

System Modeling and Top-Down Design

The OMT Method

Modeling

Goals:

- Specification of system
 - Structure, including hierarchy
 - Behavior, including concurrency
 - Control-flow
 - Data-flow
- Executable models
- Automatic code synthesis

Modeling

Purposes:

- testing a physical entity before building
- communication with customers
- visualization
- reduction of complexity
- better understanding of problem

Object Modeling Technique (OMT)

- Object Model
 - static, structural, "data" aspects of system
 - describes the objects in the system and their relationships (what)
- Dynamic Model
 - temporal, behavioral, "control" aspects of system
 - describes the aspects of a system that change over time
 - interactions among objects (when)
- Functional Model
 - transformational, "function" aspects of system
 - describes the data transformations of the system (how)

Object Model

- Object - a concept, abstraction, or thing with concrete boundaries and meaning for the problem at hand
 - promote understanding of the real world
 - provide a practical basis for computer implementation
 - all objects are distinguishable and have identity
- Class - a group of objects with similar properties, common behavior, common relationships to other objects, and common semantics
 - an object is an instance of a class

Object Model

- Attribute - a data value held by an object; a pure value not an object
- Operation - a function or transformation that may be applied to or by an object

Object Model

Class Notation

Class-Name
attribute: data-type=default value
operation(argument list): result value type

- attributes and operations may or may not be shown depending on desired level of detail

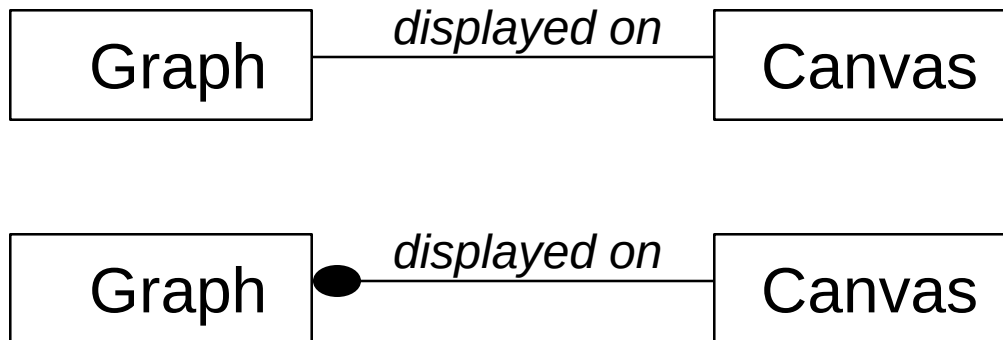
Object Model

Association - means for establishing relationships among classes

- link - a physical or conceptual connection between object instances
- a group of links with common structure and semantics
- inherently bidirectional
- may be binary, ternary, or higher order

Object Model

- Multiplicity - how many instances of one class may relate to an instance of an associated class
- Notation



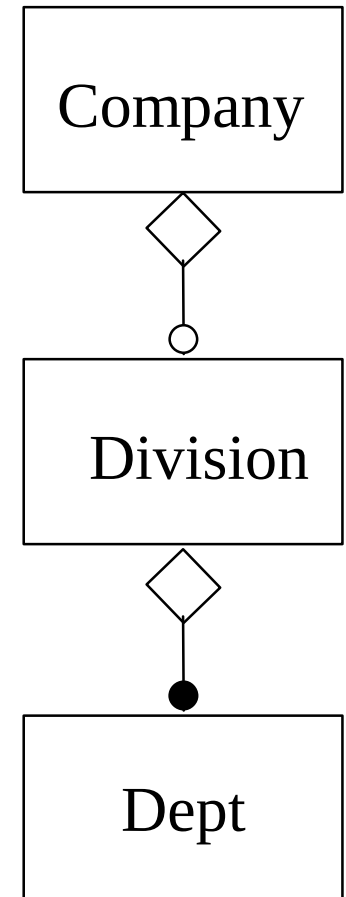
Multiplicities

●	one
○	many: zero or more
3	optional: zero or one
2+	exactly the number
3-5	two or more
2,4,6	3 to 5
	2 or 4 or 6

Object Model

Aggregation - is the "a-part-of" relationship in which objects representing the components of something are associated with an object representing the entire assembly

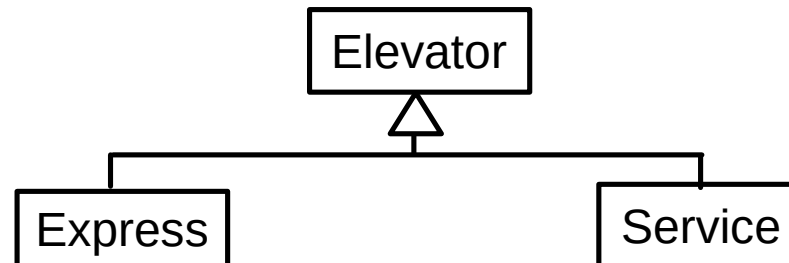
- is transitive
- is a special form of association



Object Model

Generalization - is the relationship between a class and one or more refined versions of it

- the class being refined is called the superclass
- the class being generalized is called the subclass
- each subclass is said to inherit the features of its superclass
- "is-a" relationship
- transitive across an arbitrary number of levels



Object Model

Creation Tips

- understand the problem
- keep it simple at first, then refine later
- choose class names carefully
- try to have only binary relations
- do not worry about multiplicities on first draft
- do not feel you have to use all constructs
- concentrate on WHAT
- document reasons behind the model
- refine until complete and correct

Dynamic Model

The aspects of a system that are concerned with time, changes, and control

- sequences of operations that occur in response to external stimuli
- no consideration of what the operations do, what they operate on, or how they are implemented

Dynamic Model

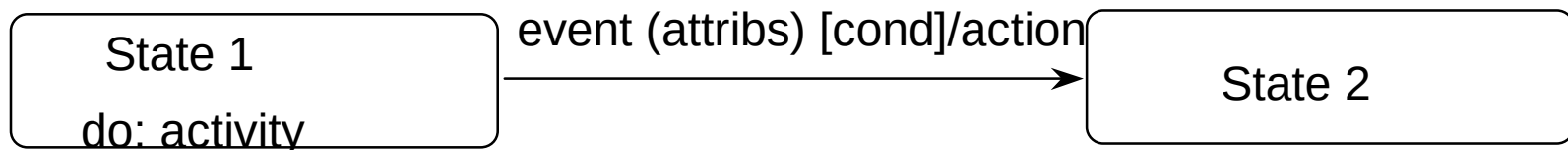
- Event - an external stimuli, something that happens at a point in time
- State - the attributes and links held by an object at a particular point in time
- State Diagram - relates the pattern of events, states, and state transitions for a given class

Dynamic Model

Model consists of multiple state diagrams

- shows the pattern of activity for an entire system

Notation: a state diagram is a graph whose nodes are states and whose directed arcs are transitions labeled by event names



Dynamic Model

- Condition - a boolean function of object values
 - can be used as a guard on transitions; transition fires only if guard is true
- Activity - an operation that takes time to complete and is associated with a state
 - starts on entry to the state and stops on exit, or after completion

Dynamic Model

- Action - an instantaneous operation that is associated with an event
- Entry and Exit Actions - performed on entry or exit from state
- Internal Action - an event can cause an action to be performed without causing a state change
 - entry and exit actions not performed
 - different from self-transition

Dynamic Model

Performance Order

- actions on incoming transition
- entry action
- do activity
- exit action
- actions on outgoing transition

Dynamic Model

Creation Tips

- only construct state diagrams for object classes with meaningful dynamic behavior
- verify consistency between diagrams for shared events
- use scenarios to begin the construction of diagrams
- let application decide on granularity and distinguish between actions and activities
- use nested states to improve understanding/readability
- distinguish state diagrams for super and subclasses
- watch for race conditions in the diagrams

Functional Model

- The functional model describes computations within a system:
 - processing of information
 - how output values are derived from input values without regard to the order of computations
- The functional model consists of multiple data flow diagrams.

Functional Model

- Process - transforms data values
- Data Flow - connects output of an object or process to the input of another object or process
- Actor - an active object that drives the data flow by producing or consuming data values
- Data Store - a passive object that stores data for later use
- Notation: a data flow diagram is a graph showing the flow of data values from their sources through processes that transform them to their destinations in objects



Functional Model

Creation Tips

- Start simple, basically one process with correct inputs and outputs
- Refine to next level by expanding the process
- Continue to refine one process at a time
- Ensure consistency between the inputs and outputs throughout your revisions