Introduction to Templates

User Guide

First published September 2007

AVEVA Solutions Ltd, High Cross, Madingley Road, Cambridge, CB3 0HB, United Kingdom

Introduction to Templates User Guide

Contents **Page**

Introduction to Templates

1 Read This First

1.1 Scope of the Guide

This guide introduces the facilities provided by AVEVA for the creation of Design Templates. These are standard design configurations which you can store for subsequent use by yourself or other designers.

The guide explains the main concepts underlying the creation of Design Templates and their incorporation into the Specifications which make them accessible to other designers. A key feature of the guide is a hands-on tutorial exercise which is incorporated throughout.

This guide does not give step-by-step instructions on how to carry out every specific template design function, since you can access such information as you work by using the on-line help facilities incorporated into the program's graphical user interface.

If you are not yet familiar with the use of the DESIGN module, it is recommended that you first work through one or more of the discipline-specific guides (*Pipework Design User Guide*, *Structural Design Using PDMS User Guide*, etc.) and practise using the various DESIGN applications. You need to be proficient in the use of DESIGN before you try to create design templates for use by others. You also need to be familiar with the main principles of managing Catalogues and Specifications in databases using the PARAGON module: the use of Design Templates provides an extension of those facilities.

For fuller information about using the whole range of facilities, refer to the sources listed in Appendix A of this guide.

1.1.1 Intended Audience

This guide has been written for engineers familiar with template design practices and with a prior knowledge of PDMS.

1.1.2 Assumptions

For you to use this guide, the sample project, Project SAM, must be correctly installed on your system, and you must have read/write access to the project databases.

It is assumed that you know:

- where to find PDMS on your computer system
- you know how to use the Windows operating system installed on your site.

Contact your systems administrator if you need help in either of these areas.

1.1.3 About the Tutorial Exercise

All the steps of the exercise are numbered sequentially throughout the guide.

1.1.4 Further Reading

You can find a list of relevant AVEVA documentation in the appendices of this guide.

1.2 Terminology

You can switch rapidly between the different parts of the program, so that the distinctions between them become almost imperceptible, but you need to recognise what is happening when you select from the different functions available to you from the various menus.

The following terms are used throughout this guide to describe what action to carry out:

Enter	Type text into the specified dialogue box, then press the Enter (or Return) key to confirm the entry.
Click	Place the mouse cursor over a specified point, then quickly press and release the designated mouse button. If no button is specified, use the left-hand mouse button.
Pick	Click on the required item to select it.
Drag	Place the mouse cursor over a specified point, then press and hold down the required mouse button while moving the cursor to a second specified point. Release the button over the second point.
Double-click	Place the mouse cursor over a specified point, then click the left-hand mouse button twice in quick succession.

1.3 How the Guide is Organised

This guide contains the following chapters and appendices:

This chapter introduces this guide and summarises its scope.

Creating a Simple Template	first introduces the concept of design templates. It then describes the geometric design of a simple template representing a structural penetration hole surrounded by a kickplate assembly.
Setting Template Properties and Rules	explains how to set up the properties and rules to parameterise the design template dimensions so that it can be used in subsequent design operations.
Selecting a Design Template	summarises the steps needed to include the design template in a catalogue, so that it can be selected by a designer via a specification.
A More Advanced Example	further illustrates the principles by means of a more complex example representing an equipment vessel with user-selectable end configurations.
Some Further Information for Template Designers	gives some additional information which you might find relevant as you gain experience of template design.

Hints and Tips	lists some points to remember to ensure success when designing templates.
Other Relevant Documentation	identifies other sources of information which supplement, and expand upon, the brief details given in this guide.

The guide concludes with an Index, allowing you to refer back to any specific topics about whose details you need to be reminded.

2 Creating a Simple Template

2.1 Basic Concepts

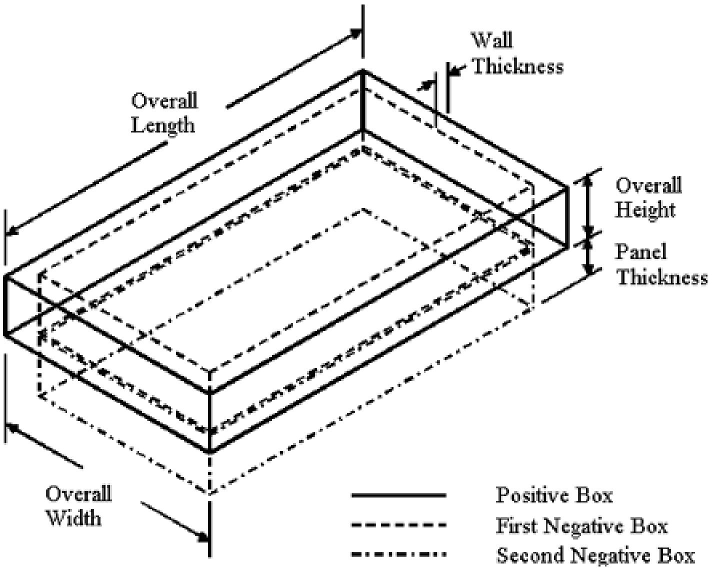
A design template is a set of design primitives, panels and nozzles that may be grouped together and then referenced from within another part of the design database as though it were a single item. In many ways, a design template behaves in a similar way to a catalogue component, except that the template items are stored in a special area of the DESIGN DB, rather than in a separate Catalogue DB, and they can use the more powerful sets of primitives and parameterisation facilities available from within DESIGN. Unlike a catalogue component, a design template can be split down into its constituent parts for selective reporting, dimensioning, MTO, etc.

A design template is used in a design by creating an instance of the template in the DESIGN DB. When a design template is instanced, the template contents are copied into the design hierarchy; they may then be modified locally as required.

A design template may be referenced from Equipment or Sub-Equipment, a Panel Fitting, a Section Fitting or a Primary Joint.

2.2 Creating an Initial Template Design

To demonstrate the main principles of design template creation, we will create a template representing a simple rectangular kickplate configuration which can be added round a penetration hole in a panel. The template will consist of three box primitives: a positive box representing the outer surfaces of the kickplate assembly; a negative box which removes most of the material from the positive box, leaving only a wall thickness representing the individual kickplates; and a second negative box which will penetrate any panel on which the template is positioned by a designer. The configuration will be as follows:



When the design template is used in a design, its dimensions will be specified by reference to a set of predefined properties. It is important, therefore, that you have thought carefully about which dimensions are to be parameterised before you create the design template (just as you would normally sketch out a new catalogue component before starting to build it up in PARAGON).

In our present example, we will define five properties representing overall length, overall width, overall height, wall thickness for the kickplates, and the panel thickness for the penetration hole below the kickplate assembly, as shown on the preceding diagram.

Exercise begins:

1. The Design Templates application makes the assumption that a new template will be based on an existing set of design elements (or on a predefined design template), so we will first use the Equipment application to create the positive box and the first negative box. (We will see later why the second negative box, representing the panel penetration hole, cannot be created at this stage.)

   Select **Design>Equipment** to enter the Equipment application.

2. From the **Equipment Application** menu, select **Create>Site** and name the new site **/TMPLSITE**.

3. Below this site create a new Zone named **/TMPLZONE** and below this an Equipment named **/Kickplate**. Leave the equipment **Position** at the default of 0,0,0.

   It is this equipment element which will own the BOX, which will in turn own the first NBOX, which will be used as the basis for the design template.

4. Select **Create>Primitives** and use the **Primitives** form to create a **Solid Box**.

   Select the **Box** and set the following data:

   Set **Y-length** to **1000, X-length** to **500** and **Z-length** to **250**. These dimensions give a starting point for the design. The template derived from this box will redefine the dimensions in terms of parameterised rules.

   Click on the **Create** button to create the box.

   Leave the **Position** at the default of **0,0,0** and the **Orientation** at the default of **Y-Axis North** and **Z-Axis Up**.

   Close the **Primitives** form.

Select **Modify>Name** and enter the name as **/Kickplate-Outer**.

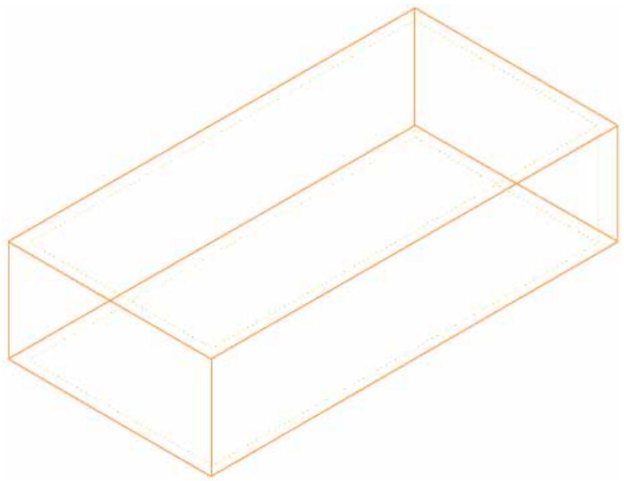
5. Using the same procedure, create a Negative Box with Name /Kickplate-Inner, Y-length 950, X-length 450, Z-length 250, default Position and Orientation.

   Note that the X and Y dimensions of the NBox are smaller than those of its owning Box by an amount corresponding to twice the wall thickness, but the Z dimensions of both boxes are the same.

6. Use the following settings for viewing the results:

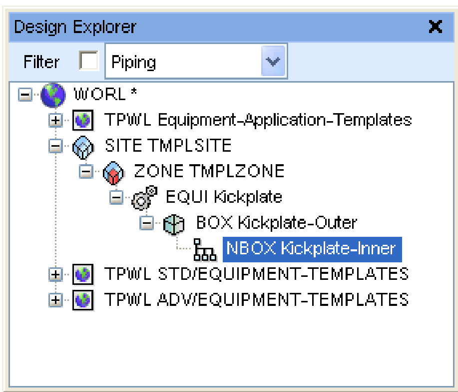
   - Click the **Walk to Draw List** button () on the 3D View Tool Bar.
   - Set the view direction to Iso 2 by selecting **Isometric>Iso 2** from the 3D View shortcut menu.
   - Set the **Representation** (**Settings>Graphics>Representation** from the main menu) to **Holes Drawn Off (not ticked)**.
   - If you currently have a colour-shaded view, switch to a wireline view (**View>Settings>Shaded** from the **3D View** menu, or press **F8**).

     The result should look like this:



Try some other view settings if you wish.

The **Design Explorer** should now show the following elements:



Save your design changes (**Design>Save Work**).

That ends the creation of the basic design, which will be copied to form the design template.

In the next part of the exercise we will create a suitable hierarchy under which to store the template and will then create the template itself.

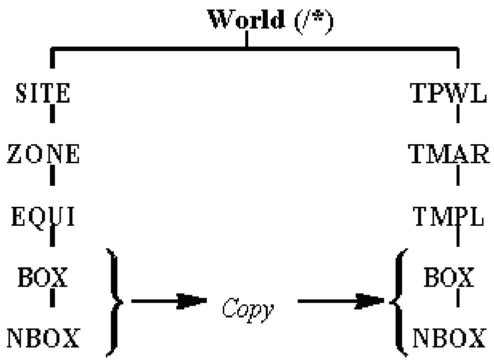
2.3 Design Template Hierarchy

Design Template (TMPL) elements are stored in a separate part of the Design DB under an administrative Template World (TPWL), which is itself divided into administrative Template Areas (TMAR).

When you are working in the Design Templates application, you need to be able to navigate concurrently in two different parts of the Design database: the area which holds the design data on which the templates are to be based, and the area which holds the templates themselves. Template creation involves, among other steps, copying data from the former area to the latter.

Navigation in the design area is carried out via the **Design Explorer** in the usual way, while navigation in the template area is carried out via a separate **Template Browser** form.

The hierarchy with which we will be concerned in the next part of the exercise is as follows:



Exercise continues:

7. In order to create the design template elements, we must first change from the Equipment application to the Design Templates application. To do so, select **Design>Design Templates** from the **Equipment Application** menu.

You will see the **Design Templates Application** menu bar plus the **Template Browser** form. The latter lists all of the design templates which have been supplied as part of the sample project. During the current exercise, we will add the templates which we will create under a new top-level Template World.

8. From the **Design Templates Application** menu, select **Create>Template World**. Set the Name to **/TESTTPWL** and set the **Description** to **Template World for exercises**.

The new template world will be created immediately below World level at the top of the current design hierarchy, regardless of the level of your current design element. Notice that **Template Browser** shows the element's description, whereas the **Design Explorer** shows its name (you will not see the latter yet unless you navigate to it).

9. Select **Create>Template Area** and name the template area **/TESTTMAR.** Enter its description as **Template Area for exercises**.

2.4 Copying Design Geometry into a Template

We are now going to create a design template whose geometry will be based on a copy of the equipment element which we have already created, namely a positive box with a negative box inside it. Later, we will modify this design within the template by adding another negative box; this will not affect the original equipment in any way.

Exercise continues:

10. When you create a design template (TMPL), it will copy its geometry from the current design element, so first navigate to the Equipment (indicated by the symbol) **Kickplate** in the **Design Explorer** and then select **Create>Template** from the **Design Templates Application** menu.

You will now see a **Create Template** form and a **Positioning Control**.

The **Copy Model** data at the bottom of the **Create Template** form shows the name of the current design item (in this case, the equipment /Kickplate).

Set the following Template Information:

Name	Kickplate-1
Purpose	Leave as Unset
Description	Protected floor penetration
Function	Rectangular
Generic Type	PENH (Penetration Hole)

(This data is optional, but is helpful when the template is to be selected for use in a design. In some cases the generic type will be deduced automatically from the original design item, but this is not possible in our current example.) Do **not** OK the form yet.

We will set the position of the template's Origin by picking it in the 3D View. To do this, you need to understand the concept of **event-driven graphics**. This is explained next.

2.5 Event-driven Graphics Mode

Before we begin the next part of the exercise, it is necessary to understand how to use the cursor to pick points in the graphical view. Whenever the status line immediately below the menu bar of the 3D View shows a prompt other than 'Navigate', as now, the graphical view is switched automatically into event-driven graphics mode. This means that when you pick a point in the displayed graphics, your action is interpreted in whatever way is appropriate to your current design operation (i.e. the current event) rather than simply as a request to navigate to a new current element. In this example, picking in event-driven graphics mode will be used to specify a position.

The position derived from your cursor pick can be the exact point at which you have placed the cursor or, more commonly, it can be a position which is related to the picked point in a specified way. The main concept involved in structural applications is that of the snap function, which automatically chooses the nearest Start, End or (optionally) Secondary Node position to the picked point, so that you do not need to be very accurate when positioning the cursor.

The full range of options available for identifying positions is extensive. We will use it in the exercise simply to pick a p-point which is already at the required position.

Exercise continues:

11. The current position of the design template origin, relative to the template geometry, is at the centre of the box, as shown by the **Template Origin** label which you will now see in the 3D View (rotate the view if necessary to see the position properly; the label may be difficult to read because the equipment origin is labelled at the same location).

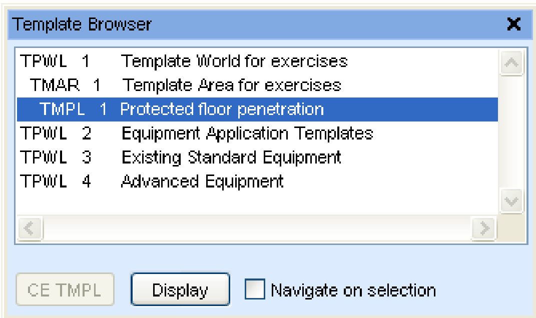
In order to provide a more convenient datum point for positioning the kickplate assembly relative to the surface of a panel, we will redefine the origin as being at the centre of the lower face of the box.

To do so, we will enter the **Origin** data on the **Create Template** form by picking an existing p-point at the required position. We are already in event-driven graphics mode (as shown by the 3D View status line which says '**Defining a Template Snap**').

On the **Positioning Control** form, set the Pick Type (left-hand box) to **Ppoint** and the Pick Method to **Snap**. This constrains the system to allow you to pick only p-points. Move the cursor into the 3D View, hold down the left-hand mouse button, and move the cursor over the box (click on any edge of the box if you are using a wireline view). The p-points appear as dots in the view and the cursor shape changes to ⌖ when it is over a p-point. The name of the selected point is shown in the status line. Release the mouse button when the cursor is over **P6 of BOX /Kickplate-Outer**.

Notice how the **Origin** coordinates are now shown in the **Create Template** form as **East 0, North 0, Down 125** and how the labelling has changed in the new 3D View.

12. Leave the **Add Properties** and **Add Points** selected and **OK** the **Create Template** form. The **Template Browser** should now show the following elements:



The **3D View** will now show both the original equipment and the template derived from it (offset by the amount by which the origin has been repositioned). From the **Template Browser**, select **Display** to show just the current template in a 3D view.

2.6 Modifying the Template Geometry

In order that the template can penetrate a panel on which it is positioned when instanced in a design, it must include a second negative box owned by the TMPL. A negative box owned by the positive box will not have the required effect, which is why we did not create it in the original equipment.

Exercise continues:

13. The new NBOX must be created below the TMPL. To achieve this, you must first navigate to the TMPL in the **Design Explorer**. The easiest way to do this is to check the **Template Browser's Navigate on selection** checkbox and then reselect the TMPL in the browser list. This element will become current in the **Design Explorer** automatically.

Alternatively, if you have the **Draw List** window open, you can choose the **Navigate To** option from the popup menu that appears.

From the **Design Templates Application** menu, select **Create> Primitives**. Create a **Negative Box**.

Set the **Y-length** to **950**, **X-length** to **450** and **Z-length** to **100**.

Leave the **Orientation** at its default setting.

We want to position the new NBOX so that its upper face is coplanar with the lower faces of the existing BOX and NBOX. To achieve this, we must move it **down** by half its Z-length, so set the **Position** to **North 0, East 0, Down 50**.

Set the **Name** to **/Panel-pene**.

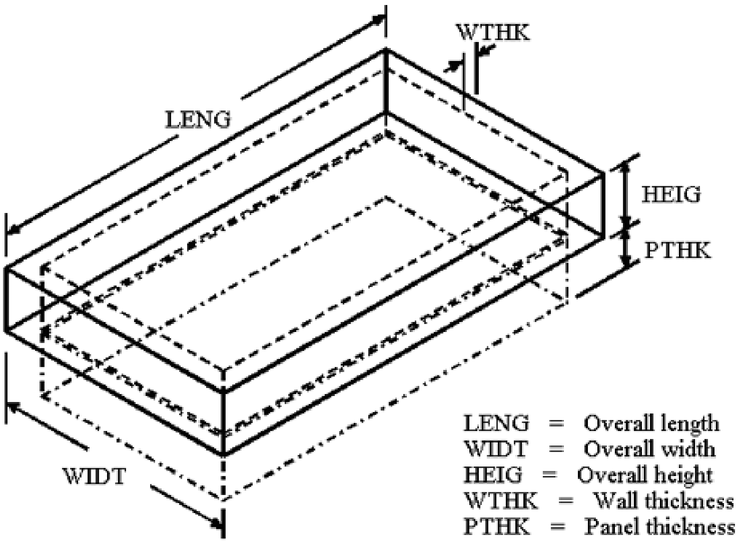
We have now completed the required design template geometry, as illustrated in the diagram near the start of *Creating a Simple Template*. In the next chapter we will see how to allocate properties and rules to the template, so that its geometry may be parameterised and adjusted to suit specific use in design contexts.

3 Setting Template Properties and Rules

Although we set the dimensions and relative positions of the Box and NBoxes to specific values to give a meaningful representation in the displayed view, these settings are largely irrelevant for the template. When the template is used in a design application, the dimensions and positions of the primitives will be reset by reference to a set of parameterised rules which adjust the geometry to suit the local circumstances. These rules are defined in terms of the template's properties, which we will now define.

3.1 Defining the Template Properties

Each property represents an 'attribute' of the template which is to be adjusted by reference to the parameterised rules when used in a design instance. In the current example, we need to define five properties corresponding to the following dimensions:



The template properties are stored in Design Data (DDAT) elements, owned by a Design Dataset (DDSE), which is itself owned by the TMPL element. When we created the template, we created an empty Design Dataset by setting the **Add Properties** button to On in Step *12.*

Exercise continues:

14. Check that the selected element in the **Template Browser** is the TMPL and then select **Modify>Property Definitions** you will see a **Define Template Properties** form which lets you set up the required list of properties.

The upper part of the form lets you specify the details for a single property, while the lower part displays a list of all of the properties which are currently defined. The list is empty at this stage.

Define the first property as follows:

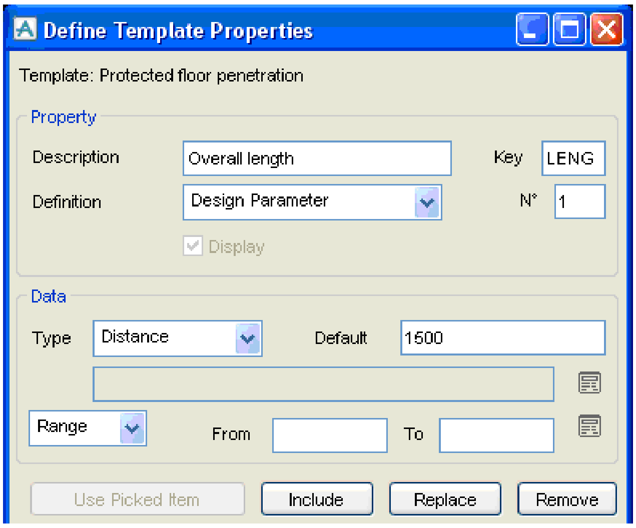
- Enter the Description as Overall length and the Key as LENG.
- The **Definition** option specifies how the value of the property will be derived when the template is used in a design. The choices are:

Design Parameter	Value taken from design data entered when template is instanced
Attribute	Value taken from named attribute (**Short Key** is set to attribute name)
Expression	Value is result of evaluating expression (as typed into text-box immediately below the option list)
Plotfile	Name of plotfile to be displayed
Specification Ref	Allows SpecRef (e.g. Profile) to be set
Sub Element Ref	Allows pointer to subsidiary element (e.g. Sub-Equipment) to be set when template is instanced

We will set the first four dimensional properties as **Design Parameters**, so select this option for the **Definition** field. In the N^{o} text-box, enter **1** for the first design parameter.

- Set the **Data Type** option to **Distance**.
- The **Default** text-box lets you specify the value which will be used if the correct value cannot be derived at any intermediate stage of the design process. Enter a default length of 1500.
- The **Range** text-boxes let you specify acceptable minimum (**From**) and maximum (**To**) values for the property value which can be entered in the design. Leave these unset. (Ignore the greyed-out **Display** button for now.)

The settings should now look like this:



Click the **Include** button to create the currently defined property in the list (there is no separate **Apply** button on this form).

15. Repeat the process to add the following property definitions to the list:

Description	Key	DesPar No	Default Distance
Overall width	WIDT	2	750
Kickplate height	HEIG	3	300
Wall thickness	WTHK	4	50

(You will notice that the application adds parentheses round the default value automatically. It is optional whether or not you enter these when you type in the default value.)

16. The fifth property, namely the thickness of the panel which the second NBOX is to penetrate, will be derived from the actual panel thickness as defined in the design. To achieve this, we will define the property as an expression which represents this thickness.

Enter the **Description** as **Panel thickness** and the **Key** as **PTHK**.

Set the **Definition** option to **Expression** and type the following expression into the expr text-box:

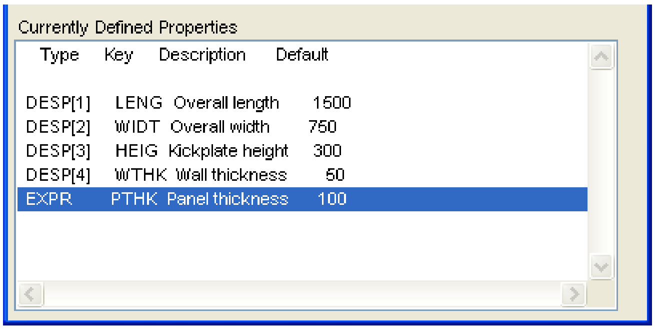
HEIG of PLOO 1 of PANEL

(since the panel thickness is represented by the Height attribute of the Panel Loop element owned by the Panel).

Set the **Default Distance** to **100**. Leave the **Range** limits **unset**.

Notice that the **Display** button is now selected. Its setting determines whether or not the value of this property will be shown on the data-entry form when the template is instanced in a design. Leave the button selected. (The property value will be shown for reference only, since it is derived automatically via the expression and cannot be set explicitly by the designer.)

The final **Currently Defined Properties** list should look like this:



Click the  button to remove the form.

3.2 Defining a Template Rule

So far, we have simply defined the properties of the template which must have their values set when the template is instanced in a design. We must now define the rules which define how the geometry of the template instance is to be parameterised in terms of those properties.

Exercise continues:

17. Check that you are still at the TMPL in the **Template Browser** and then select **Modify>Parameterisation**. You will see a **Template Member Parameters** form listing the primitives which make up the current template.

In the **Template Members** list, select the **BOX**. Notice how the **Rules** list automatically shows those attributes of the selected element for which you can set rules.

In the **Rules** list, select **X length**. You will see a **Attribute Rule (X length)** form ready to accept data relevant to the selected type of attribute. In the **Expression** text-box, enter the following expression:

CDPR LENG

This means: 'Set the X-length to the value of that property of the current template whose key is LENG' (as defined in Step *14.*). This expression will be explained more fully in the next section.

To see the effect of this rule, click the **Text Expression** button and study the result. Because no real design data has yet been created, the X length of the box has been set by the rule using the default value of the LENG property (which we specified as 1500).

Click **OK** to accept the rule. Notice how it is now shown in the **Rules** list on the **Template Member Parameters** form.

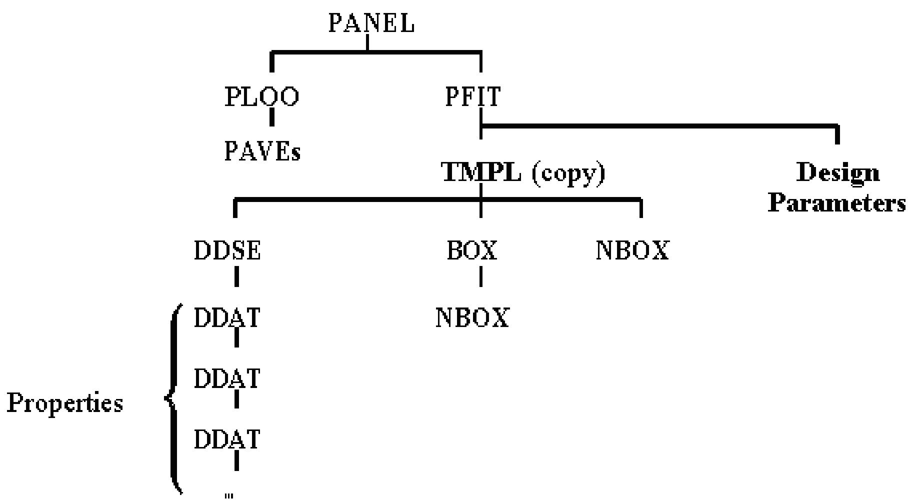
3.3 How Template data is Accessed in the Design Hierarchy

In order to understand how template data is accessed via specific cross-references in the Design database, we will consider what happens when the kickplate template which we are creating is used in the design of a panel.

To use the template, the structural designer will create a Panel Fitting (PFIT) element at the required position on the Panel and will then set the PFIT's Specification to point to the Design Template.

Note: To make the template accessible to the designer, it must have been included in a Specification in a Steelwork Catalogue database. We will see how to do this later in the exercise. The designer then selects from the specification in the usual way; there is no distinction between a catalogue component and a design template as far as the designer is concerned.

When the Specification Reference of the panel fitting is set to the design template, the TMPL is copied into the PFIT's members list (that is, an instance of the template is created automatically). The resulting Design hierarchy includes the following:



Consider the following:

- From PFIT level, you can query the properties data held in the DDATs of its TMPL by reference to a pseudo-attribute PROP whose setting is identified by the corresponding short key. For example, to query the X-length (which has the short key LENG), the command is Q PROP LENG. The system navigates down automatically via the TMPL and its DDSE to find the required property. **Query>Template Properties** lists all such properties.
- From BOX level, you can query the properties data held in the DDATs of its parent TMPL by reference to a pseudo-attribute CDPR (**C**urrent template's **D**esign **PR**operty). The system navigates up to the owning TMPL and then down via that template's DDSE. It is this setting which we used in the expression for the X-length rule in Step *17.*

3.4 Defining More Template Rules

We will now continue by defining the remaining rules for the Box.

Exercise continues:

18. Select **Y length** in the **Rules** list and define its rule as **CDPR WIDT**.

19. Select **Z length** in the **Rules** list and defined its role as CDPR HEIG.

20. Leave the **Orientation** rule unset (i.e. **None**). This means that the orientation must always be set by the designer to suit the local requirements when the template is instanced.

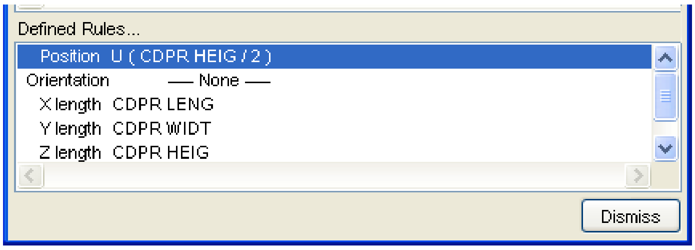
21. Select **Position**. Notice how the **Attribute Rule** form has attribute type POSITION at the top of the form. This indicates that the rule entered must be a valid position rule.

Remember that the box origin is at its centre. We want to set the position such that the kickplate assembly is positioned with the bottom face of the box coplanar with the surface of the panel. To do this, we must move the box up by half its height with respect to the PFIT.

Define the positions rule as U (CDPR HEIG / 2) (note the spaces before and after the / operator).

OK the rule definition.

The **Defined Rules** list for the Box should now appear as follows:



22. In the **Template Members** list on the **Template Members Parameters** form, select the **BOX**.

    Click the **Copy Rules** button. Now select the NBOX owned by the BOX (indented relative to the Box in the list). Now click the **Paste Rules** button to copy rules from the BOX to NBOX.

    The required X length of the NBOX (/Kickplate-Inner) is equal to the X length of the Box (/Kickplate-Outer) minus twice the wall thickness (see diagram in *Defining the Template Properties*). To achieve this, edit the X length rule to

    CDPR LENG - 2 * CDPR WTHK

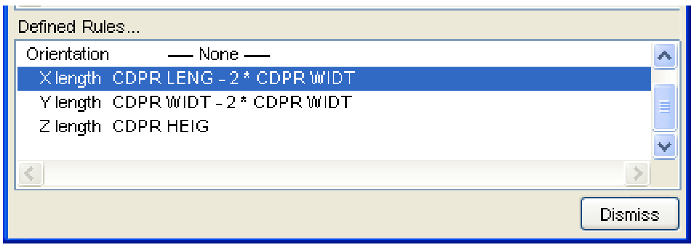
23. Set the Y length rule to

    CDPR WIDT - 2 * CDPR WTHK

    Set the rule for the **Z length** of the NBOX so that it has the same height as its owning Box.

    Because the NBOX is positioned with respect to its owning Box, its elevation will already be correct: you do **not** have to move this one by half its height. Therefore, leave both the **Position** and **Orientation** rules unset.

    The **Defined Rules** list for the NBOX should now appear as:



    The Box and NBOX together represent a rectangular kickplate assembly positioned on the surface of the panel. In the next step we will set the rules for the second NBOX, owned by the TMPL, which is to penetrate the panel below the kickplate assembly.

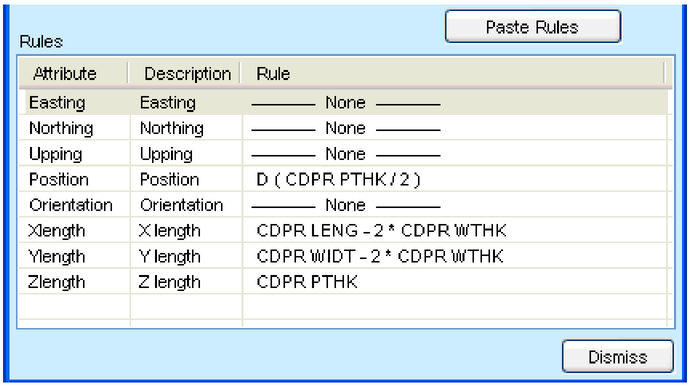
24. Set the rules for the **X length** and **Y length** of the second Nbox to be the same as those for the first NBox.

    We want to set the **Z length** of the second NBox to the thickness of the panel which it is to penetrate. To achieve this, set the **Z length** rule to **CDPR PTHK** (remembering that PTHK is itself derived from the panel thickness, as specified by the property definition).

    The position of the second NBox is to be such that its upper face is coplanar with the lower face of the Box, which means that it must be moved **down** by half its height. Set an appropriate rule to do this.

    The **Defined Rules** list for the second NBox should now appear as:

:



That concludes the definition of the kickplate design template. Check the **3D View** to ensure that the geometry looks correct. If not, correct the errors before finally saving the design.

In the next chapter we will add the template into a catalogue database specification for panel fittings, so that it can be referenced for inclusion in a structural design model.

4 Selecting a Design Template

In order to make a design template available for selection by a user, for incorporation (as an instance) into a design model, the template must be referenced from a Catalogue Specification or from a Selection Table. This chapter summarises how to achieve this.

4.1 How a Template is Accessed via a Specification

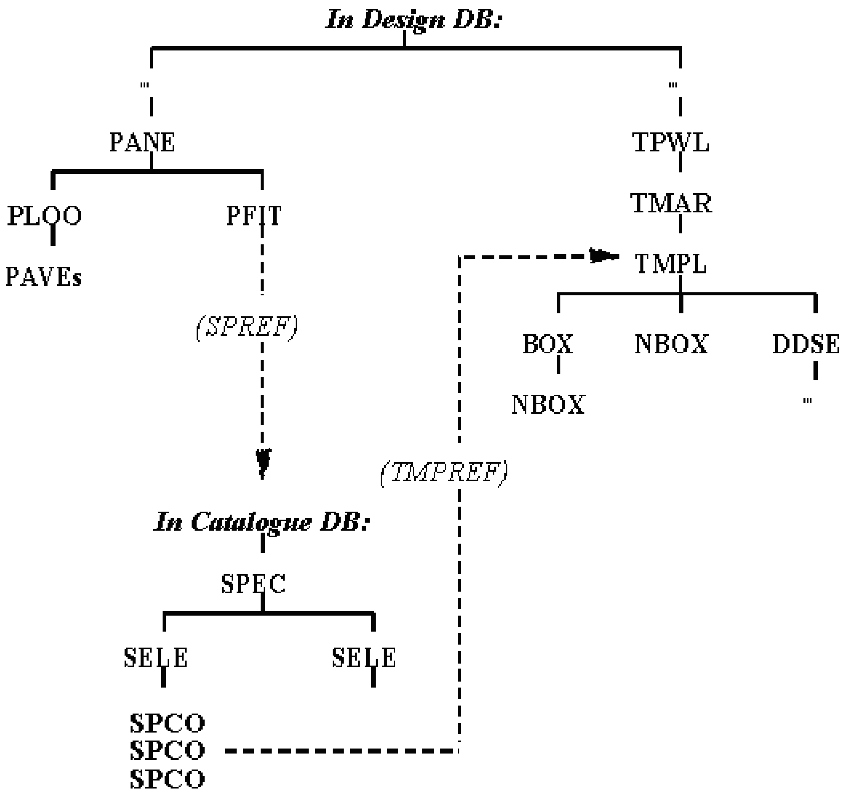
When a designer selects a catalogue component via a Specification (SPEC), the specification Selectors (SELEs) are searched in turn, using a question/answer sequence, until a Specification Component (SPCO) is found which matches all of the specified design criteria. The catalogue component selected is the one to which the Catalogue Reference attribute (CATREF) of that SPCO points.

Following exactly the same principle, a SPCO can refer instead to a design template by setting its **Template Reference** attribute (TMPREF) to point to a TMPL.

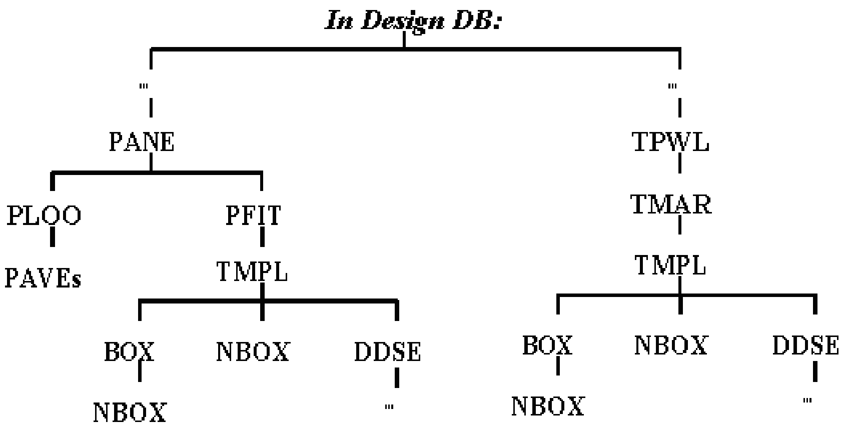
The only differences are:

- The catalogue component is stored in the Catalogue database, whereas the design template is stored in the Design database.
- When a catalogue component is selected, only the CATREF setting (or, more strictly, the SPRE setting) is stored in the design data. When a design template is selected, an instance of the design template is copied by the application into the design data, adding new elements into the design members list.

This can be illustrated as follows, using the selection of our kickplate template for addition to a panel (in the form of a panel fitting) as an example:



The SPREF of the PFIT leads to a SPCO which has its TMPREF pointing to the TMPL. This causes an instance of the TMPL to be copied below the PFIT, leading to:



4.2 Setting a Template Reference in a Specification

We will now enter the Catalogue Construction module, PARAGON, and modify an existing Specification by adding a new Specification Component. We will then set the TMPREF of this SPCO to point to our newly created design template.

Exercise continues:

25. If you are still in Design, select **Design>Modules>PARAGON...** to change to PARAGON. If you have left PDMS, log back in to project SAM as user CATS, but this time load module PARAGON from macro files. You will see the **PARAGON General Application** menu bar and the **Members List**.

The PARAGON graphical user interface does not include facilities for modifying specifications, so select **Display>Command Line** to display a **Command Input & Output** window.

Using the **Catalogue Explorer**, we will navigate to the existing SELE in which we will create the new SPCO

26. Go to the SPWL **/STRUCT/PENI/SPEC**.

From the main menu bar, select **Query>Attributes** to list the attributes of the current Specification World within the **Query** form. Note that the **Description** is set to **Structural** and the **Purpose** is set to **STL** (Steel), showing that these specifications are intended for use in structural designs.

27. Go to the SPEC **/PENH/PIPE/PFIT**.

This specification is intended for panel fittings which represent penetration holes through which pipes can pass.

28. Go to **SELE 1** (there is only one SELE under this SPEC).

29. Go to **SELE 3**.

You will see that this SELE already holds two SPCOs:

/PEN/PFIT/PENH/FITT/RECT
/PEN/PFIT/PENH/FITT/RECT2

Go to either of the SPCOs and query its attributes (if you still have the **Query** form on display the SPCO's attributes will automatically appear within it, otherwise select Query**>Attributes** again). Note that its Catalogue Reference (Catref) is set to point to a catalogue component (**/PENH/FITT/RECT or /PENH/FITT/RECT2**), while its Template Reference (**Tmpref**) is not set (**Nulref or =0/0**).

30. To create a specification component for selecting the kickplate template, type (on the command line)

NEW SPCO /TMPL/SAMPLE/KICKPLATE

To set its Template Reference, type

TMPREF /Kickplate-1

(this is the name which we allocated to the template in Step *10.*).

Query the attributes of the new SPCO, checking that its Catref is set to **Nulref** (or **=0/0**) and its **Tmpref** is set to **/Kickplate-1**.

31. Save your catalogue changes and exit from PARAGON. You can either change back to Design if you want to continue with the exercises, or leave PDMS if you want to take a break.

That completes the first introductory exercise for the creation of a design template. The template now exists in the Design database and a new specification component, pointing to the template, has been added into the Catalogue database.

If you want to see the effect from a designer's point of view, enter the Design Panels & Plates application and create a Panel Fitting by selecting the kickplate from the available Fitting specifications. If you have not yet used this application, it is suggested that you work through the exercise in *Structural Design Using PDMS User Guide*, but select the new template rather than the suggested manhole when you reach the relevant part of the exercise.

The next chapter will introduce some additional concepts, illustrating them with a more complex example.

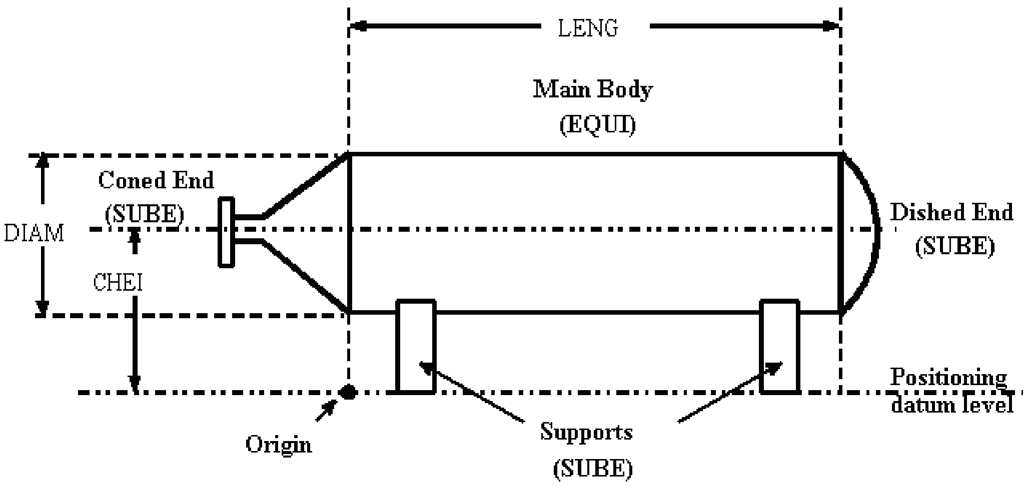
5 A More Advanced Example

To illustrate the principles of design templates more fully, particularly the ways in which rules defining property values can be specified, this chapter continues the exercise using a more complex example.

5.1 Building up a Design from Subsidiary Parts

In the preceding part of the exercise, the whole of the design model represented by the template was derived from a single Equipment element which owned three member primitives. For more complex items, it is often better to divide the design into subsidiary parts (represented by, say, Sub-Equipment elements), each of which is referenced from a parent design template.

Consider, for example, a cylindrical vessel with selectable end configurations and a set of supports. If the end configurations and the support set are defined as subsidiary designs, they can be re-used for a range of vessels. We will create a template for the following equipment design (typical end types shown for illustration only):



Exercise continues:

32. Restart PDMS if necessary and load the Design Equipment application.

Navigate to the Zone which you created for these exercises (i.e. TMPLZONE).

33. Create an Equipment element named **Test-Vessel** comprising:

- a **cylinder** with its **Z-axis** pointing **North** (i.e. the cylinder will be horizontal), and with **Height (Length) 1500** and **Diameter 400**;

- a **negative cylinder** with its **Z-Axis** pointing Up (remember that this orientation is defined with respect to the owning cylinder, whose Z/Up direction is already set to North), and with **Height (Length) 1500** and **Diameter 380**.

34. Change to the Design Templates application.

In the **Template Browser**, navigate to the Template Area which you created for these exercises (i.e. **Template Area for exercises**). In the **Design Explorer**, navigate to the Equipment on which the template is to be based (i.e. **Test-Vessel**).

Select **Create>Template** to create a design template copied from the equipment.

Set the following **Template Information**:

Name	Vessel-Main-Body
Purpose	Unset
Description	Vessel main body
Function	VESSEL
Generic Type	VESS

Leave the **Origin** at (0,0,0)

Select both **Add Properties** and **Add Points** options so that a Design Dataset (to hold design properties) and a Pointset (to hold design points) will be created ready for later use.

Click **OK** to complete the template creation. Select **Display> Selected Template** to show the new template in the **3D View**.

35. Select **Modify>Property Definitions** to display the **Define Template Properties** form (remember that you created the Design Dataset in the preceding step).

We will first create those properties which define the dimensions of the main body of the vessel and its position. We will later create some other properties which identify the subsidiary items which are to be merged with this body in the final design; namely the sub-equipments representing the vessel ends and the supports.

Before we begin, we will look at ways of restricting the options available to a designer when they create a template instance.

5.2 Restricting Property Values for use in a Design

When we set the properties for the kickplate, in the earlier steps of the exercise, we did not impose any restrictions on the dimensions which a designer could apply when using the template. The design parameters representing the kickplate's length, width, height and wall thickness could be set to any values.

As a template designer, you can restrict the values which a user can set in one of two ways. These are controlled by the **Range/Values** option near the centre of the **Define Template Properties** form, and act as follows:

- **Range** lets you specify minimum and maximum values for a design parameter. These are stored in the MinMax attribute of the Design Data (DDAT) element corresponding to that property. When used in a design, only values in the specified range will be accepted.
- **Values** lets you specify a list of discrete values for a design parameter. These are stored as the ValidV attributes of Valid Value (VVALUE) elements owned by the DDAT corresponding to that property. When used in a design, only these values will be available for selection from a list.

We will use both of these methods in the following steps.

Exercise continues:

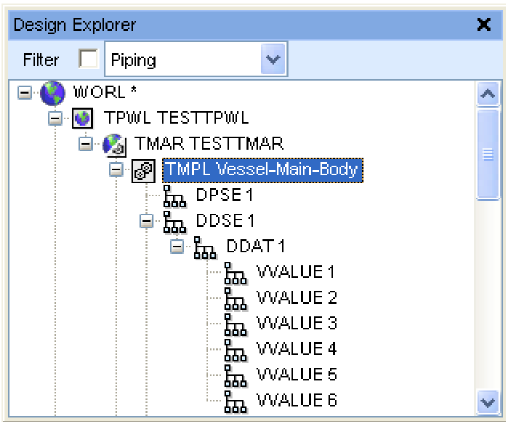
36. We will set the first property to represent the outside diameter of the vessel body. Enter the following data:

Description	Diameter of Vessel
Key	DIAM (it is good practice to use common keys in all equipment and sub-equipment templates to make their rules consistent)
Definition/No	Design Parameter 1
Data Type	Distance
Default	1000 (making this larger than the original equipment diameter of 400 lets you see the change take effect in the **3D View**)

We will restrict the user's choice of diameter values to some explicit dimensions. To do so, select the **Values** option and click the button to the right of that row of gadgets. The resulting **Set Valid Values** form contains a text-pane into which you can type your list of valid diameters. Enter the following values:

400 500 1000 1200 2000 4000

either separated by spaces or on separate lines. Note that this list should include the default value (in this case 1000), otherwise a designer cannot easily reset this if they have changed it. Click **OK** to store the valid values data, then click the **Include** button on the **Define Template Properties** form to add the property into the **Currently Defined Properties** list. Navigate to the TMPL /Vessel-Main-Body in the **Design Explorer** and check that its DDSE owns a DDAT which owns six VVALUEs (one for each valid diameter).



37. The next property will represent the length of the vessel body. Enter the following data:

Description	Length of Vessel
Key	LENG
Definition/No	Design Parameter 2

Data Type	Distance
Default	2000

For this property we will restrict the user's choice of length to any value between specified limits. To do so, select the Range option and enter **1000** and **4000** in the **From** and **To** fields.

Note: Changing to **Range** automatically clears any **Values** data stored by the **Set Valid Values** form; otherwise this data remains in force until you delete it. Take care not to set new properties to the current valid values data unintentionally.

Include the new property in the **Currently Defined Properties** list.

38. Create a third property representing the wall thickness of the vessel body. Enter the following data:

Description	Thickness of Vessel
Key	THKN
Definition/No	Design Parameter 3
Data Type	Distance
Default	20

This property will have no restrictions on its values, so keep the **Range** option selected and delete the entries from the **From** and **To** fields.

39. Create a fourth property representing the height of the vessel's centreline above the floor (the latter level will be correspond to the template's origin; see diagram at the beginning of this chapter). Enter the following data:

Description	Centreline Height
Key	CHEI
Definition/No	Design Parameter 4
Data Type	Distance
Default	600

This property will have no restrictions on its values, so leave the **Range** option as it is.

5.3 Adding Design Points

To provide convenient reference points for the relative positioning of items which make up the design template model, a design point can be created at any required location.

A design point, owned by a Design Pointset (DPSE), can be specified in three ways:

- A Cartesian Point (DPCA) is specified in terms of its coordinates only.
- A Cylindrical Point (DPCY) is specified as a position on the surface of a cylinder at a given position and with given dimensions.
- A Spherical Point (DPSP) is specified as a position on the surface of a sphere at a given position and with given dimensions.

Each design point, like a p-point, has both a position and an orientation. (The on-line help includes diagrams illustrating these concepts; refer to these if you need further clarification.)

For convenience when positioning the sub-equipments which will represent the ends of the vessel, we will create a design point at the centre of each end of the vessel body.

Exercise continues:

40. Set the view direction to **Look>East**. Check that the Vessel Main Body template is the current element. Notice that this already owns an empty Design Pointset (DPSET 1). Select **Modify>Points**. You will see a **Design Points** form showing that no points have yet been defined.

41. From the **Design Points** menu bar, select **Create>Cartesian Point**. On the resulting form enter the following data:

Number	1 (set automatically)
Description	Datum Point 1
Function	SUBE
Point Direction	
Direction of Design Point (Z)	N
Normal direction (Y)	E

(This orientation will not be relevant when positioning items relative to this design point, but is set here so that you will see the design point better in the **3D View**.)

We will position the design point by graphical picking. Click the **Pick** button, then pick **P1 of CYLINDER 1 ....**

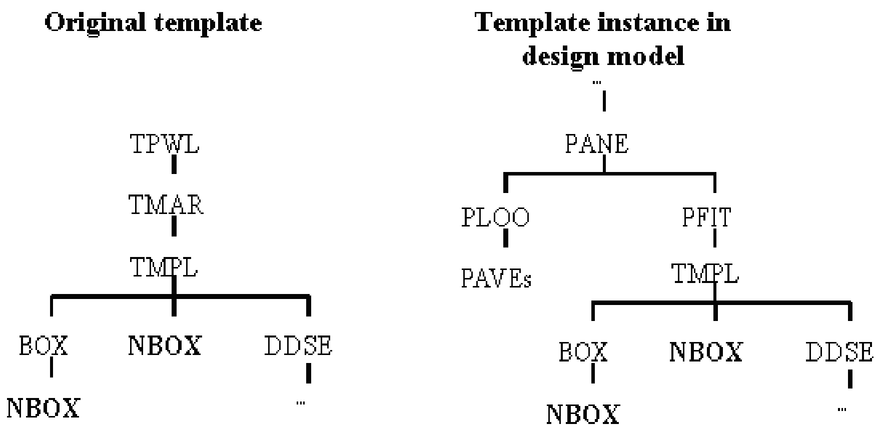
OK the **Create - Cartesian Design Point** form. Notice how the new point is shown in the list in the **Design Points** form and is also labelled (as **DP1**) in the **3D View**. To change the colour of the displayed design point, if it is difficult to see, select **Settings>Graphics>Colour** and change the **Aids** colour.

42. Create a similar design point, **Datum Point 2**, at the other end of the vessel body. The **Create Cartesian Design Point** form will have retained the previous settings, so you only need edit these slightly. The new point should have its **Z** direction set to **S** and should be positioned at **P2 of CYLINDER 1 ....**

We will pause here to introduce two new concepts which are relevant at this stage of the exercise.

5.4 Assigning Local Names to Template Elements

When a design template is copied into the design model, confusion can arise in the identification of members of the template instance. For example, the penetration/kickplate which we defined in the first part of the exercise included two negative boxes, thus:



These are identifiable in the design model only as NBOX 1 of TMPL 1 of PFIT and NBOX 1 of BOX 1 of TMPL 1 of PFIT. Even if we had named them in the design template, the names would not have been copied to the design instance because an element name cannot be used more than once in a Design database. If the design instance were modified by including other negative boxes, either or both of the original NBOXes might no longer be correctly identified as NBOX 1 and any reference to it in a rule could lead to an incorrect result.

To avoid such ambiguities in identifying members of a design template, these elements can be assigned local names. A local name is stored in the design template definition and is copied into every instance, so that references to local names within rules remain valid.

A local name can comprise up to 20 characters, and each template can store up to 500 local names. Duplicate local names are not allowed within a given design template, but the same local name may be used in any number of different design templates within a single Design database.

To assign local names, navigate to the owning TMPL and use the **Modify>Parameterisation...** menu option.

To refer to a local name in a rule or dataset expression, use the format LNID /localname to specify an element in the current template or MLNID /localname to specify an element in a template which is a member of the current element.

5.5 Specifying Priorities for Evaluating Rules

When the rules within a design template definition are executed (as, for example, when an instance of the template is copied into the design model), the rules for the member elements are evaluated, by default, in descending hierarchic order.

If, however, the rules for one element involve the current attributes of another element, the order in which the rules are executed becomes significant and the default sequence may not give the intended result. As an example, if the positions of elements /A and /B are both defined by rules, and if the position of /A is defined relative to that of /B, then the rule for positioning /A will only give the correct result if the rule for /B has already been executed.

You can control the order in which the rules for a given template will be executed by assigning a rule sequence number to any element which has a local name (see *Assigning Local Names to Template Elements*). When the template rules are re-executed, the rules for such elements will be executed in ascending order of their sequence numbers: the rules for elements without numbers will then be executed in the default sequence.

Thus, in the preceding example, you would assign a smaller sequence number to element /B than to /A, so that the rule for positioning /B is executed before the rule for /A.

To specify the sequence in which the rules for template members are to be executed, first allocate local names to the relevant elements, then use the **Modify>Order Local Names** menu option to sort the list of local names into the required order.

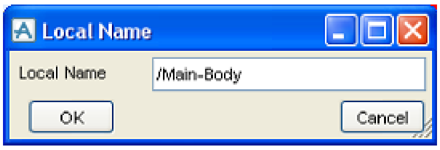
Exercise continues:

43. We will assign local names to the two design points and to the cylinder representing the main body of the vessel (the reasons will become clear when we set up the parameterisation rules).

Select **Modify>Parameterisation**. The **Template Member Parameters** form lists all members of the template.

Select **(1) DPCA** in the list and select the **Set Local Name** button. On the resulting **Local Name** form, enter **Datum-Point-1**.

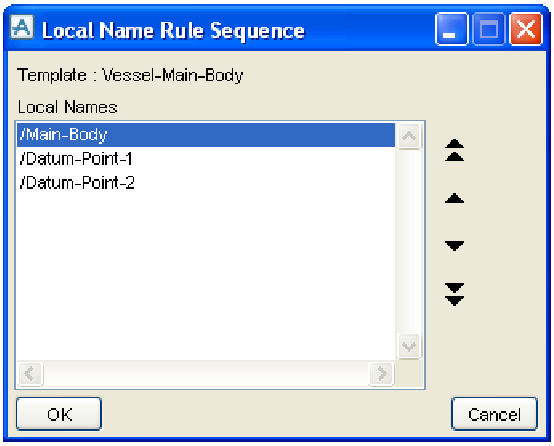
Similarly, assign local names **Datum-Point-2** to **(2) DPCA** and **Main-Body** to **CYLI 1**. You do not need to assign a local name to the negative cylinder. The result should be:



44. When the template rules are evaluated, the positions of the design points will be derived from the positions of the cylinder's p-points. It is important, therefore, that the rules for the cylinder are evaluated **before** the rules for the design points.

Select **Modify>Order Local Names**. The **Local Name Rule Sequence** form lists all elements with local names in the order in which their rules will be evaluated. By default, this is the order in which the names were assigned.

Select **/Main-Body** and click ⏫ to move this name to the top of the list, thus:



OK the change, then dismiss the form.

We will now set rules for the template parameters in terms of the properties defined so far. The effect will be to move the primitives and design points to the required positions relative to the template origin, as shown in the diagram at the beginning of this chapter.

45. Select **Modify>Parameterisation** to show the **Template Member Parameters** form. In the **Template Members** list, select **CYLI 1**. Notice how the current position and orientation of the cylinder are shown in the **3D View**. Its Origin is at its centre.

We want to move the cylinder so that Design Point 1 (at P1) is directly above the template origin, with a separation given by the Centreline Height (CHEI) property.

Select **Position** in the Rules table list and enter the following:

Easting	leave unset
Northing	CDPR CHEI
Upping	CDPR LENG / 2

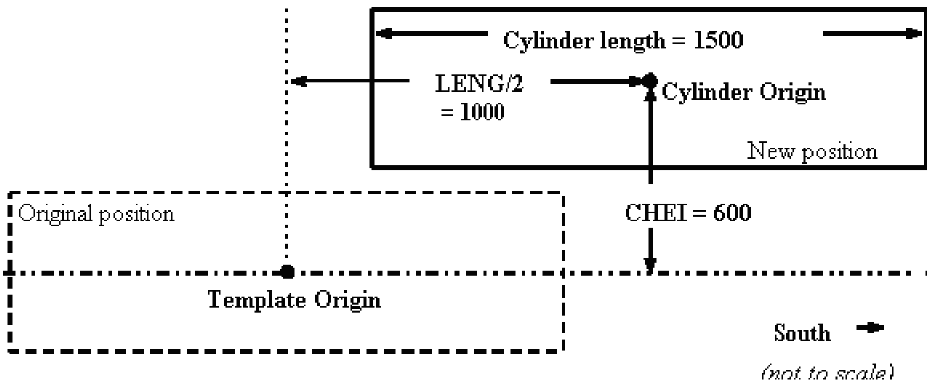
OK the change.

To display the origin and orientation for the overall design template, navigate to **/Vessel -Main-Body** in the **Design Explorer** and either select Query**>Axes** or click . On the **Define Axes** form, select **Cardinal Directions** to show **ENU** directions instead of **XYZ** directions. Select **Close>Retain Axes** to dismiss the form while still showing the axes.

Look carefully at the new position for the cylinder relative to the template origin: it may not be quite as you expected. We will now consider why this should be.

The effect of the last change was to move the cylinder up by the CHEI dimension and south by half the LENG dimension. Remember, however, that until the template is instanced in a design, and the design parameters are set, the **default** values for these dimensions apply. The actual cylinder length is currently that of the original Equipment from which the template was copied.

Consider the position along the cylinder's axis, looking East. Its length, defined by its Height attribute is 1500. The parameterisation rule has moved the cylinder along its axis by half the default LENG, which was set to 2000:



46. Select **Height** in the **Defined Rules** list and enter the rule **CDPR LENG**. Notice that because this setting increases the length of the vessel to the default value of LENG (namely 2000), the effect shown in the preceding diagram is corrected.

47. Set the rule for **Diameter** to **CDPR DIAM.** Leave the **Orientation** rule unset, so that the cylinder's orientation always defaults to that of its owning equipment when the template is instanced in a design.

48. Now select **NCYL 1** in the **Template Members** list and set appropriate rules for its **Position, Height** and **Diameter**.

Did you remember that the position of the negative cylinder is defined with respect to that of its owning cylinder (i.e. at 0,0,0)?

49. Select **(1) DPCA** and set a rule for its **Position** as follows:

- On the **Rule Definition (Position)** form, set the expression to:

  AT PP1 of LNID /Main-Body

  which positions the design point at a p-point of the locally-named cylinder (see *Assigning Local Names to Template Elements* for an explanation of why a local name is required here).

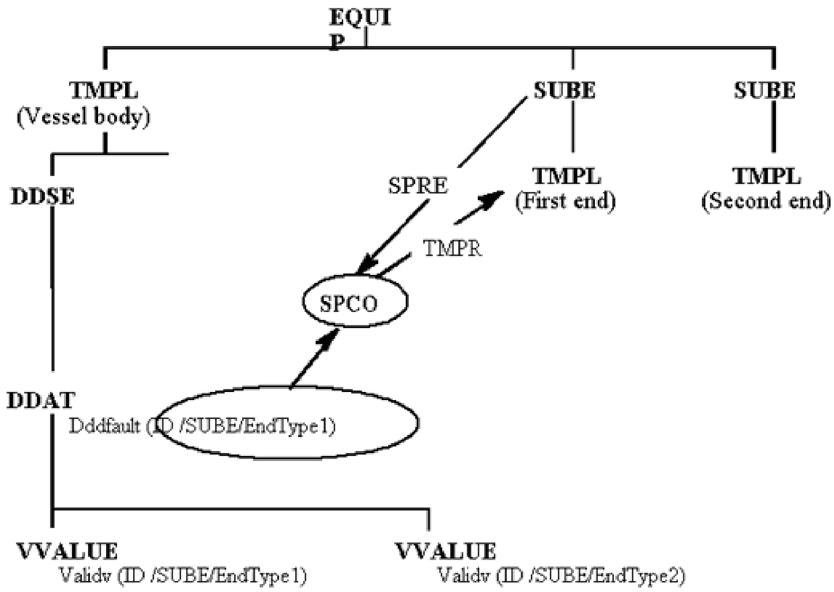
50. Repeat this sequence to set the **Position** of the second design point **(2) DPCA to PP2** of the same cylinder.

51. Switch to wireline mode, or set the graphical representation to shows holes drawn, and check that the positive and negative cylinders correctly represent the vessel body. When you are satisfied, dismiss the parameterisation form.

5.6 Setting References to Sub-equipments

The vessel ends for the equipment represented by our template will be defined in the design by setting specification references which point to other templates based on appropriate sub-equipments. For the purpose of this exercise, we will use some existing ends which are supplied as part of the sample project. In practice, you would usually have designed this yourself using the same methods as for the overall equipment, and would have included them in your catalogue specifications.

When instanced in a design, the resulting selection route would be as follows:



Each Valid Value (VVALUE) under the data element stores a reference to a Specification Component (SPCO) which has its Template Reference (TMPR) pointing to a TMPL representing a specific vessel end (as illustrated for the default setting).

The Sub-Equipment (SUBE) elements shown in the hierarchy are created under the Equipment automatically when the properties are set for the design template instance.

The equipment template which we are creating in the exercise will have two possible design types (that is, two valid values) for each end: the 'closed end' will have either a dished end or a flanged end; the 'open end' will have either a dished end or a conical end.

Note: Because the rules which define the vessel end types will not be executed until the template is instanced in a design, you will not be able to see the complete vessel while you are working in the Design Templates application. It is important, therefore, that you set the rules very carefully.

Exercise continues:

52. Navigate to **/Vessel-Main-Body** and select **Modify>Property Definitions** to show the **Define Template Properties** form again.

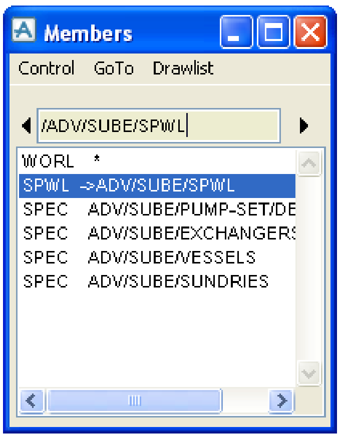
Define the property for the first end as follows:

Description	Closed End
Key	CEND
Definition	Sub Element Ref
Data Type	Sub Ref (set automatically)
Default	(We will come back to this a little later)

We will restrict the settings for this property to a choice between two end configurations. The **Values** option is set automatically (since a range of references would be meaningless), so click on to display the **Set Valid Values** form. The data which you must enter into the **List of Valid Values** field are the names of the SPCOs which point to the required end types. If you can remember them, you can type the names in directly; otherwise, you can navigate to them using the **Members** list and use the **Add CE** button to copy their names into the text pane. We will use the latter method.

The **Members** list can be displayed using **Display>Members** from the top menu bar

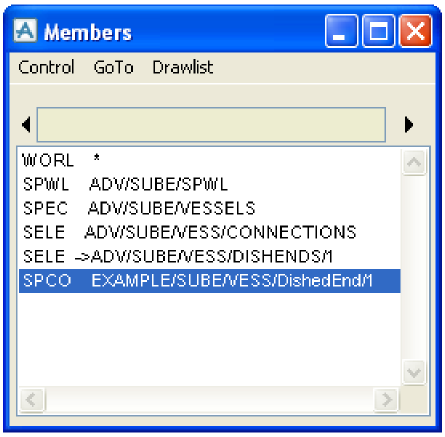
Because we are working in the Design database and the specifications are stored in the Catalogue database, you will not see the required data in the current **Members** list. We must navigate to the relevant Specification World (SPWL), which was included in the sample data. In the text-box at the top of the **Members** list, enter **/ADV/SUBE/SPWL** and press **Enter** to navigate to the named element.



Navigate down the specification hierarchy in the following sequence:

SPEC/ADV/SUBE/VESSELS
SELE/ADV/SUBE/VESS/CONNECTIONS

SELE/ADV/SUBE/VESS/DISHENDS/1
SPCO/EXAMPLE/SUBE/VESS/DishedEnd/1



This will be the first entry in our valid values list, so click the **Add CE** button on the **Set Valid Values** form.

Navigate up two levels (to **.../CONNECTIONS**) and then:

SELE/ADV/SUBE/VESS/FLGENDS/1
SPCO/EXAMPLE/SUBE/VESS/FlangedEnd/1

Click **Add CE** again to make this the second valid end type.

We must next assign the rules defining the position and orientation of this vessel end, so do not click **OK** yet.

53. The end types which we will be using have all been defined in the catalogue such that the origin is at the centre of the face which adjoins the vessel and the Y direction points away from the vessel. For example, for the dished end:

Note: When you design sub-equipment templates such as these, it is important to maintain strict naming, positioning and orientation conventions, otherwise you will cause confusion when your templates are used in a design.

On the **Set Valid Values** form, click the **Rule Definition** button.

On the **Associated Rule Definition** form, enter:

Description	Position
Purpose	Sub-Equipment Reference (set automatically)
Attribute	Sub-Equipment Position

We want to position the origin of this sub-equipment at the midpoint of one end of our vessel body. Instead of calculating this position, we will refer to the locally-named design point which we placed here for just this purpose in *Adding Design Points*. We will use the second Design Point for the closed end, so enter the **Associated Rule** as:

AT LNID /Datum-Point-2 of TMPL 1 of EQUIP

Click Include to add this into the **Rules Defined** list.

Define the next rule using the following settings:

Description	Orientation
Purpose	Sub-Equipment Reference (set automatically)
Attribute	Sub-Equipment Orientation

Enter the **Associated Rule** as:

Y is S and Z is U

and include this in the list.

OK the **Associated Rule Definition** form, then **OK** the **Set Valid Values** form.

54. We will now set the **Default** on the **Define Template Properties** form. Set this to the SPCO for the dished end, namely **/EXAMPLE/SUBE/VESS/DishedEnd/1**. You can either type this in directly, or redisplay the **Set Valid Values** form temporarily and use a copy/paste operation.

Include the property in the dataset list.

55. Repeat Steps *52.* to *54.* to achieve the following settings.

Define Template Properties form:

Description	Open End
Key	OEND
Definition	Sub Element Ref
Data Type	Sub Ref
Default	/EXAMPLE/SUBE/VESS/ConedEnd/1

Set Valid Values form:

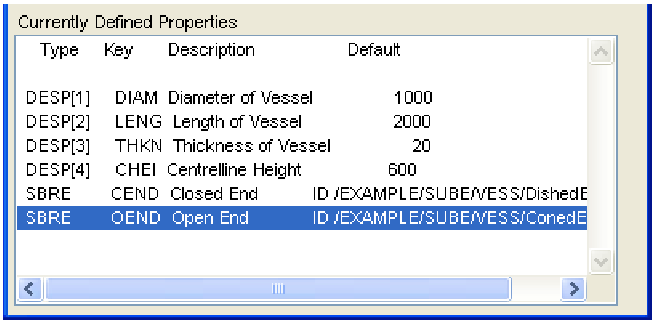
/EXAMPLE/SUBE/VESS/ConedEnd/1
/EXAMPLE/SUBE/VESS/FlangedEnd/1

Associated Rule Definition form:

Position	AT LNID /Datum-Point-1 of TMPL 1 of EQUIP
Orientation	Y is N and Z is U

(The latter is the default orientation, so you could leave this rule unset. It is, however, good practice to set such rules explicitly to avoid any uncertainty.)

At this point, your defined template properties should be:



We will pause here to see how the rules which we have just defined for positioning and orientating the sub-equipment templates are stored in the Design database.

5.7 How Rules Associated with Valid Values are Stored

Look at the Design Dataset (DDSE) members for the current template in the **Members List.**

Note: **Hint:** For a quick way of returning the focus of the **Members** list from the Catalogue database to the Design database, check that **Navigate on selection** is selected in the **Template Browser** and reselect the current template in the browser's list.

You will see that the Design Data (DDAT) elements holding the properties for the end types (probably DDATs 5 and 6) each own two Attribute Rule (ATTRRL) elements, in addition to the VVALUE elements which store the reference pointers to the end type specifications. These Attribute Rules store the associated rules, in this case for position and orientation, in their AttRule attributes.

When the template is instanced in a design, these associated rules are executed and the sub-equipments which they define are positioned and orientated accordingly.

5.8 Adding Vessel Supports

We will next add some user-selectable supports to the vessel body. These will be specified in exactly the same way as for the vessel ends, namely by setting a property which includes specification references to some pre-defined sub-equipment templates representing different support configurations.

Exercise continues:

56. Redisplay the **Define Template Properties** form and use the following property settings.

Define Template Properties form:

Description	Supports
Key	SUPP
Definition	Sub Element Ref
Data Type	Sub Ref
Default	/STD/2-Footed-Box/CL/TYPE

Set Valid Values form:

/STD/2-Footed-Box/CL/TYPE
/STD/3-Footed-Box/CL/TYPE

If you want to check these, you will find them in the Catalogue database under the following hierarchy:

SPWL/ADV/EQUIPMENT/SUPP/SPWL
SPEC/ADV/EQUIPMENT/SUPPORTS
SELE/ADV/EQUIPMENT/HORIZONTAL/VESS/SUPPS
SELE/ADV/EQUIPMENT/HORIZONTAL/VESS/SUPPS/1

Associated Rule Definition form:

The catalogue definition for each of these supports has the support position at the origin of the Equipment and the orientation the same as that of the Equipment, so no associated rules are required. You must, therefore, remove any associated rules which may already be set. The dimensions of the support will be derived from the design parameters of the owning Equipment, so that the overall support will be resized automatically to suit the dimensions of the vessel body.

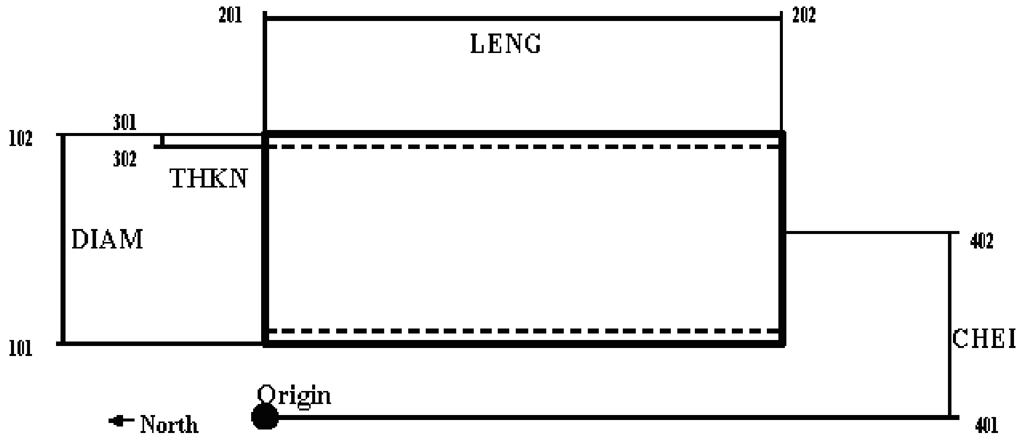
That completes the definition of the equipment template's geometry. We will finish the design by adding some dimensioning annotation which can be displayed by the user when the template is instanced in a design model.

5.9 Showing Dimensions on Template Designs

The Design Templates application incorporates a facility for creating pairs of Dimension Points (actually spherical design points; see *Adding Design Points*), for use as datum points for displaying design template dimensions. Each dimension point can have its position and associated projection line defined by a set of rules.

If you are dimensioning a new template, where no dimension point rules have yet been defined, an 'autobuild' function can be used to create pairs of dimension points automatically for all design parameters.

The dimensions which we will add to our equipment template, based on the design parameters specified in Steps *36.* to *39.*, are as follows (the numbers will be explained in the next steps):

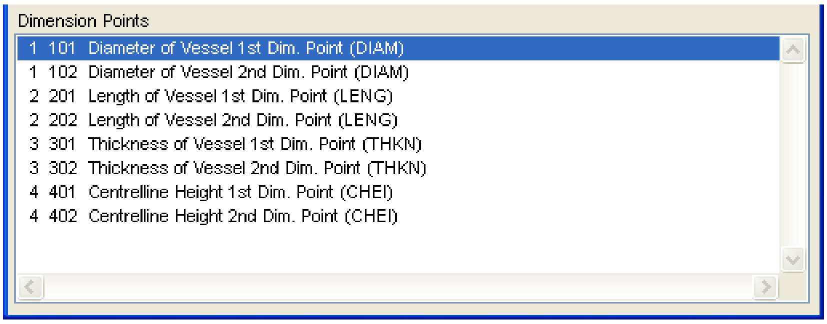


While each dimension point position and projection line extension could be defined explicitly, this could lead to an inappropriate display if, say, the vessel were made large enough to obscure them. It is best, therefore, to specify the positions and extensions in terms of rules based on the design parameter values. The dimensioning annotations are then scaled automatically as the vessel dimensions are modified.

Exercise continues

57. With **/Vessel-Main-Body** as the current element, select **Create>Dimension Points**. You will see a **Dimension Point Definition** form which lets you specify the position (**Dimension Position**), extension line length (**Dimension Extension**) and extension line direction (**Dimension Direction**) for each dimension point in each pair.

From the **Dimension Point Definition** form's menu bar, select **Control>AutoBuild**. This will create two dimension points for each of the four design parameters, as shown in the **Dimension Points** list, thus:



The points are numbered in pairs (as shown in the preceding diagram), and are given descriptions which include the design parameter DKEYs; for example, points **101** and **102** form a pair which will show the DIAM dimension.

When first created, all dimension points are positioned at the template origin (shown by the sphere in the **3D View**). What we now have to do is reposition each point and specify its projection line length and direction.

58. Select point **101**. Leave the **Design Point Description**, **Design Param No.** and **Dpoint No.** settings as they are (we will not change these for any of the points).

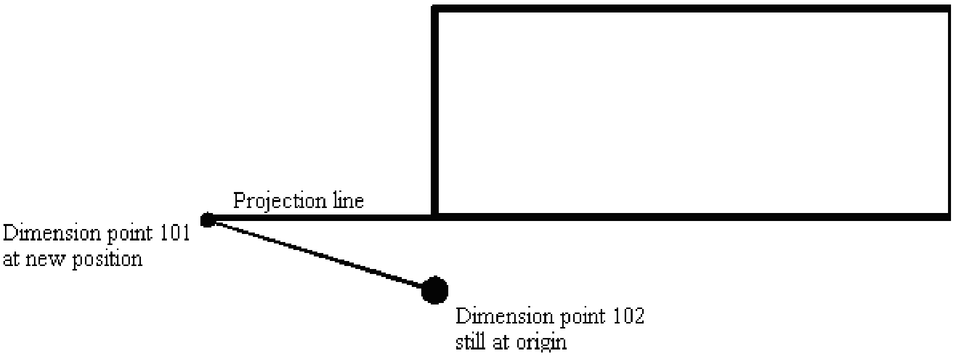
Click next to the **Dimension Position** field to display the **Position Rule Definition** 'expression builder' form. Build up the following expression and compare it with the design parameter definitions to ensure that you understand why it gives a suitable position for this point:

N (CDPR LENG / 2) U (CDPR CHEI - CDPR DIAM / 2)

(The expression builder may insert extra spaces and pairs of parentheses, but these are not essential.) The N (north) displacement, by half the length of the vessel body, should be sufficient to position the dimension point clear of the vessel's end in the completed design.

Set the **Dimension Extension** to (**CDPR LENG / 2**) and the **Dimension Direction** to **S** (south), so that the projection line from this point is directed towards, and just reaches, the end of the vessel body.

Click the **Replace** button to overwrite the data for point 101 with you new settings. The effect is as follows:

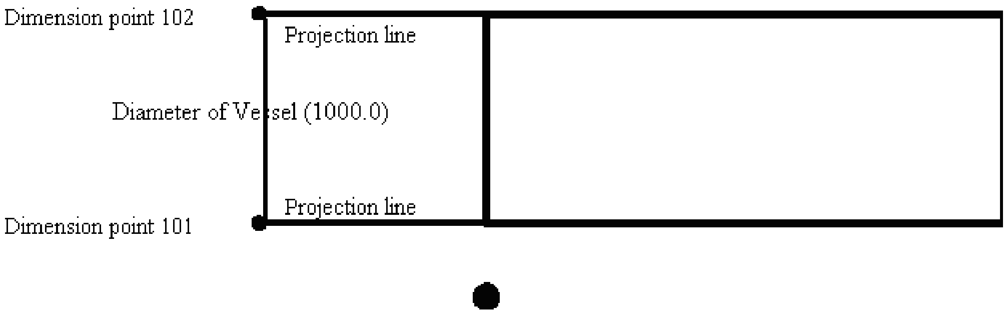


59. Select point **102**. This will differ from point **101** only in its **U** (up) coordinate, so, to save entering all the data again, click the **Copy Partner** button. This button copies all position and projection line settings from the other point in a pair; for example, from 201 to 202 or from 202 to 201.

Edit the **Dimension Position** rule by changing the - sign to a + sign, thus:

N (CDPR LENG / 2) U (CDPR CHEI + CDPR DIAM / 2)

and click **Replace**. The result should be:



60. We will set the **Thickness** dimension points next, since these will use some of the data from the **Diameter** points. (It is sensible to think about the order in which you edit dimension points, so that you minimise the amount of editing required.)

61. Select point **301**. The data for this point will be derived from the settings for point **102**, so click the **Copy Last Point** button. This button copies all position and projection line settings from the point which was selected prior to the current point (which in this case was point **102**).

62. We will position point **301** midway between point **102** and the end of the vessel body, so edit the rules thus:

Dimension Position	N (CDPR LENG / 4) U (CDPR CHEI + CDPR DIAM / 2)
Dimension Extension	(CDPR LENG / 4)
Dimension Direction	S

and **Replace**.

63. Set the rules for point **302** by using the **Copy Partner** facility.

The only rule change needed is to reduce the U(up) co-ordinate by the vessel's wall thickness, thus:

Dimension Position ... U (CDPR CHEI + CDPR DIAM / 2 - CDPR THKN)

64. We will next set the rules for the **Centreline Height** dimension points, positioning these at the opposite end of the vessel from the previous points for clarity.

Set the rules for point **401** as follows:

Dimension Position S (CDPR LENG * 1.5)

Dimension Extension (CDPR LENG * 1.5)

Dimension Direction N

Set the rules for point **402**, using **Copy Partner**, thus:

Dimension Position S (CDPR LENG * 1.5) U (CDPR CHEI)

Dimension Extension (CDPR LENG * 0.5)

Dimension Direction N

65. Finally, we will set the rules for the **Length** dimension points. We will position these points relative to the locally-named design points at the centres of the vessel body end faces.

Set the rules for point **201** thus:

Dimension Position U (CDPR DIAM * 2) from LNID /Datum-Point-1
(use the expression builder's Local Names feature)

Dimension Extension (CDPR DIAM * 1.5)

Dimension Direction D

Set the rules for point **202** thus:

Dimension Position U (CDPR DIAM * 2) from LNID /Datum-Point-2

Dimension Extension (CDPR DIAM * 1.5)

Dimension Direction D

When you are satisfied that all of the dimensioning rules are correct, select **Control>Close** to dismiss the **Dimension Point Definition** form.

5.10 Testing your Design Template

The only way to test the combined effects of all of the design template settings is to add a reference to the template into a catalogue specification (as you did for the kickplate, in *Selecting a Design Template*) and then to select an instance of the template for inclusion in a design model.

Exercise continues:

66. Change to the PARAGON module.

We will create a new Specification Component under an existing part of the Specification hierarchy. Navigate via the following route:

SPWL/ADV/EQUIPMENT/SPWL
SPEC/CADCENTRE-ADV-EQUIP
SELE CADCENTRE-ADV/VESSEL
SELE /ADV/EQUIPMENT/HORIZONTALS
SELE/ADV/EQUIPMENT/STORAGE-1

From the command line, create a new SPCO thus:

NEW SPCO /Exercise-vessel
TANS 'EXAMPLE-2'
TMPREF /Vessel-Main-Body

where the Template Reference attribute points to the equipment template which you created in the preceding steps.

67. Change back to Design, enter the Equipment application and, from a suitable hierarchic level, select **Create>Standard**.

On the **Create Standard Equipment** form, select **Specification: CADC Advanced Equip**. From the lower selection lists, select in turn:

CADCENTRE Advanced	"Advanced" Vessels
Vessel Type	"Horizontal" Vessels
Specific Type	"Horizontal" Storage Vessels with Dished Ends
Selection	"EXAMPLE-2" Vessel main body

Click the **Properties** button. Notice the following features of the property settings, each of which currently shows its default setting:

- **Centreline Height** (as defined in Step *39.*) and **Thickness of Vessel** (defined in Step *38.*) let you type in any value. No restrictions were set.
- **Length of Vessel** (as defined in Step *37.*) will only accept lengths in the range **1000** to **4000**. Only lets you select a specific dimension from the drop-down list.
- **Diameter of Vessel** (as defined in Step *36.*) only lets you select a specific dimension from the drop-down list.
- **Closed End** (as defined in Steps *52.* to *54.*), **Open End** (as defined in Step *55.*), and **Supports** (as defined in Step *56.*) each let you select appropriate sub-equipment items from the drop-down specification lists.

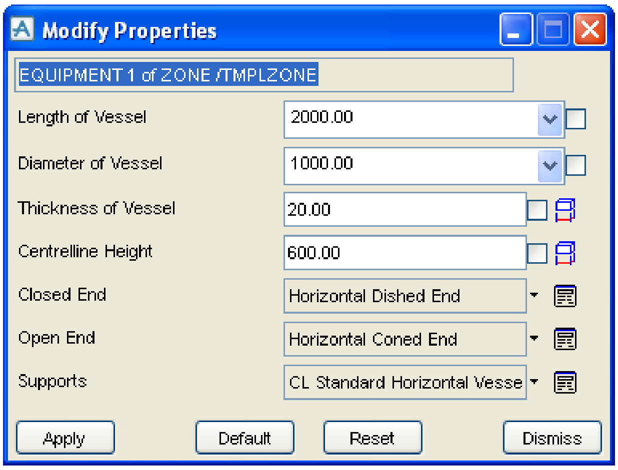
Leave all of the default settings in force for the first test of the template.

68. Click **OK** on the **Modify Properties** form, and then click **Apply** on the **Create Standard Equipment** form. Pick the position in the **3D View** at which the new equipment, based on your design template, is to be positioned.

This is the moment of truth! If you set every one of the design template properties and rules correctly, the new equipment will look similar to the diagram on the first page of this chapter. If it does, well done! If it does not, you must go back into the Design Template application and carefully check each setting.

69. When you are satisfied with the template geometry, redisplay the **Modify Properties** form in the Equipment application and change each property in turn to check that the results are all as intended.

70. The toggles (check boxes) immediately to the right of the four dimension fields on the **Modify Properties** form let you display the dimension points for the corresponding parameters (as defined in Steps *57.* to *65.*). Set all of these to **On** and check that each dimension is displayed correctly in the **3D View**.



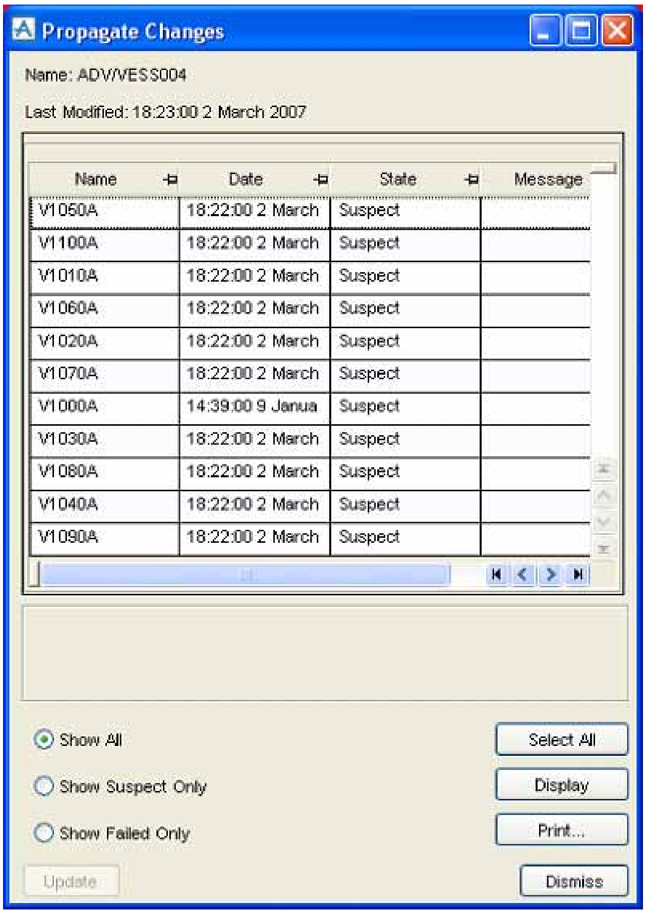
That completes the exercises. The final chapters introduce a few additional concepts and give some hints on good design practices for template designers.

5.11 Propogating Design Template Changes

The **Design Template Change Propagation** form is shown by using the Design Templates application **Utilities>Propagate Changes…** menu command.

The form below shows all linked instances of Design Template /ADV/VESS004. All of the instances are marked as **Suspect**. This means that the master Design Template has been modified either in the same session or in a subsequent session to the last modification to the Design Template instances, and consequently they should be considered for propagation of changes.

Note: Design Template instances may only be updated if the user has full access to modify each instance. If any instances are claimed by another user, or modification is prevented by Data Access Control then the update will fail.



In order to update Design Template Instances, first the user must select on the **Propagate Changes** form the Design Template instances to update. These may be selected by using the **Select All** button or by selecting one or more elements on the list.

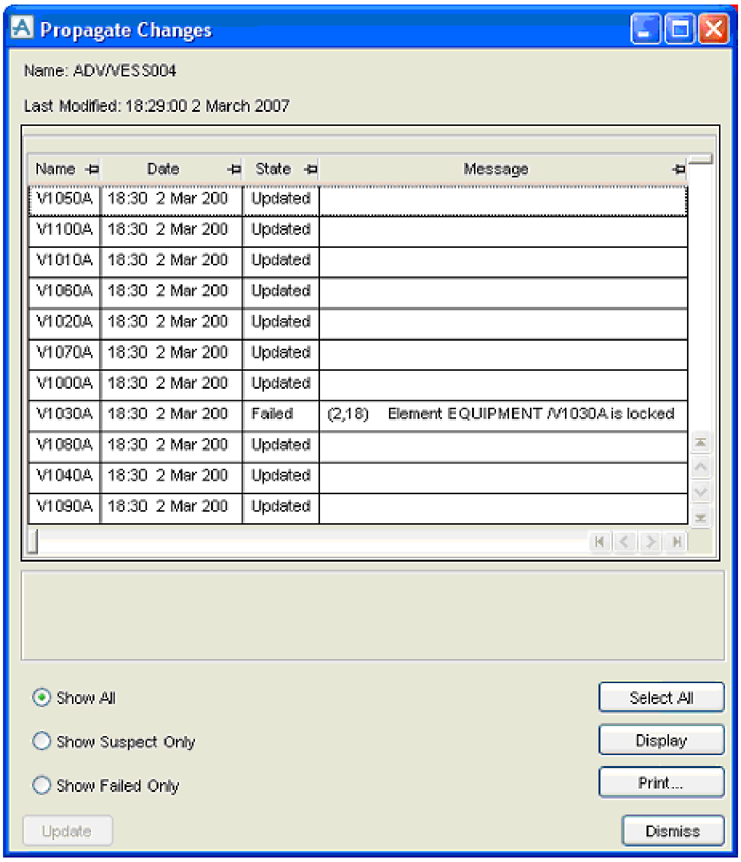
Pressing the Display button will pop up a view showing all of the selected elements.

To facilitate selection the list of Design Templates may be sorted accordingly to the State column.

You can also use the Print button to preview the current content of the form.

When Design Template Instances have been selected, the **Update** button is pressed to propagate changes to the selected elements.

During the Update operation, the system may find conditions where the Update function is unsuccessful, for example where Data Access Control has prevented the update from completing. The success or failure of the Update operation for each Design Template instance is shown on the form when the Update operation has finished. This will be indicated by the words Updated or Failed in the State column. If an update fails, a brief message is shown in the Message column and a more complete description of the problem appears in the Messages box below the table when the row containing the failed update element is individually selected.



Undo has been implemented for the **Update** command. Using the Undo command immediately after an Update will revert the model back to its state immediately before the update.

6 Some Further Information for Template Designers

This chapter gives some further information which could be relevant as you develop more complicated design templates.

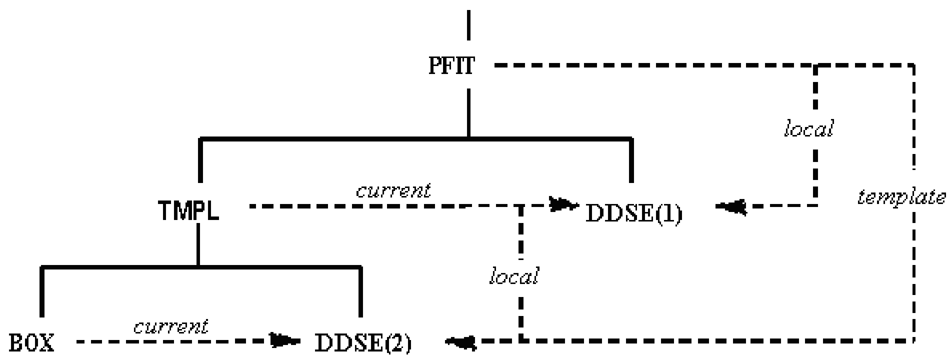
6.1 Using Pseudo-attributes for Accessing Data

(This section reiterates and expands the concept of extracting design parameter settings for use in template property rules.)

Design Datasets (DDSE) are used to store the properties of design template items, which may then be used to define the template's parameters. Each property is stored in a Design Data (DDAT) element under a DDSE, with each DDAT being identified by a keyword held in its DKEY attribute. The property definition can be specified by an expression held in the DDPR (Design Data Property) attribute, while a default value for the property (to be used if the expression cannot be evaluated for any reason) can be stored in the DDDF (Design Data Default) attribute.

Unlike a design reference to a catalogue component, which can access only a single catalogue dataset via its DTREF setting, a reference to a design template can access more than one design dataset. A local design dataset is owned directly by the current element, a template design dataset is owned by the first template below the current element, while a current design dataset is at the same level as the current element and has the same owner.

The following diagram illustrates the relative positions of these types of dataset for a simple design hierarchy, namely a Panel Fitting derived by instancing a Design Template which includes a Box primitive:



The full range of pseudo-attributes available for accessing the properties data in the various types of dataset in the Design database are as follows:

	Current Dataset	Local Dataset	Template Dataset
List of property keys	CDPL	DEPL	PRLS
Real property values	CDPR	LDPR	TDPR
Text or real property values	TCDP	DEPR	PROP
Text or real property default values	TCDD	DEPD	PRDE
Reference settings	CFDP	LFDP	TFDP

You can also derive data values from the parameters of items in the Catalogue database by using the RPRO (Reference Property) pseudo-attribute.

6.2 Associating a Plotfile with a Design Template

When a designer creates a design element by instancing a design template, the **Create ...** form incorporates a **Plotfile** button which can display an associated plotfile; typically a dimensioned plot of the parameterised item. If no plotfile exists, this button is greyed out. If there is such a plotfile, it can also be shown as part of the **Modify Properties** form (depending on the user's **Property Settings** options).

To associate such a plotfile with a template, proceed as follows:

71. Display the **Define Template Properties** form (see Step *35.*, or select **Modify>Property Definitions** for an existing template).

72. Enter, for example, Description: Plotfile and Key: PLOT. Set **Definition** to **Plotfile**.

73. In the **Data** area, set **Type** to the plot format (**Portrait** or **Landscape**).

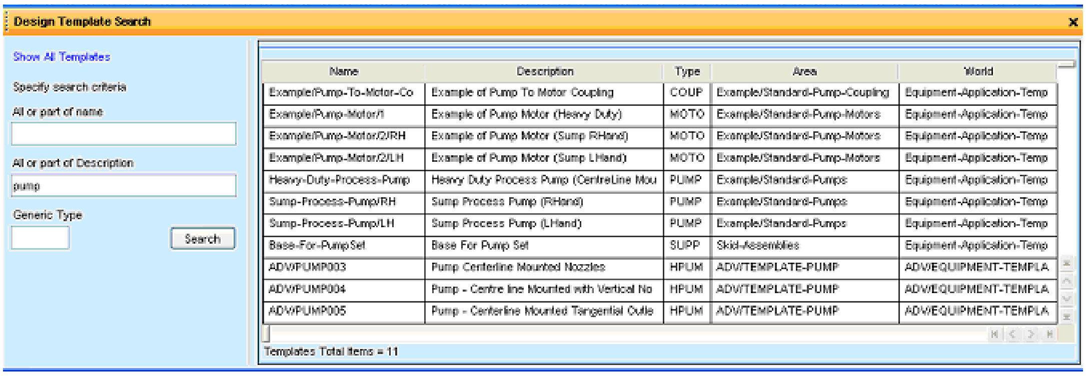
74. The default plotfile directory is as specified by the PDMSPLOTS environment variable. In the **Sub-Dir** field, enter the path from this default directory to the directory holding the plotfile; and in the field below this, enter the name of the plotfile (typically with a .plt suffix).

75. When you click **Include**, the validity of the path and filename will be checked before the property definition is accepted.

6.3 Template Application Search Tool

The Design Template Search Tool is accessed using the **Query>Template Search…** menu command in the Design Templates application. Use the search tool if you know all or part of the Design Template name, description or generic type (Gtype). You can specify several search criteria. For example, find all Design Templates with name containing 'ADV/' and description containing the word 'pump'.

Once Design Templates matching the search criteria have been listed in the table, the search can be further refined by using the table filtering function described below.



Clicking on **Show All Templates** will list all Design Templates in the current MDB in the table. Note that this operation may take a few minutes to complete for a very large set of Design Templates.

Search Criteria:

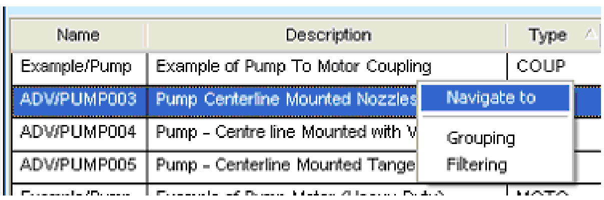
All or part of name	A Design Template will be listed if the Design Template name contains the string entered in this field. Name comparisons are case sensitive.
All or part of description	A Design Template will be listed if the Design Template description contains the string entered in this field. Description comparisons are not case sensitive.
Generic Type	A Design Template will be listed if the Design Template Generic Type (GTYPE) contains the string entered in this field. GTYPE comparisons are not case sensitive.

Once the search critieria has been specified, press the **Search** button to list the Design Templates matching that criteria.

6.3.1 Navigation using the Search Tool

Right click on a single entry on the table of Design Templates and select **Navigate to** to navigate to this Design Template in the database.

In order to use the Search Tool to select the current Design Template, first enter search criteria and perform a search, then filter the resulting table if required (see below), and finally right click on a row in the table and select **Navigate to**. This sets the current element to the selected element. Then click on the **CE TMPL** button on the Template Browser to make this the current Design Template.

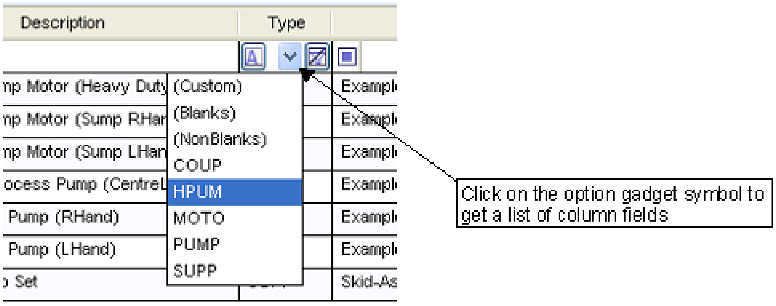


6.3.2 Search Tool Filtering

Right click on the table of Design Templates and select **Filtering** to enable table filtering. This shows a row of filtering controls above the first row of the table. These controls allow the following filtering options:

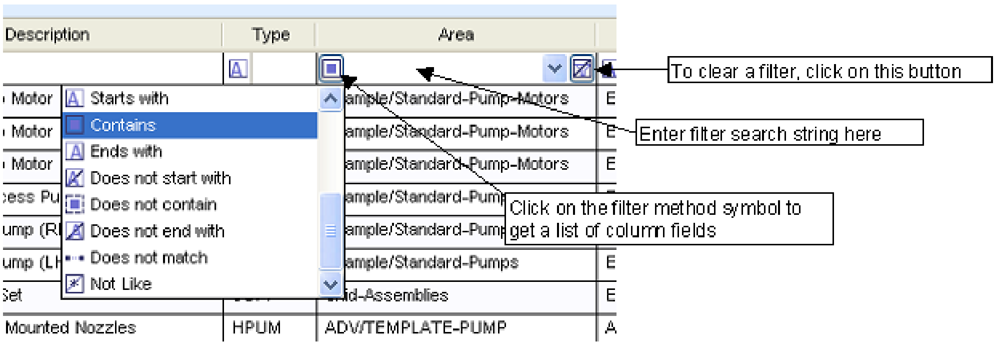
1. Pick from a pull down containing a unique list of entries in that column

   This type of filtering is normally used where the column contains repeated field entries, as in the Type column. In the example below, selecting HPUM will show only those rows with GTYPE equal to HPUM.



2. Select a filter method and supply a filter search string

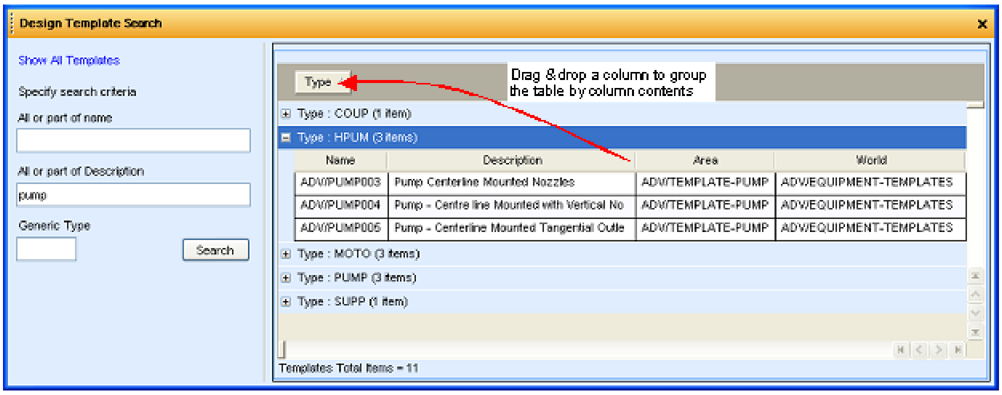
   This type of filtering is normally used where the column contains repeated field entries, as in the Type column. In the example below, selecting HPUM will show only those rows with GTYPE equal to HPUM.



6.3.3 Search Tool Grouping

Right click on the table of Design Templates and select **Grouping** to enable table grouping. This shows a region at the top of the table. Drag a column heading into this region to arrange the table as shown in the example below, where the search results have been grouped by the contents of the Type column (GTYPE).

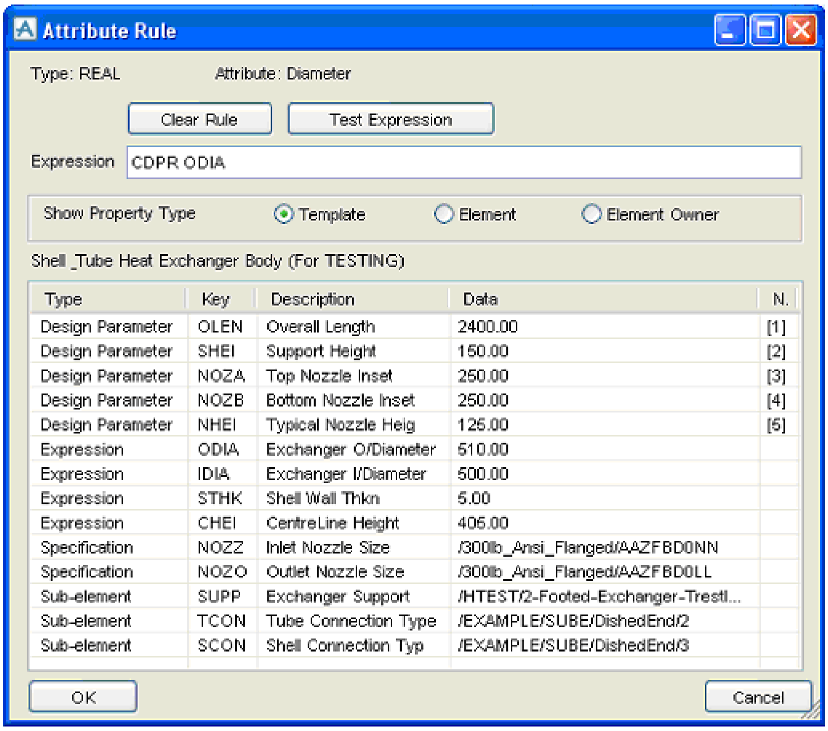
Clicking on a ⊞ button will expand a table of entries for that group.



6.4 Advanced Rules

6.4.1 Defining Attribute Rules

The **Attribute Rule** form is used to define attribute rules. It is accessed by double-clicking on a rule from within the **Template Member Parameters** form. Note: Navigation is suspended while this form is displayed. Each rule is an expression whose result must be of the correct type for the Attribute.



Clear Rule	Delete Rule
Text Expression	Validate the syntax of the expression
Expression	Input field for the expression
Table	Reference section showing properties available for use in an expression

The top part of this form is used to define the expression and test the expression syntax. Note that testing the expression here does not guarantee that the expression will work as expected in a Design model context. Correct behaviour of an expression must be verified by checking Design instances of this Template.

The lower part of this form is a reference section provided to assist the user when entering an expression. The properties shown in the table of properties is controlled by the **Show Property Type** buttons.

If **Template** is selected, the properties shown are the properties of the Template, which can be accessed by an expression like (LDPR ODIA OF TMPL).

If **Element** is selected, and the current element is a catalogue item, the list will contain local properties of that item, typically accessed using an expression like (RPROP WIDTH)

If **Element Owner** is selected, then local properties of the element that owns the current element will be displayed. If the current element is directly owned by the Template, then the Template property list is shown.

6.4.2 Copying and Pasting Rules

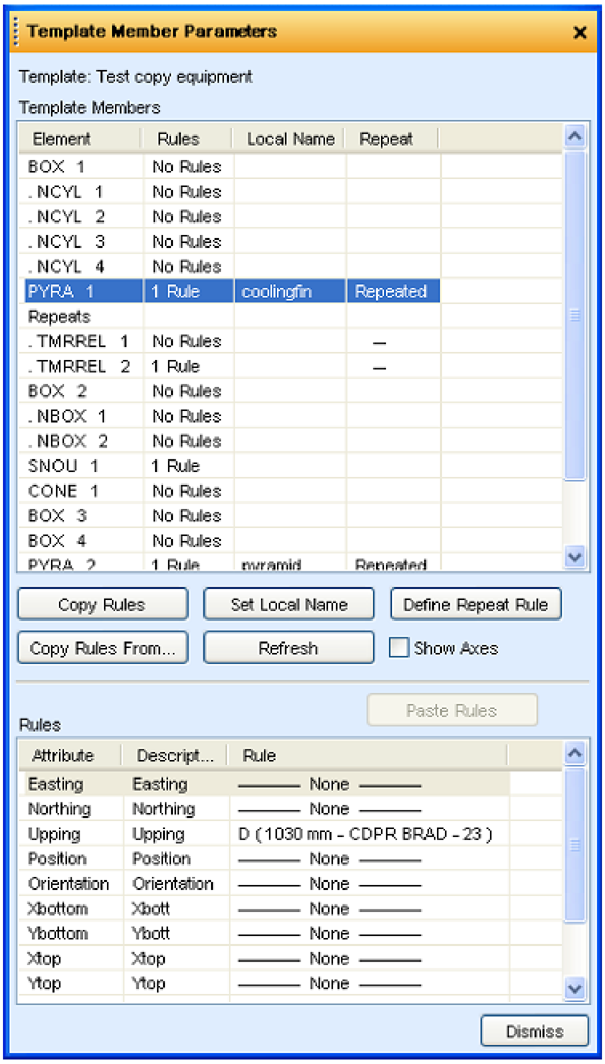
Copy and paste rules are available in the **Template Member Parameterisation** form. They allow existing rule definitions to be reused without reentering them on the Attribute Rule form. Pressing the **Copy Rules** button stores rules of the current element in the Rules Paste buffer.

Select the member to receive the copied rules and press the **Paste Rules** button. The system first checks each attribute of the current member to find an attribute rule for that attribute in the paste buffer. If a matching rule is found, the existing rule is overwritten by the rule stored in the paste buffer. If no matching attribute rule is found in the paste buffer then no change is made to that attribute.

The **Copy Rules…** button allows rules of a picked item to be pasted directly to the current element. When prompted, use the cursor to pick the item with the required rules.

6.5 Repeating Elements

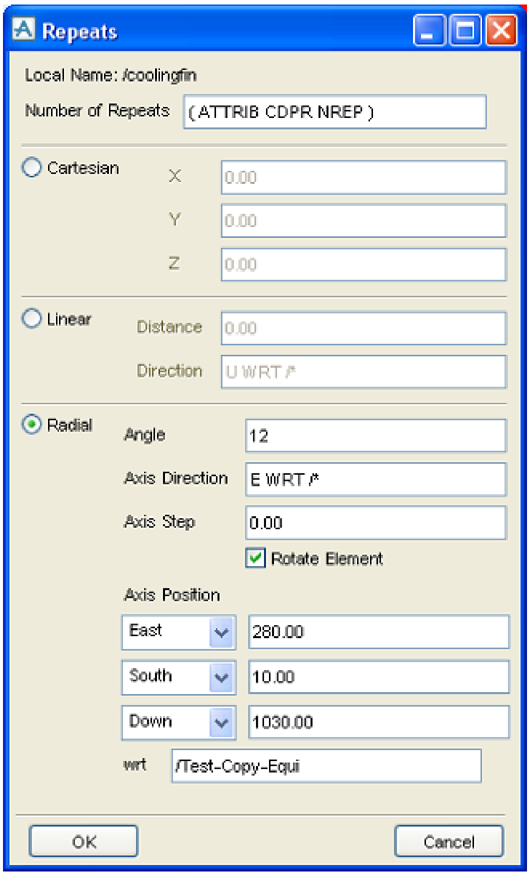
Repeat rules can be defined for an element by selecting a Template member, and pressing the **Define Repeat Rule** button.



The word “**Repeated**” in the Repeat column indicates that this Template member is linked to a Repeat Rule.

To define a Repeat Rule, select the member to be repeated in the list above and press this button.

The **Repeats Definition** form is then displayed as shown below.

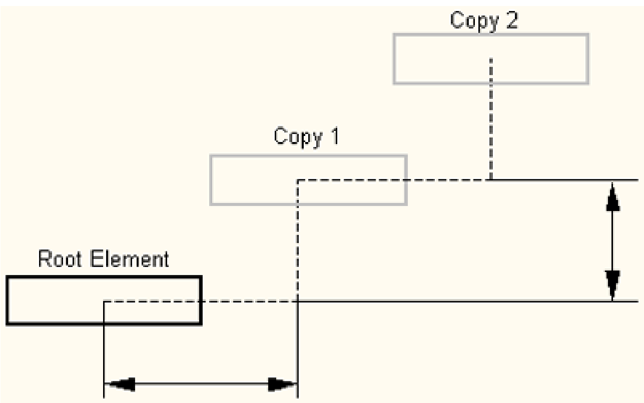


The Number of repeats is provided as an expression. The expression must result in a positive whole number which defines the number of repeats to be produced.

The three sections Cartesian, Linear and Radial define different types of Repeat pattern as described below.

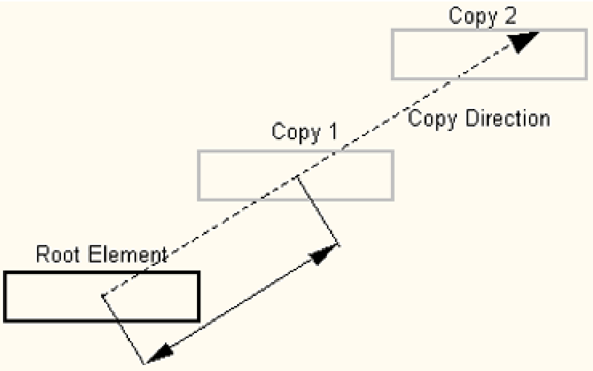
Cartesian Repeat

Each repeat element is displaced from the previous repeat element by an X, Y, Z offset.



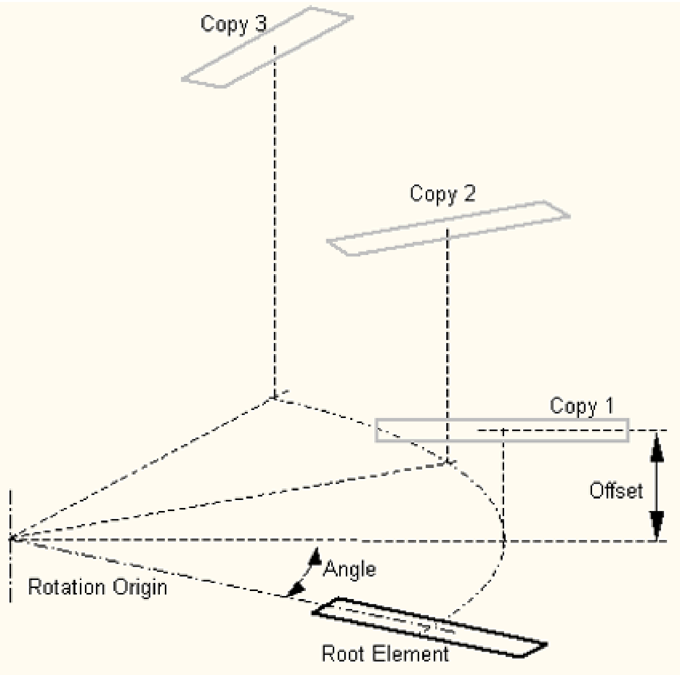
Linear Repeat

Each repeat element is displaced from the previous repeat element by distance in a given direction.

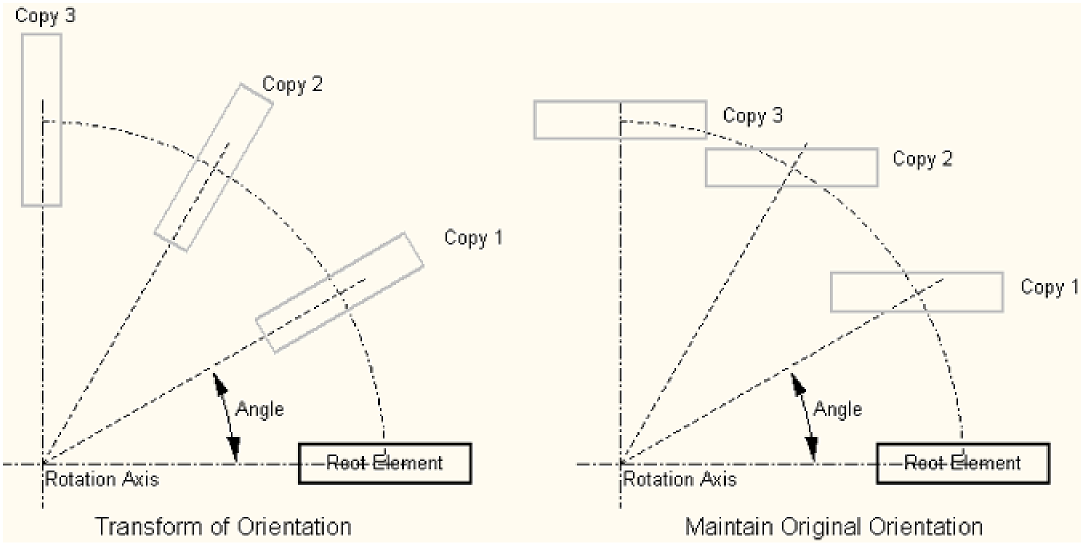


Radial Repeat

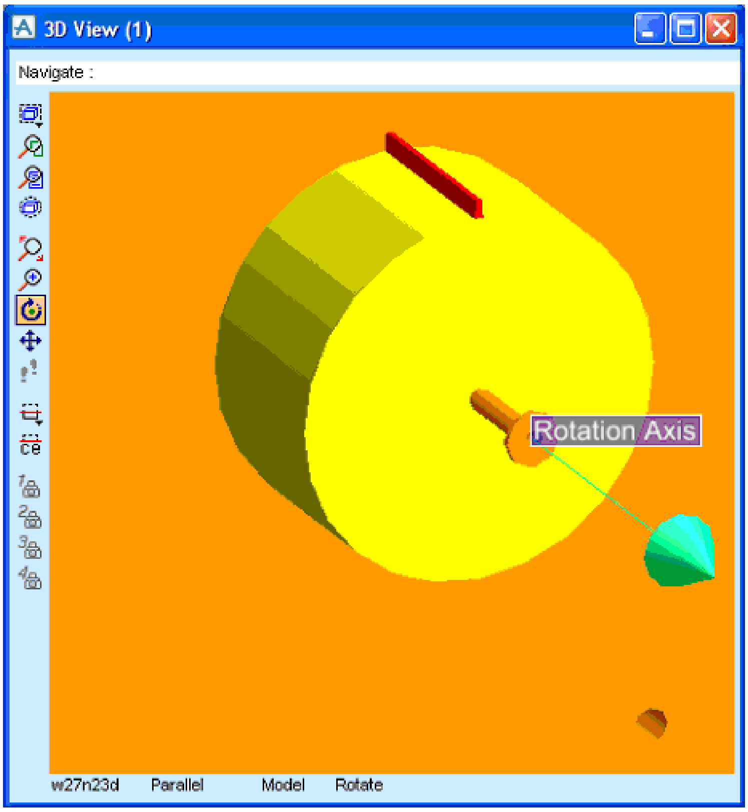
Repeats are generated at an angular displacement around a given axis of rotation. In addition to the angular displacement, each repeat element can be displaced from the previous repeat element by a given distance along the axis of rotation.



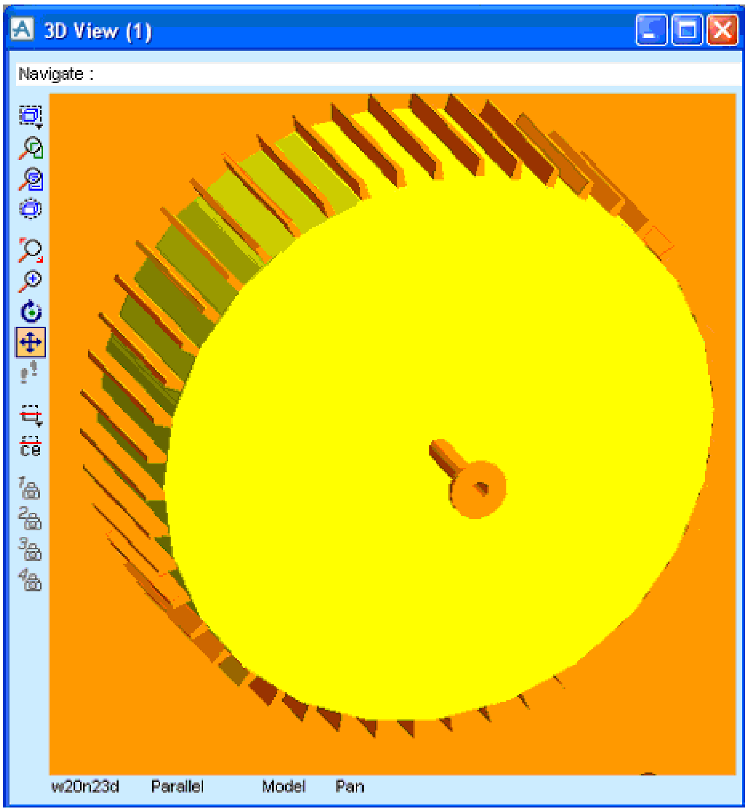
The Rotate Element option specifies whether the orientation of each repeat element is fixed and identical to the master element, or whether each repeat is rotated as shown below.



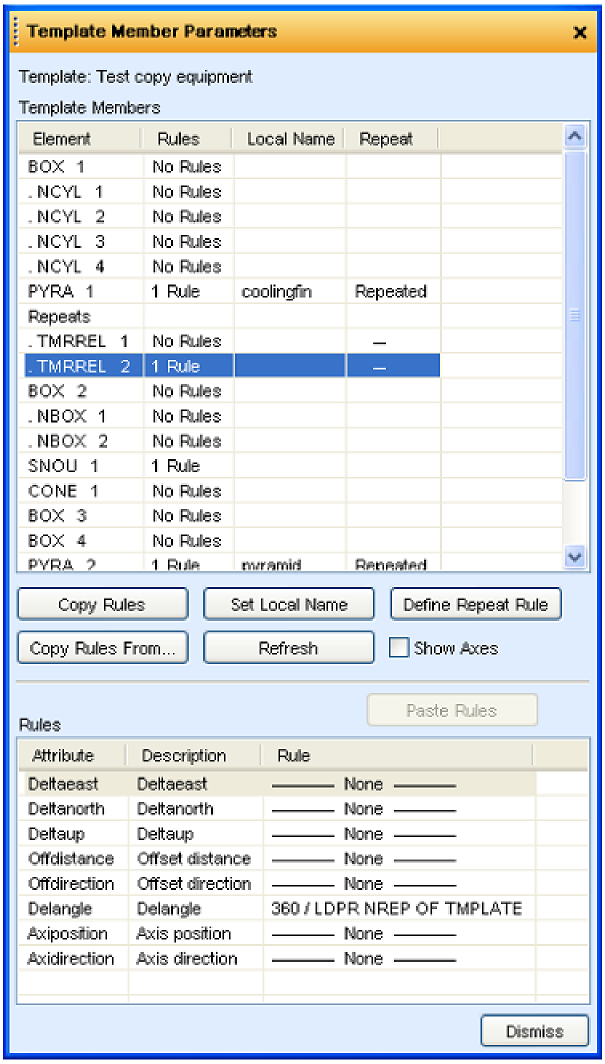
This view shows the display during definition of a radial repeat. The rotation axis is shown on the display, and the master repeat element is highlighted - in this case the pyramid shown in red.



This view shows the display of an instance of the Template showing the Repeated elements.



It is often necessary to parameterise the Repeat Rule element itself. The example below shows a rule on the angular increment.


Template Member Parameters
Template: Test copy equipment
Template Members
Element
Rules
Local Name
Repeat
BOX 1
No Rules
. NCYL 1
No Rules
. NCYL 2
No Rules
. NCYL 3
No Rules
. NCYL 4
No Rules
PYRA 1
1 Rule
coolingfin
Repeated
Repeats
. TMRREL 1
No Rules
. TMRREL 2
1 Rule
BOX 2
No Rules
. NBOX 1
No Rules
. NBOX 2
No Rules
SNOU 1
1 Rule
CONE 1
No Rules
BOX 3
No Rules
BOX 4
No Rules
Copy Rules
Set Local Name
Define Repeat Rule
Copy Rules From...
Refresh
Show Axes
Paste Rules
Rules
Attribute
Description
Rule
Deltaeast
Deltaeast
None
Deltanorth
Deltanorth
None
Deltaup
Deltaup
None
Offdistance
Offset distance
None
Offdirection
Offset direction
None
Delangle
Delangle
360 / LDPR NREP OF TMPLATE
Axiposition
Axis position
None
Axidirection
Axis direction
None
Dismiss

7 Hints and Tips

This chapter summarises the main points to remember to ensure success when creating your own range of templates.

7.1 Preliminary Planning

It is essential that you think very carefully about the geometry and parameterisation of the required template before you create anything. It is particularly important to consider the relationship between the item for which you are designing the template and the way in which it is intended to be used in a design.

A simple sketch of the intended configuration before you begin can save much wasted effort at a later stage. Decide the dataset properties, including their DKEYs and default values, at this stage, and write them on the sketch so that you can remember them when you set up the parameterisation rules. The sketches in *Defining the Template Properties* (the kickplate example) and *Building up a Design from Subsidiary Parts* (the equipment example) illustrate the principle.

This practice is the same as you would adopt when designing catalogue components. The difference is that the catalogue geometry immediately reflects the results of parametric expressions, so that errors are obvious, whereas template rules may fail if there are errors in the rules, but you may not see this until they are used in a design.

7.2 Some Specific Points to Remember

7.2.1 Get the Template Origin Right

When you first create a template by copying a design item, it is important that you immediately position the individual elements correctly to give the most appropriate origin position for the template. You need to consider not only the template origin, but also the origin and axes of its intended owner when instanced in a design. If you need to change the origin after parameterising the components, you will invalidate the rules which you have set.

7.2.2 Get the Template Orientation Right

Make sure that the orientation of the components in the template is correct. Rules must take into account the orientation of the intended owner when instanced in a design.

7.2.3 Use a Consistent Naming Convention

To make it easier to set up cross-references between design items, such as when setting rules involving specification references to, say, associated sub-equipments, it is important that you adopt and maintain a meaningful naming convention. If not, you will soon forget which element is which and, worse, other users will have great difficulty in following the logic of your designs.

Give the template a Description as well as a Name, since this will be displayed as an aid to the user when the template is selected as the basis for a newly created design element. Within the Design Template application, it is the Description which is listed in the **Template Browser**, while the Name is listed in the **Design Explorer**.

7.2.4 Do not use *External* Values Directly in Rules

An expression for a template dimension may require a value derived from a design item not included in the template; for example, the thickness of the owning panel when a panel fitting template is instanced in a design. Always define such a value as a template property (DDAT), with a default setting; do not use the external value directly in a template rule. This is because the owning design item does not exist in the template world, and so rule execution will fail unless a default value is available to be used while the template is being defined. The setting of the Panel Thickness property in Step *16.*, with its default dimension, gave an example of this.

The same principle applies when the template includes variable specification references to other items (which may, or may not, be templates themselves). Again, the specification reference must be defined in the template's dataset so that the default specification can be used to validate the template rules. The selection of the equipment end types in *Setting References to Sub-equipments* gave an example of this process.

7.2.5 Consider Adding Extra Design Points

It is often convenient to add design points into the template so that you can use their positions and/or directions for later reference. They behave in a similar way to p-points in the catalogue, and can be used for reference in other DESIGN and DRAFT operations as well as in template dataset rules. The equipment template design steps in *A More Advanced Example* gave an example of their creation and subsequent use.

7.2.6 Consider the Units Of Measurement

It is important to consider units when setting rules directly in a template. If you use a constant value in a rule, always specify the units to avoid ambiguity; for example,

Rule Set Pos (N (CDPR HEIG + 15 mm)

otherwise the current units will be assumed.

Remember that, in a metric catalogue, items are always dimensioned in mm, so a reference to these in a rule must take this into account. For example, if you use the RPROP pseudo-attribute to extract the value of a catalogue parameter, the result will always be in mm, so you must include a conversion factor in your rule if you are designing using non-metric units.

Note: This does not apply if the catalogue value is defined within a design dataset, because units are handled internally within datasets.

7.2.7 Consider Associated Negative Geometry

If negative geometry is required to cut through an owning element in a design instance, the negative primitives must be owned directly by the template and parameterised accordingly.

The kickplate template design steps in *Modifying the Template Geometry* gave an example of this.

The properties of such negative items will often be derived from the dimensions of owning elements in the design; see *Do not use External Values Directly in Rules*.

7.2.8 Always Test a New Template in a Design

Some template rules may not be able to be evaluated fully until the template is instanced in a design; for example, the rules for positioning and orientating the sub-equipments representing the vessel ends in *Setting References to Sub-equipments* cannot be tested until the specification references for the sub-equipments are set. You cannot check if these rules are correctly defined from within the Design Template application. It is important, before you release a new template to other users, that you incorporate it into a specification (which can be a dummy specification reserved for this purpose), and then create an instance in a design application to check that everything behaves as you intended. Change each of the main property settings in the design to test the effects.

A Other Relevant Documentation

This guide is intended only as an introduction to those parts of PDMS most relevant to template design. As such, it describes only the main concepts needed to get you started. Should you need more detailed information about any topic, the following documents are available.

A.1 On-Line Help

For detailed instructions on the use of the forms and menus via which you control the application, on-line help is provided as an integral part of the user interface.

The **Help** option on the menu bars gives you the following choices:

Help>Contents

This displays the **Help** window so that you can find the required topic from the hierarchical contents list.

Help>Index

This displays the **Help** window with the **Index** tab selected, so that you can browse for the topic you want to read about from the alphabetically-arranged list. You can locate topics quickly by typing in the first few letters of their title.

Help>Search

This displays the **Help** window with the **Search** tab at the front so that you can find all topics containing the keywords you specify.

Help>About

This displays information about the current operating system on your computer and about the versions of PDMS and its applications to which you have access.

Pressing the F1 key at any time will display the help topic for the currently active window.

A.2 Introductory Guides

The guide that you are now reading is one of a series of user guides which aim to introduce new users to the use of PDMS as quickly as possible. Other guides in this series include:

- *Pipework Design User Guide*
- *Structural Design Using PDMS User Guide*
- *Industrial Building Design Using PDMS*
- *HVAC User Guide*

- *Support Design Using PDMS*
- *Reporting From PDMS*
- *DRAFT User Guide*

A.3 Reference Manuals

The full documentation set includes a number of reference manuals which give detailed explanations of all the technical concepts involved. These manuals also describe the underlying command syntax which can be used to control PDMS directly (thus bypassing the forms and menus interface).

They include:

DESIGN Reference Manual	Covers concepts and commands for all design disciplines.
Catalogues and Specifications Reference Manual	Explains how to set up a catalogue and how to create tabulated specifications.

A.4 General Guides

The following guides are intended for use only by experienced users who want to write their own applications:

- *Software Customisation Guide*

  Explains how to write your own application macros using PML (AVEVA's Programmable Macro Language) and how to design your own forms and menus interface.
- *Customisation Reference Manual*

  Supplements the *Customisation Guide*. Includes a list of PML 2 Objects, Members and Methods. For Forms and Menus objects, the command syntax relating to the objects is included.

Index

M

O

P

R

S

T

V

W

Z