Geodatabase and Object Model Design Using CASE Tools

Julio Andrade

Erik Hoel

Goals

- **Develop an understanding of**
  - **when to use CASE versus ArcCatalog**
  - **how to represent data models in UML**
  - **how to run the schema wizard**
- **How to proceed forward**
  - **other UC'00 sessions**
  - **literature**

Agenda

- What is CASE
- Database design
- ArcInfo 8 Geodatabase
- Representing the Geodatabase using UML
- Running the schema wizard
- Demo

What is CASE?

CASE

- Computer Aided Software Engineering
- Used to specify data / object models
  - classes / components (software)
  - database schemas
- Graphic modeling languages
  - historical - OMT, Booch, ER
  - current - UML

CASE

- Commercial products
  - Visio Enterprise
  - Rational Rose
  - Paradigm Plus (CA)
  - Popkin System Architect
- ArcInfo 8 requirements
  - support for UML
  - support for Microsoft Repository

Database Design

Continuum of Database Design

- Natively utilize Coverages and Shapefiles
- Import data into the Geodatabase
- Utilize ArcCatalog to refine and extend existing classes
- Use CASE and UML for a ground-up redesign of a large system

CASE Wizards vs. ArcCatalog

- ArcCatalog
  - excels at tactical modifications
  - intended for modest models
  - user difficulty with large complex models
- CASE Wizards and UML
  - a strategic approach
  - very good for total system redesign
  - intended for maintaining complex models
  - learning curve for CASE tools and UML

ArcInfo 8 Geodatabase

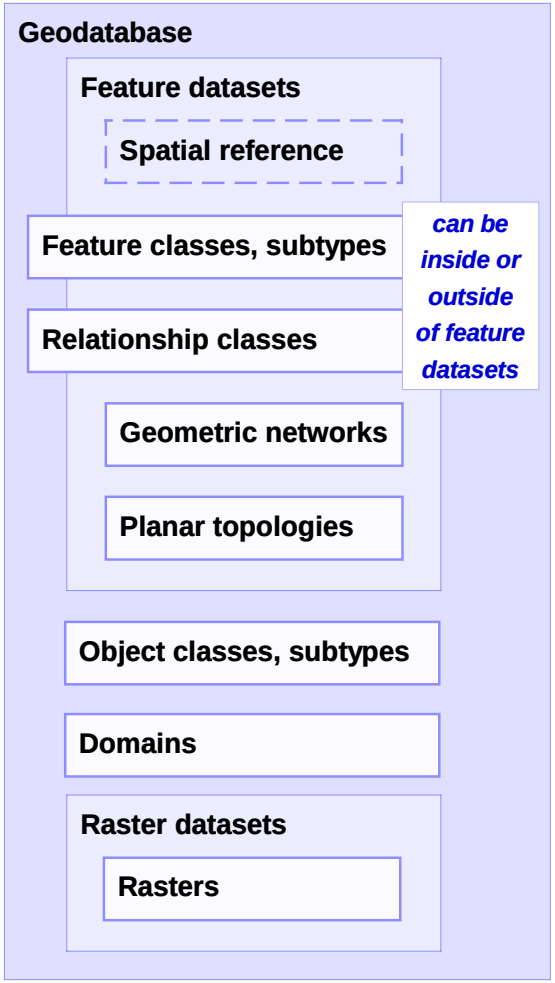
ArcInfo 8 Geodatabase

- A new object-oriented geographic data model
- All relational data storage using ArcSDE
- Versioning and long transactions
- New data access objects for application software developers
- Component based technology for developing custom objects and features

New Features at 8.1

- Dimension features
- Enhanced support for custom features in the editor
- Dynamic segmentation
- Direct import/export of geodatabase data
- New connectivity rule
- CASE tools enhancements
- Performance enhancements

Geodatabase Elements



- Objects, object classes
- Features, feature classes
- Relationships, relationship classes
- Geometric networks
- Feature datasets
- Validation rules, domains
- Spatial references
- Rasters and other dataset types in the future

Objects

A table stores an ObjectClass

A row stores an Object

OWNER

OID	Name	Address	. . .
518	Bob	38 Oak St.	

- **Objects: entities with properties and behavior**
- **An object is an instance of an object class**
- **All objects in an object class have the same properties and behavior**
- **An object can be related to other objects via relationships**

Features

FeatureClass (table)

PARCEL

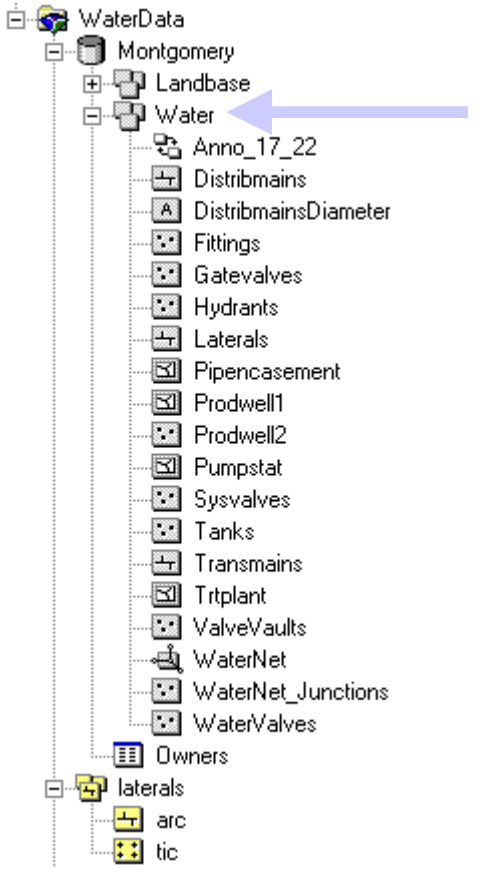
OID	Shape	Type	. . .
524	X,Y,Z,M, ...	Private	. . .

Feature (row)

- A feature is a spatial object
- Features have location
  - a spatial attribute of type geometry
- Features can participate in network and topological relationships
- A feature class is an object class that stores spatial objects (features)
- All features in a feature class have the same spatial reference

Feature Datasets

- Container for feature classes
  - shared spatial reference
- Analogous to a coverage
  - less restrictive
- May also contain
  - relationship classes
  - geometric networks



Validation Rules

- Store attribute, connectivity and spatial rules on objects as part of the geodatabase
- Pre-defined, parameter driven:
  - attribute range rule
  - attribute set rule
  - connectivity rule
- Perform custom validation by writing code

Domains

- Describe the legal values of a field type
  - used to ensure attribute integrity
- Can be shared among classes
- Uniquely named
- Types of domains
  - range
    - a tree can have a height between 0 and 300 feet
    - a road can have between 1 and 8 lanes
  - coded value (e.g., a set)
    - a tree can be of type oak, redwood, or palm
    - a road can be made of dirt, asphalt, or concrete

Subtypes

- **Partition the objects in an object class into like groups**
- **Defined by the value of a subtype code field**
- **All subtypes:**
  - **have the same attribute schema**
  - **have the same behavior schema**
  - **can have different default values and domains for each field**

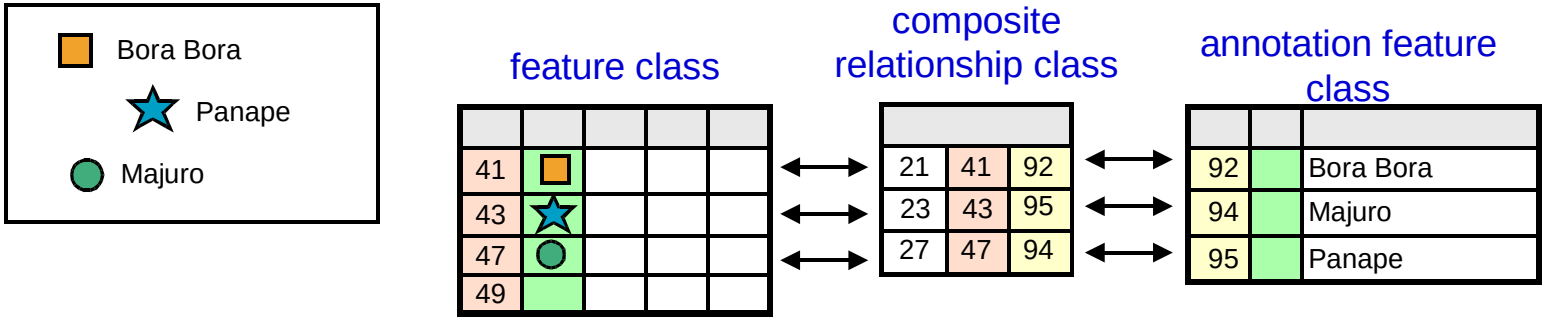
fid	geom	subtype	width	lanes	name
101		asphalt	85.3	4	Chimayo Highway
102		concrete	45.1	2	Acequia de Isabel
103		asphalt	75.9	4	Calle Petra
104		gravel	35.2	2	Maximilian Road

Relationship Classes

- A relationship class is an association between two object classes
- Relationship classes may be 1:1, 1:n, n:m
- An object class may participate in multiple relationship classes
- Related objects can message each other
  - origin to destination, destination to origin, both, neither
  - can trigger behavior (cascade delete, move to follow, custom…)

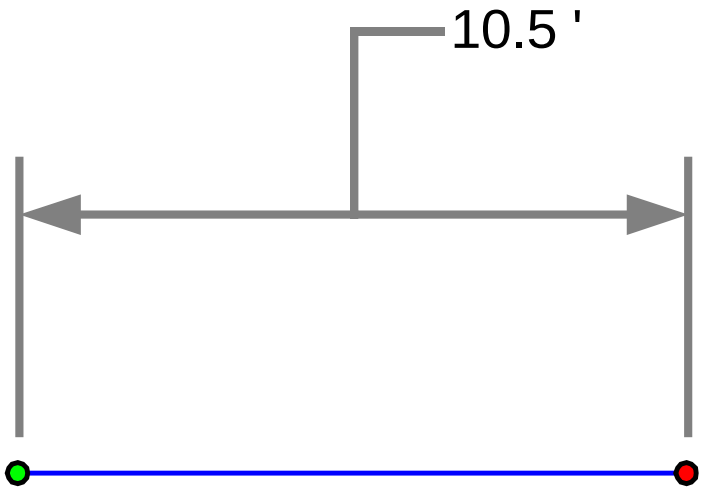
Annotation

- An example of a graphic feature class
- Annotation feature classes may be
  - feature-linked
  - non feature-linked
- Composite relationship manages link
- Can store text as well as other graphics
  - lines, arrows, boxes, etc.



Dimension Features

- Type of annotation that displays specific distances on a map
- Stored in a dimension feature class
- Graphic feature
- “Smart” feature
  - special drawing
  - special editing



Geometric Networks

- Used to model network systems
- Topological relationship between feature classes
- Each feature class has a topological role in the network (i.e., junction or edge)
- A network may have multiple feature classes in the same topological role
- Topology based upon geometric coincidence, always live
- Feature classes must be in the same feature dataset

Network Feature Classes

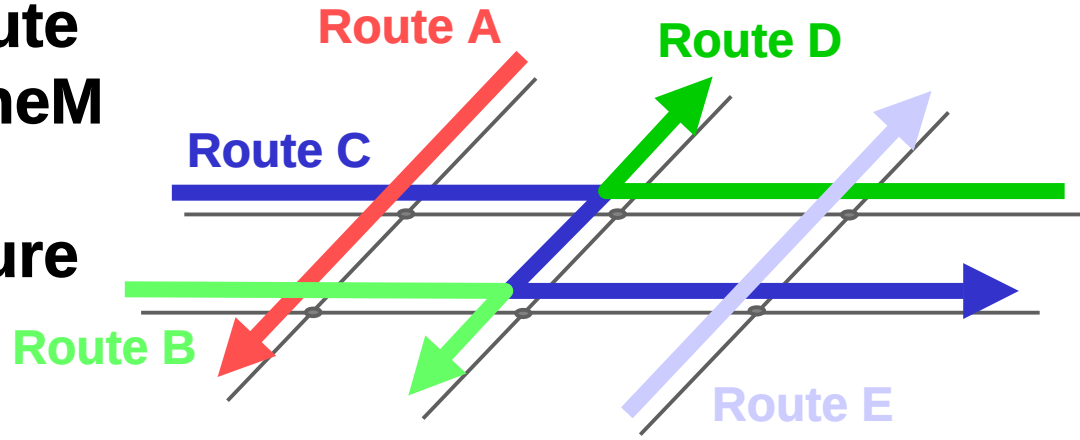
- Network features live in a geometric network
- Directly support network analysis
- Types:
  - simple junction
  - simple edge
  - complex junction
  - complex edge
- Integrity constraint:
  - edge must have a junction at each endpoint

Connectivity Rules

- Help you maintain a valid network
- Constrain permissible connectivity
  - default GN behavior allows any edge to connect to any junction
- Connectivity rules include:
  - edge-junction rules
    - cardinality
  - edge-edge rules
    - permissible junction types
    - default junction type

Dynamic Segmentation

- True dynamic segmentation (DynSeg)
  - display table or route events as layer in Map
  - interactively find a location along a route
- Event tables can be INFO, DBASE, Geodatabase, or OLE DB
- Route data can be coverage route system, PolyLineM Shapefile, or PolyLineM feature class

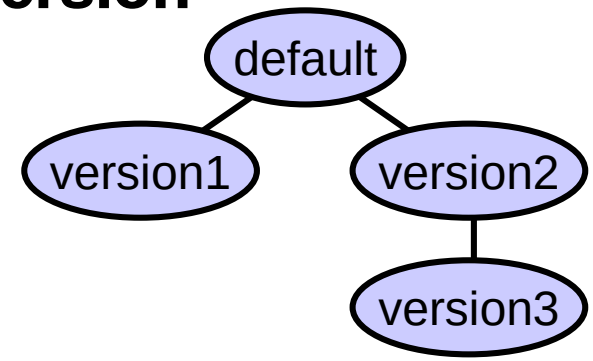


Planar Topology

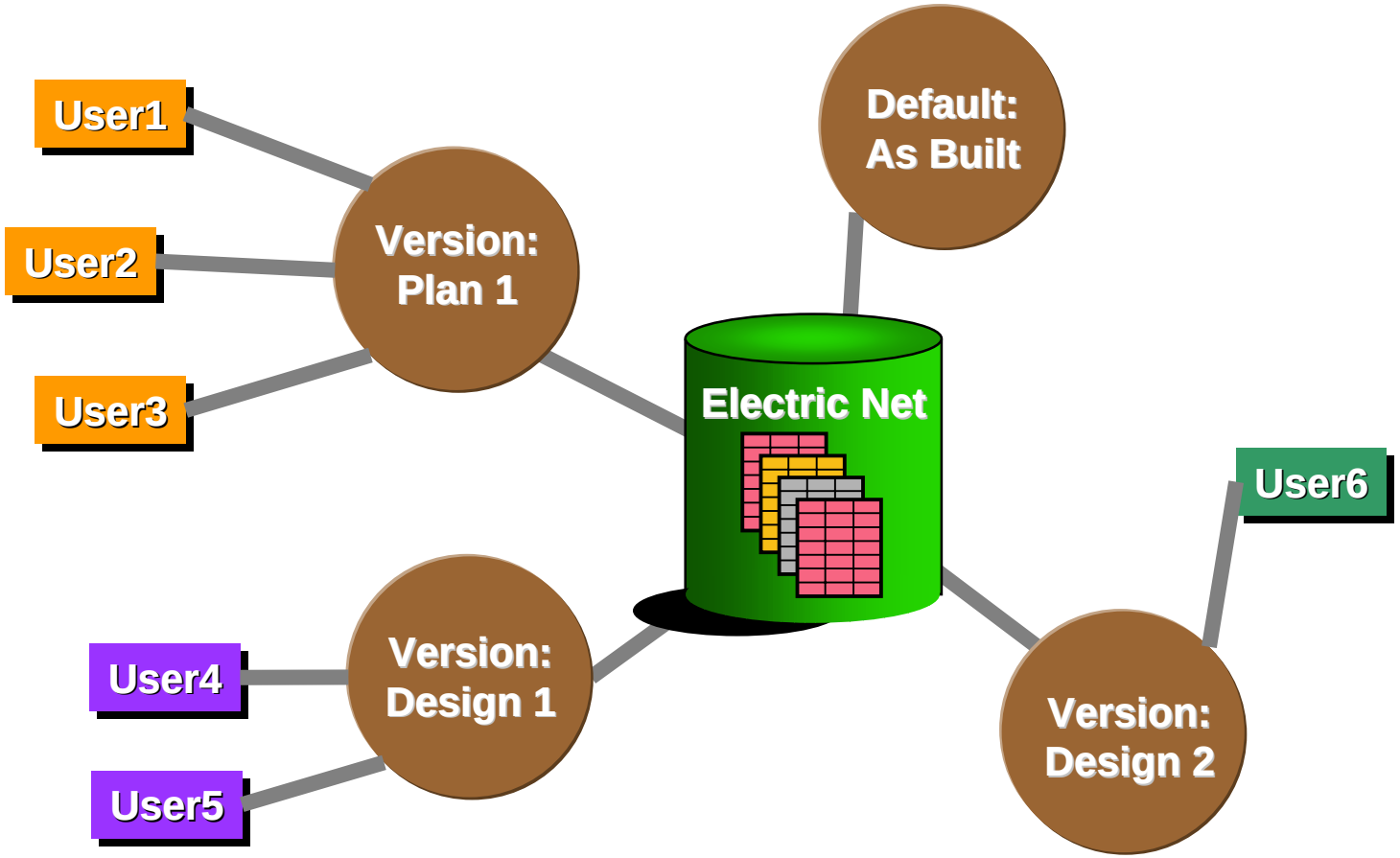
- Feature classes in an integrated feature dataset participate in a planar topology
- Features share boundaries
- Editor tools allow you to edit and maintain shared boundaries
- Use the *Integrate* command in the Editor to ensure coincident boundaries
- Use shared edge edit tool to edit shared boundaries and maintain topological relationships

Versions

- Object classes, feature classes, relationship classes, geometric and logical networks may all be versioned
- A version spans all multi-versioned objects in the database
- Schema is constant across all versions
- Versions differ only in those features or rows or elements modified in each version
- A user can connect to and work with any version of the database - majority will work with the Default version

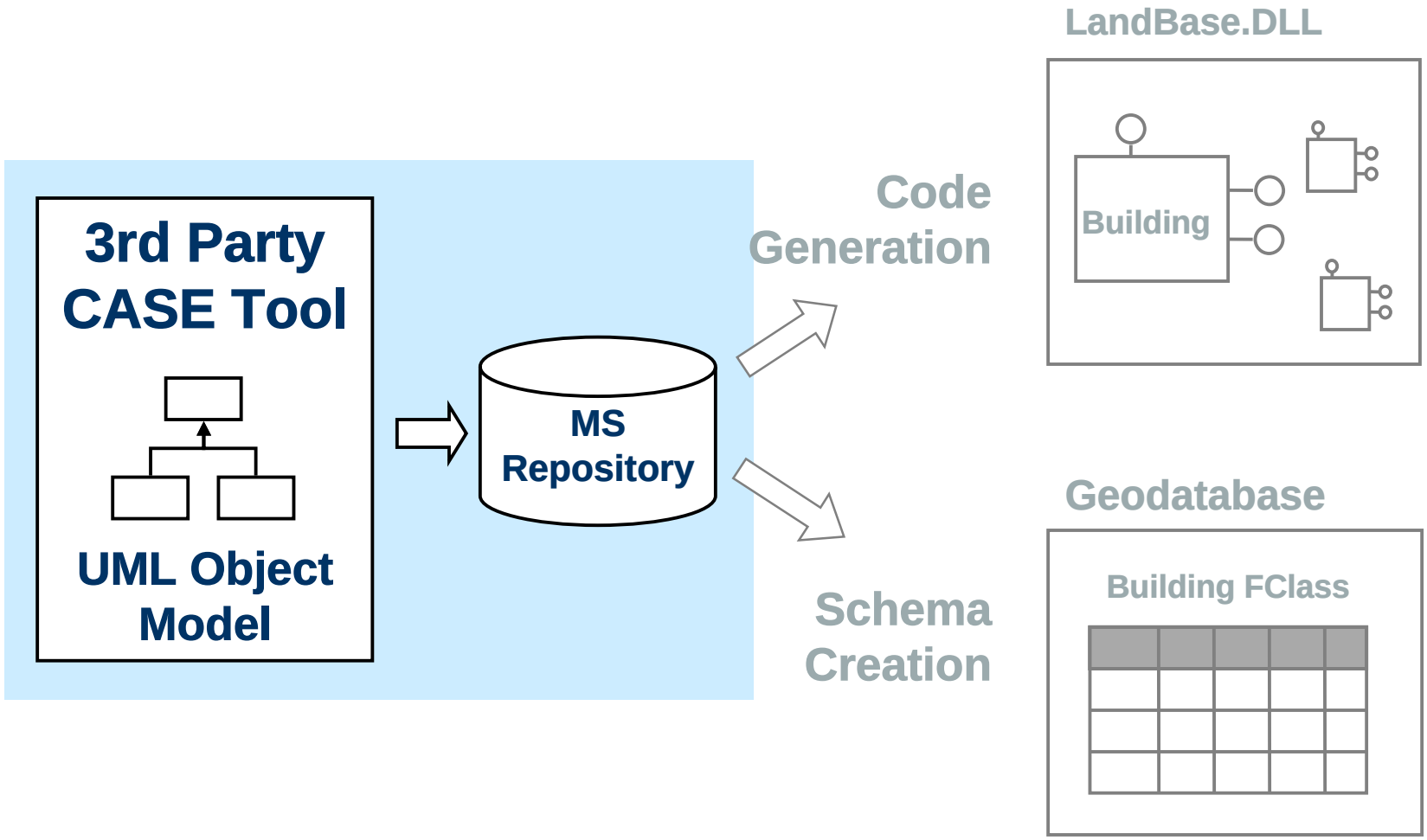


Multi-Versioned Database



Representing the Geodatabase Using UML

Data Modeling Process



CASE Tool

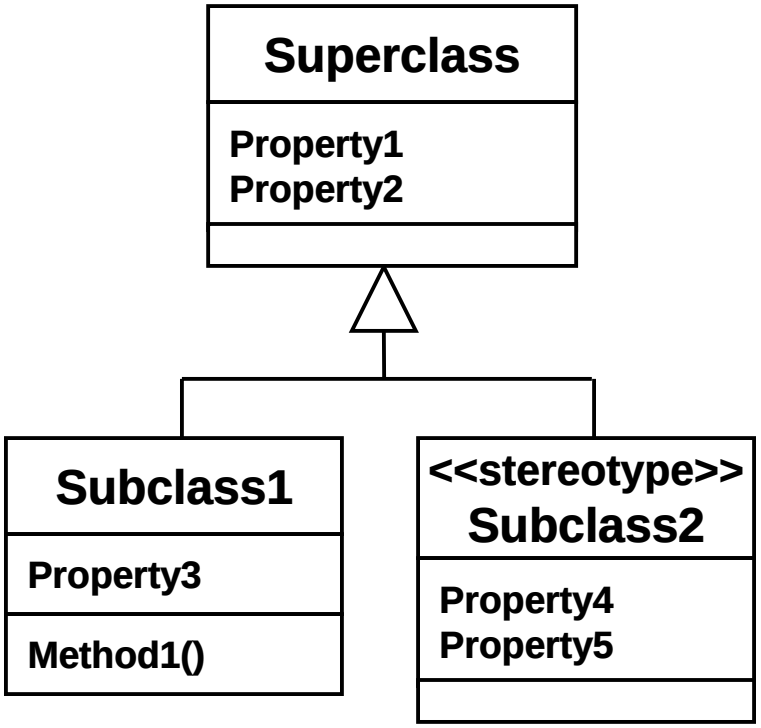
- **Basic sequence**
  - **utilize third party CASE tool**
  - **create data model**
  - **represent the data model using UML within the CASE tool**
  - **add Geodatabase configuration components**
  - **perform schema check(s)**
  - **export finished model to Repository**

UML Review

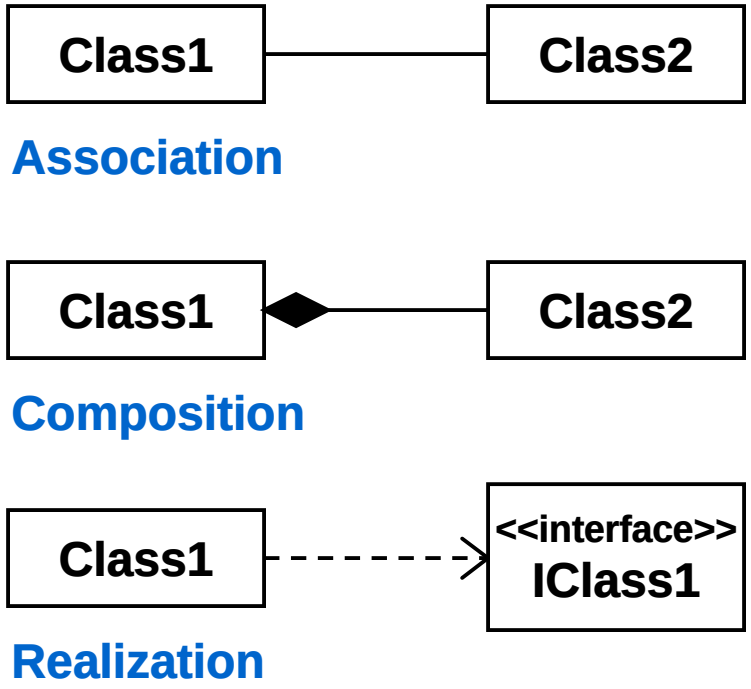
- **Unified Modeling Language**
  - **lingua franca of object modeling**
- **Developed in 1997 as a unification of the three leading methodologies**
  - **OMT (Rumbaugh)**
  - **Booch**
  - **Jacobson (use cases)**

UML Syntax

Inheritance

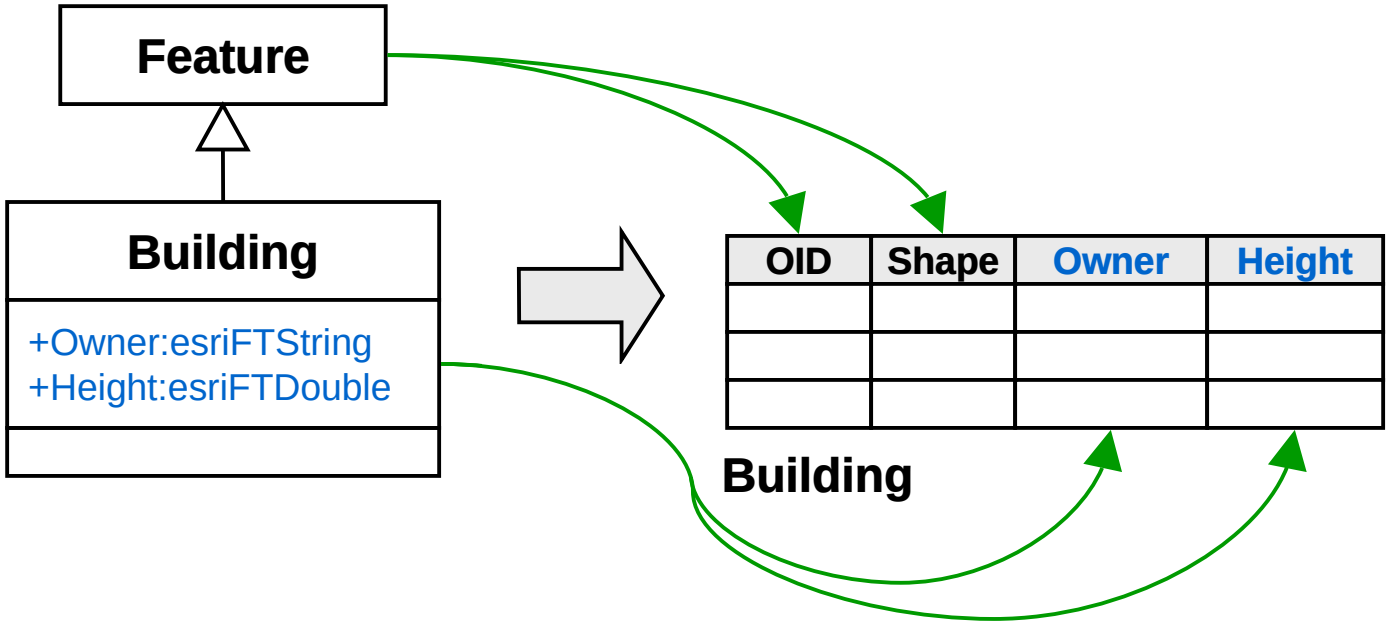


Relationships



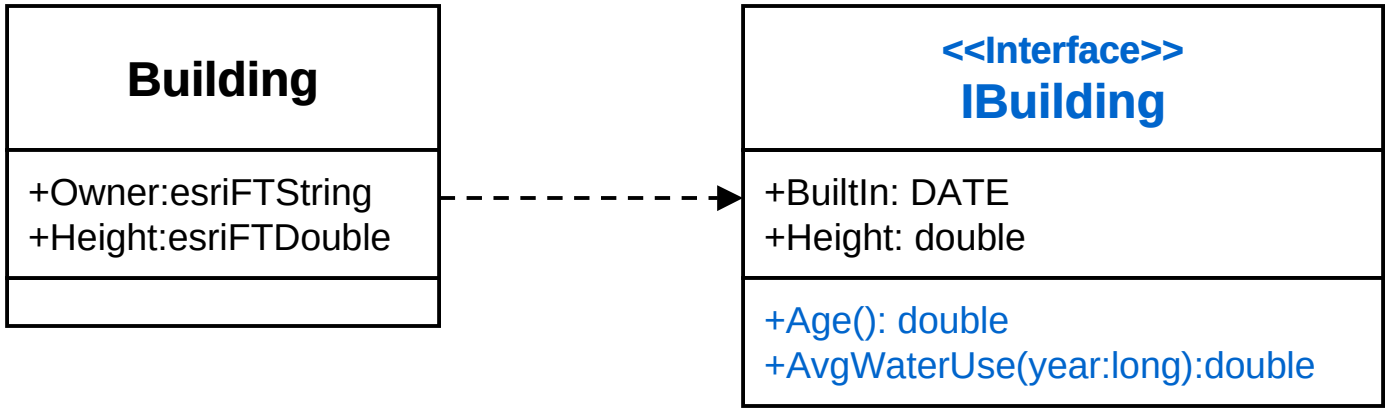
For Arc8, this is 90% of what you need to know with respect to UML

Properties



- **Properties become fields in schema**
- **Model the feature**
  - **feature class will be automatically created in the GDB during schema generation**

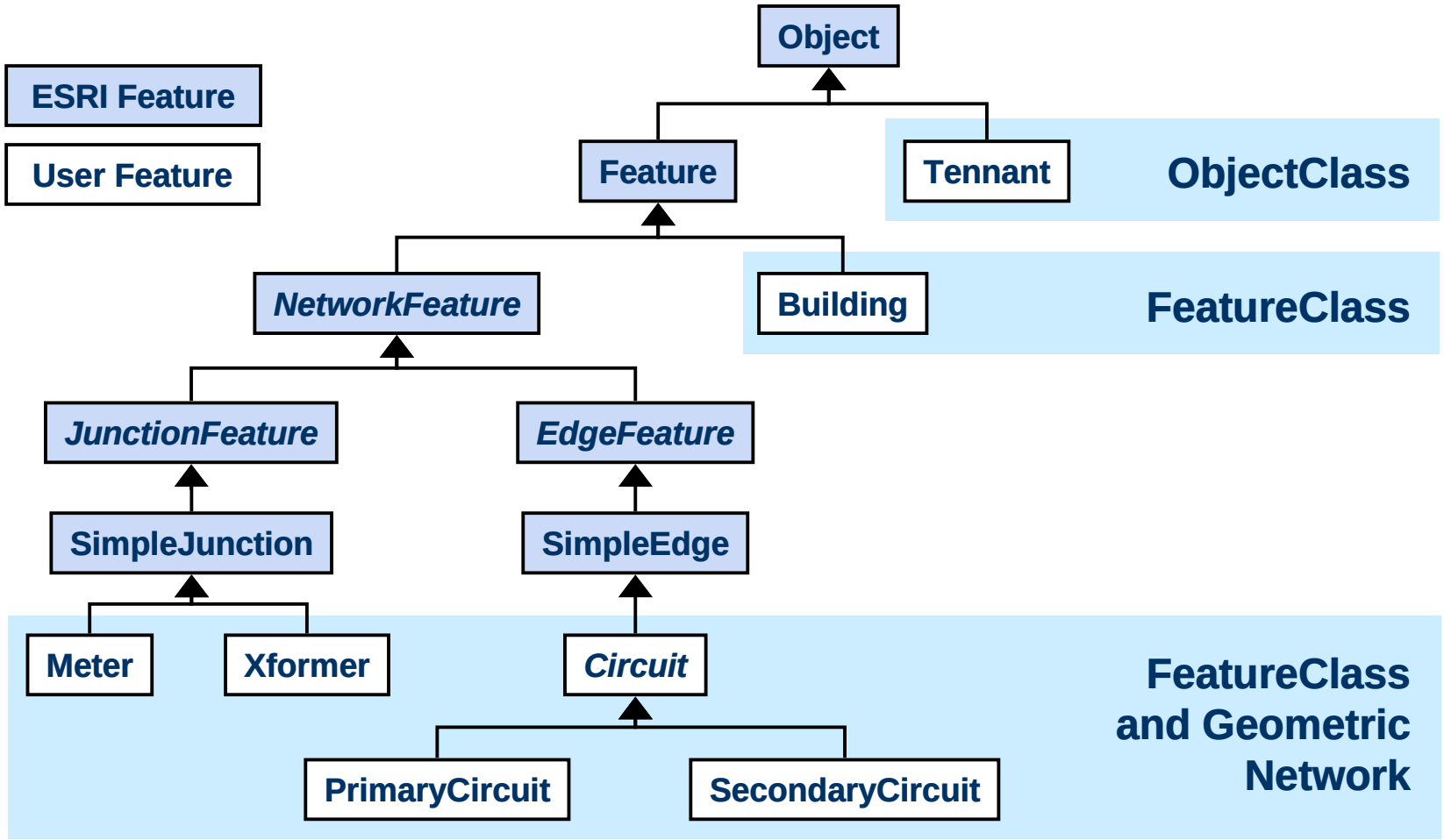
Methods



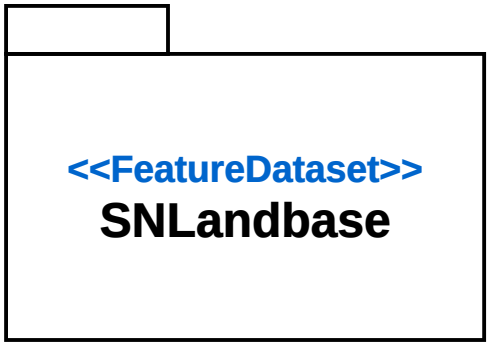
- **Methods always live in the interface**
  - **components are interface-based**
  - **class realizes an interface**

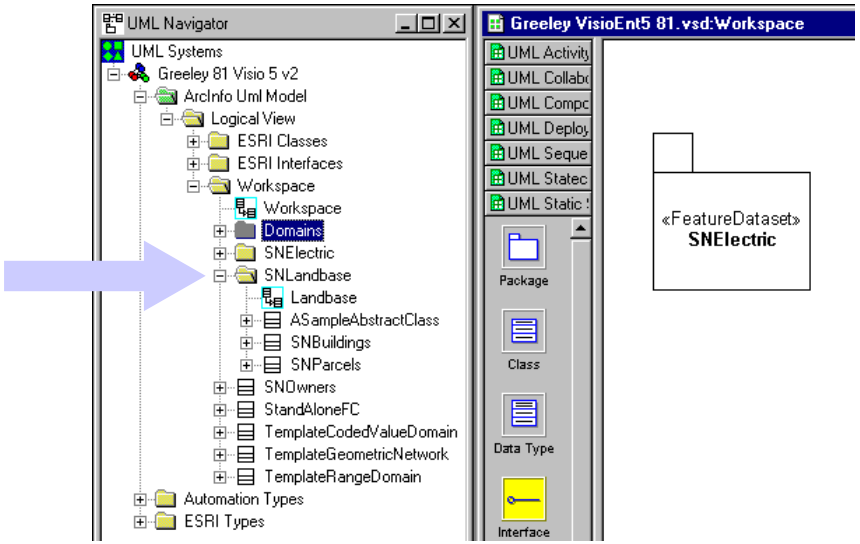
Note: this is necessary only for source code generation

Object Model Sample



Feature Datasets



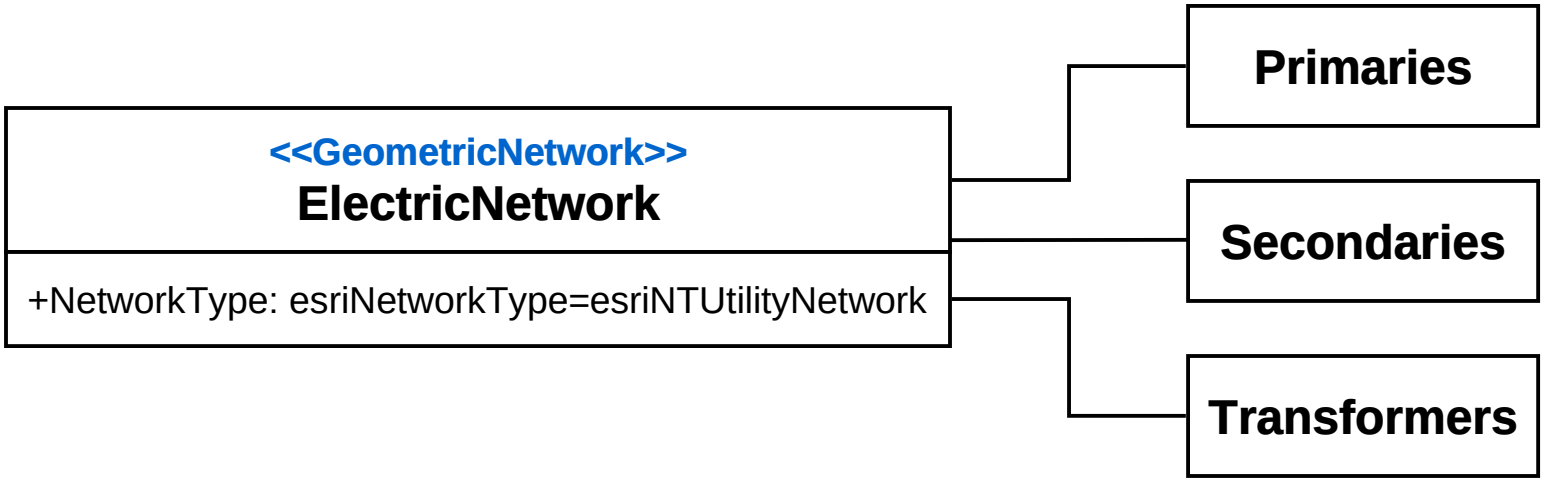


- **New for 8.1**
- **Feature datasets correspond to stereotyped packages in UML**
- **Feature classes and geometric networks added to package (tree view)**

Feature Datasets

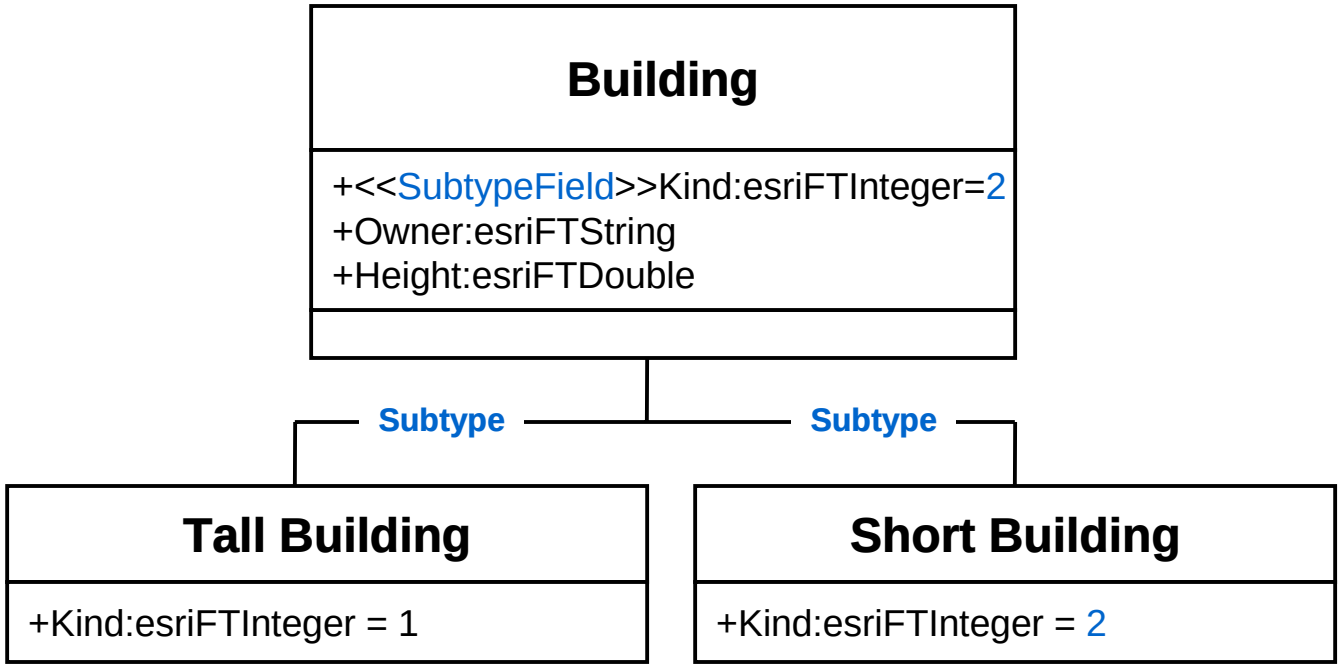
- **Modeling feature datasets as packages enables:**
  - **stand-alone feature classes**
  - **relationship classes between feature classes in different feature datasets**
  - **specification of coordinate systems within the schema generation wizard**
  - **default coordinate systems (last specified)**

Geometric Networks



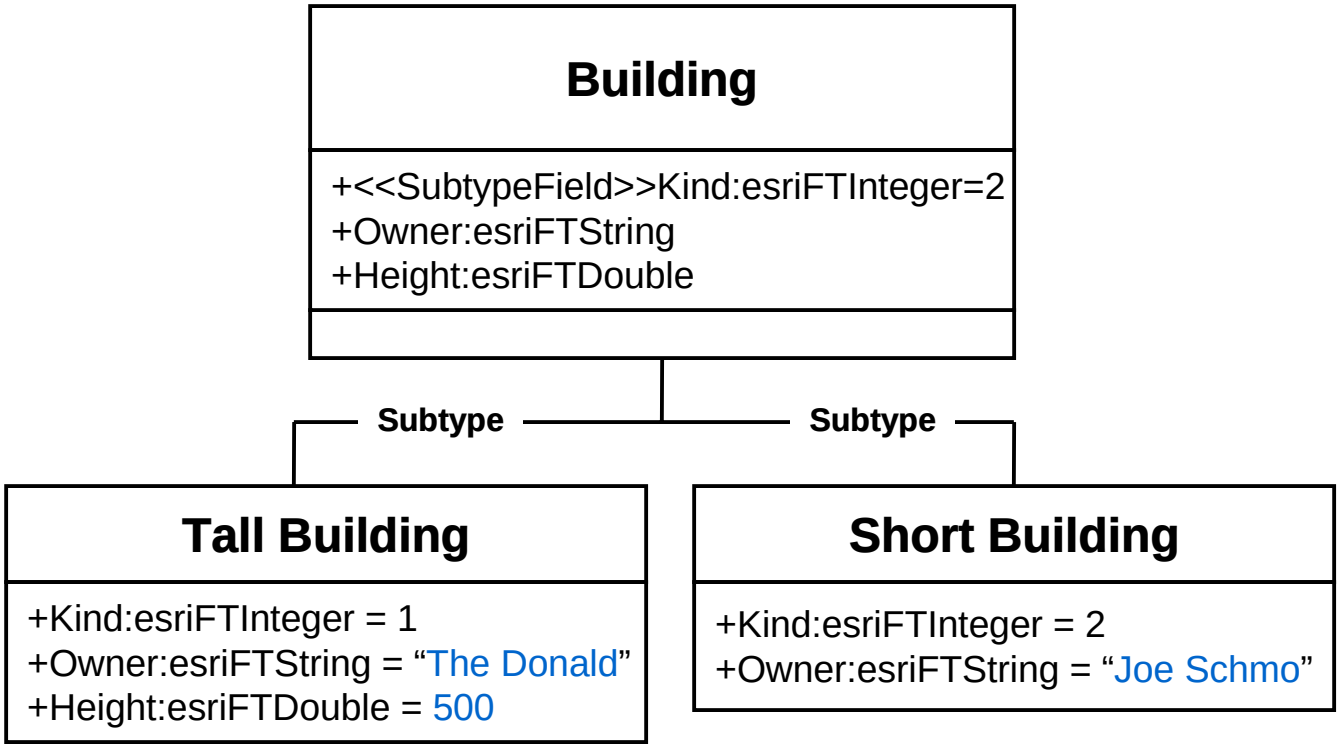
- **New for 8.1**
- **Modeled as a stereotyped class associated with all feature classes in the network**

Subtypes



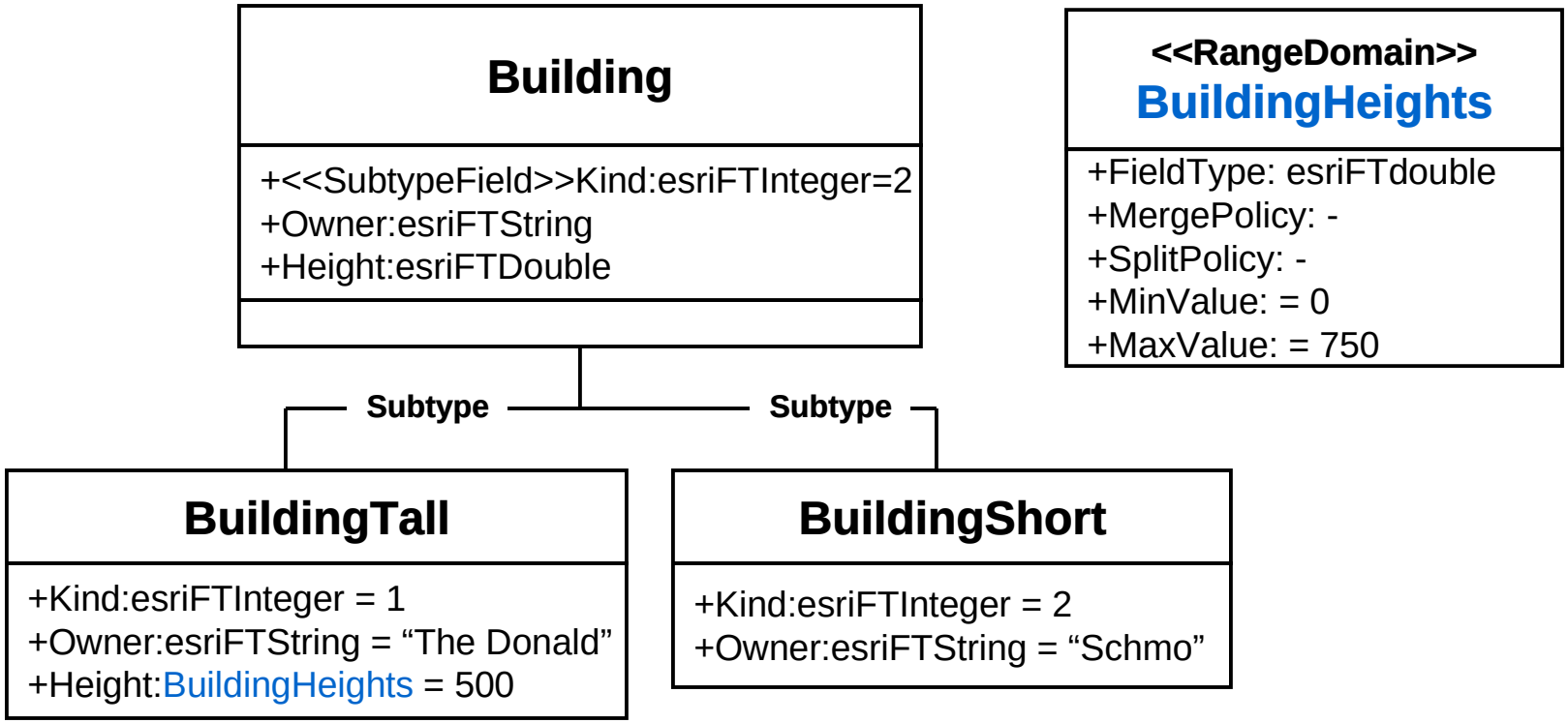
- **Subtypes based on single integer field**
- **UML Association named "Subtype"**
- **Default subtype**

Default Values



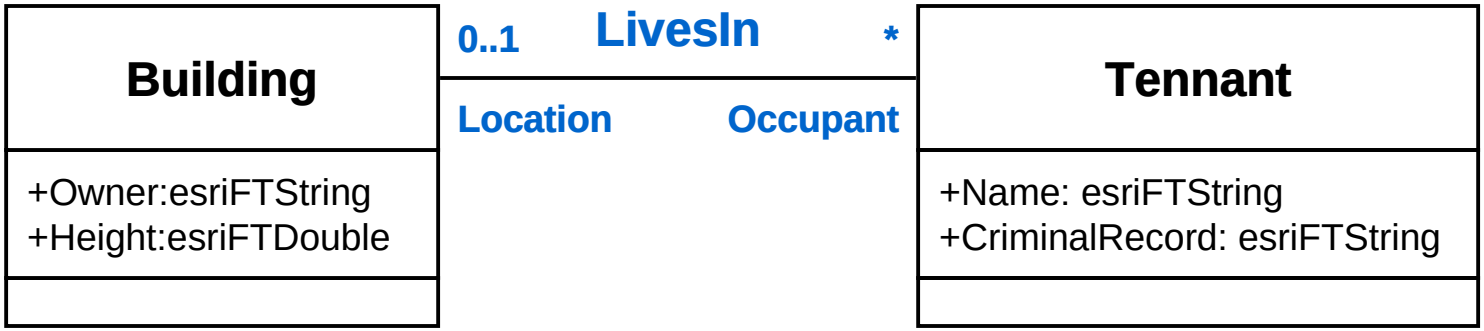
- **Assigned on a subtype basis**

Domains



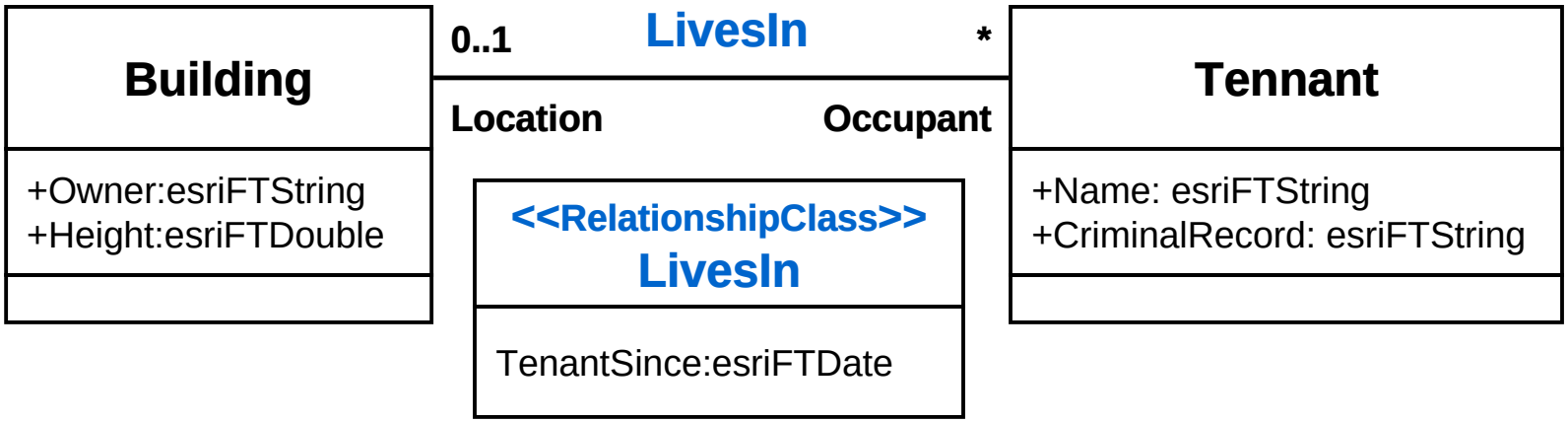
- **Stereotyped class**
- **Side effect creates an attribute rule**

Relationships



- **Relationships are named**
- **Specified cardinality**
- **Origin and destination**

Attributed Relationships

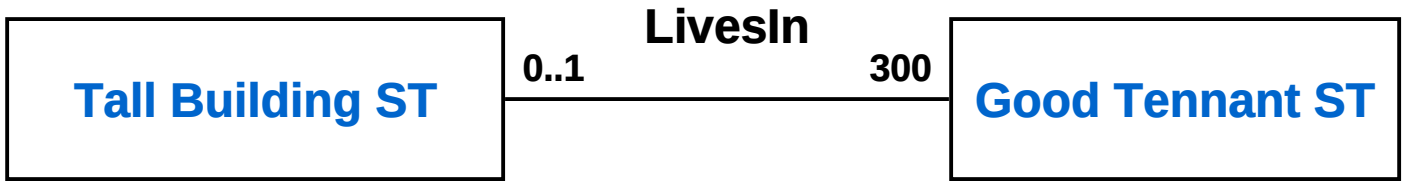


- **A separate table will be created**
- **Not restricted to many to many relationships**
- **Specified as a UML class**
  - **named after the UML association**
  - **stereotyped as <<RelationshipClass>>**

Composite Relationships

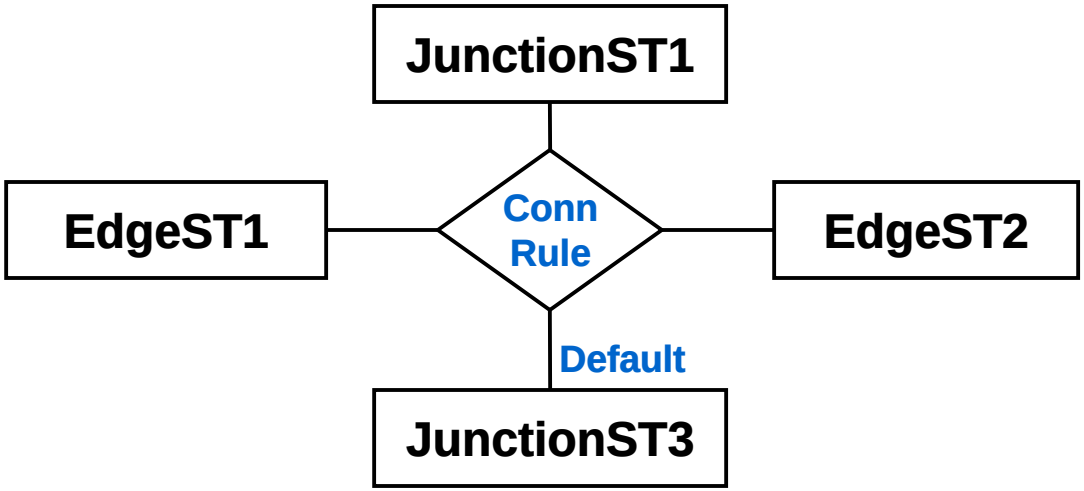
- **Part lifetime controlled by whole class (deep delete semantics)**
- **Always one to many**

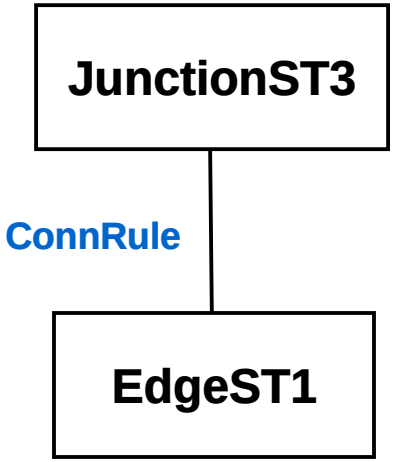
Relationship Rules



- Assigned by subtypes
- Same name as relationship
- More specific cardinality
  - but consistent with relationship

Connectivity Rules

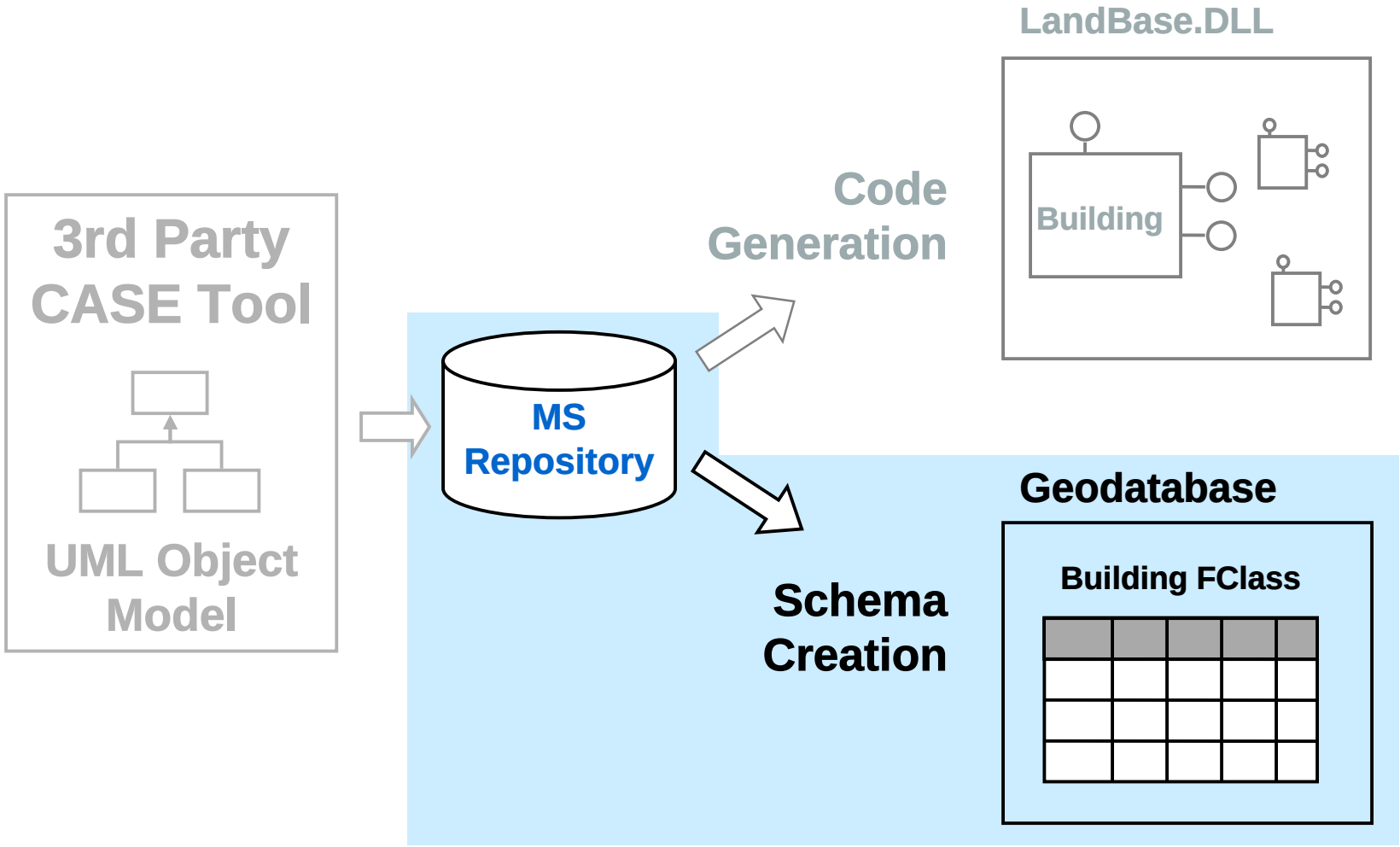




- **Edge connectivity rules**
  - **n-ary UML association**
- **Junction connectivity rules**

Schema Wizard

Data Modeling Process



Schema Wizard

- **Basic sequence**
  - **start wizard from within ArcCatalog**
  - **connect to the Repository, select the model**
  - **select the feature dataset to generate the schema for**
    - **all domains are created at this time**
    - **relationship classes only created if feature class is also being created**
  - **define schema properties for each feature**
  - **generate schema when closing wizard**

Semantics Checker

- **New for 8.1**
- **Check a model exported to the Repository**
  - **reports ALL errors at once**
  - **shortens modeling cycle**
- **Add-on that runs inside Visio**
- **Should be run before the schema or code generation wizards**

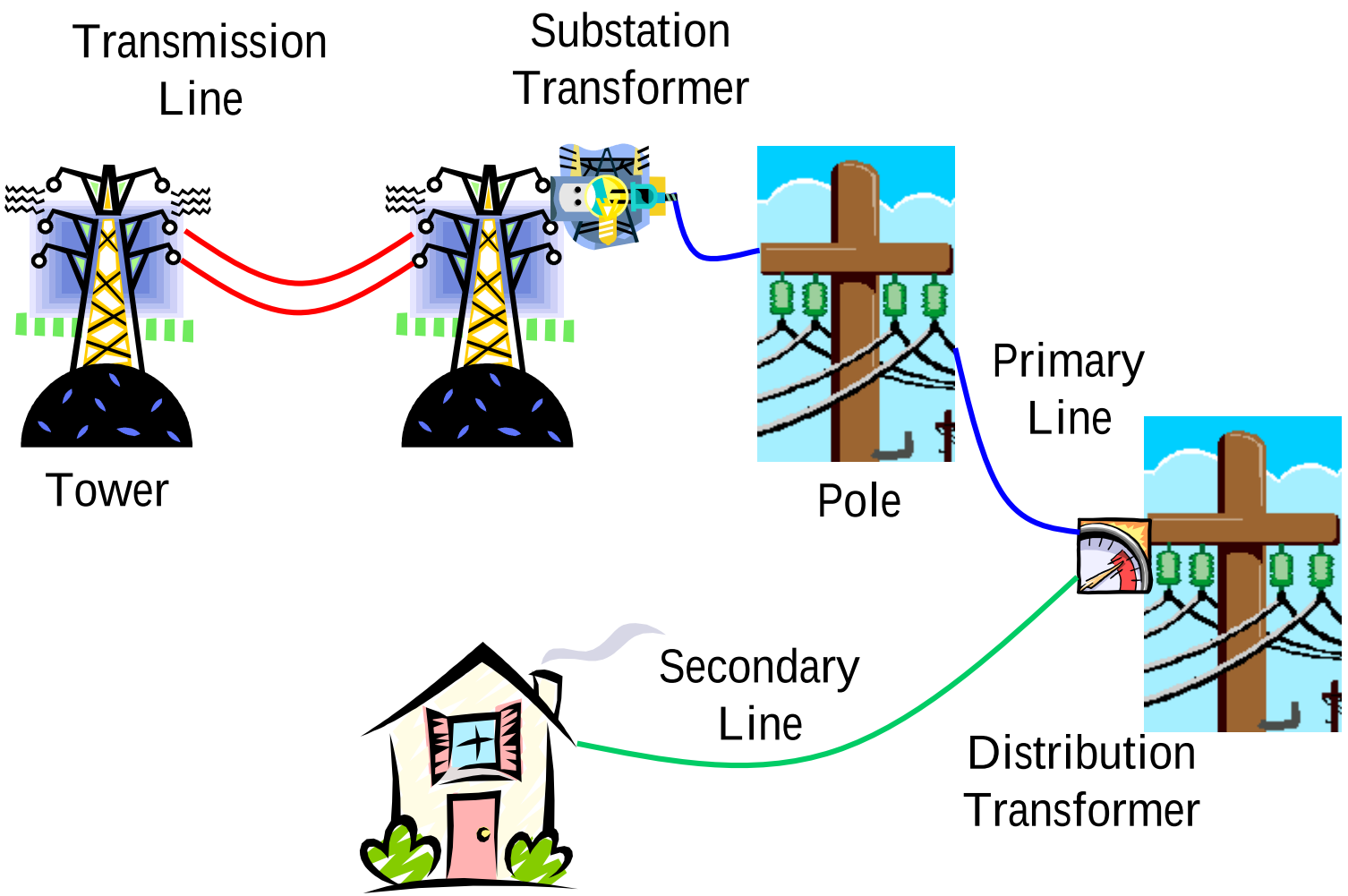
Startling Demo

Background Info: Orphan Junctions

- Simple junction feature
- Automatically added when first feature class added to network

  `<networkName>_Junctions`

- Integrity constraint:
  - edge must have a junction at each endpoint



Electrical Demo…

Conclusions

- Time spent data modeling is very beneficial in the long run
- Pay attention to performance issues
- Use ArcCatalog for tactical control of simpler systems
- Use CASE (UML and schema wizard) for modeling complex systems
- Both tools will simplify your life

For Further Info

For Further Info

- **Relevant UC sessions:**
  - ***Overview of the Geodatabase***
  - ***Designing and Using a Geodatabase***
  - ***Working with a Versioned Geodatabase***
  - ***Extending the Geodatabase with Custom Objects***
  - ***Extending the Geodatabase with Class Extensions***
  - ***Advanced Customization with ArcObjects in C++***
  - ***Managing and Editing Geometric Networks***
  - ***Working with Networks in ArcInfo 8***

For Further Info

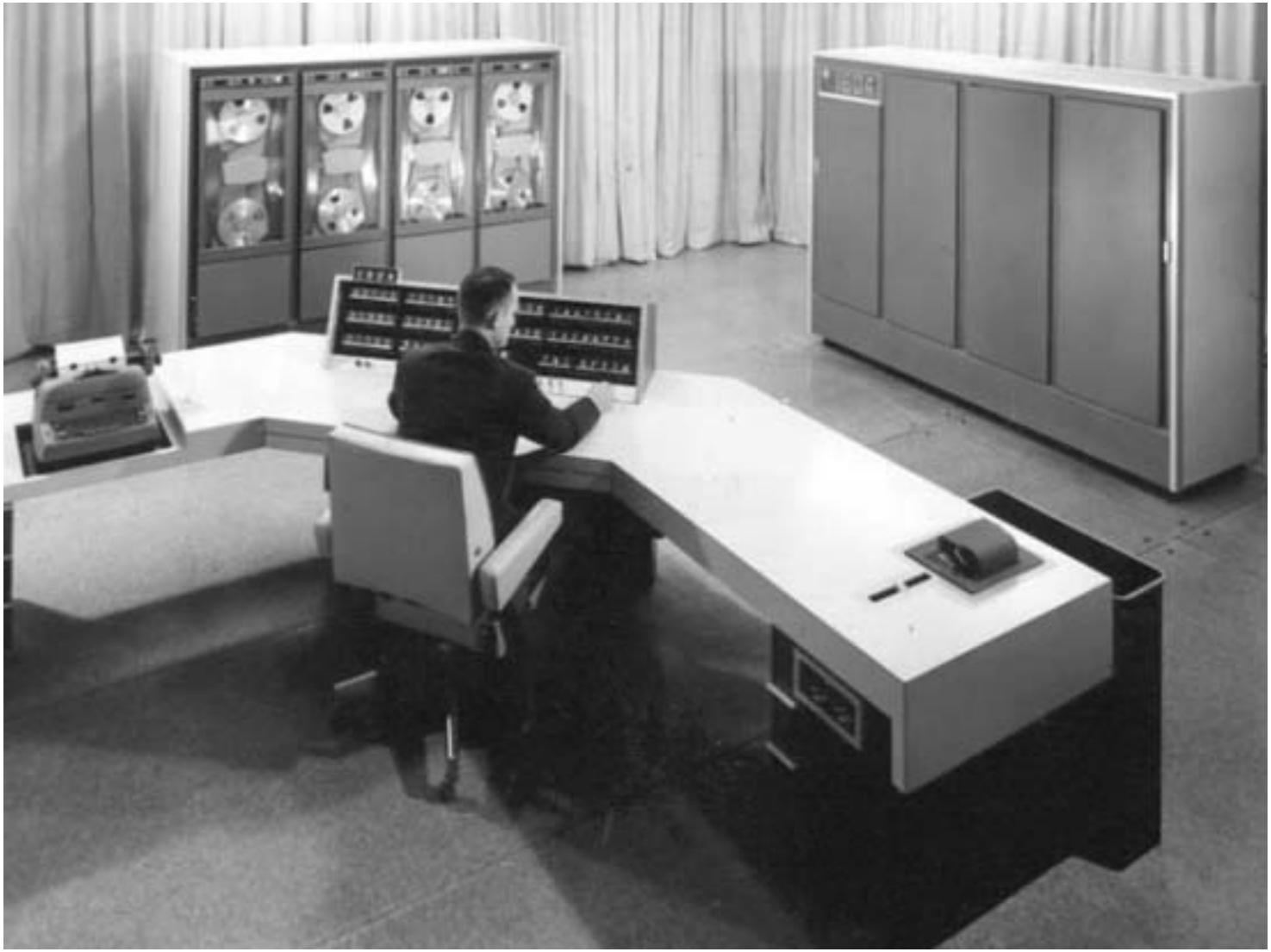
- **Geodatabase Literature**
  - **Michael Zeiler. *Modeling Our World: the ESRI Guide to Geodatabase Design*. ESRI Press, 1999.**
  - **Andy MacDonald. *Building a Geodatabase*. ESRI Press, 1999.**
  - ***Multi-user GIS Systems with ArcInfo 8*. ArcOnline White Paper, March 2000.**
  - **Erik Hoel, Julio Andrade, and Sudhakar Menon. *Modeling GIS Databases Using UML*. Submitted to the 8th International Symposium of ACM GIS.**

For Further Info

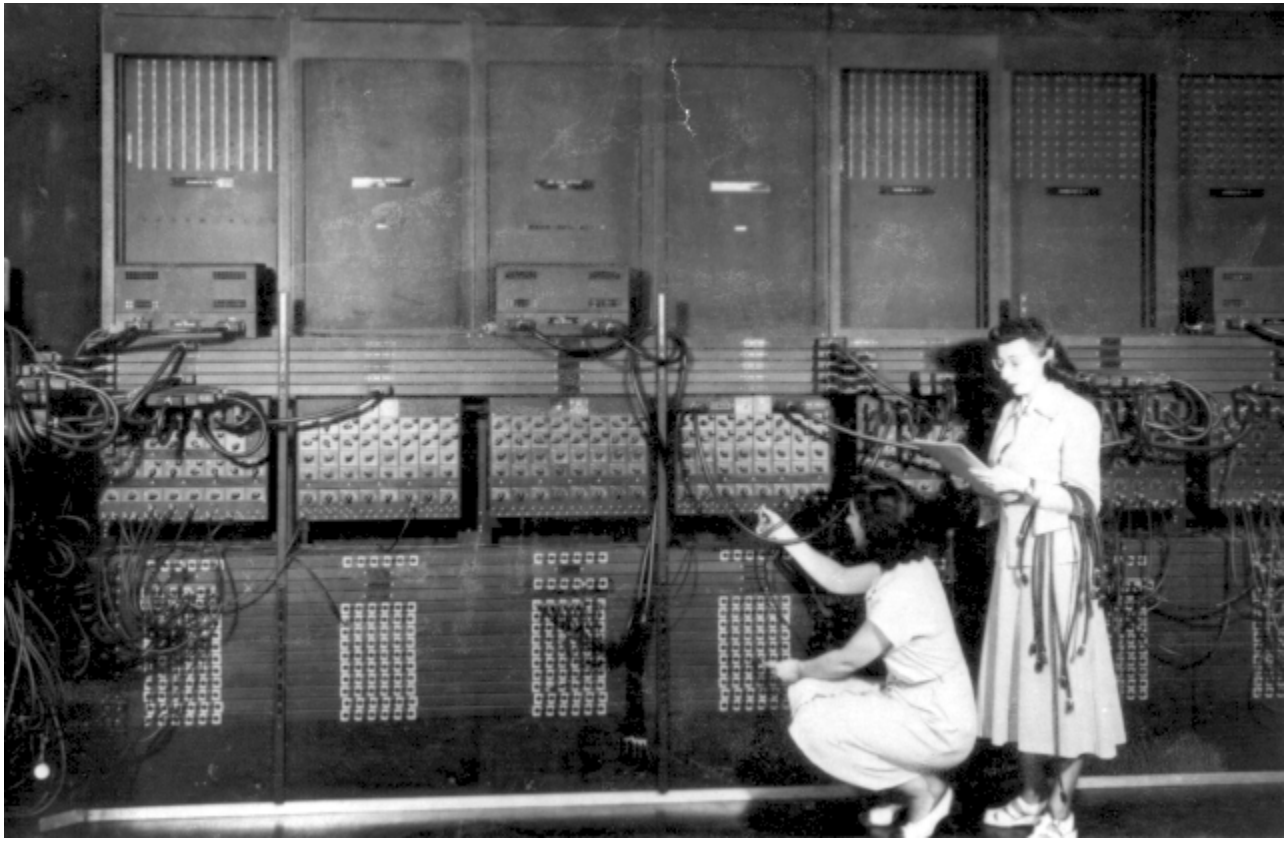
- **General Literature**
  - **David Taylor. *Object Technology: A Manager's Guide*. 2nd Ed., Addison-Wesley, 1997.**
  - **Martin Fowler, et.al. *UML Distilled: Applying the Standard Object Modeling Language*. Addison-Wesley, 1997.**
  - **Bertrand Meyer. *Object-Oriented Software Construction*. Prentice Hall, 1997.**


geography
esri
gis
our global network

Representing the GeoDatabase using UML



Startling CASE Tool Demo



Hi-Tech Schema Wizard Demo