Software Design with UML Class Diagrams

Object Oriented Software Engineering

Lecture 2

Dr. Graham McDonald

graham.mcdonald@glasgow.ac.uk

Outline

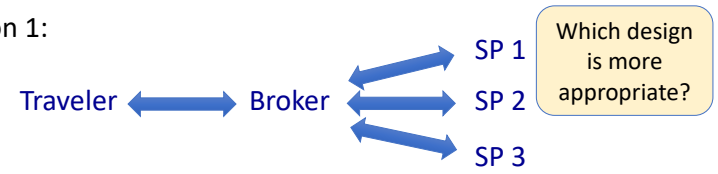
- Software Design
- What is UML?
- What is a UML class diagram?
  - What kind of information goes into it?
  - How do I create it?
  - When should I create it?
- Examples
- Tools

Software Design

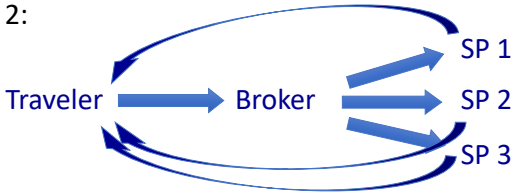
- Design: specifying the structure of a software system and its functions (behaviour)
  - Its an opportunity to get insights on design alternatives to make appropriate design choice
  - You can also evaluate the extent to which the system complies with end user expectations

Does Design Matter?

- Suppose you are tasked with designing a taxi broker service, how could you design the system?
- Option 1:



- Option 2:



Does Design Matter?

- When do you think it is most suitable to design a software system?

1. Design while implementing the code (i.e. think about design using program code as a language for system modeling)- Then automatically generate model representations (such as UML class diagram)

2. Before implementation using an abstract modeling language such as UML

1. After implementation (you generate the design automatically from the code)

Software Design

- A transition from "what" the system must do, to "how" the system will do it
  - What classes will we need to implement a system that meets our requirements?
  - What fields and methods will each class have?
  - How will the classes interact with each other?

- The outcome of a software design activity is a domain model

Object-Oriented Analysis

Domain Model:

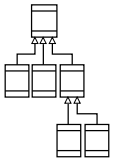
- A conceptual model of the domain that incorporates both behavior and data.
  - A formal/semiformal/informal representation of a domain with concepts, roles, datatypes, individuals, and associated rules of interaction.

What is UML

- Diagrammatic representations of an OO system
  - Programming languages are not abstract enough for OO design
  - UML is an open standard; lots of companies use it

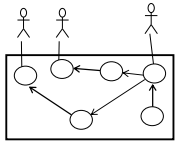
- What is legal UML?
  - A descriptive language: rigid formal syntax (like programming)
  - A prescriptive language: shaped by usage and convention
  - It's okay to omit things from UML diagrams if they are not needed

UML Modelling Notations



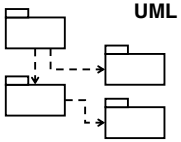
UML Class Diagrams

information structure
relationships between data items modular structure for the system



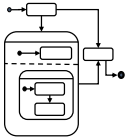
Use Cases

user's view Lists functions visual overview of the main requirements



UML Package Diagrams

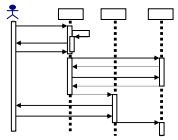
Overall architecture
Dependencies between components



(UML) Statecharts

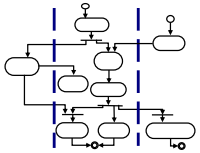
responses to events

dynamic behavior event ordering, reachability, deadlock, etc



UML Sequence Diagrams

individual scenario interactions between users and system
Sequence of messages



Activity diagrams

business processes;
concurrency and synchronization;
dependencies between tasks;