Sharing resources through ontology alignments in a semantic peer-to-peer system

Jérôme Euzenat, Onyeari Mbanefo, Arun Sharma
INRIA & LIG
Grenoble, France
Jerome.Euzenat@inrialpes.fr

SUMMARY. In heterogeneous semantic peer-to-peer systems, peers describe their resources through ontologies that they can adapt to their particular needs. In order to interoperate, queries need to be transformed with respect to alignments between their ontologies before being evaluated. Alignments are thus critical for sharing resources in such systems. We report an experiment that explores how such alignments can be obtained in a natural way. In particular, asking users to provide alignments is a heavy constraint that must be relaxed as much as possible. This can be attempted through automatic matching. However, we suggest other possible solutions.

KEY WORDS. Semantic peer-to-peer system, semantic annotation, ontology, heterogeneous annotation, resource sharing, ontology alignment, ontology matching, query, peer data management system, alignment composition, alignment inverse, PicSter, semantic web.

BACKGROUND: Semantic peer-to-peer systems

The semantic web can be described as a web for machines. For that purpose, it requires the expression of formalised knowledge on the web (in languages like RDF). This is aimed at bringing more precision to the knowledge gathered from the web than automatically indexing text documents. However, the semantic web suffers from a bootstrap problem: it can only start providing benefits when there is enough knowledge available and people will not provide knowledge if this does not return benefits.

In order to overcome this problem, we want to provide tools that allow people to start using semantic web technologies locally, for personal purposes and personal benefits, and that can spread through global and social communication. On the one hand, peer-to-peer (P2P) systems are suited for this purpose since their use for sharing resources on the web is widespread and such systems can be used by many untrained users. On the other hand, such tools are restricted in the freedom of expression given to the user: most of them do not provide much expressiveness in resource annotations or they rely on a one-size-fits-all metadata vocabulary. This restricts the precision of shared item descriptions and thus prevent their full retrieval, e.g., “Finding all recordings in which Eric Dolphy is playing flute” or “All pictures featuring a member of your family and a horse”.

We describe a kind of semantic peer-to-peer system in which users can start locally to develop the annotation scheme that suits them best. They can customise ontologies, either imported from the web or home made, so as to organise their resources the way they want. Then, users can offer their resources and annotations to their friends and relatives through peer-to-peer sharing. However, because ontologies have been designed locally, it is more difficult to interoperate between peers: therefore, alignments between ontologies are necessary. An alignment is a set of assertions, called correspondences, which express the relation between entities of two ontologies. They can be used for translating queries from one ontology to the other, as well as for other purposes (Euzenat and Shvaiko, 2007). With alignments, interaction and sharing can take place, and users do not have to abandon their own view on resources.

This chapter presents the PicSter system as a case of heterogeneous semantic peer-to-peer system. It aims to illustrate the need for ontology alignments in such systems and the difficulties of providing such alignments.

After briefly presenting the field of semantic peer-to-peer systems, we describe extensively the PicSter picture sharing system. This description highlights the need for alignments in such systems. Then we elaborate on several possible solutions for obtaining these alignments and the difficulties facing these solutions. We do so through the description of an experiment.

Related work

In principle, peer-to-peer systems are distributed systems which do not rely, in their communication, on centralised resources. In practice, they often rely on some directory for joining a network. Once they have joined, all communication goes through adjacent peers. Peer-to-peer systems are well known as resource sharing devices in which peers offer resources and can query other peers for particular resources. We are concerned with such type of systems (other systems may be built on a publish and subscribe protocol, for instance).

Semantic peer-to-peer systems are peer-to-peer systems using semantic technologies for describing shared resources. The use of ontologies allows a richer description framework for resources than tags and simple categories. In particular, it allows querying the other peers with more precise queries ("Give me the Portraits depicting Paul Cezanne") instead of ("Portrait" and "Cezanne"). It also allows peers to draw inferences from the query and their ontology. For instance, something classified as a "Self-portrait" will be qualified as an answer for the above query if a "Self-portrait" is defined as a "Portrait". Similarly, a "Painting displaying Paul Cezanne" may be considered a "Portrait of Paul Cezanne" if "Portrait" is defined as a "Painting displaying some person". In summary, semantic peer-to-peer systems aim at providing a more complete and more precise answer to the queries that are exchanged between peers.

There have been several such systems. (Staab and Stuckenschmidt, 2006) provides a good overview of issues in such systems.

Edutella (Neijdl et al., 2002) was an early project for sharing educational resources. The system used the JXTA protocol to query over the network RDF annotation of these resources. One problem addressed by Edutella was to have more precise resource descriptions and queries while avoiding the fragmentation of niche P2P systems. The use of general purpose technologies such as those of the semantic web was the solution to this problem.

A first semantic peer-to-peer system was BibSter (Haase et al., 2004) developed in the framework of the SWAP European project. In BibSter, resources are bibliographic entries. They are expressed in RDF with regard to the same RDF schema (built from BibTeX). Hence queries are unambiguous. However, peers may have different ways to describe the same entry within BibTeX (using more or less information and more or less the same categories). Hence, the project has developed techniques for matching similar items (Haase at al., 2006).

In this chapter, we focus on open semantic peer-to-peer systems in which peers are free to use any ontologies they estimate suitable to their purposes.

Peer Data Management Systems (PDMS) are peer-to-peer systems whose shared resources are data. They have gone through some popularity in database. As data management systems, they are usually made of peers supporting a database that can be queried through an query language such as SQL. Peer databases are defined by a known schema which can be different from peer to peer. Although they do not provide a particularly expressive ontology, PDMSes address the problem of heterogeneity of the schema used by each peer through the use of mappings which are usually correspondences between queries.

Piazza (Ives et al., 2004) is such a PDMS based on XML and Xquery. SomeWhere (Rousset et al., 2006) is another such system initially based on Datalog with a well-defined query evaluation procedure. It has been extended towards RDF and OWL over the years. Most of the PDMSes are made of autonomous peers, non controlled by human beings. Other systems, such as BibSter and most of the well-known internet sharing systems, are directed by someone manipulating the system. This is also the context of the system we consider here.

Semantic peer-to-peer systems are confronted with important challenges such as routing queries in the network for efficiency or accuracy, replicating resources across peers, decomposing queries or finding the correct matching between two peers. These issues are well described in (Staab and Stuckenschmidt, 2006; Zaihrayeu, 2006). This chapter considers exclusively the issue of establishing alignments between ontologies and offering solutions to end users who are not ontology nor matching specialists.

SETTING THE STAGE: The PicSter system

PicSter is an experimental semantic peer-to-peer picture annotation and sharing system developed in 2006 at INRIA (Sharma, 2006). In PicSter, both the semantic sharing aspect and the annotation aspects are important: this is because users have more freedom in modifying their ontologies that the system is heterogeneous and this makes answering queries difficult.

Semantic peer-to-peer framework

This case study considers a particular framework that we call semantic peer-to-peer. This framework involves the following elements:

- Peers: are the agents which describe and share a set of resources, e.g., pictures.
- Resources: are the resources that will shared among peers. These resources can be stored in peer directories or available on the web.
- Ontologies: define the vocabulary used for annotating resources. They can be found on the web or be locally available at each peer.
- Instances: are individuals described by the ontologies. They are stored separately in each peer. They can also be referred from the web through URIs.
- Annotations: describe resources or resource fragments using the ontologies. Agents annotate resources by indicating that they belong to particular classes of the ontology, e.g., Picture, or they have relations with particular instances, e.g., depicts Grenoble.
- Queries: are expressions used for selecting resources with regard to their annotations. They are usually expressed in a dedicated query language.
- Alignments: express relationships between ontologies so that an annotation expressed with regard to one ontology can be understood with the vocabulary of another ontology. For instance, a “Lilium” under one ontology can be understood as a subclass of “Fleur” in another ontology.

Each peer can work independently of the others as a fully functional system. As far as content is concerned, this framework covers the examples of peer-to-peer systems that we have considered previously.

This framework provides the basic functions of semantic desktops (Schumacher et al., 2008):

- annotating a resource with elements from ontologies, and
- searching for resources annotated by particular elements.

Semantic desktops provide two additional and advanced features:

- extending ontologies, and

- sharing annotations.

In this framework, resources are identified by URIs, they are annotated in RDF (Carroll and Klyne, 2004) through vocabularies defined by ontologies in OWL (Dean and Schreiber, 2004) or RDF Schema (Brickley and Guha, 2004). Ontologies can be related by alignments and queries are expressed in SPARQL (Prud'hommeaux and Seaborne, 2008). Hence, this framework takes the maximum advantage of standard semantic web technologies.

This framework is instantiated in PicSter that we describe in the remainder.

PicSter overview and architecture

Our working example is the use of semantic web technologies for annotating resources (and particularly pictures) on one's computer. This is already a very helpful application since it can help people querying their data with respect to very precise and personal vocabularies (this is more powerful than simple schemes such as those used in Web 2.0 picture sharing sites).

We have developed PicSter, a prototype for ontology-based peer-to-peer picture annotation that allows accommodating users' needs in the ontologies used for annotation. PicSter provides the ability to annotate pictures in RDF by selecting individuals and classes in chosen ontologies. The ontologies can be modified on the fly by users as they need to refine the classification, e.g., that a Fjording is a Horse, or to add individuals, e.g., that Tony is a particular Fjording. This requires very little overhead.

Since users can select and evolve ontologies as they need it, their ontologies are necessarily different from each others. Hence, when connecting to the peer-to-peer system, users have to provide (partial) alignments between the ontology they use and those of the peer they are connected to. These alignments are used for transforming queries from one peer to another so that the query can be interpreted with regard to the other peer's ontology. These alignments may be stored in an alignment server and can be retrieved on demand.

PicSter peers can be used off-line, for annotating local pictures and querying them, or online, when a peer accepts incoming queries from authorised peers and can issue queries to these peers. In order to properly work off-line, PicSter peers cache most of the information they deal with. Hence, peers store:

- List of authorised peers,
- Resources (here pictures),
- Ontologies, as well as local extensions to these ontologies,
- Instances,
- Annotations.

These resources are stored in separate are stored in several silos (actually file system directories) of individual PicSter repositories according to the following rules:

- pictures: contains all the resources as individual files, unchanged.
- ontologies: contains the original ontologies as unchanged files and the extension of these ontologies as separate prefixed files.
- instances: contains all the individuals described with regard to the ontologies. They are stored as one file per concept containing all its instances.
- annotations: contains picture annotations expressed in RDF. There is one file per picture containing all its annotations.

An example of annotation is given in appendix. These directories are inspected when launching PicSter so that it can load ontologies and pictures. Annotations and individuals are loaded at query and annotation time.

PicSter offers two main functions (see Figure 1):

- Annotation editor allowing to (1) annotate artefacts with elements of the ontology, and (2) evolving the ontology to meets the annotator's need.
- Query interface allowing to answer queries (1) against the local store, and (2) by submitting them to connected peers through their alignments.

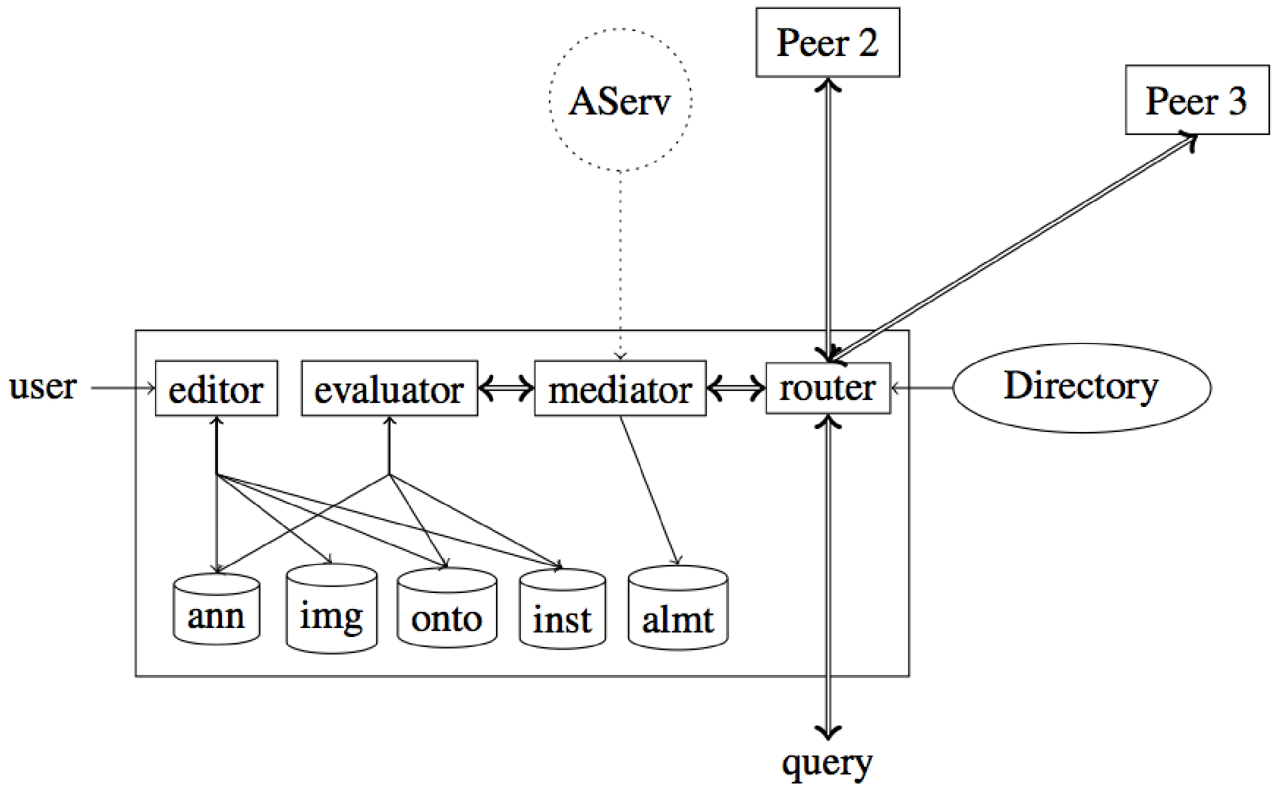


Figure 1: Architecture of a PicSter peer and relations to other elements. Simple arrows correspond to invocation and data flow. Double arrows correspond to query/answer interface. Aserv corresponds to the Alignment server, providing alignments to the mediator. It is in dashed lines because it is not available in PicSter's current implementation.

The goal of PicSter is to share the annotations and resources with other peers. For that purpose, peers are connected together and can send queries about their annotations to other peers and receive replies (under the form of lists of picture URIs satisfying the query). The queries are thus either evaluated locally or routed to other peers.

PicSter is based upon a peer-to-peer model without any centralised authority, but directories for finding peers. It is also an autonomous model: a PicSter peer without neighbours is a fully functional software. This authorises the off-line use of the tool: nothing pertaining to the peer is stored on the peer network.

The PicSter annotation tool is based on the PhotoStuff software (Halaschek-Wiener et al., 2005) that has been developed at MindLab (University of Maryland). PhotoStuff already had the ability to load images and ontologies, display images, select image areas, annotate pictures and zones with ontology concepts and properties, store the annotations in RDF and query them with SPARQL. We added the possibility to:

- modify ontologies on the fly and to store and reload these ontologies;
- load several ontologies;
- route queries to other peers and declare related peers;

- transform queries through alignments.

With regard to the description of Figure 1, PhotoStuff offered only the editor and the query evaluator. It had also a less rigourous way to store the data. This required many changes, in particular about how to name the new concepts or store the new ontologies.

PicSter step by step

We describe progressively how users can take advantage of PicSter in the way that was described at the beginning of the chapter: they first use PicSter for annotating their resources independently of any other peers. The first feature of PicSter is to help user organising their own resources without constraints. Then, users can connect to other peers and benefit from the work of others

Users start with an empty PicSter instance in which they can load “pictures” and “ontologies”. Both images and ontologies can come from their file system or the web. Under the hood, PicSter copies them in the images and ontologies directories, but it always maintain the initial URI identifying each of these objects.

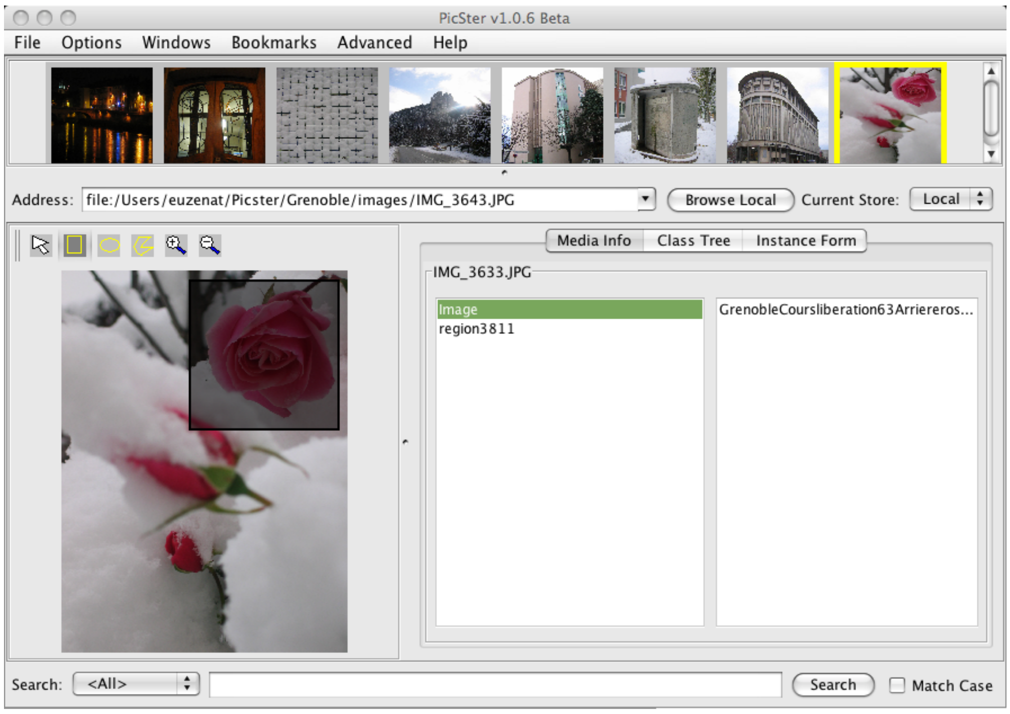
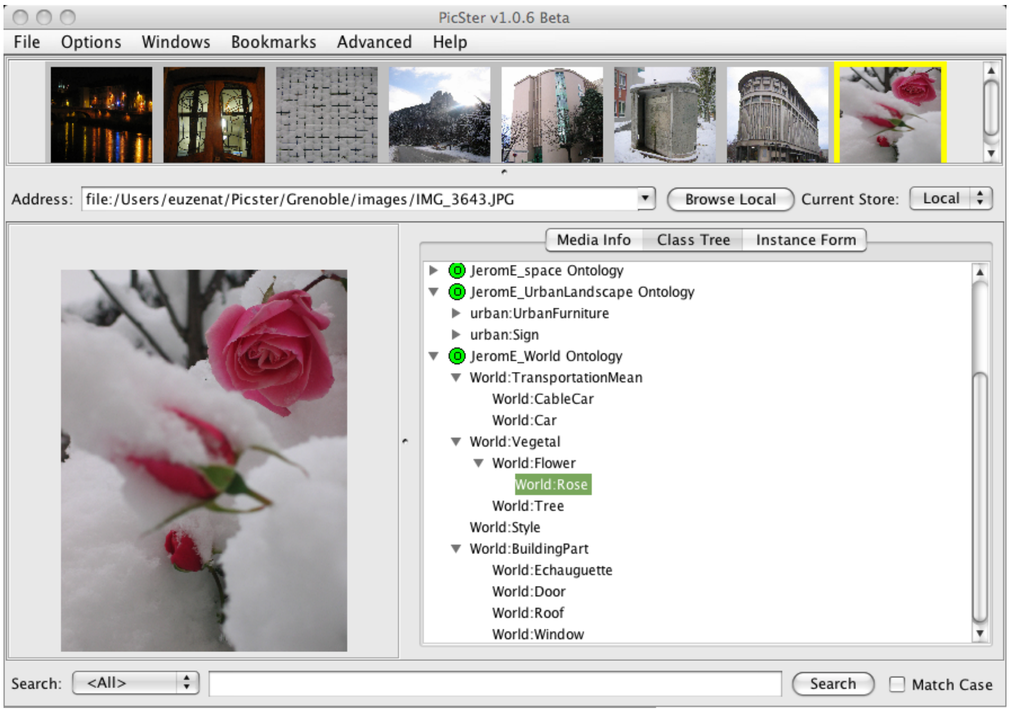

PicSter v1.0.6 Beta
File Options Windows Bookmarks Advanced Help
Address: file:/Users/euzenat/Picster/Grenoble/images/IMG_3643.JPG
Browse Local
Current Store: Local
Media Info
Class Tree
Instance Form
IMG_3633.JPG
Image
region3811
GrenobleCoursliberation63Arriereros...
Search: <All>
Search
Match Case

Figure 2: Main PicSter edition interface. On the top panel, pictures can be selected; on the left panel, a selected picture is displayed and subareas can be selected; on the right panel, annotations, ontology and instances can be displayed and selected. The bottom panel is used for expressing queries.

In order to annotate pictures, users can either select several pictures, one picture or a particular area in a picture. Then from the ontology panel, they can select a class for annotating the selected item with this class. For instance, in Figure 3, the user as selected the “Rose” class in ontology “World”. Instead of choosing classes, users can choose instances, i.e., individual objects. For instance, on Figure 2, the image has been annotated with the instance “GrenobleCoursliberation63Arriere”.



Address: file:/Users/euzenat/Picster/Grenoble/images/IMG_3643.JPG

Current Store: Local

JeromE_space Ontology
JeromE_UrbanLandscape Ontology
urban:UrbanFurniture
urban:Sign
JeromE_World Ontology
World:TransportationMean
World:CableCar
World:Car
World:Vegetal
World:Flower
World:Rose
World:Tree
World:Style
World:BuildingPart
World:Echauguette
World:Door
World:Roof
World:Window

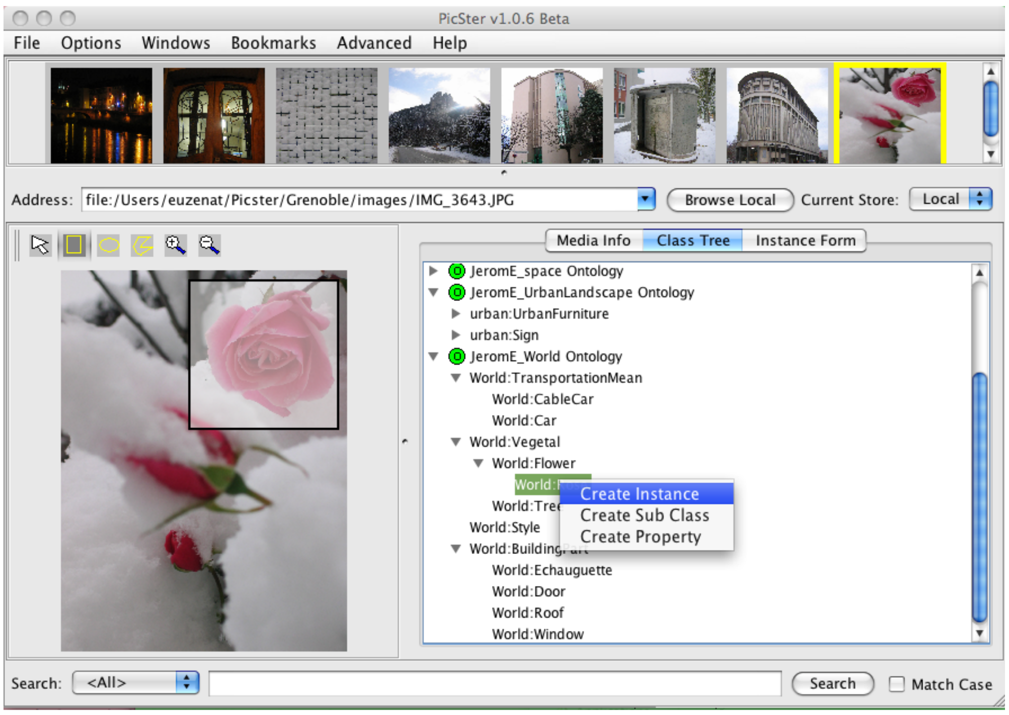
Search: <All>

Figure 3: Ontology representation and concept selection.

It may be that the ontology is not precise enough, in which case, the user can, on the fly, extend it. PicSter provides some basic ontology edition functions which are useful for quick personalisation of the picture annotation process. In particular, it allows users to extend the ontologies they use by:

- Creating new classes: The new class can be specified as a subclass of any class among the loaded ontologies. Labels and comments can be associated with it (see Figure 5).
- Creating new properties: PicSter allows users to create new properties for any of the existing or newly created ontology classes. Users can specify the range and domain of the property and hence can create both OWL Datatype and OWL Object properties. The property can also be specified as a subproperty of any property among the loaded ontologies.
- Creating new instances: of course, it is possible to create new individuals and to fill their properties, relating them to other individuals.

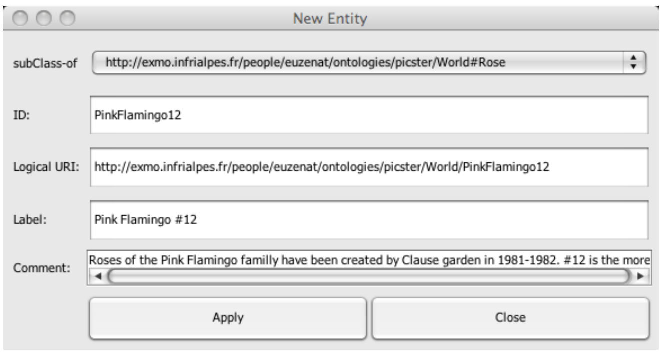
This basic ontology editing functionality ensures that users of the system do not have to stick to a base ontology for annotating their pictures. As shown in Figure 4, users can create directly from the annotation interface new instances, new subclasses and new properties if these are necessary.



Address: file:/Users/euzenat/Picster/Grenoble/images/IMG_3643.JPG

Figure 4: Addition on the fly of an instance, a subclass or a property to an existing ontology.

When selecting these items, in Figure 4, the “Create Subclass”, users are presented with a straightforward interface for providing minimal information about the new items. Here, the user knows that the depicted roses are from a particular variety, then they can create this new variety in the ontology as “PinkFlamingo12”. This change will, of course, not affect the existing ontology on the web but it will be stored in the local ontology extension of the user directory.



subClass-of	http://exmo.infrialpes.fr/people/euzenat/ontologies/picster/World#Rose
ID:	PinkFlamingo12
Logical URI:	http://exmo.infrialpes.fr/people/euzenat/ontologies/picster/World/PinkFlamingo12
Label:	Pink Flamingo #12
Comment:	Roses of the Pink Flamingo familly have been created by Clause garden in 1981-1982. #12 is the more

Figure 5: Required information for class creation.

These simple changes enabled in the annotation interface free users from having to design elaborate and extensive ontologies before annotating. However, since the ontologies are stored in genuine OWL, they can be edited with any ontology editor before or after the annotation phase. At the end of the process, the user has set up the elements depicted in Figure 6.

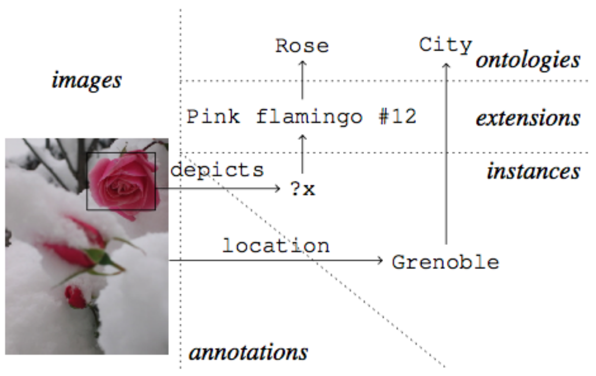


Figure 6: Resulting links between elements.

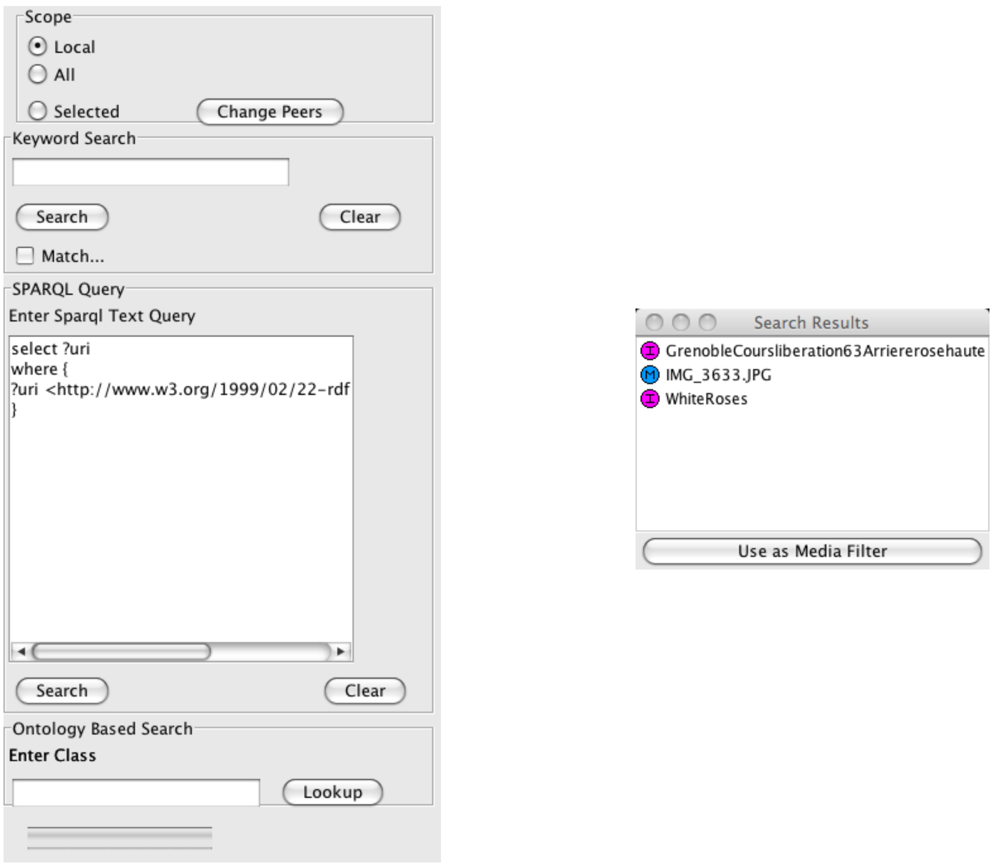
Once pictures are properly annotated, users can retrieve them through either full text search in annotations, concept search in annotations (directly from the bottom of the interface, see Figure 4) or through SPARQL querying (see Figure 7).

It is possible to evaluate simple queries like “what are the URIs of media elements representing something called Grenoble which contain a subarea which depicts a flower”:

```
PREFIX j.0: <http://www.mindswap.org/~glapizco/technical.owl#>
PREFIX j.1: <http://exmo.inrialpes.fr/people/euzenat/JeromE_world.owl#>
PREFIX j.2: <http://exmo.inrialpes.fr/people/euzenat/JeromE_space.owl#>
SELECT ?uri
WHERE
{
        ?x j.0:depicts ?y.
        ?y rdf:type j.1:Flower.
        ?x j.0:subAreaOf ?uri.
        ?uri j.2:isIn ?z.
        ?z j.2:name “Grenoble”.
}
```

This query returns the results provided in the right panel of Figure 7. The answer is provided in terms of instances and in terms of pictures. Thanks to the available ontologies it will also retrieve pictures which are not even annotated by “Flower” but by “Pink flamingo12”.

Query evaluation is implemented by using the ARQ SPARQL processor for the Jena API.



```
select ?uri
where {
?uri <http://www.w3.org/1999/02/22-rdf
}
```

GrenobleCoursliberation63Arriererosehaute
IMG_3633.JPG
WhiteRoses

Figure 7: On the left hand side, the query panel of PicSter 1.2.1. It allows to select if the query will be evaluated only locally or sent to related peers. On the right hand side, the results of query evaluation. Here these results can be either instances (I) or pictures (M).

Resource sharing

PicSter enables more than semantically annotating resources. It offers sharing resources and annotations among peers. For that purpose, peers are connected through the Java JXTA Protocol (Oaks et al., 2002). PicSter, through JXTA support, performs the tasks which are related to peer-to-peer communication. The design principles for this component are the ones which are typical of peer-to-peer systems, namely:

- A peer announces its presence to other peers when joining the network;
- Peer advertisements are sent and received in an asynchronous manner;
- All peers maintain a local cache of known peers, so that they can send queries to known peers even if they have not received an advertisement for these peers.

This peer-to-peer interface allows to declare some of the peers as friends. The support is visible in Figure 1 through the “Directory” component. It is dealt with, in each peer, through a “router” whose purpose is to route queries and answers from one peer to another.

Once a peer is connected to the network, it can evaluate its queries locally or against other peers. The interface provides the ability to send queries to all other peers or to select a subset of these (see Figure 7, left). A peer also enables other peers to send it queries to evaluate.

However, more is needed in order to evaluate queries. Indeed, since users are allowed to choose the ontology they use and to adapt them to their needs, they will end up having different ontologies. Hence, simply evaluating queries in others peers' context is deemed to fail. For solving this problem, PicSter uses mediators (see Figure 1) which transform queries with regard to a source ontology into queries with regard to a target ontology. Mediators themselves take advantage of alignments between the ontologies. Alignments are expressed using the Alignment API (Euzenat, 2004): an alignment is a set of correspondences between entities found in the ontologies. In our context, these entities could be classes, properties or individuals. For instance, the property “isIn” in one ontology may be more general than the property “city” in another one.

Query mediators translate queries. For this purpose, they take an ontology alignment and a query as input and generate the rewritten query with regard to the alignment. The current version of the mediator performs translations on the basis of equivalent correspondences given by the input alignment. The URI references within a query are replaced by their corresponding equivalent URI as mentioned in the input alignment. A further improvement would be to allow translations based on more elaborate correspondences. The mediator is implemented for queries written in the SPARQL (Prud’hommeaux and Seaborne, 2008) query language.

For instance, the following query returns the URI of all the resources which depicts some instance of “Fleur” in ontology Arun_sumo and which is situated in a “city” named “Grenoble”.

```
PREFIX j.0: <http://www.mindswap.org/~glapizco/technical.owl#>
PREFIX j.1: <http://exmo.inrialpes.fr/people/sharma/Arun_sumo.owl#>
PREFIX j.2: <http://daml.umbc.edu/ontologies/space-basic#>
SELECT ?uri
WHERE
{
        ?x j.0:depicts ?y.
        ?y rdf:type j.1:Fleur.
        ?x j.0:subAreaOf ?uri.
        ?uri j.2:city ?z.
        ?z j.1:id “Grenoble”.
}
```

In this query, the ontology j.0 is the internal ontology of PicSter providing the notions of depiction and areas. The subAreaOf relation is reflexive (so that a picture is a subarea of itself). The two other ontologies are the local extensions of respectively the SUMO and DAML space ontologies.

If the target peer also uses two ontologies, JeromE_world.owl and JeromE_space.owl, then four ontologies are involved apart from the PicSter ontology itself. If the peer finds an alignment between the ontologies providing the correspondences stating:

Arun_sumo.owl#Fleur	=	JeromE_world.owl#Flower
Arun_sumo.owl#id	=	JeromE_space.owl#name
space-basic#city	<	JeromE_space.owl#isIn

The first correspondence relates two concepts and the latter relates properties. It is noteworthy that the ontologies do not coincide one by one.

The mediator rewrites the input query according to these correspondences. This will yield the query presented in the previous section:

```
PREFIX j.0: <http://www.mindswap.org/~glapizco/technical.owl#>
PREFIX j.1: <http://exmo.inrialpes.fr/people/euzenat/JeromE_world.owl#>
PREFIX j.2: <http://exmo.inrialpes.fr/people/euzenat/JeromE_space.owl#>
SELECT ?uri
WHERE
{
        ?x j.0:depicts ?y.
        ?y rdf:type j.1:Flower.
        ?x j.0:subAreaOf ?uri.
        ?uri j.2:isIn ?z.
        ?z j.2:name “Grenoble”.
}
```

The rewritten query is able to find pictures which are not annotated by “Fleur” but by “Flower”.

This process works with SPARQL queries but it also works with more simple queries (such as those using only one concept name “Fleur”).

CASE DESCRIPTION: Obtaining alignments in peer-to-peer networks

So far, we have described how annotating pictures by ontology elements and altering ontologies themselves is made easy for end-users. This feature is important for the comfort of users but it raises the important problem of heterogeneity.
Hence, one important problem of a semantic peer-to-peer system such as PicSter is to obtain alignments so that queries can be transmitted from one peer to another. In order to solve this problem, users must obtain alignments between their ontologies. There are different ways to do this:

- manually: users can provide themselves the alignments by comparing the relevant ontologies;
- by automatic matching: users, or the system itself, can rely on matching algorithms in order to establish alignments (Euzenat and Shvaiko, 2007);
- by invert and composition: if enough alignments are available across ontologies, it is possible to combine them through algebraic operations (converse, composition, union, intersection) in order to obtain an alignment between the exact two ontologies that we have;
- by retrieving them from an alignment store: if high-quality alignments are available between ontologies, it is then possible to retrieve alignments from these stores;
- by local and aggregative interaction: it is possible to ask users for partial alignments which is sufficient for evaluating the query, then, as more queries are issued, the alignment between two ontologies will grow and eventually provide a full alignment.

It is possible to classify these techniques with regard to their expected quality and the effort required from the users, like in Table 1.

	Expected quality	Effort on the user
Manual	high	high
Automatic	low	low
Algebraic	medium (depends in input)	low
Store	high (depends on input)	low
Local	medium (correct but incomplete)	medium

Table 1: Solutions for obtaining alignments classified according to their quality and difficulty. Since, they depend on existing alignments, the algebraic and storage methods need a critical mass of alignments to be usable.

In order to evaluate if PicSter is a viable system and if Table 1 is an accurate picture of reality, we ran an experiment that we report here.

Experimental setting

In PicSter, autonomous semantic peers share resources by relying on alignments. In 2006, we ran an experiment in order to investigate the practicality of this framework. There were several goals of this experiment in relation to the work of our team:

- Test and evaluate the PicSter application and infrastructure;
- Provide experimental data for knowledge-based social network analysis;
- Provide experimental data for alignment algorithm evaluation;
- Provide experimental data on which to test alignment composition.

We have experimented PicSter on a set of pictures and users free to choose and modify the ontologies they use. Users where asked to provide an alignment with one of the other users. These alignments were used to generate new alignments through inverse and composition, and to evaluate simple automatic matching systems against the given alignments.

The experiment was made among 7 researchers of the Exmo team (named here: P, E, L, A, Z, J, S) during January and February 2006. They had a set of 133 pictures taken by themselves on the topic of Grenoble and the INRIA centre to annotate. A common subset 47 randomly selected pictures was compulsory to annotate in order to ensure a common coverage of the data set and relations in annotations.

Participants were totally free to choose the way to annotate the pictures, they were only required to use the PicSter interface to do this work. In particular, they could use existing ontologies, design their own or extend existing ones.

In the end of the experiment, this provided us with a set of ontologies per participants, a set of annotated images per participants, and a set of annotation per participants. Participants were also asked to evaluate the time they spent in annotating and to name the other participants to whom they feel to be the closest ontology-wise. This is summarised in Table 2.

Participant	#pic	#onto	#inst	#class	#prop	time	closest
S	47	3(SUMO,FOAF,SPACE)	66	158	242		
Z	47	5(AKT,office,SPACE+2)	87	141	45	18h	E
A	37	2(earthrealm,Travel)	81	738	38	11h	E+S
E	49	6(SPACE,FOAF+4)	78	390	30	35h	Z+L
J	47	1(SUMO)	49	1468	208	6h	S+L
P	30	1(SUMO)	27	1449	208	5h	L+Z
L	25	2(SUMO)	24	1395	416		

Table 2: Data from the participants in the experiment: #pic is the number of annotated pictures, #onto, the number of used ontologies, #inst, the number of instances that have been created, #class, the number of classes in the ontologies and their local extensions, #prop, the number of properties, and time, the approximate time spent for this annotation work. The closest column corresponds to the other participant, they felt their annotations would be the closest.

Participants took a relatively long time to provide the annotations. This was mostly due to the instability of the prototype.

What is surprising to us is that very few of the users (among the ontology-savvy) created their own ontologies. This is a good news for the semantic web showing that people are willing to reuse the existing resources if they are available. This can also be explained by the short time schedule or by the will of using the sharing philosophy of the web. Because this was possible in the interface, users used Swoogle (Ding et al., 2004) for finding ontologies related to the annotation they had in mind and extended them.

The second remark is that users have used upper-level ontologies (SUMO, AKT) rather than more specialised ontologies (SPACE, FOAF). This resulted in very large ontologies to be used.

After this step, each participant had been asked to evaluate and provide one alignment between his ontologies and those of another participant. In that step, we wanted to limit the burden put on users for providing alignments and yet preserve the possibility to compose alignments. The participants were organised in a circle (P → J → S → A → E → L → Z → P) in order to allow some manipulations related to reciprocity — convers — and transitivity — composition. No interface was provided to perform this task.

Alignment	#Correspondences	#Potential correspondences
P→J	14	2.127.132
J→S	17	231.944
S→A	50	116.604
A→E	75	287.820
E→L		544.050
L→Z	19	196.695
Z→P	138	204.309

Table 3: Characteristics of alignment returned by participants.

The resulting alignments were used for two distinct purposes:

- As reference alignments to compare with (simple) automatic matching techniques;
- As seeds for obtaining other alignments through converse and composition.

Additionally, we used the assessment by participants to evaluate the usability of the system and to compare their self-assessed affinity with distances on their ontologies.

Analysis

We analyse the results of this experiments with respect to several criteria. The usability of the PicSter system itself was considered fair by users. It was only a first prototype. Let consider the results concerning alignments and their computation.

Manual provision of alignments

The size of the resulting alignments is provided in Table 3. It shows a relatively low ratio of correspondences with regard to the size of the ontologies (especially if one compares with the number of potential one-to-one correspondences expressed in the third column). This easily confirms what was expected: users are not particularly willing to align ontologies by hand. This applies even if they have chosen themselves the ontologies. As a qualitative remark we had a lot of difficulties to obtain these alignments: while people have annotated pictures and altered ontologies within a few weeks, they spent several months providing the alignments.

As a result, there has been several workaround used by participants: some of them did not do the task, some of them did use very simple matchers (string equality), some of them did the task only partially. In particular, there were several people using SUMO which contains more than one thousand concepts. We could imagine that it is relatively easy to obtain an alignment from SUMO with itself and adding the few missing correspondences by hand. A simple program could be devised for doing most of this automatically. Unfortunately, these SUMO users found the task too overwhelming and returned very small alignments.

This shows that, according to our expectations, hand-made alignments indeed require a lot of effort, but, contrary to our expectations, the results were not of a high quality.

Evaluating simple matching algorithms

We wanted to compare the results of simple matching algorithms on real world data in order to have an idea of which such algorithms were the most useful (Mbanefo, 2006). We designed and run a script that: (a) computed alignments between all pairs of ontologies, and (b) computed the alignments resulting from applying threshold to these alignments. This resulted in 588 alignments. Some of these alignments could be compared, through classical precision and recall to the reference alignments provided by users. We also performed a consensus analysis in which we took the best threshold for each method, we did the intersection of the resulting alignments and we compared them to the other results.

Unfortunately, the bad quality of our reference alignments prevented to obtain good evaluation of the quality of the alignment obtained automatically. When comparing the simple matchers with the result of the participants who had used the simplest automatic matchers, then the algorithms always had 100% recall and around a 10% precision. Worse, for those participants who had used SUMO, the precision of the automatic matchers was very bad (0%), while the results were indeed correct. This was a case where automatically computed alignments were better than those provided by human subjects, but our protocol was unable to find it.

In spite of these bad conditions, we can note that, for those users who did the matching work properly, we had significantly better results with our simple matchers that for others (yet around 30% precision and recall).

Evaluating composition and inverse

The third method that we had planned for obtaining alignments was to start from existing alignments and to obtain other alignments by algebraic operations. We wanted to check the quality of the results.
Since participants were randomly organised in a circular manner, each alignment (x → y) can be computed by (a) inverting all alignments provided by participants, and (b) composing these inverses in order to go from x to y. For instance, the alignment from P to E can be obtained by $(Z \rightarrow P)^{-1}$.$(L \rightarrow Z)^{-1}$.$(E \rightarrow L)^{-1}$ or through (P→J).(J→S).(S→A).(A→E). Figure 8 shows how to start computing inverse and composition from the set of alignments provided by participants.

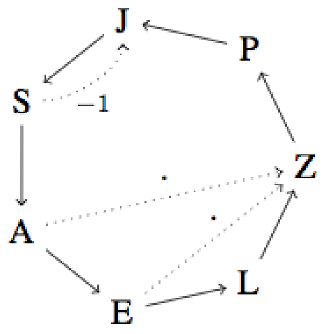


Figure 8: The set of participants and input alignments between them (plain arrows). In dotted arrows are examples of alignments obtained by composition (.) and inverse ($^{-1}$).

Once this was achieved, it would have been possible to evaluate the results with simple measures:

- by comparing the result to the given alignment;
- by having users evaluating the result;
- by evaluating the amount of lost information.

We generated the alignments corresponding to the inversion method, unfortunately, our inverse operator was not working properly. This, conjugated with the poor alignments provided by participants, prevented us to exploit results.
The algebraic operations are so highly dependent on the quality of initial alignments that we did not even perform consensus tests like comparing the results that could be obtained by the two paths above.

Conclusion

In conclusion, computer experiment is a very delicate task. It requires many people available and this is not easy for complex experiments like this one. The experiment was interesting but non conclusive due to a number of problems:

- PicSter was not the most robust and easy going system in the world. This frustrated participants.
- There was a bug in the invert function of the Alignment API which reduced our confidence in the results.
- Providing alignments by hand is a very tedious task, thus some participants (with large ontologies) did not perform it with the required quality. Hence, results were not conclusive.
- Some of the instructions were not clear enough.

Most of these problems come from the data collection and not their exploitation, hence this kind of experiment could be run again with the above problems fixed.

A specific kind of evaluation that could have been performed is introduced in (Isaac et al., 2008): instead of evaluating alignments with regard to reference alignments, it consists of using the alignments for answering queries and asking users to evaluate the answers or having reference answers. More precisely, each participant would have to ask identified queries expressed in their ontologies and to identify the answers to these queries from the annotated pictures by anyone. Then, precision and recall could be given with regard to the answer returned by transformed queries with regard to the alignments. In the case of PicSter, such an evaluation would have been relatively natural since the only purpose of the alignments is querying. It would have had the advantage of showing to the users how bad their alignments were and enabled the evaluation of automatic alignments: automatic alignments would have provided better results than partial alignments. This would certainly have required more annotated pictures, but it would evaluate both the alignments, the ontologies and the annotations of all the peers.

However, the main problem is the need for good quality alignments.

CURRENT CHALLENGES FACING THE ORGANISATION

PicSter is an experimental semantic peer-to-peer system. Beside its prototype status, it is functional but not really stable. Its characteristics are:

Semantic
: by using ontologies (as opposed to tags or keywords) for annotating and querying;

Serendipity
: by extending ontologies through annotating pictures, non obtrusively;

Sharing
: by connecting peers in a natural and personal way.

The peer-to-peer aspect of PicSter is highly relying on alignments for reducing heterogeneity. We have attempted to assess the possible ways to obtain alignments in systems like PicSter.

The main conclusion of this experiment is that the task of providing good alignments manually is very difficult for users, even if the ontologies have been chosen by them. Hence, other ways should be investigated.

In particular, it is critical for this application and, we guess, for many others to have automatic matching systems delivering alignments of reasonable quality. These alignments could, of course, be reviewed before being used, either by users when they issue queries or by curators before entering alignments in a store.

Moreover, composing alignments in such networks may be useful when one does not have a direct alignment of his or her ontology to those of another peer. On the one hand this is supposed to reduce the effort; on the other hand, composition degrades the results. It would be worth finding the paths between peers and compose the alignments along those paths which minimise the loss. It would also be useful to really measure how much is lost and to investigate the trade-offs involved in composing alignments. In any case, algebraic composition can only be achieved if there are sufficient high quality alignments it cannot rely on those provided on the spot by users.

Since good quality alignments are difficult to obtain, efforts should be devoted to maintain them. In particular, when one of the aligned ontologies changes, the alignment should not be rebuilt from scratch but the alignments should be evolved with regard to the change. This can be achieved with the help of composition of the alignment with a unique alignment between the new and the old version of the ontology. Alignment evolution should be integrated in an alignment management perspective (Euzenat et. al., 2008).
We planned from the beginning to store alignments in an Alignment server (Euzenat et al., 2008). However, at the time PicSter was developed the Alignment server was not ready so we stored the alignments within the peers themselves. There was one alignment covering all ontologies used by a pair of peers which was used for transforming queries between these two peers. An Alignment server can either store alignments or compute them on the fly with embedded matching methods. The advantages of using an Alignment server is that alignments may be shared by several peers (at least concerning the non altered parts of ontologies) and even shared with other applications of the semantic web. They can be retrieved from the server and modified with regard to the changes a peer has made to these ontologies.

Last, but not least a new trend recently emerged for facing the difficulty of obtaining high quality alignments. It consists of obtaining partial alignments from users while they accomplish their tasks (Conroy, 2008). In our case, the task may be either modifying the ontology or, more likely, querying. Instead of asking users for an alignment beforehand, the idea is to ask users for their help. When a query is issued, the system can ask the user for a partial alignment regarding the query, i.e., the terms in the query or the query as a whole. This can be done relatively easily by displaying the query and the target ontologies. Users who want to see their queries answered should show more willingness to help the system in this specific way. This local alignment could be recorded as a special type of alignment that can be reused when these alignments are necessary. The peer-to-peer system may collect many of these alignment fragments and the aggregation of many such partial alignments provided by many users over time will eventually provide high quality alignments between ontologies. Even partial, such alignments may be sufficient for answering a large proportion of the queries.
Hence, local query alignment is a cheap way to serendipitously obtain alignments from users: it seems to be a promising way to obtain the necessary alignments in semantic peer-to-peer systems.

The topic of finding alignments for heterogeneous semantic peer-to-peer systems is not closed yet. The problem of bootstrapping the semantic web with formal knowledge has resulted in that of bootstrapping with alignments. Once such alignments are available, we think that querying in semantic peer-to-peer systems will become easier. Like we promoted simple user interaction for stimulating ontology edition by users, the same could apply to alignments with local alignments: a simple interface for providing local and task-related clues to the system could go a long way.

The code of PicSter can be obtained from ftp://ftp.inrialpes.fr/pub/exmo/software/picster/PicSter-1.2.1m.zip
However, PicSter is an unsupported research prototype and is not suitable for use. The test results can be obtained from ftp://ftp.inrialpes.fr/pub/exmo/tmp/picster-tests.zip.

Acknowledgements

We thank the PhotoStuff, Jena and JXTA teams for developing and making available the software

that we have extended.
We thank Faisal Alkhateeb, Jason Jung, Sébastien Laborie, Jérôme Pierson and Antoine Zimmermann for their participation in this project.

References

Brickley, D., & Guha, V. (2004). RDF vocabulary description language 1.0: RDF schema. Recommendation. W3C.

Carroll, J., & Klyne, G. (2004). RDF concepts and abstract syntax. Recommendation. W3C.

Conroy, C. (2008). Towards semantic mapping for casual web users. I n Proceedings of the 7th International Semantic Web Conference (ISWC) doctoral consortium, Karlsruhe (DE), volume 5318 of Lecture notes in computer science (pp. 907–913).

Dean, M., & Schreiber, G. (2004). OWL Web Ontology Language: Reference. Recommendation. W3C.

Ding, L., Finin, T., Joshi, A., Pan, R., Cost, R., Peng, Y., Reddivari, P., Doshi, V., & Sachs, J. (2004). Swoogle: a search and metadata engine for the semantic web. In Proceedings of the 13th ACM Conference on Information and Knowledge Management (pp. 652-659).

Euzenat, J. (2004). An API for ontology alignment. In Proceedings of the 3rd International Semantic Web Conference (ISWC), Hiroshima (JP), volume 3298 of Lecture notes in computer science (pp. 698–712).

Euzenat, J., & Shvaiko, P. (2007). Ontology matching. Heildelberg (DE): Springer-Verlag.

Euzenat, J., Mocan, A., & Scharffe, F. (2008). Ontology alignments: an ontology management perspective. In Hepp, M., De Leenheer, P., De Moor, A., & Sure, Y. (Eds.), Ontology management: semantic web, semantic web services, and business applications (pp. 177–206). New-York (NY US): Springer.

Haase, P., Schnizler, B., Broekstra, J., Ehrig, M., van Harmelen, F., Menken, M., Mika, P., Plechawski, M., Pyszlak, P., Siebes, R., Staab, S., & Tempich, C. (2004). Bibster – a semantics-based bibliographic peer-to-peer system. Journal of Web Semantics, 2(1), 99–103.

Haase, P., Ehrig, M., Hotho A., & Schnizler, B., (2006). Personalized information access in a bibliographic peer-to-peer system. In Staab, S., & Stukenschmidt H. (Eds.), Semantic web and peer-to-peer (pp. 143–158). Heidelberg (DE): Springer.

Halaschek-Wiener, C., Golbeck, J., Schain, A., Grove, M., Parsia, B., & Hendler, J. (2005). Photostuff - an image annotation tool for the semantic web. In Proceedings of the 4th International Semantic Web Conference poster session.

Isaac, A., Matthezing, H., van der Meij, L., Schlobach, S., Wang, S., & Zinn, C. (2008). Putting Ontology Alignment in Context: Usage Scenarios, Deployment and Evaluation in a Library Case. In Proceedings of the 5th European Semantic Web Conference (ESWC), Tenerife (ES) (pp. 402–417).

Ives, Z., Halevy, A., Mork, P., & Tatarinov, I. (2004). Piazza: mediation and integration infrastructure for semantic web data. Journal of Web Semantics, 1(2), 155–175.

Mbanefo, O. (2006). Comparaison expérimentale d'algorithmes d'alignement d'ontologies.

Unpublished TER manuscript. Grenoble (FR): Université Joseph Fourier.

Nejdl, W., Wolf, B., Qu, C., Decker, S., Sintek, M., Naeve, A., Nilsson, M., Palmér, M., & Risch, T. (2002). Edutella: A P2P Networking Infrastructure Based on RDF. In Proceedings of the 11th Worldwide web conference, Honolulu (HA US) (pp. 604–615).

Oaks, S., Travaset, B., & Gong, L. (2002). JXTA in a nutshell. Sebastopol (CA US): O'Reilly.

Prud'hommeaux, E., & Seaborne, A. (2008). SPARQL query language for RDF. Recommendation. W3C.

Rousset, M.-C., Adjiman, P., Chatalic, P., Goasdoué, F., & Simon, L. (2006). Somewhere in the semantic web. In Proceedings 32nd International Conference on Current Trends in Theory and Practice of Computer Science (SofSem), Merin (CZ), volume 3831 of Lecture notes in computer science (pp. 84–99).

Schumacher, K., Sintek, M., & Sauermann, L. (2008). Combining Metadata and Document Search with Spreading Activation for Semantic Desktop Search. In Proceedings 5th European Semantic Web Conference (ESWC), Tenerife (ES) (pp. 569–583).

Sharma, A. (2006). Lightweight synchronization of ontologies. Unpublished Master's thesis. Aachen (DE): RWTH.

Staab, S., & Stukenschmidt, H. (2006). Semantic web and peer-to-peer. Heidelberg (DE): Springer.

Zaihrayeu, I. (2006). Towards peer-to-peer information management systems. PhD thesis. Trento (IT): University of Trento.

APPENDIX: RDF Annotations

In annotations/IMG3833.JPG.rdf:

```
<rdf:Description rdf:about="http://exmo.infrialpes.fr/people/euzenat/elster/images/IMG3833.JPG">
  <j.0:make>Canon</j.0:make>
  <j.0:resolutionUnit>Inch</j.0:resolutionUnit>
  <j.0:orientation>top, left side</j.0:orientation>
  <j.0:model>Canon PowerShot A80</j.0:model>
  <j.0:xResolution>180 dots per inch</j.0:xResolution>
  <j.1:depicts rdf:resource="http://exmo.infrialpes.fr/people/euzenat/elster/instances/meteo/Condition.rdf#Clouds"/>
  <j.1:depicts
        rdf:resource="http://exmo.infrialpes.fr/people/euzenat/elster/instances/euzenat/House.rdf#GrenobleCoursliberation38"/>
  <j.0:compression>JPEG compression</j.0:compression>
   <j.0:imageLength>1704</j.0:imageLength>
</rdf:Description>
```

In instances/House.rdf:

```
<rdf:Description
          rdf:about="http://exmo.infrialpes.fr/people/euzenat/elster/instances/euzenat/House.rdf#GrenobleCoursliberation38">
  <rdfs:label>38, cours de la Liberation, Grenoble</rdfs:label>
  <rdf:type rdf:resource="http://space.frot.org/rdf/space.owl#House"/>
</rdf:Description>
```

In Ontologies/space.owl:

```
<owl:Class rdf:about="http://space.frot.org/rdf/space.owl#House">
  <rdfs:comment xml:lang="EN">A (private) house.</rdfs:comment>
  <rdfs:label xml:lang="EN">House</rdfs:label>
  <rdfs:subClassOf>
    <owl:Class rdf:about="http://exmo.infrialpes.fr/people/euzenat#Inhabitation"/>
  </rdfs:subClassOf>
</owl:Class>
```