WebSphere for Dummies

An introduction to WebSphere Application Server

Tom Hackett
IBM zSeries New Technology Center
thackett@us.ibm.com

Trademarks

The following are trademarks of the International Business Machines Corporation in the United States and/or other countries.

AIX*
CICS*
DB2*
e-business logo*
HiperSockets
IBM*
IBM eServer
IBM logo*
IMS
OS/390*
Parallel Sysplex*
RACF*
Tivoli*
TotalStorage
WebSphere*
z/OS
z/VM
zSeries

* Registered trademarks of IBM Corporation

Intel is a trademark of the Intel Corporation in the United States and other countries.

Java and all Java-related trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc., in the United States and other countries.

Lotus, Notes, and Domino are trademarks or registered trademarks of Lotus Development Corporation.

Linux is a registered trademark of Linus Torvalds.

Microsoft, Windows and Windows NT are registered trademarks of Microsoft Corporation.

SET and Secure Electronic Transaction are trademarks owned by SET Secure Electronic Transaction LLC.

UNIX is a registered trademark of The Open Group in the United States and other countries.

* All other products may be trademarks or registered trademarks of their respective companies.

Sea of Acronyms*

*Several acronyms expanded at end of presentation

Agenda

- A Short History of Web Servers
- Application Servers and how they work
  - WebSphere®
- WebSphere and z/OS®
- Recent developments

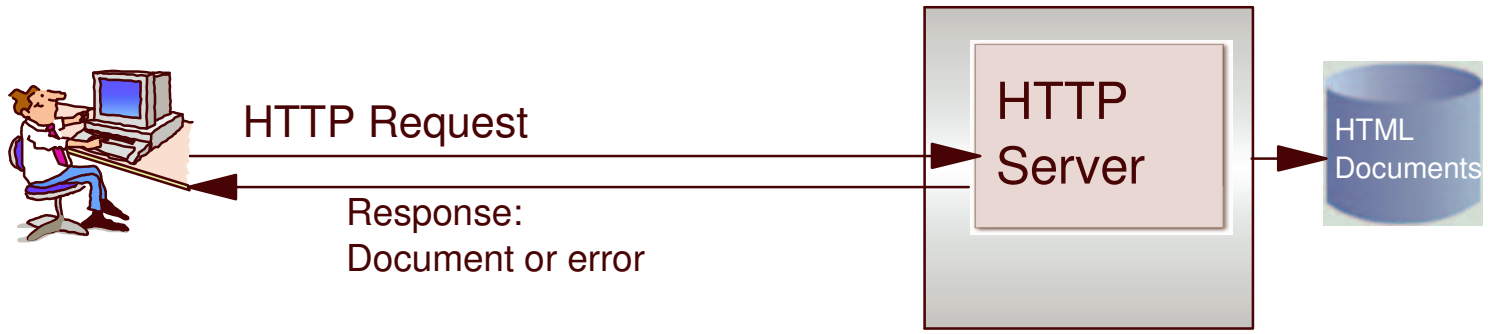
HTTP Servers

- June-December of 1980
  - Tim Berners-Lee writes "Enquire-Within-Upon- Everything"
    - Written while consulting at CERN
    - Idea for linking arbitrary nodes
- 1989
  - Berners-Lee publishes "Information Management: A Proposal"
- 10/90
  - **Started writing the first Web Browser program on a NeXT desktop**
  - **http://nxoc01.cern.ch/hypertext/WWW/TheProject.html (very first web page 11/1990)**
    - **Don't bother trying it--it's not there any longer**
- October 1994
  - World Wide Web Consortium (W3C) Created
- Web servers
  - **Original CERN, NCSA**
  - **Now Apache, Netcape®, IIS, IHS (a.k.a. ICS, ICSS, DGW), Others**

Note: This information is from the World Wide Web Consortium's "a Little History of the World Wide Web"

Client/Server based model

- Documents stored in Hypertext Markup Language
  - client (browser) interprets the HTML
- Hyper links can direct the browser to other documents (anywhere on the Web).
- Supports Applications at the server, generating the output
- Uses request/response model
  - stateless connection.
- Request carries identity of document requested and other information
  - Identity is URL - Uniform Resource Locator
    - protocol://host:port/path/document name
      - e.g. http://www.share.org/events/Anaheim/speakerchair/presentation_template.cfm
  - Other information in request header
    - expiration time, requester info, cookies, etc.



HTTP Servers

- One of the most important enablers is standards
  - W3C maintains the Standards
  - HTML for the markup language
    - Based on GML
    - Eventually leads to development of XML
  - Hypertext Transfer Protocol (HTTP) for the server
- Anyone can write a server to provide the standard functions, or a browser to process the standard HTML tags, using the same TCP/IP protocol
- Over time, application programming models were added
  - CGI's
  - API's (such as GWAPI)
- Then an Application Server

CGI's

- In the beginning there were CGI's and they were good
  - Fork/Exec model (spawn an address space) so the application ran in it's own process, so you couldn't step on another program or the server itself
  - Request passed in a standard format, so that the application could get the input request and the client information
  - Write in any supported language, based on the platform: Perl, C, C++, REXX, PL/I, Cobol, Assembler, VB, etc.
  - Only problem cost (time and cycles) of initializing and terminating the process (address space) for each request
- And then there were API's and CGI's were still good
  - API's would run faster because they were local spawns in the same process.
    - Each was a thread in the HTTP Server's (Web Server) process.
  - Loaded as .dll's, so it had to be C or C++ code (usually), or assembler for the ambitious
  - Big Issue: No isolation or integrity
    - If one had a problem, all suffered
- Lead to the need for an Application Server

Application Server

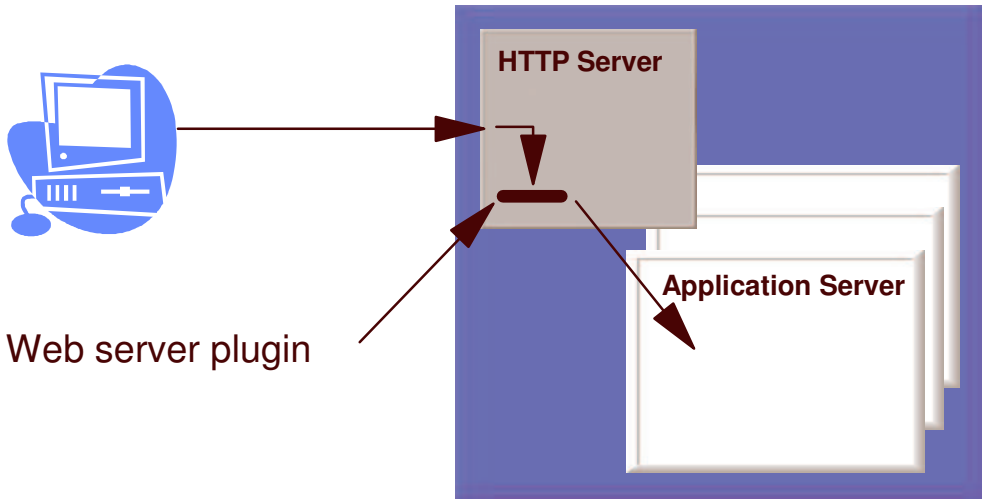
- An Application Server usually supports the following
  - Common programming model based on a standard
  - Platform neutral application programming
  - JAVA based execution model, interpreted using a JVM
- Some Available Application Servers
  - IBM WebSphere Application Server
  - BEA™ WebLogic
  - iPLANET
  - Others: Tomcat, JRUN, JSERV etc.

A Web server is not an Application Server

Extending the Web server

- The Application Server has been thought of as an extension to a Web server, consisting of 2 main components
  - A plugin for the Web server (HTTP Server) that will pass the request to the actual Application Server.

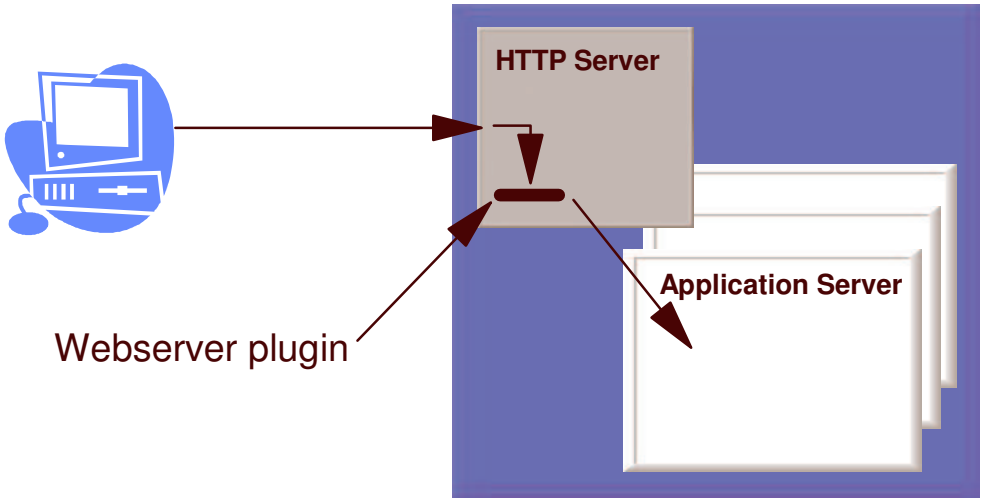
    and
  - The Application Server itself



Some Application Servers have built-in HTTP support

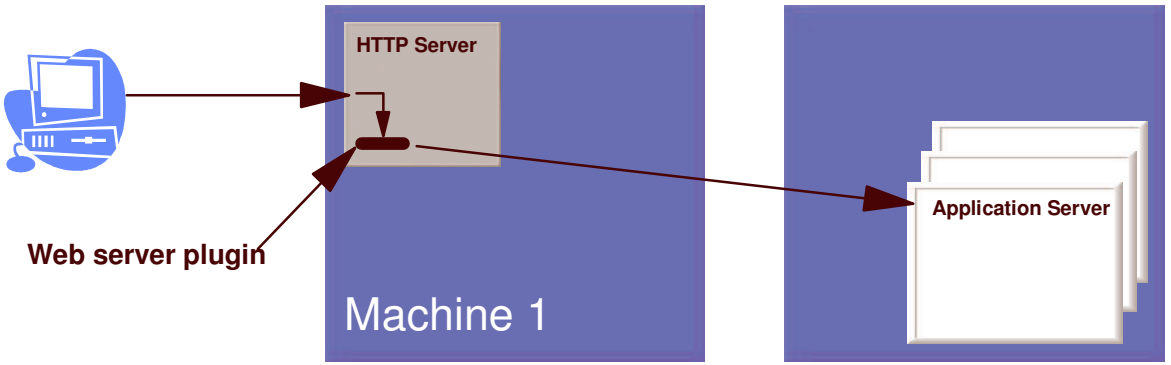
HTTP Server plugin

- This plugin is usually a .dll or .so file
- It is included in the webserver's configuration file as a plugin
  - (Service statement on z/OS)
- It usually has it's own configuration, located in a file or some other location.
- It runs just like an API application, but the vendor you get it from should support any problems.



Application Server

- Usually* runs as a separate process either on the same machine as the web server or on a different machine
  - Can be one or more processes depending on how it was setup.
- Plugin can communicate with the Application Server using several protocols depending on the Application Server vendor
  - IBM uses HTTP and HTTPS.
- Application Server uses a Java™ environment to run the executables.
  - Interpretive model.
    - Needs a Java runtime environment (JRE) or Software Development kit (SDK)
    - Key element is Java virtual machine (JVM), the interpretive runtime
- The Application Server runs components as threads in the Application Server.



*The original Servlet Express and WebSphere Application Server Standard Edition for OS/390 ran in the same address space as the HTTP Server

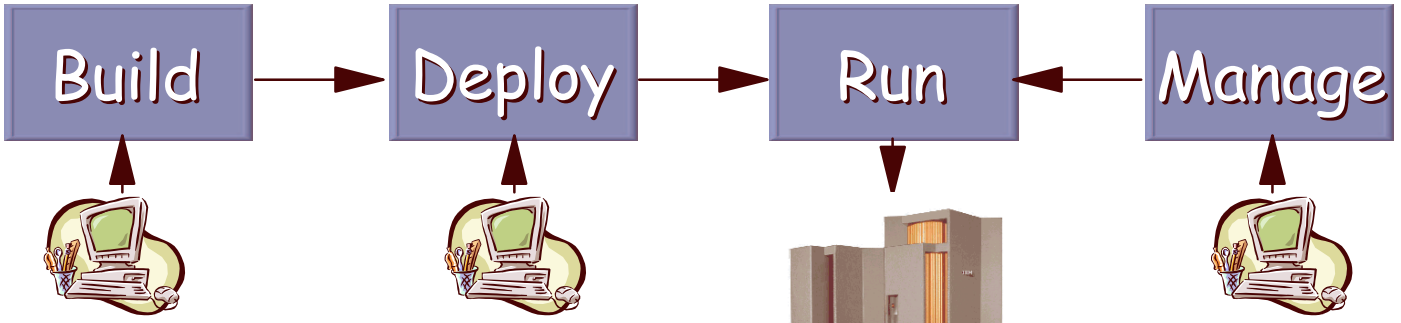
Standards

- Standards are key to platform neutral code
- J2EE™ (Java 2 Enterprise Edition) is the main standard all Application Servers are measured against
- Standard currently supported in WebSphere Application Server
  - V5 - J2EE 1.3
  - V6 - J2EE 1.4
  - Components that make up the J2EE standards also have version numbers (just to confuse you)
    - Servlets
    - JSP's
    - EJB's (including Message Driven Beans)
  - Others, Java Mail, Java Messaging Service etc.
- Standards maintained by Sun with many companies contributing

What's in J2EE

- Components
  - These are Servlets, JSP's and EJB's with other Java classes as helpers and utilities. These are executable.
- Containers
  - This is where the components are executed. There are two types of containers, Web containers (where servlets and JSP's are executed) and EJB containers (where EJB's are executed). A container on z/OS is not an address space. You can have both a Web container and an EJB container in the same server address space. The difference is the services available to the component.
- Connectors
  - These are the adapters a developer uses to get to databases and transactions. They include the CICS Transaction Gateway, IMS Connect, JDBC drivers, and JMS. In the past there were the CCF Connectors (Common Connector Framework) for Servlets. The J2EE standard now includes the J2CA (J2EE Connector Architecture) connectors.

End to end environment



- This is not ISPF
- Development is usually on a desktop using
  - WebSphere Studio Application Developer
  - Other products
- The developer packages the components into archives
  - WAR- Web archive for HTML, Graphics and JSP's
  - JAR- Java archive for Servlets, EJB's, and class files
  - EAR - Enterprise application archive

- The EAR file is deployed and installed using the administrative console or WSADMIN scripting interface

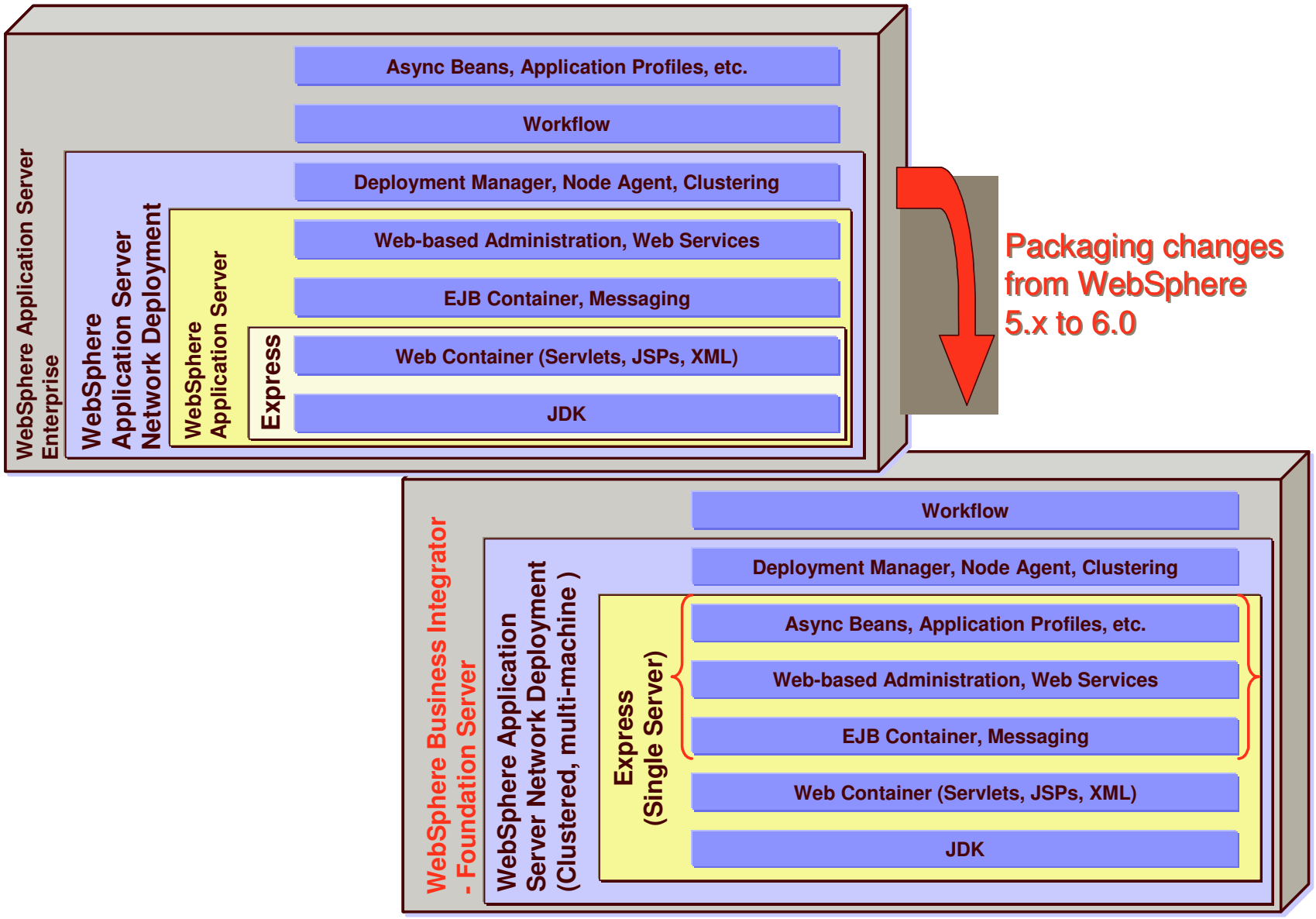
- WebSphere Application Server on OS/390 or z/OS

- Management is through MVS operator commands, the administrative console or vendor products that use the JMX API.
- SMF records are available as well.
- A number of performance monitoring tools can be employed.

WebSphere Application Server

- Runs as a threaded model
- Supports a component based architecture
  - Servlets - Base component that extends a specific Java class HTTP Servlet. All HTTP requests will be initially processed by a Servlet or a JSP. Should contain minimal business logic. Acts as a controller to call EJB's and JSP's
  - Java Server Pages (JSP's) - The component that is used to dynamically build the data sent back to the requester. This is usually dynamic HTML
  - Enterprise Java Beans (EJB's) - The component where the business logic and data access is contained. There are three main types of EJB's
    - Session Beans
      - Can be stateful or stateless
    - Entity Beans
      - Can use container managed persistence (CMP) or bean managed persistence (BMP)
    - Message Driven Beans
      - Get and process JMS messages from a queue
  - These components run in containers. Servlets and JSP's in a Web container and EJB's in an EJB Container

WebSphere Application Server

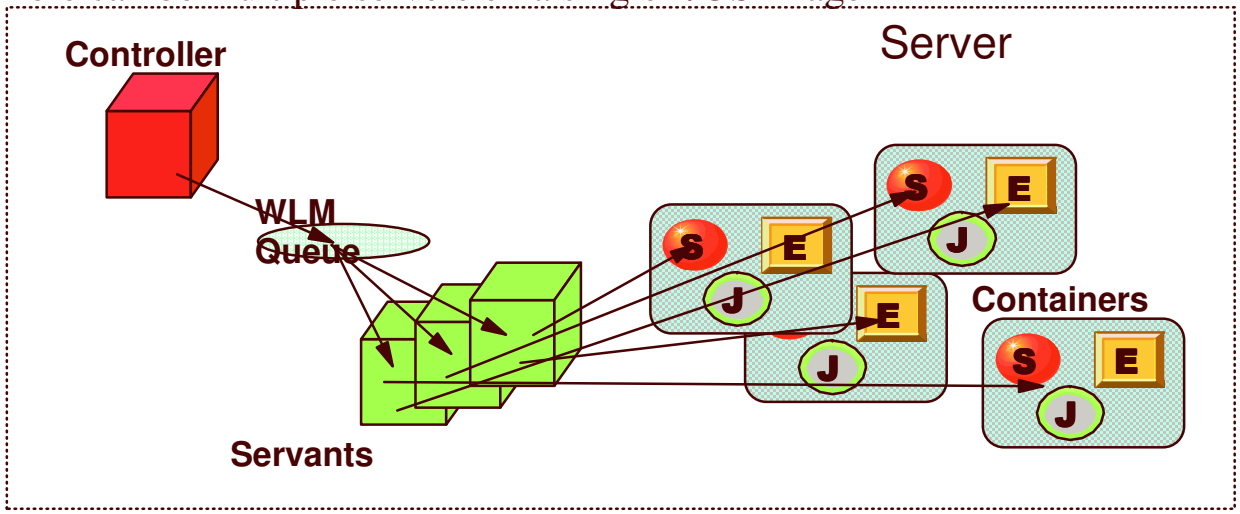


Application Servers connect to data/transactions

- Access Databases using JDBC
  - Vendor Neutral interface
  - Requires a JDBC driver
    - DB2
    - Oracle
    - Merant
- Access transactions
  - J2EE Model
    - J2EE Connector Architecture Connectors (IMS Connect, CICS Transaction Gateway)
    - JMS (Java Message Service)

Basic Runtime Structure on z/OS

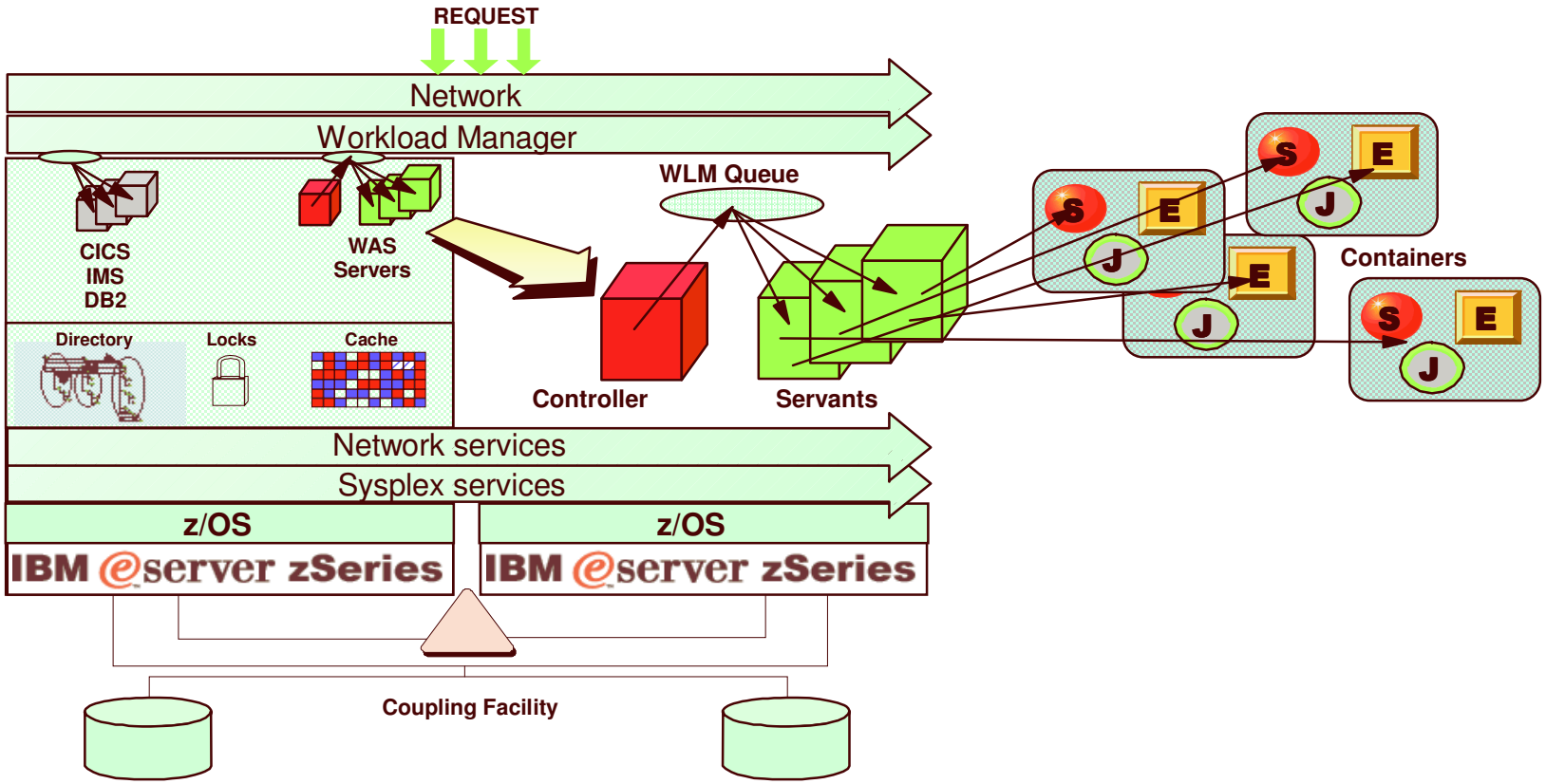
- **The basic execution environment for WebSphere on z/OS is a server. A server is a Controller/Servant configuration. Each is an address space**
  - For every Controller is one or more Servants
  - The Controller is the protocol entry point. Multiple protocols are supported: HTTP, HTTPS, IIOP and JMS.
  - The Servant is where the components execute
  - With a WLM Queue between them
  - There can be multiple servers on a single z/OS image



WebSphere Application Server for z/OS

- Optimized for z/OS
  - that's why it requires RRS, Logger, WLM Goal mode
- SMP/E installed with an ISPF dialog for the install customization
- Multiple processes (address spaces)
  - **Controller - distributes the work among server regions**
  - **Servant - does the actual work**
    - **Has 2 types of containers, a Web Container and an EJB Container**
  - **Other Daemons** for control
- Does not run under the USS Shell but does use some Unix System Services
- Configuration is performed with an administrative console (browser based) or script (JACL, Python)

Scaling



WebSphere Application Server is fully Sysplex-enabled

You can have one or more Controller/Servant configurations setup on each z/OS image in a Sysplex.

Basic Operation on z/OS

- HTTP Traffic using a Web server on z/OS or another platform
  - The client Request comes into the HTTP Server and is processed by the WAS Plugin
  - The WAS Plugin redirects the request to WebSphere using HTTP
  - A Controller in WebSphere receives the request and puts the request on a Workload Manager queue
  - A Servant takes the request off the queue and starts the requested Servlet or JSP
  - The response is passed back through the Web server and to the client
- HTTP Traffic using the HTTP Transport
  - The request comes directly into the Controller and is placed on the WLM queue and processed exactly like the above HTTP method
  - The HTTP Transport is not a full Web server and there is usually a full Web server that acts as a proxy to get the request to the HTTP Transport
- IIOP Traffic
  - The client request is usually an RMI (remote Method Invocation) over IIOP to an EJB
  - The Request comes directly into a Controller and is placed on a WLM queue
  - A Servant takes the request off the queue and starts the EJB requested and the results are returned to the client
- JMS Traffic
  - The client puts a message on a message queue
  - The Controller connected to the message queue places the message on a WLM queue
  - A Servant takes the message off the queue and invokes an MDB EJB.
  - Results are not necessarily returned to the client

Security

- Traditional security model
  - You have a userid that is connected to one or more RACF groups
  - Logon with userid and have access to applications and resources based on userid and group
- WebSphere security model
  - Access can be controlled at the level of a component or specific methods within the component
    - Might have authority to account.getBalance() but not account.makeWithdrawal()
  - Access is based on Roles
    - Roles are like RACF groups, each user could be in one or more roles
    - Principals are like userids
      - Each thread has Subject containing one or more principals
      - Principal contains information about caller's identity
      - Could be non-RACF as well as RACF identity
    - Works even if end user has no principles
  - Not the same as an ACEE
    - ACEE still controls access to non-Java resources (like datasets)

Security (cont.)

- Additional concepts
  - RunAs - This option allows the application to "Run" as the identity of the caller, role, or server.
  - sync-to-os-thread - This option causes the application to synch the thread of execution (TCB's ACEE) to the RACF identity associated with the RunAs statement.
    - If RunAs is set to caller (the default), then the thread would have the callers RACF id,
    - If RunAs is set to role, then the thread would have the RACF identity associated with a particular role (using the EJBROLE profile)
    - if RunAs is set to server, then the thread would run under the servant's RACF identity
      - which is the normal situation anyway
  - res-auth - This set for the application to identify who should control the resource authentication setup.
    - If set to application the developer would have to code a getConnection(<userid>,<password>)
      - JDBC (DB2) JMS (MQ) or J2CA (CICS, IMS)
    - If set to container, then WebSphere will pass the RunAs identity to the back end
- To find out more, attend sessions 1744, 1745 and 1746 on Thursday

What's new in V6.0

- Always new features and functions
- SDK 1.4.1
- Web Services enhancements
- Programming model extension changes
- Business process choreography enhancements
- For more information, attend 2931 on Tuesday

Web Services

- Say you were a store that wanted to sell pencils on the network but you didn't want to setup the payment process. So you need to find someone that will process the payments for you. In this case you'd be a Service Requester. Your online application would have to
  - Lookup the name and location of the service you need to use
- Say your a bank and you want to provide a service to anyone who needs to accept payments, but you don't want to tie your service directly to anyone. In this case you'd be a Service Provider. You would have to
  - Build you service and register it for someone to lookup
- Both of the above need a middle man to handle the directory and support the registration of services and the lookup. This would be the Service Broker. You would have to
  - Make your service available to the providers and requesters

Each of these three services requires a set of standards and definitions that is commonly called Web Services. Some development groups are using parts of the standard, i.e. XML or SOAP over HTTP and they say they are using Web Services

Web Services: Key Functions

Service Broker

- Owns a searchable repository of service descriptions
- Service Providers publish their services to the broker
- Service requesters access broker to find services

Service Provider

- Owns a group of services
- Provides applications as Web Service
- Publishes their services

Service Requester

- A client that requires a service
- Finds a matching service
- Invokes the service

J2EE Standards: JSR 101, JSR 109
Web Services Gateway, UDDI

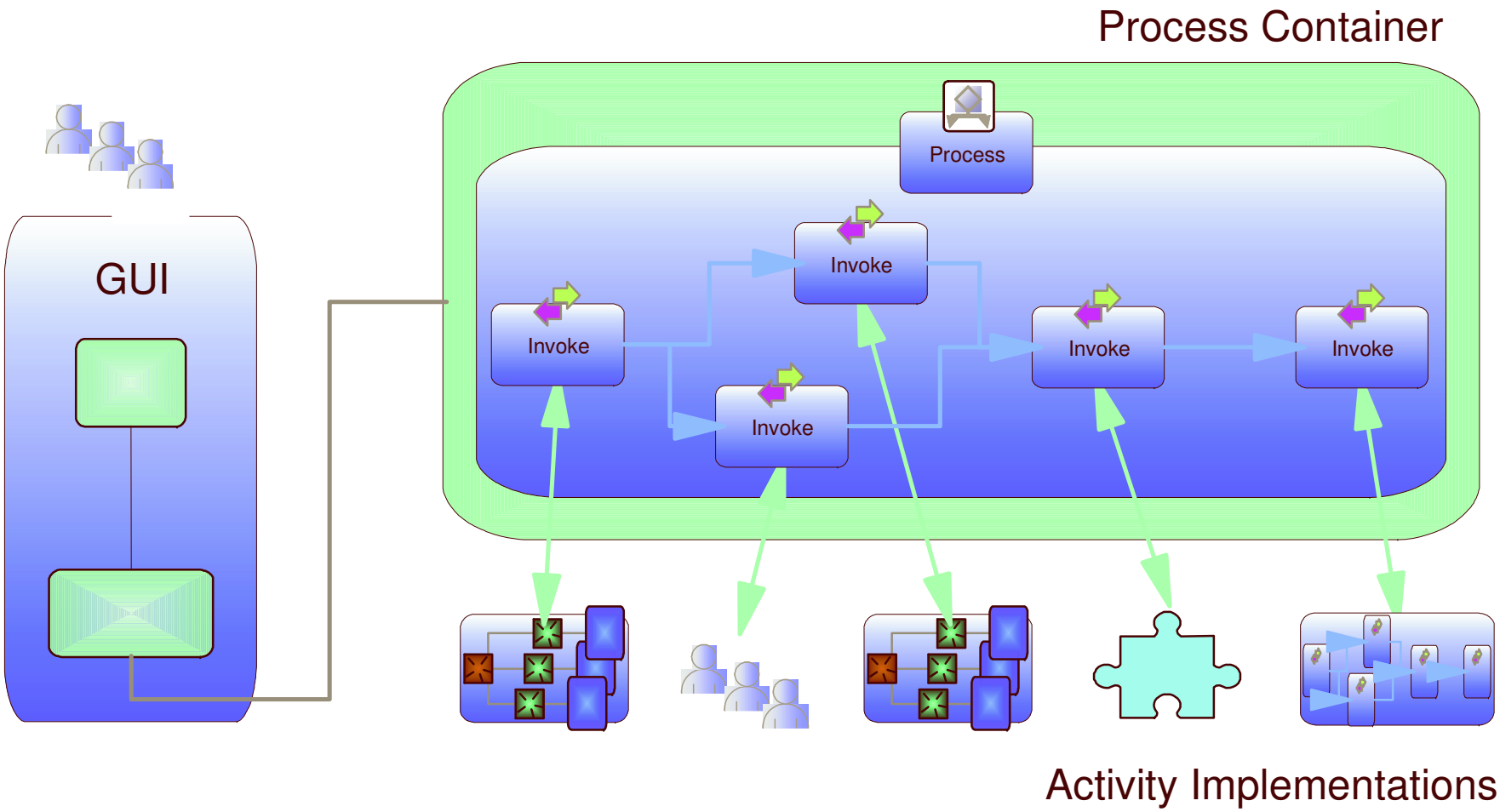
Web Services: What Does It Take?

- A structured way for exchanging information
  - XML - Extensible Markup Language
  - Provides a platform/vendor neutral way to structure data
- A service access protocol
  - SOAP - Simple Object Access Protocol
  - Provides a platform/vendor neutral application communication protocol
- A way to catalog and describe services
  - WSDL - Web Services Description Language
  - Uses XML to describe a Web service
- A way to advertise and find out about available services
  - UDDI - Universal Discovery Description and Integration
  - A structured directory or registry used for publishing and finding a Web Service

Programming model extensions

- Advanced CMP
- Container Managed Messaging (CMM)
- Asynchronous beans
- Activity sessions, last participant support
- Startup beans
- Business rule beans
- I18N
- etc.

Business process choreography



What is WebSphere?

- IBM has branded several products under the WebSphere name. They tend to fall into 4 categories
  1. **Development Tools**
     - **Rational Application Developer (follow-on to VisualAge for Java and WebSphere Studio, including WebSphere Studio Application Developer)**
  2. **WebSphere Application Servers**
     - **Several editions**
     - **Some available on specific platforms**
  3. **WebSphere Extensions**
     - **WebSphere Business Integration Server Foundation**
     - **WebSphere Commerce Suite**
     - **WebSphere Portal Server**
  4. **Management and analysis tools**
     - **WebSphere Studio Application Monitor**
     - **WebSphere Studio Workload Simulator**

The list is constantly changing!

Where do you go from here?

- Back to Assembler/COBOL/REXX ?
  - but we'd miss out on all capabilities that WebSphere brings to the table, So...
- You can get additional information on the internet and in RedBooks (http://www.redbooks.ibm.com)
  - **Make sure you bookmark**

    http://www-306.ibm.com/software/webservers/appserv/zos_os390/support/
  - **Check out the online InfoCenter and books**

    http://www-306.ibm.com/software/webservers/appserv/was/library/
- You can also use the Sun tutorial on Java components at http://java.sun.com/learning/new2java/index.html

WebSphere User Group Community – www.websphere.org

Where the WebSphere dummies hang out!

WebSphere User Groups
Global network of local activity of large and small customers, developers, consultants, vendors and IBM technical, support & sales staff …. all **endorsing, constantly evaluating, learning and entrenching** WebSphere tools, product and solutions to solve business problems

A community of over 8100 customers and business partners (~90 WebSphere User Groups)

Why is this significant?

One of the largest collection of WebSphere software users and business partners enhancing their WebSphere software skill base…

- **communicating openly**
- **expressing ideas**
- **sharing experiences**

Take advantage of …

- **technical expertise**
- **education opportunities**
- **continuous WebSphere dialogue**

How to Get Involved!

- Register as a Virtual WebSphere User Group member.
- Join a User Group in your area and personally engage with other WebSphere users and developers
- For IBMers, become an IBM liaison for a User Group within your area
- Visit www.websphere.org or contact info@groupintelligence.com

Acronyms (in order of appearance)

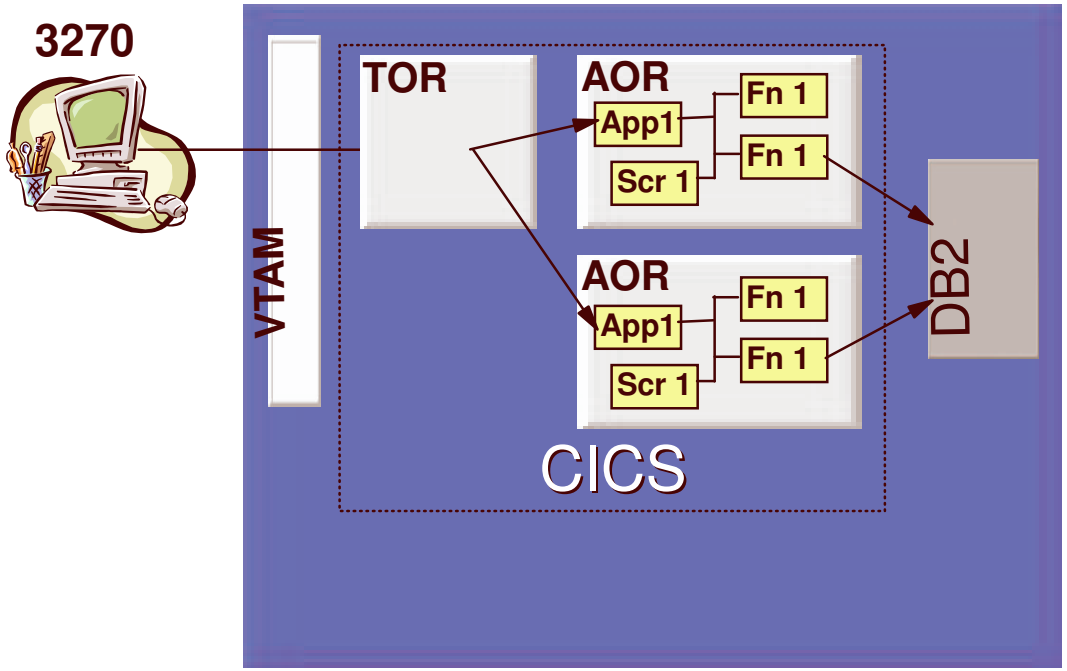
- CERN
  - Conseil European pour la Recherché Nucleaire (European Laboratory for Particle Physics; Geneva, Switzerland)
- NCSA
  - National Center for Supercomputing Applications
- IIS
  - Internet Information Server (Microsoft)
- ICS
  - IBM Connection Server
- ICSS
  - IBM Connection Secure Server
- DGW
  - Domino Go Web server
- IHS
  - IBM HTTP Server
- CGI
  - Common Gateway Interface
- API
  - Application Programming Interface
- GWAPI
  - Go Webserver API

Understanding the Transition

- The J2EE space is introducing many new acronyms and concepts, but many have parallels to those environments you are used to.
- CICS applications have an entry point Usually a (Terminal Owning Region or TOR), 3270 screens (using BMS maps), and services to support file access (File Owning Regions). These modules or components run within CICS Regions that manage the environment.
- REXX End User Applications on z/OS normally use ISPF Screens for display and ISPF Dialog Manager to display the screens and to put data into them. They also use EXECIO for file IO and other connectors to do DB2 and CICS.

The following foils are meant to provide a bridge in understanding WebSphere. They are not meant to compare the capabilities of CICS or REXX/ISPF to WebSphere

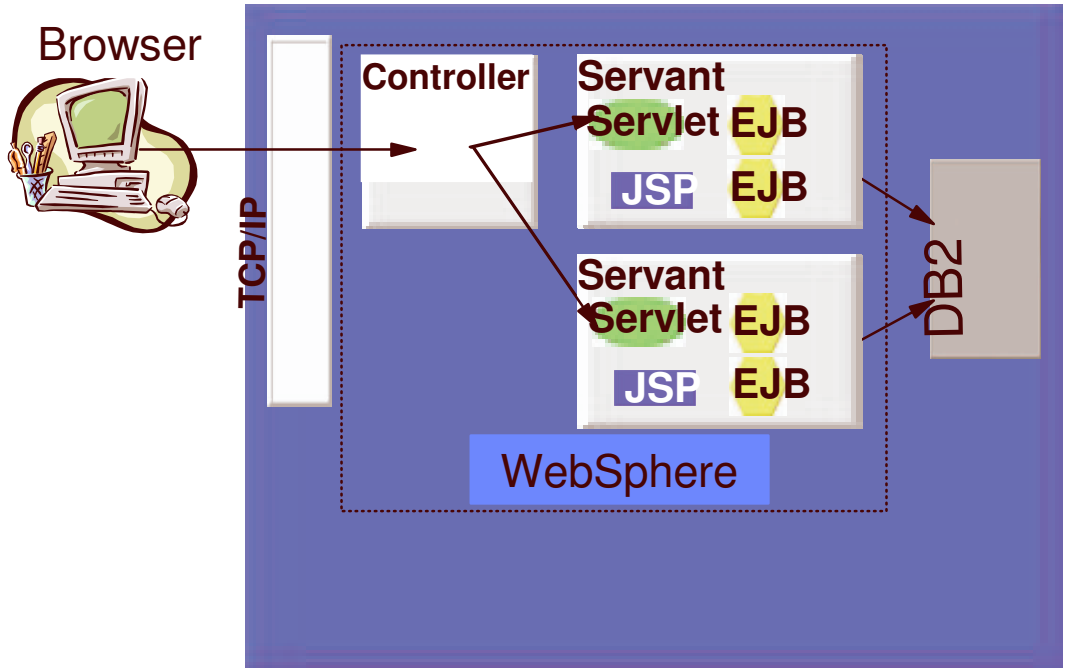
Traditional CICS Application



- **TOR - Terminal Owning Region**
- **AOR - Application Owning Region**
- **App 1- Main Application module**
- **Fn 1- Business Logic Module**
- **Fn 2- Data Access Module**
- **Scr 1- Presentation Screen module BMS mapping**

- TOR Region owns the connection to the 3270
- AOR Region is where the application (transaction) runs, Application is broken up into separate modules and connected using DLP calls,

WebSphere Application



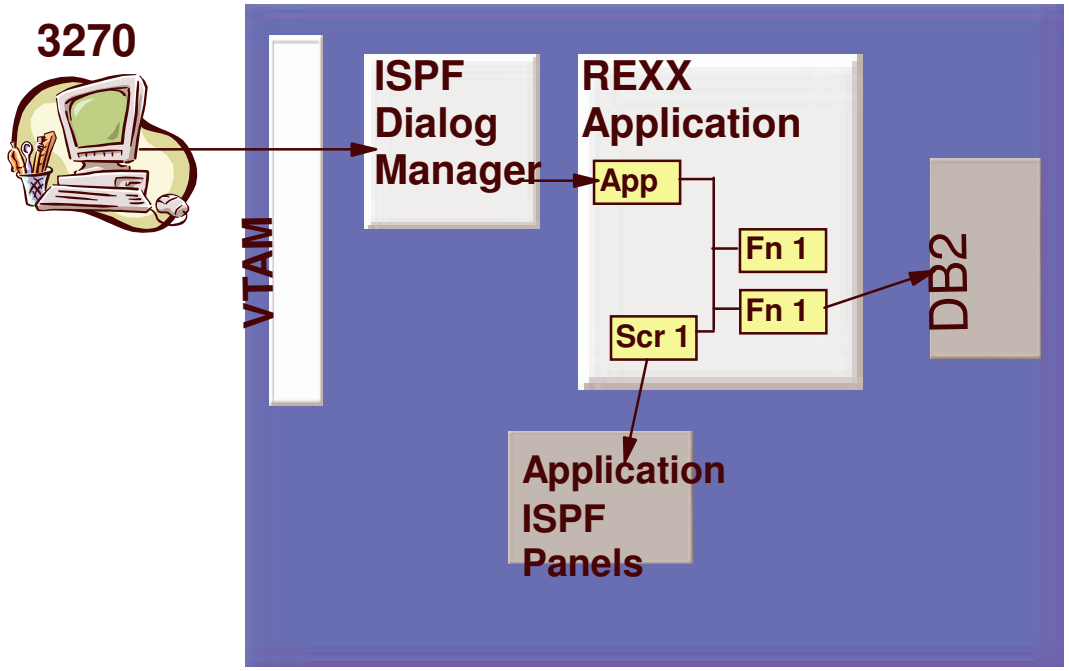
- Controller
- Servant
  - Servlet
  - JSP- Java Server Page
  - EJB
    - Session
    - Entity
    - MDB

- Controller is the entry point, from Browsers or other Clients
- Servant is where the actual application and executables are run
- Application is composed of components which use standard J2EE interfaces to communicate

Analogies with CICS Application

- TOR-Terminal Owning Region => WebSphere Controller
  - This is where WebSphere handles the incoming protocol. This can be:
    - HTTP/HTTPS - Normal browser traffic, customers are now doing XML over HTTP by having an application simulate a browser
    - IIOP - For remote EJB execution
    - JMS Java Message Service - For Message driven beans
- AOR-Application Owning Region => WebSphere Servant
- CICS Main Application => Java Servlet
- Modules for Business Logic or Data Access => EJB
- BMS Map => Java Server Page

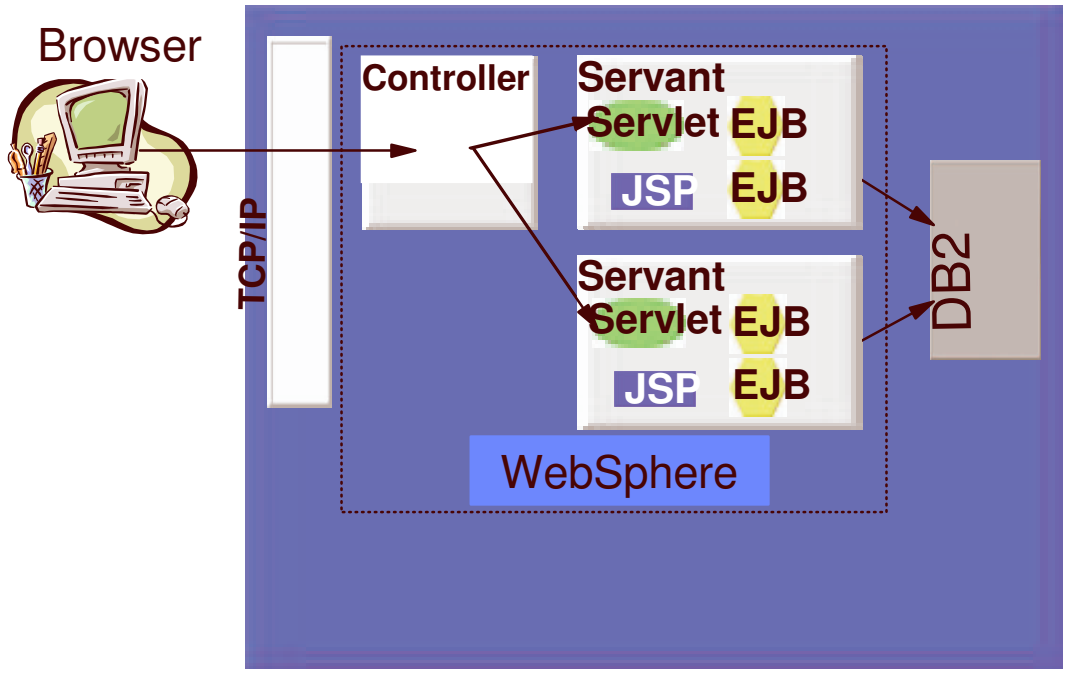
Traditional REXX ISPF Application



- ISPF Dialog Manager
- App - Main REXX Application Exec
- Fn 1- REXX Function with Business Logic
- Fn 2- REXX Function for Data Access
- Scr 1- REXX Function to setup dynamic screen data and call ISPF to display the 320 panel

- ISPF Dialog Manager owns the connection to the 3270
- Application is broken up into separate EXECS and functions and connected using normal function calls

WebSphere Application



- Controller
- Servant
  - Servlet
  - JSP- Java Server Page
  - EJB
    - Session
    - Entity
    - MDB

- Controller is the entry point, from Browsers or other Clients
- Servant is where the actual application and executables are run
- Application is composed of components which use standard J2EE interfaces to communicate