Java Tutorial: Working with Objects and Classes

Mark A. Austin

University of Maryland

austin@umd.edu
ENCE 688P, Fall Semester 2020

October 10, 2020

Overview

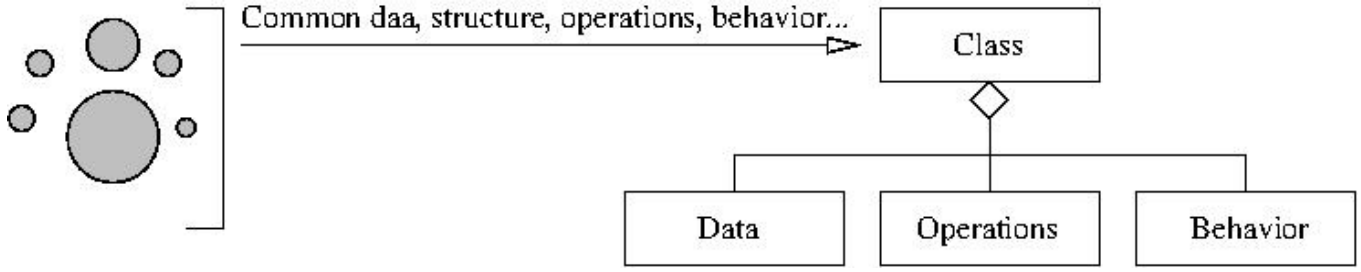
1. Working with Objects
2. Encapsulation and Data Hiding
3. Relationships Among Classes
4. Association Relationships
5. Inheritance Mechanisms
6. Composition of Object Models
7. Applications

Part 3

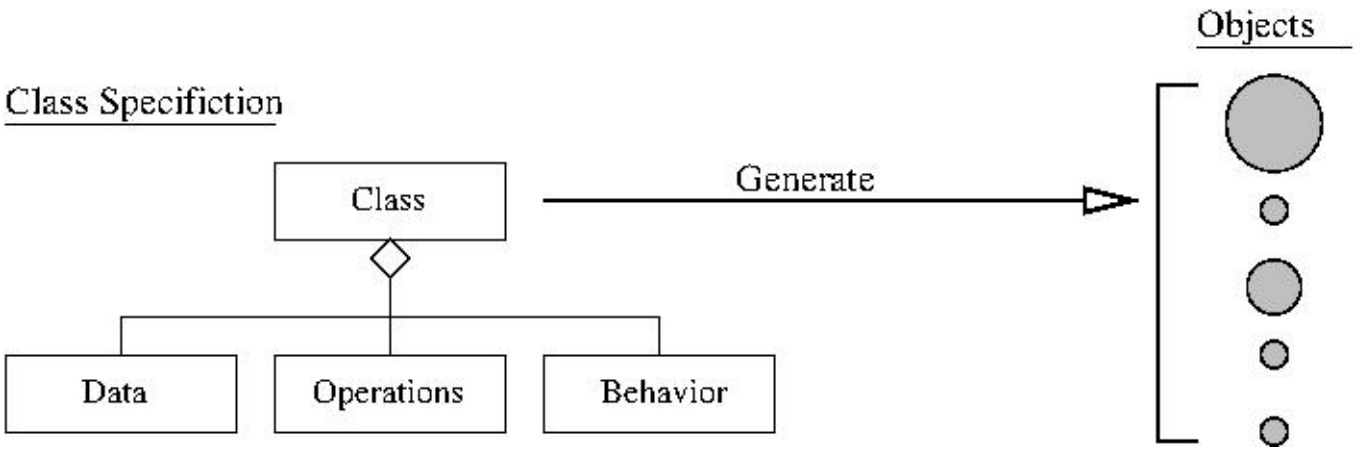
Working with Objects

Working with Objects and Classes

From Collections of Objects to Classes:

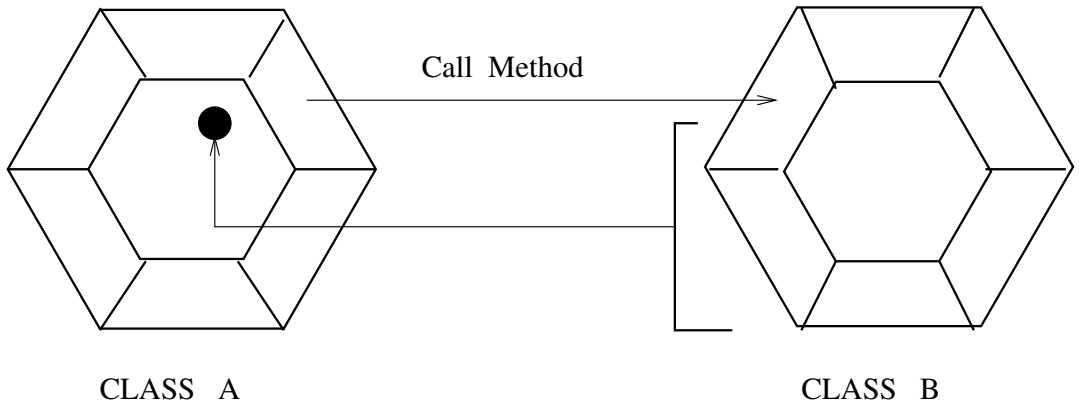


Generation of Objects from Class Specifications:



Relationships Among Classes

1. **Use:** Class A uses Class B (method call).



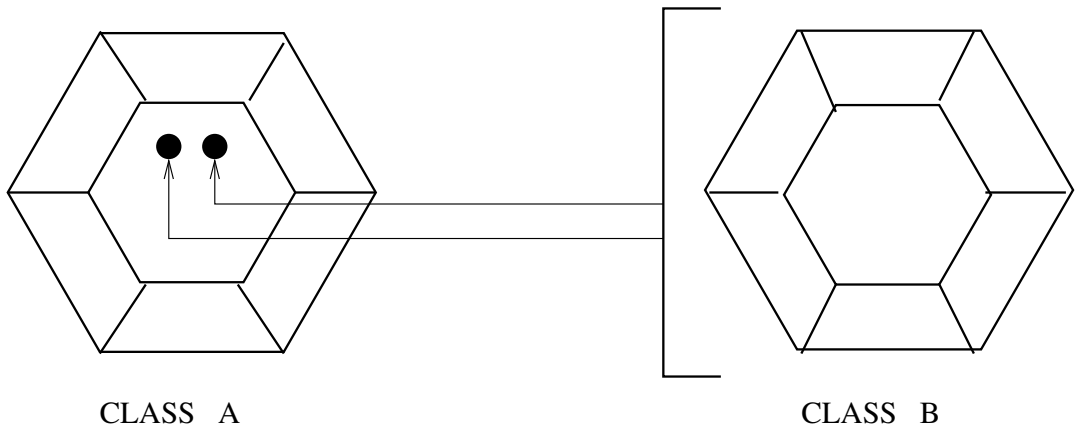
Class `A` uses Class `B` if a method in `A` calls a method in an object of type B.

Example

```
double dAngle = Math.sin ( Math.PI / 3.0 );
```

Relationships Among Classes

2. Containment (Has a): Class A contains a reference to Class B.



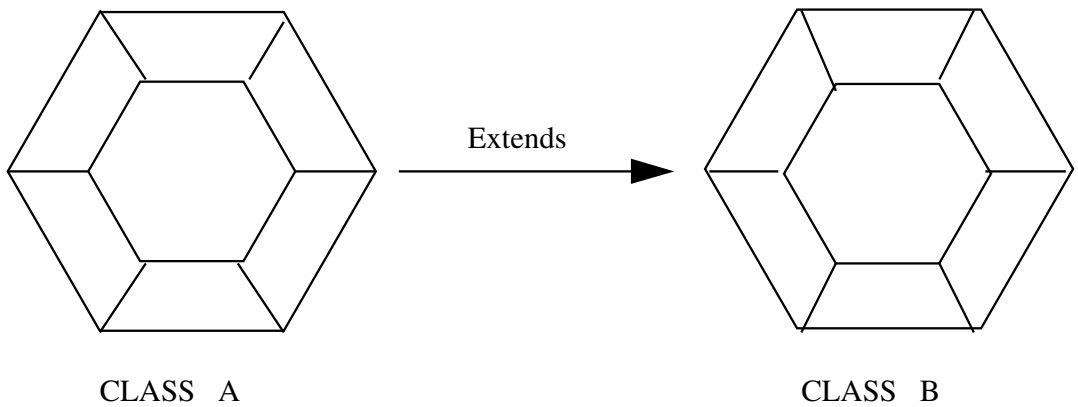
Clearly, containment is a special case of use (i.e., see Item 1.).

Example

```
public class LineSegment {
   private Point start, end;
   .......
}
```

Relationships Among Classes

3. Inheritance (Is a): In everyday life, we think of inheritance as something that is received from a predecessor or past generation. Here, Class B inherits the data and methods (extends) from Class A.



Examples of Java Code

```
public class ColoredCircle extends Circle { .... }
public class GraphicalView extends JFrame { .... }
```

Composition of Object Models

Composition of Object Models

Definition

Composition is known as is a part of or is a relationship.

The member object is a part of the containing class and the member object cannot survive or exist outside the enclosing or containing class or doesn't have a meaning after the lifetime of the enclosing object.

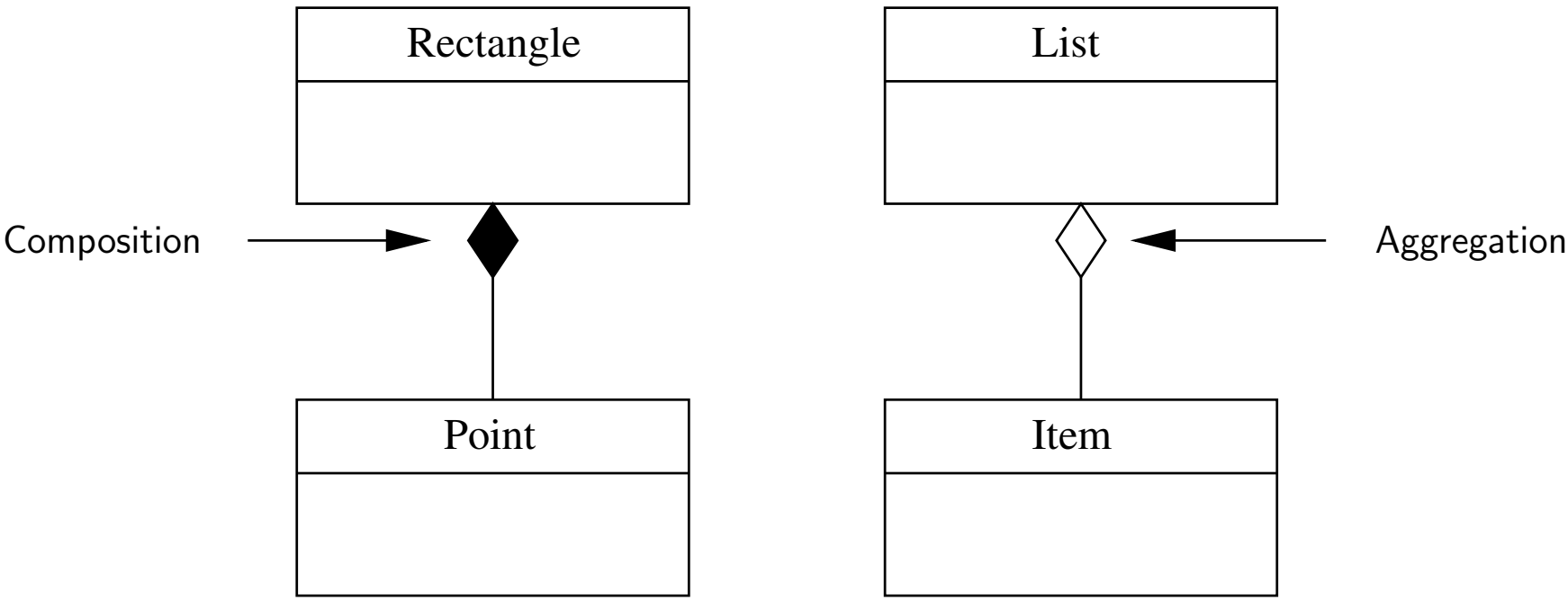
Is it Aggregation or Composition?

- Ask the question: if the part moves, can one deduce that the whole moves with it in normal circumstances?

Example: A car is composition of wheels and an engine. If you drive the car to work, hopefully the wheels go too!

Composition of Object Models

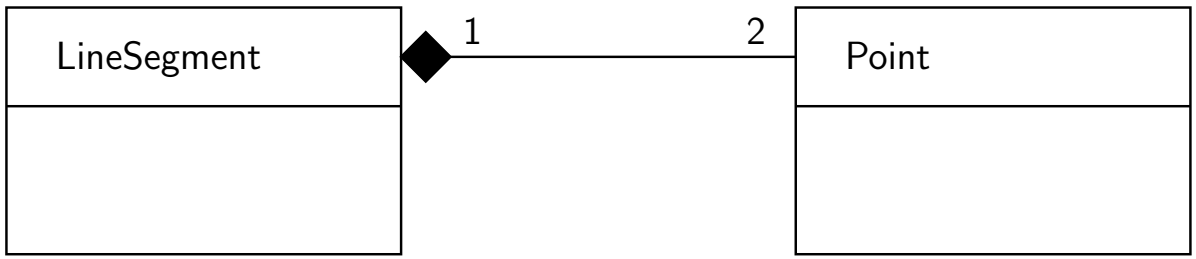
Notation for Aggregation and Composition



Recall: Aggregation is all about grouping of things ...

Example 4. Modeling Line Segments

Example 1. Line segment is composed from two points:



Source Code: Abbreviated Point.java

```
1   public class Point {
2       private int x, y;
3
4       public Point(int x, int y) {
5          this.x = x;
6          this.y = y;
7       }
8
9       public int  getX() {
10         return x;
11      }
12
13      public void setX(int x) {
14         this.x = x;
15      }
```

Example 4. Modeling Line Segments

Source Code: Point.java continued:

```
16
17      public int  getY()      { return y; }
18      public void setY(int y) { this.y = y; }
19
20      public String toString() {
21         return "(" + x + "," + y + ")";
22      }
23  }
```

Source Code: Abbreviated LineSegment.java

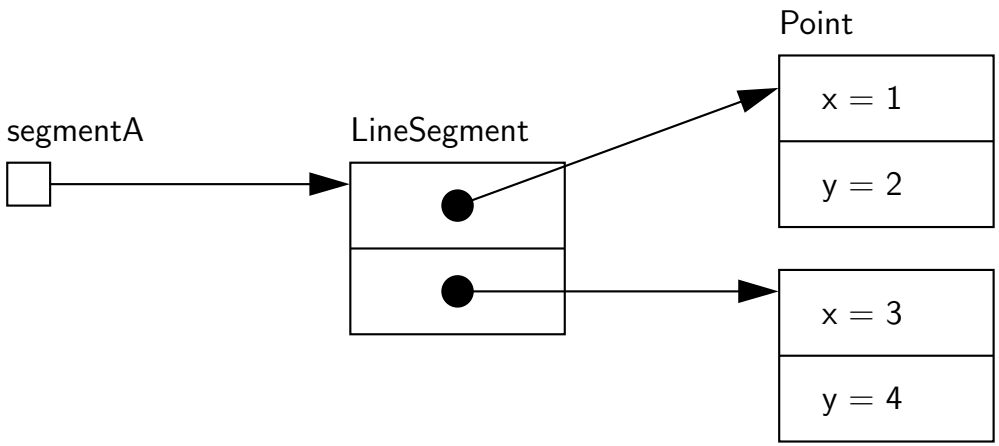
```
 1  public class LineSegment {
 2      Point begin, end;
 3
 4      public LineSegment (int x1, int y1, int x2, int y2) {
 5         begin = new Point(x1, y1);
 6         end   = new Point(x2, y2);
 7      }
 8
 9      public String toString() {
10         return "Line segment: from " + begin + " to " + end;
11      }
12  }
```

Example 4. Modeling Line Segments

Creating a Line Segment Object:

```
LineSegment segmentA = new LineSegment( 1, 2, 3, 4 );
```

The layout of memory is as follows:

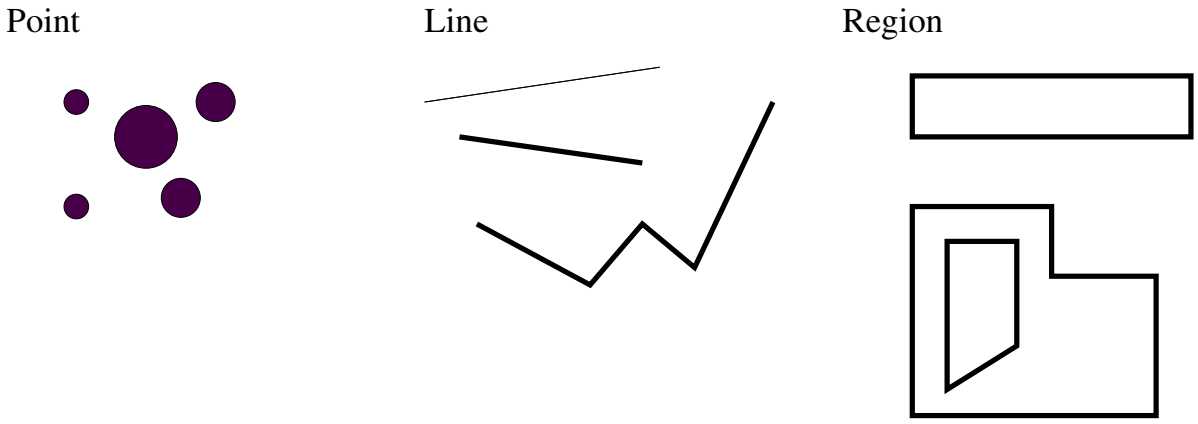


Here, segmentA refers to the memory location for the linesegment object. The linesegment object contains references to Point objects containing the (x,y) coordinates.

Spatial Applications

Spatial Models (Points, Lines, Polygons)

Points, lines and regions are fundamental spatial data types.



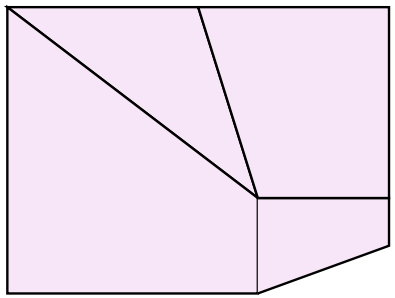
- Points are 0-dimensional entities. Lines are 1-dimensional entities. Regions are 2-dimensional entities.
- Spatial operations: union, intersection, difference.
- We need software that can compute operations on these entities in a consistent manner (e.g., Google: Java Topology Suite).

Class Diagram for GIS Domain

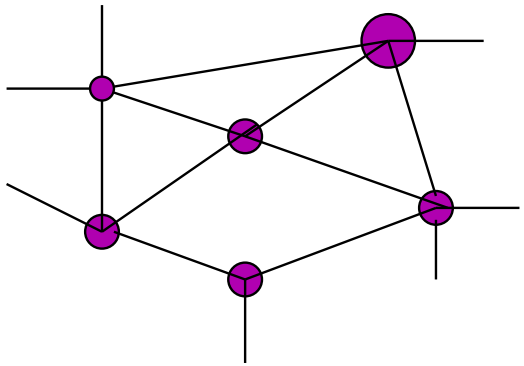
Partitions and Networks

Partitions and networks are two abstractions for modeling collections of spatial objects.

Partitions



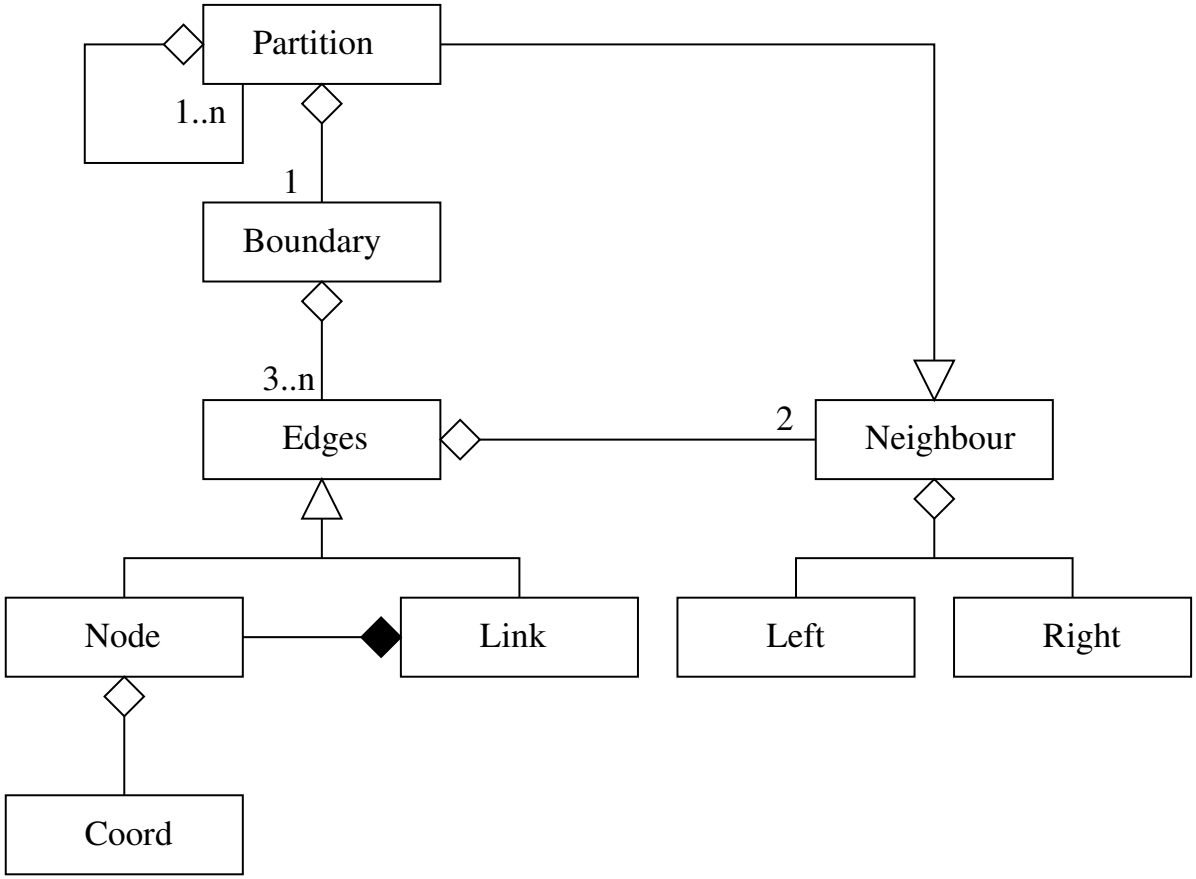
Spatially Distributed Network



- Examples of partitions: rooms in a building, districts in a state, countries in a continent.
- Examples of networks: plumbing and HVAC networks, highways and railway networks, communication and power networks.

Class Diagram for GIS Domain

Conceptual model for partition hierarchies (adapted from Chunithipaisanl S. et al., 2004)



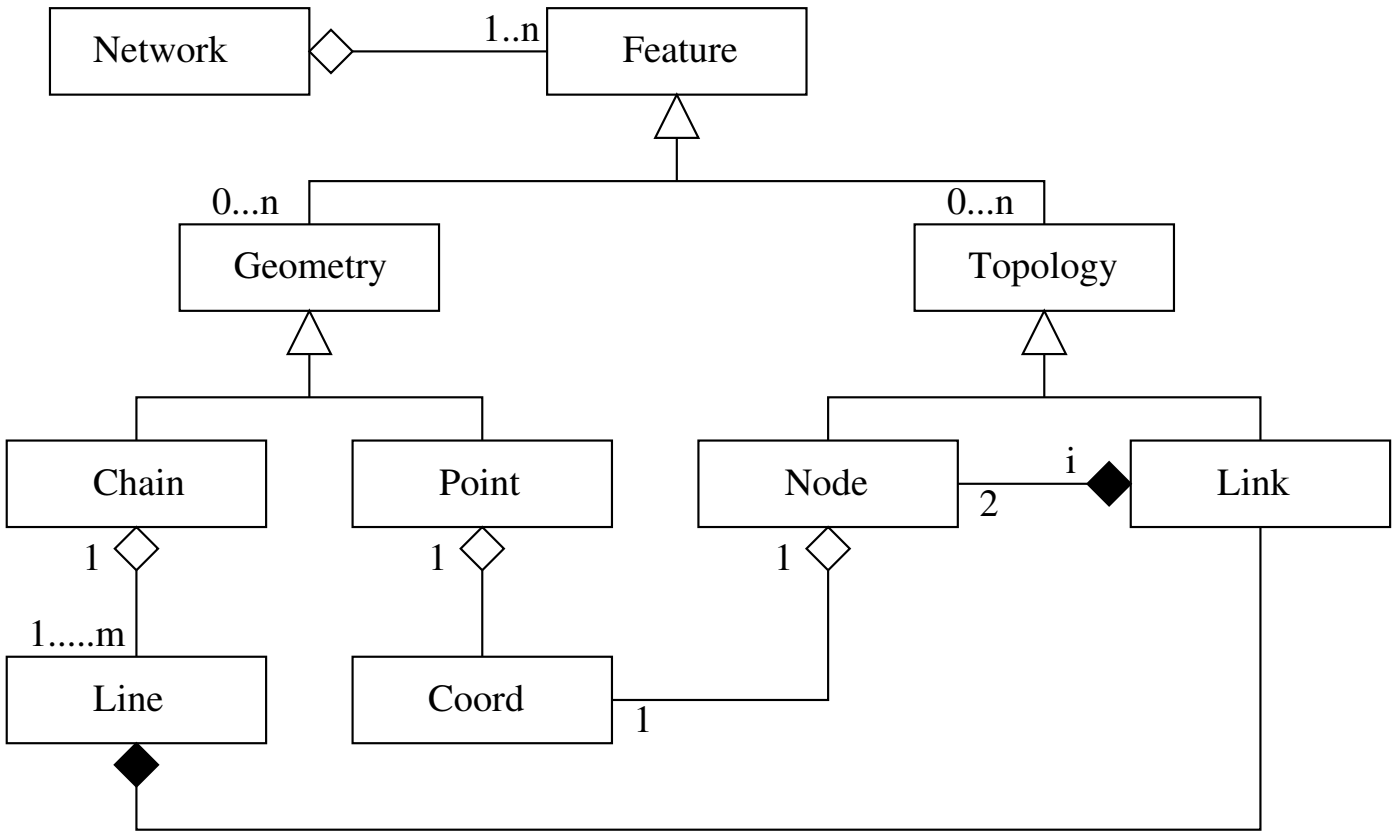
Class Diagram for GIS Domain

The conceptual model for partitions states:

- A Partition can be decomposed into 1 or more Partitions (sub-Partitions).
- Each Partition has one boundary (here we ignore the possibility of partitions containing holes).
- Boundaries are composed of edges (..at least 3 edges).
- Each Edge segment has a Node and Link.
- Nodes and Link are paired in a one-to-one correspondence.
- A Node has a coordinate.
- Edges also have Neighboring Partitions.
- Neighboring Partitions can be classified as to whether they are on the Left and Right of the Edge.

Class Diagram for GIS Domain

Conceptual model for networks (Adapted from: Chunithipaisanl S. et al., 2004).



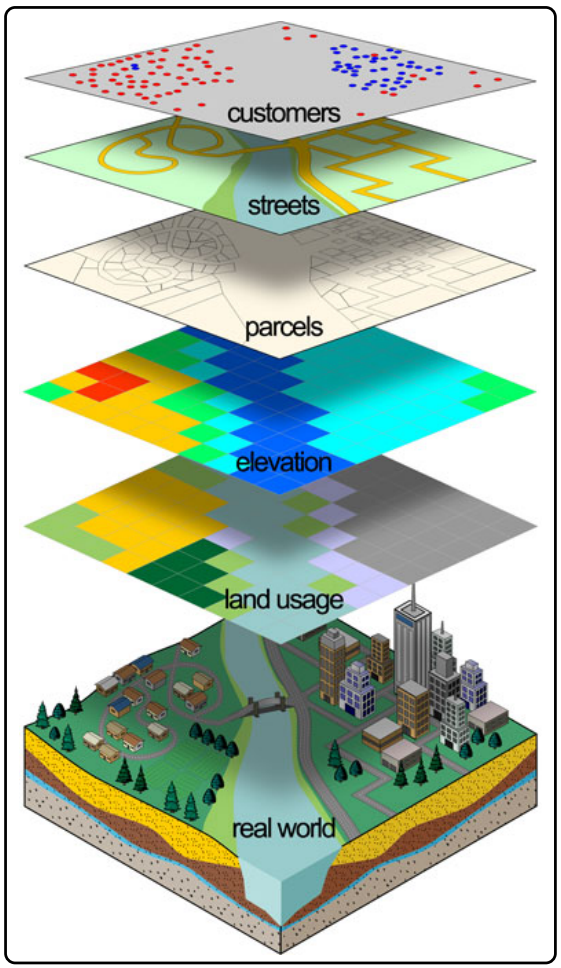
Class Diagram for GIS Domain

The conceptual model for networks states:

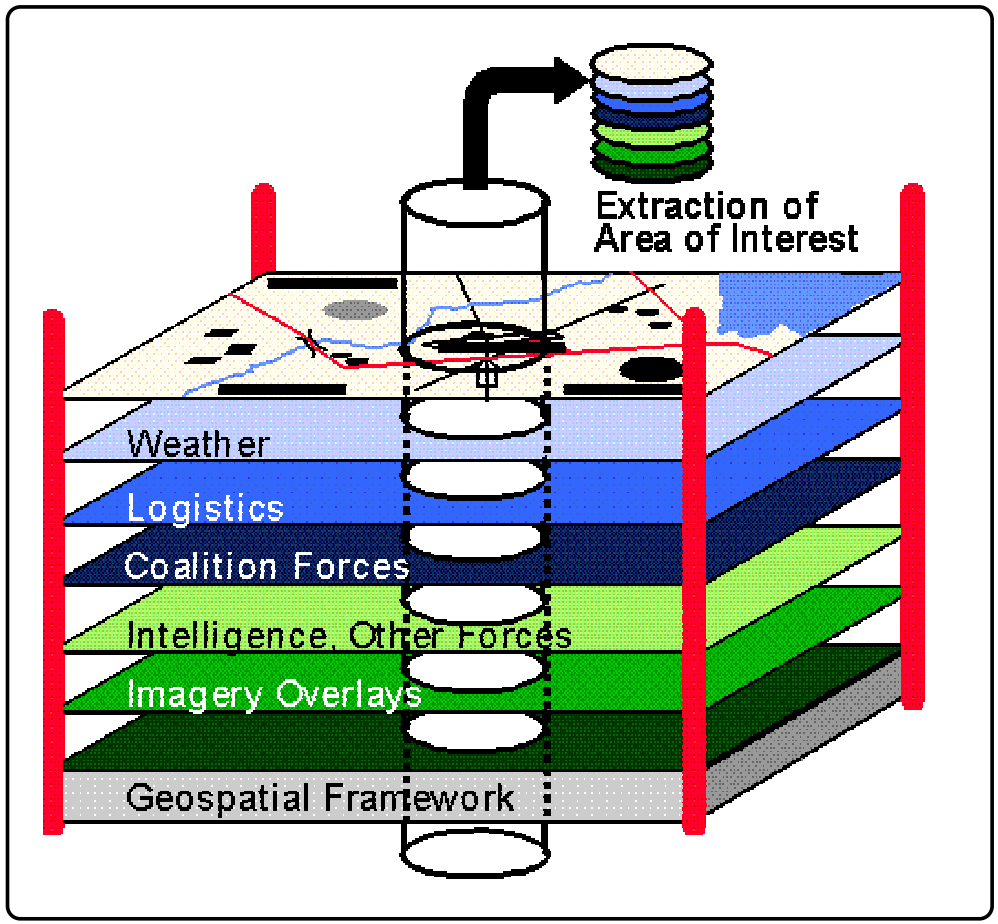
- A Network is composed of Features.
- **2.** Each Feature has Geometry and Topology.
- Geometry is a generalization for Chains and Points...
- A Chain corresponds to one or more Line segments.
- A Point has a coordinate.
- Topology is a generalization for Nodes and Links.
- Nodes also have coordinates.

Layered Organization of Attributes in Urban Data

Geographic Information System

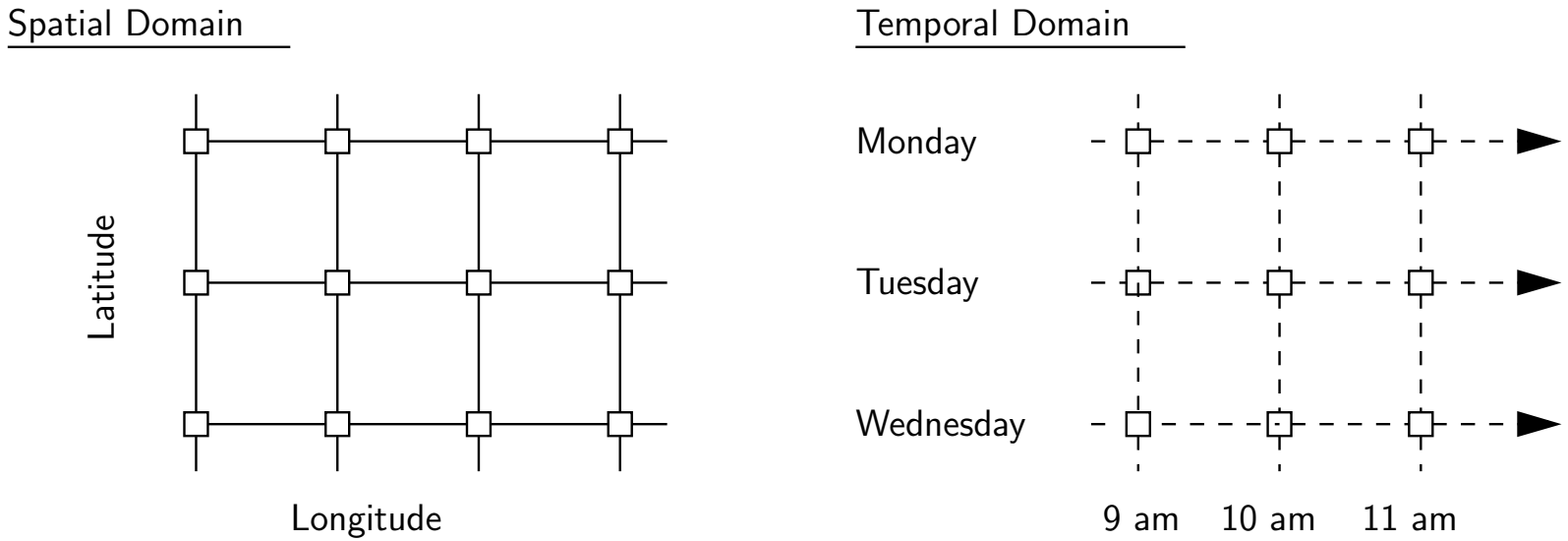


Layers of Data / Information in Military Decision Making



Spatial and Temporal Domains

Goal. We want to know that systems do the right thing (event) in the right place (spatial) at the right time (temporal).



2D Spatial Domain: OpenStreetMap, Java Topology Suite.
Temporal Domain: Calendars, Scheduling Algorithms, Ontologies of Time, UPPAAL.