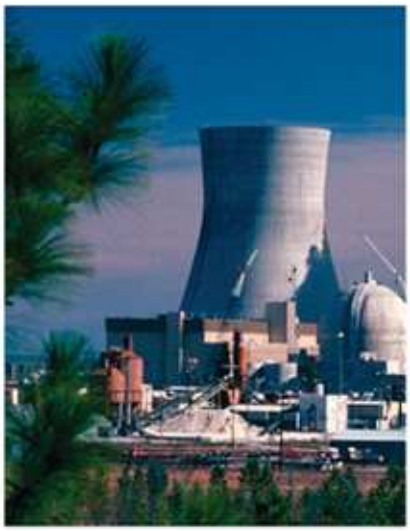
SmartPlant 3D Hangers & Support Customization

Student Workbook

Process, Power & Marine



INTERGRAPH

Version 2009 SP1 **March 2009** **N/A**

Copyright

Copyright © 2004-2008 Intergraph Corporation. All Rights Reserved.

Including software, file formats, and audiovisual displays; may be used pursuant to applicable software license agreement; contains confidential and proprietary information of Intergraph and/or third parties which is protected by copyright law, trade secret law, and international treaty, and may not be provided or otherwise made available without proper authorization.

Portions of this software are owned by Spatial Corp. © 1986-2008. All Rights Reserved.

Restricted Rights Legend

Use, duplication, or disclosure by the government is subject to restrictions as set forth below. For civilian agencies: This was developed at private expense and is "restricted computer software" submitted with restricted rights in accordance with subparagraphs (a) through (d) of the Commercial Computer Software - Restricted Rights clause at 52.227-19 of the Federal Acquisition Regulations ("FAR") and its successors, and is unpublished and all rights are reserved under the copyright laws of the United States. For units of the Department of Defense ("DoD"): This is "commercial computer software" as defined at DFARS 252.227-7014 and the rights of the Government are as specified at DFARS 227.7202-3.

Unpublished – rights reserved under the copyright laws of the United States.
Intergraph Corporation
Huntsville, Alabama 35894-0001

Warranties and Liabilities

All warranties given by Intergraph Corporation about equipment or software are set forth in your purchase contract, and nothing stated in, or implied by, this document or its contents shall be considered or deemed a modification or amendment of such warranties. Intergraph believes the information in this publication is accurate as of its publication date.

The information and the software discussed in this document are subject to change without notice and are subject to applicable technical product descriptions. Intergraph Corporation is not responsible for any error that may appear in this document.

The software discussed in this document is furnished under a license and may be used or copied only in accordance with the terms of this license.

No responsibility is assumed by Intergraph for the use or reliability of software on equipment that is not supplied by Intergraph or its affiliated companies. THE USER OF THE SOFTWARE IS EXPECTED TO MAKE THE FINAL EVALUATION AS TO THE USEFULNESS OF THE SOFTWARE IN HIS OWN ENVIRONMENT.

Intergraph is not responsible for the accuracy of delivered data including, but not limited to, catalog, reference and symbol data. Users should verify for themselves that the data is accurate and suitable for their project work.

Trademarks

Intergraph, the Intergraph logo, PDS, SmartPlant, FrameWorks, I-Convert, I-Export, I-Sketch, SmartMarine, IntelliShip, INtools, ISOGEN, MARIAN, SmartSketch, SPOOLGEN, SupportManager, and SupportModeler are trademarks or registered trademarks of Intergraph Corporation or its subsidiaries in the United States and other countries. Microsoft and Windows are registered trademarks of Microsoft Corporation. ACIS is a registered trademark of SPATIAL TECHNOLOGY, INC. Infragistics, Presentation Layer Framework, ActiveTreeView Ctrl, ProtoViewCtl, ActiveThreed Ctrl, ActiveListBar Ctrl, ActiveSplitter, ActiveToolbars Ctrl, ActiveToolbars Plus Ctrl, and ProtoView are trademarks of Infragistics, Inc. Portions of 2D DCM, 3D DCM, and HLM from D-Cubed Limited are incorporated. All rights reserved. Oracle, JD Edwards, PeopleSoft, and Retek are registered trademarks of Oracle Corporation and/or its affiliates. Other brands and product names are trademarks of their respective owners.

Table of Contents

Introduction 3

Hangers and Supports: An Overview 4

Standard Supports and Designed Supports 5

Support Parts 8

Connection Types 9

Hangers and Supports Reference Data 11

Catalog Hierarchy Sheets 14

Defining Support Parts: An Overview 17

Custom Interfaces 25

Part Symbol 28

Understanding Part Ports 29

Understanding Reference Ports 31

Defining Assemblies: An Overview 36

Assembly Information Rule Example 46
GetAssemblyCatalogParts Method 48

Understanding Constraints 50

Understanding Joints 52

GetAssemblyJoints Method 57
GetRouteConnections Method 61
GetStructConnections Method 64

Lookup Tables: An Overview 65

Face Selection Sheet 65
Face Position Selection Sheet 67
SupportJoints Sheet 68

Configuration Index Calculator 71

Bounding Box Reference Ports 74

Lab 1: Support Catalog Hierarchy 77

Lab 2: Standard Support Component Symbol (SPUBolt) 94

Lab 3a: Assembly Information Rule - Method of Take-Outs 109

Lab 3b: Assembly Information Rule - Method of Proper Joints 120

Lab 4: Assembly Information Rule Using Connection Component Parts .. 123

Lab 5: Using Table Lookup in the Assembly Information Rule 137

Lab 6: Weld Component Symbol 149

Lab 7: Assembly Information Rule for a Cantilever U-Bolt Assembly 159

Lab 8: Support Fabrication Drawings (Optional) 176

Appendix 207

IJHgrInputConfigHlpr 207
Helper Function Reference 209
SP3D References Tool 221
Debugging Your Code 223

Sample - Load Rated Parts 224

Example: Braced 45 Deg Cantilever C3X6 w/ Baseplate-UBolt Assembly . 226

Example: Unbraced Cantilever C3X6 w/ Baseplate Assembly 232

Introduction

This guide is designed as an aid for students attending the SmartPlant 3D Support Assemblies Customization class presented by Intergraph Corporation and it's a supplement to the standard product documentation.

Objective

This document provides a guideline for people who need to develop support component symbol definition and support assembly creation for the SmartPlant Support Library. This handout includes, but is not limited to the following:

- Provides an overview of customization with the SmartPlant 3D software using standard Windows™ programming tools and languages like Visual Basic™
- Understanding Support Reference Data and delivered files
- Procedures for customizing support components and assemblies

Assumptions are made here that the user has a prerequisite knowledge of the SmartPlant 3D reference data.

Course Description

Support Component Symbol and Assembly Information Rule

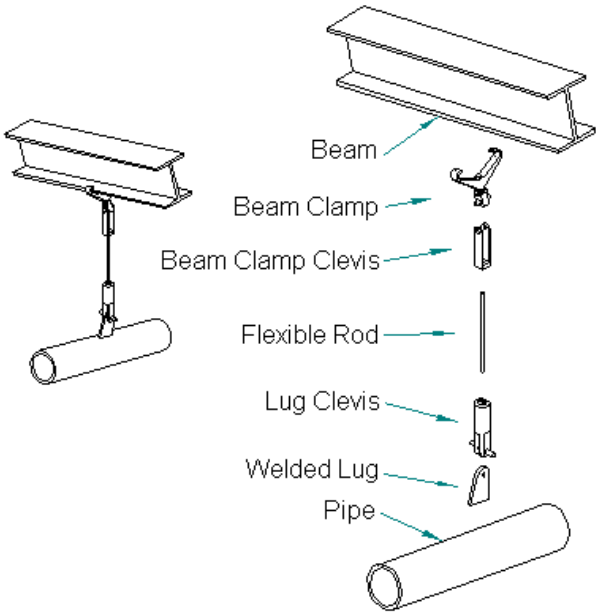
Course Reference Material

SmartPlant 3D Reference Data Guide
SmartPlant 3D Hangers and Supports Reference Data Guide
SmartPlant 3D Symbols Reference Data Guide

Hangers and Supports: An Overview

A support is an assembly of parts (also called components) that connect together, to the supported features, and to the supporting structure. The Supports and Support Parts you can use are defined in the catalog and will generally be customized by your SmartPlant 3D catalog administrator.

The following figure shows a typical support that contains five unique parts.

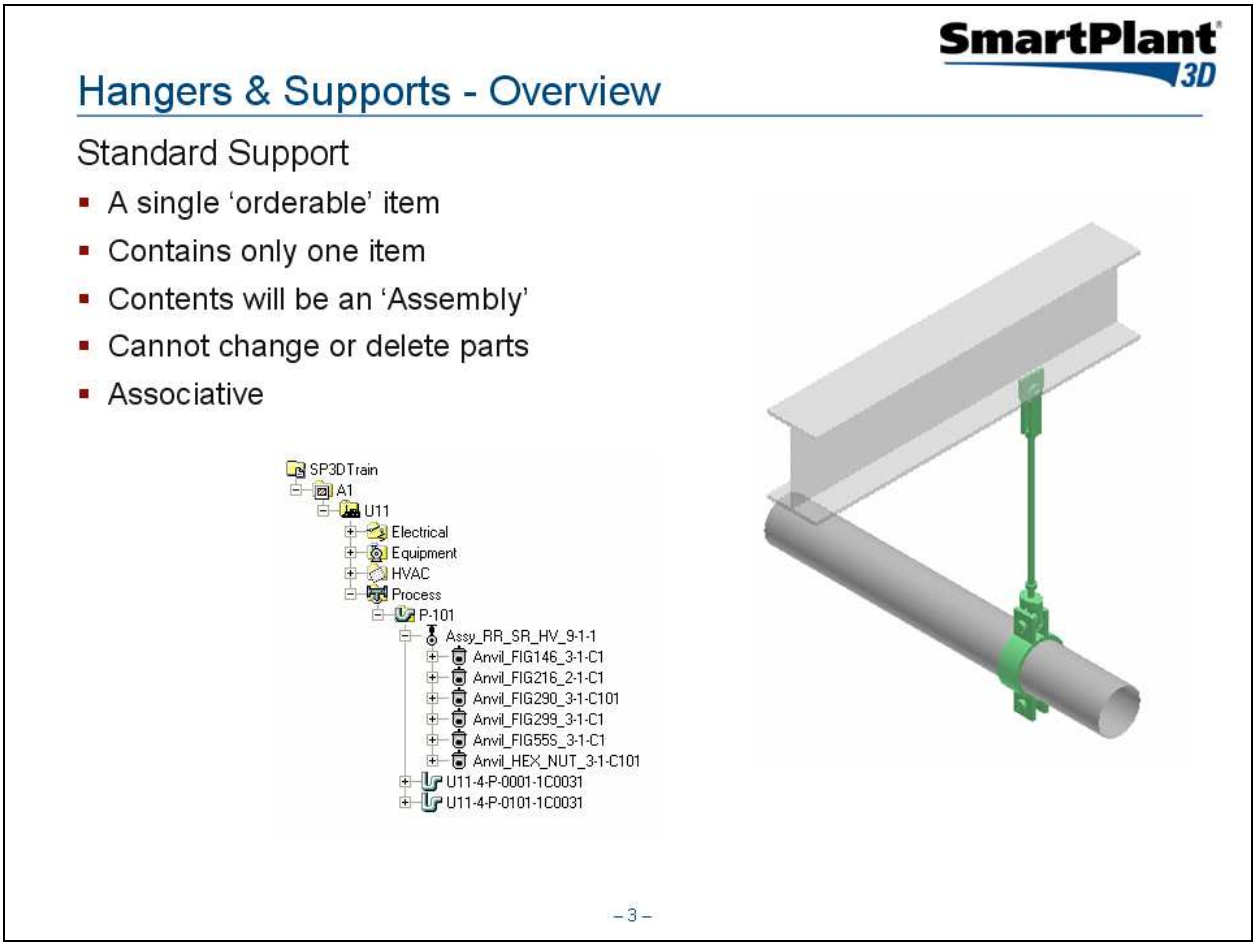


Hanger Assembly with Constituent Parts

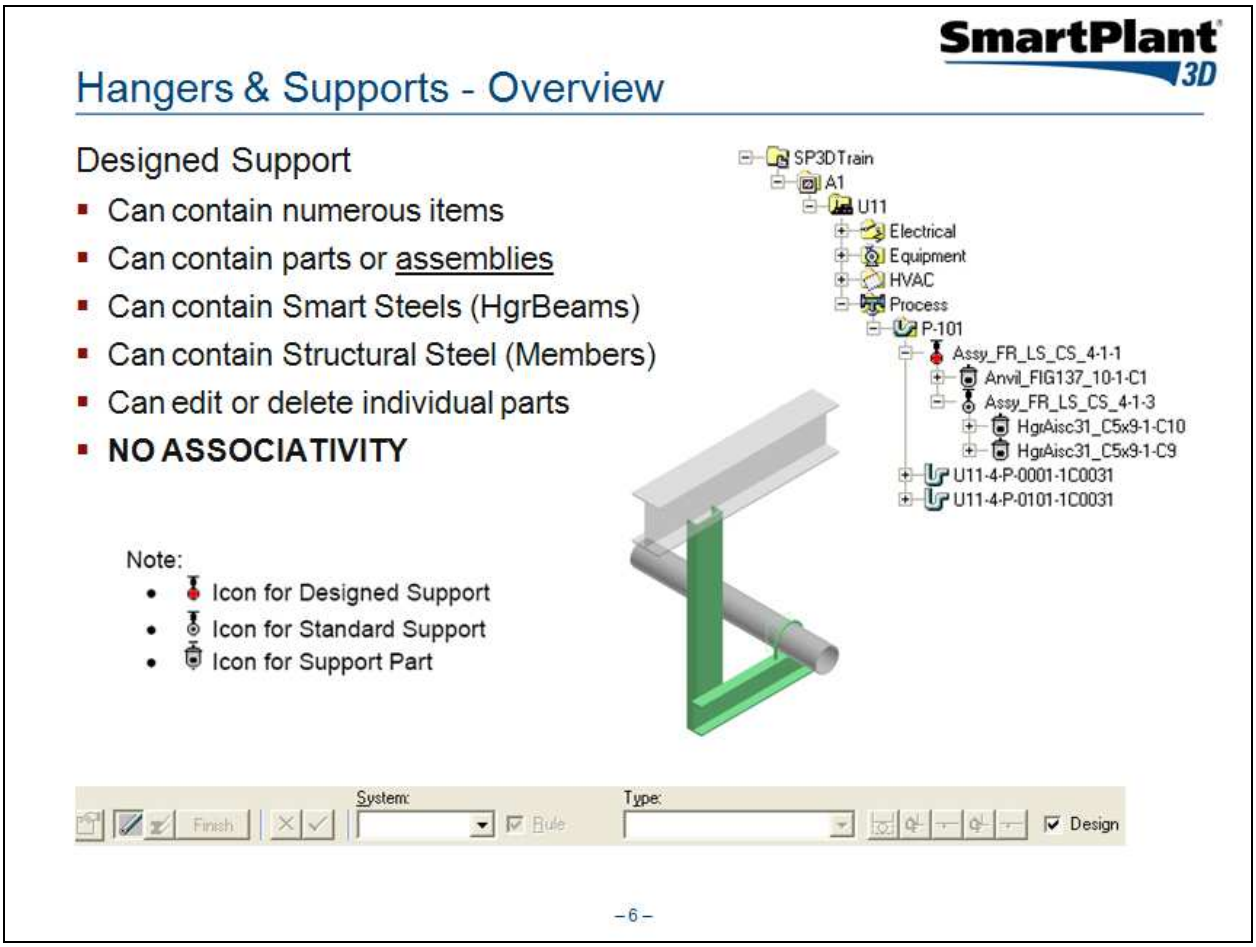
The assembly of a support's constituent parts is done using information defined in the reference data. Each assembly defined in the reference data is assigned an *assembly information rule (AIR)*. The software uses the logic contained in the assembly information rule to put together the individual support parts to form an assembly.

Standard Supports and Designed Supports

There are two basic types of supports, Standard Support and Designed Support. Standard Support will automatically select the individual parts from the catalog and position all of the parts based on the supported route features, the supporting structures, and the constraints defined in the rule. If the supported route features or the supporting structures are modified, the standard support will adjust the design of the individual parts according to the rules of the standard support. You can only modify the parts of the standard support by changing the properties of the controlling standard support.



Designed Support is a custom configuration of parts. You create designed support by selecting the option to place a designed support instead of a standard support on ribbon bar of the support placement commands. Designed Support option creates a reference graphic at the identified support location instead of placing parts. You then manually place support parts in the designed support using the Place Part command. You can use the reference graphic to help position the support parts.



You can place a standard support within a designed support by simply selecting the designed support as the system parent of the standard support. This lets you use the automation of the standard support for selecting and positioning some of the parts and while you manually place and position other parts needed for your custom designed support. You can optionally report on the nested standard support identity on material reports or report on just the individual support parts by setting the reporting properties.

Note:

- Icon for Designed Support
- Icon for Standard Support
- Icon for Support Part

If you want to modify the individual parts of a standard support after you have placed a standard support, you must convert the standard support to a designed support using the Drop Standard command. This removes the control of the standard support over the part selection and positioning and leaves all parts in their current location. It is as if you had started with a designed support and placed the parts manually in the designed support.

SmartPlant 3D

Hangers & Supports - Overview

Designed Support

- You can place a standard support within a designed support
- You must convert the standard support to a designed support using the Drop Standard command
- The Drop Standard command removes the control of the standard support over the part selection and positioning and leaves all parts in their current location

Drop Standard Command

– 7 –

SmartPlant 3D

Hangers & Supports - Overview

Parts

- You place Support Parts manually with the Place Part command
- Support Parts have predefined positioning locations called “ports”
- If you want to add support parts to a standard support, you can just pick the standard support as the system parent of the support part

Place Part Command

– 8 –

Support Parts

You place support parts manually with the Place Part command. The command will prompt you to select the parent support. If you want to add support parts to a standard support, you can just pick the standard support as the system parent of the support part. The software will automatically convert the standard support to a designed support when the part is placed.

Support Parts have predefined positioning locations called “ports”. These ports carry information about how they assemble relative to supported features, other support part ports, and the supporting structure. This information makes it easy to assemble parts that must connect. The Place Part command has a toggle that changes the port being used during placement.

SmartPlant 3D

Hangers & Supports - Overview

Parts

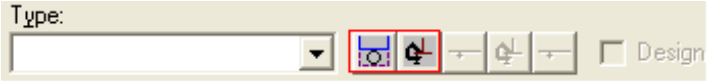
- Data Driven Entity (Support occurrence attributes)
  - Occurrence Attributes
  - Ports
- Data defined in the Catalog
- Custom programs for Selection and Graphics
  - Symbol Definition (support aspects)
  - Part Selection Rule

Output Ports for an Adjustable Swivel Ring Part Symbol

– 9 –

Connection Types

You can select the connection type by clicking on the Toggle Connection to Structure/Feature buttons as shown below on the Support Placement ribbon. The software limit the available connections based on the rules defined in the AIR (Assembly information Rule) for the specific support.



Support Placement ribbon

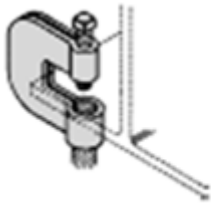
- Toggle Connection to Feature
- Toggle Connection to Structure
- Toggle Face Position

Toggle Connection to Feature – Cycles through the available placement options based on the selected feature and the support type. This option may not be available if the type that you selected does not provide for multiple options defined in the assembly information rule.

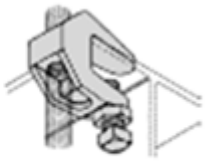
Use the GetMaxRouteConnectionValue method to defines the route toggle configurations supported by the Assembly Information Rule.

```
Private Function IJHgrAssmInfo_GetMaxRouteConnectionValue() As Long
    IJHgrAssmInfo_GetMaxRouteConnectionValue = 2
End Function
```

Toggle Connection to Structure – Cycles through the available attachment (face) options based on the supporting structure and the support type. If you are placing a support connecting to two structures, the ribbon displays two buttons, one for each structure. This option may not be available if the type you selected does not provide for multiple attachment ports.

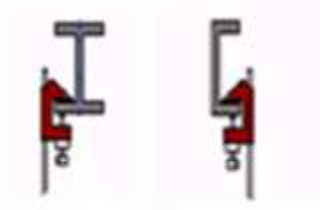


Steel C-Clamp



Light Duty
Malleable C-Clamp

Toggle Face Position - Cycles through the available face positions for the selected port (surface) on the supporting structure. For example, you can use this option to move a U-bolt to a location on the member flange where it does not interfere with the web of a beam. If you are placing a support connecting to two structures, the ribbon displays two buttons, one for each structure. This option may not be available if there are no optional face positions defined for the selected support.



Hangers and Supports Reference Data

Before working with hangers and supports reference data, you must be familiar with how SmartPlant 3D handles reference data in general. If you have not already done so, read and understand the following important concepts and procedures described in the *SmartPlant 3D Reference Data Guide*:

- Custom attributes
- Part classes
- Symbol creation
- Codelists (also referred to as "select lists")
- Bulkloading

For example, before you can define support parts, you must understand how part data relates to data on the custom interfaces sheet, and how that information relates to parameters defined when the part symbol is created.

Hangers and Supports Workbooks

The hangers and supports reference data is defined in several Microsoft Excel workbooks. The software delivers these workbooks to the *[Product Directory]*\CatalogData\Bulkload\DataFiles folder. Additional workbooks are delivered to the *[Product Directory]*\CatalogData\Bulkload\AdditionalDataFiles folder.

Bulkload Utility

In order to use reference data in the software, you must load data in the Excel workbooks to the Catalog database using the Bulkload utility.

Bulkloading is a process that imports data from the Excel workbooks into the Catalog database. You should back up the workbooks as needed before you make modifications and begin the loading process. The loading process has several modes available: New, Append, Add/Modify/Delete, and Delete and Replace.

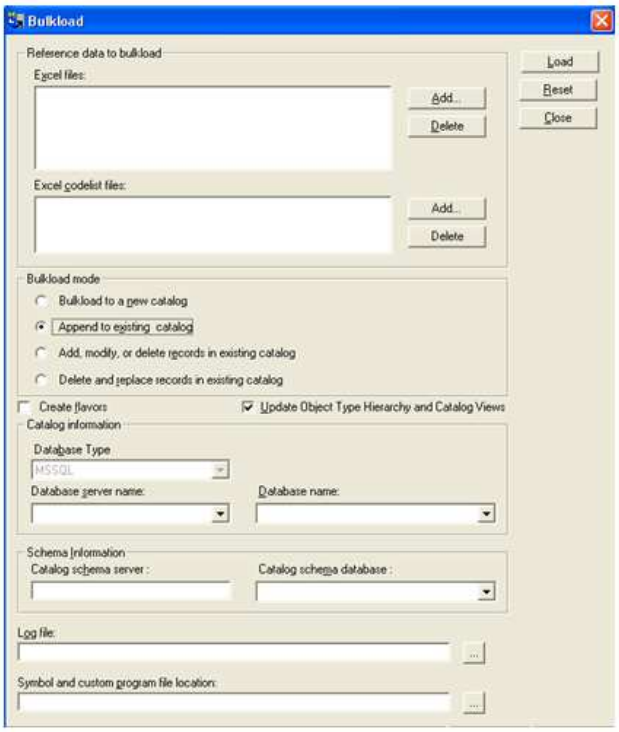
Bulkload Utility

SmartPlant 3D

The Bulkload Utility provides the primary mechanism to create/add/modify the data in the catalog database. The data for creating the database is defined using Excel workbooks.

Bulkload Mode:

- Bulkload to a new catalog: is used for creating new catalog database
- Append: Add new data only
- Add/Modify/Delete: is used for update data
- Delete & Replace: is used to replace data in the database with the data currently in the workbooks
- Update Object Type Hierarchy and Catalog Views" option: allows the catalog administrator to choose when the Business Object Classification Hierarchy (BOC) and catalog views need to be updated



– 14 –

Several sheets are common to most of the reference data workbooks. These sheets include comments about the revision history, links to the other sheets, custom interface and attribute definitions, and hierarchy information.

Other sheets in the workbooks contain component-specific information for objects in reference data. These sheets define part classes, which are groups of related objects. Each part class contains one or more parts, which are unique instances of that type of object.

Using Excel Workbooks to Configure Catalog Data

SmartPlant 3D

Common Sheets

- Revision History*
- Legend*
- Index*
- CatalogRoot
- ClassNodeType
- R-ClassNodeDescribes
- R-Hierarchy
- Custom Interfaces
- GUIDs

* The Bulkload utility does not process these sheets, so you can enter any comments and information on them.

– 7 –

Using Excel Workbooks to Configure Catalog Data

SmartPlant 3D

Bulkload Excel Format

- **The "Head", "Start" and "End" are always required in the first column. One can have as many empty rows as he/she wants to, anywhere in the sheets. Comments can be put by putting "!" in the first cell of the row in which the comment needs to be put.**
- **Letter bulkload mode A,M,D**

	A	B	C	D	E	F	G	H
1	Definition	PartClassType	SymbolDefinition	SymbolIcon	PartSelectionRule	ClassType	UserClassName	HgrSymbolPort(1):Ty
2								
3		StandardSupportComponentClas	UBoltSet.CUBoltPart		HgrPipePartSelRule.CPartByPipeSizeEqual	1		HgrSymbolPort
4								
5	Head	PartNumber	PartDescription	SymbolDefinition	MirrorBehaviorOption	HDFrom	HDTo	HDUnitType
6								
7	Start							
8		URH-075			15	0.75	0.75	in
9		URH-001			15	1	1	in
10		URH-002			15	2	2	in
11		URH-004			15	4	4	in
12		URH-006			15	6	6	in
13		URH-008			15	8	8	in
14	End							
15								

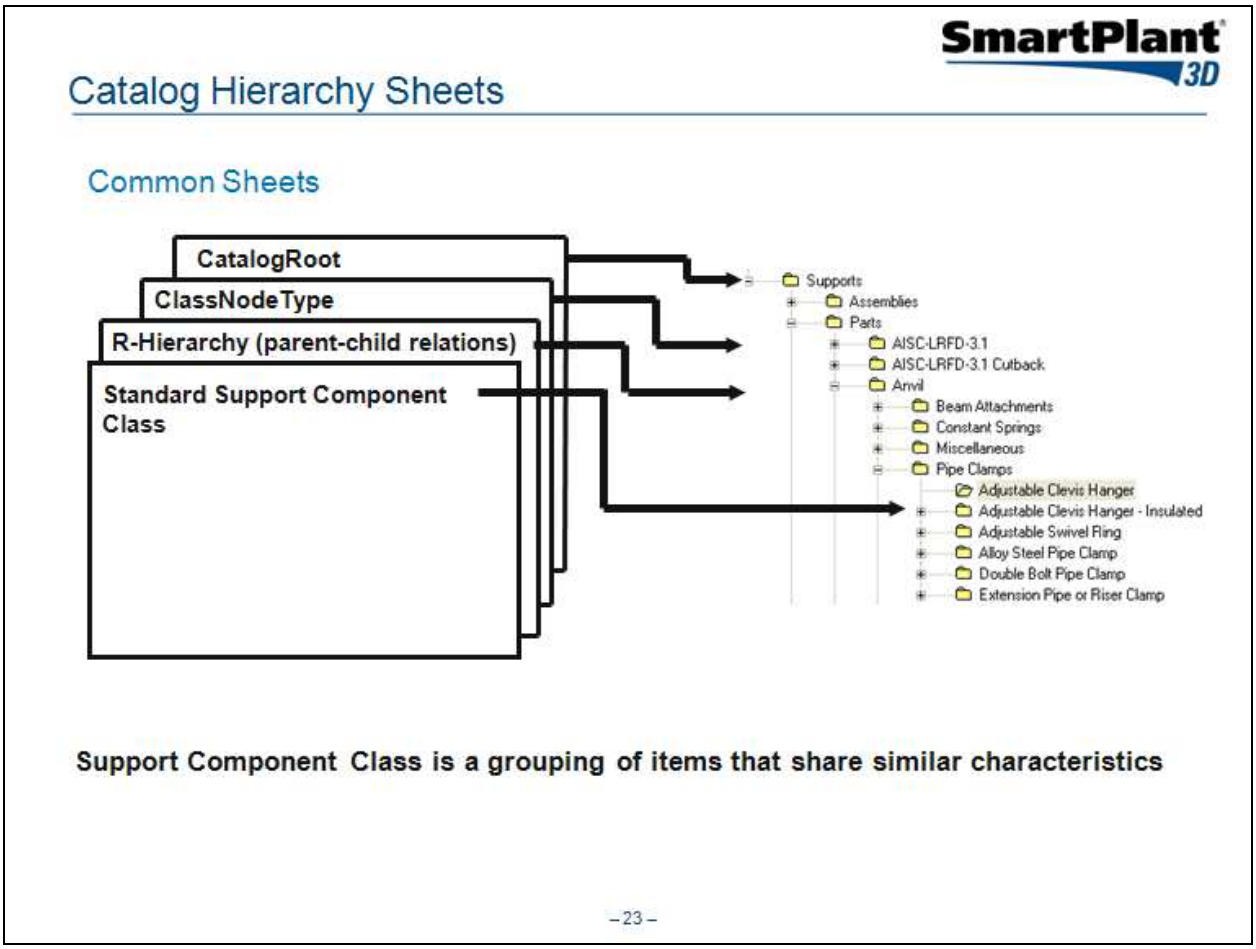
CustomInterfaces \ UBoltSet / FlexRod / WeldedBeamPart / ClassNodeType / R-Hierarchy / R-ClassNodeDescribes

– 8 –

Catalog Hierarchy Sheets

To customize the catalog hierarchy, you must edit the Excel workbooks, which contain four sheets that control the hierarchy: CatalogRoot, ClassNodeType, RHierarchy, and R-ClassNodeDescribes. Hierarchy sheets do not contain attribute names; they contain object names.

When you add a part class to the catalog hierarchy, you must update the hierarchy sheets, where necessary, in order to see the data in the Catalog task.



CatalogRoot is the RootNode of the ReferenceData Catalog Browser. It cannot have a parent node. It is a named-object. The ClassNodeType defines the folder object names and corresponding user names. The R-Hierarchy maps parent and child relationships between folders in the catalog hierarchy.

RelationSource and RelationDestination entries are ObjectNames of the objects participating in the relationship. A relationship can't be made unless the objects mentioned in RelationSource / RelationDestination columns are already created.
The R-ClassNodeDescribes is the relationship between an object from ClassNodeType as RelationSource and another object which is a PartClass as RelationDestination.

RelationSource entries are ObjectNames of the objects participating in the relationship. RelationDestination entries are names of the Part Classes sheets participating in the relationship.

SmartPlant 3D

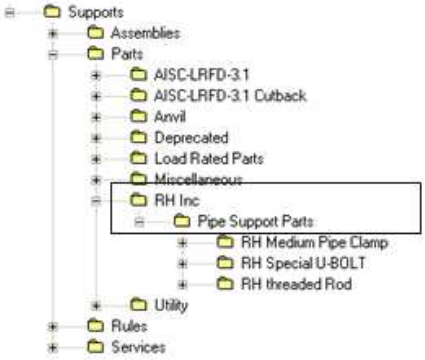
Catalog Hierarchy Sheets

ClassNodeType

allows you to create a classification folders in the Catalog hierarchy.

- **Object Name column:** **folder object name**
- **Name column:** **folder name in the GUI.**

Head	*ObjectName*	*Name*
Start		
	CLASSNODE_PART_SPCOMPANY	RH Inc
	CLASSNODE_PART_SPCOMPANY_PARTS	Pipe Support Parts
End		



– 24 –

Catalog Hierarchy Sheets

R-Hierarchy

contains the parent-child relationships between folders in the Catalog Hierarchy.

- **RelationSource column:** **specifies the parent object names.**
- **RelationDestination column:** **specifies the children object names.**

Head	*RelationSource*	*RelationDestination*
Start		
	CLASSNODE_PARTS_DEFINITION	CLASSNODE_PART_SPCOMPANY
	CLASSNODE_PART_SPCOMPANY	CLASSNODE_PART_SPCOMPANY_PARTS
End		

– 11 –

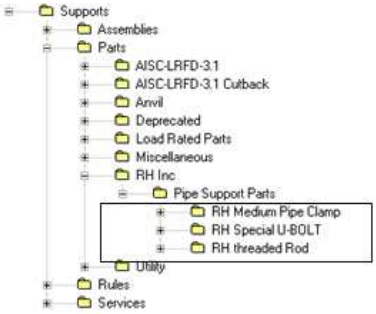
Catalog Hierarchy Sheets

R-ClassNodeDescribes

Defines the parent-child relationships between the classification folders and the part classes.

- **RelationSource column:** **specifies the classification folder.**
- **RelationDestination column:** **specifies the part class associated with the classification folder.**

Head	*RelationSource*	*RelationDestination*
Start		
!	**Support Part Classes**	
	CLASSNODE_PART_SPCOMPANY_PARTS	RH_SPCLAMP
	CLASSNODE_PART_SPCOMPANY_PARTS	RH_THREADED_ROD
	CLASSNODE_PART_SPCOMPANY_PARTS	RH_SPUBOLT
End		



– 26 –

Defining Support Parts: An Overview

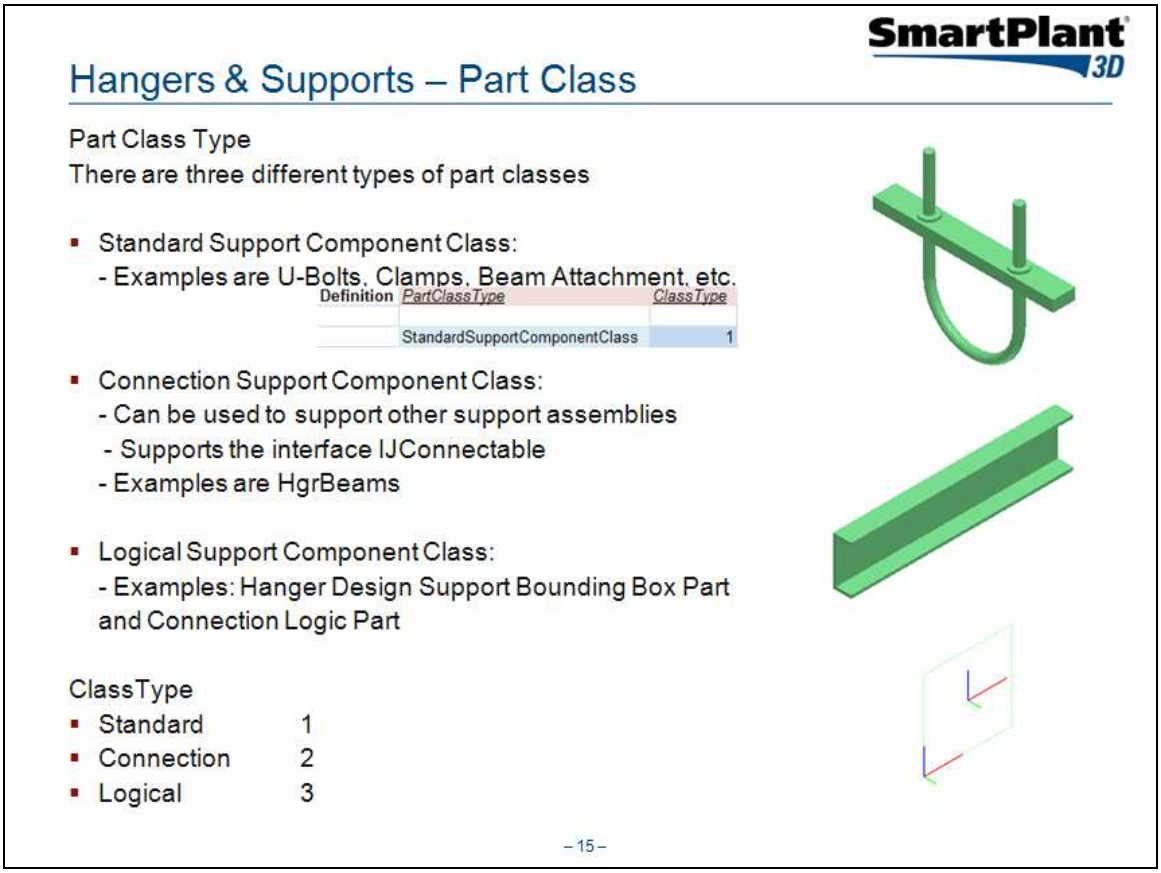
Support Parts are used to create the support assemblies that are placed in the model. Support Parts can be grouped into three categories: Supporting parts, Supported parts and Flexible parts.

Supporting Parts: Those parts physically touching structure (supporting entity) and responsible for maintaining contact between the support and structure. A physical connection exists between supporting parts and the structure.

Supported Parts: Those parts physically touching the routing part (supported entity) and responsible for maintaining contact between the support and path features. A physical connection exists between supported parts and the path feature.

Flexible Parts: Those parts responsible for maintaining connections between the supporting and supported parts are referred to as flexible parts. Physical connections exist between flexible parts and the other parts of the assembly. Any assembly part that is not a supporting or supported part is by definition a flexible part. For example, HgrBeam parts are special H&S steel symbols based on extruding the 2D steel symbols.

There are three types of part class to define these support parts.



SmartPlant 3D

Support Component Part Class

Standard Support Component Class is a grouping of support parts that share similar characteristics

A	B	C	D	E	F	G	H	I	J
Definition	PartClassType	SymbolDefinition	SymbolIcon	oa:IJO	oa:IJC	oa:IJOAI	ClassType	PartSelectionRule	HgrSymbolPort(1):Type
	StandardSupportComponentClass	HS_Anvil.Anvil_FIG82A	HangersAndSupports\PartImages\HS_Anvi				1		HgrSymbolPort

Head	PartNumber	PartDescription	SymbolDefinition	N@From	N@To	N@UnitType	IJUAHgrSize:SIZE	IJUAHgrAnvil_FIG82A:A	IJUAHgrAnvil_FIG82A:B
Start									
	Anvil_FIG82A_0	Anvil Fig 82A Size 0 Short Type A Spring				in	0	0.5	
	Anvil_FIG82A_1	Anvil Fig 82A Size 1 Short Type A Spring				in	1	0.5	
	Anvil_FIG82A_2	Anvil Fig 82A Size 2 Short Type A Spring				in	2	0.5	
	Anvil_FIG82A_3	Anvil Fig 82A Size 3 Short Type A Spring				in	3	0.5	
	Anvil_FIG82A_4	Anvil Fig 82A Size 4 Short Type A Spring				in	4	0.5	
	Anvil_FIG82A_5	Anvil Fig 82A Size 5 Short Type A Spring				in	5	0.5	
	Anvil_FIG82A_6	Anvil Fig 82A Size 6 Short Type A Spring				in	6	0.625	
	Anvil_FIG82A_7	Anvil Fig 82A Size 7 Short Type A Spring				in	7	0.625	
	Anvil_FIG82A_8	Anvil Fig 82A Size 8 Short Type A Spring				in	8	0.625	
	Anvil_FIG82A_9	Anvil Fig 82A Size 9 Short Type A Spring				in	9	0.75	
	Anvil_FIG82A_10	Anvil Fig 82A Size 10 Short Type A Spring				in	10	0.75	
	Anvil_FIG82A_11	Anvil Fig 82A Size 11 Short Type A Spring				in	11	0.75	
	Anvil_FIG82A_12	Anvil Fig 82A Size 12 Short Type A Spring				in	12	1	
	Anvil_FIG82A_13	Anvil Fig 82A Size 13 Short Type A Spring				in	13	1	
	Anvil_FIG82A_14	Anvil Fig 82A Size 14 Short Type A Spring				in	14	1.25	
	Anvil_FIG82A_15	Anvil Fig 82A Size 15 Short Type A Spring				in	15	1.25	
	Anvil_FIG82A_16	Anvil Fig 82A Size 16 Short Type A Spring				in	16	1.5	
	Anvil_FIG82A_17	Anvil Fig 82A Size 17 Short Type A Spring				in	17	1.75	
	Anvil_FIG82A_18	Anvil Fig 82A Size 18 Short Type A Spring				in	18	2	
	Anvil_FIG82A_19	Anvil Fig 82A Size 19 Short Type A Spring				in	19	2.25	
	Anvil_FIG82A_20	Anvil Fig 82A Size 20 Short Type A Spring				in	20	2.5	

Anvil_FIG98A / Anvil_FIG95 / Anvil_FIG86 / Anvil_FIG82G / Anvil_FIG82F / Anvil_FIG82E / Anvil_FIG82D / Anvil_FIG82C / Anvil_FIG82B

– 29 –

SmartPlant 3D

Connection Support Component Part Class

Connection Support Component Class is a grouping of HgrBeam parts that share similar characteristics

3	Definition	PartClassType	SymbolDefinition	SymbolIcon	oa:IJOAhsMaterial:MaterialCategory	oa:IJOAhsMaterial:MaterialGrade	oa:EndOverLength	oa:BeginOverLength	oa:Length	oa:IJCoatingInfo:CoatingType	oa:IJCoatingInfo:CoatingRequirement	oa:IJCoatingInfo:CoatingColor	oa:IJUAhsCatalog:Catalog	ClassType	UserClassName	oa:IJOAhsEmbeddedPart:EmbeddedPar	oa:IJOAhsComponentNotes:Component
4																	
5		ConnectionSupportComponentClass	HS_SmartSteel.HgrBeam	HangersAndSupports\PartImages\HS_SmartSteel\HsSteel_W.gif										2	W		
6																	

8	Head	PartNumber	PartDescription	SymbolDefinition	MirrorBehaviorOption	IJOAhsMaterial:MaterialCategory	IJOAhsMaterial:MaterialGrade	IJCoatingInfo:CoatingType	IJCoatingInfo:CoatingRequirement	IJCoatingInfo:CoatingColor	IJUAHgrOccOverLength:EndOverLength	IJUAHgrOccOverLength:BeginOverLength	IJUAhsCatalog:Catalog	IJOAhsFlexPort:FlexPortXOffset	IJOAhsFlexPort:FlexPortYOffset	IJOAhsEmbeddedPart:EmbeddedPart	IJOAhsComponentNotes:ComponentNote
9																	
10	Start																
11		HsSteel_W44X335	W44X335		15	110	135	3	3	5	0	0	AISC-LRFD-3.1	0in	0in		
12		HsSteel_W44X290	W44X290		15	110	135	3	3	5	0	0	AISC-LRFD-3.1	0in	0in		
13		HsSteel_W44X262	W44X262		15	110	135	3	3	5	0	0	AISC-LRFD-3.1	0in	0in		
14		HsSteel_W44X230	W44X230		15	110	135	3	3	5	0	0	AISC-LRFD-3.1	0in	0in		

Revision History / Legend / CustomInterfaces / W / C / L / S / 2L / ST / WT / MT / MC / HP / HSSR / HSSC / Pipe / Cutback_W / Cu

– 30 –

The delivered reference data contains some workbooks whose main purpose is to define parts. As an example, we will take the Utility_VARIABLE_CYL sheet in HS_Utility.xls. This workbook defines part classes that have the following common properties. Depending on the attributes that you define when you create a part symbol, you may have to specify additional or different attributes than the ones listed below.

Support Component Part Class Class Definition

SmartPlant 3D

Part Class Type:	Type of Class (Example: Standard Support Component Class)
Class Type:	Support Class Type: 1, 2 or 3
Symbol Definition:	Default Geometric definition for all the parts under this class
SymbolIcon:	Specifies a graphic file for the preview dialog of the parts under this class
PartSelectionRule:	Customizable program used to select the correct part definition from a part class For example: HgrPipePartSelRule.CPartByPipeSize
UserClassName:	Sets the user name for the part class that appears in the Catalog Browser
OccClassName:	Sets the occurrence name for the part class that appears in the filter dialog box
HgrSymbolPort Type:	Defines the Port Type
Occurrence Attributes:	Properties that you can change for a specific part in the model

–29–

Part Class Type - Specifies the class type of the part.
Symbol Definition - Specifies the symbol used to represent the part. For more information on creating part symbols, refer to the *SmartPlant 3D Symbols Reference Guide*.
Symbol Icon - Specifies the graphic file that contains a picture of the part symbol. The graphic file is used as a preview in the software. Specify the path to the graphic file relative to the symbols folder that is shared on your software server.
Occurrence Attributes - Depending on the part symbol, there may be one or more occurrence attributes to define.
Class Type - Enter the class type code for the part.
Part Selection Rule - Specify the part selection rule. The Part Selection Rule automates the selection of items from the catalog using certain criteria. For example, pipe size (HgrPipePartSelRule.PartByPipeSize.cls), rod attachment size

(HgrPipePartSelRule.CPartByRodSize.cls), the beam profile rule (HgrPipePartSelRule.CHgrBeamForVPad.cls), by load factor (HgrPipePartSelRule.CPartByLoadFactor.cls), and so forth.

Port Type - Enter the type of port. For example, type HgrSymbolPort to specify that the symbol port is a hangers and supports port as opposed to a piping port.

User Class Name - Specify the user class name.

Support Component Part Class Parts

SmartPlant 3D

Part section consists of the following:

- Common Properties
- Selection Properties
- Custom Properties
- Port Data

	A	B	C	D	E	F	G	H	I
1	Definition	PartClassType	SymbolDefinition	SymbolIcon	PartSelectionRule	ClassType	UserClassName	HgrSymbolPort(1):Type	HgrSymbolPort(2):Type
2									
3		StandardSupportComponentClass	UBoltSet.CUBoltPart		HgrPipePartSelRule.CPartByPipeSizeEqual	1		HgrSymbolPort	HgrSymbolPort
4									
5	Head	PartNumber	PartDescription	SymbolDefinition	MirrorBehaviorOption	NDFrom	NDTo	NDUnitType	LIUAHgrPipe_Dia:Pipe_Dia
6									
7	Start								
8		URN-075			15	0.75	0.75	in	1.050
9		URN-001			15	1	1	in	1.315
10		URN-002			15	2	2	in	2.375
11		URN-004			15	4	4	in	4.500
12		URN-006			15	6	6	in	6.625
13		URN-008			15	8	8	in	8.625
14	End								
15									

CustomInterfaces / UBoltSet / FlexRod / WeldedBeamPart / ClassNodeType / R-Hierarchy / R-ClassNodeDescribes / GUIDs

– 17 –

Support Component Part

SmartPlant 3D

Common Properties

- IJDPart
  - PartNumber
  - PartDescription
  - MirrorBehaviorOption
- IJReplacementPart
  - ReplacementPartNumber
- IJCatalogWtAndCG
  - DryWeight
  - DryCogX
  - DryCogY
  - DryCogZ
  - WaterWeight
  - WaterCogX
  - WaterCogY
  - WaterCogZ
- IJHgrPart
  - PartType

Selection Properties

- IJHgrDiameterSelection
  - NDFrom
  - NDTo
  - NDUnitType

Custom Properties

- IJHgrBOMDefinition
  - BOMType
  - BOMDefinition
- IJUAHgrPipe_Dia
  - PIPE_DIA
- IJHgrPartMaterial
  - MaterialType
  - MaterialGrade

The Custom Interfaces sheet defines the customized user interfaces and attributes (properties) for the part classes.

Port Properties

- IJHgrSymbolPort
  - Name
  - MatingType
  - Category
  - ConnectionInfo
  - PortType
  - Size
  - MaxSize
  - MinSize
  - UnitType

– 18 –

Part Number - Type a part number. This part number must be unique across the entire reference data catalog.
Part Description - Enter a description for the part.
Symbol Definition - Specifies the symbol used to represent the part. You can use this field to override the symbol definition for the part class and to instead use a different symbol for an individual part number.
ND From - Enter the minimum nominal diameter for which the part can be used. This attribute is used in the selection of the part.
ND To - Enter the maximum nominal diameter for which the part can be used. This attribute is used in the selection of the part.
ND Unit Type - Enter the units for the ND From and ND To values.
Dry Weight - Enter the dry weight of the support component.
Water Weight - Enter the wet weight of the support component.
DryCogX - Enter the X-axis location of the dry center-of-gravity.
DryCogY - Enter the Y-axis location of the dry center-of-gravity.
DryCogZ - Enter the Z-axis location of the dry center-of-gravity.
Mirror Behavior Option - Enter the code that represents the mirror behavior for the part. Valid codes are listed in the AllCodeLists.xls workbook on the Mirror Behavior Option sheet in the Codelist Number column.

HEAD	MirrorBehaviorOption ShortDescription	MirrorBehaviorOption LongDescription	Codelist Number	Sort Order
!				
START				
	May be mirrored	Component may be mirrored	5	
	Rotate 180	Component may be rotated 180 degrees, but not mirrored	10	
	Can not be mirrored or rotated	Component can not be mirrored or rotated, but can be copied and translated	15	
	Replacement required	Component must be replaced rather than mirroring, rotating, or copying	50	
END				

IJHgrBOMDefinition::BOMType (Optional) – Enter the code that defines how the BOMDescription attribute for this part will be set. Valid codes are listed in the AllCodeLists.xls workbook on the HngSupBOMType sheet. If no value is entered, the BOM description for the part will default to the PartDescription text.

HEAD	HngSupBOMType ShortDescription	HngSupBOMType LongDescription	Codelist Number	Sort Order
!				
START				
	NoDescription	No Description	1	
	PartDefinitionDescription	Part Definition Description	2	
	FromSymbolDef	From Symbol Definition	3	
	FromBOMDef	From BOM Definition ProgID	4	
	FromBOMDefProgID	From BOM Definition	5	
END				

IJHgrBOMDefinition::BOMDefinition (Optional) - Specifies the BOM description. If **BOMType is set to 4** (FromBOMDefProgID), enter the ProgID to use for calculating the BOM description. If **BOMType is set to 5** (FromBOMDef), enter the text to use for the BOM description. You do not need to enter anything in this column for other values of the BOMType attribute.

For some of the properties listed below, *(i)* is an integer corresponding to a port. When you type the port properties into the workbook, substitute a number for *(i)*.

For every port *(i)* in the symbol, several attributes must be defined as below.

HgrSymbolPort(i):Name – Enter the name of this port. Any name is valid. This name must be unique for each port name column. The name will be used when defining joints in Assembly Information Rules.
HgrSymbolPort(i):Category – Enter the code that represents the category for the port. This attribute is no longer used but exists for backward compatibility. Valid codes are listed in the AllCodeLists.xls workbook on the HngSupPortCategory sheet in the Codelist Number column.
HgrSymbolPort(i):MatingType – Enter the code that represents the mating type for the port. This attribute is no longer used but exists for backward compatibility. Valid codes are listed in the AllCodeLists.xls workbook on the HngSupMatingJointType sheet in the Codelist Number column.

HgrSymbolPort(i):ConnectionInfo – Enter the code that represents the connection information for the port. This attribute is no longer used but exists for backward compatibility. Valid codes are listed in the AllCodeLists.xls workbook on the HngSupPhysicalConnection sheet in the Codelist Number column.
HgrSymbolPort(i):PortType – Enter the code that represents the port type for the port. Valid codes can be listed in the AllCodeLists.xls workbook on the HngSupPortType sheet.

HEAD	HngSupPortType ShortDescription	HngSupPortType LongDescription	Codelist Number	Sort Order
!	Each port in a part must have a Type, this is how those numbers correspond			
!				
START				
	Undefined	Undefined	1	
	Route	Route	2	
	Structure	Structure	3	
	Revolute	Revolute	11	
	Planar	Planar	12	
	Translation	Translation	13	
	Prismatic	Prismatic	14	
	Rigid	Rigid	15	
	Spherical	Spherical	16	
	Cylindrical	Cylindrical	17	
	PlanarSlot	PlanarSlot	18	
	External Thread, RH	External Thread, RH	1000	
	Internal Thread, RH	Internal Thread, RH	1001	
	External Thread, LH	External Thread, LH	1002	
	Internal Thread, LH	Internal Thread, LH	1003	
	Rod Hole	Rod Hole	1004	
	Pin/Bolt (for eye)	Pin/Bolt (for eye)	1005	
	Pin/Bolt (for double lug)	Pin/Bolt (for double lug)	1006	
	Eye	Eye	1007	
	Single Lug	Single Lug	1008	
	Double Lug	Double Lug	1009	
	Clevis	Clevis	1010	
	Steel	Steel	1011	
	Weld	Weld	1012	
	Bolted Connection	Bolted Connection	1013	
	Beam Clamp	Beam Clamp	1014	
	Pipe	Pipe	1015	
	Pipe Clamp	Pipe Clamp	1016	
	U-Bolt	U-Bolt	1017	
	Roller	Roller	1018	
	Saddle/Shield	Saddle/Shield	1019	
	Saddle/Shield Base	Saddle/Shield Base	1020	
	Pipe Attachment	Pipe Attachment	1021	
	Strut Eye	Strut Eye	1022	
	Strut Pin	Strut Pin	1023	
	Shock Absorber	Shock Absorber	1024	
	Shock Extension	Shock Extension	1025	
	Spring	Spring	1026	
	Spring Extension	Spring Extension	1027	
	Stanchion	Stanchion	1028	
	Trunnion	Trunnion	1029	
	Other	Other	1200	
	CableTray	Cable Tray	1300	
	Conduit	Conduit	1350	
	Duct	HVAC Duct	1400	
	HVACRoundDuct	HVAC Duct Round	1410	
	HVACRectangularDuct	HVAC Duct Rectangular	1420	
	HVACOvalDuct	HVAC Duct Oval	1430	
END				

HgrSymbolPort(i):Size – Enter the physical size of this port. For example, for a pin, enter its diameter.

HgrSymbolPort(n):MinSize – Enter the minimum size of the part port that can connect to this port. For example, for a pin port, is the minimum size of the eye that can connect to it.
HgrSymbolPort(i):MaxSize – Enter the maximum size of the part port that can connect to this port. For example, for a pin port, enter the maximum size of the eye that can connect to it.
HgrSymbolPort(n):UnitType – Enter a string representing the units for the port Size, MinSize and MaxSize attributes. Currently, only in and mm are valid strings for these sizes.

Custom Interfaces

The Custom Interfaces sheet allows you to load a custom schema into the metadata of a catalog. This sheet defines the customized user interfaces and attributes (properties) for the part classes in the workbook.

SmartPlant 3D

Custom Interfaces Sheet

InterfaceName	CategoryName	AttributeName	AttributeUserName	Type	UnitsType	PrimaryUnits	OnPropertyPage	ReadOnly	SymbolParameter
IJUAHgrOccLength	Dimensions	Length	Length	Double	distance	in	1	1	Length
IJHgrDiameterSelection	Standard	NDFrom	Nominal Diameter From	Double	scalar	0	1	1	
		NDTo	Nominal Diameter To	Double	scalar	0	1	1	
		NDUnitType	Nominal Diameter Unit Type	Char	0	0	1	1	
IJHgrPartMaterial	Standard	MaterialType	Material Type	Char	0	0	1	1	MaterialType
		MaterialGrade	Material Grade	Char	0	0	1	1	MaterialGrade
IJUAHgrPipe_Dia	Standard	PIPE_DIA	Pipe Diameter	Double	distance	in	1	0	PIPE_DIA
IJOAHgrInsulat	Standard	INSULAT	Insulation Thickness	Double	distance	in	1	0	INSULAT

- The Custom Interfaces sheet defines the customized user interfaces and attributes (properties) for the part classes in the workbook
- An interface is a collection of attributes
- Provides a mechanism to map the attributes of the part class to the symbol inputs
- When you bulk load the workbook, the software looks at each user attribute on the class sheets and uses the Custom Interfaces sheet to decode the information

– 19 –

When you make changes to catalog schema, such as adding custom attributes or new part classes you need to regenerate the views for the model, catalog and reports databases.

Reference Data Management

SmartPlant 3D

Step 1

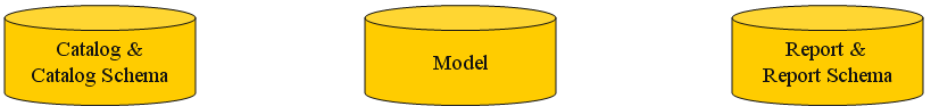
Bulk Load Utility

Step 2

View Generator Utility or Synchronize Model with Catalog Command

Step 3

Re-generate the reports database command

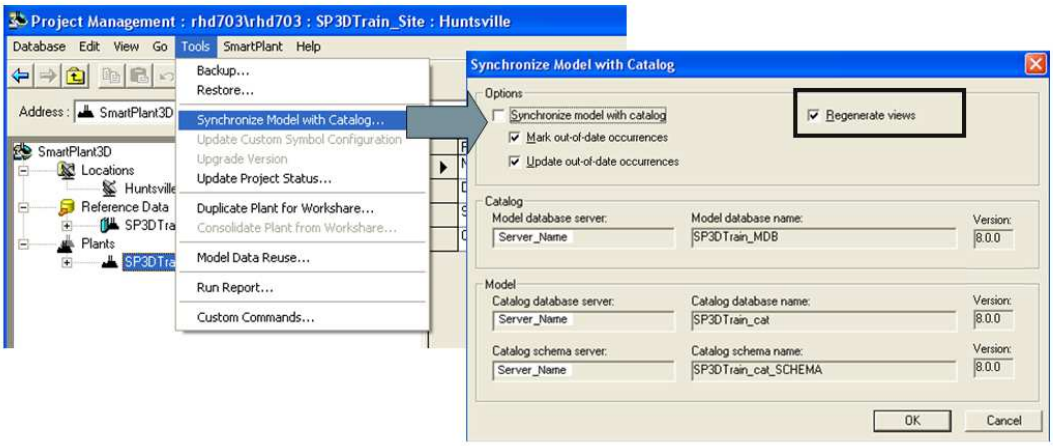


- Bulk load utility will update the views in the catalog database
- View Generator Utility will update the views in the model database
- Synchronize model with catalog command will update the views and re-compute all the part occurrences in the model
- Re-generate the report database command will re-generate the views in the report database

– 20 –

Reference Data Management

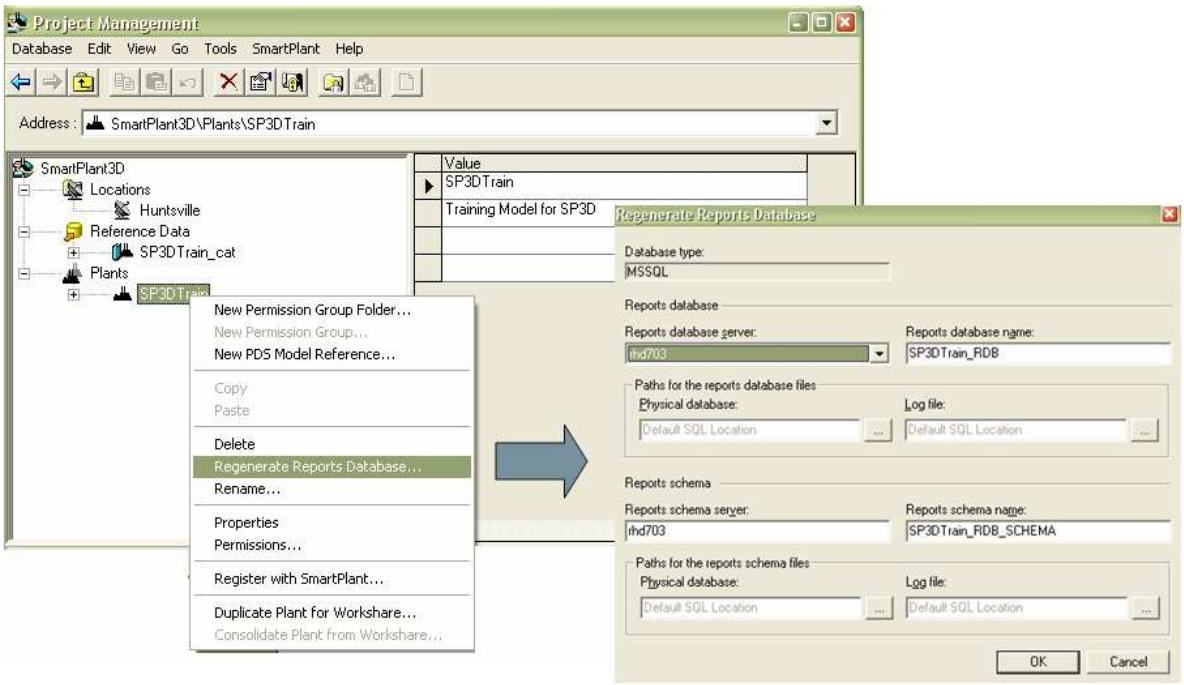
SmartPlant 3D



– 41 –

Reference Data Management

SmartPlant 3D

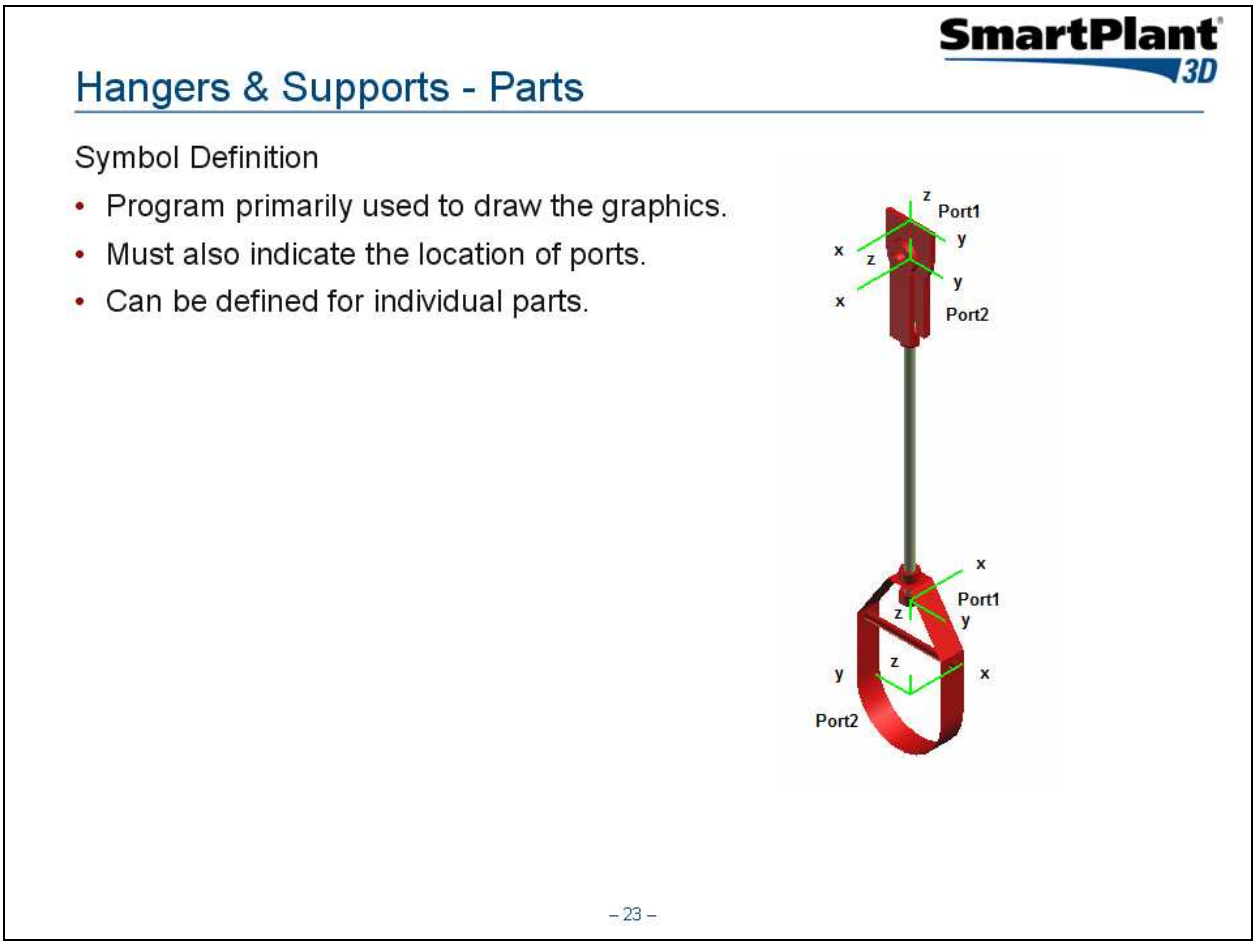


– 22 –

Part Symbol

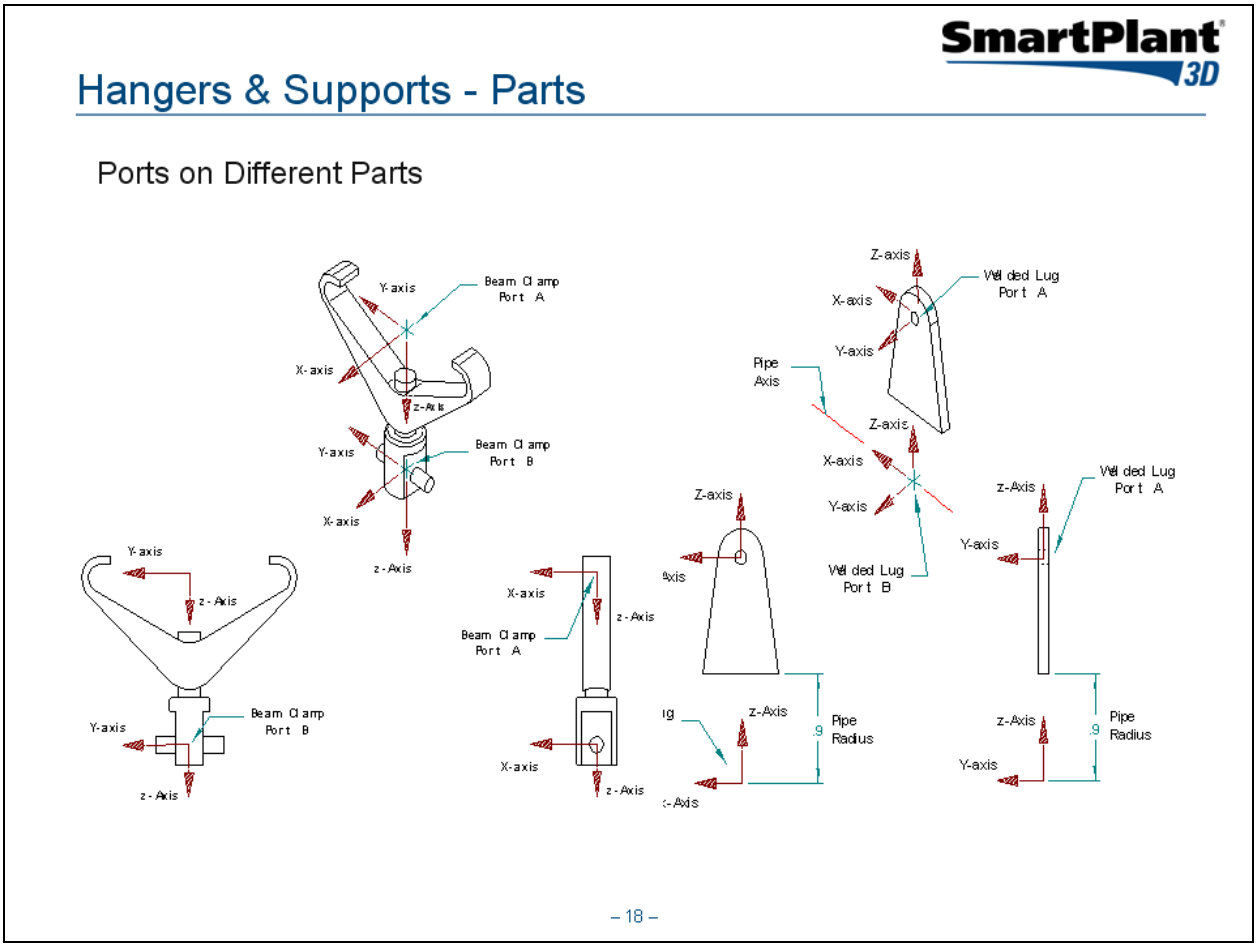
A symbol is a custom business object that provides the graphics representation of a support part.

Symbol Definition - A symbol definition is the persistent template for all symbols in a database. It is the definition of the inputs, outputs, and options of all symbols created using this symbol definition.

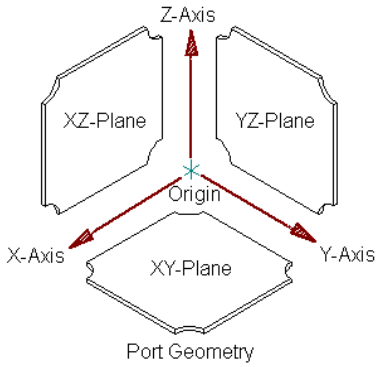


Understanding Part Ports

When creating part symbols that you intend to use in support assemblies, you should specify outputs called *ports* to aid in the placement of the part symbol. The following figure shows the output ports on different support part symbols.



The geometric information associated with each port is, in essence, a coordinate system. The port defines an origin, three axes, and three planes.



Ports convey information through their relative location with respect to the symbol graphics. In a support part symbol, one port is defined coincident with the centerline of the support part's associated pipe. The second port is positioned where the support part is connected to the other assembly part. Understand, however, that ports can be defined at any position in space and therefore do not need to be attached to the part symbol's graphical representation. In addition, the location of a port can be a function of the input of the part symbol. For example, you can define a port position for a welded lug based upon the pipe radius.

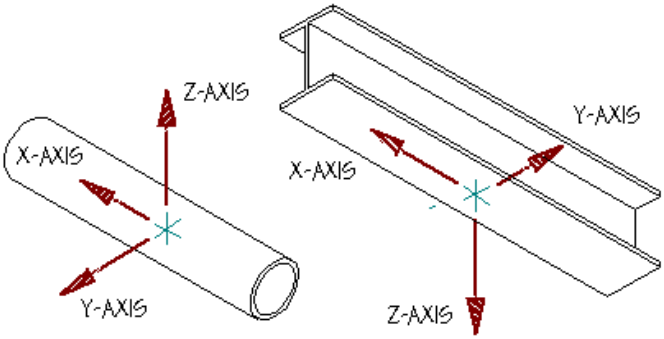
The following figure shows the part symbols and their ports arranged in the assembly. Each part has two ports defined; however, any number can be defined as required. The orientation and location of each part symbol port is independent of the other ports. During placement, however, a symbol's collection of ports is treated as rigid and will always retain their location with respect to the other ports and the geometry of the part symbol.

To display the temporary port axes in the active view, you first select the standard support component and then press **Ctrl + Shift + P** keys.

Understanding Reference Ports

While part ports define the connection relationship between parts in an assembly, *reference ports* define how a support assembly attaches to the supporting object (such as a beam) and the supported object (such as a pipe or cable tray). Unlike part symbol ports, reference ports are provided by the software and cannot be moved.

The example support mentioned earlier needs two reference ports, one on the beam and one on the pipe. The Z-axis of the supported object (in this case, the pipe) always points toward the supporting object (the beam). Conversely, the Z-axis of the supporting object (the beam) port always points toward the supported object (the pipe).

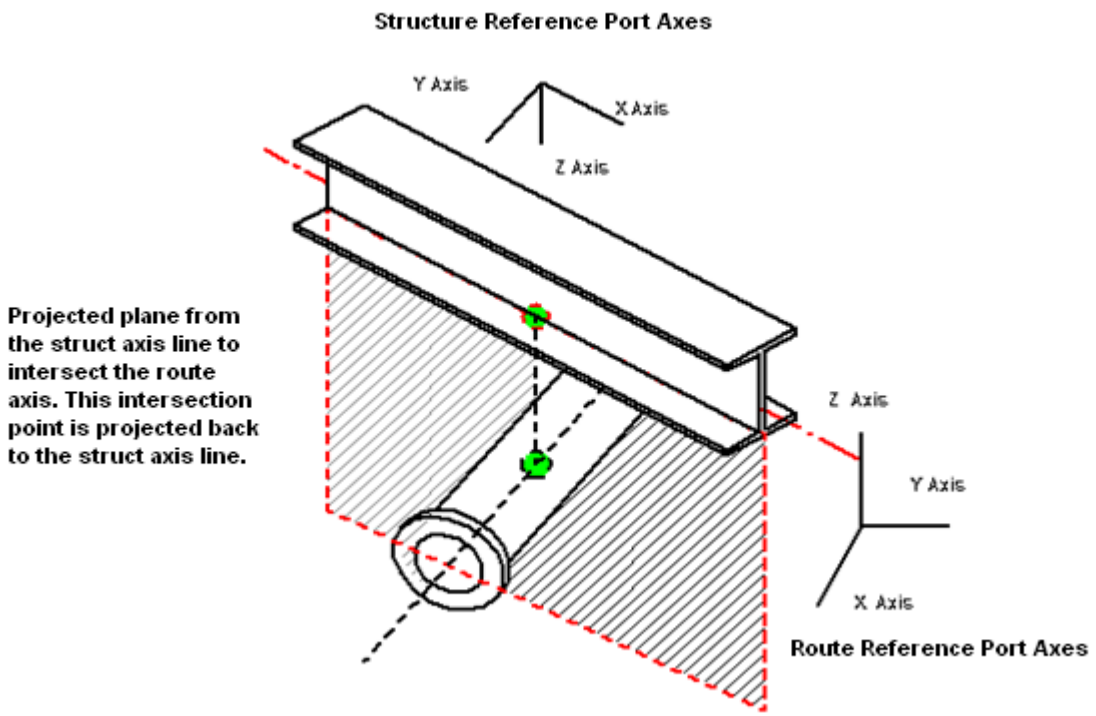


When you are defining assembly information rules in Visual Basic, the reference port on the pipe is called -1, ROUTE. The reference port on the beam is called -1, STRUCTURE. If you are connecting to more than one beam or structure, those ports are called -1, STRUCTURE; -1, STRUCT_2; and so forth.

Route Reference Port: The x direction of either the route port always aligns the flow direction. The z axis of the route port is pointing to the corresponding supporting (structure) ports. The y-axis is created using the right-handed rule.

Structure Reference Port: This port is located on the face of the supporting object to which a supported part will be attached and is aligned with the beam axis. The z axis of the structure reference port is pointing toward route reference port. The y-axis is created using the right-handed rule.

Following picture shows a simple example how the route and structure reference ports are calculated in the By Structure command case:



Local Coordinate System (LCS) for the Support

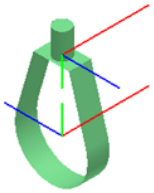
The LCS is always orientated with LCS North along the pipe. LCS east is orientated to the side of the pipe and LCS Elevation in the global up direction.

For sloped pipe, the LCS North is horizontal, not sloped with the pipe. There is one exception for vertical pipe, the LCS Elevation axis is not global up, but rather to the side of the pipe (generally torward the structure). North of the LCS is along the Route Z-axis (vertical) if Route x-axis is horizontal. Otherwise it is along the global up.

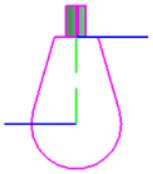
Hangers & Supports - Parts

Port Data

- Must be defined for each part
- Control the location and orientation of the part
- Define on both catalog and symbol
- Number of ports is specified in the Part Class Definition
- Type of the port must also be provided (HgrSymbolPort)



Adjustable Swivel Ring
Symbol Ports



– 32 –

Hangers & Supports - Parts

HgrSymbolPort(1):Type	HgrSymbolPort(2):Type
HgrSymbolPort	HgrSymbolPort

Port Type:
HgrSymbolPort

HgrSymbolPort(1):Name	HgrSymbolPort(1):Category	HgrSymbolPort(1):MatingType	HgrSymbolPort(1):ConnectionInfo	HgrSymbolPort(1):PortType	HgrSymbolPort(1):Size	HgrSymbolPort(1):MinSize	HgrSymbolPort(1):MaxSize	HgrSymbolPort(1):UnitType	HgrSymbolPort(2):Name	HgrSymbolPort(2):Category	HgrSymbolPort(2):MatingType	HgrSymbolPort(2):ConnectionInfo	HgrSymbolPort(2):PortType	HgrSymbolPort(2):Size	HgrSymbolPort(2):MinSize	HgrSymbolPort(2):MaxSize	HgrSymbolPort(2):UnitType
Route	2	14	5	1016	2.375	2.375	2.375	in	Pin	1	16	5	1005	1	0	1.5	in
Route	2	14	5	1016	4.500	4.500	4.500	in	Pin	1	16	5	1005	1	0	1.5	in
Route	2	14	5	1016	6.625	6.625	6.625	in	Pin	1	16	5	1005	1	0	1.5	in

Port Information:
Name
Category
Mating Type
ConnectionInfo
PortType
Size
MinSize
MaxSize
UnitType

– 36 –

Hangers & Supports - Parts

Port Data

Get the ports from the Part. These ports have been defined in the catalog

```
Set oPortFac = New HgrPortFactory

Set oElements = oPortFac.CreateHgrPortsFromPart(outputcoll.ResourceManager, part)
Set oHgrPort = GetPortByNameFromCollection(oElements, portName)

oHgrPort.SetOrigin locX, locY, locZ
oHgrPort.PutOrientation locX, locY, locZ, _
                        xDirX, xDirY, xDirZ, _
                       zDirX, zDirY, zDirZ

outputcoll.AddOutput output, oHgrPort
```

For the orientation, unit vectors need to be indicated for the x-direction and the z-direction.

Setting the Hanger Ports' Location and orientation in the symbol of the part.

**Function AddPort(portName As String, locX As Double, locY As Double, locZ As Double, _
xDirX As Double, xDirY As Double, xDirZ As Double, zDirX As Double, zDirY As Double, zDirZ As Double, _
outputcoll As IJDOutputCollection, output As String, ByVal part As Variant, Optional name As String = "None")**

– 27 –

Hangers & Supports - Parts

Port Data

- X Axis:

  If it is a part connected to the route, the x axis is preferable in the direction of the route. Similarly, the x axis should be in the direction of the orientation of the structure. If the part is internally connected, user judgment is required
- Z Axis:

  - For a Route part, the z axis should be from route to structure

  - For a Structure part, the z axis should be the direction from structure to route

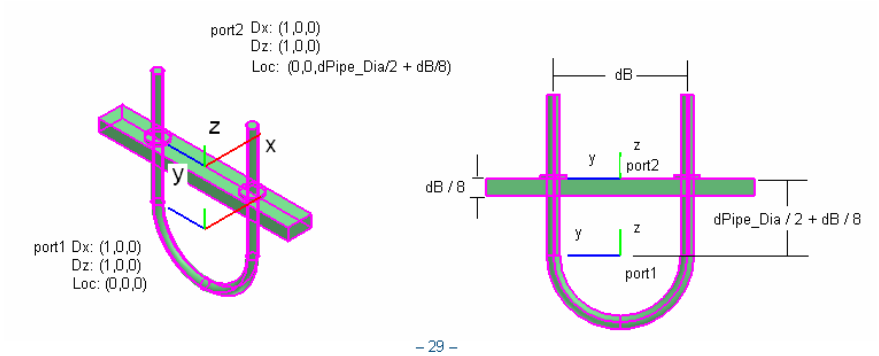
  - For a internal part, it is left to user judgment

– 28 –

Hanger & Supports - Parts

Example

```
dPipe_Dia = arrayOfInputs(2)
dA = arrayOfInputs(3)
dB = arrayOfInputs(4)
AddPort "Route", 0, 0, 0, _
        1, 0, 0,  0, 0, 1, m_outputColl, "Port1", arrayOfInputs(1), "Port1"
AddPort "Hole", 0, 0, dPipe_Dia / 2 + dB / 8, _
        1, 0, 0, 0, 0, 1, m_outputColl, "Port2", arrayOfInputs(1), "Port2"
```



– 29 –

Defining Assemblies: An Overview

The software properly positions the parts in relation to each other, the supported object (such as a pipe), and the supporting object (such as a beam) by applying *constraints* between *ports*. The assembly information rule is responsible for identifying all constraints necessary to position the parts correctly.

Specifying individual constraints provides the most flexibility and control over the assembly process. However, it is a tedious and repetitive process. To simplify things, relationships between parts are described using *joints* instead of individual constraints. Each joint represents a series of constraints that simulate the mechanical connection between parts. The assembly information rule provides a list of joints between assembly parts. The software parses this list and applies the appropriate constraints when creating the assembly.

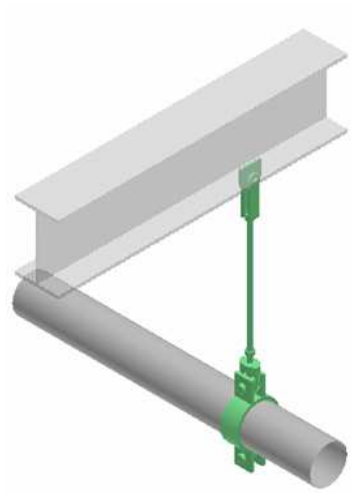
Each assembly defined in the reference data is assigned an *assembly information rule*. The software uses the logic contained in the assembly information rule to put together the individual support parts to form an assembly.

Hangers & Supports – Assembly Information Rule

SmartPlant 3D

Assembly Construction

- The graphical representation of a support is composed of a collection of symbols.
- Proper positioning of the symbols is accomplished by applying *constraints* between the *port* objects generated as output for each symbol.
- To define an assembly:
  - Define assembly catalog part (Sheet)
  - Define assembly information rule (AIR)



– 30 –

Hangers & Supports – Assembly Information Rule

SmartPlant 3D

What need to be defined in the catalog for the assembly?

- Assembly part class type
- Assembly part number
- Assembly information rule Prog ID
- Assembly attributes:

  SupportCount, DisciplineType, CommandType, TypeSelectionRule, and so on..

Optional

Occurrence attributes: This attribute can be used in the joint definition and even optional part selection

– 60 –

Hangers & Supports – Assembly Information Rule

SmartPlant 3D

Part Class Type: Type of Class (Example: PipeSupportDefinitionClass)
SymbolIcon: Specifies a graphic file for the preview dialog of the class
UserClassName: Sets the user name for the part class that appears in the Catalog Browser

Definition	*PartClassType*	SymbolIcon		UserClassName
	PipeSupportDefinitionClass			UBolt Assy Support

– 61 –

There is several assembly sheets already defined in the delivered support reference data. You can find the delivered workbook at *[Product Directory]*\CatalogData\Bulkload\DataFiles\HS_Assembly.xls.

When defining assemblies, the basic parameters are:

Part Class Type - Declare the type of support assembly. There are several types of support assemblies: PipeSupportDefinitionClass, DuctSupportDefinitionClass, CableTraySupportDefinitionClass, ConduitSupportDefinitionClass, and CombinedSupportDefinitionClass.

Occurrence Attributes - Specify any optional occurrence attributes on the part class level for the support assembly.

Part Number - Type a part number for the assembly. This name must be unique across the entire reference data catalog.

Part Description - Enter a description for the assembly.

Symbol Definition - Not used for assemblies. Leave empty.

Symbol Icon - Not used for assemblies. Leave empty.

Discipline Type - Enter the code that represents the type of object that this assembly can support. Valid codes are listed on the HngSupDiscipline sheet in the AllCodeLists.xls workbook in the Codelist Number column.

	HngSupDiscipline ShortDescription	HngSupDiscipline LongDescription	Codelist Number	Sort Order
HEAD				
!				
START				
	Piping	Piping Supports Discipline	1	
	Duct	Duct Supports Discipline	2	
	Piping & Duct	Piping & Duct Support Discipline	3	
	CableWay	Cableway Supports Discipline	4	
	CableWay & Piping	Cableway & Piping Supports Discipline	5	
	CableWay & Duct	Cableway & Duct Supports Discipline	6	
	CableWay & Piping & Duct	Cableway & Piping & Duct Supports Discipline	7	
	Conduit	Conduit Support Discipline	8	
	Conduit & Piping	Conduit & Piping Support Discipline	9	
	Conduit & Duct	Conduit & Duct Support Discipline	10	
	Conduit & Piping & Duct	Conduit & Piping & Duct Support Discipline	11	
	Conduit & Cableway	Conduit & Cablewat Support Discipline	12	
	Equipment	Equipment Support Discipline	16	
!	Others	Others		
	Combined	Combined Support Discipline	256	
	All	All Support Disciplines	511	
	Piping Designed	Piping Designed Discipline	1025	
	Duct Designed	Duct Designed Discipline	1026	
	Cable Tray Designed	Cabletray Designed Discipline	1028	
	Conduit Designed	Conduit Designed Discipline	1032	
	Combined Designed	Combined Design Discipline	1280	
END				

Command Type - Enter the code that specifies which Hangers and Supports task command can use the assembly. Valid codes are listed on the HngSupCommand sheet in the AllCodeLists.xls workbook in the Codelist Number column.

HEAD	HngSupCommand ShortDescription	HngSupCommand LongDescription	Codelist Number	Sort Order
!				
START				
	By Structure	Place By Structure Command	1	
	By Point	Place By Point Command	2	
	All	All Commands	511	
END				

Type Selection Rule - Enter the code that specifies the location of the support with respect to the route being supported. Valid codes are listed on the HngSupTypeSelectionRule sheet in the AllCodeLists.xls workbook in the Codelist Number column.

HEAD	HngSupTypeSelectionRule ShortDescription	HngSupTypeSelectionRule LongDescription	Codelist Number	Sort Order
!				
START				
	Top	Structure Above Route	1	
	Bottom	Structure Below Route	2	
	Above or Below	Structure Above OR Below of Route	3	
	Beside (Left OR Right)	Structure Besides Route	4	
	Right	Structure Right of Route	5	
	Left	Structure Left Of Route	6	
	Basic Directions	Structure Above-Below-Besides Route	7	
	Intersection	Structure Intersect with Route	8	
	Top Right	Structure Above Right of Route	10	
	Top Left	Structure Above Left of Route	11	
	Bottom Right	Structure Below Right Of Route	12	
	Bottom Left	Structure Below Left Of Route	13	
	All Directions	Structure On All Directions	15	
END				

Assembly Info Rule - Specifies the custom Visual Basic Assembly Information Rule to use to construct the assembly.

Hangers & Supports – Assembly Information Rule

Head	PartNumber	PartDescription	DisciplineType	CommandType	TypeSelectionRule	AssmInfoRule
Start						
	Assy_H2_Support_01	Assy_H2_Support_01	1	1	15	UBolt1Assy.CGENParts
	Assy_H2_Support_02	Assy_H2_Support_02	1	1	15	UBolt2Assy.CLRParts
End						

AssmInfoRule: Prog ID of VB program that defines the part and joint information to build the assembly.
DisciplineType is code listed and represents the type of discipline that this assembly can support.
CommandType is code-listed and specifies which Hangers and Supports task command can use the assembly.
TypeSelectionRule is code-listed that specifies the location of the support with respect to the route being supported.

– 33 –

Max Insulation - Not used. Set to -1.

Load Range Low – Not used.

Load Range High – Not used.

Min/Max Assembly Lengths: These are the values that are used to find out which assemblies fit the pipe steel configuration you are placing to.

Supporting Count - Specify the maximum number of supporting objects, a beam for example, to which the assembly can be attached. Enter 1+ to indicate that the support can be attached to a variable number of supporting objects.

Notes:

- Supporting count can also be 0
- Supporting/Supported – Can be set to any number 1, 2, 3, 4, 5 and we use the + to denote or more

Supported Count - Specify the maximum number of supported objects, a pipe for example, to which the assembly can be attached. Enter 1+ to indicate that the support can be attached to a variable number of supported objects.

ND From - Enter the minimum nominal diameter for the supported object. This option is only used for conduit and pipe assemblies.

ND To - Enter the maximum nominal diameter for the supported object. This option is only used for conduit and pipe assemblies.

ND Unit Type - Enter the units for the ND From and ND To values.

Nominal Width From - Enter the minimum width of the supported object for the assembly. Include the units of measurement, in or mm for example, when specifying this value.

Nominal Width To - Enter the maximum width of the supported object for the assembly. Include the units of measurement, in or mm for example, when specifying this value.

Nominal Depth From - Enter the minimum depth of the supported object for the assembly. Include the units of measurement, in or mm for example, when specifying this value. Enter 0 if the supported object is circular.

Nominal Depth To - Enter the maximum depth of the supported object for the assembly. Include the units of measurement, in or mm for example, when specifying this value. Enter 0 if the supported object is circular.

Hangers & Supports – Assembly Information Rule

SmartPlant 3D

MaxInsulation	LoadRangeLow	LoadRangeHigh	MinAssemblyLength	MaxAssemblyLength	Supporting_Count	Supported_Count	NDFrom	NDTo	NDUnitType	NominalWidthFrom	NominalWidthTo	NominalDepthFrom	NominalDepthTo
-1	0	0	0in	1000in	1	1	1	8	in	0.5in	50in	0	50in
-1	0	0	0in	1000in	1	1	1	8	in	0.5in	50in	0	50in

LoadRangeHigh/Low: load range of the support (Not used currently)
SupportingCount: max number of structure the assembly can be attached to
1 means it is applied only with one structure
1+ means it can be applied with multiple structure
SupportedCount: max number of supported object the assembly can be attached to.
1 means assembly will support one route object
1+ means assembly will support multiple route objects

Notes: Supporting count can also be 0
Supporting/Supported – Can be set to any number 1, 2, 3, 4, 5 and we use the + to denote or more.

– 50 –

Supported Family - Enter the type of supported object. Valid types are STRAIGHT, TURN, END, and PART. Using END and PART allow you to place hangers on in-line components. PART defines objects such as valves, flanges, tees. END defines end features (for example, reducers).

Face Selection Type - Enter the face selection combination that you would like to use for the assembly. This column specifies which faces on a structure member that the assembly attaches. These combinations are defined on the Face Selection sheet. You enter a number from the Selection Key column on the Face Selection sheet in this column.

Insulation Purpose - Enter the insulation purpose code for the supported object. Valid codes are listed on the Insulation Purpose sheet in the AllCodeLists.xls workbook in the Codelist Number column.

Support Type - Enter the type for the assembly. Valid codes are listed on the HngSupSupportType sheet in the AllCodeLists.xls workbook in the Codelist Number column.

HEAD	SupportsDiscipline ShortDescription	SupportsDiscipline LongDescription	SupportsType ShortDescription	SupportsType LongDescription	Codelist Number	Sort Order
!						
START						
	PipeSupport	Pipe Support			10	
			Pipe Support		11	
			Anchor		12	
			Anchor - Rotational		15	
			Longitudinal Guide Horizontal		21	
			Longitudinal Guide Vertical		22	
			Penetration Plate		23	
			Longitudinal Guide Vertical and Horizontal		30	
			Travel Stop		31	
			Shoe Longitudinal Guide Horizontal		41	
			Shoe Longitudinal Guide Vertical		42	
			Shoe Longitudinal Guide Vertical and Horizontal		43	
			Shoe Travel Stop		44	
			Rod Hanger		45	
			Rod Hanger Double		46	
			Variable Spring		54	
			Variable Spring Double		55	
			Constant Spring		60	
			Constant Spring Double		61	
			Variable Spring Base		64	
			Constant Spring Base		68	
			Dynamic Restraint		70	
			Sway Strut		72	
			Structural		74	
			Riser Clamp		80	
			Riser Clamp Variable Spring		81	
			Straight Pipe Leg Extension		90	
			Elbow Leg Extension		91	
			Dummy Leg		95	
	DuctSupport	Duct Support			100	
			Duct Support		101	
	CableSupport	Cable Support			200	
			Cable Support		201	
	ConduitSupport	Conduit Support			300	
			Conduit Support		301	
	CombinedSupport	Combined Support			400	
			Combined Support		401	
END						

Mirror Behavior Option - Enter the code that represents the mirror behavior for the assembly. Valid codes are listed in the AllCodeLists.xls workbook on the Mirror Behavior Option sheet in the Codelist Number column.

HEAD	MirrorBehaviorOption ShortDescription	MirrorBehaviorOption LongDescription	Codelist Number	Sort Order
!				
START				
	May be mirrored	Component may be mirrored	5	
	Rotate 180	Component may be rotated 180 degrees, but not mirrored	10	
	Can not be mirrored or rotated	Component can not be mirrored or rotated, but can be copied and translated	15	
	Replacement required	Component must be replaced rather than mirroring, rotating, or copying	50	
END				

Hangers & Supports – Assembly Information Rule

SmartPlant 3D

SupportedFamily	FaceSelectionType	InsulationPurpose	SupportType	MirrorBehaviorOption
STRAIGHT	0	2	11	15
STRAIGHT	0	2	11	15

SupportedFamily: type of supported feature the assembly can support
STRAIGHT, TURN, PART, END
this attribute will be used in the Assembly Selection Rule to selection appropriate assemblies for certain input objects when the rule is on

FaceSelectionType (optional): structural face selection key which defines applicable structural surfaces to be attached with the assembly

– 35 –

SmartPlant 3D

Assembly Information Rule (AIR)

- An COM object implementing the ***IJHgrAssmInfo*** interface.
- There are two major components needs to be defined within the AIR.
  - Parts definition: the parts that construct the assembly
  - Joints definition: geometrical relationship between the symbols ports
    - Prismatic
    - Spherical
    - Cylindrical, etc…

Workflow:

- Define the part class
  - specify the part classes/parts constituting the assembly
- Joints
  - define joints required to assembly the support
- Connection information
  - Define connection between symbol and route/structure

– 36 –

Assembly Information Rule Example

The assembly information for each support type is defined in assembly reference data using the AssmInfoRule attribute.

The AssmInfoRule attribute (short for Assembly Information Rule) contains the ProgId of a COM object implementing the IJHgrAssmInfo interface. In most cases, the rule is a Visual Basic class. The purpose of an Assembly Information Rule is to provide any required information about the assembly during the evaluation of a support. Information is acquired by calling the methods defined on the IJHgrAssmInfo interface.

There are five methods defined within the Visual Basic program:

- IJHgrAssmInfo::GetAssemblyCatalogParts() defines the part class and retrieves the parts.
- IJHgrAssmInfo::GetAssemblyJoints() defines the constraints between the parts.
- IJHgrAssmInfo::GetRouteConnections() defines which part is connected to the route.
- IJHgrAssmInfo::GetStructConnections() defines which part is connected to the structure.
- IJHgrAssmInfo::GetMaxRouteConnectionValue() defines the route toggle configurations supported by the Assembly Information Rule.

Assembly Information Rule (AIR)

Method of *IJHgrAssmInfo* interface

- GetAssemblyCatalogParts( )
  - Define part class/part
- GetAssemblyJoints( )
  - Specify joint data to assembly symbols into support assembly
- GetRouteConnections( )
  - Define route connection
- GetStructConnections( )
  - Define structure connection
- GetMaxRouteConnectionValue()
  - defines the route toggle configurations supported by the Assembly Information Rule

– 37 –

Assembly Information Rule (AIR)

GetAssemblyCatalogParts( )

```
Dim my_IJHgrInputConfigHlpr As IJHgrInputConfigHlpr
Set my_IJHgrInputConfigHlpr = pDispInputConfigHlpr
'=== ========== == ==== ===========
'Get Attributes on AIR
'=== ========== == ==== ===========
Dim PartClasses(3) As String
PartClasses(1) = "UBoltSet"
PartClasses(2) = "FlexRod"  'HngSupHgrAssemblySymbols.ThreadFlex (Flexible Rod)
PartClasses(3) = "G4G_1461_01"  'Beam attachment

'Return the collection of Catalog Parts
Set IJHgrAssmInfo_GetAssemblyCatalogParts =
   AddPartsByClassName(pDispInputConfigHlpr, PartClasses)
Exit Function
```

– 38 –

GetAssemblyCatalogParts Method

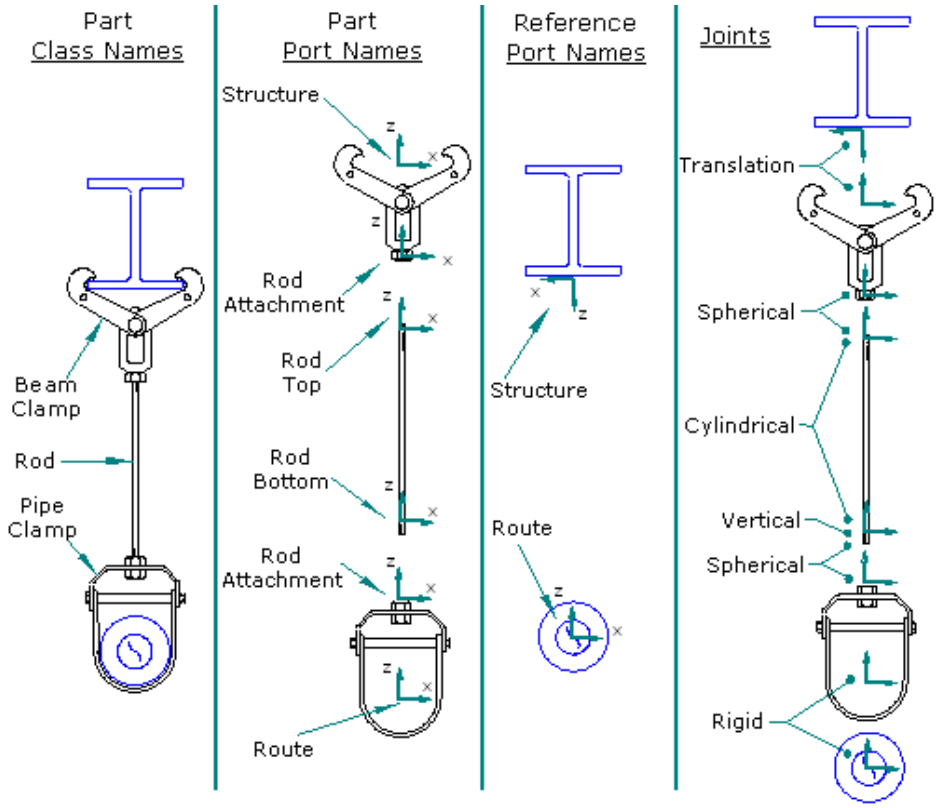
The GetAssemblyCatalogParts method of the IJHgrAssmInfo interface is responsible for specifying the catalog parts constituting the assembly. The method is defined as:

```
Function IJHgrAssmInfo_GetAssemblyCatalogParts (ByVal pDispInputConfigHlpr As Object ) As Object
```

The method should return a collection of objects implementing the IJDPart interface. The following shows a sample implementation of this method for the hanger support assembly shown in Figure 1.

```
Implements IJHgrAssmInfo
Private Function IJHgrAssmInfo_GetAssemblyCatalogParts(ByVal pDispInputConfigHlpr As Object) As Object

'Get IJHgrInputConfig Hlpr Interface of passed Helper

Dim my_IJHgrInputConfigHlpr As IJHgrInputConfigHlpr
Set my_IJHgrInputConfigHlpr = pDispInputConfigHlpr

'Create a new collection to hold the catalog parts

Dim CatalogPartCollection As New Collection

'Create the list of part classes required by the support assembly

Dim PartClasses(3) As String
PartClasses(1) = "Beam_Clamp"
PartClasses(2) = "Rod"
PartClasses(3) = "Pipe_Clamp"

'Declare an object to temporarily hold a Catalog Part

Dim PartProxy As Object

'Use the default selection rule to get a catalog part for each part class

Dim index As Integer
For index = 1 To UBound(PartClasses)
    Set PartProxy = my_IJHgrInputConfigHlpr.GetPartFromClass(PartClasses(index))
    CatalogPartCollection.Add PartProxy
Next

'Return the collection of Catalog Parts

Set IJHgrAssmInfo_GetAssemblyCatalogParts = CatalogPartCollection
End Function
```

Arguments for the method are limited to a single object named the Input Configuration Helper. This object offers access to the input specified by the user during the creation of the support as well as any services required by the Assembly Information Rule. In the above example, the GetPartFromClass method of the Input Configuration Helper is used to obtain a catalog part given a part class name.



Hanger Support with Associated Ports and Joints

Understanding Constraints

SmartPlant 3D

Constraints

Constraints allow placement of symbol occurrences relative to supporting and supported objects, as well as relative to each other. The process is managed with a user defined program, which will perform the process according to the user requested flexibility of the support.

Constraints are applied between

- Ports generated as outputs of symbols
- Reference Ports provided by the system to represent supporting and supported objects.

– 39 –

The software uses constraints to control how part symbol ports relate to each other and to reference ports. There are four constraints that can be defined by themselves or in conjunction with another constraint.

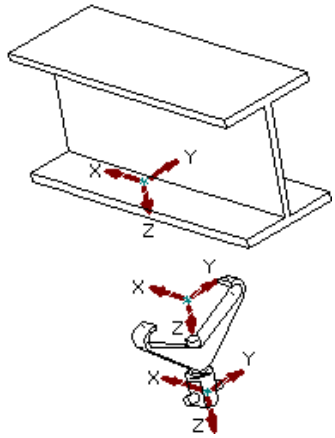
Parallel - Restricts geometries so that they are parallel. The constraint can only be applied between geometries with an associated direction. For ports, these geometries are axes and planes. A plane's direction is defined by its normal vector.

Perpendicular - Restricts geometries so that they are perpendicular. The constraint can only be applied between geometries with an associated direction. For ports, these geometries are axes and planes. A plane's direction is defined by its normal vector.

Coincident - Makes two geometries identical. Coincident constraints are only valid if one geometry can be made to touch everywhere with the other geometry.

Distance - Defines the minimum separation in three dimensions between two geometries. Unlike the other three constraints, this constraint has an associated value. A distance constraint can have different meanings depending on the geometry types and any other applied constraints.

As an example of constraints, consider this beam clamp and beam.



Two constraints allow the beam clamp to stay flush with the face of the beam while still allowing it to slide along the axis of the beam. The first constraint is parallel between the Beam XY-plane and the Beam Clamp XY-plane. The second constraint is coincident between the Beam X-axis and the Beam Clamp X-axis.

Understanding Joints

Constraints used between part symbols frequently model the mechanical attachment techniques used to connect the parts. These parts interact in a limited number of ways. The interaction of these part ports, the way in which each part can move with respect to the other part are defined by *joints*.

SmartPlant 3D

Joint Construction

- Positioning of the symbols is accomplished through the application of constraints between symbol ports
- Constraints are denoted as joints. Joints describes the relative geometric position between symbols and between symbol and reference ports
- Solver will solve the constraint sets and generate the appropriate location/orientation for each symbol

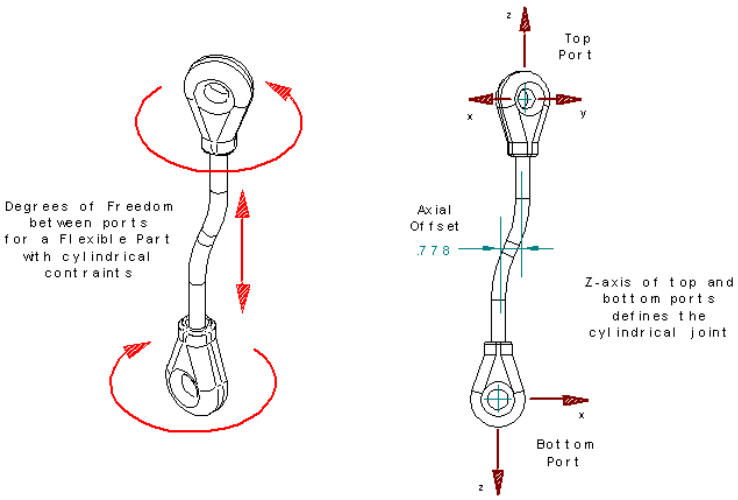
Joint	Description
Revolute	Objects can spin about the provided axis but do not translate.
Planar	Objects are free to rotate and translate while provided planes remain flush.
Translation	Objects can translate but not rotate while provided planes remain flush.
Prismatic	Objects can slide along provided axis but cannot rotate about the axis.
Rigid	Objects are rigidly fixed relative to each other.
Spherical	Object origins remain connected while objects are free to spin.
Cylindrical	Objects are free to spin about and translate along provided axis.

– 40 –

In three-dimensional space, there are six degrees of freedom: X, Y, Z, RX, RY, and RZ. A joint restricts the motion of parts with respect to each other in one or more of these degrees of freedom. By using joints, you do not have to specify individual constraints. However, you do need to specify the joint type, the part ports to join, and the required port geometry. For example, for the revolute joint shown, you need to specify the joint type (revolute), the pipe and turnbuckle ports to join, and the X-axis of rotation (port geometry) for the joint.

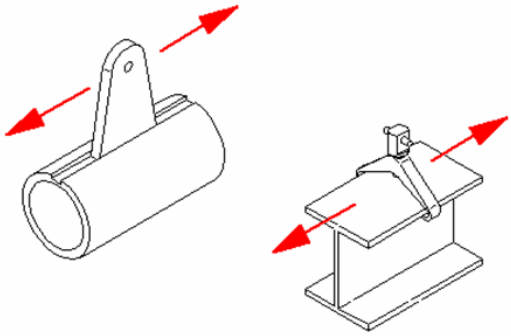
You can define more than one joint per port connection as long as the joints do not conflict with one another.

Cylindrical Joint



– 41 –

Prismatic Joint

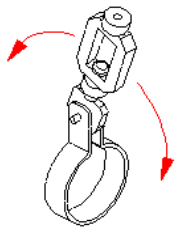
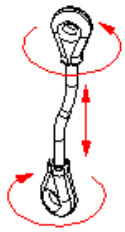
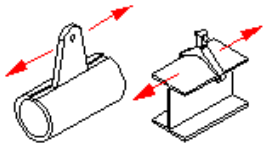
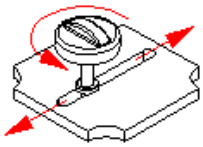
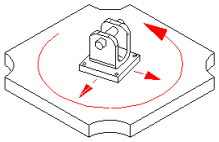
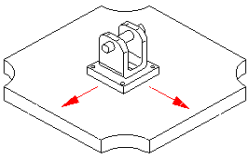


Prismatic Joints

One Translational degree of freedom

– 42 –

The following table provides pictures of some common joints.

	
Revolute Joint	*Cylindrical Joint*
	
Prismatic Joint	*Planar Slot Joint*
	
Plane Joint	*Translation Joint*

The next table lists joint and coupling types along with the associated degrees of freedom and a description of each joint.

Joint or Coupling Type	**Input Geometry**	**Associated Offset**	**Translational DOF**	**Rotational DOF**	**Description**
Revolute	1) Axis 2) Origin (implied)	1) No 2) No	0	1	Objects spin about provided axis but do not translate.
Planar	Plane	Yes	2	1	Objects free to rotate and translate while provided planes remain flush.
Translation	1) Plane 2) Axis (in specified plane)	1) Yes 2) No	2	0	Objects may translate but NOT rotate while provided planes remain flush.

Prismatic	1) Plane 2) Axis (in specified plane)	1) Yes 2) Yes	1	0	Objects may slide along provided axis but NOT rotate about the axis.
Rigid	1) Plane 2) Axis (in specified plane) 3) Origin (implied)	1) Yes 2) Yes 3) Yes	0	0	Objects are rigidly fixed relative to each other.
Spherical	Origin (implied)	No	0	3	Object origins remain in contact with each other while objects are free to spin.
Cylindrical	Axis	Yes	1	1	Objects are free to spin about and translate along provided axis.
Planar Slot	1) Plane 2) Axis (one object only)	1) Yes 2) Yes	1	1	While provided planes remain flush, objects can rotate about their normal and slide along the axis.
Horizontal	Axis/Plane (one object, world XY-plane implied for other)	No	n/a	n/a	Object becomes coincident with or lies in the horizontal plane.
Vertical	Axis/Plane (one object, world Y-axis implied for other)	No	n/a	n/a	Axis is made parallel to vertical axis; plane is forced to contain vertical axis.

SmartPlant 3D

Assembly Information Rule (AIR)

GetAssemblyJoints( )

```
'Create a Joint Factory
   Dim JointFactory As New HgrJointFactory
   'Create an object to hold the Joint
   Dim AssemblyJoint As Object
'Create the Flexible (Cylindrical) Joint between the ports of the rod
   Set AssemblyJoint = JointFactory.MakeCylindricalJoint(2, "RodTop", 2, "RodBottom")
   JointCollection.Add AssemblyJoint
   Set AssemblyJoint = JointFactory.MakeRigidJoint(1, "Route", -1, "Route")
   JointCollection.Add AssemblyJoint
   Set AssemblyJoint = JointFactory.MakeSphericalJoint(1, "Hole", 2, "RodBottom")
   JointCollection.Add AssemblyJoint
   Set AssemblyJoint = JointFactory.MakeRigidJoint(3, "Structure", -1, "Structure")
   JointCollection.Add AssemblyJoint
   Set AssemblyJoint = JointFactory.MakeSphericalJoint(3, "Rod", 2, "RodTop")
   JointCollection.Add AssemblyJoint
   Set AssemblyJoint = JointFactory.MakeVerticalJoint(2, "RodBottom")
   JointCollection.Add AssemblyJoint
```

GetAssemblyJoints Method

The GetAssemblyJoints method of the IJHgrAssmInfo interface provides the joints required to assemble the support. The method is defined as:

```
Function IJHgrAssmInfo_GetAssemblyJoints (
    ByVal pDispInputConfigHlpr As Object
    ByVal pDispPartOccCollection As Object
    ) As Object
```

The method should return a collection of objects implementing the IJHgrJoint interface. The following shows a sample implementation of this method for the hanger support assembly shown in Figure 1.

```
Implements IJHgrAssmInfo
Private Function IJHgrAssmInfo_GetAssemblyJoints( ByVal pDispInputConfigHlpr As Object,
ByVal pDispPartOccCollection As Object ) As Object

'Get IJHgrInputConfig Hlpr Interface off of passed Helper

Dim my_IJHgrInputConfigHlpr As IJHgrInputConfigHlpr
Set my_IJHgrInputConfigHlpr = pDispInputConfigHlpr
'Set Attributes on Part Occurrence
'====================
'Get the Diameter of the Primary Pipe
Dim PipeRadius As Double
PipeRadius = (my_IJHgrInputConfigHlpr.PrimaryPipeDiameter) / 2#

'Get interface for accessing items on the collection of Part Occurrences
Dim IJElements_PartOccCollection As IJElements
Set IJElements_PartOccCollection = pDispPartOccCollection

'Get the Pipe Clamp from the collection. It will be the third Part Occurrence.
Dim PipeClamp As Object
Set PipeClamp = IJElements_PartOccCollection.Item(3)

'Set the Pipe Radius Attribute on the Pipe Clamp Symbol
Dim strPipeRadius As String
strPipeRadius = "PipeRadius"
my_IJHgrInputConfigHlpr.SetSymbolInputByName PipeClamp, strPipeRadius, PipeRadius

'Create Joints
'========
'Create a collection to hold the joints
Dim JointCollection As New Collection
'Create a Joint Factory
Dim JointFactory As New HgrJointFactory
'Create an object to hold the Joint
Dim AssemblyJoint As Object

'Create the Flexible (Cylindrical) Joint between the ports of the rod
Set AssemblyJoint = JointFactory.MakeCylindricalJoint(2, "RodTop", 2, "RodBottom")
```

```
JointCollection.Add AssemblyJoint

'Add a Rigid Joint between Pipe and Pipe Clamp
Set AssemblyJoint = JointFactory.MakeRigidJoint(3, "PipeAttachment", -1, "Pipe")
JointCollection.Add AssemblyJoint

'Add a Translation Joint between the Beam and the Beam Clamp
Set AssemblyJoint = JointFactory.MakePrismaticJoint(1, "BeamAttachment", -1, "Structure")
JointCollection.Add AssemblyJoint

'Add a Spherical Joint between Beam Clamp and Top of Rod
Set AssemblyJoint = JointFactory.MakeSphericalJoint(1, "RodAttachment", 2, "RodTop")
JointCollection.Add AssemblyJoint

'Add a Spherical Joint between Pipe Clamp and Bottom of Rod
Set AssemblyJoint = JointFactory.MakeSphericalJoint(3, "RodAttachment", 2, "RodBottom")
JointCollection.Add AssemblyJoint

'Add a Vertical Joint to the Rod Z axis
Set AssemblyJoint = JointFactory.MakeVerticalJoint(2, "RodBottom")
JointCollection.Add AssemblyJoint

'Return the collection of Joints
Set IJHgrAssmInfo_GetAssemblyJoints = JointCollection
End Function
```

Two objects are passed as input to the method. The first is the Input Configuration Helper. The second is a collection of part occurrences representing the support. The part occurrences in this collection are listed in same order as the catalog parts returned by the GetAssemblyCatalogParts method. Hence, the first part occurrence corresponds to the first catalog part. The second part occurrence corresponds to the second catalog part, and so on. The part occurrences are provided to allow the Assembly Information Rule an opportunity to set inputs on the part occurrences associated symbol. In the previous example, the "PipeRadius" input is set on the pipe clamp using information provided by the Input Configuration Helper.

The collection returned by the method contains a list of Hanger Joint objects. These objects are created using a Joint Factory (HgrJointFactory object). The Factory implements the IJHgrJointFactory interface, which contains the following methods:

```
Function MakeRevoluteJoint (
    Integer PartIndex_A, String PortName_A,
    Integer PartIndex_B, String PortName_B,
    Integer ConfigIndex ) As Object
Function MakePrismaticJoint (
    Integer PartIndex_A, String PortName_A,
    Integer PartIndex_B, String PortName_B,
    Integer ConfigIndex,
    Double PlaneOffset, Double AxisOffset ) As Object
Function MakeRigidJoint (
    Integer PartIndex_A, String PortName_A,
    Integer PartIndex_B, String PortName_B,
    Integer ConfigIndex, Double PlaneOffset,
    Double AxisOffset, Double OriginOffset ) As Object
```

```
Function MakeTranslationJoint (
        Integer PartIndex_A, String PortName_A,
        Integer PartIndex_B, String PortName_B,
        Integer ConfigIndex,
        Double PlaneOffset ) As Object
Function MakeVerticalJoint (
        Integer PartIndex_A, String PortName_A,
        Integer ConfigIndex ) As Object
Function MakeSphericalJoint (
        Integer PartIndex_A, String PortName_A,
        Integer PartIndex_B, String PortName_B ) As Object
Function MakeCylindricalJoint (
        Integer PartIndex_A, String PortName_A,
        Integer PartIndex_B, String PortName_B,
        Integer ConfigIndex,
        Double AxisOffset) As Object
Function MakePrismaticJoint (
        Integer PartIndex_A, String PortName_A,
        Integer PartIndex_B, String PortName_B,
        Integer ConfigIndex,
        Double PlaneOffset, Double AxisOffset,
        Double OriginOffset ) As Object
Function MakePlanarJoint (
        Integer PartIndex_A, String PortName_A,
        Integer PartIndex_B, String PortName_B,
        Integer ConfigIndex,
        Double PlaneOffset ) As Object
```

These methods create CHgrJoint objects. A CHgrJoint object retains sufficient information to define the constraints implementing any joint. Member data for the CHgrJoint object includes:

Joint Type (1): an enum describing the type of joint. The IJHgrJointFactory method names indicate the enum stored.

Symbol Indices (2): Symbols will be indicated as indices into the catalog part list provided as output to GetAssemblyCatalogParts method. These are the PartIndex_A and PartIndex_B arguments within the IJHgrJointFactory method prototypes.

Port Names (2): Port names indicate which output of the specified symbol to use when constructing the joint. These are the PortName_A and PortName_B arguments within the IJHgrJointFactory method prototypes.

Configuration Index (1): Each joint type requires a defining set of port geometries. For example, the prismatic joint requires a plane and axis be specified for each of the two participating ports. Because the geometry associated with each port is limited (3 planes and 3 axes), all possible combinations can be encoded in a single 32-Bit integer referred to as the Configuration Index. Configuration Indices can be calculated using a utility provided by development called the Configuration Index Calculator. If no Configuration Index is provided when calling an IJHgrJointFactory method, a default representing the most common case is used.

Offsets (3): Several joint types permit the specification of offsets between their defining geometries. For the most complex case, three offsets are allowed. Offsets are represented using three doubles. These are the PlaneOffset, AxisOffset and OriginOffset arguments within the IJHgrJointFactory method prototypes.

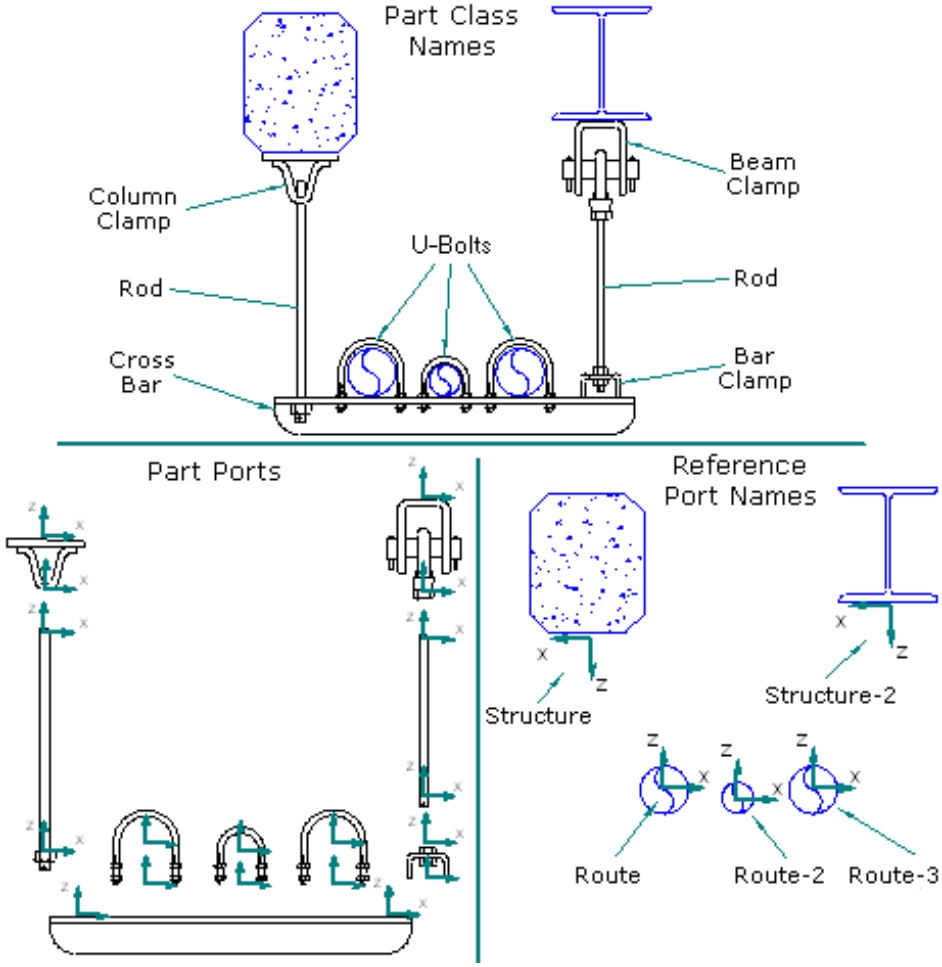
GetRouteConnections Method

The GetRouteConnections method of the IJHgrAssmInfo interface provides information specifying those components of the assembly that physically connect to the supported input objects. The method is defined as:

```
Function IJHgrAssmInfo_GetRouteConnections ( ByVal pDispInputConfigHlpr As Object) As Object
```

The method should return a collection of integer arrays. The following shows a sample implementation of this method for the structural support assembly shown in Figure 2.

```
Private Function IJHgrAssmInfo_GetRouteConnections( ByVal pDispInputConfigHlpr As Object ) As
Object
'Create a collection to hold the ALL Route connection information
Dim RouteCollofConnInfo As New Collection
'Create an ARRAY to hold Route Connection information for the Cross Bar
Dim CrossBarConnectionInfo(4) As Integer
CrossBarConnectionInfo (1) = 3 'Third Part return by GetAssemblyCatalogParts() is Cross Bar
CrossBarConnectionInfo (2) = 1 'Connects to First Route Reference Port
CrossBarConnectionInfo (3) = 2 'Connects to Second Route Reference Port
CrossBarConnectionInfo (4) = 3 'Connects to Third Route Reference Port
'Create an ARRAY to hold Route Connection information for the Left Ubolt
Dim Left_UBolt_ConnectionInfo(2) As Integer
Left_UBolt_ConnectionInfo(1) = 7 'Seventh Part return by GetAssemblyCatalogParts() is left most UBolt
Left_UBolt_ConnectionInfo(2) = 1 'Connects to First Route Reference Port
'Create an ARRAY to hold Route Connection information for the Middle Ubolt
Dim Middle_UBolt_ConnectionInfo(2) As Integer
Middle_UBolt_ConnectionInfo (1) = 8 'Eight Part return by GetAssemblyCatalogParts() is middle UBolt
Middle_UBolt_ConnectionInfo (2) = 2 'Connects to Second Route Reference Port
'Create an ARRAY to hold Route Connection information for the Right Ubolt
Dim Right_UBolt_ConnectionInfo(2) As Integer
Right_UBolt_ConnectionInfo (1) = 9 'Ninth Part return by GetAssemblyCatalogParts() is right most UBolt
Right_UBolt_ConnectionInfo (2) = 3 'Connects to Third Route Reference Port
'Add the Connection Information arrays to the returned Collection.
RouteCollofConnInfo.Add CrossBarConnectionInfo
RouteCollofConnInfo.Add Left_UBolt_ConnectionInfo
RouteCollofConnInfo.Add Middle_UBolt_ConnectionInfo
RouteCollofConnInfo.Add Right_UBolt_ConnectionInfo
'Return the collection of Route connection information.
Set IJHgrAssmInfo_GetRouteConnections = RouteCollofConnInfo
End Function
```



Trapeze Support with Associated Ports

In previous example, four arrays of integers are returned. These arrays correspond to the four assembly components physically touching the pipes: the cross bar and the three U-bolts. The first entry in each array is the index of the assembly part as referenced from the list returned in the GetAssemblyCatalogParts method. For the example, we assume the following part order (see Figure 2):

1. Column Clamp
2. Rod
3. Cross Bar
4. Bar Clamp
5. Rod
6. Beam Clamp
7. Left U-Bolt
8. Middle U-Bolt
9. Right U-Bolt

All entries following the first in each array refer to the route reference ports to which the assembly component connects. In the example, the first array returns: (3, 1, 2, 3). This is interpreted as: the third part returned by GetAssemblyCatalogParts (the cross bar), connects with the first, second, and third route reference ports (the left, middle, and right pipes). Similar arrays are returned for each of the U-Bolts; however, these connect with only a single pipe: (7, 1).This is interpreted as: the seventh part connects with the first route reference port.

GetStructConnections Method

The GetStructConnections method of the IJHgrAssmInfo interface provides information specifying those components of the assembly that physically connect to the supporting input objects. The method is defined as:

```
Function IJHgrAssmInfo_GetStructConnections ( ByVal pDispInputConfigHlpr As Object) As Object
```

The method should return a collection of integer arrays. The following shows a sample implementation of this method for the structural support assembly shown in Figure 2.

```
Private Function IJHgrAssmInfo_GetStructConnections( ByVal pDispInputConfigHlpr As Object ) As
Object
'Create a collection to hold the ALL Structure connection information
Dim StructCollofConnInfo As New Collection
'Create an ARRAY to hold Structural Connection information for the Column Clamp
Dim ColumnClampConnectionInfo(2) As Integer
ColumnClampConnectionInfo (1) = 1 'First Part return by GetAssemblyCatalogParts() is Column Clamp
ColumnClampConnectionInfo (2) = 1 'Connects to First Structural Reference Port
'Create an ARRAY to hold Structural Connection information for the Beam Clamp
Dim BeamClampConnectionInfo(2) As Integer
BeamClampConnectionInfo (1) = 6 'Sixth Part return by GetAssemblyCatalogParts() is Beam Clamp
BeamClampConnectionInfo (2) = 2 'Connects to Second Structural Reference Port
'Add the Connection Information arrays to the returned Collection.
StructCollofConnInfo.Add ColumnClampConnectionInfo
StructCollofConnInfo.Add BeamClampConnectionInfo
'Return the collection of Struct connection information.
Set IJHgrAssmInfo_GetStructConnections = StructCollofConnInfo
End Function
```

The format of the returned integer arrays is identical to those returned in the GetStructConnections method described above. For example, the two arrays returned in the above example correspond to the two parts physically touching the structural inputs, the column clamp and the beam clamp. The second of these arrays contains the following entries: (6, 2).This is interpreted as: the sixth part returned by GetAssemblyCatalogParts (the beam clamp), connects with the second structural reference ports (the beam).

Lookup Tables: An Overview

Lookup tables are used by the software to place supports in the model. The lookup tables are in the HS_System.xls workbook.

Face Selection Sheet

The Face Selection sheet in the HS_System.xls workbook defines the faces of a structural member to which a part can attach itself.

Face Name - Enter a unique name for the face selection option. This name must be unique across the entire catalog.

Description - Type a description for the face selection option.

Cross Section - Enter the cross section shape to which the face selection option applies. For example, W or L.

Selection Key - Enter a code to link this face selection option to parts. All face selection options with the same selection key code, 1 for example, will be made available to the part that has this code defined.

Face 1 - Enter the member face with the highest priority. This face will be default when the assembly is placed.

Group 1 - Enter the group key code. Valid codes are listed on the Face Position Selection sheet in the Group Key column. For more information about this code, see *Face Position Selection Sheet*, page 67.

Face 2 - Enter the member face with the next highest priority. This face will be the second choice when the assembly is placed. Enter 0 if you do not want to define another choice.

Group 2 - Enter the group key code. Valid codes are listed on the Face Position Selection sheet in the Group Key column.

Face 3 - Enter the member face with the third highest priority. This face will be the third choice when the assembly is placed. Enter 0 if you do not want to define another choice.

Group 3 - Enter the group key code. Valid codes are listed on the Face Position Selection sheet in the Group Key column.

Face 4 - Enter the member face with the four highest priority. This face will be the fourth choice when the assembly is placed. Enter 0 if you do not want to define another choice.

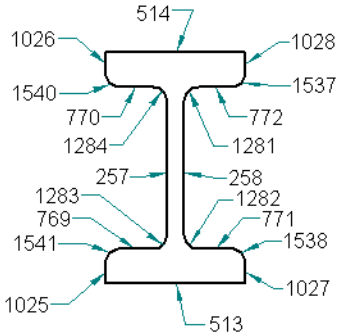
Group 4 - Enter the group key code. Valid codes are listed on the Face Position Selection sheet in the Group Key column.

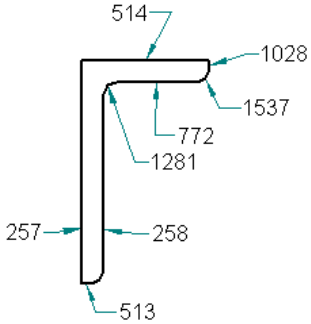
Face 5 - Enter the member face with the lowest priority. This face will be the last choice when the assembly is placed. Enter 0 if you do not want to define another choice.

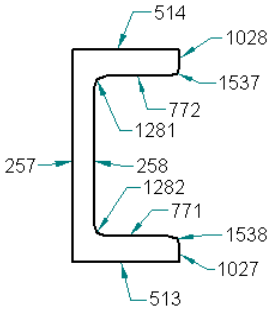
Group 5 - Enter the group key code. Valid codes are listed on the Face Position Selection sheet in the Group Key column.

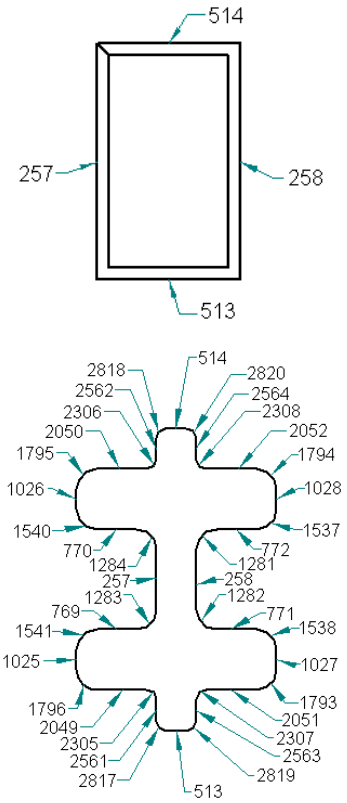
Available Face Numbers

These figures show the available face numbers for common section shapes that you can enter in the Face 1-5 boxes.









Face Position Selection Sheet

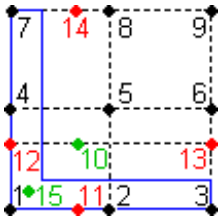
The Face Position Selection sheet in the HS_System.xls workbook defines the position on a face of a structure member where the part can locate itself. An example of when to use this sheet would be a U-bolt connected to the bottom of an I-shaped member. You would want the U-bolt to be placed on the out edges of the bottom flange so that the U-bolt did not interfere with the web of the member.

Point Sel Name - Enter a unique name for the face position selection option. This name must be unique across the entire catalog.

Description - Type a description for the face position selection option.

Group Key - Enter the group key for the face position selection option. This key is used to group face positions selection options together from which the user can select one of the face positions. This group key is also entered on the Face Selection sheet in the Group 1-5 columns.

Cardinal Point - Enter the member cardinal point from which to base the face position. There are 15 cardinal positions available. The location of cardinal points 10 (center-of-gravity) and 15 (shear center) depend on the section shape. The local z-axis of the member and the center-of-gravity point of the section define cardinal points 11 and 14. The local y-axis of the member and the center-of-gravity point of the section define cardinal points 12 and 13.



Offset - Enter the offset rule to use for the face position selection option (X-direction). The offset is measured from the cardinal point that you specified. Offset does not require a defined progID. The column can contain a numeric value.

OffSetY - Enter the offset rule to use for the Y-direction (perpendicular to face). OffsetY does not require a defined progID. The column can contain a numeric value.

SupportJoints Sheet

The SupportJoints sheet in the HS_System.xls workbook defines the joints that you can use to connect two parts. The sheet is used when you are defining supports by adding parts using the Place Part command.

Joint Key - Specifies the name of the joint key. This name must be unique within the column.

Mating Type 1 - Enter the first mating type code for the joint. Valid codes are listed on the HngSupPortType sheet in the AllCodeLists.xls workbook in the Codelist Number column.

HEAD	HngSupPortType ShortDescription	HngSupPortType LongDescription	Codelist Number	Sort Order
!	Each port in a part must have a Type, this is how those numbers corres			
!				
START				
	Undefined	Undefined	1	
	Route	Route	2	
	Structure	Structure	3	
	Revolute	Revolute	11	
	Planar	Planar	12	
	Translation	Translation	13	
	Prismatic	Prismatic	14	
	Rigid	Rigid	15	
	Spherical	Spherical	16	
	Cylindrical	Cylindrical	17	
	PlanarSlot	PlanarSlot	18	
	External Thread, RH	External Thread, RH	1000	
	Internal Thread, RH	Internal Thread, RH	1001	
	External Thread, LH	External Thread, LH	1002	
	Internal Thread, LH	Internal Thread, LH	1003	
	Rod Hole	Rod Hole	1004	
	Pin/Bolt (for eye)	Pin/Bolt (for eye)	1005	
	Pin/Bolt (for double lug)	Pin/Bolt (for double lug)	1006	
	Eye	Eye	1007	
	Single Lug	Single Lug	1008	
	Double Lug	Double Lug	1009	
	Clevis	Clevis	1010	
	Steel	Steel	1011	
	Weld	Weld	1012	
	Bolted Connection	Bolted Connection	1013	
	Beam Clamp	Beam Clamp	1014	
	Pipe	Pipe	1015	
	Pipe Clamp	Pipe Clamp	1016	
	U-Bolt	U-Bolt	1017	
	Roller	Roller	1018	
	Saddle/Shield	Saddle/Shield	1019	
	Saddle/Shield Base	Saddle/Shield Base	1020	
	Pipe Attachment	Pipe Attachment	1021	
	Strut Eye	Strut Eye	1022	
	Strut Pin	Strut Pin	1023	
	Shock Absorber	Shock Absorber	1024	
	Shock Extension	Shock Extension	1025	
	Spring	Spring	1026	
	Spring Extension	Spring Extension	1027	
	Stanchion	Stanchion	1028	
	Trunnion	Trunnion	1029	
	Other	Other	1200	
	CableTray	Cable Tray	1300	
	Conduit	Conduit	1350	
	Duct	HVAC Duct	1400	
	HVACRoundDuct	HVAC Duct Round	1410	
	HVACRectangularDuct	HVAC Duct Rectangular	1420	
	HVACOvalDuct	HVAC Duct Oval	1430	
END				

Mating Type 2 - Enter the second mating type code for the joint. Valid codes are listed on the HngSupPortType sheet in the AllCodeLists.xls workbook in the Codelist Number column.

Mating Option - Specifies the mating option. This value is not currently used in the software.

Command Type - Enter the code that specifies which Hangers and Supports task command can use the mating type combination. Valid codes are listed on the HngSupCommand sheet in the AllCodeLists.xls workbook in the Codelist Number column. This value is not currently used in the software.

Joint Type - Specifies the joint type by the joint code. Valid codes are listed on the HngSupJointType sheet in the AllCodeLists.xls workbook in the Codelist Number column. For more information about joints, see *Understanding Joints*, page 52. This value is not currently used in the software.

A short description of each joint type is listed below:

Code	Joint	Description
1	Revolute	Objects can spin about the provided axis but do not translate.
2	Planar	Objects are free to rotate and translate while provided planes remain flush.
3	Translation	Objects can translate but not rotate while provided planes remain flush.
4	Prismatic	Objects can slide along provided axis but cannot rotate about the axis.
5	Rigid	Objects are rigidly fixed relative to each other.
6	Spherical	Object origins remain connected while objects are free to spin.
7	Cylindrical	Objects are free to spin about and translate along provided axis.
8	Planar Slot	While provided planes remain flush, objects can rotate about their normal and slide along the axis.
9	Vertical	Object axis is made parallel to vertical axis, and the plane is forced to contain the vertical axis.
10	Horizontal X	Object becomes coincident with or lies in the horizontal plane.
11	Horizontal Y	Object becomes coincident with or lies in the horizontal plane.
12	Point On	Objects are point onto each other.

Configuration Index - Enter the configuration index number to use for the joint. You can calculate the configuration index number using the *Configuration Index Calculator*, page 71. This value is not currently used in the software.

Offset 1 - Specifies the plane offset between the parts at the joint. This value is not currently used in the software.

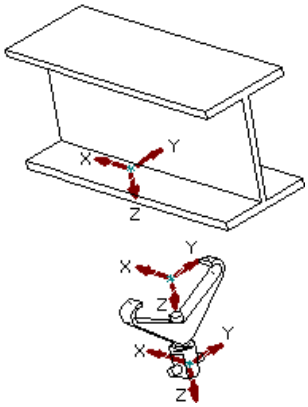
Offset 2 - Specifies the axis offset between the parts at the joint. This value is not currently used in the software.

Offset 3 - Specifies the origin offset between the parts at the joint. This value is not currently used in the software.

Configuration Index Calculator

The Configuration Index Calculator calculates the configuration index number used on the Support Joint sheet and in assembly information rules. The configuration index is an integer that specifies the geometry associated with each port in the joint. For example, the prismatic joint requires a plane and an axis be specified for each port. The configuration index tells the software which port plane and which port axis to use for each part and how those planes and axes are constrained. The Configuration Index Calculator is delivered with the software in the *[Product Directory]*\HangersAndSupports\Tools\Bin folder.

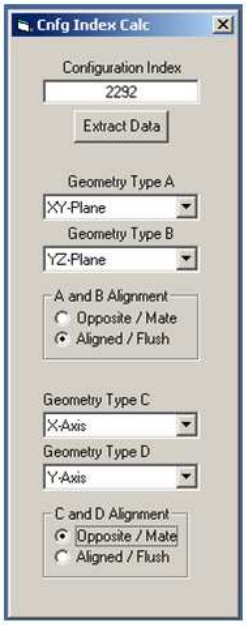
For an example of how to use the Configuration Index Calculator, look at the connection between this beam clamp and the beam. Two constraints allow the beam clamp to stay flush with the face of the beam while still allowing it to slide along the axis of the beam. The first constraint is parallel between the Beam XY-plane and the Beam Clamp XY-plane. The second constraint is coincident between the Beam X-axis and the Beam Clamp X-axis.



SmartPlant 3D

Configuration Index

- Each joint type requires a defining set of port geometries. Since the geometry associated with each port is limited (3 planes, 3 axes), all possible combinations can be encoded in a single 32-Bit integer referred to as the *Configuration Index*.
- Configuration Indices can be calculated using a utility



– 44 –

Configuration Index - Displays the configuration index number that you should enter on the Support Joint sheet in the Configuration Index column.

Extract Data - Calculates the configuration index number based on the inputs that you provide.

Geometry Type A - Select the geometry type for one of the port constraints in the connection. Using the example, you would select XY-Plane to represent the XY-Plane of the beam clamp port.

Geometry Type B - Select the geometry type for the other constraint port in the connection. Using the example, you would select XY-Plane to represent the XY-Plane of the beam port.

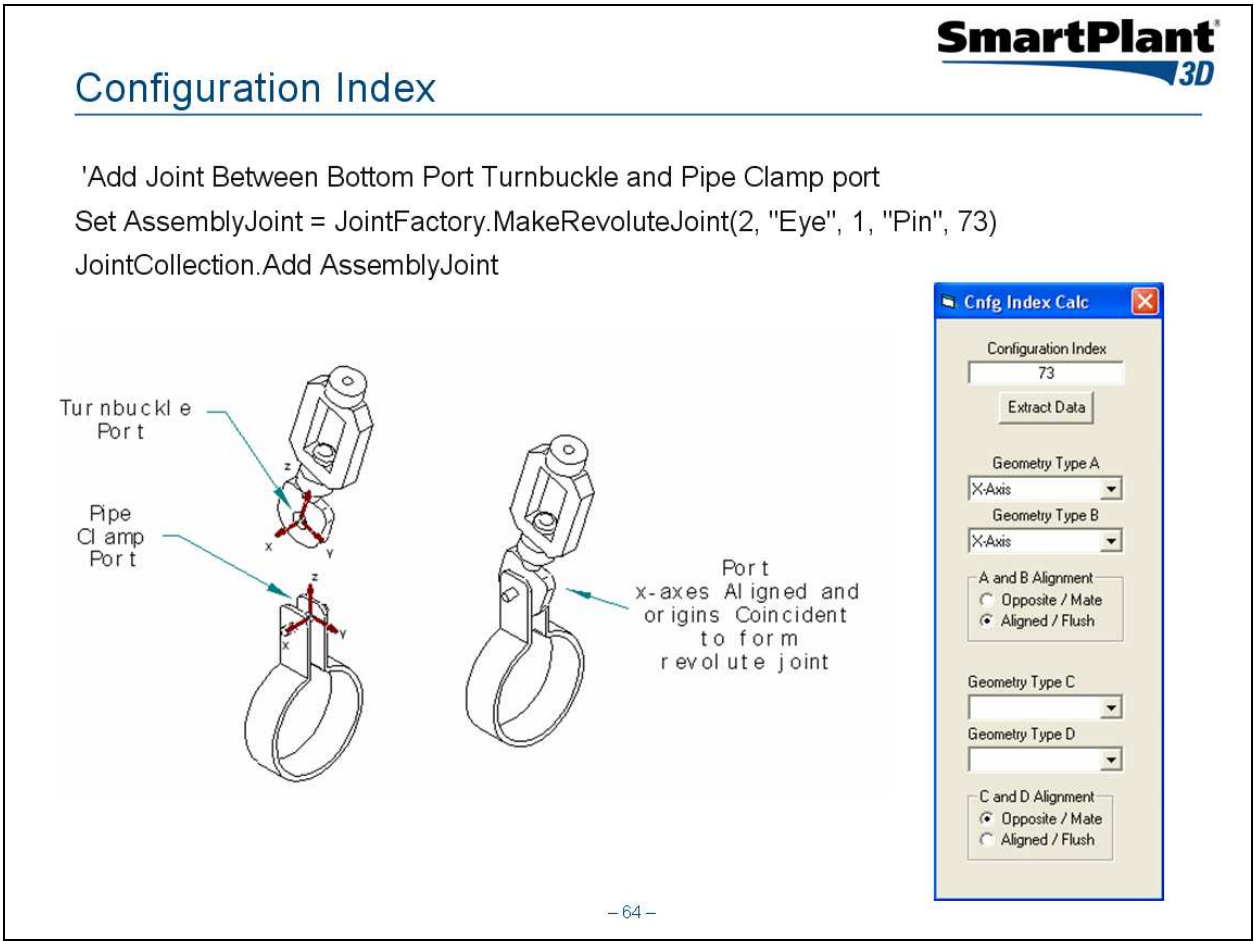
A and B Alignment - Select how you want the Geometry Type A and Geometry Type B selections to align. Using the example, you would select Opposite / Mate because you want the two XY-Planes to be co-planar, or mate.

Geometry Type C - Select the geometry type for the other port constraint in the connection. Using the example, you would select X-axis for the X-axis of the beam

clamp port. If the connection does not have a second constraint, you can leave this box blank.

Geometry Type D - Select the geometry type for the other constraint port in the connection. Using the example, you would select X-axis to represent the X-axis of the beam port. If the connection does not have a second constraint, you can leave this box blank.

C and D Alignment - Select how you want the Geometry Type C and Geometry Type C selections to align. Using the example, you would select Aligned / Flush because you want the two X-axes to be coincident.



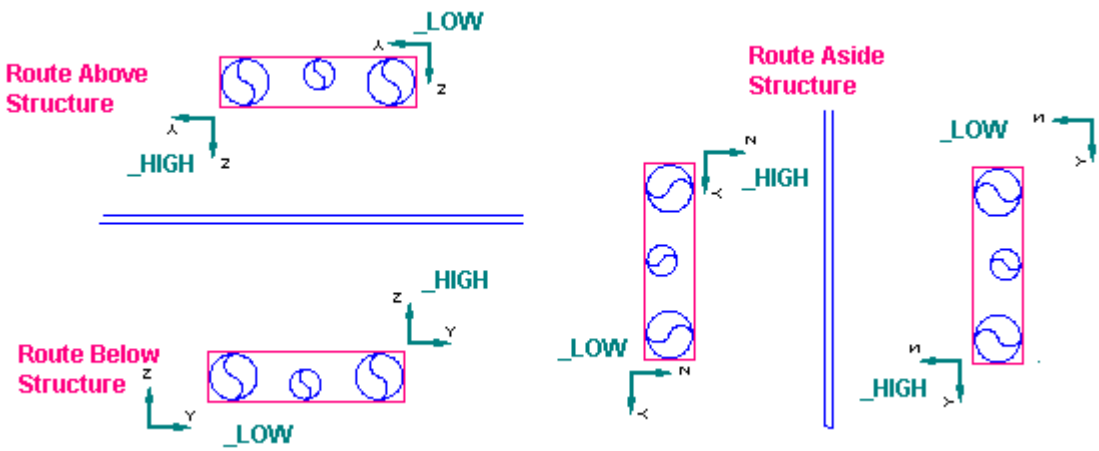
Bounding Box Reference Ports

Bounding box is a 2D box that defines the outer boundary of a collection of the route input objects. These boxes are useful to determine the range of the support in the case of multiple supported ports. In such cases, the extent of the support cannot be determined directly form the supported ports. An intersection plane and the LOW and HIGH points on the plane define bounding box. The ports defined on the bounding box also serve as reference ports and determine the orientation of the support. On every bounding box, the reference ports are defined on the end points of the diagonal from:

- the bottom left with the suffix "_Low",
- to the top right with the suffix "_High".

The prefix defines the plane used to generate the bounding box.

Following figure illustrated how the bounding box is defined:



Similar to route and structure reference port definition, the x axis of the bounding box port is pointing toward route flow direction, and z axis is pointing to the supporting structure. The port on the lower left corner of the bounding box (left means negative y direction, low means negative z direction) is defined as Low port, and port on the top right corner is defined as High port as is shown in the above picture.

Currently the following 9 types of bounding boxes are being created for the support.

- Bounding Box Route (BBR): This box is aligned along the cross-section of the primary supported port. In other words, the x-axis is identical to the route feature direction. This definition is most widely used in the By Point command support joint definition.

- Tangent_BBR: This port has been added to calculate the tangent bounding box when the number of pipes is more than one. This bounding box is aligned along the tangent to the set of pipe cross sections.

- Bounding Box Route Vertical (BBRV): This bounding box is aligned along one axis of the primary reference port and also along the global Z Axis.

- Bounding Box Structure (BBS): This bounding box is aligned along the plane of the supporting port.

- Bounding Box Structure Vertical (BBSV): This bounding box is aligned along the plane, which is vertical and aligned along the structure direction.

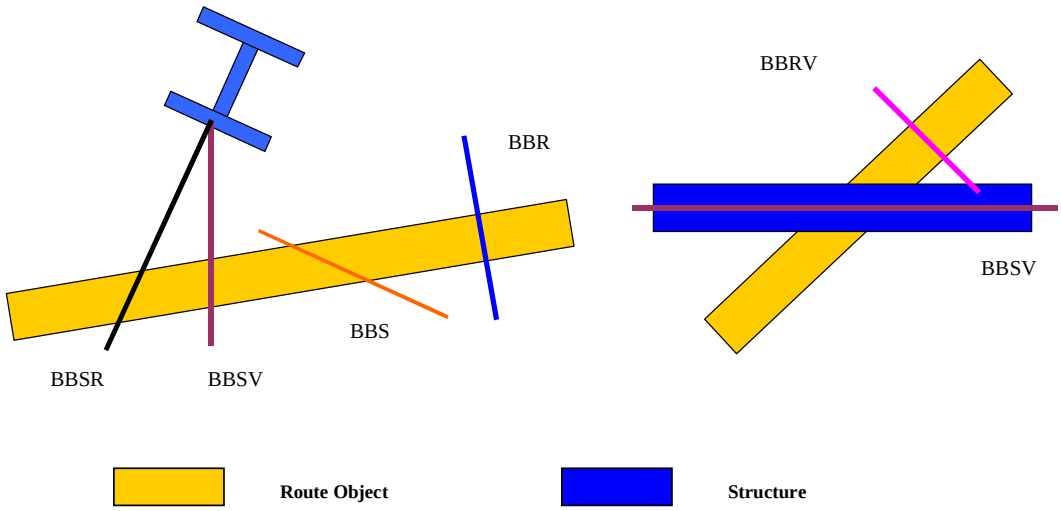
- Bounding Box Structure Route (BBSR): This bounding box is aligned along the plane perpendicular to the xy plane of the supporting reference port. This definition can be used in the By Struct command to ensure that the support is perpendicular to the structure's supporting plane.

- GBBR_X : This bounding box is created such that it's normal is aligned with the global X direction. i.e., the bounding box is in the global YZ plane.

- GBBR_Y: This bounding box is created such that it's normal is aligned with the global Y direction. i.e., the bounding box is in the global XZ plane.

- GBBR_Z: This bounding box is created such that orientation of the box is always aligned with the global Z axis.

The following pictures illustrate side view and top view of different bounding box planes:



It is important to know that all reference planes in the previous picture shows only orientation of the bounding box. The different planes have the same root point, which is the origin of the route reference port. This point is the origin of the “Route” port. The point is the intersection point which is generated by projecting the structural axis toward the pipe and intersecting with the pipe axis.

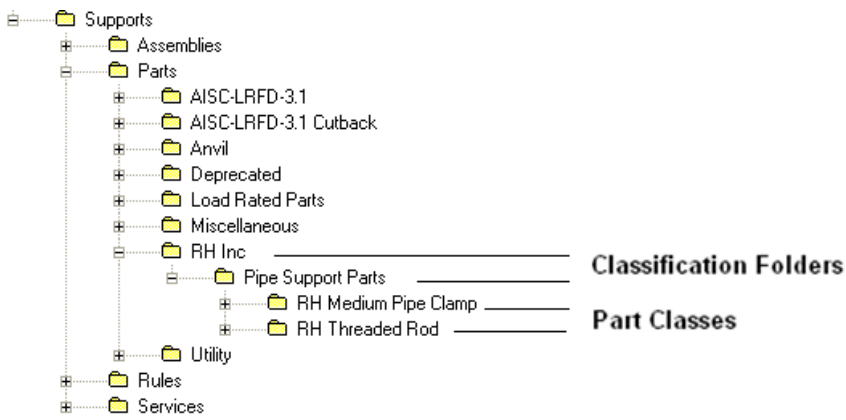
Lab 1: Support Catalog Hierarchy

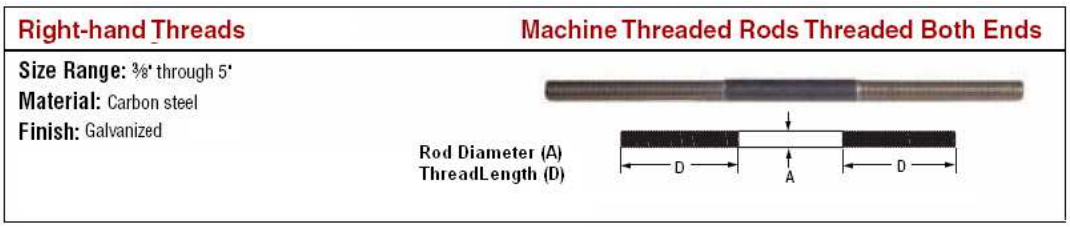
Objectives

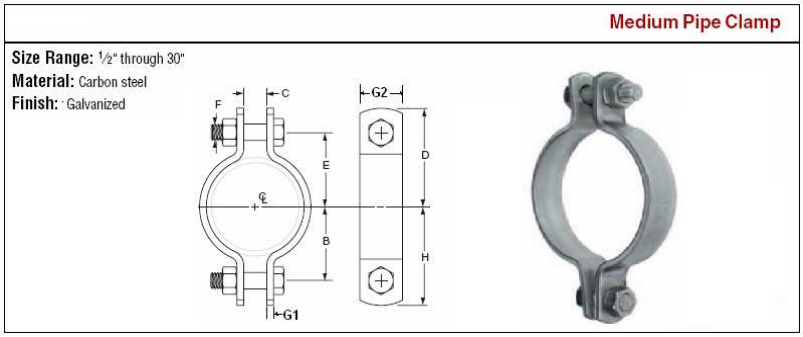
After completing this lab, you will be able to:

- Add and modify the support catalog hierarchy
- Add standard support components into your catalog
- Create standard support component part classes
- Create new interfaces

In this lab, you will learn to add new nodes in the support catalog hierarchy. You will obtain a detail understanding on how part data relates to data on the custom interfaces sheet, and how that information relates to parameters defined when the part symbol is created. Two standard support component part classes will be created to demonstrate the above concept. Delivered part symbols will be used for this exercise.







1. Create the following directory:

   c:\train

2. Copy the training files provided by the instructor to

   c:\train\HgrTrainingFiles

3. Create a directory called HS_RH_Parts under the c:\train:

   c:\train\ HS_RH_Parts

4. Copy the c:\train\HgrTrainingFiles\HgrTemplatePartV2\SPSupportTraining.xls workbook into HS_RH_Parts directory.

5. Open the SPSupportTraining.xls workbook.

6. You will add two new nodes (classification folders) called **RH Inc** and **Pipe Support Parts** into the support catalog hierarchy. To create the classification folders, you must edit the hierarchy sheets that control the catalog hierarchy: **ClassNodeType**, **R-Hierarchy**, and **R-ClassNodeDescribes**. When you add a part class to the catalog hierarchy, you must update these sheets, where necessary, in order to see the data in the catalog task or catalog browser dialog box.

7. Go to ClassNodeType sheet and add the following two rows. Remember to add the letter A to all new rows.

Head	ObjectName	Name
Start		
a	CLASSNODE_PART_SPCOMPANY	RH Inc
a	CLASSNODE_PART_SPCOMPANY_PARTS	Pipe Support Parts
End		

Note:

- *This sheet describes the nodes used to navigate the CatalogBrowser hierarchy.*
- ***ObjectName** should be unique across the entire database to identify a node uniquely. **Name** is the display name which appears in the CatalogBrowser hierarchy.*

8. Go to R-Hierarchy sheet and add the following two rows. This sheet defines where the nodes appear in the hierarchy.

Head	RelationSource	RelationDestination
Start		
a	CLASSNODE_PARTS_DEFINITION	CLASSNODE_PART_SPCOMPANY
a	CLASSNODE_PART_SPCOMPANY	CLASSNODE_PART_SPCOMPANY_PARTS
End		

Note:

- *RelationSource and RelationDestination entries are **ObjectNames** of the objects participating in the relationship. A relationship can't be made unless the objects mentioned in Relationsource / RelationDestination columns are already created.*

9. Next step, you will add two new part classes called **RH_SPCLAMP** and **RH_THREADED_ROD** into the support catalog hierarchy. You will need to create one entry in the R-ClassNodeDescribes sheet for each new part class.

10. Go to R-ClassNodeDescribes sheet and add the following two rows:

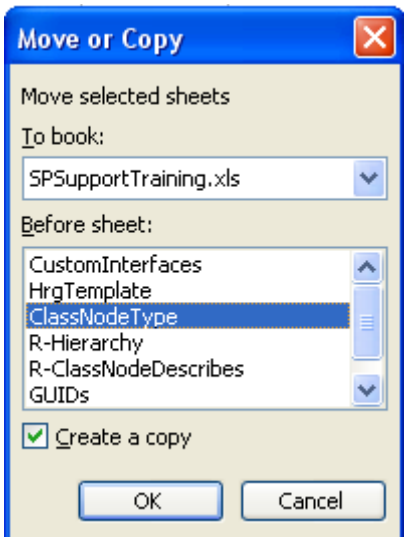
Head	RelationSource	RelationDestination
Start		
!	**Support Part Classes**	
a	CLASSNODE_PART_SPCOMPANY_PARTS	RH_SPCLAMP
a	CLASSNODE_PART_SPCOMPANY_PARTS	RH_THREADED_ROD
End		

Note:

- *'R-ClassNodeDescribes' is the relationship between an object from ClassNodeType worksheet as RelationSource and another object which is a PartClass as RelationDestination. RelationSource entries are **ObjectNames** of the objects participating in the relationship. RelationDestination entries are names of the PartClasses sheets participating in the relationship.*

11. Now, you will need to create the threaded rod part class. Use the HrgTemplate sheet as a template.

12. Select the HrgTemplate Sheet. Right Click to open the Move or Copy dialog box. Enable the copy option. Select the OK button to copy the sheet next to the ClassNodeType sheet.



13. Rename the sheet as RH_THREADED_ROD

Note:

- *Make sure you do not add any blank space or special characters for the name.*

14. Go to the class definition and edit the rows as shown below:

PartClassType	StandardSupportComponentClass
SymbolDefinition	HS_LRParts.LRParts_FIG140N
SymbolIcon	
PartSelectionRule	
ClassType	1
UserClassName	RH Threaded Rod
OccClassname	RH Threaded Rod
HgrSymbolPort(1):Type	HgrSymbolPort
HgrSymbolPort(2):Type	HgrSymbolPort

.

)efinition	PartClassType	SymbolDefinition	SymbolIcon	PartSelectionRule	ClassType	UserClassName	OccClassName	HgrSymbolPort(1):Type	HgrSymbolPort(2):Type
ı	StandardSupportComponentClass	HS_LRParts.LRParts_FIG140N			1	RH threaded Rod	RH threaded Rod	HgrSymbolPort	HgrSymbolPort

Note:

- *The UserClassName is also known as a definition class name. You can use this field to provide a more meaningful name for the class or a name with blank spaces in it. This name appears in the catalog task and catalog browser dialog box. The OccClassName appears in the business object hierarchy on the Object Type tab on the Filter Properties dialog boxes.*

15. Add occurrence attributes to this part class as shown below:

    OA:IJUAHgrOccLength::Length
    OA:IJUAHgrRHThreadedRod::WEIGHT_PER_LENGTH

OA:IJUAHgrOccLength::Length	OA:IJUAHgrRHThreadedRod::WEIGHT_PER_LENGTH

Notes:

- *Occurrence attributes are interpreted by the identifier '**OA:**' and are defined along with the definition of the PartClass in the Part Class worksheets.*
- *An attribute with the same name can exist on more than one interface. On our part class sheet, you can scope the user attributes based on their interfaces. The scoping syntax is **OA:Interface::UserAttribute.***

16. Now, you will need to add five attributes in the part definition row identified by the HEAD keyword.

    IJUAHgrRHThreadedRod::ROD_DIA
    IJUAHgrOccLength::Length
    IJUAHgrRHThreadedRod::D_USER
    IJOAHgrRodFinish::FINISH
    IJUAHgrRHThreadedRod::WEIGHT_PER_LENGTH

	PartNumber	PartDescription	SymbolDefinition	MirrorBehaviorOption	NDFrom	NDTo	NDUnitType	HgrSymbolPort(1):Name	HgrSymbolPort(1):Category	HgrSymbolPort(1):MatingType	HgrSymbolPort(1):ConnectionInfo	HgrSymbolPort(1):PortType	HgrSymbolPort(1):Size	HgrSymbolPort(1):MinSize	HgrSymbolPort(1):MaxSize	HgrSymbolPort(1):UnitType	HgrSymbolPort(2):Name	HgrSymbolPort(2):Category	HgrSymbolPort(2):MatingType	HgrSymbolPort(2):ConnectionInfo	HgrSymbolPort(2):PortType	HgrSymbolPort(2):Size	HgrSymbolPort(2):MinSize	HgrSymbolPort(2):MaxSize	HgrSymbolPort(2):UnitType	DryWeight	WaterWeight	DryCogX	DryCogY	DryCogZ	IJHgrBOMDefinition::BOMType	IJHgrBOMDefinition::BOMDefinition	IJUAHgrRHThreadedRod::ROD_DIA	IJUAHgrOccLength::Length	IJUAHgrRHThreadedRod::D_USER	IJOAHgrRodFinish::FINISH	IJUAHgrRHThreadedRod::WEIGHT_PER_LENGTH
Head																																					

17. Now, you will add three parts to the rod part class as shown below. Key-in the appropriate part number, part description and mirror behavior option (codelist).

Head	PartNumber	PartDescription	SymbolDefinition	MirrorBehaviorOption	NDFrom	NDTo	NDUnitType
Start							
a	RH-ROD_0.375	3/8 in End Threaded Rod Right Hand Thread		15			
a	RH-ROD_0.5	1/2 in End Threaded Rod Right Hand Thread		15			
a	RH-ROD_0.625	5/8 in End Threaded Rod Right Hand Thread		15			
End							

18. Next step, key-in the appropriate rod diameter, rod length, thread length, rod finish and the rod weight per length.

Note:

- Rod Finish: Valid codes can be listed in the **HS_Anvil_Codelist.xls** workbook on the **Anvil_Finish** sheet.

IJUAHgrRHThreadedRod::ROD_DIA	IJUAHgrOccLength::Length	IJUAHgrRHThreadedRod::D_USER	IJOAHgrRodFinish::FINISH	IJUAHgrRHThreadedRod::WEIGHT_PER_LENGTH
0.375	12	2.5	2	0.372
0.5	12	2.5	2	0.672
0.625	12	2.5	2	1.044

19. Key-in the appropriate port data as shown below:

Notes:

- **HgrSymbolPort(i):Name**: Enter the name of this port.

  The following attributes are not used in version 2009:
  - *HgrSymbolPort(i):Category: Valid codes are listed in the AllCodeLists.xls workbook on the HngSupPortCategory sheet in the Codelist Number column.*
  - *HgrSymbolPort(i):MatingType: Valid values can be listed in the AllCodeLists.xls workbook on the HngSupMatingJointType sheet.*
  - *HgrSymbolPort(i):ConnectionInfo:Valid codes are listed in the AllCodeLists.xls workbook on the HngSupPhysicalConnection sheet in the Codelist Number column.*

- *HgrSymbolPort(i):PortType: Valid codes are listed in the AllCodeLists.xls workbook on the HngSupPortType sheet in the Codelist Number column.*

HngSupPortType ShortDescription	HngSupPortType LongDescription	Codelist Number
External Thread, RH	External Thread, RH	1000
Internal Thread, RH	Internal Thread, RH	1001
External Thread, LH	External Thread, LH	1002
Internal Thread, LH	Internal Thread, LH	1003
Rod Hole	Rod Hole	1004
Pin/Bolt (for eye)	Pin/Bolt (for eye)	1005
Pin/Bolt (for double lug)	in/Bolt (for double lug)	1006
Eye	Eye	1007
Single Lug	Single Lug	1008
Double Lug	Double Lug	1009

- **HgrSymbolPort(i):Size** – Enter the physical size of this port.
  HgrSymbolPort(n):MinSize – Enter the minimum size of the part port that can connect to this port.
- **HgrSymbolPort(i):MaxSize** – Enter the maximum size of the part port that can connect to this port.
- **HgrSymbolPort(n):UnitType** – Enter a string In or mm representing the units for the port Size, MinSize and MaxSize attributes.

HgrSymbolPort(1):Name	HgrSymbolPort(1):Category	HgrSymbolPort(1):MatingType	HgrSymbolPort(1):ConnectionInfo	HgrSymbolPort(1):PortType	HgrSymbolPort(1):Size	HgrSymbolPort(1):MinSize	HgrSymbolPort(1):MaxSize	HgrSymbolPort(1):UnitType	HgrSymbolPort(2):Name	HgrSymbolPort(2):Category	HgrSymbolPort(2):MatingType	HgrSymbolPort(2):ConnectionInfo	HgrSymbolPort(2):PortType	HgrSymbolPort(2):Size	HgrSymbolPort(2):MinSize	HgrSymbolPort(2):MaxSize	HgrSymbolPort(2):UnitType
BotExThdRH				1000	0.375	0	4	in	TopExThdRH				1000	0.375	0	4	in
BotExThdRH				1000	0.5	0	4	in	TopExThdRH				1000	0.5	0	4	in
BotExThdRH				1000	0.625	0	4	in	TopExThdRH				1000	0.625	0	4	in

20. Key-in the appropriate BOM type as shown below:

Note:

- Valid codes can be listed in the **AllCodeLists.xls** workbook on the **HngSupBOMType** sheet.

IJHgrBOMDefinition::BOMType	IJHgrBOMDefinition::BOMDefinition
2	
2	
2	

21. Repeat the above steps to create the pipe clamp part class. Go to the HrgTemplate sheet and rename it as RH_SPCLAMP.

CustomInterfaces / RH_THREADED_ROD / RH_SPCLAMP

22. Go to the class definition and edit the following rows:

PartClassType	StandardSupportComponentClass
SymbolDefinition:	HS_LRParts.LRParts_FIG212N
SymbolIcon	
PartSelectionRule	HgrPipePartSelRule.CPartByPipeSizeEqual
ClassType	1
UserClassName	RH Medium Pipe Clamp

OccClassname	RH Medium Pipe Clamp
HgrSymbolPort(1):Type	HgrSymbolPort
HgrSymbolPort(2):Type	HgrSymbolPort

Definition	PartClassType	SymbolDefinition	SymbolIcon	PartSelectionRule
a	StandardSupportComponentClass	HS_LRParts.LRParts_FIG212N		HgrPipePartSelRule.CPartByPipeSizeEqual

ClassType	UserClassName	OccClassName	HgrSymbolPort(1):Type	HgrSymbolPort(2):Type
1	RH Medium Pipe Clamp	RH Medium Pipe Clamp	HgrSymbolPort	HgrSymbolPort

23. Now, you will need to add the pipe clamp attributes in the part definition row identified by the HEAD keyword.

    IJUAHgrPipe_Dia::Pipe_Dia
    IJUAHgrRHSPCLAMP::B
    IJUAHgrRHSPCLAMP::C
    IJUAHgrRHSPCLAMP::D
    IJUAHgrRHSPCLAMP::E
    IJUAHgrRHSPCLAMP::F
    IJUAHgrRHSPCLAMP::H
    IJUAHgrRHSPCLAMP::G1
    IJUAHgrRHSPCLAMP::G2
    IJUAHgrRHSPCLAMP::TAKE_OUT
    IJOAHgrFinish::FINISH
    IJUAHgrRHSPCLAMP::WEIGHT
    IJUAHgrRHSPCLAMP::PIPE_OD

Head	PartNumber	PartDescription	SymbolDefinition	MirrorBehaviorOption	NDFrom	NDTo	NDUnitType	HgrSymbolPort(1):Name	HgrSymbolPort(1):Category	HgrSymbolPort(1):MatingType	HgrSymbolPort(1):ConnectionInfo	HgrSymbolPort(1):PortType	HgrSymbolPort(1):Size	HgrSymbolPort(1):MinSize	HgrSymbolPort(1):MaxSize	HgrSymbolPort(1):UnitType	HgrSymbolPort(2):Name	HgrSymbolPort(2):Category	HgrSymbolPort(2):MatingType	HgrSymbolPort(2):ConnectionInfo	HgrSymbolPort(2):PortType	HgrSymbolPort(2):Size	HgrSymbolPort(2):MinSize	HgrSymbolPort(2):MaxSize	HgrSymbolPort(2):UnitType
DryWeight																									
WaterWeight																									
DryCogX																									
DryCogY																									
DryCogZ																									
IJHgrBOMDefinition::BOMType																									
IJHgrBOMDefinition::BOMDefinition																									
IJUAHgrPipe_Dia::Pipe_Dia																									
IJUAHgrRHSPCLAMP::B																									
IJUAHgrRHSPCLAMP::C																									
IJUAHgrRHSPCLAMP::D																									
IJUAHgrRHSPCLAMP::E																									
IJUAHgrRHSPCLAMP::F																									
IJUAHgrRHSPCLAMP::H																									
IJUAHgrRHSPCLAMP::G1																									
IJUAHgrRHSPCLAMP::G2																									
IJUAHgrRHSPCLAMP::TAKE_OUT																									
IJOAHgrFinish::FINISH																									
IJUAHgrRHSPCLAMP::WEIGHT																									
IJUAHgrRHSPCLAMP::PIPE_OD																									

24. Now, you will add three parts for the pipe clamp class as shown below. Key-in the appropriate part number, part description, mirror behavior option (codelist) and valid pipe size ranges.

Head	PartNumber	PartDescription	SymbolDefinition	MirrorBehaviorOption	NDFrom	NDTo	NDUnitType
Start							
a	RH-CLAMP_2	2 in Medium Pipe Clamp		15	2	2	in
a	RH-CLAMP_4	4 in Medium Pipe Clamp		15	4	4	in
a	RH-CLAMP_6	6 in Medium Pipe Clamp		15	6	6	in
End							

25. Key-in the appropriate pipe outer diameter, clamp dimensions and clamp finish.

Note:

- *Clamp Finish: Valid codes can be listed in the **HS_Anvil_Codelist.xls** workbook on the **Anvil_Finish** sheet.*

IJUAHgrPipe_Dia::Pipe_Dia	IJUAHgrRHSPCLAMP::B	IJUAHgrRHSPCLAMP::C	IJUAHgrRHSPCLAMP::D	IJUAHgrRHSPCLAMP::E	IJUAHgrRHSPCLAMP::F	IJUAHgrRHSPCLAMP::H	IJUAHgrRHSPCLAMP::G1	IJUAHgrRHSPCLAMP::G2	IJUAHgrRHSPCLAMP::TAKE_OUT	IJOAHgrFinish::FINISH	IJUAHgrRHSPCLAMP::WEIGHT	IJUAHgrRHSPCLAMP::PIPE_OD
2.375	2.12	0.5	2.75	2.12	0.5	2.75	0.25	1	1.87	2	1	2.375
4.5	3.44	0.75	4.38	3.62	0.625	4.19	0.25	1.3	3.3075	2	3	4.5
6.625	4.88	0.875	5.88	5	0.75	5.75	0.375	1.5	4.625	2	6	6.625

26. Key-in the appropriate port data as shown below:

HgrSymbolPort(1):Name	HgrSymbolPort(1):Category	HgrSymbolPort(1):MatingType	HgrSymbolPort(1):ConnectionInfo	HgrSymbolPort(1):PortType	HgrSymbolPort(1):Size	HgrSymbolPort(1):MinSize	HgrSymbolPort(1):MaxSize	HgrSymbolPort(1):UnitType	HgrSymbolPort(2):Name	HgrSymbolPort(2):Category	HgrSymbolPort(2):MatingType	HgrSymbolPort(2):ConnectionInfo	HgrSymbolPort(2):PortType	HgrSymbolPort(2):Size	HgrSymbolPort(2):MinSize	HgrSymbolPort(2):MaxSize	HgrSymbolPort(2):UnitType
Route				1016	2.375	2.375	2.375	in	Pin				1005	0.375	0	1.000	in
Route				1016	4.5	4.5	4.5	in	Pin				1005	0.5	0	1.000	in
Route				1016	6.625	6.625	6.625	in	Pin				1005	0.625	0	1.000	in

Note:

- *Port Type: Valid codes are listed in the AllCodeLists.xls workbook on the HngSupPortType sheet in the Codelist Number column.*

HngSupPortType ShortDescription	HngSupPortType LongDescription	Codelist Number	Sort Order
External Thread, RH	External Thread, RH	1000	
Internal Thread, RH	Internal Thread, RH	1001	
External Thread, LH	External Thread, LH	1002	
Internal Thread, LH	Internal Thread, LH	1003	
Rod Hole	Rod Hole	1004	
Pin/Bolt (for eye)	Pin/Bolt (for eye)	1005	
Pin/Bolt (for double lug)	in/Bolt (for double lug)	1006	
Eye	Eye	1007	
Single Lug	Single Lug	1008	
Double Lug	Double Lug	1009	
Clevis	Clevis	1010	
Steel	Steel	1011	
Weld	Weld	1012	
Bolted Connection	Bolted Connection	1013	
Beam Clamp	Beam Clamp	1014	
Pipe	Pipe	1015	
Pipe Clamp	Pipe Clamp	1016	
U-Bolt	U-Bolt	1017	
Roller	Roller	1018	

27. Key-in the appropriate clamp weight, pipe outer diameter and BOM type as shown below:

Note:

- *Valid codes can be listed in the **AllCodeLists.xls** workbook on the **HngSupBOMType** sheet.*

DryWeight	WaterWeight	DryCogX	DryCogY	DryCogZ	IJHgrBOMDefinition::BOMType	IJHgrBOMDefinition::BOMDefinition
1	0	0	0	0	3	
3	0	0	0	0	3	
6	0	0	0	0	3	

28. Next step, you will add two new custom interfaces for the rod and clamp part classes. Use the Custom Interfaces sheet to define the attributes names with associated data type and unit type.

29. Create a new interface called IJUAHgrRHThreadedRod. Go to the Custom Interface sheet and add the following rows:

InterfaceName	CategoryName	AttributeName	AttributeUserName
IJUAHgrRHThreadedRod	Dimensions	D_USER	D: Thread length if other than standard
		ROD_DIA	Rod Diameter
		WEIGHT_PER_LENGTH	Pounds per Foot

Type	UnitsType	PrimaryUnits	OnPropertyPage	ReadOnly	SymbolParameter
Double	distance	in	1	1	D_USER
Double	distance	in	1	1	ROD_DIA
Double	mass per length	lbm/ft	1	1	WEIGHT_PER_LENGTH

Notes:

- *The Custom Interfaces sheet allows you to load a custom schema into the metadata of a catalog. This sheet defines the customized user interfaces and attributes (properties) for the smart/part classes in the workbook.*

- *An interface is a collection of attributes. A complete interface definition includes one or more attribute names with associated data type, unit type, and optional symbol parameter.*

- *The appropriate unit type and the primary unit values are found using the AllCommon.xls workbook. The Category Names are listed in the Property Categories codelist. The codelist namespace for user-defined codelist is UDP.*

30. Create the second new interface called IJUAHgrRHSPCLAMP and add the following entries:

InterfaceName	CategoryName	AttributeNan	AttributeUse	Type	UnitsType	PrimaryUnits	OnPropertyPage	ReadOnly	SymbolParameter
IJUAHgrRHSPCLAMP	Dimensions	B	B	Double	distance	in	1	1	B
		C	C	Double	distance	in	1	1	C
		D	D	Double	distance	in	1	1	D
		E	E	Double	distance	in	1	1	E
		F	F	Double	distance	in	1	1	F
		H	H	Double	distance	in	1	1	H
		G1	G1	Double	distance	in	1	1	G1
		G2	G2	Double	distance	in	1	1	G2
		TAKE_OUT	TAKE_OUT	Double	distance	in	1	1	TAKE_OUT
		WEIGHT	WEIGHT	Double	force	lbf	1	1	WEIGHT
		PIPE_OD	PIPE_OD	Double	distance	in	1	1	PipeDiam

31. Save the excel workbook as SPSupportTraining.xls workbook.

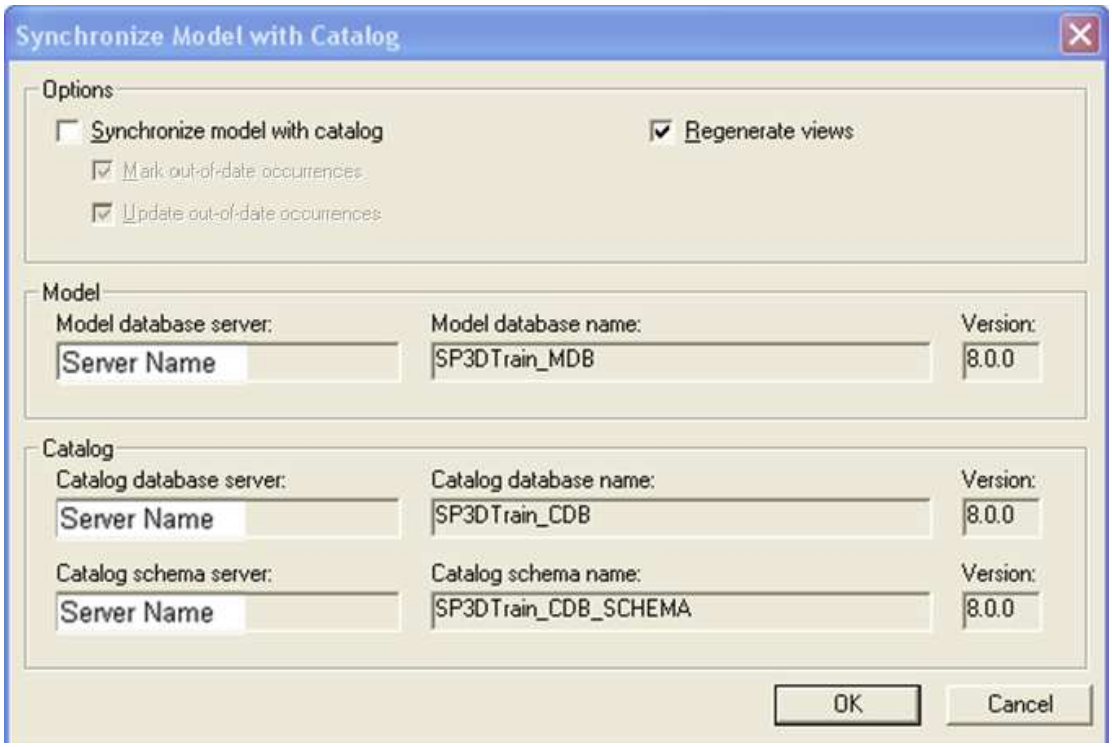
32. Load the information into the catalog using the Add/Modify/Delete Mode. Once the bulkload process is completed, review the log file.

33. Run the Project Management Task. Select the Model in the hierarchy.

34. Select Tools -> Synchronize Model with the Catalog.

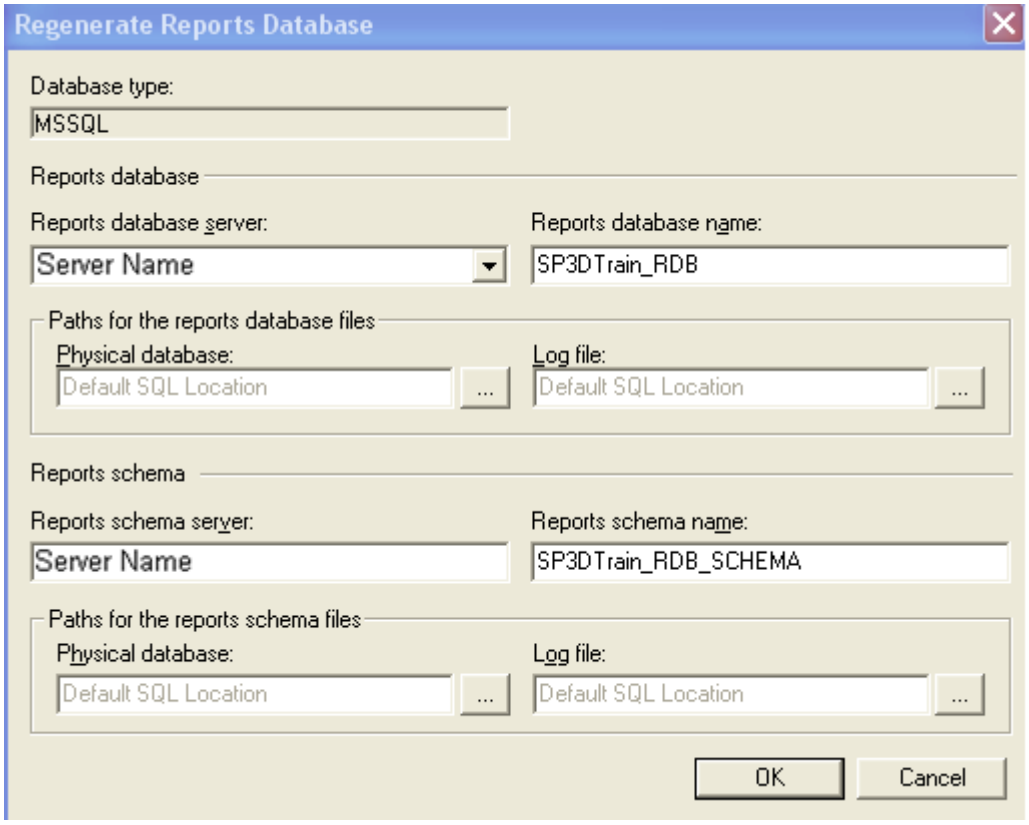
35. Uncheck the Synchronize Model with the Catalog option.

    ***Note**: You just need to update the views in the model.*



36. Hit “OK” Button.

37. Once the process is completed, Right click the training plant icon and select “Regenerate the Reports database” option to re-create the views in the report database.



38. Hit OK to start the process.

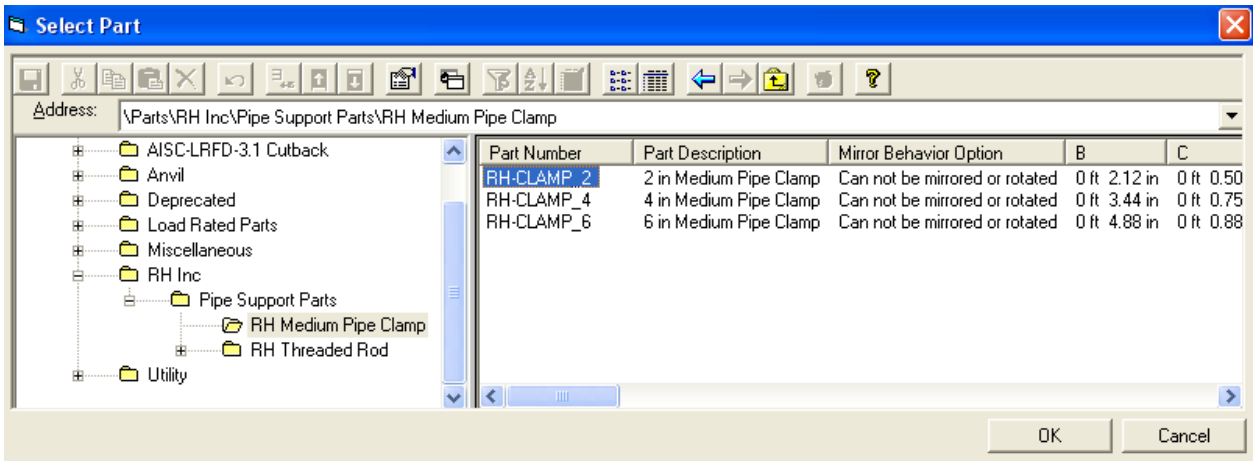
39. Once the process is completed, go to the Hangers and Supports Task.

40. Place a Design Support on a pipe.

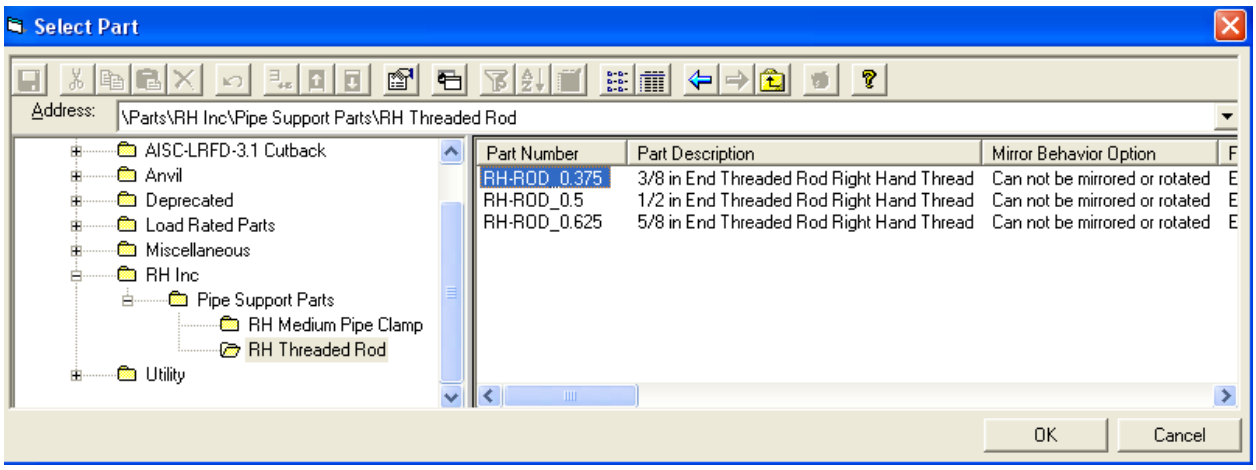
41. Select the Place Part command.

42. The Select Part dialog appears. Expand the hierarchy until you see the new parts.

43. Place the pipe clamp in the model.



44. Repeat the above steps to place the new rod in the model.



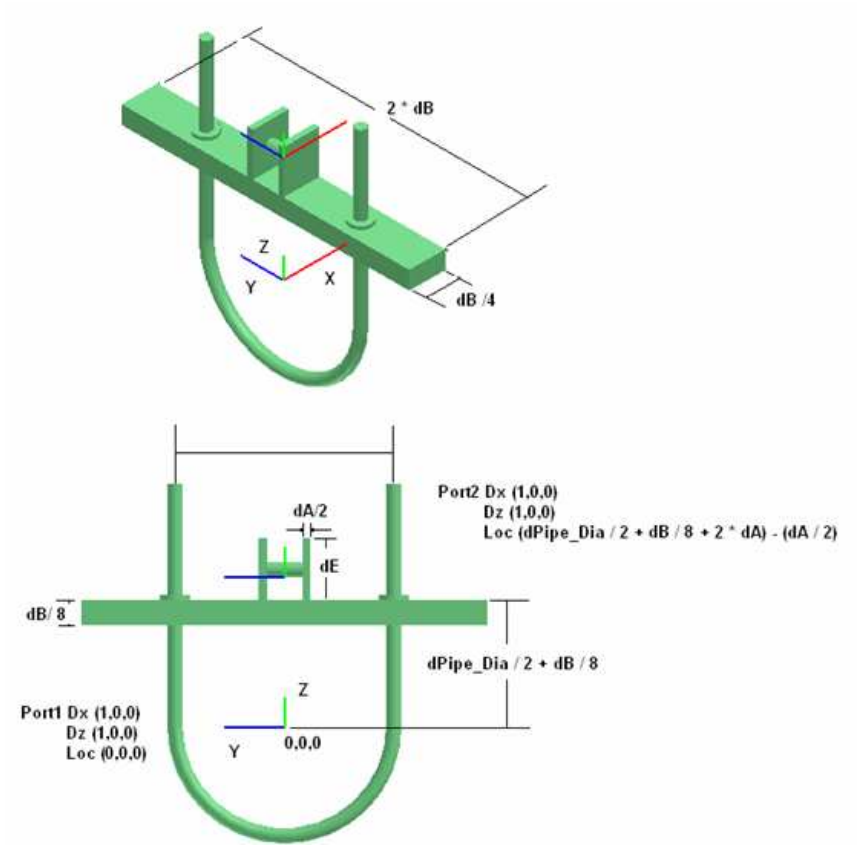
Lab 2: Standard Support Component Symbol (SPUBolt)

Objectives

After completing this lab, you will be able to:

- Create a simple standard support component symbol
- Use the symbol service to create the symbol definition
- Use the AddPort method to create the ports
- Use Helper Functions to create simple geometric shapes

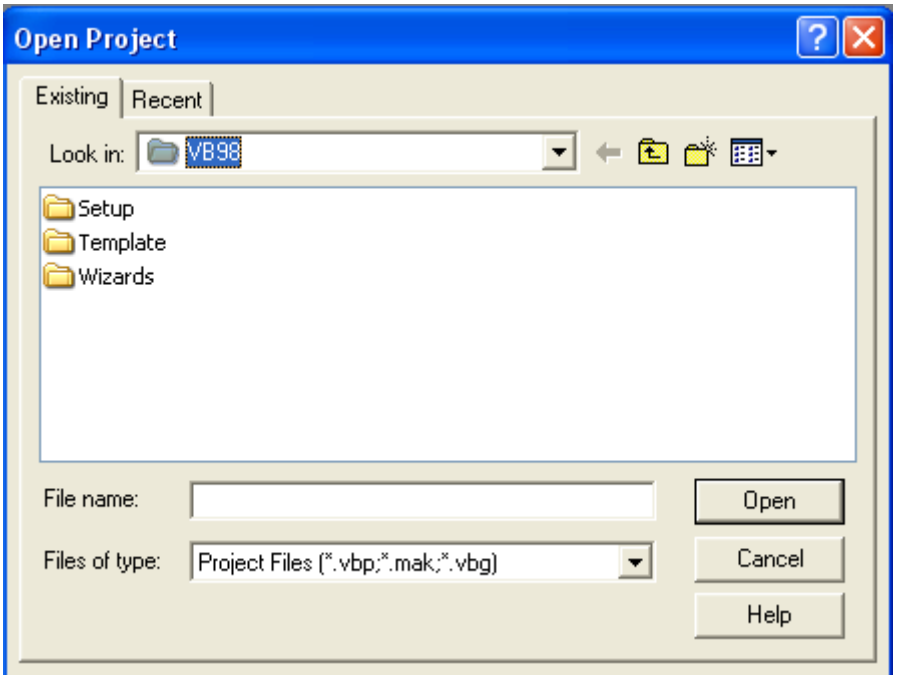
In this lab, you will create a standard support component symbol as shown below. You can use the HgrTemplatePart template provided by the instructor to create the symbol definition program. This symbol consists of nine simple geometric shapes (five cylinders, one U shape and three rectangular boxes) to define the symbol's output. Four inputs named "A", "B", "D" and "E" are required to construct this symbol.



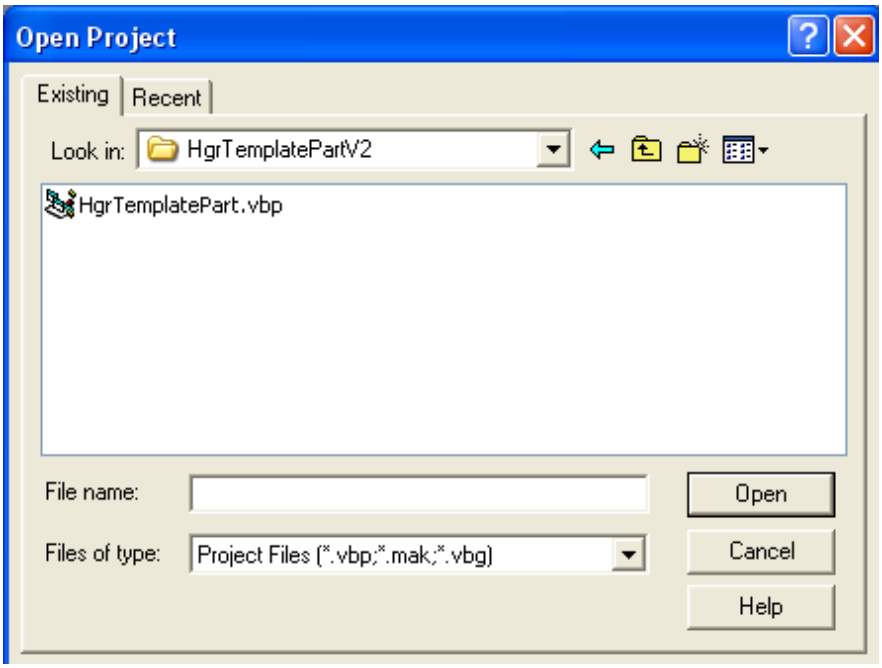
1. Create a directory called UBoltSet under c:\train\HgrTrainingFiles and run Microsoft Visual Basic 6.0

2. Close the Microsoft New Project dialog box.

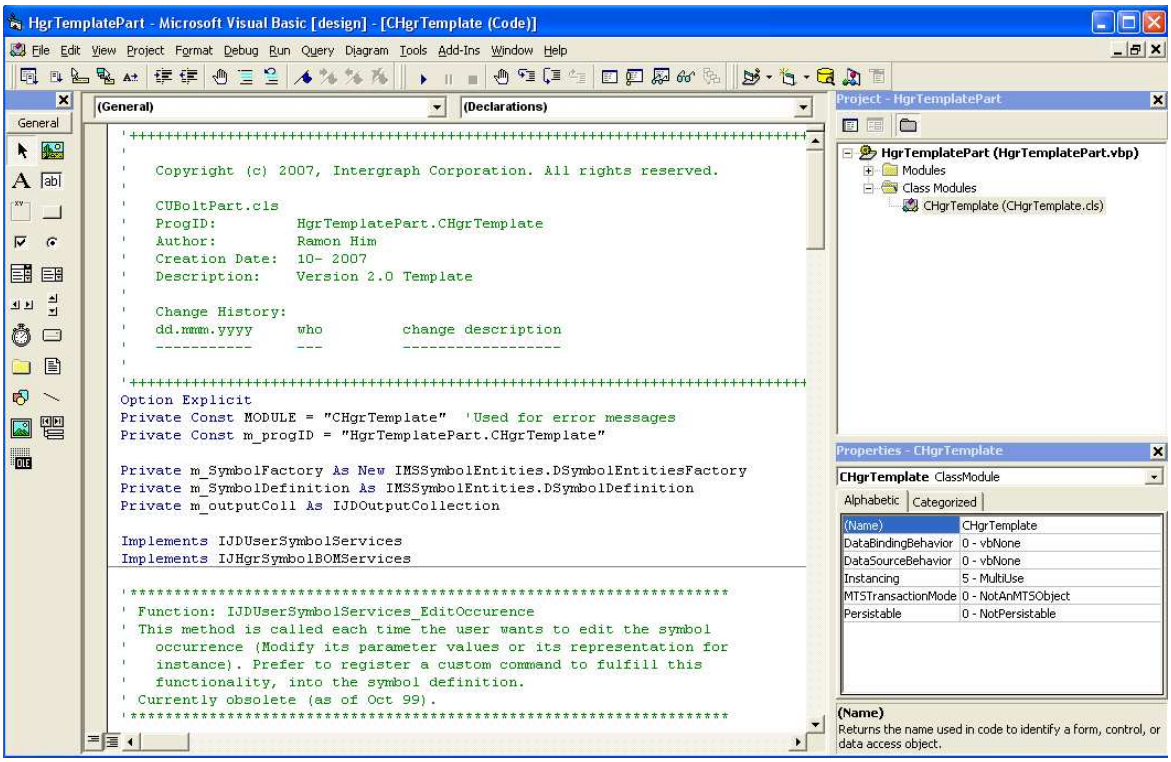
3. Select *File -> Open Project* option to open the Open Project Dialog box.



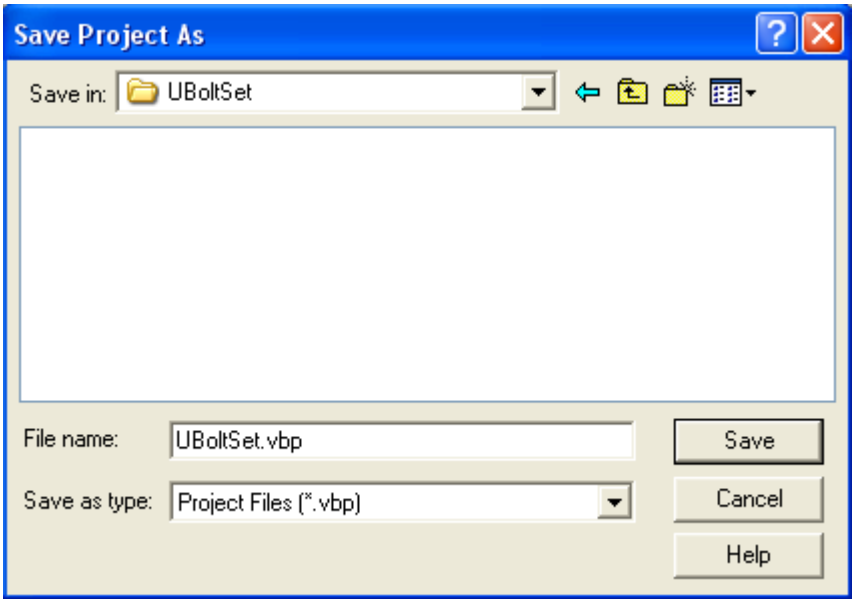
4. Navigate to C:\train\HgrTrainingFiles\HgrTemplatePartV2 and open the HgrTemplatePart Visual Basic project.



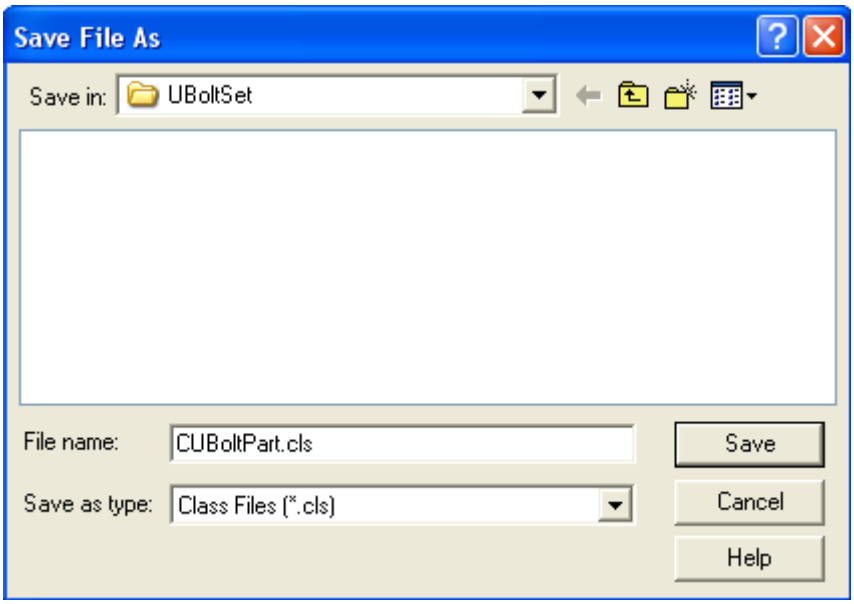
5. Setup the Visual Basic Development Environment as shown below:



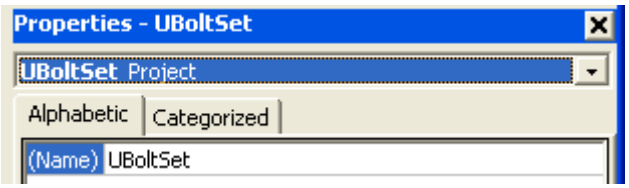
6. Go to the Explorer Window and select the Project file node. Select *File* -> *"Save Project As"* option to save the project as UBoltSet.vbp under the UBoltSet directory.

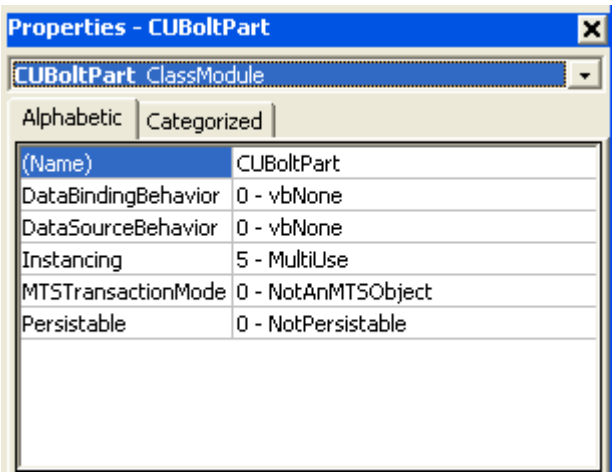


7. Go to the Explorer Window and select the CHgrTemplate class file node. Select *File -> "Save CHgrTemplate.cls As"* option to save the class module as CUBoltPart.cls under UBoltSet directory.



8. Go to the Properties Window and change the name of the Project and Class Modules, as shown here:





9. Go to the General Declarations section in CUBoltPart module and change the value of the *Constant Module variable* from *"CHgrTemplatePart:"* to *"CUBoltPart:"*

```
Private Const MODULE = "CUBoltPart"  'Used for error messages
```

10. Rename the ProgId (project name and class name) as shown below:

```
Private Const m_progID = "UBoltSet.CUBoltPart"
```

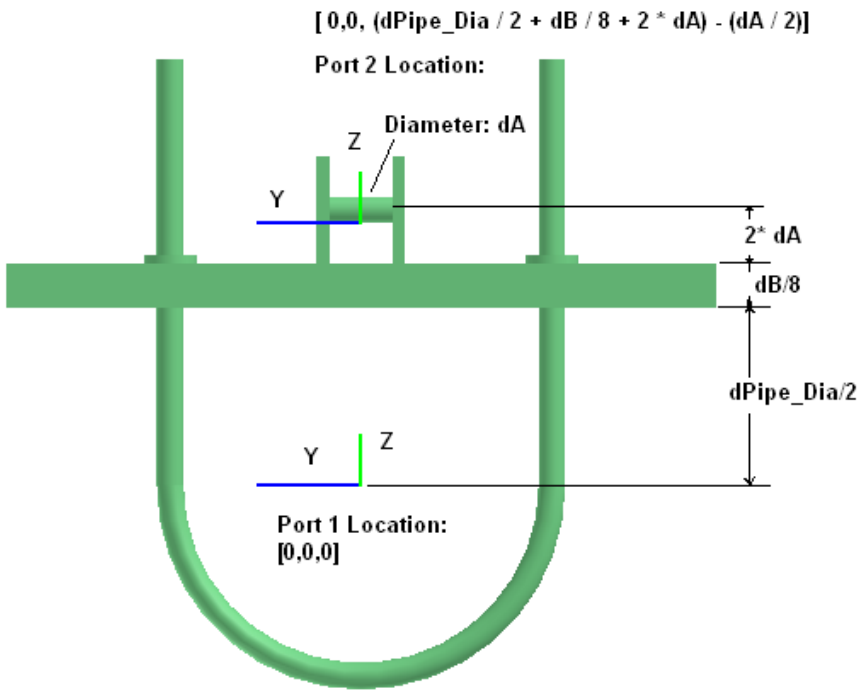
11. Go to IJDUserSymbolServices_InitializeSymbolDefinition method. Review and edit the input and output definitions as shown below:

```
'Set total number of inputs

    Const iNumInputs As Integer = 6
    Dim sInputName(iNumInputs) As String
    Dim sDataType(iNumInputs) As String

'Set total number of outputs

    Const iNumOutputs As Integer = 11
    Dim sOutputName(iNumOutputs) As String

' List the number of outputs for the symbol

    sOutputName(1) = "Port1"
    sOutputName(2) = "Port2"
    sOutputName(3) = "BEND"
    sOutputName(4) = "R"
    sOutputName(5) = "L"
    sOutputName(6) = "BASE"
    sOutputName(7) = "NUT1"
    sOutputName(8) = "NUT2"
    sOutputName(9) = "P1"
    sOutputName(10) = "P2"
    sOutputName(11) = "P3"
```

```
' List the number of inputs for the symbol

    sInputName(2) = "PIPE_DIA"
    sDataType(2) = "Double"
    sInputName(3) = "A"
    sDataType(3) = "Double"
    sInputName(4) = "B"
    sDataType(4) = "Double"
    sInputName(5) = "D"
    sDataType(5) = "Double"
    sInputName(6) = "E"
    sDataType(6) = "Double"

    SetupInputsAndOutputsForCache pSymbolDefinition, _
      iNumInputs, iNumOutputs, sInputName(), sDataType(), sOutputName(), m_progID
```

12. Go to Physical() procedure and add the following instructions to create the outputs:

```
    'Add code here

        Dim dPipe_Dia As Double
        Dim dA As Double
        Dim dB As Double
        Dim dD As Double
        Dim dE As Double

        dPipe_Dia = arrayOfInputs(2)
        dA = arrayOfInputs(3)
        dB = arrayOfInputs(4)
        dD = arrayOfInputs(5)
        dE = arrayOfInputs(6)

    'Create two ports

        AddPort "Route", 0, 0, 0, _
                1, 0, 0, _
                0, 0, 1, m_outputColl, "Port1", arrayOfInputs(1), "Port1"

        AddPort "Pin", 0, 0, (dPipe_Dia / 2 + dB / 8 + 2 * dA) - (dA / 2), _
                1, 0, 0, _
                0, 0, 1, m_outputColl, "Port2", arrayOfInputs(1), "Port2"
```



'U-Bolt shape

*AddElbow dA / 2, dB / 2 + dA / 2, 180, "ROTX(180) * " + Loc(0, 0, 0) + " * ROTZ(270)", m_outputColl, "BEND", "BEND"*

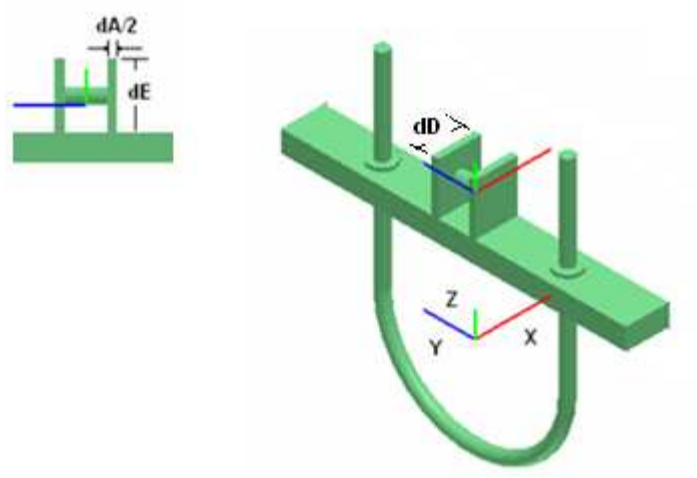
*AddCylinder 1.2 * dB, dA / 2, Loc(dB / 2 + dA / 2, 0, 0) + " * ROTZ(270)", m_outputColl, "R", "R"*
*AddCylinder 1.2 * dB, dA / 2, Loc(-dB / 2 - dA / 2, 0, 0) + " * ROTZ(270)", m_outputColl, "L", "L"*

*AddBox 2 * dB, dB / 4, dB / 8,Loc(-dB, -dB / 8, dPipe_Dia / 2) + " * ROTZ(270)", m_outputColl, "BASE", "BASE"*

'Nuts

*AddCylinder 0.6 * dA / 2, dA, _*
*Loc(dB / 2 + dA / 2, 0, dPipe_Dia / 2 + dB / 8) + " * ROTZ(270)", m_outputColl, "NUT1", "NUT1"*

*AddCylinder 0.6 * dA / 2, _*
*dA, Loc(-dB / 2 - dA / 2, 0, dPipe_Dia / 2 + dB / 8) + " * ROTZ(270)", m_outputColl, "NUT2", "NUT2"*



'Eye

AddBox dD, dA / 2, dE, _
*Loc(-dD / 2, -dD / 2, dPipe_Dia / 2 + dD / 2) + " * ROTZ(0)", m_outputColl, "P1", "P1"*

AddBox dD, dA / 2, dE, _
*Loc(-dD / 2, dD / 2 - dA / 2, dPipe_Dia / 2 + dD / 2) + " * ROTZ(0)", m_outputColl, "P2", "P2"*

AddCylinder dD, dA / 2, _
*Loc(0, dPipe_Dia / 2 + dD / 2 + 2 * dA, -dD / 2) + " * ROTX(90)", m_outputColl, "P3", "P3"*

13. Compile the Visual Basic project and save the dll as UBoltSet.dll in the c:\train\UBoltSet

14. One of the most important steps in Visual Basic programming is to preserve the binary compatibility of your program. Save the final version of your dll file to be binary compatibility in order to preserve the CLSID.

15. Save the UBoltSet project.

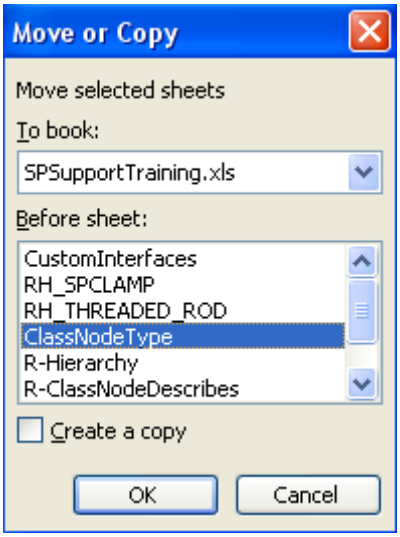
16. Open the SPSupportTraining.xls workbook located in *c:\train* HS_RH_Parts

17. Go the R-ClassNodeDescribes sheet and add the following row:

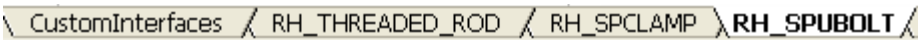
Head	*RelationSource*	*RelationDestination*
Start		
!	**Support Part Classes**	
	CLASSNODE_PART_SPCOMPANY_PARTS	RH_SPCLAMP
	CLASSNODE_PART_SPCOMPANY_PARTS	RH_THREADED_ROD
a	CLASSNODE_PART_SPCOMPANY_PARTS	RH_SPUBOLT
End		

18. Next step, you will create a new part class by copying the RH_SPCLAMP sheet:

19. Select the RH_SPCLAMP Sheet. Right Click to open the Move or Copy dialog box. Enable the copy option. Select the OK button to copy the sheet next to the ClassNodeType sheet.



20. Rename the sheet as RH_SPUBOLT.



21. Go to the class definition and edit the following row:

PartClassType	StandardSupportComponentClass
SymbolDefinition:	UBoltSet.CUBoltPart
SymbolIcon	
PartSelectionRule	HgrPipePartSelRule.CPartByPipeSizeEqual
ClassType	1
UserClassName	RH Special U-BOLT
OccClassname	RH Special U-BOLT
HgrSymbolPort(1):Type	HgrSymbolPort
HgrSymbolPort(2):Type	HgrSymbolPort

Definition	PartClassType	SymbolDefinition	SymbolIcon	PartSelectionRule
a	StandardSupportComponentClass	UBoltSet.CUBoltPart		HgrPipePartSelRule.CPartByPipeSizeEqual

ClassType	UserClassName	OccClassName	HgrSymbolPort(1):Type	HgrSymbolPort(2):Type
1	RH Special U-BOLT	RH Special U-BOLT	HgrSymbolPort	HgrSymbolPort

22. Now, you will need to add U-Bolt attributes in the part definition row identified by the keyword HEAD.

IJUAHgrPipe_Dia::Pipe_Dia
IJUAHgrSPUBolt::A
IJUAHgrSPUBolt::B
IJUAHgrSPUBolt::D
IJUAHgrSPUBolt::E
IJUAHgrSPUBolt::PIPE_OD
IJOAHgrFinish::FINISH

Head	PartNumber	PartDescription	SymbolDefinition	MirrorBehaviorOption	NDFrom	NDTo	NDUnitType	HgrSymbolPort(1):Name	HgrSymbolPort(1):Category	HgrSymbolPort(1):MatingType	HgrSymbolPort(1):ConnectionInfo	HgrSymbolPort(1):PortType	HgrSymbolPort(1):Size	HgrSymbolPort(1):MinSize	HgrSymbolPort(1):MaxSize	HgrSymbolPort(1):UnitType	HgrSymbolPort(2):Name	HgrSymbolPort(2):Category	HgrSymbolPort(2):MatingType	HgrSymbolPort(2):ConnectionInfo	HgrSymbolPort(2):PortType	HgrSymbolPort(2):Size	HgrSymbolPort(2):MinSize	HgrSymbolPort(2):MaxSize	HgrSymbolPort(2):UnitType

DryWeight	WaterWeight	DryCogX	DryCogY	DryCogZ	IJHgrBOMDefinition::BOMType	IJHgrBOMDefinition::BOMDefinition	IJUAHgrPipe_Dia::Pipe_Dia	IJUAHgrSPUBolt::A	IJUAHgrSPUBolt::C	IJUAHgrSPUBolt::D	IJUAHgrSPUBolt::E	IJUAHgrSPUBolt::PIPE_OD	IJOAHgrFinish::FINISH

23. Now, you will add three parts for the u-bolt class as shown below. Key-in the appropriate part number, part description, mirror behavior option (codelist) and valid pipe size ranges.

Head	PartNumber	PartDescription	SymbolDefinition	MirrorBehaviorOption	NDFrom	NDTo	NDUnitType
Start							
a	RH-U_2	2 in Special U Bolt		15	2	2	in
a	RH-U_4	4 in Special U Bolt		15	4	4	in
a	RH-U_6	6 in Special U Bolt		15	6	6	in
End							

24. Key-in the appropriate pipe outer diameter, U-Bolt dimensions and U-Bolt finish.

Note:

- Clamp Finish: Valid codes can be listed in the **HS_Anvil_Codelist.xls** workbook on the **Anvil_Finish** sheet.

IJUAHgrPipe_Dia::Pipe_Dia	IJUAHgrSPUBolt::A	IJUAHgrSPUBolt::B	IJUAHgrSPUBolt::D	IJUAHgrSPUBolt::E	IJUAHgrSPUBolt::PIPE_OD	IJOAHgrFinish::FINISH
2.375	0.3	2.375	0.5938	1.2	2.375	2
4.5	0.5	4.5	1.125	2	4.5	2
6.625	0.5	6.625	1.6563	2	6.625	2

25. Key-in the appropriate port data as shown here:

HgrSymbolPort(1):Name	HgrSymbolPort(1):Category	HgrSymbolPort(1):MatingType	HgrSymbolPort(1):ConnectionInfo	HgrSymbolPort(1):PortType	HgrSymbolPort(1):Size	HgrSymbolPort(1):MinSize	HgrSymbolPort(1):MaxSize	HgrSymbolPort(1):UnitType	HgrSymbolPort(2):Name	HgrSymbolPort(2):Category	HgrSymbolPort(2):MatingType	HgrSymbolPort(2):ConnectionInfo	HgrSymbolPort(2):PortType	HgrSymbolPort(2):Size	HgrSymbolPort(2):MinSize	HgrSymbolPort(2):MaxSize	HgrSymbolPort(2):UnitType
Route				1016	2.375	2.375	2.375	in	Pin				1005	0.375	0	1.000	in
Route				1016	4.5	4.5	4.5	in	Pin				1005	0.5	0	1.000	in
Route				1016	6.625	6.625	6.625	in	Pin				1005	0.625	0	1.000	in

26. Key-in the appropriate U-Bolt weight and BOM type as shown here:

Note

- Valid codes can be listed in the **AllCodeLists.xls** workbook on the **HngSupBOMType** sheet.

DryWeight	WaterWeight	DryCogX	DryCogY	DryCogZ	IJHgrBOMDefinition::BOMType	IJHgrBOMDefinition::BOMDefinition
1.5					2	
3.5					2	
6.5					2	

27. Next step, you will add a new custom interfaces for the U-Bolt part class. Use the Custom Interfaces sheet to define the attributes names with associated data type and unit type.

28. Create a new interface called IJUAHgrSPUBolt. Go to the Custom Interface sheet and add the following rows:

InterfaceName	CategoryName	AttributeName	AttributeUserName	Type	UnitsType	PrimaryUnits	OnPropertyPage	ReadOnly	SymbolParameter
IJUAHgrSPUBolt	Dimensions	E	E	Double	distance	in	1	1	E
		D	D	Double	distance	in	1	1	D
		B	B	Double	distance	in	1	1	B
		A	A	Double	distance	in	1	1	A
		PIPE_OD	PIPE_OD	Double	distance	in	1	1	PIPE_OD

29. Save and Exit the SPSupportTraining.xls workbook.

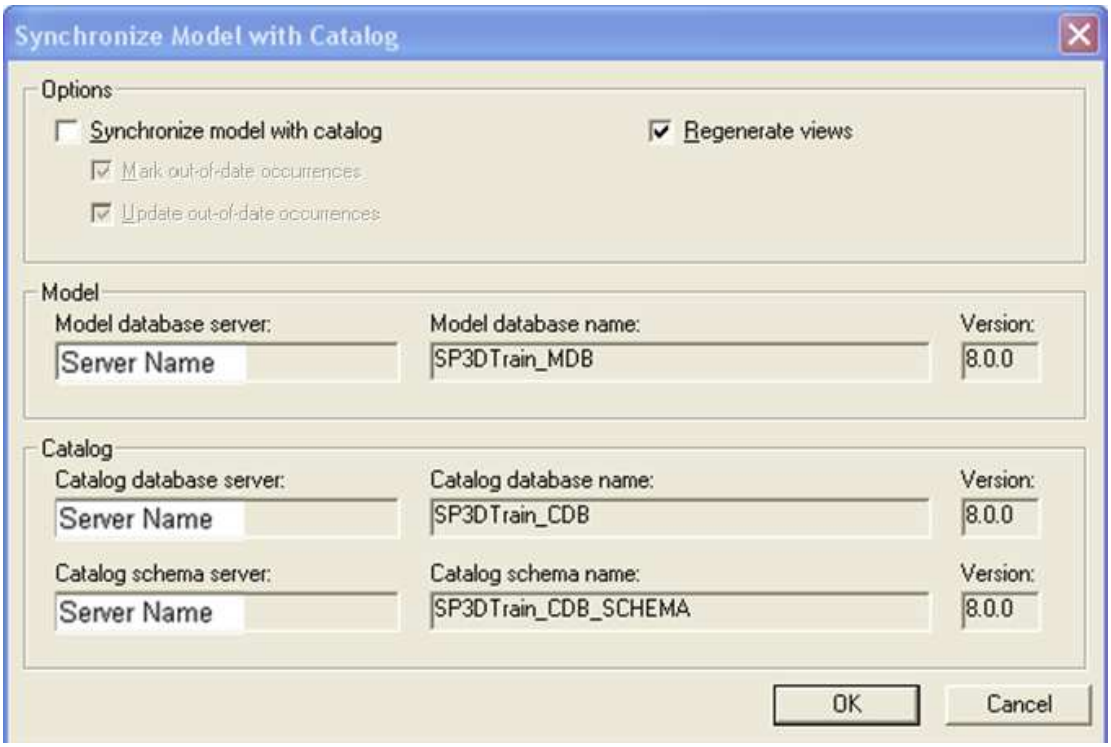
30. Load the information into the catalog using the A/M/D Mode. Once the bulkload process is completed, review the log file.

31. Run the Project Management Task. Select the Model in the hierarchy.

32. Select Tools -> Synchronize Model with the Catalog.

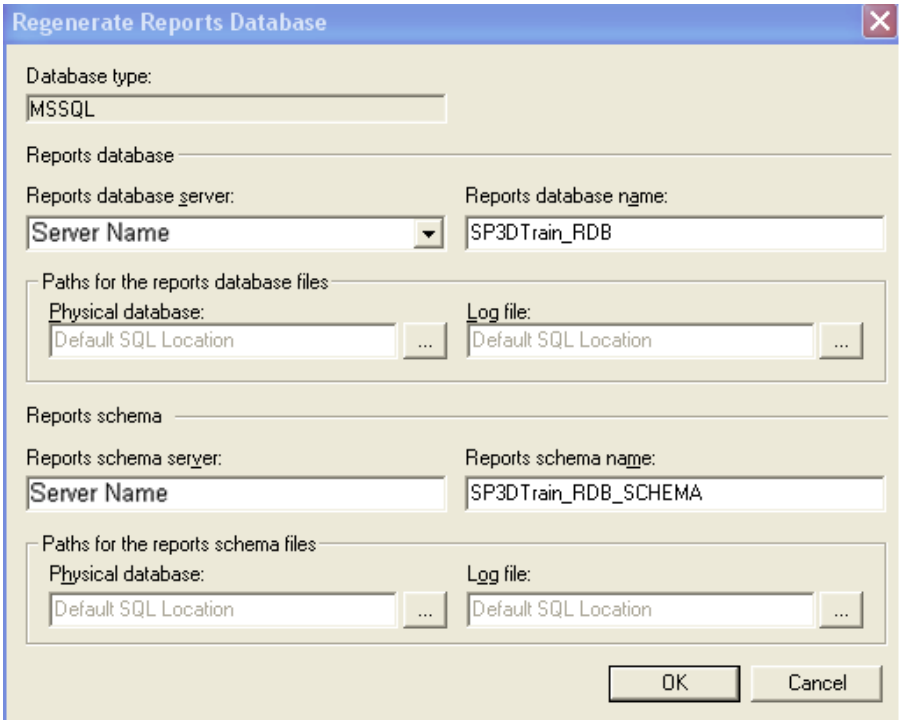
33. Uncheck the Synchronize Model with the Catalog option.

***Note**: You just need to update the views in the model.*

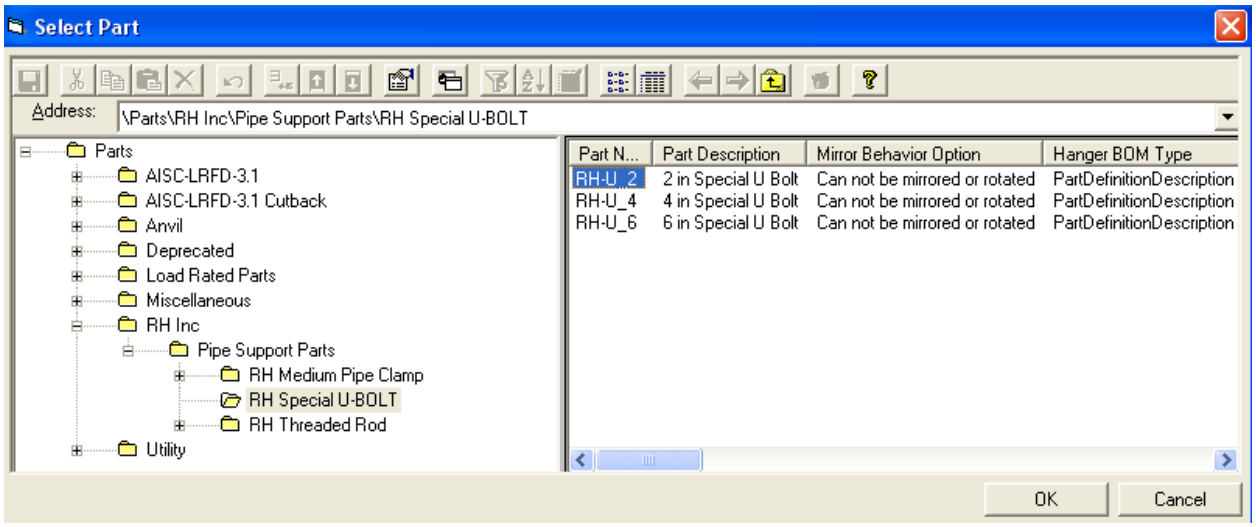


34. Hit “OK” Button.

35. Once the process is completed, Right mouse click the training plant icon and select “Regenerate the reports database” option to re-create the views in the report database.



36. Hit OK to start the process.
37. Once the process is completed, go to the Hangers and Supports Task and place the UBoltSet part using the Place Part Command.



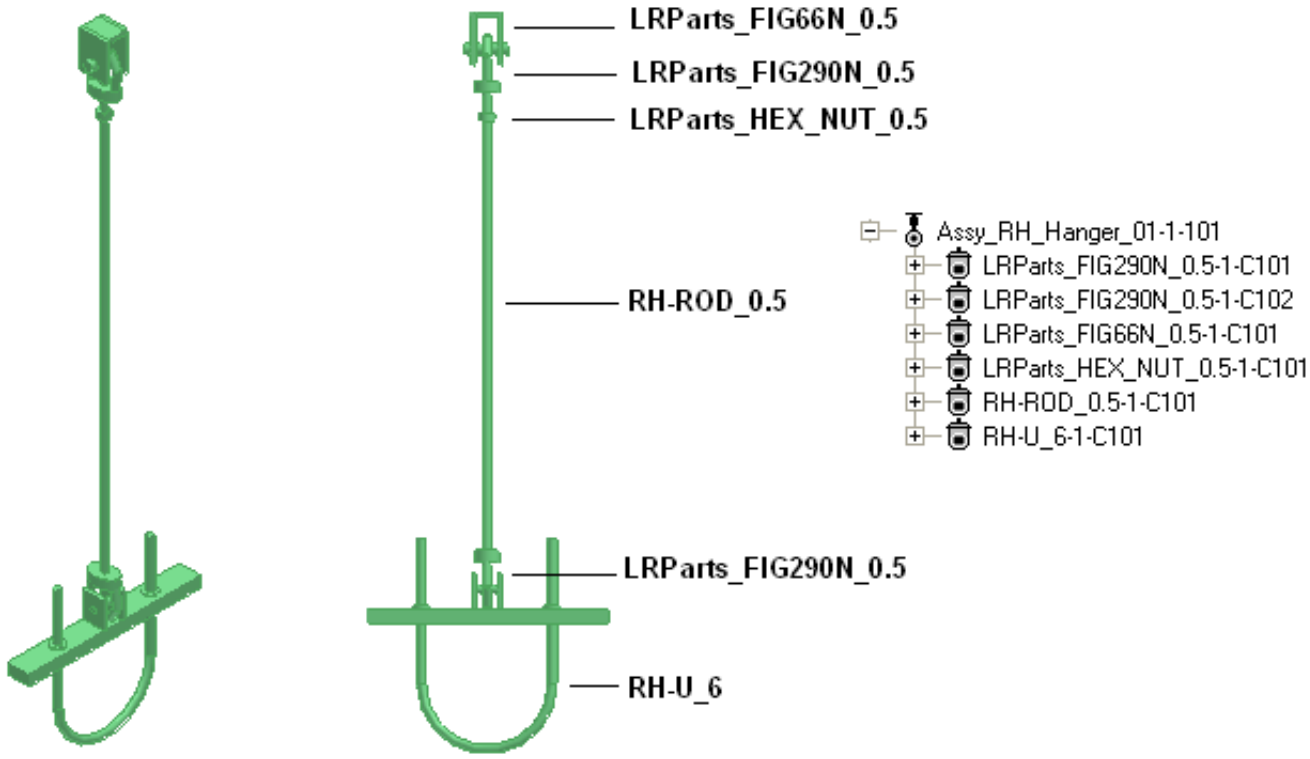
Lab 3a: Assembly Information Rule - Method of Take-Outs

Objectives

After completing this lab, you will be able to:

- Use the methods provided by IJHgrAssmInfo
- Define joint data for this assembly
- Use part selection rule
- Method of Take-Outs

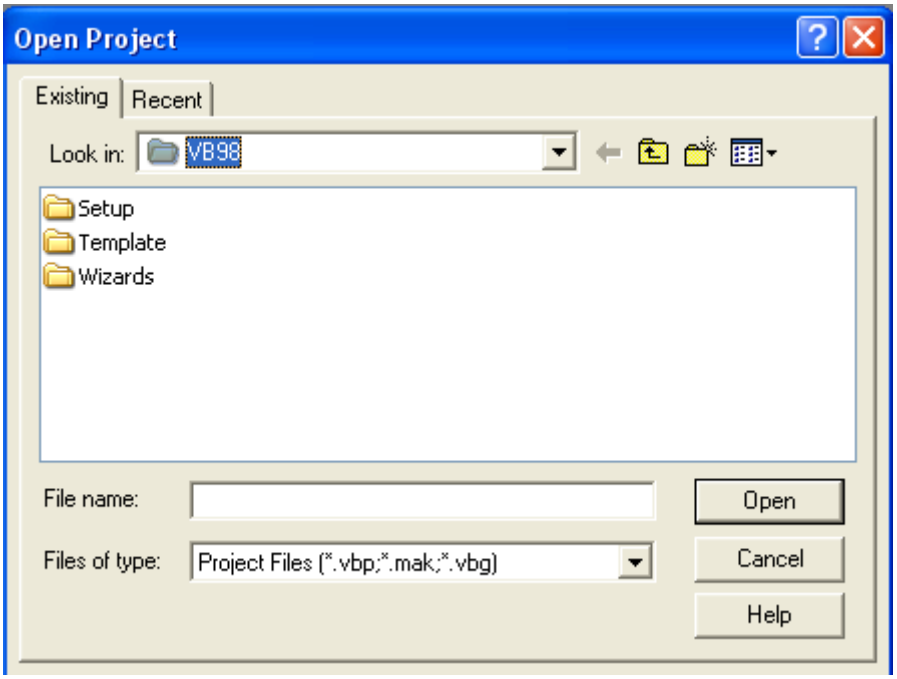
In this lab, you will create an assembly information rule to build a hanger using standard components created in previous labs as shown below. You start by using the Assy_TemplateAsm template provided by the instructor to create the rule.



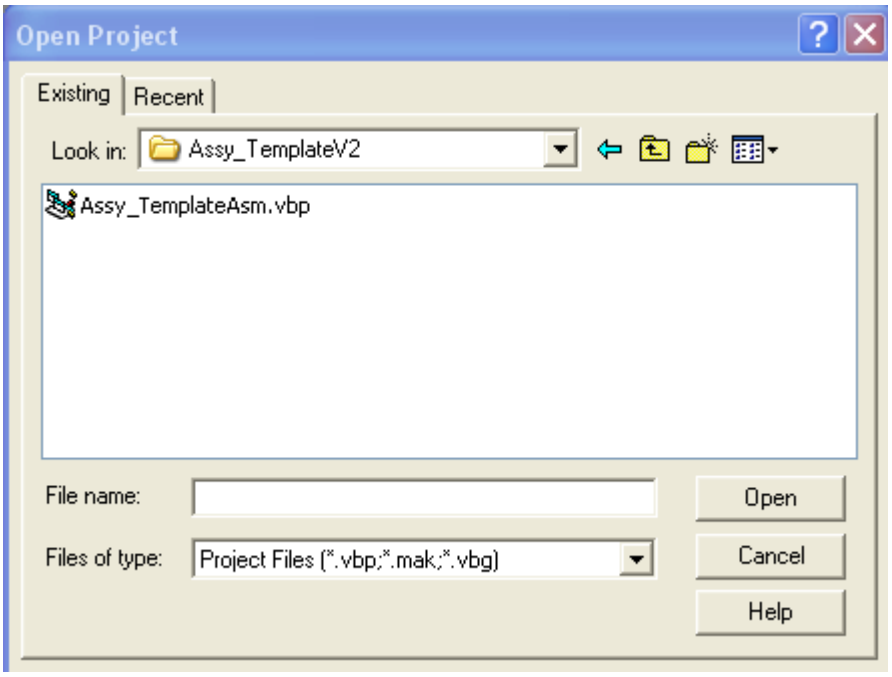
1. Create a directory called HS_RH_ASSY_Hanger1 under c:\train\HgrTrainingFiles and run Microsoft Visual Basic 6.0

2. Close the Microsoft New Project dialog box.

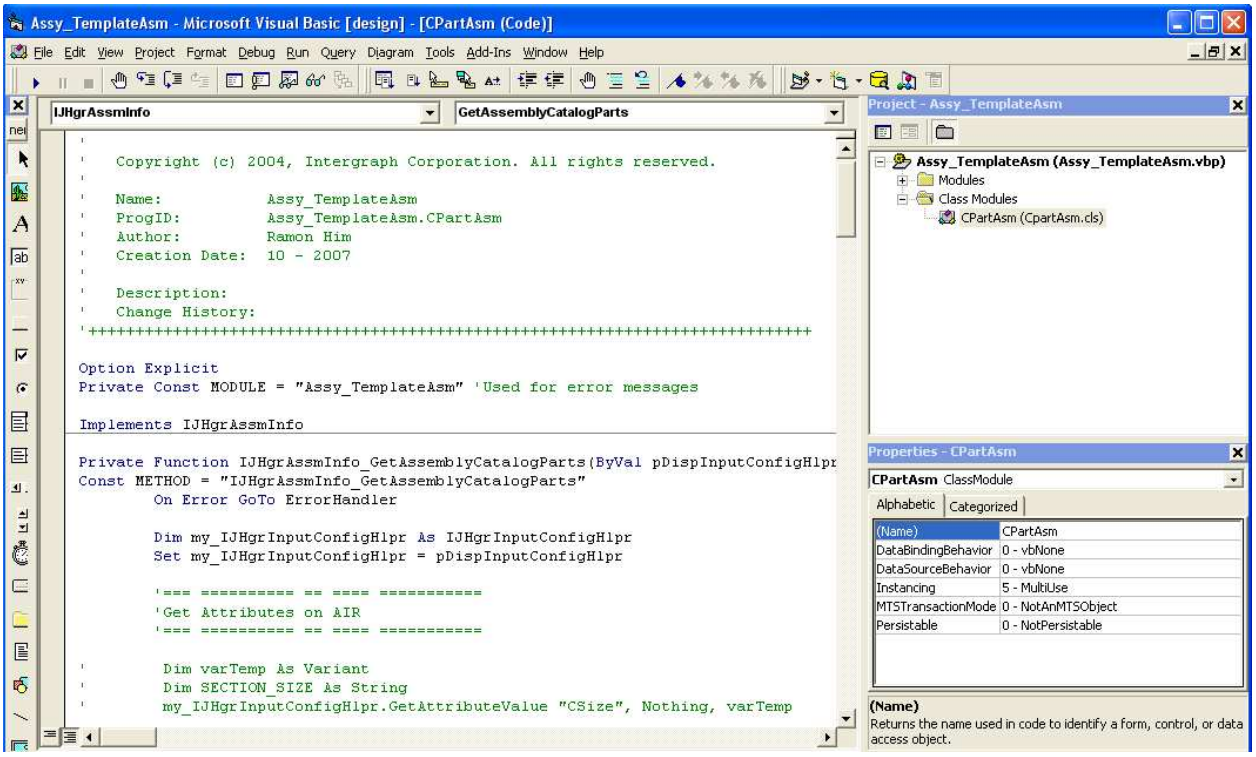
3. Select *File -> Open Project* option to open the Open Project Dialog box.



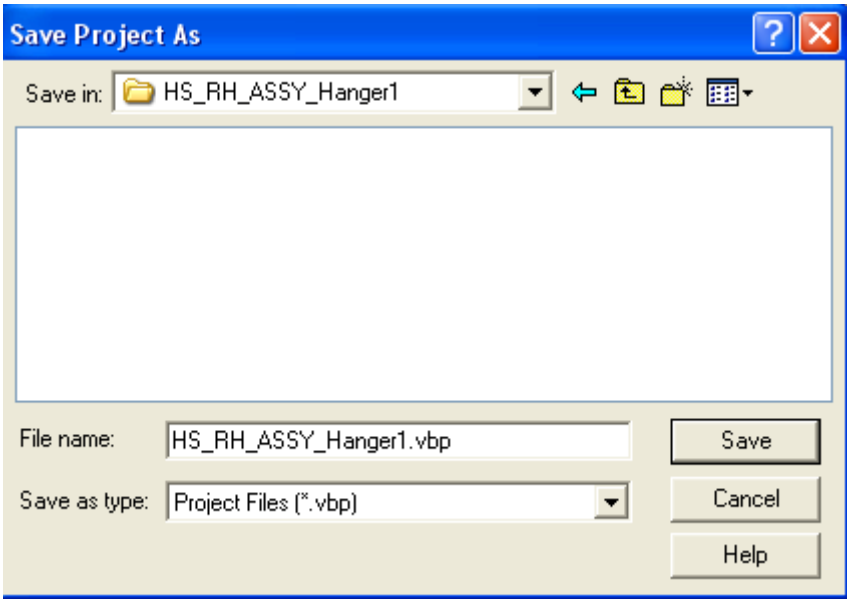
4. Navigate to c:\train\HgrTrainingFiles\ AssyTemplateV2 and open the Assy_TemplateAsm.vbp Template project.



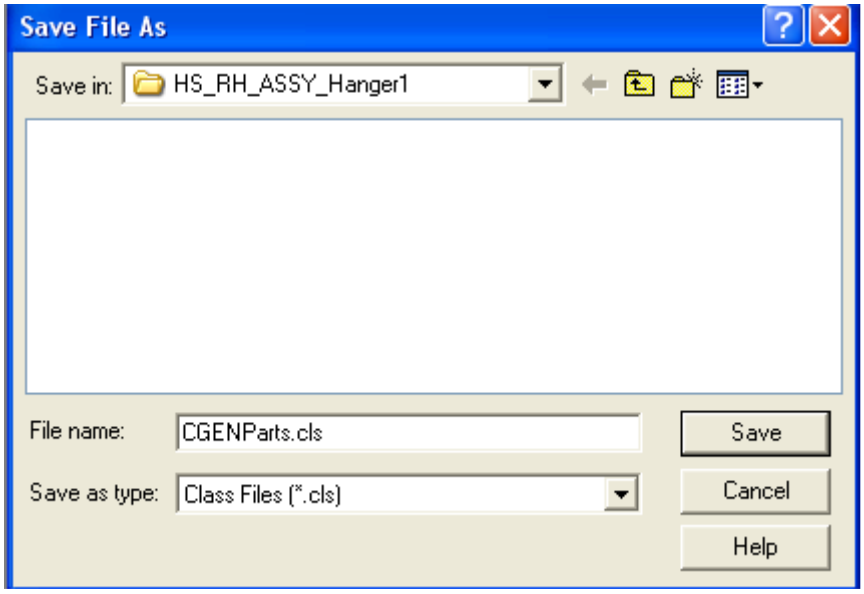
5. Setup the Visual Basic Development Environment as shown here:



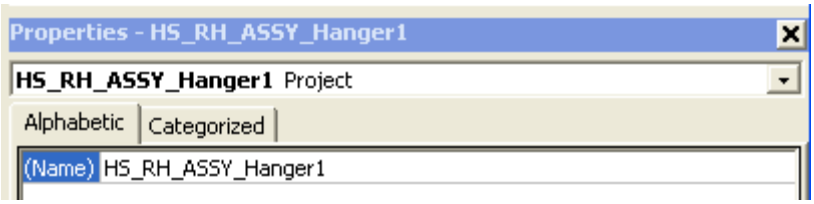
6. Go to the Explorer Window and select the Project file node. Select *File* -> *"Save Project As"* option to save the project as HS_RH_ASSY_Hanger1.vbp under the HS_RH_ASSY_Hanger1 directory.

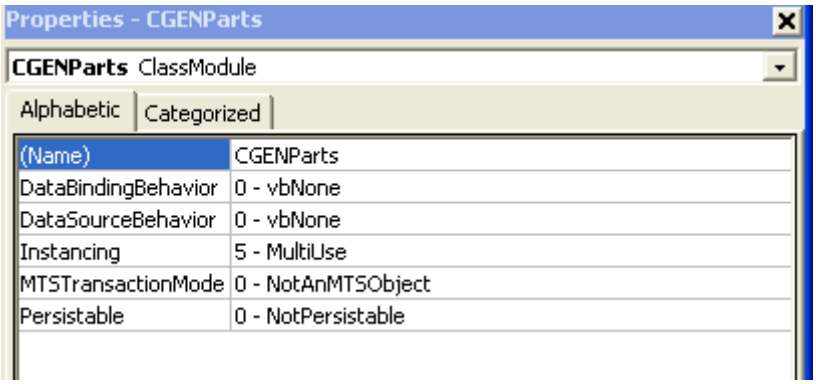


7. Go to the Explorer Window and select the CPartAsm class file node. Select *File -> "Save CPartAsm.cls" As* option to save the class module as CGENParts.cls under HS_RH_ASSY_Hanger1 directory.



8. Go to the Properties Window and change the name of the Project and Class Modules, as shown here:





9. Go to the General Declarations section in CGENPart module and change the value of the *Constant Module variable* from *"Assy_TemplateAsm"* to *" CGENPart"*

   Private Const MODULE = "CGENPart" *'Used for error messages*

10. Add global variables as shown here:

```
Private PIPEDIAM As Double
Private LENGTH As Double
Private ROD_DIA As Double

Private UBOLT_TO  As Double
Private THREADED_LENGTH_TO As Double
Private EYE_NUT_TO As Double
Private BEAM_ATT_TO As Double
```

11. Go to the IJHgrAssmInfo_GetAssemblyCatalogParts method. Add the following instructions to retrieve the pipe diameter and the distance between the reference ports.

```
PIPEDIAM = GetExternalPipeDiameter(my_IJHgrInputConfigHlpr, 1)
LENGTH = my_IJHgrInputConfigHlpr.GetDistanceBetweenPorts("Route", "Structure", VERTICAL)
```

12. Add the following lines to specify the parts constituting the assembly as shown here:

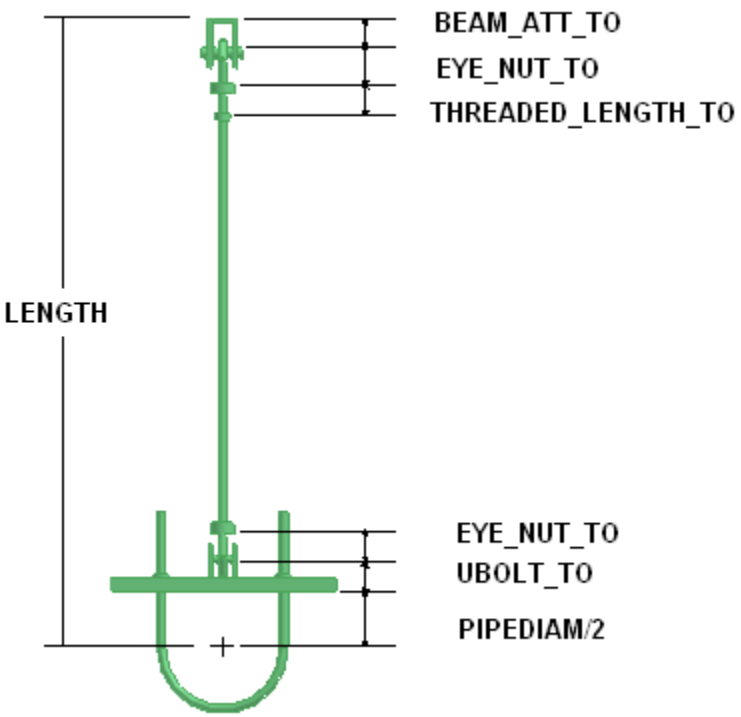
```
 Dim PartClasses(6) As String
PartClasses(1) = "RH_SPUBOLT"
PartClasses(2) = "LRParts_FIG290N_0.5"
PartClasses(3) = "RH-ROD_0.5"
PartClasses(4) = "LRParts_FIG290N_0.5"
PartClasses(5) = "LRParts_FIG66N_0.5"
PartClasses(6) = "LRParts_HEX_NUT_0.5"

Set IJHgrAssmInfo_GetAssemblyCatalogParts = _
                    AddPartsByClassName(pDispInputConfigHlpr, PartClasses)
```

Note:

- If you use the part class name, then the system will use the part selection rule to select the part.

13. Go to the IJHgrAssmInfo_GetAssemblyJoints method. Add the following lines to get the distance between the parts as shown below:

```
Dim parA As Double
Dim parC As Double

ROD_DIA = 0.5 * 0.0254
parC = 4.5 * 0.0254
parA = 0.5 * 0.0254

UBOLT_TO = (parC / 8) + 2 * parA - parA / 2
THREADED_LENGTH_TO = 2.5 * 0.0254
EYE_NUT_TO = 2 * 0.0254 - ROD_DIA / 2#
BEAM_ATT_TO = 2 * 0.0254 - (0.6875 * 0.0254 / 2#)
```



14. Add the following lines to define the joints required to assembly the support.

```
'Create a collection to hold the joints
Dim JointCollection As New Collection
'Create a Joint Factory
Dim JointFactory As New HgrJointFactory
Dim AssemblyJoint As Object

'UBolt
```

```
Set AssemblyJoint = JointFactory.MakeRigidJoint(1, "Route", -1, "Route")
JointCollection.Add AssemblyJoint

'Bottom Eye Nut
Set AssemblyJoint = JointFactory.MakeRigidJoint(2, "InThdRH", -1, "Route", 9508, PIPEDIAM / 2 +
UBOLT_TO + EYE_NUT_TO, 0, 0)
JointCollection.Add AssemblyJoint

'Rod
Set AssemblyJoint = JointFactory.MakeRigidJoint(3, "BotExThdRH", -1, "Route", 9508, PIPEDIAM / 2 +
UBOLT_TO + EYE_NUT_TO)
JointCollection.Add AssemblyJoint

'Set Rod legnth
Dim Rodlength As Double
Rodlength = LENGTH - PIPEDIAM / 2 - UBOLT_TO - EYE_NUT_TO - EYE_NUT_TO - BEAM_ATT_TO
my_IJHgrInputConfigHlpr.SetSymbolInputByName IJElements_PartOccCollection.Item(3), "Length",
Rodlength

'Nut
Set AssemblyJoint = JointFactory.MakeRigidJoint(6, "InThdRH", -1, "Route", 9508, LENGTH -
BEAM_ATT_TO - EYE_NUT_TO - THREADED_LENGTH_TO)
JointCollection.Add AssemblyJoint

'Top Eye Nut
Set AssemblyJoint = JointFactory.MakeRigidJoint(4, "Eye", -1, "Route", 9508, LENGTH -
BEAM_ATT_TO)
JointCollection.Add AssemblyJoint

'Beam Attachment
Set AssemblyJoint = JointFactory.MakeRigidJoint(5, "Structure", -1, "Route", 9380, LENGTH)
JointCollection.Add AssemblyJoint

'Return the collection of Joints
Set IJHgrAssmInfo_GetAssemblyJoints = JointCollection
```

15. Go to the IJHgrAssmInfo_ GetMaxRouteConnectionValue method and add the following line, as shown here:

```
IJHgrAssmInfo_GetMaxRouteConnectionValue = 0
```

16. Go to the IJHgrAssmInfo_ GetRouteConnections method and add the following line, as shown here:

```
Set IJHgrAssmInfo_GetRouteConnections = ConnectPartToRouteOrStruct(1, 1)
```

17. Go to the IJHgrAssmInfo_ GetStructConnections method and add the following line, as shown here:

```
Set IJHgrAssmInfo_GetStructConnections = ConnectPartToRouteOrStruct(5, 1)
```

18. Compile the Visual Basic project and save the dll as HS_RH_ASSY_Hanger1.dll in the HS_RH_ASSY_Hanger1 directory.

19. One of the most important steps in Visual Basic programming is to preserve the binary compatibility of your program. Save the final version of your dll file to be binary compatibility in order to preserve the CLSID.

20. Save and Exit the HS_RH_ASSY_Hanger1.vbp project.

21. Copy the C:\train\HgrTrainingFiles\AssyTemplateV2\ SPAssySupportTraining.xls workbook into HS_RH_ASSY_Hanger1 directory.

22. Open the SPAssySupportTraining.xls workbook.

23. Go the ClassNodeType sheet and add the following row:

Head	ObjectName	Name
Start		
a	CLASSNODE_ASSEMBLY_PIPE_SPASSEMBLY_COMPANY	RH Inc Pipe Support Assemblies
End		

24. Go the R-Hierarchy sheet and add the following row:

Head	RelationSource	RelationDestination
Start		
a	CLASSNODE_ASSEMBLY_PIPE_SPASSEMBLY	CLASSNODE_ASSEMBLY_PIPE_SPASSEMBLY_COMPANY
End		

25. Go the R-ClassNodeDescribes sheet and add the following row:

Head	RelationSource	RelationDestination
Start		
a	CLASSNODE_ASSEMBLY_PIPE_SPASSEMBLY_COMPANY	Assy_RH_Hanger
End		

26. Go to the HrgTemplateAssy sheet and rename it as Assy_RH_Hanger.

27. Go to the class definition and edit the row as shown here:

Definition	PartClassType	SymbolIcon	UserClassName	OccClassName
a	PipeSupportDefinitionClass		Assy_RH_Hanger	Assy_RH_Hanger

28. Now, you will need to add the assembly part definition as shown below:

Head	PartNumber	PartDescription	DisciplineType	CommandType	TypeSelectionRule	AssmInfoRule
Start						
a	Assy_RH_Hanger_01	Assy_RH_Hanger_01	1	1	1	HS_RH_ASSY_Hanger1.CGENParts
End						

MaxInsulation	LoadRangeLow	LoadRangeHigh	MinAssemblyLength	MaxAssemblyLength	Supporting_Count	Supported_Count	NDFrom	NDTo	NDUnitType	NominalWidthFrom	NominalWidthTo	NominalDepthFrom	NominalDepthTo	SupportedFamily	FaceSelectionType	InsulationPurpose	SupportType	MirrorBehaviorOption
-1	0	0	0in	1000in	1	1	2	6	in	0	0	0	0	STRAIGHT	0	2	11	15

29. Save and Exit the SPAssySupportTraining.xls workbook.

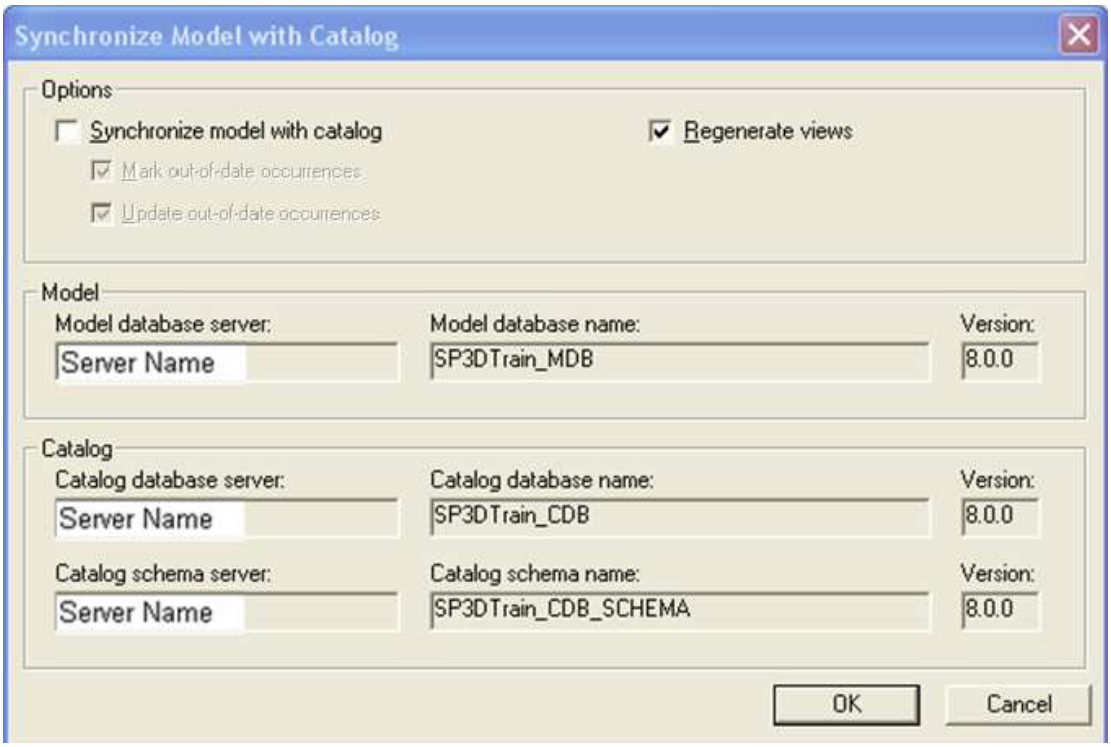
30. Load the information into the Catalog using the A/M/D Mode. Once the bulkload process is completed, review the log file.

31. Run the Project Management Task. Select the Model in the hierarchy.

32. Select Tools -> Synchronize Model with the Catalog.

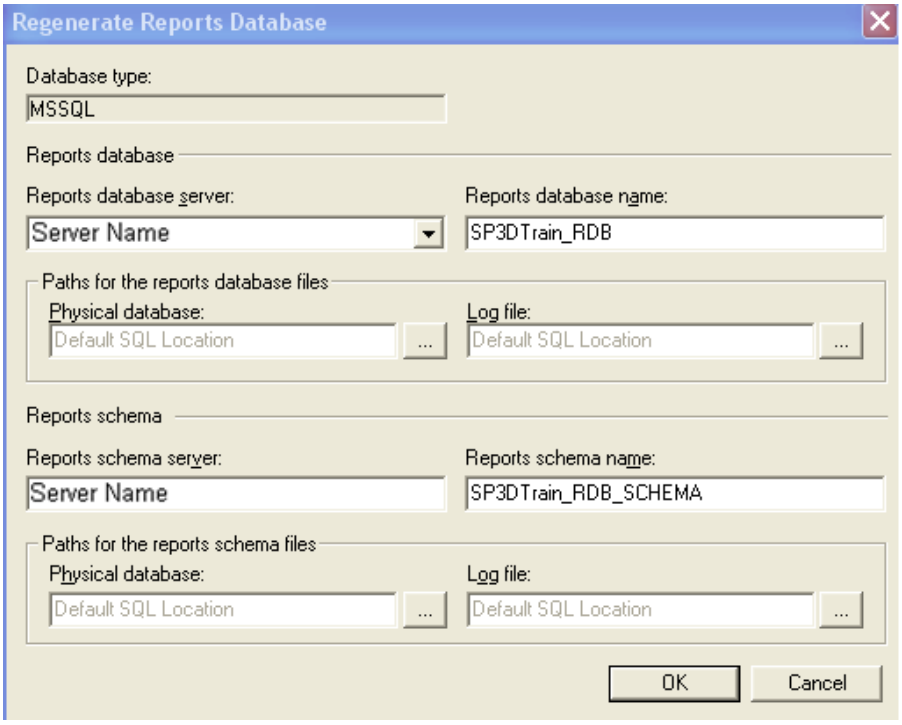
33. Uncheck the Synchronize Model with the Catalog option.

***Note**: You just need to update the views in the model.*



34. Hit “OK” Button.

35. Once the process is completed, Right click the training plant icon and select “Regenerate the reports database” option to re-create the views in the report database.



36. Hit OK to start the process.

37. Once the process is completed, go to the Hangers and Supports Task and place the Assy_RH_Hanger_01 support using the Place Support by Structure Command.

Lab 3b: Assembly Information Rule - Method of Proper Joints

Objectives

After completing this lab, you will be able to:

- Use the methods provided by IJHgrAssmInfo
- Define joint data for this assembly
- Use part selection rule
- Method of proper joints and flexible members

In this lab, you will create an assembly information rule of the hanger created in previous labs using the method of proper joints and flexible members.

1. Create a directory called HS_RH_ASSY_Hanger2 under c:\train\HgrTrainingFiles

2. Copy the all the files create in lab3a to HS_RH_ASSY_Hanger2.

3. Ope the Project HS_RH_ASSY_Hanger1.vbp

4. Remove the global variables for the take-outs. Keep the *THREADED_LENGTH_TO variable* as shown here:

```
Private PIPEDIAM As Double
Private LENGTH As Double

Private THREADED_LENGTH_TO As Double
```

5. Go to the IJHgrAssmInfo_GetAssemblyJoints method. Replace all lines as shown below:

```
'<<Get Attributes on objects

    THREADED_LENGTH_TO = 2.5 * 0.0254

'<< Get interface for accessing items on the collection of Part Occurences

    Dim IJElements_PartOccCollection As IJElements
    Set IJElements_PartOccCollection = pDispPartOccCollection

    'Resume normal error handling.
    On Error GoTo ErrorHandler:

    '=====================================================
    'Get the Current Location in the Route Connection Cycle
    '=====================================================
    Dim RouteConnectionValue As Integer
    RouteConnectionValue = my_IJHgrInputConfigHlpr.GetRouteConnectionValue
```

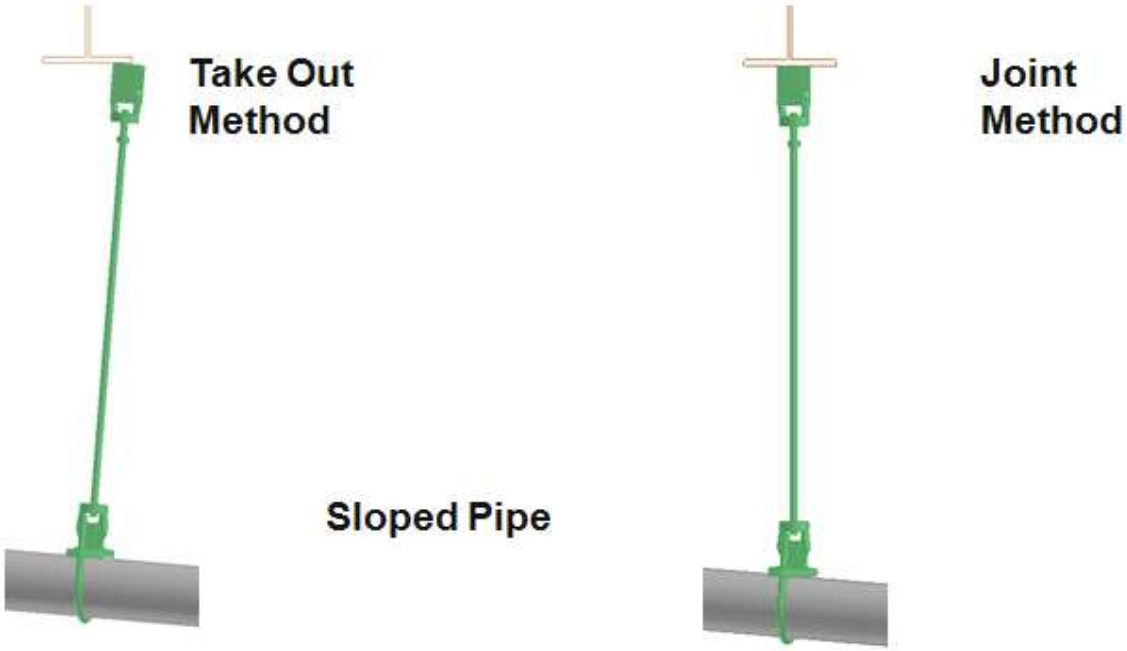
```
'<<Create Joints

        'Create a collection to hold the joints
        Dim JointCollection As New Collection
        'Create a Joint Factory
        Dim JointFactory As New HgrJointFactory
        Dim AssemblyJoint As Object

'UBolt
        Set AssemblyJoint = JointFactory. MakeRigidJoint (1, "Route", -1, "Route")
        JointCollection.Add AssemblyJoint
'Bottom Eye Nut
        Set AssemblyJoint = JointFactory.MakeRevoluteJoint(2, "Eye", 1, "Pin", 81)
        JointCollection.Add AssemblyJoint
'Rod
        Set AssemblyJoint = JointFactory.MakePrismaticJoint(3, "TopExThdRH", 3, "BotExThdRH", 3501)
        JointCollection.Add AssemblyJoint
        Set AssemblyJoint = JointFactory.MakeRigidJoint(2, "InThdRH", 3, "BotExThdRH", 10468)
        JointCollection.Add AssemblyJoint
        Set AssemblyJoint = JointFactory.MakeVerticalJoint(3, "TopExThdRH")
        JointCollection.Add AssemblyJoint
'Nut
    Set AssemblyJoint = JointFactory.MakeRigidJoint(6, "InThdRH", 3, "TopExThdRH", 9508, _
                                        -THREADED_LENGTH_TO)
    JointCollection.Add AssemblyJoint

'Top Eye Nut
        Set AssemblyJoint = JointFactory.MakeRigidJoint(4, "InThdRH", 3, "TopExThdRH", 10468)
        JointCollection.Add AssemblyJoint

        Set AssemblyJoint = JointFactory.MakeRevoluteJoint(4, "Eye", 5, "Pin", 17)
        JointCollection.Add AssemblyJoint

'Beam Attachment
        Set AssemblyJoint = JointFactory.MakePlanarJoint(5, "Structure", -1, "Structure", 9572)
        JointCollection.Add AssemblyJoint

        'Return the collection of Joints
        Set IJHgrAssmInfo_GetAssemblyJoints = JointCollection
```

6. Compile the Visual Basic project and save the dll as HS_RH_ASSY_Hanger2.dll in the HS_RH_ASSY_Hanger1 directory.

7. One of the most important steps in Visual Basic programming is to preserve the binary compatibility of your program. Save the final version of your dll file to be binary compatibility in order to preserve the CLSID.

8. Save and Exit the HS_RH_ASSY_Hanger2.vbp project.

9. Go to the Hangers and Supports Task and place the Assy_RH_Hanger_01 support using the Place Support by Structure Command.


Take Out
Method
Joint
Method
Sloped Pipe

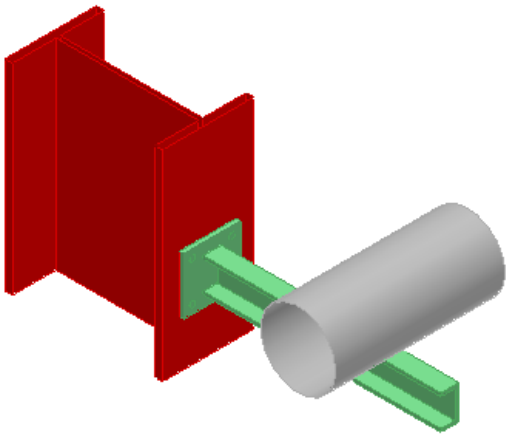
Lab 4: Assembly Information Rule Using Connection Component Parts

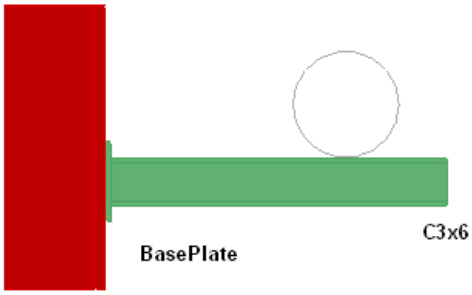
Objectives

After completing this lab, you will be able to:

- Use the methods provided by IJHgrAssmInfo
- Use the route connection toggle configurations
- Define the bounding box
- Define joint data for this assembly

In this lab, you will create the assembly information rule to build a horizontal C3x6 frame using the delivered connection component as shown below. You start by using the Assy_TemplateAsm template provided by the instructor to create the rule.

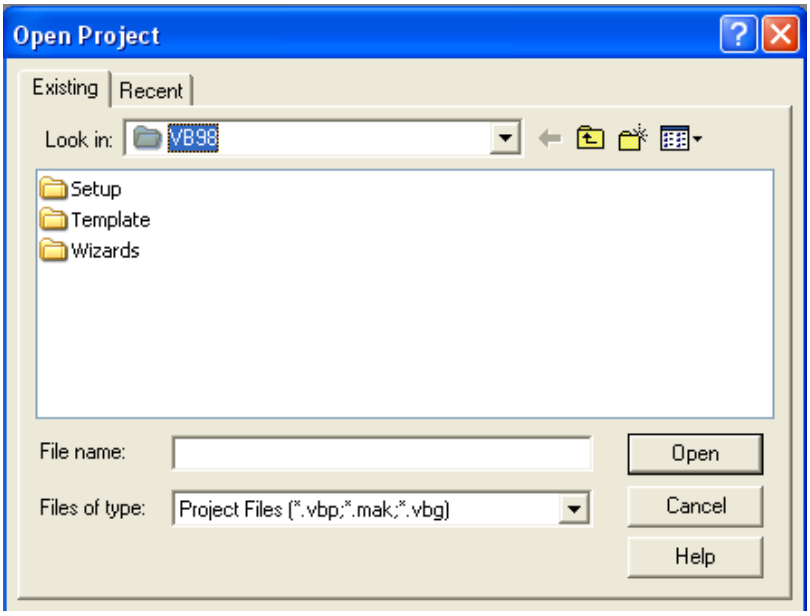




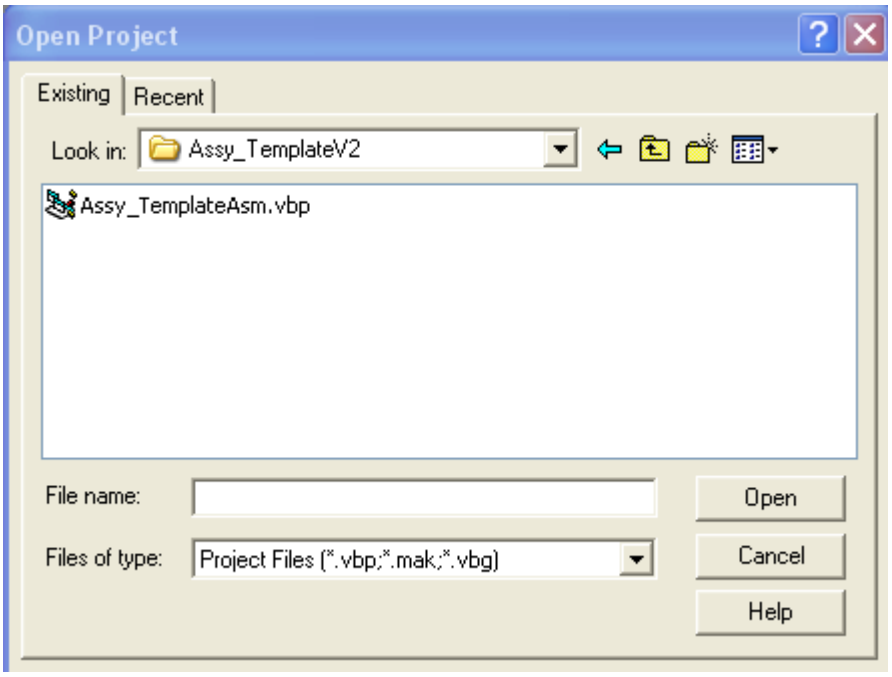
1. Create a directory called HS_RH_ASSY_SingleFrame under c:\train\HgrTrainingFiles and run Microsoft Visual Basic 6.0

2. Close the Microsoft New Project dialog box.

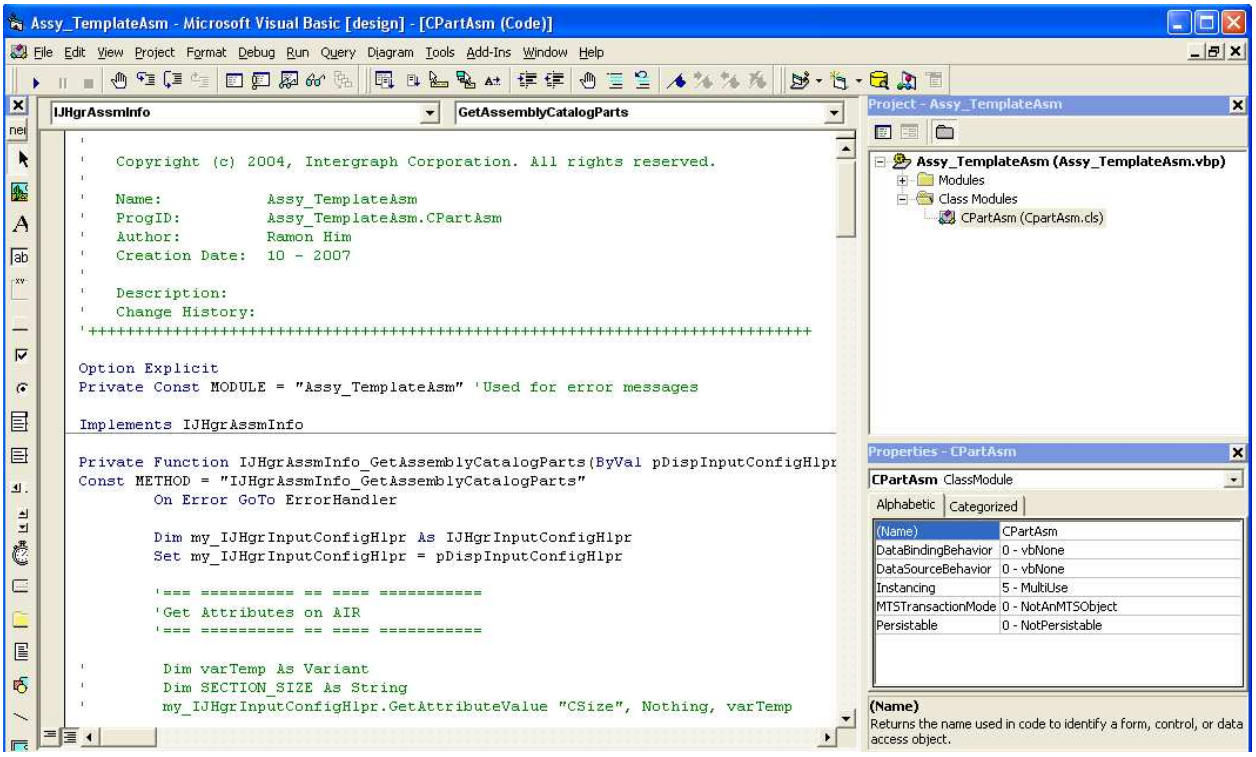
3. Select *File -> Open Project* option to open the Open Project Dialog box.



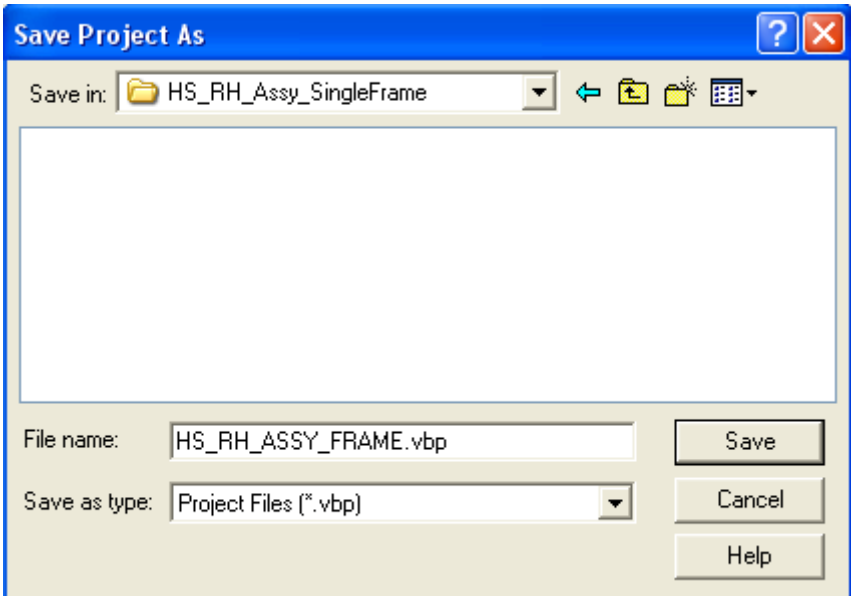
4. Navigate to c:\train\HgrTrainingFiles\ AssyTemplateV2 and open the Assy_TemplateAsm.vbp Template project.



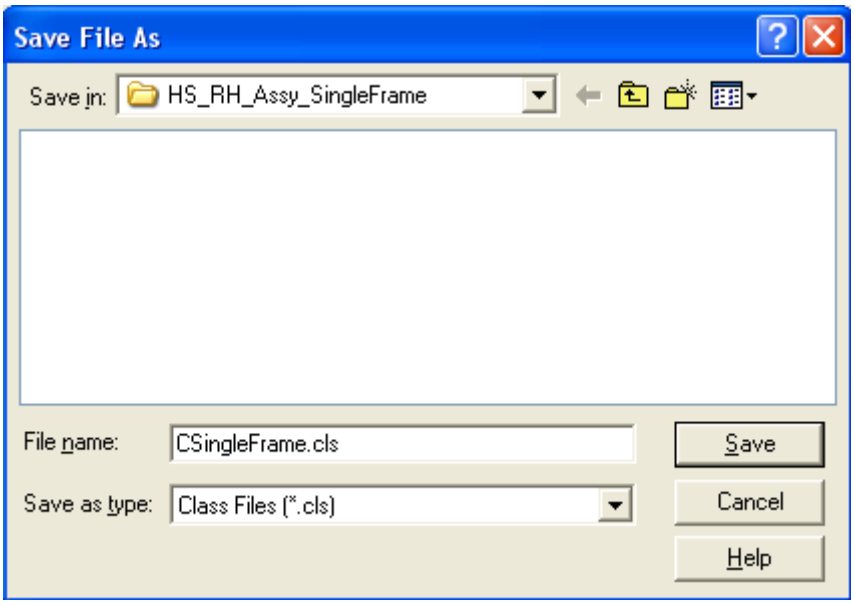
5. Setup the Visual Basic Development Environment as shown below:



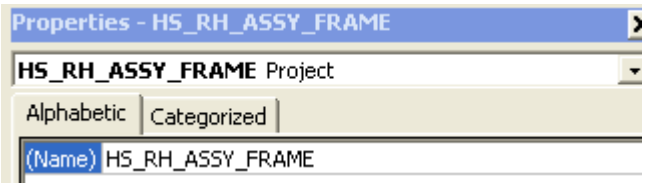
6. Go to the Explorer Window and select the Project file node. Select *File* -> *"Save Project As"* option to save the project as HS_RH_ASSY_FRAME.vbp under the HS_RH_Assy_SingleFrame directory.

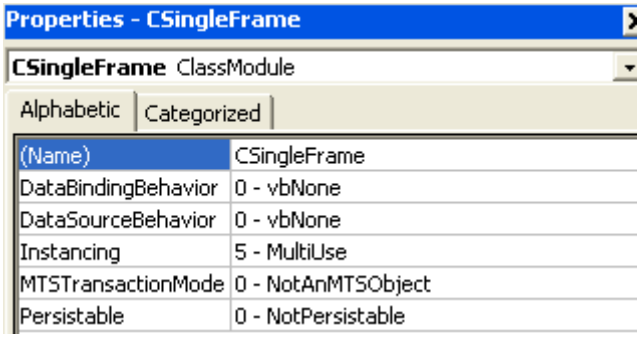


7. Go to the Explorer Window and select the CPartAsm class file node. Select *File -> "Save CPartAsm.cls" As* option to save the class module as CSingleFrame.cls under HS_RH_Assy_SingleFrame directory.



8. Go to the Properties Window and change the name of the Project and Class Modules as shown here:





9. Go to the General Declarations section in CSingleFrame module and change the value of the *Constant Module variable* from *"Assy_TemplateAsm"* to *"CSingleFrame"*

```
Private Const MODULE = "CSingleFrame" 'Used for error messages
```

10. Add global variables as shown below:

```
Private Const nSection = 1
Private Const TOP_PLATE = 2

Private OVERHANG As Double
Private SECTION_SIZE As String
Private SECTION_SIZE1 As String

Private PLATE_SIZE As String
Private BP_WIDTH As Double
Private BP_HOLE_SIZE As Double
Private BP_HOLE_INSET As Double
Private BP_THICKNESS As Double
```

11. Go to the IJHgrAssmInfo_GetAssemblyCatalogParts method. Add the following lines to retrieve dimensional data from the assembly part definition as shown here:

```
Dim varTemp As Variant
my_IJHgrInputConfigHlpr.GetAttributeValue "CSize", Nothing, varTemp
SECTION_SIZE = Trim(varTemp)
SECTION_SIZE1 = Mid(SECTION_SIZE, 9)

my_IJHgrInputConfigHlpr.GetAttributeValue "OVERHANG", Nothing, varTemp
OVERHANG = varTemp
my_IJHgrInputConfigHlpr.GetAttributeValue "PLATE_SIZE", Nothing, varTemp
PLATE_SIZE = varTemp
my_IJHgrInputConfigHlpr.GetAttributeValue "BP_HOLE_SIZE", Nothing, varTemp
BP_HOLE_SIZE = varTemp
my_IJHgrInputConfigHlpr.GetAttributeValue "BP_HOLE_INSET", Nothing, varTemp
BP_HOLE_INSET = varTemp
my_IJHgrInputConfigHlpr.GetAttributeValue "BP_WIDTH", Nothing, varTemp
BP_WIDTH = varTemp
my_IJHgrInputConfigHlpr.GetAttributeValue "BP_THICKNESS", Nothing, varTemp
BP_THICKNESS = varTemp
```

12. Add the following lines to specify the part class constituting the assembly as shown below:

```
ReDim PartClasses(2) As String
PartClasses(nSection) = Trim(SECTION_SIZE) 'L Section
PartClasses(TOP_PLATE) = "Utility_FOUR_HOLE_PLATE_" + PLATE_SIZE
```

13. Go to the IJHgrAssmInfo_ GetAssemblyJoints method. Add the following lines to retrieve the diameter and radius of the primary pipe.

```
Dim dPipeDia As Double
dPipeDia = GetExternalPipeDiameter(my_IJHgrInputConfigHlpr, 1)

Dim dPipeRadius As Double
dPipeRadius = GetPipeODwithoutInsulat(dPipeDia, oRoute) / 2

Dim dInsulat As Double
dInsulat = GetPipeInsulat(oRoute)
```

14. Add the following lines to get the bounding box dimensions.

```
'==========================
'1. Load standard bounding box definition
'==========================
Dim my_IJHgrBBXHlpr As IJHgrBBXHlpr
Set my_IJHgrBBXHlpr = pDispInputConfigHlpr
my_IJHgrBBXHlpr.LoadBoundingBox
'====== ======
'2. retrieve dimension of the bounding box
'====== ======
' Get route box geometry
'
'  ______________________
' |                      |
' |  ROUTE BOX BOUND     | dHeight
' |______________________|
'        dWidth
'
Dim dimension() As Double
Dim dWidth As Double, dHeight As Double

dimension = GetBoundingBoxDimension(pDispInputConfigHlpr)
dWidth = dimension(1)
dHeight = dimension(2)
```

15. Add the following lines to retrieve the beam dimensions and distance between the reference ports.

```
' Get dimension from HgBeam

Dim SteelWidth As Double
Dim SteelDepth As Double
```

```
Dim SteelThickness As Double
SteelThickness = GetProfileDim(nSection, "Flange", IJElements_PartOccCollection)
SteelWidth = GetProfileDim(nSection, "Width", IJElements_PartOccCollection)
SteelDepth = GetProfileDim(nSection, "Depth", IJElements_PartOccCollection)

'Get the distance between the structure and route ports
Dim dDistPorts As Double
Dim tempL As Double

dDistPorts = my_IJHgrInputConfigHlpr.GetDistanceBetweenPorts("BBR_Low", "Structure",
HORIZONTAL)
tempL = dDistPorts
```

16. Add the following lines to get the current configuration based on the route connection toggle.

```
    '=====================================================
    'Get the Current Location in the Route Connection Cycle
    '=====================================================
    Dim RouteConnectionValue As Integer
    RouteConnectionValue = my_IJHgrInputConfigHlpr.GetRouteConnectionValue

    Dim iStructConfIndex As Integer

' Set the confIndex for the toggle route connection
    If RouteConnectionValue = 1 Then
        iStructConfIndex = 10404
    ElseIf RouteConnectionValue = 2 Then
        iStructConfIndex = 2212
    ElseIf RouteConnectionValue = 3 Then
        iStructConfIndex = 2212
    Else
        iStructConfIndex = 10404
    End If
```

17. Add the following lines to set the base plate dimensions

```
'Set input to part symbols
'Plates
  my_IJHgrInputConfigHlpr.SetSymbolInputByName
IJElements_PartOccCollection.Item(TOP_PLATE), "WIDTH", BP_WIDTH       ' 5 inches
  my_IJHgrInputConfigHlpr.SetSymbolInputByName
IJElements_PartOccCollection.Item(TOP_PLATE), "DEPTH", BP_WIDTH       ' 5 inches
  my_IJHgrInputConfigHlpr.SetSymbolInputByName
IJElements_PartOccCollection.Item(TOP_PLATE), "C", BP_HOLE_INSET      ' 3/4 inches
  my_IJHgrInputConfigHlpr.SetSymbolInputByName
IJElements_PartOccCollection.Item(TOP_PLATE), "HOLE_SIZE", BP_HOLE_SIZE  ' 9/16 inches bolt
holes
  my_IJHgrInputConfigHlpr.SetSymbolInputByName
IJElements_PartOccCollection.Item(TOP_PLATE), "THICKNESS", BP_THICKNESS
```

18. Add the following lines to set the beam attributes.

```
    Dim iArrayOfKeys(2) As Integer
```

```
    iArrayOfKeys(1) = nSection

  SetValuesOfSymbolInputsAndOccAttributes my_IJHgrInputConfigHlpr,
  IJElements_PartOccCollection, iArrayOfKeys, "EndOverLength", 0
  SetValuesOfSymbolInputsAndOccAttributes my_IJHgrInputConfigHlpr,
  IJElements_PartOccCollection, iArrayOfKeys, "BeginOverLength", 0
  SetValuesOfSymbolInputsAndOccAttributes my_IJHgrInputConfigHlpr,
  IJElements_PartOccCollection, iArrayOfKeys, "Orientation", 0
  SetValuesOfSymbolInputsAndOccAttributes my_IJHgrInputConfigHlpr,
  IJElements_PartOccCollection, iArrayOfKeys, "BeginMiter", "No End Treatment"
  SetValuesOfSymbolInputsAndOccAttributes my_IJHgrInputConfigHlpr,
  IJElements_PartOccCollection, iArrayOfKeys, "EndMiter", "No End Treatment"
```

19. Add the following lines to define the joints required to assembly the support.

```
'-------------
'Create joints
'-------------
'Create a collection to hold the joints
Dim JointCollection As New Collection

'Create a Joint Factory
Dim JointFactory As New HgrJointFactory
'Create an object to hold the Joint
Dim AssemblyJoint As Object

'TOP PLATE

 If RouteConnectionValue = 1 Then
Set AssemblyJoint = JointFactory.MakeRigidJoint(-1, "Structure", TOP_PLATE, "TopStructure", 9380,
BP_THICKNESS, 0, -SteelDepth / 2 - dPipeRadius)
ElseIf RouteConnectionValue = 2 Then
Set AssemblyJoint = JointFactory.MakeRigidJoint(-1, "Structure", TOP_PLATE, "TopStructure", 9380,
BP_THICKNESS, 0, -SteelDepth / 2 - dPipeRadius)
ElseIf RouteConnectionValue = 3 Then
Set AssemblyJoint = JointFactory.MakeRigidJoint(-1, "Structure", TOP_PLATE, "TopStructure", 9380,
BP_THICKNESS, 0, SteelDepth / 2 + dPipeRadius)
Else
Set AssemblyJoint = JointFactory.MakeRigidJoint(-1, "Structure", TOP_PLATE, "TopStructure", 9380,
BP_THICKNESS, 0, SteelDepth / 2 + dPipeRadius)
End If
JointCollection.Add AssemblyJoint

'Horizontal Brace length

my_IJHgrInputConfigHlpr.SetSymbolInputByName IJElements_PartOccCollection.Item(nSection),
"Length", tempL - BP_THICKNESS + OVERHANG

If RouteConnectionValue = 1 Then
Set AssemblyJoint = JointFactory.MakeRigidJoint(nSection, "BeginCap", -1, "Structure",
iStructConfIndex, tempL + OVERHANG, SteelDepth + dPipeRadius, SteelWidth / 2)
ElseIf RouteConnectionValue = 2 Then
Set AssemblyJoint = JointFactory.MakeRigidJoint(nSection, "BeginCap", -1, "Structure",
iStructConfIndex, tempL + OVERHANG, -dPipeRadius, SteelWidth / 2)
ElseIf RouteConnectionValue = 3 Then
```

```
Set AssemblyJoint = JointFactory.MakeRigidJoint(nSection, "BeginCap", -1, "Structure",
iStructConfIndex, tempL + OVERHANG, SteelDepth + dPipeRadius, SteelWidth / 2)
Else
Set AssemblyJoint = JointFactory.MakeRigidJoint(nSection, "BeginCap", -1, "Structure",
iStructConfIndex, tempL + OVERHANG, -dPipeRadius, SteelWidth / 2)
End If
JointCollection.Add AssemblyJoint

 'Return the collection of Joints
Set IJHgrAssmInfo_GetAssemblyJoints = JointCollection
```

20. Go to the IJHgrAssmInfo_GetMaxRouteConnectionValue method and add the following line, as shown here:

```
IJHgrAssmInfo_GetMaxRouteConnectionValue = 4
```

21. Go to the IJHgrAssmInfo_ GetRouteConnections method and add the following line, as shown here:

```
Set IJHgrAssmInfo_GetRouteConnections = ConnectPartToRouteOrStruct(nSection, 1)
```

22. Go to the IJHgrAssmInfo_ GetStructConnections method and add the following line, as shown here:

```
Set IJHgrAssmInfo_GetStructConnections = ConnectPartToRouteOrStruct(nSection, 1)
```

23. Compile the Visual Basic project and save the dll as HS_RH_ASSY_FRAME.dll in the HS_RH_Assy_SingleFrame directory.

24. One of the most important steps in Visual Basic programming is to preserve the binary compatibility of your program. Save the final version of your dll file to be binary compatibility in order to preserve the CLSID.

25. Save the HS_RH_ASSY_FRAME.vbp project.

26. Copy the c:\train\HgrTrainingFiles\AssyTemplateV2\SPAssySupportTraining.xls workbook into HS_RH_Assy_SingleFrame directory.

27. Rename SPAssySupportTraining.xls to Assy_RH_SingleFrame.xls

28. Open the Assy_RH_SingleFrame.xls workbook.

29. Go the R-ClassNodeDescribes sheet and add the following row:

Head	RelationSource	RelationDestination
Start		
!	Support Assembly Classes	
a	CLASSNODE_ASSEMBLY_PIPE_SPASSEMBLY_COMPANY	Assy_RH_SingleFrame
End		

30. Go to the HrgTemplateAssy sheet and rename it as Assy_RH_SingleFrame.

CustomInterfaces | **Assy_RH_SingleFrame**

31. Go to the class definition and edit the row as shown here:

Definition	PartClassType	oa:IJUAHgrCSize::CSize	oa:IJUAHgrAssySFrame::OVERHANG	oa:IJUAHgrAssySFrame::PLATE_SIZE	oa:IJUAHgrAssySFrame::BP_WIDTH	oa:IJUAHgrAssySFrame::BP_HOLE_SIZE	oa:IJUAHgrAssySFrame::BP_HOLE_INSET	IJUAHgrAssySFrame::BP_THICKNESS	UserClassName	OccClassName
a	PipeSupportDefinitionClass								RH Single Frame	RH Single Frame

32. Now, you will need to add custom attributes in the part definition, as shown here:

Head	PartNumber	PartDescription	SymbolDefinition	SymbolIcon	DisciplineType	CommandType	TypeSelectionRule	AssmInfoRule	MaxInsulation	LoadRangeLow	LoadRangeHigh	MinAssemblyLength	MaxAssemblyLength	IJUAHgrCSize::CSize	IJUAHgrAssySFrame::OVERHANG	IJUAHgrAssySFrame::PLATE_SIZE	IJUAHgrAssySFrame::BP_WIDTH	IJUAHgrAssySFrame::BP_HOLE_SIZE	IJUAHgrAssySFrame::BP_HOLE_INSET	IJUAHgrAssySFrame::BP_THICKNESS

Supporting_Count	Supported_Count	NDFrom	NDTo	NDUnitType	NominalWidthFrom	NominalWidthTo	NominalDepthFrom	NominalDepthTo	SupportedFamily	FaceSelectionType	InsulationPurpose	SupportType	MirrorBehaviorOption

33. Now, you will need to add the assembly part definition as shown here:

Head	PartNumber	PartDescription	SymbolDefinition	SymbolIcon	DisciplineType	CommandType	TypeSelectionRule
Start							
a	Assy_RH_Single_Frame_1	Assy_RH_Single_Frame_1			1	1	4
END							

34. Continue add the appropriate data for the other attributes as shown here

AssmInfoRule	MaxInsulation	LoadRangeLow	LoadRangeHigh	MinAssemblyLength	MaxAssemblyLength
HS_RH_ASSY_FRAME.CSingleFrame	-1	0	0	24in	60in

Supporting_Count	Supported_Count	NDFrom	NDTo	NDUnitType	NominalWidthFrom	NominalWidthTo	NominalDepthFrom	NominalDepthTo	SupportedFamily	FaceSelectionType	InsulationPurpose	SupportType	MirrorBehaviorOption
1	1+	2	8	in	0	0	0	0	STRAIGHT	0	2	74	15

35. Key-in the appropriate Section name, overhang and plate dimensions as shown here:

IJUAHgrCSize::CSize	IJUAHgrAssySFrame::OVERHANG	IJUAHgrAssySFrame::PLATE_SIZE	IJUAHgrAssySFrame::BP_WIDTH	IJUAHgrAssySFrame::BP_HOLE_SIZE	IJUAHgrAssySFrame::BP_HOLE_INSET	IJUAHgrAssySFrame::BP_THICKNESS
HsSteel_C3x6	3	5	5	0.5625in	0.8	0.3750

Next step, you will add a new custom interfaces for the Assy_RH_SingleFrame class. Use the Custom Interfaces sheet to define the attributes names with associated data type and unit type.

36. Create a new interface called IJUAHgrAssySFrame. Go to the Custom Interface sheet and add the following entries:

InterfaceName	CategoryName	AttributeName	AttributeUserName	Type	UnitsType	PrimaryUnits	OnPropertyPage	ReadOnly	SymbolParameter
IJUAHgrAssySFrame	Dimensions	PLATE_SIZE	Plate Size	Char	0	0	0	1	PLATE_SIZE
		OVERHANG	Overhang	Double	distance	in	0	1	OVERHANG
		BP_WIDTH	Baseplate Width	Double	distance	in	0	1	BP_WIDTH
		BP_HOLE_SIZE	Baseplate Hole Size	Double	distance	in	0	1	BP_HOLE_SIZE
		BP_HOLE_INSET	Baseplate Hole Inset	Double	distance	in	0	1	BP_HOLE_INSET
		BP_THICKNESS	Baseplate Thickness	Double	distance	in	0	1	BP_THICKNESS

37. Save and Exit the Assy_RH_SingleFrame.xls workbook.

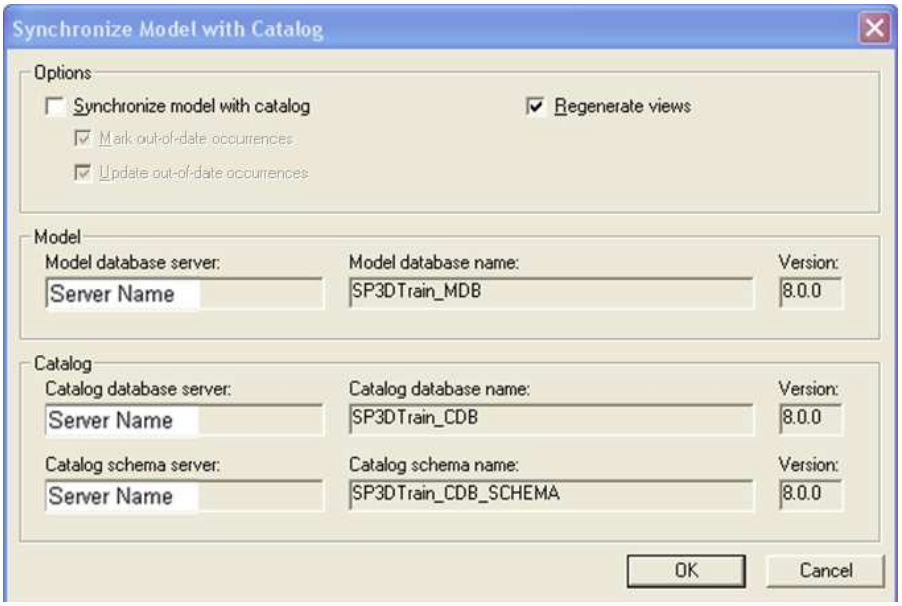
38. Load the information into the Catalog using the A/M/D Mode. Once the bulkload process is completed, review the log file.

39. Run the Project Management Task. Select the Model in the hierarchy.

40. Select Tools -> Synchronize Model with the Catalog.

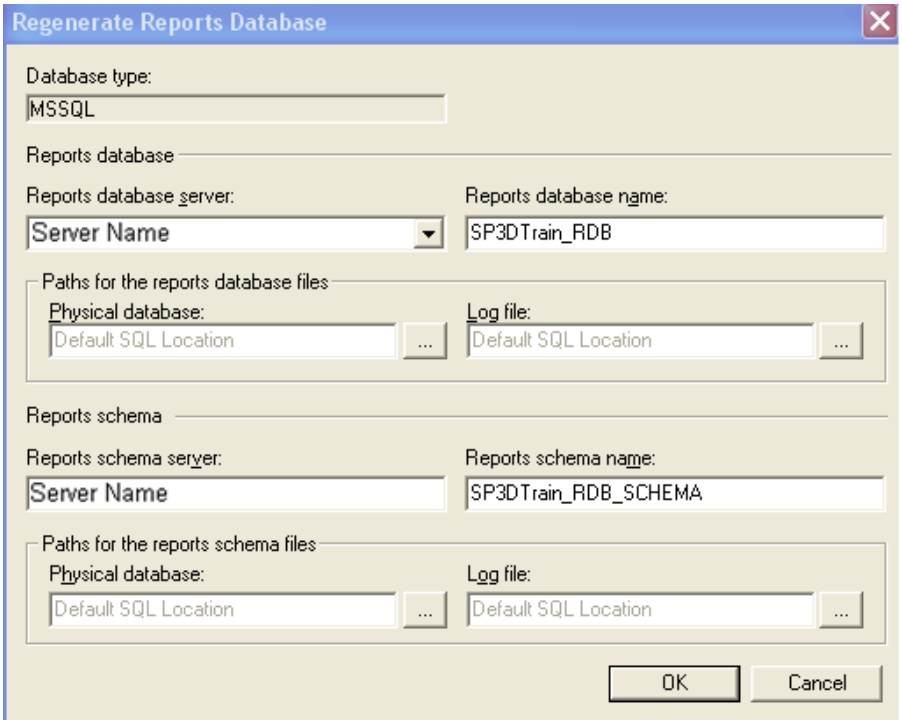
41. Uncheck the Synchronize Model with the Catalog option.

***Note**: You just need to update the views in the model.*



42. Hit “OK” Button.

43. Once the process is completed, Right click the training plant icon and select “Regenerate the reports database” option to re-create the views in the report database.



44. Hit OK to start the process.

45. Once the process is completed, go to the Hangers and Supports Task and place the Assy_RH_Single_Frame_1 support using the Place Support by Structure Command.

Lab 5: Using Table Lookup in the Assembly Information Rule

Objectives

After completing this lab, you will be able to:

- Use the methods provided by IJHgrAssmInfo
- Use table lookups
- Use helper functions to retrieve data from database
- Use the IJHgrSymbolWCGServices
- Define joint data for this assembly

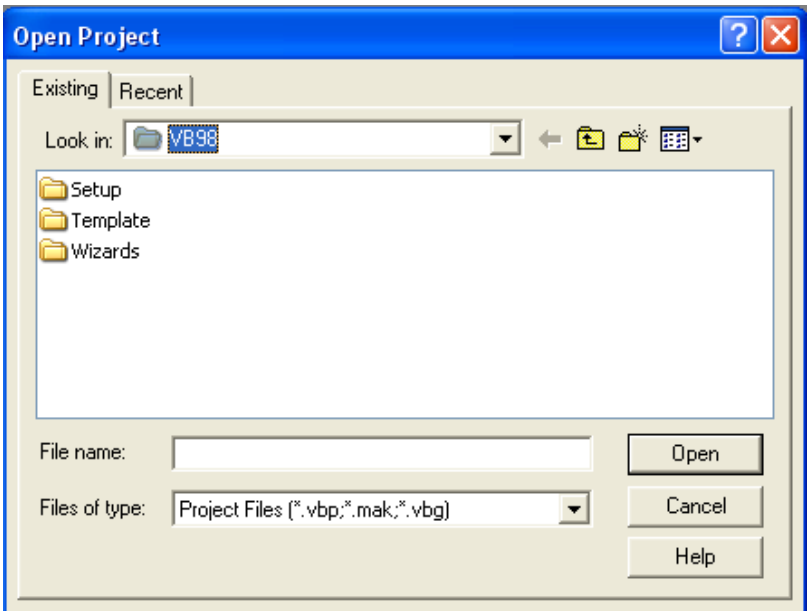
In this lab, you will create an assembly information rule to build a hanger using queries from a table lookup as shown below. The query returns a rod diameter and the part class name (U-bolt or a Pipe Clamp) based on the diameter of the pipe and the piping specification of the piperun. You start by using the Assy_TemplateAsm template provided by the instructor to create the rule.

Definition	*PartClassType*	UserClassName					
a	HgrServiceClass	RH Hangers Auxiliary Table					
Head	Name	IJUAHgrRH_HANGER_TBL::PIPE_DIAM	IJUAHgrRH_HANGER_TBL::DimensionUnitType	IJUAHgrRH_HANGER_TBL::PIPE_OD	IJUAHgrRH_HANGER_TBL::ROD_DIA	IJUAHgrRH_HANGER_TBL::ROUTE_PART	IJUAHgrRH_HANGER_TBL::SPECNAME
Start							
a	RH_HANGER_TBL_1	2	in	2.375	0.375	RH_SPCLAMP	1C0031
a	RH_HANGER_TBL_2	4	in	4.5	0.5	RH_SPUBOLT	1C0031
a	RH_HANGER_TBL_3	6	in	6.625	0.625	RH_SPUBOLT	1C0031
a	RH_HANGER_TBL_4	2	in	2.375	0.375	RH_SPUBOLT	2C0032
a	RH_HANGER_TBL_5	4	in	4.5	0.5	RH_SPCLAMP	2C0032
a	RH_HANGER_TBL_6	6	in	6.625	0.625	RH_SPCLAMP	2C0032
End							

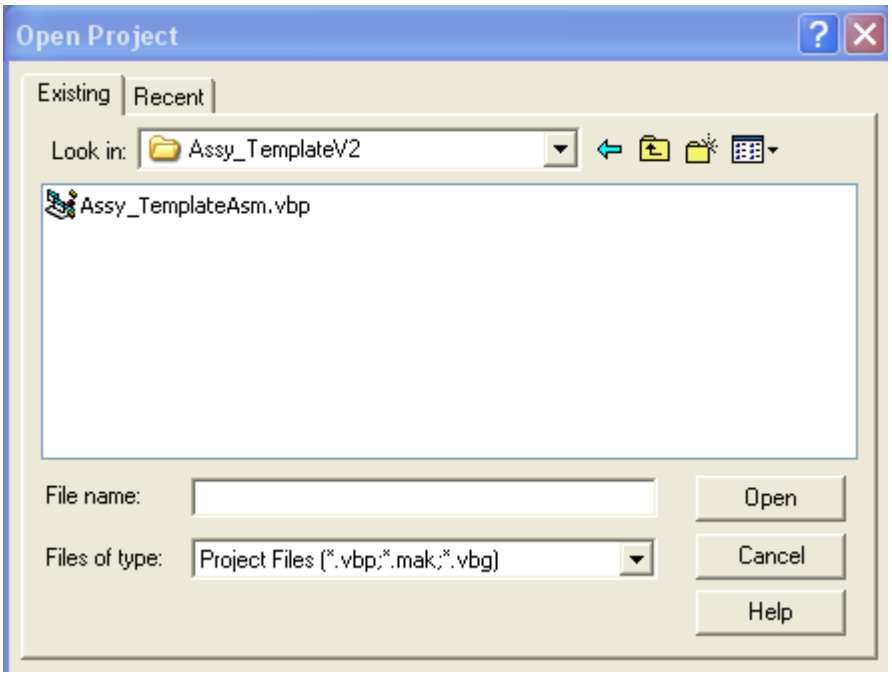
1. Create a directory called HS_RH_ASSY_Hanger3 directory under c:\train\HgrTrainingFiles and run Microsoft Visual Basic 6.0

2. Close the Microsoft New Project dialog box.

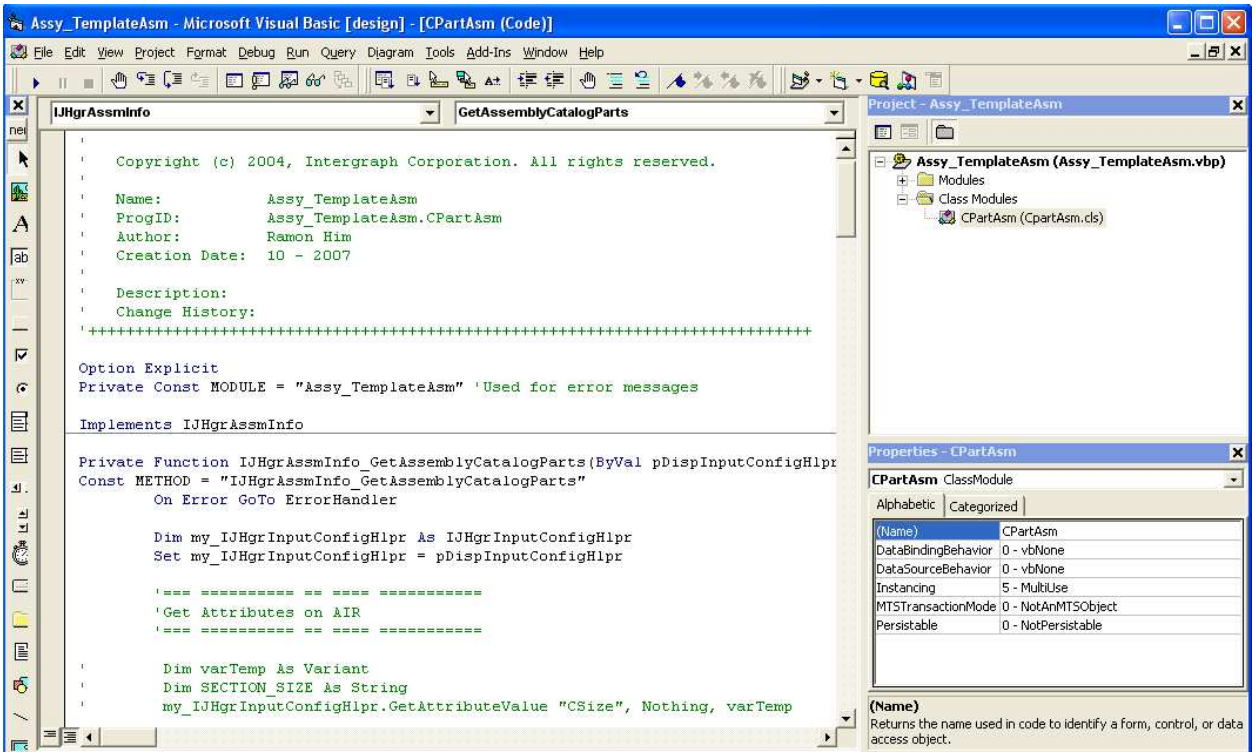
3. Select *File -> Open Project* option to open the Open Project Dialog box.



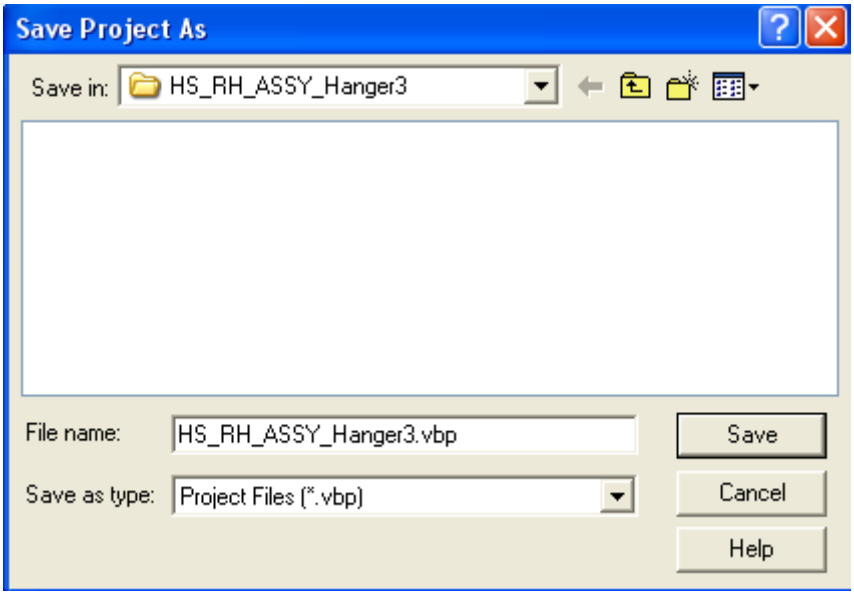
4. Navigate to c:\train\HgrTrainingFiles\ AssyTemplateV2 and open the Assy_TemplateAsm.vbp Template project.



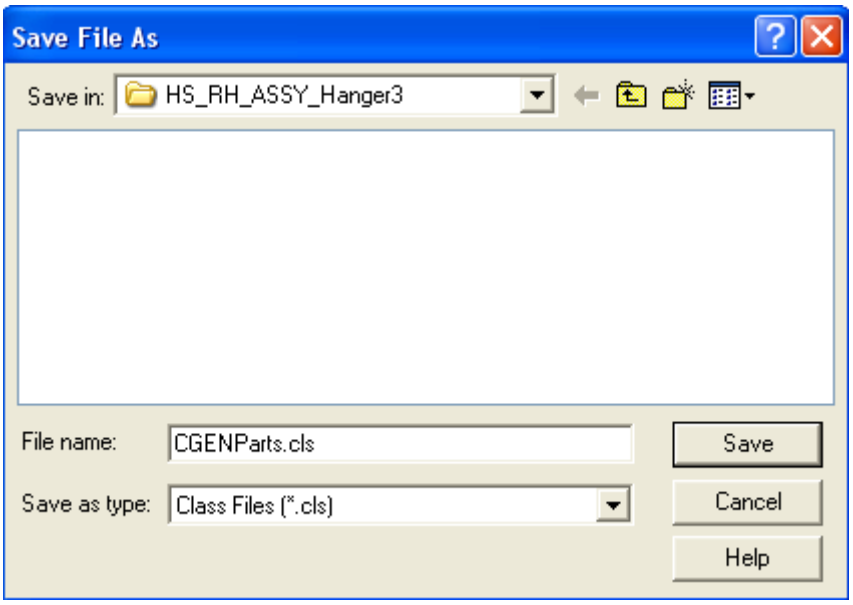
5. Setup the Visual Basic Development Environment as shown below:



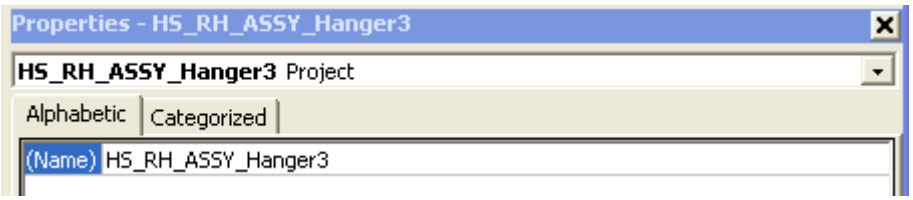
6. Go to the Explorer Window and select the Project file node. Select *File -> "Save Project As"* option to save the project as HS_RH_ASSY_Hanger3.vbp under the HS_RH_ASSY_Hanger3 directory.

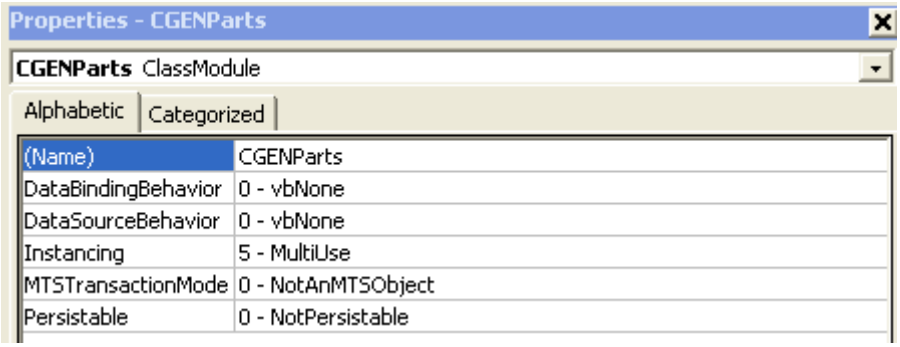


7. Go to the Explorer Window and select the CPartAsm class file in the tree. Select *File -> Save CPartAsm.cls As* option to save the class module as CGENParts.cls under HS_RH_ASSY_Hanger3 directory.



8. Go to the Properties Window and change the name of the Project and Class Modules as follows:





9. Open HS_RH_ASSY_Hanger2 project and copy all the lines to HS_RH_ASSY_Hanger3 project.

10. If you did the previous labs, Add the ROUTEPART variable and then verify all the other global variables are declared as shown here:

```
Private PIPEDIAM As Double
Private LENGTH As Double
Private ROD_DIA As Double

Private ROUTEPART As String
Private THREADED_LENGTH_TO As Double
```

11. Add the following line to specify the implementation of the IJHgrSymbolWCGServices interface, as shown here:

```
Implements IJHgrSymbolWCGServices
```

12. Go to the IJHgrAssmInfo_GetAssemblyCatalogParts method. Add the following lines to retrieve the fluid code of the primary pipe, as shown here:

```
Dim RouteIndex As Long
RouteIndex = my_IJHgrInputConfigHlpr.RouteIndex
Dim IJElements_PipeColl As IJElements
my_IJHgrInputConfigHlpr.GetPipes IJElements_PipeColl
Dim oRoute As Object
Set oRoute = IJElements_PipeColl.Item(RouteIndex)

Dim strSpecName As String
strSpecName = my_IJHgrInputConfigHlpr.GetRteSpec(oRoute)
```

13. If you did the previous labs, then you can skip this step. Add the following lines to retrieve the pipe diameter and the distance between the reference ports, as shown here:

```
PIPEDIAM = GetExternalPipeDiameter(my_IJHgrInputConfigHlpr, 1)
LENGTH = my_IJHgrInputConfigHlpr.GetDistanceBetweenPorts("Route", "Structure", VERTICAL)
```

14. Add the following lines to query the lookup table as shown below:

```
Dim tempTableName As String
ReDim pParHangerData(0 To 2) As Parameters
Dim Record1 As String
Dim Record2 As String

pParHangerData(0) = FillParameterForDouble("PIPE_OD", CStr(PIPEDIAM - 0.001))
pParHangerData(1) = FillParameterForDouble("PIPE_OD", CStr(PIPEDIAM + 0.001))
pParHangerData(2) = FillParameterForBSTR("SPECNAME", strSpecName)

tempTableName = "JUAHgr" & "RH_HANGER_TBL"
Record2 = ReadParametricData(tempTableName, "where [PIPE_OD]>= ? and [PIPE_OD]<=? and
[SPECNAME]=?")
ROD_DIA = GetNParamUsingParametricQuery(Record2, "ROD_DIA", pParHangerData)
ROUTEPART = GetSParamUsingParametricQuery(Record2, "ROUTE_PART", pParHangerData)
```

15. Add the following lines to get rod diameter in inches as shown below:

```
Dim InchRodDia As String
If (ROD_DIA >= (0.009525 - 0.001) And ROD_DIA <= (0.009525 + 0.001)) Then InchRodDia =
"0.375"
If (ROD_DIA >= (0.0127 - 0.001) And ROD_DIA <= (0.0127 + 0.001)) Then InchRodDia = "0.5"
If (ROD_DIA >= (0.015875 - 0.001) And ROD_DIA <= (0.015875 + 0.001)) Then InchRodDia =
"0.625"
If (ROD_DIA >= (0.01905 - 0.001) And ROD_DIA <= (0.01905 + 0.001)) Then InchRodDia = "0.75"
If (ROD_DIA >= (0.022225 - 0.001) And ROD_DIA <= (0.022225 + 0.001)) Then InchRodDia =
"0.875"
If (ROD_DIA >= (0.0254 - 0.001) And ROD_DIA <= (0.0254 + 0.001)) Then InchRodDia = "1"
```

16. Add the following lines to list the part classes required by this type as shown below:

```
Dim PartClasses(6) As String
PartClasses(1) = ROUTEPART 'ubolt
PartClasses(2) = "LRParts_FIG290N_" & InchRodDia
PartClasses(3) = "RH-ROD_" & InchRodDia
PartClasses(4) = "LRParts_FIG290N_" & InchRodDia  'top eye
PartClasses(5) = "LRParts_FIG66N_" & InchRodDia  'Beam attachment
PartClasses(6) = "LRParts_HEX_NUT_" & InchRodDia  'Nut
```

17. Add the following lines to retrieve the dimensional rod data from the database based on the rod diameter as shown below:

```
'GET ROD DIM
'----------
```

```
    Dim pParaRod(0 To 1) As Parameters
    pParaRod(0) = FillParameterForDouble("ROD_DIA", CStr(ROD_DIA - 0.001))
    pParaRod(1) = FillParameterForDouble("ROD_DIA", CStr(ROD_DIA + 0.001))
    Record1 = ReadParametricData("JUAHgrRHThreadedRod", "where [ROD_DIA]>= ? and
[ROD_DIA]<= ?")
    THREADED_LENGTH_TO = GetNParamUsingParametricQuery(Record1, "D_USER", pParaRod)
```

18. Go to the IJHgrAssmInfo_ GetAssemblyJoints method. Review the following lines to create the joints of the support as shown below:

```
'UBolt or Clamp
        Set AssemblyJoint = JointFactory.MakeRigidJoint(1, "Route", -1, "Route")
        JointCollection.Add AssemblyJoint
'Bottom Eye Nut
        Set AssemblyJoint = JointFactory.MakeRevoluteJoint(2, "Eye", 1, "Pin", 81)
        JointCollection.Add AssemblyJoint
'Rod
        Set AssemblyJoint = JointFactory.MakePrismaticJoint(3, "TopExThdRH", 3, "BotExThdRH",
3501)
        JointCollection.Add AssemblyJoint
        Set AssemblyJoint = JointFactory.MakeRigidJoint(2, "InThdRH", 3, "BotExThdRH", 10468)
        JointCollection.Add AssemblyJoint
        Set AssemblyJoint = JointFactory.MakeVerticalJoint(3, "TopExThdRH")
        JointCollection.Add AssemblyJoint
''Nut
        Set AssemblyJoint = JointFactory.MakeRigidJoint(6, "InThdRH", 3, "TopExThdRH", 9508, -
THREADED_LENGTH_TO)
        JointCollection.Add AssemblyJoint

'Top Eye Nut
        Set AssemblyJoint = JointFactory.MakeRigidJoint(4, "InThdRH", 3, "TopExThdRH", 10468)
        JointCollection.Add AssemblyJoint

        Set AssemblyJoint = JointFactory.MakeRevoluteJoint(4, "Eye", 5, "Pin", 17)
        JointCollection.Add AssemblyJoint

'Beam Attachment
        Set AssemblyJoint = JointFactory.MakePlanarJoint(5, "Structure", -1, "Structure", 9572)
        JointCollection.Add AssemblyJoint
```

19. Go to the IJHgrAssmInfo_ GetMaxRouteConnectionValue method and review the following line, as shown here:

```
  IJHgrAssmInfo_GetMaxRouteConnectionValue = 0
```

20. Go to the IJHgrAssmInfo_ GetRouteConnections method and review the following line, as shown here:

```
 Set IJHgrAssmInfo_GetRouteConnections = ConnectPartToRouteOrStruct(1, 1)
```

21. Go to the IJHgrAssmInfo_ GetStructConnections method, review the following line, as shown here:

```
Set IJHgrAssmInfo_GetStructConnections = ConnectPartToRouteOrStruct(5, 1)
```

22. Add the following lines to calculate the weight of the rod as shown below:

```
Private Sub IJHgrSymbolWCGServices_EvaluateLocalWCG(ByVal pSupportComp As Object, Weight
As Double, CogX As Double, CogY As Double, CogZ As Double)
Const METHOD = "IJHgrSymbolWCGServices_EvaluateLocalWCG"
    On Error GoTo ErrorHandler

    Dim dblLength As Double
    Dim dblWeightperUnitlength As Double

    dblLength = GetAttributeFromObject(pSupportComp, "Length")
    dblWeightperUnitlength = GetAttributeFromObject(pSupportComp, "WEIGHT_PER_LENGTH")

    Weight = dblWeightperUnitlength * dblLength

    CogX = 0
    CogY = 0
    CogZ = -dblLength / 2

    Exit Sub
ErrorHandler:
    Err.Raise LogError(Err, MODULE, METHOD, "").Number
End Sub
```

23. Compile the Visual Basic project and save the dll as HS_RH_ASSY_Hanger2.dll in the HS_RH_ASSY_Hanger2 directory.

24. One of the most important steps in Visual Basic programming is to preserve the binary compatibility of your program. Save the final version of your dll file to be binary compatibility in order to preserve the CLSID.

25. Save and Exit the HS_RH_ASSY_Hanger2.vbp project.

26. Copy the C:\train\HgrTrainingFiles\AssyTemplateV2\SPAssySupportTraining.xls workbook into HS_RH_ASSY_Hanger2 directory.

27. Rename SPAssySupportTraining.xls to HS_RH_ASSY_Hanger2.xls

28. Open the HS_RH_ASSY_Hanger2.xls workbook.

29. Go the R-ClassNodeDescribes sheet and add the following row:

Head	RelationSource	RelationDestination
Start		
a	CLASSNODE_ASSEMBLY_PIPE_SPASSEMBLY_COMPANY	HS_RH_ASSY_Hanger2
End		

30. Go to the class definition and edit the row as shown here:

Definition	PartClassType	SymbolIcon		UserClassName	UserClassName
a	PipeSupportDefinitionClass			HS_RH_ASSY_Hanger2	HS_RH_ASSY_Hanger2

31. Now, you will add the assembly part definition as shown below. Key-in the appropriate part number, part description, discipline type, command type and the type selection rule.

Head	PartNumber	PartDescription	DisciplineType	CommandType	TypeSelectionRule	AssmInfoRule
Start						
a	Assy_RH_Hanger_02	Assy_RH_Hanger_02	1	1	1	HS_RH_ASSY_Hanger2.CGENParts
End						

MaxInsulation	LoadRangeLow	LoadRangeHigh	MinAssemblyLength	MaxAssemblyLength	Supporting_Count	Supported_Count	NDFrom	NDTo	NDUnitType	NominalWidthFrom	NominalWidthTo	NominalDepthFrom	NominalDepthTo	SupportedFamily	FaceSelectionType	InsulationPurpose	SupportType	MirrorBehaviorOption
-1	0	0	0in	1000in	1	1	2	6	in	0	0	0	0	STRAIGHT	0	2	11	15

32. Save and Exit the HS_RH_ASSY_Hanger2.xls workbook.

33. Open the SPSupportTraining.xls workbook located in c:\train\HS_RH_Parts.

34. Create a new sheet named RH_HANGER_TBL and add the following rows:

Definition	PartClassType	UserClassName					
a	HgrServiceClass	RH Hangers Auxiliary Table					
Head	Name	IJUAHgrRH_HANGER_TBL::PIPE_DIAM	IJUAHgrRH_HANGER_TBL::DimensionUnitType	IJUAHgrRH_HANGER_TBL::PIPE_OD	IJUAHgrRH_HANGER_TBL::ROD_DIA	IJUAHgrRH_HANGER_TBL::ROUTE_PART	IJUAHgrRH_HANGER_TBL::SPECNAME
Start							
a	RH_HANGER_TBL_1	2	in	2.375	0.375	RH_SPCLAMP	1C0031
a	RH_HANGER_TBL_2	4	in	4.5	0.5	RH_SPUBOLT	1C0031
a	RH_HANGER_TBL_3	6	in	6.625	0.625	RH_SPUBOLT	1C0031
a	RH_HANGER_TBL_4	2	in	2.375	0.375	RH_SPUBOLT	2C0032
a	RH_HANGER_TBL_5	4	in	4.5	0.5	RH_SPCLAMP	2C0032
a	RH_HANGER_TBL_6	6	in	6.625	0.625	RH_SPCLAMP	2C0032
End							

Note:

- In version 2009, the support assembly is not updated if the 'piping specification' of the pipe feature it is attached to, changes.

Next step, you will add a new custom interfaces for the RH_HANGER_TBL service class. Use the Custom Interfaces sheet to define the attributes names with associated data type and unit type.

35. Create a new interface called IJUAHgrRH_HANGER_TBL. Go to the Custom Interface sheet and add the following rows:

InterfaceName	CategoryName	AttributeName	AttributeUserName	Type	UnitsType	PrimaryUnits	OnPropertyPage	ReadOnly	SymbolParameter
IJUAHgrRH_HANGER_TBL	Standard	DimensionUnitType	DimensionUnitType	Char	0	0	1	1	DimensionUnitType
		PIPE_DIAM	PIPE_DIAM	Double	distance	in	1	1	PIPE_DIAM
		PIPE_OD	PIPE_OD	Double	distance	in	1	1	PIPE_OD
		ROD_DIA	Rod Diameter	Double	distance	in	1	1	ROD_DIA
		SPECNAME	SPECNAME	Char	0	0	1	1	SPECNAME
		ROUTE_PART	ROUTE_PART	Char	0	0	1	1	ROUTE_PART

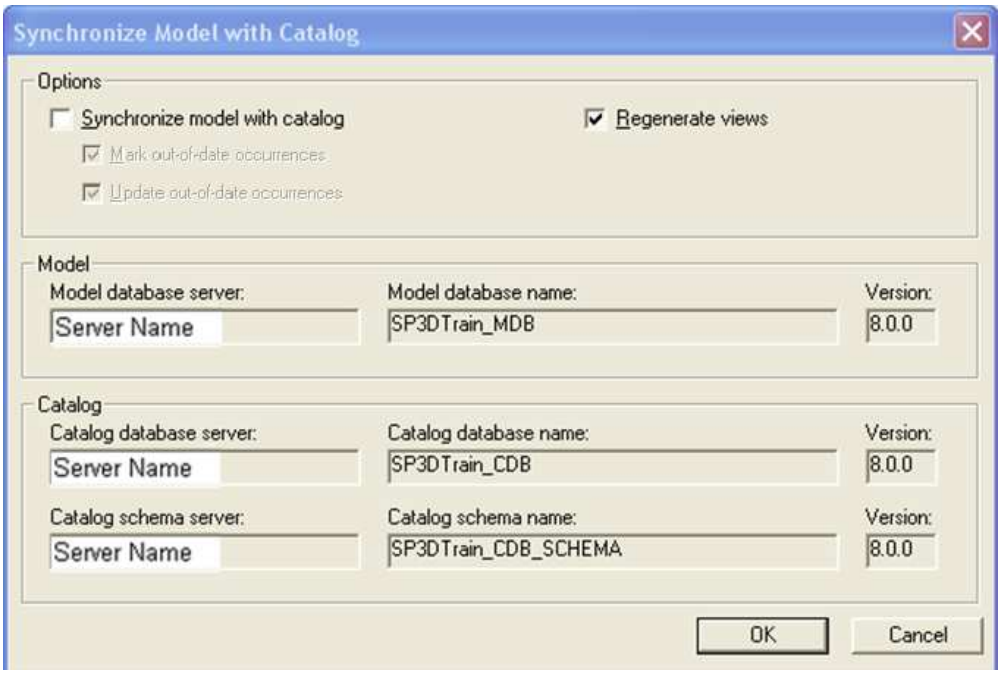
36. Save and Exit the SPSupportTraining.xls workbook.

37. Run the Project Management Task. Select the Model in the hierarchy.

38. Select Tools -> Synchronize Model with the Catalog.

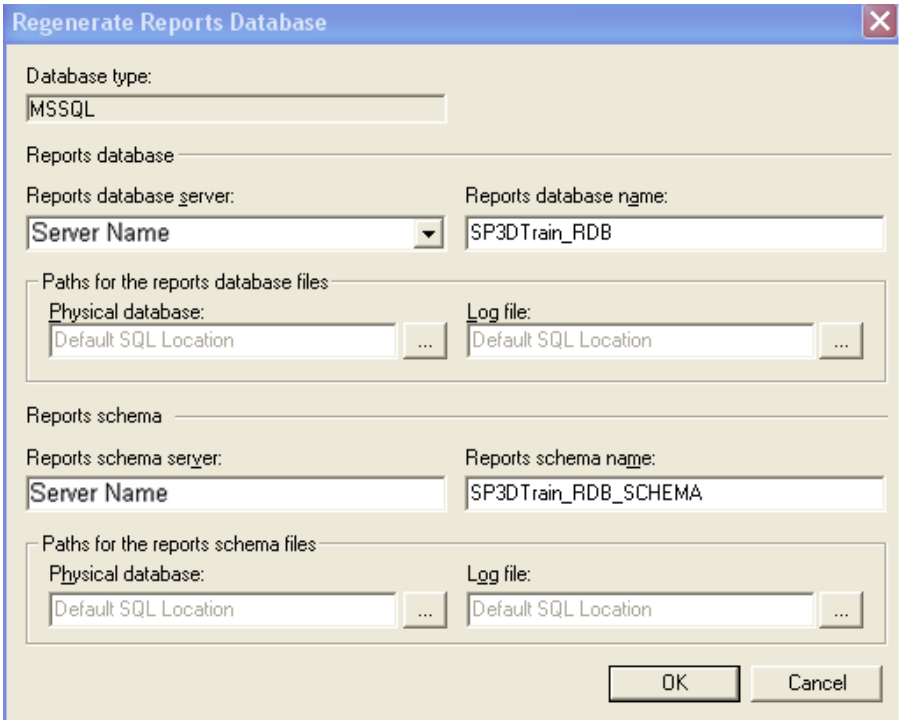
39. Uncheck the Synchronize Model with the Catalog option.

***Note**: You just need to update the views in the model.*

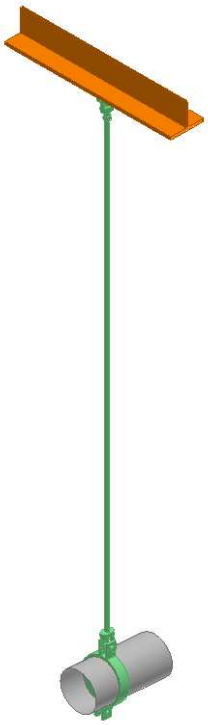


40. Hit “OK” Button.

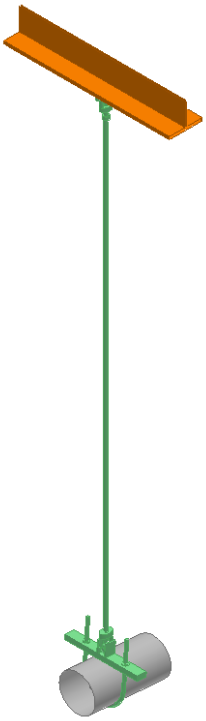
41. Once the process is completed, Right click the training plant icon and select “Regenerate the reports database” option to re-create the views in the report database.



42. Hit OK to start the process.
43. Go to the Hangers and Supports Task and place the HS_RH_ASSY_Hanger2 Assembly support using the Place Support by Structure Command.



2c0032 Pipe Specification



1c0031 Pipe Specification

Lab 6: Weld Component Symbol

Objectives

After completing this lab, you will be able to:

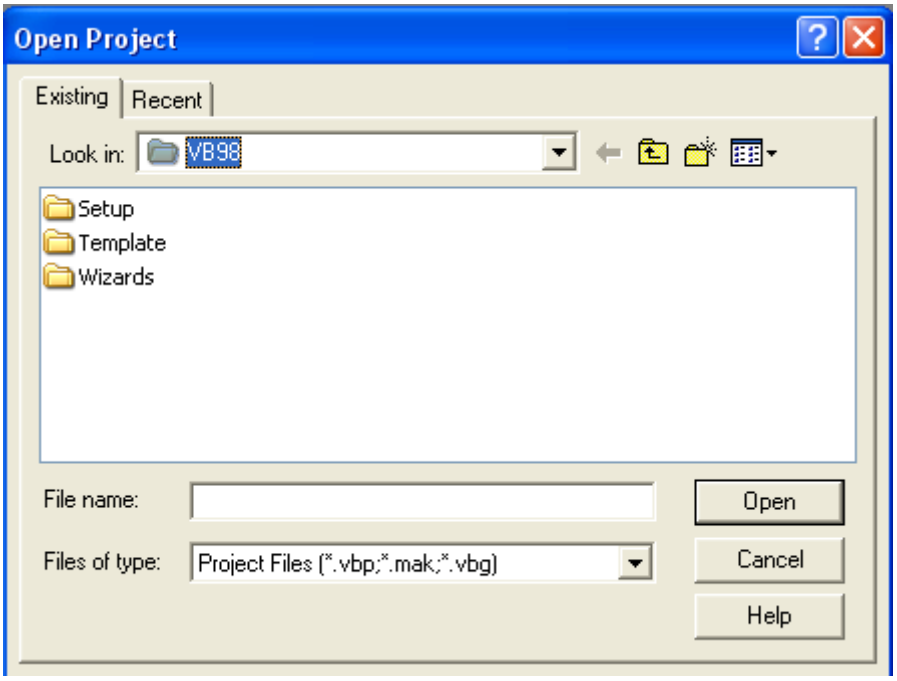
- Create a simple standard support component symbol
- Use the symbol service to create the symbol definition
- Use the AddPort method to create a port
- Use Helper Functions to create simple geometric shapes
- Implement the IJHgrSymbolBOMServices interface

In this lab, you will create a standard support component symbol to be used as a Weld item on the support assembly. You can use the HgrTemplatePart template provided by the instructor to create the symbol program. This symbol consists of a 1/4mm cube shape.

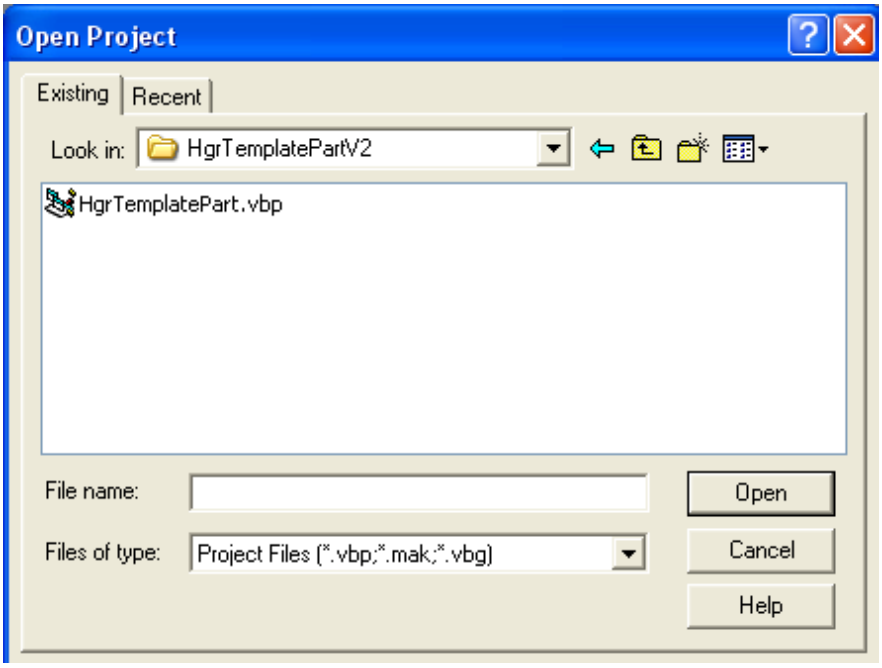
1. Create a directory called WeldBox directory under c:\train\HgrTrainingFiles and run Microsoft Visual Basic 6.0

2. Close the Microsoft New Project dialog box.

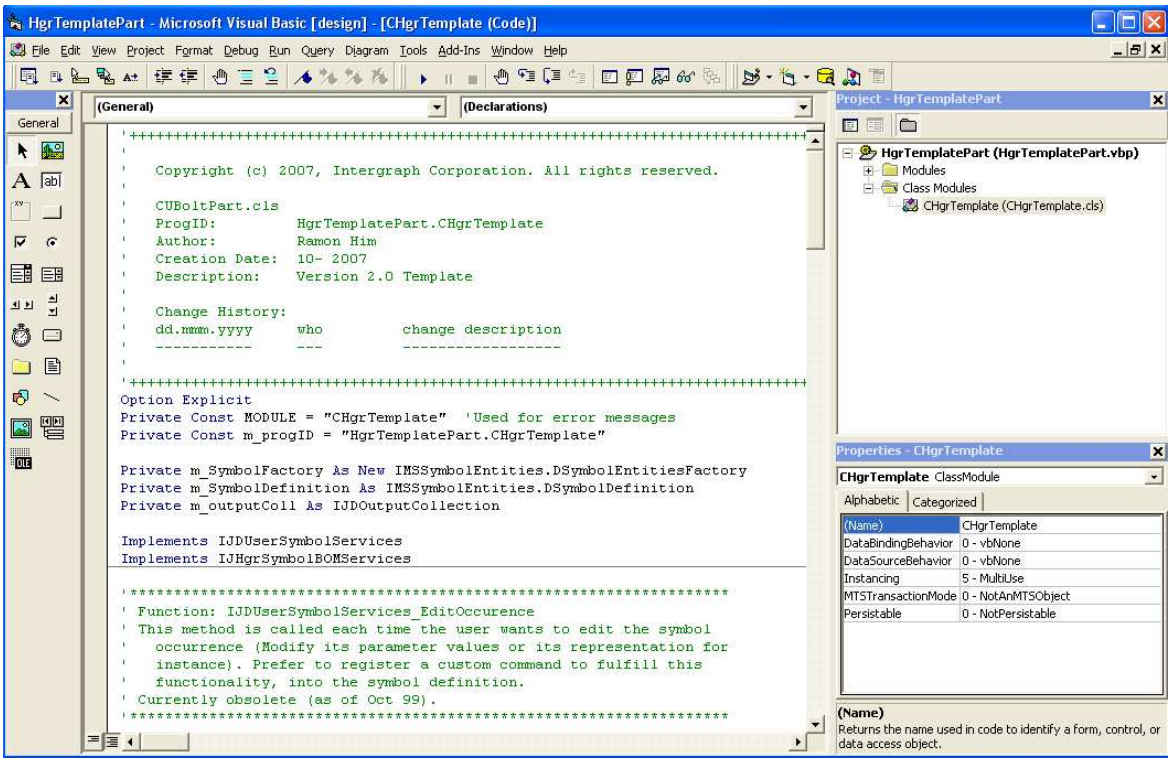
3. Select *File -> Open Project* option to open the Open Project Dialog box.

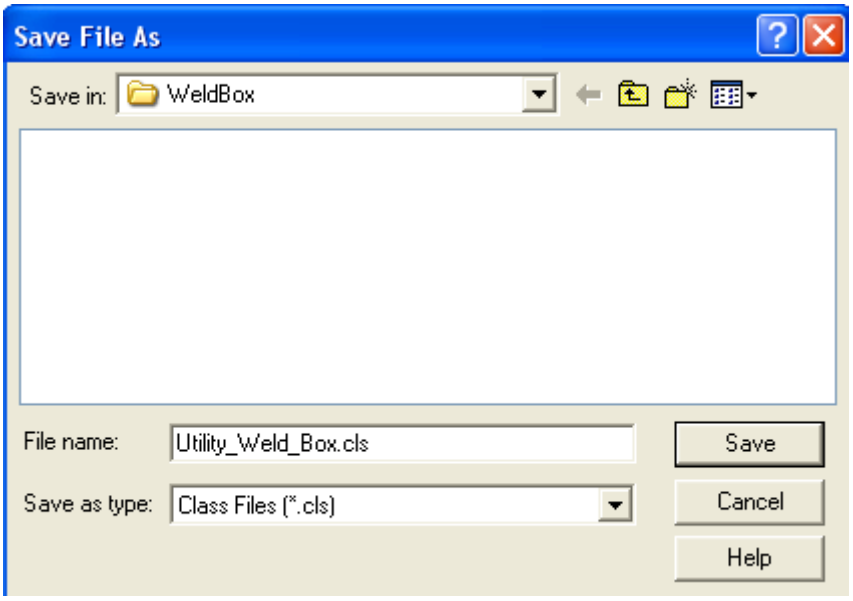


4. Navigate to C:\train\HgrTrainingFiles\HgrTemplatePartV2 and open the HgrTemplatePart Visual Basic project.

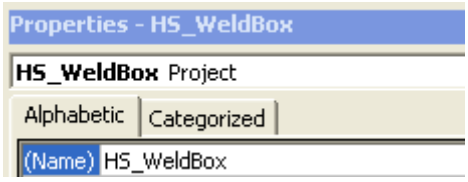


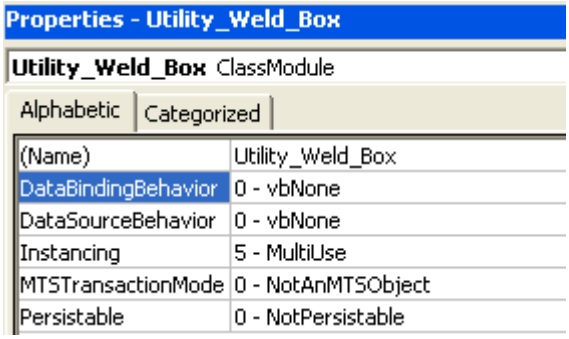
5. Setup the Visual Basic Development Environment as shown below:


HgrTemplatePart - Microsoft Visual Basic [design] - [CHgrTemplate (Code)]
File Edit View Project Format Debug Run Query Diagram Tools Add-Ins Window Help
(General)
(Declarations)
'++++++



8. Go to the Properties Window and change the name of the Project and Class Module, as shown here::





9. Go to the General Declarations section in CUBoltPart module and change the value of the *Constant Module variable* from *"CHgrTemplatePart:"* to *"Utility_Weld_Box:"*

   Private Const MODULE = "Utility_Weld_Boxt" 'Used for error messages

10. Rename the ProgId (project name and class name) as shown below:

    Private Const m_progID = "HS_WeldBox.Utility_Weld_Box"

11. Go to the IJDUserSymbolServices_InitializeSymbolDefinition method. Review and edit the input and output definitions as shown below:

```
'Inputs
Const iNumInputs As Integer = 2
Dim sInputName(iNumInputs) As String
Dim sDataType(iNumInputs) As String

'outputs
Const iNumOutputs As Integer = 2
Dim sOutputName(iNumOutputs) As String

'Inputs
sInputName(2) = "Thickness"
sDataType(2) = "Double"

'Outputs
sOutputName(1) = "Port1"
sOutputName(2) = "BODY"

SetupInputsAndOutputsForCache pSymbolDefinition, iNumInputs, _
      iNumOutputs, sInputName(), sDataType(), sOutputName(), m_progID
```

12. Go to Physical()procedure and add the following instructions to create the outputs:

```
Dim Thickness As Double
Thickness = arrayOfInputs(2)

AddPort "Other", 0, 0, 0, _
        1, 0, 0, _
        0, 0, 1, m_outputColl, "Port1", arrayOfInputs(1)

AddBox Thickness, Thickness, Thickness, Loc(-Thickness / 2, -Thickness / 2, 0) + _
        " * ROTZ(270)", m_outputColl, "BODY"
```

13. Go to the IJHgrSymbolBOMServices_EvaluateBOM method. Review and edit the input and output definitions as shown below:

```
Private Sub IJHgrSymbolBOMServices_EvaluateBOM(ByVal pSupportComp As Object,
bstrBOMDesc As String)
Const METHOD = "IJHgrSymbolBOMServices_EvaluateBOM"
   On Error GoTo ErrorHandler

   ' <Determine BOM description here>
   Dim oPartOcc As PartOcc
   Dim oPart As IJDPart
   Set oPartOcc = pSupportComp ' The part occurence
   oPartOcc.GetPart oPart      ' The associated catalog part

   bstrBOMDesc = "1/4"

   Exit Sub
ErrorHandler:
      Err.Raise LogError(Err, MODULE, METHOD, "").Number
End Sub
```

14. Compile the Visual Basic project and save the dll as HS_WeldBox.dll in the c:\Train\ WeldBox.

15. One of the most important steps in Visual Basic programming is to preserve the binary compatibility of your program. Save the final version of your dll file to be binary compatibility in order to preserve the CLSID.

16. Save the HS_ WeldBox project.

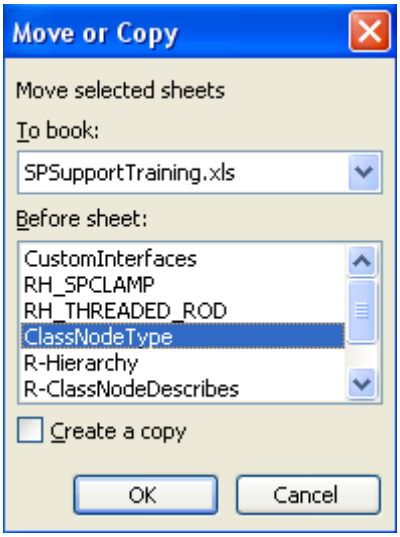
17. Open the SPSupportTraining.xls workbook located in *c:\train* HS_RH_Parts

18. Go the R-ClassNodeDescribes sheet and add the following row:

Head	*RelationSource*	*RelationDestination*
Start		
!	**Support Part Classes**	
	CLASSNODE_PART_SPCOMPANY_PARTS	RH_SPCLAMP
	CLASSNODE_PART_SPCOMPANY_PARTS	RH_THREADED_ROD
	CLASSNODE_PART_SPCOMPANY_PARTS	RH_SPUBOLT
a	CLASSNODE_PART_SPCOMPANY_PARTS	UTILITY_WELD
End		

19. Next step, you will create a new part class by copying the RH_SPCLAMP sheet:

20. Select the RH_SPCLAMP Sheet. Right Click to open the Move or Copy dialog box. Enable the copy option. Select the OK button to copy the sheet next to the ClassNodeType sheet.



21. Rename the sheet as UTILITY_WELD.

CustomInterfaces / RH_SPCLAMP \ **UTILITY_WELD** / RH_THREADED_ROD /

22. Go to the class definition and edit the row as shown here:

PartClassType StandardSupportComponentClass
SymbolDefinition: HS_WeldBox.Utility_Weld_Box
SymbolIcon
PartSelectionRule
ClassType 1
UserClassName Utility Weld
OccClassname Utility Weld
HgrSymbolPort(1):Type HgrSymbolPort
oa:IJUAHgrThickness::Thickness

Definition	PartClassType	SymbolDefinition	SymbolIcon	PartSelectionRule	ClassType	UserClassName	OccClassName	HgrSymbolPort(1):Type	oa:IJUAHgrThickness::Thickness
a	StandardSupportComponentClass	HS_WeldBox.Utility_Weld_Box			1	Utility Weld	Utility Weld	HgrSymbolPort	

23. Now, you will need to add the Thickness attribute in the part definition.

IJUAHgrThickness::Thickness

Head	PartNumber	PartDescription	SymbolDefinition	MirrorBehaviorOption	NDFrom	NDTo	NDUnitType	HgrSymbolPort(1):Name	HgrSymbolPort(1):Category	HgrSymbolPort(1):MatingType	HgrSymbolPort(1):ConnectionInfo	HgrSymbolPort(1):PortType	HgrSymbolPort(1):Size	HgrSymbolPort(1):MinSize	HgrSymbolPort(1):MaxSize	HgrSymbolPort(1):UnitType	DryWeight	WaterWeight	DryCogX	DryCogY	DryCogZ	IJHgrBOMDefinition::BOMType	IJHgrBOMDefinition::BOMDefinition	IJUAHgrThickness::Thickness

24. Now, you will add one part for the Utility Weld class as shown below. Key-in the appropriate part number, part description and mirror behavior option (codelist).

Head	PartNumber	PartDescription	SymbolDefinition	MirrorBehaviorOption	NDFrom	NDTo	NDUnitType
Start							
a	Utility_Weld_1	Utility_Weld_Box_1		15			

25. Key-in the appropriate port data, BOM type and dimension for the weld part.

Note:

- Valid codes can be listed in the **AllCodeLists.xls** workbook on the **HngSupBOMType** sheet.

HgrSymbolPort(1):Name	HgrSymbolPort(1):Category	HgrSymbolPort(1):MatingType	HgrSymbolPort(1):ConnectionInfo	HgrSymbolPort(1):PortType	HgrSymbolPort(1):Size	HgrSymbolPort(1):MinSize	HgrSymbolPort(1):MaxSize	HgrSymbolPort(1):UnitType	DryWeight	WaterWeight	DryCogX	DryCogY	DryCogZ	IJHgrBOMDefinition::BOMType	IJHgrBOMDefinition::BOMDefinition	IJUAHgrThickness::Thickness
Other				15					0	0	0	0	0	3		0.25mm

Check if the interface IJUAHgrThickness exists in the custom interface sheet. If the interface does not exist in the sheet, then add the following rows:

InterfaceName	CategoryName	AttributeName	AttributeUserName	Type	UnitsType	PrimaryUnits	CodeList	codelisttablenamespace	OnPropertyPage	ReadOnly	SymbolParameter
IJUAHgrThickness	Dimensions	Thickness	Thickness	Double	distance	in			1	0	Thickness

26. Save and Exit the SPSupportTraining.xls workbook.

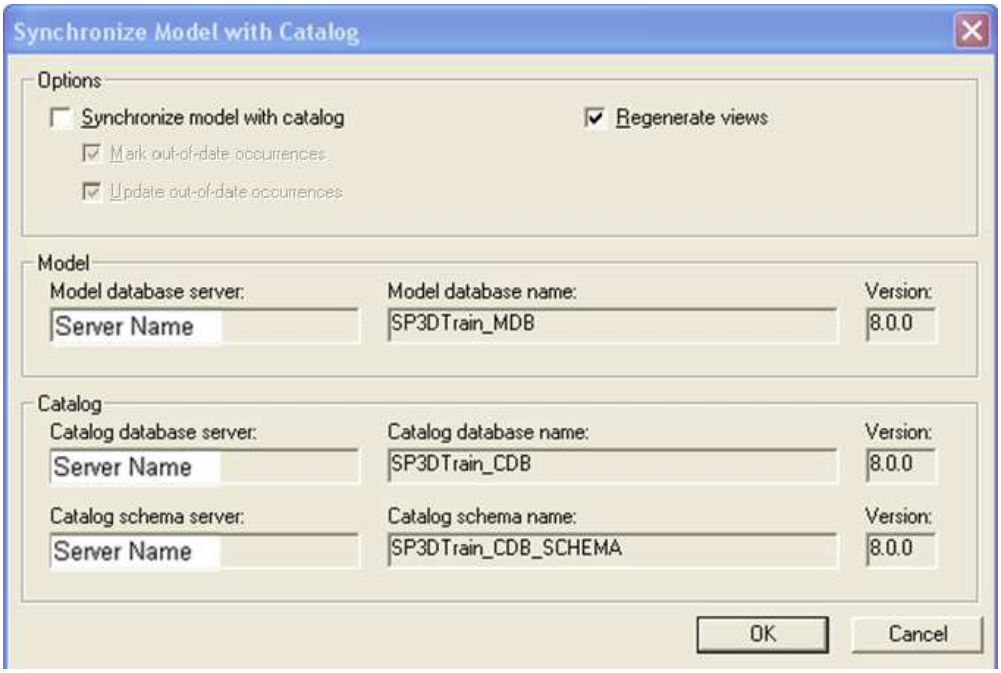
27. Load the information into the catalog using the A/M/D Mode. Once the bulkload process is completed, review the log file.

28. Run the Project Management Task. Select the Model in the hierarchy.

29. Select Tools -> Synchronize Model with the Catalog.

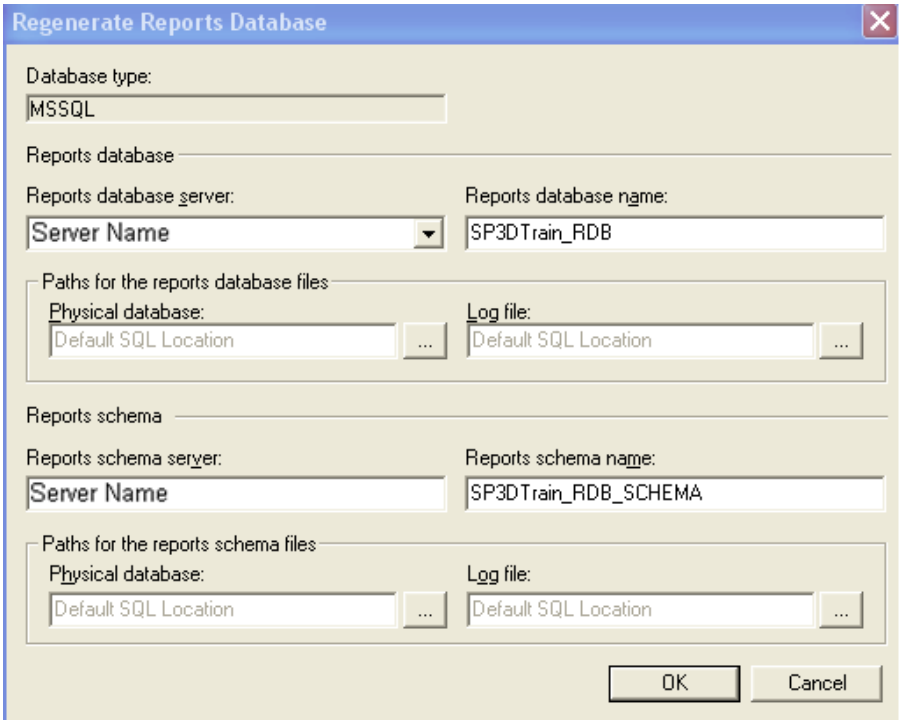
30. Uncheck the Synchronize Model with the Catalog option.

***Note**: You just need to update the views in the model.*



31. Hit “OK” Button.

32. Once the process is completed, Right mouse click the training plant icon and select “Regenerate the reports database” option to re-create the views in the report database.



33. Hit OK to start the process.

34. Once the process is completed, go to the Hangers and Supports Task and place the Utility Weld part using the Place Part Command.

35. Undo the placement of the weld part.

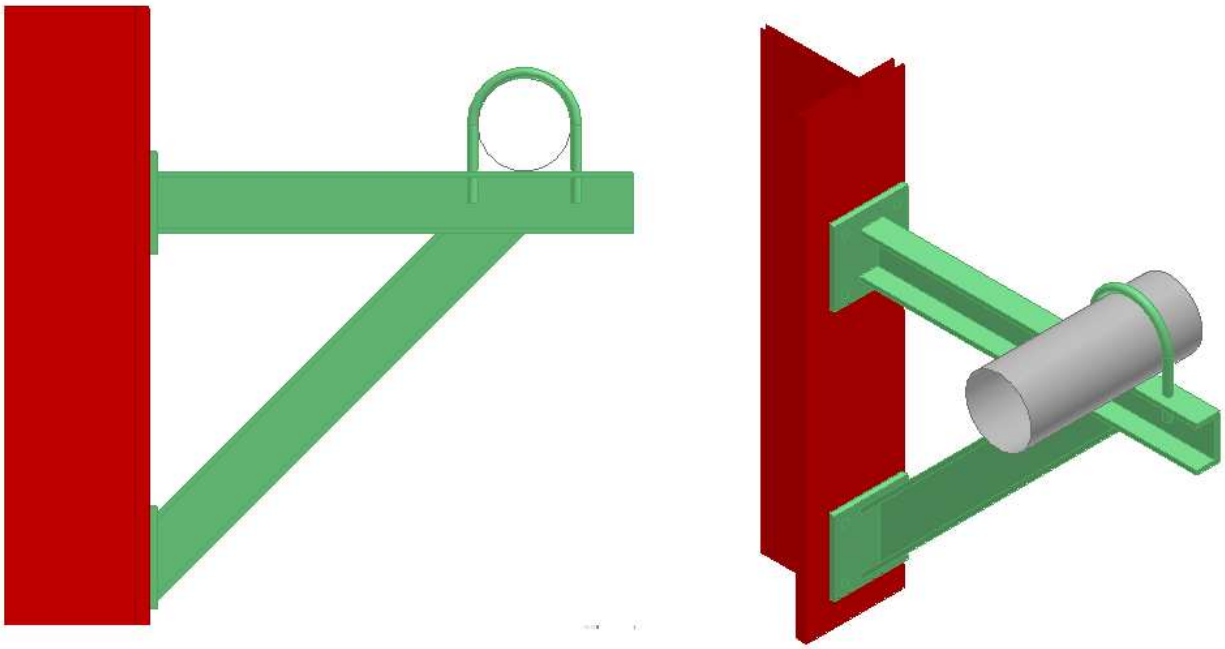
Lab 7: Assembly Information Rule for a Cantilever U-Bolt Assembly

Objectives

After completing this lab, you will be able to:

- Use the methods provided by IJHgrAssmInfo
- Place note objects on support keypoints for auto dimensioning
- Place Utility_Weld parts and set them non-reportable
- Define joint data for this assembly

In this lab, you will create the assembly information rule to build a cantilever u-bolt support as shown below. You start by using the Assy_TemplateAsm template provided by the instructor to create the rule.

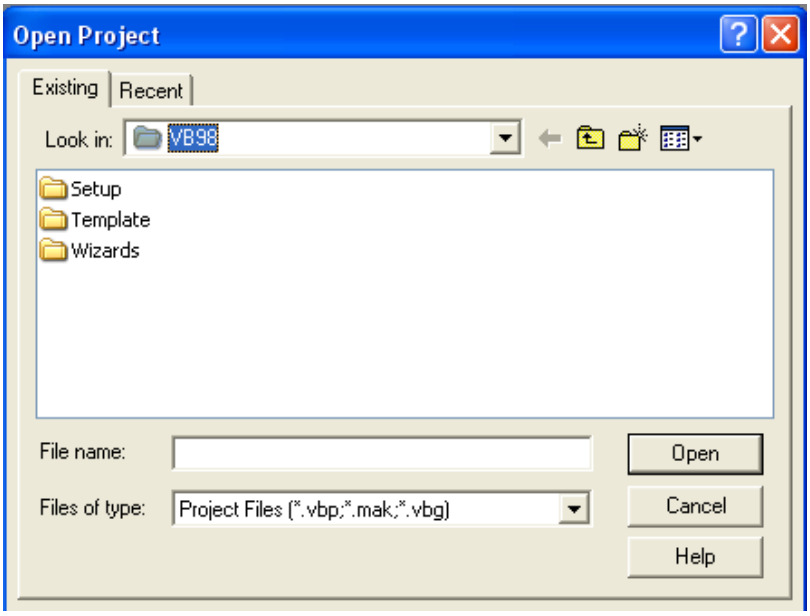


Elevation View Isometric View

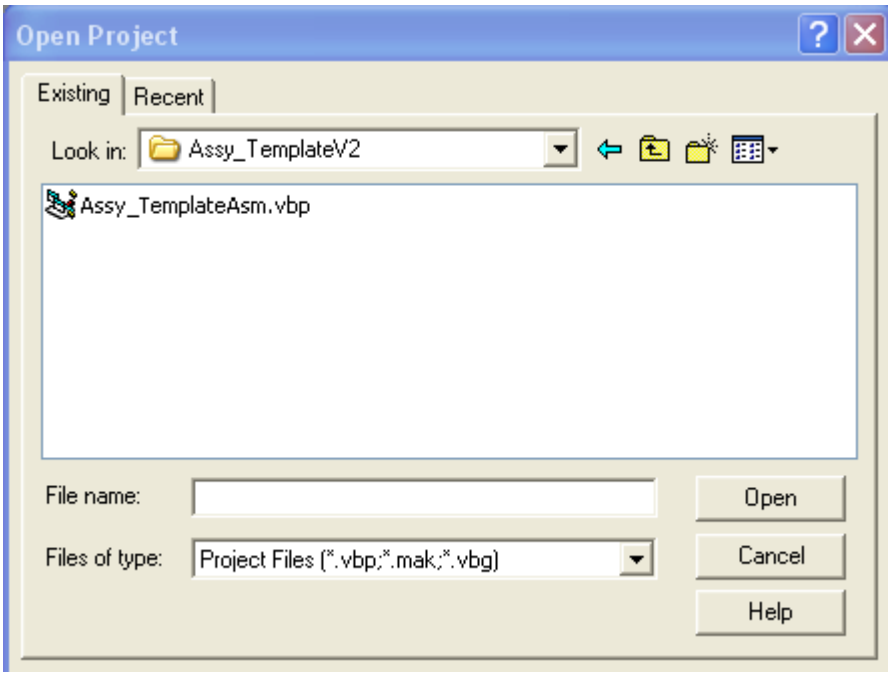
1. Create a directory called HS_RH_ASSY_BC under c:\train\HgrTrainingFiles and run Microsoft Visual Basic 6.0

2. Close the Microsoft New Project dialog box.

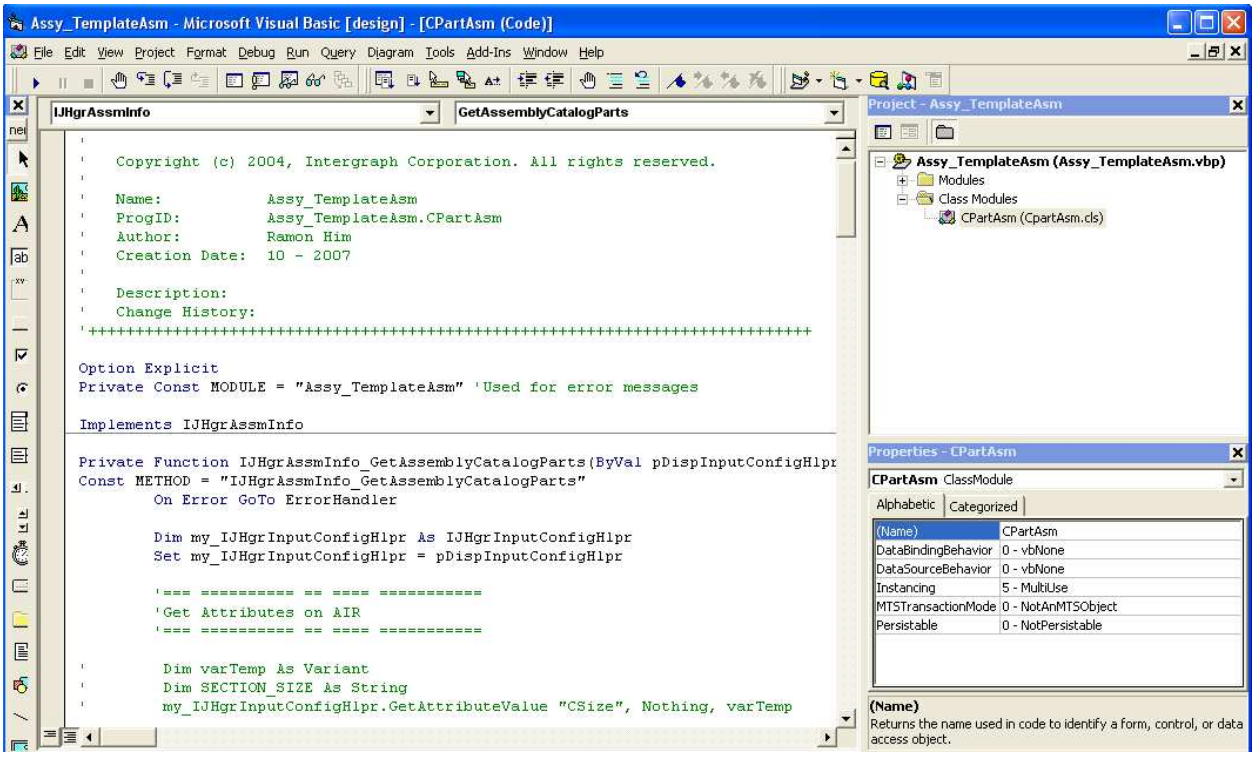
3. Select *File -> Open Project* option to open the Open Project Dialog box.



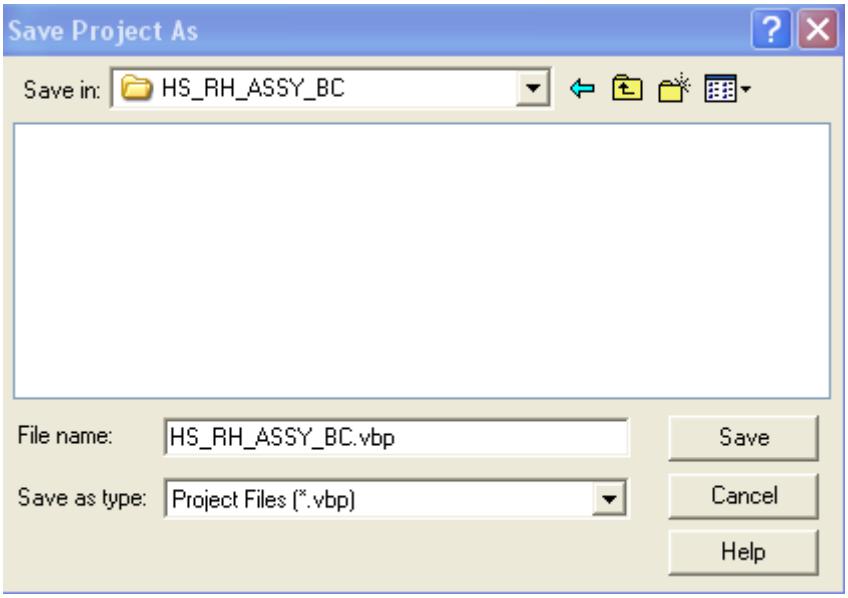
4. Navigate to c:\train\HgrTrainingFiles\ AssyTemplateV2 and open the Assy_TemplateAsm.vbp Template project.



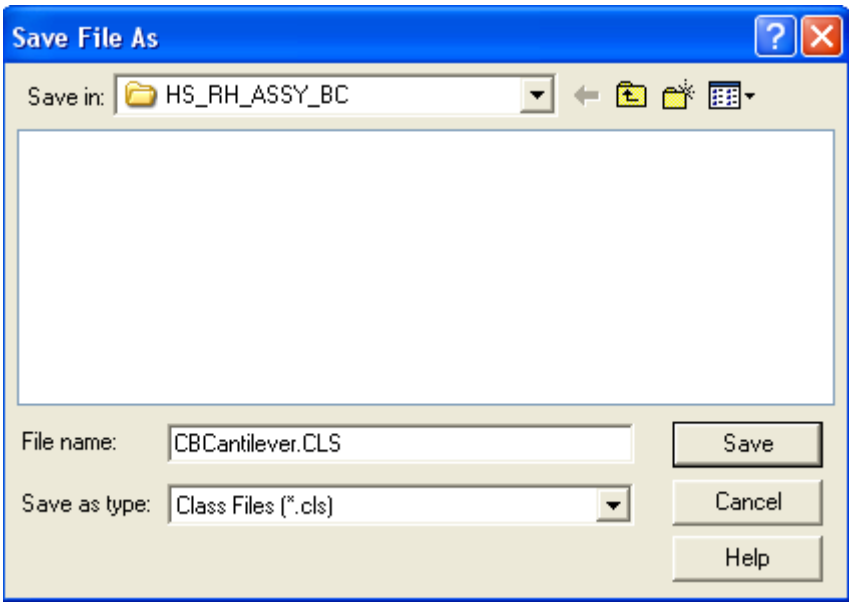
5. Setup the Visual Basic Development Environment as shown below:



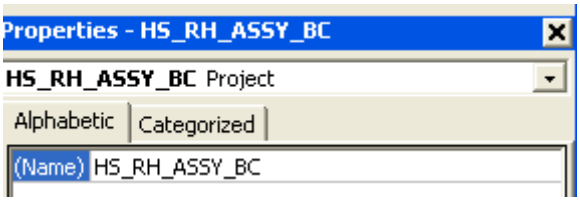
6. Go to the Explorer Window and select the Project file node. Select *File* -> *"Save Project As"* option to save the project as HS_RH_ASSY_BC.vbp under the HS_RH_ASSY_BC directory.

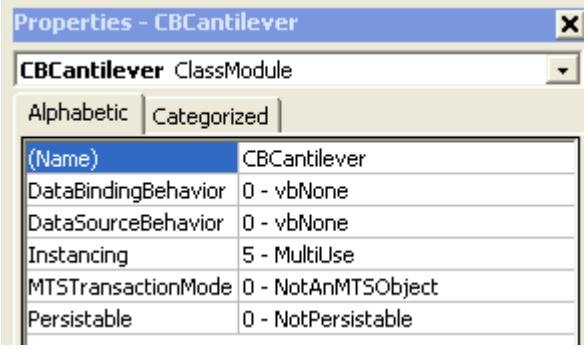


7. Go to the Explorer Window and select the CPartAsm class file node. Select *File -> Save* CPartAsm*.cls As* option to save the class module as CBCantilever.cls under HS_RH_ASSY_BC directory.



8. Go to the Properties Window and change the name of the Project and Class Modules as shown here:





9. Go to the General Declarations section in CBCantilever module and change the value of the *Constant Module variable* from *"Assy_TemplateAsm"* to *"CBCantilever"*

```
Private Const MODULE = "CBCantilever" 'Used for error messages
```

10. Add global variables as shown below:

```
Private Const nSection = 1
Private Const nBrace = 2
Private Const nConnObj1 = 3
Private Const TOP_PLATE = 4
Private Const ANGLE_PLATE = 5
Private Const WeldBox1 = 6
Private Const WeldBox2 = 7
Private Const WeldBox3 = 8
Private Const Ubolt = 9

Private OVERHANG As Double
Private BRACE_ANGLE As Double
Private SECTION_SIZE As String
Private SECTIONTYPE As String
Private SECTION_SIZE1 As String

Private BP_WIDTH As Double
Private BASEPLATE As String
Private BP_HOLE_SIZE As Double
Private BP_HOLE_INSET As Double
Private BP_THICKNESS As Double
```

11. Go to the IJHgrAssmInfo_GetAssemblyCatalogParts method. Add the following lines to retrieve dimensional data from the assembly part definition as shown below:

```
  Dim varTemp As Variant
    my_IJHgrInputConfigHlpr.GetAttributeValue "CSize", Nothing, varTemp
    SECTION_SIZE = Trim(varTemp)
    SECTION_SIZE1 = Mid(SECTION_SIZE, 9)
    my_IJHgrInputConfigHlpr.GetAttributeValue "OVERHANG", Nothing, varTemp
    OVERHANG = varTemp
    my_IJHgrInputConfigHlpr.GetAttributeValue "BRACE_ANGLE", Nothing, varTemp
    BRACE_ANGLE = varTemp

'GET BASE PLATE Dimensions from AIR
'=====================================
    my_IJHgrInputConfigHlpr.GetAttributeValue "BP_HOLE_SIZE", Nothing, varTemp
```

```
        BP_HOLE_SIZE = varTemp
        my_IJHgrInputConfigHlpr.GetAttributeValue "BP_HOLE_INSET", Nothing, varTemp
        BP_HOLE_INSET = varTemp
        my_IJHgrInputConfigHlpr.GetAttributeValue "BP_WIDTH", Nothing, varTemp
        BP_WIDTH = varTemp
        my_IJHgrInputConfigHlpr.GetAttributeValue "BP_THICKNESS", Nothing, varTemp
        BP_THICKNESS = varTemp
    '=====================================
```

12. Add the following lines to specify the part class constituting the assembly as shown below:

```
        ReDim PartClasses(9) As String
        PartClasses(nSection) = Trim(SECTION_SIZE) 'C Section
        PartClasses(nBrace) = "Utility_CUTBACK_C2_1" 'C Section Utility_CUTBACK_C2_1
        PartClasses(nConnObj1) = "Log_Conn_Part_1" 'Rotational Connection Object
        PartClasses(TOP_PLATE) = "Utility_FOUR_HOLE_PLATE_5"
        PartClasses(ANGLE_PLATE) = "Utility_FOUR_HOLE_PLATE_5"
        PartClasses(WeldBox1) = "Utility_Weld_1"
        PartClasses(WeldBox2) = "Utility_Weld_1"
        PartClasses(WeldBox3) = "Utility_Weld_1"
        PartClasses(Ubolt) = "Utility_GEN_U_BOLT_1"
```

13. Go to the IJHgrAssmInfo_ GetAssemblyJoints method. Add the following lines to retrieve the diameter of the primary pipe.

```
       'Get Structure object
        Dim oStruct As Object
        Dim oStructColl As Object
        my_IJHgrInputConfigHlpr.GetSupportingCollection oStructColl
        Set oStruct = oStructColl.Item(1)

        Dim dPipeDia As Double
        dPipeDia = GetExternalPipeDiameter(my_IJHgrInputConfigHlpr, 1)

        Dim dPipeRadius As Double
        dPipeRadius = GetPipeODwithoutInsulat(dPipeDia, oRoute) / 2

        Dim dInsulat As Double
        dInsulat = GetPipeInsulat(oRoute)
```

14. Add the following lines to set the note on the support keypoints.

```
'<< Get interface for accessing items on the collection of Part Occurences

         Dim IJElements_PartOccCollection As IJElements
         Set IJElements_PartOccCollection = pDispPartOccCollection

         'Resume normal error handling.
         On Error GoTo ErrorHandler:

         '=====================================================
         'Get the Current Location in the Route Connection Cycle
         '=====================================================
         Dim RouteConnectionValue As Integer
         RouteConnectionValue = my_IJHgrInputConfigHlpr.GetRouteConnectionValue

'Set note for dimension on Support PartOcc
```

```
Dim oNote1 As IJGeneralNote
Set oNote1 = my_IJHgrInputConfigHlpr.CreateNoteAndAssociateToPort("Dim1", _
        IJElements_PartOccCollection.Item(1), "BeginCap")
oNote1.Text = "1 Dimension"
oNote1.Purpose = 3 ' value 3 means fabrication
oNote1.Dimensioned = True

Set oNote1 = my_IJHgrInputConfigHlpr.CreateNoteAndAssociateToPort("Dim2",  _
         IJElements_PartOccCollection.Item(1), "EndCap")
oNote1.Text = "2 Dimension"
oNote1.Purpose = 3 ' value 3 means fabrication
oNote1.Dimensioned = True

Set oNote1 = my_IJHgrInputConfigHlpr.CreateNoteAndAssociateToPort("Dim3", _
     IJElements_PartOccCollection.Item(2), "StartStructure")
oNote1.Text = "Third Dimension"
oNote1.Purpose = 3 ' value 3 means fabrication
oNote1.Dimensioned = True

Set oNote1 = my_IJHgrInputConfigHlpr.CreateNoteAndAssociateToPort("Dim4",  _
      IJElements_PartOccCollection.Item(5), "TopStructure")
oNote1.Text = "4 Dimension"
oNote1.Purpose = 3 ' value 3 means fabrication
oNote1.Dimensioned = True

Set oNote1 = my_IJHgrInputConfigHlpr.CreateNoteAndAssociateToPort("Dim5",  _
      IJElements_PartOccCollection.Item(5), "BotStructure")
oNote1.Text = "5 Dimension"
oNote1.Purpose = 3 ' value 3 means fabrication
oNote1.Dimensioned = True

Set oNote1 = my_IJHgrInputConfigHlpr.CreateNoteAndAssociateToPort("Dim6",  _
         IJElements_PartOccCollection.Item(9), "Route")
oNote1.Text = "6 Dimension"
oNote1.Purpose = 3 ' value 3 means fabrication
oNote1.Dimensioned = True

Set oNote1 = Nothing
```

15. Add the following lines to retrieve the beam dimension.

```
'Get dimension from HgBeam

Dim SteelWidth As Double
Dim SteelDepth As Double
Dim SteelThickness As Double
SteelThickness = GetProfileDim(nSection, "Flange",
IJElements_PartOccCollection) 'Width, Depth, Flange, Web as 2nd argument
SteelWidth = GetProfileDim(nSection, "Width", IJElements_PartOccCollection)
'Width, Depth, Flange, or Web as 2nd argument
SteelDepth = GetProfileDim(nSection, "Depth", IJElements_PartOccCollection)
'Width, Depth, Flange, or Web as 2nd argument
```

16. Add the following lines to get the distance between the reference ports.

```
'Get the distance between the structure and route ports
    Dim dDistPorts As Double
    Dim tempL As Double

    dDistPorts = my_IJHgrInputConfigHlpr.GetDistanceBetweenPorts("Route",
"Structure", HORIZONTAL)
```

```
    tempL = dDistPorts
```

17. Add the following lines to get the current configuration based on the route connection toggle.

```
    Dim iStructConfIndex As Integer
    Dim BraceConfigIndex As Double

'Set the confIndex for the toggle route connection
        If RouteConnectionValue = 1 Then
                iStructConfIndex = 10404
                BraceConfigIndex = 9574
        Else
                iStructConfIndex = 2212
                BraceConfigIndex = 9574
        End If
```

18. Add the following lines to set the part occurrence dimensions.

```
'Set input to part symbols
'HgrSection
    my_IJHgrInputConfigHlpr.SetSymbolInputByName
IJElements_PartOccCollection.Item(nBrace), "WIDTH", SteelWidth '
    my_IJHgrInputConfigHlpr.SetSymbolInputByName
IJElements_PartOccCollection.Item(nBrace), "DEPTH", SteelDepth '
    my_IJHgrInputConfigHlpr.SetSymbolInputByName
IJElements_PartOccCollection.Item(nBrace), "T_FLANGE", SteelThickness
    my_IJHgrInputConfigHlpr.SetSymbolInputByName
IJElements_PartOccCollection.Item(nBrace), "T_WEB", SteelThickness

'Plates
    my_IJHgrInputConfigHlpr.SetSymbolInputByName
IJElements_PartOccCollection.Item(TOP_PLATE), "WIDTH", BP_WIDTH ' 5 inches
    my_IJHgrInputConfigHlpr.SetSymbolInputByName
IJElements_PartOccCollection.Item(TOP_PLATE), "DEPTH", BP_WIDTH ' 5 inches
    my_IJHgrInputConfigHlpr.SetSymbolInputByName
IJElements_PartOccCollection.Item(TOP_PLATE), "C", BP_HOLE_INSET ' 3/4 inches
    my_IJHgrInputConfigHlpr.SetSymbolInputByName
IJElements_PartOccCollection.Item(TOP_PLATE), "HOLE_SIZE", BP_HOLE_SIZE ' 9/16
inches bolt holes
    my_IJHgrInputConfigHlpr.SetSymbolInputByName
IJElements_PartOccCollection.Item(TOP_PLATE), "THICKNESS", BP_THICKNESS

    my_IJHgrInputConfigHlpr.SetSymbolInputByName
IJElements_PartOccCollection.Item(ANGLE_PLATE), "WIDTH", BP_WIDTH
    my_IJHgrInputConfigHlpr.SetSymbolInputByName
IJElements_PartOccCollection.Item(ANGLE_PLATE), "DEPTH", BP_WIDTH
    my_IJHgrInputConfigHlpr.SetSymbolInputByName
IJElements_PartOccCollection.Item(ANGLE_PLATE), "C", BP_HOLE_INSET
    my_IJHgrInputConfigHlpr.SetSymbolInputByName
IJElements_PartOccCollection.Item(ANGLE_PLATE), "HOLE_SIZE", BP_HOLE_SIZE
    my_IJHgrInputConfigHlpr.SetSymbolInputByName
IJElements_PartOccCollection.Item(ANGLE_PLATE), "THICKNESS", BP_THICKNESS

'Ubolt
    my_IJHgrInputConfigHlpr.SetSymbolInputByName
IJElements_PartOccCollection.Item(Ubolt), "D", 0.0127 '0.5 inches - diameter ubolt
    my_IJHgrInputConfigHlpr.SetSymbolInputByName
IJElements_PartOccCollection.Item(Ubolt), "R", dPipeDia / 2 + 0.0127 / 2 'bend
radius
```

```
        my_IJHgrInputConfigHlpr.SetSymbolInputByName
    IJElements_PartOccCollection.Item(Ubolt), "H", dPipeDia / 2 + SteelDepth / 2

    'Weld
        my_IJHgrInputConfigHlpr.SetSymbolInputByName
    IJElements_PartOccCollection.Item(6), "Thickness", 0.00025

        my_IJHgrInputConfigHlpr.SetSymbolInputByName
    IJElements_PartOccCollection.Item(7), "Thickness", 0.00025

        my_IJHgrInputConfigHlpr.SetSymbolInputByName
    IJElements_PartOccCollection.Item(8), "Thickness", 0.00025

Dim iArrayOfKeys(2) As Integer
iArrayOfKeys(1) = nSection

SetValuesOfSymbolInputsAndOccAttributes my_IJHgrInputConfigHlpr,
IJElements_PartOccCollection, iArrayOfKeys, "EndOverLength", 0
SetValuesOfSymbolInputsAndOccAttributes my_IJHgrInputConfigHlpr,
IJElements_PartOccCollection, iArrayOfKeys, "BeginOverLength", 0
SetValuesOfSymbolInputsAndOccAttributes my_IJHgrInputConfigHlpr,
IJElements_PartOccCollection, iArrayOfKeys, "Orientation", 0
SetValuesOfSymbolInputsAndOccAttributes my_IJHgrInputConfigHlpr,
IJElements_PartOccCollection, iArrayOfKeys, "BeginMiter", "No End Treatment"
SetValuesOfSymbolInputsAndOccAttributes my_IJHgrInputConfigHlpr,
IJElements_PartOccCollection, iArrayOfKeys, "EndMiter", "No End Treatment"
```

19. Add the following lines to define the joints required to assembly the support.

```
'-------------
'Create joints
'-------------
'Create a collection to hold the joints
Dim JointCollection As New Collection

'Create a Joint Factory
Dim JointFactory As New HgrJointFactory
'Create an object to hold the Joint
Dim AssemblyJoint As Object

'TOP PLATE

If RouteConnectionValue = 1 Then
    Set AssemblyJoint = JointFactory.MakeRigidJoint(-1, "Structure", TOP_PLATE,
"TopStructure", 9380, BP_THICKNESS, 0, -SteelDepth / 2 - dPipeRadius)
Else
    Set AssemblyJoint = JointFactory.MakeRigidJoint(-1, "Structure", TOP_PLATE,
"TopStructure", 9380, BP_THICKNESS, 0, SteelDepth / 2 + dPipeRadius)
End If
JointCollection.Add AssemblyJoint

'Angle PLATE

If RouteConnectionValue = 1 Then
    Set AssemblyJoint = JointFactory.MakeRigidJoint(nBrace, "StartStructure",
ANGLE_PLATE, "TopStructure", 3252, -((SteelDepth) / Sin(PF_HgrGeneralUtil.Rad(45))) /
2, 0, 0)
Else
    Set AssemblyJoint = JointFactory.MakeRigidJoint(nBrace, "StartStructure",
ANGLE_PLATE, "TopStructure", 3252, -((SteelDepth) / Sin(PF_HgrGeneralUtil.Rad(45))) /
2, 0, 0)
End If
```

```
JointCollection.Add AssemblyJoint

'Weld

If RouteConnectionValue = 1 Then
   Set AssemblyJoint = JointFactory.MakeRigidJoint(nConnObj1, "Connection", 6,
"Other", 9444, -0.00025, 0, 0)
Else
   Set AssemblyJoint = JointFactory.MakeRigidJoint(nConnObj1, "Connection", 6,
"Other", 9444, -0.00025, 0, 0)
End If
JointCollection.Add AssemblyJoint

If RouteConnectionValue = 1 Then
   Set AssemblyJoint = JointFactory.MakeRigidJoint(nSection, "EndCap", 7, "Other",
9444, -0.00025, 0, 0)
Else
   Set AssemblyJoint = JointFactory.MakeRigidJoint(nSection, "EndCap", 7, "Other",
9444, -0.00025, 0, 0)
End If
JointCollection.Add AssemblyJoint

If RouteConnectionValue = 1 Then
   Set AssemblyJoint = JointFactory.MakeRigidJoint(nBrace, "StartStructure", 8,
"Other", 9444, 0, 0, 0)
Else
   Set AssemblyJoint = JointFactory.MakeRigidJoint(nBrace, "StartStructure", 8,
"Other", 9444, 0, 0, 0)
End If
JointCollection.Add AssemblyJoint

'Horizontal brace length

my_IJHgrInputConfigHlpr.SetSymbolInputByName
IJElements_PartOccCollection.Item(nSection), "Length", tempL - BP_THICKNESS +
dPipeRadius + OVERHANG
If RouteConnectionValue = 1 Then
   Set AssemblyJoint = JointFactory.MakeRigidJoint(nSection, "EndCap", -1,
"Structure", iStructConfIndex, BP_THICKNESS, SteelDepth + dPipeRadius, SteelWidth / 2)
Else
   Set AssemblyJoint = JointFactory.MakeRigidJoint(nSection, "EndCap", -1,
"Structure", iStructConfIndex, BP_THICKNESS, SteelDepth + dPipeRadius, SteelWidth / 2)
End If
JointCollection.Add AssemblyJoint

'Angle Brace length

Dim dBraceL As Double
dBraceL = (tempL - BP_THICKNESS - (SteelDepth / 2) / Sin(PF_HgrGeneralUtil.Rad(45))) /
Cos(PF_HgrGeneralUtil.Rad(45))
my_IJHgrInputConfigHlpr.SetSymbolInputByName
IJElements_PartOccCollection.Item(nBrace), "L", dBraceL

'Add a Joint between horizontal section and the point that we want the angle to rotate
around.
'Move the conn port to the where you want the cutback start

Dim dPlaneOffset As Double
dPlaneOffset = dDistPorts + dPipeRadius - BP_THICKNESS
Set AssemblyJoint = JointFactory.MakeRigidJoint(nConnObj1, "Connection", nSection,
"BeginCap", BraceConfigIndex, -dPipeRadius - OVERHANG, 0)
JointCollection.Add AssemblyJoint
```

```
Set AssemblyJoint = JointFactory.MakeRigidJoint(nConnObj1, "Connection", nBrace,
"EndStructure", 3309, SteelWidth / 2, 0, 0)
JointCollection.Add AssemblyJoint

'Ubolt

If RouteConnectionValue = 1 Then
   Set AssemblyJoint = JointFactory.MakeRigidJoint(-1, "Route", Ubolt, "Route", 2533,
0, 0)
Else
   Set AssemblyJoint = JointFactory.MakeRigidJoint(-1, "Route", Ubolt, "Route", 10661,
0, 0)
End If
JointCollection.Add AssemblyJoint
```

20. Add the following lines to set the BOM of the Beam.

```
'Set the BOM for the cutback member

    Dim strBraceL As String
    Dim oUOMService As IJUomVBInterface
    Set oUOMService = New UnitsOfMeasureServicesLib.UomVBInterface
    oUOMService.SetDefaultUnits UNIT_DISTANCE, 65, 64
    oUOMService.FormatUnit UNIT_DISTANCE, dBraceL, strBraceL

     my_IJHgrInputConfigHlpr.SetSymbolInputByName
IJElements_PartOccCollection.Item(nBrace), "BOM_DESC",SECTION_SIZE1 & " , Length: " &
strBraceL & " , Angle: " & Deg(BRACE_ANGLE)
    Set oUOMService = Nothing

    my_IJHgrInputConfigHlpr.SetSymbolInputByName
IJElements_PartOccCollection.Item(Ubolt), "BOM_DESC1", "Light Weight U-Bolt with two
hex nuts, Galvanized Finish."
```

21. Add the following lines to set the weld to non-reportable.

```
'Set all Weld parts to non-mto

    Dim Attrcol As IJDAttributesCol
    Dim oAttrs As IJDAttributes
    Dim oAttr As IJDAttribute
    Dim iWeldBoxIndx As Integer

    For iWeldBoxIndx = 6 To 8
        Set oAttrs = IJElements_PartOccCollection.Item(iWeldBoxIndx)
        Set Attrcol = oAttrs.CollectionOfAttributes("IJMTOInfo")
        Set oAttr = Attrcol.Item("ReportingRequirements")
        oAttr.Value = 2
        Set oAttr = Attrcol.Item("ReportingType")
        oAttr.Value = 2
    Next
    Set oAttr = Nothing
    Set oAttrs = Nothing
    Set Attrcol = Nothing

    Set IJElements_PartOccCollection = Nothing
    Set oRoute = Nothing
```

22. Go to the IJHgrAssmInfo_GetMaxRouteConnectionValue method and add the following line as shown below:

```
IJHgrAssmInfo_GetMaxRouteConnectionValue = 2
```

23. Go to the IJHgrAssmInfo_ GetRouteConnections method and add the following line as shown below:

```
Set IJHgrAssmInfo_GetRouteConnections = ConnectPartToRouteOrStruct(nSection, 1)
```

24. Go to the IJHgrAssmInfo_ GetStructConnections() function and add the following code as shown below:

```
Set IJHgrAssmInfo_GetStructConnections = ConnectPartToRouteOrStruct(nSection, 1)
```

25. Compile the Visual Basic project and save the dll as HS_RH_ASSY_BC.dll in the HS_RH_ASSY_BC directory.

26. One of the most important steps in Visual Basic programming is to preserve the binary compatibility of your program. Save the final version of your dll file to be binary compatibility in order to preserve the CLSID.

27. Save and Exit the HS_RH_ASSY_BC.vbp project.

28. Copy the C:\train\HgrTrainingFiles\AssyTemplateV2\SPAssySupportTraining.xls workbook into HS_RH_ASSY_BC directory.

29. Rename SPAssySupportTraining.xls to HS_RH_ASSY_BC..xls

30. Open the HS_RH_ASSY_BC.xls workbook.

31. Go the R-ClassNodeDescribes sheet and add the following row.

Head	RelationSource	RelationDestination
Start		
a	CLASSNODE_ASSEMBLY_PIPE_SPASSEMBLY_COMPANY	HS_RH_ASSY_BC
End		

32. Go to the HrgTemplateAssy sheet and rename it as HS_RH_ASSY_BC.

CustomInterfaces | HS_RH_ASSY_BC

33. Go to the class definition and edit the row as shown here:

Definition	PartClassType	SymbolIcon	UserClassName	OccClassName	oa:IJUAHgrCSize::CSize	oa:IJUAHgrAssyRHBC::BRACE_ANGLE	oa:IJUAHgrAssyRHBC::OVERHANG	oa:IJUAHgrAssyRHBC::BP_WIDTH	oa:IJUAHgrAssyRHBC::BP_HOLE_SIZE	oa:IJUAHgrAssyRHBC::BP_HOLE_INSET	oa:IJUAHgrAssyRHBC::BP_THICKNESS
a	PipeSupportDefinitionClass		RH Cantilever	RH Cantilever							

34. Now, you will need to add custom attributes in the part definition.

Head	PartNumber	PartDescription	DisciplineType	CommandType	TypeSelectionRule	AssmInfoRule	MaxInsulation	LoadRangeLow	LoadRangeHigh	MinAssemblyLength	MaxAssemblyLength	Supporting_Count	Supported_Count	NDFrom	NDTo	NDUnitType

NominalWidthFrom	NominalWidthTo	NominalDepthFrom	NominalDepthTo	SupportedFamily	FaceSelectionType	InsulationPurpose	SupportType	MirrorBehaviorOption	IJUAHgrCSize::CSize	IJUAHgrAssyRHBC::BRACE_ANGLE	IJUAHgrAssyRHBC::OVERHANG	IJUAHgrAssyRHBC::BP_WIDTH	IJUAHgrAssyRHBC::BP_HOLE_SIZE	IJUAHgrAssyRHBC::BP_HOLE_INSET	IJUAHgrAssyRHBC::BP_THICKNESS

35. Now, you will need to add the assembly part definition as shown below:

Head	PartNumber	PartDescription	DisciplineType	CommandType	TypeSelectionRule	AssmInfoRule
Start						
a	Assy_RH_ASSY_BC_01	Assy_RH_ASSY_BC_01	1	1	4	HS_RH_ASSY_BC.CBCantilever
End						

36. Continue add the appropriate data for the other attributes:

MaxInsulation	LoadRangeLow	LoadRangeHigh	MinAssemblyLength	MaxAssemblyLength	Supporting_Count	Supported_Count	NDFrom	NDTo	NDUnitType	NominalWidthFrom	NominalWidthTo	NominalDepthFrom	NominalDepthTo	SupportedFamily	FaceSelectionType	InsulationPurpose	SupportType	MirrorBehaviorOption
-1	0	0	24in	60in	1	1+	2	8	in	0	0	0	0	STRAIGHT	0	2	74	15

37. Key-in the appropriate Section name, overhang and plate dimensions.

IJUAHgrCSize::CSize	IJUAHgrAssyRHBC::BRACE_ANGLE	IJUAHgrAssyRHBC::OVERHANG	IJUAHgrAssyRHBC::BP_WIDTH	IJUAHgrAssyRHBC::BP_HOLE_SIZE	IJUAHgrAssyRHBC::BP_HOLE_INSET	IJUAHgrAssyRHBC::BP_THICKNESS
HsSteel_C3x6	45	3	5	0.5625in	0.8	0.3750

Next step, you will add a new custom interfaces for the HS_RH_ASSY_BC class. Use the Custom Interfaces sheet to define the attributes names with associated data type and unit type.

38. Create a new interface called IJUAHgrAssyRHBC. Go to the Custom Interface sheet and add the following entries:

InterfaceName	CategoryName	AttributeName	AttributeUserName	Type	UnitsType	PrimaryUnits	OnPropertyPage	ReadOnly	SymbolParameter
IJUAHgrAssyRHBC	Dimensions	OVERHANG	Overhang	Double	distance	in	0	1	OVERHANG
		BP_THICKNESS	Baseplate Thickness	Double	distance	in	0	1	BP_THICKNESS
		BP_WIDTH	Baseplate Width	Double	distance	in	0	1	BP_WIDTH
		BP_HOLE_SIZE	Baseplate Hole Size	Double	distance	in	0	1	BP_HOLE_SIZE
		BP_HOLE_INSET	Baseplate Hole Inset	Double	distance	in	0	1	BP_HOLE_INSET
		BRACE_ANGLE	Brace Angle	Double	angle	deg	0	1	BRACE_ANGLE

39. Save the HS_RH_ASSY_BC.xls workbook.

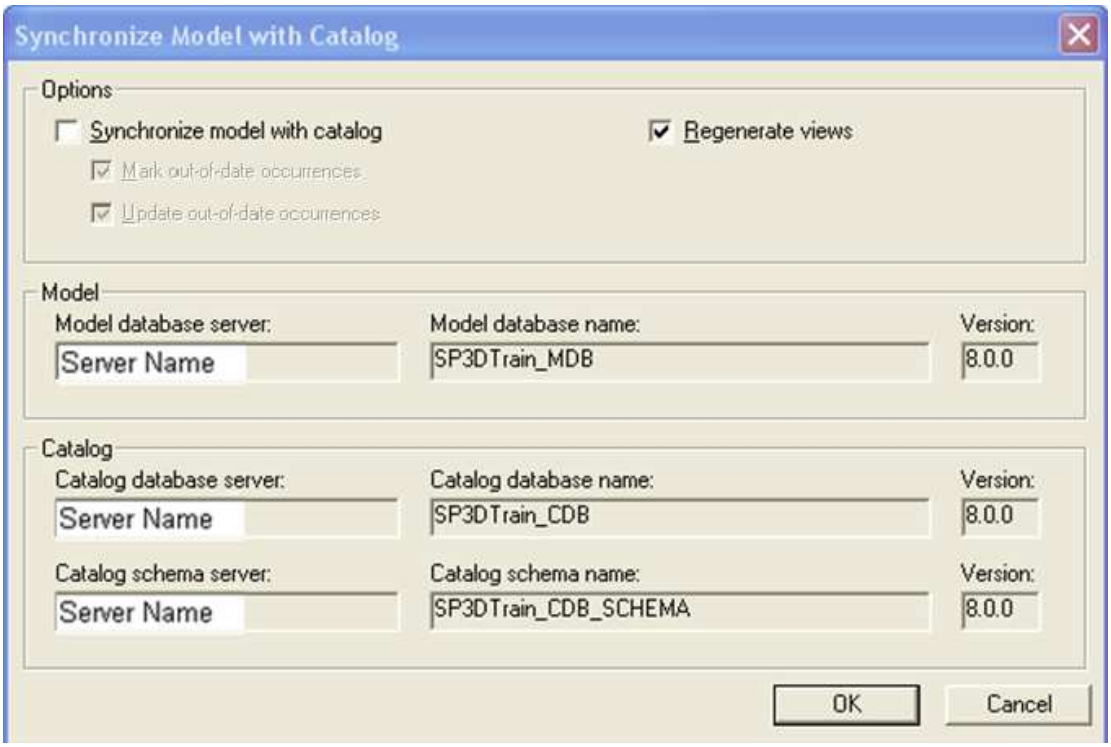
40. Load the information into the Catalog using the A/M/D Mode. Once the bulkload process is completed, review the log file.

41. Run the Project Management Task. Select the Model in the hierarchy.

42. Select Tools -> Synchronize Model with the Catalog.

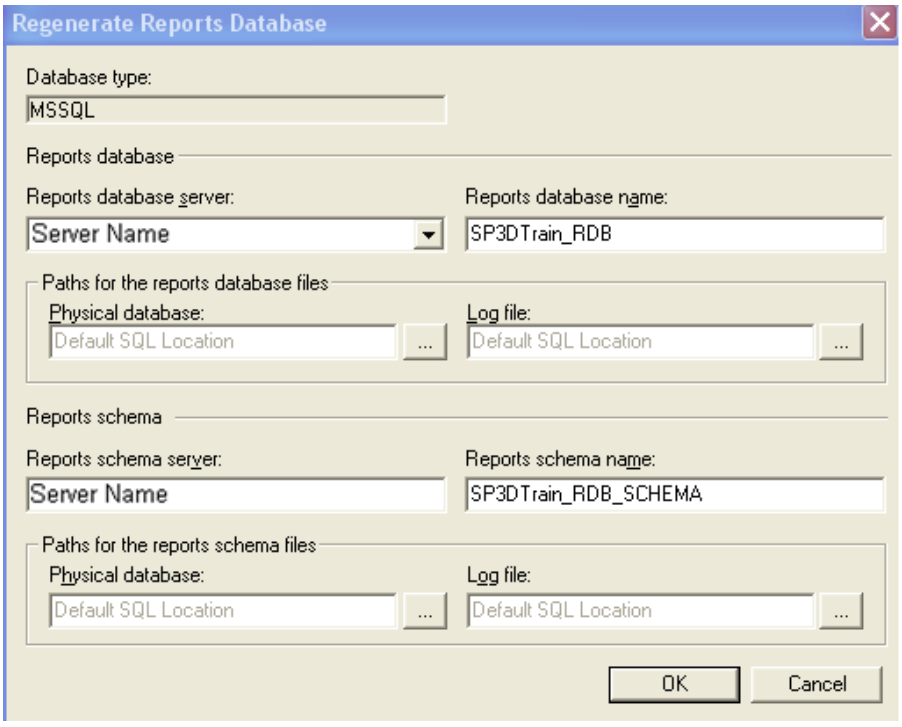
43. Uncheck the Synchronize Model with the Catalog option.

***Note**: You just need to update the views in the model.*



44. Hit “OK” Button.

45. Once the process is completed, Right click the training plant icon and select “Regenerate the reports database” option to re-create the views in the report database.



46. Hit OK to start the process.

47. Once the process is completed, go to the Hangers and Supports Task and place the Assy_RH_ASSY_BC_01 support using the Place Support by Structure Command.

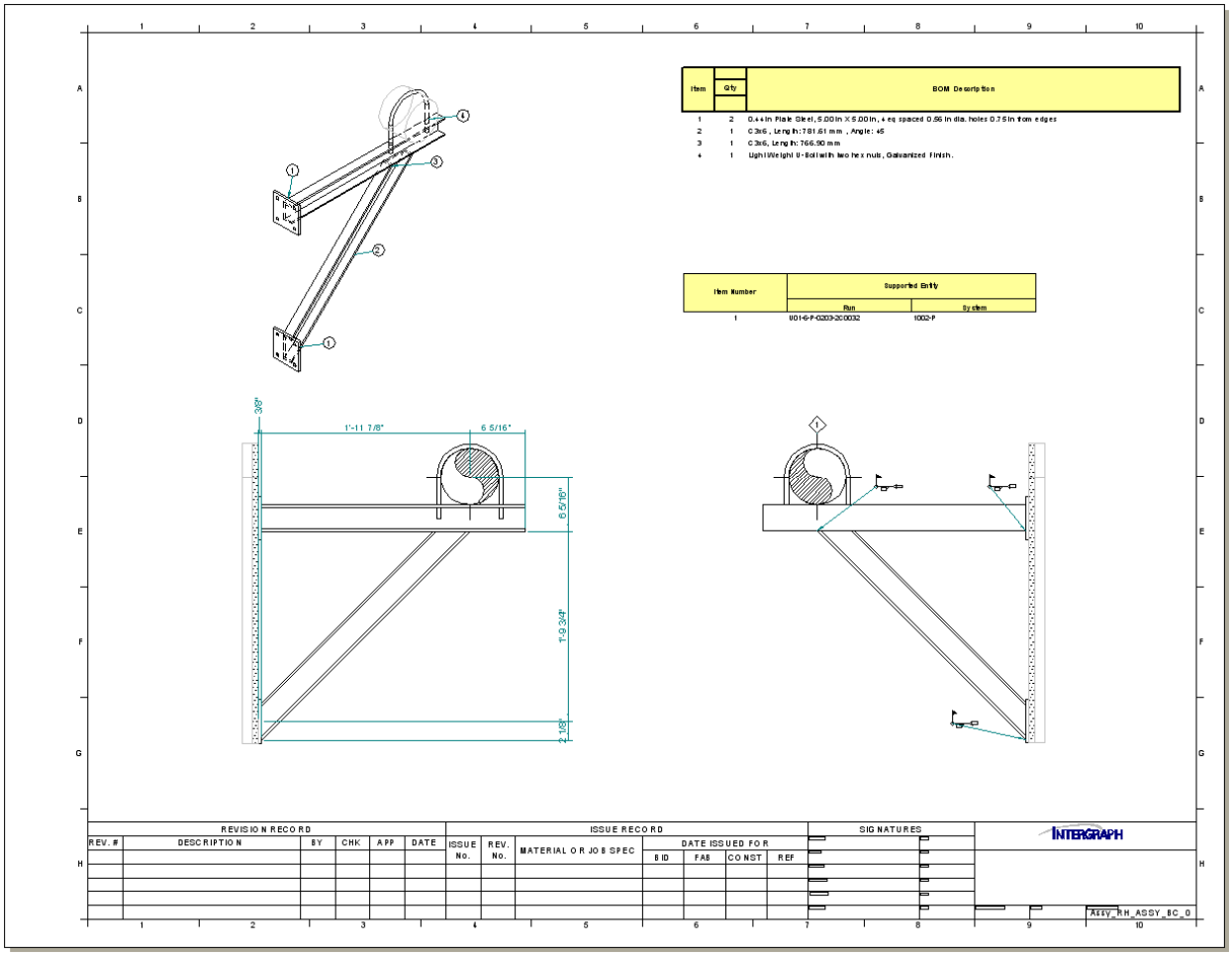
Lab 8: Support Fabrication Drawings (Optional)

Objectives

After completing this lab, you will be able to:

- Create catalog filters
- Understand the schema of the drawing label rule
- Understand the schema of the drawing dimension rule
- Create drawing view styles
- Create custom orthographic drawing package
- Use drawing by query manager Component to create support drawings

In this lab, you will setup and create all the necessary items to generate a support drawing for the cantilever u-bolt support, as shown here:

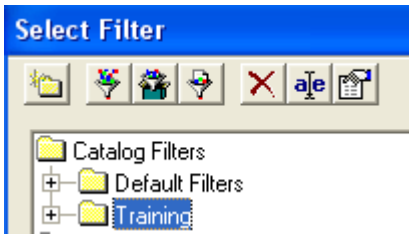


1. First step, create two catalog filters to query reportable and non-reportable pipe support components

2. Go to Common Task and select Tools-> Select by Filter option to open the Select Filter dialog box.

3. In the Select Filter dialog box, select Catalog Filters and Click New Folder icon.

4. Key-in Training for the new folder.

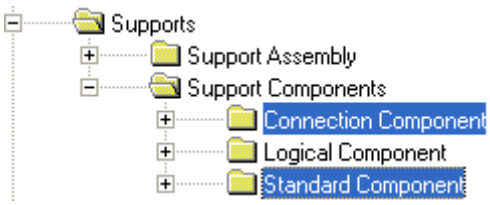


5. Next step, create a simple filter that includes reportable support components.

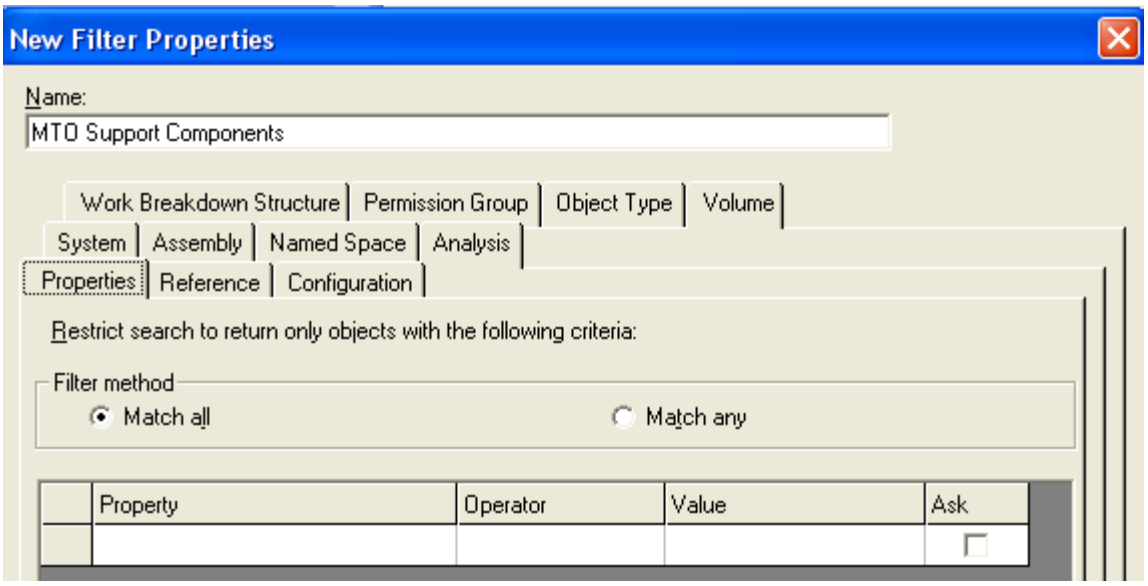
6. In the Select Filter dialog box, select Training and click the New Filter (Simple or Asking) icon to open the New Filter Properties dialog box.

7. In the New Filter Properties dialog box, type MTO Support Components in the Name box.

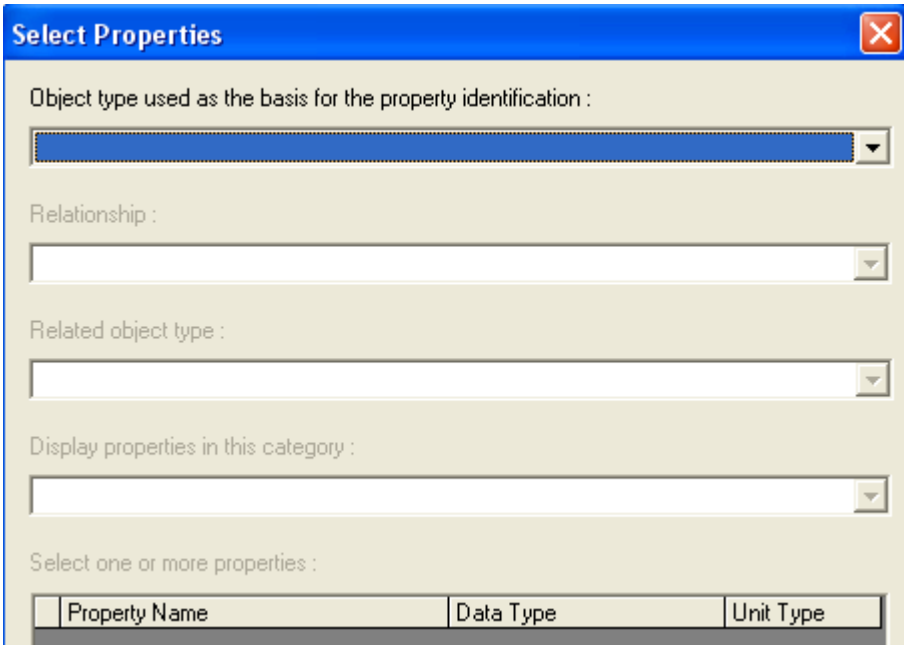
8. Click the Object Type tab and select Connection Component and Standard Component under SP3D -> Supports -> Support Components. Press and hold the CTRL key to select both nodes.



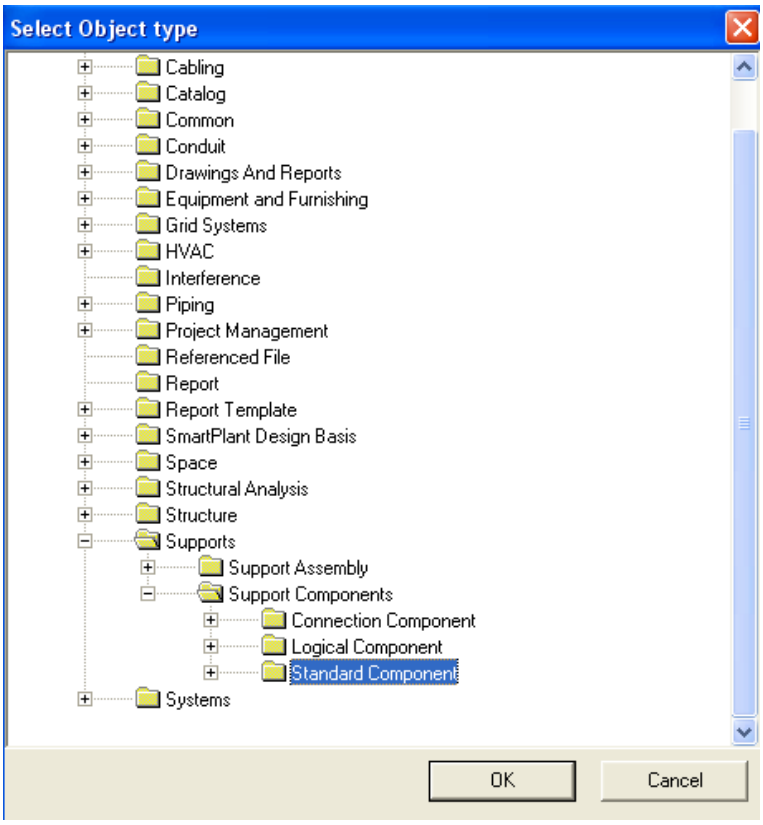
9. Click the Properties tab and select the More option on the Property column.



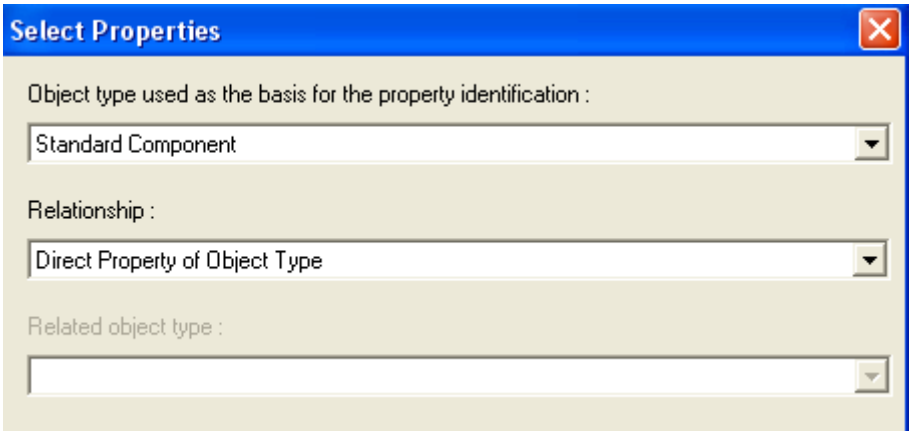
10. The Select Properties dialog appears. In the “Object type used as basis for property identification”, select More option.



11. You can select any object type that has the Reporting Requirement property. Select Standard Component under SP3D -> Supports -> Support Components and click OK.

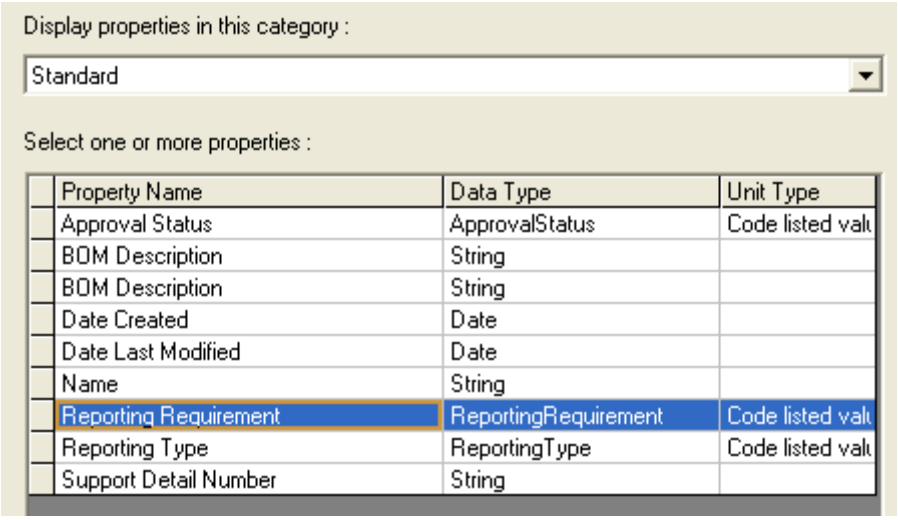


12. In the “Relationship” drop-down list, select Direct Property of Object Type.

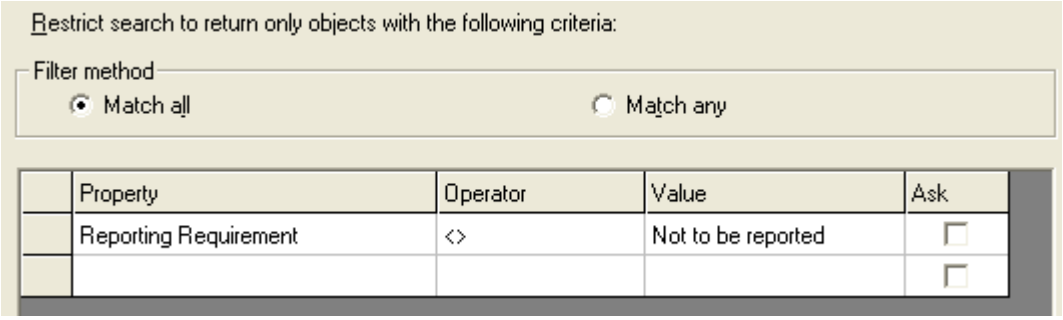


13. In the “Display properties in this category” drop-down list, select Standard

14. In the “Select one or more properties grid” area, select Reporting Requirement property and click OK.

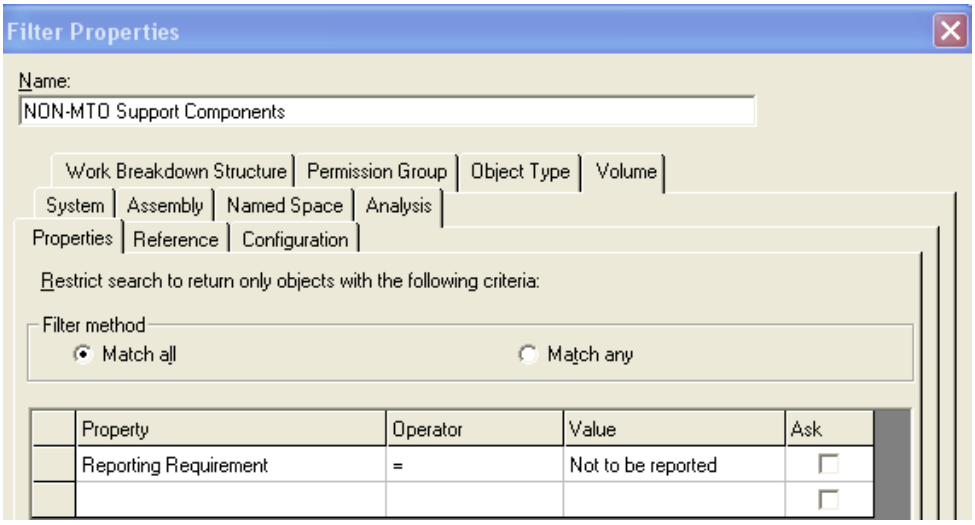


15. In the grid area, select “<>” operator in the Operator column. Select “Not to be reported” in the Value column.



16. Click OK.

17. Repeat the above steps to create another simple filter called “NON-MTO Support Components” to query only support components that are not to be reported.



18. Create a Plant Filter called “Pipe Supports in Unit 1” to query pipe supports in your Unit 1 of the model.

    Note: Use the system tab and the object type tab for this plant filter.

19. Go to the Hangers and Support Task and place few supports in this Unit 01.

20. Next step, prepare one drawing label to label the weld item.

21. Locate the drawing label files in your project symbol share. They should be located in: \\MachineName\Symbols\Drawings\Catalog\Labels\Templates

22. Locate the following files:

    - HngSup ISO_Note.rpt
    - HngSup ISO_Note.rqe
    - HngSup ISO_Note.rfm
    - HngSup ISO_Note.xml
    - HngSup ISO_Note.sym

23. Make a copy of those files and rename them as:

    - HngSup WeldType1.rpt
    - HngSup WeldType1.rqe
    - HngSup WeldType1.rfm
    - HngSup WeldType1.xml
    - HngSup WeldType1.sym

24. Open the HngSup WeldType1.rpt using a XML editor (Cooktop editor).

25. Edit the HngSup WeldType1.rpt as shown here:

```
<?xml version="1.0" encoding="UTF-8"?>
<REPORT_TEMPLATE
        Name="HngSup WeldType1"
        Description="Example of a Label for HngSup WeldType1">

  <QUERIES>
    <QUERY
        Name=" HngSup WeldType1"
        Site="User"
        Description=" HngSup WeldType1"
        Path="HngSup WeldType1.rqe" />
  </QUERIES>

  <FORMATTING
        Name=" HngSup WeldType1"
        Site="User"
        Description="HngSup WeldType1"
```

```
        Path="HngSup WeldType1.rfm" />

</REPORT_TEMPLATE>
```

26. Make sure there are no syntax errors.

27. Save the file.

28. Open and edit the HngSup WeldType1.rqe as shown here:

```
<?xml version="1.0" encoding="UTF-8"?>
<REPORT_QUERY
        Name="HngSup WeldType1"
        Description=""
        RequiresFilter="No">

  <DESIGN_TIME
        Progid="SP3DReportsQueryBuilder.COMQuery"
        Action=""
        Arg="" />

  <RUN_TIME
        Progid="SP3DRuntimeQuery.CQueryInterpreter"
        Action=""
        Arg="" />

  <RETURNED_PROPERTIES>
    <RETURNED_PROPERTY
                Name="BOM"
                SQLType="BStr">

      <PATHS>
        <PATH
        SourceType="*"
          DestinationInterface="IJHgrBOM"
          DestinationProperty="BOMdescription"
          Concatenate="No"
          PathSeparator="" />
      </PATHS>
    </RETURNED_PROPERTY>
  </RETURNED_PROPERTIES>
</REPORT_QUERY>
```

29. Make sure there are no syntax errors.

30. Save the file.

31. Open and edit the HngSup WeldType1.rfm as shown here:

```
<?xml version="1.0" encoding="UTF-8"?>
<REPORT_FORMATTING Name="HngSup WeldType1" Description="Example of Format used for
labels on instanciated objects">
  <DESIGN_TIME Progid="SP3DLabelFormatDesigner.RTFLabel" Action="" Arg="" />
```

```
  <RUN_TIME Progid="SP3DLabelsFormat.FormatLabel" Action="RTFLabel" Arg="" />

  <LAYOUT_TEMPLATE Type="Internal" />

  <FORMATTING_PARAMETERS Name="HngSup WeldType1" Site="User" Path="" />

  <RTF_LABEL>
    <BLOCKS>
      <BLOCK Action="Visible">
        <TOKENS>
          <TEXT Value="" Visible="Yes" />

          <DATA Column="BOM" Visible="Yes">
            <FORMAT_EXPRESSION Expression="&amp;" />
          </DATA>
        </TOKENS>
      </BLOCK>
    </BLOCKS>
  </RTF_LABEL>
</REPORT_FORMATTING>
```

32. Make sure there are no syntax errors.

33. Save the file.

34. Open and edit the HngSup WeldType1.xml as shown here:

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
FileName: HngSup WeldType1.xml
(Property=BOM Descr, text shape=None, position=Clear space, and Leader=true.)

Uses a label to report the BOM descr of the dummy weld part and positions the
text with a jogged leader attached in an absolute X and Y offset position.
-->
<label>
<!--Name of the label rule-->
  <name>HngSup WeldType1</name>
```

35. Use the default point generator module

```
<!--Contains settings used by the point Generator to generate points-->
  <pointGeneratorSettings>
<!--Default Point Generator used by Labels.-->
    <pgModule>DefaultLabelPointGenerator</pgModule>
  </pointGeneratorSettings>
```

36. Use the default geometric analyzer module

```
<!--The Geometric Analyzer takes points generated by a point generator and groups them
according its specifications-->
  <geometricAnalyzerSettings>
    <gaModule>DefaultLabelGeometricAnalyzer</gaModule>
```

```
  </geometricAnalyzerSettings>
```

37. Use the default label control generator module

```
<!--Determines the Module used to generate the Annotation-->
  <annotationControlGenerator>
    <acgModule>DefaultLabelControlGenerator</acgModule>
  </annotationControlGenerator>
```

38. Use the DrawingLabelHelper module. This module requires a symbol file and a report file to generate the output.

```
<!--Determines the content of the label to be placed-->
  <content>
    <contentModule value="DrawingLabelHelper" />

    <ID attributeName="BOM">HngSup WeldType1.rtp</ID>
  </content>
```

39. Define default label settings

```
<!--Label settings-->
  <labelSettings>
<!--define the maximum offset that a label can be from its connect point.-->
    <labelOffset>0.04</labelOffset>

<!--Defines how hard the system should look for clearspace: 0 course, 1 medium, 2 fine-->
    <granularity>1</granularity>

<!--This is used to tell the system at what priority this item should be considered.
      The lower the number the more likely this item will be closer to its first position.-->
    <priority>2</priority>

<!-- Defines the maximum/minimum scale factor that a label can be scaled to -->
    <maxFactor>1</maxFactor>

    <minFactor>1</minFactor>
  </labelSettings>
```

40. Define default positioning label settings

```
<!--Positioning Settings: The Positioning Modules are responsible to position the label -->
  <posSettings>
    <posModulesSets>
      <posModulesSet>
<!-- Key Point to Connect the leader to
            TopLeft=0, TopMiddle=1, TopRight=2,
            MiddleLeft=3, MiddleMiddle=4, MiddleRight=5,
            BottomLeft=6, BottomMiddle=7, BottomRight=8,
            ControlPoint=9-->
          <connectPoint>4</connectPoint>

          <positioningPoint>1</positioningPoint>

           <posModules>
```

```
        <posModule value="DrawingQuadOne"/>
        <posModule value="DrawingQuadTwo"/>
        <posModule value="DrawingQuadThree"/>
        <posModule value="DrawingQuadFour"/>
        <posModule value="DrawingAbsolute">
<!--Determine the X and Y offsets that a label is from its Connect Point.-->
            <hOffset>0.02</hOffset>
            <vOffset>-0.02</vOffset>
          </posModule>
        </posModules>
      </posModulesSet>
    </posModulesSets>
  </posSettings>
```

41. Define default leader settings

```
<!--Leader style and breakline settings.  For no leader delete this section-->
  <leaderSettings>
    <leaderModule>DwgLeaderControl</leaderModule>

<!--Name of the dimension style in the styles.sha file.-->
<style>
ANSI
</style>

<!--Determines if a breakline is placed. Off=0 On=-1-->
    <breakline>-1</breakline>
  </leaderSettings>
</label>
```

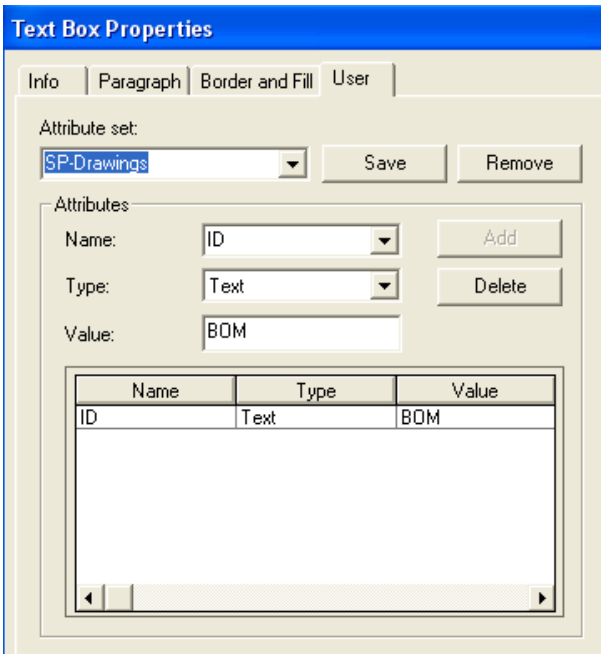
42. Make sure there are no syntax errors.

43. Save the file.

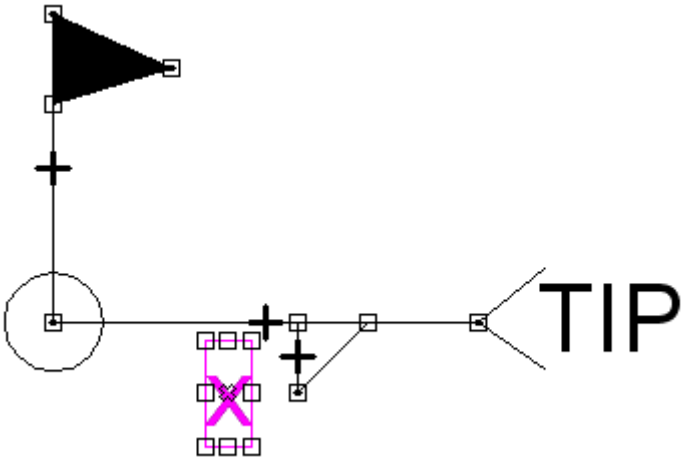
44. Open HngSup WeldType1.sym using the Shape 2D editor.

45. Right click the label to open the context menu. Select Properties to open the Text Box Properties dialog box.

46. Go to the User tab and edit the value attribute as follows:



47. Rename the text box as BOM. Draw the weld symbol as shown here:



48. Save the file.

49. Locate the dimension label definition rule HngSup ISO_Note.xml in your project symbol share. They should be located in: \\MachineName\Symbols\Drawings\Catalog\Rules\LabelRules

50. Make a copy of the file HngSup ISO_Note.xml and rename it as:

    - HngSup WeldType1.xml

51. Open the HngSup WeldType1.xml using a XML editor (Cooktop editor).

52. Edit the HngSup WeldType1.xml as shown here:

```
<?xml version="1.0" encoding="utf-8"?>
<RULE>
  <DESCRIPTION>Training: Weld Type1</DESCRIPTION>
  <ACTION>
    <CATEGORY>AddLabel</CATEGORY>

    <VALUE type="string">
      <labels>
        <label>
          <name>HngSup WeldType1</name>
        </label>
      </labels>
    </VALUE>
  </ACTION>
</RULE>
```

53. Make sure there are no syntax errors.

54. Save the file.

55. Locate the dimension labels HgrSup End_Support Assembly Vertical.xml and HgrSup End_Support Assembly Horizontal.xml in your project symbol share. They should be located in: \\MachineName\Symbols\Drawings\Catalog\Dimensions\Templates

56. Make a copy of the files and rename them as:

- RH HgrSup End_Support Assembly Vertical.xml
- RH HgrSup End_Support Assembly Horizontal.xml

57. Open the RH HgrSup End_Support Assembly Horizontal.xml using a XML editor (Cooktop editor).

58. Edit the RH HgrSup End_Support Assembly Horizontal.xml as shown here:

```
<?xml version="1.0" encoding="UTF-8"?>
<!--FileName: RH HgrSup End_Support Assembly Horizontal.xml-->
<dimension>
 <!--The following point generators are used to place dimension points on objects returned by
filters.-->
```

59. Use the point generator by note module. This module places points at the support keypoints having a note.

```
 <pointGeneratorSettings>
  <pgModule>DrawingPGByNote</pgModule>
 </pointGeneratorSettings>
```

60. Use the default geometric analyser module

```
<!--The following geometry analyzers are used to gather clusters of points to be dimensioned.-->
<geometricAnalyzerSettings>
  <gaModule>DrawingGeometricAnalyzer</gaModule>
  <placeLines>0</placeLines>
</geometricAnalyzerSettings>
```

61. Use the Drawing Dim generator module

```
<annotationControlGenerator>
  <!--This Annotation Control Generator will be used in most dimension rules.-->
  <acgModule>DrawingDimGenerator</acgModule>
</annotationControlGenerator>
```

62. Dimension Content Module for Linear Horizontal Dimensions

```
<!--The following dimension modules are used to place dimensions.-->
<dimensionContentModules>
  <!--This module places horizontal dimensions.-->
  <horiz>DrawingDimContLinHNoReplace</horiz>
</dimensionContentModules>

<dimensionAnchorSet>
</dimensionAnchorSet>
```

63. Define default dimension settings

```
<dimensionSettings>
  <!--Note: All of the following values are in meters relative to paper space.-->
  <!--This value determines the maximum horizontal distance between objects to be included-->
  <!--in a single dimension group.-->
  <maxDimOffset>0.2</maxDimOffset>
  <!--This value determines the maximum vertical distance between objects to be included in-->
  <!--a single dimension group.-->
  <maxWitnessLength>0.2</maxWitnessLength>
  <!--This option places an additional dimension over each dimension chain cluster identified-->
  <!--by this rule. Settings: -1 for true, 0 for false.-->
  <overall>0</overall>
  <!--This option determines the offset for the overall dimension from the dimension chain.-->
  <overallOffset>0.01</overallOffset>
  <!--This option determines how many attempts are made to place a dimension in clearspace.-->
  <!--Settings: 0 == Coarse, 1 == Medium, 2 == Fine.-->
  <granularity>0</granularity>
  <!--If the geometricAnalyzerSettings use the "DrawingGAMarginonly" module, this option-->
  <!--determines the distance that margin dimensions should be offset from the matchline.-->
  <perimeterOffset>0</perimeterOffset>
  <!--All annotation has a preferred placement position, this option resolves conflicts-->
  <!--between multiple rules. The lower the number, the higher the priority.-->
  <priority>1</priority>
  <!--This option determines whether horizontal dimensions are placed. Settings: -1 for-->
  <!--true, 0 for false.-->
  <horiz>-1</horiz>
  <!--This option determines whether vertical dimensions are placed. Settings: -1 for -->
  <!--true, 0 for false.-->
```

```
  <vert>0</vert>

  <keepTopOrBottomDup>1</keepTopOrBottomDup>
  <keepLeftOrRightDup>1</keepLeftOrRightDup>

  <!--This option determines Whether like dimension types should try to align with each-->
  <!--other. Settings: -1 for true, 0 for false. This value should be 0 for dimensions-->
  <!--positioned absolute.-->
  <align>0</align>
</dimensionSettings>
```

64. Define default dimension style settings

```
<dimensionStyleSettings>
  <!--This option determines what dimension style should be used from the styles.sha file-->
  <!--to create dimensions.-->
  <dimStyle>ANSI</dimStyle>
</dimensionStyleSettings>
<!--
        Positioning Module Settings
        -->
<posSettings>
  <posModulesSets>
    <posModulesSet>
      <posModules>
        <!--With settings below the dimensions placed will be 0.02 meters up and 0.02 -->
        <!--meters to the right of either the control point, the origin, or the center of-->
        <!--range of each object in the filter, regardless of where this would make the-->
        <!--dimensions end up individually.  Also, they are spaced out from the extent-->
        <!--points of the chain. They are not placed 0.02 meters up from each object itself.-->
        <horiz>
                <posModule value="DrawingDimAbsolutePos">
                <vOffset>0.02</vOffset>
                </posModule>
        </horiz>
      </posModules>
    </posModulesSet>
  </posModulesSets>
</posSettings>
</dimension>
```

65. Make sure there are no syntax errors.
66. Save the file.

67. Locate the dimension rule HgrSup End_Support Assembly Vertical.xml in your project symbol share. They should be located in: \\MachineName\Symbols\Drawings\Catalog\Rules\DimensionRules

68. Make two copies of the file HgrSup End_Support Assembly Vertical.xml rename them as:

- RH HgrSup End_Support Assembly Vertical.xml
- RH HgrSup End_Support Assembly Horizontal.xml

69. Open the RH HgrSup End_Support Assembly Horizontal.xml using a XML editor (Cooktop editor).

70. Edit the RH HgrSup End_Support Assembly Horizontal.xml as shown here:

```
<?xml version="1.0" encoding="UTF-8"?>
<RULE>
  <DESCRIPTION>Dim Line Orientation:Horizontal, Style:ANSI,
Positioning:Margin</DESCRIPTION>
  <ACTION>
    <CATEGORY>AddDimension</CATEGORY>
    <VALUE type="string">
      <dimensions>
        <dimension>
          <name>RH HgrSup End_Support Assembly Horizontal</name>
          <filter>RH HgrSup End_Support Assembly Horizontal</filter>
        </dimension>
      </dimensions>
    </VALUE>
  </ACTION>
</RULE>
```

71. Make sure there are no syntax errors.
72. Save the file.
73. Open the RH HgrSup End_Support Assembly Vertical.xml using a XML editor (Cooktop editor).
74. Edit the RH HgrSup End_Support Assembly Vertical.xml as shown here:

```
<?xml version="1.0" encoding="UTF-8"?>
<RULE>
  <DESCRIPTION>Dim Line Orientation:Vertical, Style:ANSI, Positioning:Margin</DESCRIPTION>
  <ACTION>
    <CATEGORY>AddDimension</CATEGORY>
    <VALUE type="string">
      <dimensions>
        <dimension>
          <name>RH HgrSup End_Support Assembly Vertical</name>
          <filter>RH HgrSup End_Support Assembly Vertical</filter>
        </dimension>
      </dimensions>
    </VALUE>
  </ACTION>
</RULE>
```

75. Make sure there are no syntax errors.

76. Save the file.

77. Next step, create three view styles

78. Go to Drawings and Reports Task and select Tools-> Define View Style.

79. For the “View Style Type:” property, drop the select list down and select Orthographic.

80. Select the “New Style” button from the top left of this form and rename the style to “RH Hng Sup - Iso” by select the new style that was created with the default name and using the “Rename” button.

81. Double click on the new style created and the following “View Style Properties” dialog will appear:

82. Since this is a Volume view style, you will need to change the “Filter Behavior” property to “Filters determine which objects are processed (Volume)”.

83. Select the HngSup – End for the Graphic preparation rules.

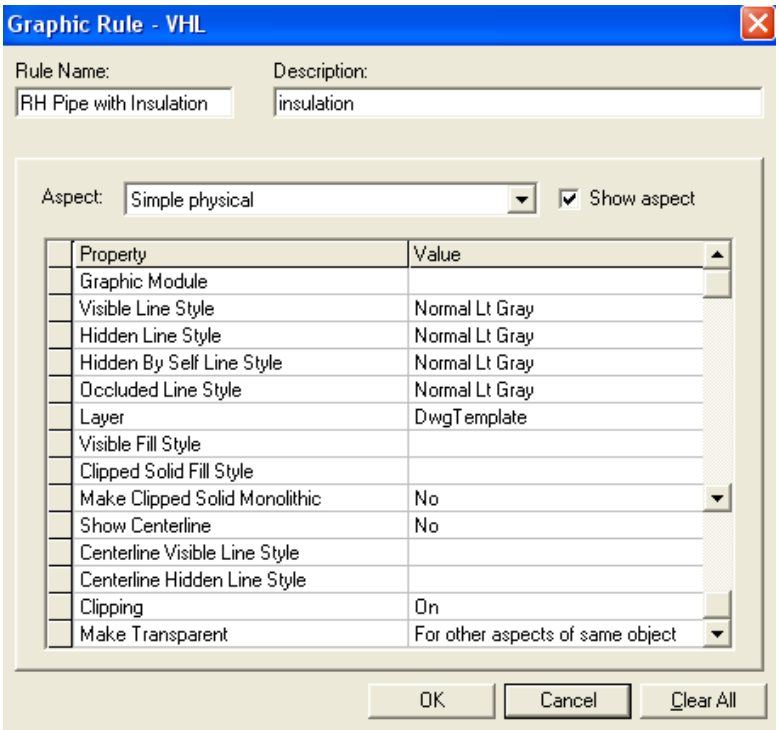
84. Define the appropriate rule as shown below:

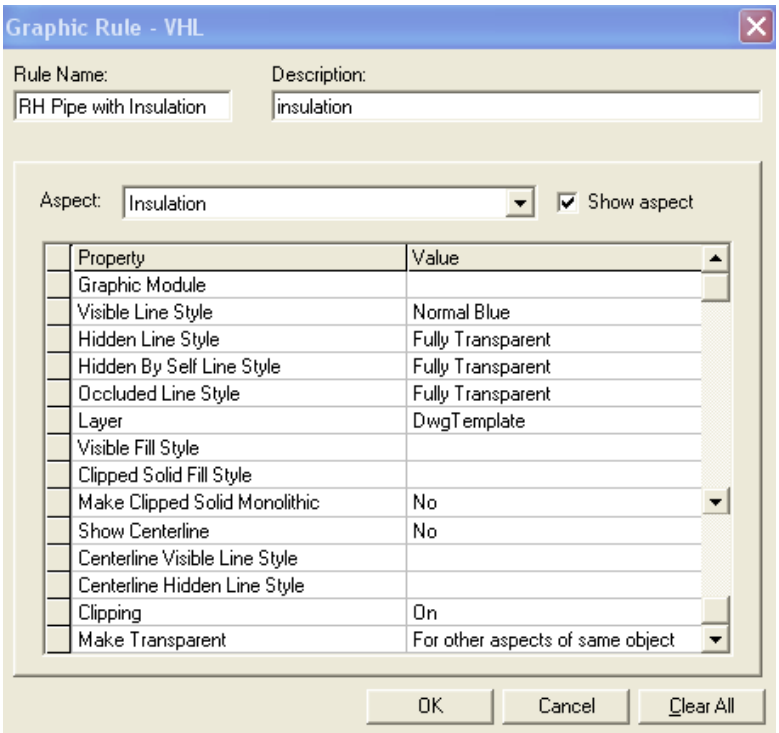
View Style Name	Description			
RH Hng Sup - Iso				
View Style Settings				
	Filter Behavior	Graphic Preparation Rule	Intersection Edges	Preserve Z Order
	spatial	HngSup - End	Low Resolution	FALSE
	Tests			
	Filter Name	Primary Orientation	Secondary Orientation	Clipping
	catalog\Catalog Filters\Default Filters\SP3D Object Filters\Object Types\Structure			
	catalog\Catalog Filters\Default Filters\SP3D Object Filters\Object Types\Piping			
	catalog\Catalog Filters\Training\MTO Support Components			

Actions		
Graphic Rule	Label Rule	Dimension Rule
HngSup End_Structure Reference		
RH Pipe with Insulation		
	HngSup ISO_Reference_Circle_CA_L.xml	

Note:

RH Pipe with Insulation Grahic Rule
The settings for this rule are shown below:





85. Click OK to save the view style.

86. Repeat the above steps to create the other two view styles using the following data:

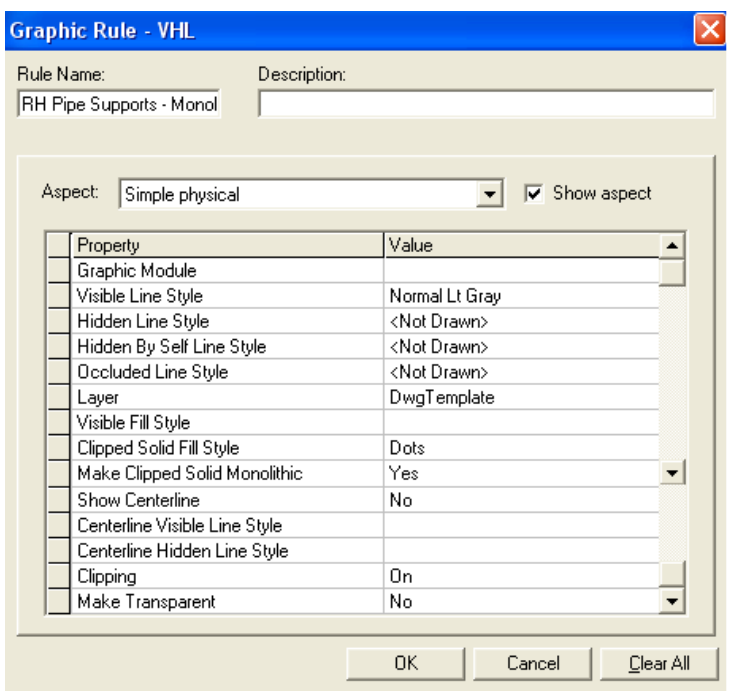
View Style: RH HngSup – End Dim

View Style Name	Description			
RH HngSup - End Dim				
View Style Settings				
	Filter Behavior	Graphic Preparation Rule	Intersection Edges	Preserve Z Order
	spatial	HngSup - End	Low Resolution	FALSE
	Tests			
	Filter Name	Primary Orientation	Secondary Orientation	Clipping
	catalog\Catalog Filters\Default Filters\SP3D Object Filters\Object Types\Structure			
	catalog\Catalog Filters\Default Filters\SP3D Object Filters\Object Types\Piping			
	catalog\Catalog Filters\Default Filters\SP3D Object Filters\Object Types\Piping\Piping Parts\Pipes	Normal		
	catalog\Catalog Filters\Default Filters\SP3D Object Filters\Object Types\Supports\Support Assembly			
	catalog\Catalog Filters\Default Filters\SP3D Object Filters\Object Types\Supports\Support Assembly			

Actions		
Graphic Rule	Label Rule	Dimension Rule
RH Pipe Supports - Monolithic Structure_Ref		
HngSup End_Piping Reference		
HngSup End_Pipe End		
		RH HgrSup End_Support Assembly Horizontal
		RH HgrSup End_Support Assembly Vertical

Note:

RH Pipe Supports – Monolithic Structure_Ref Grahic Rule.
The settings for this rule are shown below:



View Style: RH HngSup – End Weld

View Style Name	Description			
RH HngSup - End Weld				
View Style Settings				
	Filter Behavior	Graphic Preparation Rule	Intersection Edges	Preserve Z Order
	spatial	HngSup - End	Low Resolution	FALSE
	Tests			
	Filter Name	Primary Orientation	Secondary Orientation	Clipping
	catalog\Catalog Filters\Default Filters\SP3D Object Filters\Object Types\Structure			
	catalog\Catalog Filters\Default Filters\SP3D Object Filters\Object Types\Piping			
	catalog\Catalog Filters\Default Filters\SP3D Object Filters\Object Types\Piping\Piping Parts\Pipes	Normal		
	catalog\Catalog Filters\Default Filters\SP3D Object Filters\Object Types\Piping\Piping Parts			
	catalog\Catalog Filters\Default Filters\SP3D Object Filters\Object Types\Supports\Support Assembly			
	catalog\Catalog Filters\Training\NON-MTO Support Components			

Actions		
Graphic Rule	Label Rule	Dimension Rule
RH Pipe Supports - Monolithic Structure_Ref		
HngSup End_Piping Reference		
HngSup End_Pipe End		
	HngSup End_Diamond_M_L	
	HngSup WeldType1	

87. Click OK to save the view styles.

88. Next step, create a custom Orthographic Drawing Package.

89. Go to Drawings and Report Task.

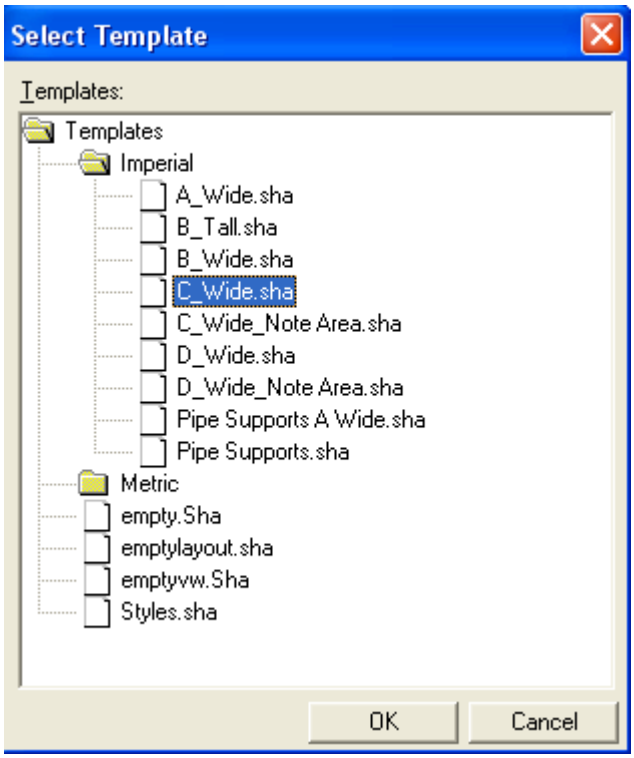
90. Right click on the plant and create a folder called Packages.

91. Right-click on Packages and add a New Orthographic Drawings by Query component.

92. Rename New Orthographic Drawings by Query to RH Pipe Supports.

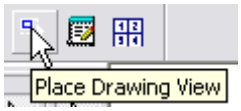
93. Right click on RH Pipe Support and Edit Template.

94. Select Imperial\C Wide.sha and click OK.

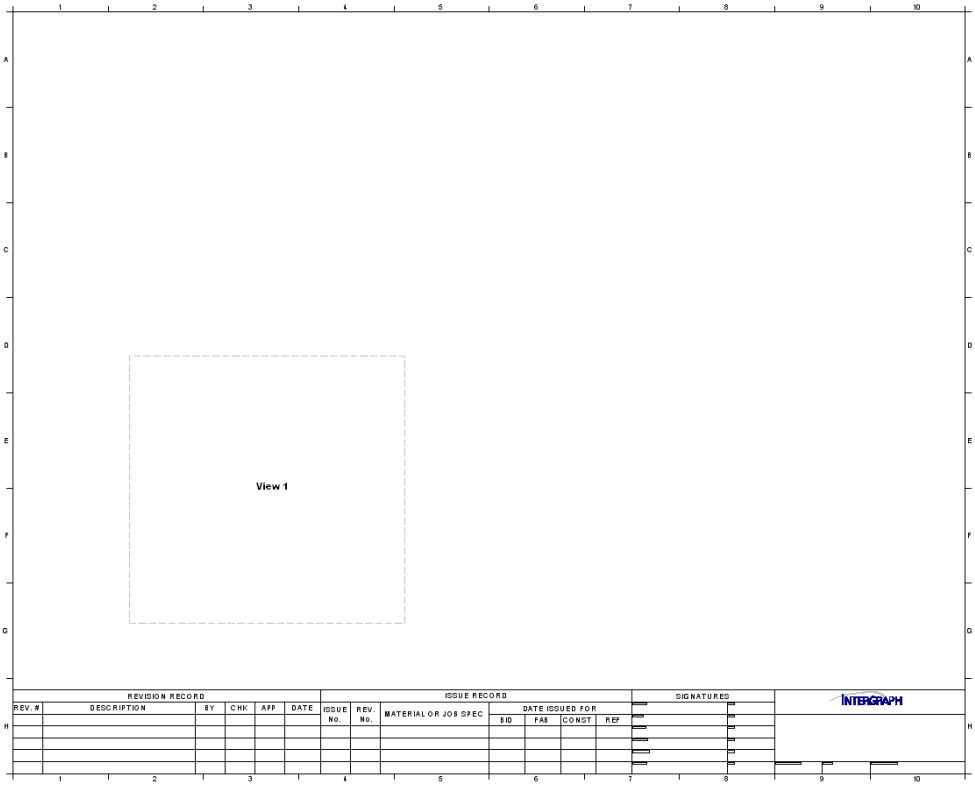


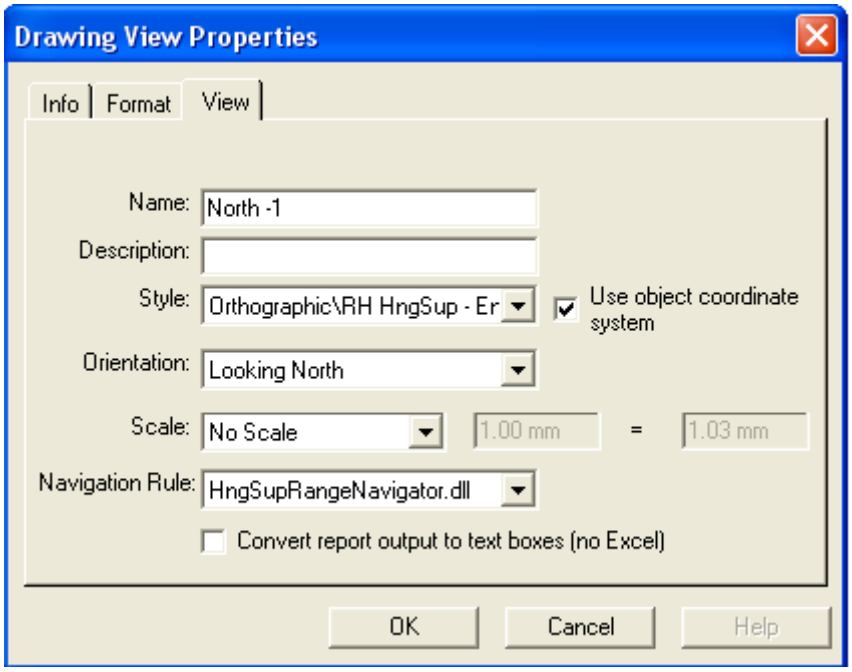
95. Maximize and fit the view in the Drawing Editor

96. Use the Place Drawing view command to draw the 2D view coincident with the view shown here.

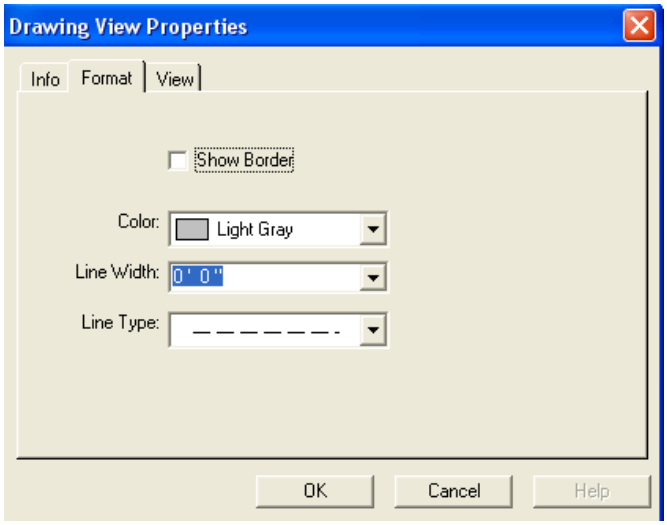


97. Fill in the Drawing View Properties as shown here:

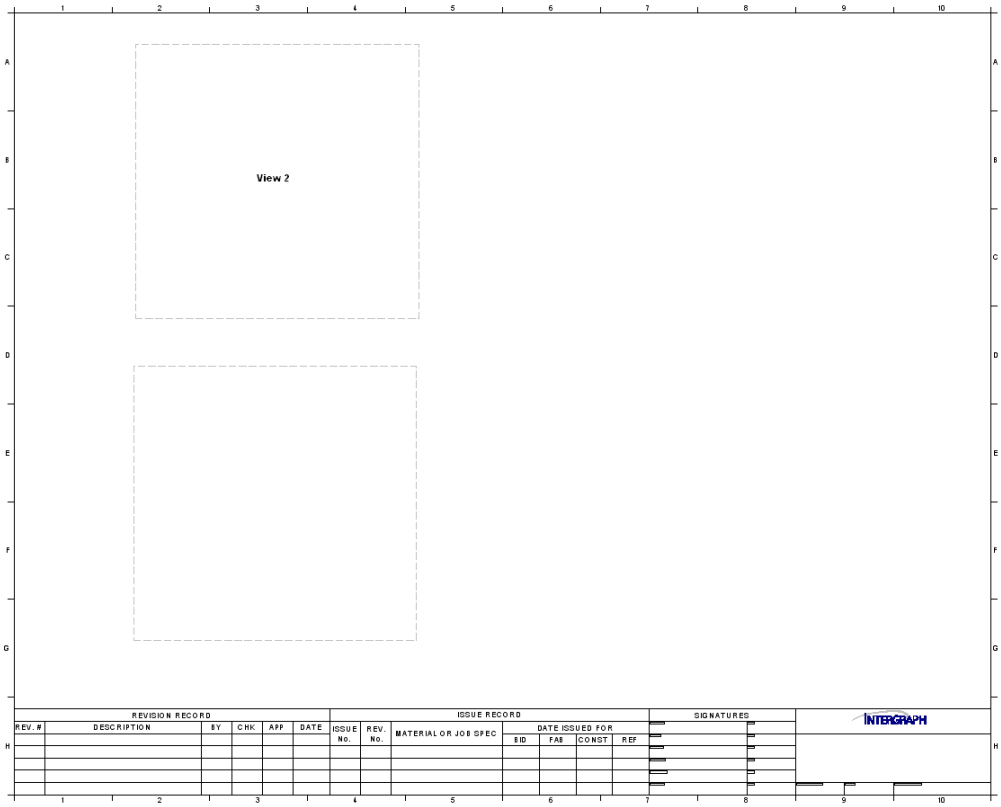


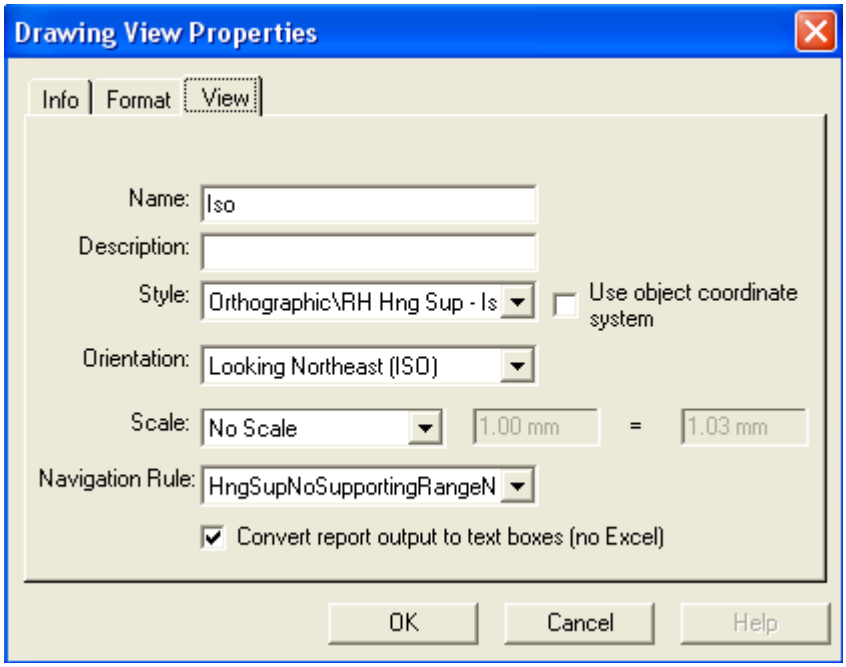


98. Switch to the Format tab and uncheck the Show Border box.

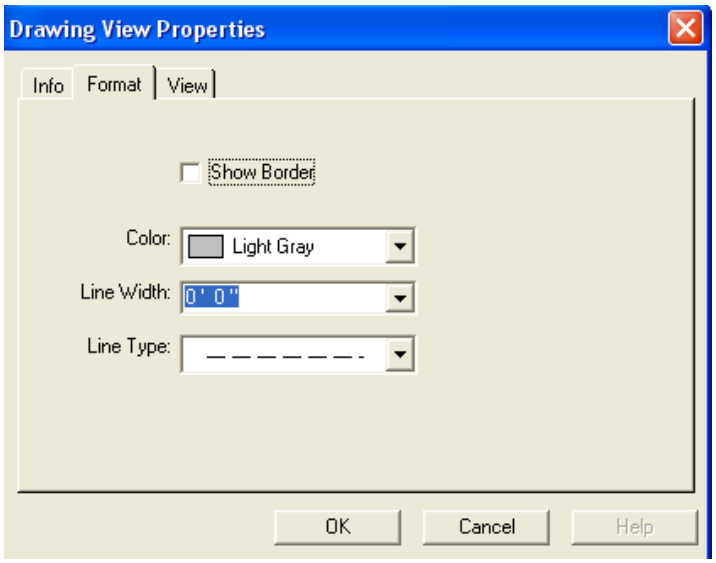


99. Repeat the above steps to place the second view. Fill in the Drawing View Properties as shown here:

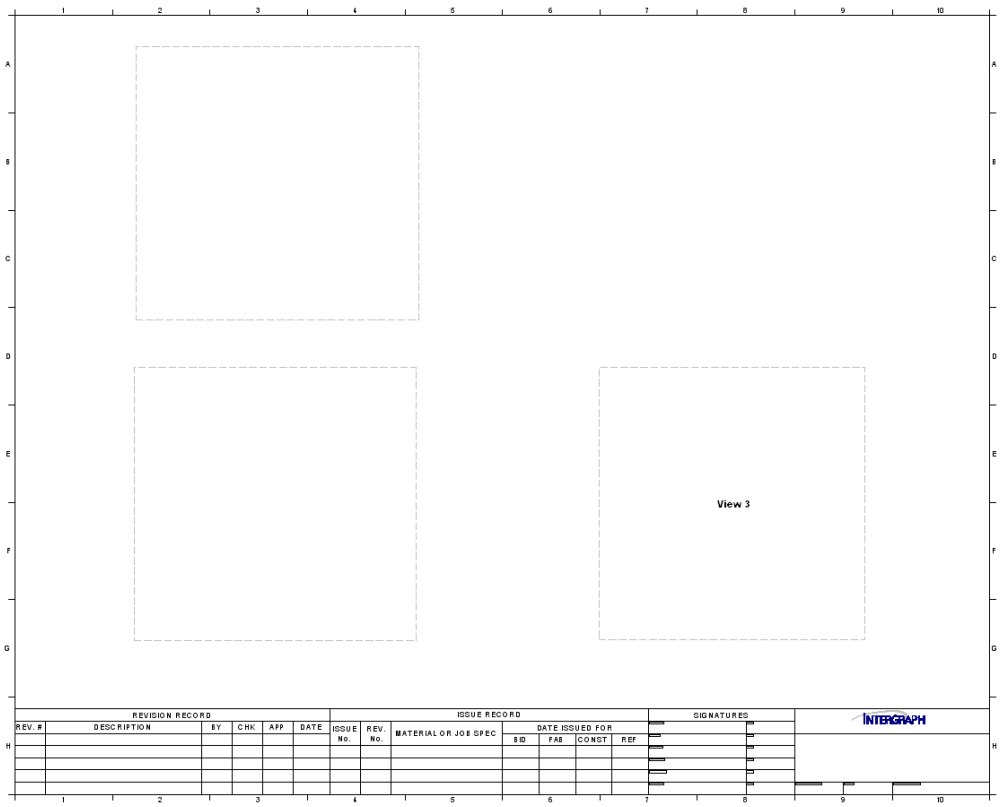


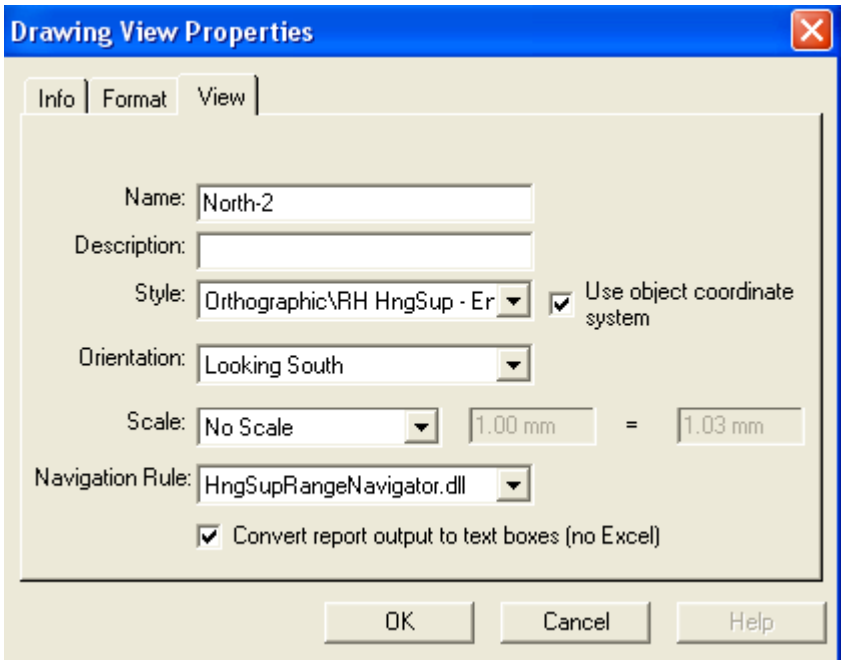


100. Switch to the Format tab and uncheck the Show Border box.



101. Repeat the above steps to place the third view. Fill in the Drawing View Properties as shown here:



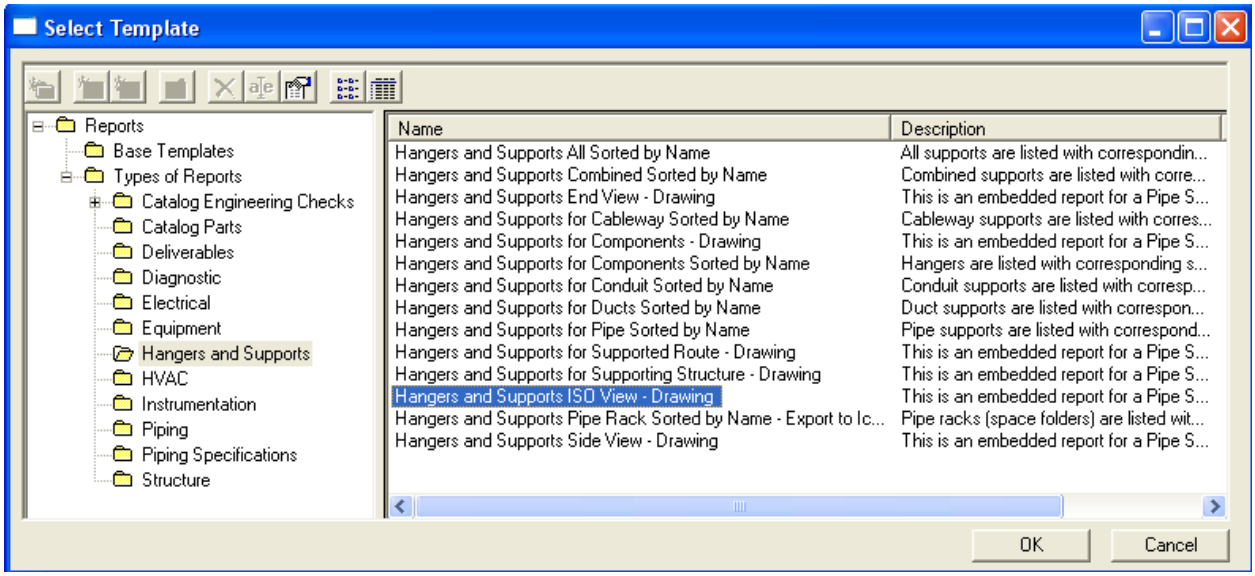


102. Select the second view to make it active.

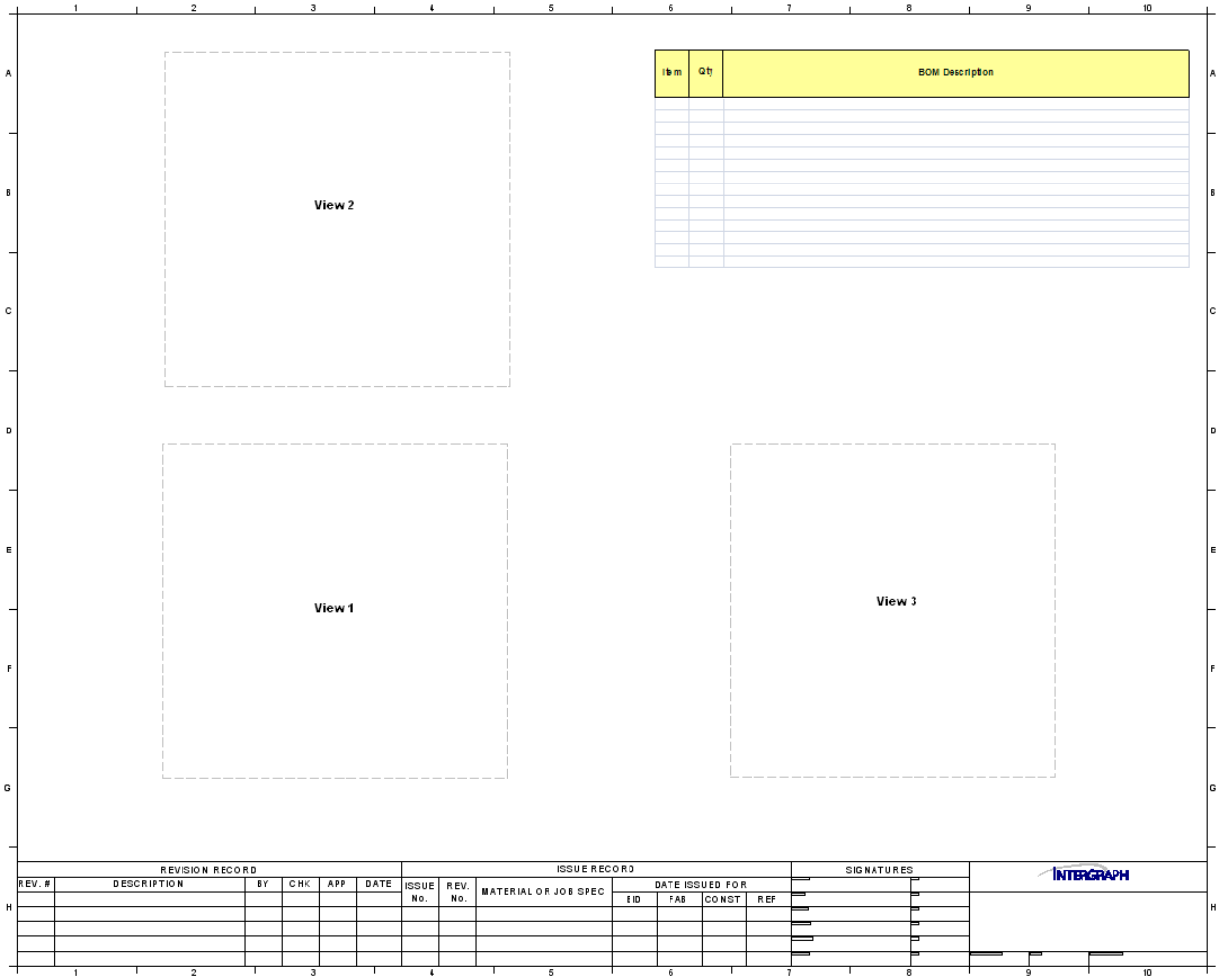
103. Select “Place report” icon on the taskbar.

104. The Select Report Template dialog appears.

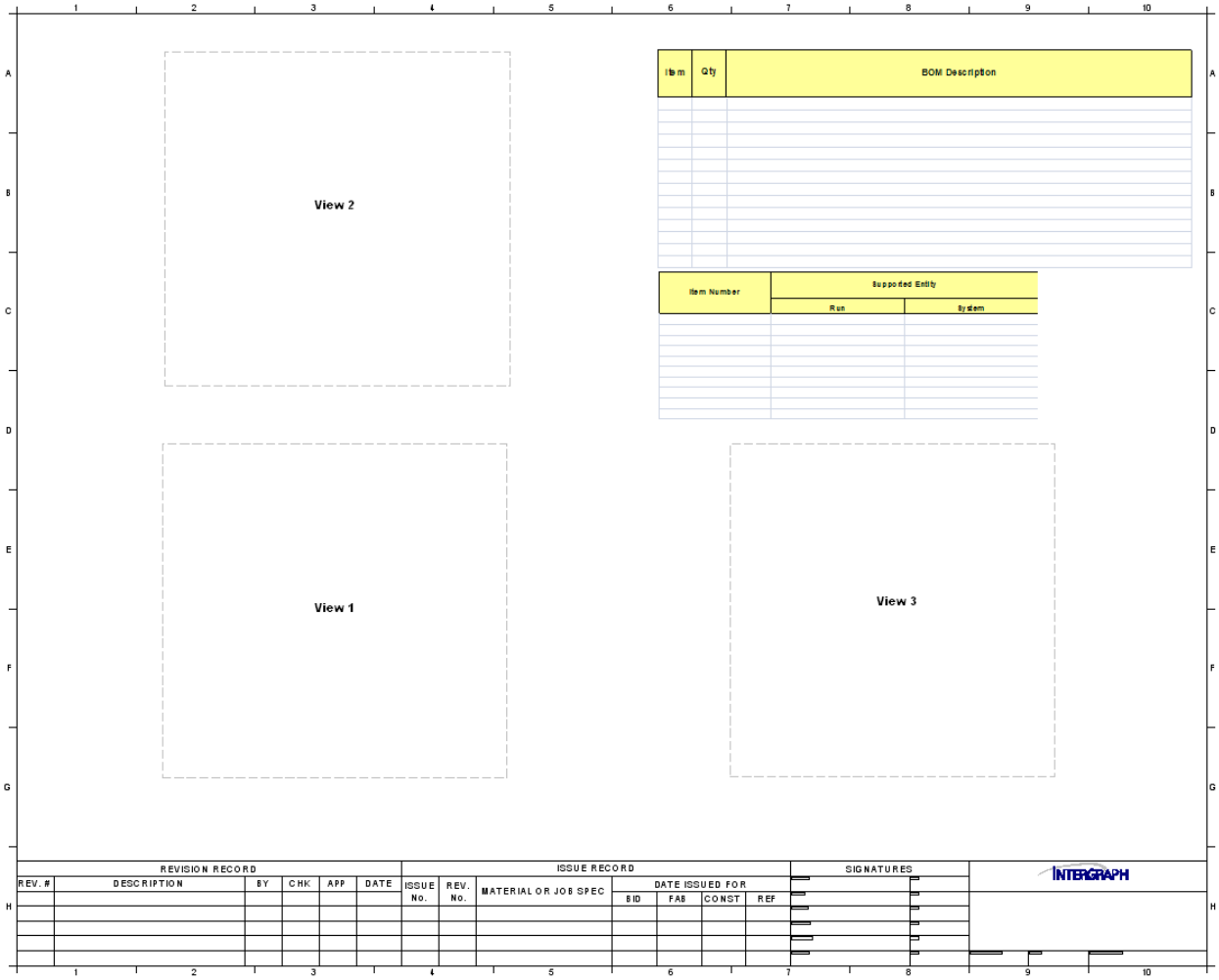
105. Select the report template “Hanger and Supports Iso View – Drawing” under Reports -> Type of Reports -> Hangers and Supports folder and click OK.



106. Place the Excel template at the top right of corner of the drawing as shown here:



107. Repeat the above steps to place the second report using the third view.

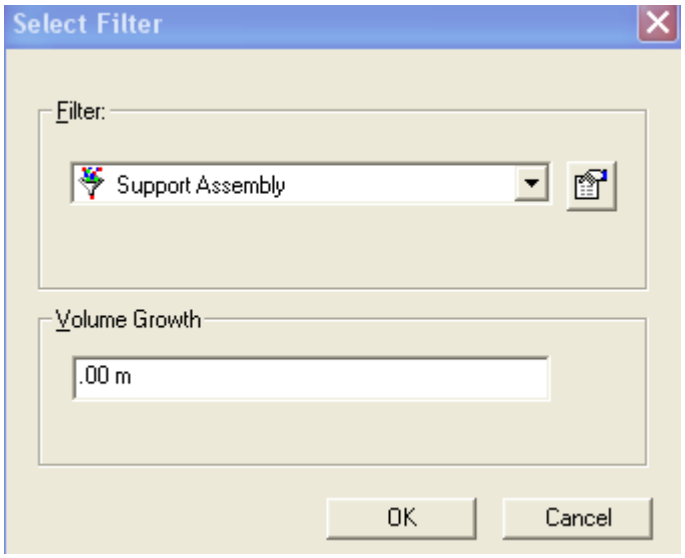


108. Save and Exit the Drawing Editor.

109. Right click on RH Pipe Support component and Select Setup

110. Fill in the Select Filter dialog and Click OK, as shown here:

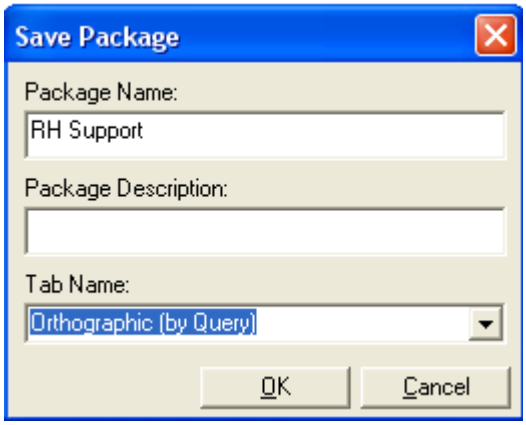
- Filter: Support Assembly under Catalog Filter ->Default Filters ->SP3D Object Filters -> Object Types -> Supports folder.
- Volume Growth: 0.0 meter



111. Right click the RH Pipe Support again and Select Save Package.

112. Key-in “RH Support” in Package Name field.

113. Select “Orthographic (by Query)” in Tab Name drop-down list and Click OK.



114. Create a new folder “Pipe_supports” under the plant in the Drawings and Reports task.

115. Add component “Drawings By Query Manager” under that folder. Rename it to “Support Drawings”.

116. Right click on Support Drawings” and Select Setup.

117. For package choose the package that you created above.

118. Select the Plant Filter “Pipe Supports in Unit 1” created at the beginning of this exercise.

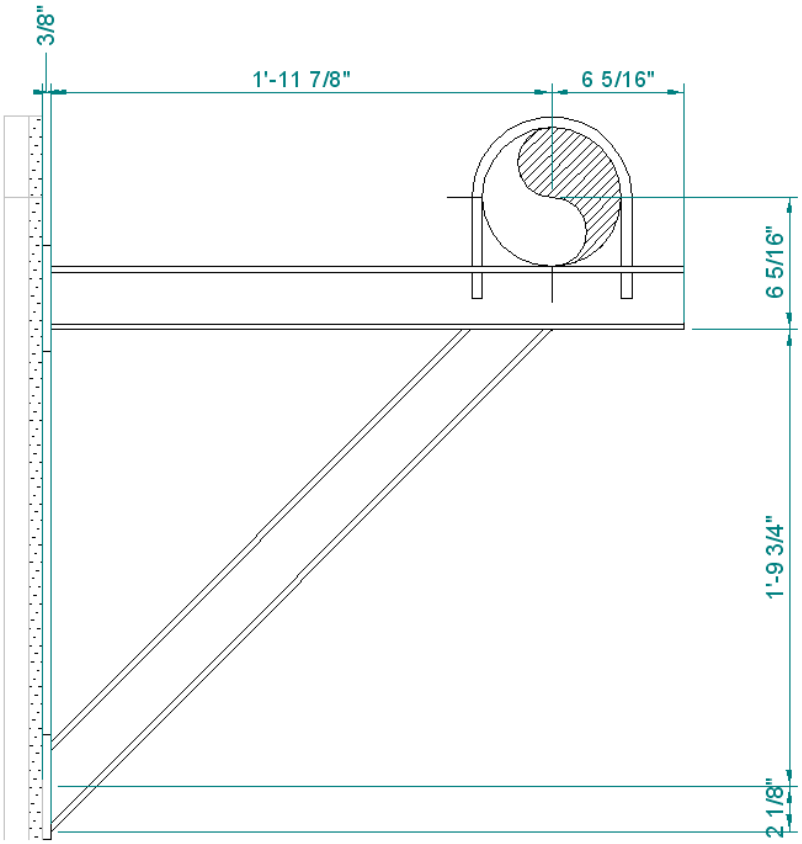
119. Run query for the level “Support Drawings” (under the main folder).

120. You should get the plant hierarchy created under “Support Drawings” component.

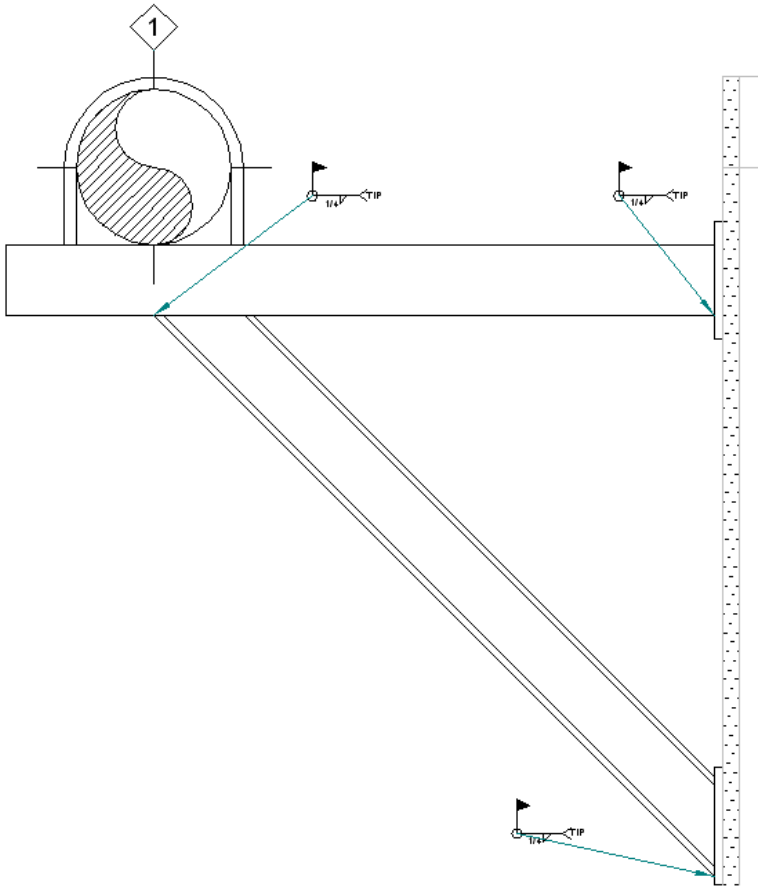
121. Select “Create Drawings” for the lowest level in hierarchy. You should get drawings created for each pipe support.

122. Select “Update Now” to create the contents to the support drawings.

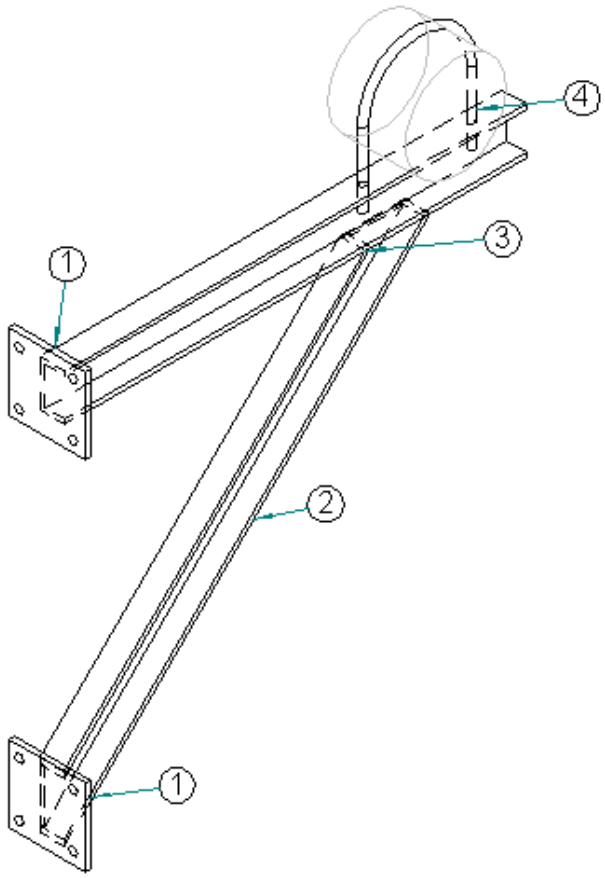
123. For viewing the drawings select “Edit” or “Open”.



Item Number	Supported Entity	
	Run	System
1	U01-6-P-0203-2C0032	1002-P



Item	Qty	BOM Description
1	2	0.44 in Plate Steel, 5.00 in X 5.00 in, 4 eq spaced 0.56 in dia. holes 0.75 in from edges
2	1	C3x6 , Length: 781.61 mm , Angle: 45
3	1	C3x6, Length: 766.90 mm
4	1	Light Weight U-Bolt with two hex nuts, Galvanized Finish.



Appendix

IJHgrInputConfigHlpr

IJHgrInputConfigHlpr is the interface that the CHgrSupport object implements. This interface is intended to be used by the end user to create assembly information rules and part selection rules. Use this interface to get information about the inputs to the active entity and various other features made available to the end user.

a. GetPipes - Method to get the collection of pipes that the hanger active entity is attached to. Use this method only if confident that the input is pipes. Otherwise use the method “GetSupportedCollection”.

b. GetDucts - Method to get the collection of ducts that the hanger active entity is attached to. Use this method only if confident that the input is ducts. Otherwise use the method “GetSupportedCollection”.

c. GetCableTray - Method to get the collection of Cable Tray that the hanger active entity is attached to. Use this method only if confident that the input is CableTray. Otherwise use the method “GetSupportedCollection”.

d. GetSupportedCollection - Use this method to retrieve the supported collection when not sure of the supported type.

e. PrimaryPipeDiameter - Method to get the diameter of the first input pipe.

f. GetPipeDiameter - Method to get Pipe Diameter of the input pipe port. Use this method when there are more than one input pipe.

g. GetRteCrossSectParams - To retrieve the cross section dimensions of the input supported object. This object can be a pipe, duct or cableway.

h. GetTurnFeatProperties - Use this method to retrieve the properties associated specifically to a turn feature**.**

i. GetExternalPipeDiameter - Method to retrieve the outer diameter of the route from the route port.

j. SupportedParts - Property to get the input route feature object parts based on index.

k. SupportedPartsCount - Property to the get the number of input route feature object parts based on index.

l. SupportedSectionOrientation - This method defines the orientation of the route object w.r.t. to the structure.

m. GetSupportedSectionOffset - This method returns the actual width and depth of a cableway object based on the TrayOffset service table defined in the catalog.

n. GetBendRadiusAndCenterOfRotation - This method returns the geometric properties of the route turn feature object.

o. GetRteSpec - Method to retrieve the spec applied on certain route object.

16. GetRteSpecVersion – Method to get the piping spec version.

p. GetRouteMaterialTypeGrade - returns the material type and grade of the given route object.

q. GetStructure - Method to get the structure collection.

r. GetSupportingProperty - Method to get property of beam cross section by the name of the property. Use this method when you are sure that the input beam to the active entity is only an SPS Member port. Also, the property being requested for must be defined the Structure Cross-Sections catalog.

s. GetStructureShape - To distinguish between whether the supporting object is a member or a slab.Useful while trying to place a support by two supporting objects.

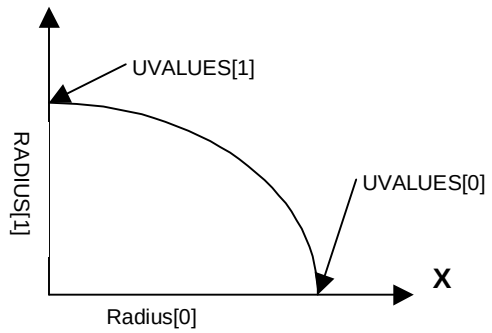
Helper Function Reference

This Appendix is a reference for all built-in functions usable in common bas files

PF_HgrPrimitives Bas

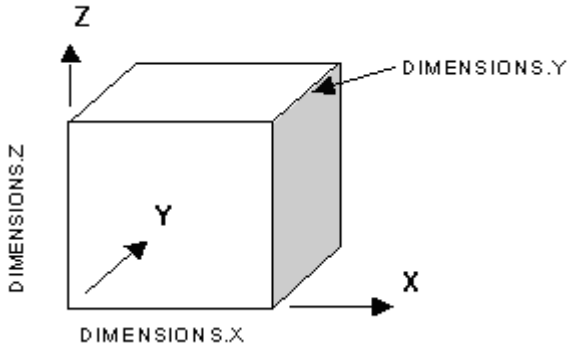
AddArc

Description: Create the graphical representation of an Arc. Specify the start and end angles, the radius and a starting point
Inputs - startAngle - double, endAngle - double, RADIUS - double, strmatrix - string, outputcoll - SP3D Output Collection Object, Output - string, Name - string



AddBox

Description: Create the graphical representation of a box
Inputs - X - double, Y - double, Z - Double, strmatrix – string, outputcoll - SP3D Output Collection Object, Output - string, Name – string

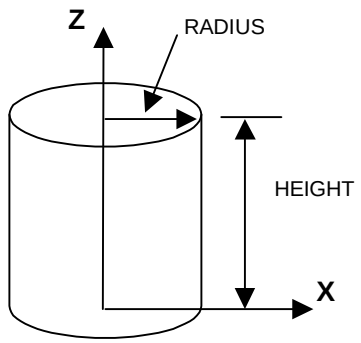


AddCylinder

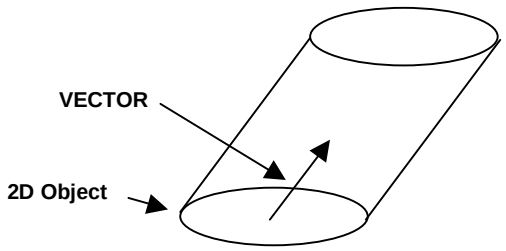
Description: Create the graphical representation of a cylinder. Specify the start point (3d matrix), length and the radius
Inputs - LENGTH - double, RADIUS - double, strmatrix - string, outputcoll - SP3D Output Collection Object, Output - string, Name – string

AddCylinderWP

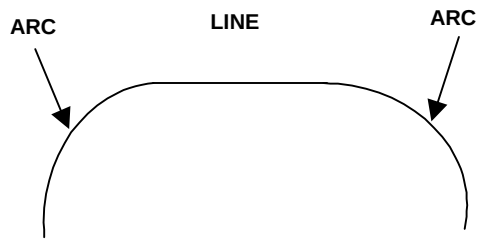
Description: Create the graphical representation of a cylinder. Specify the start point (3d), end point (3d) and the radius.
Inputs - startx - double, starty - double, startz - Double, endx - double, endy - double, endz - Double, RADIUS - double, outputcoll - SP3D Output Collection Object, Output - string, Name - string



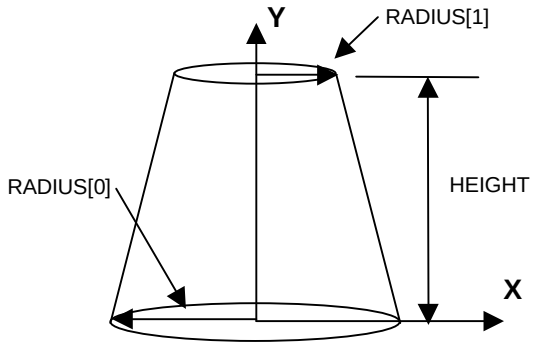
AddExtrusion
'Description: Create the graphical representation of an extrusion. Specify the amount to extrude in the x, y and z directions. Also specify if the extrusion is to be capped
Inputs - extrudeX - double, extrudeY - double, extrudeZ - double, capped – Boolean, outputcoll - SP3D Output Collection Object, Output - string, Name – string



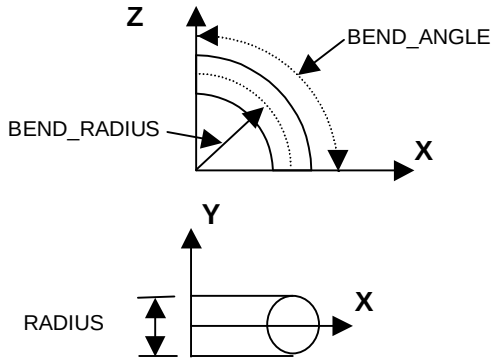
AddComposite
Description: Create representation of an Composite. Specify the number of pieces and whether it is flat or extruded
Inputs - numPieces - Integer, flatOrExtruded – string, Output - string, Name - string



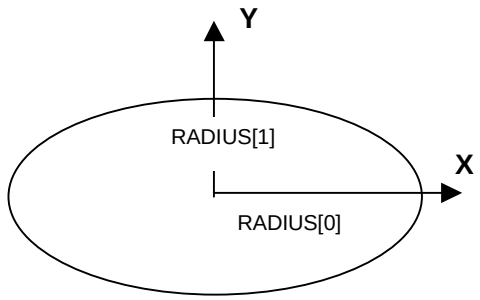
AddCone
Description: Create the graphical representation of a Cone. Specify the bottom radius, the top radius, the height and a starting point.
Inputs - bottomRadius - double, topRadius - double, height - double, strmatrix – string, outputcoll - SP3D Output Collection Object, Output - string, Name – string



AddElbow
Description: Create the graphical representation of a cylinder elbow. Specify the cylinder radius, the elbow radius, the sweep angle and the starting position (3d matrix)
Inputs - rodradius - double, elbowradius - double, SweepAngle - double, strmatrix - string, outputcoll - SP3D Output Collection Object, Output - string, Name – string

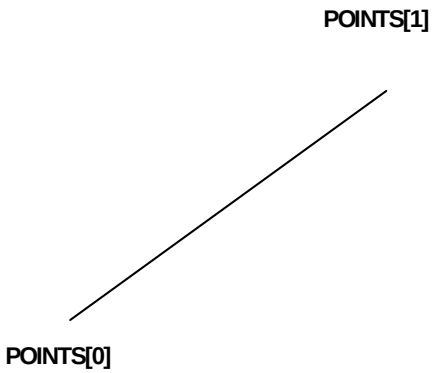


AddEllipse
'Description: Create the graphical representation of an ellipse. Specify the two radii, whether it is flat or extruded and a starting point



Inputs - radius1 - double, radius2 - double, flatOrExtruded - string, strmatrix – string, outputcoll - SP3D Output Collection Object, Output - string, Name - string

AddLine
Description: Create the graphical representation of a line. Specify the start x, y, z and end x, y, z
Inputs - startx - double, starty - double, startz - double, endx - double, endy - double, endz - double, outputcoll - SP3D Output Collection Object, Output - string, Name - string



AddPoint
Description: Create the graphical representation of a point. Specify the location (X, y, z).
Inputs - locX - double, locY - double, locZ - double, outputcoll - SP3D Output Collection Object, Output - string, Name – string

AddPolygon
Description: Create the graphical representation of a Polygon. Specify the number of points and if it will be flat or extruded
Inputs - numPoints - integer, flatOrExtruded - string,
Output - string, Name – string

Loc
Description: Creates a location string from x, y and z coordinates
Inputs - x - double, y - double, z – double

Rotx(angle), Roty(angle), Rotz(angle)
Description: String containing a rotation string appropriate for passing into graphic primitive functions when used together with the Loc function.
Inputs: - angle - Number, angle in degrees to rotate this item

PF_HgrGeneralUtil.bas:

1. SetupInputsAndOutputs - This function was created to modularize the code needed to generate the inputs and outputs that are required to develop a SP3D symbol.

2. SetupInputsAndOutputsForCache -This function was created to modularize the code needed to generate the inputs and outputs that are required to develop a SP3D symbol. This function will cache the symbol.

3. ASin - Calculate the Arc Sin of the input value.

4. ACos - Calculate the Arc Cos of the input value.

5. FIFtoMeter - - Convert a string measurement that is feet-inch-fraction to Meters (Used in many Anvil parts).

6. ReadParametricData – Create the sql statement that will be used in data look-ups.

7. CosDeg - Calculate the cosine of an angle that is in degrees.

8. SinDeg - Calculate the sine of an angle that is in degrees.

9. TanDeg - Calculate the tan of an angle that is in degrees.

10. ACosDeg - Calculate the arc cosine of an angle that is in degrees.

11. ASinDeg - Calculate the arc sine of an angle that is in degrees.

12. ATanDeg - Calculate the arc tan of an angle that is in degrees.

13. Deg - Convert the value from radians to degrees.

14. Rad - Convert the value from degrees to radians.

15. ConvertValueToLongStringValue - This function can be used to look up data in a codelist or to convert units optionally pass in bFIF, if true and the units are feet or inch, the string returned is in FIF format, 1/16 precision optionally pass in iPrecision to control fractional precision for FIF, defaults to 16ths if not passed in optionally pass in uPrimaryUnits to control the displayed units to override the custom interfaces sheet. These units must be one of the units as defined by IJUnitsOfMeasure.

16. ListAttributesOfObject - This function returns a list of all the attributes that the object has associated with it

17. LogError - default Error logger.

18. LogErrorWithContext - default Error logger, same as LogError but adds the errors with proper error context.

19. PF_EventHandler - This procedure will allow us to use the same line of code to either log a message or raise an error when an expected event occurs
    WarnOnly = False --> We raise an error with the provided information and get kicked out of the current method
    WarnOnly = True --> We write the provided info to the log file but DO NOT raise an error, the method continues

20. RoundDimension - - This function will round the given dimension of the part based on the rule HgrSupDimensionRoundFactor. This rule is from the HgrRules sheet of HS_System.xls. This rule should return a long number corresponding to the number of decimal places to round the dimension to.

21. AssignArraySizeForInput - Redimensions the Input array of a symbol.

22. AssignArraySizeForOutput - Redimensions the Output array of a symbol.

23. GetNParamUsingParametricQuery – To retrieve a numeric value from database.

24. GetSParamUsingParametricQuery - To retrieve a string value from database.

25. ParamInput – To set the parameter direction (adParamInput).

26. FillParameter - Use this function to fill all the values for the Parameter.

27. FillParameterForDouble - Use this function to fill Name and Value of a Parameter which is of type Double.

28. FillParameterForVariant - Use this function to fill Name and Value of a Parameter which is of type Variant.

29. FillParameterForBSTR - Use this function to fill Name and Value of a Parameter which is of type BSTR.

30. FillParameterForVarChar - Use this function to fill Name and Value of a Parameter which is of type VarChar.

31. FillParameterForInteger - Use this function to fill Name and Value of a Parameter which is of type Integer.

32. FillParameterForBoolean - Use this function to fill Name and Value of a Parameter which is of type Boolean.

33. SetAttributeList – Sets the attribute name and value retrieved using the DBQuery and saves them in oAttributeList() which is an array of user defined datatype AttributeList.

34. MultipleInterfaceDataLookUp - This function does a complex query between 2 different interfaces that are on the SAME part, the interfaces must have the oid property in common.

35. sDestColName - the name of the column we are looking up.

36. sDestColInterface - the interface this column exists on (with the i removed of course.

37. sPartInterface - the interface of the part we are looking up.

38. sRefColName - the name of the column we are referencing.

39. sRefInterfance - the name of the interface that the reference column exists on.

40. sRefValue - the value we are using for comparison purposes. If the option sMaxRefValue is included then this is considered the Min value.

41. [sMaxRefValue] - the max value we are using for comparison purposes.

    Returns a Variant with the result of the query.
    Return the type of structure being connected to 35. GetSupportingTypes.

42. IsStructureSlopedAcrossPipe – This function takes the inputconfighlpr and structuretype as input and returns a True if structure is sloped across pipe otherwise returns a False.

43. GetSupportingPropertyByRule - Returns the value of the property if the property is found in the database for that section.

44. GetCableTrayCSDimensions - This function returns width and height of the selected cable tray.

45. GetPipeODwithoutInsulation - Function to get PipeOD without insulation - Takes dTotalOD as double oRoute as Object as Input and returns the PipeOD without insulation.

46. GetUserAttribute - Get the given User attribute from the given interface.

47. AddInput - Used in IJDUserSymbolServices_InitializeSymbolDefinition to add inputs.

48. AddOutput - Used in IJDUserSymbolServices_InitializeSymbolDefinition to add outputs.

49. SetupInputsAndOutputsEx – This function was some extension kind of code which in turn calls the SetupInputsAndOutputsForCache that modularize the code needed to generate the inputs and outputs that are required to develop a SP3D symbol.

HgrSymbolUtilFuncs:

1. GetAttributeFromObject – Takes SupportComponent object and AttributeName as inputs and returns the AttributeValue.

2. GetPortByNameFromCollection – Takes the PortName and PortElements as inputs and returns the HgrPort Object when a match is found for the input PortName.

basGeom3D:

1. GetOrthonormal – Need better definition and a sketch would be better

2. PlaceTrCString – This function creates transient (non-persistent) complex string curve based on collection of curves

3. PlaceTrBspline – This function creates transient (non-persistent) b-spline curve based on array of points, Array should consist of at least 4 points

4. PlaceCylinder – This function creates persistent projetion of circle based on two points (axis of cylinder) and diameter

5. PlaceCone – This function creates a persistent cone based on 2 points (axis of cone) and 2 diameters

6. PlaceRevolution – This function creates persistent revolution based on curve axis of revolution and angle

7. PlaceProjection – This function creates persistent projection based on curve axis of projection and height

8. PlaceTorus – This function creates persistent Torus based on center point and 2 diameters

9. PlaceSphere – This function creates persistent Sphere based on center point and radius

10. PlaceBox – This function creates a box, it takes two opposite box corners as input parameters

11. PlaceTrArcByCenterNorm – This function creates transient (non-persistent) arc by start, end, and center

12. PlaceTrArcByCenter – This function creates transient (non-persistent) arc by start, end, and center

13. PlaceTrArcBy3Pts – This function creates transient (non-persistent) arc by start, end, and point on

14. PlaceTrCStringNoCheck - this function creates transient (non-persistent) complex string curve

PF_HgrAssyUtil.bas:

1. AddPartsByClassName – This Function (normally called from IJHgrAssmInfo_GetAssemblyCatalogParts) simplifies the specification of the parts included. Adds the catalog part collection to the supportdefinition.

2. GetPipeRadiusDBU – Gets the PrimaryPipeDiameter and converts the default unit value to database value.

3. ConnectPartToRouteOrStruct - Return the collection of Route/struct connection information. Used in IJHgrAssmInfo_GetRouteConnections and IJHgrAssmInfo_GetStructConnections functions

4. GetUserDouble – Gets an Attributevalue which is of type double, using InputConfigHlpr's GetAttributeValue

5. GetUserString – Gets an Attributevalue which is of type String, using InputConfigHlpr's GetAttributeValue

6. GetProfileDim – Get CrossSection dimension of HgrBeam

7. GetHSSDim – Return the value of an attribute for a cross section definition

8. SetValuesOfSymbolInputsAndOccAttributes – Method to set any occurrence attribute on the symbol of the part.

9. CreateGenNote – Creates a NoteObject, takes the partobject as input

StructSupServiceUtils:

1. AddConnPortBetweenStructPorts - This function adds a connection port between 2 structure ports so that some support component can be positioined properly based on this connection port position.

2. GetStructPortNamesInOrder – This function is used to get structure port names in order so that, either from left to right or from rear to front based on input order type. Basically, it can be used to check relative orientation or position of 2 structures about the route.

3. GetMaxExternalPipeDiameter - This method retrieves the diameter of the pipe with max diameter among multi pipes.

4. GetMaxExternalPipeIndex - This method retrieved the index of the pipe with max diameter among multi pipes.

5. GetExternalPipeDiameter – This function can be deleted and it will be replaced by GetExternalPipeDiameter function in InputConfigHlpr.

6. GetBoundaryObjectDimension - This method retrieves the dimension of the boundary objects on the defined bounding box.

7. GetIsLugEndOffsetApplied - This method checks whether an offset value needs to be explicitly specified on both end of the leg. If the input structure is either a slab or a non-parallel member, then the offset needs to be specified. Else no offset value needs to be specified

8. Output: boolean array (1: LeftOffsetAppliedFlag
   1. 2: RightOffsetAppliedFlag)

9. GetBoundingBoxDimension - This method returns the dimension (Width and Height) of bounding box itself

10. GetPipeStructuralASMAttributes - This method gets all the occurrence attributes defined on the pipe structural support assemblies

11. GetDuctStructuralASMAttributes - This method gets all the occurrence attributes defined on the duct structural support assemblies

12. GetAngleCSDimension – This function also returns the cross section attributes of a HgrBeam

13. GetIndexedStructPortName - This method returns the port name on left and right side of the route object. (Left is on the negative Y axis of bounding box coord. sys., right is on the positive Y axis of the bounding box coord. sys.).

14. Outputs: string array
    1: Left structure reference port name
    2: right structure reference port name

15. GetLSupportConfiguration - This method retrieves the configuration of L shape strutural support

16. There are two configuration of the L shape support
    config 1: the top supporting member is on the right of the route object
    config 2: the top supporting member is on the left of the route object
17. This method also returns whether an offset is needed for a certain end and also determines the offset value,
    offset(1) defines the offset between BBX and vertical leg
    offset(2) defines the offset between BBX and the horizontal leg

18. CrossSectionAttributeValue – To get the cross section attributes of a HgrBeam

SP3D References Tool

The software consists of hundreds of type libraries that provide the programmatic interfaces to the data model and its underlying data. These libraries consist of the data model's interfaces and their methods and properties.

The ability to integrate user-definable components into the environment is a key capability of the software. The mechanism of creating custom commands provides this extensibility.
To reference the available type libraries in Visual Basic:

- Click **Project > References**.

To perform the task of referencing your type libraries more quickly and efficiently:

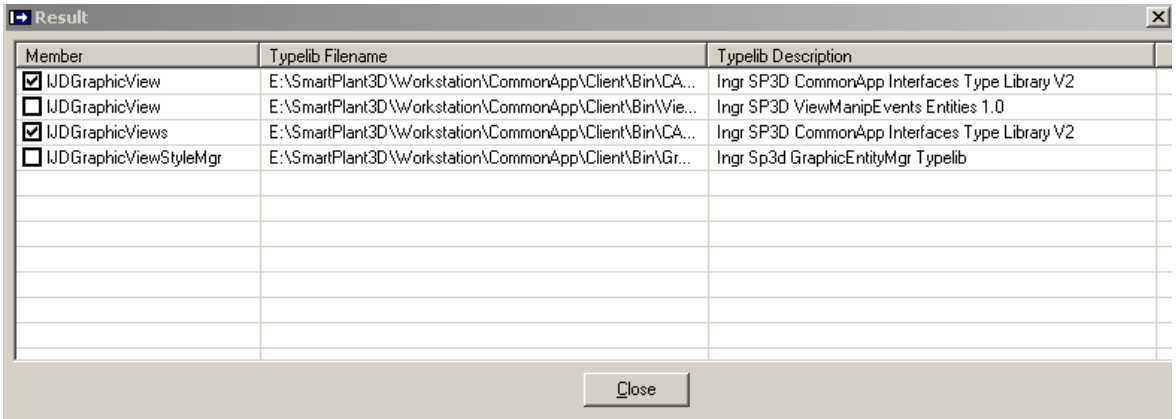
- Click **Project > SP3D References**.

Using the SP3D References Tool

The SP3D References tool is a very useful utility that you can use to locate and reference type libraries quickly and easily. You only need to know the name of your class object or variable in which to perform a search.

1. Open Visual Basic.
2. Click **Add-Ins > Add-In Manager**....
3. Select **SP3D References** and make sure that the **Loaded/Unloaded** and **Load on Startup** boxes under **Load Behavior** are both checked.
4. Click **OK**.
5. Click **Project > SP3D References** to invoke the dialog.

6. Enter a class or variable name to search..
7. Click **Find**.



8. Check the appropriate type libraries.

   Note: If this is the first time that you have invoked the tool, it begins reading your system to generate a data file that contains information about all existing registered type libraries.

Debugging Your Code

No matter how carefully you create your code, errors can occur. To handle these errors, you need to add error-handling code to your procedures.
You perform the process of locating and fixing bugs in applications by *debugging* the code.
Visual Basic provides several tools to help analyze how your application operates. These debugging tools are useful in locating the source of bugs, but you can also use the tools to experiment with changes to your application or to learn how other applications work.

Note: You must add the TaskHost project to the integrated development environment (IDE) before you can debug your Visual Basic project.

Before you can use the TaskHost project, you must set new paths in your computer's environment variables. Click Start -> Settings -> Control Panel -> System. Select the Advanced tab and then click Environment Variables. Finally add the following path statements according to the location in which you installed the software:

PATH=[*Product Directory*]\Core\Runtime; [*Product Directory*]\GeometryTopology\Runtime

Adding the TaskHost Project to your Project

1. Open your Visual Basic .vbp project to debug.
2. Click File > Add Project.
3. Select the Existing tab.
4. Open SP3DTaskHost.vbp in the following path: ..\Debug\Container\Src\Host
5. In the Project window, right-click over SP3DTaskHost and then select Set as Start Up.
6. Right-click again on SP3DTaskHost and then select SP3DtaskHost Properties...
7. On the Project Properties dialog, change the Project Type to Standard EXE.
8. Set the breakpoint in your project to debug.
9. Click Run and wait for processing to begin. Your Visual Basic project becomes active when the breakpoint is reached.
10. Click to view <your project>, which returns you back to the code view. Then step through your code.

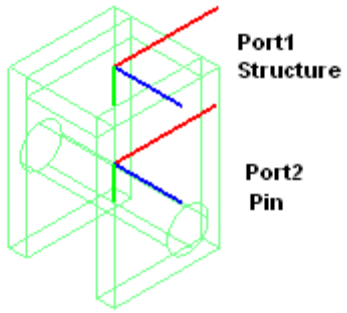
Important

Do not stop the debug process by clicking the End command. If you end processing this way, you will throw an exception, crash all the software that is running, and lose your changes.
To safely end processing, click File > Exit from the SmartPlant 3D TaskHost software.

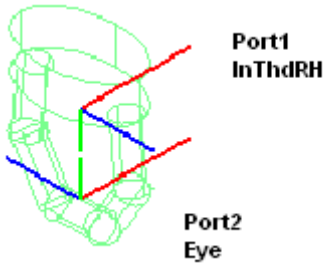
Sample - Load Rated Parts

Load Rated Parts Library

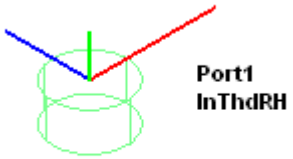
Load Rated Part FIG66N - Welded Beam Attachment (Connect to Bolt)



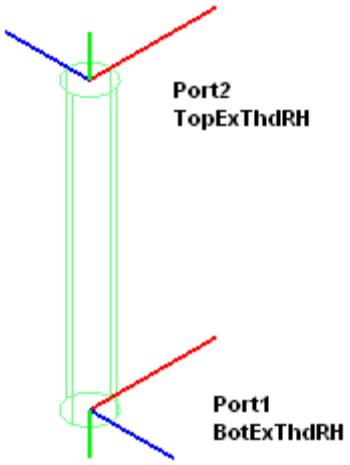
Load Rated Part FIG290N Weldless Eye Nut



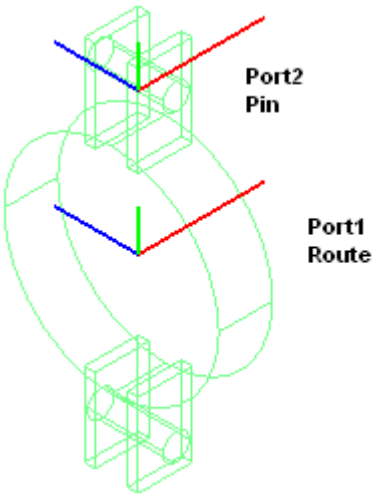
Standard Hex Nut



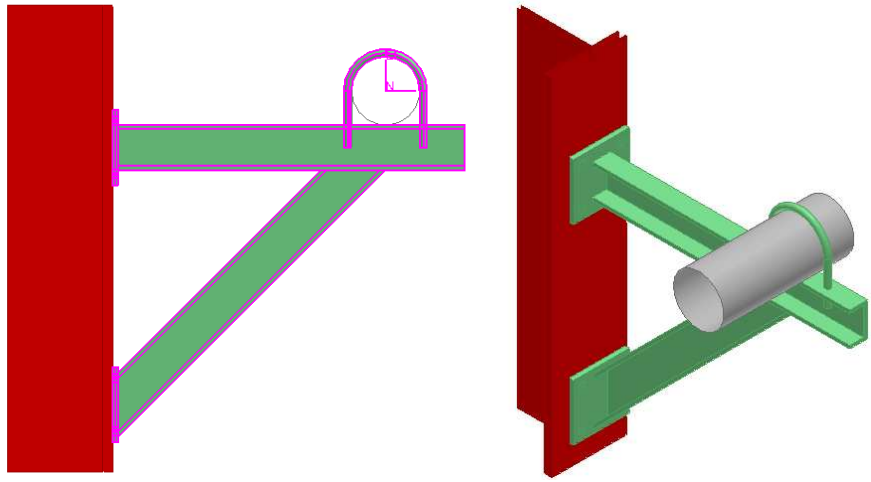
Load Rated Part FIG140N End Threaded Rod Right Hand Thread



Load Rated Part FIG212N Medium Pipe Clamp



Example: Braced 45 Deg Cantilever C3X6 w/ Baseplate-UBolt Assembly



```
'++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
'
'   Copyright (c) 2004, Intergraph Corporation. All rights reserved.
'
'   Name:           HS_RH_ASSY_BC
'   ProgID:         HS_RH_ASSY_BC.CBCantilever
'   Author:         Ramon Him
'   Creation Date:  2/2009
'
'   Description:
'   Change History:
'++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

Option Explicit
Private Const MODULE = "CBCantilever" 'Used for error messages

'Declare Global variables

Private Const nSection = 1
Private Const nBrace = 2
Private Const nConnObj1 = 3
Private Const TOP_PLATE = 4
Private Const ANGLE_PLATE = 5
Private Const WeldBox1 = 6
Private Const WeldBox2 = 7
Private Const WeldBox3 = 8
Private Const Ubolt = 9

Private OVERHANG As Double
Private BRACE_ANGLE As Double
Private SECTION_SIZE As String
Private SECTIONTYPE As String
Private SECTION_SIZE1 As String

Private BP_WIDTH As Double
Private BASEPLATE As String
Private BP_HOLE_SIZE As Double
Private BP_HOLE_INSET As Double
Private BP_THICKNESS As Double

Implements IJHgrAssmInfo

Private Function IJHgrAssmInfo_GetAssemblyCatalogParts(ByVal pDispInputConfigHlpr As Object) As Object
Const METHOD = "IJHgrAssmInfo_GetAssemblyCatalogParts"
        On Error GoTo ErrorHandler

    Dim my_IJHgrInputConfigHlpr As IJHgrInputConfigHlpr
    Set my_IJHgrInputConfigHlpr = pDispInputConfigHlpr

'<< Get Attributes on AIR
'HsSteel_C3x6

    Dim varTemp As Variant
    my_IJHgrInputConfigHlpr.GetAttributeValue "CSize", Nothing, varTemp
    SECTION_SIZE = Trim(varTemp)
    SECTION_SIZE1 = Mid(SECTION_SIZE, 9)

    my_IJHgrInputConfigHlpr.GetAttributeValue "OVERHANG", Nothing, varTemp
    OVERHANG = varTemp
    my_IJHgrInputConfigHlpr.GetAttributeValue "BRACE_ANGLE", Nothing, varTemp
```

```
    BRACE_ANGLE = varTemp

'GET BASE PLATE Dimensions from AIR
'=====================================
    my_IJHgrInputConfigHlpr.GetAttributeValue "BP_HOLE_SIZE", Nothing, varTemp
    BP_HOLE_SIZE = varTemp
    my_IJHgrInputConfigHlpr.GetAttributeValue "BP_HOLE_INSET", Nothing, varTemp
    BP_HOLE_INSET = varTemp
    my_IJHgrInputConfigHlpr.GetAttributeValue "BP_WIDTH", Nothing, varTemp
    BP_WIDTH = varTemp
    my_IJHgrInputConfigHlpr.GetAttributeValue "BP_THICKNESS", Nothing, varTemp
    BP_THICKNESS = varTemp


'=====================================

'<< Insert the list of part classes required by the type

    ReDim PartClasses(9) As String
    PartClasses(nSection) = Trim(SECTION_SIZE) ' C Section
    PartClasses(nBrace) = "Utility_CUTBACK_C2_1" 'C Section Utility_CUTBACK_C2_1
    PartClasses(nConnObj1) = "Log_Conn_Part_1" 'Rotational Connection Object
    PartClasses(TOP_PLATE) = "Utility_FOUR_HOLE_PLATE_5"
    PartClasses(ANGLE_PLATE) = "Utility_FOUR_HOLE_PLATE_5"
    PartClasses(WeldBox1) = "Utility_Weld_1"
    PartClasses(WeldBox2) = "Utility_Weld_1"
    PartClasses(WeldBox3) = "Utility_Weld_1"
    PartClasses(Ubolt) = "Utility_GEN_U_BOLT_1"

'<< Return the collection of Catalog Parts
    Set IJHgrAssmInfo_GetAssemblyCatalogParts = AddPartsByClassName(pDispInputConfigHlpr, PartClasses)
    Exit Function

ErrorHandler:
        Err.Raise LogError(Err, MODULE, METHOD, "").Number
End Function
Private Function IJHgrAssmInfo_GetAssemblyJoints(ByVal pDispInputConfigHlpr As Object, ByVal pDispPartOccCollection As Object) As
Object
Const METHOD = "IJHgrAssmInfo_GetAssemblyJoints"
        On Error GoTo ErrorHandler:


        'Get IJHgrInputConfig Hlpr Interface
        Dim my_IJHgrInputConfigHlpr As IJHgrInputConfigHlpr
        Set my_IJHgrInputConfigHlpr = pDispInputConfigHlpr

        Dim RouteIndex As Long
        RouteIndex = my_IJHgrInputConfigHlpr.RouteIndex
        Dim IJElements_PipeColl As IJElements
        my_IJHgrInputConfigHlpr.GetPipes IJElements_PipeColl

        Dim oRoute As Object
        Set oRoute = IJElements_PipeColl.Item(RouteIndex)

'<<Get Attributes on objects

    'Get Structure object
    Dim oStruct As Object
    Dim oStructColl As Object
    my_IJHgrInputConfigHlpr.GetSupportingCollection oStructColl
    Set oStruct = oStructColl.Item(1)

    Dim dPipeDia As Double
    dPipeDia = GetExternalPipeDiameter(my_IJHgrInputConfigHlpr, 1)

    Dim dPipeRadius As Double
    dPipeRadius = GetPipeODwithoutInsulat(dPipeDia, oRoute) / 2


    Dim dInsulat As Double
    dInsulat = GetPipeInsulat(oRoute)


'<< Get interface for accessing items on the collection of Part Occurences

        Dim IJElements_PartOccCollection As IJElements
        Set IJElements_PartOccCollection = pDispPartOccCollection

        'Resume normal error handling.
        On Error GoTo ErrorHandler:

        '=====================================================
        'Get the Current Location in the Route Connection Cycle
        '=====================================================
        Dim RouteConnectionValue As Integer
        RouteConnectionValue = my_IJHgrInputConfigHlpr.GetRouteConnectionValue

'Set note for dimension on Support PartOcc

        Dim oNote1 As IJGeneralNote
        Set oNote1 = my_IJHgrInputConfigHlpr.CreateNoteAndAssociateToPort("Dim1", IJElements_PartOccCollection.Item(1),
"BeginCap")
        oNote1.Text = "1 Dimension"
        oNote1.Purpose = 3 ' value 3 means fabrication
        oNote1.Dimensioned = True
```

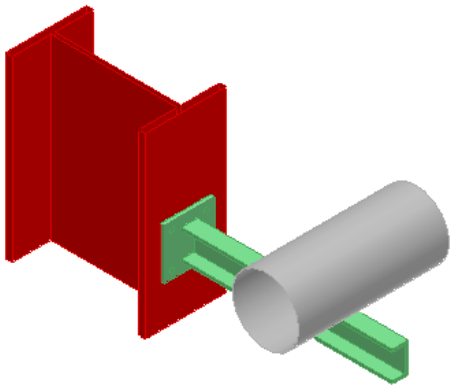
```
        Set oNote1 = my_IJHgrInputConfigHlpr.CreateNoteAndAssociateToPort("Dim2", IJElements_PartOccCollection.Item(1), "EndCap")
        oNote1.Text = "2 Dimension"
        oNote1.Purpose = 3 ' value 3 means fabrication
        oNote1.Dimensioned = True

        Set oNote1 = my_IJHgrInputConfigHlpr.CreateNoteAndAssociateToPort("Dim3", IJElements_PartOccCollection.Item(2),
"StartStructure")
        oNote1.Text = "Third Dimension"
        oNote1.Purpose = 3 ' value 3 means fabrication
        oNote1.Dimensioned = True

        Set oNote1 = my_IJHgrInputConfigHlpr.CreateNoteAndAssociateToPort("Dim4", IJElements_PartOccCollection.Item(5),
"TopStructure")
        oNote1.Text = "4 Dimension"
        oNote1.Purpose = 3 ' value 3 means fabrication
        oNote1.Dimensioned = True

        Set oNote1 = my_IJHgrInputConfigHlpr.CreateNoteAndAssociateToPort("Dim5", IJElements_PartOccCollection.Item(5),
"BotStructure")
        oNote1.Text = "5 Dimension"
        oNote1.Purpose = 3 ' value 3 means fabrication
        oNote1.Dimensioned = True


        Set oNote1 = my_IJHgrInputConfigHlpr.CreateNoteAndAssociateToPort("Dim6", IJElements_PartOccCollection.Item(9), "Route")
        oNote1.Text = "6 Dimension"
        oNote1.Purpose = 3 ' value 3 means fabrication
        oNote1.Dimensioned = True

        Set oNote1 = Nothing

' Get dimension from HgBeam

    Dim SteelWidth As Double
    Dim SteelDepth As Double
    Dim SteelThickness As Double
        SteelThickness = GetProfileDim(nSection, "Flange", IJElements_PartOccCollection) 'Width, Depth, Flange, Web as 2nd
argument
        SteelWidth = GetProfileDim(nSection, "Width", IJElements_PartOccCollection) 'Width, Depth, Flange, or Web as 2nd argument
        SteelDepth = GetProfileDim(nSection, "Depth", IJElements_PartOccCollection) 'Width, Depth, Flange, or Web as 2nd argument

    'get the distance between the structure and route ports
    Dim dDistPorts As Double
    Dim tempL As Double

    dDistPorts = my_IJHgrInputConfigHlpr.GetDistanceBetweenPorts("Route", "Structure", HORIZONTAL)
    tempL = dDistPorts

    Dim iStructConfIndex As Integer
    Dim BraceConfigIndex As Double

' Set the confIndex for the toggle route connection
        If RouteConnectionValue = 1 Then
                iStructConfIndex = 10404
                BraceConfigIndex = 9574
        Else
                iStructConfIndex = 2212
                BraceConfigIndex = 9574
        End If

'Set input to part symbols
'HgrSection
    my_IJHgrInputConfigHlpr.SetSymbolInputByName IJElements_PartOccCollection.Item(nBrace), "WIDTH", SteelWidth '
    my_IJHgrInputConfigHlpr.SetSymbolInputByName IJElements_PartOccCollection.Item(nBrace), "DEPTH", SteelDepth '
    my_IJHgrInputConfigHlpr.SetSymbolInputByName IJElements_PartOccCollection.Item(nBrace), "T_FLANGE", SteelThickness
    my_IJHgrInputConfigHlpr.SetSymbolInputByName IJElements_PartOccCollection.Item(nBrace), "T_WEB", SteelThickness

'Plates
    my_IJHgrInputConfigHlpr.SetSymbolInputByName IJElements_PartOccCollection.Item(TOP_PLATE), "WIDTH", BP_WIDTH ' 5 inches
    my_IJHgrInputConfigHlpr.SetSymbolInputByName IJElements_PartOccCollection.Item(TOP_PLATE), "DEPTH", BP_WIDTH ' 5 inches
    my_IJHgrInputConfigHlpr.SetSymbolInputByName IJElements_PartOccCollection.Item(TOP_PLATE), "C", BP_HOLE_INSET ' 3/4 inches
    my_IJHgrInputConfigHlpr.SetSymbolInputByName IJElements_PartOccCollection.Item(TOP_PLATE), "HOLE_SIZE", BP_HOLE_SIZE ' 9/16
inches bolt holes
    my_IJHgrInputConfigHlpr.SetSymbolInputByName IJElements_PartOccCollection.Item(TOP_PLATE), "THICKNESS", BP_THICKNESS

    my_IJHgrInputConfigHlpr.SetSymbolInputByName IJElements_PartOccCollection.Item(ANGLE_PLATE), "WIDTH", BP_WIDTH
    my_IJHgrInputConfigHlpr.SetSymbolInputByName IJElements_PartOccCollection.Item(ANGLE_PLATE), "DEPTH", BP_WIDTH
    my_IJHgrInputConfigHlpr.SetSymbolInputByName IJElements_PartOccCollection.Item(ANGLE_PLATE), "C", BP_HOLE_INSET
    my_IJHgrInputConfigHlpr.SetSymbolInputByName IJElements_PartOccCollection.Item(ANGLE_PLATE), "HOLE_SIZE", BP_HOLE_SIZE
    my_IJHgrInputConfigHlpr.SetSymbolInputByName IJElements_PartOccCollection.Item(ANGLE_PLATE), "THICKNESS", BP_THICKNESS


'ubolt
    my_IJHgrInputConfigHlpr.SetSymbolInputByName IJElements_PartOccCollection.Item(Ubolt), "D", 0.0127 '0.5 inches - diameter
ubolt
    my_IJHgrInputConfigHlpr.SetSymbolInputByName IJElements_PartOccCollection.Item(Ubolt), "R", dPipeDia / 2 + 0.0127 / 2 'bend
radius
    my_IJHgrInputConfigHlpr.SetSymbolInputByName IJElements_PartOccCollection.Item(Ubolt), "H", dPipeDia / 2 + SteelDepth / 2


'weld
        my_IJHgrInputConfigHlpr.SetSymbolInputByName IJElements_PartOccCollection.Item(6), "Thickness", 0.00025

        my_IJHgrInputConfigHlpr.SetSymbolInputByName IJElements_PartOccCollection.Item(7), "Thickness", 0.00025
```

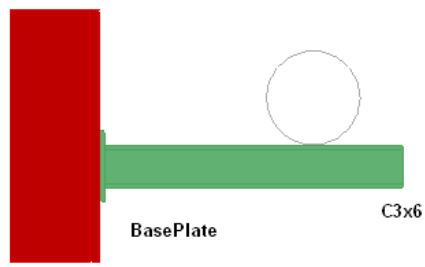
```
        my_IJHgrInputConfigHlpr.SetSymbolInputByName IJElements_PartOccCollection.Item(8), "Thickness", 0.00025


        Dim iArrayOfKeys(2) As Integer
        iArrayOfKeys(1) = nSection

        SetValuesOfSymbolInputsAndOccAttributes my_IJHgrInputConfigHlpr, IJElements_PartOccCollection, iArrayOfKeys,
"EndOverLength", 0
        SetValuesOfSymbolInputsAndOccAttributes my_IJHgrInputConfigHlpr, IJElements_PartOccCollection, iArrayOfKeys,
"BeginOverLength", 0
        SetValuesOfSymbolInputsAndOccAttributes my_IJHgrInputConfigHlpr, IJElements_PartOccCollection, iArrayOfKeys,
"Orientation", 0
        SetValuesOfSymbolInputsAndOccAttributes my_IJHgrInputConfigHlpr, IJElements_PartOccCollection, iArrayOfKeys,
"BeginMiter", "No End Treatment"
        SetValuesOfSymbolInputsAndOccAttributes my_IJHgrInputConfigHlpr, IJElements_PartOccCollection, iArrayOfKeys, "EndMiter",
"No End Treatment"


'<<Create Joints

        'Create a collection to hold the joints
        Dim JointCollection As New Collection
        'Create a Joint Factory
        Dim JointFactory As New HgrJointFactory
        Dim AssemblyJoint As Object

'TOP PLATE

        If RouteConnectionValue = 1 Then
            Set AssemblyJoint = JointFactory.MakeRigidJoint(-1, "Structure", TOP_PLATE, "TopStructure", 9380, BP_THICKNESS, 0, -
SteelDepth / 2 - dPipeRadius)
        Else
            Set AssemblyJoint = JointFactory.MakeRigidJoint(-1, "Structure", TOP_PLATE, "TopStructure", 9380, BP_THICKNESS, 0,
SteelDepth / 2 + dPipeRadius)
        End If
        JointCollection.Add AssemblyJoint

'Angle PLATE

        If RouteConnectionValue = 1 Then
            Set AssemblyJoint = JointFactory.MakeRigidJoint(nBrace, "StartStructure", ANGLE_PLATE, "TopStructure", 3252, -
((SteelDepth) / Sin(PF_HgrGeneralUtil.Rad(45))) / 2, 0, 0)
        Else
            Set AssemblyJoint = JointFactory.MakeRigidJoint(nBrace, "StartStructure", ANGLE_PLATE, "TopStructure", 3252, -
((SteelDepth) / Sin(PF_HgrGeneralUtil.Rad(45))) / 2, 0, 0)
        End If
        JointCollection.Add AssemblyJoint

'Weld

        If RouteConnectionValue = 1 Then
            Set AssemblyJoint = JointFactory.MakeRigidJoint(nConnObj1, "Connection", 6, "Other", 9444, -0.00025, 0, 0)
        Else
            Set AssemblyJoint = JointFactory.MakeRigidJoint(nConnObj1, "Connection", 6, "Other", 9444, -0.00025, 0, 0)
        End If
        JointCollection.Add AssemblyJoint

        If RouteConnectionValue = 1 Then
            Set AssemblyJoint = JointFactory.MakeRigidJoint(nSection, "EndCap", 7, "Other", 9444, -0.00025, 0, 0)
        Else
            Set AssemblyJoint = JointFactory.MakeRigidJoint(nSection, "EndCap", 7, "Other", 9444, -0.00025, 0, 0)
        End If
        JointCollection.Add AssemblyJoint

        If RouteConnectionValue = 1 Then
            Set AssemblyJoint = JointFactory.MakeRigidJoint(nBrace, "StartStructure", 8, "Other", 9444, 0, 0, 0)
        Else
            Set AssemblyJoint = JointFactory.MakeRigidJoint(nBrace, "StartStructure", 8, "Other", 9444, 0, 0, 0)
        End If
        JointCollection.Add AssemblyJoint


'  Horizontal brace length

        my_IJHgrInputConfigHlpr.SetSymbolInputByName IJElements_PartOccCollection.Item(nSection), "Length", tempL - BP_THICKNESS
+ dPipeRadius + OVERHANG
        If RouteConnectionValue = 1 Then
                Set AssemblyJoint = JointFactory.MakeRigidJoint(nSection, "EndCap", -1, "Structure", iStructConfIndex,
BP_THICKNESS, SteelDepth + dPipeRadius, SteelWidth / 2)
        Else
                Set AssemblyJoint = JointFactory.MakeRigidJoint(nSection, "EndCap", -1, "Structure", iStructConfIndex,
BP_THICKNESS, SteelDepth + dPipeRadius, SteelWidth / 2)
        End If
        JointCollection.Add AssemblyJoint

'  Angle brace length

        Dim dBraceL As Double
        dBraceL = (tempL - BP_THICKNESS - (SteelDepth / 2) / Sin(PF_HgrGeneralUtil.Rad(45))) / Cos(PF_HgrGeneralUtil.Rad(45))
        my_IJHgrInputConfigHlpr.SetSymbolInputByName IJElements_PartOccCollection.Item(nBrace), "L", dBraceL

'Add a Joint between horizontal section and the point that we want the angle to rotate around.
'Move the conn port to the where you want the cutback start
```

```
        Dim dPlaneOffset As Double
        dPlaneOffset = dDistPorts + dPipeRadius - BP_THICKNESS
        Set AssemblyJoint = JointFactory.MakeRigidJoint(nConnObj1, "Connection", nSection, "BeginCap", BraceConfigIndex, -
dPipeRadius - OVERHANG, 0)
        JointCollection.Add AssemblyJoint

       Set AssemblyJoint = JointFactory.MakeRigidJoint(nConnObj1, "Connection", nBrace, "EndStructure", 3309, SteelWidth / 2, 0,
0)
       JointCollection.Add AssemblyJoint

'Ubolt

        If RouteConnectionValue = 1 Then
                Set AssemblyJoint = JointFactory.MakeRigidJoint(-1, "Route", Ubolt, "Route", 2533, 0, 0)
        Else
                Set AssemblyJoint = JointFactory.MakeRigidJoint(-1, "Route", Ubolt, "Route", 10661, 0, 0)
        End If
        JointCollection.Add AssemblyJoint


        'Return the collection of Joints
        Set IJHgrAssmInfo_GetAssemblyJoints = JointCollection


' set the BOM for the cutback member

    Dim strBraceL As String
    Dim oUOMService As IJUomVBInterface
    Set oUOMService = New UnitsOfMeasureServicesLib.UomVBInterface
    oUOMService.SetDefaultUnits UNIT_DISTANCE, 61
    oUOMService.FormatUnit UNIT_DISTANCE, dBraceL, strBraceL

    my_IJHgrInputConfigHlpr.SetSymbolInputByName IJElements_PartOccCollection.Item(nBrace), "BOM_DESC", SECTION_SIZE1 & " ,
Length: " & strBraceL & " , Angle: " & Deg(BRACE_ANGLE)
    Set oUOMService = Nothing

    my_IJHgrInputConfigHlpr.SetSymbolInputByName IJElements_PartOccCollection.Item(Ubolt), "BOM_DESC1", "Light Weight U-Bolt with
two hex nuts, Galvanized Finish."

' Set all Weld parts to non-mto

    Dim Attrcol As IJDAttributesCol
    Dim oAttrs As IJDAttributes
    Dim oAttr As IJDAttribute
    Dim iWeldBoxIndx As Integer

    For iWeldBoxIndx = 6 To 8
        Set oAttrs = IJElements_PartOccCollection.Item(iWeldBoxIndx)
        Set Attrcol = oAttrs.CollectionOfAttributes("IJMTOInfo")
        Set oAttr = Attrcol.Item("ReportingRequirements")
        oAttr.Value = 2
        Set oAttr = Attrcol.Item("ReportingType")
        oAttr.Value = 2
    Next
    Set oAttr = Nothing
    Set oAttrs = Nothing
    Set Attrcol = Nothing


    Set IJElements_PartOccCollection = Nothing
    Set oRoute = Nothing


        Exit Function

ErrorHandler:
        Err.Raise LogError(Err, MODULE, METHOD, "").Number

End Function

Private Function IJHgrAssmInfo_GetMaxRouteConnectionValue() As Long

       IJHgrAssmInfo_GetMaxRouteConnectionValue = 2

End Function

Private Function IJHgrAssmInfo_GetRouteConnections(ByVal pDispInputConfigHlpr As Object) As Object
Const METHOD = "IJHgrAssmInfo_GetRouteConnections"
        On Error GoTo ErrorHandler

        Set IJHgrAssmInfo_GetRouteConnections = ConnectPartToRouteOrStruct(nSection, 1)   'partindex, routeindex
        Exit Function

ErrorHandler:
        Err.Raise LogError(Err, MODULE, METHOD, "").Number
End Function

Private Function IJHgrAssmInfo_GetStructConnections(ByVal pDispInputConfigHlpr As Object) As Object
Const METHOD = "IJHgrAssmInfo_GetStructConnections"
        On Error GoTo ErrorHandler

        Set IJHgrAssmInfo_GetStructConnections = ConnectPartToRouteOrStruct(nSection, 1) 'partindex, structindex
        Exit Function

ErrorHandler:
```

```
        Err.Raise LogError(Err, MODULE, METHOD, "").Number
End Function

Public Function GetPipeInsulat(oRoute As Object) As Double
Const METHOD = "GetPipeInsulat"

    On Error GoTo ErrorHandler

    Dim oInsulationObject As IJRteInsulation
    Set oInsulationObject = oRoute
    GetPipeInsulat = oInsulationObject.Thickness
    Set oInsulationObject = Nothing

Exit Function

ErrorHandler:
    Err.Raise LogError(Err, MODULE, METHOD, "").Number
End Function

Public Function GetPipeODwithoutInsulat(dTotalOD As Double, oRoute As Object) As Double
    Dim oInsulationObject As IJRteInsulation
    Set oInsulationObject = oRoute

    Dim dInsulat As Double
    dInsulat = oInsulationObject.Thickness
    GetPipeODwithoutInsulat = dTotalOD - 2 * dInsulat
    Set oInsulationObject = Nothing
End Function
```

Example: Unbraced Cantilever C3X6 w/ Baseplate Assembly






```
'+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
'
'   Copyright (c) 2004, Intergraph Corporation. All rights reserved.
'
'   Name:           HS_RH_ASSY_FRAME
'   ProgID:         HS_RH_ASSY_FRAME.CSingleFrame
'   Author:         Ramon Him
'   Creation Date:  2/2009
'
'   Description:
'   Change History:
'+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

Option Explicit
Private Const MODULE = "CSingleFrame" 'Used for error messages

'Declare Global variables

Private Const nSection = 1
Private Const TOP_PLATE = 2

Private OVERHANG As Double
Private SECTION_SIZE As String
Private SECTION_SIZE1 As String

Private PLATE_SIZE As String
Private BP_WIDTH As Double
Private BP_HOLE_SIZE As Double
Private BP_HOLE_INSET As Double
Private BP_THICKNESS As Double

Implements IJHgrAssmInfo

Private Function IJHgrAssmInfo_GetAssemblyCatalogParts(ByVal pDispInputConfigHlpr As Object) As Object
Const METHOD = "IJHgrAssmInfo_GetAssemblyCatalogParts"
        On Error GoTo ErrorHandler

    Dim my_IJHgrInputConfigHlpr As IJHgrInputConfigHlpr
    Set my_IJHgrInputConfigHlpr = pDispInputConfigHlpr

'<< Get Attributes on AIR
'HsSteel_C3x6

    Dim varTemp As Variant
    my_IJHgrInputConfigHlpr.GetAttributeValue "CSize", Nothing, varTemp
    SECTION_SIZE = Trim(varTemp)
    SECTION_SIZE1 = Mid(SECTION_SIZE, 9)

    my_IJHgrInputConfigHlpr.GetAttributeValue "OVERHANG", Nothing, varTemp
    OVERHANG = varTemp
    my_IJHgrInputConfigHlpr.GetAttributeValue "PLATE_SIZE", Nothing, varTemp
    PLATE_SIZE = varTemp
    my_IJHgrInputConfigHlpr.GetAttributeValue "BP_HOLE_SIZE", Nothing, varTemp
    BP_HOLE_SIZE = varTemp
    my_IJHgrInputConfigHlpr.GetAttributeValue "BP_HOLE_INSET", Nothing, varTemp
    BP_HOLE_INSET = varTemp
    my_IJHgrInputConfigHlpr.GetAttributeValue "BP_WIDTH", Nothing, varTemp
    BP_WIDTH = varTemp
    my_IJHgrInputConfigHlpr.GetAttributeValue "BP_THICKNESS", Nothing, varTemp
    BP_THICKNESS = varTemp

'<< Insert the list of part classes required by the type

    ReDim PartClasses(2) As String
    PartClasses(nSection) = Trim(SECTION_SIZE) 'C Section
    PartClasses(TOP_PLATE) = "Utility_FOUR_HOLE_PLATE_" + PLATE_SIZE
```

```
'<< Return the collection of Catalog Parts
    Set IJHgrAssmInfo_GetAssemblyCatalogParts = AddPartsByClassName(pDispInputConfigHlpr, PartClasses)
    Exit Function

ErrorHandler:
        Err.Raise LogError(Err, MODULE, METHOD, "").Number
End Function
Private Function IJHgrAssmInfo_GetAssemblyJoints(ByVal pDispInputConfigHlpr As Object, ByVal pDispPartOccCollection As Object) As
Object
Const METHOD = "IJHgrAssmInfo_GetAssemblyJoints"
        On Error GoTo ErrorHandler:

        'Get IJHgrInputConfig Hlpr Interface
        Dim my_IJHgrInputConfigHlpr As IJHgrInputConfigHlpr
        Set my_IJHgrInputConfigHlpr = pDispInputConfigHlpr

        Dim RouteIndex As Long
        RouteIndex = my_IJHgrInputConfigHlpr.RouteIndex
        Dim IJElements_PipeColl As IJElements
        my_IJHgrInputConfigHlpr.GetPipes IJElements_PipeColl

        Dim oRoute As Object
        Set oRoute = IJElements_PipeColl.Item(RouteIndex)

'<<Get Attributes on objects

    Dim dPipeDia As Double
    dPipeDia = GetExternalPipeDiameter(my_IJHgrInputConfigHlpr, 1)

    Dim dPipeRadius As Double
    dPipeRadius = GetPipeODwithoutInsulat(dPipeDia, oRoute) / 2

    Dim dInsulat As Double
    dInsulat = GetPipeInsulat(oRoute)

        '==========================
        '1. Load standard bounding box definition
        '==========================
        Dim my_IJHgrBBXHlpr As IJHgrBBXHlpr
        Set my_IJHgrBBXHlpr = pDispInputConfigHlpr
        my_IJHgrBBXHlpr.LoadBoundingBox
        '====== ======
        '2. retrieve dimension of the bounding box
        '====== ======
        ' Get route box geometry
        '
        '  ____________________
        ' |                    |
        ' |  ROUTE BOX BOUND   | dHeight
        ' |____________________|
        '        dWidth
        '
        Dim dimension() As Double
        Dim dWidth As Double, dHeight As Double

        dimension = GetBoundingBoxDimension(pDispInputConfigHlpr)
        dWidth = dimension(1)
        dHeight = dimension(2)

        Dim IJElements_PartOccCollection As IJElements
        Set IJElements_PartOccCollection = pDispPartOccCollection

        'Resume normal error handling.
        On Error GoTo ErrorHandler:

        '=====================================================
        'Get the Current Location in the Route Connection Cycle
        '=====================================================
        Dim RouteConnectionValue As Integer
        RouteConnectionValue = my_IJHgrInputConfigHlpr.GetRouteConnectionValue

' Get dimension from HgBeam

    Dim SteelWidth As Double
    Dim SteelDepth As Double
    Dim SteelThickness As Double
    SteelThickness = GetProfileDim(nSection, "Flange", IJElements_PartOccCollection)
    SteelWidth = GetProfileDim(nSection, "Width", IJElements_PartOccCollection)
    SteelDepth = GetProfileDim(nSection, "Depth", IJElements_PartOccCollection)

'Get the distance between the structure and route ports
    Dim dDistPorts As Double
    Dim tempL As Double

    dDistPorts = my_IJHgrInputConfigHlpr.GetDistanceBetweenPorts("BBR_Low", "Structure", HORIZONTAL)
    tempL = dDistPorts

' Set the confIndex for the toggle route connection
    Dim iStructConfIndex As Integer

        If RouteConnectionValue = 1 Then
                iStructConfIndex = 10404
        ElseIf RouteConnectionValue = 2 Then
                iStructConfIndex = 2212
```

```
        ElseIf RouteConnectionValue = 3 Then
                iStructConfIndex = 2212
        Else
                iStructConfIndex = 10404
        End If

'<< Set Attributes on Part Occurrences

'Plates
    my_IJHgrInputConfigHlpr.SetSymbolInputByName IJElements_PartOccCollection.Item(TOP_PLATE), "WIDTH", BP_WIDTH ' 5 inches
    my_IJHgrInputConfigHlpr.SetSymbolInputByName IJElements_PartOccCollection.Item(TOP_PLATE), "DEPTH", BP_WIDTH ' 5 inches
    my_IJHgrInputConfigHlpr.SetSymbolInputByName IJElements_PartOccCollection.Item(TOP_PLATE), "C", BP_HOLE_INSET ' 3/4 inches
    my_IJHgrInputConfigHlpr.SetSymbolInputByName IJElements_PartOccCollection.Item(TOP_PLATE), "HOLE_SIZE", BP_HOLE_SIZE ' 9/16
inches bolt holes
    my_IJHgrInputConfigHlpr.SetSymbolInputByName IJElements_PartOccCollection.Item(TOP_PLATE), "THICKNESS", BP_THICKNESS

    Dim iArrayOfKeys(2) As Integer
    iArrayOfKeys(1) = nSection

SetValuesOfSymbolInputsAndOccAttributes my_IJHgrInputConfigHlpr, IJElements_PartOccCollection, iArrayOfKeys, "EndOverLength", 0
SetValuesOfSymbolInputsAndOccAttributes my_IJHgrInputConfigHlpr, IJElements_PartOccCollection, iArrayOfKeys, "BeginOverLength", 0
SetValuesOfSymbolInputsAndOccAttributes my_IJHgrInputConfigHlpr, IJElements_PartOccCollection, iArrayOfKeys, "Orientation", 0
SetValuesOfSymbolInputsAndOccAttributes my_IJHgrInputConfigHlpr, IJElements_PartOccCollection, iArrayOfKeys, "BeginMiter", "No
End Treatment"
SetValuesOfSymbolInputsAndOccAttributes my_IJHgrInputConfigHlpr, IJElements_PartOccCollection, iArrayOfKeys, "EndMiter", "No End
Treatment"

'<<Create Joints

        'Create a collection to hold the joints
        Dim JointCollection As New Collection
        'Create a Joint Factory
        Dim JointFactory As New HgrJointFactory
        Dim AssemblyJoint As Object

'TOP PLATE

If RouteConnectionValue = 1 Then
Set AssemblyJoint = JointFactory.MakeRigidJoint(-1, "Structure", TOP_PLATE, "TopStructure", 9380, BP_THICKNESS, 0, -SteelDepth /
2 - dPipeRadius)
ElseIf RouteConnectionValue = 2 Then
Set AssemblyJoint = JointFactory.MakeRigidJoint(-1, "Structure", TOP_PLATE, "TopStructure", 9380, BP_THICKNESS, 0, -SteelDepth /
2 - dPipeRadius)
ElseIf RouteConnectionValue = 3 Then
Set AssemblyJoint = JointFactory.MakeRigidJoint(-1, "Structure", TOP_PLATE, "TopStructure", 9380, BP_THICKNESS, 0, SteelDepth / 2
+ dPipeRadius)
Else
Set AssemblyJoint = JointFactory.MakeRigidJoint(-1, "Structure", TOP_PLATE, "TopStructure", 9380, BP_THICKNESS, 0, SteelDepth / 2
+ dPipeRadius)
End If
JointCollection.Add AssemblyJoint

'Horizontal Brace length

my_IJHgrInputConfigHlpr.SetSymbolInputByName IJElements_PartOccCollection.Item(nSection), "Length", tempL - BP_THICKNESS +
OVERHANG

If RouteConnectionValue = 1 Then
Set AssemblyJoint = JointFactory.MakeRigidJoint(nSection, "BeginCap", -1, "Structure", iStructConfIndex, tempL + OVERHANG,
SteelDepth + dPipeRadius, SteelWidth / 2)
ElseIf RouteConnectionValue = 2 Then
Set AssemblyJoint = JointFactory.MakeRigidJoint(nSection, "BeginCap", -1, "Structure", iStructConfIndex, tempL + OVERHANG, -
dPipeRadius, SteelWidth / 2)
ElseIf RouteConnectionValue = 3 Then
Set AssemblyJoint = JointFactory.MakeRigidJoint(nSection, "BeginCap", -1, "Structure", iStructConfIndex, tempL + OVERHANG,
SteelDepth + dPipeRadius, SteelWidth / 2)
Else
Set AssemblyJoint = JointFactory.MakeRigidJoint(nSection, "BeginCap", -1, "Structure", iStructConfIndex, tempL + OVERHANG, -
dPipeRadius, SteelWidth / 2)
End If
JointCollection.Add AssemblyJoint

        'Return the collection of Joints
        Set IJHgrAssmInfo_GetAssemblyJoints = JointCollection

        Exit Function

ErrorHandler:
        Err.Raise LogError(Err, MODULE, METHOD, "").Number

End Function

Private Function IJHgrAssmInfo_GetMaxRouteConnectionValue() As Long

        IJHgrAssmInfo_GetMaxRouteConnectionValue = 4

End Function

Private Function IJHgrAssmInfo_GetRouteConnections(ByVal pDispInputConfigHlpr As Object) As Object
Const METHOD = "IJHgrAssmInfo_GetRouteConnections"
        On Error GoTo ErrorHandler

        Set IJHgrAssmInfo_GetRouteConnections = ConnectPartToRouteOrStruct(nSection, 1)   'partindex, routeindex
        Exit Function
```

```
ErrorHandler:
        Err.Raise LogError(Err, MODULE, METHOD, "").Number
End Function

Private Function IJHgrAssmInfo_GetStructConnections(ByVal pDispInputConfigHlpr As Object) As Object
Const METHOD = "IJHgrAssmInfo_GetStructConnections"
        On Error GoTo ErrorHandler

        Set IJHgrAssmInfo_GetStructConnections = ConnectPartToRouteOrStruct(nSection, 1) 'partindex, structindex
        Exit Function

ErrorHandler:
        Err.Raise LogError(Err, MODULE, METHOD, "").Number
End Function

Public Function GetPipeInsulat(oRoute As Object) As Double
Const METHOD = "GetPipeInsulat"

    On Error GoTo ErrorHandler

    Dim oInsulationObject As IJRteInsulation
    Set oInsulationObject = oRoute
    GetPipeInsulat = oInsulationObject.Thickness
    Set oInsulationObject = Nothing

Exit Function

ErrorHandler:
    Err.Raise LogError(Err, MODULE, METHOD, "").Number
End Function

Public Function GetPipeODwithoutInsulat(dTotalOD As Double, oRoute As Object) As Double
    Dim oInsulationObject As IJRteInsulation
    Set oInsulationObject = oRoute

    Dim dInsulat As Double
    dInsulat = oInsulationObject.Thickness
    GetPipeODwithoutInsulat = dTotalOD - 2 * dInsulat
    Set oInsulationObject = Nothing
End Function
```