ArcGIS™: Working With Geodatabase Topology

An ESRI® White Paper • May 2003

ESRI 380 New York St., Redlands, CA 92373-8100, USA • TEL 909-793-2853 • FAX 909-793-5953 • E-MAIL info@esri.com • WEB www.esri.com

J-9037

ArcGIS: Working With Geodatabase Topology

An ESRI White Paper

Contents **Page**

ArcGIS: Working With Geodatabase Topology

Overview

Topology—the spatial relationship between geographic features—is fundamental to ensuring data quality. Topology enables advanced spatial analysis and plays a fundamental role in ensuring the quality of a geographic information system (GIS) database. In the ArcInfo™ coverage model, hundreds of thousands of GIS users worldwide learned the benefits of topology by performing the build and clean operations on their databases. ESRI® ArcGIS™ 8.3 introduces a new suite of editing tools necessary to construct and maintain user-defined topological relationships within a geodatabase. In ArcGIS, the *Validate Topology* capability will ensure data integrity by validating the features of a geodatabase against a set of topology rules.

One of the primary objectives of the ArcGIS 8.3 release is to add full topology to the geodatabase. Until the ArcGIS 8.3 release, topology was a feature of the ArcInfo coverage data model. The introduction of the ArcGIS geodatabase provides an opportunity to expand what topology means for GIS users as well as its possible uses in modeling spatial data. Geodatabase topology is supported in ArcEditor™ 8.3 and ArcInfo 8.3.

Introduction

So what is really meant by topology, and why is it important in the maintenance of a geographic database in ArcGIS?

The word topology has several meanings when discussed in the GIS context. It can refer to a

- Theory or mathematical model of features in space
- Mechanism that allows features in the same or different feature classes to share geometry
- Set of editing tools that works with features in an integrated fashion
- Physical data model for feature data
- Set of validation rules for geographic features
- Mechanism for navigating between features using topological relationships

These aspects of topology are integrated in the coverage data model. By understanding the physical data model of a coverage with its arcs, nodes, polygons, routes, and regions,

one can also understand how a coverage represents space, how it is edited, and how it can be used to constrain and validate features as part of a database design. For example, if you need to represent nonoverlapping area features, such as soil polygons, you define them as polygons rather than regions. Or to build polygons, you digitize arcs and label points, and then run CLEAN.

In ArcGIS, topology is not based on a single physical data model such as the coverage. Topology has been generalized to operate in large, continuous databases, and the implementation is not tied to a specific physical data structure such as the coverage.

This white paper discusses the definition of topology as it applies to maintaining data within ArcGIS, how topology works within a geodatabase, and how to incorporate topology in your workflow.

How Is Topology Used?

Topology is used fundamentally to ensure data quality and aid data compilation. It is also a way to model the integrated behavior of different feature types (shared geometry). Examples of how a topology helps to model the real world in an integrated way include modeling a continuous fabric of land parcels, soil polygons, or county boundaries; a transportation network with street centerlines and bus routes; and a nested hierarchy of features such as census blocks, block groups, and tracts.

Topology for ESRI users has traditionally been associated with the coverage as a spatial data structure that is used primarily to ensure that associated data formats are consistent and have a clean topological fabric ("clean" implies that polygons close, lines that are supposed to snap together do, etc.). An alternative view of topology is that it is a collection of editing tools and techniques that are coupled with an approach to modeling that integrates the behavior of different feature types and supports different types of key relationships. In other words, topology may be considered from the feature geometry and behavior context, not just from the data structure perspective. This alternative view may also be employed to ensure that the associated data forms a clean and consistent topological fabric from the end user's perspective. ArcGIS uses this latter view to support topology within the geodatabase.

Users can set the following fundamental requirements of topology:

- Declare and limit how features share geometry.
- Create features from unstructured geometry (e.g., aggregation and snapping).
- Edit features while supporting a shared geometry model and support the management of topological constraints.
- Support a continuous, multiuser database architecture that leverages the relational model and handles very large data sets.

Setting Topology Rules

Topology is a set of rules about how features within collections of feature classes share geometry. This set of rules is used to improve the user experience for editing features, validating features, and tracing through the geometric relationships between features.

- Features can share geometry within a topology in the following ways:
  - Line features can share endpoints (arc–node topology).
  - Area features can share boundaries (polygon topology).
  - Line features can share segments with other line features (route topology).
  - Area features can be coincident with other area features (region topology).
  - Line features can share endpoint vertices with other point features (node topology).
  - Point features can share geometry with line features.
- Integrity rules are used to organize and define topology as follows:
  - Snapping and editing behavior (snapping of vertices during CLEAN/*Validate*)
  - Calculation of shared geometry
  - Shared geometric elements (vertices and line segments) among features
  - Relationships between features
  - Validation rules using topological relationships
  - Logical networks using topological relationships

Data Models and Topology

There are a variety of topological data models that users commonly require. These data models may contain exclusively network (one-dimensional) topologies (e.g., a utility network), areal (two-dimensional) topologies (e.g., a land parcel cadastre), or a combination of the two (e.g., a census database containing street networks, address ranges, and legal boundaries).

A topology manages spatial relationships between features for a set of feature classes within the same feature data set (the feature data set being used to organize feature classes for a topology). This includes modeling geometric elements, which are shared between features (e.g., the common boundary between adjacent polygon features). The vertical integration of a set of features by defining a common set of underlying shared topological elements is the essential concept of geodatabase topology.

A topology is a database object that defines a formal mathematical model integrating geometry from one or more feature classes in a feature data set. When feature classes are integrated in a topology, they share a common set of topological elements. The topology manages these elements including their relationships to one another and to the features in the contributing feature classes.

J-9037

A topology supports two views of the feature information. First, it supports the standard feature-oriented view of information. In this view, a topology consists of a set of geometric objects organized into thematic feature classes. In addition, a topology supports the view of geographic information as a graph of topology elements, in which each element can be associated with zero or more thematic features.

In general, the geodatabase model defines a generic model for geographic information. This model allows you to define relationships between objects, together with rules for maintaining the referential integrity between objects. ArcGIS topology, as implemented on top of a geodatabase, fits into this paradigm. Topology on top of a geodatabase uses a feature data set and its collection of simple feature classes (a generic model), a topology that defines relationships and integrity rules, and a set of tools to maintain the relationships.

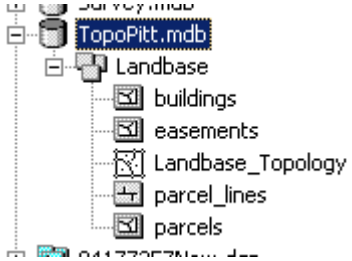
Understanding How It Works

The Basics

Topology in ArcGIS begins with the definition of parameters to be included. These parameters are defined through a wizard in ArcCatalog™. Topologies have the following characteristics:

- They live within feature data sets (Figure 1) since all participating feature classes must have the same spatial reference.
- There can be multiple topologies within a data set.
- Feature classes can only participate in one topology.
- Feature classes cannot participate in both a topology and a geometric network.
- A topology can contain multiple point, line, and polygon feature classes.

Figure 1



Topologies (Landbase_Topology in this case) live within a feature data set.

In addition to feature classes participating, there are other parameters that help define a topology. These additional parameters are

- Cluster tolerance
- Relative ranks for each feature class
- Rules

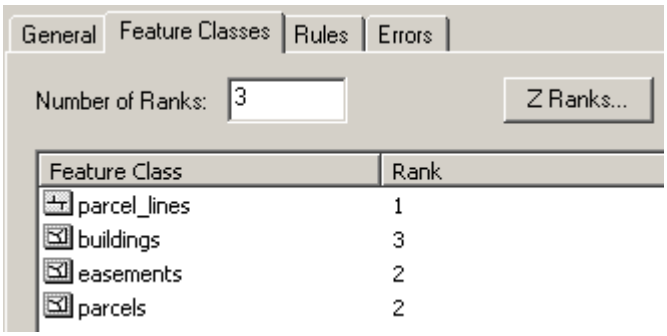
Cluster Tolerance

A single cluster tolerance is defined for the topology. The cluster tolerance is a distance range in which all vertices and boundaries are considered identical or coincident. Vertices and endpoints falling within the cluster tolerance are snapped together during a *Validate Topology* process. The default value for this tolerance is the precision defined for the spatial reference of the feature data set. A general rule of thumb is that the cluster tolerance should be at least one magnitude less than the accuracy of your data. For example, if your data set has two-meter accuracy, the cluster tolerance should be 0.2 meter or less.

Ranks

Ranks are defined at a feature class level, and they control how much the features in that class can potentially move in relation to features in other classes when a topology is validated (Figure 2). If your feature classes contain *z* values, you can apply a *z*-level ranking as well. The higher the rank, the less the features may move in relation to features from other classes. The highest rank is 1. For example, a topology is defined with two feature classes: parcels and parcel lines. These two features classes need to be coincident, but in some cases they are not. In the cases in which the feature classes are not coincident, the parcels should move and snap to the parcel lines. To accomplish this, the parcel lines are given a higher rank than the parcel boundaries. If the two classes have the same rank, then averaging is used to determine the location of the snapped features.

Figure 2



Ranks are used to specify the relative potential of one feature class moving versus another during a Validate *operation. Define the rank classification scheme for the topology by setting the number of ranks a topology can have. The rank values in the figure indicate that the parcels will move to the parcel lines when necessary.*

Topology Rules

Every topology in the geodatabase is associated with a set of topology rules. Users define topology integrity by adding and removing rules from this set. A topology rule defines a condition in the topology, which is a problem or possible problem in the topology features. Examples of topology rules are

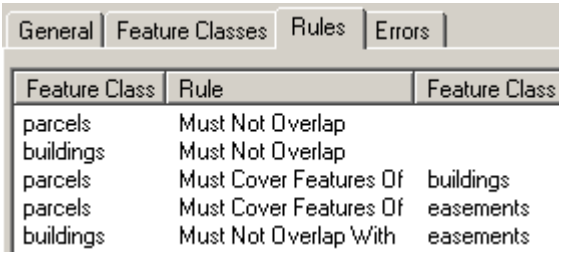
- Land parcels Must Not Overlap.
- Census blocks Must Be Contained Within Block Groups.

Topology rules can be defined for the features within a feature class or alternatively for features between two feature classes. When a topology is validated, the rules are tested. A topology error is generated for each instance of a rule violation. For example, when a dangling edge is found, a flag is generated for the edge (assuming a rule has been defined

for this condition). This flag contains information necessary to draw and understand the topology rule violation. Flags are used to understand the integrity status of the topology, to draw "error plots" for quality assurance, and as objects used in editing workflows to look up and fix problems in the topology.

Rules help you manage the topological relationships between features within a feature class or between feature classes (Figure 3). Coverages also have a set of rules associated with them, but these rules are only between the features in the coverage and are fixed. All the rules inherent in coverage topology are represented in ArcGIS topology.

Figure 3



Rules are used to specify the spatial relationships between and within feature classes. The rules in the figure demonstrate some of the options for polygon feature classes.

Defining a rule in your topology does not guarantee it will never be violated, but it does ensure occurrences in violation of the rule will be marked as errors. Allowing errors to exist and persist in the database creates an environment in which many diverse workflows can be applied. Most workflows call for the features to be in violation of specified rules at some point during the operation. With ArcGIS topology, the user has the ability to decide when rules will be checked by choosing when to run a *Validate Topology* operation.

A topology has three states. A topology is *not Validated* if there are areas in the topology that are undefined as a result of edits on the topology. The undefined areas are called *dirty areas*. Dirty areas are examined by the *Validate Topology* command, resulting in one of the other two states of a topology. *Validated with errors* indicates the *Validate Topology* command has been run but there are errors in the topology. *Validated without errors* indicates there are no errors or dirty areas in the topology.

Dirty Areas

Dirty areas represent regions where the spatial integrity of the topology has not been validated (i.e., *Validate Topology* has not been run). When a topology is initially created, the entire area covered by the participating feature classes is dirty. All this means is that the database has not been checked to ensure vertices within the cluster tolerance are snapped and there are no violations of the topology's integrity rules. Dirty areas may contain existing or undiscovered errors inside them. The *Validate Topology* command is used to integrate geometry, check dirty areas, and discover errors. After the initial *Validate,* new dirty areas are created by performing any type of spatial edit (e.g., a move, split) or certain types of attribute updates (e.g., changing the subtype of a feature will create a dirty area if the subtypes are involved in a rule) on one of the features participating in the topology. When an edit occurs, the area around the feature that was edited is marked as dirty.

It is important to note here that dirty areas are allowed to persist in the database. The user is not required to run *Validate Topology* on dirty areas before saving changes or even reconciling and posting to a multiuser database. However, there is no way to ensure there are no rule violations or features that should be snapped together within a dirty area unless a *Validate* operation has been run.

Users can define their own workflow for running *Validate Topology* on dirty areas. Some may choose to run *Validate* after every edit operation, while others will want to perform all their edits and then run *Validate* a single time before saving.

The *Validate Topology* Command

The section above discusses the different parameters that define a topology. Once these parameters have been set and the topology has been created, it is the *Validate Topology* command that is used to snap features together and check for violations of the defined rules. The *Validate Topology* operation can be run against all dirty areas, dirty areas in the visible extent, or dirty areas within a user-specified extent.

Validate Topology begins by snapping together all vertices that fall within the cluster tolerance, taking into account the ranking value of each feature class in the process. It is important to note that snapping only occurs when the vertices of features fall within the cluster tolerance. If you have made sure all your features snapped together during your edits, then there is less chance of further movement of features because of the *Validate Topology* command. Another way to limit the potential of feature movement as a result of running the *Validate* operation is to keep a small cluster tolerance. If feature vertices are found within the cluster tolerance, the features from the feature class with the lowest relative rank will be moved to the features with the higher relative rank. As part of the snapping routine, *Validate* will also add vertices where features intersect and there is not already a vertex.

As part of the snapping process within *Validate,* all features need to be aligned to an underlying integer grid. The resolution of this grid is based on the precision of the data set. The more accurate the data is, the smaller the resolution of the integer grid. When *Validate* is run on a data set for the first time, all features will be aligned to this grid, which means there is a chance for coordinate movement. Since all features are aligned with the grid, you can be assured that features that are coincident will remain coincident. The same potential for movement is there when you run CLEAN on a coverage in ArcInfo Workstation. After the initial *Validate,* any potential movement can be minimized by making sure new features always snap to existing features.

After the *Validate* operation performs checks for vertices within the cluster tolerance, the rules defined within the topology are verified. Any rule violations that are found are marked as errors. In ArcMap™, errors can be displayed (if the proper symbology is turned on) or listed in the Error Inspector. A complete error listing is available in the properties of the topology (accessible in both ArcCatalog or ArcMap). Once an error has been marked, it is the user's responsibility to fix the error or mark it as an exception. The Fix Topology Error tool or the Error Inspector in ArcMap can be used for resolving errors or marking them as exceptions. Subsequent executions of *Validate* will not regenerate errors that have been marked as exceptions, but the user always has the option of displaying exceptions and remarking them as errors. As with dirty areas, errors and exceptions can be persisted in the database.

Validate Topology *Versus BUILD*

Topology users who are familiar with ArcInfo Workstation are most likely familiar with the BUILD command. BUILD is used in ArcInfo Workstation to rebuild the topology and attribute tables of a coverage without executing the workstation CLEAN command.

In ArcGIS, the *Validate Topology* command is used to ensure data integrity by validating the features in a topology against the topology rules. To validate the feature for errors, geodatabase topology requires that the following three basic conditions be met:

1. There are no intersecting line segments (either from a line or polygon feature class). If two lines intersect, they do not need to be split to satisfy this condition; a vertex needs to exist at the intersection point for each feature.

2. No two vertices are within the cluster tolerance.

3. No vertex is closer than the cluster tolerance to a line segment.

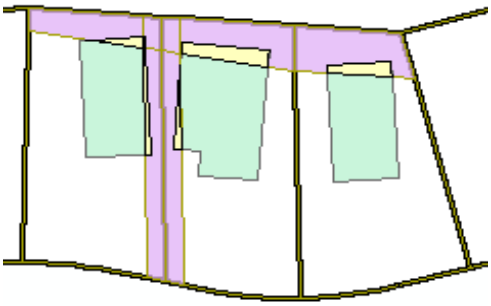
In these conditions, a vertex is considered a point or any shape defining the point of a line or polygon. Once these conditions are met, rule verification can occur.

Errors and Exceptions

A major part of topology is the defining and subsequent validation of rules. Rules are defined as part of the schema definition for a topology. When dirty areas in a topology are examined with *Validate Topology,* an error is generated for each instance of a topology rule that is determined to be invalid. For example, if a rule is set that states buildings cannot overlap with easements, then an error will be generated for each building feature that overlaps an easement feature. Figure 4 demonstrates how the errors are symbolized in ArcMap, while Figure 5 shows the same errors listed in the Error Inspector.

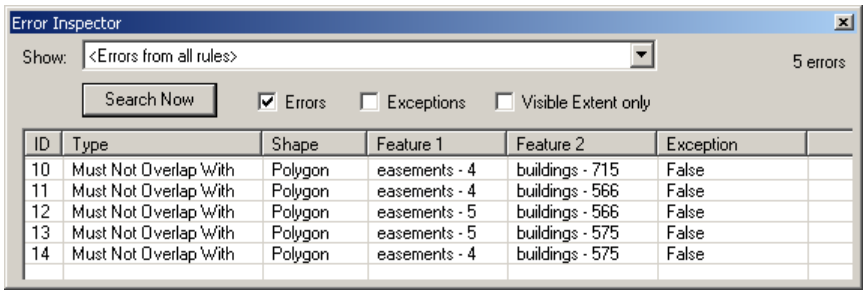
J-9037

Figure 4



Newly added building features (in green) are in violation of the topology rule that says buildings must not overlap easements (shown in magenta). After running Validate *on the dirty areas, five errors (shown in yellow) are generated. There are five errors because two buildings (the one on the left and the one in the center) overlap multiple easement features.*

Figure 5



ID	Type	Shape	Feature 1	Feature 2	Exception
10	Must Not Overlap With	Polygon	easements - 4	buildings - 715	False
11	Must Not Overlap With	Polygon	easements - 4	buildings - 566	False
12	Must Not Overlap With	Polygon	easements - 5	buildings - 566	False
13	Must Not Overlap With	Polygon	easements - 5	buildings - 575	False
14	Must Not Overlap With	Polygon	easements - 4	buildings - 575	False

The Error Inspector can be used to display the errors and exceptions present in the topology. In this case the Error Inspector lists the errors generated for buildings overlapping easements.

As mentioned previously, errors are persisted in the database. Users have the option of saving edits (as well as reconciling and posting) without fixing errors if they choose. The system was designed this way to ensure all possible workflow variations can be supported and the feature class is always usable (e.g., polygons can be drawn at any time without having to run *Validate*).

When an error is discovered by *Validate,* the user has the following three basic options:

1. Leave the error unresolved in the database.
2. Fix the error using the Fix Topology Error tool or some other method.
3. Mark the error as an exception.

The Fix Topology Error tool offers a variety of methods for resolving an error depending on the error and the feature type. For instance, using the building overlapping an easement example, the resulting error shape is the area of the overlap or a polygon. The Fix Topology Error tool has options for resolving polygon errors: create a new feature, subtract the area from both features, or merge the overlap with one of the two features. In this particular example, the user could also resolve the error by selecting the building and moving it to a new location.

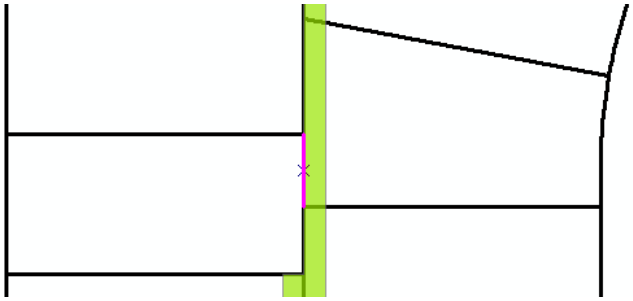
When resolving errors, the user always has the option of marking an individual error or a collection of errors as exceptions. There are instances when the occurrence of a defined error may actually be acceptable. In these cases the error should be marked as an exception. For instance, in the building and easement example, it may be admissible to have buildings that fall within a designated easement feature. Once an error has been marked as an exception, it remains so until it is reset back to an error. Running *Validate Topology* on the same area will not generate an error for an instance that has been marked as an exception.

Shared Geometry

What can be done with the data as a result of running *Validate?* One of the real benefits of running *Validate* and having vertices snap together when necessary is that it results in shared geometry. In ArcGIS, polygons are stored as closed rings, meaning the boundary between the polygons is stored with each polygon feature. A coincident boundary is discovered when it is required on the fly. When you click on a line or point with the Topology Edit tool, the tool determines the features that share geometry with that location.

In Figure 6 the Topology Edit tool was used to select the shared geometry between two parcels and an easement feature (show in green). The selected edge (highlighted in magenta) is stored with each parcel and the easement feature. Edits can now be performed on the shared geometry, and they will affect all three features.

Figure 6

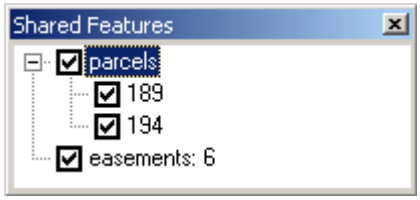


Shared geometry (such as the common boundary between the two parcels and the easement in green) is discovered during the editing process.

There are advantages to discovering shared geometries during edits as opposed to storing them in the database. The foremost is performance. Another compelling reason for using the discovered approach is the flexibility it offers for editing. Using the features shown in Figure 6, suppose you find the easement feature really did not follow the boundaries of the parcels but should be 10 feet to the left instead. If you are working in an environment in which the boundary is only stored once, it would be a very time-consuming process to redigitize the easement line 10 feet to the left while still maintaining the original width. In the ArcGIS "discovered" environment, you select the feature and move it without having to worry about the impact on the parcels that share a boundary. If you want to

move the parcel boundaries without taking the easement along, you can use the Show Shared Features command, which will give you a list of the features sharing the selected edge. Figure 7 shows the listing of shared features. Unchecking the easement feature allows you to update the two parcels independent of the easement. Doing this in the stored topology environment requires redigitizing lines and several other steps.

Figure 7



The Show Shared Features command will display a dialog listing the features that share geometry with the selected edge.

Coverage Model Versus Geodatabase Model

The coverage model has provided ArcInfo users with many benefits for storing data. There are also many benefits to storing data in the new geodatabase model. If you look closely at the two models, the following similarities exist:

- Both are a collection of feature classes.
- Both offer the capability to share geometry (although they use a different approach).
- Both employ a CLEAN/*Validate* process with integrity rules/flags for maintaining topology.

With coverages, if you want to have different feature classes that share the same geometry, you can create either regions or routes on top of the underlying features. For instance, suppose you want to maintain parcels, lots, and easements within the database and you want them to share geometry where appropriate. You would have a coverage that contained arc and polygon features along with three region subclasses representing the parcels, lots, and easements. You would be required to manage the atomic polygon features even though the features you are really interested in are in the region feature classes. The topological rules for the coverage are hardwired and unavoidable (you cannot violate them).

When working with a geodatabase, the parcels, lots, and easements are separate feature classes within a data set. The polygon feature class is not necessary, and lines are only necessary if attributes are maintained on them (e.g., symbology information). In each case the edit tools allow the user to maintain and manipulate the common boundaries between the features as shared geometry. In addition, with the geodatabase model, users can define and enforce rules. For example, the lots and parcels cannot overlap, but the easements can overlap. The leap of faith that has to be made from a coverage to geodatabase topology is in understanding that topological relationships are still supported and maintained; they simply are not stored in a table that can be viewed.

Similar to features in a coverage, features in a geodatabase topology are generic point, line, and polygon features managed by the system. They are not specialized custom features with hidden characteristics and traits the user does not have access to. One thing to keep in mind, though, is that all line features in a geodatabase are essentially routes and all polygon features are regions. Lines and polygons in a geodatabase can overlap each other, so they function in the way advanced features (e.g., routes and regions) do in coverages.

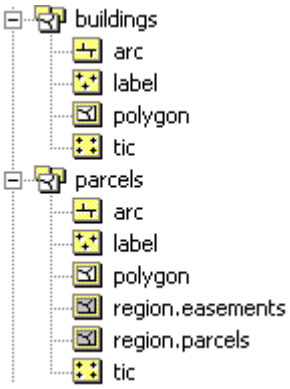
In a coverage polygon, line feature classes are necessary to support regions and routes. In a geodatabase, they are not. This approach requires additional effort to make sure polygons and lines do not overlap when they should not. This is where topology comes in. Adding topology to a geodatabase allows users to define integrity rules on the feature classes. Rules such as no overlaps or gaps for my polygons (regions), routes must be on top of other routes, and points must be on top of route endpoints allow the user to maintain coverage-like integrity on the data when it is necessary.

Working With Topology

Converting Coverage Data to Geodatabase Feature Classes

To migrate a coverage to a geodatabase data set with topology, begin by observing the coverages in ArcCatalog (Figure 8).

Figure 8

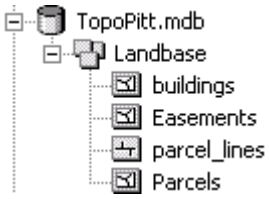


The feature classes in two coverages are shown as they appear in ArcCatalog.

Only those feature classes with feature attribute tables (FAT) need to be considered for migration to the geodatabase. For the buildings coverage, the polygon feature class is of interest. For the parcels coverage, the two region subclasses (easements and parcels) along with the arc feature class need to be migrated. Arcs are only being migrated in order to maintain attributes on them for symbolization. In the case of the polygon feature class, it does have a feature attribute table associated with it; however, it is only the region classes defined on top that are of interest. As a result, it is not necessary to move the polygon class to the geodatabase. The first step in the process then is to use the capabilities of ArcCatalog or ArcToolbox™ to import the three coverage feature classes from the parcels coverage and the one feature class from the buildings coverage into a geodatabase data set. Figure 9 shows the new data set after the import.

J-9037

Figure 9



After creating the TopoPitt database and Landbase data set, the Import command was used to bring the coverage feature classes into the geodatabase.

When coverage feature data is originally loaded into a geodatabase, the result is simple feature classes. At this point it would be possible to go into ArcMap and begin editing the feature classes, but there would not be any tools to help ensure the features share geometry when appropriate. Building a topology on the feature classes that were just loaded enables you to create integrity rules for the data and expose the shared geometry editing capabilities.

Building a Topology

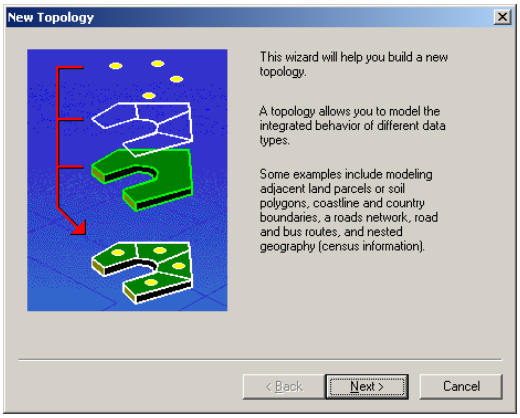
A topology can be created in ArcCatalog by right-clicking on the data set name and selecting New -> Topology from the context menu. A wizard appears to guide the user through the creation of the topology. For this example, the data set loaded from coverages in Converting Coverage Data to Geodatabase Feature Classes above will be used.

Topology Wizard in ArcGIS

Use the topology wizard to build a topology from existing simple feature classes. The reason certain options are chosen in the wizard will be discussed.

1. The first page of the wizard (Figure 10) offers a description of what a topology is and why you would want to create one. Click Next to move to the second page.

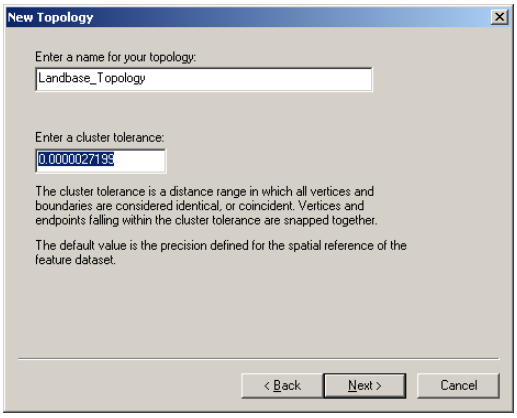
Figure 10



When the wizard starts, it explains what a topology is and why you would want to use one.

2. The second page of the wizard (Figure 11) lets you set the name of the topology that you are about to create and specify the cluster tolerance to be used when analyzing the topology. For this example, the topology is named Landbase_Topology and the default cluster tolerance of 0.0000027199 is used.

Figure 11

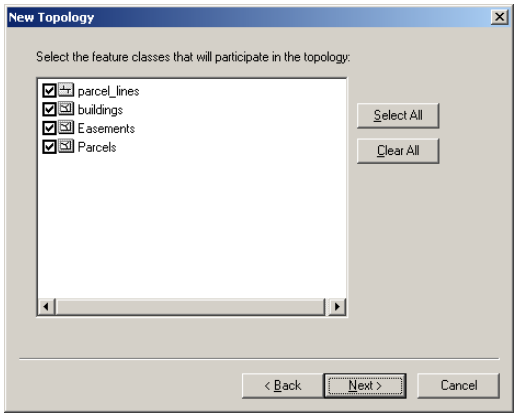


Page 2 of the wizard allows the user to enter a unique name for the topology and specify the cluster tolerance. The default cluster tolerance is the inverse of the precision of the feature data sets' spatial reference.

3. The third page of the wizard (Figure 12) is for specifying the feature classes that will participate in the topology. The user can specify any number of point, line, and

polygon feature classes that can participate, but all of the feature classes must reside in the same data set. For this example, all of the feature classes are used.

Figure 12

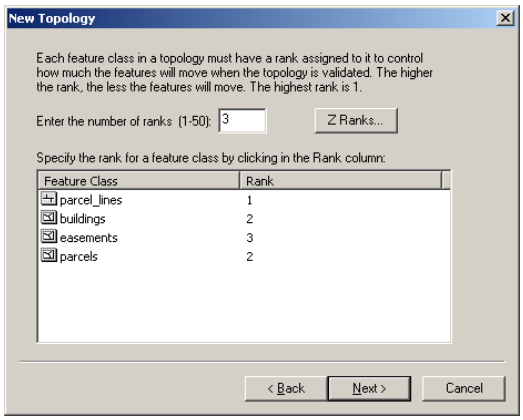


Page 3 of the wizard lists all the feature classes in the feature data set that can participate in the topology. Classes that would not show up include those that are already in a topology and those that participate in a geometric network.

4. The fourth page of the wizard (Figure 13) is used to specify the relative ranking of each feature class within the topology. Only the feature classes selected on the previous page of the wizard will show up in the list. For this example, the rank of the parcel_lines layer is increased and the rank of the Parcels layer is decreased because the parcels need to snap to the parcel_lines when necessary. As mentioned previously, snapping only occurs when vertices fall within the cluster tolerance.

J-9037

Figure 13



Ranks are used to control the accuracy a vertex in a given feature class has relative to vertices in other feature classes. Ranks are only applied to vertices within the cluster tolerance of each other. You can set ranking schemes based on relative x,y *and* z *coordinate space. The feature class that contains a rank value of one contains the highest accuracy.*

Note that this is only significant for vertices within the cluster tolerance of each other. By default, rank values for all feature classes are the same. In the case in which rank values are the same, relative location will be averaged.

5. The fifth page of the wizard may be the most critical. This is where you define the integrity rules to apply to your database. Rules are verified during the *Validate Topology* operation, and violations can be reported in either ArcCatalog or ArcMap. Tools for fixing rule violations (errors) can be found on the Topology toolbar in ArcMap.

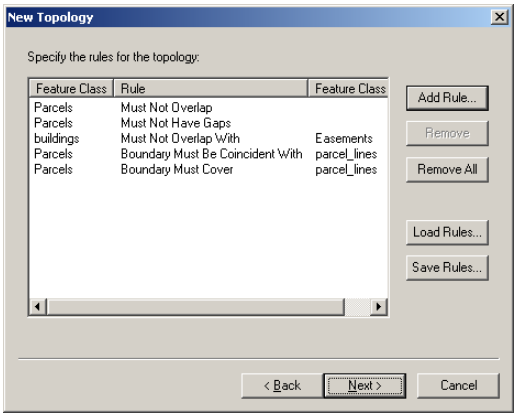
   For this example database, the following rules are of interest:

   - Parcels cannot overlap or have gaps. The gap rule requires the user to have features for road rights-of-way or to mark these areas as exceptions when the errors are generated.
   - Buildings cannot overlap with easements or each other.
   - Parcel_lines must coincide with the boundaries of parcels.

   The no gap requirement for parcels requires features for road rights-of-way or to mark these areas as exceptions when the errors are generated. The coincidence requirement for parcel_lines is included in order to symbolize the boundaries of the parcels based on attributes on the parcel_lines and to maintain coordinate geometry (bearing and distance) information on the parcel_lines. The requirements specified result in the following rules (Figure 14) being created (referenced by the feature classes involved):

- Parcels—Must Not Overlap
- Parcels—Must Not Have Gaps
- Buildings/Easements—Must Not Overlap With
- Parcels/parcel_lines—Boundary Must Be Coincident With
- Parcels/parcel_lines—Boundary Must Cover

Figure 14



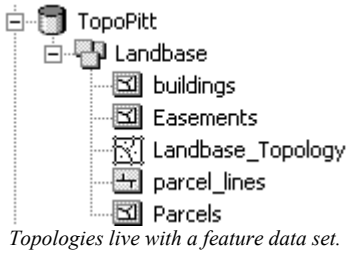
Rules help you manage the topological relationships between features within a feature class or features between feature classes. Rules may be set at both the feature class and subtype level.

6. The final panel is a summary page that lists all of the choices made in the wizard. Click Finish to create the topology. After the topology has been created, another dialog box appears asking if you would like to run *Validate Topology* against the database at this time. Keep in mind that upon creation of the topology the entire database is dirty, and it must be validated at some point. Having a database with dirty areas does not necessarily mean there are rule violations or vertices to snap together; it just means that the *Validate Topology* operation needs to be run on the area(s) to check for these conditions.

   When the topology has been created, it shows up as an entry within the data set. Figure 15 shows the topology created in the Landbase data set. Once the topology has been generated and validated, it is ready for editing.

J-9037

Figure 15



Topologies live with a feature data set.

Editing a Topology

In the previous two sections converting coverages to a geodatabase and building a topology from the resulting feature classes were discussed. In this section the editing of a topology is discussed. As mentioned at the end of Building a Topology, the entire database is dirty after building a new topology. *Validate* does need to be run on the database at some point to validate the dirty areas. Any time *Validate Topology* is run, there is the potential to generate errors (violations of the specified rules). Errors are listed in either ArcCatalog or ArcMap, but they can only be repaired within ArcMap. *Validate* was run on the database created in the previous sections, and there were no errors initially, so the editing process can now begin.

First, the topology needs to be loaded into ArcMap. Adding the Landbase_Topology class to ArcMap will result in all the participating feature classes being added to the map. Topology has its own set of symbology users can turn on and off depending on what they would like to see displayed. The user has the option of displaying dirty areas, errors (broken down by feature type and type of error), and exceptions (broken down by feature type).

When editing topological data sets in ArcMap, the user has access to all of the general functionality of the Editor. Features can be selected, updated, added, and deleted just as they always could. Having a topological data set does not restrict what you can do with the Editor in the general sense, but there are two important points to remember. The first one is that all nonattribute edits will create dirty areas. The second one is that using the Topology Edit tool and the Show Shared Features command is the easiest way to work with shared geometries.

Dirty areas have been discussed in various sections of this paper. The one point to consider here is your workflow in dealing with dirty areas. Dirty areas can persist in the database so it is not necessary to validate these areas at any specific time. Users have the option of validating dirty areas after each edit, before saving, before reconciling and posting, or at any other time they desire. *Validate Topology* can be run from ArcCatalog, but you will most likely run it from ArcMap as part of an edit session. When running *Validate* from ArcCatalog, the only option available is to process all the dirty areas. Within ArcMap the user has the option of analyzing a single dirty area, selected areas, areas within the visible extent, or all dirty areas. The most common workflow is to perform all necessary edits, validate topology, and resolve errors before saving.

Working with shared geometries is another topic that has been discussed but without much detail about how to perform edits. The sample database was defined with rules such as parcel_lines need to be coincident with parcels. While editing, there is nothing to stop you from activating the Edit tool, selecting a parcel_line feature, and moving it to an area where it would not be coincident with a parcel feature. When *Validate* is run on the

resulting dirty areas, it will generate some errors based on the edits performed. The question becomes, "What is the easiest way to edit without generating a number of errors?" The easiest way is to use the Topology Edit tool to select and edit shared geometries (coincident points and edges). This tool has its own context menu with editing options and can also be used in conjunction with the Sketch tool to manipulate the shared geometries. The result is that all coincident features have their points and edges updated at the same time and no coincidence errors will be generated. Although the Topology Edit tool is the most direct way to deal with shared geometries, the user can still access the standard Editor toolbar because it offers the greatest amount of flexibility. There are valid workflows that will use the general purpose tools of the Editor and, as such, no restrictions should be placed on them.

Summary

This white paper has introduced the concepts of topology within a geodatabase. There are many similarities between this topology model and the one employed by the coverage, but there are also many differences. The basic premise to keep in mind is that the makeup of the coverage data model and the geodatabase data model is quite similar (you maintain the feature classes for which you have attributes). The most significant difference in designing your topology is how the topology is represented. The coverage model stores the topological information within the database and employs a rigid set of tools for operating on the features. The result is a tightly controlled environment in which the workflow is dictated to the user and topological integrity is steadfastly maintained. The ability to pick up a polygon feature and move it is not allowed because of the effect it would have on all the associated features and geometries. In some environments this is good since it keeps the user from separating regions from their supporting polygons.

Geodatabase topology stores very little information in the database. Only the parameters of the topology (e.g., participating feature classes, cluster tolerance, relative ranks, and rules), dirty areas, errors, and exceptions are stored. All other information is discovered on the fly during the manipulation of the participating feature classes. The result is a very loosely controlled environment in which just about any workflow can be employed and topological integrity is validated only at designated times. In geodatabase topology, the responsibility for maintaining topologically correct features falls on the user, although this can be controlled by using the Topology Edit tool when working with shared geometries. In the end, it offers a more flexible environment and the ability to define and apply a wider set of integrity rules and constraints.

Topology in the geodatabase is supported in ArcEditor 8.3 and ArcInfo 8.3. For more information, visit www.esri.com/arcgis.