Visualizing a Temporally – Enhanced Ontology

Katifori Akrivi
Dept. of Informatics and Telcommunications[1],
University of Athens
+302107275241
vivi@di.uoa.gr

Vassilakis Costas
Dept. of Computer Science and Technology[2],
University of Peloponnese
+302710372203
costas@uop.gr

Lepouras Georgios
Dept. of Computer Science and Technology[2],
University of Peloponnese
+302710372201
G.Lepouras@uop.gr

Daradimos Ilias
Dept. of Informatics and Telcommunications[1],
University of Athens
+302107275241
drid@mm.di.uoa.gr

Halatsis Constantin
Dept. of Informatics and Telcommunications[1],
University of Athens
+302107275201
halatsis@di.uoa.gr

ABSTRACT
Most ontology development methodologies and tools for ontology management deal with *ontology snapshots*, i.e. they model and manage only the most recent version of ontologies, which is inadequate for contexts where the history of the ontology is of interest, such as historical archives. This work presents a modeling for entity and relationship timelines in the Protégé tool, complemented with a visualization plug-in, which enables users to examine entity evolution along the timeline.

Categories and subject descriptors
H.3.3 Information Search and Retrieval, H.5.2 User Interfaces – Graphical User Interfaces (GUI), I.3.6 COMPUTER GRAPHICS - Methodology and Techniques – Interaction Techniques.

General Terms
Design.

Keywords and phrases
Temporally enhanced ontology, visualization method, human-computer interaction, entity timeline.

1. INTRODUCTION
The recent progress in the area of digital libraries and the semantic web has lead to new ways of digitizing, organizing and presenting the material and the incorporation of semantics in search mechanisms. A very useful tool to this end is an ontology, which presents an overview of the domain related to a specific area of interest and may be used for browsing and query refinement. Ontologies model classes and relationships in a high level of abstraction, providing rich semantics for humans to work with and the required formalism for computers to perform mechanical processing and reasoning.

According to [3], an ontology is an explicit specification of a conceptualization. The term "conceptualization" is defined as an abstract, simplified view of the world that needs to be represented for some purpose. It contains the objects, concepts and other entities that are presumed to exist in some area of interest and their relations.

Therefore, as defined in [7], an ontology is a formal explicit description of concepts (or classes) in a domain of discourse, attributes of each class describing various features and properties of the class (also called slots), and restrictions on attributes. Each attribute has a type and could have a restricted number of allowed values. An instance of a class has a concrete value for each attribute of the class An ontology together with a set of individual instances of classes constitutes a *knowledge base*.

However, in some contexts, such as that of a Historical Archive, a very important factor is the concept of time. Sometimes the material that the ontology refers to covers a very large time span, which could be a few years, decades or even centuries. This fact results in instances and/or relationships between them that change with the passage of time. In such cases, the ontology should be able to reflect the evolution of the real-world, providing facilities for designating the instants (e.g. Jan 31, 2000 19:37) or time periods (e.g. [Jan 31, 2000-Feb 17, 2000]) [4] for which each represented real-world state is valid.

Most ontology development methodologies and tools for ontology management deal with ontology snapshots, i.e. they model and manage only the most recent version of ontologies, which is inadequate for some contexts - such as that of an historical archive - where the organization ontology will have probably changed in the time period covered by the archive. In this paper, we present a modelling approach for ontologies evolving over time and a visualization tool enabling users to efficiently explore the ontology evolution. The rest of the paper is organized as follows: section 2 presents related work in the areas of temporal ontology modelling and visualization. Section 3 introduces the requirements for temporal ontology vizualization, while section 4 presents the proposed visualization methodology, including some information for the modelling of the ontology where appropriate. Finally, section 5 concludes the paper and outlines future work.

[1] University Campus, Ilissia, Athens, 157 84, Greece

[2] End of Karaiskaki, 22100 , Tripolis, Greece

2. RELATED WORK

There are several recent works relative to the subjects of ontology versioning and evolution. The system PromptDiff [6] has been developed in the context of a collaborative environment for managing ontologies in order to support ontology versioning. Given two versions of an ontology, it allows users to: (1) examine the changes between versions visually; (2) understand the potential effects of changes on applications; and (3) accept or reject changes. PromptViz [11] is a tool providing advanced visualizations using treemaps to help users understand the location, impact, type and extent of changes that have occurred between versions on an ontology. In [5], a different approach for reconciling the different ontology versions with each other is introduced. The presented framework, that aims to provide means for reasoning based on a complete versioning history, includes the generic notion of *change bridge* for describing ontology resource changes, and a basic set of particular change bridge types that constitute the class hierarchy of a change bridge ontology. Ontology changes are represented as instances of the change bridge types relating classes in successive ontology versions with each other. The change bridge ontology is represented using the Resource Description Framework (RDF). In [2] the changes in an ontology are handled as database operations and triggers, whereas in [10] the abstract data type *knowledge base* is introduced which contains a description logic representation and a basic set of operations to work on it. A formalization of operations that change the knowledge base is presented, in order to support the evolution of ontologies. The operations can be extended or changed to satisfy local needs.

Finally, [1] briefly presents requirements for visualizing changes to ontologies and [3] attempts a comparative evaluation of ontology editors concerning the subject of function supporting ontology evolution. A survey of techniques for visualizing differences is included in [11].

3. TEMPORALLY – ENHANCED ONTOLOGY VISUALIZATION REQUIREMENTS

An ontology may evolve along the timeline in the following ways:

1. New instances may be created or existing instances may be deleted. A single instance may also be split into two distinct instances (e.g. Czechoslovakia is split to the Czech Republic and Slovakia) or two instances may be merged into a single one (e.g. East and West Germany are united to form Germany).

2. The value of instance attributes may change. Besides changes of simple values (e.g. an academic's rank could be "Lecturer" for the period [Sep 2001-Mar 2004] and "Assistant Professor" for the period [Apr 2004-now]), this includes changes in relationships between classes (e.g. the relationship "President" between a "Department" and a "Faculty member"), as well as changes in multi-valued attributes.

3. New classes may be added or deleted, or existing ones may be modified by adding or removing attributes, changing their types, changing inheritance relationships between classes, merging or splitting classes, modifying constraints and so forth.

The first two items in the list above effectively include changes in the *content* of the knowledge base, while the third item refers to changes on the *structure* (or *schema*) of the knowledge base. If a system maintains a record of all states of the knowledge base content, then *history support* [4] is provided. A system accommodates *schema evolution* if the schema can be changed without data loss, while *schema versioning* is provided if a record of all schemas is maintained and each (past or current) schema can be used for browsing and querying [12]. In this paper we will limit our discussion to visualizing content evolution only. For the efficient visualization of an ontology that accommodates history support, the following requirements can be identified:

1. Direct identification of instances, attributes and relationships that have evolved over time, as opposed to those that have retained a constant value.

2. For items that have evolved over time, presentation of the evolution timeline. A common query would be that of retrieving a person's biography, i.e. if a certain person was ever student in the University, when the person graduated, when the person became professor or employee in the university, etc. Or, it would be useful to know, in the case of the University of Athens Historical Archive Ontology, when the department of Informatics was created and if it originated from another department.

3. For instances that have split or merged, it should be possible to identify their predecessors or successors.

4. It should be possible to extract a specific *point-in-time* of the ontology, i.e. creation of an ontology which contains the classes, relationships and instances that are valid for a designated time instance. The result of this extraction is a non-temporal ontology. This is similar to focusing on a specific time instant.

5. Facilities for extracting a specific *time period* of the ontology, i.e. creation of an ontology which contains the classes, relationships and instances that are valid for a designated time period. The result of this extraction is a temporal ontology. This is similar to focusing on a specific time period.

6. A holistic view of the timeline that the archive covers should be offered. The system should provide an overview of the time period covered by the ontology with the ability to zoom in and out and select specific sub-periods or time points in order to view the corresponding ontology.

7. Visualization of the *co-evolution* of different instances or attributes should be provided. A common method of analysis for historians is to correlate the evolution of different parameters (e.g. urban population and economic growth), to extract useful conclusions.

4. PROPOSED APPROACH

In order to address the issues concerning the management of a temporal ontology, there is a need for an integrated approach that will incorporate versioning and evolution while enriching the instances with temporal characteristics. The following sections summarize an approach to augmenting an ontology management system with functions related to time.

4.1 Managing Instance Evolution

In the context of a temporal ontology the instances may change the values of some of their attributes. For example, the salary of a secretary or the name of a university department may change at some point. This is also the case with more complex attributes such as the members of a council that may be different in different time periods.

In order to record these changes in the instances, the class of *temporally enhanced attribute* has been introduced. This data type offers a set of pairs of the following form:

<attribute-value1>, <time-period1>

<attribute-value2>, <time-period2>

…

<attribute-valuen>, <time-periodn>

where <attribute-value> is the value of the attribute, either one of the simple data types such as String, Integer or Float or an instance of a class within the ontology and <time-period> is the valid time for this value, which has a starting point and an ending point. There is a constraint defined so as the starting point cannot be greater than the ending point. Temporal attributes are subdivided to *single-valued temporal attributes* and *multivalued temporal attributes.*

Single-valued temporal attributes may have only one value at a given time period – equivalently, the time periods associated with different attribute values may not overlap. For example, the name attribute of a university department may be the following:

Department of Informatics [16/06/1989 – 30/6/2002]

Department of Informatics and Telecommunications [1/7/2002 – today]

The absence of overlapping is enforced by an appropriate constraint.

Multi-valued temporal attributes may have multiple values for a single time point. For example, the members of a council may be defined as follows:

John Black (12/3/2000 - 2/4/2001)

Mary Peterson (5/5/2000 – 5/6/2002)

Sheila White (3/12/2000 – 1/6/2002)

As seen from this example there are periods where more than one person was a member of the council.

For accommodating the evolution of the knowledge base contents, the ontology meta-schema has been modified, to include the following elements:

1. Classes for representing time instants and time periods at different granularities (e.g. year, month, second) and calendars (e.g. Gregorian calendar, academic calendar etc).
2. A template for single-valued temporal attributes. Effectively, this template maps to a multi-valued attribute with pairs of the form (attribute-value, time-period), plus an constraint for non-overlapping of time periods.
3. A template for multi-valued temporal attributes. Effectively, this template maps to a multi-valued attribute with pairs of the form (attribute-value, time-period), without any non-overlapping constraint.
4. A number of attributes have been added to the class template, to allow tracking of instances that have split into multiple ones or multiple instances that have merged to a single one. These attributes are named “split-to” and “merged-from”, with their inverses being “split-from” and “merged-to”, respectively.

4.2 Temporal Ontology Visualization

In order for the user to benefit from all the available meta-data information concerning time in the context of the archive ontology, there is a need for an appropriate visualization tool. This tool could allow the user to navigate in the ontology or select specific entities in order to view their course in time. Protégé [9] was selected as the ontology management tool the functionality of which would be augmented. This is due to a number of advantages it provides related to extensibility. It is developed with Java and its source code available with detailed documentation for creating plug-ins. The plug-in, named “Entity Timeline”, consists of four parts:

An **explorer – like view** of the ontology, namely the one provided by the Protégé Class Browser. Classes with temporal attributes are designated by a small clock next to their description.

An **Instance View** window were all the instances of a selected class are presented. Instances that have evolved along the time axis (any attribute has changed its value) are again designated by a small clock next to their instance icon. The user may tune this indication to appear if any of some selected attributes have evolved. The instance view allows the user to select the portion of the time axis s/he wants to focus on. Note that by changing the time axis portion, instances may disappear (when their lifeline ceases to intersect with the displayed portion) or appear (if their lifeline did not formerly intersect with the selected time axis portion, but now does). The user may tune the visualization to grey-out instances that “do not exist” in the selected time axis portion, rather than totally hide them.

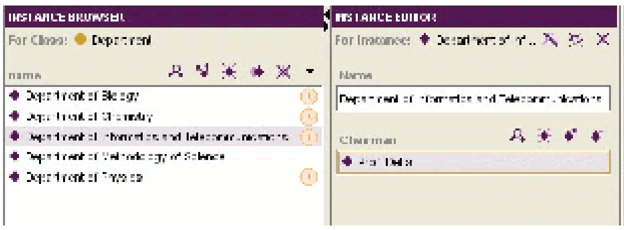

Figure 1. The Instance View with clocks next to the instances that have changed during the selected time period.

A **Timeline** that is visualized as an horizontally placed bar at the top of the main visualization window.

The **Main Visualization** window which offers several options. The user may select time periods or time points from the Timeline and view the ontology (or ontologies) that correspond to this period. When more than one versions are presented, the Main window is divided in the appropriate number of parts for their representation and the parts that have changed in a version are highlighted.

Furthermore, the user may select a class or instance and view its course over the selected period. This is accomplished with the Entity Timeline visualization. This visualization is somewhat similar to the Lifelines [8] visualization used to represent information related to a specific person along a time axis but it has been adapted to the special characteristics of an ontology. The user may select which attributes of the selected class or instance s/he wishes to investigate on the selected time-period. The simple type non-temporal attributes like names, dates of birth, etc are displayed on the Entity Timeline. Temporal attributes of type "Instance" or "Class" are links to their information.

This is presented in the following example. The user has chosen to visualize the department of Informatics and Telecommunications for the time period 1980–2005 and has checked the attributes "Name", "Establishment-Date" and "Chairman" in the appropriate Attribute Panel which appears when clicking on "Select Attributes". The requested information is visualized in Figure 2.

On the upper part the Timeline is presented. The user may perform actions with it like selecting time points and periods by clicking on the Timeline. His/her selections are highlighted. When needed the Timeline expands or retracts accordingly. The user may also select an instance or class and view its lifeline, as in the figure.

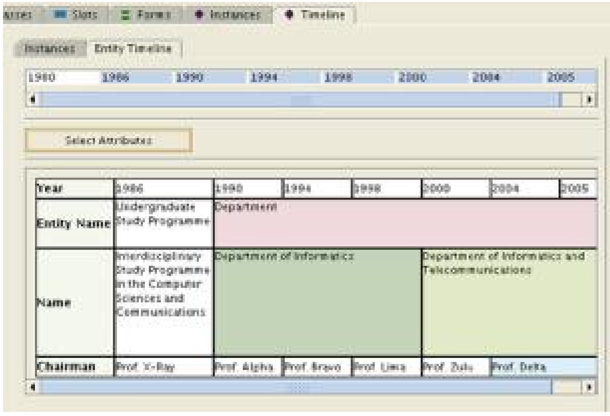

Figure 2. Entity Timeline plug – in Protégé.

The upper part of the table is a timeline of dates (in this example years) when changes to the selected entity (class or instance) occurred, within the time period selected by the user. The second row is the entity name and the following ones contain the selected attribute values. In this case, the user may see that the department of our example started in 1985 as an Undergraduate Studies Program and became a Department in 1990 named "Department of Informatics". S/he may also see that its name changed to "Department of Informatics and Telecommunications" in 2000. The names of the professors that served as chairmen during this period are also visible.

Besides the visualization tabs described above, two more operations are available from the menus, allowing the user to extract a specific point-in-time of the ontology and a specific time period of the ontology, respectively. The result of the former operation is a standard "snapshot ontology" which may be visualized using any standard Protégé plugin, whereas the result of the latter operation is a temporal ontology, on which any visualization method described above may be applied.

5. CONCLUSIONS AND FUTURE WORK

This work presents an environment for the definition and visualization of different ontology versions and changes in instances that cover a specific time span. It is implemented as a Protégé plug-in. The current version does not contain the implementation of a query mechanism supporting complex queries. As a future step, this mechanism will be implemented. Furthermore, a thorough evaluation of the system will be carried out, in order to investigate its efficiency. Incorporation of ontology evolution and ontology versioning capabilities will also be considered.

6. REFERENCES

[1] Blundell, B., and Pettifer, S., Abstract: Graph Visualization to Aid Ontology Evolution in Protégé, 7th nternational Protégé Conference, 6th - 9th, July 2004, Bethesda, Maryland, http://protege.stanford.edu/conference/2004/

[2] Ceravolo, P., Corallo, A., Elia, G. and Zilli, A., Managing Ontology Evolution Via Relational Constraints, Eighth International Conference on Knowledge-Based Intelligent Information & Engineering Systems (KES'04), Wellington, New Zealand, September 2004.

[3] Gruber, T. R., A Translation Approach to Portable Ontology Specifications, Knowledge Acquisition, Special issue: Current issues in knowledge modelling, Vol 5, Issue 2, pp. 199-220, 1993

[4] Jensen, C. S. et al., The Consensus Glossary of Temporal Database Classs, Technical Report, Department of Computer Science, Aalborg University, available at http://www.cs.auc.dk/~csj/Glossary/

[5] Kauppinen, T., and Hyvönen, E., Bridging the Semantic Gap between Ontology Versions, Web Intelligence Symposium, Finnish AI Conference, 20 year Anniversary (SteP 2004), Heureka - the Finnish Science Centre, Vantaa,September 2-3, 2004

[6] Noy, N. F., Kunnatur, S., Klein, M., and Musen, M. A., Tracking Changes During Ontology Evolution, Third International Conference on the Semantic Web (ISWC-2004), Hisroshima, Japan

[7] Noy, N. F., and McGuiness, D. L.,Ontology Development 101: A Guide to Creating Your First Ontology, Stanford Knowledge Systems Laboratory Technical Report KSL-01-05 and Stanford Medical Informatics Technical Report SMI-2001-0880, March 2001

[8] Plaisant, C., Milash, B., Rose, A., Wido, S., and Schneiderman, B., LifeLines: Visualizing Personal Histories. In Proc. of the ACM CHI Conference, Vancouver, Canada, April 1996

[9] Protégé, http://protege.stanford.edu/

[10] Sindt, T., Formal Operations for Ontology Evolution, Proceedings of the International Conference on Emerging Technologies, Minneapolis, Minnesota, August 25 – 26, 2003

[11] Steven, D., and Perrin, J., PROMPT-Viz: Ontology Version Comparison Visualizations with Treemaps, Master Of Science Thesis in the Department of Computer Science, University of Victoria, 2004. Retrieved from http://www.cs.uvic.ca/~chisel/thesis/David_Perrin_Thesis.pdf

[12] Völkel, M., Enguix ,C. F., Ryszard Kruk, S., Zhdanova, A. V. Stevens, R., and Sure, Y., SemVersion - Versioning RDF and Ontologies KnowledgeWeb Deliverable D2.3.3.v1, Institute AIFB, University of Karlsruhe, June 2005