Lecture 3: Making features “smart”

GEOG 419: Advanced GIS
Dr. Scott
Spring 2004

Qualities of features

► Shape
  • Geometry feature class
    ▶ Points and multipoints
    ▶ Polylines (line segments that may or may not be connected)
    ▶ Polygons (set of either disjoint or embedded rings)
      ▶ Ring: set of connected, closed, nonintersecting line segments
      ▶ Line segments can be straight, circular arcs, Bézier curves, and elliptical arcs

Qualities of features II

► Spatial reference
  • Specifies how the x,y coordinates of a set of features are mapped onto the earth’s surface
► Attributes
  • Fields in a feature class table
  • Standard or custom properties of features
  • Numeric, textual, descriptive
► Subtypes
  • Feature class is homogenous set of features
  ▶ May be considerable variation among features
  • Buildings ➔ Residential, Commercial, Industrial
  • Subtypes give more control over qualities of features

Qualities of features III

► Relationships
  • Among geographic features
  ▶ Between geographic features and nonspatial objects
► Attribute constraint
  • Attribute domain: numeric range or list of valid values
► Rule Validation
► Topology
  • Precise spatial relationships
  • Planar topology
  • Geometric network
► Complex behavior
  • Custom editing interaction, intrinsic analytical capabilities, sophisticated cartographic rendering

Steps to making features smart

► Progressively adding intelligence
► Select feature type and topology
  • Inventory all of the types of objects you need to model
  • Establish feature datasets to group feature classes that are bound by spatial reference, topology, and thematic content
    ▶ Nonspatial: create object classes
    ▶ Spatial: create simple feature classes (point, line polygon)
    ▶ Topological: create a graph with topological feature classes in a common feature dataset

Steps to making features smart II

► Set attribution and subtypes
  • Subtype – special attribute
    ▶ Major groupings of objects
    ▶ Can express diversity among similar objects/features without creating lots of classes
    ▶ Improve data integrity with attribute domains, default values, connectivity rules, relationship rules
► Define attribute domains and validation rules
► Establish object relationships
  • Relationship classes, relationship rules
► Create custom objects
  • Visual Basic programming
Geodatabase Design Considerations

Keep features/objects in same geodatabase if:
- Objects/features have relationships
- Features that have topological associations
- In order to concurrently edit features
  - Can only edit one geodatabase at a time

Separate features/objects in separate geodatabases if:
- Different departments have responsibility for different datasets
- Use of different commercial RDBMS
- Personal geodatabases have practical size limits

Geodatabase Design Considerations II

Classes (object, feature, relationship) can either stand alone or reside in a feature dataset

Group classes in a feature dataset if:
- If feature classes are topologically related
- If you wish to enforce a common spatial reference for a set of feature classes
- Logical organization

Geodatabase Design Considerations III

Should a group of related features be separate feature classes or subtypes of the same feature class?

The more feature classes, the worse the geodatabase performance

Why split into feature classes?
- Each group of features requires distinct custom behavior
- When feature attributes are different
- If you need different access privileges for different classes
- Some features are to use versioning and some not

Storing Data in Tables

All objects, features, and relationships are stored in tables

Predefined fields
- For identifying objects and storing feature shapes
- Managed by ArcInfo and should NEVER be modified via another RDBMS

Custom fields
- All others

Adding Simple Behavior with Subtypes

Primarily used to verify the integrity of objects

Subtypes (lightweight subdivision)
- Can name them
- Can define distinct attribute domains
  - Constraints on data values
  - Split and merge policies
- Can define distinct default values
- Can define distinct validation rules
- Can prescribe relationship types
- With programming, you can add custom rules

Relationships Among Objects

Topological
- Connectivity, adjacency

Spatial
- Touches, coincides with, overlaps, is inside of, is outside of

General
- Persistent tie between objects/features