Semantic indexing modelling of resources in personal and collective memories based on a P2P approach

Cristian LAI

Thesis Director: Claude MOULIN
Outline

- Motivation
- Semantic Indexing
- Indexing Patterns
- Community of Users
- Conclusion
- Future Work
Motivation

Context

- Loose community of users
  - Private and shared resources.
  - Experts and general users.
- Resources of different types.
- Any nature of community
  - Possible focus on community of teachers and students.
Motivation

Issues

- How to manage publication and retrieval contexts?
  - How to match the description made during the retrieval context with a description made during the publication context?

- How to transform a user understandable description to machine understandable one?
  - How to create a formal description from the user input?

- How to make possible the life of a decentralized community?
  - How to manage a certain level of communication among members?
  - How to manage elements allowing the indexing of resources?
Motivation

Contribution

- How to manage publication and retrieval contexts?
  - Description extension.
  - The description is enlarged during publication to foresee different possible retrieval situations.

- How to transform a user understandable description to machine understandable one?
  - Model of Indexing Patterns.

- How to make possible the life of a decentralized community?
  - A distributed Semantic Wiki of the community and a distributed system of Notes.
  - A set of Core resources is managed for allowing the indexing.
Outline

- Motivation
- Semantic Indexing
- Indexing Patterns
- Community of Users
- Conclusion
- Future Work
Semantic Indexing
A P2P system

search key

closest peer to the requested key

request propagation

retrieving peer

Distributed Index

Key_1

Key_2

...

Key_j

values

URL

Data content
Semantic Indexing

Indexing

- P2P networks require the Boolean indexing
  - We choose to adopt the same Boolean indexing also for the personal memory.

- Indexing is the process of creating or updating an index
  - Given a list of resources it is necessary to create their proper descriptions different from the only title.

- The only *title* of a resource does not give a meaning universally known
  - In traditional filesharing systems it is usually used the title or a set of keywords to identify a resource.

- We consider semantic descriptions built manually by users.
Semantic Indexing
Description and query

- A *description* is supplied during publication
- In a Boolean Index, for retrieving a resource it is required the same description
  - Exact matching between descriptions.
- A *query* is supplied during retrieval
  - It is equivalent to a description of a potential set of resources.
Semantic Indexing
Ontologies and Knowledge bases

- Expert community members are able to
  - Find the proper *ontologies*.
  - Build a population of an ontology grouping the most prominent individuals of the domain: knowledge base.

- Ontologies and knowledge bases are available within the community.

- *Open description*
  - Add keywords to the description.
  - Guidelines for preventing typing ambiguities.
Semantic Indexing
Semantic description

The user thinks the possible description of the resource

Description in NL

The user provides the necessary input

Semantic Description (formal representation)

Final Description

The system analyzes the ontologies

The system produces the Semantic Description
Semantic Indexing
Types of Descriptions

- Resource type
  - Address resources giving elements of description concerning the resource itself and not its content.
  - *Document written by Chomsky.*

- Content type
  - Address resources giving elements of description that concern their content.
  - *Document about Chomsky.*
Semantic Indexing
Resource Type

- *Document written by Chomsky.*
- An ontology of domain is necessary. It should contain:
  - A concept that can represent the resource.
  - A concept that can represent an author.
  - A relation that binds the document to the author.
- The resource is considered as an instance of the concept that represents the resource itself.
- The system must show the concepts of the ontology that can represent the resource: *Entry Point*
  - The ontology provider has to declare what are the Entry Points.
Semantic Indexing
Content Type

- *Document about Chomsky*
- An ontology that represents the resource and its content is necessary.
- We have defined the *System Ontology* that contains
  - The concept *system:Document* that represents the resource.
  - The property *system:hasInterest* that paraphrases *about*.
- It is necessary to have an ontology of domain for extracting the concept describing the content
  - It is necessary to represent the author *Chomsky*. 
The whole figure represents the *Formal Description*: it is an RDF Graph.

The bordered part is used for the *Final Description*.
Semantic Indexing
Description Tree: Resource Type

- The whole figure represents the *Formal Description*: it is an RDF Graph.
- The bordered part is used for the *Final Description*. 
A Simple Description is a description where the root of the Description Tree has only one child.

- The general form of the part used for the Final Description.
- The blank nodes are virtual instances of concepts.
- The last node is a real individual of a concept defined in the ontology.
Semantic Indexing
Complex description: Description Tree

A Complex Description contains several paths. Each path starts from the root and relates a Simple Description $SDes$ of the same document.
Semantic Indexing
Complex description: definition

Definition
A Complex Description is a Description Tree where the root has more than one child. The tree is the merging of $n$ simple descriptions. A Complex Description $CDes$ is defined by the union of simple descriptions:

$$CDes = SDes_1 \lor SDes_2 \lor \ldots \lor SDes_n$$
Semantic Indexing
Complex description: publication and retrieval

- A resource \( R \) with \( n \) \( SDes \) is published \( n \) times, once with each \( SDes \).
- A query with one of the \( n \) descriptions must answer positively with the resource \( R \).
- A query requesting for resources having a complex description is considered as a set of elementary queries (corresponding at a simple description). The result of the query is the intersection of the elementary query results.

\[
\text{Result}(Q_{\text{CDes}}) = \bigcap_{i=1, p} \text{Result}(Q_{\text{SDes}_i})
\]
A key used in the index is a representation of the semantic description of a resource and is written in a language based on RDF.

The semantic description is an RDF graph (the *Description Tree*)

- That contains blank nodes useless for indexing because they do not contain semantic information necessary for describing a resource.
Semantic Indexing
An example of Description Tree

Very difficult documents.

```
lom:LearningObject
  rdf:type
  _:lo

_lom:has_lomEducational

lom:LomEducationalCategory
  rdf:type
  _:lec

_lom:has_difficulty

lom:Difficulty
  rdf:type
  lom:very_difficult

rdfs:label
  "Very Difficult"
```
Semantic Indexing
A small knowledge base

The description contains the following triples:

```n3
_:lo rdf:type lom:LearningObject .
_:lo lom:has_lomEducational _:lec .
_:lec rdf:type lom:LomEducationalCategory .
_:lec lom:has_difficulty lom:very_difficult .
```

The N3 notation synthesizes the description as follows:

```n3
[ a lom:LearningObject ] lom:has_lomEducational
[a lom:LomEducationalCategory ;
 lom:has_difficulty lom:very_difficult .]
```
Semantic Indexing
Format of the key

Key:
- \{\text{rdf}:\text{type}, \text{lom}:\text{LearningObject}\}
- \{\text{lom}:\text{has\_lomEducational}\}
- \{\text{lom}:\text{has\_difficulty}, \text{lom}:\text{very\_difficult}\}
Users should be able to find a resource with other characteristics than those exactly used for publishing

- The System must also publish a resource with descriptions corresponding to these expected characteristics.

The extension of keys produces a *Complex Description*

- The *Simple Description* supplied by the resource provider is combined with others generated by the system.
- The resource is published with each of them.
Semantic Indexing
Keys extension: subsumption

- *Documents about Stack.*
- The ontology *Theory of Languages* contains the concept *lt:Stack* and the super-concept: *lt:Data_Structure*.
- Any request of resources concerning Data Structure should also return resources concerning Stack.

<table>
<thead>
<tr>
<th>Key_initial:</th>
</tr>
</thead>
<tbody>
<tr>
<td>{rdf:type,system:Document}</td>
</tr>
<tr>
<td>{system:hasInterest, lt:Stack}</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Key_extended:</th>
</tr>
</thead>
<tbody>
<tr>
<td>{rdf:type,system:Document}</td>
</tr>
<tr>
<td>{system:hasInterest, lt:Data_Structure}</td>
</tr>
</tbody>
</table>
Very difficult documents.

A resource may be published with a specific difficulty level (lom:very_difficult).

We consider also interesting to look for resources where the difficulty level has been defined.

A request of resources where the difficulty level has been defined, should also return resources published with a specific difficulty level (instances of the concept lom:Difficulty).

Key_initial:
{rdf:type,lom:LearningObject}
{lom:has_lomEducational}
{lom:has_difficulty,lom:very_difficult}

Key_extended:
{rdf:type,lom:LearningObject}
{lom:has_lomEducational}
{lom:has_difficulty}
Documents about Chomsky.

We consider that if the content of a resource is about a particular author, it is also about the concept of Author.

Key_initial:
  \{\text{rdf:type,system:Document}\}
  \{\text{system:hasInterest, lt:chomsky}\}

Key_extended:
  \{\text{rdf:type,system:Document}\}
  \{\text{system:hasInterest, lt:Author}\}
Documents about "Jeffrey D. Ullman".

The ontology *Theory of Languages* does not contain any individual of the concept *lt:Author* referring to the author "Jeffrey D. Ullman"

- We considered the possibility for the System to create a virtual individual of the concept *lt:Author*
- And let the user enter the string "Jeffrey D. Ullman" as value of its property *lt:hasName*

Key_initial:
{rdf:type,system:Document}
{system:hasInterest,lt:Author}
{lt:hasName,"Jeffrey D. Ullman"^^xsd:string}
Semantic Indexing

Keys extension: keyword (II)

- A resource whose content is about a particular author, is also about the concept of Author (Category extension).

  
  Key_extended:
  
  `{rdf:type,system:Document}
  {system:hasInterest, lt:Author}

- A resource whose content is associated to a string, is also about a keyword (Keyword extension).

  
  Key_extended:
  
  `{rdf:type,system:Document}
  {system:hasKeyword,
   "Jeffrey D. Ullman"^^xsd:string}`
Outline

- Motivation
- Semantic Indexing
- Indexing Patterns
- Community of Users
- Conclusion
- Future Work
### Indexing Patterns

#### Cases of indexing

<table>
<thead>
<tr>
<th>C.T.</th>
<th>N.V.</th>
<th>1 Step</th>
<th>2 Steps</th>
<th>Example</th>
<th>Extension</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>concept</td>
<td>treating of Family.</td>
<td>Subsumption</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>property</td>
<td>about the semantic density of a LO.</td>
<td>Subsumption</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>individual</td>
<td>treating of Chomsky.</td>
<td>Category</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>V.</td>
<td>individual property</td>
<td>treating of Ullman.</td>
<td>Keyword</td>
<td></td>
</tr>
<tr>
<td></td>
<td>≥1 Step</td>
<td>&gt;1 Step</td>
<td>Example</td>
<td>Extension</td>
<td></td>
</tr>
<tr>
<td>R.T.</td>
<td>N.V.</td>
<td>individual</td>
<td>having a known contributor.</td>
<td>Facet</td>
<td></td>
</tr>
<tr>
<td></td>
<td>V.</td>
<td>individual property</td>
<td>having an unknown contributor.</td>
<td>Facet + Keyword</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>K.T.</th>
<th>1 Step</th>
<th>Example</th>
<th>Extension</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>string</td>
<td>about Medieval Italy.</td>
<td></td>
</tr>
</tbody>
</table>

**Legenda:**

- **C.T.:** Content Type
- **R.T.:** Resource Type
- **K.T.:** Keyword Type
- **V.:** Virtual
- **N.V.:** Not Virtual
Indexing Patterns

Objective

- An Indexing Pattern is a model of a case of indexing.
- An Indexing Pattern allows to follow a path within an ontology, defining a sequence of steps
  - At each step, the user interacts only with the necessary part of the ontology. The unnecessary parts are hidden.
- An Indexing Pattern is used
  - For presenting the ontologies to users in a friendly and easy-to-use way
    - Developers can provide a User Interface able to guide the user through the ontologies.
  - For creating the keys of indexing.
Indexing Patterns
Definition

We call Indexing Pattern a triple $IP = (D, P, A)$ where:

- $D$ is a *Description Template*, the generalized description of a resource. It contains some variables that are fixed during the steps followed by users for creating the description.

- $P$ is a *User Process*, the sequence of steps necessary for determining the values of the variables. It is composed of a sequence of assignments involving either SPARQL queries, or other types of user inputs.

- $A$ is an *Algorithm*, the sequence of computations used for creating the keys of indexing.
Indexing Patterns
Description Template: Iterative Pattern

Individual: _:d
Types: <T_0>
loop (k=1,n)
Facts: <P_k> _:i_k
Individual: _:i_k
Types: <T_k>
end loop
<i_v> ← _:i_n
<V> ← <T_n>
Indexing Patterns
User process: Iterative Pattern

\[ \begin{align*}
\emptyset & \leftarrow \text{userOntologyChoice()} \\
T_0 & \leftarrow \text{user(entry_point(\emptyset))} \\
k & \leftarrow 0 \\
i_v & \leftarrow \text{null} \\
\text{repeat} \\
\quad k & \leftarrow k++ \\
\quad S_k & \leftarrow (\gamma_{T_{k-1}}, \emptyset), \text{select } ?p ?r) \\
\quad \text{with } \gamma_{T_{k-1}} & \{ \\
\quad \quad ?p & \text{rdf:type owl:ObjectProperty} . \\
\quad \quad ?p & \text{rdfs:domain } <T_{k-1}> . \\
\quad \quad ?p & \text{rdfs:range } ?r . \} \\
\quad <p_k, T_k> & \leftarrow \text{user(res}(S_k)) \\
S_F & \leftarrow (\gamma_{T_k}, \emptyset), \text{select } ?i) \\
\quad \text{with } \gamma_{T_k} & \{ ?i \text{ rdf:type } <T_k> . \} \\
\quad \text{if (res}(S_F) \neq \emptyset \\
\quad \quad <i_v> & \leftarrow \text{user(res}(S_F)) \\
\text{until } <i_v> \neq \text{null}
\end{align*} \]
Indexing Patterns
User process: Iterative Pattern

\[ \Theta \leftarrow \text{userOntologyChoice()} \]

\[ T_0 \leftarrow \text{user(entry_point(\Theta))} \]

\[ k \leftarrow 0 \]

\[ i_v \leftarrow \text{null} \]

repeat

\[ k \leftarrow k++ \]

\[ \text{repeat} \]

\[ k \leftarrow k++ \]

\[ S_k < T_{k-1} > \leftarrow (\gamma < T_{k-1} >, \{ \Theta \}, \text{select ?p ?r}) \]

\[ \text{with } \gamma < T_{k-1} > = \{ \]

\[ ?p \text{ rdf:type owl:ObjectProperty} . \]

\[ ?p \text{ rdfs:domain } < T_{k-1} > . \]

\[ ?p \text{ rdfs:range } ?r . \} \]

\[ < p_k, T_k > \leftarrow \text{user(res(S_k < T_{k-1} >))} \]

\[ S_F < T_k > \leftarrow (\gamma < T_k >, \{ \Theta \}, \text{select ?i}) \]

\[ \text{with } \gamma < T_k > = \{ ?i \text{ rdf:type } < T_k > . \} \]

\[ \text{if } (\text{res(S_F < T_k >))} \neq \emptyset \]

\[ < i_v > = \text{user(res(S_F < T_k >))} \]

until \[ i_v \neq \text{null} \]
Indexing Patterns
User process: Iterative Pattern

\[ O \leftarrow \text{userOntologyChoice()} \]
\[ T_0 \leftarrow \text{user(entry_point}(O)) \]
\[ k \leftarrow 0 \]
\[ i_v \leftarrow \text{null} \]
repeat
\[ k \leftarrow k++ \]
\[ S_k<T_{k-1}>\leftarrow(\gamma<T_{k-1}>,\{\emptyset\}, \text{select } ?p ?r) \]
with \( \gamma<T_{k-1}> = \{ \)
\[ ?p \text{ rdf:type } \text{owl:ObjectProperty} . \]
\[ ?p \text{ rdfs:domain } <T_{k-1}> . \]
\[ ?p \text{ rdfs:range } ?r . \} \]
\[ <p_k, T_k> \leftarrow \text{user(res}(S_k<T_{k-1}>)) \]
\[ S_F<T_k>\leftarrow(\gamma<T_k>, \{\emptyset\}, \text{select } ?i) \]
with \( \gamma<T_k> = \{ ?i \text{ rdf:type } <T_k> . \}
if (\text{res}(S_F<T_k>) \neq \emptyset)
\[ <i_v> = \text{user(res}(S_F<T_k>)) \]
until \( <i_v> \neq \text{null} \)

\[ <T_0> \leftarrow \text{lom:LearningObject} \]
\[ _d \leftrightarrow \text{rdf:type} \]
\[ <T_1> \leftarrow \text{lom:LomEducationalCategory} \]
\[ <P_1> \leftarrow \text{lom:has_lomEducational} \]
\[ _i1 \leftrightarrow \text{rdf:type} \]

PhD Defense – July 6, 2011
Indexing Patterns
User process: Iterative Pattern

\[ O \leftarrow \text{userOntologyChoice()} \]
\[ T_0 \leftarrow \text{user(entry_point}(O)) \]
\[ k \leftarrow 0 \]
\[ i_v \leftarrow \text{null} \]
repeat
\[ k \leftarrow k++ \]
\[ S_k^{<T_{k-1}>} \leftarrow (\gamma^{<T_{k-1}>}, \{\emptyset\}, \text{select } ?p \ ?r) \]
\[ \text{with } \gamma^{<T_{k-1}>} = \{ \]
\[ ?p \text{ rdf:type owl:ObjectProperty} . \]
\[ ?p \text{ rdfs:domain } <T_{k-1}> . \]
\[ ?p \text{ rdfs:range } ?r . \} \]
\[ <p_k, T_k> \leftarrow \text{user(res}(S_k^{<T_{k-1}>})) \]
\[ S_F^{<T_k>} \leftarrow (\gamma^{<T_k>}, \{\emptyset\}, \text{select } ?i) \]
\[ \text{with } \gamma^{<T_k>} = \{ ?i \text{ rdf:type } <T_k> . \} \]
\[ \text{if } (\text{res}(S_F^{<T_k>})) \neq \emptyset \]
\[ <i_v> = \text{user(res}(S_F^{<T_k>})) \]
until \[ <i_v> \neq \text{null} \]
Outline

- Motivation
- Semantic Indexing
- Indexing Patterns
- Community of Users
- Conclusion
- Future Work
Community of Users
Community and users

A Community is composed of users interested in collaborative activities

- **Expert users**: experts in the domain of interest of the community. They are in charge of the activities of providing the ontologies, their description and their publication.
- **Provider users**: they don’t have an high level role. They usually publish and retrieve resources.
- **Consumer users**: they have a passive participation because they don’t provide any contribution to the community. They just retrieve resources.
Community of Users
Community and resources

- Community resources
  - Documents
    - The resources shared by users through the Shared Memory.
  - Core resources
    - *Ontologies*: used for creating the keys of indexing.
    - *Notes*: free text provided by a user to include additional information in the System.
    - *Wiki*: a unique space of the System shared by all users.
    - Are published in the network with specific keys using the System Ontology.
Community of Users
Core resources: Ontologies

- Are published in the network by expert users with a small additional description:
  - a textual description concerning domain of the ontology;
  - the set of Entry Points.
- The publication is made thanks a key of indexing assigned automatically by the System.
- Are retrieved from the network when the user starts the system.
Community of Users
Core resources: Notes

- The use of Notes is considered of general purpose
  - The content of the Note may be any topic of interest for the user: messages for other users, memos, comments on certain resources, etc.
  - Notes are published using a keyword.
- Notes are published with a key assigned by the System.
Community of Users
Core resources: Wiki

- The Wiki of the Community is composed of only one physical document containing several parts that may link to other resources, distributed in the network
  - Links are *Semantic*, refer to keys of indexing, are embedded in the HTML *link* tag.
- When a new community is created the System publishes the Wiki in the network from a template containing the skeleton with only the essential structure.
Community of Users
Community and tools

A Community is supported by a Web platform equipped with a set of tools

- **Indexing Tool**: is used for choosing the ontologies retrieved from the network and for creating the keys of indexing.

- **Indexing Pool**: is a temporary container of (key, resource) pairs. It allows users to select the resource they want to index and to associate the key of indexing built with the Indexing Tool.

- **Notes Editor**: is a tool that enables users to create personal notes that are associated to keys of indexing and published.

- **Retrieval Tool**: allows users to submit queries to the system. It retrieves results and displays them.
Community of Users
Web Platform
Community of Users
Architecture of a Peer
Outline

- Motivation
- Semantic Indexing
- Indexing Patterns
- Community of Users
- Conclusion
- Future Work
**Conclusion**

- **A model of Indexing Patterns**
  - For transforming a user understandable description to a machine understandable one.

- **Description extension mechanism**
  - Form managing publication and retrieval contexts.
  - The description is enlarged during publication to foresee different possible retrieval situations.

- **A Web platform**
  - That makes feasible the life of a decentralized community.
  - Contains a set of *Core resources* for allowing the indexing.
  - Contains a distributed Semantic Wiki and a distributed system of Notes.
Outline

- Motivation
- Semantic Indexing
- Indexing Patterns
- Community of Users
- Conclusion
- Future Work
Future Work

- Advanced navigation system for ontologies
  - A richer navigation system for ontologies for better organize the visual composition of represented data.

- Exchange with an external system.
  - It may query our system by creating a semantic description of potential resources based on RDF

- Multilingual issues.
  - It concerns resources indexed on keywords or indexed on virtual individuals because the user has to add at least one string in order to describe this individual.

- Evaluation.
  - Experiments should prove that the system can really support a real community.
Publications (I)

- In proceedings
Publications (II)

- Journal

- Book chapters