CS 315 - Oct 12, 2015

Chapter 14: Design

- Analysis Model to Design Model
    - Usage based design is mapped onto a structured, well-defined model

- Design and Abstraction
    - Classical Design Activities
        - Architectural Design
            - Input: Specifications
            - Output: Modular Decomposition
            - Architectural Styles
                - Data/database center
                    - Clients connect to central repository of information

                - Pipe and Filter
                    - Produce an output with a given input
                    - Examples
                        - Image Manipulation
                        - Encryption
                        - Transfer protocols

                    - Special Case: Batch Sequential
                        - Everything happens a certain sequence in a straight line

                - Layered
                    - Layers of functionality above a core layer
                    - Operating Systems follow this model
                    - Layers only have to interact with the layer below them

                - Call and Return
                    - Traditional programming
                    - Break computations into tiny pieces, calculate results, then combine them

                - Service Oriented
                    - Good for distributed business systems
                    - Interface/Client tier, Business Logic/Application tier, database tier

        - Detailed Design
            - Each module is designed

      - Specific algorithms, data structures

- Object Oriented Design
  - Aim
    - Design the product in terms of the classes extracted during object oriented analysis
  - Two Steps
    1. Complete the class diagram
       - Determine the formats of the attributes
         - To minimize rework, **never** add an item to a UML diagram until strictly necessary
       - Principle A:
         - Information Hiding
       - Principle B:
         - If an operation is invoked by many clients of an object, assign the mothod to the object, not the clients
       - Principle C:
         - Responsibility-Driven Design
         - What actions is this object responsible for?
         - What information does this object share?
       - Assign each method, eitehr to a class or to a client that sends a message to an object of that class
    2. Perform the detailed design
  - Package Diagram
    - Package is a group of related classes
    - Primary mechanism to indicate encapsulation in UML