

Template Editor User Guide

Version 3.4

Introduction

The DTS Template Editor package provides facilities for the creation and execution of DTS Editor Templates: structured forms for the entry of DTS Concept and Term information. A Template data entry panel gives an alternate, and more structured, way for Concepts/Terms and Concept/Term information to be created and edited compared to the existing DTS Editor Concept/Terms Detail panel. The Template Editor Module contains a Template execution component for selecting and running Templates and a Template builder component for creating and editing the Templates which drive the form-based data entry. The builder component also supports the “publishing” of Templates as independent DTS Editor Modules.

Installation

Extract the files in `TemplateEditor-3.4.zip` into your *DTSInstall* directory. Be sure the `Use folder names` box is checked. This will place all the Template Editor files into the appropriate folders:

<u>Folder</u>	<u>Files</u>
<i>DTSInstall</i> \bin\templateeditor	TemplateEditor.bat
<i>DTSInstall</i> \lib\modules	templateeditor.jar templatemodule.jar pluginutils.jar xmldigester.jar tools.jar
<i>DTSInstall</i> \docs	templateeditoruserguide.pdf
<i>DTSInstall</i> \docs\help	templateeditoruserguide.htm

As desired, edit `TemplateEditor.bat`, which runs the TemplateEditor as a standalone application, to accommodate your particular DTS environment.

Operation

A Template Editor *Template* is a specification for a data entry “form” that can be used to review and update selected DTS Concepts or Terms. The form is made available as part of a Template Editor *entry panel* which consists of a set of *fields*; each field corresponding to an *attribute* of a DTS Concept or Term. Supported attributes include Name, Code, Id, Kind (display only), Primitive/Defined, Defining Concepts, Defining Roles, Synonyms (Concept Templates only), Properties, Associations, and Qualifiers. (Kind, Primitive/Defined, Defining Concept and Defining Role attributes are only available in Templates associated with Ontylog Extension Namespaces). Each field, in turn, has a number of *parameters* which determine the way the field’s *value* is presented, selected, and validated. The Template itself is created by the Template Editor’s *build panel*.

Both entry and build panels are available within the DTS Editor and from the standalone TemplateEditor application (accessed via TemplateEditor.bat). The only difference between the two platforms is that drag-and-drop of Concept/Term Names from DTS Editor panels to Template Editor fields is available in the former. The following sections describe the operation of the components of the Template Editor package.

Template Entry

Figure 1 below shows an example of a Template Editor entry panel. The panel can be accessed from the Template Editor tab in the right DTS Editor tab panel, from the DTS Editor Tools menu as a floating panel, or from the Template Editor tab in the standalone TemplateEditor application (as shown in Figure 1). The components of the entry panel are the Template Select Area and the Template Entry Area which are separated by a split-pane slider. Clicking on the up arrow in the slider hides the Template Select Area increasing the screen space available to the Template Entry area.

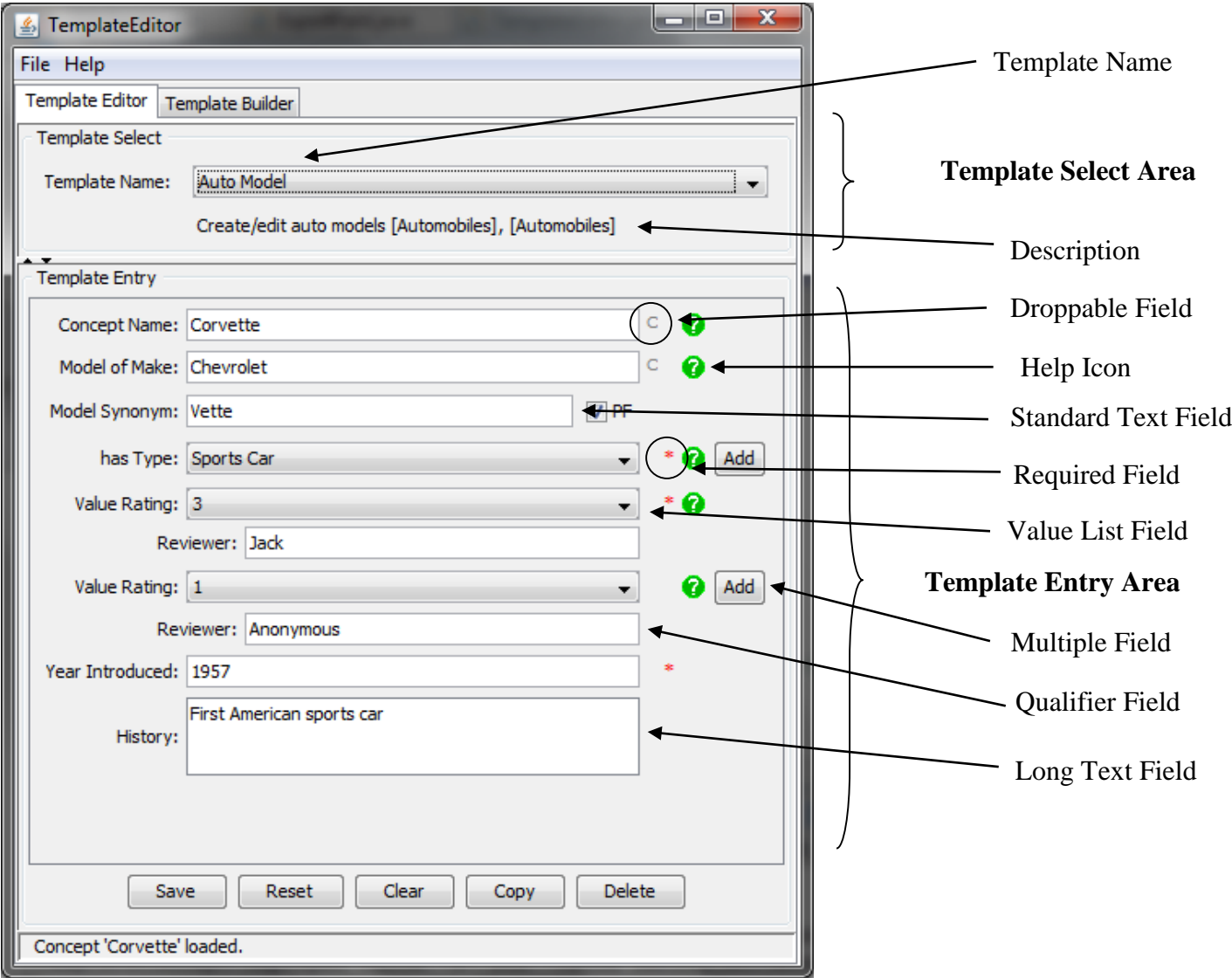


Figure 1 – Template Editor Entry Panel

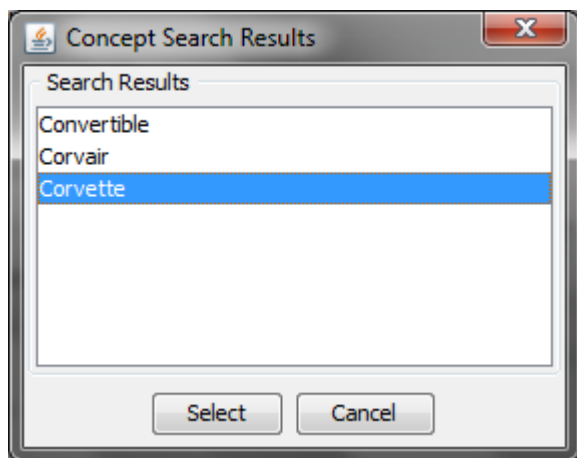
Template Selection

To load an existing Template into the entry area, select the Template's name from the `Template Name` combo box. The selected Template will be loaded, and filled with any default attribute values. The DTS Namespaces associated with the selected Template (`Concept/Term Namespace` and `Update Namespace`) are shown next to the Template name. An asterisk is appended to the Namespace if the Template is `Edit Only` (see below).

Data Entry

Begin using the Template by entering a name in the `Concept Name` (or `Term Name`) field. (Note that this field may have a different label if so designated in the Template specification. This will, however, always be the first field in the Template.) If the name corresponds to an existing Concept/Term in the Template's `Concept/Term Namespace`, the current values of Template attributes are loaded into their associated fields.

If the name does not correspond to an existing Concept/Term, a wild-card search in the Template's `Concept/Term Namespace` is attempted by appending an asterisk ("`*`") to the end of the name. (If desired, an asterisk can be included in the entered name to initiate an immediate search.) Resulting matches are displayed in a `Search Results` panel as shown below. If no match is selected, and this is not an `Edit Only` Template, an option to create a new Concept/Term with the entered name is offered. (See the **Template Creation** section below for further information on the `Edit Only` parameter.)



Upon loading, multiple instances of an attribute will result in the display of multiple occurrences of the associated field, displayed in alphabetic order by the attribute values. Concept/Term attributes which do not have associated Template fields will not be displayed.

Template fields are of three types: standard text, long text, and value list. Standard text fields provide a single line entry field. Data values may not contain `Tab` or `Enter`. Long text fields provide an extended text area for the value. Automatic text wrapping is supported in the area (which does not create line breaks in the value), and both `Tab` and `Enter` can be included if desired.

Value list fields are represented by non-editable combo boxes. Only one of the value list items from the associated drop-down box can be selected. Examples of these three fields are shown in **Figure 1**.

Field values are entered and edited using standard Windows mouse and keyboard procedures. New values in standard text fields **MUST** be followed by the `Tab` or `Enter` keys to be recognized by the panel. Both `Tab` and `Enter` move the focus to the next data entry field if the new value is valid (see **Data Validation** below), unless the field is long text. Values in long text fields are recognized when the cursor is moved to another field.

Grayed fields are display-only fields and may not be modified. See the **Attribute Detail Panel** section below for further information on the `Display Only` parameter.

When running in the DTS Editor, the `Concept Name` (or `Term Name`) field and fields associated with Defining Concepts, Defining Roles, Concept/Term Associations and Inverse Concept/Term Associations support drag-and-drop from other DTS Editor panels. (`Concept Name` and `Term Name` drops are only supported from the Template's designated Concept/Term Namespace.) Droppable fields (those that require Concept/Term values and thus support drag-and-drop) are marked with a small "C" (for Concept) or "T" (for Term) to the right of the field's entry area (see **Figure 1**). These marks are displayed whether or not the Template Editor is running as a panel in the DTS Editor.

Droppable fields can be defined so that allowable Concept values must come from Concepts participating in a specific DTS Subset. In this case, a small "S" is displayed to the right of the field's entry area. The Template Editor provides a default Help message for the field which gives the name of the Subset.

Like the Name field, droppable fields support partial name entry. If an exact match for an entry value in the Template's Concept/Term Namespace is not found, a wild card search is performed in the appropriate Concept/Term context: all Namespaces for Associations, and the Template and base Namespaces for Defining Concepts and Defining Roles. Selected Concepts/Terms from Namespaces other than the Template's Concept/Term Namespace will be post-fixed with the Namespace Name: `MyConcept [SNOMED CT]`. If desired, entered values can contain post-fixed Namespaces in this format.

Attribute Qualifiers are supported on Property and Association fields. Qualifiers are indented in the entry area (see **Figure 1**).

Templates can permit the entry of multiple instances of an attribute. (Existing multiple instances of attributes are always displayed.) Fields that allow the entry of multiple instances have an `Add` button at the far right of last field line for the attribute (see **Figure 1**). To create a new instance of the field, click on the button. (Adding a new attribute also adds the modifiers associated with the attribute in the Template.) Note that adding a new field does not necessarily mean that a new Concept/Term attribute will be created. The attribute will only be created if the field has a non-blank value when the `Save` button for the Template is pressed (see **Entry Panel Buttons** below).

Existing attributes can be deleted by clearing their associated field.

Changing the Concept/Term Name Value

After Concept/Term data has been loaded into the Template, additional behaviors are available when the value in the Name field is changed. The default behavior is to look-up the newly entered name. If there are no matching Concepts/Terms, or `Cancel` is selected from the `Search Results` panel, three results are possible depending on the parameters of the Template (see the **Template Creation** section below for details on these parameters):

1. On a Template whose `Concept/Term Namespace` and `Update Namespace` are different, the change is rejected.
2. On an `Edit Only` Template when the `Concept/Term Namespace` and `Update Namespace` are the same, a dialog is shown asking if the Name of the Concept/Term should be changed. If this option is accepted, the new name is displayed, but the Concept/Term is not renamed until the

Save button is pressed. (This option is not available if the Name field has been designated as Display Only in the Template definition.)

3. On a Template that is not Edit Only and whose Name field has not been designated Display Only, a dialog is shown asking if a name change is desired or a new Concept/Term should be created. If the latter is chosen, the Template is cleared. To create a new Concept/Term with the same field values as the current Concept/Term, use the Copy button described below.

Data Validation

Validation is performed at both the field and Concept/Term levels. At the field level, new values can be validated through the use of standard field validation parameters: regular expressions (RegExs), integer range specifications, and Namespace, Parent or Subset participation. Custom field level validation is also available using user-written Template Validator Java Classes.

Field validation occurs immediately after entry of a new value. If the field requires a Concept/Term value, i.e., it is droppable, existence of the entered Concept/Term is checked, then Subset participation (if specified) is tested. The new value is next validated against any RegEx (if present), then any integer range (if present). Finally it is passed to the Validator Class (if such a Class is specified in the Template) which can accept the value, reformat the value, or reject the value. If a field value fails any of these validations, an error message is displayed and the previous value is restored.

To assist in correct field entry, the field can have a Help message that provides additional information on the desired value. The message can be viewed (as a Tool Tip) by moving the cursor over the green help icon (question mark) on the field line (see **Figure 1**).

When the Save button is pressed upon completion of field entry, all required fields (those marked with a red asterisk – see **Figure 1**) are checked for non-blank values. An error message is displayed if a required value is not present. Then, if a Template Validator Class has been specified, the new Concept/Term object is passed to the Class for final “global” validation. The Class can accept or reject the Concept/Term. An accepted Concept/Term is finally written to the DTS Knowledgebase.

Appendix A gives detailed instructions on writing a Template Validator Class.

Entry Panel Buttons

There are five buttons at the bottom of the entry panel: Save, Reset, Clear, Copy and Delete.

Save	Writes the Template data into the DTS Knowledgebase. Before saving, all required fields are checked for non-blank values and, if a Template Validator Class has been specified, the Concept/Term is validated.
------	--

Reset	Restores the Template data to its most recent saved, or initial, state. If the designated Concept/Term exists, the fields are reloaded from the DTS Knowledgebase. If the named Concept/Term has not yet been saved, the fields are reset to their Template default values.
-------	---

Clear	Sets all fields to their Template default (or blank) values, including the initial Concept/Term name field.
Copy	Copies the current Template field values into a new Concept/Term. A dialog requesting the new Concept/Term Name (and Code and Id if these are requested in the Template definition) is displayed. Note that the Name (and optionally Code and Id) values must be unique in their Namespace. The Copy button is disabled on Edit Only Templates.
Delete	Deletes the specified Concept/Term from the DTS Knowledgebase. A confirmation dialog box is displayed before deleting. There is no way to undo this Delete. The Delete button is disabled on Edit Only Templates.

Template Creation

A Template is created in the Template Editor build panel. The build panel can be accessed from the DTS Editor Tools menu Template Builder item as a floating panel, or from the Template Builder tab in the standalone TemplateEditor application. Each template specification is stored in a separate XML file named *template name.tpl* located in the */bin/templateeditor* folder.

Figure 2 shows a floating build panel for the Auto Model Template displayed in **Figure 1**. The components of the panel are the top **Template Select Area**, the **Base Information Area**, the **Attributes Area** (with associated buttons), and the bottom **Button Area**.

The screenshot shows the Template Editor window with the following components:

- Template Select Area:** Contains a 'Template Name' dropdown set to 'Auto Model', a 'Load' button, and a 'New' button.
- Base Information Area:** Contains fields for 'Description' (Create/edit auto models), 'Template Type' (Concept), 'Concept/Term Namespace' (Automobiles), 'Concept Subset' (- None -), 'Update Namespace' (Automobiles), and 'Validator Class'. There are also checkboxes for 'Request/Show Code and Id' and 'Edit Only'.
- Attributes Area:** Contains a table of attributes with columns for the attribute name, a value, and an 'Edit' button.
- Button Area:** Contains buttons for 'Insert', 'Delete', 'Up', 'Down', 'Save', 'Copy', 'Publish', 'Delete', 'Reset', and 'Clear'.

Attribute Name	Value	Action
Concept Name	R	Edit
[Inv]Parent Of[Automobiles] (A)	B	Edit
Model Synonym[Automobiles] (S)	N	Edit
has Type[Automobiles] (A)	HRMP	Edit
Value Rating[Automobiles] (P)	HRMVQ	Edit
Year Introduced[Automobiles] (P)	RI	Edit
History[Automobiles] (P)	L	Edit

Figure 2 – Template Builder Panel

Template Selection

To edit an existing Template's specification, select the Template's name from the `Template Name` combo box and press the `Load` button. The data for the selected Template will be loaded. To create a new Template, click on the `New` button. A dialog will appear prompting for the name of the new template.

Base Information

The **Base Information Area** contains the basic parameters for the Template. These eight parameters define the overall context and behavior of the Template.

The `Description` parameter provides for an optional line of descriptive text to be associated with the Template.

The three “primary” Template parameters are the `Template Type`, `Concept/Term Namespace` and the `Update Namespace`. Values for all three of these parameters must be selected before many of the other parameters, including the `Templates Attributes`, are available. Changing the value of any of these three parameters will force a reset (clear) of all other Template specification data.

First select either `Concept` or `Term` from the `Template Type` combo box. This selection determines whether the Template will display, and allow editing of, `Concept` or `Term`-based information.

Next, select a Namespace from the `Concept/Term Namespace` dropdown. The `Concept/Term Namespace` is the namespace of the Template's primary object, i.e., the Namespace from which `Concepts` or `Terms` will be retrieved, or into which new or updated `Concepts` or `Terms` will be saved.

The `Update Namespace` is the third of the three primary Template parameters. The `Update Namespace` is the Namespace from which editable DTS `Concepts`, `Terms` and `Attributes` are drawn. Since these objects must be updatable, the values in the `Update Namespace` combo only contain the names of `Thesaurus` and `Ontylog Extension` Namespaces. Select the desired `Update Namespace` from the dropdown. To create mappings from one Namespace to another, for example, the `Concept/Term Namespace` should be the source Namespace (`Ontylog` or `Thesaurus`) of the mapping, while the `Update Namespace` should be the Namespace of the mapping Association. Restrictions on the mapping's target Namespace can be specified in the `Attribute validation details` (see the **Attribute Detail Panel** section below). To add new `Concepts` or `Terms`, the `Update Namespace` must be the same as the `Concept/Term Namespace`. Finally, note that if the `Update Namespace` is an `Ontylog Extension Namespace`, then the `Concept/Term Namespace` must also be the same `Ontylog Extension Namespace`.

The remaining four parameters provide additional Template-wide specifications. Some of these parameters may be disabled based on other parameter settings.

If the Template `Concept's` (or `Term's`) `Code` and `Id` should be displayed, or explicitly entered for new `Concepts/Terms`, select the `Request/Show Code and Id` checkbox. This causes `Code` and `Id` attribute fields to appear after the `Name` field in the Template's `Attribute` area and data entry form.

Otherwise, if creation of new Concepts/Terms is permitted, DTS's internal Code and Id generator will be used to provide values for these Attributes. (In the sections that follow, the Name, Code, and Id Attributes will be referred to as the *Fixed Attributes* of the Template.)

If only *existing* Concepts/Terms should be loaded into the Template, check the `Edit Only` checkbox. If this parameter is selected, the Template entry panel will not prompt for, or permit, the creation of new Concepts/Terms. This parameter is only enabled if the Concept/Term Namespace is the same as the Update Namespace. Otherwise, this parameter is disabled and the Template is Edit Only by default.

The `Concept Subset` dropdown is only enabled after values for the three primary Template parameters have been provided, and if the Template Type is Concept and the Template is Edit Only. Values in the dropdown are the names of those Subsets associated with the Concept/Term Namespace. If a Subset is selected, only Concepts from the designated Subset may be loaded. An "s" will be shown to the right of the Concept Name field in the associated Template entry panel (rather than a "c") if a Concept Subset has been specified.

Finally, the fully-qualified Class name of any desired Template Validator Class can be entered in the `Validator Class` field. (See **Appendix A** for further information on the Template Validator.)

Template Attributes

Concept/Term Attributes are added to the Template by entering them into the table in the Attributes Area. Attributes will be displayed in the Template's data entry form in the same order as in this table. To add an Attribute, click on `Insert` in the attribute button list. A new Attribute row will be added above the currently selected row (or at the bottom of the list). (Note that Attributes cannot be inserted before or between Fixed Attributes. The Fixed Attributes always appear at the top of the form in Name, Code, and Id order. An attempt to insert a new Attribute into this area will result in the Attribute being added after the last Fixed Attribute.)

Next, double click in the first cell of the row, the cell showing "(select)", and select the desired Attribute. The dropdown list contains all the Attribute Types from both the Concept/Term Namespace and the Update Namespace. Items in the list are ordered by the class of the attribute, e.g., Extension Attributes, Synonym Attributes, Property Attributes, Association Attributes and Role Attributes. Within each class (shown by a parenthesized code at the end of each Attribute name), the order is by Namespace, then alphabetic by Name of the Attribute. The "signature" of the Attribute's parameters are displayed in the second cell of an Attribute's row. To add or edit the parameters of any attribute, click on the `Edit` button at the end of the attributes row. This opens an Attribute Detail panel for the attribute. Only one Attribute Detail panel can be open at one time. For details on the operation of this panel, see the **Attribute Detail Panel** section below.

To delete an existing attribute, select it by clicking in the second (signature) column of the attribute row, then click on `Delete` in the attribute button area. Fixed Attributes cannot be deleted.

Attributes appear in the entry form in the same order as in the attribute area. To move an attribute row up or down in the table, select the row and click on the `Up` or `Down` button. Fixed Attributes cannot be moved from their defined positions.

Button Area

There are six buttons at the bottom of the build panel: Save, Copy, Publish, Delete, Reset, and Clear.

Save	Updates the Template's specification with the current parameters.
Copy	Copies the current Template specification to a new Template. If the specifications have changed since the last save, a confirmation dialog will be displayed asking if the current Template should be updated. Then a new Template name is requested.
Publish	Publishes the selected Template specification as an independent DTS Editor Module. See <i>Publishing Templates</i> below for further information.
Delete	Deletes the specified Template from the Template File. A confirmation dialog box is displayed before deleting. There is no way to undo this Delete.
Reset	Restores the Template specification data to its most recent saved, or initial, state. If the designated Template exists, the fields are reloaded from the Template file. If the current Template has not yet been saved, the fields are cleared.
Clear	Clears (sets to blanks) all parameters in the specification.

Attribute Detail Panel

The screenshot shows the 'Template Attribute Detail' dialog box. It is divided into three main sections: 'Basic Data', 'Values', and 'Qualifiers'.
- **Basic Data:** The 'Attribute' field is 'Value Rating[Automobiles] (Property)'. The 'Label' field is empty. The 'Help' field contains the text 'Select the value of this model.'. There are checkboxes for 'Hidden', 'Display Only', 'Required' (which is checked), 'Long Value', and 'Multiple' (which is checked).
- **Values:** There is a 'Default Value' text field. Below it are radio buttons for 'No Validation', 'Namespace', 'Parent', 'Subset', 'RegEx', and 'Value List' (which is selected). The 'Value List' is currently showing a list with items '1', '2', and '3'. Below these are 'Int. Range' spinners for 'Lower' and 'Upper', both set to '0'.
- **Qualifiers:** There is a list box containing 'Reviewer[Automobiles] (PQ)' and an 'Edit' button next to it.
At the bottom of the dialog are buttons for 'Insert', 'Delete', 'Up', 'Down', 'Save', 'Clear', 'Test', and 'Cancel'.

The Attribute Detail panel presents attribute-specific parameters used by the Template Editor for the presentation and processing of the associated form entry field. **Figure 3** shows the Attribute Detail Panel for the Value Rating attribute in the Auto Model Template. Available parameters vary by the type (Fixed, Kind, Primitive/Defined, Defining Concept, Defining Role, Synonym, Property, Association, or Qualifier) of attribute. Each parameter has a “signature character” (underlined in the label of the parameter) that is displayed in the build panel Attribute area (see above). Areas of the detail panel are Basic Data, Values, Qualifiers, and action buttons.

Figure 3 – Attribute Detail Panel

Basic Data for an attribute includes:

- | | |
|--------|--|
| Label | An optional label, or prompt, for the data entry field. If label is blank, the name of the attribute is used. For Inverse Concept Associations, the label defaults to its inverse name, if one has been specified. The <u>L</u> abel signature character is “B”. |
| Help | An optional help string for the field. A non-blank value causes a help icon to be placed to the right of the field on the Template’s entry form (see Figure 1). A non-blank value also overrides any default help message, such as that provided for droppable fields. The help string is shown as a Tool Tip when the user hovers over the icon. The string is also used in field validation error messages. The <u>H</u> elp signature character is “H”. |
| Hidden | If checked, the attribute is not displayed on the entry form. This permits attributes with fixed values, e.g. audit entries, to be automatically created whenever a Template entry form is saved. The value associated with the <u>H</u> idden attribute is that given in the <u>D</u> efault <u>V</u> alue parameter. This must be non-blank for the attribute to be |

created. Other Value Area options are disabled. Hidden is not available for Synonyms. The Hidden signature character is “N”.

- Display Only** If checked, the attribute is “read-only”, i.e., a new value cannot be entered. Display Only is typically selected in two situations. First, when a Template is Edit Only and certain fields of the Concepts/Terms should be displayed but not modified. Second, as an alternative to Hidden. As with Hidden, all Value Area parameters are disabled except the Default Value parameter. For new Concepts/Terms, the Default Value will be loaded into the Attribute, which will be displayed, but not editable. Display Only is automatically set (and the field disabled) for the Kind attribute. Display Only is also automatically set on any Attribute that is not from the Update Namespace. Finally, note that Display Only has a unique interpretation when used on the Name field. For the Name field, value entry is obviously enabled, but if Display Only is selected, the Concept/Term cannot be renamed. See the **Changing the Concept/Term Name Value** section above for further details. The Display Only signature character is “O”.
- Required** If checked, a value for the attribute is required. The Template Concept form cannot be saved as long as any required field is blank. Required is set automatically for the Primitive/Defined attribute. The Required signature character is “R”.
- Long Value** If checked, an extended text area is displayed in the attribute’s form field. This field permits entry of Enter and Tab as well as standard characters. Text in the area will automatically wrap for improved visibility. Wrapping does not modify the contents of the field value. The Long Value parameter is only available for Properties. The Long Value signature character is “L”.
- Multiple** If checked, multiple occurrences of the attribute can be created. The Multiple parameter places an Add button next to the last occurrence of the attribute in the Template data entry form. See the **Data Entry** section above for further information. If both Required and Multiple are active for an attribute, only the first instance of the attribute in the entry form will be required, i.e., new instances will NOT be marked as required. The Multiple signature character is “M”.

The Value area contains the following parameters: (This area is disabled for Fixed and Synonym Attributes.)

- Default Value** This is the default value for the attribute’s data entry field. If a Parent, Subset, RegEx, Value List or Int. Range validation (see below) is present, the default must satisfy the validation. For Properties (and Qualifiers), the value can include the Template Editor code strings “@D”, “@T” and “@U” which will be replaced at Concept/Term save with the current date, current time, and DTS User ID respectively. When running in the DTS Editor, if the attribute is Concept/Term valued, the Default Value parameter area supports drag-and-drop from other DTS Editor panels. The Default Value signature character is “D”.

The following parameters are selected by clicking on the associated radio button and completing the associated parameter fields. **Note: Only one of the Namespace, Parent, Subset, RegEx, Value List or Int. Range parameters may be active for the Attribute.**

- | | |
|------------|--|
| Namespace | The Namespace parameter specifies that the (Concept/Term) value of the Attribute must be from the selected Namespace. The <code>Namespace</code> combo contains the names of all Namespaces. The namespace signature character is “N”. |
| Parent | The value of the Parent parameter is the name of a Concept (from any Namespace) the Concept Names of whose children will be the Value List for the attribute. In other words, only the Concept Names of the specified Parent’s children will be permitted as values of the attribute. The names will be displayed (uneditable) in the Value List parameter (see below). When the Template is executed, the allowed values will also be displayed in a non-editable combo box. Even though the associated attribute may be Concept/Term-valued, this combo box will not support drag-and-drop. When running in the DTS Editor, the Parent field supports drag-and-drop from other DTS Editor panels. If the Parent’s Namespace is of Thesaurus type, the <code>Parent Of Association</code> is used to determine children. The Parent signature character is “P”. |
| Subset | The Subset parameter specifies that values for the attribute must be Concept names from Concepts that participate in the named Subset. Available Subsets are given in the parameter’s combo box. The Subset signature character is “S”. |
| RegEx | The RegEx parameter specifies that the associated attribute value must satisfy the entered Regular Expression. Validity of the Regular Expression is tested by the Attribute Detail Panel. The <code>RegEx</code> signature character is “X”. |
| Value List | The Value List parameter specifies the list of permitted values for the attribute. When specifying a Value List, enter the values in the text area with each value separated by “Enter”, i.e., on a separate line. Blank lines are ignored. The Primitive/Defined attribute automatically creates a Value List with the values “Primitive” and “Defined”. This default validation cannot be overridden. When the Template is executed, the specified values will be displayed in a non-editable combo box. The Value List signature character is “V”. |
| Int. Range | The Integer Range parameter specifies the lower and upper limits for an integer attribute value. A value can be entered into either of the fields, or the up and down arrows (“spinners”) can be used. The lower value must be less than the upper value. In the example in Figure 3 , an integer range could have been used instead of the Value List specification. The Integer Range signature character is “I”. |

The Qualifiers area is a list of Qualifier attributes that will be associated with the base attribute in the entry form. This area is enabled when the base attribute is a Property and (Inverse) Association and the Template’s Namespace contains corresponding Qualifiers. The Qualifier area is similar in operation to the Template Attributes area in the builder panel as described above. After inserting and selecting a Qualifier row, clicking on the row’s `Edit` button opens a new (cascaded) Attribute Detail Panel for the Qualifier. This panel contains the basic, value, and action button areas. The Qualifiers signature character is “Q”.

Attribute Detail Panel Buttons

The Attribute Detail Panel action buttons are `Save`, `Clear`, `Test` and `Cancel`:

<code>Save</code>	Updates the attribute's specification with the parameter data and closes the panel.
<code>Clear</code>	Clears (set to blanks) all the parameter values.
<code>Test</code>	Displays a value test window. Entered values are tested against the panel's validation parameters. This button is only available when the <code>Regex</code> or <code>Int. Range</code> validators are selected.
<code>Cancel</code>	Closes the detail panel without updating any parameters.

Publishing Templates

Templates can be published to create a DTS Editor Module that consists of an implementation of a Template entry panel running (only) the associated Template specification. The Template thus becomes an independent Module application. The publishing feature, available in the Template Builder panel, bundles the specification and all required executables into a standard DTS Editor Module kit (zip file). The kit can subsequently be loaded into any DTS Editor instance. A published Template Module does not include the full Template Editor application and the Template Editor Module does not need to be present in the target DTS Editor instance.

To create a Template Module, open the Template Builder panel (see **Figure 2** above), select the desired Template and click on the `Publish` button. This opens the Publish Template dialog.

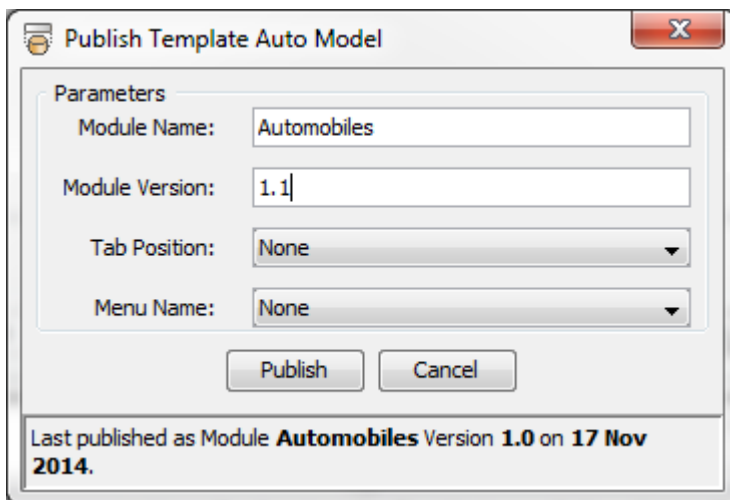
Complete the publish parameter fields described below:

Module Name	Required. Enter the desired Module Name. This value will be used in the name of the Module distribution kit, as the name of the Module's menu item in any menu and as the title of the Module's dialog.
Module Version	Required. Enter the version designator value for the Module. The format is a list of period-delimited numbers such as "1.3" or 12.4.7".
Tab Position	Select the location of the Module in the default DTS Editor layout if the Module should be shown as a tab in one of the tab panels. Available values are <code>None</code> (the Module will not be shown), <code>Left</code> and <code>Right</code> . Note that this setting is only considered if the default DTS Editor layout is used and the <code>EnablePlugins</code> layout parameter is true.
Menu Name	Select the name of the DTS Editor standard layout menu into which an item for the Template Module will be added. The name of the item will be <code>New <Module Name></code> . Available values are <code>None</code> (the Module will not be referenced in any menu except the <code>Help</code> menu), <code>File</code> , <code>Tools</code> , and <code>Options</code> . Note that an entry in the standard <code>Help</code> menu will always be added.

Note that a Template Module can be placed anywhere in a custom (user-written) DTS Editor layout (and menu tree) independent of the saved values of `Tab Position` and `Menu Name`.

Click on `Publish` to initiate the operation. A confirmation dialog will be shown. During publishing, progress is shown in the message area. If the publish operation is successful, a Module kit with the name `<Module Name>-<Module Version>.zip` is created and placed in the *install* directory of the current DTS Editor instance.

If a Template has previously been published, information on the published Module is supplied in the dialog's message area and the `Module Name` field is automatically set to the previously published name (see below). A different Module name can still be entered.



Revision History

Version 1.0	Initial release.
Version 1.1	Concept and Term Template Types. Fixed Attributes appear in Attribute list and have selected Attribute details. Help and RegEx parameters for Attributes. Clear button in entry panel.
Version 1.2	DTS 3.4. Help option (this User Guide) in plug-in and standalone implementations.
Version 1.3	Integer range validation for Attributes.
Version 1.4	Concept/Term searches. Edit-Only templates and Display-Only Attributes. Load button in Builder.
Version 1.5	Subset added as a validation parameter. Wild card searches supported on all Concept/Term-valued attributes. Ontylog Extension Namespaces are supported along with new Attributes: Kind, Primitive/Defined, Defining Concept, and Defining Role. Default Help strings supplied for Concept and Term valued fields. Concept, Term, and Subset markers added for droppable fields.
Version 1.6	Separate Concept/Term and Update Namespaces. Namespace attribute validation parameter. Subset parameter on Template Concept Names. Drop error messages. @U code string added for DTS User ID. Bug fixes on Extension Namespace support and qualifier data entry.
Version 2.0	Separate template files with automatic conversion from Version 1.x formats. Delayed template load errors. Improved keyboard navigation. Renaming and Copy functions added. Alphabetic sorting of multiple attributes. Test button for Regex and Int. Range value parameters. About dialog.
Version 2.1	Improved Term lookups and processing. Bug fixes.
Version 3.0	Updated for DTS Version 4.0. Qualifier fixes.
Version 3.1	Update for DTS Version 4.1. Bug fixes.
Version 3.2	Publishing of Templates to DTS Editor Modules.
Version 3.3	Bug fixes. Update for DTS 4.3 Layout Manager. Template Editor and published templates can be use as Layout panels.
Version 3.4	Bug fixes for handling multiple Terms with the same name.

Appendix A – Writing a Template Validator Class

The Template Editor supports custom field and Concept validation through a Template Validator Class. A Template Validator Class is a Java Class that implements the `TemplateEditor.TemplateValidator` Interface. This Class is found in the `com.apelon.modules.dts.editor.templateeditor` package.

A validation class must implement the following three methods:

```
public String validateAttribute(String value, String attribute);

public String validateConcept(DTSConcept con);

public boolean reviewConcept(DTSConcept con);

public String validateTerm(Term term);

public boolean reviewTerm(Term term);
```

The `validateAttribute` method is called with a (non-blank) prospective value and the name of the associated DTS attribute name (Synonym, Property, Association, Inverse Association, or Qualifier). The Concept Name, Code, and Id (and Term Name, Code and Id) are passed for validation using the Template Editor constants:

```
TemplateEditor.TemplateAttribute.CONCEPT_NAME,
TemplateEditor.TemplateAttribute.CONCEPT_CODE,
TemplateEditor.TemplateAttribute.CONCEPT_ID,
TemplateEditor.TemplateAttribute.TERM_NAME,
TemplateEditor.TemplateAttribute.TERM_CODE, and
TemplateEditor.TemplateAttribute.TERM_ID.
```

`validateAttribute` is called after a new (non-blank) data value has been entered in a field, and the value has passed any specified validation. The method should return the (possibly updated) value if valid, or the empty string ("") on validation error. An updated value can be used to format an item in a standard way, e.g. "203-431-2530" for a telephone number.

The `validateConcept` method is called with a complete prospective `DTSConcept` object when the form Save button is pressed on a Concept Template. This method should return the empty string ("") on success, or an error message on failed validation.

The `reviewConcept` method is called after a Concept has been saved. Return true to force a reload of the Concept into the Template from the DTS Knowledgebase. This method is used when the `validateConcept` method directly modifies the (stored) version of the Concept.

The `validateTerm` method is called with a complete prospective `Term` object when the form Save button is pressed on a Term Template. This method should return the empty string ("") on success, or an error message on failed validation.

The `reviewTerm` method is called after a Term has been saved. Return true to force a reload of the Term into the Template from the DTS Knowledgebase. This method is used when the `validateTerm` method directly modifies the (stored) version of the Term.

The `TemplateValidatorAdapter` class (part of the `com.apelon.modules.dts.editor.templateeditor` package) provides default (accept all) implementations of the required validator methods. By writing a class that extends `TemplateValidatorAdapter`, only needed methods be implemented.

To install a Template Validator Class, add its `.class` file to a `.zip` or `.jar` file and place the archive file into the `...DTS/lib/modules` folder in the DTS installation directory. If using `TemplateEditor.bat`, also add the archive file's path to the `CLASSPATH` variable.