SFWR ENG 3A04: Software Design II

Dr. Ridha Khedri

Department of Computing and Software, McMaster University
Canada L8S 4L7, Hamilton, Ontario

Term 2

Acknowledgments: Material based on *Software Architecture Design* by Tao et al. (Chapter 9)

Outline of Part I

1. Questions?

Outline of Part II

2. Overview

3. Model-View-Controller
   - MVC-I
   - MVC-II

4. Presentation-Abstraction-Control (PAC) Architecture

Part I

Review of Previous Lecture

Overview

Model-View-Controller

Presentation-Abstraction-Control (PAC) Architecture

Part II

Today's Lecture

Interaction Oriented Software Architecture

Overview

- More and more software applications that involve user input and output interactions
- We focus on the software architecture that best supports user interaction
- Interaction oriented software architecture decomposes the system into three major partitions
  - Data module (provides the data abstraction& core business logic)
  - Flow control module (determines the flow controls, view selections, communications between modules, job dispatching, and certain data initializations and configurations)
  - View presentation module (responsible for visual or audio data output presentation)

Interaction Oriented Software Architecture

Overview

- This architecture allows the separation of user interactions from data abstraction and business data processing
- Allows multiple views for a same data set
- Even for a specific view presentation, the interfaces may need to change very often (the loose coupling between data abstractions and its presentations is very helpful)
- A control module plays a central role that mediates the data module and view presentation modules
- All three modules may be completely connected

Interaction Oriented Software Architecture

Overview

- There are two categories of interaction oriented architecture:
  - Presentation-Abstraction-Control (PAC)
  - Model-View-Controller (MVC).
- They are different in their flow controls and structure organization
- The MVC does not have a clear hierarchical structure and all three modules are connected together

Interaction Oriented Software Architecture

Overview

- The PAC is an agent based hierarchical architecture
  - The system is decomposed into many cooperating agents
  - Each agent has three components (Presentation, Abstraction, and Control)
  - The Control component in each agent is in charge of communications with other agents
  - The top-level agent provides core data and business logics
  - The bottom level agents provide detailed specific data and presentations
  - A middle level agent may play a role of coordinator of low-level agents
  - There are no direct connections between Abstraction component and Presentation component in each agent

Interaction Oriented Software Architecture

Model-View-Controller

- Most of Web developers are familiar with MVC architecture
  - Widely adopted for Web server site interactive application design such as online shopping, online survey, online student registration, etc.
- MVC architecture is specifically used in applications where user interfaces are prone to data changes all the time
- MVC architecture typically supports "look and feel" features in GUI application
- The Java Swing components and Java Swing layout managers are designed in MVC architecture

Interaction Oriented Software Architecture
Model-View-Controller

Summary:

- The Controller
  - manages the user input requests
  - controls the sequence of user interactions
  - selects desired views for output displays
  - manages all initialization, instantiations, and registrations of other modules in the MVC system
- The Model module
  - provides all core functional services and encapsulates all data details
  - does NOT depend on other modules, and it does not know which views are registered with or attached to it
- The View module
  - is responsible for displaying the data provided by the Model module and updating the interfaces whenever the data changes

Interaction Oriented Software Architecture
Model-View-Controller (MVC-I)

- The MVC-I is a simple version of MVC architecture
- The system is simply decomposed into two sub-systems:
  - Controller/View
    - It handles input and output processing and their interfaces
    - It registers with (attaches to) data module
  - Model
    - It copes with all core functionality and data
    - It notifies the Controller-View module of any data changes in the Model module

Interaction Oriented Software Architecture
Model-View-Controller (MVC-I)

- The connection between the Controller/View and the Model can be designed in a pattern of subscribe/notify
- The Controller/View subscribes the Model and the Model notifies the Controller/View of any changes
- The Controller/View is an observer to the data in the Model of MVC
- Read the example given in Chaptre 9, Section 9.2.1 to see how MVC-I architecture concretely works

Interaction Oriented Software Architecture

Model-View-Controller (MVC-I)

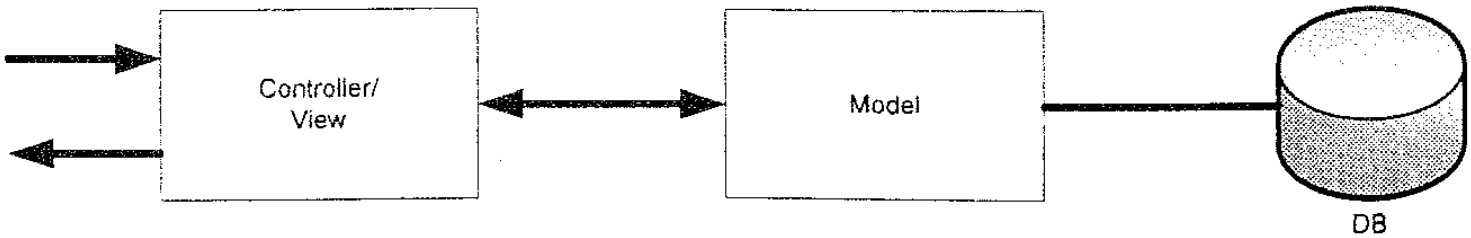

Figure: MVC-I architecture

Interaction Oriented Software Architecture
Model-View-Controller (MVC-II)

MVC-II architecture

- The Model module provides all core functionality and data supported by database (Same as MVC-I)
- The View module displays the data from the Model module
- The Controller module
  - It takes input requests, validates input data, initiates the Model and the View and their connection
  - It dispatches tasks

Interaction Oriented Software Architecture
Model-View-Controller (MVC-II)

- The Controller and the View register with the Model module
- Whenever the data in the Model module is changed the View module and the Controller module are notified
- Comparison to MVCI
  - In both MVC-I and MVC-II, Model module plays an active role
  - In MVC-II architecture, the View module and the Controller module are completely separated

Interaction Oriented Software Architecture

Model-View-Controller (MVC-I)

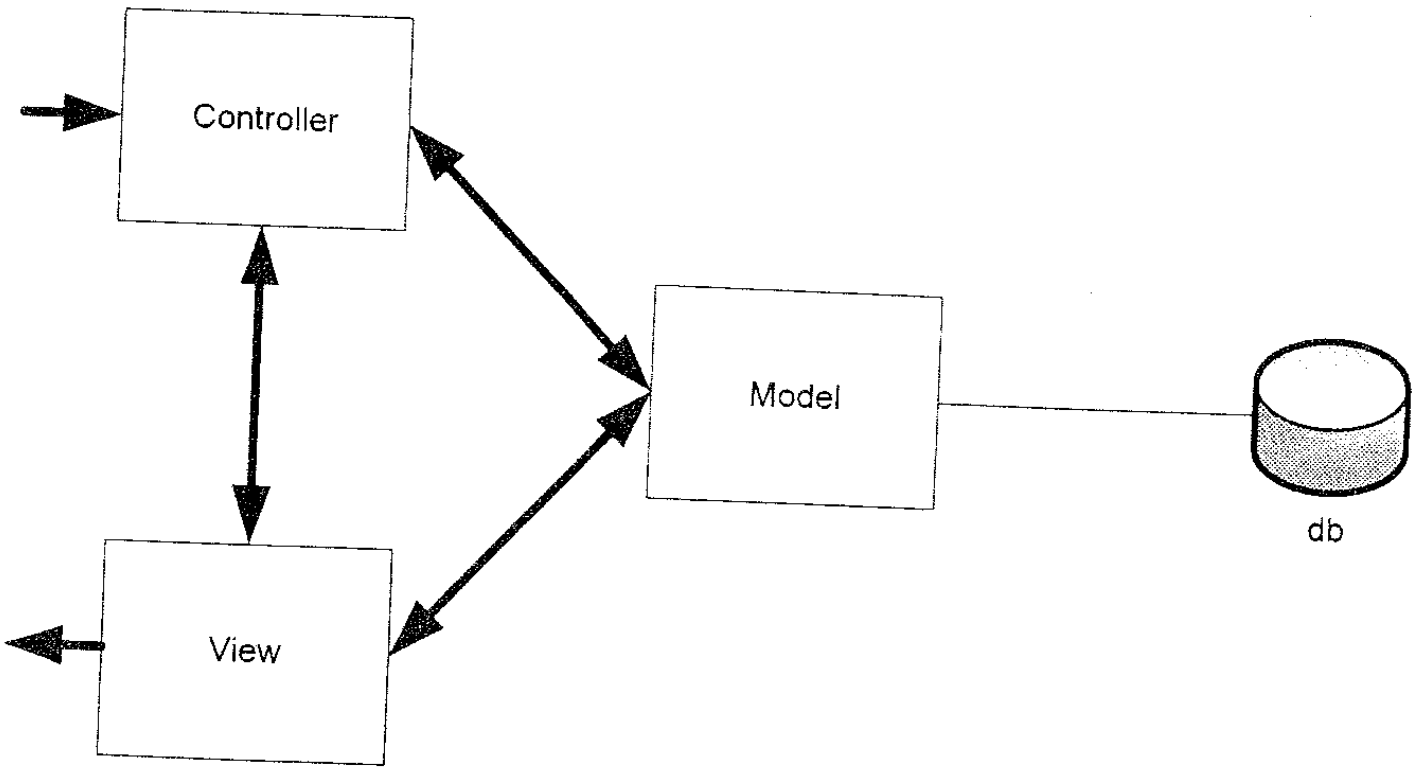

Figure: MVC-II architecture

Interaction Oriented Software Architecture

Model-View-Controller

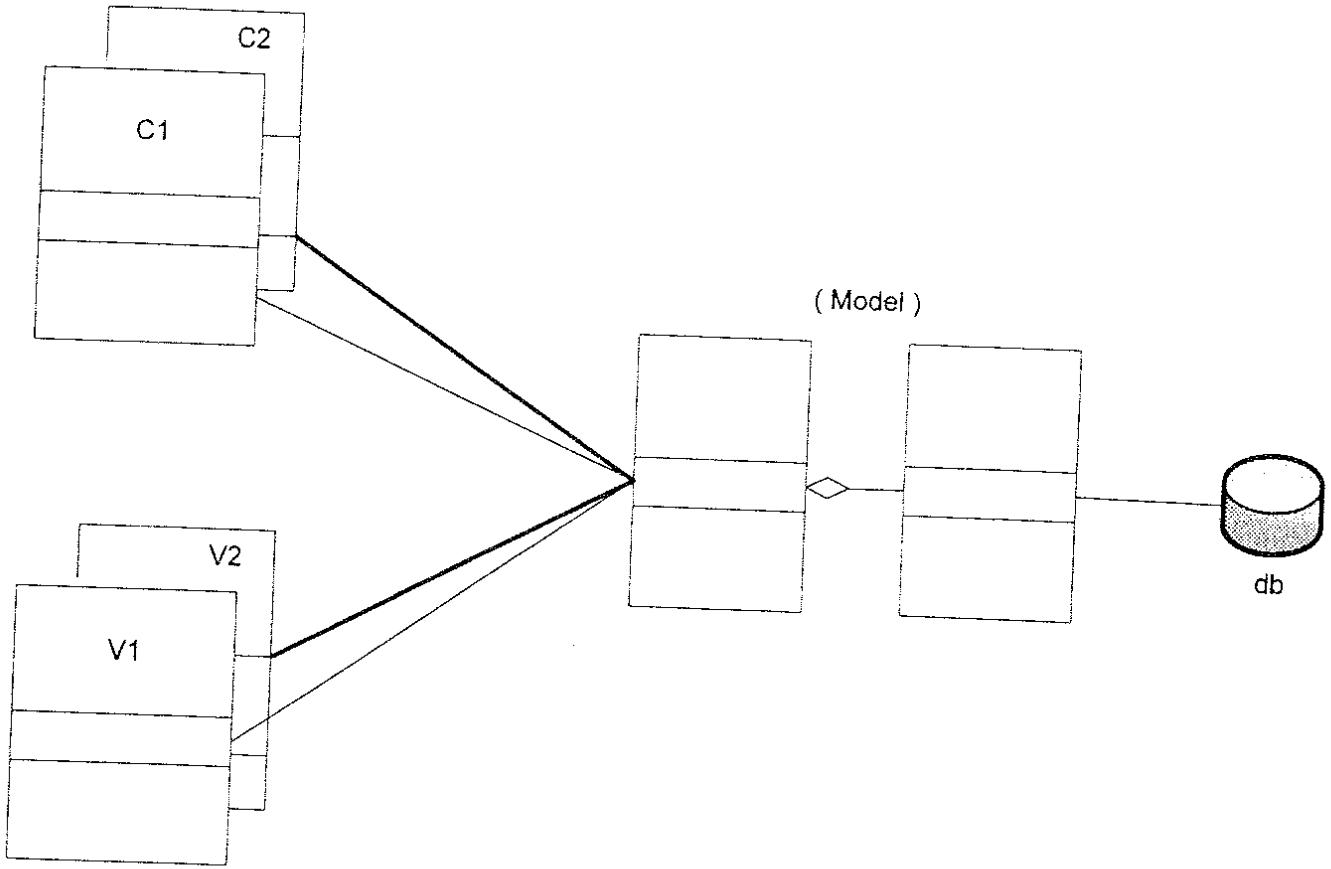

Figure: A detailed MVC-II architecture

Interaction Oriented Software Architecture

Model-View-Controller

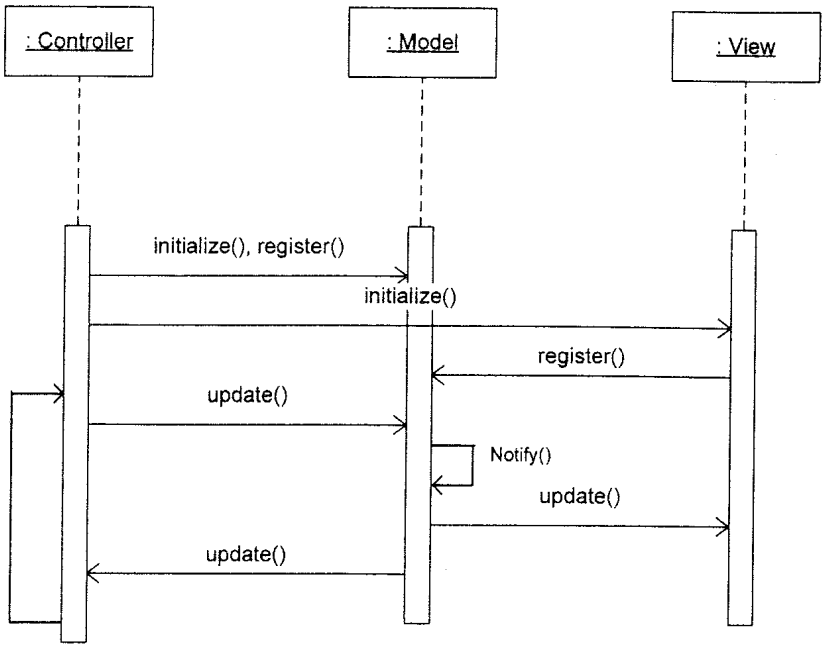

Figure: Sequence diagram for MVC architecture

Model-View-Controller

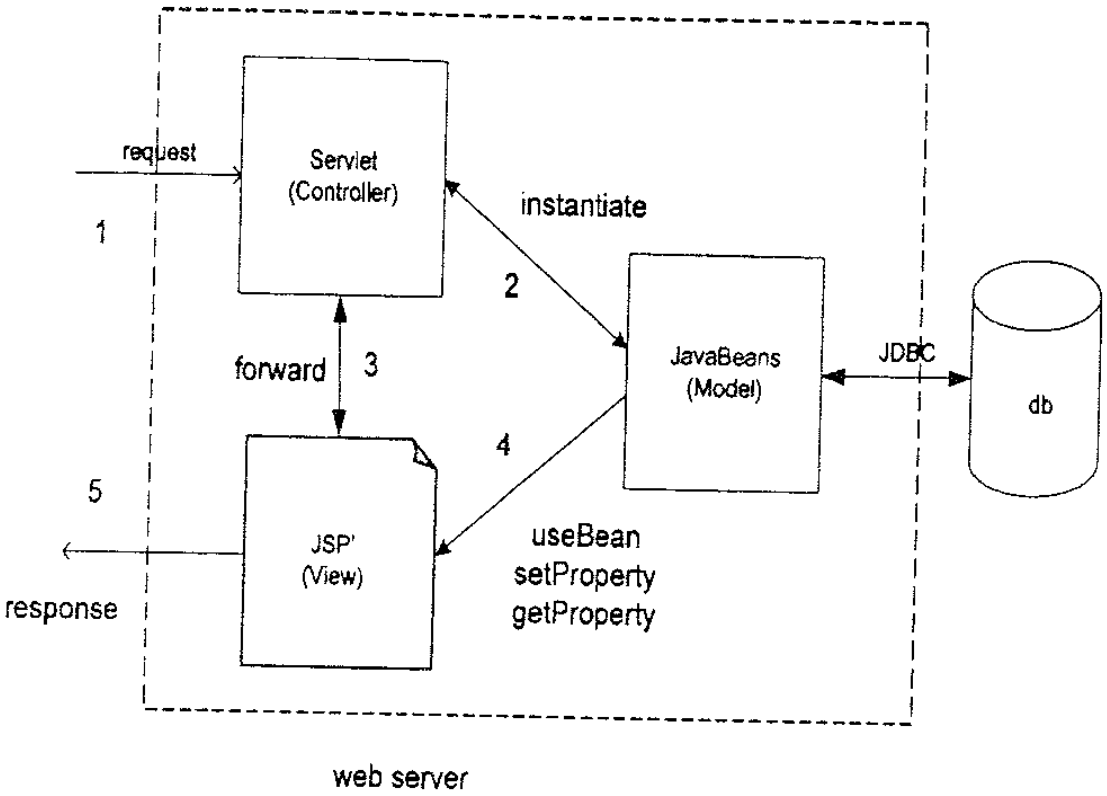

Figure: MVC architecture on Java Web platform

SFWR ENG 3A04: Software Design II

Dr. R. Khedri

Overview

Model-View-Controller

MVC-I

MVC-II

Presentation-Abstraction-Control (PAC) Architecture

Interaction Oriented Software Architecture
Model-View-Controller

- Applicable domain of MVC architecture
  - Suitable for interactive applications (multiple views are needed + volatile graphics interfaces)
  - There are clear divisions between controller, view, and data modules (different professionals can be assigned to work on different aspects of the system)
- Benefits
  - Many MVC vendor frameworks available
  - Multiple views synchronized with same data model
  - Easy to plug in new or change interface views, update interface views with new technologies
  - Very effective for developments (team = graphics, programming, and data base professionals)

Model-View-Controller

- Limitations
  - Does not fit agent-oriented application such as interactive mobile, robotics applications
  - Multiple pairs of controllers and views based on the same data model make any data model change expensive
  - The division between the View and the Controller is not very clear in some cases
- Related architecture
  - Implicit invocation architecture such as event-based, Multi-tierarchitecture, and Presentation-Abstraction-Control (PAC)

SFWR ENG 3A04: Software Design II

Dr. R. Khedri

Overview

Model-View-Controller

MVC-I

MVC-II

Presentation-Abstraction-Control (PAC) Architecture

Presentation-Abstraction-Control (PAC) Architecture

- The PAC architecture is quite similar to MVC
- The PAC was developed from MVC to support the application requirement of multiple agents in addition to the interactive application requirement
- The PAC three components concepts are applied to all concrete sub-system architecture
- It is very suitable for any distributed system where each remote agent has its own functionalities with data and interactive interface
- Another feature: all agents need to communicate with other agents in a well structured way

SFWR ENG 3A04: Software Design II

Dr. R. Khedri

Overview

Model-View-Controller

Presentation-Abstraction-Control (PAC) Architecture

Presentation-Abstraction-Control (PAC) Architecture

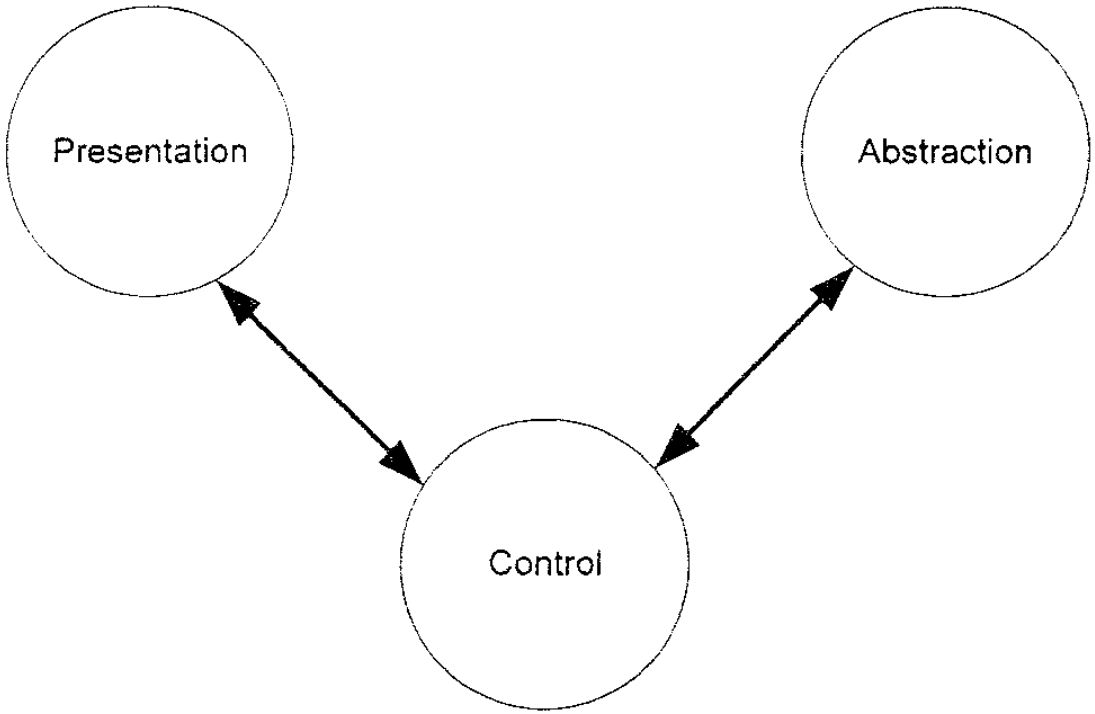

Figure: A single agent in PAC

Presentation-Abstraction-Control (PAC) Architecture

- Applicable domain of PAC architecture
  - Interactive system where the system can be divided into many cooperating agents in a hierarchical structure (Each agent has its specific job)
  - The coupling among the agents is expected very loose (change of one agent will not affect the others)
- Benefits
  - Supporting multi-tasking, multi-viewing
  - Supporting agent reusability and extensibility
  - Easy to plug in new agent or replace an existing agent
  - Supporting concurrency (agents are in different threads or different devices or computers)

SFWR ENG 3A04: Software Design II

Dr. R. Khedri

Overview

Model-View-Controller

Presentation-Abstraction-Control (PAC) Architecture

Presentation-Abstraction-Control (PAC) Architecture

- Limitations
  - Overhead due to
    - the control bridge between presentation and abstraction
    - the communications of controls of many agents
  - Difficult to determine the right numbers of the agents based on the loose couplings between agents and high independence of each other
  - Development complexity: due to complete separation of presentation and abstraction (communications between agents only take place between the controls of agents)
  - Increased complexity of the system design and implementation
- Related Architecture
  - Layered architecture, multi-tier architecture, MVC architecture

SFWR ENG 3A04: Software Design II

Dr. R. Khedri

Overview

Model-View-Controller

Presentation-Abstraction-Control (PAC) Architecture