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 THE REFINEMENT OF THE PROPOSED MODELS, METHODS AND SEMANTIC SEARCH

Pierre Andrews, Ilya Zaihrayeu, Juan Pane,
Aliaksandr Autayeu and Marin Nozhchev

December 2010

Technical Report # DISI-10-067

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

INSEMTIVES

Deliverable <2.4>

Report on the refinement of the proposed models, methods and semantic search

Editor:	Pierre Andrews, DISI, University of Trento
Deliverable nature:	Report (R)
Dissemination level: (Confidentiality)	Public (PU)
Contractual delivery date:	30.11.2011
Actual delivery date:	30.11.2011
Version:	1.0
Total number of pages:	69
Keywords:	semantic annotation models, ontology evolution, annotation dataset, evaluation, semantic search

Abstract

This deliverable builds on top of the previously submitted deliverables in Workpackage 2.It defines how the models and methods described in the previous deliverables need to be refined in order to better meet the evolved requirements coming from the use case partners. The deliverable reports on the results of the conducted validations of the proposed models and methods that led to the necessity of their refinement. The deliverable provides a formalisation of the necessary extensions for the annotation model, a reproducible description of the refinements to the algorithms with a discussion of how these refinements relate to the state-of-the-art in relevant areas as well as a novel evaluation method. Last but not least, the deliverable presents a platform for creating golden standards for semantic annotation systems and describes such a dataset that was created using the platform and that was used for the evaluation of some of the proposed algorithms. The dataset is exported to RDF and is currently undergoing the process of its inclusion to the Linking Open Data cloud.

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] D2.4 Report on the refinement of the proposed models, methods and semantic search

[Editor: Name, company] Pierre Andrews, DISI, University of Trento

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

Copyright notice

©2009-2012 Participants in project INSEMTIVES

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 involve the users more heavily in the generation of semantic contents, i.e., contents with machine processable formal semantics. The goal of Workpackage 2 (Models and Methods for the Creation and Usage of Lightweight, Structured Knowledge) is to develop models and methods for storing and processing these semantics contents produced by the users as well as for helping the user in the annotation process. Because the end user is not supposed to be knowledgeable in the semantic technologies field, these models need to be suitable for storing *lightweight* semantic contents that, for example, can be generated by an ordinary user as part of her everyday activities.

The previous deliverables of this Workpackage proposed models and methods based on the requirements collected from the use case partners and based on the analysis of the state-of-the-art. These deliverables are: D2.1.1 [23] (Report on the state-of-the-art and requirements for annotation representation models), D2.1.2 [21] (Specification of models for representing single-user and community-based annotations of Web resources), D2.2.1 [10] (Report on methods and algorithms for bootstrapping Semantic Web content from user repositories and reaching consensus on the use of semantics), D2.2.2/D2.2.3 [22] (Report on methods and algorithms for linking user-generated semantic annotations to Semantic Web and supporting their evolution in time), D2.3.1 [24] (Requirements for information retrieval (IR) methods for semantic content), and D2.3.2 [34] (Specification of information retrieval (IR) methods for semantic content).

The proposed models and methods were then validated against evolved requirements from the use case partners and the areas of refinements were identified. This deliverable provides a detailed account on the results of the validation and on the refinements that need to be introduced to the models and to the algorithms. In particular, the following algorithms are detailed in this deliverable: (i) the semantic convergence algorithm that supports the computation of concepts from user annotations and positioning of these concepts in an ontology; (ii) the annotation evolution algorithm that supports the recomputation of links from annotations to the underlying ontology as the ontology evolves; (iii) the summarization algorithm that is capable of computing short summaries for concepts from the ontology to help users decide which concepts to use in the annotation process; (iv) semantic search algorithm that uses the underlying ontology in order to provide the user with more relevant results. The algorithms are described at the reproducible level of details and their relation to the state-of-the-art is reported, whenever possible.

The deliverable also presents a platform for creating golden standards for semantic annotation systems and describes a golden standard dataset that was created using the platform and that was used for the evaluation of some of the proposed algorithms. To the best of our knowledge, it is the first attempt to develop such a platform that would facilitate the creation of golden standard datasets for annotation systems in the Semantic Web community. The aforementioned dataset is exported to RDF and is currently undergoing the process of its inclusion to the Linking Open Data could. The platform and the dataset represent a valuable contribution to the community, where the need for golden standard datasets, which can be used for a comparative analysis of existing approaches, has been realised.

The deliverable is the concluding deliverable on annotation models and methods in Workpackage 2. Further possible refinements of the models and methods will be reported in publications in scientific conferences, journals, and other venues.

 ©INSEMTIVES consortium 2009-2012

List of authors

Company	Author
University of Trento	Pierre Andrews
University of Trento	Ilya Zaihrayeu
University of Trento	Juan Pane
University of Trento	Aliaksandr Autayeu
Ontotext AD	Marin Nozhchev

Contents

 ©INSEMTIVES consortium 2009-2012

List of Figures

 ©INSEMTIVES consortium 2009-2012

1 Introduction

The aim of the INSEMTIVES project is to involve the users more heavily in the generation of semantic contents, i.e., contents with machine processable formal semantics. The goal of Workpackage 2 (Models and Methods for the Creation and Usage of Lightweight, Structured Knowledge) is to develop models and methods for storing and processing these semantic contents produced by the users as well as for helping the user in the annotation process. Because the end user is not assumed to be knowledgeable in the semantic technologies field, these models need to be suitable for storing *lightweight* semantic contents that, for example, can be generated by an ordinary user as part of her everyday activities.

The previous deliverables of WP2 proposed such models and methods. These deliverables are: D2.1.1 [23] (Report on the state-of-the-art and requirements for annotation representation models), D2.1.2 [21] (Specification of models for representing single-user and community-based annotations of Web resources), D2.2.1 [10] (Report on methods and algorithms for bootstrapping Semantic Web content from user repositories and reaching consensus on the use of semantics), D2.2.2/D2.2.3 [22] (Report on methods and algorithms for linking user-generated semantic annotations to Semantic Web and supporting their evolution in time), D2.3.1 [24] (Requirements for information retrieval (IR) methods for semantic content), and D2.3.2 [34] (Specification of information retrieval (IR) methods for semantic content). The current deliverable defines how the models and methods described in these deliverables need to be refined in order to better meet the evolved requirements coming from the use case partners. In the following we provide some details on the key achievements reported in this deliverable.

First of all, we consider the case where users provide annotation of resources in an uncontrolled manner. For instance, in the Telefónica use case, the users can enter free tags to annotate an image on a Web page or a segment of text. These free tags are not mapped to the controlled vocabulary and thus cannot be used for semantic search and navigation. Mapping to the controlled vocabulary can be performed by standard Word Sense Disambiguation (WSD) schemes, however, as we show in Section 3, many terms might not be present in the controlled vocabulary. For instance, if we consider WordNet as the available control vocabulary, the term "RDF" is not present in WordNet but is now a term used across the web. Another example is the one of "ajax" which is present in WordNet but with the sense: "A mythical Greek hero"; while it is now more often used to refer to the web development technology when found online. There is thus an issue of discovering new concepts in the set of free text terms used by the users when annotating resources and to attach them to an existing controlled vocabulary. This is what we call "semantic convergence" and we discuss in Section 4 an automatic algorithm to perform this task semi-automatically.

While the semantic convergence algorithm takes care of updating the knowledge base and keeping the controlled vocabulary up to date with the knowledge used by the users, another track of research takes care of keeping the annotations created by the users aligned with the controlled vocabulary. If we consider that the vocabulary can evolve (for instance with the semantic convergence algorithm, or with manual editing), then some annotation might end up referring to the "wrong" concept. For instance, if a user A tagged a resource R with the concept "sea – body of water" and later on a user B adds two more specific concepts to the knowledge base: "Black sea" and "Mediterranean sea"; to improve the semantic search and navigation service, we need to know if the resource R is referred to one of these two new sub-concepts or to the more general "sea" concept. This process is what we call "annotation evolution" and is described in Section 4.

In order to evaluate the performance of the proposed algorithms, a golden standard dataset of semantically annotated content was created. In order to create this dataset, a platform for creating golden standards for semantic annotation systems was developed. To the best of our knowledge, it is the first attempt to develop such a platform that would facilitate the creation of golden standard datasets for annotation systems in the Semantic Web community. The aforementioned dataset is exported to RDF and is currently undergoing the process of its inclusion to the Linking Open Data could. The platform and the dataset represent a valuable contribution to the community, where the need for golden standard datasets, which can be used for a comparative analysis of existing approaches, has been realised. The details on the platform and on the datasets are reported in Section 3, while in Section 4.2 we describe an algorithm for using this golden standard dataset for generating evaluation datasets for the semantic convergence process in knowledge evolution scenarios.

The aforementioned platform includes a word sense disambiguation algorithm that is used to disambiguate the meaning of terms provided by the user by mapping them to concepts in the underlying ontology. As we show in Section 3.2, it is hard to reach more than 60% of accuracy in this task and, therefore, the involvement

of the user is required at this phase of the annotation process. In order to help the user grasp fast the meaning of candidate concepts, we developed and evaluated a concept summarization algorithm that, given a concept, computes a one word summary for the concept; this summary can be used together with the concept name in order to clarify the meaning of the concept to the user. The details of the algorithm and the results of the evaluation are reported in Section 5.

Last but not least, we provide details of the required refinements for the semantic search algorithm, as reported in Section 6.

2 Annotation models

In this section we provide an analysis of the generic semantic annotation model detailed in deliverable D2.1.2 [20]; we describe how the initial requirements from the three use case partners can be accommodated within the proposed model; we also point to the new requirements that triggered modification in the initial annotation model (Section 2.1); we then propose an extension of the semantic annotation model to accommodate the new requirements (Section 2.2) and show how the extended model can be exported to RDF to facilitate the sharing and reuse of the model and data across semantic web applications as well as uploading of the semantic annotations to the LOD cloud (Section 2.3).

2.1 Use case validation

In this section we present a mapping between the collected annotation requirements coming from the use case partners and the requirements presented in deliverable 2.1.2 [21]. Due to the evolution in the scenario specification in each of the use case partners that were introduced after the analysis presented in deliverable 2.1.1 [23], some Model Objects (MO) require to be updated to fit these new requirements. In the following subsections we refer to model objects that were added or updated in order to satisfy the new requirements with **bold** (**MO#**); they will later be presented in Section 2.2. The complete definition of the Model Objects (MO#) not in bold can be consulted in the Appendix B.

2.1.1 Telefónica

Telefónica's use case will provide a semantic annotation platform for the users of their intranet corporate portal, however, the resulting semantic annotation platform will be applicable to other corporate portals as it is developed as an extension of the Google Chrome browser. The users will be able to annotate blog entries and forum posts in HTML format but will also be able to annotate news, videos, etc. For all type of resources, Telefónica requires the possibility to annotate parts of the resource with tags, with attribute annotations, or with relations to other resources. These annotations will mostly come from a controlled vocabulary that will be built bottom up in the envisioned scenario, requiring collaboration features as all annotations are shared publicly and the users can all collectively annotate resources, but also edit the vocabulary (see deliverable 5.1.2 [45]).

The semantic annotation model was initially presented in deliverable 2.1.2 [21] that analysed the requirements of the use case as reported in deliverable 5.1.2 [45]. In what follows, we present the relation between the relevant requirements from [45] and how they are supported by the Model Objects (MO) as defined in the semantic annotation model [21];

- Requirement WP5.FUN03. ANNOTATION. *The final users MUST be able to annotate certain portions of the content but also the content as a whole*. This is supported by MO2 (Resource) and MO19 (Resource with parts);
- Requirement WP5.FUN05. MULTIMEDIA ANNOTATIONS.- *The final users SHOULD be able to annotate not only text but almost any content type (images, video, audio, etc.)*. This is supported by MO20 (Textual Segment), MO21 (Image segment) and MO22 (Video Segment);
- Requirement WP5.FUN07. INTERNATIONALIZATION.- *The semantic annotation capabilities and tools SHOULD support multiple languages* .... This is supported by MO5 (Natural Language Dictionary);

 ©INSEMTIVES consortium 2009-2012

- Requirement WP5.FUN08. INTERNAL CONTENT RELATIONSHIPS.- *The semantic annotation capabilities and tools MUST allow the final users to define semantic relationships amongst certain portions of the exposed content* .... This requirement is supported by MO14 (Uncontrolled relational annotation), **MO15** (Controlled relational annotation) applied over MO2 (Resource) and MO19 (Resource with parts);

- Requirement WP5.FUN09. NON EXPLICIT ANNOTATIONS.- *The semantic annotation capabilities and tools MUST allow the final users to add additional semantic annotations about information not explicitly exposed on the content* .... This requirement demands for the addition of a new Model Object (**MO10**) called *Controlled Term* (ct) for encoding single semantic annotation that refers to multiple concepts in the same annotation (see Section 2.2.1), e.g., in the case of multi-word annotations. Then, this ct is introduced in the updated definitions of **MO11** (Controlled tag annotation), **MO13** (Controlled attribute annotation), **MO15** (Controlled relational annotation) as presented in Section 2.2.2.

- Requirement WP5.FUN10. ANNOTATION CORRECTION MECHANISM.- *The semantic annotation capabilities and tools MUST provide mechanisms to correct erroneous or false annotations* .... This requirement is supported by the accuracy element α in MO7 (Uncontrolled tag annotation), **MO11** (Controlled tag annotation), MO12 (Uncontrolled attribute annotation), **MO13** (Controlled attribute annotation), MO14 (Uncontrolled relational annotation), **MO15** (Controlled relational annotation) and **MO23** (History element); and

- Requirement WP5.FUN13. ANNOTATION AUTHOR.- *The final users SHOULD be able to know who made each annotation*. This requirement is supported by the user element u in MO7 (Uncontrolled tag annotation), **MO11** (Controlled tag annotation), MO12 (Uncontrolled attribute annotation), **MO13** (Controlled attribute annotation), MO14 (Uncontrolled relational annotation), **MO15** (Controlled relational annotation). This requirement triggers the update of **MO23** (History element) as each annotation will have only one owner, and if a particular annotation were to be changed by another user, it is assumed that this other user will create a copy of the annotation where s/he is the owner, and thus the need for multiple users in the **MO23** (History element) is deprecated.

2.1.2 Seekda!

Seekda!'s use case will provide annotation tools for the annotation of Web services. Again, this will require the annotations of the following types: attributes, tags and relations to other resources. Part of the vocabulary used will be controlled by a pool of experts while some annotations will be left open to the user's vocabulary.

At the moment of writing this deliverable, the final version of the Requirements specification deliverable (D6.1.2) was not yet available, therefore, we base our validation on:

- the initial requirements specification [18],

- the "Human-driven Annotation Tool for Web Services. Users guide"[1];

- the screencasts of the current semantic web service annotation tool[2];

- the questionnaire for determining annotation needs of the use case partners reported in deliverable 2.1.1 [23]; and

- the many direct interactions with Seekda! in order to understand the evolution of the scenario as initially defined in the above mentioned documents.

The following list presents the most relevant requirements for the Seekda! use case and how they are supported by the Model Objects (MO) presented in [21] and refined in this deliverable:

- *Support for annotating web services*. This requirement is supported by **MO13** (Controlled attribute annotation), **MO15** (Controlled relational annotation), **MO11** (Controlled tag annotation), MO7 (Uncontrolled tag annotation);

[1] http://www.insemtives.eu/tools/human_driven_annotation_for_Web_services_manual.pdf
[2] http://seekda.com/annotation-tool/index.htm

- *Named annotation such as: availability, performance, main category, cost, pricing model, payment modality, average rating, user rating or feedback and coverage based on location, type of users and language*. This requirement is supported by **MO13** (Controlled attribute annotation);
- *Support for related resources such as: item provider, related documents by the crawler and the user, mashup url*. This requirement is supported by **MO15** (Controlled relational annotation);
- *Descriptive tags linked to a general purpose controlled vocabulary where each tag may contain multiple words*. This requirement is supported by the new Model Object **MO10** (Controlled term) and the updated vesion of **MO11** (Controlled tag annotation), as a single description encoded in a single tag, might refer to more than one concept;
- *Free text for short descriptions*. This requirement is supported by MO7 (Uncontrolled tag annotation); and
- *Expert created controlled vocabulary. Currently Seekda! uses the Service Finder Service Category Ontology*[3]. This requirement is supported by MO6 (Taxonomy).

2.1.3 PGP

This use case provides an annotation platform for users of an online virtual world called MyTinyPlanets[4] where the process of creating semantic annotation missions becomes a kind of treasure hunt task with clues, locations and links from one point to another. PGP's use case will use rather subjective annotations of the virtual world by its users, and in this sense, the collaboration might not be as strong. However, all annotations, as set by the requirements in Section 3.3 of deliverable 2.1.1 [23] will be available to all users.

At the moment of writing this deliverable, the final version of the Requirements specification deliverable (D7.1.2) was not yet available, therefore, we base our validation on:

- the initial requirements specification [31];
- the questionnaire for determining annotation needs of the use case partners reported in deliverable 2.1.1 [23]; and
- the many direct interactions with PGP in order to understand the evolution of the scenario as initially defined in the above mentioned documents.

The following list presents the most relevant requirements for the MyTinyPlanets semantic annotation application use case and how they are supported by the Model Objects (MO) presented in [21] and refined in this deliverable:

- *Due to legal reasons the use case cannot allow free text to be input and will have to be drag and drop annotation*. This requirement is supported by **MO13** (Controlled attribute annotation) excluding the possibility of using MO7 (Uncontrolled tag annotation), MO12 (Uncontrolled attribute annotation) and MO14 (Uncontrolled relational annotation);
- *Support for annotation of still images*. This requirement is supported by **MO13** (Controlled attribute annotation) and MO21 (Image segment);
- *Support the recognition of characteristic image features and regions (and their outlines) which include among other stars, planets, galaxies, moons, comets, asteroids, nebula, meteors, constellations, i.e., annotation of parts of the image*. This requirement is supported by **MO13** (Controlled attribute annotation) and MO21 (Image segment);
- *The images would be "discovered" within a space context and semantically annotated accordingly, referencing of existing ontologies to describe the content including the ontology of geological or astronomical*. This requirement is supported by MO5 (Natural language dictionary);

[3] http://www.service-finder.eu/ontologies/ServiceCategories
[4] http://mytinyplanets.com/

 ©INSEMTIVES consortium 2009-2012

- *Support for recording user activity*. This requirement is supported by the user u and timestamp ts in **MO13** (Controlled attribute annotation) and **MO23** (History element); and
- *Support for (simple) second and even third level definitions such as red stars, spiral galaxies, gas planets*. This requirement is supported by MO6 (Taxonomy) and **MO10** (Controlled term) for multi-word annotations.

2.2 Refinements

Given the validation and requirements of the use case partners presented in Section 2.1 over the model defined in deliverable 2.1.2 [21] we introduce new elements to the annotation model. In particular, the changes are made considering the need to support the disambiguation of free-text multi-word annotations into their corresponding concepts. This is particularly interesting given that a tag can be split in several ways (if no space was used when creating the annotation). For example, the tag "javaisland" can be split into {"java" "island"} as well as {"java" "is" "land"} and they can both be correct. In this particular case, only the annotating user would know the intended correct split (and the related concepts), therefore it is important to be able to know in which cases to ask for user intervention, interrupting the annotation process as little as possible.

In the following we present the necessary changes to the annotation model. Note that all the model objects not explicitly mentioned here remain unchanged as defined in deliverable 2.1.2 [21].

2.2.1 New model objects

Model Object 10 (Controlled Term)
A controlled term ct is a tuple $ct = \langle t, \{lc\}\rangle$, where t is a non-empty finite sequence of characters normally representing natural language words or phrases such as "bird", "java island" or "sea"; $\{lc\}$ is an ordered list of linguistic concepts that link word (also referred to as token) in t to the controlled vocabulary.

2.2.2 Updated model objects

Model Object 9 (Linguistic Concept)
A linguistic concept lc is a tuple $lc = \langle c, nlid, st\rangle$, where c is a concept ($c \in C$); $nlid$ is the identifier of a natural language dictionary defining the language of the term; and st is a term that belongs to the set of terms of the synset that is mapped to c and $nlid$ by the concept mapping function, i.e., $st \in cmf(c, nlid)$.

Model Object 11 (Controlled tag annotation)
A controlled tag annotation tag^c is a tuple $tag^c = \langle ct, r, [u,]ts[,\alpha]\rangle$, where ct is the controlled term ct (MO10) encoding the tag text and the related concept c; r is the annotated resource ($r \in R$); when present, u is the user who created this annotation; and ts is the timestamp recorded when the annotation was created. When the annotation is added by an automatic or semi-automatic algorithm, α is provided to indicate the accuracy of the algorithm that created the annotation.

Model Object 13 (Controlled attribute annotation)
A controlled attribute annotation $attr^c$ is a tuple $attr^c = \langle can, av, r, [u,]ts[,\alpha]\rangle$, where can is a controlled term ct denoting the attribute name whose linguistic concepts $\{lc\}$ is linked to a taxonomy (i.e., $c \in tx$); av is the attribute value that can belong to any of the primitive data types (e.g., date, float, string) or that can be a controlled term ct encoding the attribute value whose linguistic concept lc is linked to a taxonomy (i.e., $c \in tx$); r is the annotated resource ($r \in R$); when present, u is the user who created this annotation; and ts is the timestamp recorded when the annotation was created. When the annotation is added by an automatic or semi-automatic algorithm, α is provided to indicate the accuracy of the algorithm that created the annotation.

Model Object 15 (Controlled relation annotation)
A controlled relation annotation rel^c is a tuple $rel^c = \langle sr, tr, crel, [u,]ts[,\alpha]\rangle$, where sr is the source resource (i.e., the resource being annotated); tr is the target resource (i.e., the resource used as an annotation object); $crel$ is a controlled term ct that denotes the name of the relation that exists between sr and tr and whose linguistic concepts $\{lc\}$ is linked to a taxonomy (i.e., $c \in tx$); when present, u is the user who created this annotation; and

ts is the timestamp recorded when the annotation was created. When the annotation is added by an automatic or semi-automatic algorithm, α is provided to indicate the accuracy of the algorithm that created the annotation.

Model Object 23 (History Element)
An history element is defined as a triple: $he = \langle action[, ov], ts[, \alpha]\rangle$. Where $action$ is the type of action that was performed $action \in \{$`added`, `removed`, `updated_name`, `updated_value`, `updated_source`, `updated_target`$\}$, ov – when applicable – is the value of the changed annotation before $action$ was performed; ts is the timestamp when this operation was performed. When the annotation is added or changed by an automatic or semi-automatic algorithm, α is provided to indicate the accuracy of the algorithm that created the annotation.

As noted in model objects 11, 13 and 15, each annotation is a relation between the annotation value ann (or name), the user u and the resource r; therefore we have chosen to remove the user u from the history element as defined in deliverable 2.1.2 since users maintain their own annotations and therefore other users cannot change them. Users can of course use the same annotation for the same resource, but in this case we would create a new annotation entry.

Each annotation element then refers to a set of history elements containing the modifications that a specific annotation received. Thus, we modify the generic definitions of the Model Objects 7, 11, 12, 13, 14 and 15 as defined in this deliverable and in the complete model reference in the Appendix B (which updates deliverable 2.1.2 [21]) to replace the $ts[, \alpha]$ provenance information by a set of history elements $\{he\}$.

For instance, the Model Object 11 becomes:

Model Object 24 (Controlled tag annotation)
A controlled tag annotation tag^c is a tuple $tag^c = \langle ct, r, [u,]\{he\}\rangle$, where ct is controlled term encoding the tag text and the related concepts c; r is the annotated resource ($r \in R$); when present, u is the user who created this annotation; and $\{he\}$ is a set of history elements storing the different versions of this annotation element.

2.3 RDF Mapping for the Annotation Model

To make the annotations that we collect within the INSEMTIVES project easy to distribute as Linked Open Data (LOD) datasets, we have mapped most of our model to existing vocabularies in the LOD. Figure 1 provides an overview of the links that can be used in our model:

The model is divided in three main components representing the different dimensions that we defined in deliverables D2.1.1 [23].

Annotation The annotation uses as main element the `PKM:Mention` class from the Proton KM[5] ontology that represents a section of a (or a full) resource that is linked to an entity. For both uncontrolled and controlled annotation, the semanticless string representing the name of the annotation (i.e., the tag, the attribute name or the relation name) can be stored using the `dc:title` predicate [19].

Only one new property definition `insem:hasValue` is created to use the `PKM:Mention` class as an attribute annotation (see definition in Figure 2). This property links to one of the following elements:

- a literal – defining a simple value for the attribute (a String, a Date, etc.);
- a `PTOP:InformationResource` – defining a relational annotation; or
- a `PTOP:LexicalResource` to represent a controlled attribute annotation. The ambiguous natural language string used for the annotation value is stored as a `dc:title` predicate while, as explained in the next point, the meaning of this lexical entry is disambiguated with a (list of) `ctag:means` predicate(s). This class is used to represent a controlled term ct, as defined in **MO10** while the `ctag:means` predicate provides links of the ct to the controlled vocabulary.

Vocabulary The annotations can be linked to a controlled vocabulary. When they are simple mentions (or controlled tagging), they are linked to entities or concepts through the `PKM:mentions` link or to an `rdf:Seq` of[6] concepts through the `ctag:means` link provided by the CommonTags ontology [4].

[5] `http://proton.semanticweb.org/2006/05/protonkm`
[6] in the case where the label of the annotation is composed of multiple tokens

 ©INSEMTIVES consortium 2009-2012

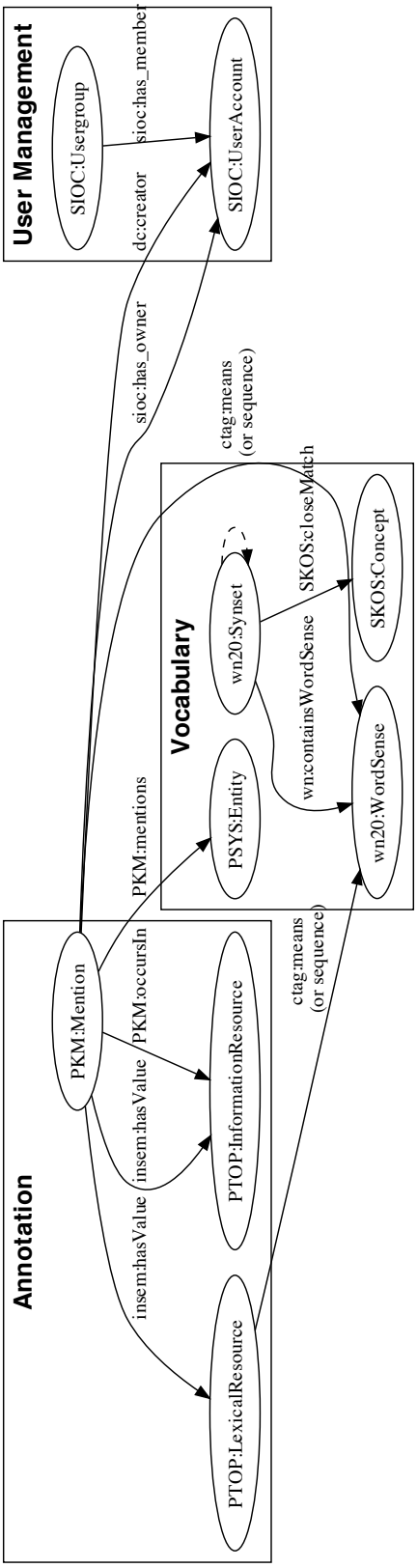


Figure 1: RDF Mapping for the Annotation Model

```
<owl:ObjectProperty rdf:about="&insem;hasValue"
     rdfs:label="hasValue"
     rdfs:comment="When the PKM:Mention is used to annotate an attribute,
                   its value is defined by this property">
    <rdfs:domain rdf:resource="&pkm;Mention"/>
    <rdfs:range rdf:resource="&ptop;InformationResource"/>
    <rdfs:range rdf:resource="&rdfs;Literal"/>
    <rdfs:range rdf:resource="&ptop;LexicalResource"/>
</owl:ObjectProperty>
```

Figure 2: Definition of the `insem:hasValue` property

In this way, the `dc:title` that stores the semanticless representation of the annotation name can be disambiguated.

In the same way, if the value of an attribute has to be linked to a controlled vocabulary, it is represented as a `PTOP:LexicalResource` instead of a simple string and is linked through an ordered list (`rdf:Seq`) to a set of concepts in the controlled vocabulary with `ctag:means`.

As in our experiments and in the Seekda! and Telefónica use cases we used the WordNet vocabulary to provide the concepts that the users can use for annotation, we link the controlled annotations to the WordNet-rdf mapping [1]. The language independent abstraction is dealt with by using the existing `wn20:WordSense` class and the language features provided by RDF[7]. Thus, the RDF representation of the Natural Language Dictionary (NLD) in our model[8] will be as follows: An NLD with name "en-us" will be represented in RDF as a set of interconnected `wn20:Word` whose labels are literals with language "en-us". An advantage of this solution is that it allows to use the `langMatches` semantics in SPARQL construct to make searches for specific languages or language groups.

User Management The user management being mainly dealt with by the use case implementation and presenting privacy issues when exporting to the LOD, this part of the mapping is kept simple and only the ownership of the annotation is exported. The SIOC [17] and FOAF[9] vocabularies can then be used to describe the users metadata. To link an annotation (`PKM:Mention`) to its creator, both the dublin core [19] `dc:creator` and the SIOC `SIOC:has_owner` can be used.

3 Semantic Annotation Dataset

3.1 A Platform for Creating Golden Standards of Semantic Annotation Systems

Since the work on the extraction of formal semantics from folksonomies became an active topic of research in Semantic Web and related communities, the research community realised that the lack of golden standards and benchmarks significantly hinders the progress in this area: *"The current lack of transparency about available social tagging datasets holds back research progress and leads to a number of problems including issues of efficiency, reproducibility, comparability and ultimately validity of research"* [37]; and in particular about finding semantics in folksonomies *"We have noticed the lack of testbeds and standard evaluation metrics that allow proper comparisons of the different research works"* [29, 37, 5].

Actually, as reported in [37], some datasets are available of different folksonomies, however, none of them is yet annotated with links to an ontology disambiguating the tags' meaning (the tags' semantic). For instance, in the del.icio.us folksonomy[10], if a bookmark is tagged with "javaisland", to know the real semantic of such tag, we need to split the tag in the two tokens "java" and "island" (tag cleaning) and then connect each token to its corresponding unambiguous sense: "java – an island in Indonesia to the south of Borneo" and "island – a land mass that is surrounded by water" (disambiguation, as illustrated in Figure 3).

[7] As suggested by [2]

[8] compliant with Appendix H of the W3C WordNet-in-OWL working draft

[9] `http://www.foaf-project.org/`

[10] `http://del.icio.us`

 ©INSEMTIVES consortium 2009-2012

In fact, if we look at some of the latest publications in the field, we can see that such call for evaluation datasets is well grounded. For instance, in [38], the authors try to extract ontological structures from folksonomies but only report a "subjective evaluation" where they study manually a generated ontology with four branches. To perform a real quantitative evaluation of their algorithm, the authors would need a folksonomy of which the tags are already linked to an existing ontology.

Another example is the work of [25], where they study the relation between distributional semantic measures of similarity between tags and a measure based on the WordNet ontology. To do this, they link a tag to its sense in WordNet and use a distance measure in the WordNet *is-a* hierarchy. However, their assumption for disambiguation of homograph tags to the right sense is to use the closest related tag in the context; as we show in Section 3.1.3 the accuracy of such disambiguation is only around 60%, if they had taken the most frequent sense[11] in the english language, they would also have a similar disambiguation accuracy [53]. This means that in around 40% of the tags they are considering, the results for the WordNet based distances are wrong and cannot be used for comparison. Furthermore, as we show in Section 3.2.4, a number of tags appear in WordNet but do not have any sense available. For instance "ajax" is only present in WordNet as "a mythical Greek hero" while the most often used sense in the delicious dataset they use is the one relating to the programming technology used on the web applications[12].

Some studies can thus be conduced on the semantic-less tags, but the use of semantics in folksonomy, and its automatic extraction, can only be evaluated if we have available an explicit disambiguation of the tags used on each resources in the folksonomy. Up to now, as we have illustrated here, this *semantification* of the folksonomy is done with an imperfect automatic method or manually, which can only lead to small scale qualitative studies or limited quantitative evaluations that yield results with a large margin of error.

In line with what has been requested in other publications [29, 37, 5], we thus believe that a new type of dataset is needed that provides a real golden standard for the links between the folksonomy tags and their semantics. To guarantee the validity of such a golden standard, the tag cleaning and disambiguation has to be performed manually by a team of annotators. In the following section, we describe such annotation process, and some results obtained on a golden standard that we built on a sample of del.icio.us.

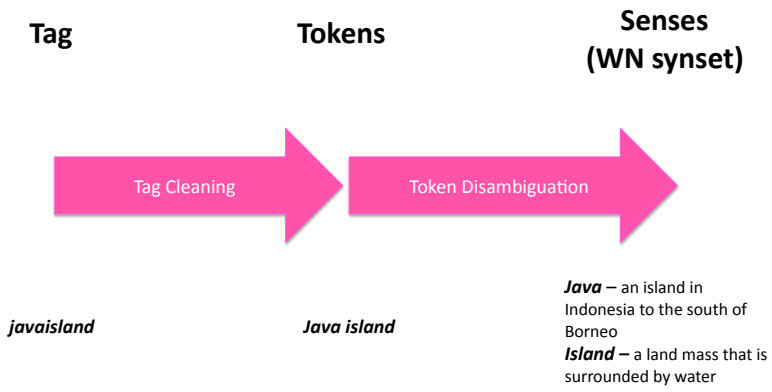


Figure 3: Semantic Folksonomy Dataset Creation Process

In order to help address the problem mentioned above and to create a golden standard dataset to be used for the evaluation, we developed a platform for creating golden standards for semantic annotation systems, called *Tags2Con* (an abbreviation for "from tags to concepts").

The *Tags2Con* platform's aim is to provide a manual validation interface for quiclly annotating large samples of folksonomies to create the required golden standards.

The platform provides a set of utilities and tools for the semi-automatic conversion of tags in annotation systems such as del.icio.us into controlled annotations in which tags are linked to concepts in a controlled vocabulary such as WordNet. The platform supports the annotator in many respects, for instance: they propose a split for a tag consisting of several concatenated words (e.g., "javaisland" can be split into "java island" or "java is land"); they propose a sense for a word in a tag by running a word sense disambiguation algorithm; they

[11] as defined by WordNet.

[12] http://en.wikipedia.org/wiki/Ajax_(programming)

provide an export facility to RDF; and others. The platform comes with a comprehensive set of user interfaces and allows for multiple annotators to assign concepts to tags and then computes the level of agreement between the annotators. Last but not least, the platform includes a module for the evaluation of a simple semantic search algorithm that uses concepts instead of tags for searching annotated resources. The platform can be downloaded from `http://www.sourceforge.net/projects/insemtives/`.

In the following we describe some of the key steps that the platform enables in order to convert uncontrolled tag annotations to controlled tag annotations ($t \rightarrow ct$), give examples and screenshots, and provide other information in order to help the reader understand better the essentials of the platform. In this description, we use the Del.icio.us bookmarks as an example of annotated resources, but the platform supports any kind of resources dereferencable with HTTP URLs (e.g., images, bibsonomy entries). In Section 3.2, we describe a first dataset of semantic folksonomy that we manually annotated with the platform described here.

3.1.1 Manual validation

The *Tags2Con* platform main task is to ask validators to provide the true disambiguation of tags. The process of validation includes the selection of the correct tag split, and for all the tokens in the tag split, the selection of the correct sense. Figure 4 shows the user interface developed for the validation task. In order to reach a certain degree of certainty that the validated sense is the correct one and not due to a mistake (or chance); we control annotator agreement by having all the tags validated by two different validators.

The manual validation process will be better explained with an example. Figure 4 shows the manual validation user interface presented to the validator, for example, the URL `http://1videoconference.com/`. For this URL all the tags applied by the del.icio.us user u "83942" are presented, these are: "tools", "collaboration", "conference", "videoconferencing" and "opensource". The validation interface contains five main parts: the *Tag list*, the *Ignore* options for a particular resource r, the *Split and Disambiguation* list of the selected tag, for each possible split an *Ignore* option, and finally, the *Ignore* options applicable to the whole tag.

The validation process is summarized as follows:

1. A $\langle resource, user \rangle$ pair is selected automatically to ensure that all tags for the $\langle r, u \rangle$ pairs are validated twice to reach a certain confidence in the validation process. Once the resource is selected, it is visualised in another window of the browser, so that the validator can check the Web page and therefore understand the context and the possible intended meanings of the tags. The validator has the option of ignoring the given resource if, for example, the Web site is not available anymore, it is in a foreign language, among others (see Figure 5);

2. the validator selects a particular tag for the given $\langle r, u \rangle$ pair as shown in Figure 6.

3. the validator selects the correct tag split from the existing list as shown in Figure 6. The validator is also given the option to ignore the selected tag if there is no meaningful tag split or sense is suggested as shown in Figure 8. The validator also is presented with the option of suggesting the correct tag split, if this one is missing;

4. the validator selects the correct sense for each of the tokens in the tag. For example, for the word "video", WordNet shows three senses as shown in Figure 9. The user can also choose to ignore the token (word) if the correct sense is not present in the list, indicating the reason; for example because the correct sense is missing from the controlled vocabulary;

5. the validator submits the results and validates the next tag, until all the tags have been validated for the $\langle r, u \rangle$ pair, in which case the validation page goes to the next $\langle r, u \rangle$ pair.

Selecting the right tag split and the right disambiguation for each token in such split is a tedious task for the human annotators and *Tags2Con* tries to make this task as straightforward as possible. It also provides some supporting tools to simplify the work of the validators and streamline the annotation process. With this platform, a team of three annotators have already annotated a sample of one thousand bookmarks from a del.icio.us crawl in less than a week (See Section 3.2). To enable such streamlined annotation, some pre-annotation is performed automatically so that the most probable splits are already available to the validators and the most probable disambiguation is also proposed. These supporting tools are described in the following sections.

 ©INSEMTIVES consortium 2009-2012

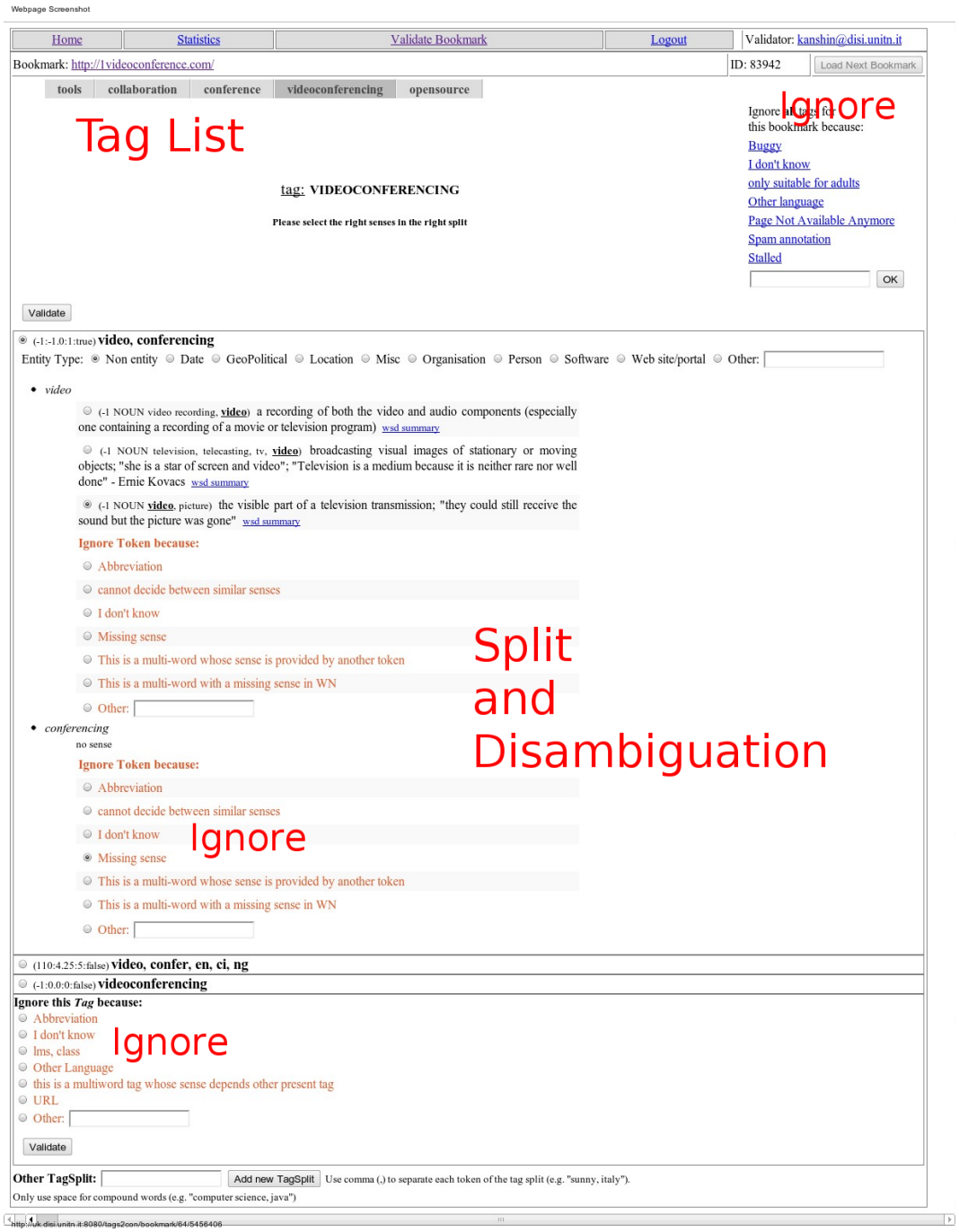


Figure 4: Annotation Page for the Manual Tag Disambiguation

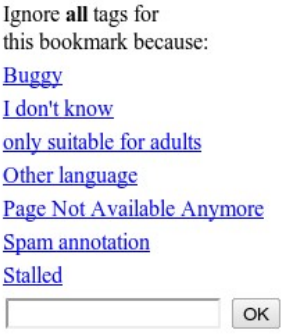


Figure 5: Ignoring options for a URL

Figure 6: URL and list of Tags to be validated

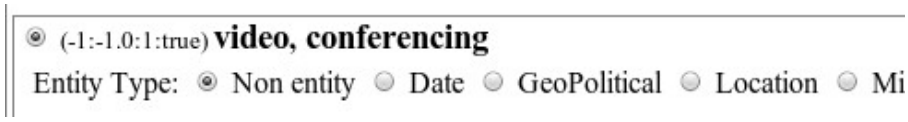


Figure 7: A proposed split for a tag and possible Entity annotations

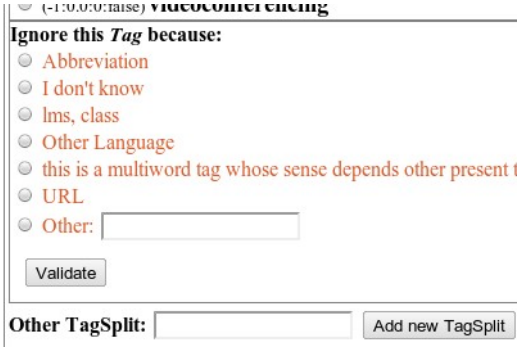


Figure 8: Ignoring options for a Tag and a field to propose a new split

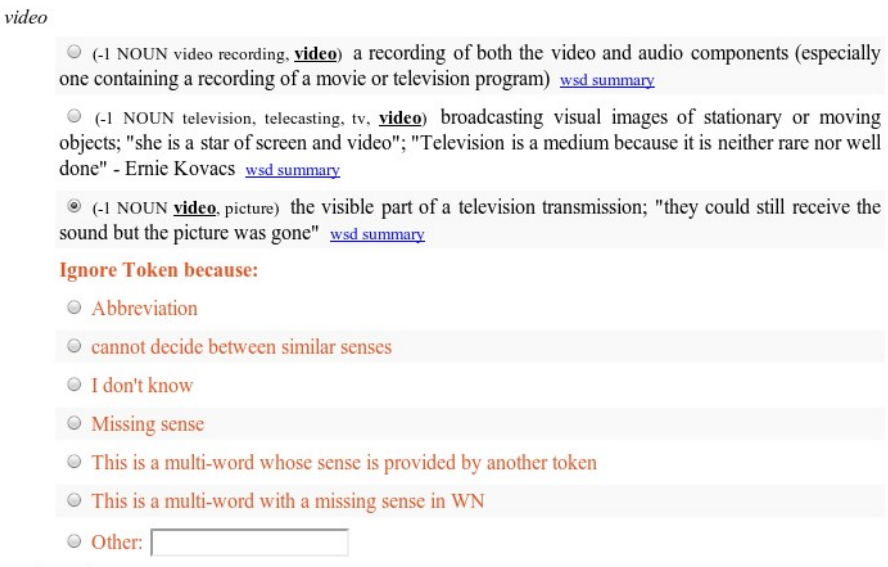


Figure 9: Choice of disambiguations for a Token of a tag

 ©INSEMTIVES consortium 2009-2012

3.1.2 Preprocessing

The goal of the preprocessing step is to recognise a word sequence in a tag that may consist of several concatenated tokens which might have been written with syntactic variations (e.g,. plurals, exceptional forms). This step is composed of the following sub-steps:

1. **Tag split**: split the tags into the component tokens. This step is needed considering the fact that many annotation systems such as del.icio.us do not allow spaces as word separators[13] and, therefore, users concatenate multi-word and phrasal annotations using the Camel case (javaIsland), slashes (java-island), underscores (java_island) or other separator they deem useful. We also found that in many cases, heuristics based on case ans special characters of the annotation could not be applied as users just entered the multi-word tags without any sort of separator, e.g., "javaisland". The tag split preprocessing runs a search on the controlled vocabulary (such as WordNet) and tries to place *splits* when it recognises valid tokens. This preprocessing can generate different splits for the same tag, for instance, the tag "javaisland" can be split into {"java", "island"} or {"java", "is", "land"}. The output of this step is ranked to present the most plausible split to the annotator first. The ranking prefers proposals with a fewer number of splits and with the maximum number of tokens linked to the controlled vocabulary.

2. **Lemmatization**: in order to reduce different forms of the word into a single form (that can later be found in a vocabulary such as WordNet), a number of standard lemmatization heuristics are applied. For example, "banks" would be preprocessed as "bank".

3.1.3 Automatic disambiguation

In this step we run an automatic disambiguation algorithm in order to suggest the validator the possibly correct sense of the word (as preprocessed in the previous step). The algorithm is an extension of the one reported in [54] and based on the following idea that collocated tags provide context for disambiguating (as generalised in the survey by Garcia-Silva [5]). In our approach, given a token within a tag split, we consider three levels of context:

1. the other tokens in the tag split provide the first level of context,

2. the tokens in the tag splits for the other tags used for the annotation of the same resource by the *same* user provide the second level,

3. the tokens in the tag splits for the tags used for the annotation of the same resource by *other* users, provide the third level of context.

The senses of the tokens from the contexts of the different levels are then used to disambiguate the sense of the given token by mining the possible semantic relations between the senses of the token and the senses of the tokens from the contexts. When a relation is found, the score of the corresponding word sense is boosted by a predefined value. The used relations are as follows (in decreasing order of their boost value):

1. synonymy (e.g., "image" and "picture");

2. specificity, measured as the length of the is-a path between two senses (e.g., "dog (Canis familiaris)" is more specific than "animal (a living organism)"); and

3. relatedness, measured as the sum of the lengths of the paths from the two given senses to the nearest common parent sense (e.g., "table (a piece of furniture)" is related to "chair (a seat for one person) through the common parent sense "furniture (furnishings that make a room)").

For the specificity and relatedness relations, the scores are adjusted according to the length of the path (the shorter the length, the higher the score). The scores for all the relations are also boosted according to the level of the used context (level one leads to higher scores, whereas level three leads to lower scores). The algorithm

[13] For example, if the user enters a space in the annotation interface of Del.icio.us the annotation becomes two different tag annotations.

then uses two other heuristics to boost the scores of word senses, namely: boosting the sense of a word if the part-of-speech (POS) of the sense is the same as the one returned by a POS tagger (which can reach more than 97% in accuracy on metadata labels as shown in [54]); and boosting the sense of a word according to the frequency of usage of the sense (this data is available in linguistic resources such as WordNet [41]).

The sense with the highest score is then proposed to the validator as the suggested meaning of the token. If more than one sense has the highest score, then the following heuristic is applied: nouns are preferred to verbs, verbs are preferred to adjectives, and adjectives are preferred to adverbs – it follows the distribution of the tag tokens by POS in annotation systems such as del.icio.us as reported in [27] and confirmed in our own analysis (see Figure 18). Finally, if more than one candidate remains, then the sense with the highest frequency of usage is selected.

Word Sense disambiguation (WSD) is known to be a difficult problem [52, 7], especially when it is applied in the domain of short metadata labels such as those used to name the categories in a Web directory such as DMOZ[14] or the like [54, 15]. Some work also exists on the disambiguation of tags in the domain of folksonomies, see [5] for a survey of existing approaches. According to the classification presented in the survey, our approach falls under the *ontology-based* category in which the meaning of tags is defined by an explicit association of an ontology element (such as a class or an instance). With respect to the disambiguation task, the most noticeable approaches are [12, 29]. In [12], the authors define the context of a tag as the other tags that co-occur with the given tag when describing a resource, and they use the senses of the tags from the context for the disambiguation of the sense of the tag by using the Wu and Palmer similarity measure between the senses [50]. Our approach is different in that it is using different measures for the computation of the similarities between senses as well as in that it uses a POS tagger and the frequency of senses to further refine the selection of the tag token sense. The approach presented in [29] uses Wikipedia[15] as the source of possible meanings of a tag. In order to compute the sense candidate, the disambiguation activity then uses the cosine function on the vectors that represent the tag with its context and the most frequent terms in the retrieved Wikipedia pages. As pointed out earlier, without having golden standards and benchmarks, it is difficult to conduct a comparative analysis with the existing approaches. This consideration was also made in [29]. Therefore, for the time being we can only describe relevant approaches pointing to the differences in algorithms with respect to our approach without providing any quantitative measure to compare the different approaches, however, a qualitative evaluation of our algorithm is provided in Section 4.

3.2 Semantifying a del.icio.us Dataset

In the INSEMTIVES project, we are interested in the annotation of resources with a mix of uncontrolled annotations (for example tags) and controlled annotations (which are linked to a controlled vocabulary). As we mentioned previously, we believe that this mix of uncontrolled and controlled terms can be used to extend semi-automatically the controlled vocabulary (see Section 4). However, to study the feasibility of this task and to test and evaluate the proposed algorithms, we require a dataset that represents the problem we are looking at. The most suited dataset in the project would be one coming from the use cases, in particular Telefónica and Seekda! as they both intend to use a mix of controlled and uncontrolled annotations (while PGP, for legal reason, can only use a controlled vocabulary). But, as these use cases are still under development within the INSEMTIVES project, there is not yet a suitable, large size dataset that we could use.

In the current state of the art of the semi-automatic knowledge building field, folksonomies such as Del.icio.us[16] or Flickr[17] are used as data providers to study the automatic construction of ontologies and the evolution of uncontrolled vocabularies. To be aligned with the state of the art work, we have chosen to also study the del.icio.us folksonomy as a replacement to a use case specific dataset. We believe that this resource is close to the Telefónica and Seekda! use case in that it annotates resources (web site urls) with uncontrolled tags. This is quite similar to what users would do in Seekda!'s use case when annotating documentation webpages linked to a particular webservice or to what users of the Telefónica's use case would do with the OKenterprise plugin.

[14] http://www.dmoz.org
[15] http://www.wikipedia.com
[16] http://delicious.com
[17] http://flickr.com

 ©INSEMTIVES consortium 2009-2012

del.icio.us is a simple folksonomy as was defined by [47] and formalised by [40] in that it links resources to users and tags in a tripartite graph. However, these tags are totally uncontrolled and their semantic is not explicit. In the current datasets provided by Tagora[18] and others [37], no-one has yet, to the best of our knowledge, provided a golden standard with such semantics. As mentioned earlier, we are interested in a mix of these free-text annotations with controlled annotations whose semantic is defined by their link to a controlled vocabulary (such as WordNet for instance). In that, the del.icio.us dataset is not perfectly what we are looking for. The Faviki[19] website could provide such dataset, however it does not contain so many users and annotations as del.icio.us and the quality of the disambiguations is not guaranteed. To make the del.icio.us dataset fit our problem statement, we have thus decided to extend a subset of a del.icio.us dump with disambiguated tags using the *Tags2Con* platform described in Section 3.1. We used WordNet 2.0 as the underlying controlled vocabulary for finding and assigning senses for tag tokens.

Given that the algorithms to be discussed in Section 4.1 and Section 5.1 are to be evaluated over the semantics of the tags, we need a golden standard dataset that complies with **Model Object 10** (Controlled tag annotation), i.e., we need to map the tags t to controlled terms ct and therefore split the tags into its components (tokens) and map each of them to a linguistic concept lc.

3.2.1 del.icio.us Sample

We obtained the initial crawled data from del.icio.us between December 2007 and April 2008 from the authors of [49]. After some initial cleaning (eliminate invalid URLs, tags with only numbers and non-ASCII characters) the dataset contains *5 431 804* unique tags (where the uniqueness criteria is the exact string match) of *947 729* anonimized users, over *45 600 619* unique URLs on *8 213 547* different website domains. This data can be considered to follow the **Model Object 7** (Uncontrolled tag annotation) $\langle t, r, u \rangle$[20] where the resource r is the URL being annotated, containing a total of *401 970 328* uncontrolled tag annotations. Note that this is the larger crawled dataset of del.icio.us currently available.

To make the del.icio.us dataset fit our problem statement, we have thus decided to extend a subset of a del.icio.us dump with disambiguated tags, this is, convert $t \rightarrow ct$. This means that for each tag t in this subset, we have explicitly split it in its component tokens and marked it with the WordNet synset (its sense) it refers to.

The golden standard dataset we have built includes annotations from users which have less than 1 000 tags and have used at least ten different tags in five different website domains. This upper bound was decided considering that del.icio.us is also subject to spamming, and users with more than one thousand tags could potentially be spammers or machine generated tags as the original authors of the crawled data assumed [49]. Furthermore, only $\langle r, u \rangle$ pairs that have at least three tags (to provide diversity in the golden standard), no more than ten tags (to avoid timely manual validation) and coming from users who have tags in at least five website domains (to further reduce the probability of spam tags) are selected. Only URLs that have been used by at least twenty users are considered in the golden standard in order to provide enough overlap between users in order to allow the semantic convergence and evolution algorithms to be evaluated. After retrieving all the $\langle r, u \rangle$ pairs that comply with the previously mentioned constraints, we randomly selected 500 pairs. Table 1 summarizes the characteristics of the resulting subset of the dataset used to build the golden standard.

Item	Count
$\langle r, u \rangle$ pairs	500
total number of tags	4707
average tags per $\langle r, u \rangle$	4.3
unique tags	871
number of URLs	299
number of users	500
website domains	172

Table 1: Statistics about the golden standard dataset.

[18] http://www.tagora-project.eu/data/

[19] http://faviki.com/

[20] The actual definition of the Uncontrolled tag annotation is $\langle t, r, u, ts[, \alpha] \rangle$ but considering the obtained dataset did not contain the time stamp we omit it for simplicity.

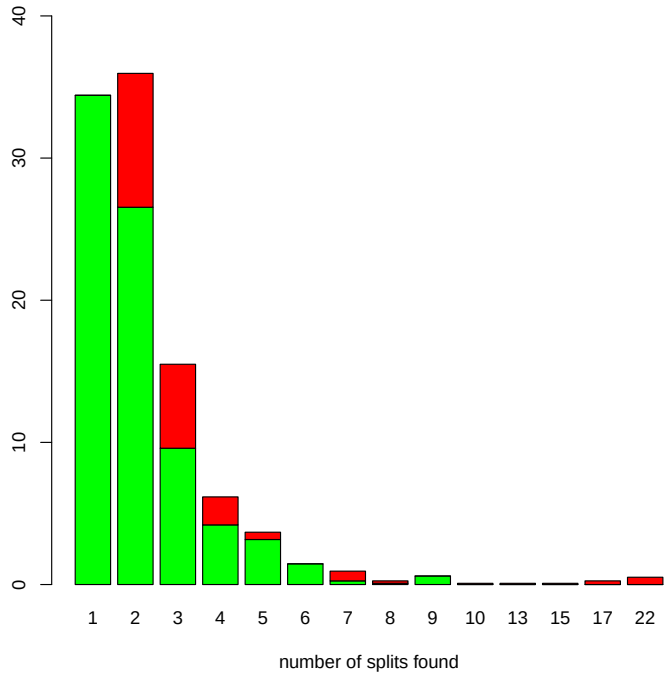


Figure 10: Accuracy of the Preprocessing algorithm by the number of possible splits proposed to the validator.

3.2.2 Results

The *Tags2Con* platforms provides a set of semi-automatic tools to help the annotation process. In the following paragraphs we describe a first evaluation of the validity of these support tools based on the annotated sample from del.icio.us.

Preprocessing The accuracy of the preprocessing step (see Section 3.1.2) in this validation task reached 97.24%. In Figure 10 we provide a detailed analysis of the accuracy of the algorithm for different numbers of possible splits. The Y axis corresponds to the distribution of tags per number of possible splits proposed to the validator, the top box is the amount of wrong split while the bottom one represents the amount of accurate splits that were ranked top by the preprocessing algorithm. The figure should be read as follows: the number of tags with two possible splits is ∼35% and the accuracy of the algorithm for these cases is ∼80% (see the second bar from the left).

We believe that the current accuracy of the preprocessing algorithm can be improved by some simple improvements on the lemmatization heuristics as well as by using a lexicon of existing words in the English language.

Word Sense Disambiguation The average polysemy of the tag tokens in the dataset was 4.68, i.e., each tag token had 4.68 possible senses on average. The proposed WSD algorithm performed at 59.37% in accuracy. In Figure 11 we provide a detailed analysis of the accuracy of the algorithm for different levels of polysemy. The Y axis corresponds to the distribution of tokens per polysemy, the top box is the amount of wrong disambiguation while the top one represents the amount of accurate disambiguations that were ranked top by the WSD algorithm. The figure should be read as follows: the number of cases with two possible senses in the controlled vocabulary is ∼22% and the accuracy of the algorithm for these cases is ∼90% (see the second bar from the left).

It is worth noting that, on Figure 11, we can see that the WSD algorithm has an accuracy lower than 50% for the tokens with many available senses, however, the biggest amount of tokens only had two senses available and in this case, the WSD algorithm performed at an accuracy close to 90%.

 ©INSEMTIVES consortium 2009-2012

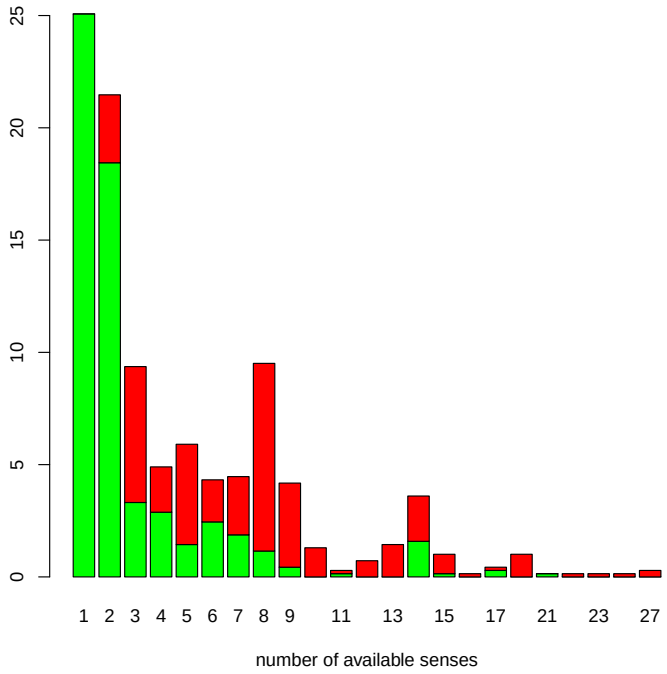


Figure 11: Accuracy of the WSD algorithm by the level of polysemy

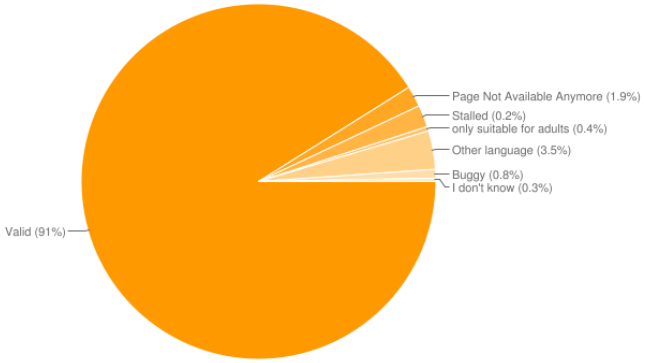


Figure 12: Distribution of Validated Dataset Entries

From the result we conclude that the WSD problem can be harder in its application in the domain of tag annotations than in its application in the domain of web directory labels, which are closer to tags in their structure than well formed sentences. This suggests that the annotators should not fully rely on the result of the WSD algorithm and that they may need to check and/or provide the input manually at this annotation phase.

Validation In order to guarantee the correctness of the assignment of tag splits and tag token senses, two different validators validated each $\langle URL, u\rangle$ pair. As the result, we obtained one thousand validations that are summarized in Figure 12. As we can see, 91% of $\langle URL, u\rangle$ pairs in the dataset were marked as valid, i.e., the user selected a tag split and disambiguation; 8,9% were marked as invalid (see Figure 5) either because, for instance, the Web page was not available anymore, it was in another language, the page was only suitable for adults.

Among the 911 $\langle URL, u\rangle$ pairs marked as valid, 84.69% of the tags could be validated by the validators while the other tags had to be ignored (as shown in Figure 8, this is mostly because the validator did not know what the tag meant, the tag was in another language, or consisted to be part of a multiword tag[21].)

[21] this ignore option was used when the meaning of one tag was dependent on another different tag, for example, when two tags as in

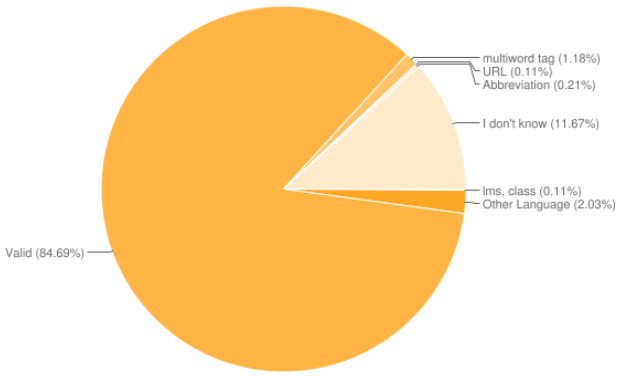


Figure 13: Distribution of Ignored Tags

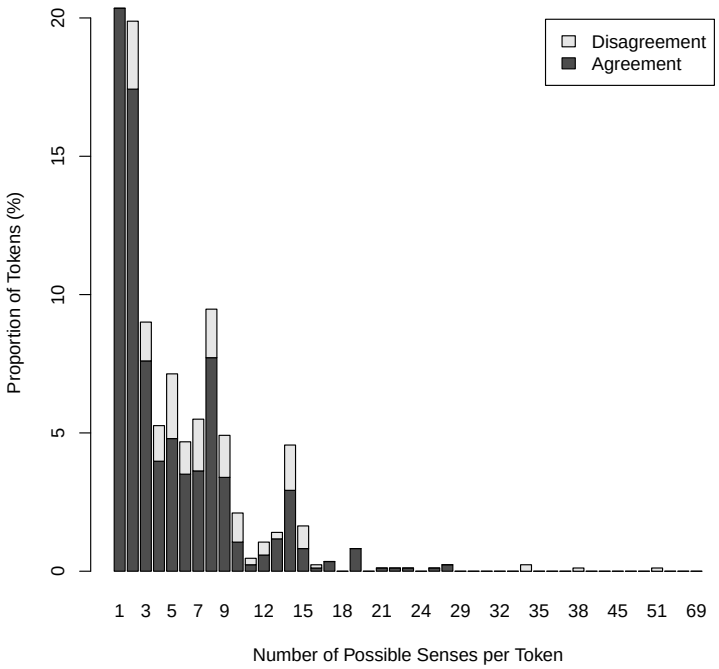


Figure 14: Agreement between Annotators on Sense Validation, per Number of Available Senses

 ©INSEMTIVES consortium 2009-2012

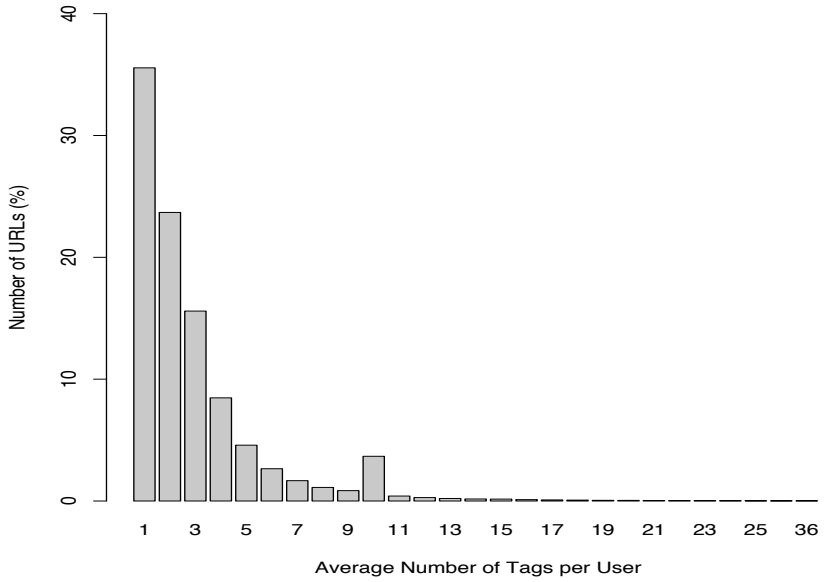


Figure 15: Number of Tags per URL per User

The "agreement without chance correction" [13] between users in the task of disambiguation of tokens is 0.76. As mentioned in [13], there is not yet any accepted best measure for the agreement in the task of sense annotation and thus we currently report only the raw agreement. It is intuitive to think that the agreement will fall when there are more available senses for one token as the annotators will have more chance to choose a different sense. This could also happen because, as we show in Figure 19, sometimes the annotators cannot decide between too fine grained sense in the controlled vocabulary. Figure 14 shows a more detailled view of the effect of number of available senses on the annotators' agreement.

3.2.3 Considerations on the Dataset Uncontrolled Vocabulary

del.icio.us is used in many research groups that work on folksonomies as a large dataset showing how users use tags to organise and share their resources. We have thus started by a basic analysis of how users used tags in the dataset and what we could observe from this. In the following paragraphs, we discuss the analysis that we performed on the whole dataset of 45 600 619 URLs, with all the users and tags available. The analyis and first conclusion on the manual disambiguation batch of 500 $\langle URL, u \rangle$ pairs is discussed in the next section.

In the INSEMTIVES project, we are interested in the generation of semantic annotations for resources, in particular for online resources. The del.icio.us dataset is thus of great interest to us as, even if it does not provide the semantics of the annotations, it illustrates the annotation behaviour of many users in a real application. A first hypothesis in the project is that we will need incentives and semi-automatic methods to help the users create these semantic annotations as the task is difficult and not usually motivating for the user. While the annotation task on del.icio.us is quite simpler as it does not require the specification of semantics, we can already see that the users are not motivated to provide a large amount of annotations. Note that we cannot make any conclusions on why this might be the case as this would require a direct users study, however, as illustrated by Figure 15, we can see that in 35.5% of the cases, users use only one tag per bookmark and only in 12.1% of the cases they would add more than five tags per bookmark.

This might be because each user only uses very specific tags to classify/categorize the bookmark and thus does not require many indexing terms to find the resource in the future. This assumption would be a "dream" scenario as it would mean that the users are already ready to provide very specific descriptors for their resources and if these descriptors are linked to the underlying controlled vocabulary, we can retrieve them using synonymous and/or more general terms very easily. However, it might just be that the users are not bothered to add more tags as they do not see the value of adding many indexing terms for future retrieval.

"George" and "Bush" were applied – which derives from the issue of using space as separator in del.icio.us.

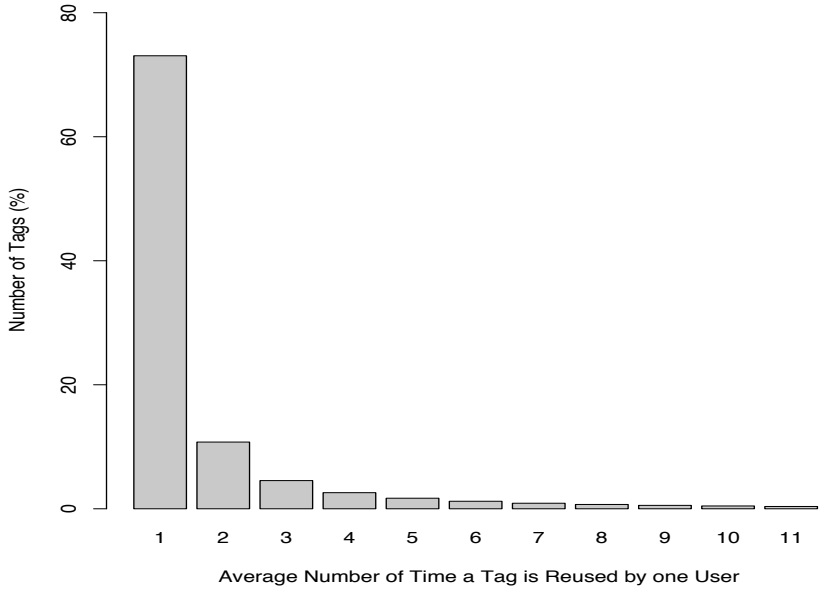


Figure 16: Number of Time a Tag is Reused by the same User on all the Bookmarks

An interesting point is that there is an out-of-the-norm peak at ten tags per bookmark that seems too strong to be coincidental. We have not yet studied in details why this happens but hypothesise that it might be created by spambots providing a lot of bookmarks with exactly ten tags.

In Figure 16, we consider another interesting feature of the tagging behaviour of users on del.icio.us. While an often used assumptions in folksonomy study algorithms is that we can learn a lot from tag collocations on different resources, we can see that users do not often reuse the same tag more than once.

In fact, from our analysis, in 73% of the cases, a tag is used only once on the whole set of bookmarks by a single user. This means that in a majority of the cases, a tag will not be found located on different resources, at least not by the same user. Only in 7.3% of the cases a tag is reused on more than seven resources.

This might support our previous assumption that the users use very specific tags when they annotate resources and thus they do not use them on multiple documents. However, this might create difficulties when sharing knowledge between users as they might not use the same vocabulary (as they use very specific/personal terms). It might also impair the ontology learning algorithms [5] that are based on the measure of collocation of tags.

When annotating shared goods such as web pages, if their is no agreement between the users on what the resource means, it is difficult to reuse these annotations to improve search and ranking of resources. It is also difficult to learn the meaning of the resource or of the annotations attached to it. We have thus done a preliminary analysis of the general agreement of the users in the del.icio.us dataset when they tag a resource. Here we are interested to see how many tags are used by more than one user on the same resource.

To do this, we have adopted a naïve measure of agreement where we count how many users have used the same tag on the same resource. For instance, if there is user U_1 who tagged a resource R_1 with T_1 and T_2 while user U_2 tagged this resource with T_3 and T_4, then there is only one user using any of the four tags. If U_3 tagged R_2 with T_5 and T_6, U_4 tagged it with T_6 and T_7 and U_5 with T_8 and T_9, then there are two users agreeing on at least one tag for that resource. Note that we only consider URLs in the dataset bookmarked by at least two users.

Figure 17 shows the results of this measure. In 67.5% of the cases, there is only one user "agreeing" on at least one tag, which means different users used different tags on the same resources. In only 9.3% of the cases more than three users agreed on at least one tag.

In a sense this is a good result in that users do provide very diverse tags for the same resource and thus we can learn more about the resource itself. However, if there is no agreement between the users, it is difficult to consider that tags are valid as they might be very personal or subjective.

 ©INSEMTIVES consortium 2009-2012

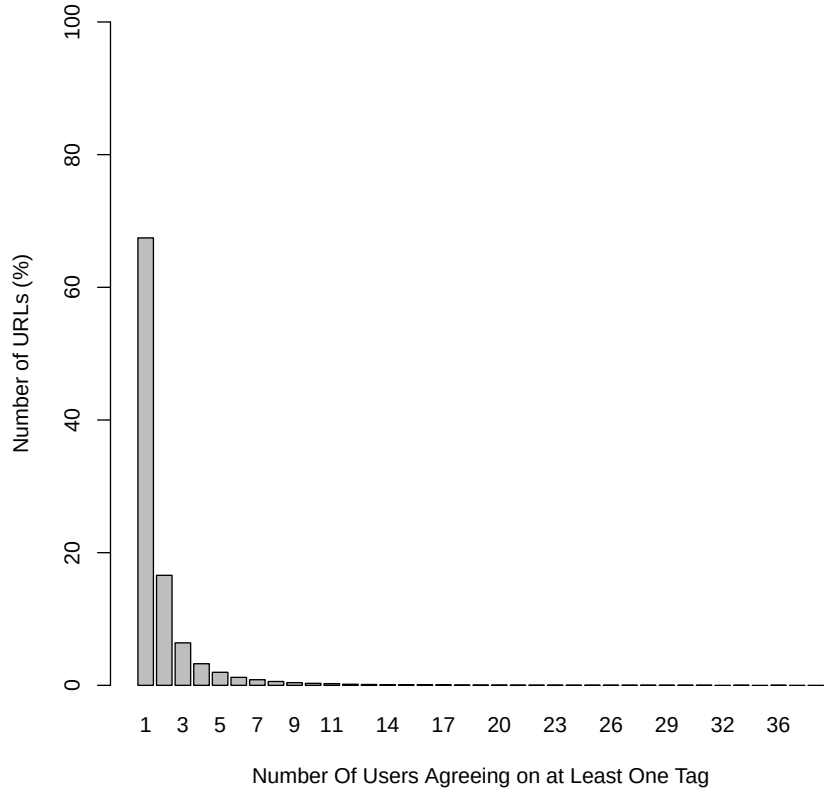


Figure 17: Average Agreement on Tags for the same Resource

It is interesting to note that these percentages apply on millions of tags, resources and users and in this, a small percentage still represent a large mass of resources and users on which automatic semantic extraction algorithms can be applied. Also, these figures were computed without any preprocessing of the different forms of tags, or without their disambiguation. As we show in the next section, this might be an important factor for the lack of overlap of tags between resources and users that we are seeing.

However, seeing these results, it is clear that there is a need to create better incentives for the users to provide annotations. In particular, they should be motivated to provide diverse annotation, but also annotations that create a consensus on the meaning of the resources as both these factors are important for leveraging the power of semantic search, navigation and knowledge learning.

The validation application for creating this dataset is available as open source code on the sourceforge repository of INSEMTIVES[22] and the first batch discussed here has been distributed as a LOD RDF dataset with the schema presented in Appendix A at `http://disi.unitn.it/~knowdive/dataset/delicious/`. This RDF export will grow as we extend the dataset with new samples of del.icio.us.

3.2.4 Consideration on the Dataset Controlled Vocabulary

While in the previous section we discussed some observations that could be made on the uncontrolled tags, we have developed a subset of these uncontrolled tags that are cleaned and disambiguated to a controlled vocabulary (see Section 3.1). It is thus interesting to analyse this subset to see the tagging behaviour when tags are disambiguated to the terms in a controlled vocabulary. In the following paragraphs we present some first conclusions on the use of a controlled vocabulary and how it maps to the users' vocabulary. In the following analysis, we only consider entries that were validated and agreed upon by two validators.

[22] `http://www.sourceforge.net/projects/insemtives/`

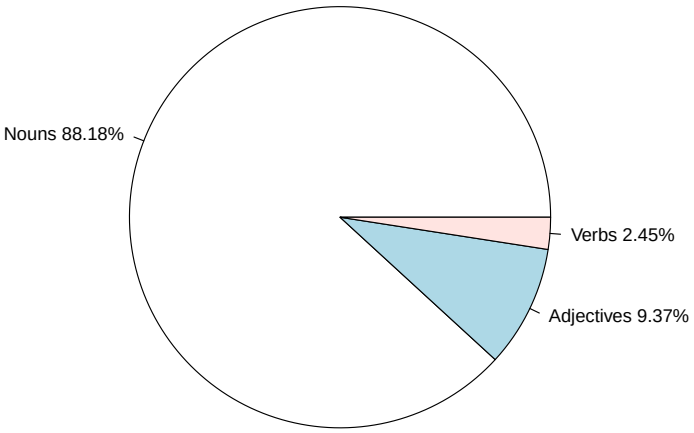


Figure 18: Distribution of Part of Speech on the validated Tokens

Use of Nouns, Verbs and Adjectives In a previous study [27] points out that the users of del.icio.us tend to use mainly nouns as descriptors of the urls. In the current dataset we have a validated sense (with all its metadata provided by WordNet) for each term and thus we can easily reproduce such observation.

Figure 18 shows that we can come to the same conclusions as [27]. In fact, nouns are used most of the times (88.18%) while verbs and adjectives, while being used sometimes cannot be found in great numbers in the annotations.

Note that Adverbs seem to be never used, at least in the sample of del.icio.us that we are studying.

Controlled Vocabulary vs. the Users' Vocabulary While disambiguating the tags to a sense in WordNet, the manual annotators could decide that no sense provided by the controlled vocabulary was adequate to express the sense meant by the user. For example, the tag "ajax" was found in the dataset and usually referred to the ajax technology used in web applications[23]. However, the only sense present in WordNet for this tag is "a mythical Greek hero".

As shown in Figure 19, the case of the missing sense happened in 35.8% of the cases. However, the validators were able to find a matching sense in WordNet for 48.7% of the terms used in the validated batch. For diverse reasons (the users use abbreviations, there is no sense in WordNet, etc.) less than half of the vocabulary used by the users can be mapped to the WordNet controlled vocabulary.

This is an important observation as it shows the inadequacy of fully automatic folksonomy processing systems based on fixed controlled vocabularies such as WordNet. For instance, if we consider the issue of Word Sense Disambiguation, the state-of-the-art tools cannot often achieve more that 60% accuracy. However, given the fact that only half of the terms from our dataset can be found in a vocabulary such as WordNet, from the end user perspective, it means that in much less number of cases than 60% the user will be suggested the right sense for a given tag token.

This is why we believe that relying on a purely static controlled vocabulary curated by experts is not a scalable option. First of all, we think that the automatic ontology learning field should be extended and find reliable methods to tackle this issue, and for this we propose some solutions in Section 4. We also believe that some strong incentives have to be found for users to help extend the existing controlled vocabulary.

[23] http://en.wikipedia.org/wiki/Ajax_(programming)

 ©INSEMTIVES consortium 2009-2012

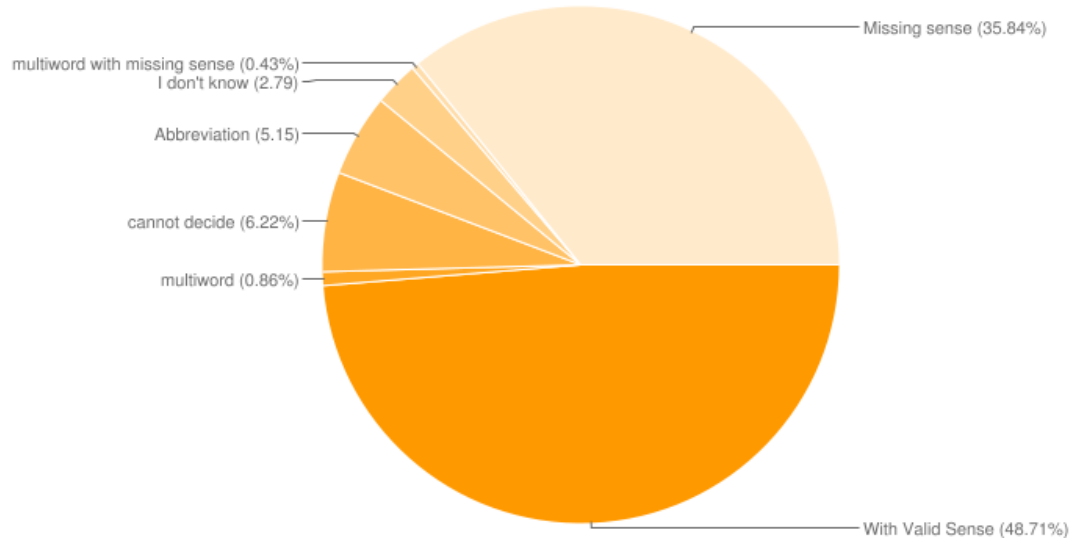


Figure 19: Distribution of Ignored Tokens (part of a Tag)

Figure 20: Decrease in the Amount of ambiguities after pre-processing and after sense disambiguation

Sense Disambiguation One of the issues presented in the raw analysis we discussed in Section 3.2.3 is that there is not a great agreement between users in the tags they use and there is not a great overlap in their personal vocabularies. One of the hypothesis for this is that there are many lexical variations of the same term that cannot be matched without preprocessing the tags (for example, “javaisland”, “java_island”, “java” and “island”, etc.) and as we have already discussed earlier, there are different terms that can be used for the same concept (for example, “trip” and “journey”).

In the validation process for the batch, we have actually cleaned all these issues by collapsing different lexical variations and linking them to their relevant concepts. We can thus evaluate the amount of ambiguity that is added by these different type of variations.

Figure 20 shows a summary of this decrease in ambiguity when going from *tags* – that can represent the same word in different forms – to *tokens* – that are preprocessed tags collapsed to the normal form of the world – and then to *synsets* – that disambiguate the meaning of the tag. The top bar represents the number of tags we started from (742), the middle bar represents the number of tokens to which they collapse (265) and the bottom bar represent the number of synsets from WordNet to which these tokens can be mapped.

We can thus see that by preprocessing alone (splitting and lemmatazing tags), the vocabulary size shrinks by 64.7%, thus reducing the ambiguity of the annotations significantly without the need to disambiguate them to the terms in a controlled vocabulary (e.g., a user searching for “blog” will be able to find bookmarks tagged with “blogs”, “coolblog”, “my_blog”, etc.).

The disambiguation provided by the linking to the controlled vocabulary, in the current batch does not actually provide a great amount of reduction in the vocabulary size. In fact, in the current batch, only seven tokens can be mapped to a smaller set of synsets. This means that there is not a great amount of synonymy in the tags that we have studied.

We believe that this is not a general feature of the full del.icio.us folksonomy and that synonyms will happen in a bigger number in different domains. We are now extending the size of our study batch to observe this hypothesis. In fact, in the current batch, the main topic was focused on computer and web technologies that use a very restricted vocabulary where words do not often have synonyms. We believe that this phenomenon might appear more often in less technical domains and are thus extending our study to the domains of cooking, education and travel.

3.3 Evaluating the Quality of Service in Semantic Annotation Systems

It is often argued that the quality of search would improve if the explicit semantic of the resources were known by the search engine [3]. In order to evaluate this improvement of the Quality of Service (QoS) of search in annotation systems such as del.icio.us, we implemented and evaluated the performance of a semantic search algorithm. The key difference from keywords-based search algorithms is that instead of using strings as query terms, the algorithm uses concepts from the controlled vocabulary and searches results in the semantically annotated dataset of del.icio.us discussed in Section 3.2.

We built queries from validated tag tokens, i.e., tokens for which an agreement on their meaning was reached amongst the validators. The key intuition here was that if the user used these tags to annotate web resources, then she is likely to use the same tags and in the same meaning to find these and other resources.

In order to implement searching, we built two indexes: a *keyword index* and a *concept index*. The keyword index contained mappings from tag tokens (e.g., "java") to all the resources annotated with this tag token (e.g., pages about the Java island but also about the programming language, the coffee beverage, etc). The concept index contained mappings from the concepts of the validated tag tokens to all the resources annotated with this tag token in the meaning represented by the concept (e.g., given the token "java" in the meaning of the Java island, the index would point to all resources about the java island but *not* about the programming language or about the coffee beverage). We used the dataset described in Section 3.2 that resulted into 377 entries in the concept index, 369 entries in the keyword index, and 262 resources pointed to in both indexes.

Given a number of tag tokens (which corresponds to the desired number of query terms) we built two queries: a *keywords-based query* and a *concepts-based query*. The keywords-based query is the conjunction of the token strings, whereas the concepts-based query is the conjunction of the corresponding validated concepts of the tokens. The keywords-based query was then executed on the keyword index. The results might be incorrect and incomplete due to, among other things, the following factors deriving from the ambiguous nature of the natural language:

Polysemy query terms may have an ambiguous interpretation. For instance, the term "Java" may refer to the *Java island* or to the *Java programming language*; thus, users looking for resources related to the programming language using this term may also get some irrelevant results related to the island (therefore, reducing the precision);

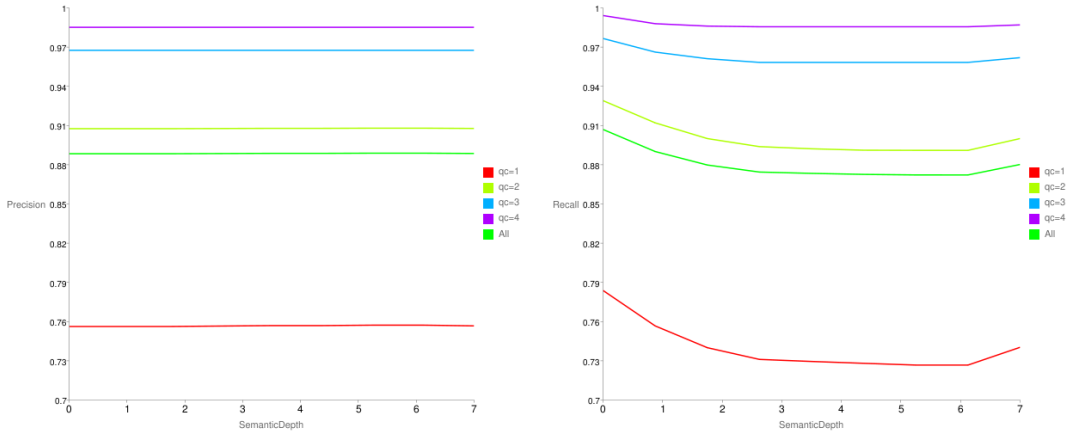
Synonymy Syntactically different query terms may have the same meaning. For example, query terms `image` and `picture` may be used interchangeably by users but will be treated by the system as two different query terms because of their different spelling; thus, retrieving resources using only one of the terms may yield incomplete results as the computer is not aware of the synonymy link;

Specificity gap This problem comes from a difference in the specificity of terms used in resource descriptions and queries. For example, the user searching with the term `cheese` will not find resources tagged with "cheddar[24]" if no link connecting these two terms exists in the system.

The concepts-based query was then executed on the concept index. The results were computed by matching concepts in the query to those in the index. Thus, a query with a particular concept would return all and only resources that have this concept amongst its tag tokens independently of any linguistic variation used to denote this concept in the tag token (e.g., synonymy, polysemy, as from above). Therefore, the results of concepts-based queries are *correct* and *complete* as long as the meaning of tag tokens in the resource annotations and of the terms in the concepts-based queries is properly disambiguated, which is the case for the analysed dataset due to its manual disambiguation as described in Section 3.2.

[24] which is a kind of cheese

 ©INSEMTIVES consortium 2009-2012



a) b)

Figure 21: Precision and Recall vs. Semantic Depth

In order to address the specificity gap problem, the concepts-based search described above was extended to support searching in more specific terms. In this we followed the approach described in [30]. In short, we introduced a variable *"semantic depth"* parameter that indicated the maximum distance between a query concept and a concept according to the `is-a` hierarchy of concepts in the underlying taxonomy in order for a resource annotated with such a concept to be considered as a query result for this query concept. For example, given the following path in the taxonomy: `transport` → `vehicle` → `car` and the query `transport`, then if the semantic depth parameter is set to 1, then resources annotated with the concept `transport` and/or with the concept `vehicle` will be returned as results; if this parameter is set to 2, then the resources annotated with the concept `car` will also be returned.

Number of Query Terms	Number of Queries	Average Number of Search Resutls	
		Concept Search	Keyword Search
1	2 062	9.25	8.87
2	2 349	4.38	4.20
3	1 653	2.40	2.32
4	1 000	1.44	1.44

Table 2: Number of Queries

Queries with different number of query terms and different values for the semantic depth parameter were generated and executed as described above (see Table 2 for details). Given that concepts-based queries, by construction, always yield correct and complete results, their results were taken as the golden standard for the evaluation of the performance of the keywords-based search. The measures of *precision* and *recall* were used for the evaluation. The results of the evaluation for these measures are presented in Figures 21a) and 21b) respectively.

As can be seen in Figure 21a), the precision of the keywords-based search with one query term is about 76%, i.e., about 24% of results may not be relevant to the user query. The precision improves for keyword queries with more query terms as the combination of more keywords disambiguates implicitly each keyword (e.g. if we search for the two terms "java island", resources about the programming language sense of "java" will rarely be returned). Note that, in theory, the precision of the keyword search should drop with the increase of the semantic depth. In deed, the concepts-based search should produce more results with a greater levels of semantic depth[25], and thus the number of resources that the keyword-based search should find to be the same should also increase. However, the keywords-based search will always produces the same number of results as

[25] see explanation of this feature in the previous paragraphs

it cannot solve the *specificity gap* issue without the conceptual taxonomy.

In practice, in the current evaluation golden standard, the different tags used on resources are very far apart in the taxonomy and thus increasing the semantic depth does not change the number of results returned by the concept search. This is a weak point of our evaluation dataset that is yet not big enough to show the *specificity gap* effect. We are currently working on extending this dataset to get more accurate results.

The recall of the keywords-based search with one query term and with the semantic depth of one is about 79%, i.e., about 21% of *correct* results are *not* returned by the query. With the increase of the semantic depth the recall decreases. This is explained by the fact that concepts-based search is capable of retrieving resources annotated with more specific terms than those used in the query, as discussed above. Therefore, concepts-based search returns more relevant results, whereas the keywords-based search always returns the same results, which leads to a lower recall. Again, as the number of query terms increases, the recall of the keywords-based search improves as the implicit semantic of the keywords is disambiguated by the other keywords.

As the evaluation results show, the introduction of formal semantics for tags and query terms allows to significantly improve the precision and recall of search in annotation systems such as del.icio.us. More evaluation is needed to be performed on larger datasets with a greater coverage of the controlled vocabulary in order to evaluate the impact in full extent.

4 Convergence and Evolutions Algorithms

As we explained earlier and in deliverable D2.2.1 [11] the semantic convergence algorithm we intend to develop should be able to understand the meaning the user wanted to use when an uncontrolled tag was added to a resource. This is slightly different from Word Sense Disambiguation (WSD) in that we are dealing with free text tags that we know are *not* in the controlled vocabulary and thus in addition to performing WSD, we need to find where to create a *new concept* in this controlled vocabulary. As we have shown in the preliminary study of the del.icio.us dataset (see Section 3.2.4), this case actually happens more than 50% of the time in a real use case.

While the computer does not "know" the actual meaning of the free-text tag used, the users always know the meaning they wanted to use when they tagged a resource. So if they tagged a bookmark with "java", in their mind, at the time of tagging, they knew exactly if they meant the "indonesian island" or the "programming language". This principle has already been widely illustrated in the automatic ontology building field where social network analysis methods were introduced [40] to extract the so-called "emergent semantics" [6].

[5] provides a good survey of the field of semantic discovery in folksonomies and we recommend the reading of this article to understand better the field. Here we will concentrate on the main principles used in the different approaches, their weak points and how we are different from them.

The method used to extract the semantics from folksonomies is what is called *tag clustering* and its principle is based on machine learning clustering algorithms [51]. This clustering is based on the principle that similar tags will have the same semantic and can thus be attached to the same *"concept"* in the created vocabulary. For instance, if the algorithm finds out that "opposition" and "resistance" are similar, then it can associate it to one concept for that meaning. One of the main issue is thus to compute the similarity between tags to run the clustering algorithms that will attach similar tags together. To do this, all the methods available currently use a mix of measures based on the collocation of tags on resources and their use by users. If two tags are often used by the same user on different resources or by the different users on the same resource, then they can be considered similar.

This assumption on the computation of the similarity of tags is, in our opinion, one of the first weak points of these approaches as it makes the assumption that one tag can only have one meaning. Thus these algorithm are good to find synonyms of the most popular sense but cannot deal with the polysemy of the words. For example, if the tag "java" is collocated with "indonesian island" on 200 resource and with "programming language" on 1000 resources, then it will be considered to be similar to the latter and the fact that it has a second meaning is lost. However, [55] show that tags are often ambiguous in folksonomies (their study is also based on del.icio.us) and can bare more than one meaning. In the algorithm we propose, we add an extra step to the clustering to first identify the diverse senses of polysemous tags and in the following clustering steps, we do not consider tags directly, but the unique word senses that they can take (see Section 4.1).

In [26], the authors have shown that by combining the usual folksonomy collocation measures with con-

 ©INSEMTIVES consortium 2009-2012

tent based measure (such as SIFT [39] based similarity between images), more information can be detected about each tag. While there are not enough resources in the INSEMTIVES project to extensively study the applicability of this to our algorithms, we would recommend future research to go in this direction.

We have discussed, until now, our "semantic convergence" approach in the perspective of the semi-automatic construction and extension of a controlled vocabulary from the uncontrolled annotations. Thus creating new knowledge based on the consensus on the meaning of tags reached within the community. However, the expansion of such a controlled vocabulary poses the issue of evolution of the existing annotations. In particular, the semantic convergence algorithm might introduce new, more specific, concepts in the controlled vocabulary taxonomy as illustrated in Figure 22. This might also happen when the users manually extend the shared controlled vocabulary and introduce new concepts.

When some new concepts $\{C_N1, C_N2, C_N3, \ldots\}$ are introduced under the concept C_K, we cannot ask the user to re-annotate all the resources that have been already annotated with a controlled annotation linking to C_K to these more specific concepts. However, keeping the annotations always linked to a concept as specific as possible is recommended to leverage the semantic services (such as semantic search for instance, as discussed in Section 3.3). We are thus proposing to reuse some of the techniques used in the convergence algorithm to classify the existing annotations into the most appropriate new concept.

4.1 Algorithms

We propose to adopt a parametric based clustering approach a bit different from the standard KMeans and KNN algorithms that are often discussed in the state-of-the-art of ontology construction from folksonomy (see, for a review [5]). In fact, these algorithms, while being the most popular in the clustering field, are not well tailored to our application domain as they take as an input parameter the number of expected clusters (the K in the name). The state-of-the-art approaches on ontology building from folksonomies cluster all the tags together to find all the concepts that they represent (see figures two and four in the review of Garcia-Silva et al. [5]). In this case, they can optimise the K parameter to find the best overall number of clusters for their dataset. However, in our approach, we have added an extra step where clustering is applied to detect the different senses of one term. In this case, we cannot find an overall optimal value for the number of clusters to look for as each term might have a different number of senses.

Thus, we need to use a clustering algorithm that can work without this parameter as input. We have decided to use the DBScan algorithm [28] to do a density based clustering. This approach to clustering has various advantage for our application:

- it does not require as input the number of clusters to be found. Instead it takes two parameters: ϵ, the minimum distance between two items to put them in the same cluster and m the minimum number of items in a cluster.

  ϵ is easier to optimize in our use case than to compute the K parameter as we can find it by studying the annotated dataset that we are currently building. If we assume that we also have a mix of controlled and uncontrolled tag, we can optimise this parameter dynamically as the annotations base grows.

  m can also be optimized on the annotated dataset that we are building with the del.icio.us data and the minimum number of resources, and senses to be used can be studied to optimize the accuracy of the algorithm (see the evaluation in Section 4.2);

- while the KMean and KNN algorithms assign each item in the clustering space to a cluster, the DBScan algorithm can decide that some of the items to be clustered are just noise and should not be considered. This is very important in our application domain as it allows to leave out very personal or subjective uses of a term that might not be aligned with the rest of the community understanding of the term; and

- the DBScan algorithm can detect clusters that have more complex shapes than the standard hyperspherical clusters returned by vector quantization based clustering such as the KMeans and KNN[51].

While there is already some research done on diverse similarity measures applicable to concept detection and learning in the Natural Language Processing field (for instance [8] or [33]), the existing clustering techniques discussed in the folksonomy field are only considering raw collocation counts (of tags, resources or users) as a similarity measure between tags. For instance, [8] proposes to combine four different measures

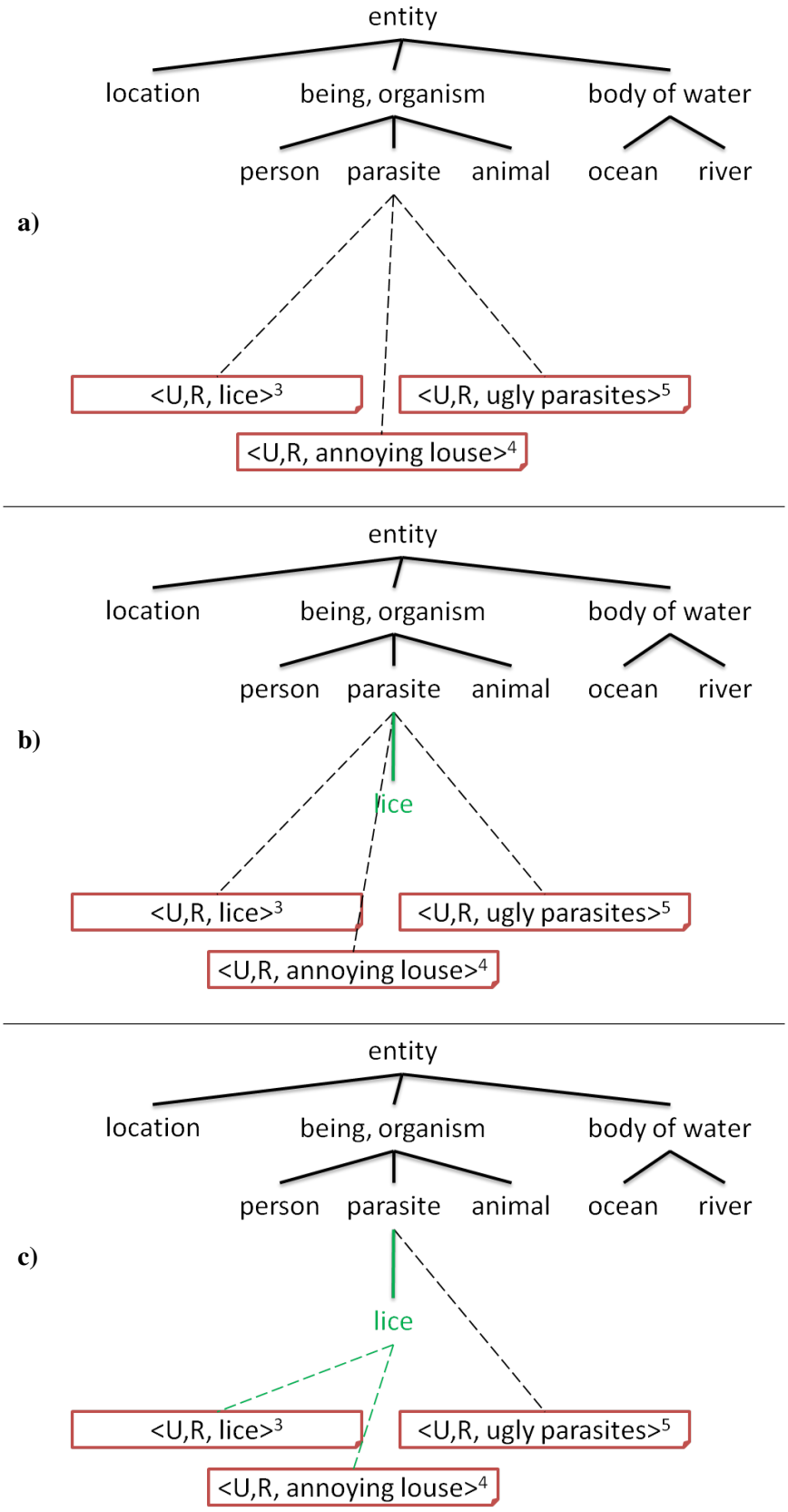


Figure 22: Evolution of Annotations: **a)** Set of controlled annotations; **b)** The underlying controlled vocabulary changes by adding *lice* as a child of *parasite*; **c)** The controlled annotations need to be checked, and the corresponding tags need to be updated to poing to the newly created concept.

 ©INSEMTIVES consortium 2009-2012

to compute sense similarities: the *topic signature*, the *subject signature*, the *object signature* and the *modifier signature*. While most of these measures can only be applied to textual documents as they require to know noun-verb relationships in a sentence, the *topic signature* is interesting in the domain of folksonomy where one of the only context we have for computing the distances is the list of collocations. However, these collocations can be considered and weighted in different way and [33] points out that simple vector distances or cosinus distances between *topic signatures* are not always powerful enough. The authors show that information based measures – such as the Kullback-Leibler divergence of word distribution, the mutual information – can be used to have more powerful measures of semantic distances between concepts based on the Distributional Semantics principles. The authors of [48] have proven that this measure can be applied with success to the domain of folksonomies to disambiguate tag senses.

For the algorithms of convergence and evolution that we discuss in this section (see Alg. 1 and Alg. 2 we use clustering algorithms relying on distance measures between User-Resource pair and between tag senses. We are currently experimenting with different measures, from the standard tag collocation measures proposed in the current state of the art to the more advanced distributional measures described above.

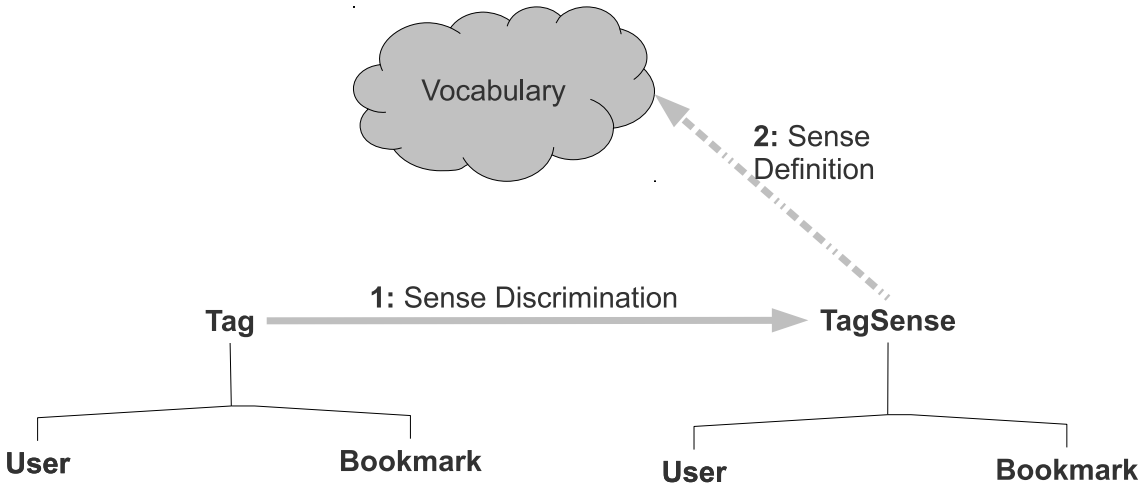


Figure 23: Sense-User-Bookmark Tripartite graph

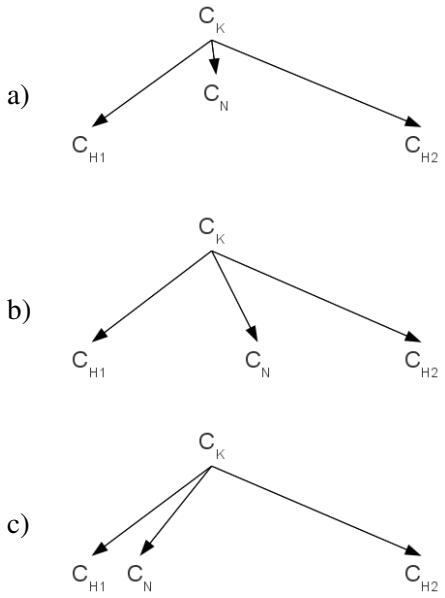
To enrich the controlled vocabulary with a new concept from an uncontrolled tag, we propose to do the concept detection in three stages as described in the Algorithm 1:

1. For each tag, we cluster the user-resource bipartite graph that are attached to this tag. By doing so, as was hinted by [14], we can discover the different meaning of the tag. By considering each cluster to be a tag sense, we can replace the tag in the user-resource-tag tripartite graph by its senses and the tripartite graph becomes a user-resource-sense graph as illustrated in Figure 23.

2. We can now apply the same principle as the one discussed in the-state-of-the-art on the user-resource-sense tripartite graph to cluster similar senses together (see [5] for a review). In this way, if we consider our previous example, the tag "java" will be split in two senses: `java-1`, similar to "indonesian island" and `java-2`, similar to "programming language".

3. Once the tag senses have been clustered together, we can identify new concepts for each of the clusters. This process is equivalent to finding the relation (in particular hypernym/hyponym relations) of the new concept (represented by the cluster of tag senses) in the controlled vocabulary. This can be achieved in two ways in the INSEMTIVES setup:

    - the first solution is provided by the mixed model of annotations. As we have uncontrolled tag annotations as well as controlled tag annotations, we consider the collocated controlled annotations to the tags of the cluster we are considering as possible hypernym or hyponym of the new concept. For example, if we have a new tag sense `mediterranean-1` which is often collocated with the *known* controlled tag linking to the concept `sea-watermass` then we can place this new concept under the know concept;



$D_{N-K} \leq D_{N-H}$
$D_{N-K} < \frac{1}{\rho} \times D_{K-H}$

C_K and C_N are "synonymous"

$D_{N-K} \leq D_{N-H}$
$D_{N-K} \geq \frac{1}{\rho} \times D_{K-H}$

C_K is more general of C_N

$D_{N-K} > D_{N-H}$

C_N should be placed under C_{H1}

Figure 24: Decisions to Extend the Concept Taxonomy

- the second solution requires taking a hierarchical classification approach similar to the one proposed in by [8]. In their approach to ontology building, they consider a similarity measure between a *known* concept C_k in the vocabulary and a new concept C_n. If the distance between these two concepts is smaller than the distance between C_n and any of the hyponyms of C_k, then C_n is considered to be the hyponym of C_k. Otherwise, they continue the search down the conceptual hierarchy. We alter this approach by splitting it in three cases as we believe that there can also be cases in which the new concepts C_n are actually synonyms of an existing concept C_k. The updated solution is as follows:
  - if the new concept C_n is closer to the existing concept C_k than to any of the its children concepts, but much more similar to C_k than any of its child concepts, then we say it is most likely that C_n is a synonym of C_k (Figure 24 a));
  - if the new concept C_n is closer to the existing concept C_k than to any of the its children concepts, but not much more similar to C_k than any of its child concepts, then we say it is most likely that C_n is more specific than C_k (Figure 24 b));
  - if the new concept C_n is closer to the a child concept of C_k than C_k, then we continue our search under this most similar child concept (Figure 24 c));

  We can apply this search procedure on our controlled vocabulary, starting from the root of its conceptual hierarchy.

The two algorithms presented in listings 1 and 2 are parametric as they depend on the values of ρ and θ. ρ specifies the threshold to decide if a new concept is more specific than an existing concept or is just a synonymous that extends its synset. θ is a similar threshold but alows to decide when an existing annotation has to be evolved towards a new more specific concept. This parameters will be different depending on the specific application domain and will decide of how much specific the controlled vocabulary will get. We are currently running evaluations to show the behaviour of these algorithms with different values of the parameters and will report on these in future publications.

4.2 Evaluation

While there is existing research on the automatic construction of ontologies from folksonomies, [5] points out that there is not yet any agreed evaluation dataset. In fact, from our knowledge of the state of the art approach, there is not yet an appropriate evaluation methodology in the field. As pointed out earlier, this is mostly due to the lack of a golden standard evaluation dataset and thus the evaluation of the existing methods were often only evaluated *subjectively* [38] by checking manually some extracted clusters [29, 46, 43].

 ©INSEMTIVES consortium 2009-2012

Algorithm 1 convergence(ST, ρ, θ)

Require: $Annot$ {⟨User,Resource,Tag⟩, set of all (uncontrolled and controlled) tag annotations}
Require: $annot \in Annot$ {one tag annotation}
Require: $Annot^S$ {⟨User,Resource,Tag$_{sense}$⟩, set of all the tag annotations which have an attached tag sense}
Require: $annot^S \in Annot^S$ {one tag sense annotation}
Require: $Terms^U | Term \in Annot^U$ {The set of all the uncontrolled terms used as uncontrolled tag annotations}
Require: CV {and existing Controlled vocabulary}
Require: $C_K \in CV$ {Known concept in CV}
Require: $C_N \notin CV$ {New concept not in CV}
Require: $Rel = \langle C_N, C_K, R \rangle$ {Relation between the new concept C_N and the known concept C_K with the given relation $R \in \{is-a, synonym\}$}
Require: TS {tag synset or set of synonymous tags}
{Step 1: Compute the tag senses}
for all $\alpha \in Term^U$ **do**
$Annot_\alpha \in Annot$ = subset of $Annot$ created by users using α or resources annotated with α
C_α = $\{annotCluster\} = \text{cluster}(Annot_\alpha)$ {each annotCluster is a subset of of $Annot^U_\alpha$ }
$Senses_\alpha = \emptyset$
for all $cluster^i_\alpha \in C_\alpha$ **do**
S^i_α = new tagsense for T_α discriminated in $cluster^i_\alpha$
$Senses_\alpha = Senses_\alpha \cup S^i_\alpha$
replace T_α by S^i_α in $cluster^i_\alpha$ {this converts an $annot$ = ⟨User,Resource,Tag⟩ into $annot^S$ = ⟨User,Resource,Tag^S⟩}
end for
end for{The output of this part is the split of single Terms into their Senses}
{Step 2: Compute the tag synsets}
$C^{Annot^S_\alpha} = \{cluster_{annot^S}\} = \text{clusterSenseAnotations}(Annot^S_\alpha)$
$NewTSs = \emptyset$
for all $cluster^j_\alpha \in C^{Annot^S_\alpha}$ **do**
$\{S_\alpha\} = \text{SensesIn}(cluster^j_\alpha)$ {$SensesIn$ is a function that returns the senses $S \; \forall \; annot^S \in cluster^j_\alpha$ }
$TS = \emptyset$
for all $S^K_\alpha \in \{S_\alpha\}$ **do**
$TS = TS \cup T_\alpha$
end for
$NewTSs = NewTSs \cup TS$
end for
{Step 3: Place the new concepts in the existing controlled vocabulary}
for all $TS \in newTSs$ **do**
$Rel = \emptyset$
if $\exists ct \in TS$ **then**
$C_K = C \in ct$
C_N is a new concept TS
$Rel = \langle C_N, C_K, is-a \rangle$
else
$Stack = \emptyset$
$C_K = \text{root}(CV)$ {assign the root of the Controlled Vocabulary to C_K}
$\text{push}(Stack, C_K)$
while $Stack \neq \emptyset$ & $Rel = \emptyset$ **do**
C_K = $\text{pop}(Stack)$
H = more-specific(C_K) {set of more specific concept than C_K}
if $H = \emptyset$ **then**
$Rel = \langle C_N, C_K, is-a \rangle$
apply evolution(C_K, C_N)
else
$D_{N-K} = \text{distance}(C_N, C_K)$
$D_{N-H} = \arg\min_{C_i \in H} \text{distance}(C_N, C_i)$ {minimum distance between C_N and the more specific concepts of C_K}
$D_{K-H} = \arg\min_{C_i \in H} \text{distance}(C_K, C_i)$ {minimum distance between C_K and its more specific concepts}
if $D_{N-K} > D_{N-H}$ **then**
C_K = more specific concept with the smallest D_{N-H}
$\text{push}(Stack, C_K)$
else if $D_{N-K} < \frac{1}{\rho} \times D_{K-H}$ **then**
$Rel = \langle C_N, C_K, synonym \rangle$
else
$Rel = \langle C_N, C_K, is-a \rangle$
apply evolution(C_K, C_N, θ)
end if
end if
end while
end if
$CV = CV \cup Rel$
end for

Algorithm 2 evolution(C_K, C_N, θ)

Require: $ann^c = \langle u, r, ct = \langle t, \{lc\}\rangle\rangle$ {is a controlled tag annotation as in (**MO10**) and ct is a controlled term as defined in (**MO23**)}

```
CA = annotationsUsing(C_K)
D̄_K = arg mean_{ann_i^c ∈ CA} distance(ann_i^c, C_K))
sd = arg stdev_{ann_i^c ∈ CA} distance(ann_i^c, C_K))
for all ann^c ∈ CA do
  if |D̄_K − distance(ann^c, C_N)| > θ × sd then
    update lc in ann_c to link to C_N
  end if
end for
```

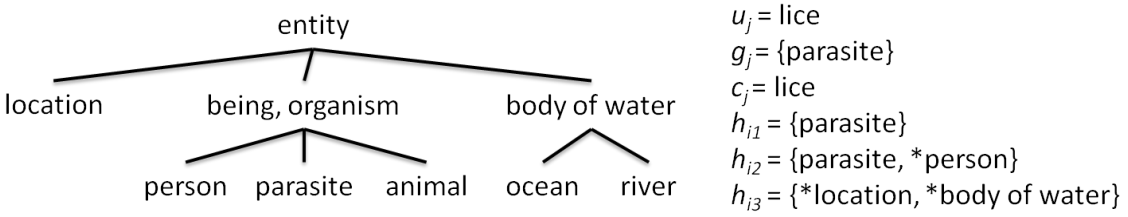


Figure 25: Example of taxonomy, an unknown relevant concept u_j, its correct generalisations g_j and the generalisations proposed by three hypothetical algorithms h_{ik} (from [9])

However, as pointed out earlier, the NLP field has already tackled the issue of concept extraction from text and has considered different evaluation measures for this task. [9] describes the evaluation problem as follows:

> Let us suppose that we have a set of unknown concepts that appear in the test set and are relevant for a specific domain: $U = \{u_1, u_2, \ldots, u_n\}$. A human annotator has specified, for each unknown concept u_j , its maximally specific generalisations from the ontology: $G_j = \{g_{j,1}, g_{j,2}, \ldots, g_{j,m_j}\}$.
>
> Let us suppose that an algorithm decided that the unknown concepts that are relevant are $C = \{c_1, c_2, \ldots, c_l\}$. For each C_i, the algorithm has to provide a list of maximally specific generalisations from the ontology: $H_i = \{h_{i,1}, h_{i,2}, \ldots, h_{i,p_i}\}$. (See Figure 25)

From this definition, a number of evaluation metrics can be computed:

Accuracy the amount of correctly identified hypernyms,

Parsinomy the amount of concepts for which a correct set of generalisations is identified,

Recall the amount of concepts that were correctly and to which at least one relevant hypernym was found,

Precision the ratio of concepts that were correctly attached to their hypernym to the total of concepts identified

Production the amount of proposed hypernyms per concept.

Learning Accuracy the distance, in the concept hierarchy, from the concept proposed placement to its true placement (from [32]).

As can be seen from this proposed measures, a golden standard needs to be available that provides the "maximally specific generalisations" (G_j) for each concept (U). Alfoncesca and Manandhar [9] use a dataset of textual documents that is manually annotated for this purpose. However, for our application in the annotation domain, we need to evaluate the algorithm within a folksonomy. To do this, we can use the evaluation dataset described in Section 3 as it provides a manually validated disambiguation for each tag in a the del.icio.us folksonomy. The dataset contains the disambiguation from the free text tag to its corresponding concept in a controlled vocabulary, in this case WordNet.

The measures listed above can be measured on this dataset by applying a leave one out approach to the evaluation. That is, we iterate through all controlled tag annotations available in the golden standard; we "forget" the senses of one tag at a time and apply the convergence algorithm on this tag; we can then compare

 ©INSEMTIVES consortium 2009-2012

the senses and their place in the taxonomy that are generated for this tag to the real sense that the golden standard defined.

While this is a good evaluation procedure to evaluate the final output of the convergence algorithm, the current dataset is not in a form that allows to evaluate the intermediate results. In particular, to optimise the ϵ and m parameters of the clustering algorithm, we have to be able to evaluate the accuracy of such an algorithm. In the same way, we need to be able to evaluate the distance metrics used and compare different approach. For this, we need a clustering golden standard, that provides the "true cluster" (class) of each user-resource pairs in the dataset so that we can compare the found clusters to this golden standard results. In the following paragraphs we discuss a strategy to generate such a clustering golden standard.

Let us assume that we have a manually validated dataset of annotations with their corresponding concepts in a Controlled Vocabulary. We assume this manually validated dataset to be the golden standard GS, where the set of unknown concepts $U = \{u_1, u_2, \dots u_n\}$ to be clustered are the terms used in the annotation; for each unknown concept u_j the maximally specified generalization G_j from the ontology is the parent concept in CV. In this case, the set of unknown concepts that the algorithm decides to be relevant (C) are the same as the manually validated ones in U, as this set will be generated by the procedure described bellow. Finally, for each $u_i \in U$[26] the clustering algorithm produces a set of results $H_i = \{h_{i,1}, h_{i,2}, \dots, h_{i,p_i}\}$ to be compared with the expected results in G_j.

In our case, GS is the validated dataset presented in Section 3. The set of unknown concepts U will be the terms t in the annotations. The set of maximally specified generalization G_j is the set of parent concepts of the manually disambiguated terms u^i, considering the hypernym relation in WordNet. The construction of the golden standard consists in aligning the set of concepts U for which we have the resulting clusters G_j and the set of concepts C_i to be given as input to the clustering algorithm, we will describe the process bellow.

When building the golden standards GS^k we want to automatically generate the set of unknown concepts (U and C_i) to be clustered, their classes, and the maximally specified generalization G_j. In order to do so, we perform the following steps:

1. we define G_j to be a concept in our Controlled Vocabulary (CV) for which there is more than one child concept[27] that has more than one manually validated associated term in the annotation[28]. In our example in Figure 26 **a)**, the concept "being, organism" has two children concepts ("person" and "parasite") that contain more than one annotation attached to it[29], also the concept "body of water" contains two children concepts ("ocean" and "river") that have more than one annotation attached to it. Each of the complying concepts ("being, organism" and "body of water") will generate a golden standard evaluation dataset $GS^1_{being,organism}$ and $GS^2_{bodyofwater}$.

2. forget momentarily that each of the children concepts of G_j exist, which will convert our controlled tag annotation (MO10) into a uncontrolled tag annotation (MO7), and therefore, the input C_i for the clustering algorithm. Since for each of these children C_i we know their corresponding annotations, we create a class for each deleted concept, and define the boundary of the GS^k to these particular classes, i.e., as mentioned previously, each set of siblings will produce a new golden standard GS^k. In our example in Figure 26 **b)**, we can see that two golden standard have been created: GS^1 for "being, organism" and GS^2 for "body of water", each of them containing two classes (one for each deleted concept).

3. We can repeat the process by further "forgetting" concepts higher in the hierarchy and thus creating more golden standard sets of increasing difficulty as the higher we go in the hierarchy, the more classes will be created by GS^k. In our example in Figure 26 **c)**, we further forget the concepts "being, organism" and "body of water" and create another golden standard GS^3 for "entity", creating four classes.

If we apply the above mentioned process on the Dataset depicted in Figure 26 we would obtain three GS dataset as shown in Table 3:

The purpose of each golden standard GS^k is twofold:

[26] or c_i, as they are the same

[27] for example, using the is-a relation

[28] i.e., considering that the annotation is manually validated, we know the relate concept in the CV

[29] It is assumed that the annotation is linked to the Controlled vocabulary using MO10 (Controlled tag annotation), therefore the annotation is linked to the controlled vocabulary via the used term

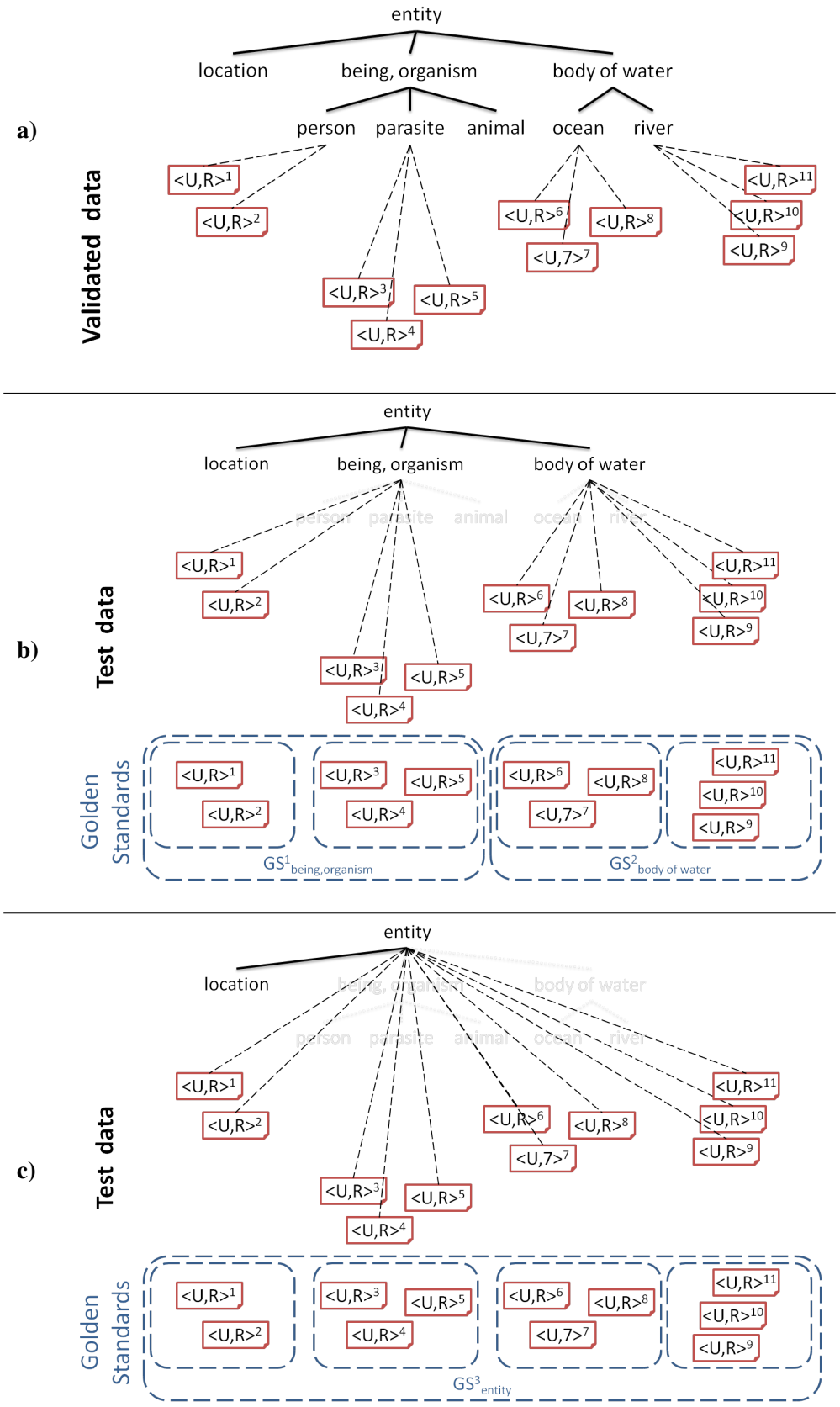


Figure 26: Process of constructing an evaluation dataset for a clustering algorithm in folksonomies: **a)** The validated data for which the concepts is known, e.g., the Semantified del.icio.us dataset presented in Section 3.2; **b)** Creation of two validation clusters by deleting the children of *being, organism* and *body of water*; **c)** Creation of a third validation cluster by further deleting *being, organism* and *body of water*.

 ©INSEMTIVES consortium 2009-2012

GS^k	C, U	G_j	Classes and new concepts
$GS^1_{being,organism}$	$C = U =$ {person, parasite}	$G_1 =$ {"being, organism"}	person $= \langle U, R\rangle^1, \langle U, R\rangle^2$ parasite $= \langle U, R\rangle^3, \langle U, R\rangle^4, \langle U, R\rangle^5$
$GS^2_{bodyofwater}$	$C = U =$ {ocean, river}	$G_2 =$ {"body of water"}	ocean $= \langle U, R\rangle^6, \langle U, R\rangle^7, \langle U, R\rangle^8$ river $= \langle U, R\rangle^9, \langle U, R\rangle^10, \langle U, R\rangle^{11}$
GS^3_{entity}	$C = U =$ {person, parasite, ocean, river}	$G_3 =$ {"entity"}	person $= \langle U, R\rangle^1, \langle U, R\rangle^2$ parasite $= \langle U, R\rangle^3, \langle U, R\rangle^4, \langle U, R\rangle^5$ river $= \langle U, R\rangle^9, \langle U, R\rangle^10, \langle U, R\rangle^{11}$ river $= \langle U, R\rangle^9, \langle U, R\rangle^10, \langle U, R\rangle^{11}$

Table 3: Resulting golden standards GS^k for the evaluation of the semantic convergence algorithm.

First: Evaluate step one of the semantic convergence algorithm presented in the previous Section 4.1, where the input is a set of uncontrolled tags C and the output is a set of clusters of similar tags which represent a new concept. In our example in Figure 26, we would be calling the clustering algorithm with each of the golden standard GS^k. Then, to compute the accuracy of the clustering, we could compare the produced results H_i with the classes of the golden standard with standard cluster evaluation metric such as Purity, Accuracy and Precision/Recall.

Second: Considering that we know the parent concept G_j for each golden standard GS^k, we can also evaluate step 3 of the semantic algorithm where for each cluster produced a new concept also has to be added to the CV as more specific than a existing concept in CV. The concept created by the semantic convergence is compared to the name of the cluster (the forgotten concept), and the parent concept in CV designated by the semantic convergence algorithm should be compared to G_j. In our example in Figure 26, if we pass GS^1 to the semantic convergence algorithm, it should create concepts for "person" and "parasite", and put them as child concepts of "being, organism".

We are currently running evaluations of different distance metrics and parameters to our algorithms by applying the novel evaluation methodology described here. However, the currently annotated sample from del.icio.us is yet too small to report significant results. Indeed, to get interesting results for the clustering, each class (i.e. new tag sense) should contain a minimum number of user-resource pairs. Currently, our manually annotated dataset contains 500 user-resource pairs for 742 different tags distributed on 258 senses, which actually translates by a mean of 2.53[30] user-resource pairs per sense with a maximum of 57 user-resource pairs for the biggest class. We are now extending this golden standard and will report on results of this new approach in upcoming publications.

5 Summarization Algorithm

In the evaluation of the del.icio.us dataset presented in Section 3, we have found that fully automatic Word Sense Disambiguation is particularly hard, as was already pointed out in the state of the art [53]. It is indeed difficult to reach precision values over 66%. However, in the INSEMTIVES project, we are interested in *semi-automatic* methods that can help the users in providing more semantic annotations as discussed in deliverable D2.2.1 [10]. Thus, a good way to approach the WSD issue is to provide an adequate user interface so that the users can provide the right sense for a term directly when inserting an annotation [10]. However, when dealing with polysemous words[31], displaying a simple term is not enough and a visual disambiguation of the term is required. Currently, the only available way to perform such disambiguation is by displaying the definition of the term, for example, as shown in Figure 9. For instance, for the "java" term, that would be the three definitions[32]:

- Java (an island in Indonesia to the south of Borneo; one of the world's most densely populated regions)
- coffee, java (a beverage consisting of an infusion of ground coffee beans) "he ordered a cup of coffee"
- Java (a platform-independent object-oriented programming language)

[30] standard deviation: 5.27

[31] e.g. "java" has at least three senses in the WordNet controlled vocabulary.

[32] from WordNet.

Displaying such long definitions in an annotation interface is not always practical as the screen real estate is limited. In this section, we propose a *summarization* algorithm that can produce a "single word" disambiguation label for a term. For instance, for the "java" term, we would have the three following summaries:

- Java – island
- java – beverage
- Java – programming

These disambiguating labels do not provide a full definition but are sufficient for the user to disambiguate between all the ambiguous senses of a single word.

While we have found a large body of research about free text summarisation, the problem statement is very different and the available work in that field is not relevant to our domain. There is also some state of the art in synonymous detection, comparable to the ontology building research track discussed in Section 4, however, here again the approaches are far from our problem statement. As far as we know, there is not yet any work on this issue. In the following section, we describe the algorithm that we developed and the evaluation that was performed to validate our heuristics.

5.1 Algorithm

We consider that each Part Of Speech (POS) – such as noun, verb, adjective, adverb – can have a different summary, and that the summarization algorithm might need different heuristic depending on the POS. We thus explore the summarization algorithm considering the particular characteristics of each of them. In the following paragraphs we provide the characteristics of each of the POS, and report some results based on WordNet 2.1.

5.1.1 Noun summarization

In WordNet 2.1 there is a total of 117 097 nouns, out of which 13.47% (15 776) have more than one sense, referring to 43 783 different senses, giving us an average of 2.7 senses per an ambiguous noun. In order to create a summary label for a given ambiguous noun, we defined four heuristics, choosing (in the presented order) the first available heuristic among the following:

1. return the first shortest unused lemma among the available words of the same synset;
2. return the first shortest unused lemma among the available words in the hypernym's (parent) synset;
3. return the first shortest unused lemma among the available words in the hyponyms' (child) synsets;
4. if there are no hypernym synsets available, return the noun itself (original token)

We always use the "first unused lemma" when possible as different sense might have the same terms in their synset and thus we choose only the first term we can find that has not yet been selected as a summary for another of the senses.

We rely on the length of the lemma and choose the shortest among the available choices for two main reasons: first, very often the shortest word is the simplest one and second, to save the screen space (altering as little as possible the normal flow of the annotation process). There are a few cases (1.48%) where several senses share the same summary; in particular, some of them (0.82%) have the same summary for all the senses – in which cases it is impossible to help the user disambiguate with such a summary and the definition of the synset will have to be used. Roughly half of the ambiguous noun senses produce a shorter summary label, on the average 2.31 characters, and in the other half of the cases (22001 out of 43783 senses) the summary label produced is longer than the original word, namely, on the average 4.84 characters longer. Table 4 shows some examples of summary labels for nouns.

 ©INSEMTIVES consortium 2009-2012

Sense	Gloss	Obtained via	Summary
triangle#1	a three-sided polygon	SYNSET	trigon
triangle#2	something approximating the shape of a triangle; ”the coastline of Chile and Argentina and Brazil forms two legs of a triangle”	HYPERNYMS	form
triangle#3	a small northern constellation near Perseus between Andromeda and Aries	SYNSET	Triangulum
abacus#1	a tablet placed horizontally on top of the capital of a column as an aid in supporting the architrave	HYPERNYMS	tablet
abacus#2	a calculator that performs arithmetic functions by manually sliding counters on rods or in grooves	HYPERNYMS	calculator
circle#1	ellipse in which the two axes are of equal length; a plane curve generated by one point moving at a constant distance from a fixed point; ”he calculated the circumference of the circle”	HYPERNYMS	oval
circle#2	an unofficial association of people or groups; ”the smart set goes there”; ”they were an angry lot”	SYNSET	set
circle#3	something approximating the shape of a circle; ”the chairs were arranged in a circle”	HYPERNYMS	form
circle#4	movement once around a course; ”he drove an extra lap just for insurance”	SYNSET	lap
athens#1	the capital and largest city of Greece; named after Athena (its patron goddess); ”in the 5th century BC ancient Athens was the world’s most powerful and civilized city”	SYNSET	Athinai
athens#2	a town in southeast Ohio	HYPERNYMS	town
athens#3	a university town in northeast Georgia	HYPERNYMS	town

Table 4: Summarization Examples for Nouns

5.1.2 Verb summarization

In WordNet 2.1 there is a total of 11 488 verbs, out of which 45.49% have more than one sense, referring to 18 629 different senses, giving us an average of 3.56 senses per an ambiguous verb. In order to create summary label for a given ambiguous verb we defined four heuristics, choosing (in the presented order) the first available heuristic among the following:

1. return the first shortest unused lemma among the available words of the same synset;
2. return the first shortest unused lemma among the available words in the hypernym’s synset;
3. return the first shortest unused lemma among the available words in the hyponyms’ synsets;
4. return first word of the gloss (often well-known verb such as “to cause”, “to have” or “to be”).

In a few cases (1.38%) where several senses share the same summary and some of them (0.23%) have the same summary label for all senses. In a majority of cases (65%) the summary label is, on the average, 1.84 characters shorter, while in 35% of cases the produced summary label is, on the average, 2.59 characters longer. Table 5 shows some examples of summary labels for verbs.

Sense	Gloss	Obtained via	Summary
abstain#1	refrain from voting	HYPERNYMS	refrain
abstain#2	choose not to consume; "I abstain from alcohol"	SYNSET	desist
accost#1	speak to someone	SYNSET	address
accost#2	approach with an offer of sexual favors; "he was solicited by a prostitute"; "The young man was caught soliciting in the park"	SYNSET	hook
add#1	make an addition (to); join or combine or unite with others; increase the quality, quantity, size or scope of; "We added two students to that dorm room"; "She added a personal note to her letter"; "Add insult to injury"; "Add some extra plates to the dinner table"	HYPERNYMS	increase
add#2	state or say further; "'It doesn't matter,' he supplied"	SYNSET	append
add#3	bestow a quality on; "Her presence lends a certain cachet to the company"; "The music added a lot to the play"; "She brings a special atmosphere to our meetings"; "This adds a light note to the program"	SYNSET	lend
belong#1	be owned by; be in the possession of; "This book belongs to me"	HYPERNYMS	be
belong#2	originate (in); "The problems dwell in the social injustices in this country"	SYNSET	dwell
belong#3	be in the right place or situation; "Where do these books belong?"; "Let's put health care where it belongs–under the control of the government"; "Where do these books go?"	SYNSET	go
belong#4	be classified with; "The whales belong among the mammals"	GLOSS	be

Table 5: Summarization Examples for Verbs

5.1.3 Adjective summarization

In WordNet 2.1 there are a total of 22 141 adjectives, out of which 23.72% have more than one sense, referring to 14 413 different senses, giving us an average of 2.7 senses per an ambiguous adjective. In order to create a summary label for a given ambiguous adjective we defined four heuristics, choosing (in the presented order) the first available heuristic among the following:

1. return the first shortest unused lemma among the available words of the same synset;
2. return the first shortest unused lemma among available the words in the satellite synsets (using the `similar_to` relation);
3. return the first shortest unused lemma among the available words in the pertainyms' (related noun) synsets;
4. return the first shortest unused lemma among the available words in the `see_also` synsets;
5. return the first shortest unused lemma among the available words in the antonyms' synsets;
6. return the first word of the gloss.

There are few cases (0.36%) where several senses share the same summary label, some of them (0.11%) have the same summary label for all senses. In a majority of cases (56%) the summary label is on the average 2.05 characters shorter than the original word. In 44% of the cases the summary label is on the average 2.89 characters longer than original word. Table 6 shows some examples of summary labels for adjectives.

 ©INSEMTIVES consortium 2009-2012

Sense	Gloss	Obtained via	Summary
aboriginal#1	of or pertaining to members of the indigenous people of Australia; ”an Aboriginal rite”	PERTAINYM	Abo
aboriginal#2	characteristic of or relating to people inhabiting a region from the beginning; ”native Americans”; ”the aboriginal peoples of Australia”	SYNSET	Native
aboriginal#3	having existed from the beginning; in an earliest or original stage or state; ”aboriginal forests”; ”primal eras before the appearance of life on earth”; ”the forest primeval”; ”primordial matter”; ”primordial forms of life”	SYNSET	primal
young#1	(used of living things especially persons) in an early period of life or development or growth; ”young people”	SYNSET	immature
young#2	(of crops) harvested at an early stage of development; before complete maturity; ”new potatoes”; ”young corn”	SYNSET	new
young#3	suggestive of youth; vigorous and fresh; ”he is young for his age”	SYNSET	youthful
young#4	being in its early stage; ”a young industry”; ”the day is still young”	GLOSS	being
young#5	not tried or tested by experience; ”unseasoned artillery volunteers”; ”still untested in battle”; ”an illustrator untried in mural painting”; ”a young hand at plowing”	SYNSET	unseasoned
latin#1	of or relating to the ancient Latins or the Latin language; ”Latin verb conjugations”	PERTAINYM	Latin
latin#2	having or resembling the psychology or temper characteristic of people of Latin America; ”very Latin in temperament”; ”a Latin disdain”; ”his hot Latin blood”	SIMILAR	Emotional
latin#3	relating to people or countries speaking Romance languages; ”Latin America”	PERTAINYM	Romance
latin#4	relating to languages derived from Latin; ”Romance languages”	SYNSET	Romance
latin#5	of or relating to the ancient region of Latium; ”Latin towns”	PERTAINYM	Lazio

Table 6: Summarization Examples for Adjectives

5.1.4 Adverb summarization

In WordNet 2.1 there is a total of 4 601 adverbs, out of which 16.32% have more than one sense, referring to 1 870 different senses, giving us an average of 2.49 senses per an ambiguous adverb. In order to create a summary label for a given ambiguous adverb we defined four heuristics, choosing (in the presented order) the first available heuristic among the following:

1. return the first shortest unused lemma among the available words of the same synset;
2. return the first shortest unused lemma among the available words in derived synsets of this adverb (using the `derived_from` relation);
3. return the first shortest unused lemma among the available words in the antonyms' synsets;
4. return the first word of the gloss.

There are a more cases than for the other POS (6.66%) where several senses share the same summary and some of them (2.4%) have the same summary label for all the senses. This is due to the lower number of relations for the adverbs in WordNet. In a majority of cases (66%) the produced summary label is on the average 2.56 characters shorter than the original word. In 34% of the cases the summary label is on the average 3.24 characters longer than the original. Table 7 shows some examples of summary labels for adverbs.

Sense	Gloss	Obtained via	Summary
aboard #1	part of a group; ”Bill's been aboard for three years now”	GLOSS	part
aboard #2	on a ship, train, plane or other vehicle	SYNSET	onboard
aboard #3	on first or second or third base; ”Their second homer with Bob Allison aboard”	SYNSET	on_base
aboard #4	side by side; ”anchored close aboard another ship”	SYNSET	alongside
aloft#1	at or on or to the masthead or upper rigging of a ship; ”climbed aloft to unfurl the sail”	GLOSS	at
aloft#2	upward; ”the good news sent her spirits aloft”	GLOSS	upward
aloft#3	at or to great height; high up in or into the air; ”eagles were soaring aloft”; ”dust is whirled aloft”	GLOSS	at
aloft#4	in the higher atmosphere above the earth; ”weather conditions aloft are fine”	GLOSS	in

Table 7: Summarization Examples for Adverbs

	total	with multiple senses (%)	number of senses	ambiguity	overlapping summaries (%)	no distinct summaries (%)
Nouns	117 097	13.47	43 783	2.7	1.48	0.82
Verbs	11 488	45.49	18 629	3.56	1.38	0.23
Adjectives	22 141	23.72	14 413	2.7	0.36	0.23
Adverbs	4 601	16.32	1 870	2.49	6.66	2.4

Table 8: WordNet 2.1 Word Ambiguity per POS

5.2 Evaluation

5.2.1 Scenario

As ultimately these summaries should be used to disambiguate terms visually when the user is providing a controlled annotation, we evaluated their quality and precision by evaluating them directly with the users. For each word, we evaluated all the defined heuristics applicable to the POS this word belongs to. This gave us a comparable measure of the quality of each heuristic for each POS.

The evaluation methodology is based on the fact that the purpose of the summary labels is twofold:

1. the summary should represent the meaning of the gloss,
2. it should discriminate well enough between the difference senses of the same word.

We have thus introduced two scenarios to evaluate the summary heuristics.

In the first scenario, the user is presented with a word and its summary in the form of the following question:

Among all these senses of the word **word** select the one(s) which mean(s) **summary**.

The user is then presented with a list of senses that are expressed by their glosses obtained from WordNet (see for instance Figures 27 and 28). The user is asked to select the right sense(s) corresponding to the summary shown. In case of doubts on the answer, the user also has the option of skipping the current question.

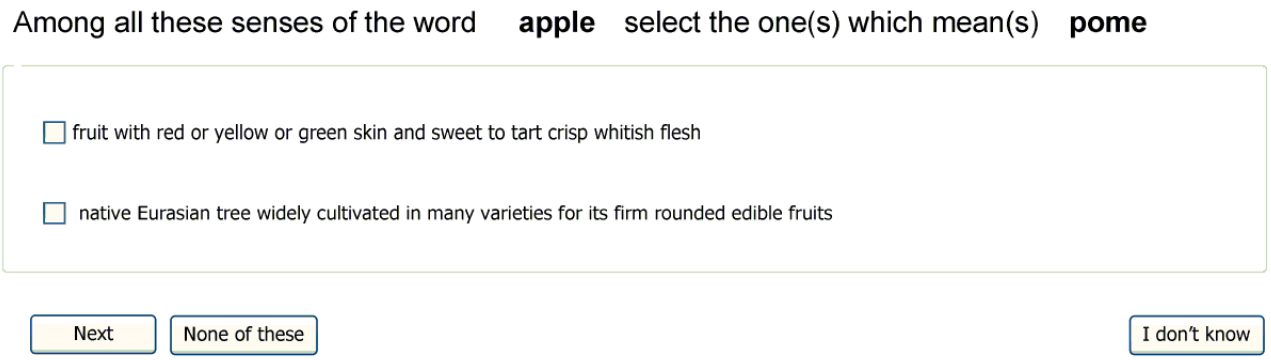


Figure 27: Example of the first validation scenario question for the word “Apple”

The second example of the first validation scenario provided in Figure 28 illustrates the rationale behind allowing the user to choose multiple senses as an answer to a question. In WordNet, there are senses that may be too fine-grained and the user may perceive the produced summary label as related to several displayed definitions. These too fine grained senses in WordNet might decrease the disambiguation precision of the summary label during the annotation task.

In order to assess the quality of the results of the first validation scenario we have devised six answer categories. These answer categories are not mutually exclusive and serve to demonstrate different properties of the summary heuristic in question. The categories and their purpose are:

unknown when the user clicked “I don’t know” button. This category shows that the user is not familiar with the presented word and/or summary label;

 ©INSEMTIVES consortium 2009-2012

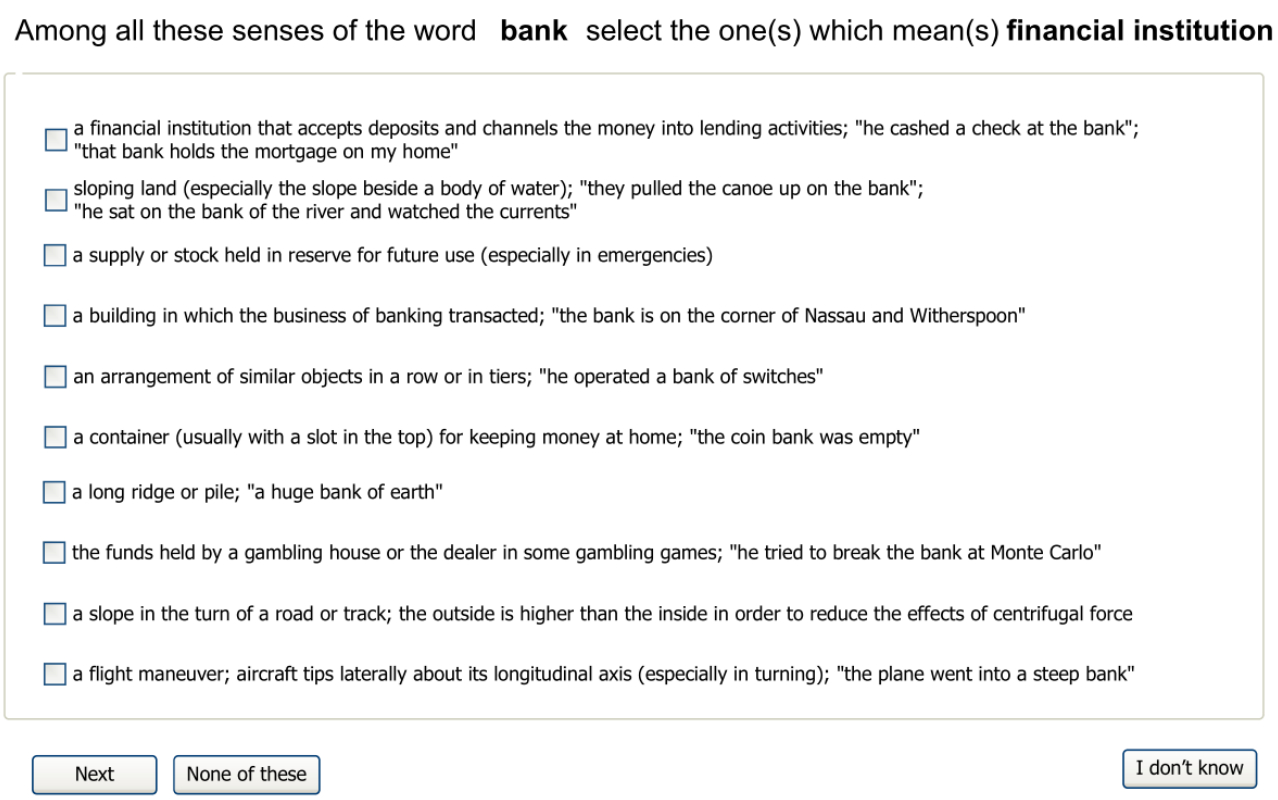


Figure 28: Example of the first validation scenario question for the word "Bank"

none when the user clicked "None of these" button. This category shows that user was not able to associate the summary with a sense;

correct when the user selected one sense and this sense is correct. This category shows that the summary label is good, as the user was able to associate it the correct sense;

semicorrect when the user selected more than one sense, and a correct one is among them. This category shows that the summary is potentially good, because the user was able to associate it the correct sense, however, probably due to very fine-grained senses in WordNet, the user was not able to make a proper distinction;

incorrect when the user selected one incorrect sense. This category shows that the summary label is potentially bad, because the user is not able to associate the correct sense; and

more than 1 selected sense when the user selected more than one sense. This category shows the cases where the senses are too fine-grained and confuses the user.

In the second validation scenario (illustrated in Figure 29) we test the *discrimination power* of the summary labels to disambiguate between the different senses without displaying a definition. To ease the cognitive load on the users, and thus simplify the validation task, we do this for pairs of senses of a word, instead of all senses at once by asking the following question:

If we talk about **word**, does the word **summary#sense1** mean the same as **summary#sense2**?

The user is then given the choice between three answers:

Yes means that the user understood the question but the discrimination is bad (i.e., the two summary mean the same thing to the user);

No means that the user understood the question and the discrimination is good (i.e., the two summaries represent different senses); and

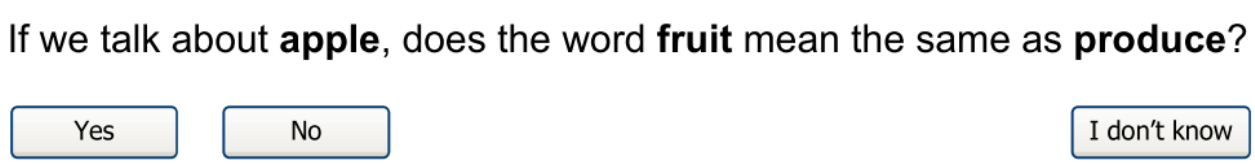


Figure 29: Example of the second validation scenario's question for the word "Apple" with summaries "fruit" and "produce"

I don't know means that the user did not understand something in the question and could not answer the question.

Given that the answers to these questions can be subjective or error prone (depending on the summaries generated, some questions were quite ambiguous), we decided to evaluate the agreement between different users for the same exact question. We thus showed each instance of the questions to more than one user so that the agreement between the users could be computed (see Section 5.2.4) and therefore, we could guarantee a certain accuracy for the results.

5.2.2 Dataset

POS	Summary count	Frequency of use
ADJECTIVE	1104	5.71
NOUN	4318	26.86
ADVERB	271	16.07
VERB	3621	11.00
TOTAL	9314	

Table 9: Summaries by Part of Speech (POS)

The summarization algorithm was tested on a subset of the WordNet containing summaries for 9 314 words (see Table 9 for details). Before conducting the evaluation we performed several dryrun evaluations with a limited set of users to test the phrasing of the questions and the design of the experiment.

In many cases, some users found the questions difficult to understand or to answer. In fact, the major reason for this difficulty was that, in some parts of WordNet, there is a very fine grained definition and specification of senses. In these cases, some of the participants that were shown very domain specific terms might never had seen them before and could not answer correctly the questions.

To tackle this issue, we have exploited the *use count* that is provided on a subset of WordNet and describes the "frequency of use" of the terms in a representative corpus of English. We have thus generated summaries only for words having non-zero use count in WordNet. This limited the questions to the most used words and thus, potentially, better known words, addressing the limited vocabulary problem to some extent.

Heuristic	Summary count	Frequency of use
CHILD	1 253	21.17
DERIVED	125	7.97
GLOSS	1 300	13.08
PARENT	3 804	24.57
SIMILAR_TO	756	6.58
SYNSET	2 076	11.24
TOTAL	9314	

Table 10: Summaries by Heuristics

 ©INSEMTIVES consortium 2009-2012

It is worth noting that different heuristics apply in different proportions to each POS (see Table 10 for details):

1. The "Synset" heuristic applies to all parts-of-speeches;
2. the "Parent" and the "Child" heuristics apply to both nouns and verbs;
3. the "Gloss" heuristic applies only to verbs due to the way in which the glosses are written;
4. the "Derived" heuristic applies only to adverbs; and
5. the "Similar_to" applies only to adjectives.

5.2.3 Participants

51 users participated in the evaluation, including representatives of both genders, various age groups (between 20 and 60) and various cultures. While the majority of the users were non-native English speakers, some native speakers as well as bilingual users participated in the evaluation. Fifteen users answered more than 100 questions each, and a top contributor answered 700 questions. 25 users answered at least 40 questions each while eleven users answered less than 20 questions each. On the average we collected 83 answers per user.

5.2.4 Users' Agreement

In order to ensure the quality of the validations, a subset of the questions were handed out to at least two different users. This is a standard procedure in the construction of language datasets [13], as it allows the evaluations of the validity and reproducibility of the results in annotation tasks that might be ambiguous due to their natural language nature. It is accepted, in the state of the art, that presenting all the questions to every user would require too much resources and thus, a representative subset of the questions needs to be validated by more than one user [13].

We have collected 308 double-rated questions for the first scenario and 301 questions for the second scenario. To compute the user-agreement, we used the "agreement without chance correction" [13] where we considered an answer to a question as an item. One question could have more than one answer, each given by a different user. It is also useful to keep in mind that the questions from the first scenario can have multiple independent answers and as such are more difficult to agree upon for the users. Tables 11, 12, and 13 provide details on the overall proportion of agreement by the type of answers across all heuristics.

Answer Type	First Scenario Questions	Second Scenario Questions
UNKNOWN	0.18	0.225
NONE	0.23	n/a
CORRECT	0.59	0.72
SEMICORRECT	0.04	n/a
INCORRECT	0.35	0.43

Table 11: Proportion of user agreement by question and answer type

Answer type	child	derived	gloss	parent	similar_to	synset
UNKNOWN	0.5	0.5	0	0.13	0	0
NONE	0.5	0	0	0.09	0.4	0
CORRECT	0.37	0	0.53	0.62	0.61	0.66
SEMICORRECT	0	0.09	0.05	0	0	0.07
INCORRECT	0.5	0.5	0	0.3	0.36	0.4

Table 12: Proportion of user agreement by heuristic and answer type for questions in the first scenario.

Answer type	child	derived	gloss	parent	similar_to	synset
UNKNOWN	0.18	1	0.25	0.25	0	0
CORRECT	0.82	0	0.72	0.75	0.60	0.53
INCORRECT	0	0	0.47	0.38	0.66	0.50

Table 13: Proportion of user agreement by heuristic and answer type for questions in the second scenario.

5.2.5 Precision Results

The first scenario allows us to measure the precision of the summarization heuristics. That is: given its summary, with what precision can a user select the right definition for a word within all its senses?

We report the amount of users that used each type of answers (see Section 5.2.1) and by POS. As mentioned earlier, each heuristic applies to a different proportion (if it applies) to each POS category and thus it is more interesting to consider the results separately.

Tables 14, 15, 16 and 17 provide a detailed of the distribution of answers per POS, heuristic and type of answers given by the user. Note that the type of answers are not exclusive, in fact *more than one selected sense* includes some of the *incorrect* answers[33] and the *semicorrect* ones[34].

	Heuristic					
Type	**child**	**derived**	**gloss**	**parent**	**similar_to**	**synset**
none	27.9	0	0	10.8	0	6.3
correct	**37.2**	0	0	**56.9**	0	**58.3**
semicorrect	2.3	0	0	4.3	0	4.2
incorrect	25.6	0	0	15.0	0	17.7
more than 1 selected sense	6.9	0	0	12.9	0	13.5
unknown	17	0	0	11	0	9

Table 14: Distribution (%) of the answers for the first scenario (Precision) questions for the Nouns.

	Heuristic					
Type	**child**	**derived**	**gloss**	**parent**	**similar_to**	**synset**
none	0	0	0	0	4.1	5.2
correct	0	0	0	0	**64.3**	**57.73**
semicorrect	0	0	0	0	2.0	7.1
incorrect	0	0	0	0	22.4	19.4
more than 1 selected sense	0	0	0	0	7.1	10.2
unknown	0	0	0	0	4	12

Table 15: Distribution (%) of the Answers for the first scenario (Precision) questions for the Adjectives.

[33] when the user selected more than one sense and none were the correct one.

[34] when the user selected more than one sense and one of them was the correct one.

 ©INSEMTIVES consortium 2009-2012

	Heuristic					
Type	**child**	**derived**	**gloss**	**parent**	**similar_to**	**synset**
none	24.1	0	8.2	16.1	0	1.9
correct	**39.1**	0	**63.3**	**44.1**	0	**53.4**
semicorrect	4.6	0	5.1	5.4	0	9.7
incorrect	18.7	0	13.3	22.6	0	13.6
more than 1 selected sense	11.5	0	10.2	11.8	0	21.4
unknown	17	0	6	12	0	7

Table 16: Distribution (%) of the Answers for the first scenario (Precision) questions for the Verbs.

	Heuristic					
Type	**child**	**derived**	**gloss**	**parent**	**similar_to**	**synset**
none	0	6.4	0	0	0	2.9
correct	0	**39.4**	0	0	0	**44.8**
semicorrect	0	11.0	0	0	0	15.2
incorrect	0	24.8	0	0	0	17.1
more than 1 selected sense	0	18.34	0	0	0	19.9
unknown	0	4	0	0	0	11

Table 17: Distribution (%) of the Answers for the first scenario (Precision) questions for the Adverbs.

For most of the cases, all the heuristics perform well, with the users answer either correctly or semi-correctly, in particular, the *synset* heuristic that can be applied to all the POS can help the users choose between multiple senses precisely in 62.7% of the cases.

5.2.6 Discrimination Power Results

In an annotation application, it is often more important that the summary is good at discriminating between the multiple senses of the term that it applies to. Actually, the user has to choose the right concept among several choices when annotating. It is thus more important that the summary is good at discriminating between the different terms displayed than how precise it is at defining the term.

In the case of the Nouns, we can see that the *similar_to* relationship in WordNet provides summaries that are able to discriminate sense quite effectively for the users (see Table 18). However, using other terms in the same *synset* does not provide a very strong discrimination between senses; this might be due to the fact that the other terms in the synsets are themselves quite ambiguous.

For the Adjectives, it is very difficult to generate a good discriminating summary and another strategy might be needed to help the user choose the right sense. For instance, it might be more informative, even if it takes more space, to show to the user an example of use of the adjective instead of a single word summary.

	Heuristic					
Type	**child**	**derived**	**gloss**	**parent**	**similar_to**	**synset**
incorrect	31.1	0	0	23.6	0	55.6
correct	**68.9**	0	0	**76.4**	0	**44.1**
unknown	10	0	0	11	0	7

Table 18: Distribution (%) of the Answers for the second scenario (Discriminating Power) questions for the Nouns.

	Heuristic					
Type	**child**	**derived**	**gloss**	**parent**	**similar_to**	**synset**
incorrect	0	0	0	0	59.4	64.4
correct	0	0	0	0	**40.6**	**25.6**
unknown	0	0	0	0	4	11

Table 19: Distribution (%) of the Answers for the second scenario (Discriminating Power) questions for the Adjective.

For Verbs (see Table 20), which are usually very ambiguous (see Table 8), most of the heuristics are actually providing a good discriminating power. It is interesting to see that the *gloss* heuristic has the better discriminating power. This is probably due to the fact that, in WordNet, the glosses for verbs often start with another verb that can give more information to the action described by the verb. For instance, for the term "run", we can find: "cause", "move", "perform", etc.

	Heuristic					
Type	**child**	**derived**	**gloss**	**parent**	**similar_to**	**synset**
incorrect	35.5	0	33.3	36.9	0	53.2
correct	**64.5**	0	**66.7**	**63.0**	0	**46.7**
unknown	7	0	10	8	0	8

Table 20: Distribution (%) of the Answers for the second scenario (Discriminating Power) questions for the Verbs.

	Heuristic					
Type	**child**	**derived**	**gloss**	**parent**	**similar_to**	**synset**
incorrect	0	59.7	0	0	0	66.3
correct	0	**40.3**	0	0	0	**33.7**
unknown	0	29	0	0	0	14

Table 21: Distribution (%) of the Answers for the second scenario (Discriminating Power) questions for the Adverbs.

Table 21 shows that it is also very hard to generate good disambiguating summaries for Adverbs. However, if we consider the dataset of del.icio.us that we discuss in Section 3.2 as a good example of users' annotation behaviour, then we can see that Adverbs are never used (see Figure 18. Thus, using the full definition for the Adverbs for discriminating their senses will not often impair the interaction with the user.

5.3 Conclusion

As we have shown in the previous discussions in this deliverable, doing fully automatic disambiguation of annotations to a controlled sense in the underlying vocabulary is difficult. We thus need to provide ways for the users to provide as much possible disambiguation when they are creating the annotations. In the INSEMTIVES project, this is done by studying incentives that would motive the users, but we also believe that providing more streamlined annotation interfaces is a prerequisite.

By providing a one word summary for all the senses of ambiguous words, it will be easier to display the choices of senses to the users at the annotation time and thus improve the disambiguation process. In future work, it would be interesting to study the effect of "examples of use" as provided in lexical resources such as WordNet as an additional, short visual disambiguation help. Also, while we have studied the disambiguation power of the generated summaries, it would be interesting to study their direct effect on a real annotation task.

 ©INSEMTIVES consortium 2009-2012

6 Semantic Search

6.1 Summary

In the deliverable D3.2.1 [36] we have described the initial release of the Semantic Content Management Platform, the source code of which can be found at: `http://insemtives.svn.sourceforge.net/viewvc/insemtives/platform/trunk/`. We evaluated the provided capabilities for semantic information retrieval (semantic search) against the Telefónica's use case requirements for semantic search as listed in deliverable D5.1.2 [45]. At the date of delivery of the current deliverable, the final requirements for the Seekda! and PGP use cases were not yet available and thus we cannot mention them here. Alignment of the search methods will be performed iterativelly with these use case partners.

In this section we explain the validation process and the improvements that will be done to the semantic information retrieval capabilities.

6.2 Use case validation

The search capabilities of the Semantic Content Management Platform were evaluated on a User Advisory Board meeting, held at the ESWC in Iraklio, Crete, on June 4th and 5th, 2010. The semantic search and navigation tool was presented with a video walkthrough. It was analyzed given the requirements of the use case as reported in D5.1.2 [45]. In what follows, we present the relation between the relevant UI requirements and how they are supported by the search capabilities of the Semantic Content Management Platform as defined in the deliverable "Specification of Information Retrieval Methods for Semantic Content" D2.3.2 [34].

- Requirement WP5.FUN18. SEARCH. *The final user MAY be able to search the content exposed by the application based on the semantic annotations already added to that content. In particular, some kind of faceted search would be useful, letting the final users to search the content based on either one facet or a combination of them.* This is supported by the retrieval method described in D.2.3.2 [34], subsection 2.4 Co-occurrence and ranking of entities.

- Requirement WP5.FUN19. SEARCH SUGGESTIONS. *The semantic search capabilities MAY include some kind of guidance to compose the final semantic search query depending on the data already included in the semantic annotation platform.* This was an optional general requirement, which needed to be further defined in order to be supported by the retrieval methods. After the initial development of the OKenterprise use-case tools, it has been specified from "suggestion of possible query results" to "suggestion of existing concepts or synsets that match a $ConceptualQuery$". The revised requirement is supported by the method described in the Refinements subsection below.

- Requirement WP5.FUN23. ONTOLOGIES SEARCH. *The final users MAY be able to search the semantic annotation ontologies used, looking for concepts and/or properties.* This optional requirement is not covered by the the retrieval methods specification D2.3.2 [34]. Basic retrieval of concepts and properties is supported by the Background Knowledge Service, alternatively known as Entity Service, as described in D3.1 [42], section 3.

- Requirement WP5.PER01. RESPONSE TIMES. *The semantic annotation platform MUST respond in less than 1 second for annotation adding operations and in less than 5 seconds for searching operations.* The retrieval methods specification D2.3.2 [34] does not contain any analysis on the implementation of the methods against response time. Since the initial version of the implementation D3.3.1 [35] also does not comment on that issue, we propose to modify the requirements of the the retrieval methods implementation in D2.3.2 [34] to allow time limited execution of searches, retrieving potentially limited results.

Below we explain how the improved auto-suggest function of the semantic retrieval of entities will affect some of the use case tools and simplify the end user participation.

6.3 Refinements

The refinements in the semantic search affect especially the retrieval of named entities. We need to add a new concept or synset-centric search method to the ones existing in D2.3.2 [34] "Specification of Information Retrieval Methods for Semantic Content", so that the use case tools can meet the requirements for quick and focused entity search.

To enable the users to quickly select entities, we will add the method *getEntities*. Note that by entity we mean the union between concept and synset as defined in the Annotation Model [D2.1.2] [21].

$getEntities$
INPUT 'PREFIX': $string$ - the prefix of a token in the entity name such as: "Bara" when searching for "Barack Obama"
INPUT 'LABELPREDICATE': URI - the predicate that connects the entity to its label (optional parameter; default: rdfs:label). See below for details.
INPUT 'LANGUAGE': $string$ - a language identifier such as "en-us" or "fr" (optional parameter; default: empty). An empty string is considered as any language.
INPUT 'CQUERY': $ConceptualQuery$ - a conceptual query that further specifies the returned entities (optional parameter).
INPUT 'RANKINGTYPE': $oneof\{rdfrank, none\}$ - the ranking approach to be used (optional parameter; default: none). See below for details.
INPUT 'LIMIT': $integer$ - the maximum number of entities to be returned (optional parameter; default: -1). The value of -1 is considered as no limit.
OUTPUT: $EntitySet$
EXCEPTION: $StorageException$
DESCRIPTION: Retrieves all entities where one of their name tokens starts with the given prefix. The entity names are all determined as literals connected to the entity through the properties defined by the labelPredicate URI. If no entities match the prefix requirement, the result is an empty $EntitySet$. If a ranking type is selected, the entities are ordered by their RDFRank or other criteria in descending order.

Here follows the more detailed description of the method parameters.

The **'prefix' parameter** specifies a prefix of a token in a concept or synset name. The prefix is not case sensitive. The API behavior reflects the way most users search for concepts. They start typing one of the words in the name from the beginning, for example "Aga" for "Andre Agasi".

The **'labelPredicate' parameter** controls the semantics of the relation between the concept or synset and its name. Depending on the specific case, the user interface may require searching only in the preferred names of concepts, in all names of concepts including the alternative and rarely used ones, in the labels of the different terms that compose a synset, etc. Instead of restricting the name semantics to a predefined option, the label predicate approach allows the use-case tools developers to define custom types of names with the semantics defined in OWL 2 RL or RDFS. Still, the default behavior adheres *strictly* to the concept name definition in the Annotation Model [D2.1.2], which is *the lemma of any term part of a synset connected to the concept*. Additionally, these standard types of name semantics are supported by using well-known predicates. For example:

- rdfs:label - The default value of the parameter searches in all names of the concepts, according to the Annotation Model [D2.1.2].
- skos:prefLabel - Searches only in preferredLabels of concepts.
- wn20:senseLabel - Searches in all terms related to synsets. Note that, as stated above, in the standard RDF representation of the Annotation Model, WordNet 2.0 Basic describes the synsets and terms.

The **'language' parameter** allows selecting a specific language for the labels. In effect, this parameter specifies the Natural Language Dictionary (NLD) as defined in the Annotation Model D2.1.2 [21]. The synset and conceps labels that belong to a given NLD are marked with the RFC-4647-compliant language identifier of the NLD. The 'language' identifier restriction is applied according to the matching algorithm, also defined in the RFC. For example, if you look for "en" labels, and there is no NLD which language is exactly "en", the method will return "en-uk" labels.

 ©INSEMTIVES consortium 2009-2012

The **'cquery' parameter** adds existing semantic restrictions to the prefix query. For details about the ConceptualQuery restriction behavior, see the existing "entity-centric" search method in D2.3.2 [34].

The **'rankingType' parameter** controls what kind of ranking is used when ordering the entity set. Only the most generic option 'rdfrank' is specified here, while the use case developers can extend the semantic search by implementing domain-specific ranking algorithms. RDF Rank identifies the more important or more popular entities in the repository by examining their interconnectedness. The popularity of entities can then be used to order query results in a similar way to internet search engines, such as how Google orders search results using PageRank.

In conclusion, this new search method will allow the use case developers and the tools designers to include smart user-friendly suggestions for tag names, attributes, or relations that will simplify the annotation process for the end user.

6.4 Existing work

Two public services exist that provide similar capabilities to the described method.

The first one is DBPedia Autocomplete - `http://lookup.dbpedia.org/autocomplete.aspx`. It provides prefix search over the latest DBPedia compilation.

The other one is FactForge Autocomplete - `http://factforge.net/autocomplete.json?callback=p\&q=Bara\&limit=10`. It is a JSONP service that provides prefix search with support for ranking of the concepts in `http://factforge.net`.

Both services are quite limited to be used in the platform. In their current form they run over a predetermined data set, which makes them unsuitable for the use-cases. They are open source, so it is possible to integrate them in the Semantic Content Management Platform. However, the lack of support for specific languages, different name semantics, and additional conceptual restrictions mean that these services would not be able to satisfy the requirements of the tools and use cases.

7 Conclusions

The work reported in this deliverable builds on top of the results reported in the previous deliverables from Workpackage 2. These deliverables are: D2.1.1 [23] (Report on the state-of-the-art and requirements for annotation representation models), D2.1.2 [21] (Specification of models for representing single-user and community-based annotations of Web resources), D2.2.1 [10] (Report on methods and algorithms for bootstrapping Semantic Web content from user repositories and reaching consensus on the use of semantics), D2.2.2/D2.2.3 [22] (Report on methods and algorithms for linking user-generated semantic annotations to Semantic Web and supporting their evolution in time), D2.3.1 [24] (Requirements for information retrieval (IR) methods for semantic content), and D2.3.2 [34] (Specification of information retrieval (IR) methods for semantic content).

In this deliverable we reported on the refinements of the models and methods proposed in the aforementioned deliverables. The deliverable reports on the results of the conducted validations of the proposed models and methods that led to the necessity of their refinement. In addition, we provide a formalisation of the necessary extensions for the annotation model, a reproducible description of the refinements to the algorithms with a discussion of how these refinements relate to the state-of-the-art in relevant areas. Apart from proposing refinements to the models and methods to respect the evolved requirements from the use case partners, the deliverable makes the following contributions that can have an impact on a larger scale:

A platform for creating golden standards. It has been realised that the lack of golden standard social tagging datasets significantly hinders the research progress and leads to a number of problems including issues of efficiency, reproducibility, comparability and ultimately validity of research [37, 29, 5]. As part of the work on this deliverable, a platform for creating golden standards for semantic annotation systems was developed and made available to the community (see Section 3 for details). The platform can help solve the problem mentioned above by providing the dataset developers with a tool for creating golden standard datasets that support the developers at different phases of the dataset generation process;

A golden standard dataset. In order to evaluate the performance of the proposed algorithms, a golden standard dataset was created using the above mentioned platform. The dataset represents a real use case as it is built from a subset of a De.licio.us dump by extending it with manually validated links from tags to concepts in the WordNet ontology. To the best of our knowledge, it is the first manually created dataset with the above mentioned characteristics with a guaranteed quality measured with multi-annotators agreement. The dataset is made available to the community and can be used as a benchmark, against which multiple approaches can be compared. The dataset is available in the RDF format and is currently undergoing the process of its inclusion to the Linking Open Data cloud to facilitate the dissemination and reuse of the dataset. The fact that the dataset was developed using the above mentioned platform confirms the applicability of the platform for research purposes;

Quality of Service in semantic annotation systems. The experimental results reported in Section 3.3 confirm the hypothesis that adding explicit semantics to annotations can improve the quality of services such as search in social annotation systems such as De.licio.us. This hypothesis is validated thanks to the aforementioned golden standard dataset. In the same section we provide a quantitative analysis of the improvements brought by semantic annotations to search – one of the most typical operations in social annotation systems. The reported results further motivate the need of involving the user in the process of generating semantic contents as it shows the possible added value from doing this;

Semantic convergence with multiple senses identification. As shown in Section 4, the proposed semantic convergence algorithm is capable of computing more than one candidate meaning for a tag that is used for the annotation of different resources. It is fundamentally different from the existing approaches that make the simplifying assumption that one tag corresponds to one sense only. As we discuss in the same section, even if the most frequently-used sense is used, it will correspond to 60%-70% of the cases, whereas the other possible meanings are lost. This can have a negative effect on the quality of service (e.g., see the previous item) and demotivate the user from providing semantic annotations. Thus, this study shows a possible oversight in the existing approaches that need to be addressed in future studies. While there is an existing number of approaches for this semantic convergence issue, they do not provide reproducible and comparable evaluations, Section 4.2 presents a novel, quantitative, method of evaluating such semantic convergence algorithms that will allow a better comparison of the different approach and, we believe, an improvement in the quality of the research in that field; and,

Concept Summarization. As discussed in Section 5, the proposed concept summarization algorithm is capable of computing a one word summary given a concept that can be used to guide the user in the selection of an ontology element for the annotation of resources. This should allow for the user interfaces and interaction modalities that do not require the user to know that there is an ontology at all and, thus, generate semantic annotations in an easy and natural way. To the best of our knowledge, this is a novel algorithm and, therefore, can be considered as a contribution to the state-of-the-art. Developing such user-oriented approach to sense disambiguation is very important as it has been shown multiple time that, yet, it is very difficult to use fully automatic disambiguation algorithms.

The work reported in this deliverable and in the preceding ones provides the basis for further research directions, amongst which: (a) the generation of a larger golden standard dataset for a more realistic evaluation of the proposed algorithms and for a more in-depth study of the properties of semantic annotation systems; (b) further improvements of the algorithms and their evaluation on the real data of the use case partners; and (c) further improvement of the golden standard generation platform based on the feedback from its usage within the community, if any. The results of research on these and possible other directions will be reported in publications in international conferences, workshops, and other venues.

 ©INSEMTIVES consortium 2009-2012

References

[1] Rdf/owl representation of wordnet – editor's draft. Technical report, W3C, 2006. http://www.w3.org/2001/sw/BestPractices/WNET/wn-conversion.

[2] Rdf/owl representation of wordnet – w3c working draft. Technical report, W3C, 2006. http://www.w3.org/TR/wordnet-rdf/.

[3] Self-adaptation of ontologies to folksonomies in semantic web. *Proc World Acad Sci Eng Tech*, 33(September):335–341, 2008.

[4] Vocabulary summary: Common tag. Technical report, Zemanta, 2010. http://www.commontag.org/Specification.

[5] Garca-Silva A., Corcho O., Alani H., and Gmez-Perez A. Review of the state of the art: Discovering and associating semantics to tags in folksonomies. *The Knowledge Engineering Review*, 2010, (To be published).

[6] Karl Aberer, Philippe C. Mauroux, Aris M. Ouksel, Tiziana Catarci, Mohand S. Hacid, Arantza Illarramendi, Vipul Kashyap, Massimo Mecella, Eduardo Mena, Erich J. Neuhold, and Et. Emergent Semantics Principles and Issues. In *Database Systems for Advances Applications (DASFAA 2004), Proceedings*, pages 25–38. Springer, March 2004.

[7] E. Agirre and G. Rigau. A proposal for word sense disambiguation using conceptual distance. In *the First International Conference on Recent Advances in NLP*, Tzigov Chark, Bulgaria, September 1995.

[8] Enrique Alfonseca and Suresh Manandhar. Extending a lexical ontology by a combination of distributional semantics signatures. In *Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web*, EKAW '02, pages 1–7, London, UK, 2002. Springer-Verlag.

[9] Enrique Alfonseca and Suresh Manandhar. Proposal for evaluating ontology refinement methods. *Language Resources and Evaluation*, 2002.

[10] Pierre Andrews, Juan Pane, and Ilya Zaihrayeu. Insemtives deliverable 2.2.1:report on methods and algorithms for bootstrapping semantic web content from user repositories and reaching consensus on the use of semantics. Technical report, UNITN, June 2009.

[11] 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.

[12] Sofia Angeletou, Marta Sabou, and Enrico Motta. Semantically enriching folksonomies with flor. In *In Proc of the 5th ESWC. workshop: Collective Intelligence & the Semantic Web*, 2008.

[13] R. Artstein and M. Poesio. Inter-Coder Agreement for Computational Linguistics. *Journal of Computational Linguistics*, 34(4), 2008.

[14] Au, N. S. Gibbins, and N. Hadbolt. Understanding the Semantics of Ambiguous Tags in Folksonomies. In *The International Workshop on Emergent Semantics and Ontology Evolution (ESOE2007) at ISWC/ASWC 2007*, November 2007.

[15] Aliaksandr Autayeu, Fausto Giunchiglia, and Pierre Andrews. Lightweight parsing of classifications into lightweight ontologies. In *ECDL*, pages 327–339, 2010.

[16] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider. *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge University Press, 2003.

[17] Diego Berrueta, Dan Brickley, Stefan Decker, Sergio Fernndez, Christoph Grnand Andreas Harth, Tom Heathand Kingsley Idehen, Kjetil Kjernsmo, Alistair Miles, Alexandre Passant, Axel Polleres, and Luis Polo. Sioc core ontology specification. Technical report, DERI, 2010. http://rdfs.org/sioc/spec/.

[18] Christoph Blank, Michal Zaremba, Markus Rohde, Lin Wan, Fahri Yetim, and Roberta Cuel. Insemtives deliverable 6.1.1: Extended market place. requirements specification (initial version). Technical report, Seekda!, Siegen, UNITN, 2009.

[19] DCMI Usage Board. Dcmi metadata terms. Technical report, Dublin Core Metadata Initiative Limited, 2010. http://dublincore.org/documents/dcmi-terms/.

[20] Tobias Burger, Olga Morozova, Ilya Zaihrayeu, Pierre Andrews, and Juan Pane. Insemtives deliverable 2.1.1: Specification of models for representing single-user and community-based annotations of web resources. Technical report, UIBK, UNITN, 2009.

[21] Tobias Bürger, Olga Morozova, Ilya Zaihrayeu, Pierre Andrews, and Juan Pane. Insemtives deliverable 2.1.2: Specification of models for representing single-user and community-based annotations of web resources. Technical report, UIBK, UNITN, June 2009.

[22] Tobias Burger, Olga Morozova, Ilya Zaihrayeu, Pierre Andrews, and Juan Pane. Insemtives deliverable 2.2.2/2.2.3: Report on methods and algorithms for linking user-generated semantic annotations to semanticweb and supporting their evolution in time. Technical report, UNITN, November 2009.

[23] 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. Technical report, UIBK, UNITN, ONTOTEXT, June 2009.

[24] Tobias Bürger, Ilya Zaihrayeu, Uladzimir Kharkevich, and Borislav Popov. Insemtives deliverable 2.3.1: Requirements for information retrieval (ir) methods over semantic content. Technical report, UNITN, ONTO, September 2009.

[25] C. Cattuto, D. Benz, A. Hotho, and G. Stumme. Semantic grounding of tag relatedness in social bookmarking systems. In Amit P. Sheth, Steffen Staab, Mike Dean, Massimo Paolucci, Diana Maynard, Timothy W. Finin, and Krishnaprasad Thirunarayan, editors, *Proc.Intl. Semantic Web Conference 2008*, page 615631. Springer.

[26] David Crandall, Lars Backstrom, Dan Huttenlocher, and Jon Kleinberg. Mapping the World's Photos. In *Proc. of the Int. World Wide Web Conference*, 2009.

[27] B. Dutta and F. Giunchiglia. Semantics are actually used. In *International Conference on Semantic Web and Digital Libraries*, pages 62–78, Trento, Italy, September, 8-11 2009. University of Trento.

[28] Martin Ester, Hans peter Kriegel, Jrg S, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96)*, pages 226–231. AAAI Press, 1996.

[29] A. Garca-Silva, M. Szomszor, H. Alani, and O. Corcho. Preliminary results in tag disambiguation using dbpedia. In *Proc. of the First International Workshop on Collective Knowledge Capturing and Representation (KCAP)*, USA, 2009.

[30] Fausto Giunchiglia, Uladzimir Kharkevich, and Ilya Zaihrayeu. Concept search. In *ESWC*, pages 429–444, 2009.

[31] Carl Goodman and Tobias Burger. Insemtives deliverable 7.1.1: Virtual worlds. requirements specification (initial version). Technical report, PGP, UIBK, 2009.

[32] Udo Hahn and Klemens Schnattinger. Towards text knowledge engineering. In *IN AAAI/IAAI*, pages 524–531, 1998.

 ©INSEMTIVES consortium 2009-2012

[33] Salma Jamoussi. Une nouvelle reprsentation vectorielle pour la classification smantique. In *Apprentissage automatique por le TAL*, volume 50, pages 23–57. TAL, 2009.

[34] Uladzimir Kharkevich, Mihail Konstantinov, and Borislav Popov. Insemtives deliverable 2.3.2: Specification of information retrieval methods for semantic content. Technical report, UNITN, ONTO, November 2009.

[35] Mihail Konstantinov, Marin Nozhchev, Atanas Ilchev, Anton Andreev, and Gergana Petkova. Insemtives deliverable d.3.3.1: Implementation of information retrieval (ir) methods for semantic content (initial version). Technical report, ONTO, March 2010.

[36] Mihail Konstantinov, Marin Nozhchev, Atanas Ilchev, Reneta Popova, and Gergana Petkova. Insemtives deliverable d.3.2.1: Semantic content management platform (initial version). Technical report, ONTO, March 2010.

[37] Christian Körner and Markus Strohmaier. A call for social tagging datasets. *SIGWEB Newsl.*, pages 2:1–2:6, January 2010.

[38] Huairen Lin, Joseph Davis, and Ying Zhou. An integrated approach to extracting ontological structures from folksonomies. In Lora Aroyo, Paolo Traverso, Fabio Ciravegna, Philipp Cimiano, Tom Heath, Eero Hyvnen, Riichiro Mizoguchi, Eyal Oren, Marta Sabou, and Elena Simperl, editors, *The Semantic Web: Research and Applications*, volume 5554 of *Lecture Notes in Computer Science*, pages 654–668. Springer, Berlin / Heidelberg, 2009.

[39] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60:91–110, 2004.

[40] Peter Mika. Ontologies are us: A unified model of social networks and semantics. *Web Semantics: Science, Services and Agents on the World Wide Web*, 5(1):5–15, March 2007.

[41] G. Miller. *WordNet: An electronic Lexical Database*. MIT Press, 1998.

[42] Katharina Siorpaes, Mihail Konstantinov, and Borislav Popov. Insemtives deliverable d.3.1: Requirement analysis and architectural design of semantic content management platform. Technical report, UIBK, ONTO, September 2009.

[43] L. Specia and E. Motta. Integrating folksonomies with the semantic web. In *Proc. of the European Semantic Web Conference (ESWC2007)*, volume 4519 of *LNCS*, pages 624–639, Berlin Heidelberg, Germany, July 2007. Springer-Verlag.

[44] Chengzheng Sun, Xiaohua Jia, Yanchun Zhang, Yun Yang, and David Chen. Achieving convergence, causality preservation, and intention preservation in real-time cooperative editing systems. *ACM Trans. Comput.-Hum. Interact.*, 5(1):63–108, 1998.

[45] German Toro del Valle. Insemtives deliverable 5.1.2: Corporate knowledge management portal. requirements specification (final version). Technical report, Telefónica Investigacion y Desarrollo, 2010.

[46] Cline Van Damme, Martin Hepp, and Katharina Siorpaes. Folksontology: An integrated approach for turning folksonomies into ontologies. In *Bridging the Gep between Semantic Web and Web 2.0 (SemNet 2007)*, pages 57–70, 2007.

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

[48] Kilian Quirin Weinberger, Malcolm Slaney, and Roelof Van Zwol. Resolving tag ambiguity. In *Proceeding of the 16th ACM international conference on Multimedia*, MM '08, pages 111–120, New York, NY, USA, 2008. ACM.

[49] Robert Wetzker, Carsten Zimmermann, and Christian Bauckhage. Analyzing Social Bookmarking Systems: A del.icio.us Cookbook. In *Proceedings of the ECAI 2008 Mining Social Data Workshop*, pages 26–30. IOS Press, 2008.

[50] Zhibiao Wu and Martha Palmer. Verb semantics and lexical selection. In *Proc. of the 32nd annual meeting on Association for Computational Linguistics*, pages 133–138, 1994.

[51] Rui Xu and Donald Wunsch. Survey of clustering algorithms. *IEEE Transactions on Neural Networks / a Publication of the IEEE Neural Networks Council*, 16(3):645–678, May 2005. PMID: 15940994.

[52] C. Yang and J. C. Hung. Word sense determination using wordnet and sense co-occurrence. In *Proceedings of the 20th International Conference on Advanced Information Networking and Applications - Volume 1*, pages 779–784, Washington, DC, USA, 2006. IEEE Computer Society.

[53] Ilya Zaihrayeu, Lei Sun, Fausto Giunchiglia, Wei Pan, Qi Ju, Mingmin Chi, and Xuanjing Huang. From web directories to ontologies: Natural language processing challenges. In *In 6th International Semantic Web Conference (ISWC*. Springer, 2007.

[54] Ilya Zaihrayeu, Lei Sun, Fausto Giunchiglia, Wei Pan, Qi Ju, Mingmin Chi, and Xuanjing Huang. From web directories to ontologies: Natural language processing challenges. In *ISWC/ASWC*, pages 623–636, 2007.

[55] Lei Zhang, Xian Wu, and Yong Yu. Emergent Semantics from Folksonomies: A Quantitative Study. In *Journal on Data Semantics VI*, Lecture Notes in Computer Science, chapter 8, pages 168–186. 2006.

 ©INSEMTIVES consortium 2009-2012

A del.icio.us RDF Model

In the following appendix we provide examples of the RDF tuples used to export the del.icio.us dataset described in Section 3.2 to the LOD.

Concept N3 notation

```
@prefix : <#> .

@prefix skos: <http://www.w3.org/2004/02/skos/core#> .
@prefix wn30: <http://purl.org/vocabularies/princeton/wn30/> .

@prefix tags2con: <http://dataset.science.unitn.it/delicious/tags2con#> .

: a skos:Concept;
  skos:prefLabel "holiday"@en ;
  skos:altLabel "holidays"@en ;
  skos:closeMatch wn30:synset-holiday-noun-1 .
  tags2con:meaningOfToken
      <http://dataset.science.unitn.it/delicious/word/holidays/1#holiday>
```

Bookmark N3 notation

```
@prefix : <#> .

@prefix tags: <http://www.holygoat.co.uk/owl/redwood/0.1/tags> .
@prefix dc: <http://purl.org/dc/elements/1.1/> .
@prefix bookmark: <http://www.w3.org/2002/01/bookmark#> .

@prefix user: <http://dataset.science.unitn.it/delicious/user/> .


: a bookmark:Bookmark;
  bookmark:recalls <http://en.wikipedia.org/wiki/Java>;
  tags:tag [a tags:Tagging;
            tags:associatedTag
               <http://dataset.science.unitn.it/delicious/tag/javaisland/1>,
               <http://dataset.science.unitn.it/delicious/tag/holidays/1>;
            tags:taggedBy user:1234];
  dc:creator user:1234.
```

Tagging N3 notation

```
@prefix : <#> .

@prefix dc: <http://purl.org/dc/elements/1.1/> .
@prefix tags: <http://www.holygoat.co.uk/owl/redwood/0.1/tags> .
@prefix skos: <http://www.w3.org/2004/02/skos/core#> .
@prefix ctag: <http://commontag.org/ns#> .
@prefix sioc: <http://rdfs.org/sioc/ns#> .

@prefix deltag: <http://www.delicious.com/tag/> .
```

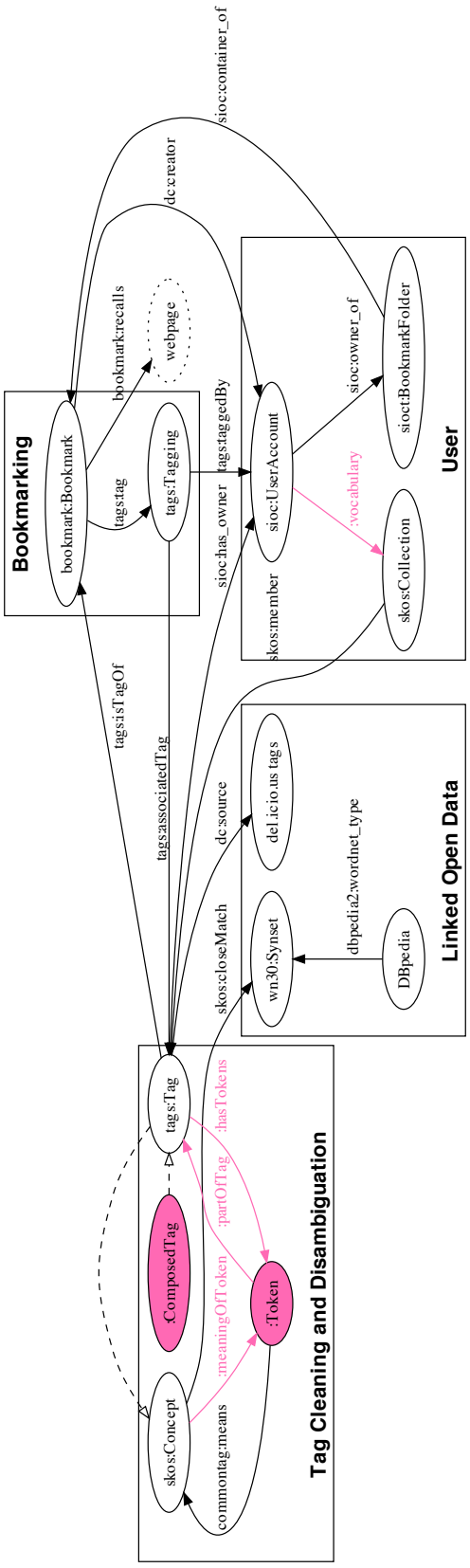


Figure 30: Dedicated RDF Model for the del.icio.us dataset in LOD

 ©INSEMTIVES consortium 2009-2012

```
@prefix wn30: <http://purl.org/vocabularies/princeton/wn30/> .

@prefix user: <http://dataset.science.unitn.it/delicious/user/> .
@prefix concept: <http://dataset.science.unitn.it/delicious/concept/> .
#our own vocabulary
@prefix tags2con: <http://dataset.science.unitn.it/delicious/tags2con#> .

: a :ComposedTag;
  tags:tagName "javaisland";
  dc:source deltag:javaisland;
  tags2con:hasTokens :java, :island;
  sioc:has_owner user:1234, user:2344;
  tags:isTagOf
     <http://dataset.science.unitn.it/delicious/user/1234/bookmark/5678>,
     <http://dataset.science.unitn.it/delicious/user/2344/bookmark/5245> .

:java a tags2con:Token;
      dc:label "java"@en;
      tags2con:position 0;
      tags2con:partOfTag :;
      ctag:means concept:Java-noun-1 .

:island a tags2con:Token;
        dc:label "island"@en;
        tags2con:position 1;
        tags2con:partOfTag :;
        ctag:means concept:island-noun-1 .
```

User Account N3 notation

```
@prefix : <#> .

@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix skos: <http://www.w3.org/2004/02/skos/core#> .
@prefix sioc: <http://rdfs.org/sioc/ns#> .
@prefix sioct: <http://rdfs.org/sioc/types#> .
@prefix tags: <http://www.holygoat.co.uk/owl/redwood/0.1/tags> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#>.
@prefix dc: <http://purl.org/dc/elements/1.1/> .

@prefix userbk: <http://dataset.science.unitn.it/delicious/user/1234/bookmark/> .

@prefix tags2con: <http://dataset.science.unitn.it/delicious/tags2con#> .



: a sioc:UserAccount;
          sioc:id "1234";
          tags2con:vocabulary :tags;
          foaf:interest <http://en.wikipedia.org/wiki/Java>;
          sioc:owner_of :bookmarks.

:tags a skos:Collection;
```

```
      skos:prefLabel "user 1234 vocabulary";
      skos:member
        <http://dataset.science.unitn.it/delicious/user/1234/tag/holidays/1>;
      skos:member
        <http://dataset.science.unitn.it/delicious/user/1234/tag/javaisland/1>.

:bookmarks a sioct:BookmarkFolder;
                          sioc:container_of userbk:5678, userbk:5680.
```

 ©INSEMTIVES consortium 2009-2012

B The complete annotation model

Here we present the updated annotation model considering the changes included in Section 2.2. This Appendix should be used as the up to date reference in replacement of deliverable D2.1.2 [21].

GENERIC OBJECTS These objects are the basic elements that constitute the annotation model.

Model Object 1 (User)
A user u is a tuple $u = \langle id, name \rangle$, where id is a unique identifier of the user; and $name$ is the name of the user. We write U to denote the set of all users.

Model Object 2 (Resource)
A resource r is a tuple $r = \langle id, name \rangle$, where id is a unique identifier of the resource; and $name$ is the name of the resource. We write R to denote the set of all resources.

Model Object 3 (Term)
A term t is a non-empty finite sequence of characters. We write T to denote the set of all possible terms. Normally, terms represent natural language words such as "sea", "bird", or "location".

CONTROLLED VOCABULARIES The generic model defines a set of controlled vocabularies to restrict the meaning of terms describing a resource:

Model Object 4 (Synset)
A synset syn is a tuple $syn = \langle sid, ST, pt, desc \rangle$, where sid is a unique identifier of the synset; ST is a set of synonymous terms ($ST \subseteq T$); pt is a term from ST (i.e, $pt \in ST$) which is called the *preferred term* of the synset; and $desc$ is a natural language description of the meaning of the synonymous terms.

The set of terms ST models the fact that the same concept may be referred to with several synonymous words (e.g., "image" and "picture"). Preferred term serves rather for presentational purposes and is used, for example, as a synset name in the user interface. The description d serves to help a human user understand the meaning of the synonymous terms (e.g., "a visual representation (of an object or scene or person or abstraction) produced on a surface").

Model Object 5 (Natural Language Dictionary)
A natural language dictionary nld is a tuple $nld = \langle nlid, LT, SYN \rangle$, where $nlid$ is a unique identifier of the dictionary language; LT is a set of terms of this language ($LT \subseteq T$); and SYN is a set of synsets, such that the set of terms ST of any synset in SYN is a subset of the set of terms of the language, i.e., $ST \subseteq LT$.

Hereinafter we use the notion of *concept* as it is defined in description logics [16]. We write c to denote a concept and we write C to denote the set of all concepts.

Model Object 6 (Taxonomy)
A taxonomy tx is a rooted tree where each node is a concept $c \in C$ and each parent concept pc is more general than any of its child concept cc, i.e., $pc \sqsupseteq cc$. We write TX to denote the set of all taxonomies. We write $c \in tx$ to mean that concept c belongs to the set of nodes of tx.

Note that the term computation and concept computation functions can be used to map from natural language terms to the concepts in a taxonomy and vice versa.

ANNOTATION ELEMENTS In the following definitions, the user element u is made optional to model the situation in which an unregistered user annotates a resource.

Model Object 7 (Uncontrolled tag annotation)
An uncontrolled tag annotation tag^u is a tuple $tag^u = \langle t, r, [u,]ts[, \alpha] \rangle$, where t is the tag term used for the annotation ($t \in T$); r is the annotated resource ($r \in R$); when present, u is the user who created this annotation; and ts is the time stamp recorded when the annotation was created. When the annotation is added by an automatic or semi-automatic algorithm, α is provided to indicate the accuracy of the algorithm that created the annotation.

Model Object 8 (Concept Mapping Function)
Concept mapping function cmf is a function that takes a concept and a language identifier as input and returns

a synset as output, i.e., $cmf(c, nlid) \rightarrow syn$, s.t. syn belongs to the natural language nld dictionary with identifier $nlid$.

The concept mapping function serves for the computation of natural language representation of set-theoretic concepts.

Model Object 9 (Linguistic Concept)
A linguistic concept lc is a tuple $lc = \langle c, nlid, st \rangle$, where c is a concept ($c \in C$); $nlid$ is the identifier of a natural language dictionary defining the language of the term; and st is a term that belongs to the set of terms of the synset that is mapped to c and $nlid$ by the concept mapping function, i.e., $st \in cmf(c, nlid)$.

Model Object 10 (Controlled Term)
A controlled term ct is a tuple $ct = \langle t, \{lc\} \rangle$, where t is a non-empty finite sequence of characters normally representing natural language words or phrases such as "bird", "java island" or "sea"; $\{lc\}$ is an ordered list of linguistic concepts that link word (also referred to as token) in t to the controlled vocabulary.

Model Object 11 (Controlled tag annotation)
A controlled tag annotation tag^c is a tuple $tag^c = \langle ct, r, [u,]ts[, \alpha] \rangle$, where ct is the controlled term ct (MO23) encoding the tag text and the related concept c; r is the annotated resource ($r \in R$); when present, u is the user who created this annotation; and ts is the timestamp recorded when the annotation was created. When the annotation is added by an automatic or semi-automatic algorithm, α is provided to indicate the accuracy of the algorithm that created the annotation.

Model Object 12 (Uncontrolled attribute annotation)
An uncontrolled attribute annotation $attr^u$ is a tuple $attr^u = \langle an, av, r, [u,]ts[, \alpha] \rangle$, where an is a term denoting the attribute name ($an \in T$); av is the attribute value which can belong to any of the primitive data types (e.g., dates, floats, strings); r is the annotated resource ($r \in R$); when present, u is the user who created this annotation; and ts is the time stamp recorded when the annotation was created. When the annotation is added by an automatic or semi-automatic algorithm, α is provided to indicate the accuracy of the algorithm that created the annotation.

Model Object 13 (Controlled attribute annotation)
A controlled attribute annotation $attr^c$ is a tuple $attr^c = \langle can, av, r, [u,]ts[, \alpha] \rangle$, where can is a controlled term ct denoting the attribute name whose linguistic concepts $\{lc\}$ is linked to a taxonomy (i.e., $c \in tx$); av is the attribute value that can belong to any of the primitive data types (e.g., date, float, string) or that can be a controlled term ct encoding the attribute value whose linguistic concept lc is linked to a taxonomy (i.e., $c \in tx$); r is the annotated resource ($r \in R$); when present, u is the user who created this annotation; and ts is the timestamp recorded when the annotation was created. When the annotation is added by an automatic or semi-automatic algorithm, α is provided to indicate the accuracy of the algorithm that created the annotation.

Model Object 14 (Uncontrolled relation annotation)
An uncontrolled relation annotation rel^u is a tuple $rel^u = \langle sr, tr, rel, [u,]ts[, \alpha] \rangle$, where sr is the source resource (i.e., the resource being annotated); tr is the target resource (i.e., the resource used as an annotation object); rel is a term that denotes the name of the relation that exist between sr and tr ($rel \in T$); when present, u is the user who created this annotation; and ts is the time stamp recorded when the annotation was created. When the annotation is added by an automatic or semi-automatic algorithm, α is provided to indicate the accuracy of the algorithm that created the annotation.

Model Object 15 (Controlled relation annotation)
A controlled relation annotation rel^c is a tuple $rel^c = \langle sr, tr, crel, [u,]ts[, \alpha] \rangle$, where sr is the source resource (i.e., the resource being annotated); tr is the target resource (i.e., the resource used as an annotation object); $crel$ is a controlled term ct that denotes the name of the relation that exists between sr and tr and whose linguistic concepts $\{lc\}$ is linked to a taxonomy (i.e., $c \in tx$); when present, u is the user who created this annotation; and ts is the timestamp recorded when the annotation was created. When the annotation is added by an automatic or semi-automatic algorithm, α is provided to indicate the accuracy of the algorithm that created the annotation.

ACCESS CONTROL The following model objects are extensions tailored for the use case partners that require an access control extension to manage the read and write permissions on the resources (as expressed in the deliverable D2.1.1 [23] section 3.

 ©INSEMTIVES consortium 2009-2012

Model Object 16 (User Group)
A user group g is a tuple $g = \langle id, name \rangle$, where id is a unique identifier of the group; and $name$ is the name of the group. We write G to denote the set of all user groups.

Model Object 17 (Group Membership)
A group membership gm is a tuple $gm = \langle u, g \rangle$, where u is a registered user of the system such as $u \in U$ as defined in 1 and g is a user group to which this user belongs such as $g \in G$ as defined in model object 16.

Model Object 18 (Access Rule)
An access rule is a relation between a resource and a group defining the type of access right this group has on the annotations of that resource[35]. It is represented as a triple $ac = \langle r, g, right \rangle$ where $r \in R$ as defined in model object 2 and $g \in G$ as defined by model object 16. $right \in \{\texttt{read}, \texttt{write}\}$.

A user u has access to a resource annotations r if there exist an ac for this resource with a group to which the user belongs. A `read` access right allows the group to view and search the annotations of that resource while a `write` access right allows the group to add annotations, remove annotations and edit the existing annotations of a resource.

GRANULARITY The following model objects are extensions tailored for the use case partners that require a more granular annotation of resources where also its parts (e.g. a paragraph in a textual document, a part of an image) can be described by an annotation. For this purpose, we extend the resource definition given in MO2 to be able to refer to parts of resources.

Model Object 19 (Resource)
A resource r is a triple $r = \langle id, name[, part] \rangle$, where id is a unique identifier of the resource; and $name$ is the name of the resource. $part$ is an optional descriptor that identifies the part of the full resource identified by $name$.

The actual definition of a part is implementation dependent and will depend on the specific requirements of each use case and resource type. A part can take multiple form:

Model Object 20 (Textual Segment)
A textual segment is part of a textual document and is defined by a tuple: $part^t = \langle beginning, end \rangle$ where $beginning$ and end represent the number of characters from the beginning of the document to the start of the segment and the end of the segment, respectively.

Model Object 21 (Image Segment)
A image segment is part of an image and is defined by a singleton: $part^i = \langle region \rangle$ where $region$ denotes a geometrical region of the image.

Model Object 22 (Video Segment)
A video segment is part of a video and is defined by a triple: $part^v = \langle beginning, end, region \rangle$ where $beginning$ and end represent the number of seconds from the beginning of the video to the start of the segment and the end of the segment, respectively. $region$ denotes a static geometrical region of the video during this segment of time.

VERSIONING In the generic model, the annotation elements have a provenance information that stores the user and time of addition of an annotation to a resource. In some use cases, this provenance information must be extended to a full version history system that allows to store when an annotation was *added* or *removed* or when its value has been *updated*.

Because of the centralised organisation of the platform in the Insemtives project, there is no issue of synchronisation, consistency checking and merging branches in the versioning system. However, the way it is implemented in our model is similar to the popular patch system that can be found in many source versioning systems (e.g. git, svn, cvs) or wiki applications. It can thus be used to implement algorithms based on the operation intention model as discussed in [44] for instance.

To store this information, we introduce an history object to the model.

[35] Note that the specific access right systems for the resources are use case specific and cannot be encoded in this model.

Model Object 23 (History Element)
An history element is defined as a triple: $he = \langle action[, ov], ts[, \alpha]\rangle$. Where $action$ is the type of action that was performed $action \in \{$`added`, `removed`, `updated_name`, `updated_value`, `updated_source`, `updated_target`$\}$, ov – when applicable – is the value of the changed annotation before $action$ was performed; ts is the timestamp when this operation was performed. When the annotation is added or changed by an automatic or semi-automatic algorithm, α is provided to indicate the accuracy of the algorithm that created the annotation.
Note that the $action$ is specific to particular annotation objects:

- `updated_name`, applies to all, and implies that the attribute or relation name has changed. ov will take the name of the annotation before it was changed.
- `updated_value` applies to both attribute annotation objects (12 and 13) and implies that the value of an attribute has changed. ov will take the value of that attribute before the change.
- `updated_source` and `updated_target` apply to the relation annotation objects. ov will take the value of the source or target (respectively) before it was changed.

Each annotation element then refers to a set of history elements containing the modifications that specific annotation received. Thus, we modify the generic definitions of the Model Objects 7, 11, 12, 13, 14 and 15 to replace the $ts[, \alpha]$ provenance information by a set of history elements $\{he\}$.

Note that in the case of tag annotations (7, 11) an $update\_name$ is equivalent to the a $removed$ operation followed by an $added$ operation of a new annotation, but we still keep the operation to provide a more detailed tracking of an annotation history.

For instance, the Model Object 11 becomes:

Model Object 24 (Controlled tag annotation)
A controlled tag annotation tag^c is a tuple $tag^c = \langle ct, r, [u,]\{he\}\rangle$, where ct is controlled term encoding the tag text and the related concepts c; r is the annotated resource ($r \in R$); when present, u is the user who created this annotation; and $\{he\}$ is a set of history elements storing the different versions of this annotation element.
[end of document]

 ©INSEMTIVES consortium 2009-2012