Component Architectures

29 March 2011
CMPT166
Sean Ho
Trinity Western University

TRINITY WESTERN UNIVERSITY

Software components

- Pre-fabricated, reusable building blocks for software systems
- Rapid development, consistent reuse
- Not tying together libraries (chunks of code)
  - But coordinating running code (dynamic cooperation of live objects with active state)
- Bigger than single objects, can combine
- Relate "peer-to-peer", not in hierarchy
- May operate across a suite of applications
  - Provide consistent interfaces to applications

Component concepts

- Applications use a palette of components
  - The programmer composes or “wires” them together to make a complete application
- Requires:
  - Ways to define new components
  - Standards to specify component interfaces
  - Each component has “hooks” (methods) by which other components interact with it
- Compare to hardware components:
  - Transistors, integrated chips, etc.

Components vs. code

- Hardware components are black boxes with spec sheets: wires connect them together
- Software components are represented as black boxes with interfaces: you write code to connect them up
- Software companies may sell components as binaries (black boxes) with API documentation
  - Need not sell the actual source code
  - e.g., NVIDIA binary graphics drivers for Linux
  - e.g., Scantron ClassClimate and myTWU portal

Component-based apps

- Components are assembled into containers
  - The finished assembly is the application
- May also take document-centric view:
  - e.g., a container document may hold text, images, videos, buttons, etc.
  - Editing any item passes control to the appropriate component: text editor, image editor, etc.
  - The document *is* the application!
  - Peer-to-peer: no one component is “boss”

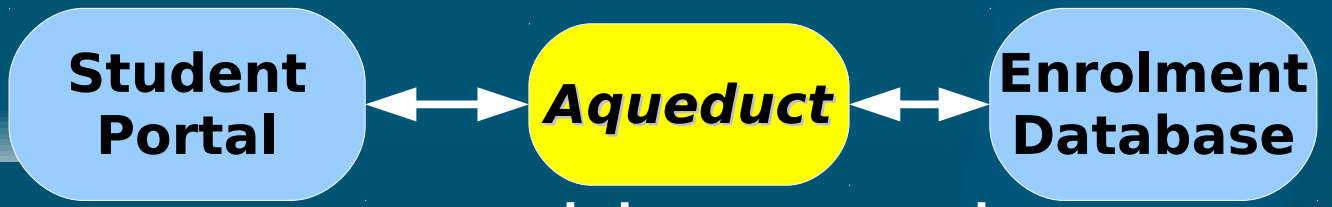
Component-based develop.

- Deliver solutions by building or buying interoperable components
- Don't reinvent the wheel: write once, deploy many times (server, desktop, handheld, ...)
- Rigid adherence to software infrastructure:
  - Standards of how components work together
- Fits into distributed, multi-language, multi-platform heterogeneous environments:
  - Don't care what language it's written in
  - Don't care where it runs

Component structures can...

- Tie together departments within a company (ERP – enterprise resource planning):
  - Accounting, invoicing, human resources
- Connect data of mergers of banks, hospitals
- Use rich, complex data stores:
  - Data mining, pattern recognition, image analysis, genomics, StatsCan, ...
- Add multimedia to a field salesperson's laptop/handheld
  - Use same back-end applications as at office

Layering



- Sometimes component architecture is deployed as “middleware”:
  - A set of components that allow a variety of database stores or applications to be manipulated by a common interface
  - Other applications must go through the middleware in order to access the datastore
- Security, ease of debugging, simplicity
- Allows format of back-end database to change while preserving the front-end UI for users

Examples of component arch.

- Application plug-in interface: Firefox, Eclipse
- LAPACK/BLAS: std. linear algebra library
- ActiveX/COM: interoperation of MS apps
  - e.g., graphics, outlining, cut-and-paste
- .NET: Microsoft's (2002) component arch.
  - CLR (Common Language Runtime) is the equivalent of the JVM
  - C#, but may use other languages, too
- JavaBeans: components for Java

Example: ODBC

- Open DataBase Connectivity
  - Standard API to relational database systems: MS-SQL, Oracle, DB/2, mySQL, PostgreSQL, ...
  - Simplifies use of standard SQL commands
    - Structured Query Language: query/edit the DB

```
SELECT Titles.Title, Authors.Name
    FROM Titles, Authors
    WHERE Titles.ISBN = Authors.ISBN
```

    - Can also access vendor-specific commands
  - Cross-platform, cross-language
    - The Java implementation is called JDBC

Example: JavaBeans

- Introspection: components advertise what features they provide (methods, events, ...)
- Use Swing's event model to communicate
  - All Swing components are JavaBeans!
- Persistence: Beans can save/restore state
- Builder: IDE to assemble components
  - JavaStudio, NetBeans, Eclipse, IntelliJ, etc.
- Enterprise JavaBeans (EJB) integrate with the JavaEE ecosystem

Applets vs. Servlets

- Applets run embedded in the browser
  - User must install JVM and browser plug-in

- Servlets are compiled Java programs that run on the web server (servlet container)
  - Return a XHTML web-page to be sent to client

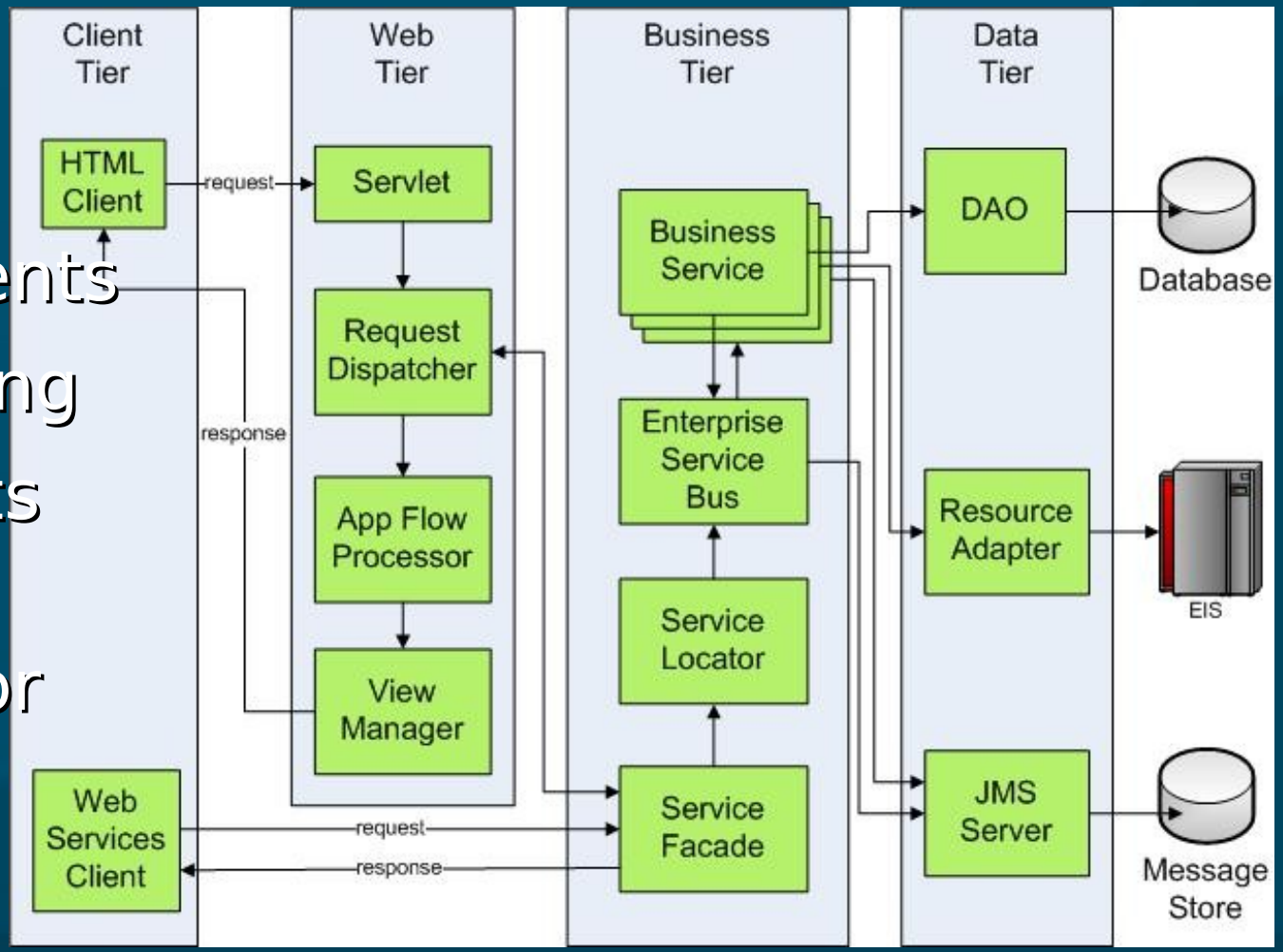
Servlets and JSP

- Java Server Pages (JSP) are interpreted by the servlet container and compiled into servlets as needed
  - Pre-compiled servlet is used if available
- Scriptlet code mingled with HTML markup
  - Similar to MS Active Server Pages (ASP), PHP, WSGI + Python, etc.

```
<html><head>
<%! int servervar = 1; %>
<% int localvar = 1; %>
<h1><%= toStringOrBlank("hello, world!") %></h1>
```

Beyond Servlets: JavaEE

- JavaEE is a component ecosystem for enterprise applications:
  - Servlets
  - EJB: components
  - JMS: messaging
  - JSF: UI widgets ("faces")
  - JCA: connector to datastores



JavaEE software

- Official reference implementation: Oracle
  - Glassfish Open-source edition
  - Glassfish Enterprise (commercial)

- Apache (open-source):
  - Geronimo (JavaEE) + Tomcat (Servlets)

- IBM WebSphere Application Server:
  - Community Edition (based on Geronimo)
  - Application Server (commercial)
- RedHat JBoss (open-source)
- SAP NetWeaver (commercial)