Extensible Support for Measurement Units

Simon White, Catalysoft Ltd., UK

Introduction

All too often, Information Systems represent quantities by *assuming* a particular unit of measurement, rather than explicitly representing it. The quantity might be represented simply as a floating point number in the programming language of choice, with the units of measurement added as canned text to the user-interface. With this approach it becomes difficult, if not impossible, to automatically convert values from one unit to another or to reliably compare values originating from different information sources. In the age of the Semantic Web, we cannot afford to be so remiss.

Information Systems and the ontologies that specify their concepts should explicitly represent units, so that 'unitized values' can be machine-processed for conversion and comparison. We believe the representation should also be extensible so that units can easily be inserted or modified without the need to rebuild the system from source code.

We present our plug-in Protégé Units, which is *reusable*, *flexible* and *extensible*. It is reusable because it extends the system classes of Protégé Frames and provides a domain-independent abstraction upon which other ontologies can be built. It is flexible because the concepts used to support measurement units are represented as a mini-ontology within Protégé's own framework, rather than being hidden away in Protégé's application code. Therefore the relationships between concepts in the user's own ontology and units can be explicitly represented. It is extensible because it enables users to insert both units and dimensions at run-time, as well as to configure their unit conversion factors. This allows users to specify unusual or custom measures, such as the foaminess of a detergent or the peatiness of a whisky.

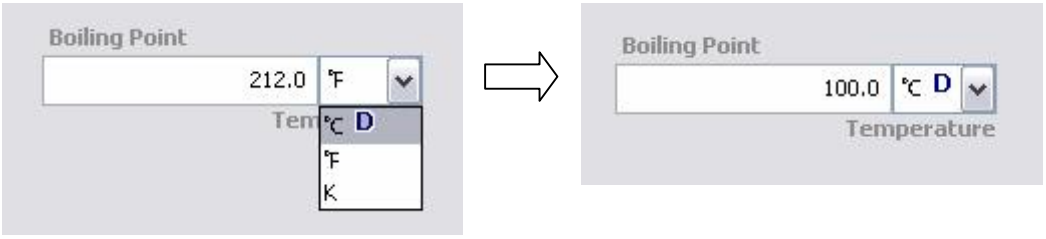

Fig. 1: Changing the units of a slot belonging to the 'Temperature' dimension

Figure 1 shows the use of the slot widget to change the unit associated with the slot value of an instance. The unit conversion is performed automatically based on conversion factors retrieved from the unit extension, and the choice of unit is stored with the instance.

Conceptual Framework

We call a quantity that has been assigned a unit of measurement a `Unitized Number`. Examples of unitized numbers are 42 kilometres, 21.2 °C, and 100 kiloWatts. The unit of a unitized number has a `Dimension` — our three examples illustrate the dimensions of distance (or length), temperature and power. A dimension states *what* we are measuring, whereas a `Unit` represents a *precise amount* of that quantity. Many different units can be used to measure the same dimension; for example, nanometres, inches, kilometres, nautical miles and light years all measure distance. Yet despite the common dimension of distance, this list also demonstrates diversity, both in scale as well as origin and/or context of usage. To accommodate this diversity we also provide the ability to modularise units into meaningful `Unit Groups`. Using this facility, a system can be constrained to present only metric units, if desired, or perhaps simply to reduce the number of possible units to a manageable number, for example, by removing 'light year' as a possible selection in the context of a molecular modelling application.

To facilitate unit conversion, every dimension has a `Standard Unit`. As the standard unit for distance is metre, a conversion from inches to kilometres would use two conversion factors: one for the conversion from inches to metres and another from metres to kilometres. This two-stage approach eliminates the need to store a conversion factor for every possible conversion. Figure 2, below, shows an outline of the conceptual framework.

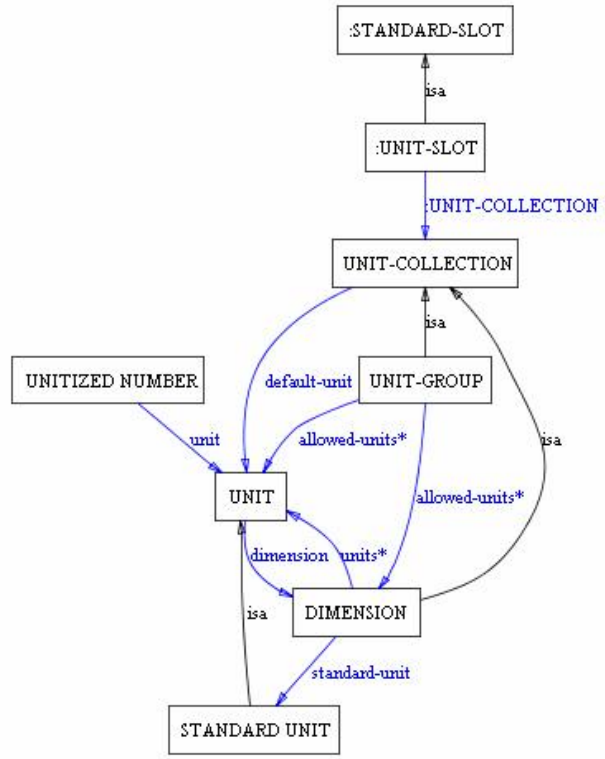

Figure 2: Outline of the Conceptual Framework (diagram generated by the OntoViz plugin)

Units and Protégé

To add unit support to Protégé, we have introduced a new slot type that is a subtype of `:STANDARD-SLOT`, called `:UNIT-SLOT`. Using this new meta-class, we can create a new slot with unit support, say `boiling point`, by setting its slot meta-class to `:UNIT-SLOT`. This extended slot class enables us to associate a dimension (or unit group) to the slot, so for our `boiling point` example, we would specify the dimension of `temperature`. When we use the boiling point slot, we set its value to a type of `UNITIZED NUMBER` so that it carries both a numeric value and a unit, and, in the form design we elect to use the `UnitWidget` for the slot so that we see not only a text field for the input of a number, but also a drop-down for the selection of a unit from the relevant set of units.

To facilitate user input, both dimensions and unit groups also specify a `default unit` (signified by a 'D' icon in the user-interface), which may be different from the standard unit. For example, in specifying the temperature dimension, we might choose Celsius as the default unit and Kelvin as the standard unit.

Catalysoft Protégé Units comprises the following:

- an ontological model of measurement units, their dimensions and the relationships among them;
- a set of commonly-used dimensions and their units;
- a slot widget with a drop-down for the selection of a unit;
- a tab widget to support the management of dimensions and their units;
- a user manual.

A license for Catalysoft Protégé Units will be available at the conference for a modest fee.