High-level Design

Overview

- What is software architecture?
- Classic architecture styles
- UML Package Diagram
- How to do architecture design?

What is Software Architecture?

- "The architecture of a system is comprehensive framework that describes its form and structure -- its components and how they fit together"

--Jerrold Grochow

What is Architectural Design?

- Design overall shape & structure of system
  - the components
  - their externally visible properties
  - their relationships
- Goal: choose architecture to reduce risks in SW construction & meet requirements

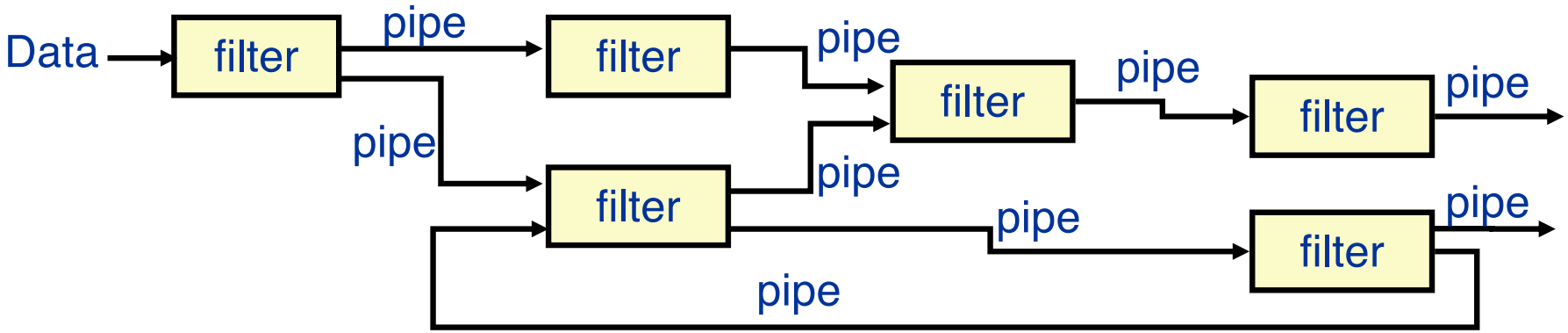
SW Architectural Styles

- Architecture composed of
  - Set of components
  - Set of connectors between them
    - Communication, co-ordination, co-operation
  - Constraints
    - How can components be integrated?
  - Semantic models
    - What are the overall properties based on understanding of individual component properties?

Architecture Patterns

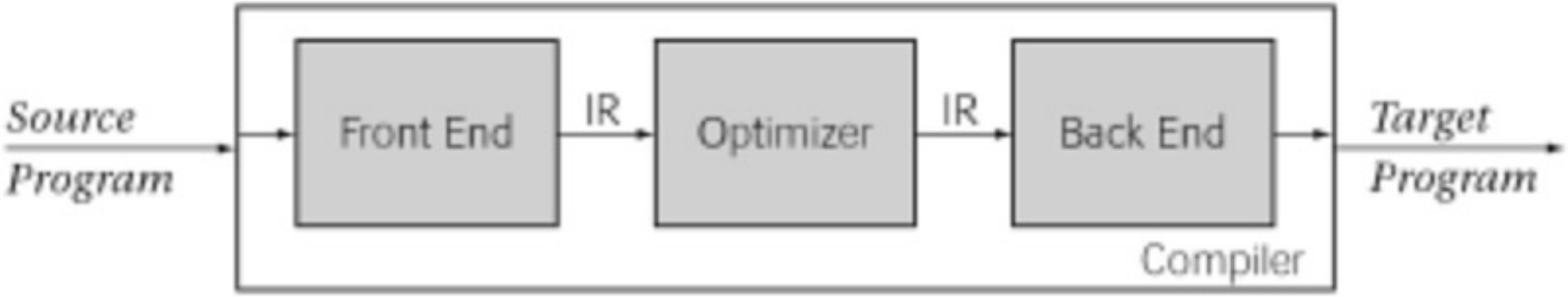
- Common program structures
  - Pipe & Filter Architecture
  - Event-based Architecture
  - Layered Architecture

Pipe & Filter Architecture

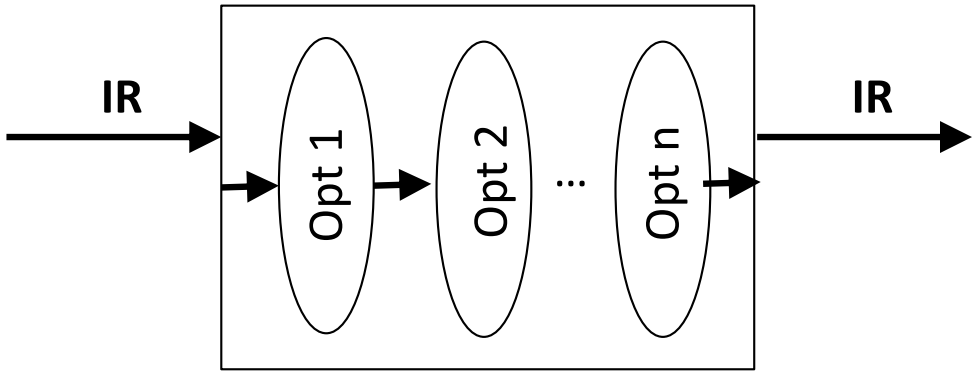


- A pipeline contains a chain of data processing elements
  - The output of each element is the input of the next element
  - Usually some amount of buffering is provided between consecutive elements

Example: Optimizing Compiler



Compiler Structure



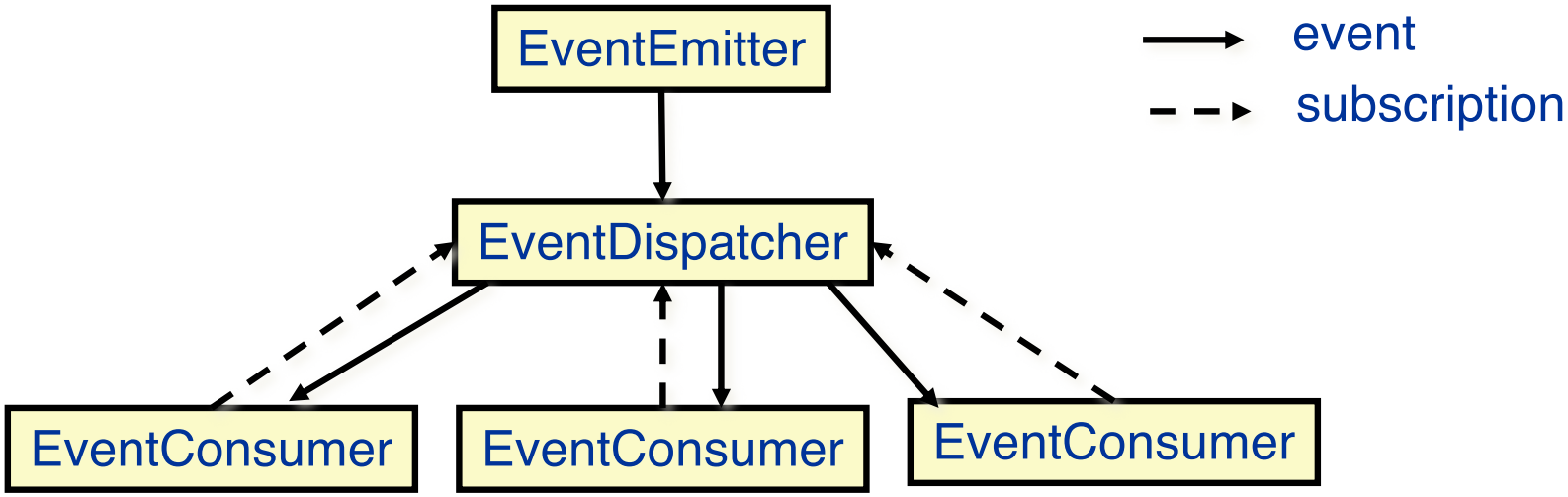
Compiler Optimization

[Engineering a Compiler, K. D. Cooper, L. Torczon]

Pros and Cons

- Other examples
  - UNIX pipes, signal processors
- Pros
  - Easy to add or remove filters
  - Filter pipelines perform multiple operations concurrently
- Cons
  - Hard to handle errors
  - May need encoding/decoding of input/output

Event-based Architecture



- Promotes the production, detection, consumption of, and reaction to events
- More like event-driven programming

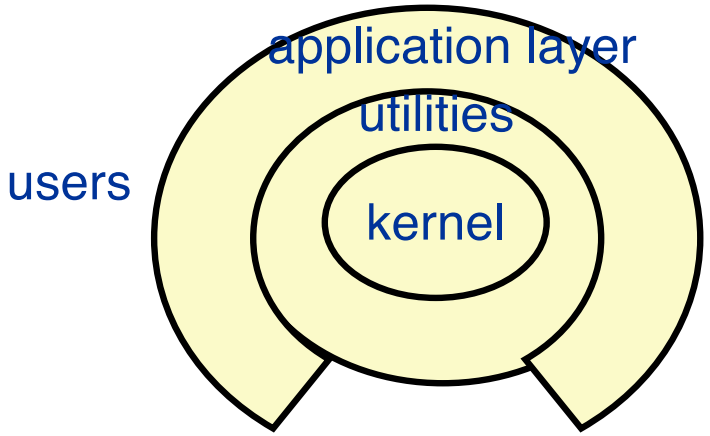
Example: GUI


Please Enter Data...
accountNumber
firstName
lastName
phone
( ) -
balance
OK
Cancel

Pros and Cons

- Other examples:
  - Breakpoint debuggers, phone apps, robotics
- Pros
  - Anonymous handlers of events
  - Support reuse and evolution, new consumers easy to add
- Cons
  - Components have no control over order of execution

Layered/Tiered Architecture

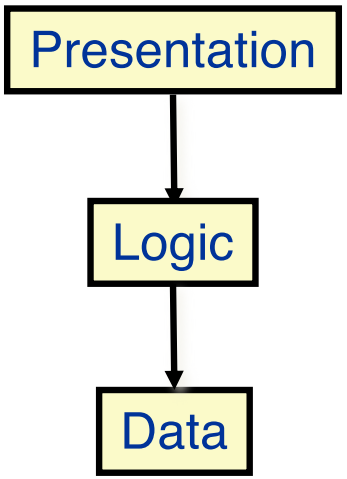


- Multiple layers are defined to allocate responsibilities of a software product
- The communication between layers is hierarchical
- Examples: OS, network protocols

Variant architectures

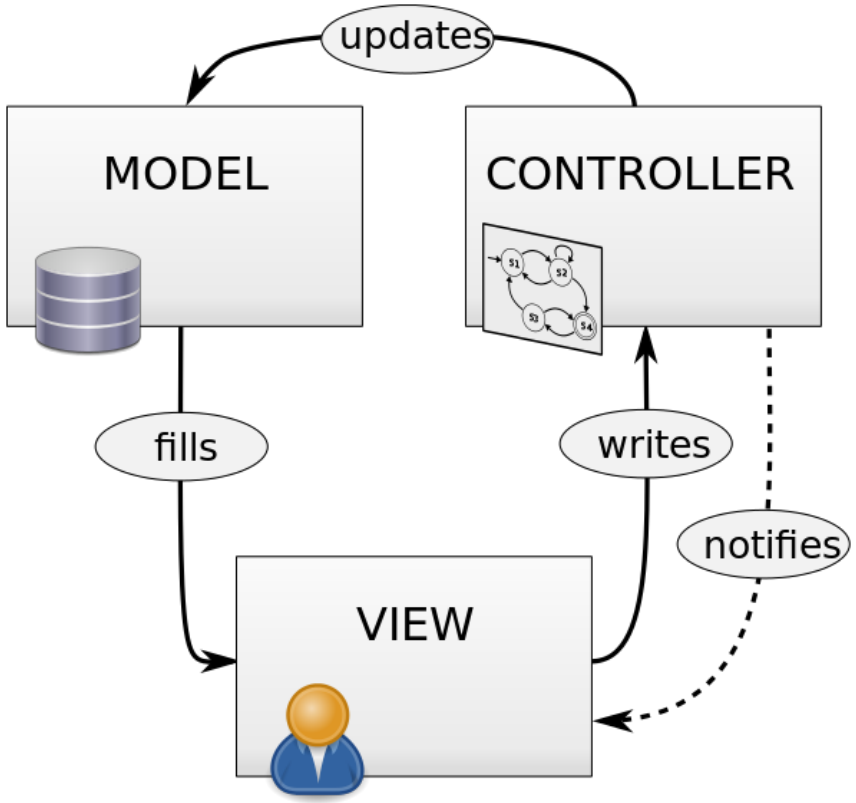
- 2-layer architecture
  - Client-Server Architecture
- 3-layer architecture
  - Model-View-Controller

3-layer Architecture



- Presentation: UI to interact with users
- Logic: coordinate applications and perform calculations
- Data: store and retrieve information as needed

Model-View-Controller



Design of Finite State Machine Drawing Tool

https://commons.wikimedia.org/wiki/File:MVC_Diagram_(Model-View-Controller).svg

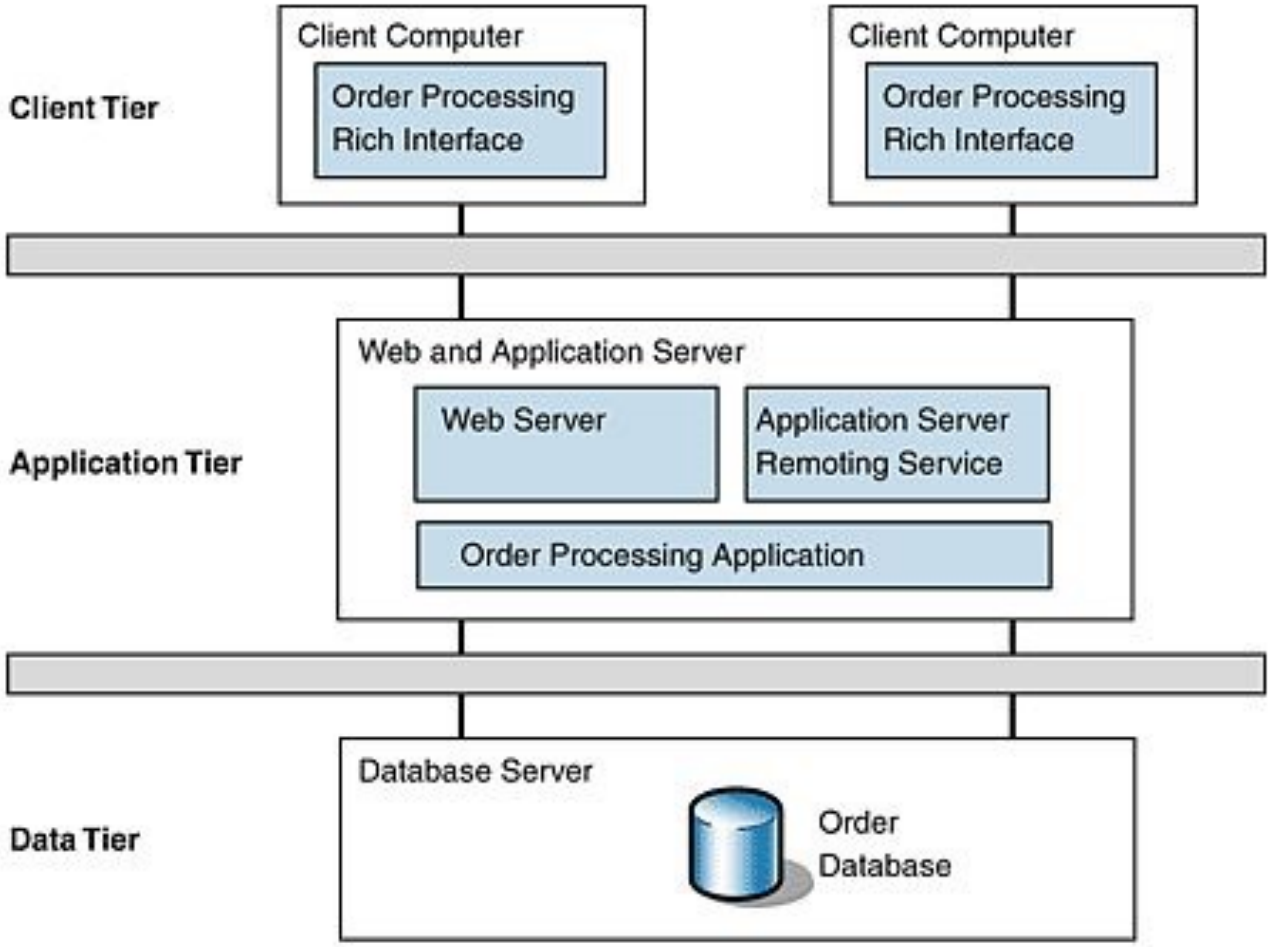
Key Points about MVC

- View layer should not handle system events
- Controller layer has the application logic to handle events
- Model layer only respond to data operation

3 layer: Pros and Cons

- Pros
  - Clear separate concerns
    - Easy to develop, change & reuse
- Cons
  - Hard to maintain when changes in one layer can affect other layers

Example: Online Ordering System



http://www.cardisoft.gr/frontend/article.php?aid=87&cid=96

Layered Architecture: Pros and Cons

- Pros
  - Support increasing levels of abstraction during design
  - Support reuse and enhancement
- Cons
  - The performance may degrade
  - Hard to maintain

How to Do Architecture Design?

- When decomposing a system into subsystems, take into consideration
  - how subsystems share data
    - data-centric or data-distributed
  - how control flows between subsystems
    - as scheduled or event-driven
  - how they interact with each other
    - via data or via method calls