COURSE DESCRIPTION

An application server is a rich, highly portable software program that runs on a middle tier server and handles all application operations between client applications running on pervasive devices and back-end databases and business applications. Application servers provide a platform independent programming interface for developing portable applications in a variety of programming languages. Application servers also facilitate the integration of legacy applications via on-the-fly transformation of XML messages, and support a wide variety of XML-enabled client channels that include traditional web clients and a growing set of smart devices. Application servers enable programmers to implement distributed applications using a variety of architectural patterns including traditional Model-View-Controller (MVC) architectures, Service Oriented Architectures (SOAs), Point-to-Point (P2P) Architectures, Grid-Computing Architectures, etc. Using an SOA architectural pattern, and emerging standards such as SOAP, application servers enable a new generation of "web services" that allow systems to make remote procedure calls to other systems over the Internet. Today, J2EE, .Net, and CORBA 3 application servers set the stage as enabling platforms for Web Services initiatives, Web appliances, and wireless applications.

This course concentrates on architecting, designing, and developing persistent software applications using application server technology. Throughout the course, students are exposed to the evolution of application server architectures that started in the mid 1990s, and experiment with corresponding approaches based on traditional client-server technology, CGI frameworks, page-based extended HTML environments, distributed object computing platforms, object management architectures, component-based
computing environments, and frameworks based on MVC, SOA, P2P, etc. The course conveys the necessary skills to select the proper application server architecture based on projects' business and technical requirements. The course also explains how to integrate an application server into an existing Web site, as well as how to implement an application server-based Web application from the ground up. Students will learn how to configure and operate application servers in production environments, taking advantage of features available in mainstream commercial frameworks such as scalability, concurrency, security, fault tolerance, auto-deployment, communications support, development environment, and monitoring tools.

As they design and implement applications using commercial and open source application server technologies (e.g., IBM WAS, BEA WebLogic, .Net, OpenCCM, etc.), students will learn how to identify application patterns that lead to the most effective use of the various services provided within application server frameworks. The design and implementation of the persistence and legacy application integration layers, using related application server technology, will be particularly emphasized. Case studies, provided as part of the course, will demonstrate how medium- to large-size sites manage the complexities inherent in these endeavors. These case studies will help students get a firm understanding of how application servers work and how to best deploy complex applications in a real-world situation. Although, the course will strive to provide a comprehensive coverage and classification of application server technologies, attempts will be made whenever possible to select open source technologies for experimentation purpose. As part of the course, students will be exposed to next generation application server technologies including Model Driven Architectures (MDAs), Pattern Driven Architectures (PDAs), as well as reflective, multimedia and agent-enabled frameworks.

**COURSE OBJECTIVES**

The objectives of the course are as follows:

1. Convey the evolution of application server architectures that started in the mid 1990s.
2. Provide a broad exposure to legacy, mainstream, and upcoming application server technology.
3. Expand the in-depth understanding of mainstream component-based application server environments based on the EJB, CORBA 3, and .Net/COM+ component infrastructures and models.
4. Identify and study core application server technologies.

5. Provide exposure to advanced design and development techniques using application server technology.

6. Clarify the key role of XML in application server’s support for application integration, Web Service platforms initiatives, application server appliances, and wireless applications.

7. Demonstrate the use of application server technology for business critical applications that require scalability, concurrency, security, fault tolerance, auto-deployment, communications support, development environments, and monitoring tools.

8. Provide a snapshot of upcoming technology directions that will guarantee a lasting expertise of application server technology.

TEXTBOOKS

Building J2EE Applications with IBM WebSphere
Dale R. Nilsson, Louis E. Mauget

Microsoft .NET Distributed Applications: Integrating XML Web Services and .NET Remoting
Matthew MacDonald
Microsoft Press, ISBN: 1-735-61933-6 (02/03)

PREREQUISITES

Students enrolling in this class are expected to have taken G22.2110 (i.e., Programming Languages) and G22.2250 (i.e., Operating Systems) and their prerequisites or to have equivalent knowledge. Students are also expected to have taken a Java and/or C# intermediate course, and to have basic knowledge of the Core JFC classes, and the ability to program in Java and/or C#. Some exposure to the topics and techniques covered in G22.3033-02 (i.e., Programming for the World Wide Web) is a plus.

WEB SITES
Download the latest Java 2 SDK from http://www.javasoft.com/. Other sources are listed under References on the course main page. These include information available on application servers portals, application servers FAQs, Web Services platforms such as Sun's Open Net Environment (ONE), HP's NetAction/e-speak, Oracle's Dynamic Services and Portlets (Oracle 9i and 10g), Microsoft's .NET, Novell's Directory-Enabled Net Infrastructure Model (DENIM), IBM's WebSphere Architecture (WSA), BEA Systems’ WebLogic, and Web Services initiatives conducted by the Apache Group, ActiveState, ObjectSpace, XMethods.com, DevelopMentor, Rogue Wave, and BowStreet. Additional sources will be provided as applicable during the course sessions.
COURSE SESSIONS

1. **Legacy Application Server Technology**
   - Network Programming Review
   - Traditional Client-Server Technology
   - Object-Oriented Client-Server Environments
   - CGI Frameworks
   - Introduction to Component Technologies
   - Introduction to Pattern Driven Architectures
   - Ongoing Project Overview

   **READINGS:** TBA
   Handouts posted on the course web site

2. **Page-Based Application Servers (Part I)**
   - ColdFusion 6.1 and MX Environment
   - PHP 4 Environment
   - XML-Based Application Servers

   **READINGS:** TBA
   Handouts posted on the course web site

3. **Page-Based Application Servers (Part II)**
   - ASP Environment (i.e., IIS with COM+, and ASP)
   - Servlets and Servlet Engines
   - JSPs and JSP Engines
   - TomCat/JRun
   - Cocoon 2/XSPs

   **READINGS:** TBA
   Handouts posted on the course web site

4. **Virtual Machines/Distributed Object Computing Platforms**
   - VMs and Component Technologies
   - JVM and .Net CLR
   - Abstract Component Infrastructures
   - CORBA
   - RMI and RMI-IIOP
   - .Net/COM+
   - DOC Platform Interoperability
   - Web-Enabled DOC Applications
5. **Object Management Architectures**
   - Object Management Architectures
   - Java-Based/J2EE Application Servers
   - Windows Services
   - Introduction to Enterprise Component Development
   - Introduction to Component-Based Architectures Design

6. **J2EE Component-Based Computing Environments (Part I)**
   - EJB Component Model
   - J2EE Services (JNDI, JMS, JTS, CMP/BMP/JDBC, MDBs)
   - Security in J2EE Application Servers

7. **J2EE Component-Based Computing Environments (Part II)**
   - WebLogic
   - WebSphere
   - Open Source J2EE Environments (JBoss)

8. **J2EE Component-Based Computing Environments (Part III)**
   - Other J2EE Application Servers
   - Inprise, iPlanet, Sybase EAServer, etc.
   - Enterprise Component Development with J2EE
   - Designing J2EE Component-Based Architectures

9. **CORBA 3 Component-Based Computing Environments**
   - CORBA 3 CCM
   - CORBA 3 Environments
10. **.Net/COM+ Component-Based Computing Environments**
   - COM/DCOM Component Model, and OLE
   - COM+ MTS services: Transactions, and Security
   - COM+ Message Queuing
   - .Net/COM+ Integrated Services
   - Enterprise Component Development with .Net/COM+
   - Designing .Net/COM+ Component-Based Architectures

   **READINGS:** TBA
   Handouts posted on the course web site

11. **XML-Based Capabilities in Component-Based Environments**
   - EAI Environments
   - B2Bi Environments (e.g., WebMethods Framework)
   - Channel Independence and Pervasive Devices

   **READINGS:** TBA
   Handouts posted on the course web site

12. **Service Oriented Architectures / Web Services (Part I)**
   - XML-RPC
   - SOAP, DIME, and ROPE
   - UDDI, DISCO
   - WSDL

   **READINGS:** TBA
   Handouts posted on the course web site

13. **Service Oriented Architectures / Web Services (Part II)**
   - Mainstream UDDI Registries and Browsers
   - Mainstream Web Services Toolkits

   **READINGS:** TBA
   Handouts posted on the course web site

14. **Upcoming Application Server Technologies**
- Component Development Summary
- Model Driven Architectures
- Reflective Application Servers
- Intelligent Agents
- Multimedia/Broadband Application Servers
- Other Component Models Directions

**READINGS:** TBA

Handouts posted on the course web site
READINGS

Assigned readings for the course will be from the textbooks, from various web sites documentation, and from trade magazines and recommended books listed on the course web site.

ASSIGNMENTS

Homework and project assignments completion will be required.
Quizzes will be administered.
The final exam will be a take-home exam.

GRADING POLICY

25% Assignments
35% Projects
30% Final Exam
10% Attendance and Participation
Extra credit will be granted periodically for particularly clever or creative solutions.