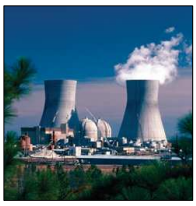
Process, Power and Marine Division

SmartPlant 3D Programming I

Ramon Him
Rhim@ingr.com
August 2006


INTERGRAPH

What will be covered in this training?

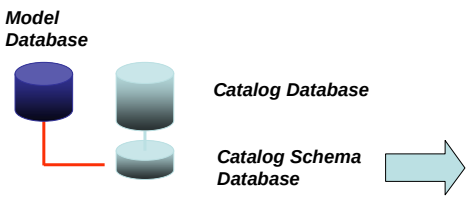
SmartPlant 3D

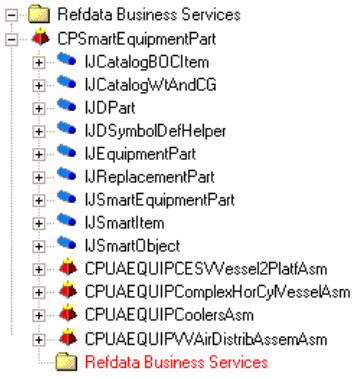
Overview:

- Understanding the SP3D data Model
- Introduction of the Schema Browser Tool
- Naming Rules
- Attribute Helper services
- Relation Helper services
- Symbol Definition for Route Objects
- SmartPlant 3D Tools
  - Symbol Wizard
  - Reference Tool
- Symbol Helper Components/Services
- Geometry Factories
- Symbol Deployment and Guidelines
  - Changing the symbol definition
  - Synchronization between model & catalog
- Debugging your code

SmartPlant 3D Data Model

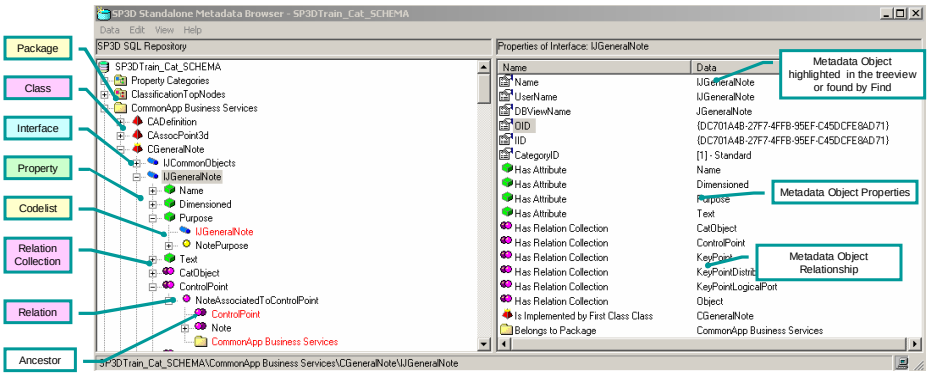
Use the Schema Browser Tool to view the data model





3

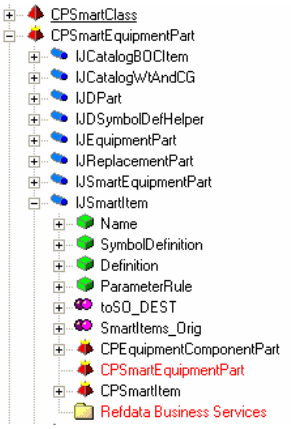
Schema Browser Tool



4

Schema Browser Tool

- A regular folder represents a package i.e. metadata of a service
- A red pyramid represents a class. A red pyramid overlayed by a yellow crown represents a first class, i.e. a class that is a leaf in the BOC (Business Objects Classification) hierarchy
- A green box represents a property.

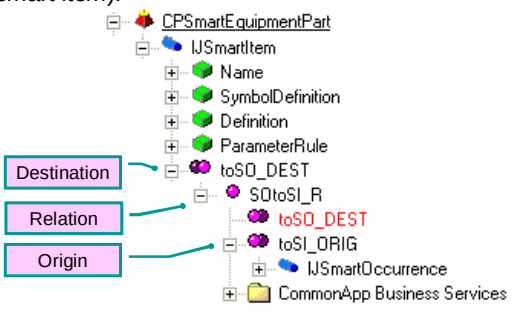


Schema Browser Tool

Relationships

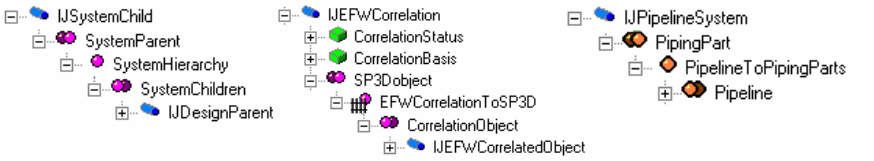
- Relations are binary
- A relation is between two and only two entities.
- These entities are known as **origin** and **destination** of the relation

Example: SOtoSI relationship (There is a relation between smart occurrence and the smart item).



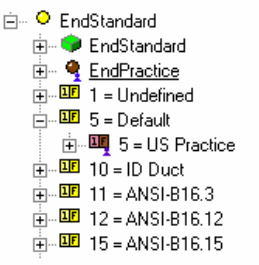
Schema Browser Tool

- A single magenta sphere represents a relationship. A single magenta sphere overlayed by a fence represents a private relationship.
- Double magenta/purple spheres represent a relationship collection (a role).
- An orange diamond represents an Edge, i.e a virtual relationship that "bridges" several relationships.
- Double Orange diamonds represent an edge collection (a role).



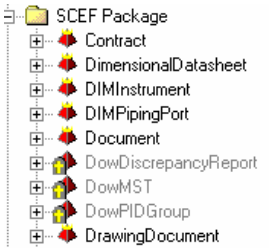
Schema Browser Tool

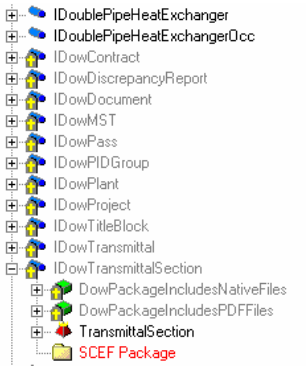
- A yellow sphere represents a codelist table.
- A yellow domino with a number and a letter represents a codelist value.
- A brown sphere overlayed by a blue human outline represents a parent codelist table.
- A brown domino with a number and a letter overlayed by a blue human outline represents a parent codelist value.



Schema Browser Tool

- Any icon overlayed by a tombstone represents a metadata object marked as deleted.

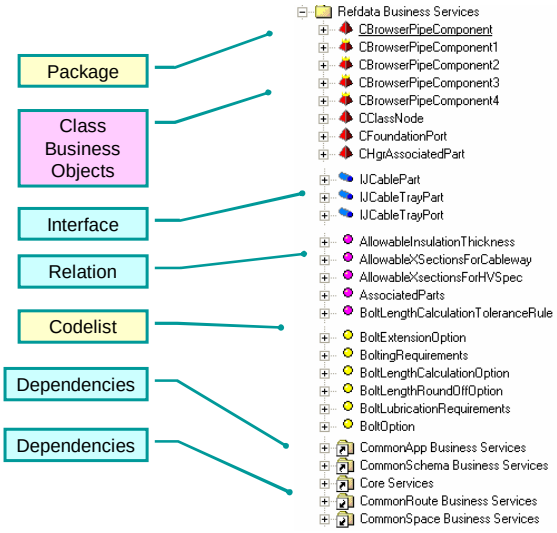




Schema Browser Tool

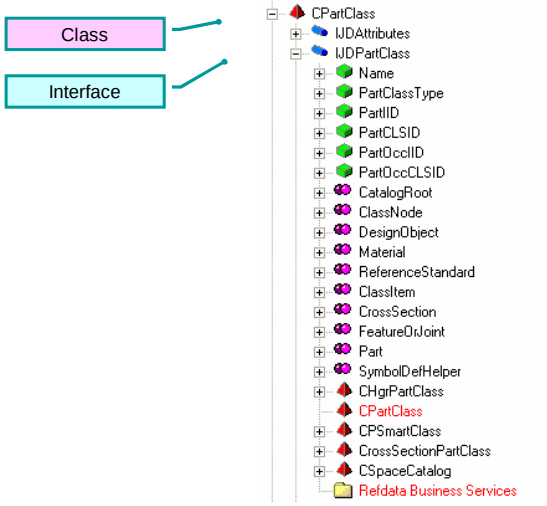
- Under View -> Units of measure, you can see the units for each Unit type. For each Unit type, the blue unit represents the (SI) Database unit (DBU). Right click on any unit allow you to convert a value in DBU.
- Under view -> Enumerations, you can know the different values of all the enumerations used by the metadata.
- If you re-open often the same schema, you should choose "Auto open open DB dialog" in Options.
- You can switch from displaying metadata object internal Names (System) to Usernames (User).

Reference Data Model



Part Class Entity Data Model

Part Classes



SQL Basics

Relational Databases are made of:

- Tables
  - Where the data are persisted. Tables are made of columns (the properties) and rows (the occurrences or records).
- Views
  - A predefined selection of a subset (or a superset) of columns and/or a subset of rows from one or more table(s).
- Relations
  - They are not “hard-coded”, they are established on request by performing a join (usually equality of values of identical columns of different tables).

SQL Basics

Query Language

```
SELECT t1.col1, t2.col2
    FROM  t1 JOIN  t2  ON  (t1.col2 = t2.col4)
    WHERE t1.col1>200
    ORDER by t1.col1
```

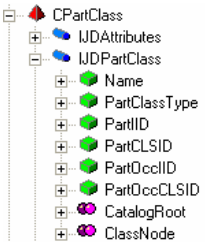
- SELECT: means list, return, give me…
  - List of columns from which data must to be listed. It defines a subset of the columns to be returned. * means All.
- FROM: defines the view (or starting point view) from which the columns and data must be listed. If a join is specified, a superset of the columns of the view is accessible.
  - JOIN: the way to navigate relations, defines a second view to be joined.
    - ON: specifies the condition of the join.
- WHERE: criteria to establish a filter (subset) of the rows to be listed
- ORDER BY: sorting criteria

Part Class Entity Data Model

Part Classes

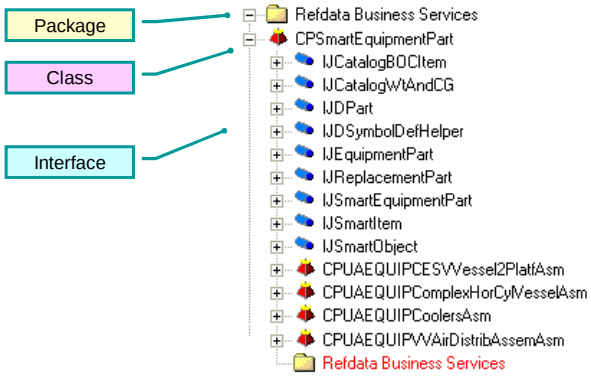
select count(*),
PartClassType
from JDPartClass
group by PartClassType

select * from
JDPartClass
where PartClassType = 'EquipmentAssemblyClass'



Equipment Reference Entity Data Model

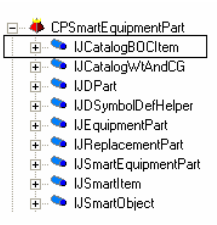
CPSmartEquipmentPart

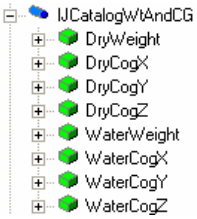


Equipment Reference Entity Data Model

Common interfaces

- The IJCatalogBOCItem interface allows the Business Object to participate in the Catalog BOC (Business Objects Classification) hierarchy.
- The IJCatalogWtAndCG interface provides methods to get the center of gravity and properties relating to an object's weight and center of gravity.



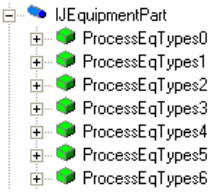


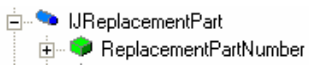
17

Equipment Reference Entity Data Model

Common interfaces

- The IJEquipmentPart interface provides code-listed properties to describe the Equipment Classification Type for all equipment parts.
- The IJReplacementPart interface provides a property to get the equipment part number when the mirror behavior option is set to Replacement required.



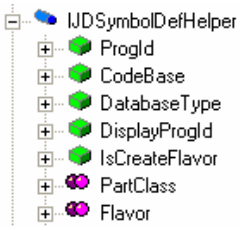


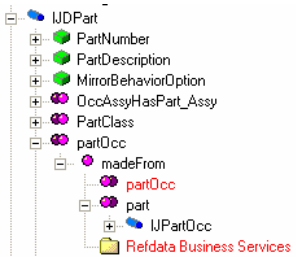
18

Equipment Reference Entity Data Model

Common interfaces

- The IJDSymbolDefHelper interface provides methods to assists in the definition of the cached symbol and properties relating to the symbol definition.
- The IJDPart interface is a generic interface that permits obtaining information about a part or part occurrence.

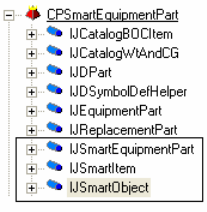


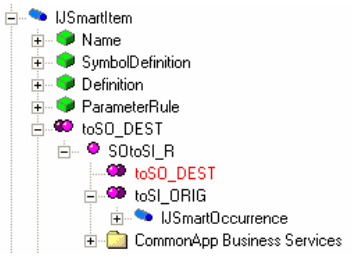


Equipment Reference Entity Data Model

Common interfaces

- The IJSmartItem interface provides properties relating to the smart item and permits obtaining information about a smart item or smart occurrence.
- The IJSmartObject interface permits obtaining information about a smart class.
- The JSmartEquipmentPart interface provides methods to obtain information about Smart Equipment.





Equipment Reference Entity Data Model

Common Properties

- IJSmartItem
  - Name
  - SymbolDefinition
  - Definition
  - ParameterRule
- IJDPart
  - PartNumber
  - PartDescription
  - MirrorBehaviorOption
- IJReplacementPart
  - ReplacementPartNumber
- IJEquipmentPart
  - ProcessEqTypes0
  - ProcessEqTypes1
  - ProcessEqTypes2
  - ProcessEqTypes3
  - ProcessEqTypes4
  - ProcessEqTypes5
  - ProcessEqTypes6
- IJCatalogWtAndCG
  - DryWeight
  - DryCogX
  - DryCogY
  - DryCogZ
  - WaterWeight
  - WaterCogX
  - WaterCogY
  - WaterCogZ

Custom Properties

- IJUABoilerSpecs
  - BodyDiameter
  - BodyLength
  - MaxPressure
  - MaxStress
  - InspectionOpenings

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

Equipment Reference Entity Data Model

CPEquipmentComponentPart

Class

Interface

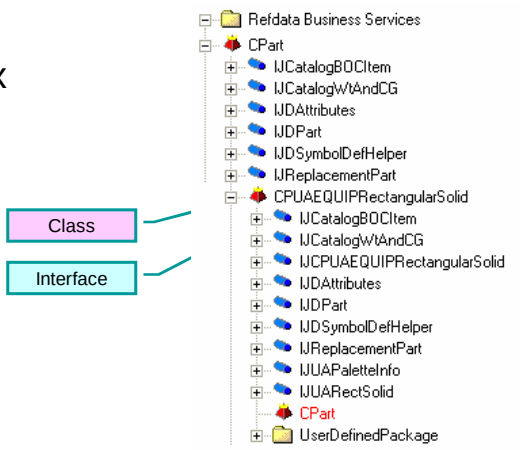
- Refdata Business Services
- CPEquipmentComponentPart
  - IJCatalogBOCItem
  - IJCatalogWtAndCG
  - IJDAttributes
  - IJDPart
  - IJDSymbolDefHelper
  - IJEquipmentComponentPart
  - IJReplacementPart
  - IJSmartItem
  - IJSmartObject
  - CPUAEQUIPE205_CVerCylEqpSkCompAsm
  - CPUAEQUIPE210_SVerCylSkirtCompAsm
  - CPUAEQUIPE215_SimVCyEqLegCompAsm
  - CPUAEQUIPE230_SphericalVesselCompAsm
  - CPUAEQUIPVesselSkirtAsm
  - Refdata Business Services

Equipment Reference Entity Data Model

SmartPlant 3D

Equipment Shapes

CPUAEQUIPxxxxxx



Equipment Reference Entity Data Model

SmartPlant 3D

Equipment Shapes

	partnumber	partdescription	SequenceNumber
1	RtCircularCylinder 001	Cylinder	1
2	RtCircularCone 001	Cone	2
3	EccentricCone 001	EccentricCone	3
4	Sphere 001	Sphere	4
5	SemiEllipticalHead 001	SemiEllipticalHead	5
6	RectangularSolid 001	RectangularSolid	6
7	TriangularSolid 001	TriangularSolid	7
8	CircularTori 001	Tori	8
9	RectangularTorus 001	RectangularTorus	9
10	EccentricRectangularPrism 001	EccentricRectangularPrism	10
11	TransitionElement 001	TransitionElement	11
12	TruncatedRectangularPrism 001	TruncatedRectangularPrism	12
13	EccentricTransitionElement 001	EccentricTransitionElement	13
14	Platform1 001	Platform	14
15	Platform2 001	Platform	15
16	OctogonalSolid 001	OctogonalSolid	16
17	HexagonalSolid 001	HexagonalSolid	17
18	DatumShape 001	Datum Shape	18

```
select
partnumber,
partdescription,
SequenceNumber
from JDPart x1
join JUAPaletteInfo x2 on x2.oid = x1.oid
order by SequenceNumber
```

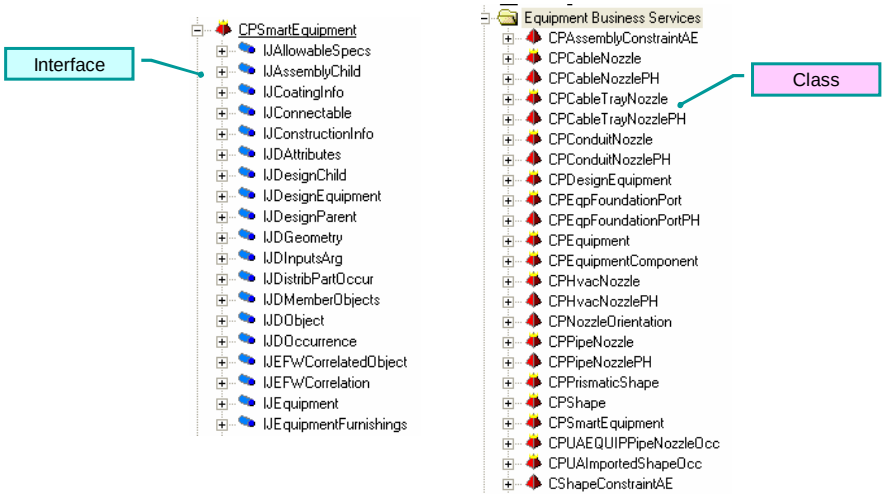
Equipment Reference Entity Data Model

Equipment Shapes

	partnumber	partdescription
1	SKETCH_CROSS_SECTION_01	Circle
2	SKETCH_CROSS_SECTION_02	Rectangle
3	SKETCH_CROSS_SECTION_03	Triangle
4	SKETCH_CROSS_SECTION_04	Ellipse
5	SKETCH_CROSS_SECTION_05	Hexagon
6	SKETCH_CROSS_SECTION_06	Sector
7	SKETCH_CROSS_SECTION_07	Road
8	SKETCH_CROSS_SECTION_08	TrapezeR
9	SKETCH_CROSS_SECTION_09	TrapezeC

```
select
partnumber,
partdescription
from
JDPart x1
join XPartClassContainsParts x2 on x2.oiddestination = x1.oid
join JDPartClass x3 on x3.oid = x2.oidorigin
where x3.PartClassType like '%SketchCrossSectionsClass%'
```


Equipment Entity Data Model



Equipment Entity Data Model

SmartPlant 3D

Generic Object interfaces

- The IJDObject interface is a required interface for almost all objects. This interface provides access to the permission group for the object, the status, name of the user who created and last modified the object and the date and time for creation and last modification.
- The IJDAttributes interface is required for the equipment object to support user-defined attributes.
- The IJNamedItem interface provides the name property, a type string and an icon for all named objects.

Equipment Entity Data Model

SmartPlant 3D

Display/Locate interfaces

- IJGeometry: This interface is a required if the BO has geometry.
- IJGraphicEntity: This interface is a required if the BO is displayable.
- IJModelLocate: This interface is used to allow client components to graphically locate all of the graphical components of a standard equipment object.
- IJOccurrence: This interface provides a property that allows the matrix to be retrieved or set. A matrix defines the origin and orientation of an equipment object.

Equipment Entity Data Model

SmartPlant 3D

Equipment specific interfaces

- IJEquipment: This interface is used to provide a common interface for both catalog equipment and design equipment. This interface also provides methods to get and set the position and orientation of the equipment.
- IJEquipmentFurnishings: This interface is used to type all business objects that are required to show up under the Equipment&Furnishings node.

Equipment Entity Data Model

SmartPlant 3D

- IJWeightCG: This interface provides both the wet and dry weights and center of gravity for an object.
- IJFoulCheck: The CPSmartEquipment class implements the IJFoulCheck interface which is used to identify which objects should be checked for interferences with other object.
- IJSystemChild: In order for an object to participate in the System Hierarchy, it must implement either IJSystemChild and establish a relationship to a design parent.
- IJMfgParent and IJAssemblyChild: Equipment Assembly objects can also participate in the Assembly hierarchy. As a part, it can be associated to an Assembly parent object using the Assembly relationship that is established between IJMfgParent and IJAssemblyChild. This relationship allows the equipment part to be associated to one assembly parent object.

Equipment Entity Data Model

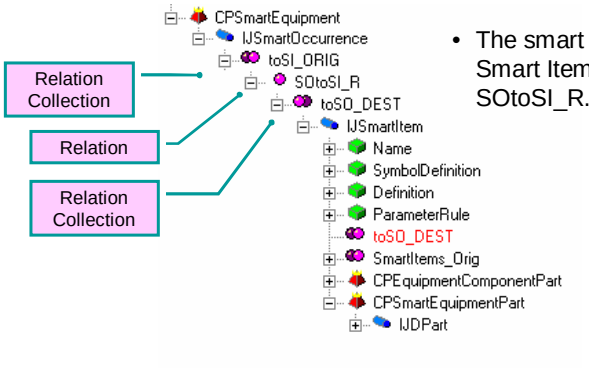
- IJCoatingInfo: This interface provides code-listed properties to describe the coating properties for SP3D parts. These include coating type description and coating color.
- IJConstructionInfo: This interface provides information on the construction status of a part. The status indicates whether the part is new, existing, future, etc.
- IJFabricationInfo: This interface provides information on who is responsible for the fabrication of the part.

Equipment Entity Data Model

- IJDistribPartOccur: This interface is used to establish a relationship to the distribution ports.
- IJOccInsulation: This interface is used to define the purpose, material and thickness for insulation for the standard equipment.
- IJConnectable: This interface is used to return a set of equipment foundation ports.

Equipment Entity Data Model

Smart Occurrence to Smart Item Relationship



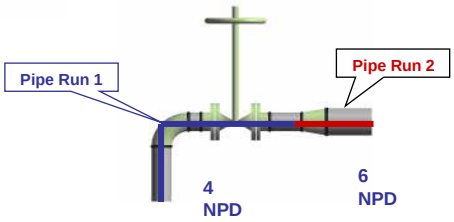
- The smart occurrence is related to the Smart Item with a relationship named SOtoSI_R.

Piping Entities Data Model

PipeRun Interfaces

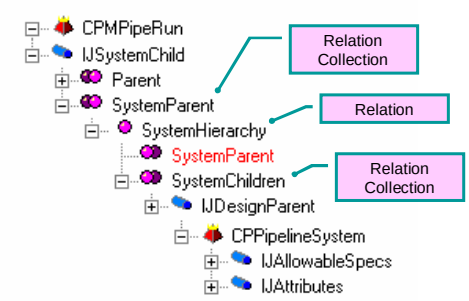
A PipeRun identifies one or more path features that share a common pipe specification, flow direction, size, temperature, pressure, etc….

One or more runs make up a pipeline.



Piping Entities Data Model

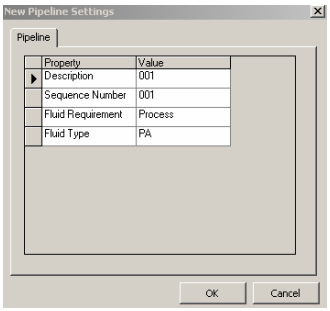
PipeRun and Pipeline Relationship



Pipeline is a high-level grouping of Pipe Runs that are created in System and Spec Task environment.

System Hierarchy Relation

The system hierarchy relationship is used to specify that a run (IJSystemChild) has a parent system (JDesignParent)

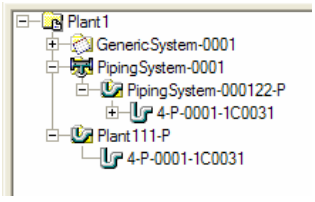


Piping Entities Data Model

Example: returns pipe runs and pipeline names

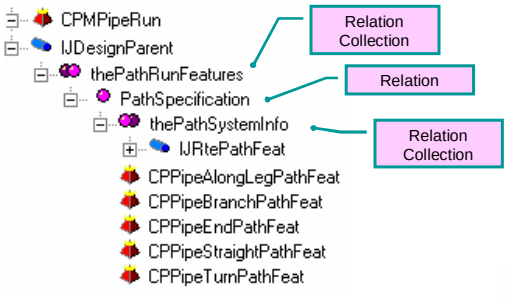
```
select
x4.ItemName as PipeRunName,
x3.ItemName as Parent_System
from JRtePipeRun x1
join JNamedItem x4 on x4.oid = x1.oid
join  XSystemHierarchy x2 on x2.oiddestination = x1.oid
join JNamedItem x3 on x3.oid = x2.oidorigin
```

	PipeRunName	Parent_System
1	4-P-0001-1C0031	PipingSystem-000122-P
2	6-P-0002-1C0031	Plant111-P



Piping Entities Data Model

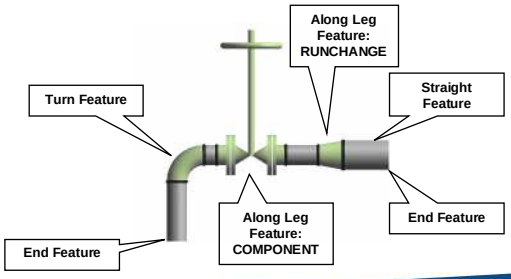
PipeRun and Features Relationship



Path Specification Relation

All path features are related to a run. This relationship allows the path feature to retrieve the specification and rules that it needs to meet from the run.

Features define the geometry path of the run and your design intent that occurs along the path. When you route a pipe run, you place features.



Piping Entities Data Model

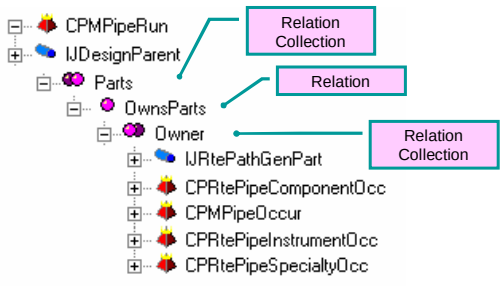
Example: Select empty runs:

```
select distinct x1.oid from JRtePipeRun x1
WHERE NOT  EXISTS
(select 1 from XPathSpecification x2 where x1.oid = x2.oidorigin)
```

Piping Entities Data Model

SmartPlant 3D

PipeRun and Parts Relationship



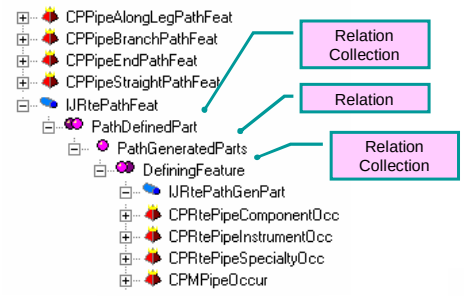
Owns Parts Relation

The OwnsParts relationship is used to specify that a part occurrence is a child of a run.

Piping Entities Data Model

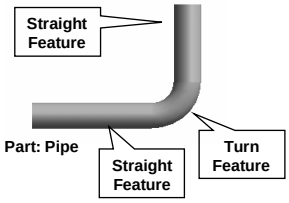
SmartPlant 3D

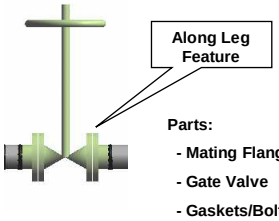
Feature and Part Relationship



PathGeneratedParts Relation

This relationship identifies the part occurrences that are generated from a given feature based on specifications. For a feature representing a component, there may be one or more generated parts. For a pipe, there may be multiple features associated with a single generated part.





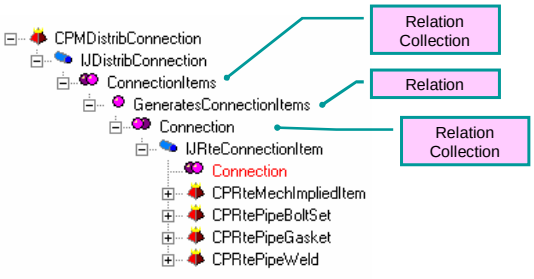
Parts:

- **Mating Flanges**
- **Gate Valve**
- **Gaskets/Bolts/Nuts**

Piping Entities Data Model

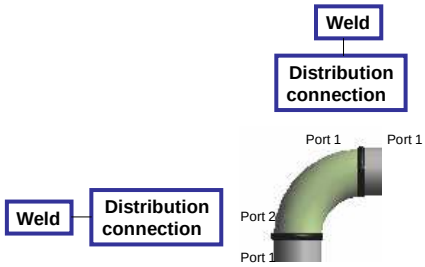
SmartPlant 3D

Distribution Connection Relationship



GeneratesConnectionItems Relation

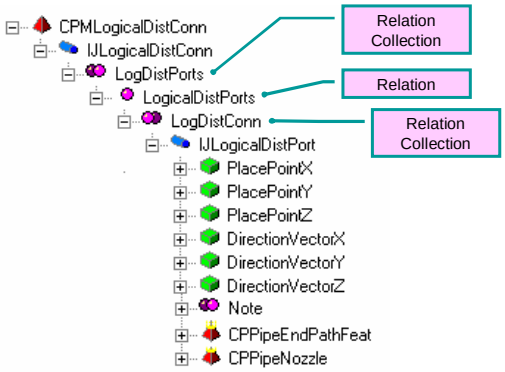
This relationship identifies the connection items that are generated from a given distribution connection.



Piping Entities Data Model

SmartPlant 3D

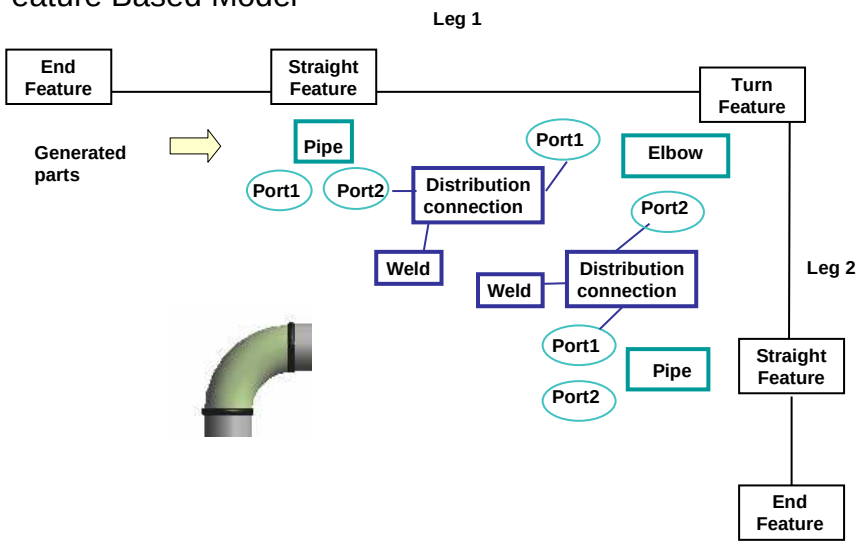
Logical Distribution Connection Relationship



LogicalDistPorts Relation

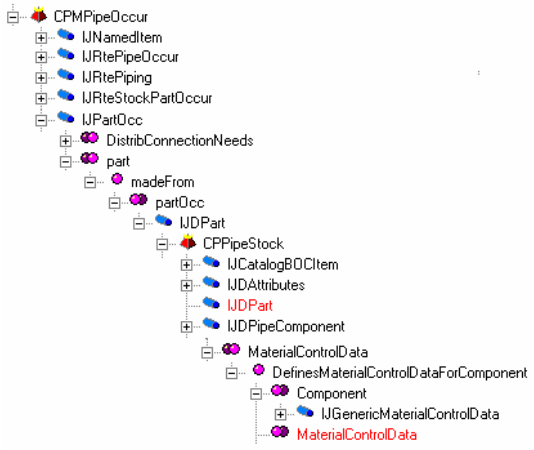
This relationship identifies the connection between an equipment nozzle and the end feature of a path leg.

Feature Based Model



Piping Data Model

Example: Query all pipe occurrences



Piping Data Model

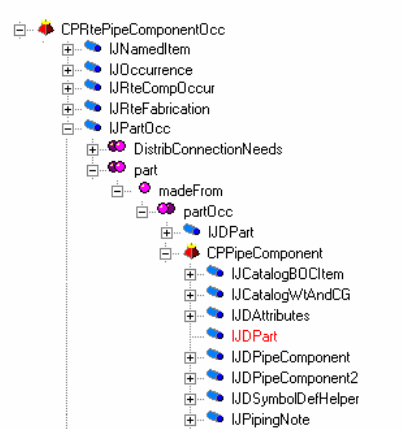
Example: Query all pipe occurrences

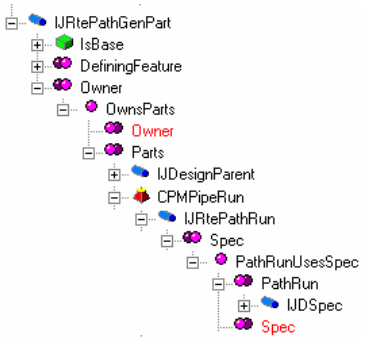
```
Select
x3.IndustryCommodityCode,
x3.PrimarySize,
x3.PriSizeNPDUnits,
Sum (x1.length) as 'TotalLength (m)'
from JRteStockPartOccur x1
Join XmadeFrom x2 on x2.oidorigin = x1.oid
Join JDPipeComponent x3 on x3.oid = x2.oiddestination
Group by x3.IndustryCommodityCode, x3.PrimarySize,
   x3.PriSizeNPDUnits
```

	IndustryCommodityCode	PrimarySize	PriSizeNPDUnits	TotalLength (m)
1	PAAZZBOZZABAABOAAZZUS	4.0	in	85.0995059293430 81
2	PAAZZBOZZABAABOAAZZUS	12.0	in	4.9714296169264518
3	PAAZZBPZZABAABSAAZZUS	1.0	in	11.275046528340001
4	PAAZZBOZZABAABOAAZZUS	3.0	in	3.1506374075455992

Piping Data Model

Example: Query all pipe component occurrences





Piping Data Model

Example: Query all piping component occurrences

	IndustryCommodityCode	SpecName	qty
1	MCMZZBOZZAAEADCZZUS	1C0031	2
2	MDJZZBOZZAAEADCZZUS	1C0031	1

```
select
x3.IndustryCommodityCode,
x9.SpecName,
count(*) as qty
from JRteCompOccur x1
JOIN XMadeFrom x2 ON (x2.OidOrigin = x1.Oid)
JOIN JDPipeComponent x3 ON (x3.Oid = x2.OidDestination)
JOIN XOwnsParts x5 ON (x5.oiddestination = x1.oid)
JOIN JRtePipeRun x6 on (x6.oid = x5.oidorigin)
JOIN XPathRunUsesSpec x7 ON (x7.OidOrigin = x6.oid)
JOIN JDSpec x8 ON (x8.oid = x7.oiddestination)
JOIN JDPipeSpec x9 on (x9.oid = x8.oid)
group by x3.IndustryCommodityCode, x9.SpecName
```


Piping Data Model

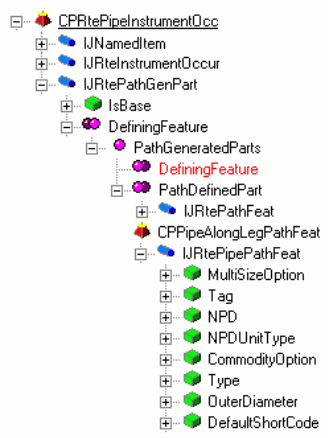
Example: Query all valves occurrences

	IndustryCommodityCode	PipeRun_Name	CommodityType	qty
1	VBGAHABAHAFEADAZZZZUS	4-P-0001-1C0031	CKS	1
2	VALAHABAHACWADAZZZZUS	4-P-0001-1C0031	GLO	1
3	VAAAHABAHADJADAZZZZUS	6-A-0001-1C0031	GAT	2
4	VAAAHABAHADJADAZZZZUS	2-A-0002-1C0031	GAT	1

```
Select
x3.IndustryCommodityCode,
x4.ShortStringValue as 'CommodityType',
count(*) as qty
from JRteCompOccur x1
Join XMadeFrom x2 ON (x2.OidOrigin = x1.Oid)
Join JDPipeComponent x3 ON (x3.Oid = x2.OidDestination)
Join CL_PipingCommodityType x4 ON (x4.ValueID =
  x3.CommodityType)
WHERE (x3.CommodityClass = 5)
Group by x3.IndustryCommodityCode, x4.ShortStringValue
```

Piping Data Model

Example: Query all Instrument occurrences



Piping Data Model

Example: Query all Instrument occurrences

```
select
x1.oid,
'Tag/Name'=
Case
  when x4.tag = 'undefined'
  then x4.tag
  else x2.Itemname
End,
x4.NPD,
x4.NPDUnitType

from JRteInstrument x1
join JNamedItem x2 ON x2.oid = x1.oid
join XPathGeneratedParts x3 on x3.oiddestination = x1.oid
join JRtePipePathFeat x4 on x4.oid = x3.oidorigin
```

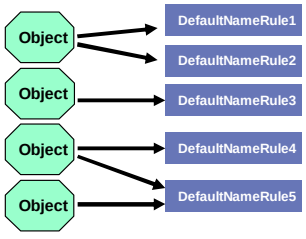
	oid	Tag/Name	NPD	NPDUnitType
1	000138B6-0000-0000-0B02-45EAD3432204	Component-0001	4.0	in

Naming Rules

- Customized component that allows users to define rules for formatting the names of business objects
- Currently, the naming rule is a Visual Basic component
- Name can be comprised of the following:
  - Object type
  - Object properties
  - Primary organizational relationships
    - System
    - Space
    - Assembly
    - Work Breakdown
  - Other relationships
    - Connection

Naming Rules - Reference Data

- Naming rules are associated to an object class
- Multiple rules can be defined for each class of object
  - If multiple rules are provided in the catalog, the user must select the rule to apply when objects are created
- A rule can also be shared by multiple object classes



Naming Rules - Reference Data

- Mapping of rules to object classes
  Example: GenericNamingRules.xls
  …\CatalogData\BulkLoad\Datafiles

Object Type Name in GUI VB Component ProgID

TypeName	Name	SolverProgID
CPEquipment	DefaultNameRule	NameRules.CommonNameRule
CPEquipment	UniqueNameRule	NameRules.UniqueName
CPEquipment	System Parent Name Rule	EquipNamingRules.CommonRule
CPDesignEquipment	DefaultNameRule	NameRules.CommonNameRule
CPDesignEquipment	UniqueNameRule	NameRules.UniqueName
CPDesignEquipment	System Parent Name Rule	NameRules.SystemNameRule
CPMSystem	DefaultNameRule	SystemNameRules.DefaultNameRule
CPMSystem	User Defined	SystemNameRules.UserDefinedNameRule
CPStructuralSystem	DefaultNameRule	SystemNameRules.DefaultNameRule
CPStructuralSystem	User Defined	SystemNameRules.UserDefinedNameRule

Naming Rules - Creation

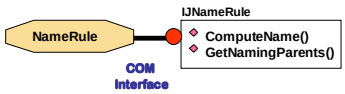
- Visual Basic component
- Implements IJNameRule

ComputeName

This method is used to format the name of an object

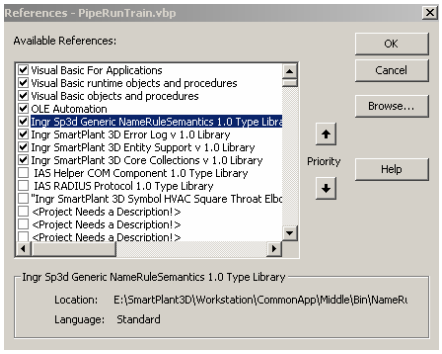
GetNamingParents

This method is used to return the related parent object or objects that are used in forming a name



Naming Rules - Reference Libraries

- Ingr Sp3D Generic NameRuleSemantics 1.0 Type Library
- Ingr SmartPlant 3D Error Log v1.0 Library
- Ingr SmartPlant 3D Entity Support v 1.0 Library
- Ingr SmartPlant 3D Core Collections v 1.0 library



IJNameRule_ComputeName

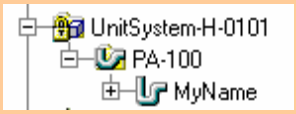
- This method is used to format the name of an object

```
Private sub IJNameRule_ComputeName
    (
    ByVal oObject as object  ‘ Object being named
    ByVal oParents as IJElements ‘ Naming Parents collection
    ByVal oActiveEntity as object ‘Naming rules active entity
    )
```

IJNameRule_ComputeName

Example (Constant Name)

```
Private Sub IJNameRule_ComputeName(ByVal oObject As Object, _
    ByVal elements As IJElements, ByVal oActiveEntity As Object)

    Dim oChildNamedItem As IJNamedItem
    Dim strChildName As String
    Set oChildNamedItem = oObject
    strChildName = "MyName"
    oChildNamedItem.Name = strChildName
    Set oChildNamedItem = Nothing
```



Naming Rules - Error Handling

When errors occur in retrieving properties or related objects, proper error handling is important

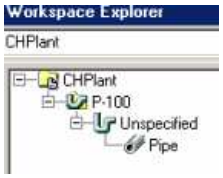
- Use the middle tier error log to write specific information

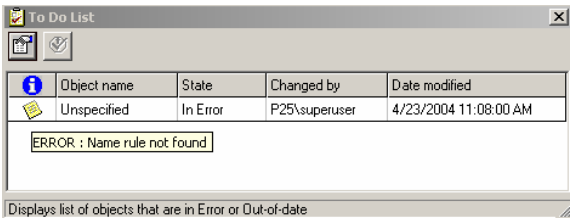
```
Dim oErrors as IJEditErrors  ' middle tier errors
Private Const E_FAIL = -2147467259
                        …..

Private Sub Class_Initialize()
Set oErrors = new IMSErrorLog.JServerErrors
End Sub
                        ….
label:
oErrors.Add err.Number, Method, err.Description
Err.Raise E_FAIL
```

Naming Rules - Error Handling

If a name rule returns an error, the generic semantic will trap the error

- Set the name to “unspecified”
- Add the object to the To Do list



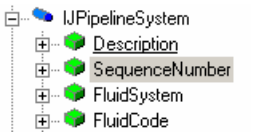


Naming Rules - Services

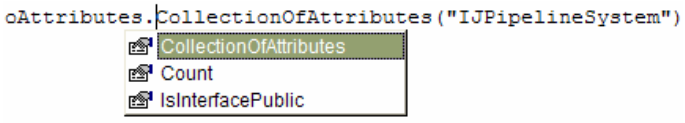
- Attribute Helper service
- Relation Helper service
- Naming counter service
- GetNamingParents Method

Attribute Helper Service

- Ingr SmartPlant 3D Attributes v 1.0 Library
- Get Object attributes
- Need to know the interface and property names.

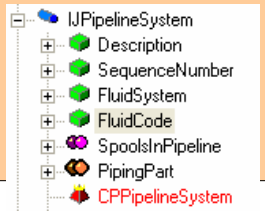



```
Dim oAttributes As IJDAttributes
Set oAttributes = oObject
Dim strSeqNum As String
strSeqNum =
    oAttributes.CollectionOfAttributes("IJPipelineSystem").Item("SequenceNumber").Value
```



Attribute Helper Service

```
Dim FluidCodeID As Long
Dim strFluidCode As String
strFluidCode = vbNullString
Dim m_oCodeListMetadata As IJDCodeListMetaData

Set m_oCodeListMetadata = GetCatalogResourceManager
FluidCodeID = oAttributes.CollectionOfAttributes("IJPipelineSystem").Item("FluidCode").Value
strFluidCode = m_oCodeListMetadata.ShortStringValue("FluidCode", FluidCodeID)

strChildName = strFluidCode & "-" & strSeqNumber
oChildNamedItem.Name = strChildName

Set oChildNamedItem = Nothing
Set oAttributes = Nothing
```



Attribute Helper Service (Cont's)

Example 1: Pipeline (Final code)

```
Dim oChildNamedItem As IJNamedItem
Dim strChildName As String
Set oChildNamedItem = oObject
strChildName = vbNullString
Dim oAttributes As IJDAttributes
Set oAttributes = oObject
Dim strSequenceNumber As String
strSequenceNumber =
oAttributes.CollectionOfAttributes("IJPipelineSystem").Item("SequenceNumber").Value
Dim FluidCodeID As Long
Dim strFluidCode As String
strFluidCode = vbNullString
……..
```

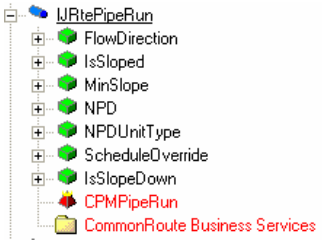

Attribute Helper Service (Cont's)

```
…..
Set m_oCodeListMetadata = GetCatalogResourceManager
FluidCodeID =
oAttributes.CollectionOfAttributes("IJPipelineSystem").Item("FluidCode").Value
strFluidCode = m_oCodeListMetadata.ShortStringValue("FluidCode", FluidCodeID)

strChildName = strFluidCode & "-" & strSequenceNumber
oChildNamedItem.Name = strChildName

Set oChildNamedItem = Nothing
Set oAttributes = Nothing
```

Attribute Helper Service

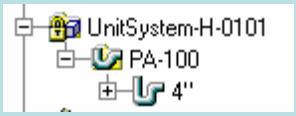
Example 2: PipeRun

```
Dim A as IJDAttributes
Set A = oObject
'get the NPD of the current Run from the IJRtePipeRun interface
strNPD = A.CollectionOfAttributes("IJRtePipeRun").Item("NPD").Value
```



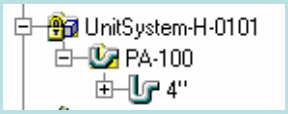
Attribute Helper Service (Cont's)

```
Dim oChildNamedItem As IJNamedItem
Dim strChildName As String
Set oChildNamedItem = oObject
strChildName = vbNullString
Dim oAttributes As IJDAttributes
Set oAttributes = oObject
' Get NPD and Unit
Dim strNPD As String
Dim strNPDUnitType As String
strNPD = CStr(oAttributes.CollectionOfAttributes("IJRtePipeRun").Item("NPD").Value)
strNPDUnitType =
oAttributes.CollectionOfAttributes("IJRtePipeRun").Item("NPDUnitType").Value
```



Attribute Helper Service (Cont's)

```
If strNPDUnitType = "in" Then
    strNPDUnitType = Chr(34)
  End If
strChildName = strNPD & strNPDUnitType
oChildNamedItem.Name = strChildName
Set oChildNamedItem = Nothing
Set oAttributes = Nothing
```



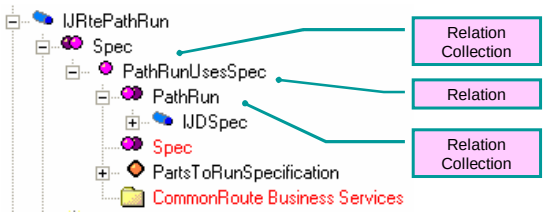
Relation Service

- Ingr SmartPlant 3D Relation v 1.0 Library
- Provides access to related objects

- To use this service you must know the relation collection name and the interface name

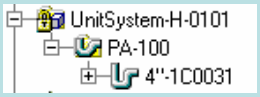
  Interface name: IJRtePathRun

  Relation collection name: Spec



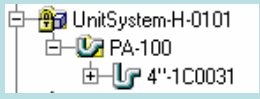
Relation Service (Cont's)

```
Dim oRelationHelper As IMSRelation.DRelationHelper
Dim oCollection As IMSRelation.DCollectionHelper
Set oRelationHelper = oObject
Dim oPipeSpec As IJDSpec
Dim strSpecName As String
Set oCollection = oRelationHelper.CollectionRelations("IJRtePathRun", "Spec")
Set oPipeSpec = oCollection.Item(1)
strSpecName = oPipeSpec.SpecName
```



Relation Service (Cont's)

```
oChildNamedItem.Name = strChildName  & "-" & strSpecName
Set oChildNamedItem = Nothing
Set oRelationHelper = Nothing
Set oCollection = Nothing
Set oPipeSpec = Nothing
```



IJNameRule_GetNamingParents

- This method is used to return the related parent object or objects that are used in forming a name

```
Private function IJNameRule_GetNamingParents
    (
     ByVal oEntity as object   ' Object being named
    )
    As IJElements
```

- All the Naming Parents that need to participate in an objects naming are added here to the IJElements collection.

IJNameRule_GetNamingParents

- Example (Return No Parent)

```
Private Function IJNameRule_GetNamingParents(ByVal oEntity As Object) As IJElements
    Const METHOD = "IJNameRule_GetNamingParents"
    On Error GoTo label
    Set IJNameRule_GetNamingParents = Nothing
' If you need the Parent objects, then Add code Here to get the Naming Parents that will participate in the naming.
Exit Function
label:
    ' log the error in middle tier and propagate the error code to the caller
    m_oErrors.Add Err.Number, "IJNameRule_GetNamingParents", Err.Description
    Err.Raise E_FAIL
End Function
```

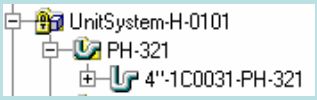
IJNameRule_GetNamingParents

- Example (Return Parent Name)

```
Private Function IJNameRule_GetNamingParents(ByVal oEntity As Object) As IJElements
    Const METHOD = "IJNameRule_GetNamingParents"
    On Error GoTo label
    Set IJNameRule_GetNamingParents = New IMSCoreCollections.JObjectCollection
    Dim oSysChild As IJSystemChild
    Set oSysChild = oEntity
    Dim oSysParent As IJSystem
    Set oSysParent = oSysChild.GetParent
    If Not (oSysParent Is Nothing) Then
       Call IJNameRule_GetNamingParents.Add(oSysParent)
    End If
    Set oSysChild = Nothing
    Set oSysParent = Nothing
.......
```


GetNamingParent

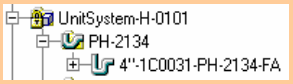
```
'Get Parent
.......
    Dim oParentNamedItem As IJNamedItem
    Dim strParentName As String
    strParentName = vbNullString
    Set oParentNamedItem = elements.Item(1)
    strParentName = oParentNamedItem.Name
    oChildNamedItem.Name = strChildName  & "-" & strParentName
```



Attribute Service

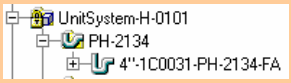
Ingr SmartPlant 3D Attributes v 1.0 Library
m_oCodeListMetadata.ShortStringValue("Tablename", ValueID)
Get short description of a code list attribute

```
Set m_oCodeListMeta = GetCatalogResourceManager
Dim HeatTracedID As Long
Dim strHeatTraced As String
strHeatTraced = vbNullString
HeatTracedID =
oAttributes.CollectionOfAttributes("IJHeatTracingInfo").Item("HeatTracingRequirement").Value
strHeatTraced = m_oCodeListMeta.ShortStringValue("HeatTracingRequirement",
   HeatTracedID)
Dim HeatMediumID As Long
Dim strHeatMedium As String
strHeatMedium = vbNullString
HeatMediumID =
   oAttributes.CollectionOfAttributes("IJHeatTracingInfo").Item("HeatTracingMedium").Value
strHeatMedium = m_oCodeListMeta.ShortStringValue("HeatTracingMedium", HeatMediumID)
```



Attribute Service

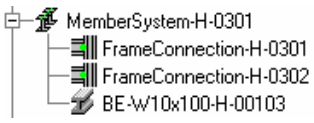
```
 If HeatTracedID = 1 Then
       strChildName = strNPD & strNPDUnitType & "-" & strSpecName & "-"
  & strParentName
   Else: strChildName = strNPD & strNPDUnitType & "-" & strSpecName &
  "-" & strParentName & "-" & strHeatTraced & "-" & strHeatMedium
End If

   oChildNamedItem.Name = strChildName

'Release BO's
   Set oChildNamedItem = Nothing
   Set oAttributes = Nothing
   Set oRelationHelper = Nothing
   Set oCollection = Nothing
   Set oPipeSpec = Nothing
   Set oParentNamedItem = Nothing
```

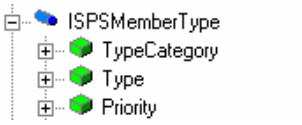


Attribute Helper Service

Example 3: Member Part

```
Dim oAttributes As IJDAttributes
Set oAttributes = oObject
Dim MemberTypeID As Long
MemberTypeID =
    oAttributes.CollectionOfAttributes("ISPSMemberType").Item("TypeCategory").Value
```

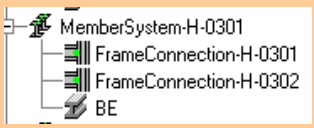




Member Category Desc + Section Name + Location + IndexCounter

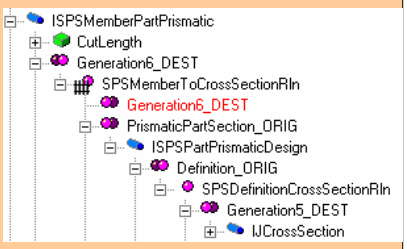
Attribute Helper Service

```
Dim strTableName As String
Dim strMemType As String
strTableName = "StructuralMemberTypeCategory“
Dim m_oCodeListMetadata As IJDCodeListMetaData

If m_oCodeListMetadata Is Nothing Then
    Set m_oCodeListMetadata = GetCatalogResourceManager
End If
strMemType = UCase(Left(m_oCodeListMetadata.ShortStringValue(strTableName,
    MemberTypeID), 2))
```



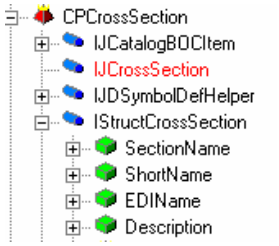
Relation Helper Service

```
Dim oRelationHelper As IMSRelation.DRelationHelper
Dim oCollectionHelper As IMSRelation.DCollectionHelper
Set oRelationHelper = oObject
Set oCollectionHelper = oRelationHelper.CollectionRelations("ISPSMemberPartPrismatic",
    "Generation6_DEST")
Set oRelationHelper = oCollectionHelper.Item(1)
Set oCollectionHelper = Nothing
Set oCollectionHelper = oRelationHelper.CollectionRelations("ISPSPartPrismaticDesign",
    "Definition_ORIG")

Dim oMembCrossSection As IJCrossSection
Set oMembCrossSection = oCollectionHelper.Item(1)
```



Attribute Helper Service

```
Dim strSectionName As String
Set oAttributes = oMembCrossSection
strSectionName =
oAttributes.CollectionOfAttributes("IStructCrossSection").Item("SectionName").Value
```



Naming Counter Service

- Sp3d NameGenerator 1.0 Type Library
- Naming counter service - generates a unique counter given a basis string
- GetCount(): This method returns a new index number if the basis string is different from the NamingParentString
- GetCountEx(): This method returns the count and the location prefix.

```
Dim oNameCounter as IJNameCounter
Set oNameCounter = New NameGeneratorService

nCount = oNameCounter.GetCount(oModelResourceMgr, strBasis)
Sname = strBasis + Str$(nCount)
  Or….
nCount = oNameCounter.GetCountEx(oModelResourceMgr, strBasis,
  strLocation)
Sname = strBasis + Str$(nCount)
```


Naming Service Architecture

SP3D (Taskhost.exe)

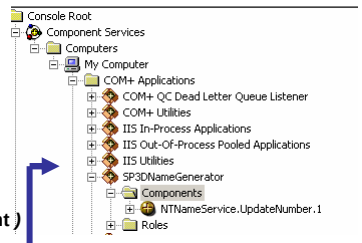
Property	Value
Pipeline	P-200
Name	P-200-1C0049-6 in-Bi-directional-H-1
Name Rule	PipeRunLineID
Specification	1C0049
Nominal Diameter	6 in
Flow Direction	BIDIRECTIONAL

NameRule component (PipeRunLineID)

GetCount(pResourceManager, strBasename, * nCount)

NameGeneratorService

SP3DNameGenerator
COM+ Application On the Administrator machine



UpdateNumber

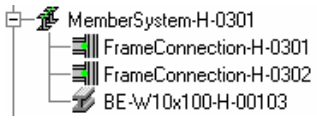
Naming Counter Service

```
Dim oNameCounter As IJNameCounter
Set oNameCounter = New GSCADNameGenerator.NameGeneratorService
Dim strLocation As String
strLocation = vbNullString
Dim nCount As Long
Set oModelResourceMgr = GetModelResourceManager
nCount = oNameCounter.GetCountEx(oModelResourceMgr, strChildName, strLocation)
If Not (strLocation = vbNullString) Then
    strChildName = strChildName + "-" + strSectionName + "-" + strLocation + "-" +
    CStr(nCount)
 Else
    strChildName = strChildName + "-" + strSectionName + "-" + CStr(nCount)
End If
oChildNamedItem.Name = strChildName
```


Naming Counter Service

Format the counter

```
Private Const strCountFormat = "00000“
....
 strChildName = strChildName + "-" + strLocation + "-" + Format(nCount, strCountFormat)
```

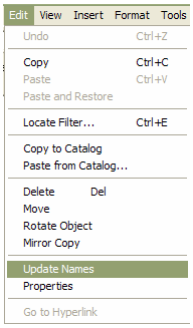


Binary Compatibility

- A VB ActiveX DLL/OCX is a COM Class
  - That is, it has a CLSID
- When you build DLL, VB generates a *new* CLSID
- Project that have your ActiveX DLL as a reference are then "broken"
  - Reference is marked "Missing"
- Solution: when you get a good build, save the DLL as a Binary-Compatible reference mark
  - Make all future builds binary compatible to it
  - This preserves the CLSID (& other project's references!)

Naming Rules - Re-compute

- Naming rule semantic is responsible for calling the ComputeName when the following actions occur:
  - A property on the named object is modified.
  - A property on the naming parent object is modified.
  - System parent or assembly parent is changed.
- Use the Update Name to re-fire the naming rule semantic to all objects in the select set

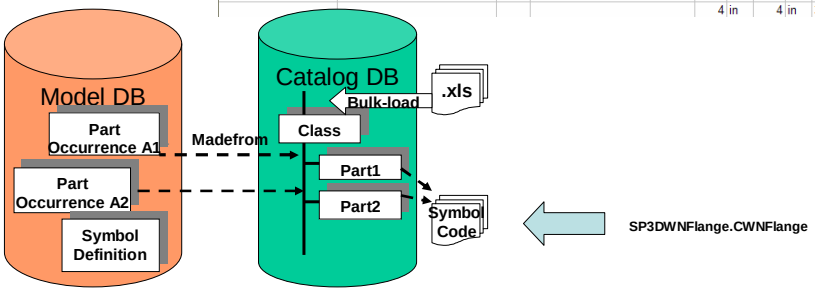


Overview

Pre-defined parts are defined in the Catalog.

Ten_Specs_CatalogData.xls

CommodityPart									
Head	IndustryCommodityCode	CommodityType	SymbolDefinition	Npd[1]	NpdUnitType[1]	Npd[2]	NpdUnitType[2]	FacetoFace	
	FAAAHDCZZAADABQHPUS	FWN	SP3DWNFlange.CWNFlange	1.5	in	1.5	in	2.4375in	
	FAAAHDCZZAADABQZZUS	FWN	SP3DWNFlange.CWNFlange	2	in	2	in	2.5in	
				3	in	3	in	2.75in	
				4	in	4	in	3in	



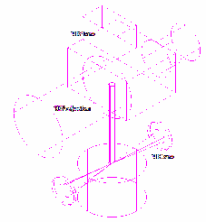
Symbol Overview

Symbols: What is it ?

……..Reusability of Graphical Entities

Probably the most used entity by SmartPlant 3D Applications.
Example:

- Piping
- Electrical
- Hvac
- Equipment
- Structure



The end user creates a graphical or programmed construction of the symbol, save it and expose it through the Reference Data

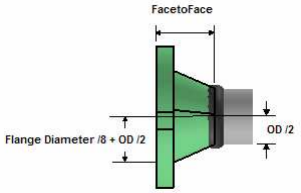
Symbol Overview

SmartPlant 3D

Parametric symbols: It means that some parameters (inputs of the symbol) can 'drive' the outputs (graphical entities).

IndustryCommodityCode	FirstSizeSchedule	SecondSizeSchedule	CommodityType	GeometryType	GraphicalRepresentationOrNot	SymbolDefinition	PipingPointBasis[1]	Id[1]	PressureRating[1]	EndPreparation[1]	EndStandard[1]	ScheduleThickness[1]	FlowDirection[1]	PipingPointBasis[2]	Id[2]	PressureRating[2]	EndPreparation[2]	EndStandard[2]	ScheduleThickness[2]	FlowDirection[2]	Npd[1]	NpdUnitType[1]	Npd[2]	NpdUnitType[2]	FacetoFace
FAAAHDCZZAADABQZZUS		S-STD	FWN	15		SP3DFWN.CSP3DFWN	15		150	21	5	3	15			301	5	S-STD	3		2 in		2 in		2.5in
																					3 in		3 in		2.75in
																					4 in		4 in		3in
																					6 in		6 in		3.5in
																					8 in		8 in		4in

NominalPipingDiameter	NominalDiameterUnits	PressureRating	EndPreparation	EndStandard	FlangeOutsideDiameter	FlangeThickness	FlangeThicknessTolerance
2 in		150	21	5	6in	0.75in	0.12in
3 in		150	21	5	7.5in	0.9375in	0.12in

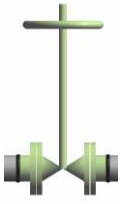


Symbol Overview

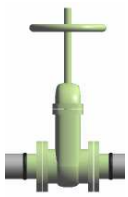
SmartPlant 3D

Functionality

- Support Multiple Representations: allows to have several different aspects of the same components.
- it is up to the symbol definition designer to define the aspects that the symbol will support.
- The aspects to be displayed is controlled by the client tier (Format View dialog).



Simple Physical Aspect

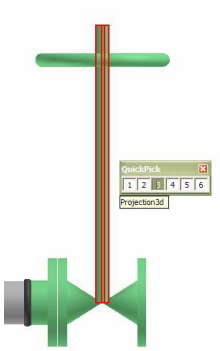


Detailed Physical Aspect

Symbol Overview

Functionality

- Support the functionality of penetrating a symbol occurrence at locate time to identify some of its members or constituent.



Symbol Overview

Functionality

- Support Nesting: Nesting means that a symbol can be made of symbols.

Symbol Machinery

Definition

- Symbol: Displayable graphic entities (An instantiation of the symbol definition)
- Symbol Definition: The definition of a symbol

Symbol
Graphical entities:
-3Dprojections
-3Dpoints
-3Dlines

Symbol Definition
Descriptions of:
-Inputs
-Representations
-Server
-User methods
-Evaluate Functions

Visual Basic Component

VB Symbol

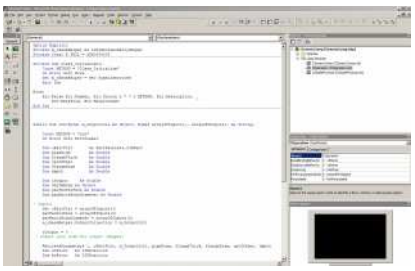
Solid Edge file

Shape 2D file

Symbol Definition – Visual Basic

Visual Basic Project (.vbp) and Class Modules (.cls) files:

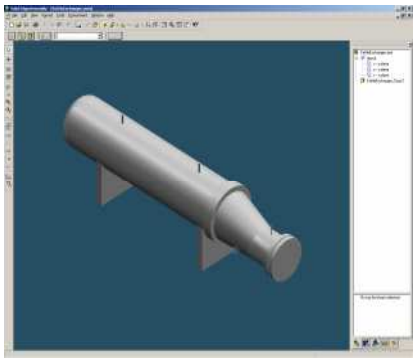
- **Each Class Module file represents a different Aspect of the symbol.**
- **The Project is a collection of supporting code that identifies the number of aspects and outputs for the symbol.**
- **Quick Symbol Template creation is achieved through SP3D's Part Definition Wizard.**
- **The basGeom3D contains functions to create many of the common construction shapes. They are utilized when creating the output constructions in the class modules.**
- **The final item of this symbol is a .dll file or cab file. (The source code should be preserved for future changes, but does not need to be distributed with the .dll file in order for the symbol to work.)**



Symbol Definition – Solid Edge

Solid Edge Assembly (.asm) and Part (.par) files:

- **Each SE Part file represents a different aspect of the symbol.**
- **The Assembly is a collection of part files that references position of the parts to one another.**
- **Ports are identified on symbol by executing the Solid Edge Port macro and selecting the construction that represents the placement for the port.**
- **The final item of this symbol is one .asm file per symbol, and one .par file for every aspect that the symbol lives.**

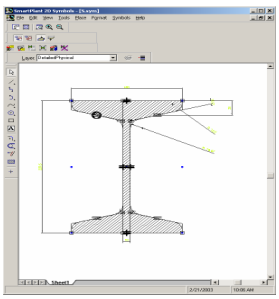


Symbol Definition – 2D Symbols

2D Symbols Cross-section file (.sym):

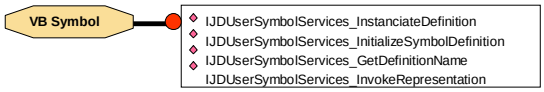
Each Cross-section file contains layers.

The information contained on the default layer represents the cross-section Physical Aspect. There are also other layers that contain dimension data for the symbol.



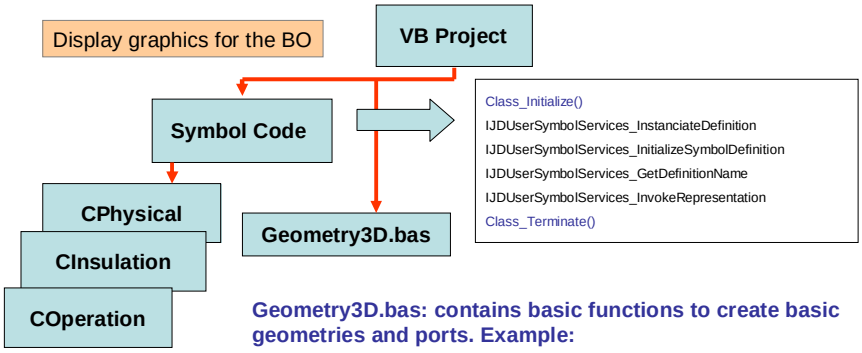
Symbol Definition – Visual Basic

- VB Symbol is a Visual Basic component
- Implements IJDUserSymbolServices
  - IJDUserSymbolServices_InstanciateDefinition
  - IJDUserSymbolServices_InitializeSymbolDefinition
  - IJDUserSymbolServices_GetDefinitionName
  - IJDUserSymbolServices_InvokeRepresentation



Symbol

Visual Basic Project (.vbp) and Class Modules (.cls) files:



Geometry3D.bas: contains basic functions to create basic geometries and ports. Example:

```
Public Function PlaceProjection(ByVal objOutputColl As Object, _
                                ByVal objCurve As Object, _
                                ByVal axisVector As AutoMath.DVector, _
                                height As Double, _
                                isCapped As Boolean)
```

Symbol Definition – Visual Basic

- User Symbol Services Object (IJDUserSymbolServices):
- Set the following information (Class_Initialize routine):
  - Inputs: are entries that drive the graphic representation of the symbol
  - Outputs: Graphics Output Description
  - Representation(s): Description

IJSymbolHelper

This interface provides methods to help in creating the definition of a VB symbol. It provides the implementation of the IJDUserSymbolServices interface as well as provide support for declaring the inputs and outputs of the symbol.

Symbol – Visual Basic

Inputs

The graphic representations (aspects) of a symbol can be driven by a set of inputs that are numeric values (parameter).

An input is identified by a *name*, has some properties and optionally a default value.

```
m_oSymbolHelper.NumInputs = 1
m_oSymbolHelper.AddInputDef 1, "FacetoFace", "FacetoFace", 4
```

AddInputDef(Count As Integer, Name As String, Description As String, DefaultValue As Double)	
Description:	Adds the input definition to the collection of inputs defined for the symbol
Parameters:	
[in] count	Index for the input parameter
[in] Name	Name of the input parameter
[in] Description	Description of the input parameter
[in] DefaultValue	Default value for the input parameter

Symbol – Visual Basic

Outputs

- The representation is described as a multi-level hierarchy of outputs characterized by a name, description and Aspects (Representation id).

```
m_oSymbolHelper.NumOutputs = 3
m_oSymbolHelper.AddOutputDef 1, "Body1", "Body1", 1
m_oSymbolHelper.AddOutputDef 2, “port1", “port1", 1
m_oSymbolHelper.AddOutputDef 3, “port2", “port 2", 1
```

AddOutputDef(Count As Integer, Name As String, Description As String, aspect as integer)	
Description:	Adds the output definition to the collection of outputs defined for the symbol
Parameters:	
[in] count	Index for the output parameter
[in] Name	Name of the output parameter
[in] Description	Description of the output parameter
[in] aspect	Aspect number for the output

Symbol – Visual Basic

Representation

- An representation is identified by a *name, representation id* and optionally a description.
- Representation id is defined as the aspect value written in powers of two.

```
m_oSymbolHelper.NumAspects = 1
m_oSymbolHelper.AddAspectDef 1, "SimplePhysical", "SimplePhysical", 1
```

AddAspectDef (Count As Integer, Name As String, Description As String, aspect as integer)	
Description:	Adds the aspect definition to the symbol
Parameters:	
[in] count	Index for the aspect
[in] Name	Name of the aspect
[in] Description	Description of the aspect
[in] aspect	Aspect number for the output

SP3D Part definition VB Wizard

What is it?

What does it do?

SP3D Part definition VB Wizard

..\SmartPlant3D\Programming\Tools\SymbolWizard

Workstation Machine:

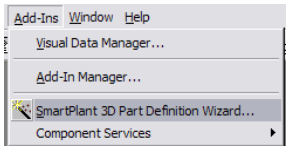
SP3D Programming Resources available under

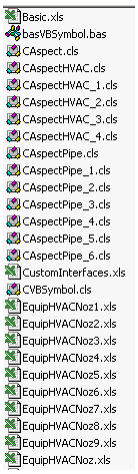
[Install Directory]\Programming Resources

Source code of all symbols, rules, and example codes

SP3D Part definition VB Wizard

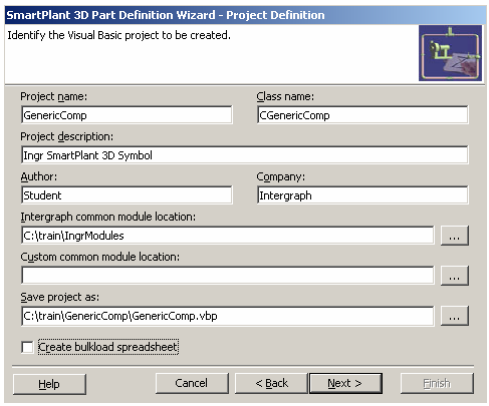
- ..\SP3D Symbol Wizard\Templates
- Run Wizard from Add-ins or Tool bar





SP3D Part Definition VB Wizard

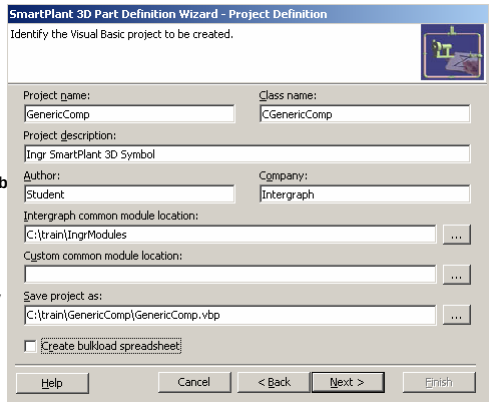
- Create Symbols for Piping Components with user defined Inputs
- Standard Data Types Attributes:
  - Double
  - String
- Automatic positioning and orientation of key ports**.**



SP3D Part Definition VB Wizard

- Shared Modules:
  - Intergraph Geometrical functions
  - Company standard
- Generation of Error Handling module
- Automatic attachment of key References:

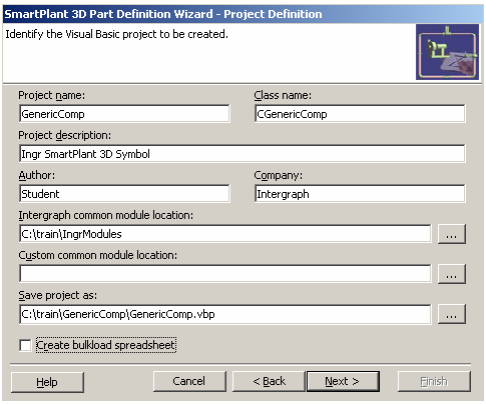
Ingr Sp3d RefDataGlobal 1.0 Type Library.
Ingr Cmn App Symbols Custom Cache 1.0 Type Lib
Ingr Sp3d Nozzle Entities 1.0 Type Library
Ingr Sp3d Nozzle Facelets 1.0 Type Library
Ingr Sp3d PartFacelets 1.0 Type Library
Ingr Sp3d Symbol Services 1.0 Type Library
Ingr SmartPlant 3D Client Collections v 1.0 Library
Ingr SmartPlant 3D Symbol Entities v 1.0 Library
Ingr SmartPlant 3D Units of Measure v 1.0 Library
Ingr SmartPlant Error Log v 1.0 Library
RefDataCableTrayFacelets 1.0 Type Library



Creating a Symbol: Wizard Page 1

Fields:

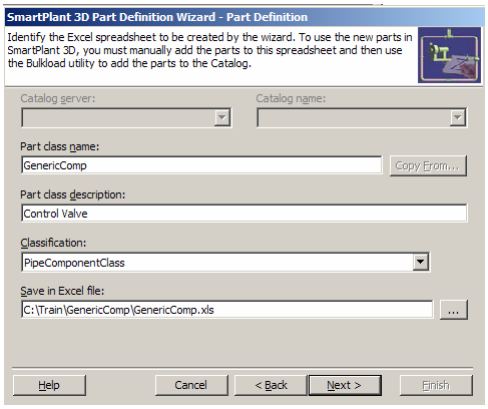
- **Project Name**
- **Class Name**
- **Project Description**
- **Author**
- **Company**
- **Intergraph common module location**
- **Custom common module location**
- **Save Project as**
- **Create bulkload spreadsheet**



Creating a Symbol: Wizard Page 2

Fields:

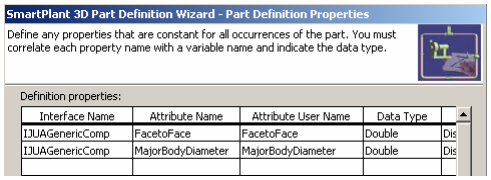
- **Catalog server**
- **Catalog name**
- **Part class name**
- **Part class description**
- **Classification**
- **Save in Excel file**

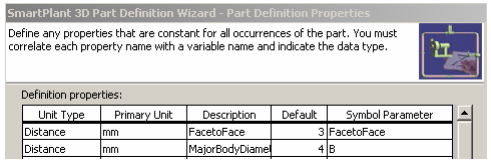


Creating a Symbol: Wizard Page 3 & 4

Fields:

- **Interface Name**
- **Attribute Name**
- **Attribute User Name**
- **Data Type**
- **Primary Unit**
- **Description**
- **Default**
- **Symbol Parameter**



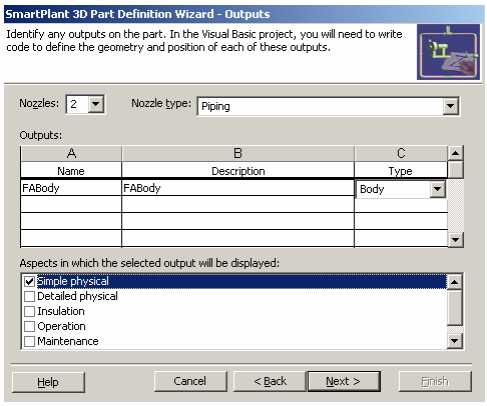


InterfaceName	AttributeName	AttributeUserName	Type	UnitsType	PrimaryUnits	OnPropertyPage	ReadOnly	SymbolParameter
IJFaceToFace	FacetoFace	Face to Face	Double	distance	mm	TRUE	FALSE	FacetoFace
IJUAMajorBodyDia	MajorBodyDiameter	Major Body Diameter	Double	distance	mm	TRUE	FALSE	B

Creating a Symbol: Wizard Page 5

Fields:

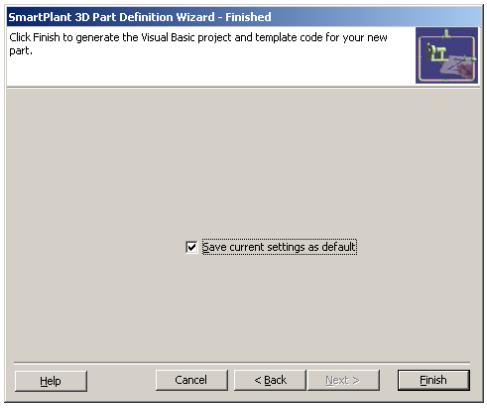
- **Nozzles**
- **Nozzle Type**
- **Outputs**
  - **Name**
  - **Description**
  - **Type**
  - **Aspects…**



Creating a Symbol: Wizard Page 6

Fields:

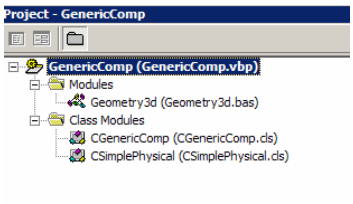
- **Save current settings as default**



What does the Wizard Produce?

A VB Project that includes:

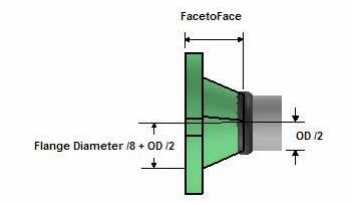
- Supporting Classes and procedures for the Symbol
- Classes and procedures based on the User input during the Wizard Execution
- Excel templates based on the User input during the Wizard Execution



Inputs & Outputs

SmartPlant 3D

Example 1: (Input & Output Definitions)



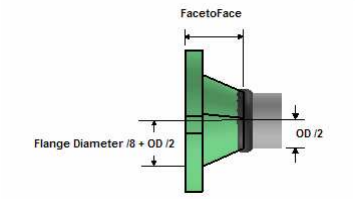

```
' Inputs
   m_oSymbolHelper.NumInputs = 1
   m_oSymbolHelper.AddInputDef 1, "FacetoFace", "FacetoFace", 2
' Outputs
   m_oSymbolHelper.NumOutputs = 3
   m_oSymbolHelper.AddOutputDef 1, "Body1", "Body", 1
   m_oSymbolHelper.AddOutputDef 2, "PipingNoz1", "Nozzle 1", 1
   m_oSymbolHelper.AddOutputDef 3, "PipingNoz2", "Nozzle 2", 1
' Aspects
   m_oSymbolHelper.NumAspects = 1
   m_oSymbolHelper.AddAspectDef 1, "SimplePhysical", "SimplePhysical", 1
```


Physical

SmartPlant 3D

Example 1: Code (Body)

```
   iOutput = 0
' Insert your code for output (FABody)
```



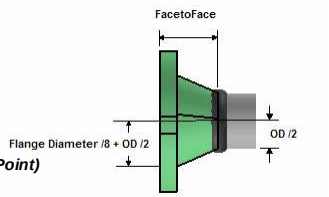

```
RetrieveParameters 1, oPartFclt, m_OutputColl, pipeDiam, flangeThick, flangeDiam,
   cptOffset, depth
   Dim stPosition   As IJDPosition
   Dim enPosition   As IJDPosition
   Set stPosition = New DPosition
   Set enPosition = New DPosition
   stPosition.Set -parFacetoFace / 2 + flangeThick, 0, 0
   enPosition.Set parFacetoFace / 2, 0, 0
   iOutput = iOutput + 1
   Set ObjBody1 = m_oGeomHelper.CreateCone(arrayOfOutputs(iOutput), stPosition,
   enPosition, pipeDiam  + flangeDiam / 4, pipeDiam )
```

Physical

SmartPlant 3D

Example 1: Code (Pipe Nozzle)

```
' Place Nozzle 1
    RetrieveParameters 1, oPartFclt, m_OutputColl, pipeDiam, flangeThick, flangeDiam, cptOffset, depth
    Dim oPlacePoint As AutoMath.DPosition
    Dim oDir       As AutoMath.DVector
    Dim objNozzle   As GSCADNozzleEntities.IJDNozzle
    Dim faceToFace  As Double
    Set oPlacePoint = New AutoMath.DPosition
    Set oDir = New AutoMath.DVector
    faceToFace = arrayOfInputs(2)
    oPlacePoint.Set -faceToFace / 2 - cptOffset + depth, 0, 0
    oDir.Set -1, 0, 0
    Set objNozzle = CreateNozzle(1, oPartFclt, m_OutputColl, oDir, oPlacePoint)
' Set the output
    iOutput = iOutput + 1
    m_OutputColl.AddOutput arrayOfOutputs(iOutput), objNozzle
    Set objNozzle = Nothing
' Place Nozzle 2
    RetrieveParameters 2, oPartFclt, m_OutputColl, pipeDiam, flangeThick, flangeDiam, cptOffset, depth
    oPlacePoint.Set faceToFace / 2 + cptOffset - depth, 0, 0
    oDir.Set 1, 0, 0
    Set objNozzle = CreateNozzle(2, oPartFclt, m_OutputColl, oDir, oPlacePoint)
' Set the output
    iOutput = iOutput + 1
    m_OutputColl.AddOutput arrayOfOutputs(iOutput), objNozzle
```



Methods

SmartPlant 3D

Geometry3D.bas

BodyOD
CreateCableTrayPort
CreateConduitNozzle
CreateNozzle
CreateNozzleJustaCircle
CreateNozzleWithLength
getOrthonormal
InsulationDiameter
InsulationLength
PlaceBox
PlaceFAN
PlaceProjection
PlaceRevolution
PlaceTorusSweep
PlaceTrapezoid
PlaceTrArcBy3Pts
PlaceTrArcByCenter
PlaceTrArcByCenterNorm
PlaceTrBspline
PlaceTrCString
PlaceTrCStringNoCheck
PlaceTrArcByCenterNorm
PlaceTrBspline
PlaceTrCString
PlaceTrCStringNoCheck
PlaceTrLine
PlaceTruncatedCylinderByHeights
ReportUnanticipatedError
RetrieveCableTrayPortProperties
RetrieveParameters
RetrieveParametersForThruBoltedEnds
RetrieveParameterswithPressureRating
RetrievePipeOD

IJSymbolGeometryHelper

Members of 'IJSymbolGeometryHelper'
- AddGeometry
- AutoTransformUpdate
- CreateChildPartOcc
- CreateCone
- CreateCylinder
- CreateMiteredTorus
- CreatePolygon
- CreatePrism
- CreateProjectedRectangle
- CreateProjectedShape
- CreateProjectedShapeByPoints
- CreateProjectedTriangle
- CreateRectangularTorus
- CreateSemiEllipsoid
- CreateSphere
- CreateTorus
- CreateTransitionalElement
- OutputCollection
- Transform

Custom Interfaces Sheet

InterfaceName	CategoryName	AttributeName	AttributeUserName	Type	UnitsType	PrimaryUnits	OnPropertyPage	ReadOnly	SymbolParameter
IJDryWeight1		DryWeight1	Dry Weight 1	Double	mass	lbm	TRUE	FALSE	
IJInsulationThickness		InsulationThickness	Insulation Thickness	Double	distance	mm	TRUE	FALSE	InsulationThickness
IJFaceToFace		FacetoFace	Face to Face	Double	distance	mm	TRUE	FALSE	FacetoFace
IJFaceToCenter		FacetoCenter	Face to Center	Double	distance	mm	TRUE	FALSE	FacetoCenter
		Face1toCenter	Face1 to Center	Double	distance	mm	TRUE	FALSE	Face1toCenter
		Face2toCenter	Face2 to Center	Double	distance	mm	TRUE	FALSE	Face2toCenter
IJOlet		HoleinHeader	Hole in Header	Double	distance	mm	TRUE	FALSE	C
		FacetoHeaderSurface	Face to Header Surface	Double	distance	mm	TRUE	FALSE	A
IJBendBranchAngle		Angle	Angle	Double	angle	deg	TRUE	FALSE	Angle
IJSecOperOrient		Rotation	Rotation	Double	angle	deg	TRUE	FALSE	HandwheelAngle

- 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.

Part Class

IndustryCommodityCode	FirstSizeSchedule	SecondSizeSchedule	CommodityType	GeometryType	GraphicalRepresentationOnNot	SymbolDefinition	PipingPointBasis[1]	Id[1]	PressureRating[1]	EndPreparation[1]	EndStandard[1]	ScheduleThickness[1]	FlowDirection[1]	PipingPointBasis[2]	Id[2]	PressureRating[2]	EndPreparation[2]	EndStandard[2]	ScheduleThickness[2]	FlowDirection[2]	Npd[1]	NpdUnitType[1]	Npd[2]	NpdUnitType[2]	FacetoFace
FAAAHDCZZAADABQZZUS		S-STD	FWN	15		SP3DFWN.CSP3DFWN	15		150	21	5		3	15			301	5	S-STD	3	2	in	2	in	2.5in
																					3	in	3	in	2.75in
																					4	in	4	in	3in
																					6	in	6	in	3.5in
																					8	in	8	in	4in

Inputs & Outputs

Example 2: (Input & Output Definitions)

```
' Inputs
   m_oSymbolHelper.NumInputs = 2
   m_oSymbolHelper.AddInputDef 1, "FacetoFace", "FacetoFace", 4
   m_oSymbolHelper.AddInputDef 2, "B", "MajorBodyDiameter", 3
' Outputs
   m_oSymbolHelper.NumOutputs = 3
   m_oSymbolHelper.AddOutputDef 1, "FABody", "FABody", 1
   m_oSymbolHelper.AddOutputDef 2, "PipingNoz1", "Nozzle 1", 1
   m_oSymbolHelper.AddOutputDef 3, "PipingNoz2", "Nozzle 2", 1
' Aspects
   m_oSymbolHelper.NumAspects = 1
   m_oSymbolHelper.AddAspectDef 1, "SimplePhysical", "SimplePhysical", 1
```


Physical

Example 2: Code (Body)

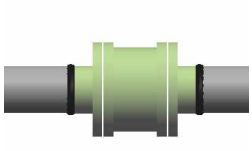
```
   iOutput = 0
' Insert your code for output (FABody)

   RetrieveParameters 1, oPartFclt, m_OutputColl, pipeDiam, flangeThick, flangeDiam,
    cptOffset, depth
   Dim stPoint   As IJDPosition
   Dim enPoint   As IJDPosition
   Set stPoint = New DPosition
   Set enPoint = New DPosition
   stPoint.Set -parFacetoFace / 2 + flangeThick, 0, 0
   enPoint.Set parFacetoFace / 2 - flangeThick, 0, 0
' Set the output
   iOutput = iOutput + 1
   Set ObjFABody = m_oGeomHelper.CreateCylinder(arrayOfOutputs(iOutput), stPoint,
    enPoint, parMajorBodyDiameter)
```

Physical

SmartPlant 3D

Example 2: Code (Pipe Nozzle)

```
' Place Nozzle 1
   RetrieveParameters 1, oPartFclt, m_OutputColl, pipeDiam, flangeThick, flangeDiam, cptOffset, depth
   Dim oPlacePoint As AutoMath.DPosition
   Dim oDir       As AutoMath.DVector
   Dim objNozzle   As GSCADNozzleEntities.IJDNozzle
   Dim faceToFace  As Double
   Set oPlacePoint = New AutoMath.DPosition
   Set oDir = New AutoMath.DVector
   faceToFace = arrayOfInputs(2)
   oPlacePoint.Set -faceToFace / 2 - cptOffset + depth, 0, 0
   oDir.Set -1, 0, 0
   Set objNozzle = CreateNozzle(1, oPartFclt, m_OutputColl, oDir, oPlacePoint)
' Set the output
   iOutput = iOutput + 1
   m_OutputColl.AddOutput arrayOfOutputs(iOutput), objNozzle
   Set objNozzle = Nothing
' Place Nozzle 2
   RetrieveParameters 2, oPartFclt, m_OutputColl, pipeDiam, flangeThick, flangeDiam, cptOffset, depth
   oPlacePoint.Set faceToFace / 2 + cptOffset - depth, 0, 0
   oDir.Set 1, 0, 0
   Set objNozzle = CreateNozzle(2, oPartFclt, m_OutputColl, oDir, oPlacePoint)
' Set the output
   iOutput = iOutput + 1
   m_OutputColl.AddOutput arrayOfOutputs(iOutput), objNozzle
```



SmartPlant 3D

Instrument & Piping Specialty Placement

We have two types of piping specialty/instrument Parts:

- 1. **Stock item:** Stock items represent those piping items that are purchased from a manufacturer's catalog, where no real engineering is required other than selecting the correct size, material, etc.
- 2. **Custom-engineered item:** custom engineered items are built items according to the process.

Instrument & Piping Specialty Placement

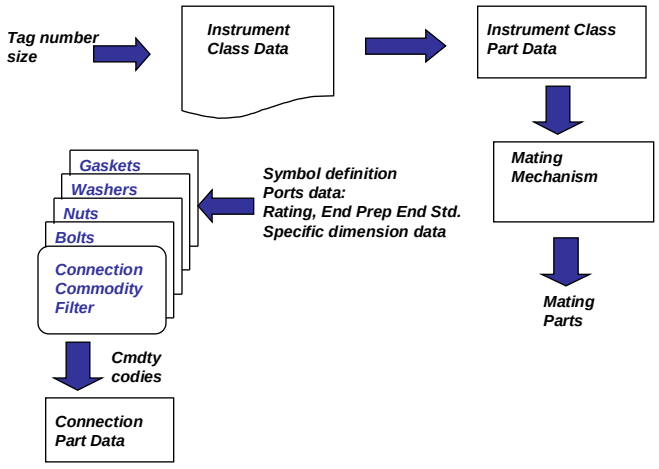
For custom engineered piping specialty/instrument:

- The catalog data will be based on the tag number or the generic tag number and the size.

For stock piping specialty/instrument:

- The catalog data will be determined in the same manner as a piping commodity, i.e. based on the contractor commodity code and the size.

Table look up for an engineered instrument



Insert an engineered instrument by tag number

Instrument class data

TagNumber	GenericTagNumber	SpecName	FirstSizeFrom	FirstSizeTo	FirstSizeUnits	SecondSizeFrom	SecondSizeTo	SecondSizeUnits	MultiSizeOption	RequisitionType	ShortMaterialDescription	FabricationType	SupplyResponsibility	ReportingType	GasketRequirements	BoltingRequirements	WeldingRequirement
F001										10	Generic Component	7	10		5	5	50

Generic Instrument

IndustryCommodityCode	CommodityType	GeometryType	RequisitionType	SymbolDefinition	PipingPointBasis[1]	Id[1]	PressureRating[1]	EndPreparation[1]	EndStandard[1]	ScheduleThickness[1]	FlowDirection[1]	PipingPointBasis[2]	Id[2]	PressureRating[2]	EndPreparation[2]	EndStandard[2]	ScheduleThickness[2]	FlowDirection[2]	DryWeight	Npd[1]	NpdUnitType[1]	Npd[2]	NpdUnitType[2]	FacetoFace	MajorBodyDiameter
F001	121	15	10	GenericComp.CGenericComp			150	21	5		3			150	21	5		3		4	in	4	in	6in	8in

Troubleshooting Symbols

Sources of Errors

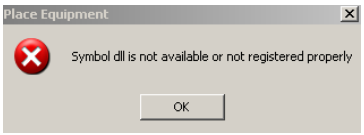
- Setting incorrect parameter values in the catalog. For example, Missing the outside pipe diameter in the generic table and therefore the pipe outside diameter is set to zero
- Check for incomplete or wrong definition
- Check for wrong versions of a symbol definition
- Check for input mismatches
- Check for output mismatches

Troubleshooting Symbols

Sources of Errors

Incorrect Class Name or incorrect number of inputs

***** Error *****
Time : 06/27/05 13:00:39
ID : 0x800401f8(-2147221000)(504);
Source: M:\Refdata\Middle\Entities\PartDef\DEX\PartClass.cpp
Desc: get_SymbolDefinition : Failed to get the SymbolDefinition from ProgId.
Help: ;0
Src : M:\Refdata\Middle\Entities\PartDef\DEX\PartClass.cpp;980.



Troubleshooting Symbols

Sources of Errors

Incorrect mapping to the symbol parameters

- ***** Error *****
- **Time : 06/27/05 13:07:20**
- **ID : 0x80040402(-2147220478)(1026);Symbol**
- **Source: GSCADPartOcc.PartOcc.1**
- **Desc: FaceToFace**
- **Help: ;0**
- **Src : M:\CommonApp\Middle\Include\OccurrenceObjectImpl.h;138**
- ***** Error *****
- **Time : 06/27/05 13:07:20**
- **ID : 0x80040402(-2147220478)(1026);**
- **Source: M:\CommonApp\Middle\Include\OccurrenceObjectImpl.h**
- **Desc: SetParamsFromProperties:SetParamsFromProperties2 failed**
- **Help: ;0**
- **Src : M:\CommonApp\Middle\Include\OccurrenceObjectImpl.h;1009**

Troubleshooting Symbols

Sources of Errors

Incorrect number of outputs

***** Error *****
Time : 06/27/05 13:33:56
ID : 0x80004005(-2147467259)(16389);Source: IMSSymbolEntities.SblDefinition.1
Desc: The VB function for the Representation SimplePhysical of Symbol GenericComp.CGenericComp failed. Please check your source code
Src : X:\Middle\Symbol\Src\SblEntities\SblDefinition.cpp;1368

Troubleshooting Symbols

Sources of Errors

Incorrect values passing to any geometry factory function

Set ObjNozzleBody = m_oGeomHelper.CreateCylinder(arrayOfOutputs(iOutput), pos1, pos2, pipeDiam)

***** Error *****
Time : 06/27/05 13:51:33
ID : 0x80004005(-2147467259)(16389);UserSymbol
Source: GenericComp.CSimplePhysical::Run
Desc: Invalid procedure call or argument
Help: ;0
Info:
Data:
Src : ;0

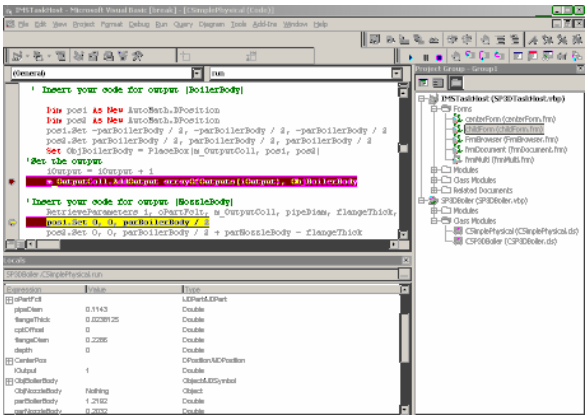
Troubleshooting Symbols

Debugging the symbol

Adding the TaskHost Project

Environment variable:

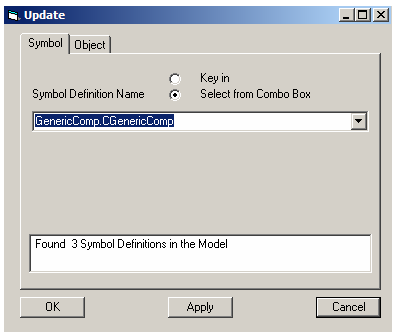
PATH=[*Product Directory*]\Core\Runtime;
[*ProductDirectory*]\GeometryTopology\Runtime



Troubleshooting Symbols

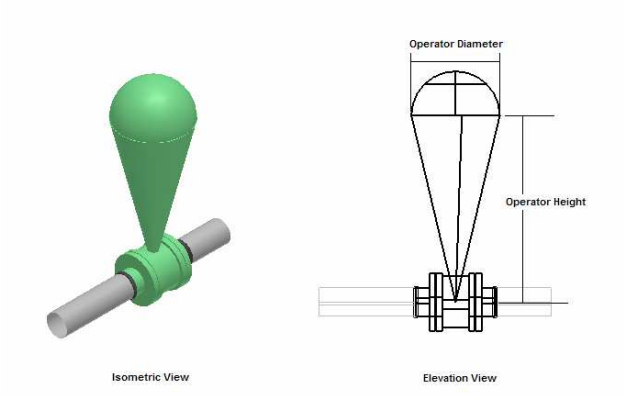
Custom Command

Update Symbol Occurrences after making changes on the Symbol Definition (SymbolTestCmds.CUpdateSymbolDefinition)



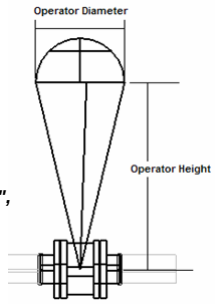
Valve Operator Symbol

Engineered Instrument with valve operator



Inputs & Outputs

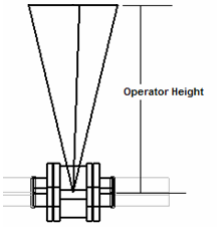
Example 2: (Input & Output Definitions)




```
' Inputs
   m_oSymbolHelper.NumInputs = 2
   m_oSymbolHelper.AddInputDef 1, "OperatorHeight", "OperatorHeight", 1
   m_oSymbolHelper.AddInputDef 2, "OperatorDiameter", "OperatorDiameter",
    0.5

' Outputs
   m_oSymbolHelper.NumOutputs = 3
   m_oSymbolHelper.AddOutputDef 1, "OPBody1", "OPBody1", 1
   m_oSymbolHelper.AddOutputDef 2, "OPBody2", "OPBody2", 1
   m_oSymbolHelper.AddOutputDef 3, "OPBody3", "OPBody3", 1

' Aspects
   m_oSymbolHelper.NumAspects = 1
   m_oSymbolHelper.AddAspectDef 1, "SimplePhysical", "SimplePhysical", 1
```

Physical

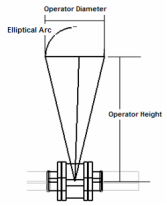
Example 2: Code (OP431)




```
iOutput = 0
' Insert your code for output (OPBody)
Dim ConeCenterBasePt    As IJDPosition
Dim ConeCenterTopPt     As IJDPosition
Set ConeCenterBasePt = New DPosition
Set ConeCenterTopPt = New DPosition

'   A value of 0.0000001 is used to avoid symbol placement failure (gives assertion errors).
ConeCenterTopPt.Set 0, parOperatorHeight, 0
ConeCenterBasePt.Set 0, 0, 0.0000001

Dim ObjOPBody  As IngrGeom3D.Cone3d
iOutput = iOutput + 1

'   A value of 0.00001 is used to avoid symbol placement failure (gives assertion errors).

Set ObjOPBody = m_oGeomHelper.CreateCone(arrayOfOutputs(iOutput), ConeCenterBasePt, _
ConeCenterTopPt, 0.00001, parOperatorDiameter)
```


Physical

Example 2: Code (OP431)




```
Dim oGeomFactory As New IngrGeom3D.GeometryFactory
Dim oGeomFactory As New IngrGeom3D.GeometryFactory
Dim dCircleCenterX As Double, dCircleCenterY As Double, dCircleCenterZ As Double
Dim oEllipticalArc As IngrGeom3D.EllipticalArc3d
Dim oRevolution As IngrGeom3D.Revolution3d
Dim PI As Double
Dim dRadius As Double

'Normal is North=  z-axis
PI = 4 * Atn(1)
dRadius = parOperatorDiameter / 2
Set oEllipticalArc =
   oGeomFactory.EllipticalArcs3d.CreateByCenterNormalMajAxisRatioAngle(Nothing, _
                    0, parOperatorHeight, 0, 0, 0, 1, 0, dRadius, 0, 1, PI * 1.5, PI / 2)

Set oRevolution = oGeomFactory.Revolutions3d.CreateByCurve(m_OutputColl.ResourceManager,
                    oEllipticalArc, 0, 1, 0, 0, 0, 0, 2 * PI, False)
iOutput = iOutput + 1
m_OutputColl.AddOutput arrayOfOutputs(iOutput), oRevolution
```

Physical

Example 2: Code (OP431)

```
Dim oCircle As IngrGeom3D.Circle3d
Set oCircle =
   oGeomFactory.Circles3d.CreateByCenterNormalRadius(m_OutputColl.ResourceManager, _
         0, parOperatorHeight, 0, 0, 1, 0, dRadius)

iOutput = iOutput + 1
m_OutputColl.AddOutput arrayOfOutputs(iOutput), oCircle
```

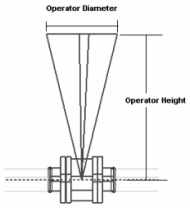

Physical

Example 2: Code (OP431)

```
Set oCircle = Nothing
   Set oRevolution = Nothing
   Set oEllipticalArc = Nothing
   Set oGeomFactory = Nothing
   Set ConeCenterBasePt = Nothing
   Set ConeCenterTopPt = Nothing
```

Inputs & Outputs

Example 2: (GenericComp)

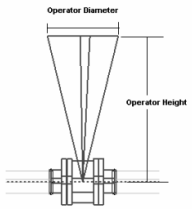



```
' Outputs
    m_oSymbolHelper.NumOutputs = 4
    m_oSymbolHelper.AddOutputDef 1, "FABody", "FABody", 1
    m_oSymbolHelper.AddOutputDef 2, "PipingNoz1", "Nozzle 1", 1
    m_oSymbolHelper.AddOutputDef 3, "PipingNoz2", "Nozzle 2", 1
    m_oSymbolHelper.AddOutputDef 4, "ValveOperator", "ValveOperator", 1
```


Physical

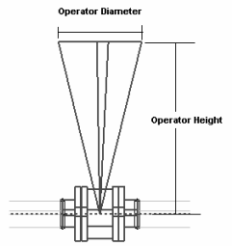
Example 2: (GenericComp)

```
Dim oSymbolHelper As IJSymbolGeometryHelper
    Set oSymbolHelper = New SP3DSymbolHelper.SymbolServices
    oSymbolHelper.OutputCollection = m_OutputColl

    On Error Resume Next
    Dim oDirX As IJDVector
    Dim oDirY As IJDVector
    Dim oDirZ As IJDVector
    Set oDirX = New DVector
    Set oDirY = New DVector
    Set oDirZ = New DVector

    oDirX.Set 1, 0, 0
    oDirY.Set 0, 1, 0
    oDirZ.Set 0, 0, 1
```



Physical

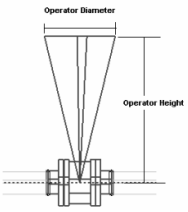
Example 2: (HandwheelAngle)




```
Dim oDirX As IJDVector
Dim oDirY As IJDVector
Dim oDirZ As IJDVector
Set oDirX = New DVector
Set oDirY = New DVector
Set oDirZ = New DVector

oDirX.Set Cos(parHandwheelAngle), 0, Sin(parHandwheelAngle)
oDirY.Set 0, 1, 0
oDirZ.Set -Sin(parHandwheelAngle), 0, Cos(parHandwheelAngle)
```


Physical

Example 2: (GenericComp)




```
Dim oPipeComponent As IJDPipeComponent
   Set oPipeComponent = oPartFclt
   On Error GoTo ErrorLabel
   Dim oOperatorPart As IJDPart
   Dim oOperatorOcc   As IJPartOcc
   If Not oPipeComponent Is Nothing Then
      Set oOperatorPart = oPipeComponent.GetValveOperatorPart
      If Not oOperatorPart Is Nothing Then
         Dim OpOrigin As IJDPosition
         Set OpOrigin = New DPosition
         OpOrigin.Set 0, 0, 0
         Set oOperatorOcc = oSymbolHelper.CreateChildPartOcc("ValveOperator",
 oOperatorPart, OpOrigin, oDirX, oDirY, oDirZ)
      End If
   End If
   Set oSymbolHelper = Nothing
   Set oOperatorPart = Nothing
   Set oPipeComponent = Nothing
   Set oOperatorOcc = Nothing
```

Operator Part Class

SmartPlant 3D

Definition	PartClassType	SymbolDefinition
	ValveOperatorClass	SP3DOP431.CSP3DOP431

CommodityPart

Head	ValveOperatorNumber	ValveSize	ValveSizeUnits	SymbolDefinition	MirrorBehaviorOption	ValveOperatorIsRotatable	DryWeight	DryCogX	DryCogY	DryCogZ	OperatorHeight	OperatorDiameter
Start												
	GAT-BLT-150-431	4	in		5						29in	13.75in

Instrument Class Data

SmartPlant 3D

TagNumber	GenericTagNumber	SpecName	FirstSizeFrom	FirstSizeTo	FirstSizeUnits	SecondSizeFrom	SecondSizeTo	SecondSizeUnits	MultiSizeOption	RequisitionType	ContractorCommodityCode	InstrumentType	GeometryType	SubstCapScrewDiameter	TappedHoleDepth	ValveOperatorType	ValveOperatorGeoIndStd	ValveOperatorCatalogPartNumber
F001										10			15			431	2080	GAT-BLT-150-431

OperatorPartNumber	ShortMatlDescription	LocalizedShortMaterialDescription	LongMaterialDescription
GAT-BLT-150-431	Diaphragm		Diaphragm

Symbol Modification

SmartPlant 3D

Some rules when editing a symbol definition

Parametric and associative inputs

- The deletion of an input is not effective since the input will be just invalidated, and its index won't be reused.
- A new input can be added to the definition. Must increase the version number of the dll and Update the symbol definition using bulkload utility.

Representations and outputs

- Symbol must define the physical representation.
- The name of an output cannot be modified.
- A new output can be added to a representation.
- A new representation can be added. Must increase the version number of the dll and Update the symbol definition using bulkload utility.
- The deletion of a representation is not effective since the representation will be just invalidated.

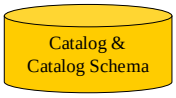
Change on the construction of the graphical entities

Reference Data Management

SmartPlant 3D

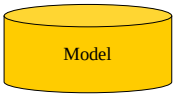
Step 1

Bulk Load Utility



Step 2

View Generator Utility or Synchronize Model with Catalog Command



Step 3

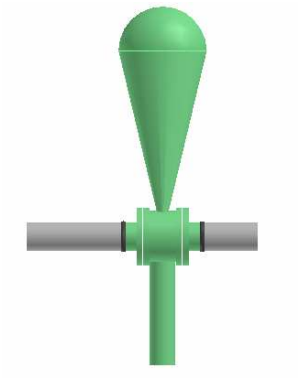
Re-generate the reports database command

- 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.

Symbol with Multiple Representations

SmartPlant 3D

Engineered Instrument with operation volume



Inputs & Outputs

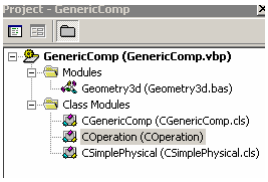
SmartPlant 3D

Example 3: (GenericComp)

```
' Outputs
   m_oSymbolHelper.NumOutputs = 5
   m_oSymbolHelper.AddOutputDef 1, "FABody", "FABody", 1
   m_oSymbolHelper.AddOutputDef 2, "PipingNoz1", "Nozzle 1", 1
   m_oSymbolHelper.AddOutputDef 3, "PipingNoz2", "Nozzle 2", 1
   m_oSymbolHelper.AddOutputDef 4, "ValveOperator", "ValveOperator", 1
   m_oSymbolHelper.AddOutputDef 5, "Operation1", "Operation1", 64

 m_oSymbolHelper.NumAspects = 2
   m_oSymbolHelper.AddAspectDef 1, "SimplePhysical", "SimplePhysical", 1
   m_oSymbolHelper.AddAspectDef 2, "Operation", "Operation", 64
```

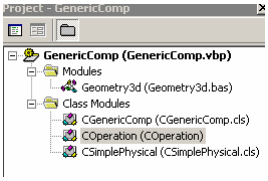
Operation Class Module

Example 3: (GenericComp)




```
Option Explicit
Private m_oGeomHelper As IJSymbolGeometryHelper
Private Const E_FAIL = &H80004005

Private Sub Class_Initialize()
   Const METHOD = "Class_Initialize"
   On Error GoTo Errx
  Set m_oGeomHelper = New SymbolServices
Exit Sub
Errx:
   Err.Raise Err.Number, Err.Source & " " & METHOD, Err.Description, _
      Err.HelpFile, Err.HelpContext
End Sub

Private Sub Class_Terminate()
   Set m_oGeomHelper = Nothing
End Sub
```


Operation Class Module



Example 3 (GenericComp)

```
Dim stPoint   As New AutoMath.DPosition
   Dim enPoint   As New AutoMath.DPosition
   stPoint.Set 0, - parMajorBodyDiameter / 2, 0
   enPoint.Set 0, - parMajorBodyDiameter / 2 - 2 *
   parMajorBodyDiameter, 0

' Set the output
   iOutput = iOutput + 1
   Set ObjFABody =
   m_oGeomHelper.CreateCylinder(arrayOfOutputs(iOutput), stPoint,
   enPoint, parMajorBodyDiameter / 2)
   Set stPoint = Nothing
   Set enPoint = Nothing
```

Binary Compatibility

- A VB ActiveX DLL/OCX is a COM Class
  - That is, it has a CLSID
- When you build DLL, VB generates a *new* CLSID
- Project that have your ActiveX DLL as a reference are then "broken"
  - Reference is marked "Missing"
- Solution: when you get a good build, save the DLL as a Binary-Compatible reference mark
  - Make all future builds binary compatible to it
  - This preserves the CLSID (& other project's references!)

Recommendation

- Always test the symbol on a test catalog.
- Backup your catalog before bulkload your test reference data.
- A Symbol Definition may have many thousands of Symbols connected to it. Therefore, Use the Synchronize Model with Catalog command to re-compute the symbol definition instead of using the Custom Command.

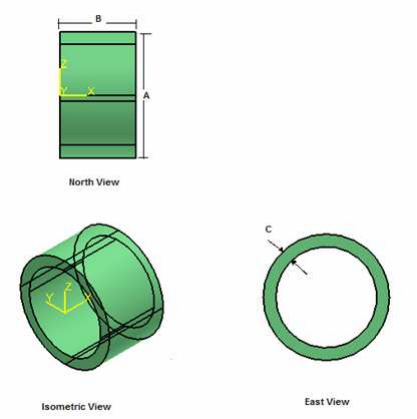
Excel

- Worksheet size 65,536 rows by 256 columns
- Column width 255 characters
- Sheets in a workbook Limited by available memory

Shape Symbol

SmartPlant 3D

Equipment Shape symbol



Inputs/Outputs

SmartPlant 3D

Example:

```
' Inputs
   m_oSymbolHelper.NumInputs = 3
   m_oSymbolHelper.AddInputDef 1, "A", "A", 2
   m_oSymbolHelper.AddInputDef 2, "B", "B", 1
   m_oSymbolHelper.AddInputDef 3, "C", "C", 0.4

' Outputs
   m_oSymbolHelper.NumOutputs = 8
   m_oSymbolHelper.AddOutputDef 1, "Body1", "Body1", 1
   m_oSymbolHelper.AddOutputDef 2, "Body2", "Body2", 1
   m_oSymbolHelper.AddOutputDef 3, "Body3", "Body3", 1
   m_oSymbolHelper.AddOutputDef 4, "Body4", "Body4", 1
   m_oSymbolHelper.AddOutputDef 5, "Body5", "Body5", 1
   m_oSymbolHelper.AddOutputDef 6, "Body6", "Body6", 1
   m_oSymbolHelper.AddOutputDef 7, "Body7", "Body7", 1
   m_oSymbolHelper.AddOutputDef 8, "Body8", "Body8", 1

' Aspects
   m_oSymbolHelper.NumAspects = 1
   m_oSymbolHelper.AddAspectDef 1, "SimplePhysical", "SimplePhysical", 1)
```

Physical

```
' Inputs
    Set oPartFclt = arrayOfInputs(1)
    parA = arrayOfInputs(2)
    parB = arrayOfInputs(3)
    parC = arrayOfInputs(4)
    m_oGeomHelper.OutputCollection = m_OutputColl

    iOutput = 0

        Dim oErrors As IJEditErrors
    Set oErrors = New JServerErrors
    If parA <= 0 Or parB <= 0 Or parC <= 0 Then
        oErrors.Add E_FAIL, "CSP3DHollowCy", "Shape Dimensions
should be greater than zero", "ZeroOrNegative"
        GoTo Errx:
    End If
```


Physical

```
Dim oGeomFactory As New GeometryFactory
    Dim oCircle(2) As Circle3d
    Dim oProjection As Projection3d
    Dim oDir As IJDVector
    Set oDir = New DVector
    oDir.Set 1, 0, 0
'create the cylinders
    Set oCircle(1) =
oGeomFactory.Circles3d.CreateByCenterNormalRadius(m_OutputColl.Resource
Manager, 0, 0, 0, 1, 0, 0, parA / 2)
    Set oCircle(2) =
oGeomFactory.Circles3d.CreateByCenterNormalRadius(m_OutputColl.Resource
Manager, 0, 0, 0, 1, 0, 0, parA / 2 - parC)

    iOutput = iOutput + 1
    m_OutputColl.AddOutput arrayOfOutputs(iOutput), oCircle(1)
    iOutput = iOutput + 1
    m_OutputColl.AddOutput arrayOfOutputs(iOutput), oCircle(2)
```

Physical

```
Set oProjection = PlaceProjection(m_OutputColl, oCircle(1), oDir, parB, False)

   iOutput = iOutput + 1
   m_OutputColl.AddOutput arrayOfOutputs(iOutput), oProjection

   Set oProjection = PlaceProjection(m_OutputColl, oCircle(2), oDir, parB, False)

   iOutput = iOutput + 1
   m_OutputColl.AddOutput arrayOfOutputs(iOutput), oProjection
```


Physical

```
'create the left face
   Dim oPlane  As IngrGeom3D.Plane3d
   Set oPlane =
oGeomFactory.Planes3d.CreateByPointNormal(m_OutputColl.ResourceManager, _
0, 0, 0, 1, 0, 0)
   Dim oElements      As IJElements
   Dim objCStr        As IngrGeom3D.ComplexString3d
   Dim i As Integer

   Set oElements = New JObjectCollection

   For i = 1 To 2
      oElements.Add oCircle(i)
      Set objCStr = oGeomFactory.ComplexStrings3d.CreateByCurves(Nothing, oElements)
      oPlane.AddBoundary objCStr
      oElements.Clear
      objCStr.RemoveCurve True
   Next i
   iOutput = iOutput + 1
   m_OutputColl.AddOutput arrayOfOutputs(iOutput), oPlane
```

Physical

```
'create the right face
    Set oCircle(1) = oGeomFactory.Circles3d.CreateByCenterNormalRadius(m_OutputColl.ResourceManager, parB, 0, 0, 1, 0, 0, parA / 2)

    Set oCircle(2) = oGeomFactory.Circles3d.CreateByCenterNormalRadius(m_OutputColl.ResourceManager, parB, 0, 0, 1, 0, 0, parA / 2 - parC)

    iOutput = iOutput + 1
    m_OutputColl.AddOutput arrayOfOutputs(iOutput), oCircle(1)

    iOutput = iOutput + 1
    m_OutputColl.AddOutput arrayOfOutputs(iOutput), oCircle(2)
```


Physical

```
'create the right face

    Set oPlane = oGeomFactory.Planes3d.CreateByPointNormal(m_OutputColl.ResourceManager, parB, 0, 0, 1, 0, 0)
    For i = 1 To 2
        oElements.Add oCircle(i)
        Set objCStr = oGeomFactory.ComplexStrings3d.CreateByCurves(Nothing, oElements)
        oPlane.AddBoundary objCStr
        oElements.Clear
        objCStr.RemoveCurve True
    Next i
    iOutput = iOutput + 1
    m_OutputColl.AddOutput arrayOfOutputs(iOutput), oPlane
```

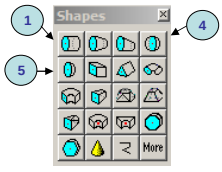
Physical

```
Set oProjection = Nothing
   Set oCircle(1) = Nothing
   Set oCircle(2) = Nothing
   Set oGeomFactory = Nothing
   Set oPlane = Nothing
   Set oDir = Nothing
   Set oElements = Nothing
   Set objCStr = Nothing
```


Shape Palette

Shape Part Class

Definition	PartClassType	SymbolDefinition	SymbolIcon	oa:IJUAHollowCy::A	oa:IJUAHollowCy::B	oa:IJUAHollowCy::C	
a	ShapesClass	SP3DHollowCy.CSP3DHollowCy	SymbolIcons\HollowCy.bmp				
Head	PartNumber	PartDescription	SymbolDefinition	IJUAHollowCy::A	IJUAHollowCy::B	IJUAHollowCy::C	IJUAPaletteInfo::SequenceNumber
Start							
a	HollowCy 001	Hollow Cylinder		100	60	10	19
End							



Shape Palette

SmartPlant 3D

ShapeTypes.xml

Located in symbol share \\Symbols\ShapeTypes

```
<ShapeTypes>
   <ShapeType name="DatumShape" picture= "DatumShape.bmp">
</ShapeType>
   <ShapeType name="Sphere" picture= "Sphere.bmp">
</ShapeType>
   <ShapeType name="CircularTori" picture= "CircularTori.bmp">
</ShapeType>
   <ShapeType name="EccentricCone" picture= "EccentricCone.bmp">
</ShapeType>
```