ArcGIS® 9

Designing Geodatabases With Visio®

Copyright © 1999–2005 ESRI
All rights reserved.
Printed in the United States of America.

The information contained in this document is the exclusive property of ESRI. This work is protected under United States copyright law and the copyright laws of the given countries of origin and applicable international laws, treaties, and/or conventions. No part of this work may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying and recording, or by any information storage or retrieval system, except as expressly permitted in writing by ESRI. All requests should be sent to Attention: Contracts Manager, ESRI, 380 New York Street, Redlands, CA 92373-8100, USA.

The information contained in this document is subject to change without notice.

CONTRIBUTING WRITERS

Andrew Perencsik, Eddie Idolyantes, Bob Booth, and Julio Andrade.

U.S. GOVERNMENT RESTRICTED/LIMITED RIGHTS

Any software, documentation, and/or data delivered hereunder is subject to the terms of the License Agreement. In no event shall the U.S. Government acquire greater than RESTRICTED/LIMITED RIGHTS. At a minimum, use, duplication, or disclosure by the U.S. Government is subject to restrictions as set forth in FAR §52.227-14 Alternates I, II, and III (JUN 1987); FAR §52.227-19 (JUN 1987) and/or FAR §12.211/12.212 (Commercial Technical Data/Computer Software); and DFARS §252.227-7015 (NOV 1995) (Technical Data) and/or DFARS §227.7202 (Computer Software), as applicable. Contractor/Manufacturer is ESRI, 380 New York Street, Redlands, CA 92373-8100, USA.

ESRI, ArcView, ArcInfo, ArcSDE, ArcCatalog, and ArcGIS are trademarks, registered trademarks, or service marks of ESRI in the United States, the European Community, or certain other jurisdictions.

The names of other companies and products herein are trademarks or registered trademarks of their respective trademark owners.

Designing Geodatabases With Visio

IN THIS DOCUMENT

- **Designing the object model in Microsoft Visio**

The PDF file Introduction_to_CASE_tools introduces you to designing geodatabases in UML.

This document discusses how you can use Microsoft® Visio® to construct your UML and how you can use CASE tools in ArcCatalog™ to generate the schema for your geodatabase. You will learn how to:

- Create UML packages and static structure diagrams.
- Create feature datasets.
- Create feature classes.
- Create relationship classes.
- Create domains.
- Create subtypes.
- Create geometric networks.
- Create connectivity rules.
- Extend classes with custom behavior.
- Export your UML model.
- Generate schema.

CASE tool functionality does not apply to ArcView® licenses.

Designing the object model in Microsoft Visio

When you begin creating your UML model, you will start with one of the ArcInfo® UML Model diagrams that were installed with ArcGIS®. These Visio drawing templates are located under your ArcGIS installation in the casetools\UML Models directory.

The ArcInfo UML Model diagram contains the object model required for using UML to model your geodatabase. The object model has five packages:

- Logical View
- ESRI Classes
- ESRI Interfaces
- ESRI Network
- Workspace

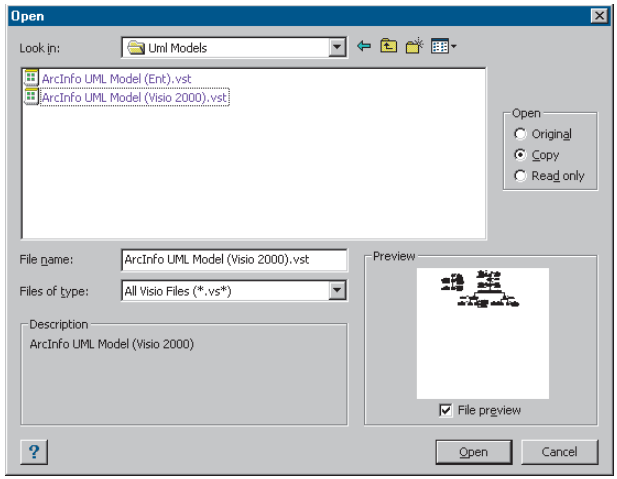
These UML packages act as directories where different parts of the entire object model are maintained. The Logical View package is the root level and contains the other three packages. Database designers and developers can use the Workspace package to create their object and database designs. It is possible to create more packages if the complexity of the model requires you to do so.

The ESRI® Classes package contains the portion of the GeoData Access Components necessary to create object models. Classes in this package represent components that are used to access spatial data sources, including geodatabases. Feature classes and object classes in your object models will inherit from these classes. The ESRI Interfaces package contains the definition of the interfaces implemented by the components shown in the ESRI Classes package. The interfaces are used only for code generation when creating custom objects.

The tasks in this document demonstrate how you can use the ArcInfo UML Model diagram to model the pieces of your geodatabase design. For additional information on UML modeling, see *http://support.esri.com/geodatabase/uml*.

Designing the object model

1. Start Visio.
2. Click File and click Open.
3. Browse to the CaseTools\UML Models folder where you installed ArcGIS. Double-click the ArcInfo UML Model template file.



Creating UML packages and static structure diagrams

Packages are a convenient way to organize your UML model. They act as folders where you can group model elements. As with folders on a disk, you can create a hierarchy of packages in your model.

You can create as many packages as you want. For example, you could have a model that has the following packages: Domains, ElectricUtils, and Landbase.

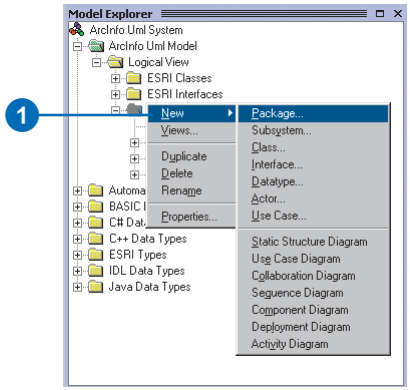
Before you begin, create a new diagram using the ArcInfo UML Model Visio templates. These templates are located under your ArcGIS installation in the casetools\UML Models directory.

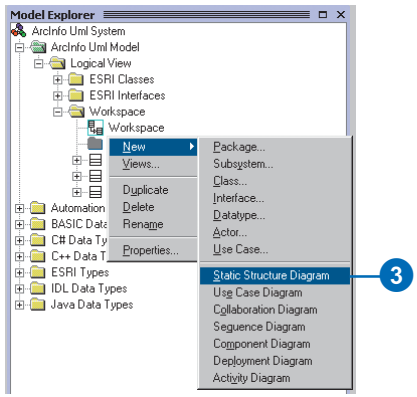
All of the tasks for creating UML diagrams that are discussed here are performed within the Visio application.

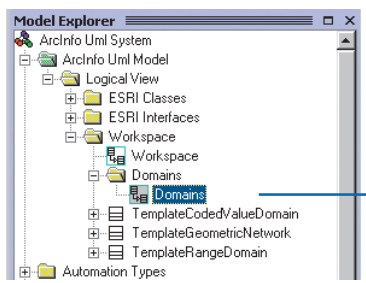
See Also

Special packages are used to model feature datasets. See 'Creating feature datasets' later in this document.

1. In the Model Explorer tree, right-click Workspace, point to New, then click Package.
2. Type the name of the package and click OK.
3. Right-click the new package. Point to New and click Static Structure Diagram to create a new class diagram.







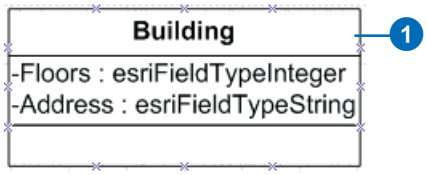
A new class diagram is created in the Model Explorer.

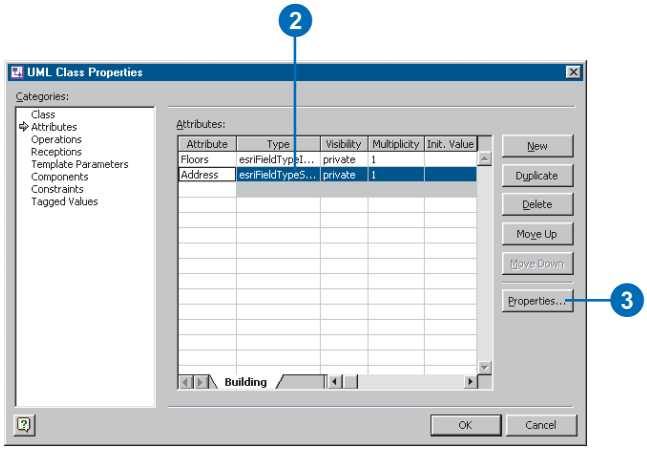
Setting tagged values

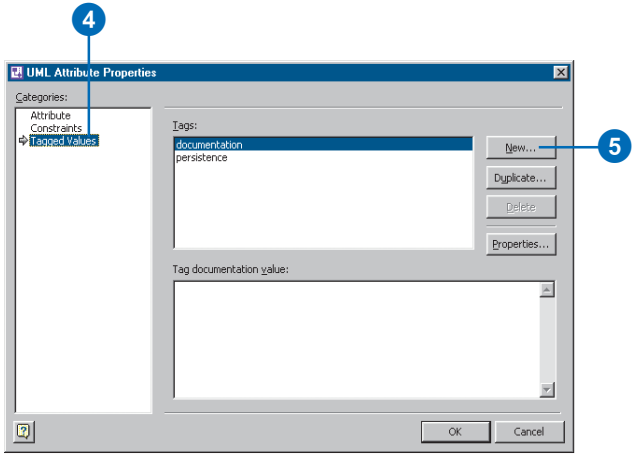
Tagged values are used to set additional properties of UML elements. For example, you can set the length (in characters) of a string field by using a tagged value.

Tagged values are recognized on several other UML elements: class, attribute, association, and so on.

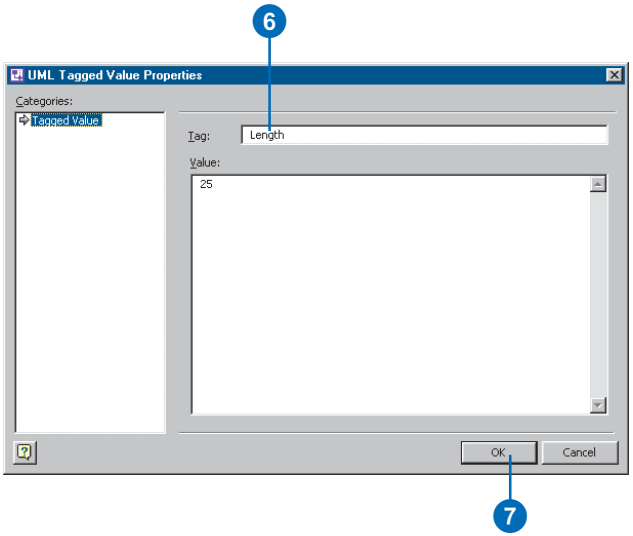
1. Double-click the UML class in the Visio diagram.
2. Click Attributes in the Categories window and double-click a string-typed attribute.
3. Click Properties.
4. Click Tagged Values in the Categories window.
5. Click New to create a new tagged value. ▸







6. Type “Length” in the Tag text box and type the length of the field in the Value text box.
7. Click OK.



Creating feature datasets

Feature datasets are modeled in UML as stereotyped packages. Other geodatabase elements—feature classes, for example—defined under the package will be created under the feature dataset. The name of the UML package will become the name of the feature dataset.

The spatial reference of a feature dataset is set while running the Schema Wizard.

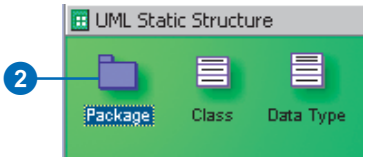
Tip

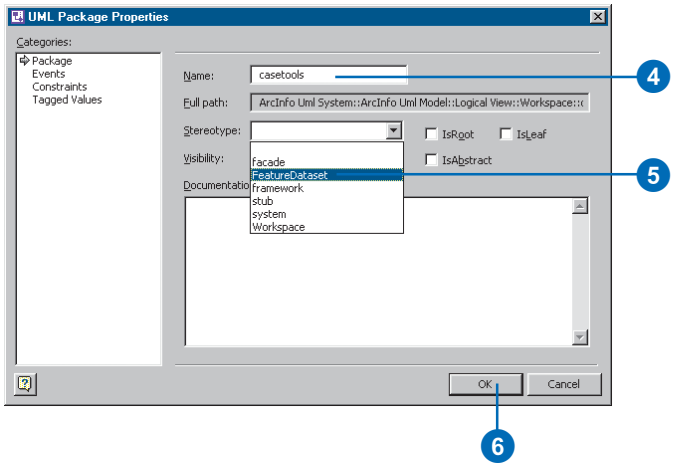
Class diagrams

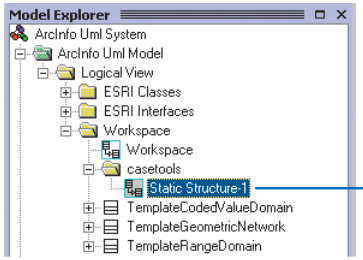
Creating several static structure diagrams can help you reduce diagram cluttering. A package can contain many static structure diagrams, and UML elements can appear in any diagram.

1. In the Model Explorer, double-click the Workspace diagram to open it.
2. Click Package in the UML Static Structure stencil and drag it onto the diagram.
3. Double-click the new package.
4. Type a name for the package.
5. Click the Stereotype dropdown arrow and click FeatureDataset.
6. Click OK.

   A new package and a new drawing are created in the Model Explorer.







A new package and a new drawing are created in the Model Explorer.

Creating feature classes

Feature classes are represented by UML classes in the model. You can model fields, geometry type, and other characteristics of the feature class in the UML class.

Feature classes can be created in the Workspace UML package or in a feature dataset package.

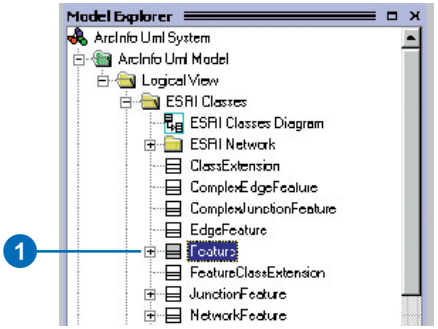
You can add fields to a feature class by adding attributes to the UML class.

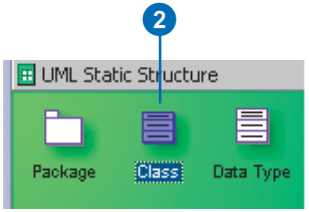
The field type is one of the values of the esriFieldType enumeration—for example, esriFieldTypeInteger.

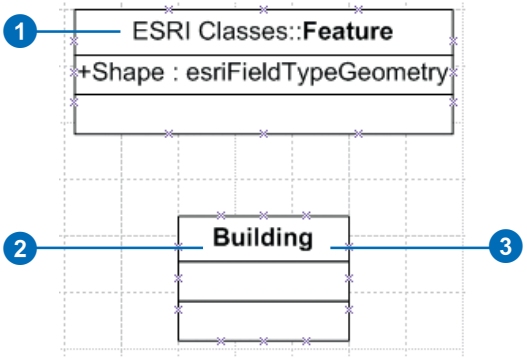
A domain created beforehand can be used as the attribute type as well.

Creating a feature class

1. In the Model Explorer tree under ESRI Classes, click the parent class and drag and drop it onto the diagram.
2. Click Class in the UML Static Structure stencil and drag and drop a new UML class onto the diagram.
3. Double-click the new class in the diagram. ▸







Tip

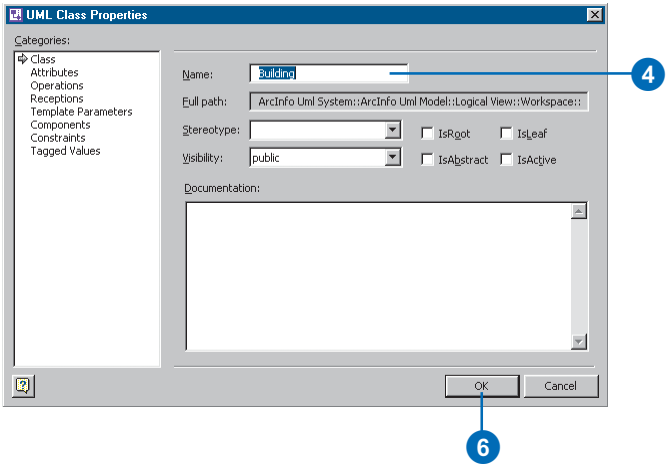
Abstract classes

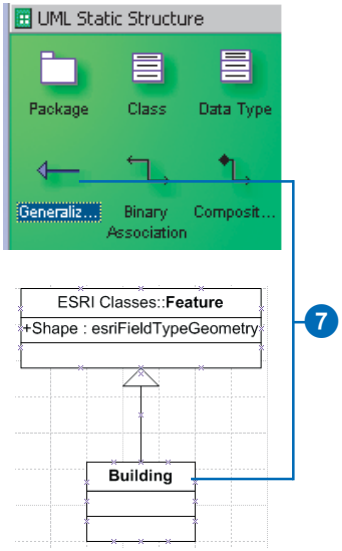
Your model may include abstract classes. Fields in abstract classes are inherited by the class descendants. Check the IsAbstract check box on the Class tab on the Class Properties dialog box to mark a class as abstract.

See Also

Tagged values allow you to fully specify UML elements in the model. See 'Setting tagged values' in this document.

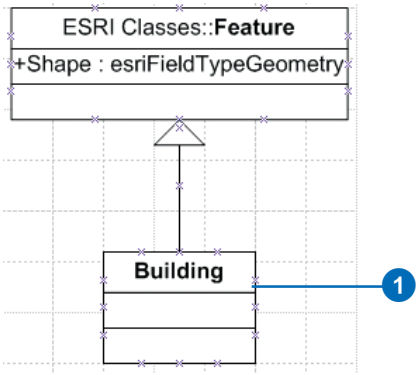
4. Type the name of the feature class.
5. To set tagged values, follow steps 2 through 6 of 'Setting tagged values' in this chapter; otherwise, skip to step 6.
6. Click OK to accept the changes.
7. Click Generalization in the UML Static Structure stencil and drag and drop it onto the diagram.
8. Drag the ends of the generalization arrow and connect the new class with its parent.

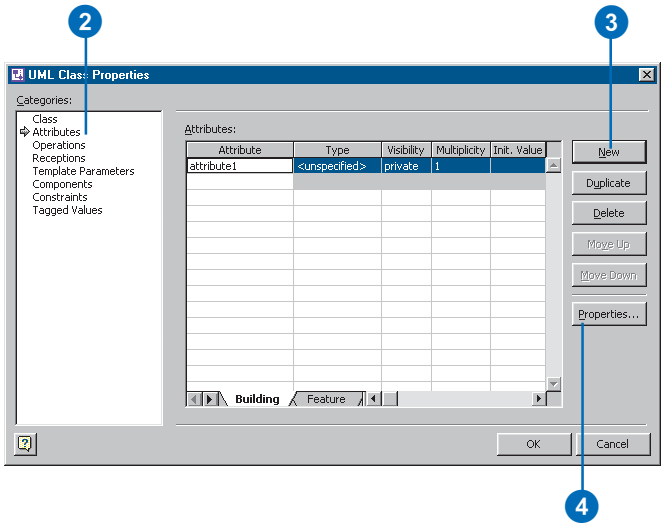




Adding fields to a feature class

1. Double-click the class in the diagram.
2. Click Attributes in the Categories window and double-click a string-typed attribute.
3. Click New to add a new attribute.
4. Click Properties to edit the field property. ▸





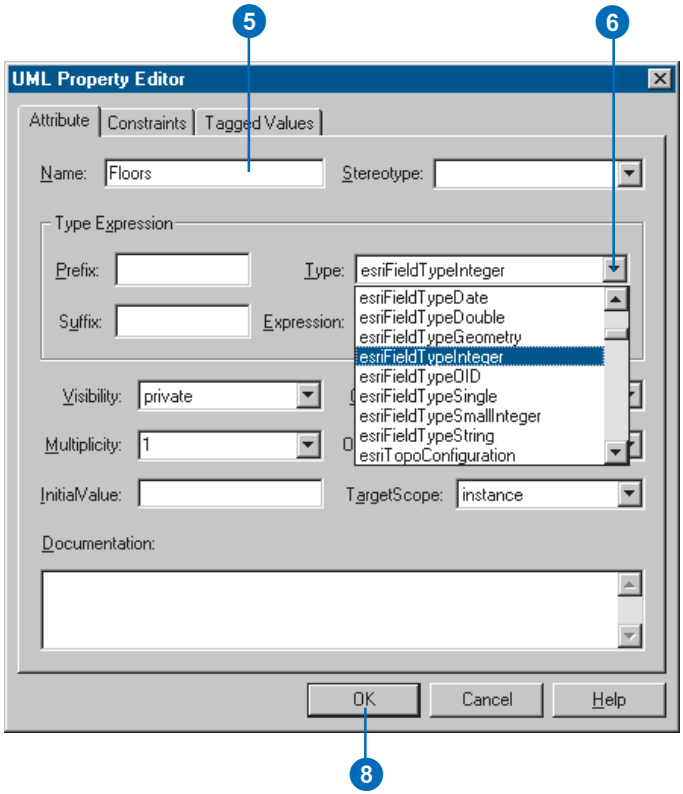
Tip

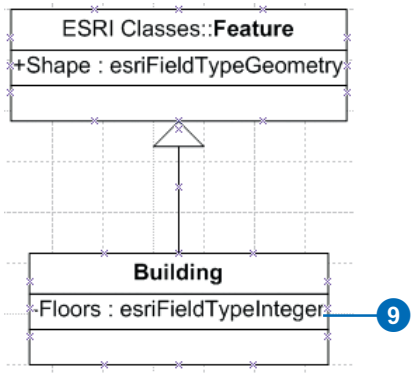
Default values

You can set the default for a field by entering the value in the Initial Value text box on the Attribute tab of the UML Property Editor dialog box. Default values are optional.

5. Type a name for the new field.
6. Click the Type dropdown arrow and click the field type.
7. Follow steps 2 through 6 of 'Setting tagged values' in this chapter to set tagged values; otherwise, skip to step 8.
8. Click OK.
9. Repeat steps 3 through 8 until you have added all the fields for your new class.

   The fields appear in the diagram as attributes of the class.





Creating relationship classes

UML associations represent relationship classes among feature classes and object classes (tables).

Relationship classes can be created only among leaf classes. Primary and foreign key fields must be present in the classes.

Tagged values for relationship classes include:

- OriginClass: the name of the origin class

  Example: PowerPole

- OriginPrimaryKey: the name of the primary key field in the origin class

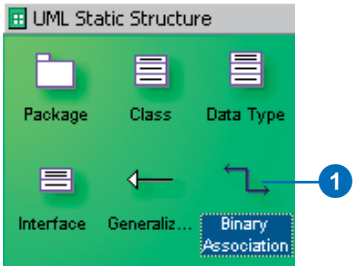
  Example: OriginPrimaryKey=OBJECTID

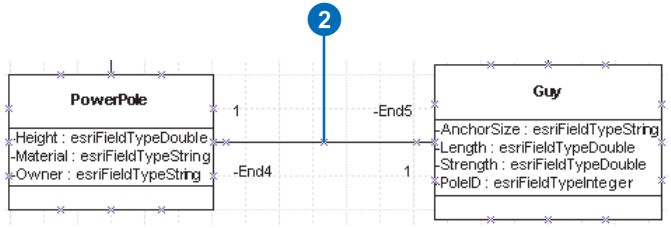
- OriginForeignKey: the name of the foreign key field in the destination class

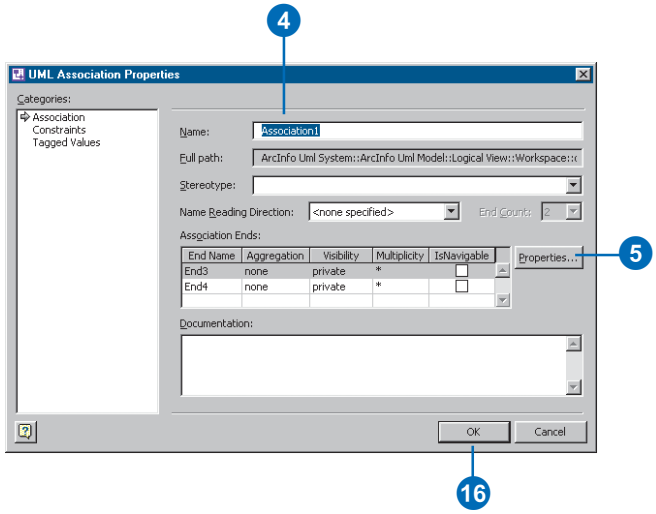
  Example: OriginForeignKey=PoleID ▸

Creating nonattributed relationship classes

1. In the UML Static Structure stencil, click Binary Association and drag and drop it onto the diagram.
2. Connect the two classes. The left end of the association is the origin class, and the right end is the destination class.
3. Double-click the association.
4. Type a name for the association.
5. Click one of the association ends and click Properties. ▸







- Notification: one of the values of the esriRelNotification enumeration

  Example: Notification=esriRelNotificationBoth

The cardinality of the relationship class is derived from the multiplicity of both association ends. Because the cardinality of a relationship class can be only 1–1, 1–M, or M–N, the only valid multiplicity values are 1 and * (many). The name of an association end becomes a relationship class path label.

A relationship class can have its own set of attributes. You can model such relationship classes by adding a UML class named after the relationship and stereotyped as RelationshipClass. The fields of the relationship class are modeled in the same way as fields in any other class. Foreign keys must be included in the class representing the attributed relationship. Many-to-many relationships are always attributed.

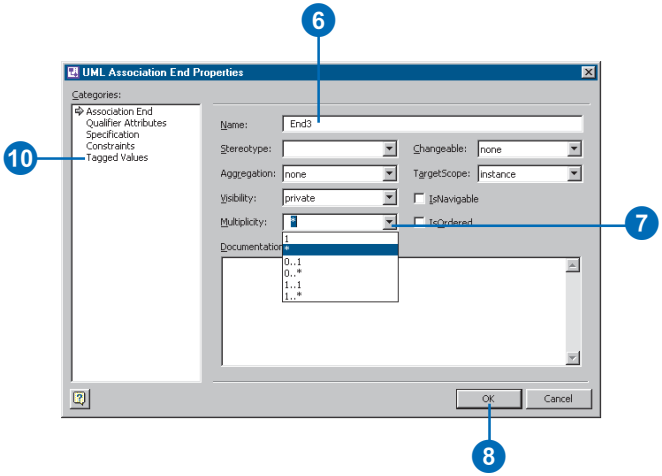
The following tagged values are recognized for attributed relationship classes:

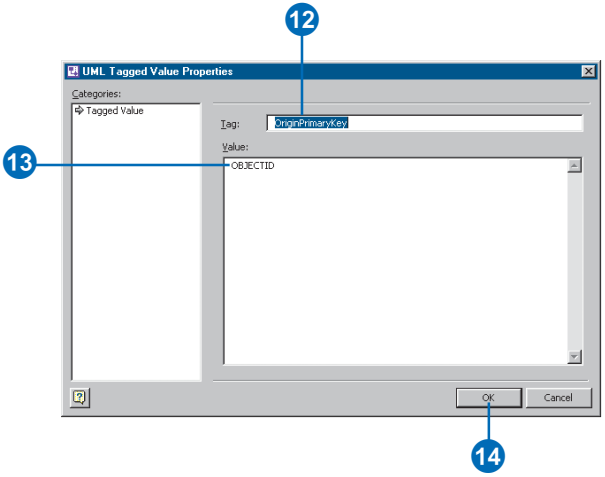
- IsAttributed: should be True ▸

6. Type the association end name.
7. Click the Multiplicity dropdown arrow and click the association end multiplicity.
8. Click OK.
9. Repeat steps 5 through 8 to set the name and multiplicity of the second end.

   The keys for the relationship class are set in UML using tagged values.

10. Click Tagged Values in the Categories window.
11. Click New.
12. Type “OriginPrimaryKey” for the tag name.
13. Type the name of the origin primary key field.
14. Click OK.
15. Repeat steps 11 through 14 to set the origin foreign key and notification tags. ▸





- OriginPrimaryKey: the name of the primary key field in the origin class

  Example: OriginPrimaryKey=OBJECTID

- OriginForeignKey: the name of the foreign key field in the destination class

  Example: OriginForeignKey=OwnerID

- DestinationPrimaryKey: the name of the primary key field in the destination class

  Example: DestinationPrimaryKey=OBJECTID

- DestinationForeignKey: the name of the foreign key field in the attributed relationship

  Example: DestinationForeignKey=FittingID

- Notification: one of the values of the esriRelNotification enumeration

  Example: Notification=esriRelNotificationBoth

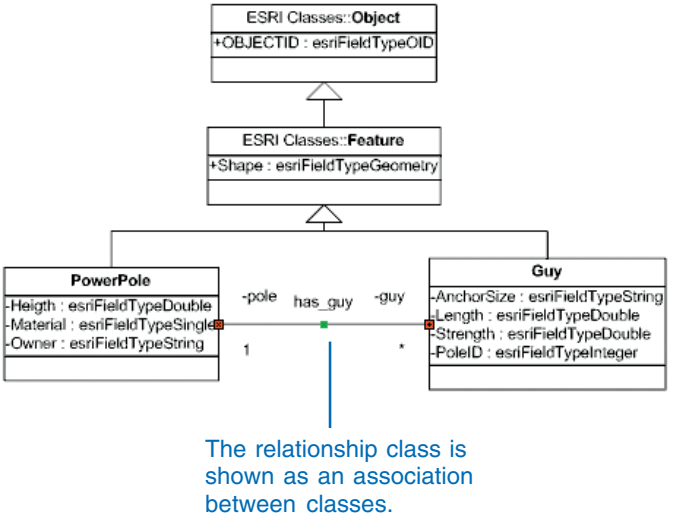
Tip

Using OBJECTID as a key field

OBJECTID is defined in the Object class and is inherited by all other classes. If a field typed as OID is used as the primary key, then the foreign key must be typed as esriFieldTypeInteger.

16. Click OK.

    The relationship class is shown as an association between two classes. The name of the association is the name of the relationship class.



The relationship class is shown as an association between classes.

Tip

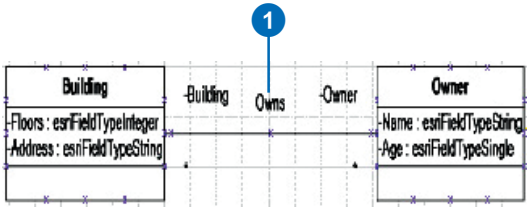
Composite relationships

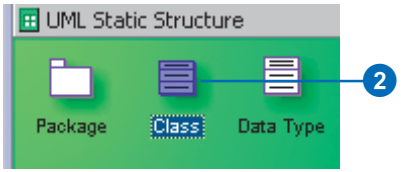
In a composite relationship, one of the objects controls the lifetime of the associated objects. You can create a composite relationship by using a Composition instead of a binary association.

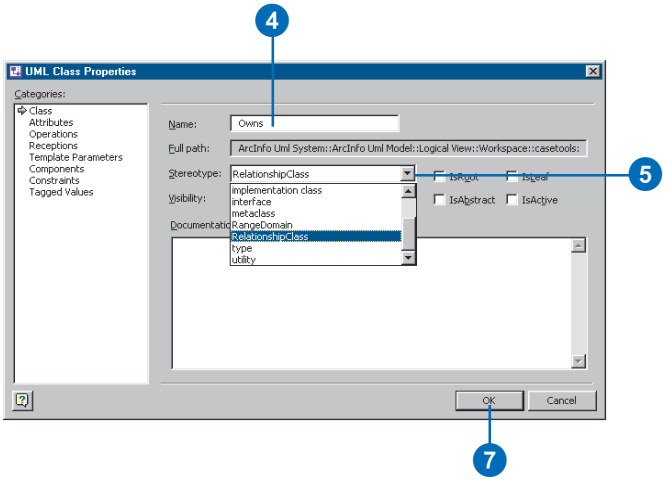
Creating attributed relationship classes

1. Create the UML association representing the relationship class by following steps 1 through 16 of 'Creating nonattributed relationship classes', described in the previous task.
2. In the UML Static Structure stencil, click Class and drag and drop a new UML class onto the diagram.
3. Double-click the new class.
4. Type the name of the association as the name of the new class.
5. Click the Stereotype dropdown arrow and click RelationshipClass.
6. Follow steps 1 through 8 of 'Adding fields to a feature class' in this chapter to add fields to the relationship class.
7. Click OK.

   The keys for the relationship class are set in UML using tagged values. ▸







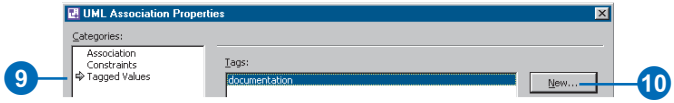
Tip

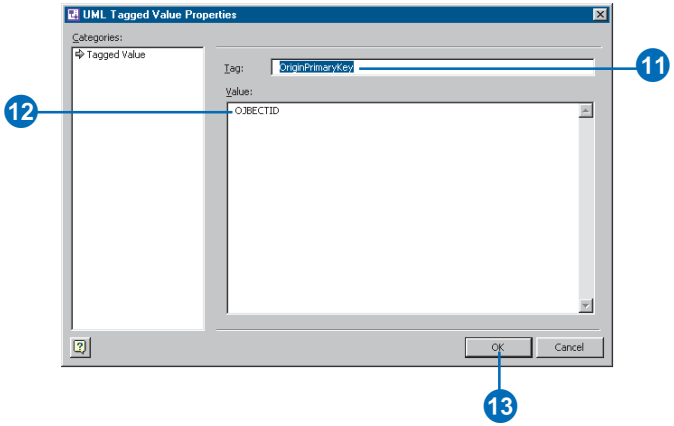
Association tagged values

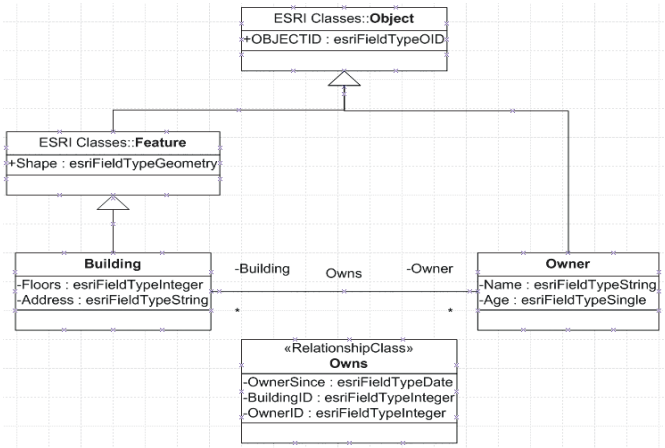
The tagged values should be in the association and not in the class with the attributes of the relationship.

8. Double-click the association in the diagram.
9. Click Tagged Values in the Categories window.
10. Click New.
11. Type “OriginPrimaryKey” for the tag name.
12. Type the name of the origin primary key field.
13. Click OK to close the UML Tagged Value Properties dialog box.
14. Repeat steps 10 through 13 for the destination primary key, the origin and destination foreign keys, and the notification tags.
15. Click OK to close the UML Association Properties dialog box.

    The attributed relationship is represented by a class with the same name as the relationship association. Its attributes appear as properties of the class.







Creating domains

Domains are modeled in UML as stereotyped classes. Domains can be used as the type of a field to define the type and valid values for the field.

The first three attributes in the class define the field type, the *merge policy*, and the *split policy*. The type for any of these attributes is not important and can be left unspecified. The initial value, however, is the actual setting. For example, the following is a valid MergePolicy:

- Attribute name: MergePolicy
- Attribute type: <unspecified>
- Attribute Initial Value: esriMPTDefaultValue

The valid initial values for FieldType, MergePolicy, and SplitPolicy are taken from the esriFieldType, esriMergePolicy, and esriSplitPolicy enumerations.

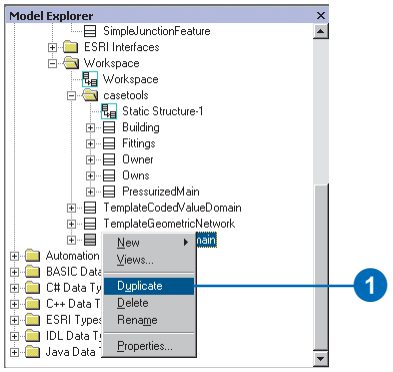
In addition to the standard attributes of domains (FieldType, MergePolicy, and SplitPolicy), range domains have MinValue and MaxValue. ►

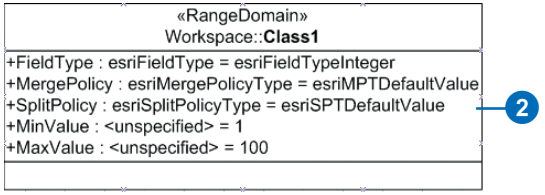
Creating a range domain

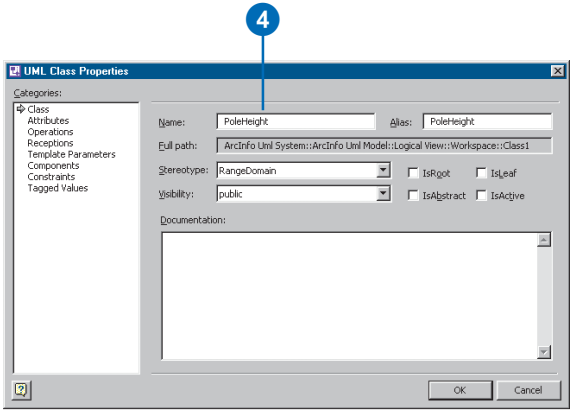
1. In the Model Explorer under the Workspace package, right-click TemplateRangeDomain and click Duplicate.

   A copy of the TemplateRangeDomain is created under the Workspace package.

2. Drag and drop it onto the diagram.
3. Double-click the new class.
4. Type the name of the domain. ►



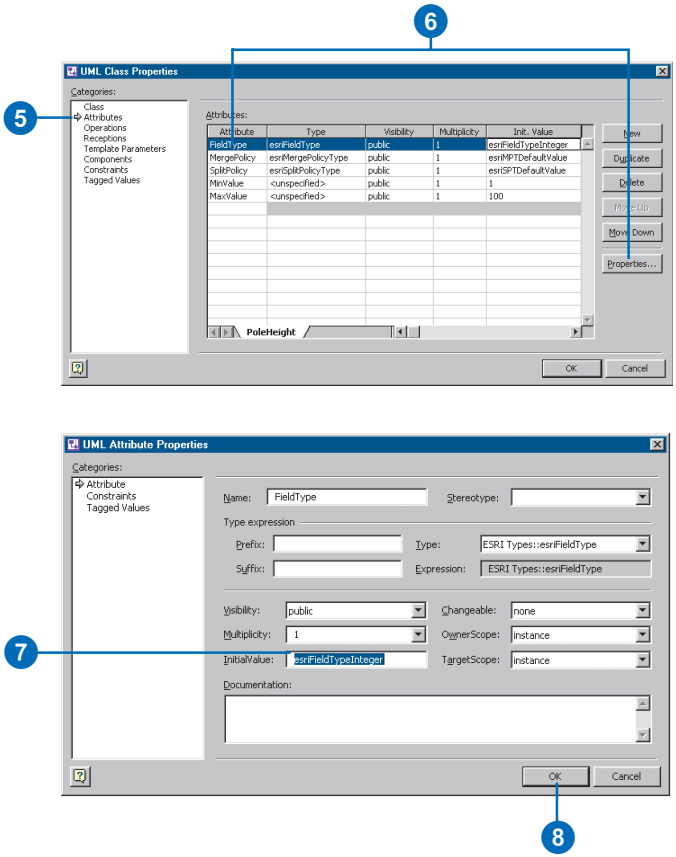




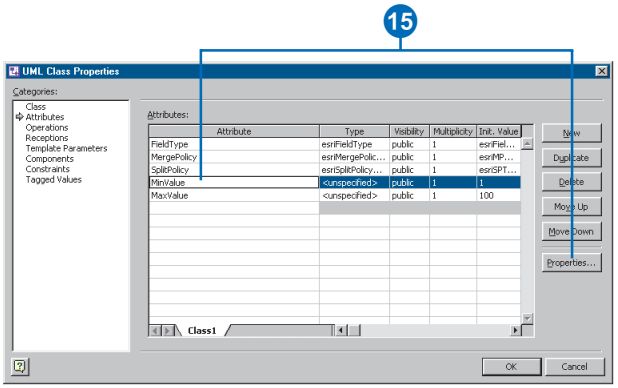
The initial values in MinValue and MaxValue define the actual range. The type of these attributes is not important and can be left unspecified.

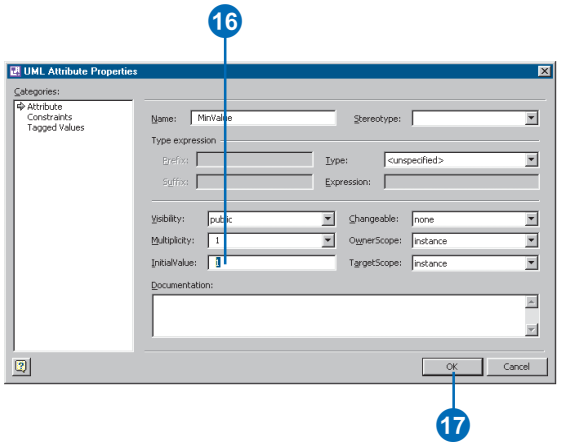
Coded value domains can have any number of UML attributes representing the set of permissible values. The initial value defines the valid code, and the name of the attribute is the name of the code. The type of these attributes is not important and can be left unspecified.

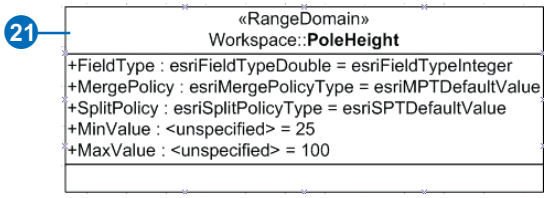
5. Click Attributes in the Categories window.
6. Click FieldType and click Properties.
7. Input the type of field with which this domain will be associated in the InitialValue text box.
8. Click OK.
9. Click MergePolicy and click Properties.
10. Type the merge policy in the InitialValue text box.
11. Click OK.
12. Click SplitPolicy and click Properties.
13. Type the split policy in the InitialValue text box.
14. Click OK. ▸



15. Click the MinValue attribute and click Properties.
16. Type the minimum value for the range domain in the InitialValue text box.
17. Click OK.
18. Click the MaxValue attribute and click Properties.
19. Type the maximum value for the range domain in the InitialValue text box.
20. Click OK.
21. Right-click the domain in the diagram and click Shape Display Options. Click the Attribute check box to hide the attribute types.

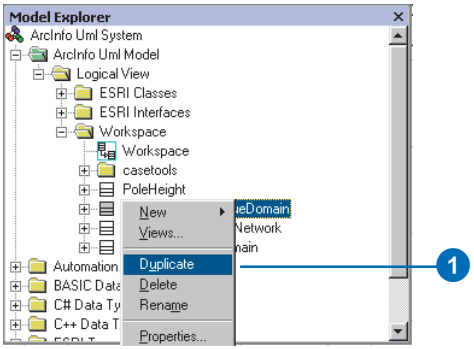


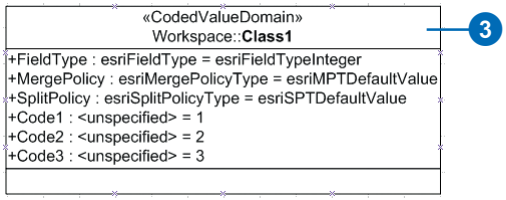


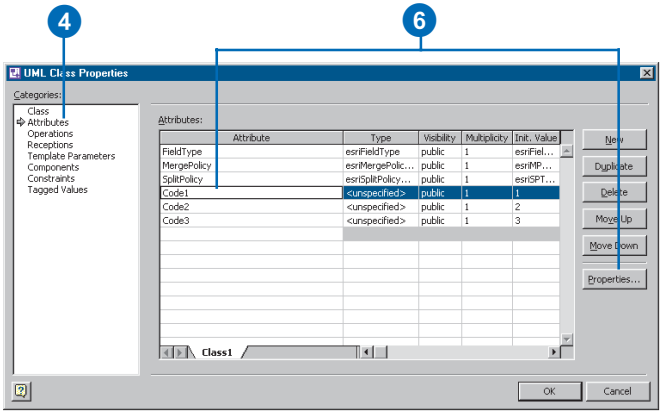


Creating coded value domains

1. In the Model Explorer under the Workspace package, right-click TemplateCodedValueDomain and click Duplicate.
2. Drag and drop onto the diagram the copy of the TemplateCodedValueDomain that is created under the Workspace package.
3. Double-click the domain in the diagram and type the name of the domain.
4. Click Attributes in the Categories window.
5. Follow steps 6 through 14 of 'Creating a range domain' in this chapter to set the field type and the split and merge policies.
6. Click the Code1 attribute and click Properties. ▸





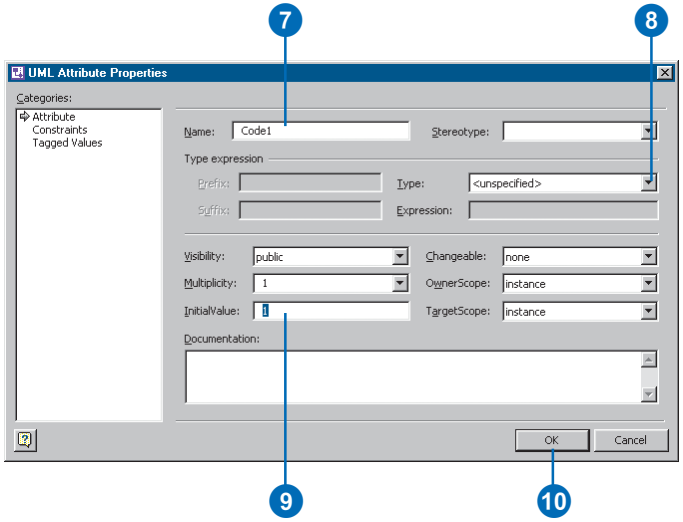


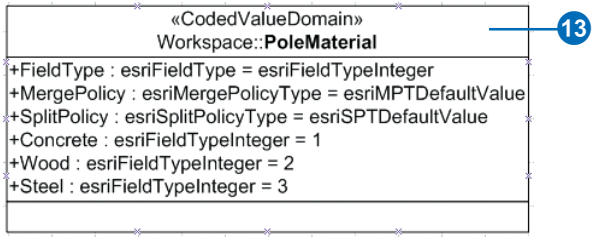
Tip

Adding additional codes

In the UML Class Properties Editor, you can add additional codes by clicking Attributes in the Categories window. Click New and click Properties for the attribute and set the name and initial value for the code.

7. Type the code name.
8. Click the Type dropdown arrow and select the type of field.
9. Type the code in the InitialValue text box.
10. Click OK.
11. Repeat steps 6 through 10 for the other codes.
12. Click OK.
13. Right-click the domain in the diagram and click Shape Display Options. Click the Attribute check box to hide the attribute types.





Creating subtypes

Subtypes are modeled in UML as classes related to the parent class through an association stereotyped as Subtype. You can specify domains and default values for the fields in the subtype.

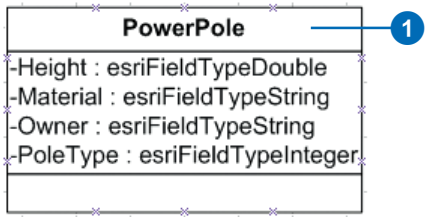
A field in the parent class must be stereotyped as SubtypeField. Its initial value is the subtype code of the default subtype. The subtype field must be typed as "esriFieldTypeInteger" in the parent class.

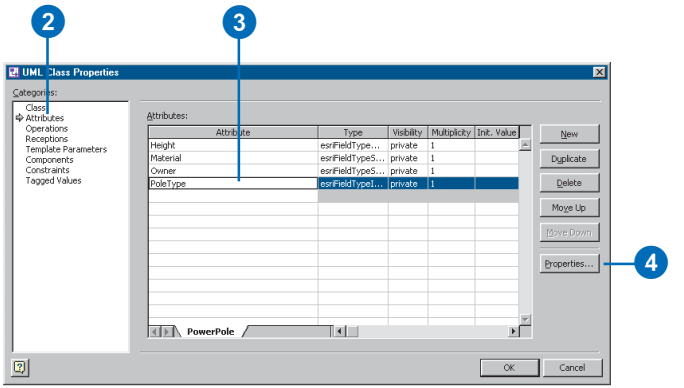
All subtypes must have a unique value for the subtype field—this unique value is its subtype code. Fields in the subtype must match those of the parent class in name and type, but not all the fields of the parent class have to be present in the subtype. The subtype field, however, is a required field in the subtype.

Initial values must be type-compatible with the type of the field. The type of a field can be a domain previously created.

Defining the subtype field for the feature class

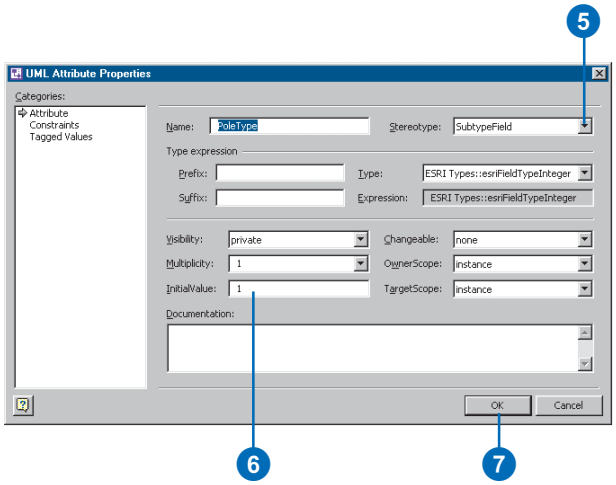
1. Double-click in the diagram the class for which you want to create a subtype.
2. Click Attributes in the Categories window.
3. Click the attribute that will be the subtype field.
4. Click Properties. ▸

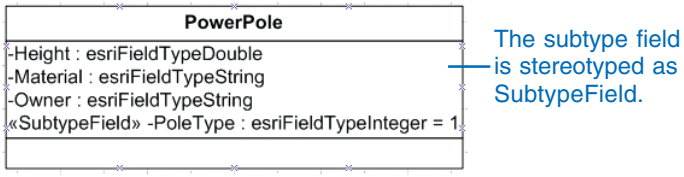




5. Click the Stereotype dropdown arrow and click SubtypeField.
6. Type the subtype code of the default subtype in the InitialValue text box.
7. Click OK.

   The subtype field appears as a property of the class with a stereotype of SubtypeField.





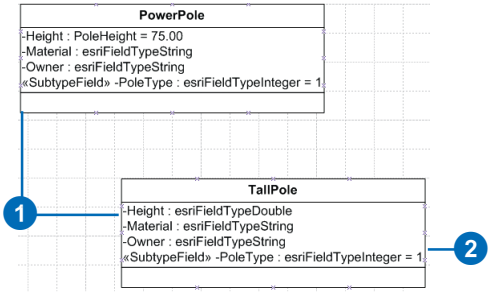
Tip

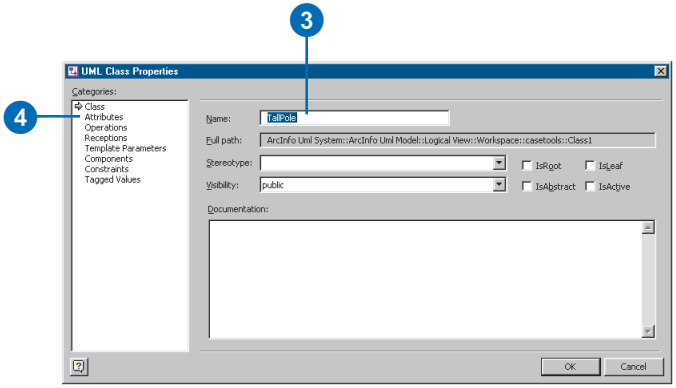
When to include a field in a subtype

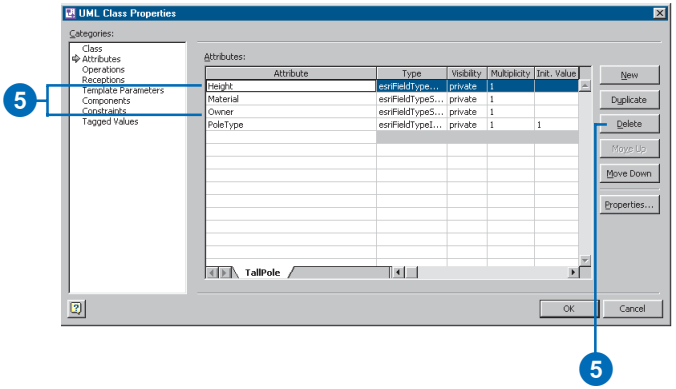
A field should be included in a subtype only when you want to associate a default value or domain with it. Fields inherited from abstract classes can be included in subtypes.

Creating a subtype

1. Copy the class in the diagram for which you want to create a subtype. Paste it onto the diagram.
2. Double-click the new class.
3. Type a name for the subtype.
4. Click Attributes in the Categories window.
5. Click each field for which you don't want to associate a default value or domain for this subtype, then click Delete to remove each one. ▸



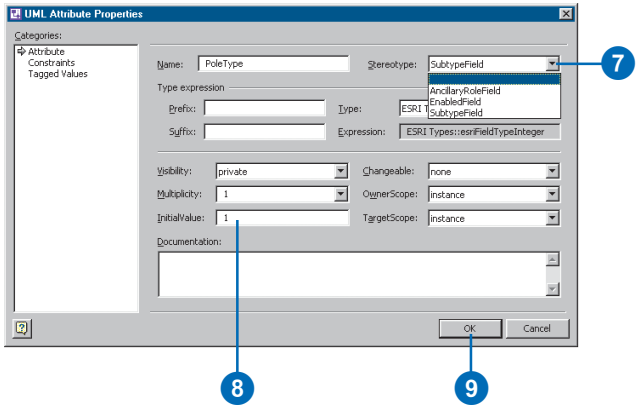


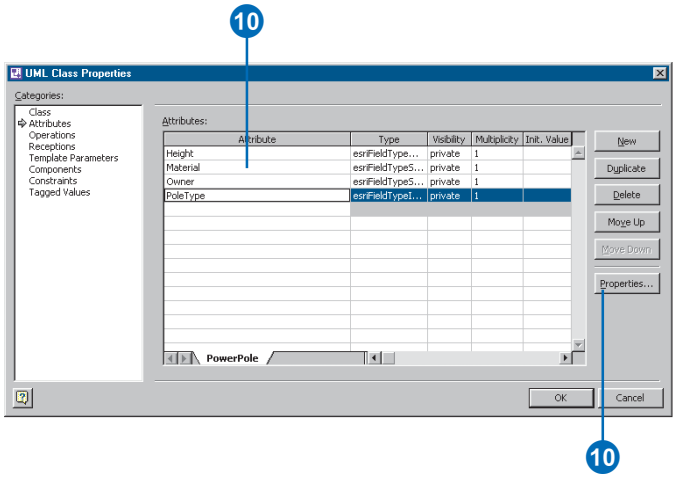


6. Click the subtype field and click Properties.
7. Click the Stereotype dropdown arrow and click the blank stereotype to set its stereotype to nothing.
8. Type the code for this subtype in the InitialValue text box.
9. Click OK.

   Repeat steps 6 through 9 until you have associated default values and domains with all the attributes.

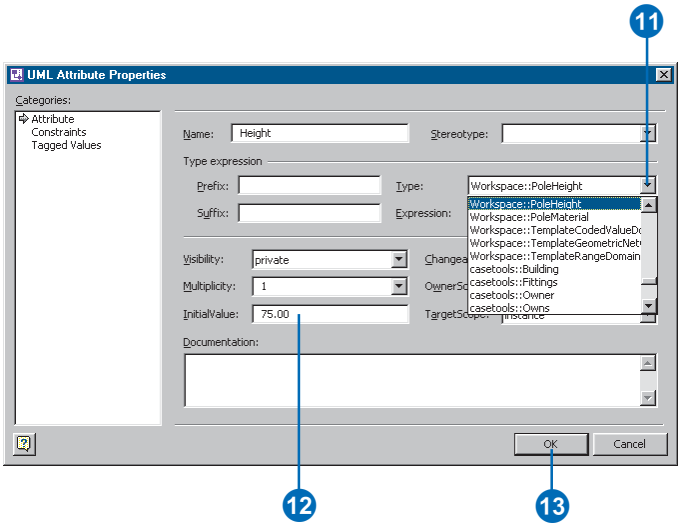
10. Click the attribute and click Properties to set default values and domains for a field. ▸

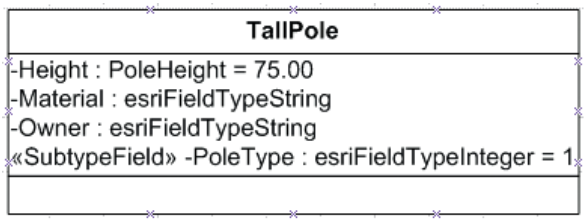




11. Click the Type dropdown arrow and click the domain you want to associate with the attribute as the type.
12. Type the default value in the InitialValue text box to associate a default value with this attribute.
13. Click OK to close the UML Attribute Properties dialog box.
14. Click OK to close the UML Class Properties dialog box.

    The field appears in the class with its domain as its type and its default value as its initial value.





The domain is displayed as the field type, and the default value as the initial value.

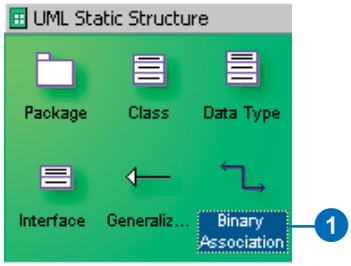
Tip

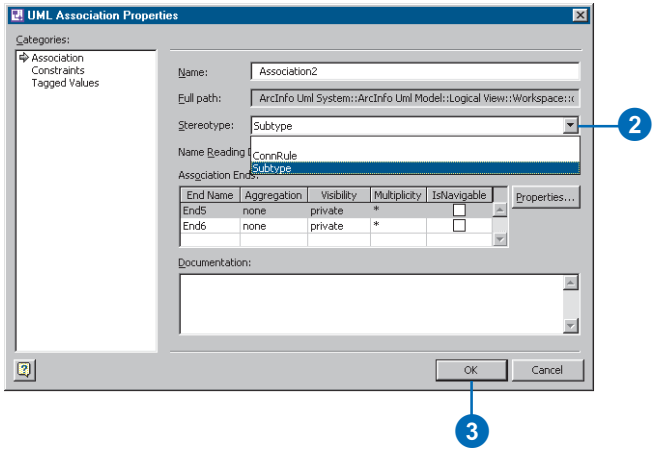
Setting the initial value for the subtype field

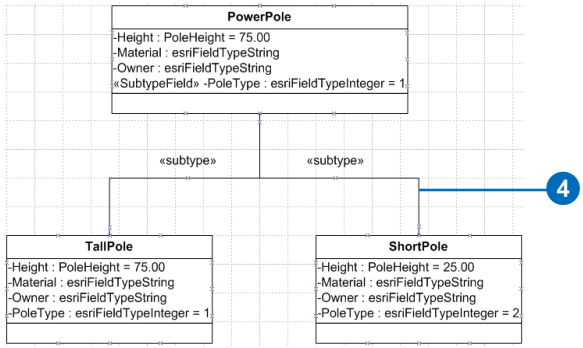
Be careful when setting the initial value for the subtype field in your subtypes. Its value has to be unique across all the subtypes of a class.

Associating the subtype with its parent class

1. In the UML Static Structure stencil, click Binary Association and drag and drop it onto the diagram. Connect the parent and subtype classes.
2. Double-click the association in the diagram. Click the Stereotype dropdown arrow and click Subtype.
3. Click OK.
4. Right-click the association and click the Shape Display Options to show its name and stereotype and to hide the end names and cardinalities.







Creating relationship rules

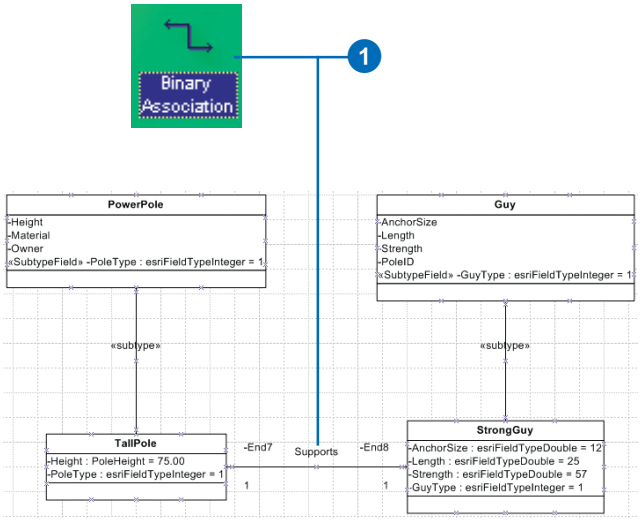
Relationship classes can have a set of relationship rules. These rules control which subtypes of objects from the origin class can be related to which subtypes of the destination class.

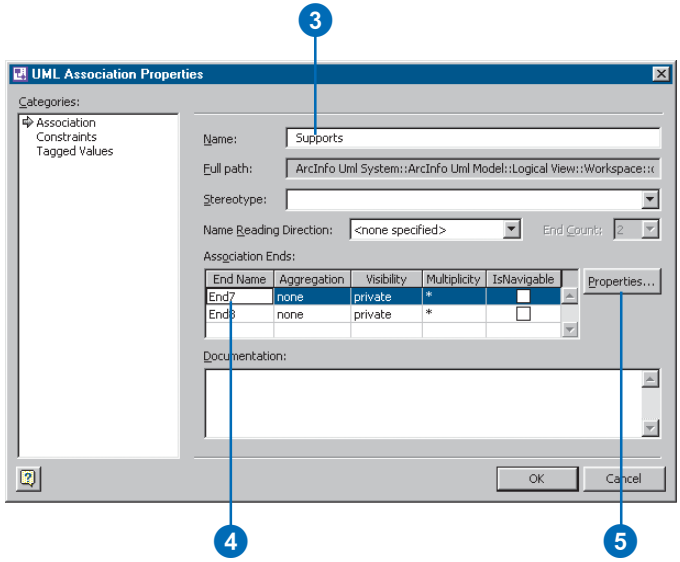
Relationship rules are represented by a UML association between subtypes. Multiplicity in association ends is used to set the cardinality of the relationship rule.

A value of 1..3 means at least one and no more than three objects of this subtype can be related.

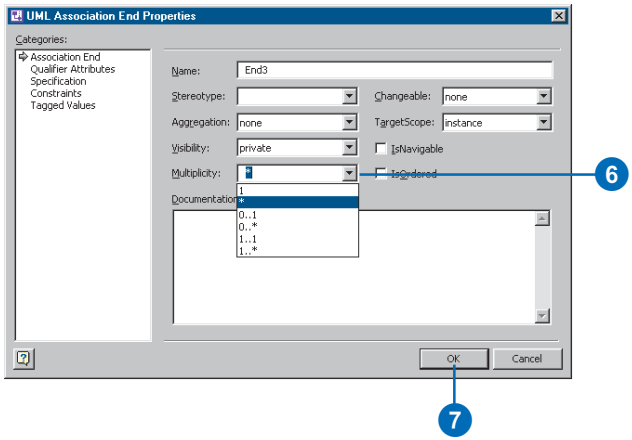
The cardinality range of both ends must be compatible with the cardinality of the parent relationship class (in the example, 1–1..3 is compatible with 1–M).

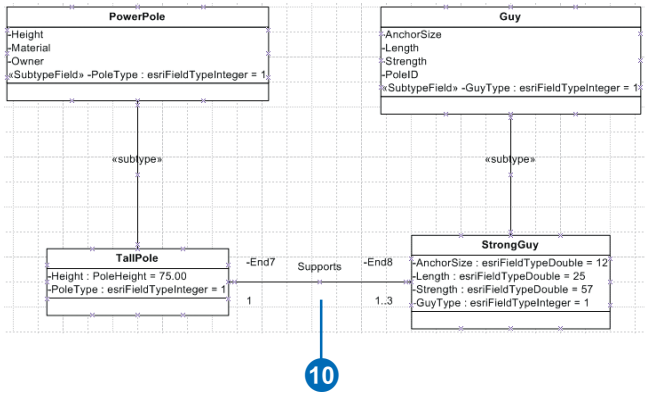
1. In the UML Static Structure stencil, click Binary Association and drag and drop it onto the diagram. Connect the two subtypes between which you want to create a relationship rule.
2. Double-click the association.
3. Type the name of the relationship between the parent classes as the name for this association.
4. Click one of the association ends.
5. Click Properties. ▸





6. Select the valid cardinality as the multiplicity of the association end.
7. Click OK to close the UML Association End Properties dialog box.
8. Repeat steps 4 through 7 for the other association end.
9. Click OK to close the UML Association Properties dialog box.
10. Right-click the association and click the Shape Display Options to show its name and stereotype and to hide the end names and cardinalities.





Creating geometric networks

A geometric network is modeled with a UML class stereotyped as GeometricNetwork. UML binary associations are used to associate the network feature classes in the model with the geometric network.

The geometric network UML class and all associated feature classes must be created under the same feature dataset UML package.

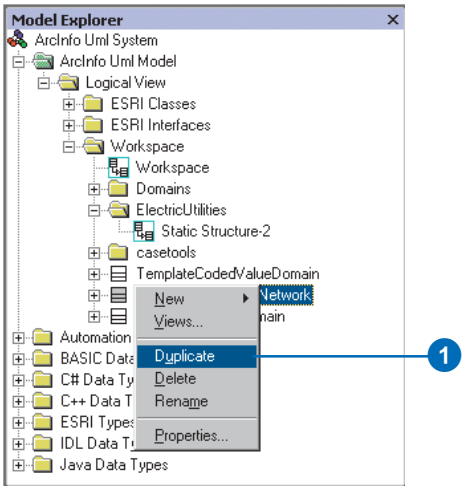
The only attribute in a geometric network class defines the network type. When you copy the template, the attribute is already in the class.

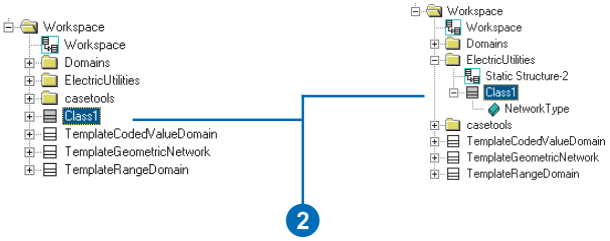
Tip

Feature datasets

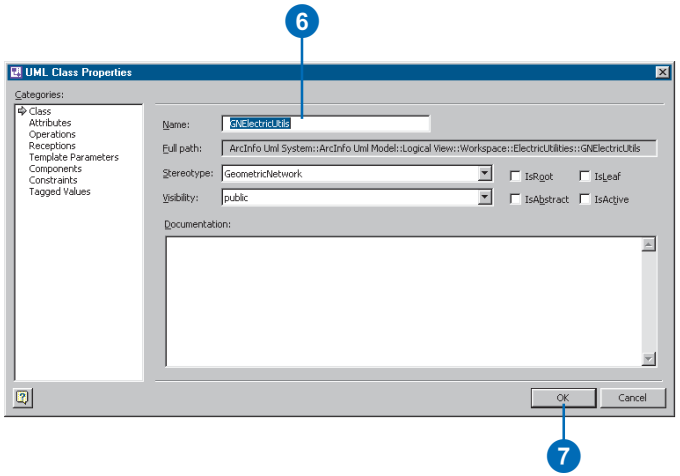
Make sure you have created a feature dataset before creating a geometric network. A geometric network can exist only inside a feature dataset.

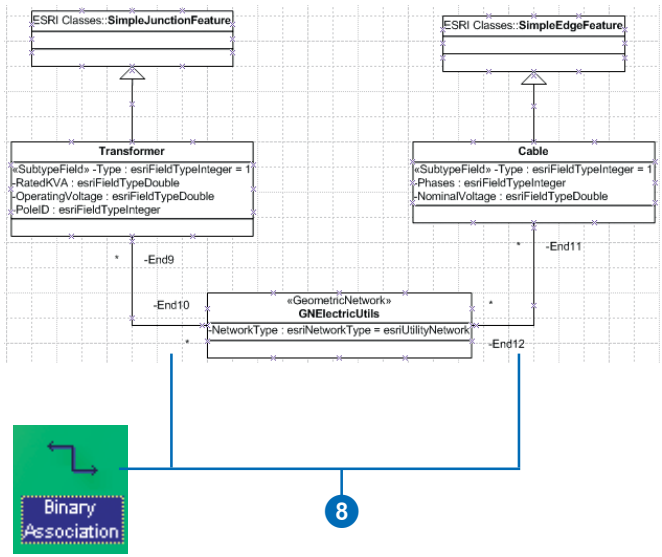
1. Right-click TemplateGeometricNetwork under the Workspace package in the Model Explorer and click Duplicate.
2. Drag and drop the copy of TemplateGeometricNetwork, which is created in the Model Explorer under the Workspace package, inside the package representing the feature dataset. ▸





3. Double-click the feature dataset diagram to open it.
4. Drag and drop the new geometric network onto the diagram.
5. Double-click the geometric network to open its properties.
6. Type a name for the geometric network.
7. Click OK.
8. In the Model Explorer, click Binary Association and drag and drop it onto the diagram. Connect the network feature classes to the geometric network.





Creating connectivity rules

Connectivity rules can be defined among subtypes of network feature classes modeled in a geometric network. UML associations are used to define connectivity rules. Two types of connectivity rules can be created: edge–edge rules and edge–junction rules.

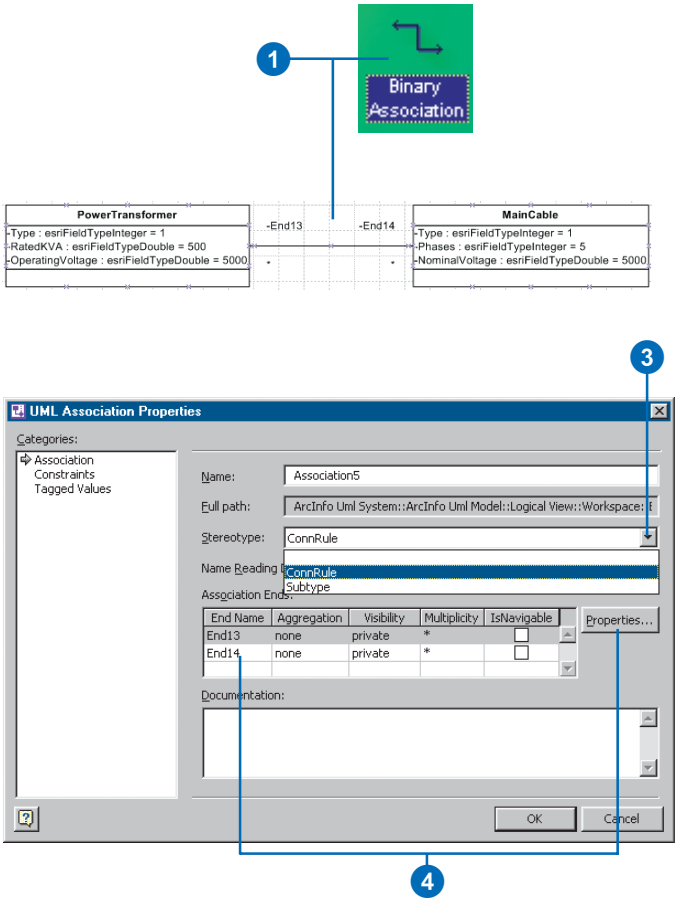
Edge–junction connectivity rules can have specific cardinalities for each subtype involved. In this example, one power transformer can be connected to up to two main cables. One of the junction subtypes can be marked as the default junction by stereotyping the association end as Default.

Edge–edge rules are represented by a UML N-ary association that involves two edges and any number of junctions. One of the junction subtypes must be marked as the default junction by stereotyping the association end as Default.

All geometric networks have a default or generic junction subtype, also called the orphan junction type. You can create connectivity rules that include the generic junction. ▸

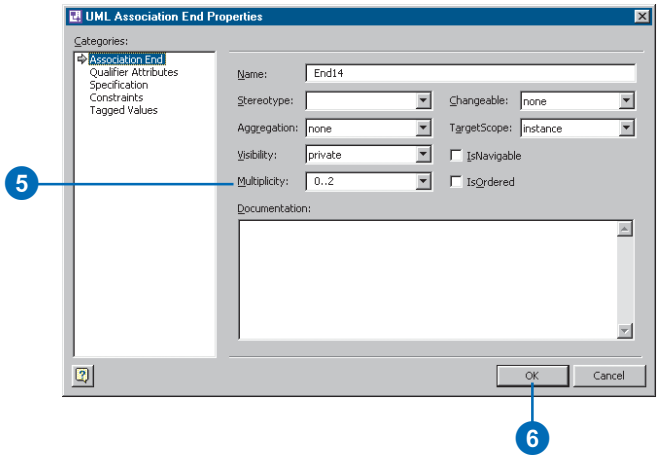
Creating an edge–junction rule

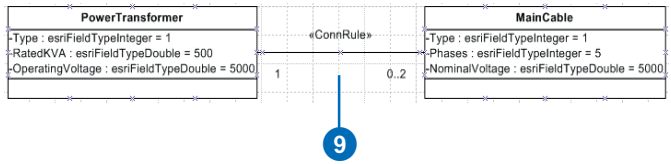
1. In the UML Static Structure stencil, click Binary Association and drag and drop it onto the diagram. Connect the edge subtypes and junction subtype.
2. Double-click the association.
3. Click the Stereotype dropdown arrow and click ConnRule.
4. Click one of the association ends and click Properties. ▸



Both edge–junction and edge–edge rules can involve the generic junction.

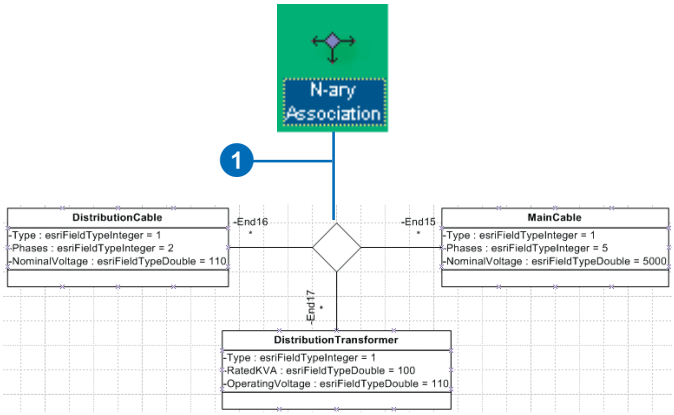
5. Type the cardinality for the subtype.
6. Click OK to close the UML Association End Properties dialog box.
7. Repeat steps 4 through 6 to set the cardinality for the second subtype.
8. Click OK to close the UML Association Properties dialog box.
9. Right-click the association and click the Shape Display Options to show its name and stereotype and to hide the end names and cardinalities.

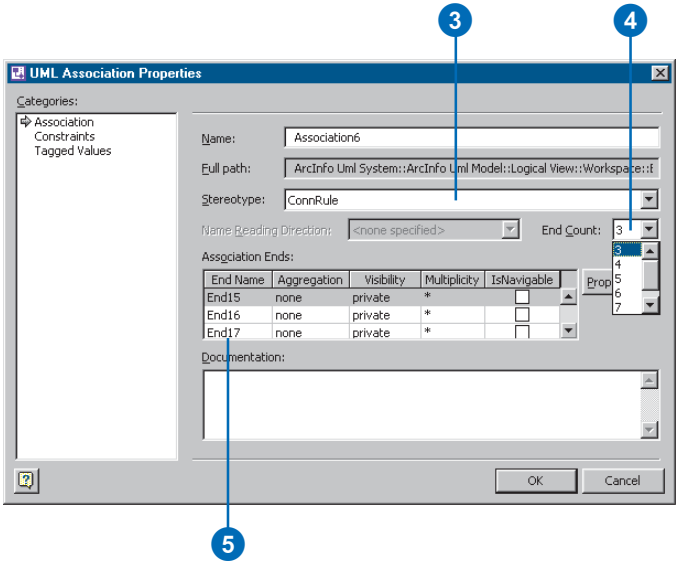




Creating an edge–edge rule

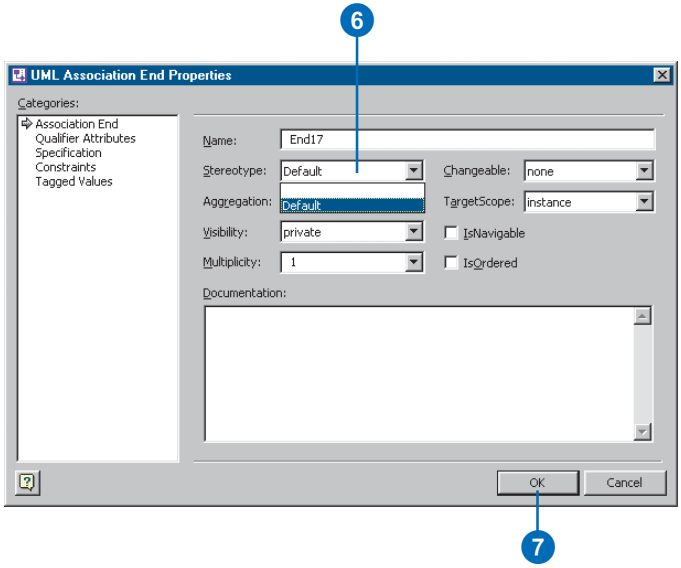
1. In the UML Static Structure stencil, click N-ary Association and drag and drop it onto the diagram. Connect two edge subtypes and a junction subtype.
2. Double-click the association.
3. Click the Stereotype dropdown arrow and click ConnRule.
4. Click the End Count dropdown arrow if there is more than one junction subtype for this edge–edge rule and click the number of junction subtypes.
5. Click one of the association ends and click Properties. ▸

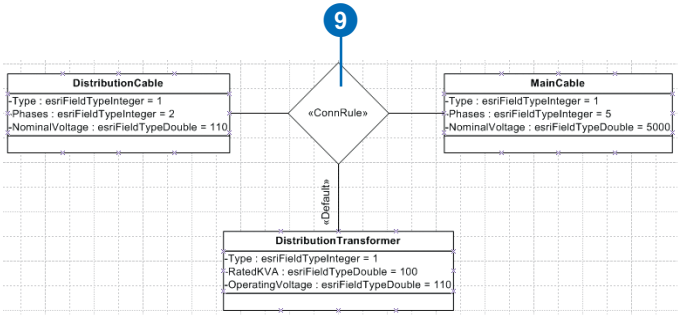




6. Click the Stereotype dropdown arrow and click Default to mark the junction subtype as the default junction subtype in the edge–edge rule.
7. Click OK to close the UML Association End Properties dialog box.
8. Click OK again to close the UML Association Properties dialog box.
9. Right-click the association and click the Shape Display Options to show its name and stereotype and to hide the end names and cardinalities.

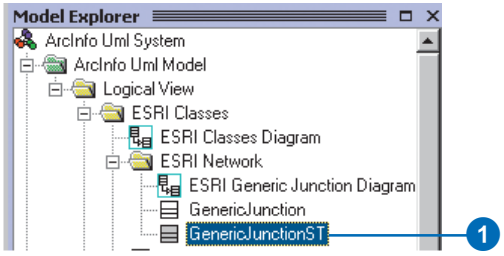
   The rule in this example can be read as “Distribution cables can be connected to main cables through distribution transformers”.

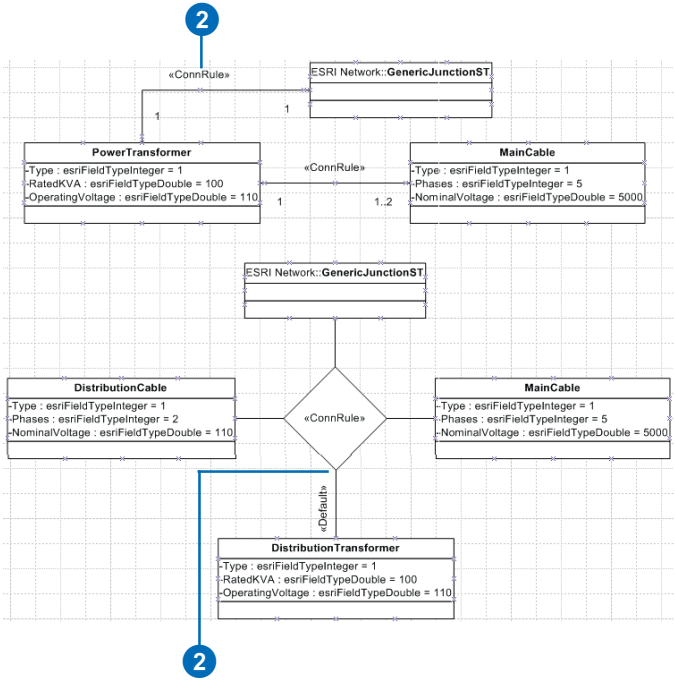




Using the generic junction subtype

1. In the Model Explorer, navigate to and click GenericJunctionST. Drag and drop it onto the diagram.
2. Follow steps 1 through 9 of ‘Creating an edge–junction rule’ or steps 1 through 9 of ‘Creating an edge–edge rule’, as described previously.





Extending classes with custom behavior

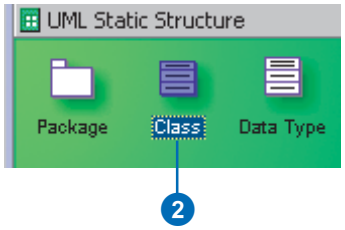
In addition to designing schema, you can use UML and CASE tools to generate code to create custom feature behavior. There are two modeling tasks associated with this: creating new interfaces and creating class extensions. It is important to understand that interfaces and class extensions in UML models are used only when code is generated for the model.

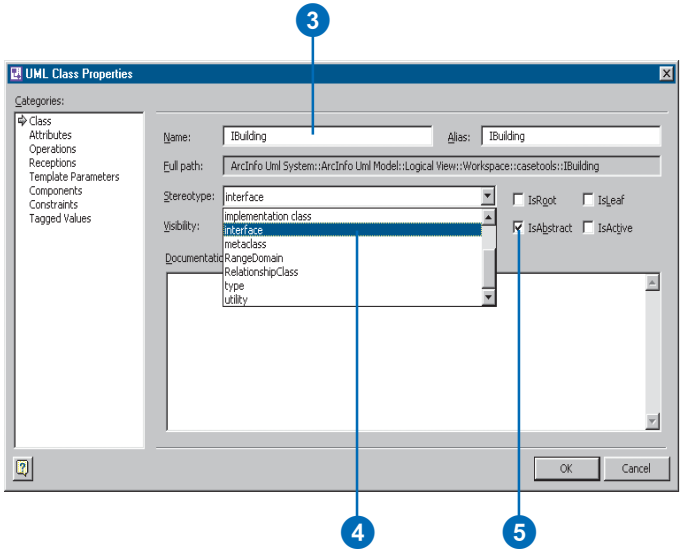
Interfaces are a set of related methods that a custom feature agrees to implement. A custom feature may implement any number of interfaces. Interfaces are inherited. An interface implemented by a parent class is also implemented by its children.

Types used in interfaces should be either other interfaces or automation-compatible types, such as long, double, DATE, and VARIANT_BOOL. UML attributes will become properties. They can be used as Object.Prop = value or var = Object.Prop. ▸

Creating an interface

1. In the UML Static Structure stencil, click Class and drag and drop it onto the diagram.
2. Double-click the class.
3. Type the name of the interface in the Name box.
4. Click the Stereotype dropdown arrow and click interface.
5. Check the IsAbstract check box. ▸





UML operations will become methods. They can be used as Object.Method(argument:type).

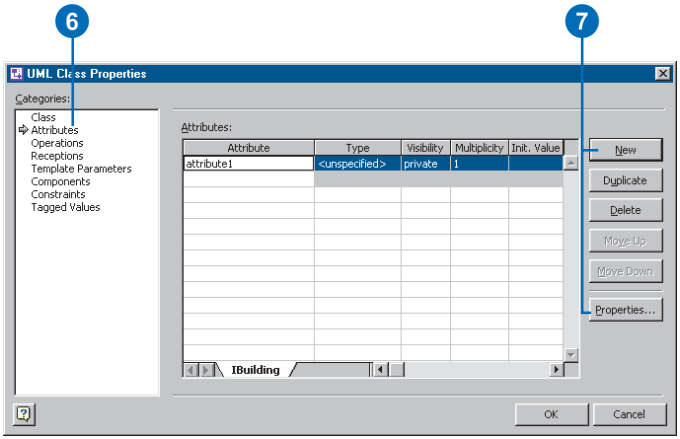
To create a read-only property, create an operation with the prefix get_. Likewise, use the prefixes put_ or putref_ to create write-only properties.

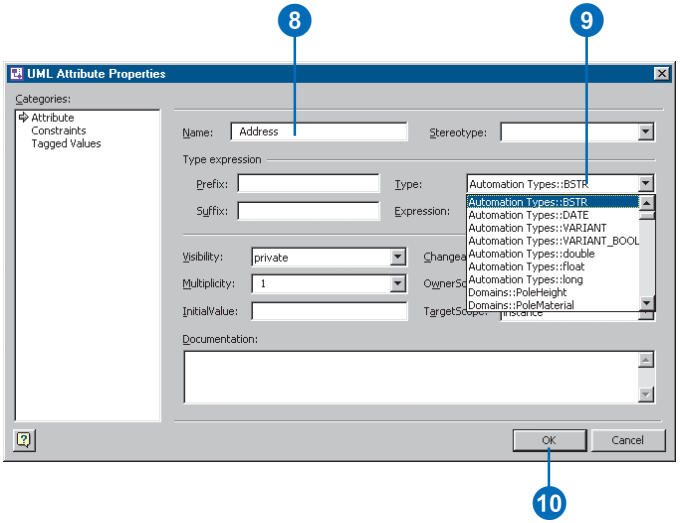
You can model class extensions when you want custom behavior associated with the feature class. Class extensions require a naming convention. The name must be formed by concatenating the name of the feature class or table with ClassExtension.

Class extensions for tables are derived from ObjectClassExtension. Likewise, class extensions for feature classes are derived from FeatureClassExtension.

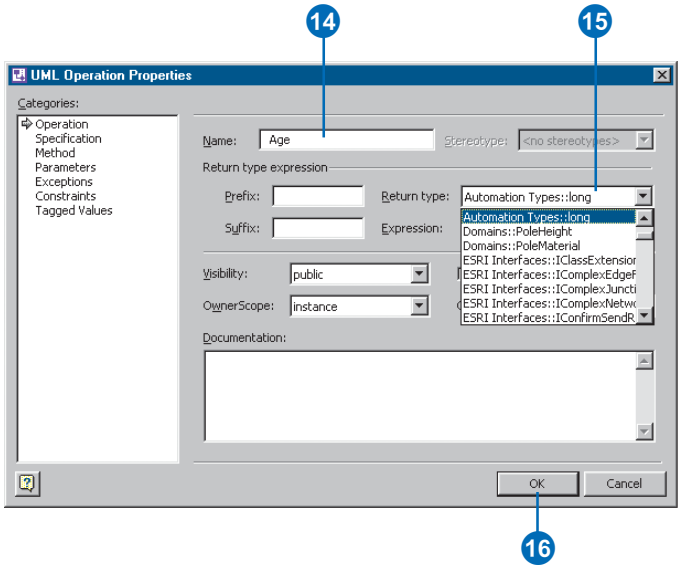
Class extensions may implement optional interfaces, such as IObjectInspector. In addition, they may implement their own interfaces (for example, IBuildingClassExtension).

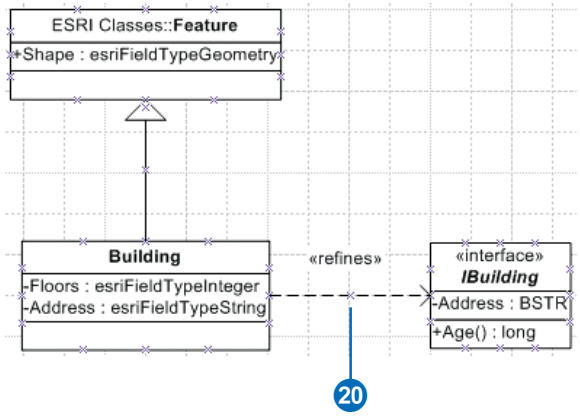
6. Click Attributes in the Categories window.
7. Click New to add a new attribute and click Properties.
8. Type a name for the new property.
9. Click the Type dropdown arrow and click the field type.
10. Click OK.
11. Repeat steps 7 through 10 until you have added all the properties of the new interface. ▸





12. Click Operation in the Categories window.
13. Click New to add a new method and click Properties.
14. Type the name of the method.
15. Click the Return type dropdown arrow and click the method type if it returns something.
16. Click OK.
17. Repeat steps 13 through 16 until you have added all the methods of the new interface.
18. Click OK.
19. In the UML Static Structure stencil, click Refinement and drag and drop it onto the diagram.
20. Connect the custom feature and the interface.





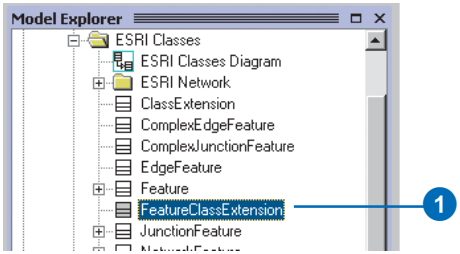
Tip

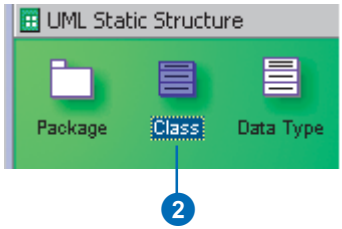
Class extension naming convention

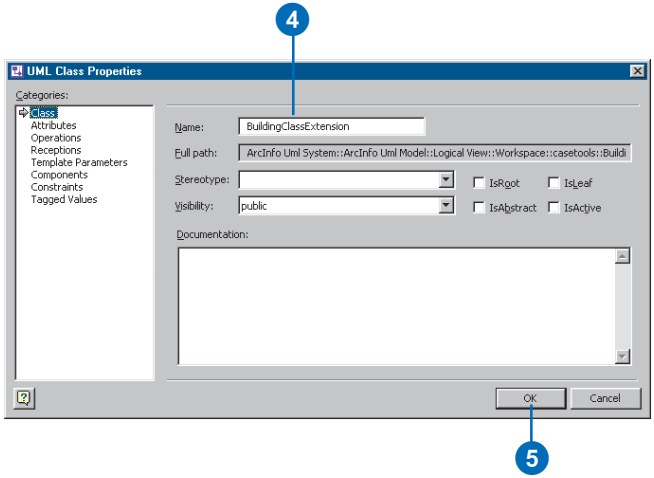
Class extensions must have the name <class>ClassExtension. For example, a class extension for a Parcels class would be ParcelsClassExtension.

Creating a class extension

1. In the Model Explorer tree under ESRI Classes, click FeatureClassExtension and drag and drop it onto the diagram.
2. In the UML Static Structure stencil, click Class and drag and drop a new UML class onto the diagram.
3. Double-click the new class in the diagram.
4. Type the name of the class following the class extension naming convention.
5. Click OK. ▸



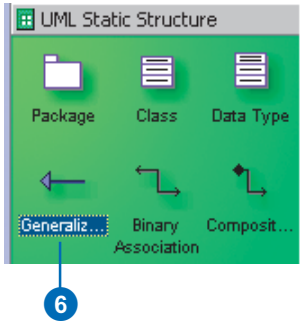


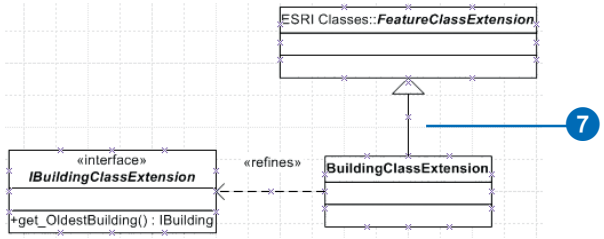


6. In the UML Static Structure stencil, click Generalization then drag and drop it onto the diagram.

7. Drag the ends of the generalization arrow and connect the new class with its parent.

   To add your own interfaces, follow the instructions for 'Creating an interface' in this chapter. There are a number of predefined optional interfaces you can implement in your class extension. ►



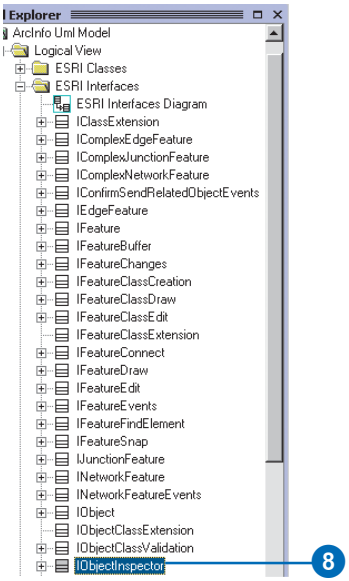


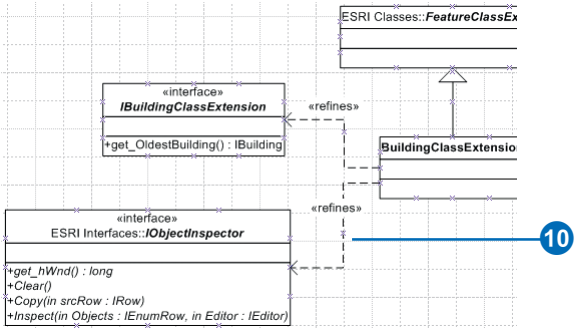
Tip

Optional interfaces

Your class extension can implement all or none of the optional class extension interfaces.

8. In the Model Explorer tree under ESRI Interfaces, click the optional interface to select it. Drag and drop it onto the diagram.
9. In the UML Static Structure stencil, click Refinement and drag and drop it onto the diagram.
10. Drag the ends of the refinement and connect the class extension with the interface.
11. Repeat steps 8 through 10 until you have added all of the optional interfaces you want to implement.





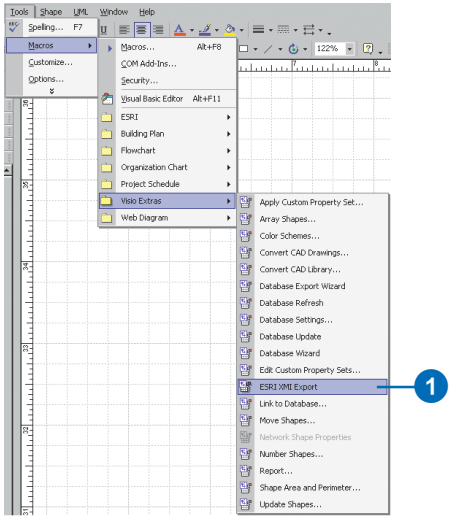
Exporting a model to XMI

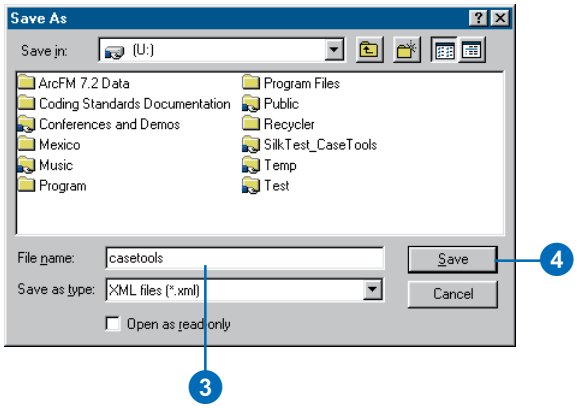
Before you can generate your geodatabase schema from your UML model, you must first export it to XMI.

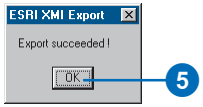
The tools to export your UML model to XMI are contained within Visio.

Exporting to XMI

1. Click Tools in Visio, point to Macros, point to Visio Extras, then click ESRI XMI Export.
2. Browse to navigate to a directory where you will save your XMI file.
3. Type the name of the XMI file.
4. Click Save to export the UML model.
5. Click OK.







Checking your model for errors

The semantics checker can be used to make sure your models are valid. If errors are found, a report will be automatically created. The report can be printed or exported to a number of formats.

The semantics checker works on models already exported to the Microsoft Repository or XMI.

You should run the semantics checker before using the Schema Wizard or Code Generation Wizard.

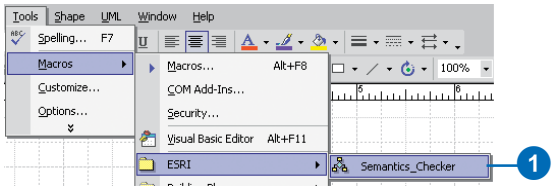
Tip

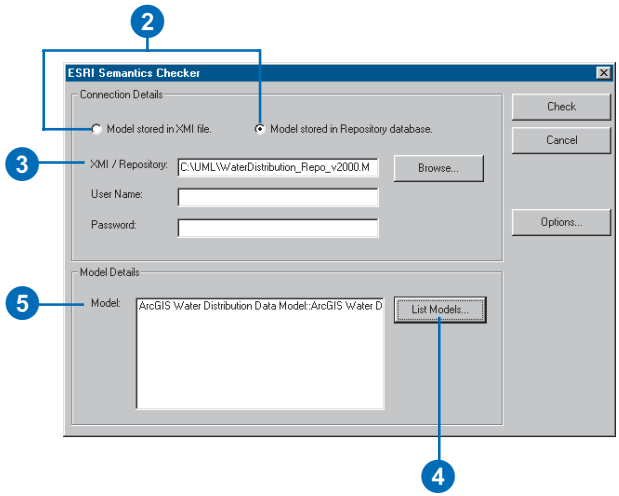
Options

Changing the options for the report allows you to modify the error report. By default the first two options will be checked.

Running the semantics checker

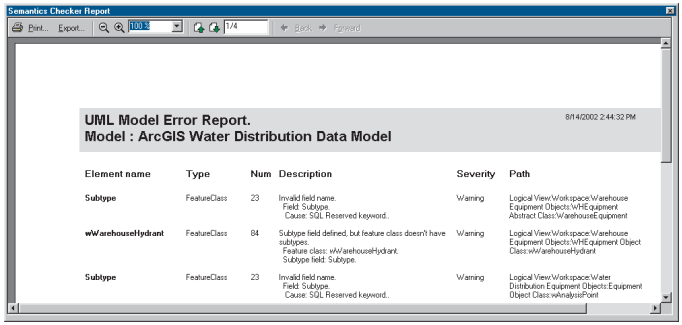
1. Click Tools in Visio, point to Macros, point to ESRI, then click Semantics_Checker.
2. Select the source of your model. Models can be stored in XMI files or repository databases.
3. Type the path to the XMI file or repository database, or click Browse to navigate to it.
4. Click List Models if you are using a repository.
5. Select the model you want to verify. ▸





6. Click Check.

   If errors are found, a report is generated listing all of the modeling errors.



A report is generated listing all of the modeling errors.