

Conceptual and Logical Modeling

GIS Applications
Spring 2009

Conceptual model

A formal model in which every entity being modeled in the real world has a corresponding object in the model.

Describes the semantics of some phenomena and represents a series of assertions about its nature.

Describes the things of significance to an organization (*entity classes*), about which it is inclined to collect information, and characteristics of (*attributes*) and associations between pairs of those things of significance (*relationships*).

Logical model

The **logical** model organizes data in terms of a particular data management technology.

Given current predominance of relational databases, logical model generally conforms to relational theory - contains **fully normalized entities**.

For a logical data model to be normalized, it must include the **full population of attributes** and those attributes must be defined in terms of their domains or **logical data types** (e.g., character, number, date, pict)

Conceptual modeling

- A formal analysis and design method that uses a set of guidelines and rules to capture the semantics of a domain
- A conceptual model is built from a limited but well defined set of constructs
- Constructs, along with a notation, and a small set of rules constitute a formal language or formalism.
- Formal methods include textual or graphical notations to create, present, validate, and manipulate data models
- Clarifies identification of entities, attributes, and associations
- Provide a basis for discussion and refinement

Creating conceptual models

- Early on formal models were made manually
- Computer Aided Software Engineering (CASE) tools have helped to automate this process
- CASE tools provide graphic constructs, drawing and editing tools

Basic Components

Entities, Attributes, Associations

Entities: principal data objects about which information is collected - usually person, place, thing, event

A particular occurrence of an entity is called an **entity instance**

Attributes: characteristics of entities or relationships that provide descriptive details about them

Associations: represent real world associations among one or more entities

OO approaches and UML

UML – Unified Modeling language

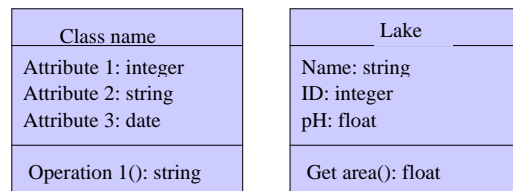
- models objects, encompassing properties (attributes)
- models relations between objects
- models aggregation of objects into more complex objects
- models generalization or specialization of the types of objects to more general or more specific types

<http://www-306.ibm.com/software/rational/uml/>

Representation of a class

An entity or class is represented as a rectangle containing three compartments stacked vertically

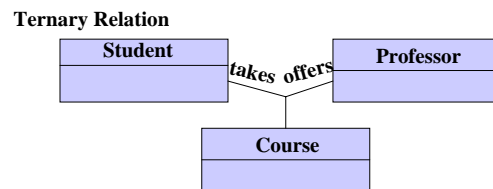
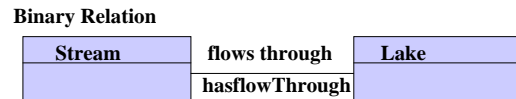
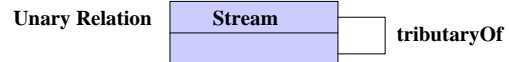
- The top compartment shows the class name
- The middle compartment lists the class's attributes
- The bottom compartment lists the class's operations or methods



Associations

- **Associations** are described in terms of degree, multiplicity and role
- The **degree** of a relationship is the number of entities associated in the relationship
- Unary, binary, ternary are special cases where the degree is 1, 2 or 3.
- An n-ary relationship is the general form for any degree n
- **Binary** relationships are the most common
- A **ternary** relationship relates 3 entities to each other in such a way that it cannot be decomposed into equivalent binary relationships

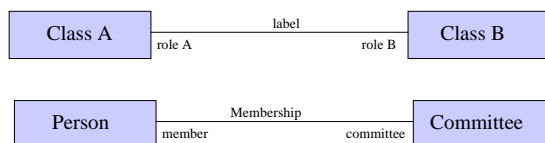
Associations



Associations

Label describes the association, typically consists of a noun or verb followed by a preposition

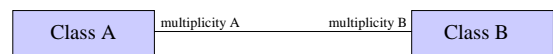
Role indicates a role played by the class in the association. The UML notation expresses the role using a noun on each side of the association.



Associations

Multiplicity describes the constraint on the number of entity instances that are related through an association

- It is important to determine the multiplicity on each side of a relationship individually



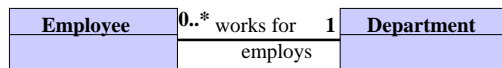
Indicator	Meaning
0..1	Zero or one
1	One only
0..*	Zero or more
1..*	One or more
n	Only n (where n > 1)
0..n	Zero to n (where n > 1)
1..n	One to n (where n > 1)

Associations

Multiplicity determined from semantic rules

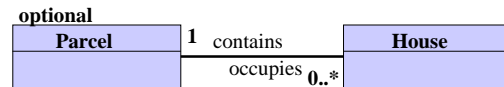
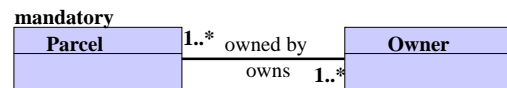
Each department may be the employer of 0 or more employees

Each employee may work for at most one department



Associations

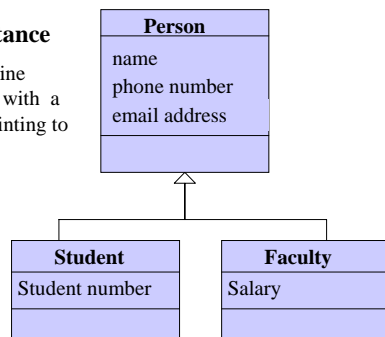
- Some entities existence depends on the existence of another entity – **existence dependency or just existence.**
- Existence is defined as either **mandatory** or **optional**
- Optional existence is indicated by a zero multiplicity



Associations

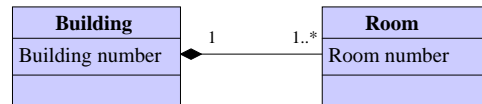
Modeling Inheritance

Indicated by a solid line drawn from subclass with a closed arrowhead pointing to the super class

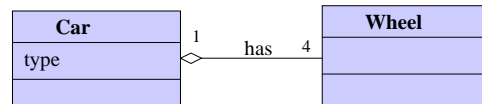


Associations

Composition: indicates that one class is part of another class, child instance lifespan is dependent on the parent, indicated by a solid line and filled diamond on the parent class



Aggregation: The child class can exist independently of the parent class. Indicated by a solid line with an unfilled diamond on the parent class



Example

Entities

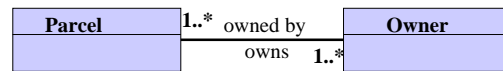
Attributes

parcel	ID, assessed value, size
owner	Name, billing address, phone
building	ID, assessed value, size, construction date
permit	Number, type, date issued
zone	Type, permitted uses

Example

Parcel – owner association

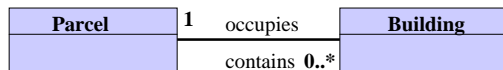
- A parcel must have at least one owner
- A parcel can have more than one owner
- Owner can own one or more parcels



Example

Parcel – building association

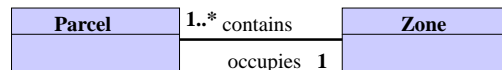
- A parcel may have zero or more buildings
- A building can occupy only one parcel



Example

Parcel –zone association

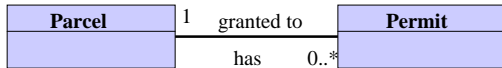
- A parcel belongs to only one zone
- A zone contains one to many parcels



Example

Parcel-permit association

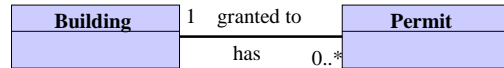
- A parcel can have zero to many permits
- An instance of a permit is associated with only one parcel



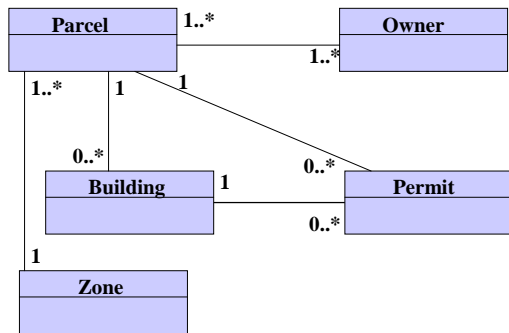
Example

Building-permit association

- A building can have zero to many permits
- An instance of a permit is associated with only one building

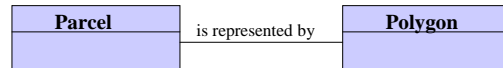


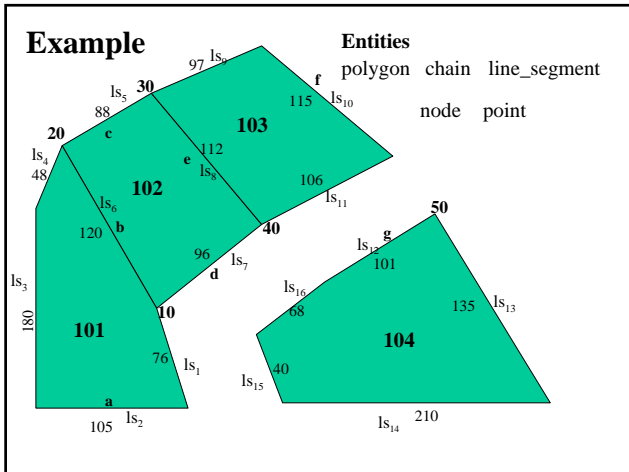
Example



Example

Spatial databases need to consider entities and their spatial representations





Spatial Object Definitions - SDTS

Zero-Dimensional Objects

2.3.1.1 Point (NP).
 A zero-dimensional object that specifies geometric location. One coordinate pair or triplet specifies the location

2.3.1.2 Node (NO, NN).
 A zero-dimensional object that is a topological junction of two or more links or chains, or an end point of a link or chain.

http://mcmcweb.er.usgs.gov/sdts/SDTS_standard_nov97/part1b10.html

Spatial Object Definitions - SDTS

One-Dimensional Objects

2.3.2.1 Line Segment.
 A direct line between two points.

2.3.2.2 String (LS).
 A connected non-branching sequence of line segments specified as the ordered sequence of points between those line segments. Note: A string may intersect itself or other strings).

2.3.2.3 Arc (AC, AE, AU, AB).
 A locus of points that forms a curve that is defined by a mathematical expression.

http://mcmcweb.er.usgs.gov/sdts/SDTS_standard_nov97/part1b10.html

Spatial Object Definitions - SDTS

One-Dimensional Objects continued

2.3.2.4 Link (LQ)
 A topological connection between two nodes. A link may be directed by ordering its nodes.

2.3.2.5 Chain.
 A directed non-branching sequence of non-intersecting line segments and (or) arcs bounded by nodes, not necessarily distinct, at each end.

2.3.2.5.1 Complete chain (LE).
 A chain that explicitly references left and right polygons and start and end nodes. It is a component of a two-dimensional manifold.

2.3.2.6 Ring.
 A sequence of nonintersecting chains or strings and (or) arcs, with closure. A ring represents a closed boundary, but not the interior area inside the closed boundary.

http://mcmcweb.er.usgs.gov/sdts/SDTS_standard_nov97/part1b10.html

Spatial Object Definitions - SDTS

One-Dimensional Objects continued

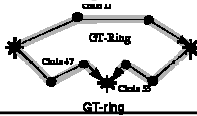
2.3.2.6.1 G-ring (RS, RA, RM).

A ring created from strings and (or) arcs.



2.3.2.6.2 GT-ring (RU).

A ring created from complete and (or) area chains.



Spatial Object Definitions - SDTS

Two-Dimensional Objects

2.3.3.2 G-Polygon (PG).

An area consisting of an interior area, one outer G-ring and zero or more nonintersecting, nonnested inner G-rings. No ring, inner or outer, must be collinear with or intersect any other ring of the same G-polygon

2.3.3.3 GT-Polygon (PR, PC).

An area that is an atomic two-dimensional component of one and only one two-dimensional manifold. The boundary of a GT-polygon may be defined by GT-rings created from its bounding chains. A GT-polygon may also be associated with its chains (either the bounding set, or the complete set) by direct reference to these chains. The complete set of chains associated with a GT-polygon may also be found by examining the polygon references on the chains.

http://mcmweb.er.usgs.gov/sdts/SDTS_standard_nov97/part1b10.html

Example –semantic rules

GT_Polygon–Chain association

- A polygon is bound by one or more chains
- An instance of a chain bounds 1 or 2 GT_polygons

Chain-Node association

- A chain is bound by one to two nodes
- A node bounds 1 or more chains

Line_segment-Chain association

- A chain contains 1 or more line segments
- An instance of a line_segment is part of one and only one chain.

Example – semantic rules

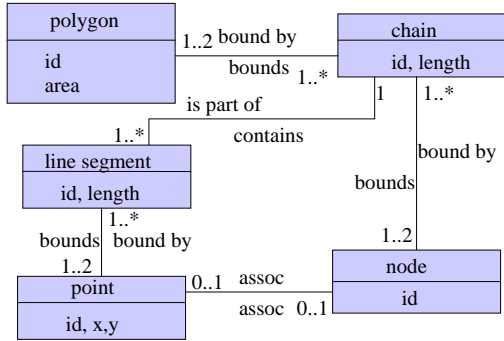
Line_segment-point association

- A line segment is bound by 2 points.
- An instance of a point bounds one or more line segments.

Node-point association

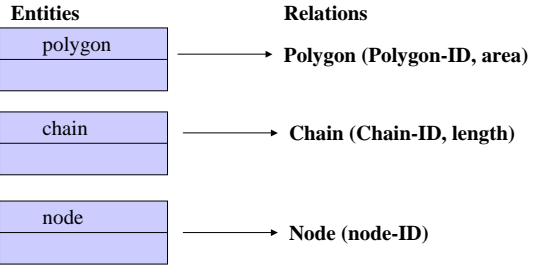
- A node can be associated with one point.
- An instance of a point can be associated with at most one node.

Example – Geometry and Topology



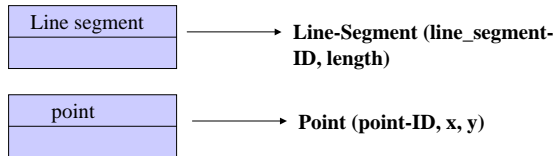
Logical Modeling

Verify model logic, coherence, and completeness of the model
 Convert conceptual model to a specific implementation model
 ie - relational model



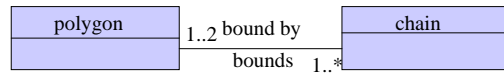
Logical Modeling

Entities converted to relations



Logical Modeling

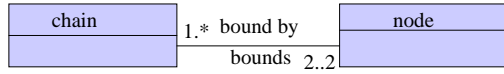
Associations converted to relations



Polygon-chain(polygon-ID, chain-ID)

Polygon-ID	Chain-ID
101	a
101	b
102	b
102	c
102	d
102	e
103	e
103	f
104	g

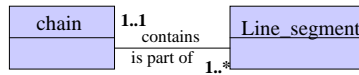
Logical Modeling



Chain-node(chain-ID, from-node-ID, to-node-ID)

Chain-ID	From-Node-ID	To-Node-ID
a	10	20
b	20	10
c	20	30
d	10	40
e	30	40
f	30	40
g	50	50

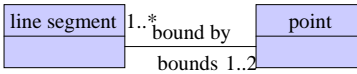
Logical Modeling



Chain-Line_Segment(chain-ID, line_segment-ID)

Chain-ID	Line-Segment ID
a	ls ₁
a	ls ₂
a	ls ₃
a	ls ₄
b	ls ₅
c	ls ₆
d	ls ₇
e	ls ₈
f	ls ₉
f	ls ₁₀
f	ls ₁₁
g	ls ₁₂
g	ls ₁₃
g	ls ₁₄
g	ls ₁₅
g	ls ₁₆

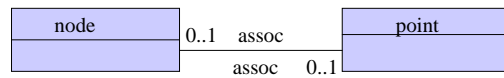
Logical Modeling



Line-Point(line_segment-ID, Start_point-ID, End_point-ID)

Line_Seg ment_ID	Start Point_ID	End Point_ID
ls ₁	1	2
ls ₂	2	3
ls ₃	3	4
ls ₄	4	5
ls ₅	5	6
ls ₆	5	1
ls ₇	1	7
ls ₈	6	7
ls ₉	6	8
ls ₁₀	8	9
ls ₁₁	9	7
ls ₁₂	10	11
ls ₁₃	11	12
ls ₁₄	12	13
ls ₁₅	13	14
ls ₁₆	14	12

Logical Modeling



Node-Point(node-ID, point-ID)

Node-ID	Point-ID
10	1
20	5
30	6
40	7
50	10