EECS4314 week 2

Jerry Wu

2024-01-17

Contents

Lecture 1

UML overview

Short history of analysis and design notations

"STD is state transition diagram, not the other STD" - H.V. Pham 2024

•

"How many graphics cards do you want? Yes." - H.V. Pham 2024

Class diagrams review

A class is a description of a set of objects that share the same attributes, operations, relationships, and semantics. Graphically, a class is drawn as a rectangle including its name, attributes, and operations in their own separate (3) compartments:

- Class name (eg. Person, Car, etc.)
- Attributes (instance variables with their respective types)

  $+ \equiv$ public

  $\# \equiv$ protected

  $- \equiv$ private

  $/ \equiv$ derived

- Operations (getters, setters, misc. methods) and their respective signatures
  - eg. `newEntry(n:Name, a:Address, p:PhoneNumber, d:Description)`

Class relationships

In UML, pbject interconnections (logical or physical), are modelled as relationships

- Dependencies - a semantic relationship between two or more elements
- Generalization - connection from subclass to superclass. Denotes inheritance of attributes and behaviors from the superclass to the subclass and indicates a specialization in the subclass of the more general superclass
- Association - if two classes in a model need to commmunicate with each other, but are not dependent on each other, then there will be a link denoted between the two classes. Dual associations are possible
  - Aggregation - specifies a whole part relationship between an aggregate (whole) and a constituent part where the part can exist independently from the aggregate. Denoted by a hollow diamond. For example (car has an owner and a manufacturer, but owner and manufacturer and owner can exist without car)
  - Composition - Same as aggregation, but is a strong ownership relationship, so the constituent parts have to exist for the composite to exist. Denoted by a filled diamond.
- Interfaces - a named set of operations (no instance variables) that speifies the behavior of objects without showing their inner structure. It can be rendered in the model by a one or two compartment rectangle with the **stereotype** `<<interface>>` above the interface name
  - Interface relationships are called **realization**, which means a concrete class `implements` an interface
- Enumeration - a user defined data type that consists of a name and ordered list of enumeration literals. Denoted using `<<enumeration>>`
- Exceptions - can be customized like any other class, denoted using `<<exception>>`

Package

A package is a container like element for organizing other elements into groups. Packages can contain classes, other packages, and diagrams. They can provide controlled access between classes in different packages. Denoted using a folder like shape.

- Packages can depend/relate on/to other packages just like classes

Component diagrams

-

Dynamic modelling using UML

Use cases

- A use case specifies the behavior of a system or part of a system
- A description of a set of sequences of actions, including its variants. Always produces an observable result of the actor(user)
- An **actor** is either a user of the system, or another system, subclass, or class