Ontology of Folksonomy: A New Modeling Method

Francisco Echarte, José Javier Astrain, Alberto Córdoba, Jesús Villadangos
Dpt. de Ingeniería Matemática e Informática
Campus de Arrosadia
Pamplona Navarra (Spain)
+34 948 159556

{patxi@eslomas.com}{josej.astrain, alberto.cordoba, jesusv}@unavarra.es

ABSTRACT

Ontologies and tagging systems are two different ways to organize the knowledge present in Web. The first one has a formal fundamental that derives from descriptive logic and artificial intelligence. The other one is simpler and it integrates heterogeneous contents, and it is based on the collaboration of users in the Web 2.0. In this paper we propose a method to model tagging systems like folksonomies using ontologies. In our proposal, structured information (ontologies) can be extracted from knowledge built in a simple and collaborative way (folksonomies). Furthermore, we provide an analytical expression to evaluate the system requirements to store the derived ontology.

Categories and Subject Descriptors

H.1.1 [**Models and Principles]**: Systems and Information Theory – *Information theory.*

General Terms

Algorithms, Management, Languages.

Keywords

Folksonomies, Ontologies, OWL DL, Semantic Web, Web 2.

1. INTRODUCTION

Capturing knowledge by using markup techniques and by supporting semantic annotations is a major technique for creating metadata. Currently, in the WWW, folksonomies are more extended as tagging system due to its easy use. Folksonomies allow the integration of heterogeneous resources and the collaboration of users in the resource tagging process. However, it is difficult to work with such information because it has not any structure and it is user dependent. On other hand, ontologies provide a framework to handle structured information and to extract conclusions from such structured information. It is very interesting to provide mechanisms to turn existing syntactic resources into knowledge structures and therefore, modeling folksonomies with ontologies is an important issue.

Folksonomies and ontologies are two different ways to organize the knowledge present in the current Web.

A folksonomy [22] is a set of free text labels assigned to heterogeneous objects (images, documents, CAD drawings, virtual reality motion scenes, web resources, etc.) that are introduced by users in a very simple way that allows collaboration between them. The information of the folksonomy is the data (images, documents, virtual reality scenes, etc.), which is annotated by users. The labels are user dependent and represent the knowledge, interest, or interpretation of the users about the information contained in the folksonomy. Folksonomies depend totally on users and their opinion on the contents, because they annotate the content directly and freely. Therefore, the main characteristic of folksonomies is the absence of structure in the classification of the information and we consider that this characteristic is the main factor for the success of folksonomies. In fact, folksonomies have been applied in some social networks in Internet and for example it has helped to build the Web 2.0.

Ontologies [12] provide a domain vocabulary and define, in different formalization levels, the vocabulary semantic and the structure that relates the different terms of the vocabulary. Ontological engineering refers to the set of activities related with ontological process development, ontological life cycle, methods and methodologies to build ontologies, and the set of tools and languages supporting ontologies. In the last years, ontologies have been the focus of attention in multiple fields of research like Knowledge Engineering and Artificial Intelligence. They also have been applied in very different areas like Knowledge Management, Natural Language Processing, eCommerce, Intelligence Systems Integration, Bioinformatics, Education, and in the emergent area of Semantic Web [10].

1.1 Folksonomies

Folksonomies are the result of a collaborative annotation process. A folksonomy is composed of text labels, called tags, and a set of resources associated to those tags. A priori, in the annotation process there is not any restriction and, therefore, there is not any predefined hierarchy or restriction to define the tags, and it relies upon users criteria. In the last years, folksonomies have been used in different social networks like Flickr[1] and del.icio.us[2], where users annotate images and links respectively. In these social networks, metadata is assigned (or knowledge captured) following a decentralized process. Another interesting characteristic of

[1] http://www.flickr.com

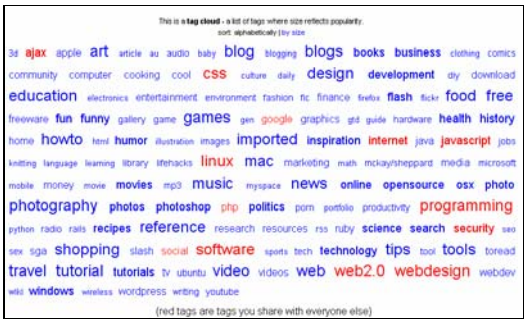
[2] http://del.icio.us

folksonomies is collaboration due to the fact that users can interact with the resources provided by others adding notes on them. Therefore, folksonomies contain a view of contents and knowledge wider than the contents they represent.

One usual example of folksonomy is del.ici.ous, a social network where users store URLs and annotate them. In this case, users can annotate all URLs present in the system and each URL can be located by means of tags assigned to it. Therefore users can access to any web resource annotated by any user. In this case, users can also collaborate to assign tags. Hence, the semantic of the resources and the used tags are included in a collaborative way.

Another example of social network, similar to del.ici.ous, is Flickr. In this case, tags can be used by different users, but each user annotates only its own resources (photos, images, etc.). In this way, the annotation process is individual. The semantic of the tags corresponds with the different resources that have been labeled by different users.

In the above examples and similar ones, the navigation process is based on clouds of words, called *tag clouds*, avoiding the taxonomic organization of tags. In Figure 1, we show an example of a *tag clouds* obtained with the most used tags in del.ici.ous in a particular date. Depending on its popularity (that is, the number of resources annotated with such tag), each word has a different size. Similarly, different colors indicate that tags are shared with other users.



This is a **tag cloud** - a list of tags where size reflects popularity.
sort: alphabetically | by size

3d ajax apple art article au audio baby blog blogging blogs books business clothing comics community computer cooking cool css culture daily design development diy download education electronics entertainment environment fashion fic finance firefox flash flickr food free freeware fun funny gallery game games gen google graphics gtd guide hardware health history home howto html humor illustration images imported inspiration internet java javascript jobs knitting language learning library lifehacks linux mac marketing math mckay/sheppard media microsoft mobile money movie movies mp3 music myspace news online opensource osx photo photography photos photoshop php politics porn portfolio productivity programming python radio rails recipes reference research resources rss ruby science search security seo sex sga shopping slash social software sports tech technology tips tool tools toread travel tutorial tutorials tv ubuntu video videos web web2.0 webdesign webdev wiki windows wireless wordpress writing youtube

(red tags are tags you share with everyone else)

Figure 1. Tags cloud

The knowledge captured with folksonomies has some drawbacks, (polysemy, synonymy, granularity) which are similar to the ones present in other classification systems like taxonomies or thesaurus [9][3].

Another problem specific of Folksonomies is the semantic of the tags. Although one can expect that tags are directly related with the information, it is not always the case. The flexibility of folksonomies allows users to assign tags based on their relation to the information. That is, when users define a tag they take into account not only the information to be annotated but also the purpose of the tag. Consequently, different tag types can be identified depending on its purpose [9]:

1. Identify "what about and who". These tags are used to identify what and who is the content about.
2. Identify "what it is". These tags indicate the type of the annotated resource: *blog*, *book*, etc.
3. Identify "the proprietary". These tags are used to establish who is the author or the proprietary of the content.
4. Categories. Some users use particular tags to simulate hierarchies.
5. Identify "characteristics of the content". These tags are usually adjectives (*funny*, *bored*, etc.) representing the opinion of the user that annotates the content.
6. Self reference. Such tags represent the relationship of the user with the content. Usually they begin with "*my*". For example, *mythings*, *myjob*, *mycomments*, etc.
7. Organize tasks. Such tags are used to simulate content classification in order to organize, for example, the work. Tags that fit into this class are *toread*, *todo*, *search-work*, etc.

Another additional problem present in Folksonomies arises from its collaborative philosophy. The direct implication of the final user in the construction of the knowledge structure, makes such structure very sensible to SPAM. That is, the uncontrolled action of malicious users providing for instance several wrong tags, can derive into poor knowledge models.

The problems we have described above refer to the annotation process. However, there are also some other problems related with the navigation for searching and accessing contents. Navigation problems can be grouped in two blocks [1]: (1) reduced search capabilities; and (2) limited exploration.

As the searching result is restricted to the specific tags used in the annotation process, search capability is reduced because of the linguistic and semantic limitations of tags. For example, if a user assigns the tag "cat" to a resource, and another one looks for the word "animal", that resource will not be shown; or if a user searches resources related to the tag "television", resources labelled with a "tv" tag, will not be shown.

In Folksonomies, navigation is a very simple and intuitive process that allows finding out interesting contents. There are two basic ways of navigation through the tag space: a) search and refine, b) use visualization tools of the tag space like the *cloud of words*. However, as we explain below, such mechanisms are not effective enough.

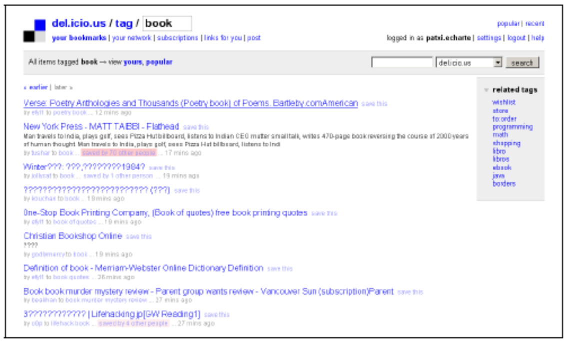
Search and refine is based on the selection or the search of a tag followed by a posterior refinement of the results. For example, when we look for the word "book" in del.ici.ous, we obtain a result similar to the one shown in Figure 2. This figure contains the set of web pages labelled with the word "book", and the set of related tags that can be used to extend the searching process.

The related tags provide a very basic way to continue the searching process or to refine the search. However, some generic related tags, e.g. those that identify personal aspects like *to:order* or *wishlist*, are not useful in practice. Many times it is better to access to the returned links to refine the search in them.

The other navigation method, that is, the use of visualization tools of the tag space, is the one used most frequently. The *cloud of words* belongs to this class (see example in Figure 1). In this case, the size of words is used to indicate the tag's frequency of use visually.

Though this navigation mechanism seems to be a good method of exploration, its utility decreases as the number of contents and tags increase [16]. The reasons, which should be avoided in other developments, are the following ones:

1. Tags are organized in alphabetical order without taking into account the relationship between them.
2. Low frequency tags are not showed hiding the point of view of some users. Therefore, one of the most interesting characteristic of Folksonomies, namely the integration of diverse points of view, is reduced.
3. There are many redundant tags, like *blog*, *blogs*, *blogging*, etc. that could be resumed.
4. Personal tags, like *toread*, which do not really provide additional semantic information interesting for other users are shown
5. This visualization is provided at the first level only, and it is not used in other places.


del.icio.us / tag / book
popular | recent
your bookmarks | your network | subscriptions | links for you | post
logged in as patxi.echarte | settings | logout | help
All items tagged book → view yours, popular
delicio.us search
« earlier | later »

Verse: Poetry Anthologies and Thousands (Poetry book) of Poems. Bartleby.comAmerican save this
by elyll to poetry book ... 12 mins ago

New York Press - MATT TAIBBI - Flathead save this
Man travels to India, plays golf, sees Pizza Hut billboard, listens to Indian CEO mutter smalltalk, writes 470-page book reversing the course of 2000 years of human thought. Man travels to India, plays golf, sees Pizza Hut billboard, listens to Indi
by tushar to book ... saved by 70 other people ... 17 mins ago

Winter???: ???,????????1984? save this
by jollyrat to book ... saved by 1 other person ... 19 mins ago

?????????????????????????? (???) save this
by kouchan to book ... 19 mins ago

One-Stop Book Printing Company, (Book of quotes) free book printing quotes save this
by elyll to book of quotes ... 19 mins ago

Christian Bookshop Online save this
????
by godlemercy to book ... 19 mins ago

Definition of book - Merriam-Webster Online Dictionary Definition save this
by elyll to book quotes ... 26 mins ago

Book book murder mystery review - Parent group wants review - Vancouver Sun (subscription)Parent save this
by bookman to book murder mystery review ... 27 mins ago

3???????????? | Lifehacking.jp[GW Reading1] save this
by c0p to lifehack book ... saved by 4 other people ... 27 mins ago

related tags
wishlist
store
to:order
programming
math
shopping
libro
libros
ebook
java
borders

Figure 2. Search and Refine

1.2 Ontologies

In [12] T. Gruber defined ontologies as the formal and explicit specification of a shared conceptualization. In this definition, *explicit* refers to the requirement to name all concepts and elements of the domain; *formal* means that a formal language must be used; and *shared* means that the points of view of all users involved in the domain are included in the ontology. Ontologies are structured knowledge where concepts, instances, attributes and relations are modelled. Thus, as the ontology models rigorously a domain, the ontology is a set of concepts that can be used by agents to dialogue with a common language.

The semantic of the ontology is supported by the descriptive logic formalism. We have the necessary axioms and inference rules to derive conclusions from the information contained in the ontology. Ontologies can be defined in different ways due to the existence of different languages to represent them. Therefore, it is very important to use mature standards to define it. From this point of view, OWL [21][6] is the best option. Proposed by the Wide Web Consortium (W3C), it is one of the main elements of the Semantic Web, a project initiated by T. Berners-Lee to include semantic in the Web [2]. OWL is supported by different technologies (XML, XML Schema, RDF [11], RDF Schema [3]), and all together provide~~s~~ a way to define a structure for documents and the explicit semantic relationship between different resources. All these technologies are open standards, tested and accepted.

However, ontologies have some drawbacks due to its difficulty to be built and used. If we consider ECM (Enterprise Content Management) systems, based on Web technologies, it is very difficult, for example, to represent a priori the whole scope of the domain for any enterprise. It is also difficult to anticipate the whole domain and the complete set of point of views of all possible users. Thus, ECM system deployments are continuously delayed because the domain is very frequently changed trying to consider all the possibilities. Usually such systems consider only a reduced version of the complete set of information. In addition, most users are simply information providers and they are not familiar with ontologies. Consequently, the insertion of new contents is a very difficult process for most users, because several criteria must be taken into account, and users neither are familiar with these criteria nor with the tools to work with ontologies. Another difficulty is that the structure of the information can vary with time and, therefore, it is necessary a monitoring process to update the structure of the information and the classification criteria. In conclusion, ontologies are difficult to maintain, mainly in changing environments.

1.3 Motivation

Ontologies are based on a solid formal model like the Descriptive Logic and an explicit, formal and shared representation of knowledge domain. On the other hand, folksonomies are based on a collaborative classification system, very simple to use and to implement. It produces a representation of the knowledge domain that emerges from the collaboration between the users with respect not only to the information annotated but also to the tags used. Although both techniques have their strong points, they have also theirs disadvantages.

In folksonomies, the collaborative way of classifying information in such an anarchical way, trying to extract patterns of order from certain chaos, has the important disadvantage that when the volume of information is high, or when the number of tags increase, the quality of the system decrease dramatically [14]. One of the reasons is the use of words badly written, plural instead of singulars, spaces or scripts between words, etc. Hence, when the number of tags increases, the navigation through the folksonomy becomes more complex and it is more difficult to find concrete resources. In addition, tags that are used less frequently are diluted among tags that are used most frequently, loosing some points of view that could be interesting for some users.

Furthermore, the absence of any type of formalism makes ~~it~~ impossible to classify in a simple way or make inferences more complex that the ones based in the relationship between contents based in the share of some tag.

The main problem of classification systems and ontologies in general, and OWL-DL in particular, comes from the expressivity power and their formalism, that usually makes it very complicated to create and use ontologies. Although in many situations it is convenient or even indispensable the creation of classification information, it can be expensive to implement and presents important problems of scalability [8].

The above analysis suggests that it would be convenient that current information web systems could take advantage of the best of both worlds, offering the possibility of structuring the knowledge on a formal way using ontologies, but without losing the dynamism that folksonomies offer. Since ontologies allow the representation of knowledge in a formal way, it seems reasonable to try to model folksonomies with them, incorporating these models to the structure of knowledge of the current information web systems based on semantic technologies.

On the one hand, this would allow having a common vocabulary referring to folksonomies that guarantees the interoperability between different applications, as the creation of common tools and libraries. On the other hand, this modeling could also be used to try to solve or to reduce the inherent folksonomies disadvantages, directly from the model, using the inference capabilities of reasoners over ontologies, or using external tools.

1.4 Contribution

Once folksonomies and ontologies have been analyzed, we remark that the success of folksonomies is the easy of use. However, knowledge structured in ontologies can be processed in a more efficient way allowing more elaborated conclusions due to the use of reasoners. In this paper, we propose a method to model folksonomies with ontologies, the first one to our knowledge. The method is composed of, on one side, a generic ontology structure that represents any folksonomy, and, on the other side, an algorithm to integrate the information contained in the folksonomy with the generic ontology. Thus, we provide an algorithm to obtain an ontology that contains the tagged information from the folksonomy. In addition, in this paper we provide a measure of the ontology size generated by our algorithm. This measure allows analyzing the system requirements before the transformation.

The main advantage of the proposed method is that it captures transparently the knowledge. The user annotates the content, resources, documents, etc., and dynamically and without user intervention, the proposed modelling method stores the information in the ontology. In addition, the method allows resolving two typical problems of folksonomies: (i) tag variability (for example, *blog*, *blogs*, *blogging*) and (ii) tags defined in terms of the objective of the tag and not on the content (for example, *toread*, *whislist*, etc.).

2. MODELING METHOD

Gruber, in his article "*Ontology of Folksonomy: a mash-up of Apples and Oranges*" [13], presents the different roles of ontologies and folksonomies in Semantic Web. He shows that both techniques are not completely opposed to each other but it is possible to get they complement each other. He gives some basic ideas on how to get it.

In his proposal, Gruber considers tagging as the activity in which some user annotates some content with one or more tags.

```
   Tagging(object, tag, tagger)
```

Gruber also considers two more characteristics, a first one centred in the sharing of taggings between different sources or applications, and a second one oriented to indicate the polarity (positive or negative) of each tagging, in an attempt to reduce problems derived from spam or incorrect taggings.

```
   Tagging(object, tag, tagger, source, + or -)
```

The work of Gruber presents the basic concepts involved in folksonomies and offers a starting point for the creation of a method for modelling folksonomies with ontologies. However, in order to create this method, it is necessary to detail more explicitly the characteristics of folksonomies and to do it in a stricter way, using some knowledge representation language like OWL DL. These languages offer the power and formalism of descriptive logics and allow the expressivity needed to model the different characteristics.

2.1 Description of the Modeling Method

The model consists on an ontology[3] designed in OWL language using the tool Protégé [18]. This ontology defines the following classes: Source, Resource, Tag, User, Annotation, AnnotationTag and Polarity. These classes have the objective to represent the model of knowledge of folksonomies.

- Source: as proposed by Gruber, it represents the sources or applications that use or feed the folksonomy.
- Resource: it represents any resource susceptible to be annotated. It has been renamed from the class "object", as Gruber uses in his article, to get a broader meaning. In an application of the ontology to a concrete domain this class could be specialized to represent documents, pictures, urls or whatever other thing.
- User: the aim of this class is to represent the users who do the tagging.
- Tag: it represents the tag concept. This class has several properties associated that are used to represent with a unique instance several tag variations like syntactic variations, incorrect spellings, or even synonyms. Two subclasses are also created: *TagPersonal* and *TagCommon*. Its objective is to be able to classify the existing tags according to their type, separating the ones of personal type, like those related to the planning of personal tasks or self-reference tags (*TagPersonal*), from the rest of tags (*TagCommon*).
- Annotation: it represents the action by which a user assigns a set of tags to a resource. Unlike the proposal of Gruber, who bases each annotation on the relationship of a tag with an object, this class represents better the habitual behaviour of users, consisting of assigning several tags to the annotated resources.
- AnnotationTag: as the Annotation class represents a set of assigned tags to a resource, this new class has been created to allow the relationship of each annotation to the assigned tags.
- Polarity: this class represents the polarity of each annotation, negative or positive. This polarity is associated to the instances of AnnotationTag. Therefore, it is possible to represent the assignation of several tags to a resource, and the polarity associated to each one of the assigned tags.

The ontology also contains a series of properties in the model. In Table 1, these properties are described indicating their name, domain, range, and a brief description.

It is necessary to explain more in depth some of these properties, specially *hasAltLabel*, *hasHiddenLabel* and last, *hasPosition*.

With respect to the two first, *hasAltLabel* and *hasHiddenLabel*, their objective is to represent the different variations of a tag, including singular and plurals, verbal tenses, synonyms, misspellings, incorrect syntactic forms, etc., from the tag's

[3] http://www.eslomas.com/tagontology-1.owl

preferred representation. For example, the tag with preferred value “rubyonrails”, could have associated to *hasAltLabel* the strings “ror”, “ruby”, and to *hasHiddenLabel* the strings “rubi”, “rubyonrials”, etc.

Table 1. Ontology properties

Property	Description
hasPrefLabel Domain: Tag, Range: string	The preferred lexical label of a tag.
hasAltLabel Domain: Tag, Range: string	Tag variations or synonyms of a tag, related to hasPrefLabel.
hasHiddenLabel Domain: Tag, Range: string	Tag variations or incorrect spellings of a tag, that should be hidden.
hasRelatedResource Domain: Tag, Range: Resource	It relates a Tag to the Resources that represents. This direct relation is created for efficiency with respect to the existing one through AnnotationTag and Annotation.
hasSourceName Domain: Source, Range: string	Name of the source/application.
hasUserName Domain: User, Range: string	Name of the user.
hasURI Domain: Resource, Range: string	Resource’s URI.
hasSource Domain: Annotation, Range: Source	It relates an annotation with a source.
hasUser Domain: Annotation, Range: User	It relates an annotation with a user.
hasResource Domain: Annotation, Range: Resource	It relates an annotation with a resource.
hasDateTime Domain: Annotation, Range: dateTime	It represents the date and time of an annotation.
hasAnnotationTag Domain: Annotation, Range: AnnotationTag	It relates one annotation with the AnnotationTags instances that represent the tags assigned.
hasTag Domain: AnnotationTag, Range: Tag	It relates each AnnotationTag with the corresponding tag.
hasLabel Domain: AnnotationTag, Range: string	It relates each AnnotationTag with the real text used by the user to tag the resource. This is necessary because hasTag associates one AnnotationTag instace with one Tag instance, but this can contain different representations among its hasPrefLabel, hasAltLabel and hasHiddenLabel properties. This property must correspond to one of those values.
hasPolarity Domain: AnnotationTag, Range: Polarity	It relates an AnnotationTag with the polarity (Negative o Positive) associated to the tag.
hasPosition Domain: AnnotationTag, Range: int	It relates each AnnotationTag with the position of the tag in the set of tags assigned to the resource in the Annotation.

With respect to *hasPosition* property, the convenience of this property is determined by the existence of studies that indicate that when a user assigns a series of tags to a resource, the order in which he does it, is not accidental, and that different annotations made by different users, agree more frequently in the first tags that in the last ones [9]. This characteristic could be used for example to help in the creation of taxonomical structures over tags.

To represent the implicit order of the different tags in an annotation, the most convenient solution would have been to define each *Annotation* like an ordered set of *AnnotationTag* instances. However, this is not possible with the actual specification of OWL. The reason is that it does not exist any explicit mechanism in OWL to define ordered lists of elements, and, although some work exists in the literature representing sequences of elements as linked lists of elements [7], the solution based in *hasPosition* property has been adopted, because of its simplicity and efficiency, allowing to get this information directly, without having to cross linked structures.

The considered approach produces that this ordenation is not directly accessible from a model based on OWL using a reasoner, but simplifies the usual requirements of recovering the tags assigned to a resource, or the resources tagged with a tag, using SPARQL [19] queries for that.

The proposed method is completed with a set of restrictions (Table 2) applied to the classes and properties described, that help to represent the knowledge contained in the folksonomy, and allow to validate the information generated in the ontology.

Table 2. Restrictions

Class	Restrictions
Source	Cardinality(hasSourceName) = 1
Resource	Cardinality (hasURI) = 1
User	Cardinality (hasUserName) = 1
Tag	Cardinality (hasPrefLabel) = 1
Annotation	Cardinality (hasSource) = 1 Cardinality (hasResource) = 1 Cardinality (hasUser) = 1 Cardinality (hasDateTime) = 1 Cardinality (hasAnnotationTag) >= 1
AnnotationTag	Cardinality (hasLabel) = 1 Cardinality (hasPolarity) = 1 Cardinality (hasPosition) = 1 Cardinality (hasTag) = 1

With the described model it is possible to have a complete representation of any folksonomy. However, it is necessary to specify a method that allows the transformation of existing folksonomies into the described model. On the other hand, it is necessary to consider that a folksonomy is not a static classification system because it evolves as users create new annotations on resources. Therefore, these annotations must be incorporated to the model. With the aim of solving both situations, namely, the transformation of a folksonomy to the described model and its evolution in time, an algorithm is proposed. This algorithm contains the set of actions necessary to model the annotations made by the users and creates a set of elements in the ontology. Figure 3 shows the algorithm of the method that models in the ontology an annotation made by a user.

```
- PRE: User and Source previously defined

If not exists Resource then
        Create Resource
        Resource.hasURI = uri
End If

Create Annotation
Annotation.hasDateTime = annotation date
Annotation.hasSource   = Source
Annotation.hasResource = Resource
Annotation.hasUser     = User

For Each tag assigned to the resource
        If not exists Tag with the same prefLabel or altLabel or hiddenLabel then
                Create Tag
                Tag.prefLabel = text of the assigned tag
        End If
        Create AnnotationTag
        AnnotationTag.hasLabel = text of the assigned tag
        AnnotationTag.Tag      = Tag
        AnnotationTag.Position = position of the tag in the set of assigned tags
        AnnotationTag.Polarity = [Positive|Negative]

        Tag.hasRelatedResource += Resource
        Annotation.hasAnnotationTag += Annotation
End For
```

Figure 3. Method of transformation and evolution

2.2 Ontology Size Estimation Analysis

Aiming at comparing our model with other ones based on ontologies, it is necessary to have some type of measure that allows us to estimate the amount of resources used. This measure can be based for example on the amount of information that the model contains.

This volume of information will be function of the existing instances at every moment in the modelled folksonomy, formed by the number of sources, users, resources, tags and annotations. Since the information represented in ontologies based in OWL is stored as RDF triplets or statements, the number of triplets represented by an ontology can be considered as a good measure of the necessary resources.

From the proposed method it is possible to define two formulas that allow us to get the number of instances represented in the model and the number of triplets contained in the ontology. They are based on a set of variables that define the state of the folksonomy, like the number of sources, users, resources, tags and annotations. Figure 4 shows both formulas. Observe that they do not considerer the use of any reasoner on the model that could produce the deduction of new triplets.

Figure 4 shows both formulas. They correspond to the calculation of the number of instances and the number of triplets of the ontology, without the use of a reasoner applied on the model that could produce the deduction of new triplets.

```
s: number of sources
r: number of resources
u: number of users
t: number of tags
a: number of annotations
α: average number of tags by annotation
β: average number of tags assigned to a resource

Number of instances = s + u + r + t + (1+α)a
Number of statements = 2(s+u+r) + (2+β)t + (5+6α)a
```

Figure 4. Size of the resulting ontology

Two averaged values are used in these formulas, namely, the average number of tags used by annotation and the average number of tags assigned collaboratively to a resource. These values depend on several things, such as the possibilities of the user interface where the annotations are created, the type of users, or the concrete field where the folksonomy is being applied. These issues can be studied in different papers (see [9][5][4]) or in the web site of some applications like del.icio.us.

To test the above formulas, some information has been captured for 48 hours from del.icio.us through the RSS channel offered by the web site. This information has been stored initially in a SQL database, getting 27.462 users, 50.728 resources, 30.412 tags and 53.698 annotations. This initial set of data has been reduced, taking only those resources with a minimum of 10 annotations. In this way we have obatained a data subset of 439 resources, 979 users, 1.219 tags and 1.069 annotations. Next, this subset has been transformed into ontology in OWL language, through the application of the transformation algorithm described in our method.

To get the values of the variables α and β, the computations have been done over the initial set of data. In this way, the values obtained are $\alpha=2.55$ and $\beta=3.54$.

An application in Java, using the semantic web framework Jena [15], has been created to compare real data with the estimated results obtained from the previous formulas. The application loads the model stored in OWL and returns the number of instances and triplets of the ontology without using any reasoner over the model, to avoid the creation of new inferred triplets.

Table 2. Estimation and real size of the testing ontology

	Estimated	**Real**
Instances Number	6.432	7.055
Triplets Number	30.305	33.688

The differences between estimated and real values are due to the use of the averaged values calculated for the initial dataset. However, if they were calculated with the reduced dataset, they would increase their values to $\alpha=3.14$ and $\beta=6.34$. With these values the estimated number of instances would be 7.064 and the number of triplets 33.456, much closer to the real values obtained.

2.3 Experimental Results

Several queries in SPARQL language have been designed to validate the proposed method against the searching and exploration problems in folksonomies with regard to tag variations, tag objectives or tag types used by the users.

With respect to the problem of tag variations, the model proposes the use of several properties associated to instances of the *Tag* class, like *hasPrefLabel*, to represent the preferred text of the tag; *hasAltLabel,* to represent different syntactic variations or even synonyms; and *hasHiddenLabel*, to represent incorrect variations that should not be shown. Consequently, one instance of *Tag* class is able to represent several text variations of a tag. In Table 3 we show the results we have got with a reduced dataset obtained from del.icio.us. We see how a user should make 4 different searches in del.icio.us, with different results each one, to get all the data on the programming language RubyOnRails. These 4 searches correspond to the different variations used usually by users to refer to this language, like "Ruby", "Ror", "Rails" and "Rubyonrails". However, with the proposed model, all this variations would be grouped in a unique *Tag* class instance, having one of them as preferred text, and the others as alternative or hidden variations. Hence, with the proposed model, it would be sufficient to make only one search with any of the tag variations, to get all the associated results.

Table 3. Number of results to each search

	Ruby	**Ror**	**Rails**	**Rubyonrails**
del.icio.us	4	1	7	3
Proposed model	15	15	15	15

A second characteristic of our model is that the grouping of the different variations of a tag in a unique *Tag* class instance decreases the number of instances of the *Tag* class with regard to the initial number of tags existing in the dataset obtained from del.icio.us. Table 4 shows that the reduced dataset obtained from del.icio.us has 1.210 tags whereas the number of *Tag* class instances in the proposed model, after the grouping, is only 886 (that is, 324 less).

Table 4. Number of tags

	Number of Tags
del.icio.us	1.210
Proposed model	886

With respect to the problem of objective or type tags used by users, the proposed model has two subclasses wich are specializations of *Tag* class. The aim of these subclasses is to divide the existent tags into two different subgroups. One of them (*TagPersonal*), correspons to tags that have a personal objective, like self-reference tags or task-planning tags. The other one (*TagCommon*), corresponds to tags that have a meaning and semantic common for all users.

The 886 tags existent in the ontology have been distributed in these two subclasses, obtaining a total of 61 tags in the

TagPersonal class and 825 in the *TagCommon* class, as is shown in the following table (Table 5).

In Table 5 we show how the 886 tags in the ontology have been distributed in these two subclasses: a total of 61 tags fit in the class *TagPersonal* whereas 825 tags fit in the *TagCommon* class.

Table 5. Number of tags based on its objective or type

	Number of Tags
TagPersonal	61
TagCommon	825

This division offers the possibility of improving the habitual browsing mechanism used in folksonomies, as *clouds of tags*, showing only the common tags *(TagCommon)*, and discarding the personal tags that usually are not interesting in a general browsing. Furthermore, this reduction in the number of tags by grouping different tag variations in only one *Tag* class instance, also improves the quality and utility of these *clouds of tags* because it reduces the number of some of the problems detected in them, and allows the inclusion of some tags that would have remained hidden.

3. CONCLUSIONS

A new method for modelling folksonomies with ontologies has been proposed. This method consists of: (1) an ontology able to be used to represent any kind of folksonomy, and (2) an algorithm to transform folksonomies into the proposed ontology and to update the resulting ontology as the folksonomy evolves in time. Estimation on the size of the ontology has also been proposed that allows knowing the system requirements before the transformation.

The representation model proposed in the ontology allows to solve two of the main problems identified in folksonomies in relation to searches and browsing limitations, namely (1) tag variations like plurals, verb tenses, synonyms, misspellings, etc., and (2) the different objective or type of the tags used by users, considering a separation between personal and common tags.

The proposed method has been verified with a dataset obtained from del.icio.us, obtaining the expected improvement in the searching and browsing process.

The representation model proposed here also offers a base for proposing solutions to other existing problems in folksonomies, like the deficiency of relations or structure between the existent tags [17]. In addition it also makes possible the incorporation of this type of systems of classification to information systems based on semantic technologies.

4. ACKNOWLEDGMENTS

Research partially supported by the Spanish Research Council TIN2006-14738-C02-02

5. REFERENCES

[1] Begelman, G., Keller P., Smadja, F. Automated Tag Clustering: Improving search and exploration in the tag space. *WWW2006*, May 22-26, 2006, Edinburgh, U.K.

[2] Berners-Lee, T., Hendler, J. and Lassila O. The Semantic Web. *Scientific American*, May 2001.

[3] Brickley D. and Guha, R.V. *RDF Vocabulary Description Language 1.0: RDF Schema.* W3C Recommendation 10 February 2004.

[4] Brooks, C.H. and Montanez, N. Improved Annotation of the Blogosphere via Autotagging and Hierarchical Clustering. *WWW 2006*, May 23-26, 2006, Edinburgh, UK.

[5] Damianos, L. E., Cuomo, D., Griffith, J., Hirst, D. M., Smallwood, J. Exploring the Adoption, Utility, and Social Influences of Social Bookmarking in a Corporate Environment. In *Proceedings of the 40th Hawaii International Conference on Systems Sciences*. 2007.

[6] Dean, M. and Schreiber, G. *OWL Web Ontology Language Reference. 2004*. W3C Recommendation 10 February 2004

[7] Drummond, N., Rector, A., Stevens, R., Moulton, G., Horridge, M., Wang, H.H., Seidenberg, J. Putting OWL in Order: Patterns for Sequences in OWL. *OWLed 2006*.

[8] Duval E., Hodgins W., Sutton S., Weibel S.L. Metadata Principles and Practicalities. *D-Lib Magazine*, Vol.8, No.4., 2002.

[9] Golder, S.A., Huberman, B.A. The Structure of Collaborative Tagging Systems. *Journal of Information Science 32*, 2, 198-208, 2005.

[10] Gómez-Pérez, A, Fernández-López, M., Corcho, O. *Ontological Engineering*, London, Springer-Verlag, 2003

[11] Graham K. and Jeremy J.C. Resource Description Framework (RDF): Concepts and Abstract Syntax. W3C Recommendation 10 February 2004.

[12] Gruber, T. A Translation Approach to Portable Ontology Specifications. *Knowledge Acquisition, 5*(2):199-220, 1993.

[13] Gruber, T. Ontology of Folksonomy: A Mash-up of Apples and Oranges. *AIS SIGSEMIS Bulletin*, 2005. *2*(3&4).

[14] Guy, M., Tonkin, E. Folksonomies - Tidying up Tags? *D-Lib Magazine*, January 2006, Volume 12 Number 1. ISSN 1082-9873

[15] Jena - A Semantic Web Framework for Java. *http://jena.sourceforge.net/*

[16] Mathes, A. Folksonomies - Cooperative Classification and Communication Through Shared Metadata. *Computer Mediated Communication*, Dec 2004.

[17] Passant, A. Using Ontologies to Strengthen Folksonomies and Enrich Information Retrieval in Weblogs. *International Conference on Weblogs and Social Media*, ICWM 2007. March 2007.

[18] Protégé. *http://protege.stanford.edu/*

[19] Prud'hommeaux, E., Seaborne A.. *SPARQL Query Language for RDF*. W3C Working Draft 4 October 2006.

[20] Sirin, E., Parsia, B., Cuenca Grau, B., Kalyanpur, A., Katz. Y. Pellet: *A practical owl-dl reasoner*. Technical report, University of Maryland Institute for Advanced Computes Studies (UMIACS), 2005-68, 2005. Available online at http://www.mindswap.org/papers/PelletDemo.pdf.

[21] Smith, M.K., Welty, C., McGuinness, D.L. *OWL Web Ontology Language Guide.* W3C Recommendation 10 February 2004.

[22] Vander Wal. T. Folksonomy. *http://vanderwal.net/folksonomy.html,* February 2, 2007.