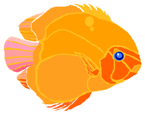
III. Class and Object Diagrams

Classes, Attributes and Operations
Objects and Multi-objects
Generalization and Inheritance
Associations and Multiplicity
Aggregation and Composition
Business Objects and Rules

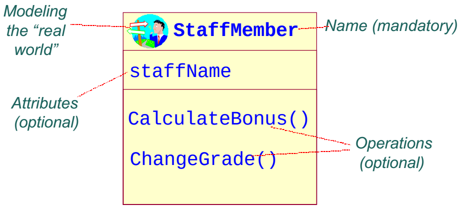


© 2004 John Mylopoulos

Classes

- A class describes a group of objects with
  - ✓ similar properties (attributes),
  - ✓ common behaviour (operations),
  - ✓ common relationships to other objects,
  - ✓ and common meaning ("semantics").
- Finding classes: Listen to the domain experts (…the people who know the domain you are modeling!)

Diagrammatic Notation for Classes



Modeling the "real world"

StaffMember

Name (mandatory)

staffName

Attributes (optional)

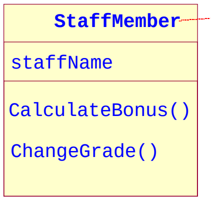
CalculateBonus()

ChangeGrade()

Operations (optional)

System Classes

This is a Java class to be included in the design of the new system



StaffMember

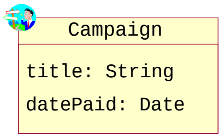
staffName

CalculateBonus()

ChangeGrade()

Attributes

- Each class can have ***attributes*** which represent useful information about instances of a class.
- Each attribute has a ***type***.
- For example, **`Campaign`** has attributes **`title`** and **`datePaid`**.

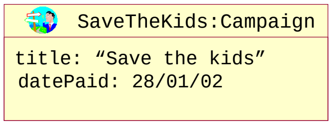


Campaign

title: String

datePaid: Date

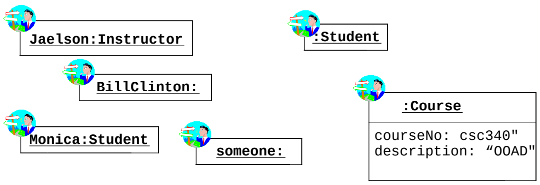
Objects are Class Instances



SaveTheKids:Campaign

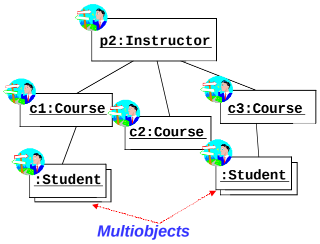
title: "Save the kids"
datePaid: 28/01/02

Object Diagrams



Multiobjects

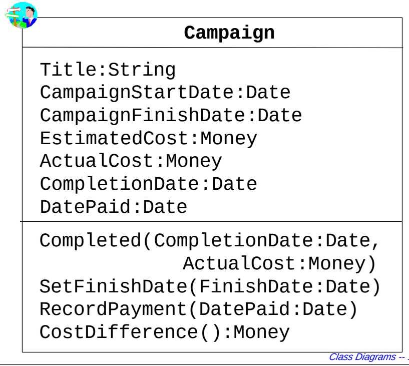
A ***multiobject*** is a set of objects, with an undefined number of elements



Operations

- *Often derived from action verbs in the description of the application.*
- *Operations describe what can be done with the instances of a class.*

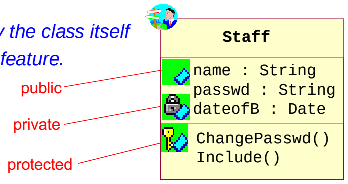
Operations



```
Campaign

Title:String
CampaignStartDate:Date
CampaignFinishDate:Date
EstimatedCost:Money
ActualCost:Money
CompletionDate:Date
DatePaid:Date

Completed(CompletionDate:Date,
              ActualCost:Money)
SetFinishDate(FinishDate:Date)
RecordPayment(DatePaid:Date)
CostDifference():Money
```

Visibility

- *As with Java, attributes and operations can be declared with different visibility modes:*
  - + ***public****: any class can use the feature (attribute or operation);*
  - # ***protected****: any descendant of the class can use the feature;*
  - - ***private****: only the class itself*
  - - *can use the feature.*

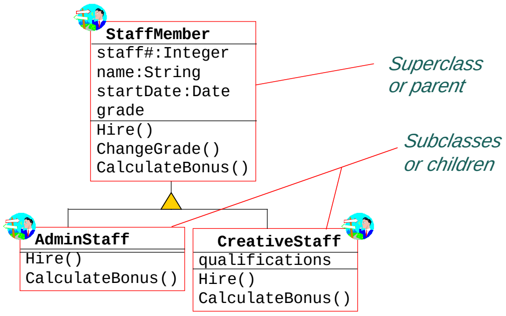


```
Staff

name : String
passwd : String
dateofB : Date

ChangePasswd()
Include()
```

Relationships

- *Classes and objects do not exist in isolation from one another*
- *A relationship represents a connection among things.*
- *In UML, there are different types of relationships:*
  - ✓ *Generalization*
  - ✓ *Association*
    - ✓ *Aggregation*
    - ✓ *Composition*
  - ✓ *…more…*

Generalization


StaffMember
staff#:Integer
name:String
startDate:Date
grade
Hire()
ChangeGrade()
CalculateBonus()

AdminStaff
Hire()
CalculateBonus()

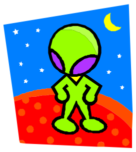
CreativeStaff
qualifications
Hire()
CalculateBonus()

Superclass or parent

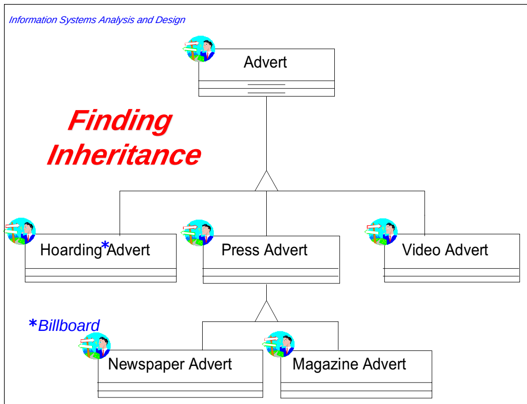
Subclasses or children

Inheritance

- Inheritance of attributes
- Inheritance of operations
- Overriding inherited attributes or operations.



Finding Inheritance


Advert

Hoarding*Advert

Press Advert

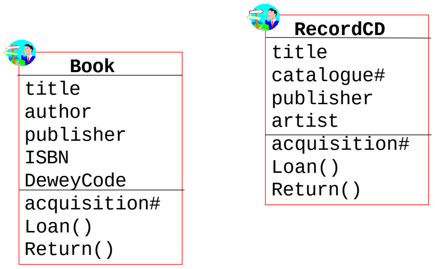
Video Advert

Newspaper Advert

Magazine Advert

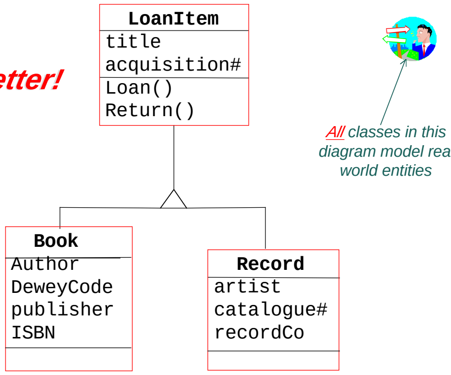
*Billboard

Finding Inheritance, ... Bottom Up


Book
title
author
publisher
ISBN
DeweyCode
acquisition#
Loan()
Return()

RecordCD
title
catalogue#
publisher
artist
acquisition#
Loan()
Return()

...Better!


LoanItem
title
acquisition#
Loan()
Return()

Book
Author
DeweyCode
publisher
ISBN

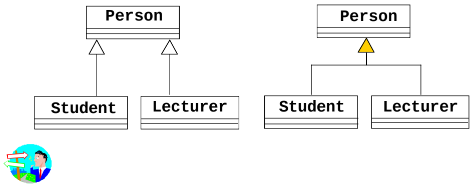
Record
artist
catalogue#
recordCo

All classes in this diagram model real world entities

Generalization Notation

Possibly overlapping
Maria is both **Lecturer** and **Student**

Mutually exclusive
a lecturer can't be a student and vice versa

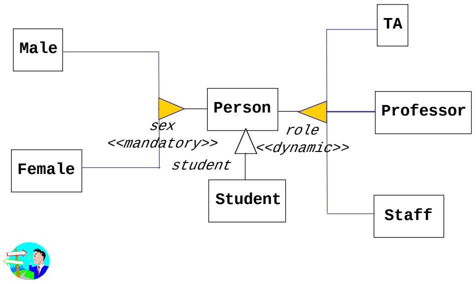

Person
Student
Lecturer

Person
Student
Lecturer

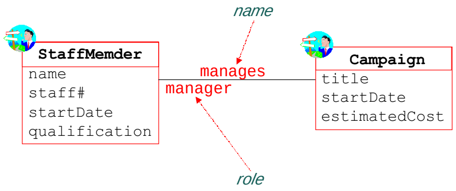
Classification

- *This is the relationship between an object and the classes of which it is an instance.*
- *Traditional object models assume that classification is **single** and **static**.*
- ***Multiple** classification allows an object to be an instance of several classes that are not is-a related to each other; for example,* **`Maria`** *may be an instance of* **`GradStudent`** *and* **`Employee`**.
- ***Dynamic** classification allows an object to change its type during its lifetime.*

Multiple Classification

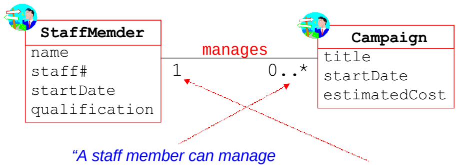


Association Relationships



Association Multiplicity

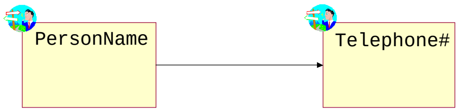
- *How many instances of a class can participate in an association of a particular type?*



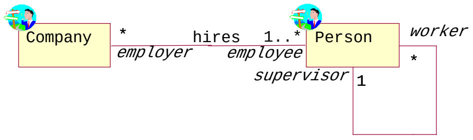
"A staff member can manage zero or more campaigns"

"Each campaign is managed by exactly one staff member"

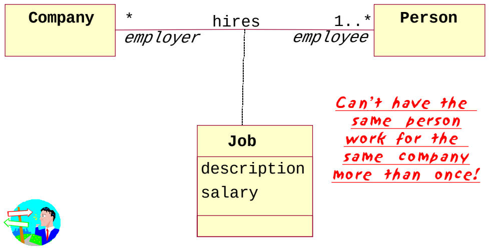
Association Navigation: Uni-Directional Associations



Associations and Roles

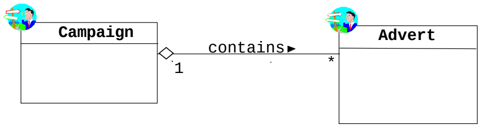


Association Classes



Can't have the same person work for the same company more than once!

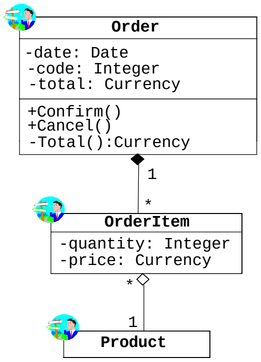
Aggregation Relationship



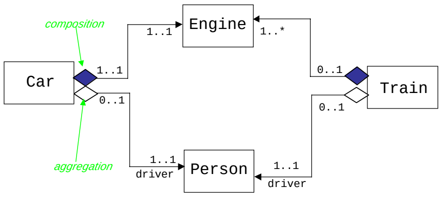
Composition Relationship

- A composition relationship implies strong ownership of the part by the whole.
- For example, the relationship between a person and her head is a composition relationship, and so is the relationship between a car and its engine.
- In a composition relationship, the whole is responsible for the disposition of its parts, i.e. the composite must manage the creation and destruction of its parts.

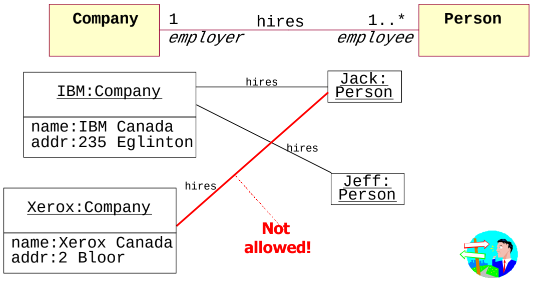
An Example



Another Example



Object Diagrams, Again



Not allowed!

Business Objects and Rules

- Business objects and rules document -- in a structured way -- a class diagram.
- Such a documentation is also called ***data dictionary.***
- Business objects are represented as classes.
- A business rule can be:
  - ✓ an ***integrity constraint*** on the data of the application,
  - ✓ a ***derivation rule***, whereby information can be derived from other information within a class diagram.

Examples of Business Objects

Classes	**Description**	**Attributes**	**Identifier**
EMPLOYEE	Employee working in the company.	Code, Surname, Salary, Age	Code
PROJECT	Company project on which employees are working.	Name, Budget, ReleaseDate	Name
....	....	....	....

Associations	**Description**	Classes **involved**	**Attributes**
MANAGEMENT	Associate a manager with a department.	Employee (0,1), Department (1,1)	
MEMBERSHIP	Associate an employee with a department.	Employee (0,1) Department (1,N)	StartDate
....	....	....	....

Examples of Business Rules

Constraints
(BR1) The manager of a department must belong to that department. **(BR2)** An employee must not have a salary greater than that of the manager of the department to which he or she belongs. **(BR3)** A department of the Rome branch must be managed by an employee with more than 10 years' employment with the company. **(BR4)** An employee who does not belong to a particular department must not participate in any project. ....
Derivations
(BR5) The budget for a project is obtained by multiplying the sum of the salaries of the employees who are working on it by 3. ....

Additional Readings

- *[Booch99] Booch, G. et al. The Unified Modeling Language User Guide, Addison-Wesley, 1999. (Chapters 4, 5, 8, 9, 10.)*
- *[Fowler97] Fowler, M. Analysis Patterns: Reusable Object Models, Addison-Wesley, 1997.*