

# OMT

## Object Modeling Technique

### Sources

- Rumbaugh.J. (& others) *Object Oriented Modeling and Design* Prentice-Hall
- Booch. G. *Object Oriented Analysis and Design* Benjamin/Cummings Publishing

### What is Object-Oriented ?

#### Identity

- Data existing as discrete, distinguishable entities called objects
- In the real world an object simply exists.
- In a software environment each object has a unique handle by which it can be uniquely referenced.

### What is Object-Oriented ?

#### Classification

- Objects with the same data structures (attributes) and the same behavior (operations) are grouped together into a class.

### What is Object-Oriented ?

#### Classification

- A class is an abstraction which describes those properties regarded as important to an application and which ignores whatever properties analysts/designers regard as irrelevant. Any choice of classes is therefore subjective and problem-specific.
- Each object is said to be an instance of some class.

### What is Object-Oriented ?

#### Classification

- Each object is said to be an instance of some class.
- An object contains an implicit reference to its class - 'an object knows what kind of thing it is'

## What is Object-Oriented ?

### • **Polymorphism**

- The same operation may produce different behavior for objects from different classes
- e.g., asking a window to move produces one kind of behavior
- asking a chess piece to move produces a different kind of behavior

## What is Object-Oriented ?

### **Inheritance**

- Sharing attributes and operations among classes based on a hierarchical relationship.
- Each subclass incorporates (inherits) all of the properties of its super-class and adds its own unique properties.

## What is Object-Oriented Development ?

- Object Orientation encourages thinking in the application domain for most of the software life cycle. e.g., analyze and design using concepts like invoice, policy, radio tuner, weather station, whatever.
- Thinking in the programming domain (e.g., concepts like data structure ) only where required (at which point another layer of OO ideas can be highly relevant).

## What is Object-Oriented Development ?

- The harder part of software development is visualizing and manipulating the 'essence of the problem', not dealing with the accidental arbitrary problems which development tools / programming languages inflict upon the process.

## Object Oriented Themes

### **Abstraction**

- Selective examinations of certain aspects of a problem.
- Isolation of relevant details, suppression of irrelevant details.
- A good abstraction / model describes crucial aspects of the world and ignores the rest.

## Object Oriented Themes

### **Encapsulation**

- Also known as Information Hiding
- Separating the external aspects of an object, which are accessible to other objects, from the internal aspects, which are hidden from other objects.

## Object Oriented Themes

### Combining Data and Behavior

- Each class implements its own operations or specifically inherits them from super-classes. The caller of an operation need not be concerned with how many implementations of an operation exist and which one to use. The class hierarchy will sort out which implementation of an operation to use.

## OMT Overview

### Analysis stage

- Building a model of the real-world situation showing just its important properties.
- The model is a precise abstraction which describes what a system does not how it does it.
- Implementation detail is selectively and deliberately ignored.

## OMT Overview

### System Design

System designer makes high level decisions about overall architecture.

Target system organized into subsystems based upon analysis structure and proposed architecture. Decisions made about (for example) what performance characteristics to optimize, and strategies for achieving them.

## OMT Overview

### Object Design

- Building a design model based upon the analysis model but containing implementation details.
- Adding detail to the design model according to the strategies established during System Design.

## OMT Overview

### Object Design

- Object classes from analysis are still meaningful but are now augmented with computer domain data structures and algorithms.
- Focus is on data structures and algorithms used to implement each class.

## OMT Overview

### Object Design

- Application-domain objects and computer-domain objects are described using the same OO concepts and notations.

## OMT Overview

### Implementation

- Object classes and relationships developed during Object Design are finally translated into a particular programming language, database, or hardware implementation. (in principle, this is the easy stage)

## Three Models within OMT

- OMT is a ternary approach.
- The three models in OMT provide complementary views of the same system. (three different perspectives) as in Structured Methods.

## The Object Model

- Describes the static structure of the objects in a system and the relationships between those objects.
- OMT uses Object Diagrams. (Booch)
- Object Diagrams are graphs where the nodes represent classes and the arcs represent relationships among classes. (fairly similar to ERD diagrams and doing much the same job)

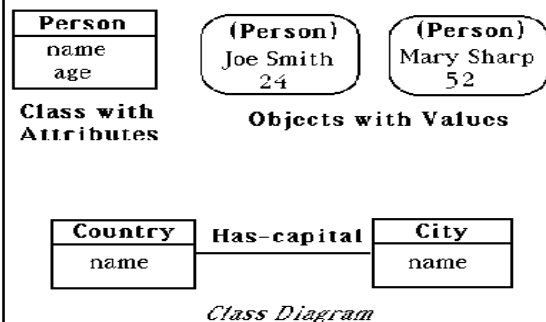
## The Dynamic Model

- Describes the aspects of the system that change over time. The dynamic model is used to specify and implement the control aspects of the system.
- OMT uses State Transition Diagrams.
- A State Diagram is a graph where the nodes represent states and the arcs represent transitions between states caused by events.

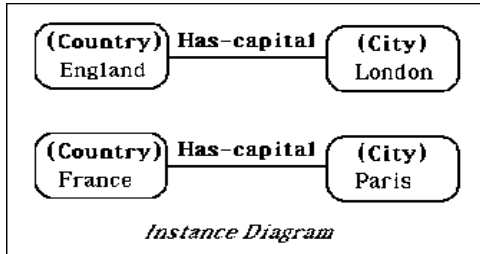
## The Functional Model

- Describes the data value transformations within a system.
- OMT uses Data Flow Diagrams.
- A Data Flow Diagram is a graph where the nodes represent processes (or external entities or data stores) and the arcs represent data flows.

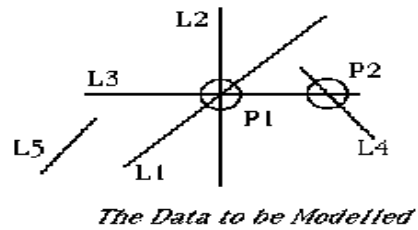
## Object Diagrams - Notation



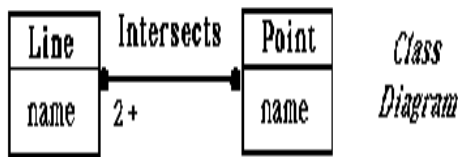
### Instance Diagrams



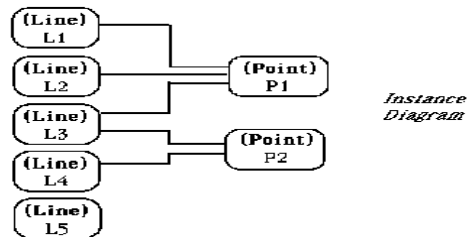
### Example



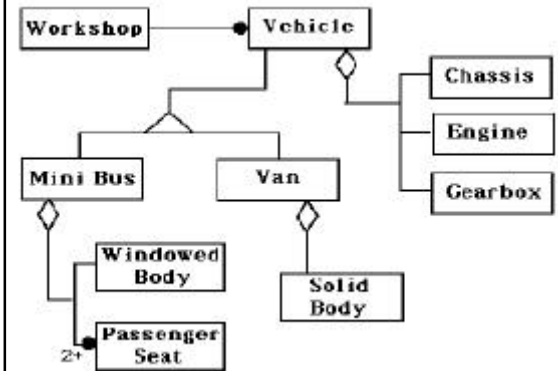
### Class Diagram



### Instance Diagram



### Class Diagram - Object Modeling Technique



### Association Symbols

#### Association Symbols

'Has-capital' is an example of a 1 to 1 Association.

