Module 2

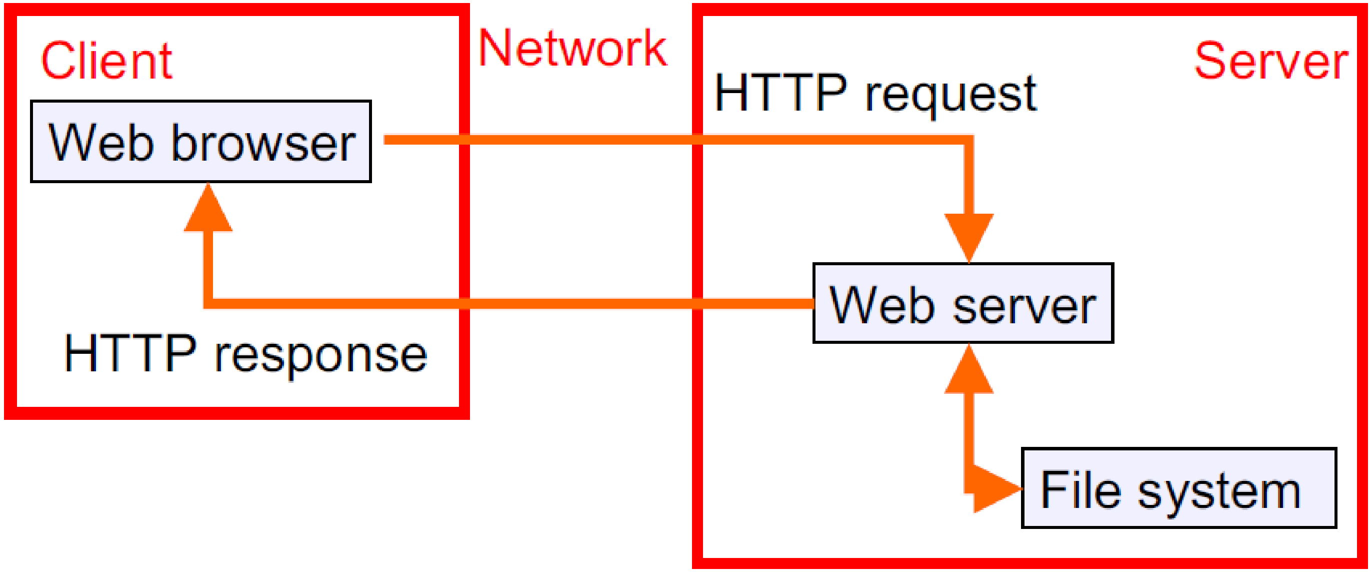
Java EE Web Component Introduction

Phoenix
one
The Knowledge Institute

Course Objectives

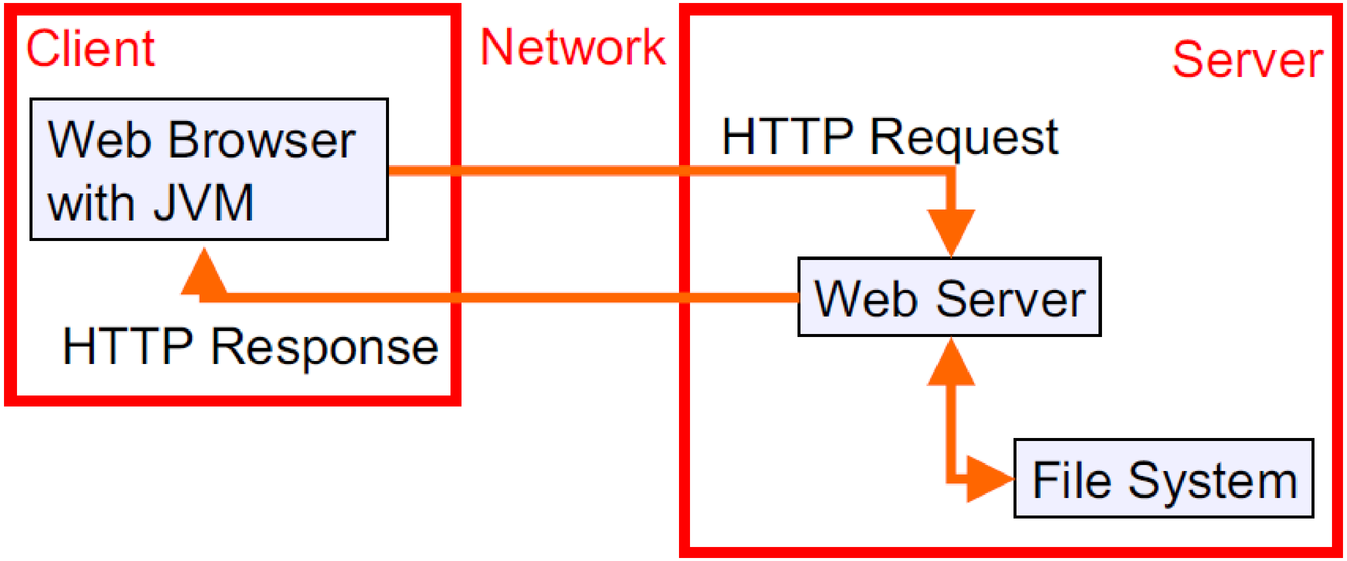
- After completing this course, you should be able to:
  - Understand the stages developing Web applications
  - Describe the model-view-controller (MVC) design pattern
  - Understand the importance of MVC in Java EE applications
  - Describe the Java EE web component and container
  - Describe the architecture of Java EE

Web Application Evolution: static



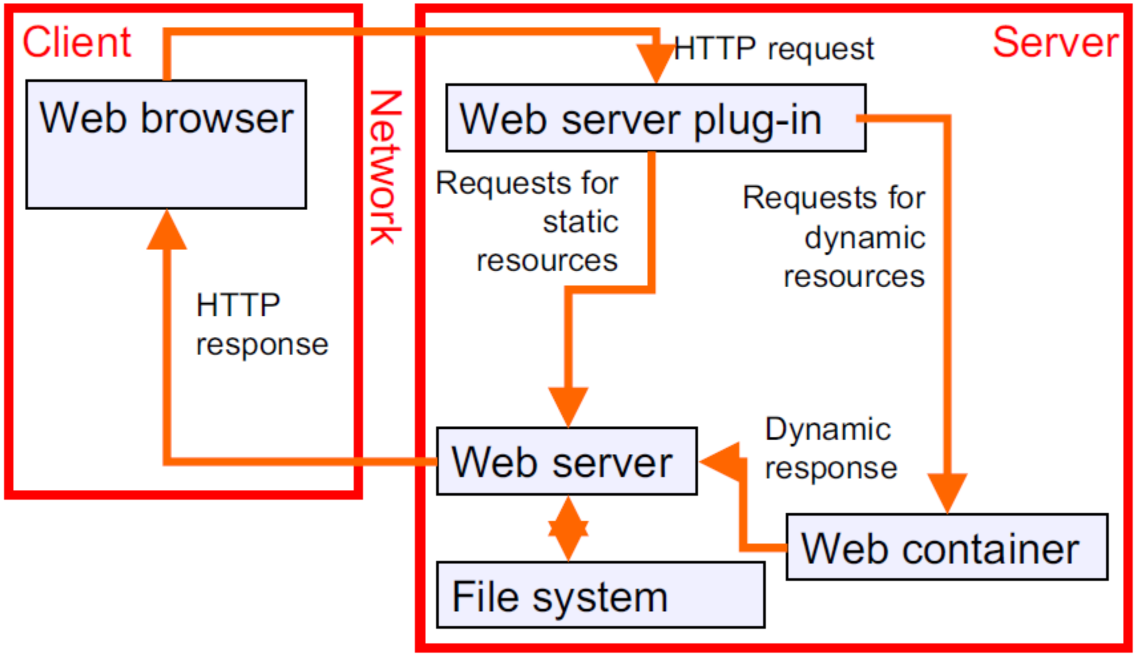
- Organizations want to make their information available to as many people in the world as possible
- This can be achieved by using the Web, delivering the information as static HTML pages

Web application evolution: applets



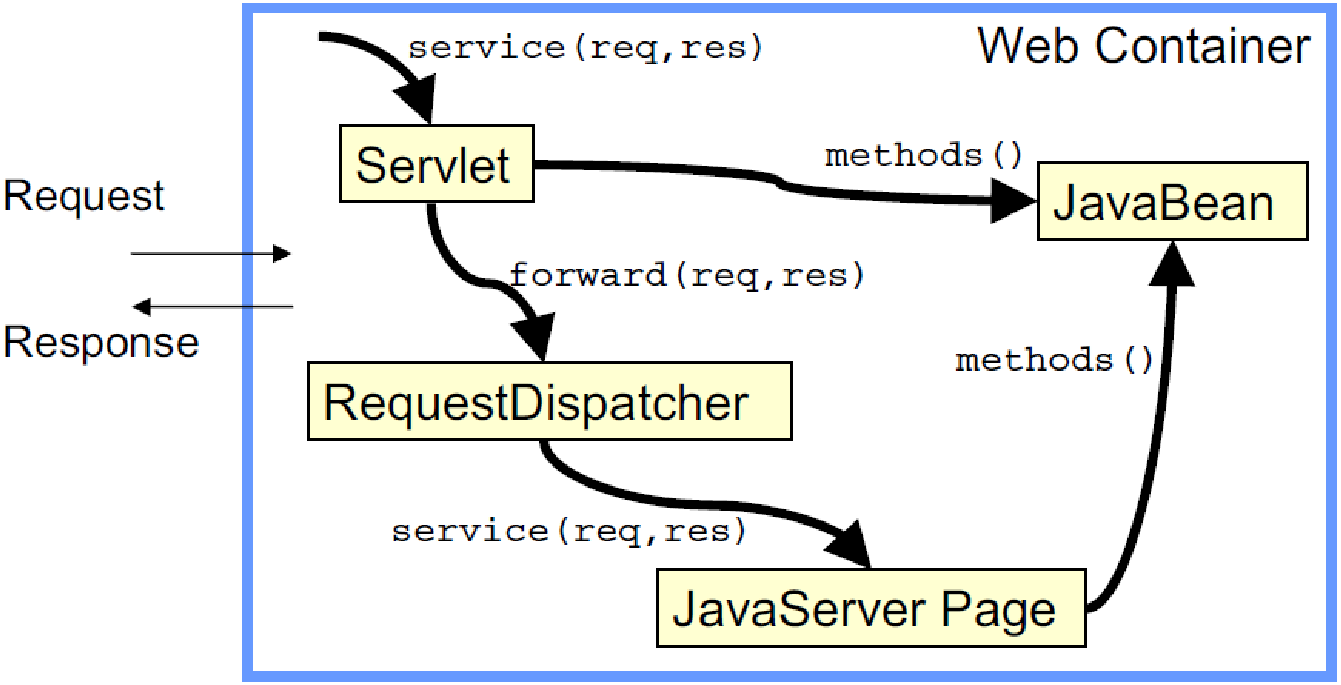
- With static HTML, users see passive page presentations that are always the same
- Presentation can be improved with Java applets or the client-side programs

Web application evolution: servlets



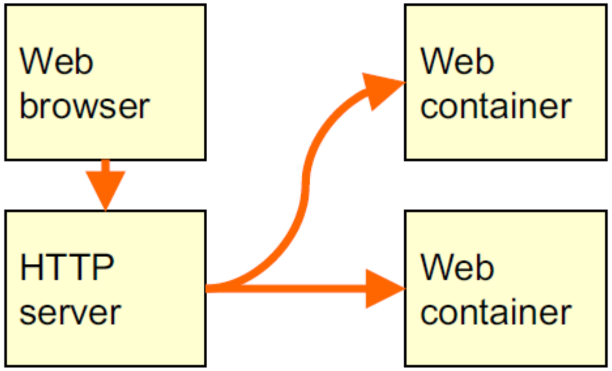
- Applets cannot access data on back-end systems
- A Web container can provide server-side components, to generate dynamic content.

Web application evolution: JSP pages



- Servlet provide poor presentation between business logic and presentation logic
- Use JSP pages and beans to improve session

Web application evolution: scalability

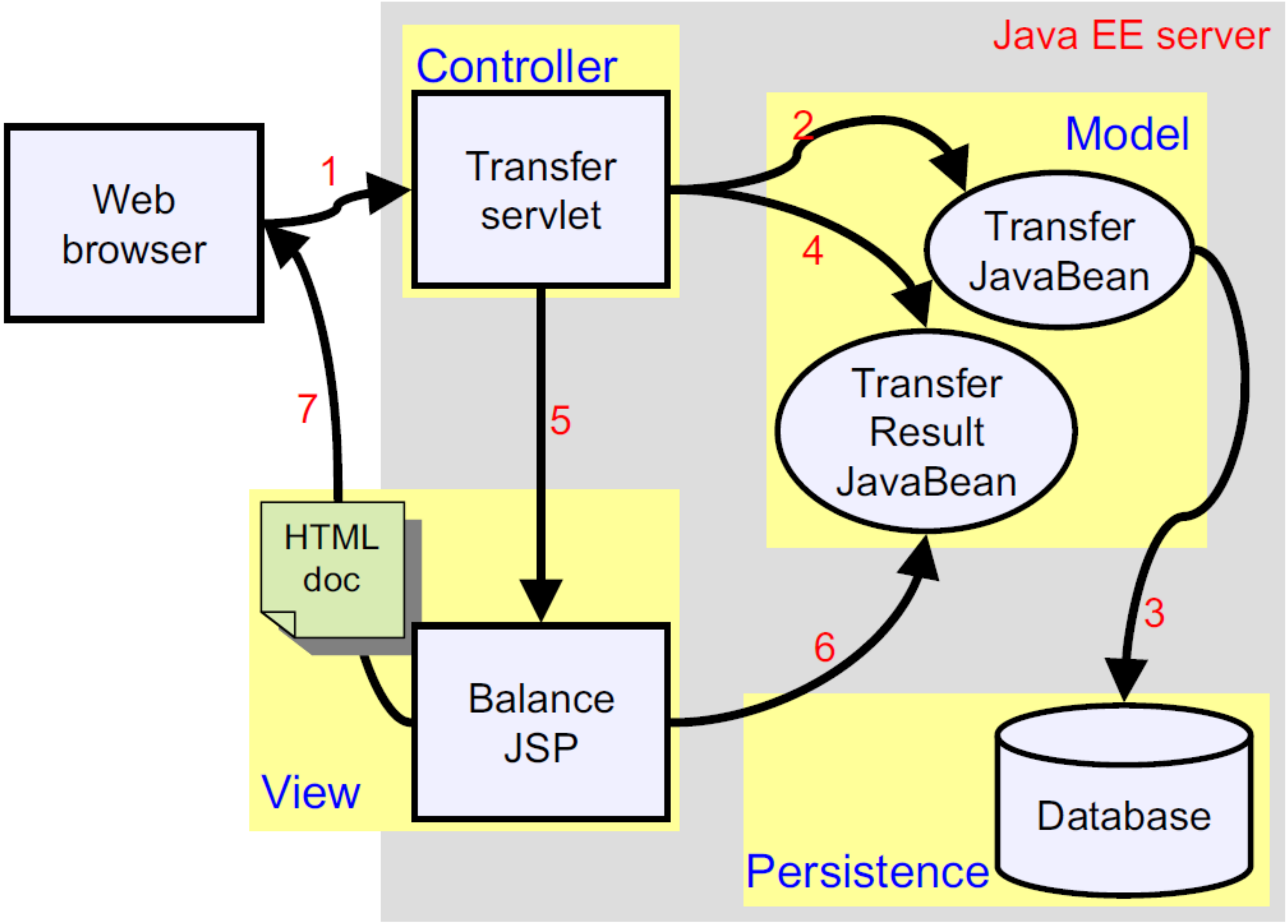


- Business requirements often involve high availability
- Improved performance may be required as business grows
- Both these requirements can be achieved through scaling
  - Servers can provide redundancy in the system
  - By sharing the load between servers, performance can be enhanced

The Model-View-Controller pattern

- Model-View-Controller (MVC) is a design pattern
  - Very widely used and regarded as a core concept in Java EE development
  - Various practical benefits
    - ❑ Promotes code reuse
    - ❑ Reduces development time
  - The model
    - ❑ Represents the underlying data and business logic in one place
    - ❑ Contains no information about the user interface
  - The view
    - ❑ The user interface: things that the user can see and respond to
    - ❑ Represents a window into the model: there can be many of these
  - The controller   Connects the model and the view   Is used to communicate between the model and view
- A fourth layer – persistence – is often added to the pattern

MVC: Application to Java EE



MVC: Benefits

- Promotes code reuse
  - The purpose of the model is to provide business logic and data access in one place
  - You can reuse this logic in many applications at the same time, without the need for any extra coding
- Reduces development time
  - The model, view, and controller are developed in parallel
- Is more maintainable
  - You can change the view without affecting the model
  - You can change the model without affecting the view
  - You can move data without affecting the view or model

Java EE Introduction

- The Java Platform, Enterprise Edition (Java EE) Specification consists of:
  - The Java EE Platform
    - A standard platform of containers, services, and communications
  - The Java EE Compatibility Test Suite
    - To verify that a Java EE platform product complies with the standard
  - The Java EE Reference Implementation
    - For prototyping Java EE applications, and providing operational definition of the Java EE platform
  - Java EE is developed using the Java Community Process
    - Collaboration between major enterprise software vendors
    - IBM actively contributes to the Java EE development process

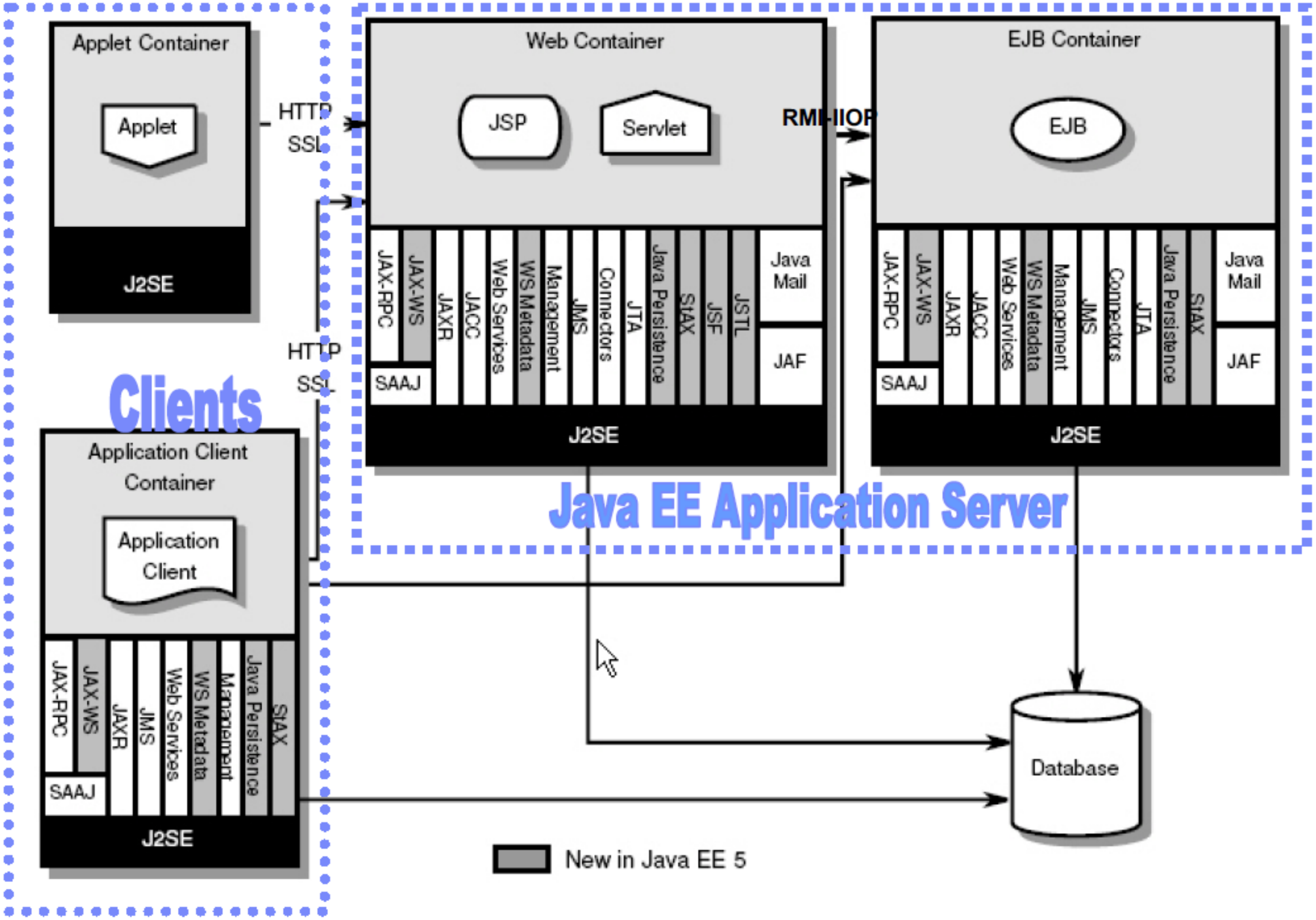
Java EE components

- Java EE defines different types of components, which must be supported by any Java EE product
  - This course will only cover Web components
    - Servlets and JavaServer Pages
    - These provide the controller and view functionality in Java EE
- Component deployment can be managed using deployment descriptors (except applets)

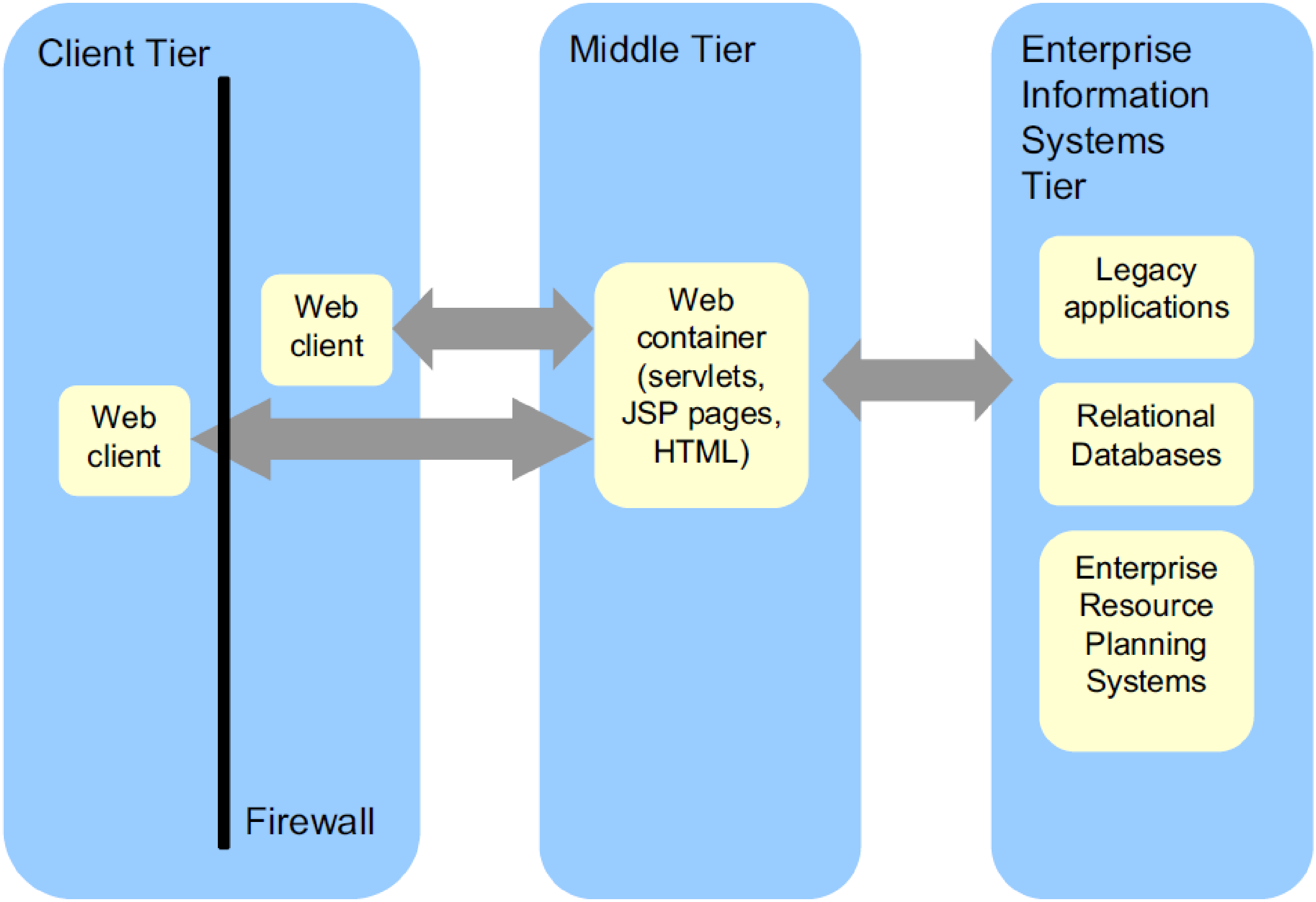
Java EE containers

- Java EE defines different types of containers.
  - Containers manage the components that they contain.
  - This course will only cover Web component containers
- Java EE components never interact with other Java EE components directly
  - They depend on the runtime support of containers
  - Interaction takes place using services provided by containers

Java EE architecture



Java EE multi-tier model


Client Tier
Web client
Web client
Firewall
Middle Tier
Web container (servlets, JSP pages, HTML)
Enterprise Information Systems Tier
Legacy applications
Relational Databases
Enterprise Resource Planning Systems

Java EE benefits

- Standards
  - A wide range of standard services, components, clients, and tools are supported
  - Applications are portable across Java EE platforms
- Distribution
  - Provides scalability for performance and availability
- Common services
  - Most commonly required services are provided by Java EE servers
- Component model
  - Good separation of development responsibilities
  - Good reuse of code and opportunities to share logic between applications
- Interoperability
  - Integration with other systems using standard protocols

Module Summary

- Having completed this module, you are able to:
  - Differentiate the stages in the development of Web applications
  - Describe the model-view-controller (MVC) design pattern
  - Relate the importance of MVC in Java EE applications
  - Explain the Web components and containers
  - Discuss the Web architecture of Java EE