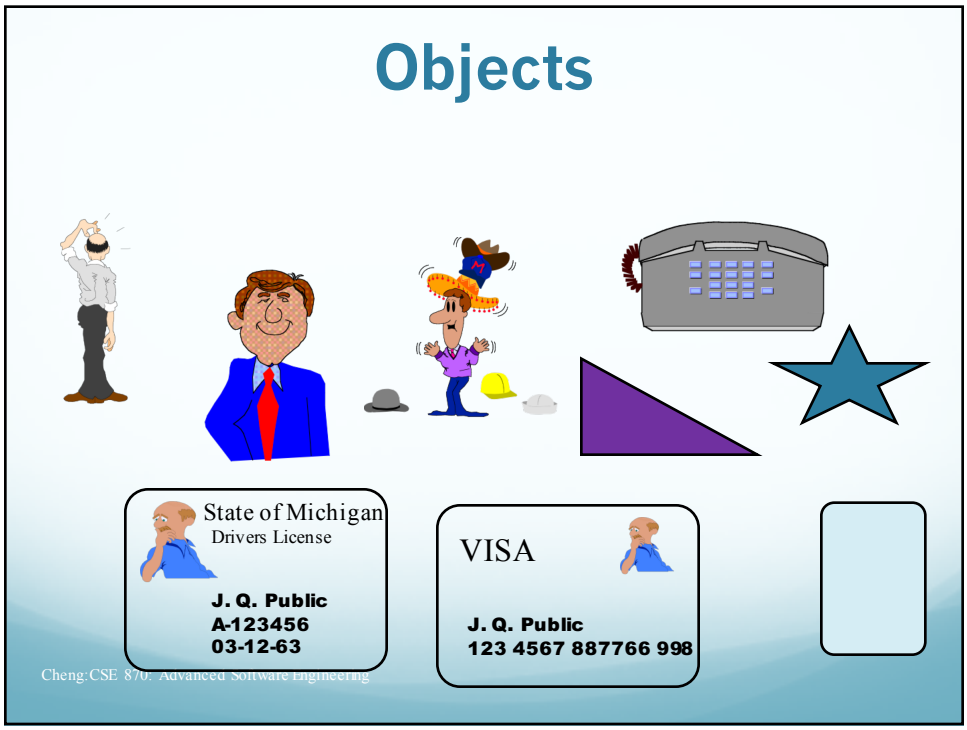
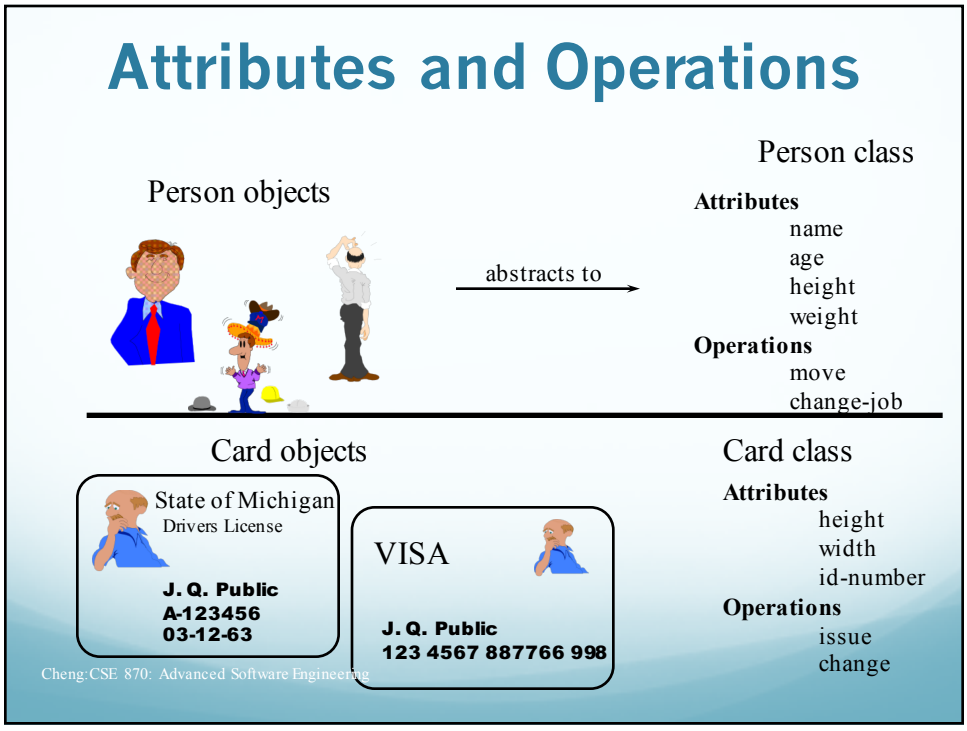
Object-Oriented Modeling

One paradigm of development

The OO Solution

- The OO model closely resembles the problem domain
  - Base your model on the objects in the problem domain
- Iteratively refine the high-level model until you have an implementation
  - Attempt to avoid big conceptual jumps during the development process


Objects
State of Michigan
Drivers License
J. Q. Public
A-123456
03-12-63
VISA
J. Q. Public
123 4567 887766 998
Cheng:CSE 870: Advanced Software Engineering


Attributes and Operations
Person objects
abstracts to
Person class
Attributes
name
age
height
weight
Operations
move
change-job
Card objects
State of Michigan
Drivers License
J. Q. Public
A-123456
03-12-63
VISA
J. Q. Public
123 4567 887766 998
Card class
Attributes
height
width
id-number
Operations
issue
change
Cheng:CSE 870: Advanced Software Engineering

Characteristics of Objects

- Identity
  - Discrete and distinguishable entities
- Classification
  - Abstract entities with the same structure (attributes) and behavior (operations) into classes
- Polymorphism
  - The same operation may behave differently on different classes
- Inheritance
  - Sharing of attributes and operations based on a hierarchical relationship

The Class Diagrams

Objects

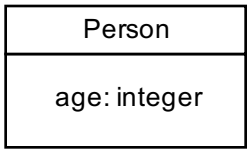
- Something that makes sense in the application context (application domain)
  - J.Q. Public
  - Joe's Homework Assignment 1
  - J. Q. Public's drivers license
- All objects have identity and are distinguishable
- NOT objects
  - Person
  - Drivers license

Classes

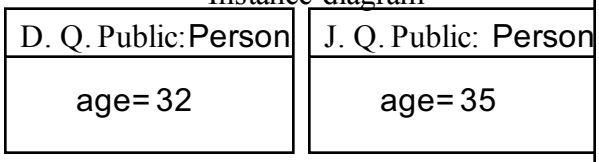
- Describes a group of objects with similar properties (attributes), common behavior (operations), common relationships to other classes, and common semantics
- Person
  - J. Q. Public
  - Joe Smith
  - D. Q. Public
  - Card
    - Credit card
    - Drivers license
    - Teller card

Class Diagrams

Class diagram

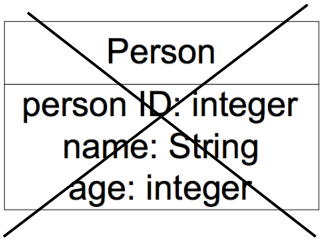


Instance diagram



Class with attributes

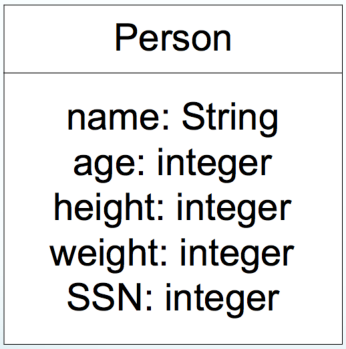
Objects with values

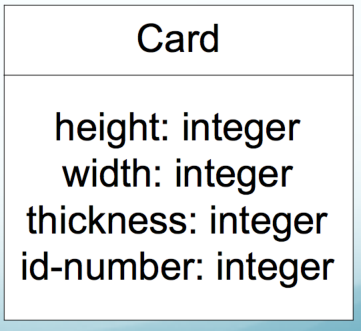


Objects have an identity

Do not explicitly list object identifiers
SSN OK!

Examples





Cheng:CSE 870: Advanced Software Engineering

Operations and Methods

- Transformation that can be applied to or performed by an object
- May have arguments

Card
height: integer width: integer thickness: integer id-number: integer
issue() revoke()

Shape
height: integer width: integer
rotate(angle: integer) move(x: integer, y: integer)

Cheng:CSE 870: Advanced Software Engineering

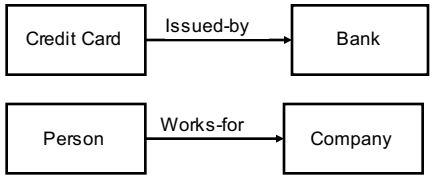
Object Notation - Summary

Class name
attribute-1 : data-type-1 = default-value-1 attribute-2 : data-type-2 = default-value-2 attribute-3 : data-type-3 = default-value-3
operation-1(argument-list-1) : result-type-1 operation-2(argument-list-2) : result-type-2 operation-3(argument-list-3) : result-type-3

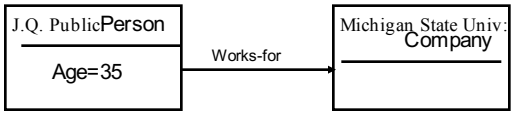
Cheng:CSE 870: Advanced Software Engineering

Associations

- Conceptual connection between classes
  - A credit card is issued-by a bank
  - A person works-for a company



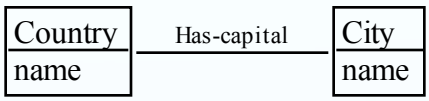
Class diagrams

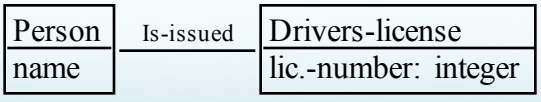


Instance diagram

Associations are Bi-directional

- There is no direction implied in an association (Rumbaugh - OMT)

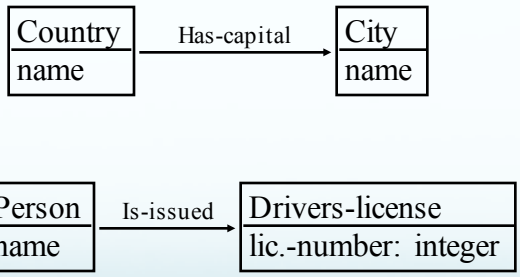




Cheng:CSE 870: Advanced Software Engineering

Associations Have Direction

- Unified adds a direction indicator
  - Inconsistently used



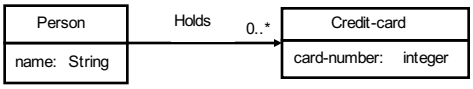
Cheng:CSE 870: Advanced Software Engineering

Multiplicity

One person holds one credit card

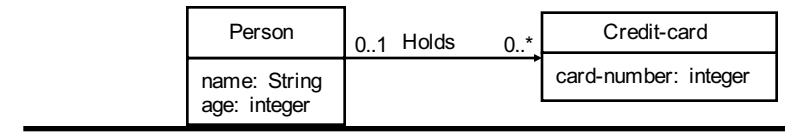
- One object can be related to many objects through the same association

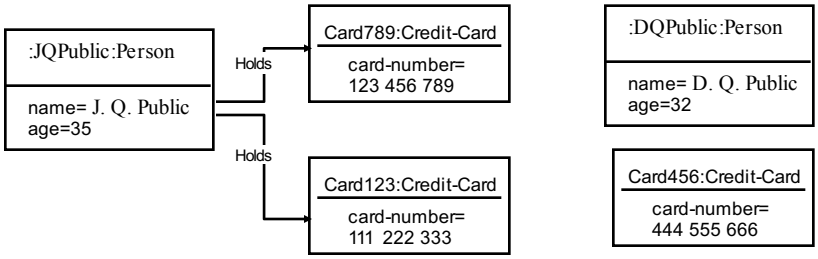
One person can hold zero or more credit cards



Multiplicity (Cont.)

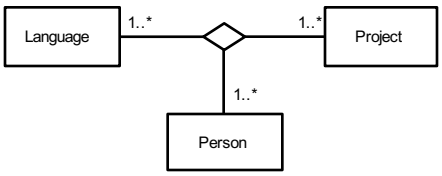
- One person can hold zero or more credit cards (0..*)
- Each card has zero or one holder (0..1)



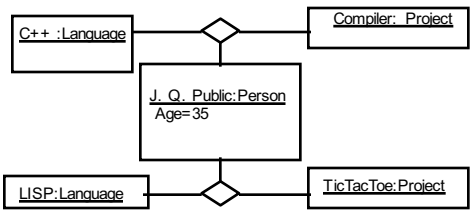


Higher order associations

- Ternary association
  - Project, language, person
- Seldom needed (and should be avoided)

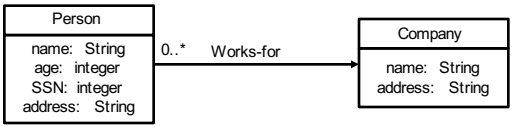


Note: hexagons should be rectangles to represent instances

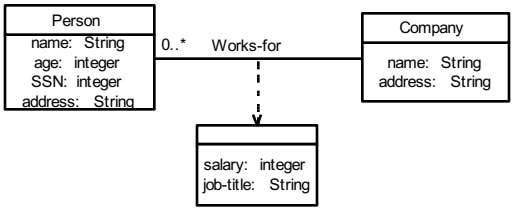


Link Attributes

- Associations can have properties the same way objects have properties

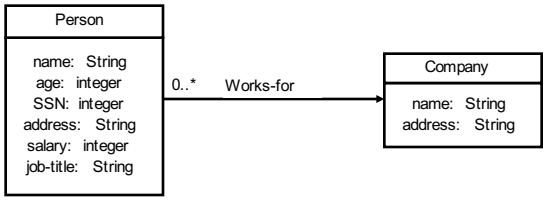


How to represent salary and job title?



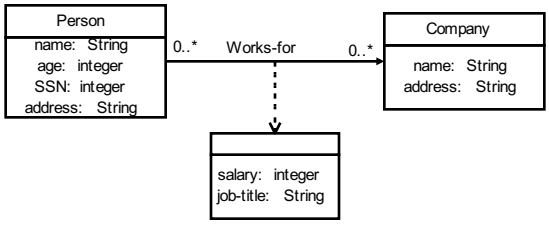
Use a link attribute!

Folding Link Attributes



Why not this?

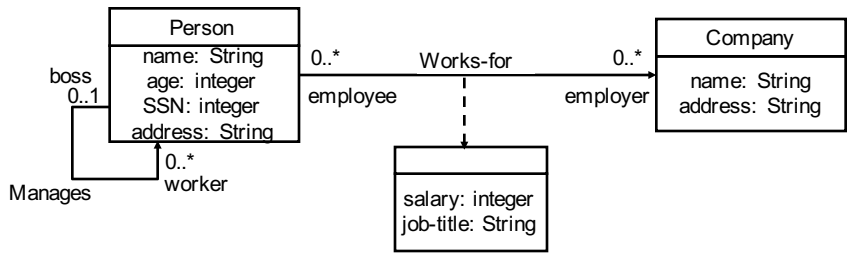
Salary and job title are properties of the job **not** the person



In this case, a link attribute is the only solution

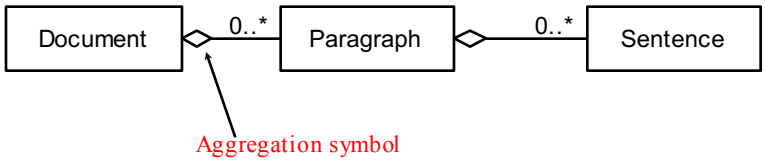
Role Names

- Attach names to the ends of an association to clarify its meaning



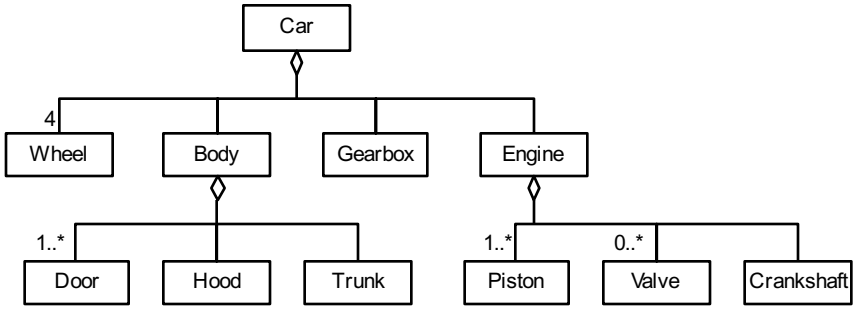
Aggregation

- A special association, the is-part-of association
  - A sentence is part of a paragraph (a paragraph consists of sentences)
  - A paragraph is part of a document (a document consists of paragraphs)

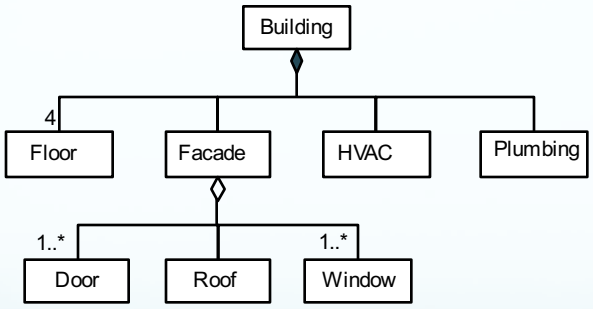


Aggregation (Cont.)

- Often used in parts explosion



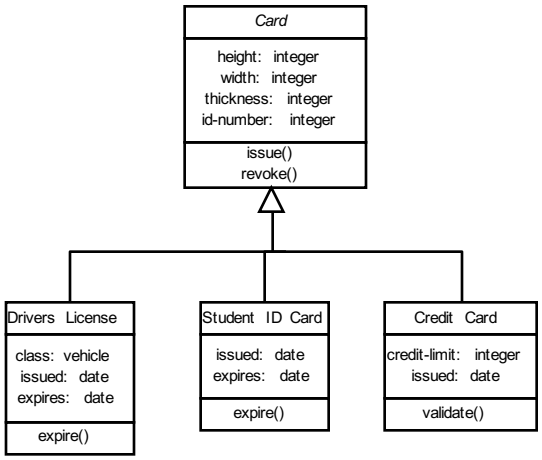
Composition



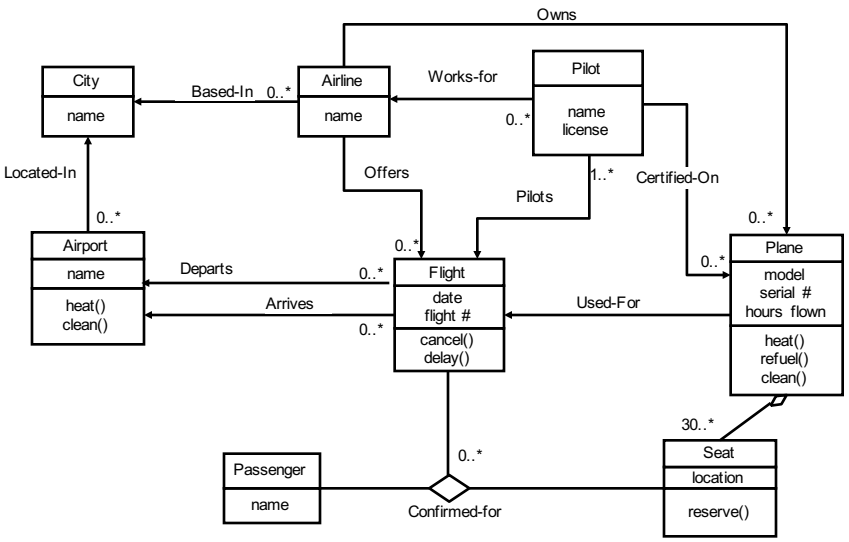
Cheng:CSE 870: Advanced Software Engineering

Generalization and Inheritance

- The is-a association
  - Cards have many properties in common
  - Generalize the common properties to a separate class, the base-card
  - Let all cards inherit from this class, all cards is-a base-card (plus possibly something more)

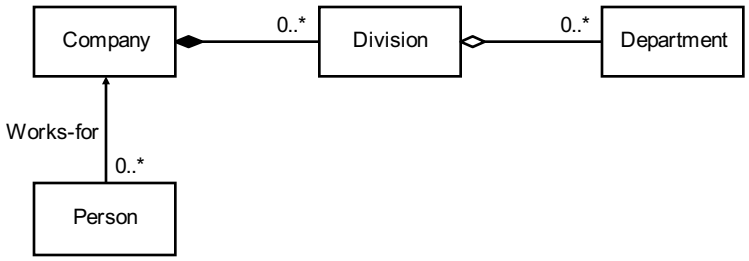


Example



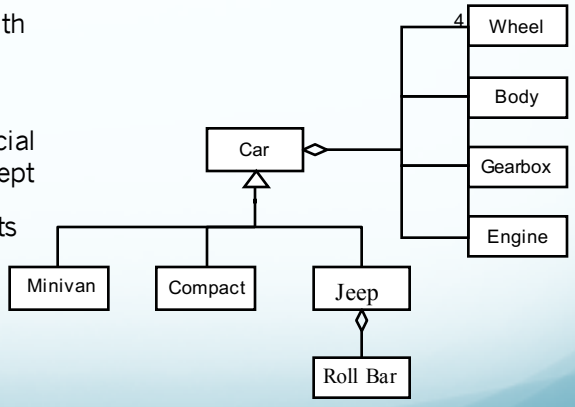
Aggregation Versus Association

- Can you use the phrase is-part-of or is-made-of
- Are operations automatically applied to the parts (for example, move) - aggregation
- Not clear what it should be.....



Aggregation Versus Inheritance

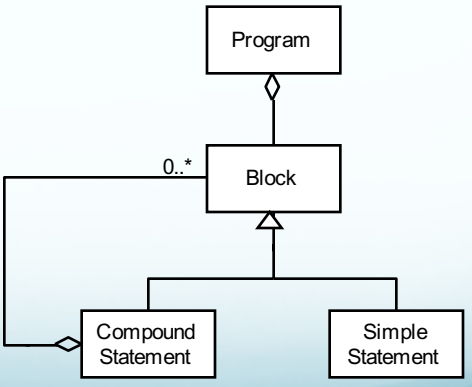
- Do not confuse the is-a relation (inheritance) with the is-part-of relation (aggregation)
- Use inheritance for special cases of a general concept
- Use aggregation for parts explosion



Cheng:CSE 870: Advanced Software Engineering

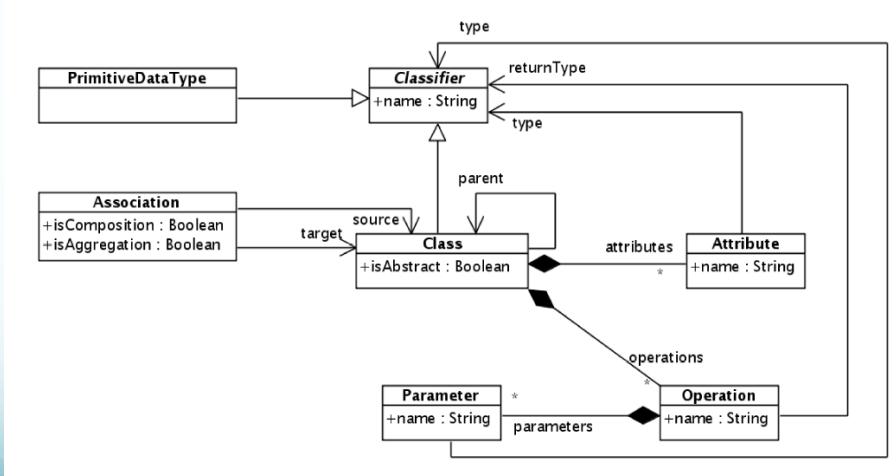
Recursive Aggregates

- A recursive aggregate contains (directly or indirectly) an instance of the same kind of aggregate



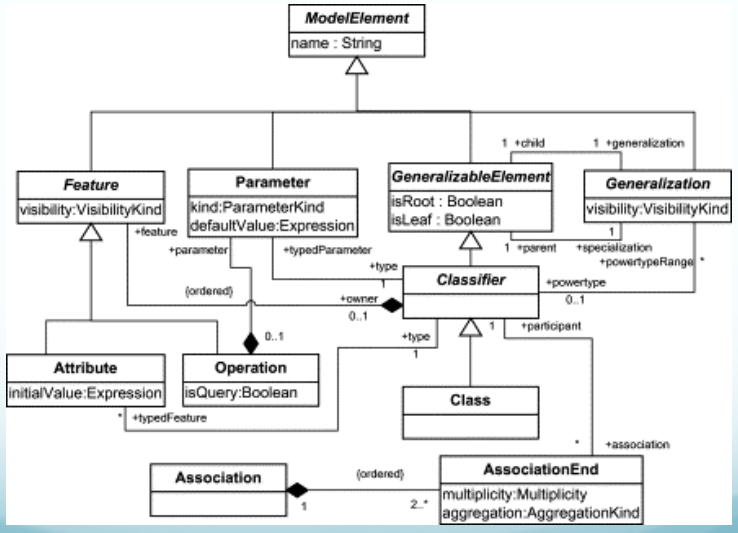
Cheng:CSE 870: Advanced Software Engineering

Class diagram Metamodel I



Cheng:CSE 870: Advanced Software Engineering Eclipse.org

Class diagram Metamodel II



Cheng:CSE 870: Advanced Software Engineering

Science direct

Use Case Metamodel I

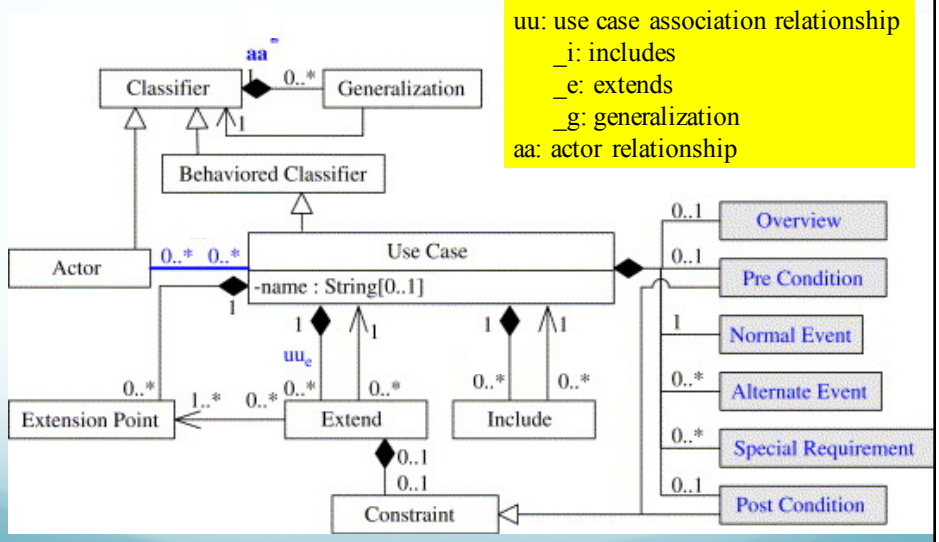


Image: Science Direct: Visual Modeling for Software Intensive Systems, Kooper et al, 2006.

Use Case Metamodel II

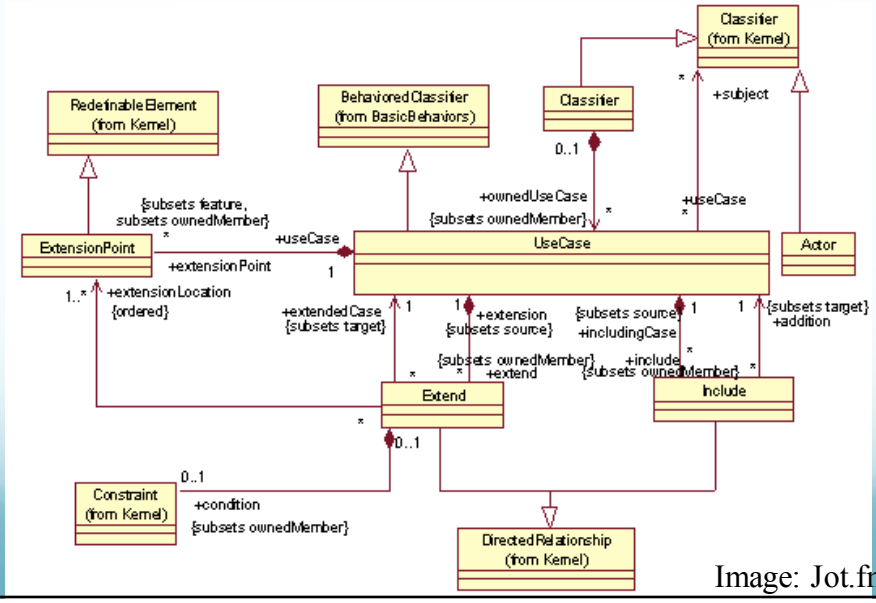
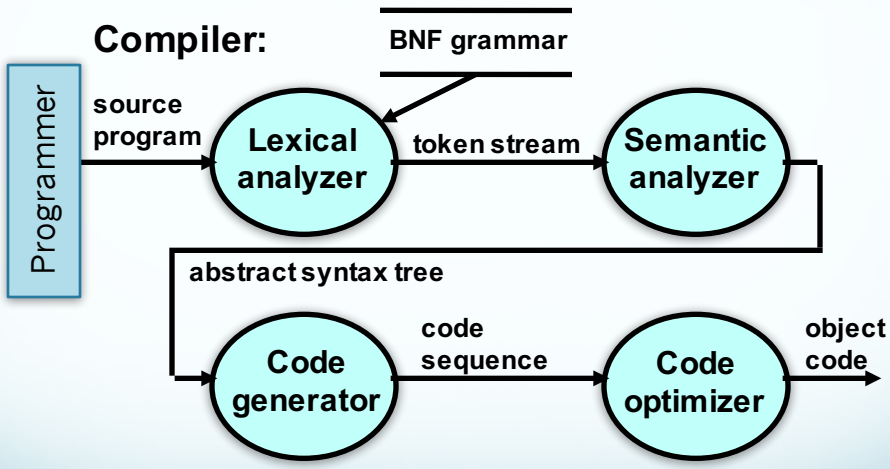


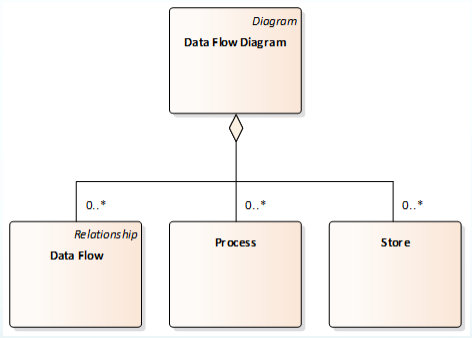
Image: Jot.fm

Data flow diagram



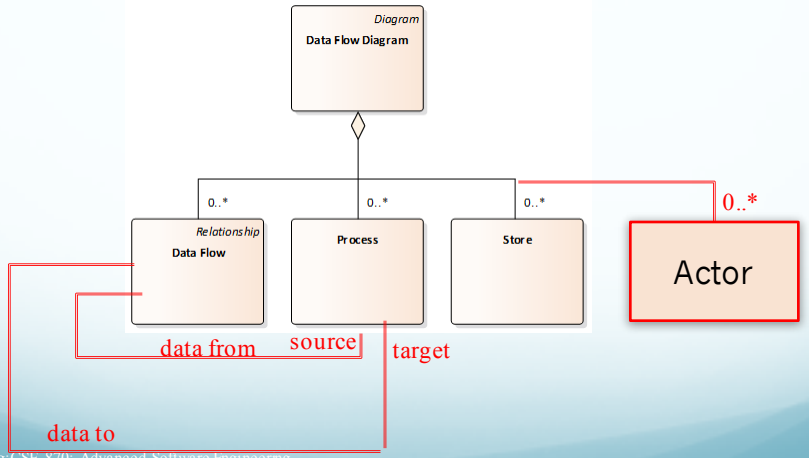
Cheng:CSE 870: Advanced Software Engineering

Possible Metamodel for DFD



Cheng:CSE 870: Advanced Software Engineering

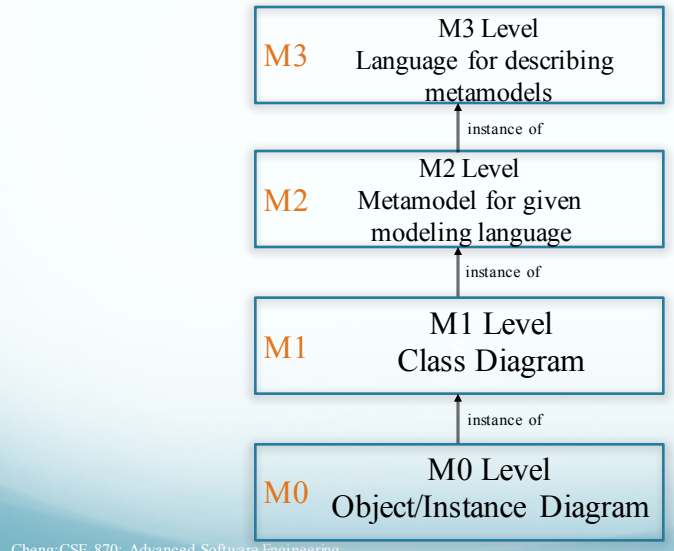
Revised Metamodel for DFD



Cheng:CSE 870: Advanced Software Engineering

PC:https://community.sparxsystems.com/white-papers/tag/metamodel

Metamodel Architecture

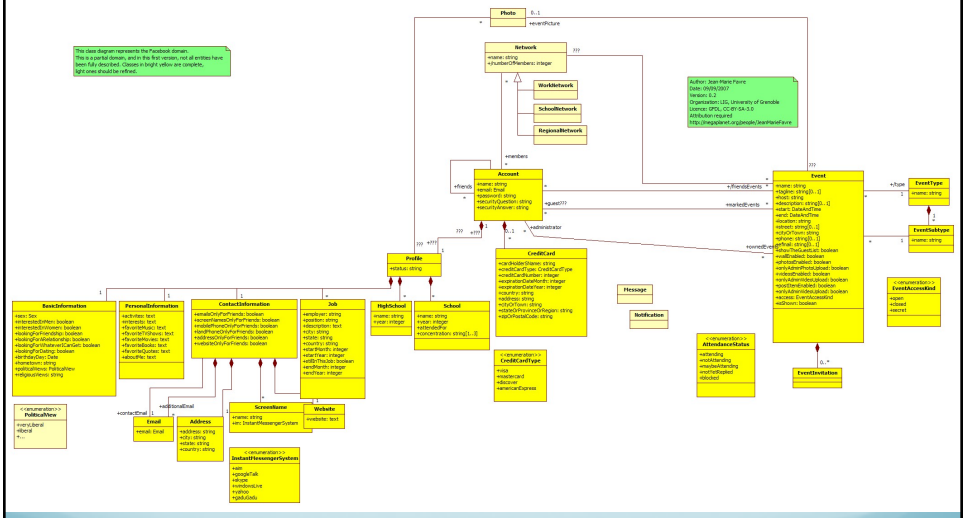


Cheng:CSE 870: Advanced Software Engineering

Fun Metamodels

Cheng:CSE 870: Advanced Software Engineering

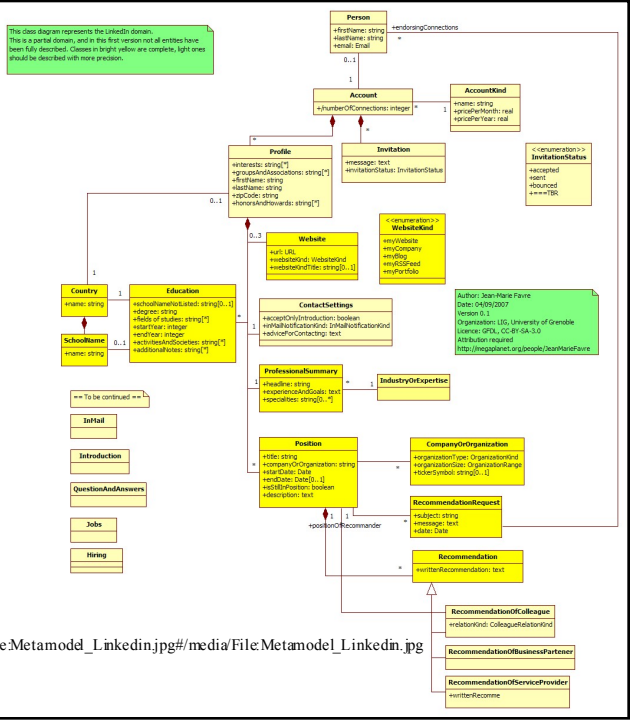
Metamodel for Facebook



Source: https://commons.wikimedia.org/wiki/File:Metamodel_of_Facebook.jpg#/media/File:Metamodel_of_Facebook.jpg

Cheng:CSE 870: Advanced Software Engineering

Meta model for LinkedIn



Source: https://commons.wikimedia.org/wiki/File:Metamodel_Linkedin.jpg#/media/File:Metamodel_Linkedin.jpg

Cheng:CSE 870: Software Engineering

Object Modeling Summary

- Classes
  - Name
  - Attributes
  - Operations
- Associations
  - Roles
  - Link attributes
- Aggregation/Composition
- Inheritance