CSIT882 Data Management Systems

Design Patterns

Subject coordinator: Chen Chen

School of Computing and Information Technology - University of Wollongong

Disclaimer: subject materials are sourced from previous offerings of CSIT882

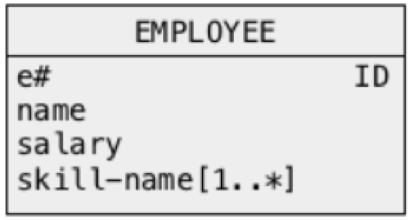
Design Patterns

- Outline

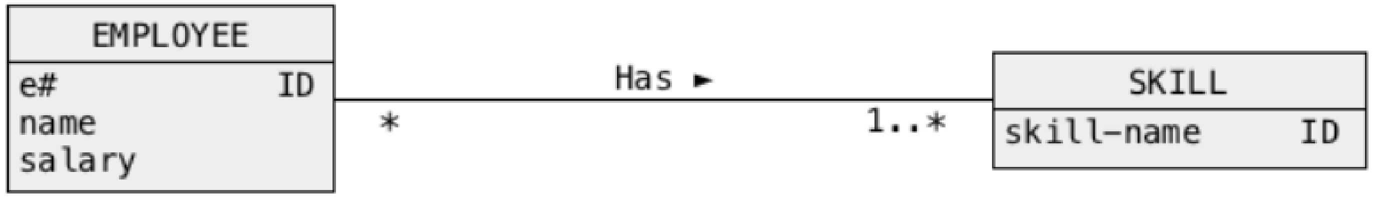
- Reification Transformation
- Design Patterns

Reification Transformation

- Reification is the promotion of something that is not an object into an object
- Employees are described by a unique number, name, salary, and names of skills possessed

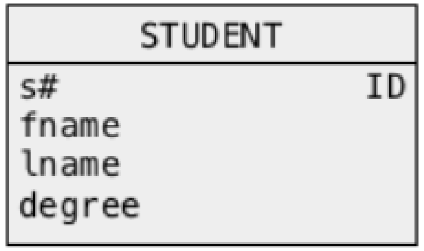


- Employees have skills

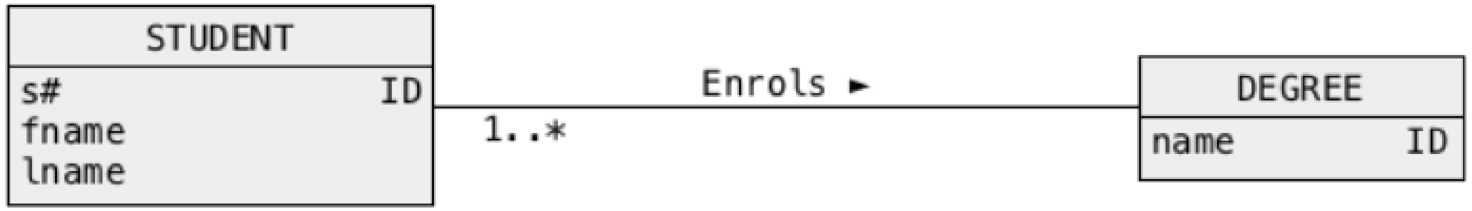


Reification Transformation

- Students are described by a student number, first name, last name, address, and name of degree enrolled



- Students enrol degrees



Design Patterns

- Outline

- Reification Transformation
- Design Patterns
  - Simple tree pattern
  - Complex tree pattern
  - Item description pattern
  - Qualification pattern
  - Homomorphism pattern

Why are patterns important

- Pattern: a model fragment that is profound and recurring
- **Enriched modeling language**. Patterns provide a higher level of building blocks than modeling primitives. Patterns are prototypical modeling fragments that distill the knowledge of experts.
- **Improved documentation**. Patterns offer standard forms that improve modeling uniformity.
- **Reduced modeling difficulty**. Many developers find modeling difficult because of the intrinsic abstraction. Patterns are all about abstraction and give developers a better place to start.
- **Faster modeling**. Developers do not have to create everything from scratch and can build on the accomplishments of others.
- **Better models**. Patterns reduce mistakes and rework. Carefully considered patterns are more likely to be correct and robust than an untested, custom solution.

Drawbacks of Patterns

- You cannot build a model by just combining patterns. Typically you will use only a few patterns, but they often embody key insights.

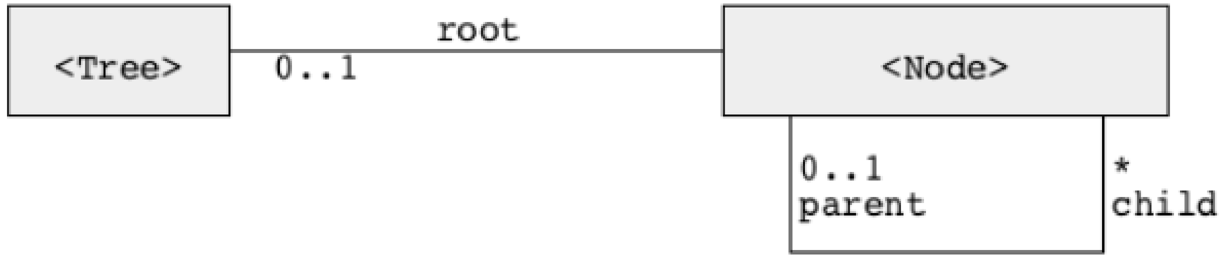
- It can be difficult to find a pattern, especially if your idea is ill-formed.

- Patterns are an advanced topic and can be difficult to understand.

- There has been a real effort in the literature to cross reference other work and build on it. However, inconsistencies still happen.

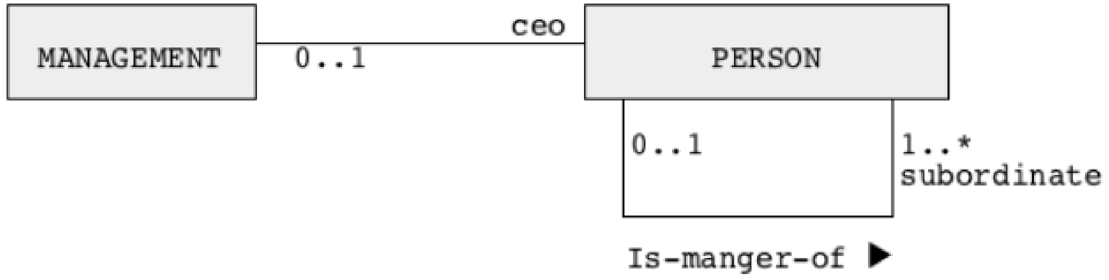
Simple Tree Pattern

- Tree structure has one root node
- Parent node has many child nodes
- Child node has at most one parent node

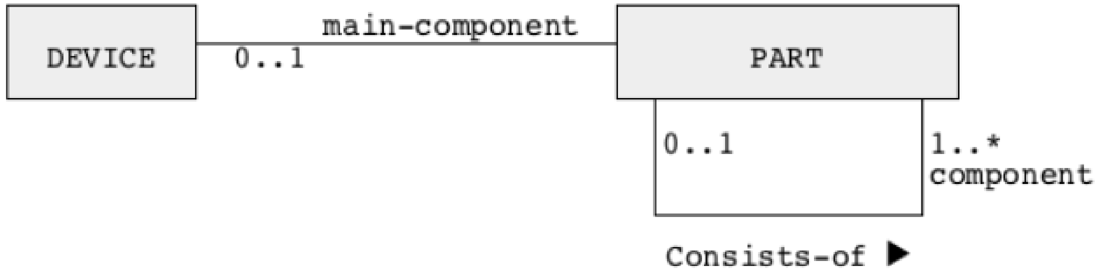


Simple tree pattern

- Management structure has one ceo. Manager has many subordinates. Subordinate has at most one manager.

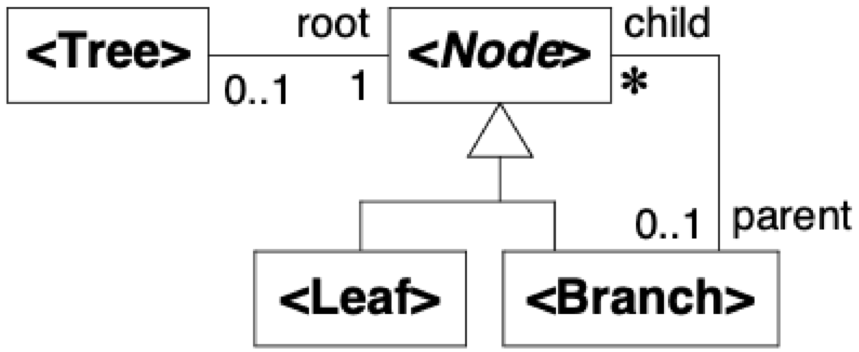


- Device looks like one part. Part consists of many parts. Part belongs to at most one other part.



Complex tree pattern

- Tree structure has one root node
- Leaf node is a node
- Branch node is a node
- Branch node has many child nodes
- Child node has at most one parent node which is a branch node



Complex tree pattern

File hierarchy has one root file

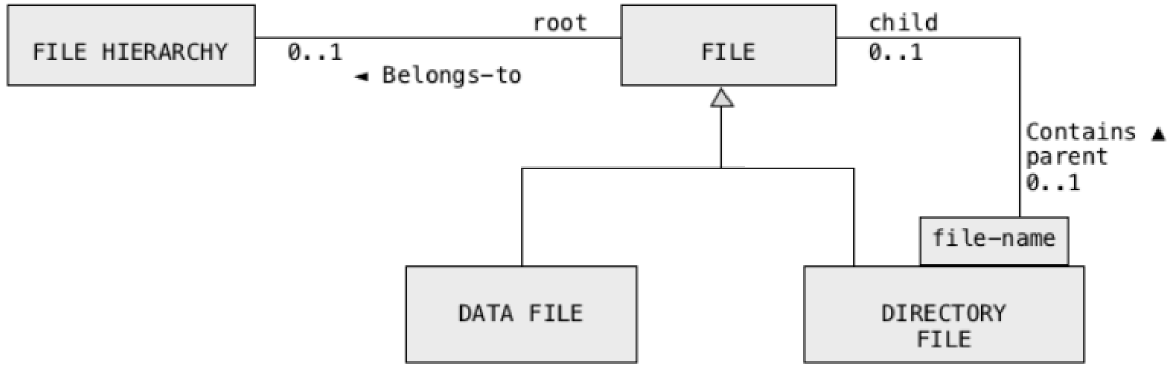
Data file is a file

Directory file is a file

Directory file contains many child files

Child file belongs to at most one directory file

File name is a local identifier of file in a directory file



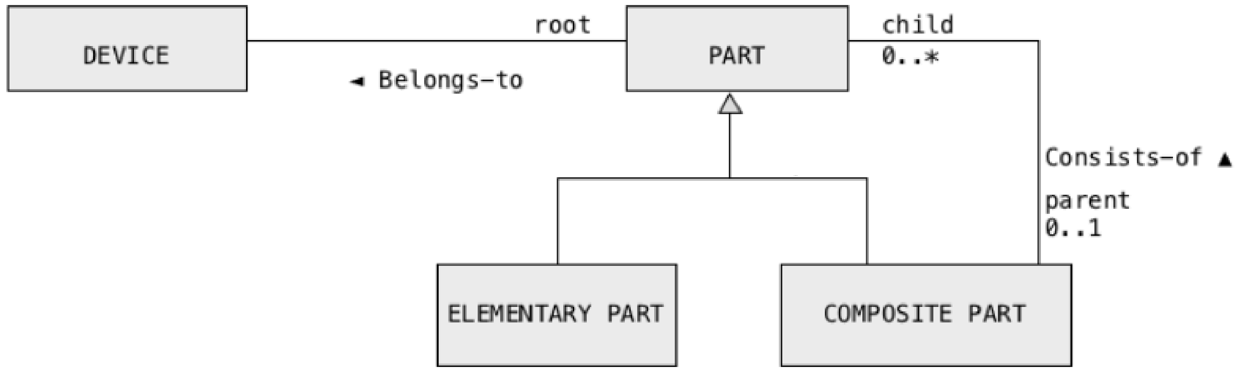
Complex tree pattern

A device has one part which is either elementary or composite

Elementary part is a part

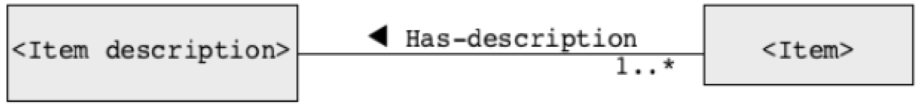
Composite part is a part

Composite part consists of many parts

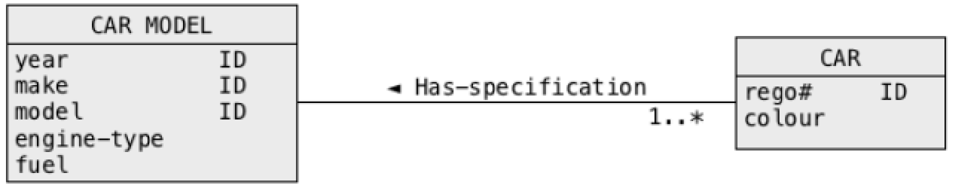


Item description pattern

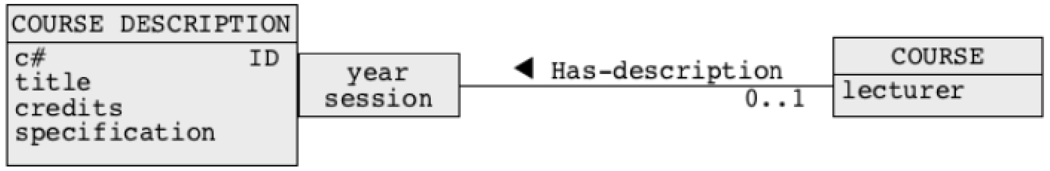
Item has item description



Car has assembly specification

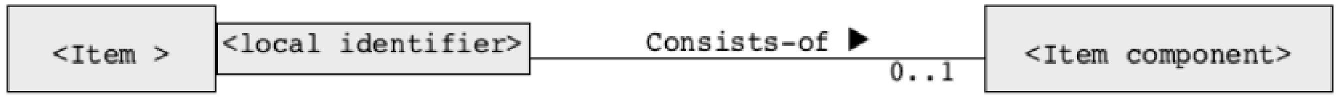


Course has description

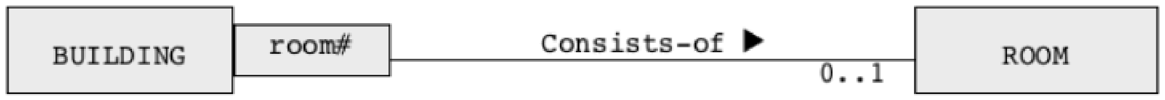


Qualification pattern

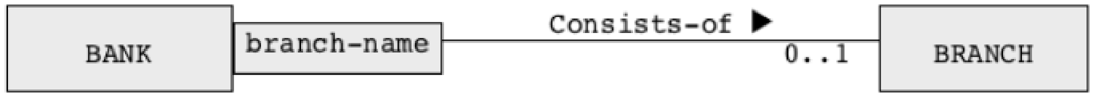
Attribute is a local identifier.



Room number is a local identifier of a room in a building.



Branch name is a local identifier of a branch in a bank



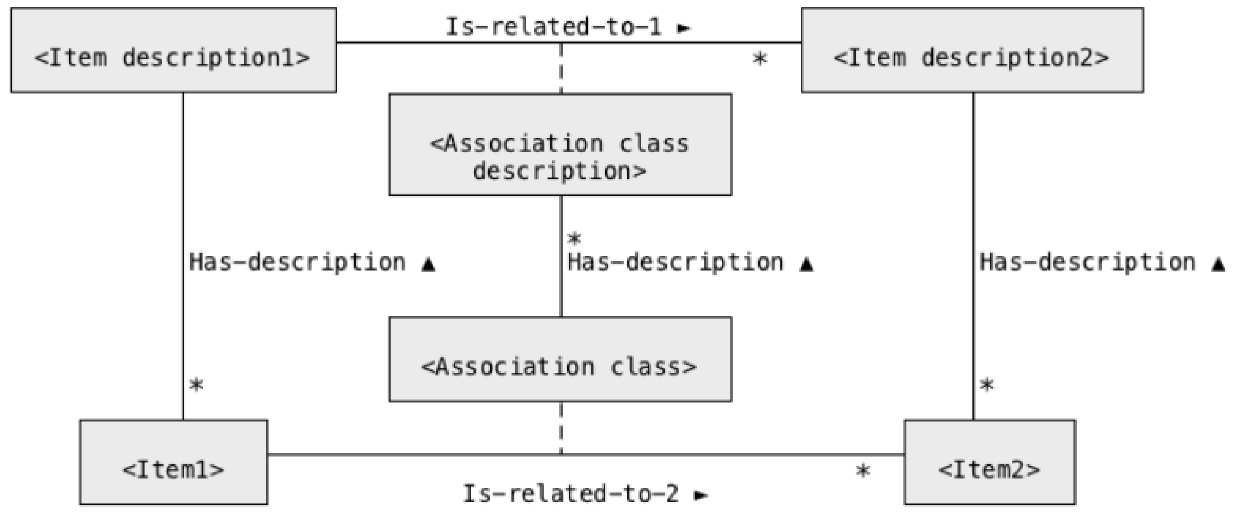
Homomorphism pattern

Item-1 has description-1

Item-2 has description-2

Item-1 is related to item-2

Description-1 is related to description-2



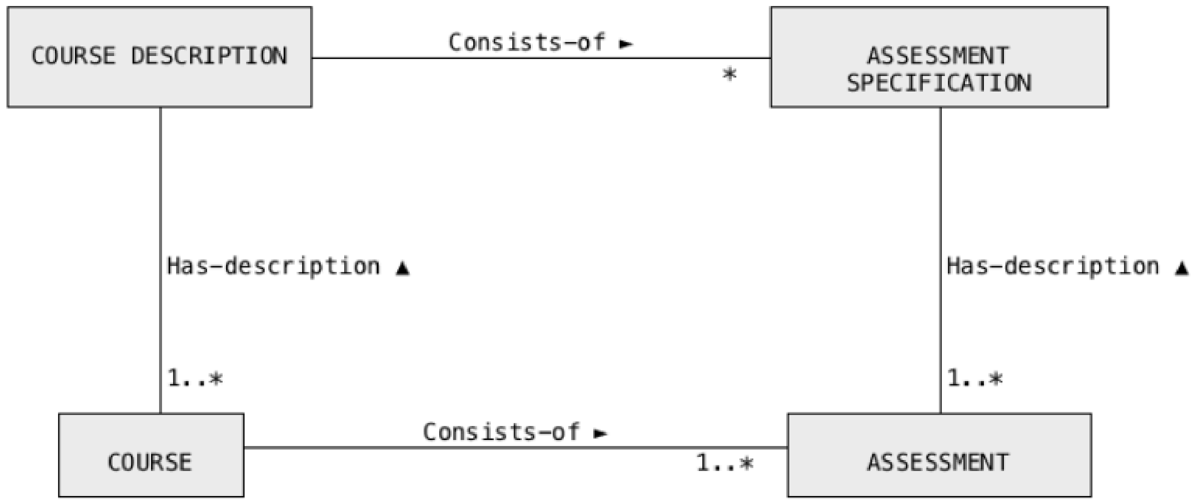
Homomorphism pattern

Course description consists of assessment specification

Course has description

Assessment has assessment specification

Course consists of assessment



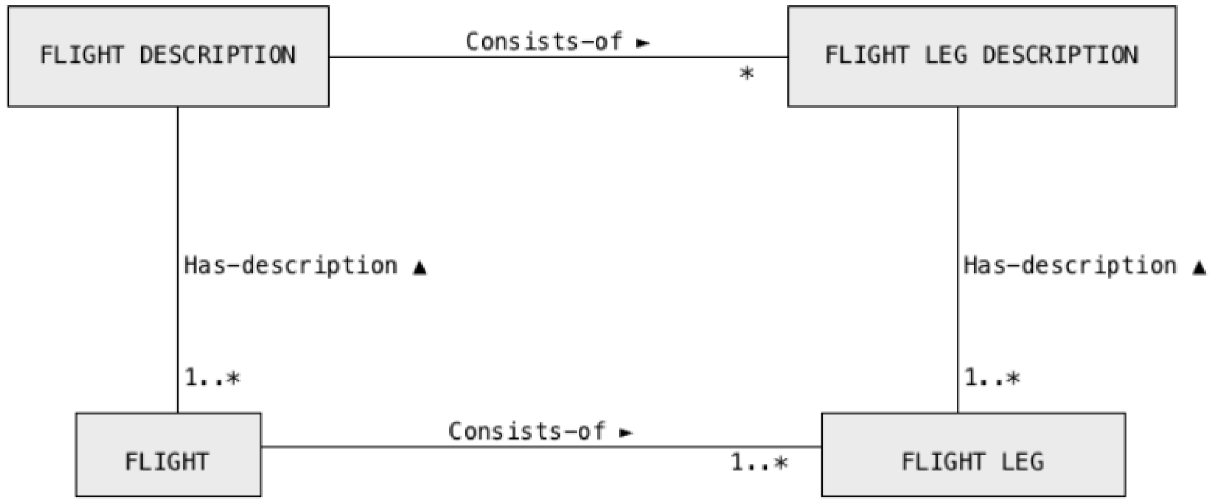
Homomorphism pattern

Flight description consists of flight leg descriptions

Flight has description

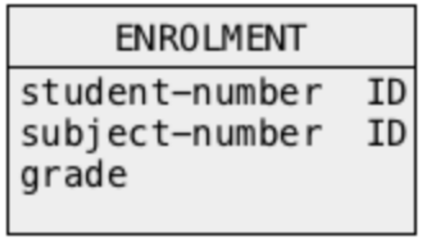
Flight leg has Flight leg description

Flight consists of flight legs



Exercise – Logical design

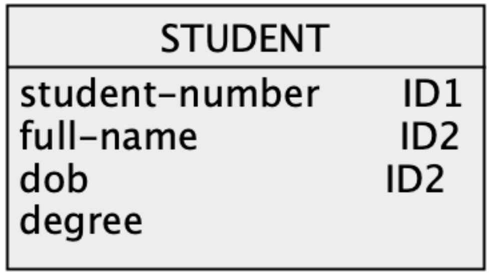
- A process of logical design transforms a class of objects



- into:

Exercise – Logical design

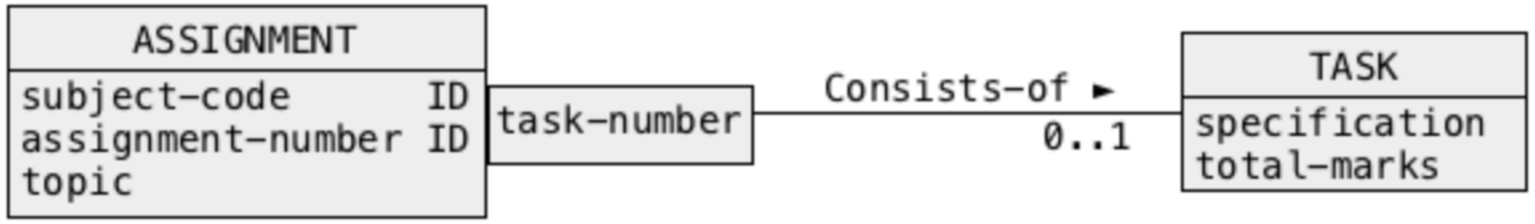
- A process of logical design transforms a class of objects



- into:

Exercise – Logical design

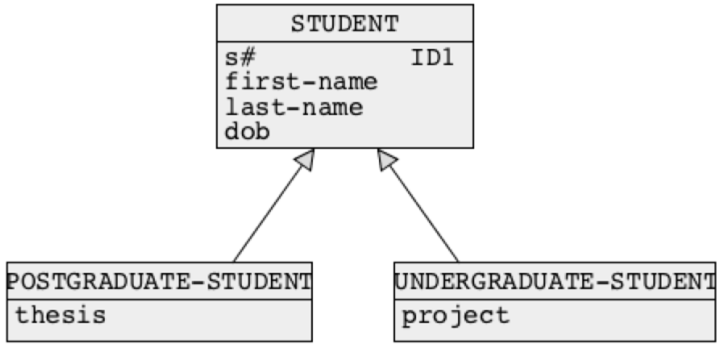
- A process of logical design transforms a class of objects



- into

Exercise – Logical design

- A process of logical design transforms a class of objects



- Into (using **superset** method)

References

- Blaha M., Pattern of Data Modelling, CRC Press, 2010, chapters 1-7
- Blaha M., PREMERLANI W., Object-Oriented modelling and Design for Database Applications, Prentice Hall, 1998, chapter 4