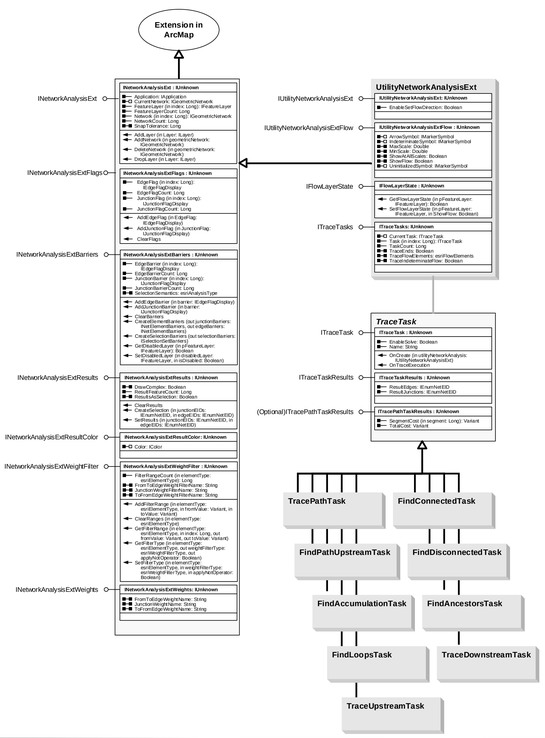
EditorExt Object Model

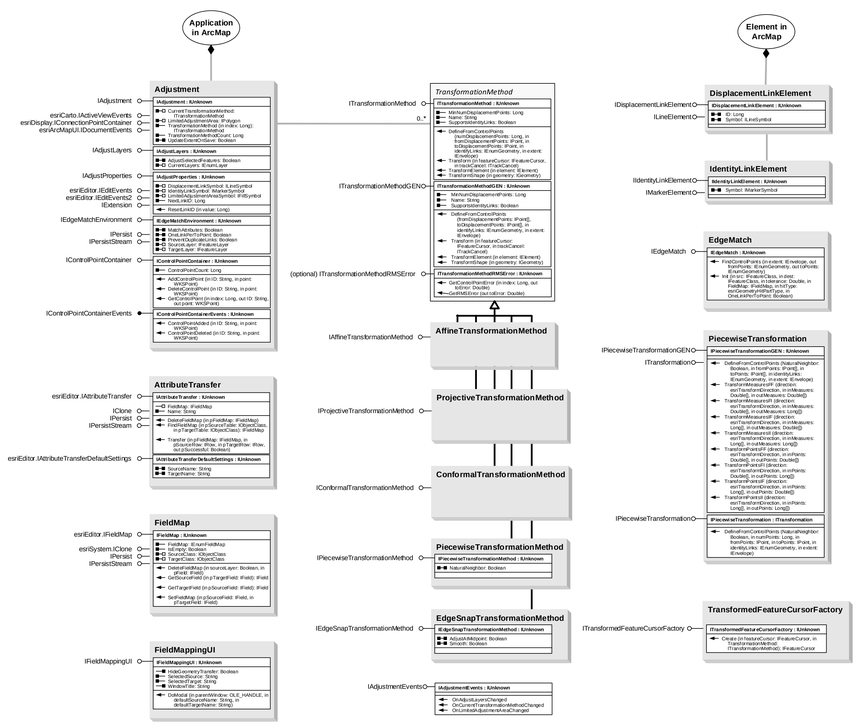
Esri® ArcGIS® 10.5

Copyright © 1999-2017 Esri. All rights reserved. Esri, ArcGIS, ArcObjects, ArcPad, and ArcMap are trademarks, registered trademarks, or service marks of Esri in the United States, the European Community, or certain other jurisdictions.

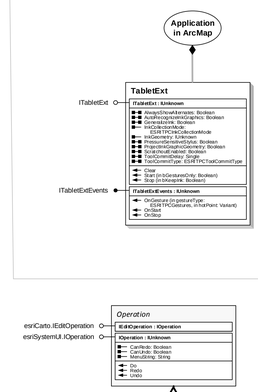
Utility Network Object Model



Spatial Adjustment Object Model



Tablet Extension Object Model



Topology Object Model

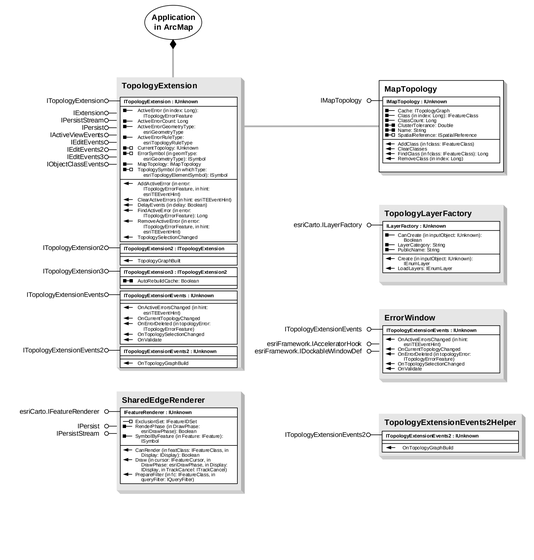
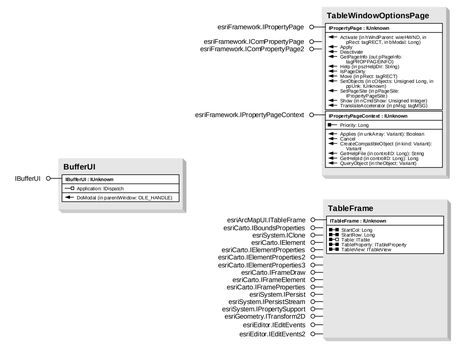
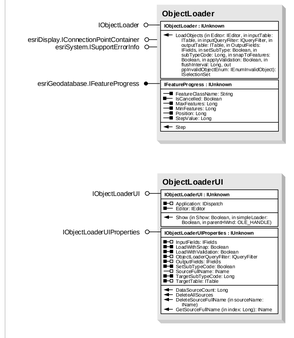


Table Window Object Model



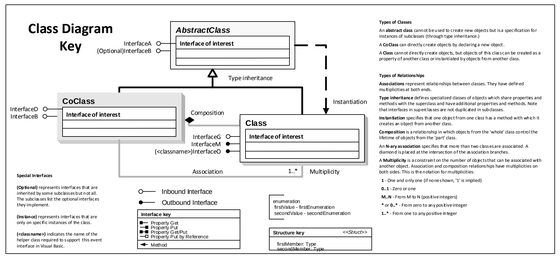
DataConverter Object Model



Enumerations

```
esriAnalysisType
0 - esriAnalysisOnAllFeatures
1 - esriAnalysisOnSelectedFeatures
2 - esriAnalysisOnNonSelectedFeatures

esriTEEventHint
0 - esriTEInit
1 - esriTEFixesActiveErrors
2 - esriTEDeleteActiveErrors
3 - esriTEValidate

esriTopologyElementSymbol
1 - esriTESelectedNodeSymbol
2 - esriTESelectedEdgeSymbol
3 - esriTENodeSymbol

esriTPCGestures
0 - ETPC_DawnLeft
1 - ETPC_RightUp
2 - ETPC_SerrateRight
3 - ETPC_ChevronLeft
4 - ETPC_ChevronRight
5 - ETPC_ChevronUp
6 - ETPC_ChevronDown
7 - ETPC_SerrateCircleLeft
8 - ETPC_SerrateCircleRight
9 - ETPC_ArrowLeft
10 - ETPC_ArrowRight
11 - ETPC_ArrowUp
12 - ETPC_ArrowDown

esriTPCVisibleCollectionMode
0 - ETPC_NotCollecting
1 - ETPC_GesturesOnly
2 - ETPC_HAndGestures

esriTPCTotCommitType
0 - ETPC_WinLoss
1 - ETPC_AutoCollapse
2 - ETPC_AutoFixed
```

Class Diagram Key



Special Interfaces

(Optional) represents interfaces that are inherited by some subclasses but not all. The subclasses list the optional interfaces they implement.

(Incoming) represents interfaces that are only on specific instances of the class.

(<classname>) indicates the name of the helper class required to support this event interface in Visual Basic.

Types of Classes

An **abstract class** cannot be used to create new objects but is a specification for instances of subclasses (through type inheritance.)

A **CoClass** can directly create objects by declaring a new object.

A **Class** cannot directly create objects, but objects of this class can be created as a property of another class or instantiated by objects from another class.

Types of Relationships

Associations represent relationships between classes. They have defined multiplicities at both ends.

Type inheritance defines specialized classes of objects which share properties and methods with the superclass and have additional properties and methods. Note that interfaces in superclasses are not duplicated in subclasses.

Instantiation specifies that one object from one class has a method with which it creates an object from another class.

Composition is a relationship in which objects from the 'whole' class control the lifetime of objects from the 'part' class.

An **N-ary association** specifies that more than two classes are associated. A diamond is placed at the intersection of the association branches.

A **Multiplicity** is a constraint on the number of objects that can be associated with another object. Association and composition relationships have multiplicities on both sides. This is the notation for multiplicities:

1 - One and only one (if none shown, '1' is implied)

0..1 - Zero or one

M..N - From M to N (positive integers)

***** or **0..*** - From zero to any positive integer

1..* - From one to any positive integer