



**University of Peloponnese**

**Department of Computer Science and Technology**

**Software and Database Systems Laboratory**

t-Protégé – A Temporal Extension for Protégé

Technical Report TR-SSDBL-06-001

Costas Vassilakis, George Lepouras, Akrivi Katifori

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

June, 2006

Tripoli, Greece

# Table of Contents

<i>Table of Contents</i>	<i>1</i>
<i>1. Introduction</i>	<i>3</i>
<i>2. Downloading and Installing</i>	<i>4</i>
<i>3. Available extensions</i>	<i>5</i>
3.1 The “Date” Data type	5
3.2 The “Period” Data type	5
3.3 The “t-String” Data Type	6
3.4 The “t-Boolean”, “t-Instance”, “t-Integer” and “t-Float” Data types	7
<i>4. Notes and known issues</i>	<i>8</i>

# 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.

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. 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]).

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

## 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 protege.zip file; this is the compiled, ready-to-run version of the extension.
  - Primary download site: <http://sdbcs.cst.uop.gr/files/protege.zip>
  - Alternative download site (better speed): <http://oceanis.mm.di.uoa.gr/pened/protege.zip>
4. Unzip the downloaded 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.

You may also download the source code (original Protégé code plus the modifications and additions to implement the extension) from the following locations:

- Primary download site: [http://sdbcs.cst.uop.gr/files/protege\\_src.zip](http://sdbcs.cst.uop.gr/files/protege_src.zip)
- Alternative download site (better speed): [http://oceanis.mm.di.uoa.gr/pened/protege\\_src.zip](http://oceanis.mm.di.uoa.gr/pened/protege_src.zip)

### 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]).

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.

## 4. Notes and known issues

1. Ontologies using any of the extensions can be opened only by installations that use the t-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 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](mailto:tprotege@uop.gr)