University of Peloponnese

Department of Computer Science and Technology

Software and Database Systems Laboratory

wt-Protégé – An Extension for Protégé Supporting Temporal and Weighted Data

Technical Report TR-SSDBL-07-001

Costas Vassilakis, George Lepouras, Akrivi Katifori

costas@uop.gr, gl@uop.gr, katifori@uop.gr

September, 2007

Tripoli, Greece

Table of Contents

1. Introduction

In the past few years, ontologies have emerged as a valuable tool for representing the semantic context of a domain; ontologies provide the required formalism for computers to perform automated reasoning and the high level semantics for humans to work with. A number of tools have emerged that facilitate the creation, maintenance, usage and visualization of ontologies, such as Protégé, KAON, etc. Insofar, however, ontology tools are targeted to maintaining *ontology snapshots*, i.e. the most recent version of ontologies; past values of properties or relationships are not maintained, neither the period that the values/relationships were in effect can be stored. Some tools encompass ontology versions, which though cannot be effectively used for maintaining entity evolution since, generally, the number of changes in properties and/or relationships is too high, resulting in an excessive and unmanageable number of version. Moreover, facilities for entering weighted information are very useful for a class of environments and applications, such as uncertain and fuzzy data management. Current provisions for entering such data suffer from the same drawbacks listed for historical data above.

This work is an extension of the Protégé tool to accommodate the modeling and presentation of

- entity history, i.e. past values of properties and/relationships; each such value is timestamped with the period that it was (or will be) valid in the real world.

- weighted data, i.e. data where each value is associated with a real number, its weight. This can serve as a degree of confidence for fuzzy data or for any other application-defined purpose.

To this end, the presented extension provides:

1. integrated support for data types expressing time quantities – more specifically dates (individual points in time) and periods (anchored segments of the time axis).

2. data types for storing histories of properties of different types (strings, integers, floats, booleans and instances [i.e. relationships]).

3. data types for storing weighted properties of different types (strings, integers, floats, booleans and instances [i.e. relationships]).

The extension can be used in contexts that the modeling of entities' history is important, such as historical archives, museums, etc as well as in contexts where storing value weights is important.

2. Downloading and Installing

For installing the extension you need to perform the following steps:

1. Download Protégé Version 3.1.1 from the Protégé site (http://protege.stanford.edu)

2. Install the Protégé software following the instructions provided by the Protégé team

3. Download the wt-protege.zip file; this is the compiled, ready-to-run version of the extension. URL http://sdbs.cst.uop.gr/files/wt-protege.zip

4. Unzip the downloaded wt-protege.zip file in the Protégé installation directory replacing the protege.jar and protege_text.properties files therein (in Windows-based systems, the default location is C:\Program Files\Protege_3.1).

You may now run Protégé as usual, and the extended version will be loaded.

3. Available extensions

The extended version of Protégé encompasses support for:

1. data types expressing time quantities – more specifically dates (individual points in time) and periods (anchored segments of the time axis).

2. data types for storing histories of properties of different types (strings, integers, floats, booleans and instances [i.e. relationships]).

3. data types for storing weighted properties of different types (strings, integers, floats, booleans and instances [i.e. relationships]).

These extensions are presented in the following paragraphs.

3.1 The "Date" Data type

The "Date" data type provides support for expressing individual points in time. The precision with which the time point may be specified varies from year-level to the level of a second. Dates are given and displayed in the ISO-standard format, i.e. YYYY-MM-DD hh:mm:ss (YYYY = year in the range 0000-9999, MM = arithmetic month of the Gregorian calendar in the range 1-12, DD = day within the month in the range 1-31, hh = hour in the range 0-23, mm = minute in the range 0-59 and ss = second in the range 0-59). The following examples illustrate acceptable values for a date slot:

- 2006
- 2006-03
- 2006-03-25
- 2006-03-25 10
- 2006-03-25 10:58
- 2006-03-25 10:58:43

The precision used to denote the time quantity will be termed as *granularity* in this document. Note that the year is mandatory. Two more values are acceptable as dates, more specifically the strings *unknown* and *now*. The *unknown* value can be used to signify that the date is not known, while the *now* value should be interpreted as always being equal to the current "wall clock" indication.

3.2 The "Period" Data type

The "Period" data type provides support for expressing anchored segments of the time axis, i.e. time intervals with a specific beginning and a specific end. A period is entered as a pair of dates, enclosed in square brackets ([]) and separated with a comma. The dates may be expressed in different granularities, but it is required that the starting date should be less or equal to the end date. The following examples illustrate acceptable values for a period slot:

- [2002, 2006]
- [2002-01, 2002-01]
- [2002-01-18 10:32:11, 2006-03-25 10:58:43]
- [2002-01-18 10:32:11, 2006-03]

Contrary, the following examples illustrate unacceptable values for a period slot:

- [2006-03-25, 2002-01-18] (ending date is before the starting date)
- [2006, 2006-10-01] (starting date represents the whole of the year 2006, which includes dates that are after the ending date).

Semantically, a period is considered to include both the starting and the ending time point, e.g. the period [2002-01, 2002-02-12] is considered to include the whole of January 2002 and the twelve first dates of February. This is particularly important for the notion of overlapping periods used in the context of non-multiple temporal types (discussed in the next paragraphs). According to the adopted semantics, periods [2002-01, 2002-02-12] and [2002-02-12, 2002-03] *do* overlap (since they both include February 12, 2002), while periods [2002-01, 2002-02-12] and [2002-02-13, 2002-03] *do not* overlap.

The special date values *unknown* and *now* may be used for the starting or the ending date of a period (or both). In such a case, no check is performed that the starting date should be before the ending date.

3.3 The "t-String" Data Type

The "t-String" data type allows for entering string-typed values that retain their evolution through time. For example, consider the case that we want to retain history of the addresses at which some organization is installed; for this purpose we would use a *t-String* typed slot named "Address". In the instance editing window, the slot would be rendered as standard Protégé list box, and we would use the "Add Value" button to enter the proper values for the organization whereabouts, each one tagged with a period expressing *when* the organization was installed at the specific location, e.g.:

[2000-01-30, 2004-12-06]	Somewhere, Someplace 23, 12345, Neverland
[2004-12-07, 2006-03-08]	Elsewhere, Otherplace 42, 34567, Neverland
[2006-04, now]	Anywhere, Anyplace 4, 67890, Neverland

Note that it is allowed to use mixed granularities across periods timestamping different values (i.e. the first two rows use day-level granularity, while the third one employs month-level); the use of "now" and "unknown" is also permitted.

The semantics for the "multiple" check box in the slot property window are modified for the t-String data type as follows:

1. if the "multiple" check box is not checked, then it is not allowed for any two list entries to have overlapping timestamps i.e. *not multiple* means that *for any*

given point in time, at most one value is allowed. If entries with overlapping timestamps are detected in the list, the slot value is considered erroneous and the list is highlighted with a red border, while the tooltip text is set to indicate the offending periods.

2. if the "multiple" check box *is* checked, then the timestamp overlap check is not performed, effectively allowing any number of list entries to have overlapping timestamps. This may be used to model cases that multiple values for a single point in time *are* allowed in the real world, such as organizations with multiple installations (e.g. headquarters, agencies, warehouses etc), lists of professors serving in a University department and so forth.

Under these definitions, the value list depicted in the following table *is not valid* for a t-String slot characterized as "non-multiple" because the second and third rows have overlapping timestamps; the same value list *is* valid for a t-String slot characterized as "multiple".

[2000-01-30, 2004-12-06]	Somewhere, Someplace 23, 12345, Neverland
[2004-12-07, 2006-03-08]	Elsewhere, Otherplace 42, 34567, Neverland
[2005-11, 2006-04-18 10:11]	Wherever, Everyplace 88, 98765, Neverland
[2006-04, now]	Anywhere, Anyplace 4, 67890, Neverland

No overlap check is performed for timestamps using the "now" and "unknown" date representations at either end.

3.4 The "t-Boolean", "t-Instance", "t-Integer" and "t-Float" Data types

These data types are similar regarding the semantics and operation with the *t-String* data type, differing only in the type of the data that may be entered. Booleans, in particular, can be entered as "true" or "false", while instances can be picked from the standard Protégé "Select Instance" window.

3.5 The "w-String" Data Type

The "w-String" data type allows for entering string-typed values that are associated with a weight. For example, consider the case that we want to model the fact that an artifact is of the Protocycladic era with a confidence of 0.8 and of the Mesocycladic era with a confidence of 0.2. For this case we would use a weighted string slot, named era, checking the "multiple" option in the slot properties window. Subsequently, in the instance editing window, the slot would be rendered as standard Protégé list box, and we would use the "Add Value" button to enter the proper values for the artefact's era, each one tagged with a real number representing our degree of belief that the artifact belongs to the specific era, e.g.:

Weight	**Value**
20	Mesocycladic
80	Protocycladic

Note that no norm is imposed on the weight value, and the values 0.80 and 0.20 could be used as well. The weighted string list box provides an additional button to perform *weight normalization*, i.e. translate all weights to the range [0, 1]. During this process,

list rows equal values are also merged into a single row – for example, if the list contents were

Weight	Value
10	Protocycladic
20	Mesocycladic
70	Protocycladic

before the normalization (note that the 1st and 3rd row have equal values), then **after** the normalization the list contents will become:

Weight	Value
0.2	Mesocycladic
0.8	Protocycladic

In the "multiple" check box is not selected, then a weighted string slot is rendered as two labeled input areas, where the user may enter the weight and the value of the weighted string, respectively.

In all cases, it is not allowed to enter a value without entering a weight. Weights are non-negative real numbers. The value of 0 is allowed to be used as a weight, and can be used as a placeholder for "unknown weights".

3.6 The "w-Boolean", "w-Instance", "w-Integer" and "w-Float" Data types

These data types are similar regarding the semantics and operation with the *w-String* data type, differing only in the type of the data that may be entered. Booleans, in particular, can be entered as "true" or "false", while instances can be picked from the standard Protégé "Select Instance" window.

4. Notes and known issues

1. Ontologies using any of the extensions can be opened only by installations that use the wt-Protégé enhancements. Thus, if you send such an ontology to another user or transfer it to another computer, make sure that the extended software is also available, or it will not be possible to load the ontology in Protégé.

2. The window allowing for creating, viewing and modifying timestamped and weighted instances does not use the standard Protégé instance handling widget.

3. The current support for the "now" date is quite limited; better support for displaying these dates and for performing validation checks should be provided.

4. February 29 is accepted for any year, not just leap ones.

5. Source code implementing the extensions is in need of some cleanup, documentation and beautification.

Please report bugs and suggestions to tprotege@uop.gr