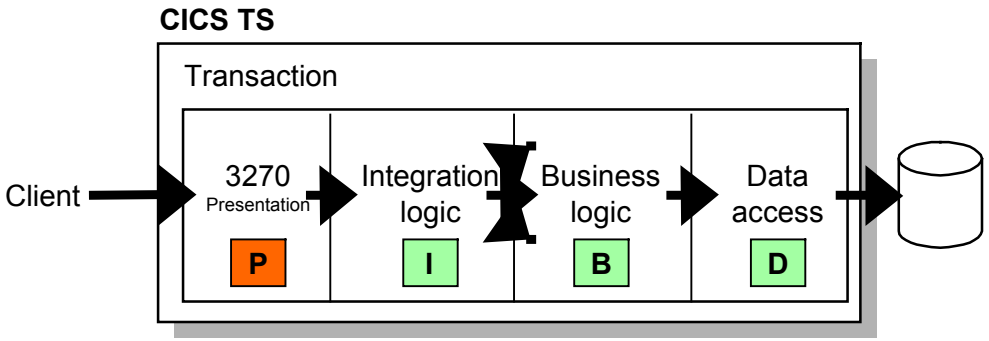
IBM
Software Group
Modern Application Architecture
SOA : Part 2
Paolo Chieregatti
Certified IT Specialist
paolo.chieregatti@it.ibm.com
© 2007 IBM Corporation

Agenda

- **Introduction**
- **Middle Tier**
  - WebSphere Application Server
    - J2EE
    - Servlets, JSP's and JSF
    - EGL
- **Client**
  - HTML
- **Connectivity**
  - Web Services, XML, SOAP, WSDL
- **Business Tier**
  - CICS
  - COBOL

© 2007 IBM Corporation

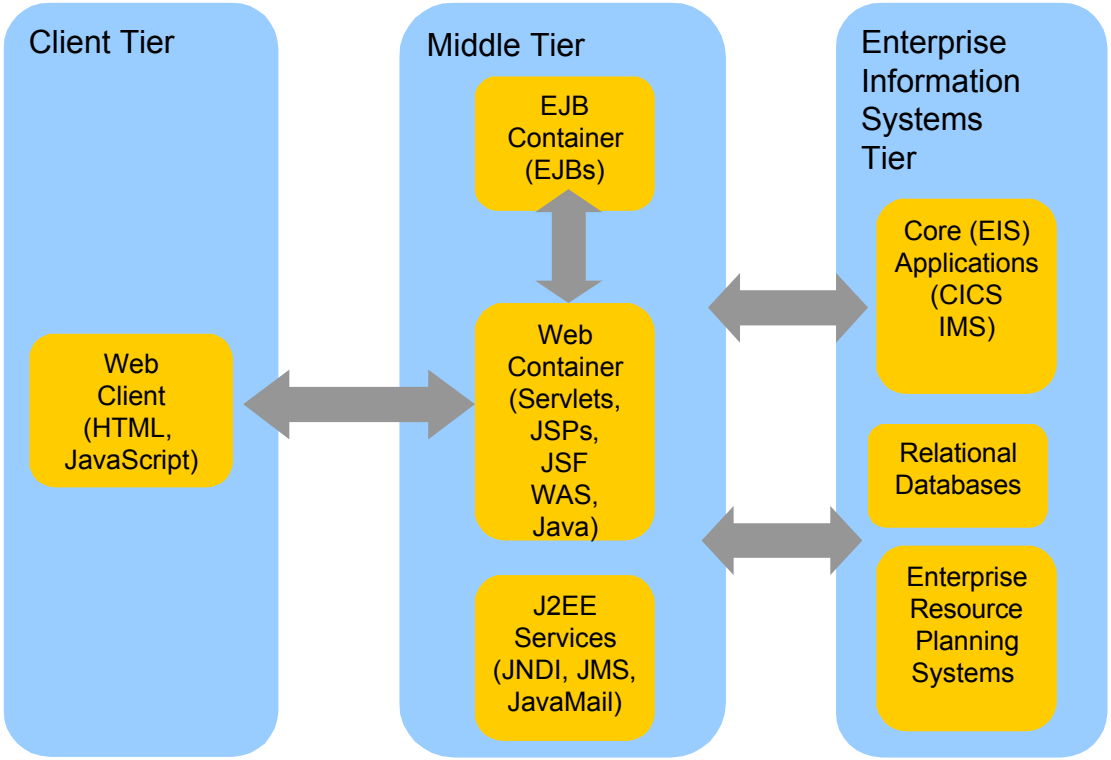
Modern “CICS” architecture



- **Best practice in CICS application design is to separate key elements of the application, in particular:**
  - Presentation logic — 3270, HTML, XML
  - Integration or aggregation logic — Menu, router, tooling
  - Business logic — COBOL, PL/I, Reusable component
  - Data access logic — VSAM, DB2, IMS, …
- **Provides a framework for reuse and facilitates separation of concerns, clear interfaces, ownership, and optimisation**

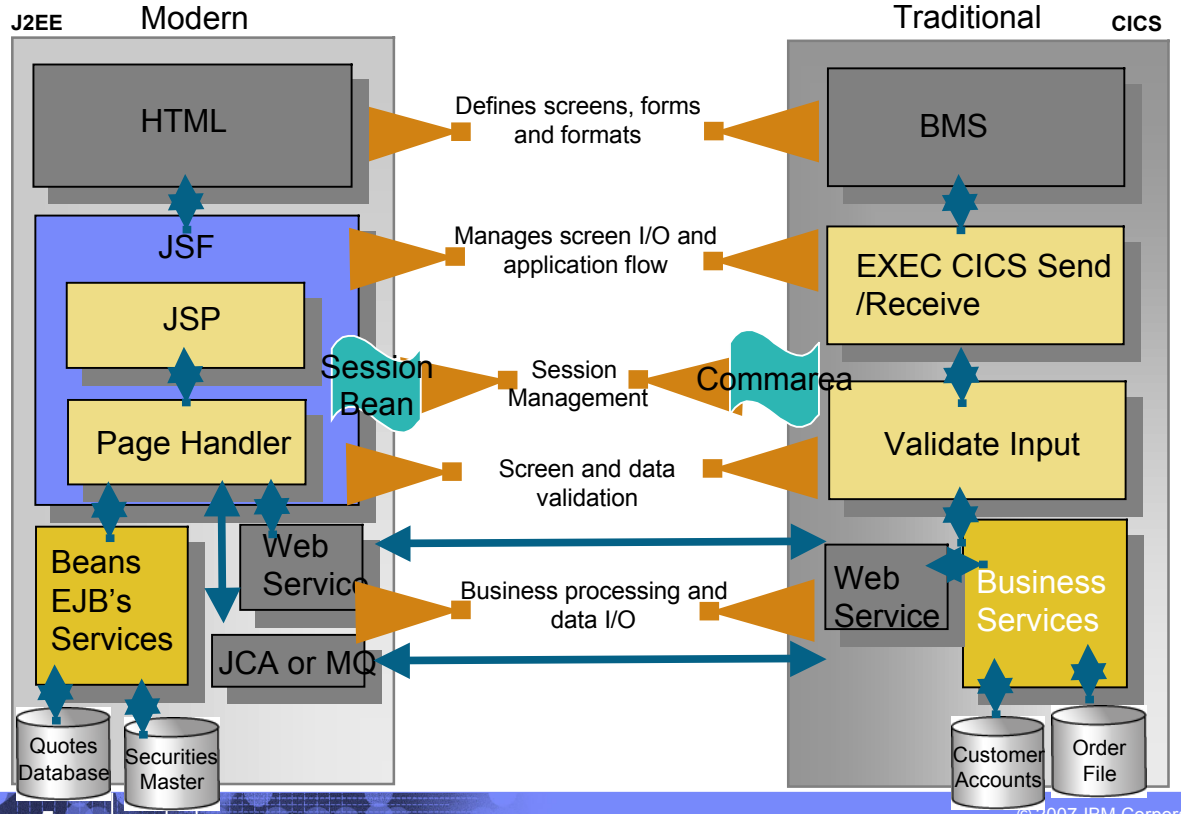
© 2007 IBM Corporation

“Modern” Multitier Architecture



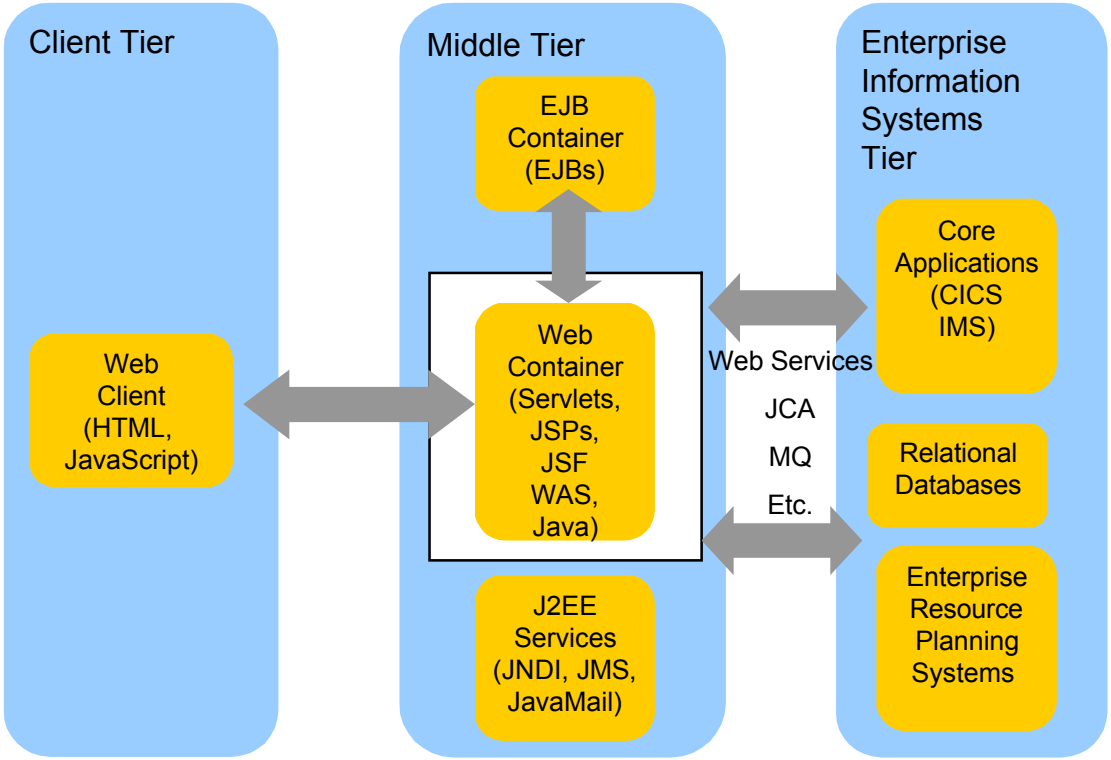
© 2007 IBM Corporation

It's not that different



© 2007 IBM Corporation

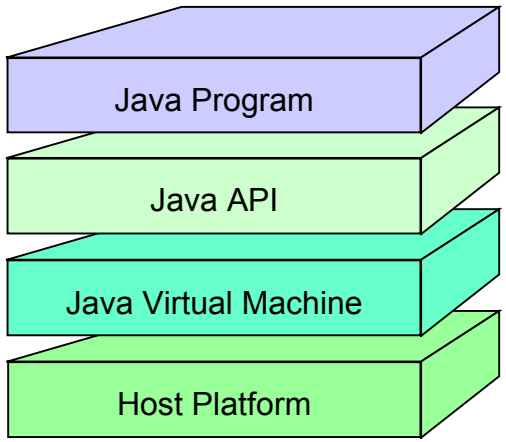
The Middle Tier



© 2007 IBM Corporation

The Java platform

- Java is an object-oriented programming language developed by Sun Microsystems
- Java has a set of standardized class libraries that support predefined reusable functionality
- Java has a runtime environment that can be embedded in Web browsers and operating systems
- Many popular UI / Session frameworks are built on Java processing



© 2007 IBM Corporation

Procedural and object oriented approaches – example

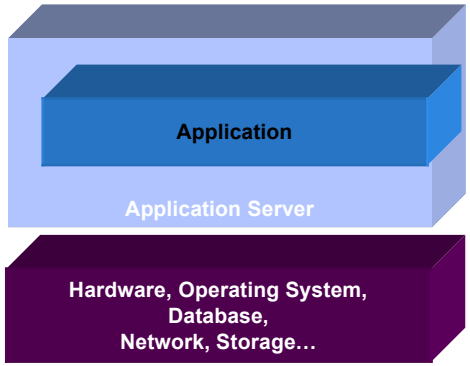
- **System requirement**
  - Banking system model withdrawing money from a savings account
- **Procedural approach**
  - Identify where the data is stored
  - List the algorithmic steps necessary to perform the action
- **Object approach**
  - Identify what objects are involved; these objects will directly relate to real life objects (Bank, SavingsAccount, Teller and Transaction)
  - Show how these objects interact:
    - To enforce business rules for withdrawals
    - To modify the balance

Both have advantages in SOA – in the right place

© 2007 IBM Corporation

What is an Application Server?

- **Provides the infrastructure for running applications that run your business**
  - **Insulates** applications from hardware, operating system, network…
  - Provides a common environment and programming model for applications
    - **Write once, run anywhere** (J2EE)
    - Platform for developing and deploying **Web Services**
  - Provides a **scalable, reliable** transaction engine for your enterprise



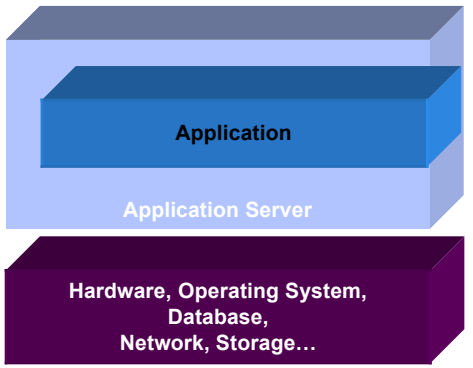
What is WebSphere Application Server?

- **WebSphere Application Server is a platform on which you can run Java-based business applications**
- **It is an implementation of the Java 2 Enterprise Edition (J2EE) specification**
- **It provides services (database connectivity, threading, workload management, and so forth) that can be used by the business applications**

© 2007 IBM Corporation

What is J2EE?

- **J2EE – Java 2 Enterprise Edition**
  - A run-time platform used for developing, deploying, and managing multitier server-centric applications on an enterprise-wide scale

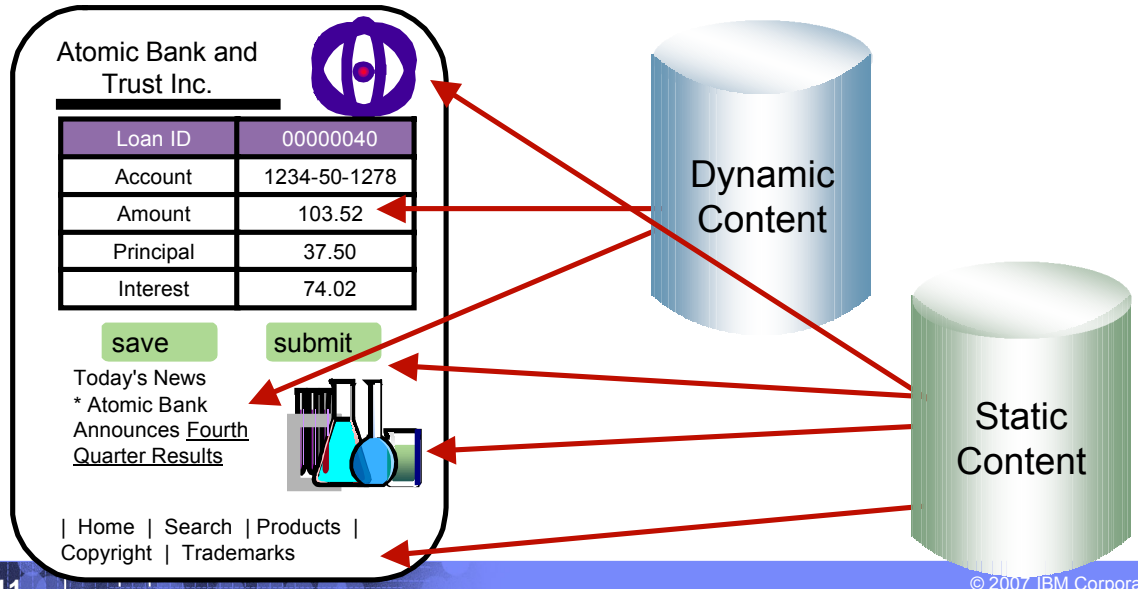


- **J2EE defines four types of components which must be supported by any J2EE product**
  - Applets
    - Graphical Java components which typically execute within a browser
    - Can provide a powerful user interface for other J2EE components
  - Application client components
    - Java programs which execute on a client machine and access other J2EE components
  - Web components
    - Servlets and JavaServer Pages
    - These provide the controller and view functionality in J2EE
  - Enterprise JavaBeans
    - Distributed, transactional components for business logic and database access

© 2007 IBM Corporation

Web page content

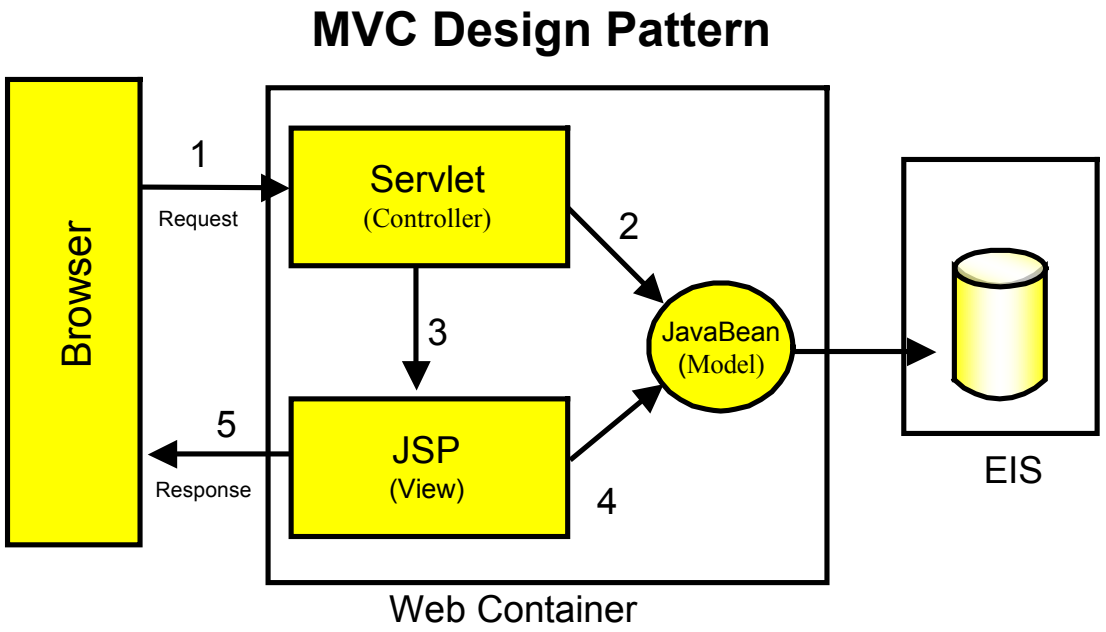
- **Content delivered to a client is composed from:**
  - Static or non-customized content
  - Customized content
- **Page layout and style are managed through HTML, XSL**



© 2007 IBM Corporation

Typical J2EE Web Application Model

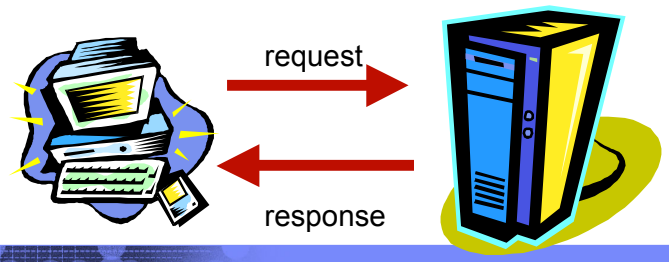
- **A request is sent to a servlet that generates dynamic content and calls a JSP page to send the content to the browser, as shown:**



© 2007 IBM Corporation

What Is a Servlet?

- **A servlet is a standard, server-side component of a J2EE application which executes business logic on behalf of an HTTP request**
  - Runs in the server tier (and not in the client)
  - A pure Java alternative to other technologies, such as CGI scripts
  - Managed by the Web container
- **Servlets form the foundation for Web-based applications in J2EE**



© 2007 IBM Corporation

A Simple Java Servlet Example

```
package com.ibm.example.servlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.ServletException;
import java.io.IOException;
import java.io.PrintWriter;
public class VerySimpleServlet extends HttpServlet {
  public void doGet(HttpServletRequest request,
                    HttpServletResponse response)
                         throws ServletException, IOException {
        String browser = request.getHeader("User-Agent");
        response.setStatus(HttpServletResponse.SC_OK);  // default
        response.setContentType("text/html");          // default
        PrintWriter out = response.getWriter();
        out.println("<HTML><HEAD><TITLE>Simple servlet");
        out.println("</TITLE></HEAD><BODY>");
        out.println ("Browser details: " + browser);
        out.println("</BODY></HTML>");
}
}
```

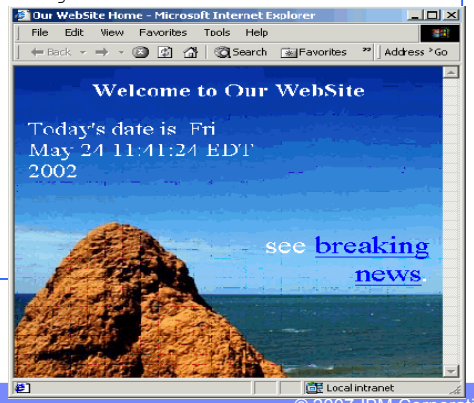

© 2007 IBM Corporation

What is JSP (JavaServer Pages)?

- JavaServer Pages is a technology that lets you mix static HTML with dynamically generated HTML
- JSP technology allows server-side scripting
- A JSP file (has an extension of .jsp) contains any combination of:
  - JSP syntax
  - Markup tags such as HTML or XML

```
<HTML>
<HEAD><TITLE>Our WebSite Home</TITLE></HEAD>
<BODY background="image.jpg" text="#ffffff">
<TABLE>
<TR><TD>
<H1>Welcome to Our WebSite</H1>
</TD></TR><TR><TD>
<H3>Today's date is
<%= new java.util.Date() %>
</H3></TD>
<TD>see <A href="breaking.html">
breaking news</A>.
</TD></TR>
</TABLE>
</BODY>
</HTML>
```

A simple JSP example



© 2007 IBM Corporation

JSP or Servlet?

- **Writing HTML code in a servlet is tedious and difficult to maintain**
- **Java code embedded in a JSP is difficult to reuse and maintain**
- **Use servlets to:**
  - Determine what processing is needed to satisfy the request
  - Validate input
  - Work with business objects to access the data and perform the processing needed to satisfy the request
  - Control the flow through a Web application
- **Use JSP pages to format and displaying the content generated by your servlets**

© 2007 IBM Corporation

What is JavaServer Faces?

- **JavaServer Faces (JSF) is a *framework* for developing Web-based applications.**
  - A framework is a skeleton or foundation of an application
    - Provides code, resources, concepts and best practices upon which applications are constructed
- **The main components of JavaServer Faces are:**
  - An API and reference implementation for:
    - representing UI components and managing their state
    - handling events, server side validation, and data conversion
    - defining page navigation
    - supporting internationalization and accessibility
    - providing extensibility for all of these features
  - A JavaServer Pages (JSP) custom tag library for expressing UI components within a JSP page
  - **EGL**, IBM's enterprise generation or *business* language supports JSF

© 2007 IBM Corporation

What is EGL?

Enterprise Generation Language (EGL)

- Is a development environment and programming language that lets users write full-function applications quickly
- Can be used to create text-based user interfaces for migration of existing applications
- Focus is on the business problem rather than on software technologies
- Is written independently of the target platform
- Can be generated into Java or COBOL programs
- Is well-suited to procedural programmers
- Is a high-level language which promotes iterative development and testing early in the development cycle

© 2007 IBM Corporation

EGL - generation

- **Runtime code generated for appropriate platform**
  - **Java** for Windows, Linux, and so forth
  - **COBOL for z/OS**
  - Uses SQL transparently
- **EGL can be used to create "full" Web-based applications including Web UI's**
  - JavaServer Faces application is generated for runtime code
  - Runs on WebSphere Application Server

© 2007 IBM Corporation

EGL – key high level language abstractions

- **Data Access:**
  - Common Verbs for data access (Get, Add, Replace, Delete)
  - Abstracts access to SQL, Indexed, Relative, Serial, DL/I, MQ, Services
  - Allows complete access to SQL statement if needed
  - Common Error Handling

```
function allLoans()
      loans LoanRec[];
      get loans;
end

function loansInFlorida()
      loans LoanRec[];
      get loans with #sql{
            select *
            from LOANREC
            where state = "FL"};
end
```

- **Validation/Editing Rules**
  - Define formatting & validation rules once in common place
  - Reuse data items for Records, screens, reports

```
DataItem Password char(10) {
      validatorFunction = "passwordValidation",
      displayUse = secret,
      displayName = "Enter your Password",
      inputRequired = yes}
end
```

- **Remote Invocation**
  - Call COBOL, RPG, C, Java
  - Linkage information separated from code…simplifies development

```
function callHelloWorldOniSeries()
      salutation char(30);
      call helloworld salutation;
end
```

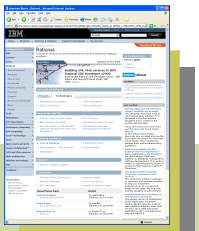
- **Transaction Control:**
  - JDBC, CICS, IMS


```
function loansInFlorida()
      startTransaction(myLoanTransaction);
      ...
      commit();
      ...
      rollback();
end
```


© 2007 IBM Corporation

EGL – simple programming model

- **Page Handlers**
  - Contain functions and data related to a .jsp
  - Should be mostly "Controller Logic"



```
PageHandler customerInfoPage {
      view = "customerInfoPage.jsp",
      title = "Customer Information",
      onPageLoadFunction = "onPageLoad"}

function onPageLoad()
      loans LoanRec[];
      get loans;
end
...
end
```

- **Report Handler**
  - Call-out to EGL "ReportHandler"
  - Open Source Reporting Framework

```
ReportHandler customerList

function afterPageInit()
...
end

end
```

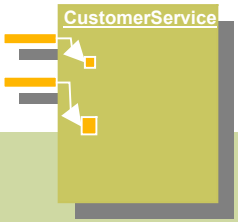
Controller Logic/User Interaction

Business Logic

- **Programs**
  - Used for single point of entry situations
  - TUI program, Batch program, GUI program

```
Program MyProgram

function myFunction()
      salutation char(30);
      call helloworld salutation;
end
```

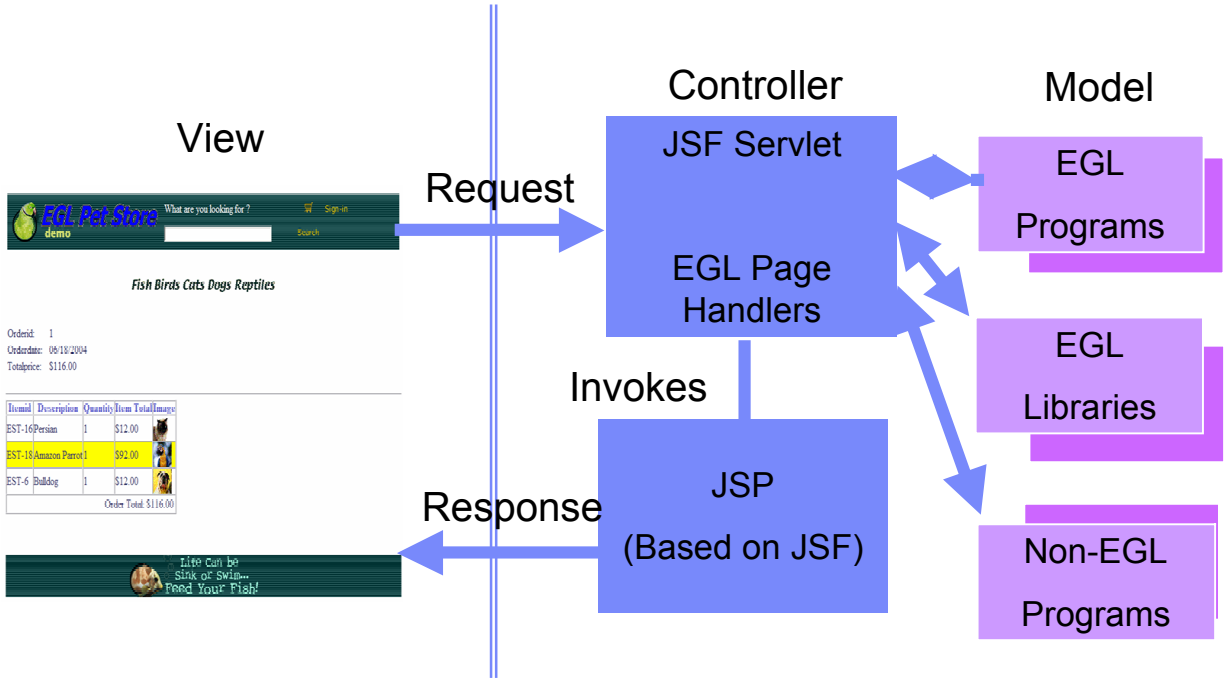
- **Services**
  - "Business Logic" for web apps
  - Multiple entry points
  - Invoke by function




```
Service Customer

Function getAllCustomers()
end
...

end
```


© 2007 IBM Corporation

EGL and JSF



© 2007 IBM Corporation

JSP's / JSF / EGL and COBOL

- **JSP's are synonymous with EXEC CICS Send Map and Receive Map processing**
  - If a CICS program only processed screens – to request business processing – or work – it would need to either Link, XCTL or Calls in COBOL.
  - JSP is a similiar concept.
- **Java Server Faces provides a framework to build UI oriented forms linked with processes such as Web Services.**
  - Performs similar function as existing CICS programs which perform send/receive processing and input validation.
- **Java server faces consist of Java Server pages – which handle the build and catching of forms and user information – and page handlers which validate information and provide control calls into back end services.**

© 2007 IBM Corporation

EGL Web – C.I.C.S. Programming Similarities

Page Data
~
BMS Map Fields

Load values from the database

"Send map"

"Receive map"

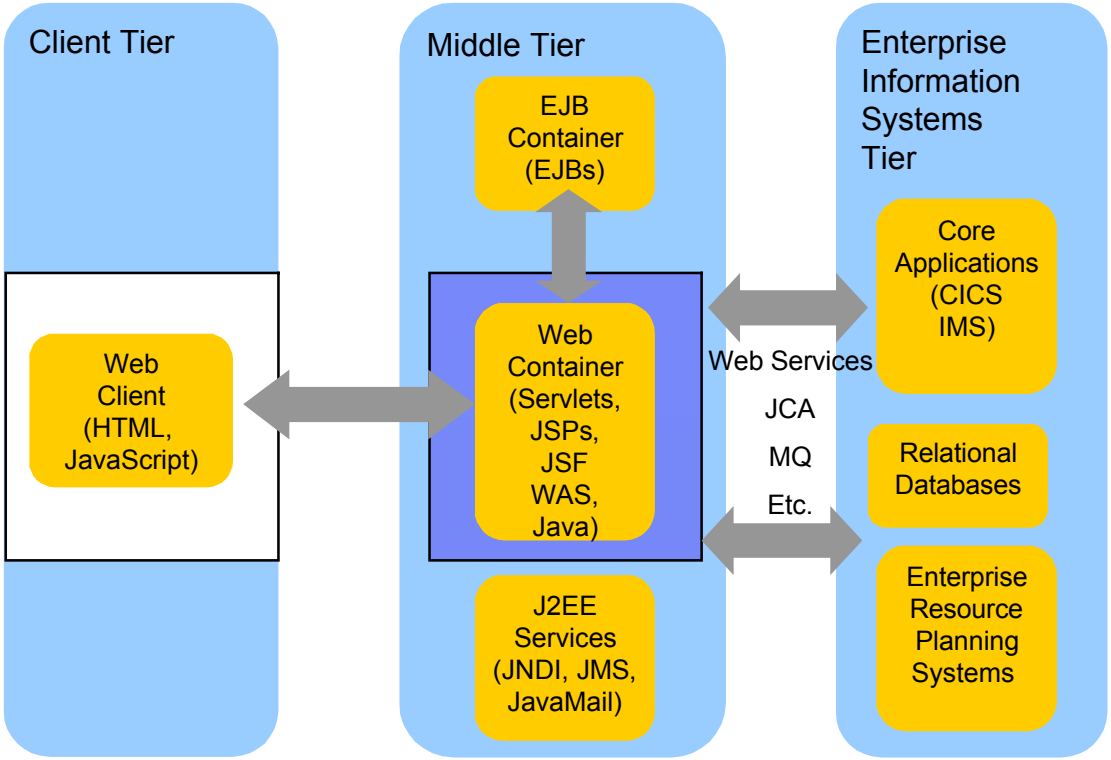
Process user-input values

Update Database

```
package pagehandlers;
import data.*;

PageHandler ordersbycustomer {view="ordersbycustomer.jsp", onPageLoad=onPageLoad}

//Page data - equivalent to I/O area for screen values
    customer Customer;
    dt  char(33);
    orders order[];
    sel int[] {selectFromList=Orders};  //Integer array - bind to Row Selection
    OrderRec Order; //Single Order record - for update of checked rows

    //vars for combo-box
    comboBoxSel char(12) {selectFromList=valueListArray,selectType=value}; //Display
    valueListArray  char(12)[];     //Temp holding array for state values
    j int ;             //Loop ctr - max number of rows in Customers dynamic array
    i int ;
    s int;

    Function onPageLoad(cid  int) //Receives control upon entry
    dt=sysvar.currentFormattedDate;
    customer.CUSTOMER_ID=cid;
    CustomerLib.getCustomer(Customer); //Load customer data from the database
    s = sqlcode;
    OrderLib.ordersByCustomer(cid, Orders); //Load order data from the database

   end

    Function updateOrders() //Receives control upon button-clicked event
        arrayMax int;   //sel (array) is created to the size of # of checked rows
        arrayMax = size(sel);   //Get this size (= # of checked rows)
        i int;  //Array loop ctr
        i = 1;  //Initialize loop ctr
        j int;  //Declare temp variable to hold indexed value in sel
        while (i <= arrayMax)   //Loop through all checked rows in Sel array
            j = sel[i];     //assign each sel[i] value to temp var.
            move  Orders[j] to OrderRec byname; //Move the fields
            orderrec.ORDER_STATUS = comboBoxSel;
            OrderLib.updateOrder(OrderRec); //Update the DB
            i = i+ 1;       //Don't forget to increment the array loop ctr
        end
```


© 2007 IBM Corporation

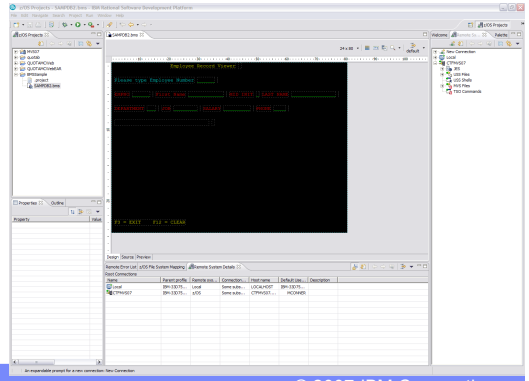
The Client



© 2007 IBM Corporation

HTML

- **HTML performs similar processing as BMS or MFS maps. It defines the screens and fields, colors, and interactions, although the technologies and implementations of course are different.**
- **Hypertext Markup Language consists of:**
  - **Hypertext.** The way of creating web documents – and of linking multiple documents together. HTML offers support for both document as well as multimedia links.
  - Tags or controls: Pieces of code that are used to create links. All browsers let you know when you've selected an active area of the screen.
    - For example <head> marks where a heading starts and </head> marks where it ends.
    - Popular tags include:
      - Text Tags – Logical structure for content
      - Link Tags – to links such as hyperlinks, image links
      - Style sheet tags – how content is rendered
      - and many more….
- **See the green screenshot – displayed inside of the WDz BMS Map Editor, together with with the BMS Macros that are input to generate the code – that upon execution causes the "green screen" to be displayed.**
  - WDz provides similar support for HTML screens



© 2007 IBM Corporation

BMS and HTML

BMS

Name and overall format of map - Includes items such as input/output, whether keyboard should be enabled, types of terminal, colors, size etc. are defined.

```
SAMPDB2  DFHMSD
    TYPE=&SYSPARM,MODE=INOUT,LANG=COBOL,STORAGE=AUTO,
    *
         CTRL=FREEKB,EXTATT=YES,TERM=3270-2,TIOAPFX=YES,      *
         MAPATTS=(COLOR,HILIGHT,OUTLINE,PS,SOSI),             *
         DSATTS=(COLOR,HILIGHT,OUTLINE,PS,SOSI)
MAP1     DFHMDI SIZE=(24,80),                                 *
         COLUMN=1,                                            *
         LINE=1
```

Headings and text fields. Defined with DFHMDF macro. You see position, length, initial value, and field attribute below.

```
     DFHMDF POS=(3,1),LENGTH=27,                              *
         INITIAL='Please type Employee Number',               *
         ATTRB=(PROT,NORM)
```

Input Fields. Defined with DFHMDF macro. You see a name (which ultimately defines storage size (and Cobol copybook field definition), and a difference with the field defined as unprotected – information can be entered.

```
EMPONUMINPUT DFHMDF POS=(3,29),LENGTH=6,                      *
         ATTRB=(UNPROT,NORM),HILIGHT=UNDERLINE
```

HTML

Headings – Overall definition, including whether Java Server faces tags will be used, a heading, and stylesheet definition.

```
<HEAD>
<%@ taglib uri="http://java.sun.com/jsf/core" prefix="f"%>
<%@ page language="java" contentType="text/html; charset=CP1252"
    pageEncoding="CP1252"%>
<META http-equiv="Content-Type" content="text/html; charset=CP1252">
<META name="GENERATOR" content="IBM Software Development Platform">
<META http-equiv="Content-Style-Type" content="text/css">
<LINK href="theme/Master.css" rel="stylesheet" type="text/css">
<TITLE>MAP1</TITLE>
```

Text headings including location definition, colors, attributes, etc.

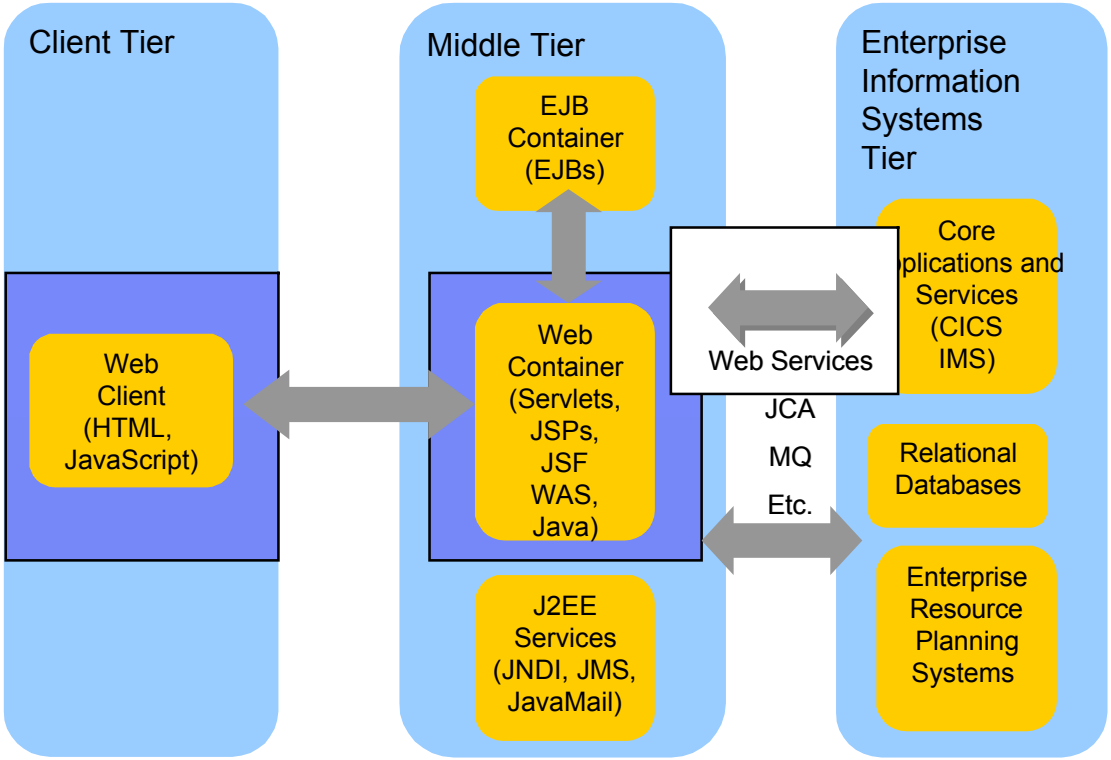
```
f:view> <BODY>
<hx:scriptCollector id="scriptCollector1"><h:form styleClass="form" dir="ltr"
    id="form1"><table><tr><td colspan="20"> </td>
<td colspan="22" nowrap><font color="#ffff00">Employee Record Viewer</font></td>
<td> </td>
<td nowrap><font color="#0000ff"></font></td>
<td colspan="36"> </td>
<tr><td colspan="80"> </td>
<tr><td> </td>
<td colspan="27" nowrap><font color="#00ffff">Please type Employee Number</font></td>
```

Input fields

```
<td> </td>
<td colspan="6" nowrap><h:inputText
    value="#{pc_MAP1Page.map1Bean.emponuminput}" required="false" style="color:
    #00ff00" size="6" id="emponuminput"></h:inputText></td>
<td> </td>
<td colspan="44"> </td>
<tr><td colspan="80"> </td>
<tr><td> </td>
```


© 2007 IBM Corporation

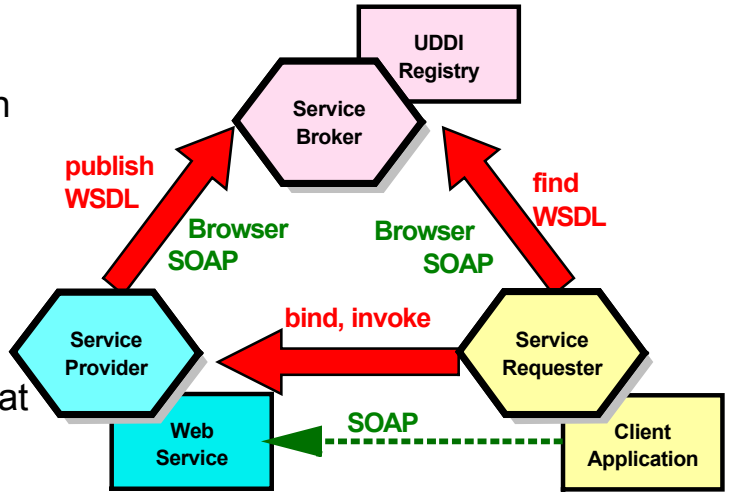
Connectivity



© 2007 IBM Corporation

Web Services

- **Architecture for**
  - Application to application
    - Communication
    - Interoperation
- **Definition:**
  - Web Services are **software components described via WSDL** that are capable of being accessed via **standard** network protocols such as SOAP over HTTP
- **WS-I.org (Web Services Interoperablity Organization)**
  - Ensure interoperability



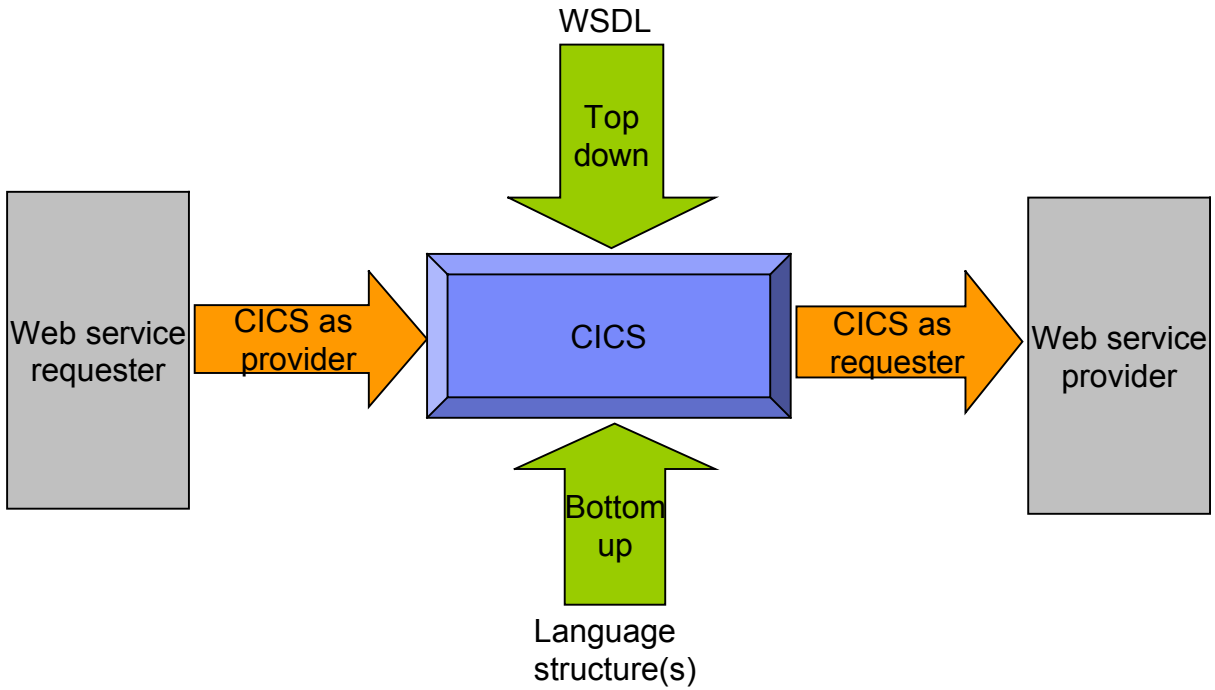
WS-I WEB SERVICES INTEROPERABILITY ORGANIZATION

uddi.org

The entire industry is agreeing on one set of standards !!

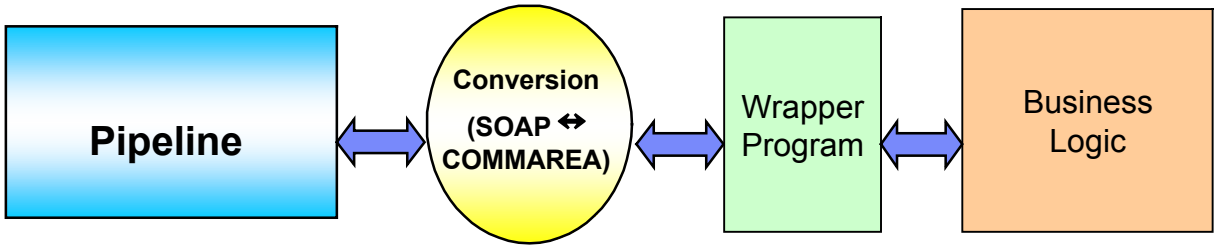
© 2007 IBM Corporation

Web Services Enablement Styles



© 2007 IBM Corporation

Where a wrapper program fits in



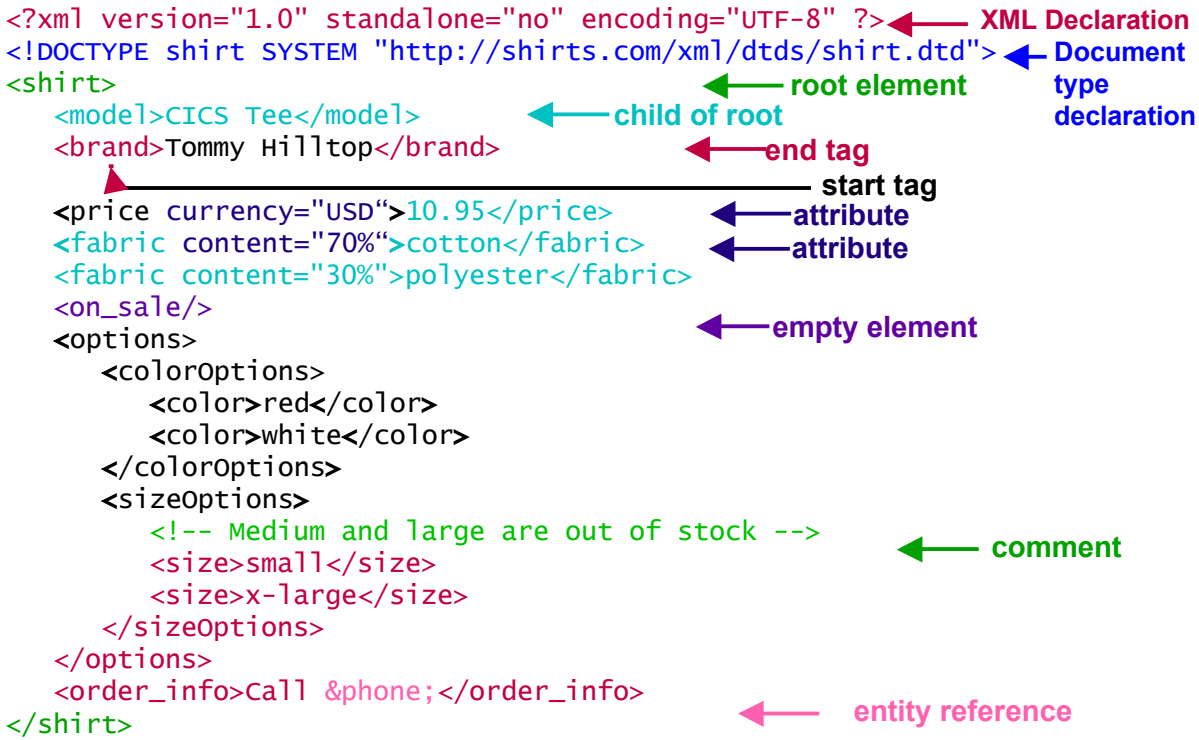
© 2007 IBM Corporation

XML Terminology

- **SOAP and WSDL are based on XML**
- **A tag / attribute based syntax**
- **Format of XML file described in**
  - **DTD** – Document Type Definition
  - **XSD** – XML Schema Definition
- **XML files are**
  - Well-formed (syntax is ok – matching tabs, etc.)
  - Valid (obeys rules in DTD or XSD) (CICS can validate)
- **Namespaces**
  - Avoids name collisions
  - A set of names (XML tags) that apply to a certain space in a document

© 2007 IBM Corporation

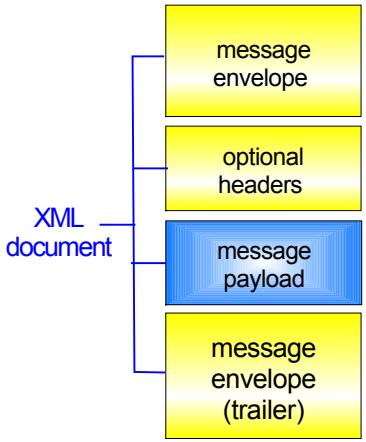
XML – Basic Parts



© 2007 IBM Corporation

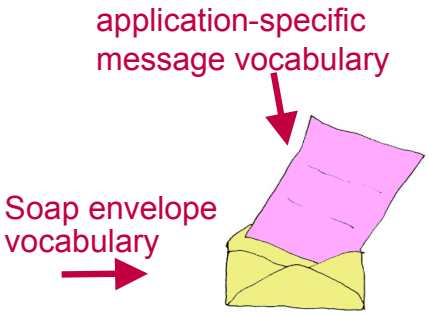
Simple Object Access Protocol (SOAP)

- An XML-based protocol for exchanging of information in a decentralized, distributed environment
- An open standard whose main goal is to facilitate interoperability
- A protocol which is not tied to any operating system, transport protocol, programming language, or component technology



- ***XML Message Envelope***
  - service requested
  - routing information
  - message type
  - date/time stamp
- ***XML Message Headers***
  - authentication
  - transaction context
- ***XML Message Payload***
  - data understood by target application
- ***XML Message Trailer***
  - closing tags
  - optional message digest

SOAP spec defines how to do this!



© 2007 IBM Corporation

SOAP: Request Message

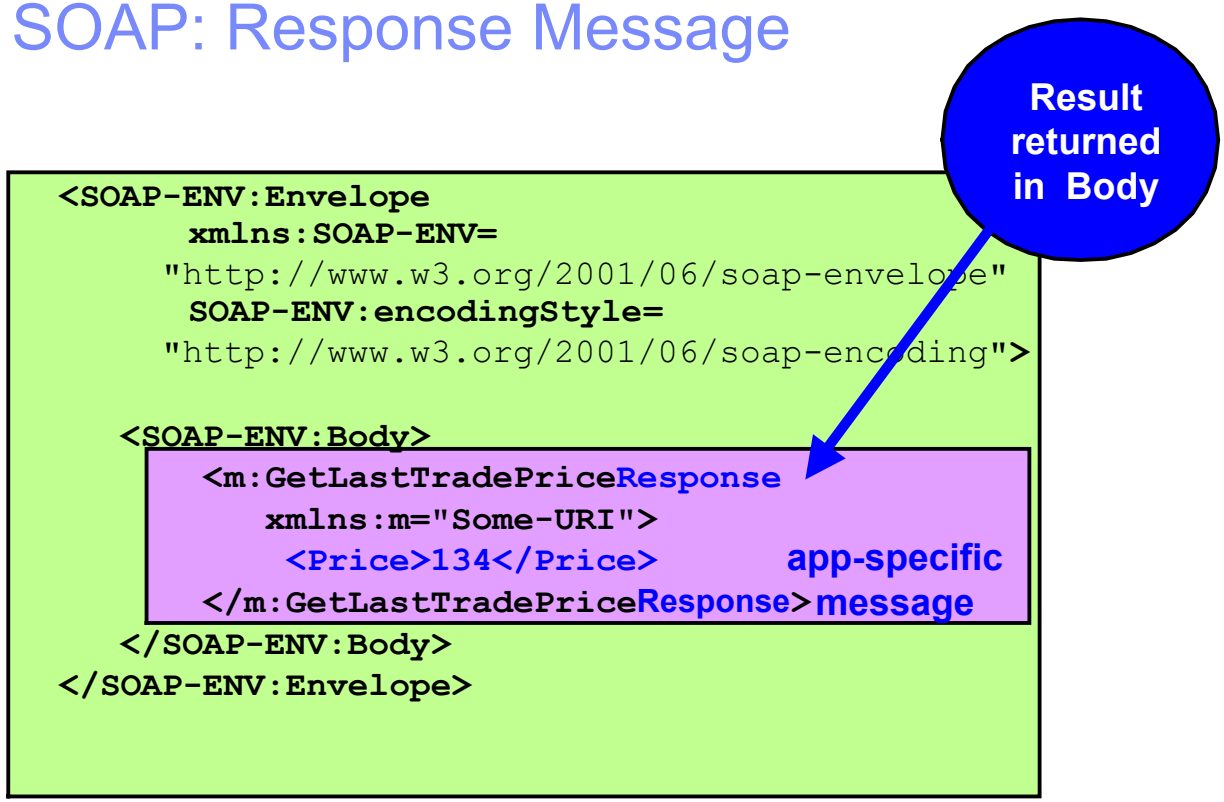
```
<SOAP-ENV:Envelope
       xmlns:SOAP-ENV=
      "http://www.w3.org/2001/06/soap-envelope"
       SOAP-ENV:encodingStyle=
      "http://www.w3.org/2001/06/soap-encoding">

   <SOAP-ENV:Body>
        <m:GetLastTradePrice xmlns:m="Some-URI">
            <symbol>IBM</symbol>
        </m:GetLastTradePrice>
   </SOAP-ENV:Body>

</SOAP-ENV:Envelope>
```

app-specific message

SOAP envelope

© 2007 IBM Corporation

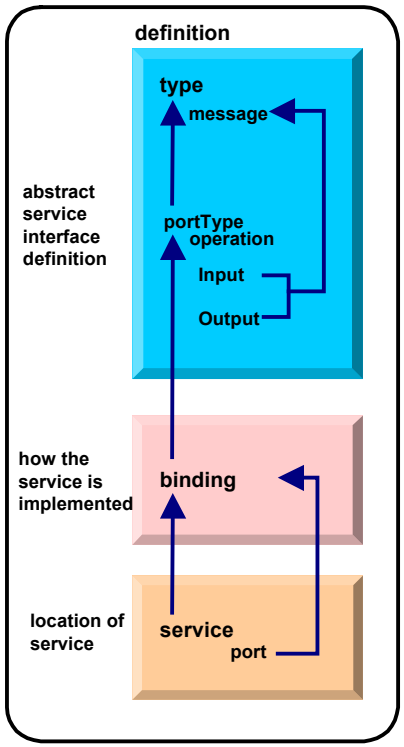
SOAP: Response Message



© 2007 IBM Corporation

WSDL - Web Service Description Language

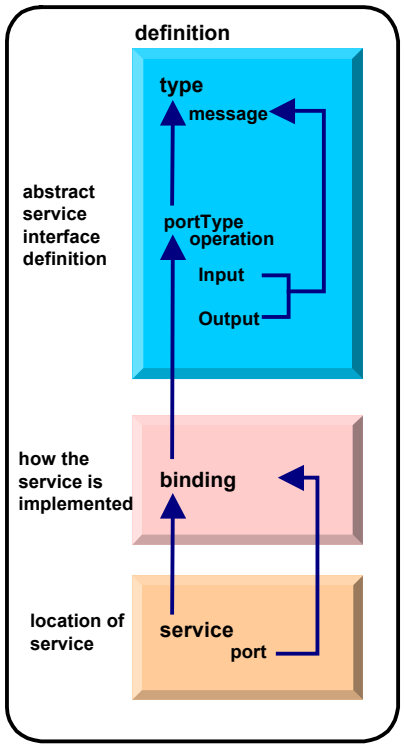
- Open Standard
- XML resume describing what a Web Service can do, where it resides, and how to invoke it
- Machine readable, generated, used by IDEs
- Similar in purpose to IDL, but in XML form
- Can be One or multiple documents
- Major sections are:
  - Service Interface (operations, input, output)
  - Service binding (protocol binding)
  - Service implementation (location of service)



© 2007 IBM Corporation

WSDL: Logical Contents

- **Service Interface**
  - Operation (business functions)
    - Input Message (0 or 1) and Output Message (0 or 1)
      - 1 or more parts
      - Parts may be simple or complex
      - Complex parts may have multiple elements
- **Service binding**
  - Definition of the physical service interface implementation
- **Service Implementation**
  - Location of the service



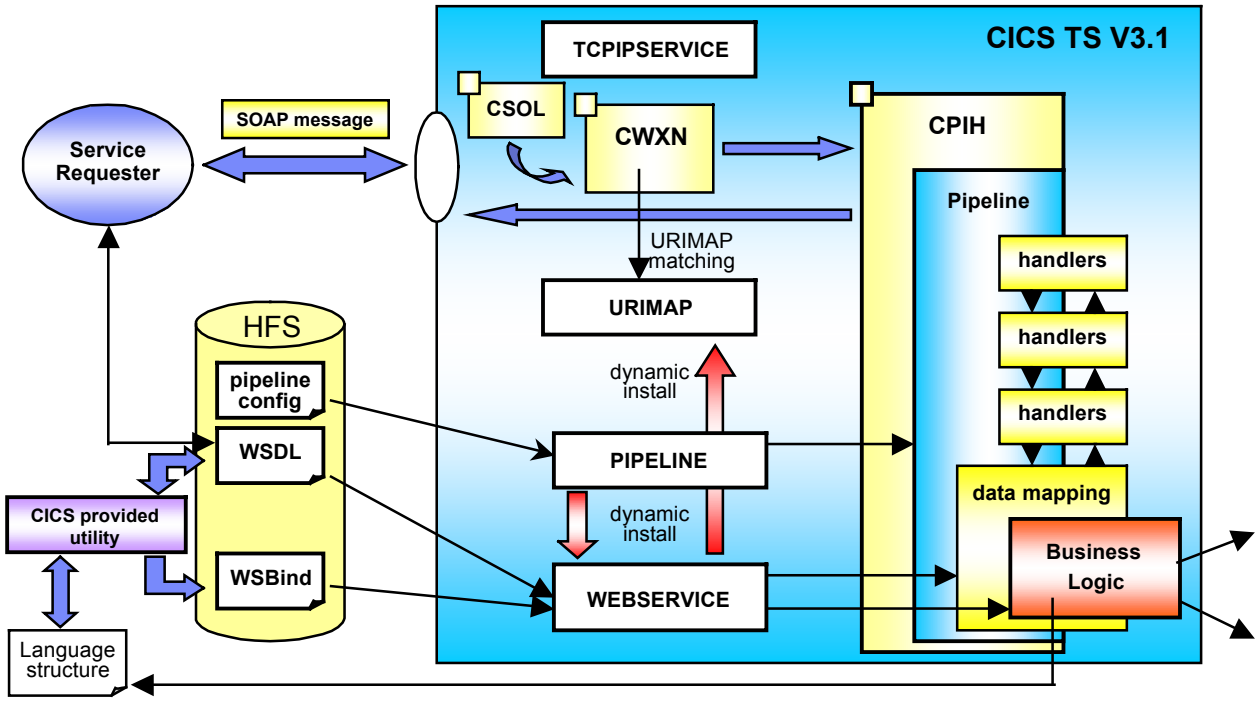
© 2007 IBM Corporation

WSDL: Physical Contents

- **Definitions – highest level tag**
  - **types** – definition of complex parts
  - **message** – a grouping of 1 or more parts
    - **parts** – simple or complex (complex points to a type)
  - **portType** – a grouping of operations
    - **operation** – correspond to business functions
      - **input** – points to input message
      - **output** – points to output message
      - **fault** – can be returned when stuff goes wrong
  - **binding** – physical associations to operations
    - **operation** – implementation of a portType operation
  - **service** – grouping of ports
    - **port** – location of associated binding

© 2007 IBM Corporation

CICS as a service provider



© 2007 IBM Corporation

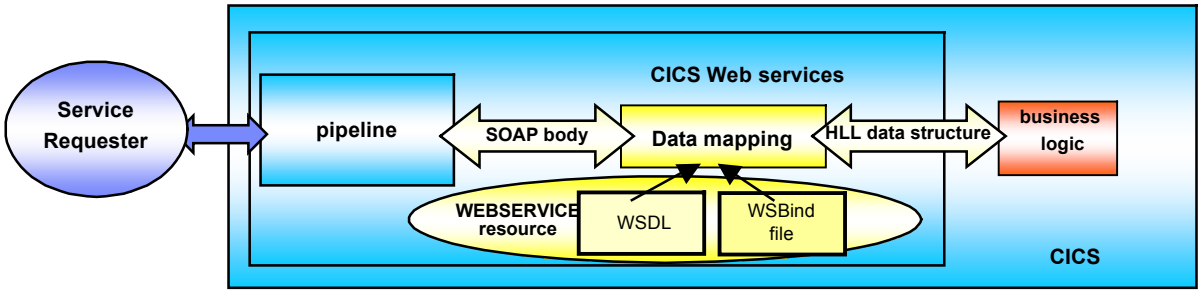
Defining the CICS Web Services Resources

- **Define a TCPIPSERVICE (or WMQ) and a PIPELINE**
- **Then install the PIPELINE definition and issue CEMT PERFORM PIPELINE SCAN**
- **CICS uses the PIPELINE definition to**
  - Locate the WSBind file
  - From the WSBind file, CICS will dynamically create a WEBSERVICE resource
  - CICS will also dynamically create a URIMAP definition
- **Can define everything individually if preferred**

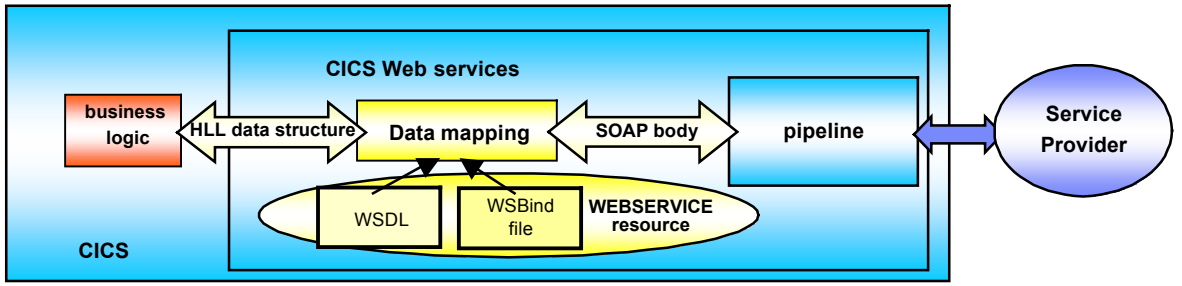
© 2007 IBM Corporation

CICS usage of the WSBind file

- **CICS as a service provider**

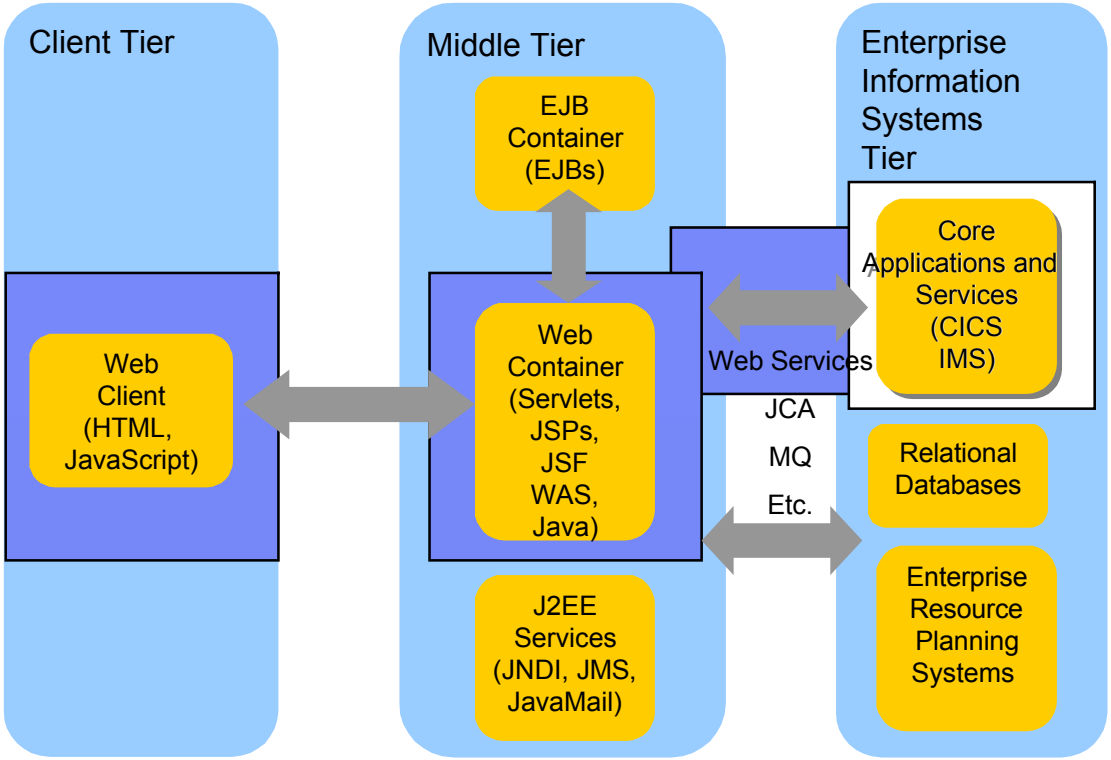


- **CICS as a service requester**



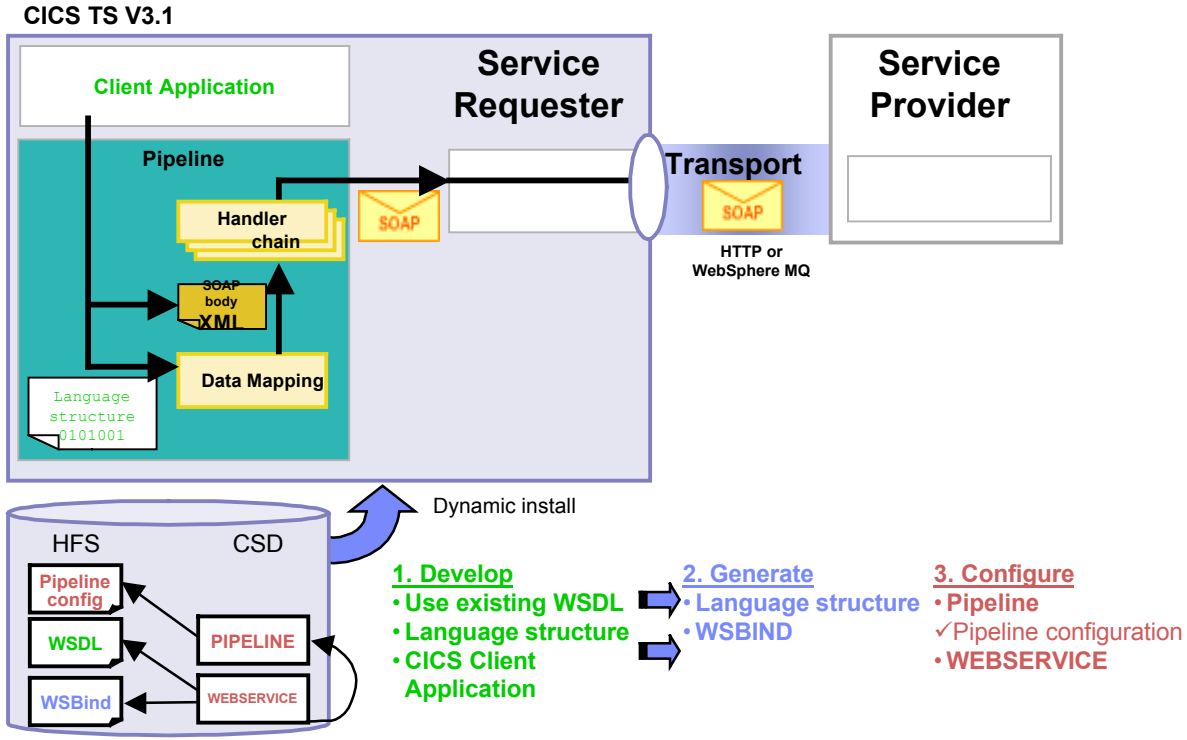
© 2007 IBM Corporation

The Business Tier



© 2007 IBM Corporation

CICS as a Web service requester



© 2007 IBM Corporation

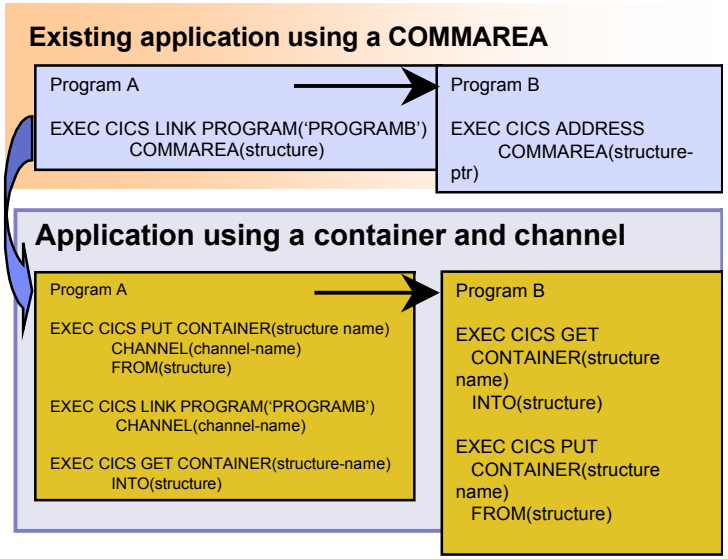
CICS API's

- **Invoking a Web Service from a CICS application program**
  - CICS as a service requester
    - EXEC CICS INVOKE WEBSERVICE ( ) CHANNEL ( ) URI ( ) OPERATION ( )
      - WEBSERVICE: name of the Web Service to be invoked
      - CHANNEL: name of the channel containing data to be passed to the Web Service (DFHWS-DATA container)
      - URI: Universal Resource Identifier of the Web Service (optional)
      - OPERATION: name of the operation to be invoked

© 2007 IBM Corporation

Data Exchange between CICS programs with Containers and Channels

- Offers a more flexible and intuitive alternative to the COMMAREA
- Enables large amounts of data to be passed between CICS applications
  - Not subject to 32KB restriction
- Optimized and managed by CICS
- Requires minimal application changes required to use

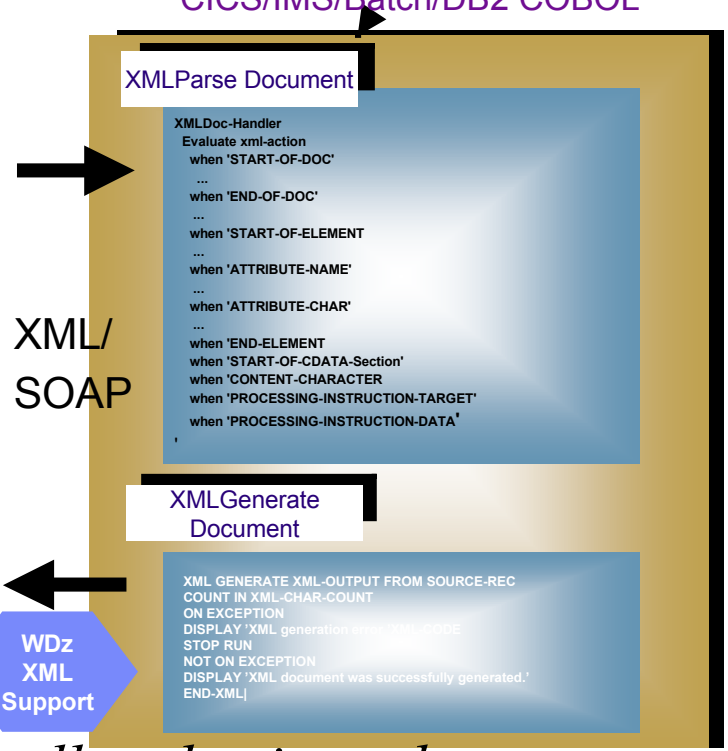


© 2007 IBM Corporation

IBM Enterprise COBOL

CICS/IMS/Batch/DB2 COBOL

- **XML Language based generation from COBOL data structure**
  - XMLGenerate Verb
  - WebSphere EJB support
  - DB2 V8
- **High speed XML Sax based parsing**
- **Object Oriented Support for Java COBOL Interoperability**
- **Unicode support**
- **CICS and DB2 integrated preprocessor**
- **Raise 16Mb COBOL data size limit**
  - Picture clause replication:
    01 A PIC X(134217727).
  - OCCURS::
    05 V PIC X OCCURS 134217727 TIME



COBOL is an excellent business language

© 2007 IBM Corporation

Why COBOL?

- **Large portfolios**
- **Many developers**
- **High performance**
- **Self documenting**
- **Proven Maintainability**
- **Business oriented, eases technology burden**

© 2007 IBM Corporation

Summary

- **MVC application model provides high levels of flexibility**
- **CICS provides leading edge support of Web Services**
  - Allows for re-use of existing business assets and new development of high QOS assets
- **Developers need "complete" application skills**
- **CICS and WebSphere Application Server are strategic middleware products that together…**
  - Interoperate - Web services, JCA, Enterprise JavaBeans
  - Exploit and complement z/OS qualities of service
  - Have high qualities of service, low cost per transaction, excellent security.

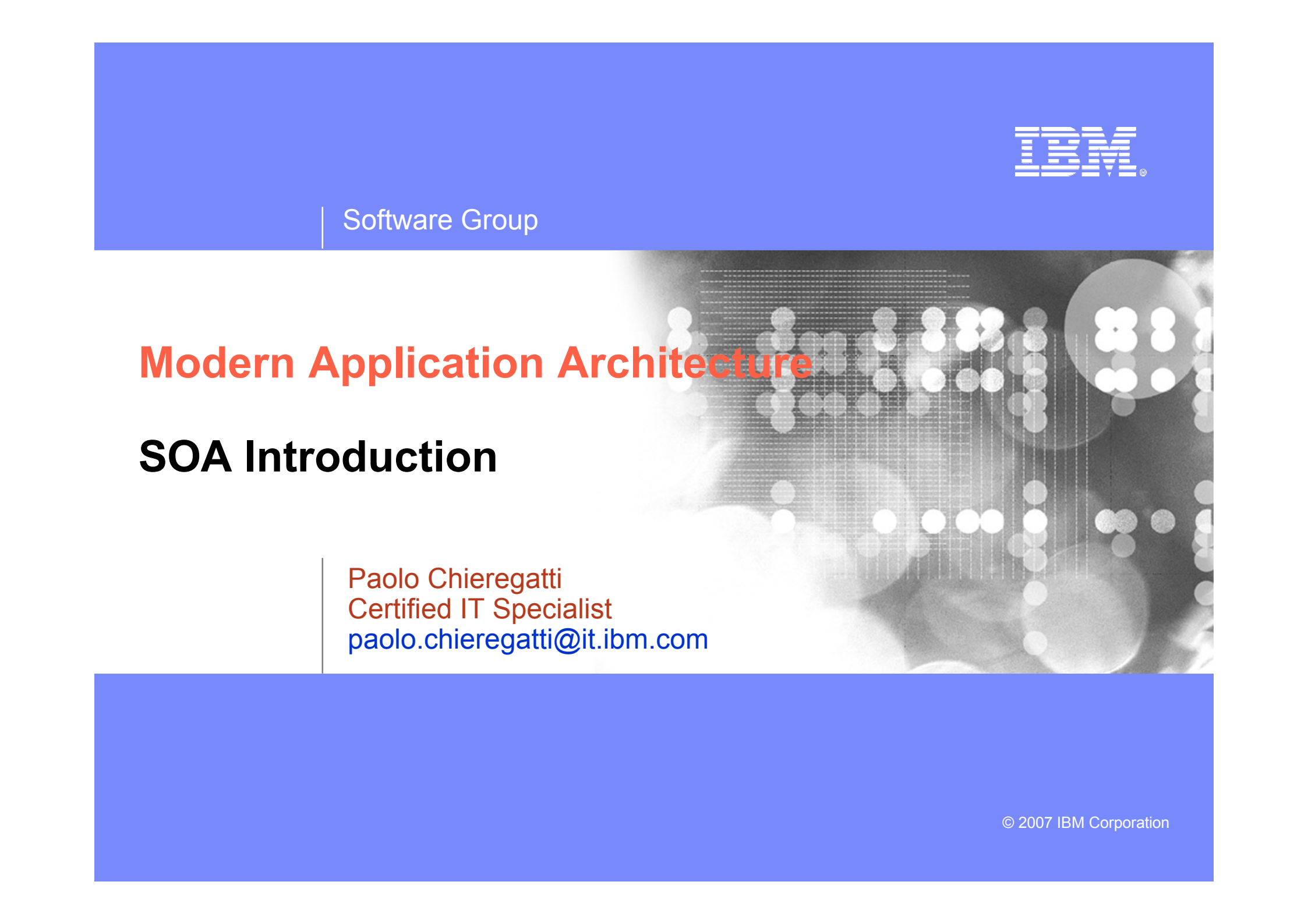
© 2007 IBM Corporation

Demo

Modern Application Architecture – Building and testing a JSF/COBOL process.

- **Demo of WDz used to create a simple, understandable visual and business application process for deployment.**
- **The session shows how to build and deploy composite CICS and WebSphere applications using the IBM WebSphere Studio tooling and the Enterprise Compilers. Composite applications are applications which are assembled from independent component parts, using Web and Web Services standards.**

© 2007 IBM Corporation


IBM
Software Group
Modern Application Architecture
SOA Introduction
Paolo Chieregatti
Certified IT Specialist
paolo.chieregatti@it.ibm.com
© 2007 IBM Corporation

Agenda

- **IT Market : trend & directions**
- **Service Oriented Architecture**
- **COBOL & Enterprise Application : Today**
- **SOA and System z Application Lifecycle**

© 2007 IBM Corporation

IT Market : trend & directions

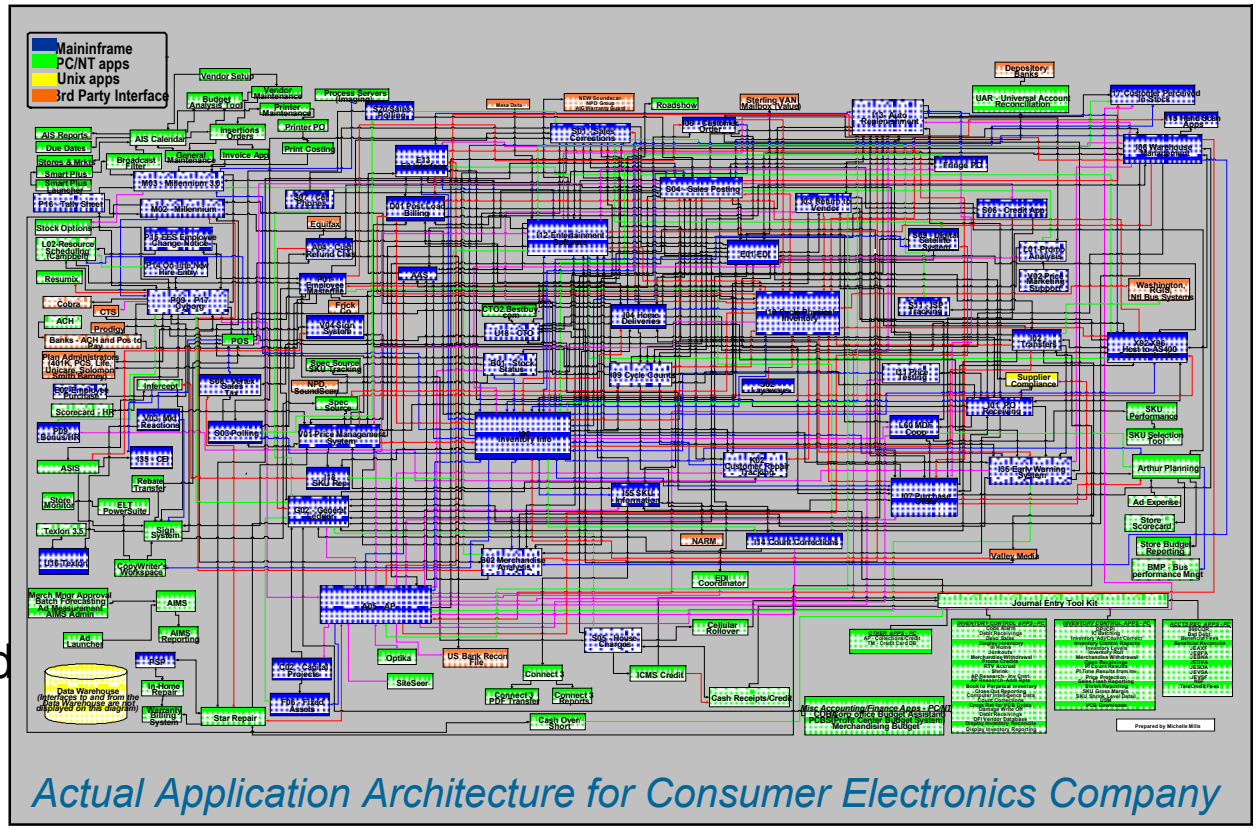
- **SOA : Service Oriented Architecture**
- **Virtualization & Consolidation**
- **Web 2.0 : WOA Web Oriented Architecture**

- **Second Life : ?**

© 2007 IBM Corporation

Architectural Challenges

- Application dependencies are extraordinarily complex, and exist at multiple levels
- Dependencies cross technologies and environments
- Need to support application maintenance, development and test
- Need to support application integration and service / component creation



Actual Application Architecture for Consumer Electronics Company

© 2007 IBM Corporation

What is Service Oriented Architecture (SOA)?

… a service?

A **repeatable business task** – e.g., check customer credit; open new account

… service orientation?

A way of integrating your **business as linked services** and the outcomes that they bring

… service oriented architecture (SOA)?

An IT **architectural style** that supports service orientation

… a composite application?

A set of **related & integrated** services that support a business process built on an SOA

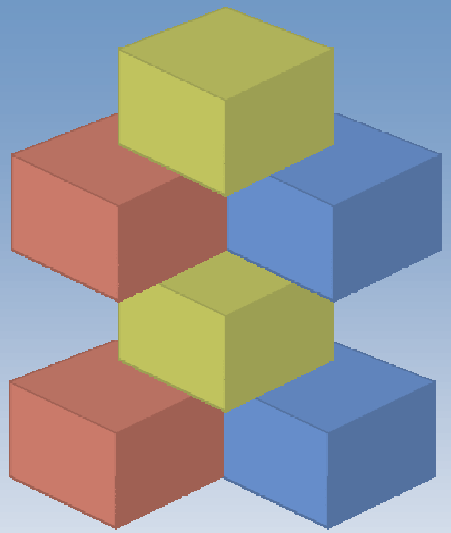
© 2007 IBM Corporation

SOA: The focus is on Flexibility and Reuse

Business Perspective

Modern UI's linked with Business Process

- Orchestrated sequence of
- Activities
- Separated elements
  - Activity sequence
  - Activity hand-off
  - Activity content



IT Perspective

Web User Interfaces and Composite Application

- Orchestrated flows of Services
  - Tooling
- Separated logic
  - Process flow
  - Connectivity
  - Business
- Flexible high QOS Business Functions

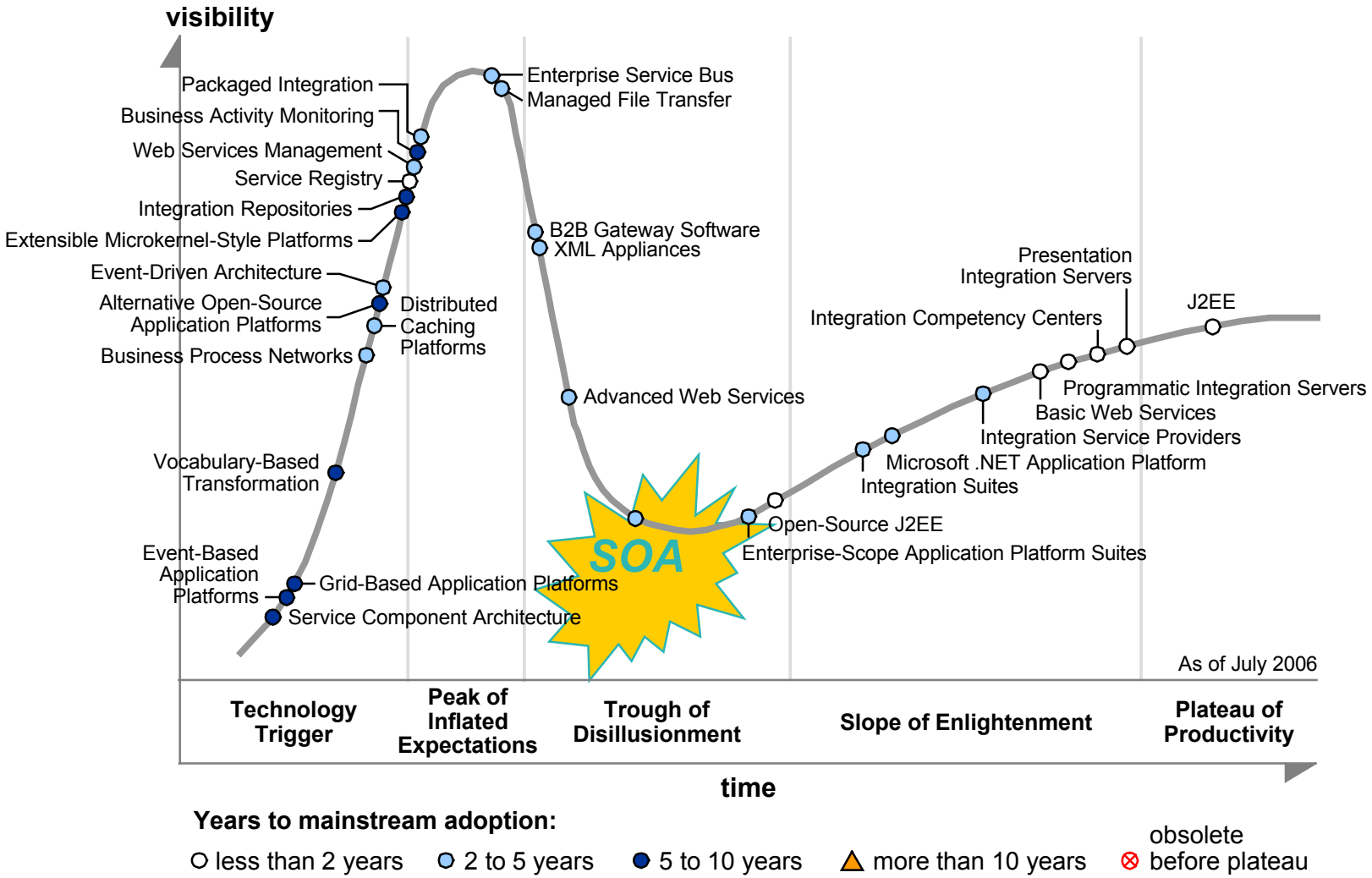
Why Service Oriented Architecture? …

- Enables re-use of existing assets
- Enhances system flexibility through logic isolation
- Supports simplified integration of new assets with existing assets

© 2007 IBM Corporation

SOA in the Trough of Disillusionment: Bad News or Good News?

Application Integration & Platform Middleware Hype Cycle



Source: Gartner

© 2007 IBM Corporation

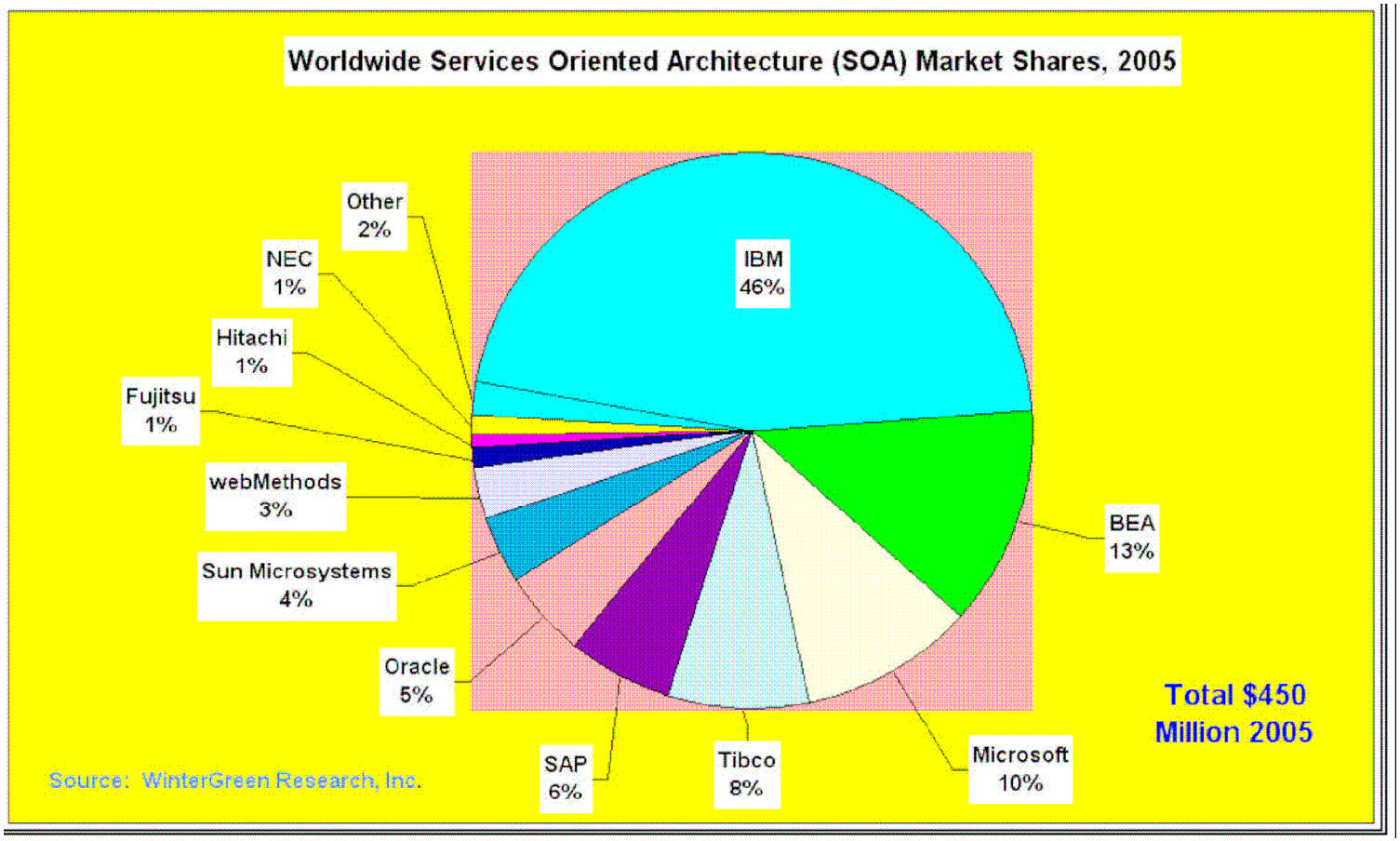
IT Market : Analysts

- **“Service oriented architecture (SOA) markets at $450 million in 2005 are expected to reach $18.4 billion by 2012. Market growth comes because SOA enables the flexible IT architecture that is needed to respond to market shifts brought by speeded product cycles and competitive challenges.”**

WinterGreen Research

© 2007 IBM Corporation

…. IT Market : Analysts



© 2007 IBM Corporation

Stages of SOA Adoption

	Stage 1 Introduction	Stage 2 Spreading	Stage 3 Exploitation	Stage 4 Plateau
Business Goals	Address Specific Pain (e.g., Customer Portal)	Process Integration (e.g., B2B)	Process Flexibility (e.g., Time to Market)	Continuous Adaptation & Evolution
IT Goals	Proof of Concept	Establish Technology Platform	Leverage Services Sharing	Enterprise SOA Infrastructure
Scope	Single Application	Multiple Applications (Single BU)	Multiple Applications (Cross BUs)	Virtual Enterprise
No. of Published Services*	<25	<100	<500	>500
No. of Service Consumers*	<5	<25	<50	>50
No. of Service Calls/Day*	<10,000	<100,000	<1,000,000	>1,000,000
No. of Service Developers*	<10	<20	<100	>100
Enabling Technology (cumulative)	Application Server, Portal, Adapters	ESB, WSM Integr. Suite, B2B	SOA Reg/Rep BPM Policy Mgmt.	Enterprise SOA Backplane

Source: Gartner

** =These figures represent typical scenarios, but they may vary considerably depending on the specific organization's requirements.*

© 2007 IBM Corporation

Business Flexibility Is A Top Priority

Business Flexibility the ability to be more responsive to changing market conditions, including opportunities, customers and competitive actions.

Business Flexibility is best achieved with SOA.

Flexible business requires a flexible IT… SOA is an evolutionary approach to building flexible IT systems focused on solving business problems

Gartner

"Business flexibility is a key enabler of business innovation."
David Cearley, Gartner

zapthink

"The power of business flexibility is the power of business to take advantage of new and unexpected opportunities, and this is what makes businesses competitive."
Ron Schmelzer, ZapThink

HURWITZ & ASSOCIATES

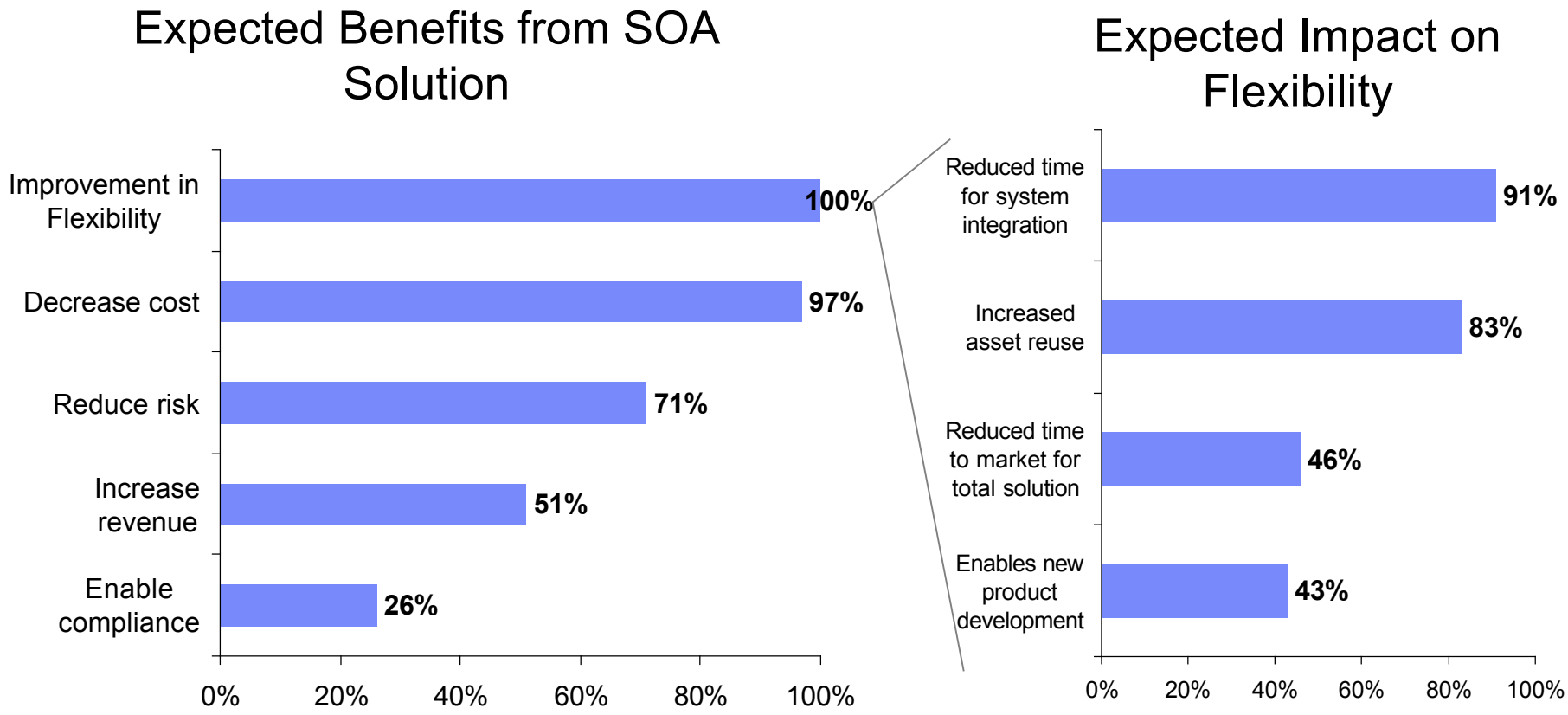
"The only way customers gain competitive advantage is through business flexibility. Without flexibility at the business and technical level, innovation isn't achievable."
Judith Hurwitz, Hurwitz & Associates

Amy Wohl
Candid opinions about the technology industry

"When we can respond to rapid business changes with the flexibility to optimize our business opportunities, we will have moved to the next stage of IT."
Amy Wohl, Wohl Associates

© 2007 IBM Corporation

SOA projects improved business flexibility 100% of the times



© 2007 IBM Corporation

Agenda

- **IT Market : trend & directions**
- **Service Oriented Architecture**
- **COBOL & Enterprise Application : Today**
- **SOA and the System z Application Lifecycle**

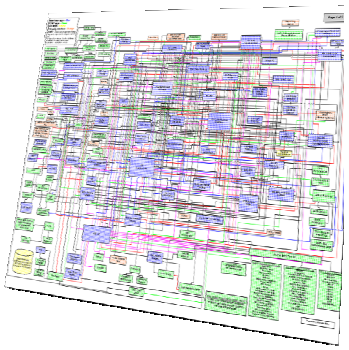
© 2007 IBM Corporation

SOA & Application Development

- **Reuse enterprise application**
- **Componentization**
- **Modernization**
- **Adoption of open standard**
- **New SOA application**

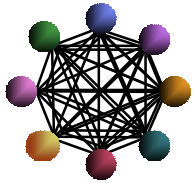
© 2007 IBM Corporation

SOA
the next step



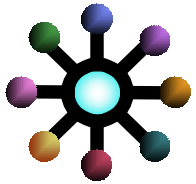
Direct Connectivity

Point-to-Point connection between applications

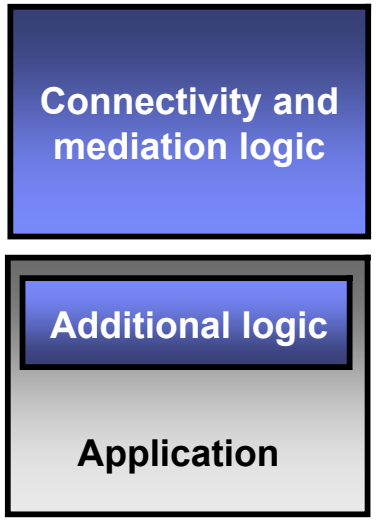


Message Queuing

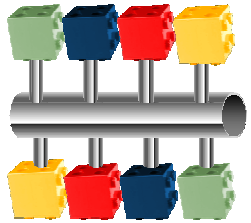
Applications via a centralized hub



Message Brokering



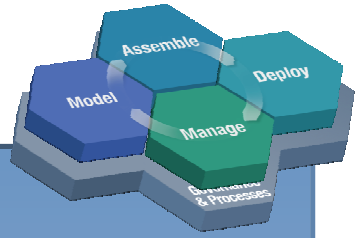
Integration and choreography of services through an Enterprise Service Bus



Service Orientation

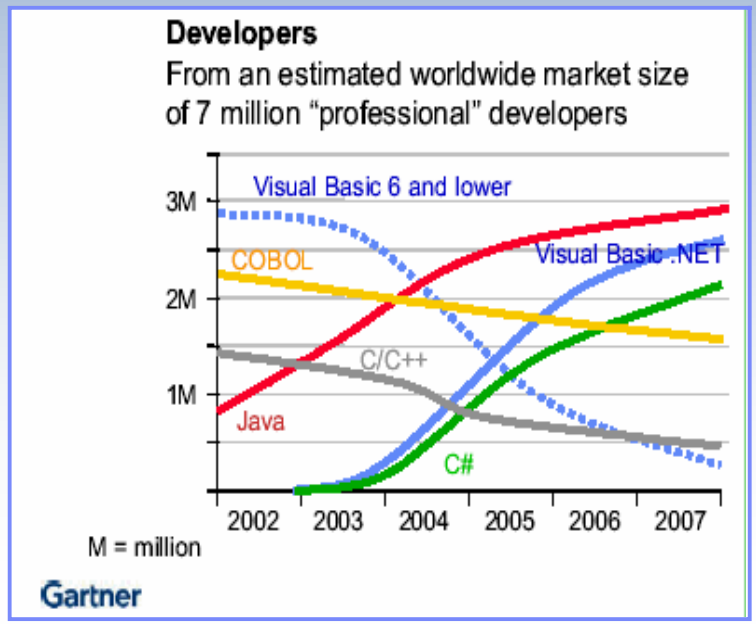
© 2007 IBM Corporation

What about “before SOA”?



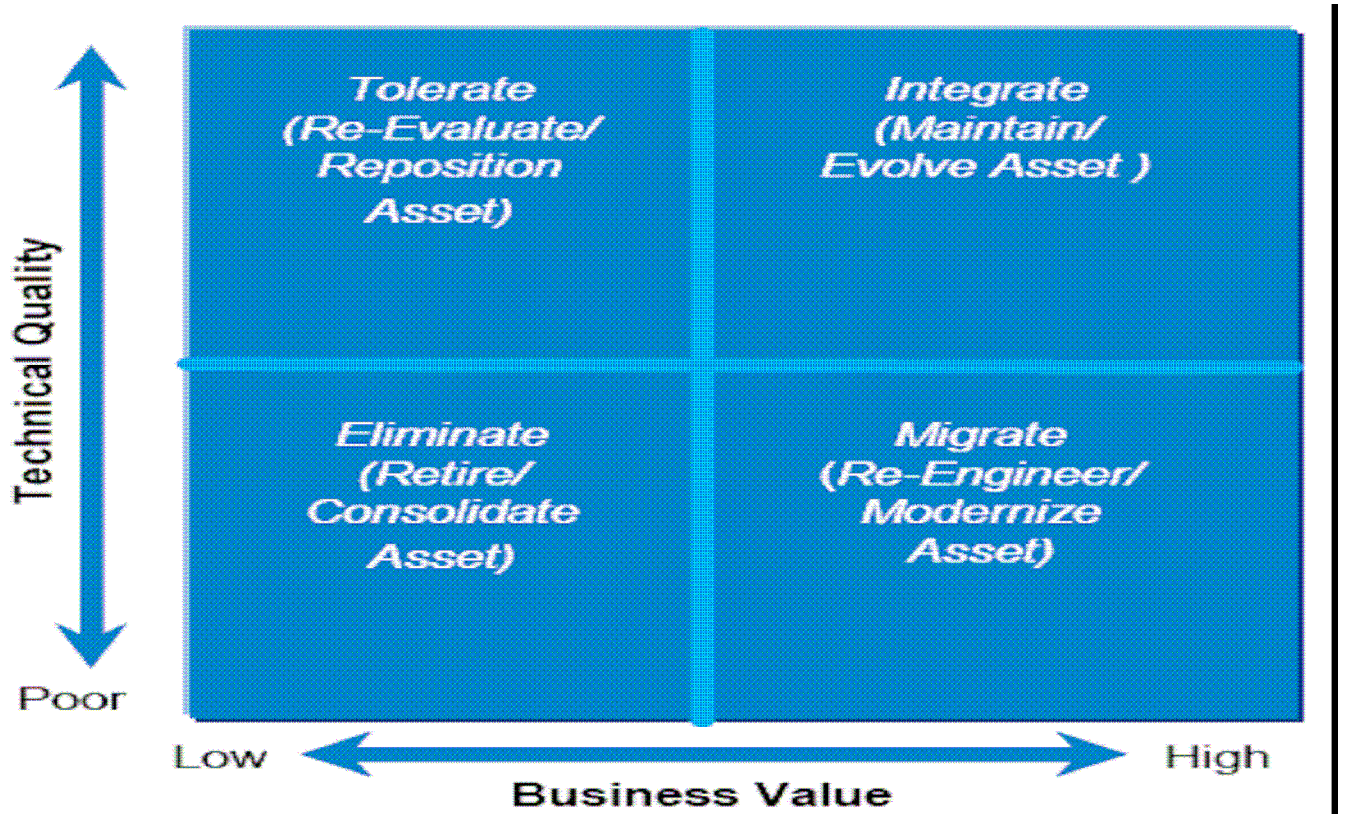
- **Significant business intelligence exists in core systems**
  - "200 Billion lines of COBOL code in existence" ***eWeek***
  - "5 Billion lines of COBOL code added yearly“ ***Bill Ulrich, TSG Inc.***
  - “2 Million COBOL developers" ***Gartner***
  - "Majority of customer data still on mainframes“ ***Computerworld***
  - “Replacement costs $20 Trillion” ***eWeek***

- **Rewriting - is it an option.....**
  - How long will it take? (lose strategic benefit)
  - Who will do it? (who has the business knowledge?)
  - How much will it cost?
  - Risk?



© 2007 IBM Corporation

Application Portfolio Analysis



© 2007 IBM Corporation

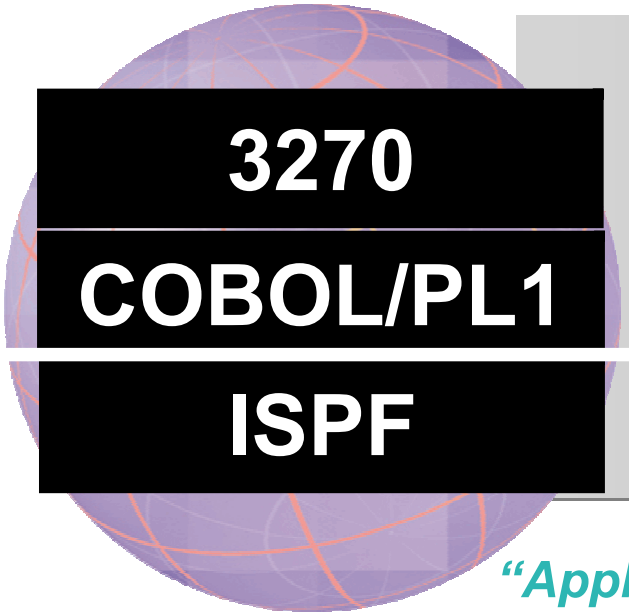
COBOL Today and the future

- **COBOL (COmmon Business Oriented Language)**
  - The predominant programming language of business applications for over 40 years
  - Specifically designed for business applications
    - Two million programmers write up to 5 Billion lines of COBOL code every year.

- **COBOL : main key points**
  - Strong presence of COBOL vendors
  - IBM continues to deliver value in its COBOL compiler products.
  - COBOL is easy to learn and maintain over time, with or without formal training.
  - The mainframe delivers superior operational efficiency due to its centralized design.

© 2007 IBM Corporation

Investment Challenges



- **Many zSeries developers still:**
  - Focused on creating or enhancing 3270 applications
  - Using traditional, host-based development environment

"Application maintenance consumes between 60 – 80 percent of IT budgets" - Phil Murphy, Forrester

Issues: How do I?
- Increase productivity of business developers working on traditional applications
- Enabling broad business developer community in SOA and Web Based infrastructures
- Improve Time to market and IT responsiveness

© 2007 IBM Corporation

Technology Challenges

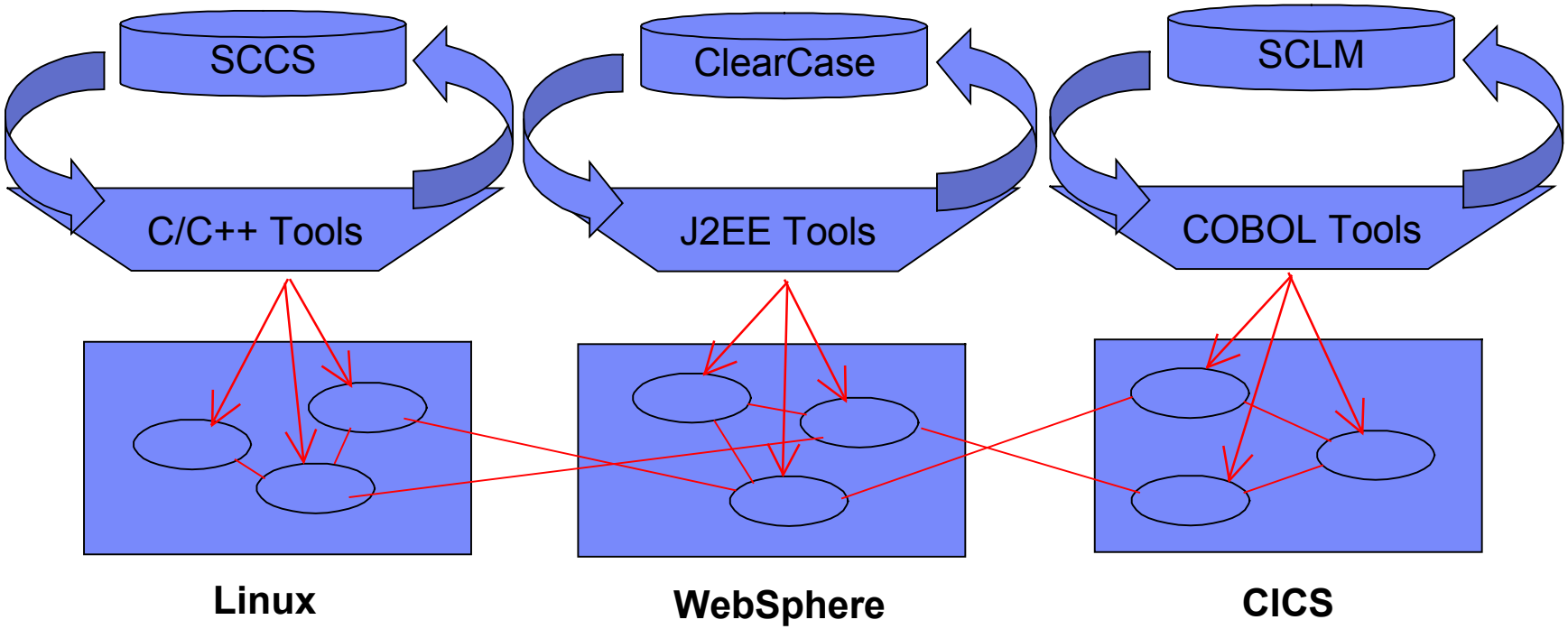
Issues: How do I?

- Enable experts on Core Applications in modern technologies
- Leverage business skills
- Create the SOA infrastructure without throwing everything else away

© 2007 IBM Corporation

Organizational Challenges

- Lack application components & skills sharing
- Ineffective / Uncoordinated development of integrated application

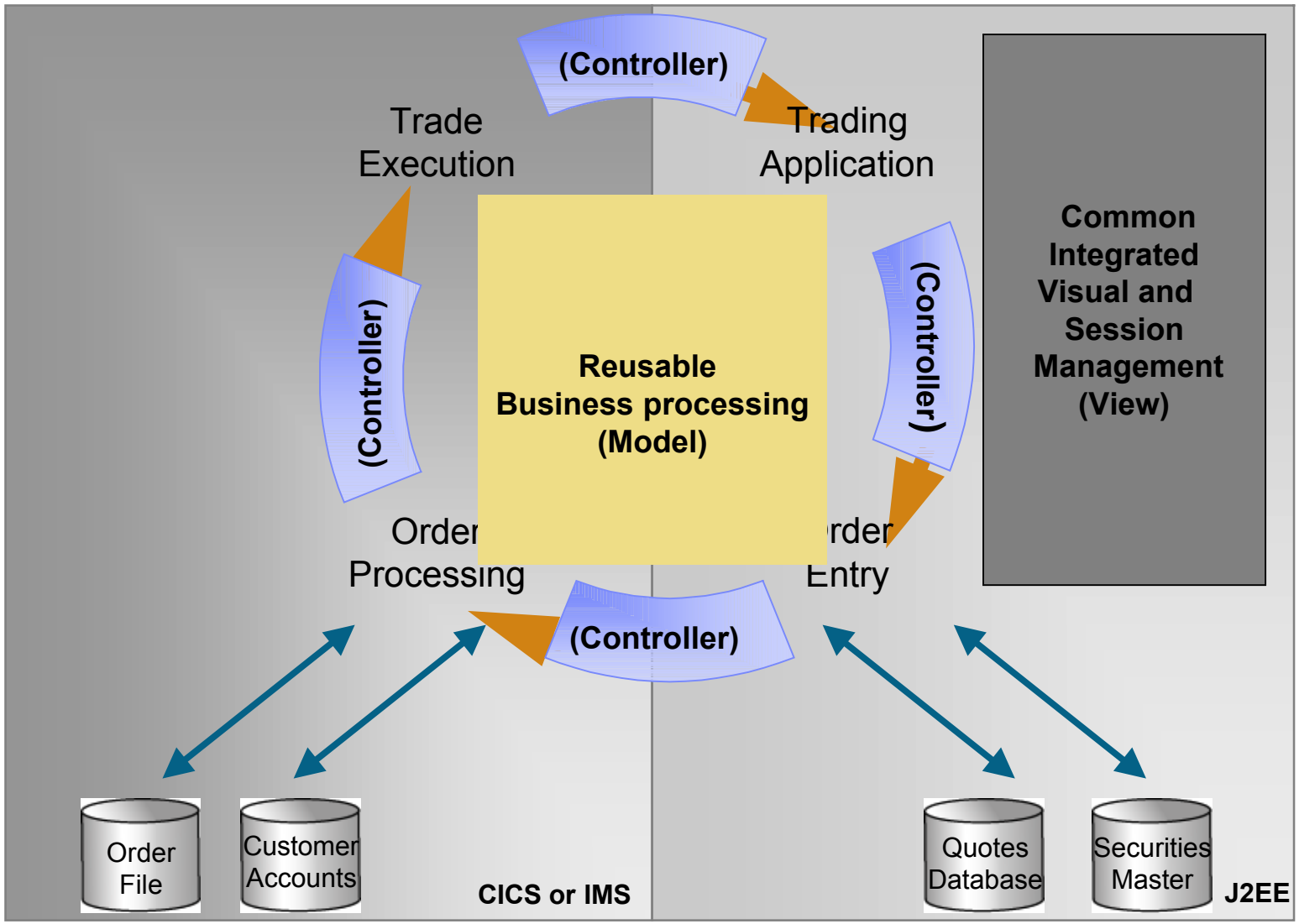


Issues: How do I?

- Manage change across geographically distributed development teams
- Communicate available services and resources
- Leverage existing code – and process – at the same time improving quality

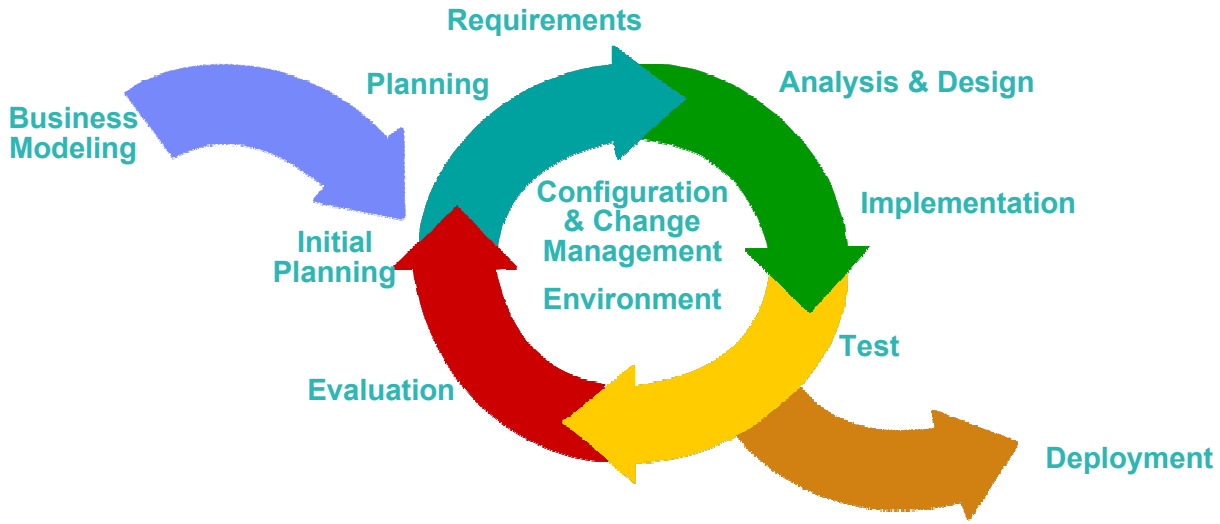
© 2007 IBM Corporation

Composite Workload Application Components



© 2007 IBM Corporation

Strategy 1 - Bring iterative model driven development paradigms to composite applications



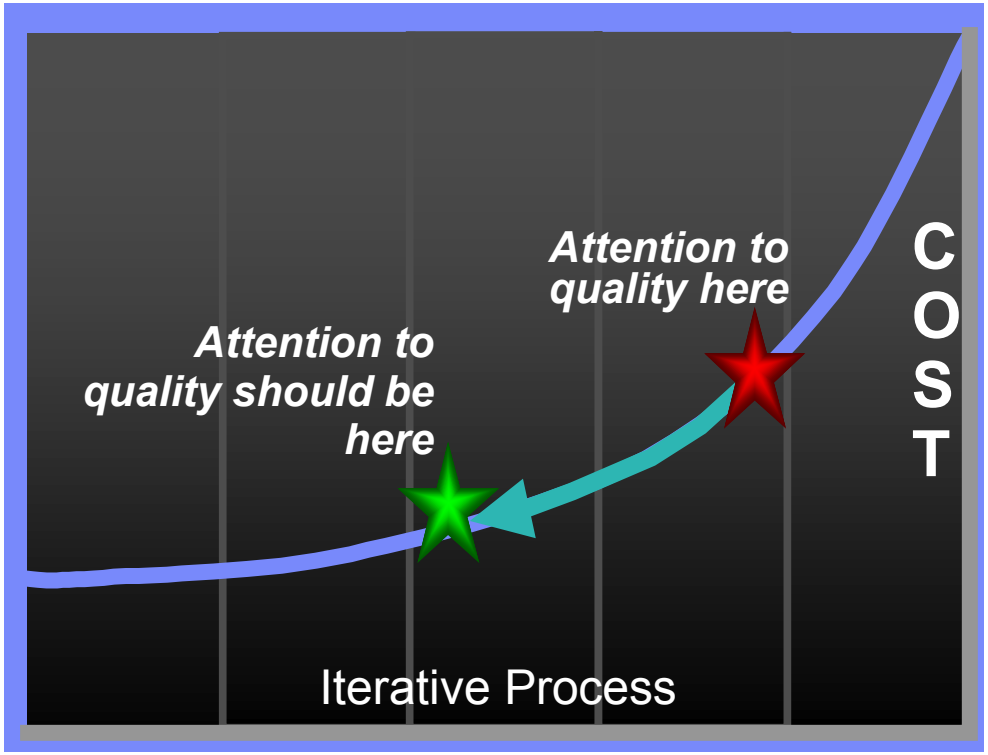
- Adopt a flexible process for both J2EE & traditional z/Series applications
- Tools integration across the lifecycle (Model and Discover, Develop and Assemble, & Deploy and Manage)
- Manage mixed workload requirements

Issues: How do I?
- Leverage modern development techniques across broad developer organizations
- Generate complex SOA architectures, versus hand coding
- Improve documentation and speed the development to test cycle

© 2007 IBM Corporation

Strategy 2 -Prevent, detect, diagnose and remove defects

- Improve application quality and test process
- Provide early warnings of activities susceptible to failure
- Analyze across disciplines to understand root causes

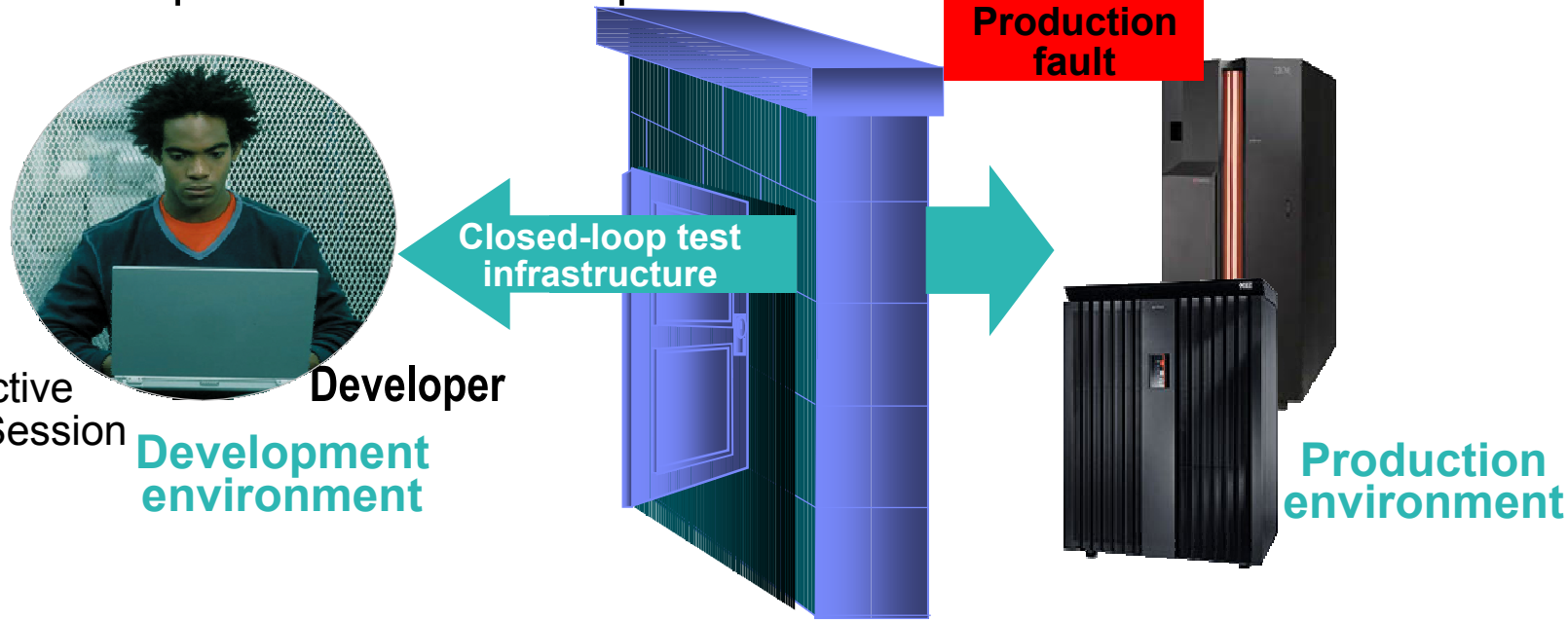


Issues: How do I?
- Find problems in development, before system test and production
- Debug SOA applications cross programs, platforms, languages, etc.
- Perform risk analysis on quality of deliverables

© 2007 IBM Corporation

Strategy 3 - Reduce application downtime

- Find and fix errors post-deployment quickly
- Speed application rebuild and redeploy
- Bridge development teams and operation teams



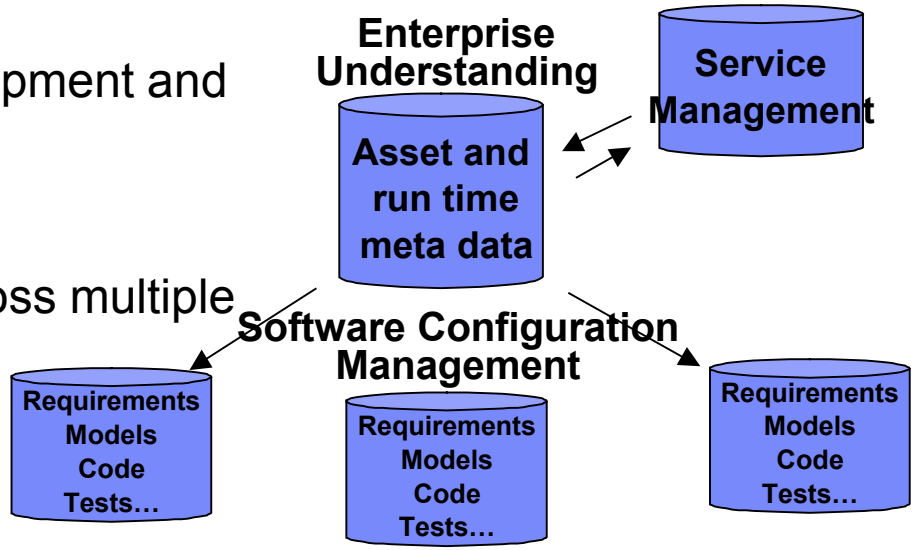
Issues: How do I?

- Manage quality in a SOA environment
- Solve applciation faults when multiple runtimes are involved
- Leverage business knowledge during problem determination process – i.e., common skills across developer bases

© 2007 IBM Corporation

Strategy 4 - Manage change and assets as services

- Manage change across multiple development and operational environments
- Manage diverse assets
- Automate and accelerate workflow across multiple development teams



Business Benefits

- Quickly respond to change
- Develop anytime, anywhere, in parallel
- Enable reuse and protect assets

Technology Benefits

- Flexible workflow and process support
- Distributed team management
- Traceability across the lifecycle

Issues: How do I?

- Govern processes and enable reuse
- Track who is working on what
- Merge changes from multiple teams
- Support vastly increased numbers of artifacts across the lifecycle

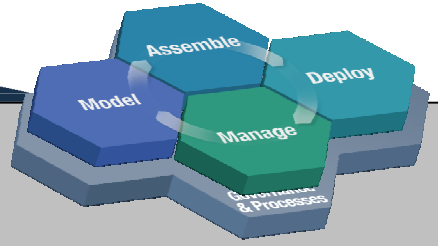
© 2007 IBM Corporation

Agenda

- **IT Market : trend & directions**
- **Service Oriented Architecture**
- **COBOL & Enterprise Application : Today**
- **SOA and the System z Application Lifecycle**

© 2007 IBM Corporation

System z Application Lifecycle



Rational Software Architect

WebSphere Business Modeler

WSAA / ATW / CICS IA

Monitor and manage Business processes

Common Processes and Software Configuration Management

WebSphere Developer for zSeries / HATS

Tivoli WS Business Monitor

Monitor Business

Model Business

Model Applications

Discover / Understand

Develop

System z Application Lifecycle

Monitor Applications

Test

Debug/ Deploy

Manage Data

Assemble

Fault Analyzer ITCAM Omegamon Application Performance Analyzer

RPT/RFT

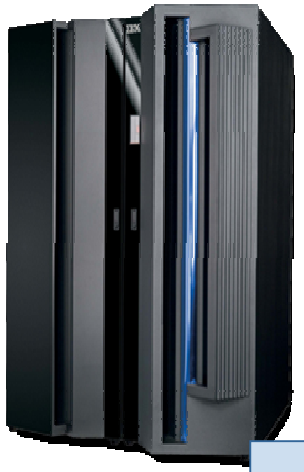
Debug Tool Utilities

File Manager

WebSphere Integration Developer

© 2007 IBM Corporation

IBM System z Technology
An Evolution ……..



From Datacenter

1964 S/360

1968 IMS

1969 CICS

1979 MVS

1983 DB2

1986 NetView

1994 MQ Series

2000 z/OS

2002 IBM TCO Tools

2003 WAS V5.0 for z/OS

2004 DB2 V8 IMS V9 Candle

2005 CICS V3 MQ V6 z/OS V1.7

2006 WAS V6 ESB Process Server Portal DB2 V9

To SOA Server

System z Software
Evolution, not revolution
Modernizing applications

© 2007 IBM Corporation

System z
Openness and Standards

Linux

UNIX ***SOA***

SAN ***Java***

Web Services ***J2EE***

Grid & Autonomic Sys. Mgmt

© 2007 IBM Corporation

Enabling a robust, flexible SOA runtime environment

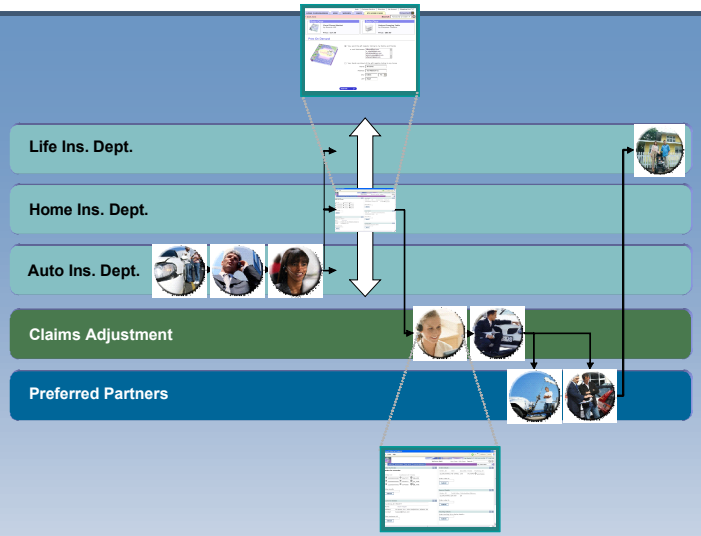
While maximizing the value of existing assets

Fully SOA capable!

WebSphere Application Server V6

CICS Transaction Server V3.1

IMS Transaction and Database V9



#1 in market share for Application Server software

IBM WebSphere Application Server comes out on top

35+ years of maturity and innovation in transaction and data systems

© 2007 IBM Corporation

Tools to realize Enterprise Modernization

WebSphere Studio Asset Analyzer (WSAA)

- Application Understanding
- Impact Analysis

Enterprise-wide app discovery and insight; find dependencies across applications and lines of business

Architects, project leaders, DBAs, developers, system programmers

Asset Transformation Workbench (ATW)

- Application Analysis
- Business Rule Management
- Components for reuse

Project-level workbench for deep application analysis and transformation

Architects, project leaders, analysts, developers

WebSphere Developer for zSeries (WDz)

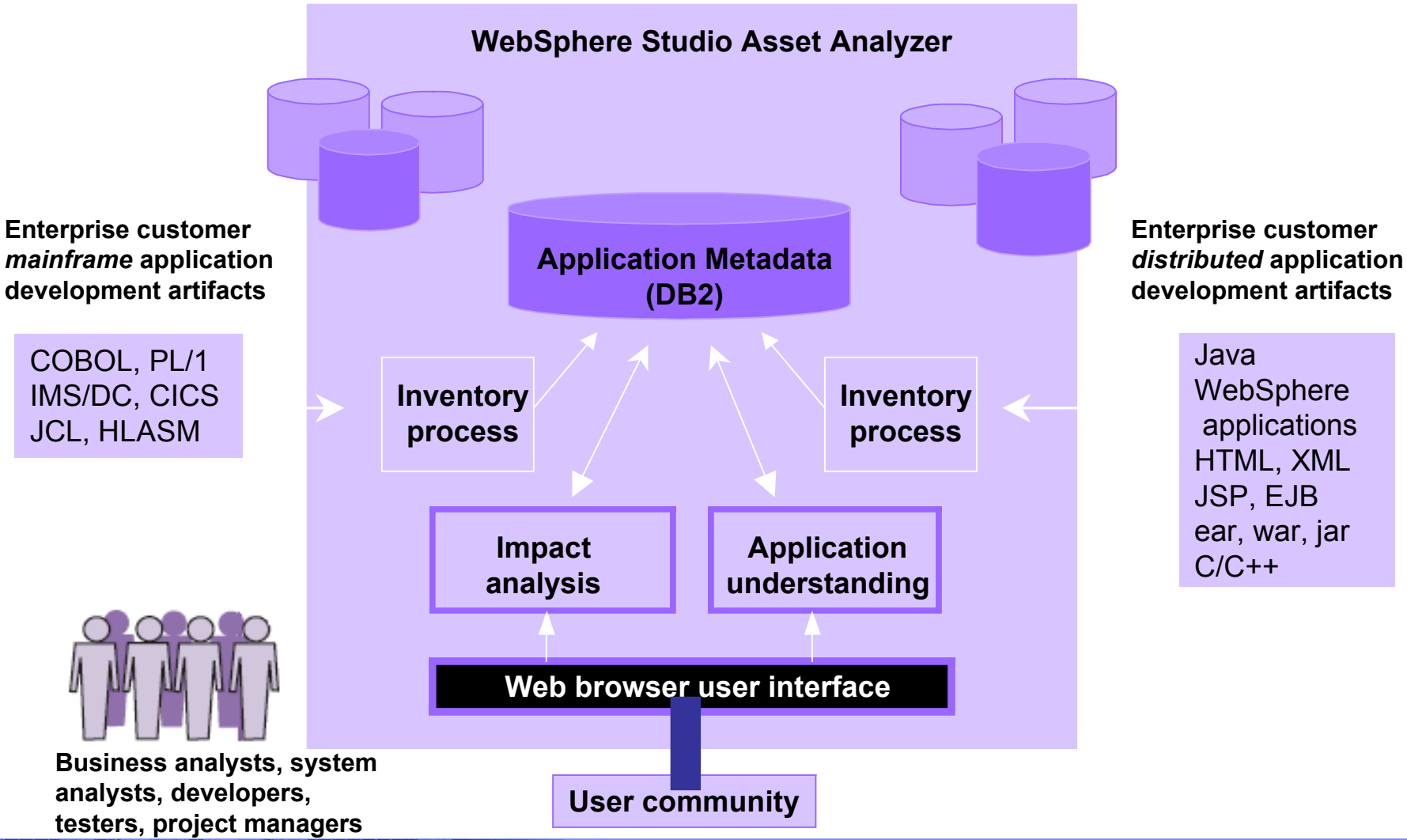
- Traditional Development
- Web Development
- Services Development

Common IDE for COBOL, PL/I, J2EE and Web services development

Architects, Developers

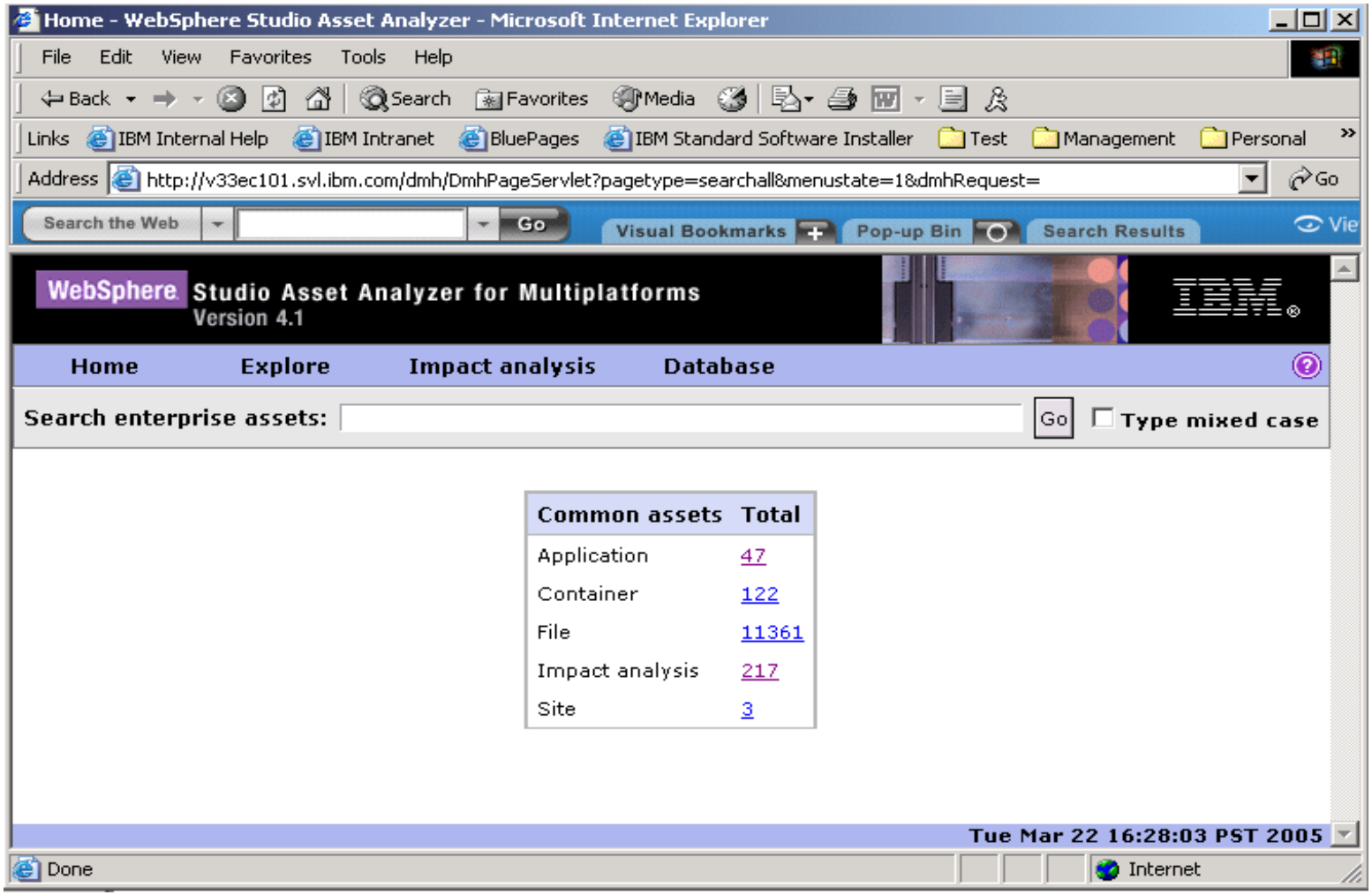
© 2007 IBM Corporation

WebSphere Studio Asset Analyzer



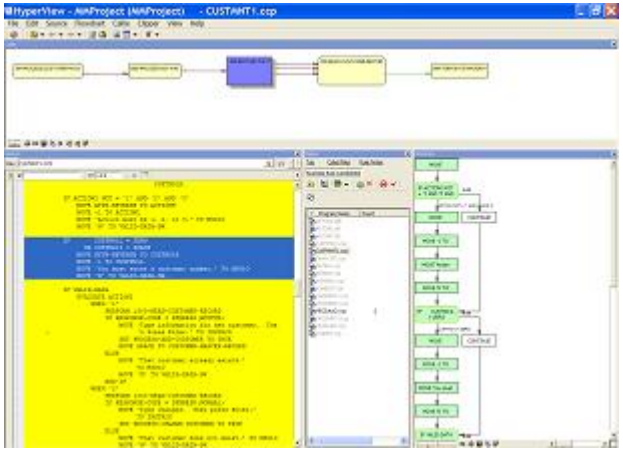
© 2007 IBM Corporation

Searching for Application Assets

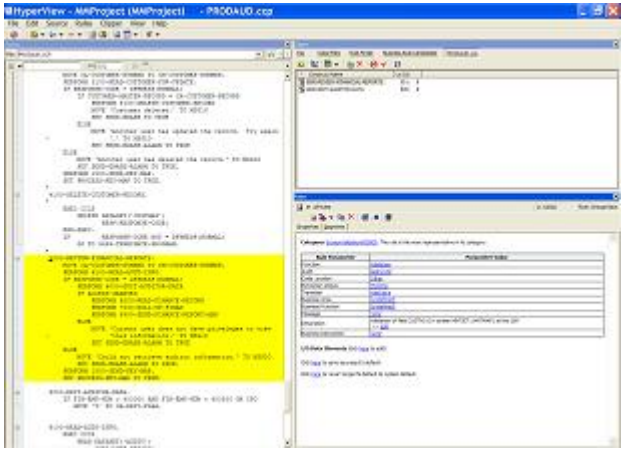


© 2007 IBM Corporation

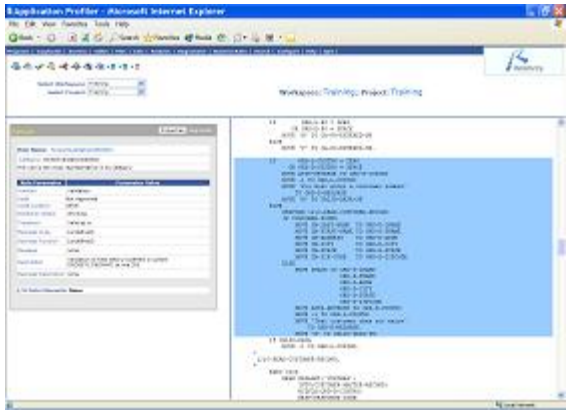
Asset Transformation Workbench



Business rule discovery



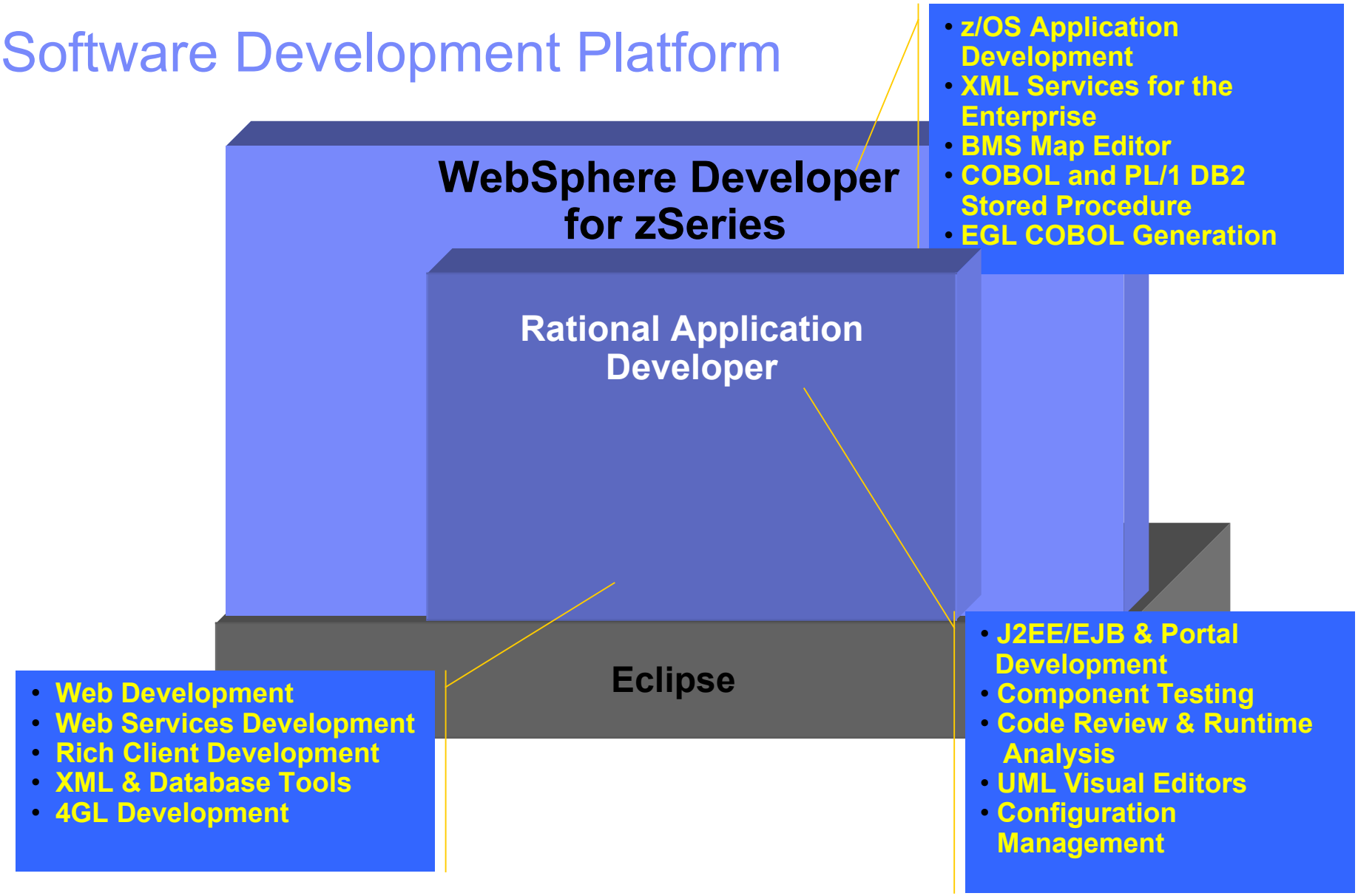
Business rule management



Integration with Application Profiler

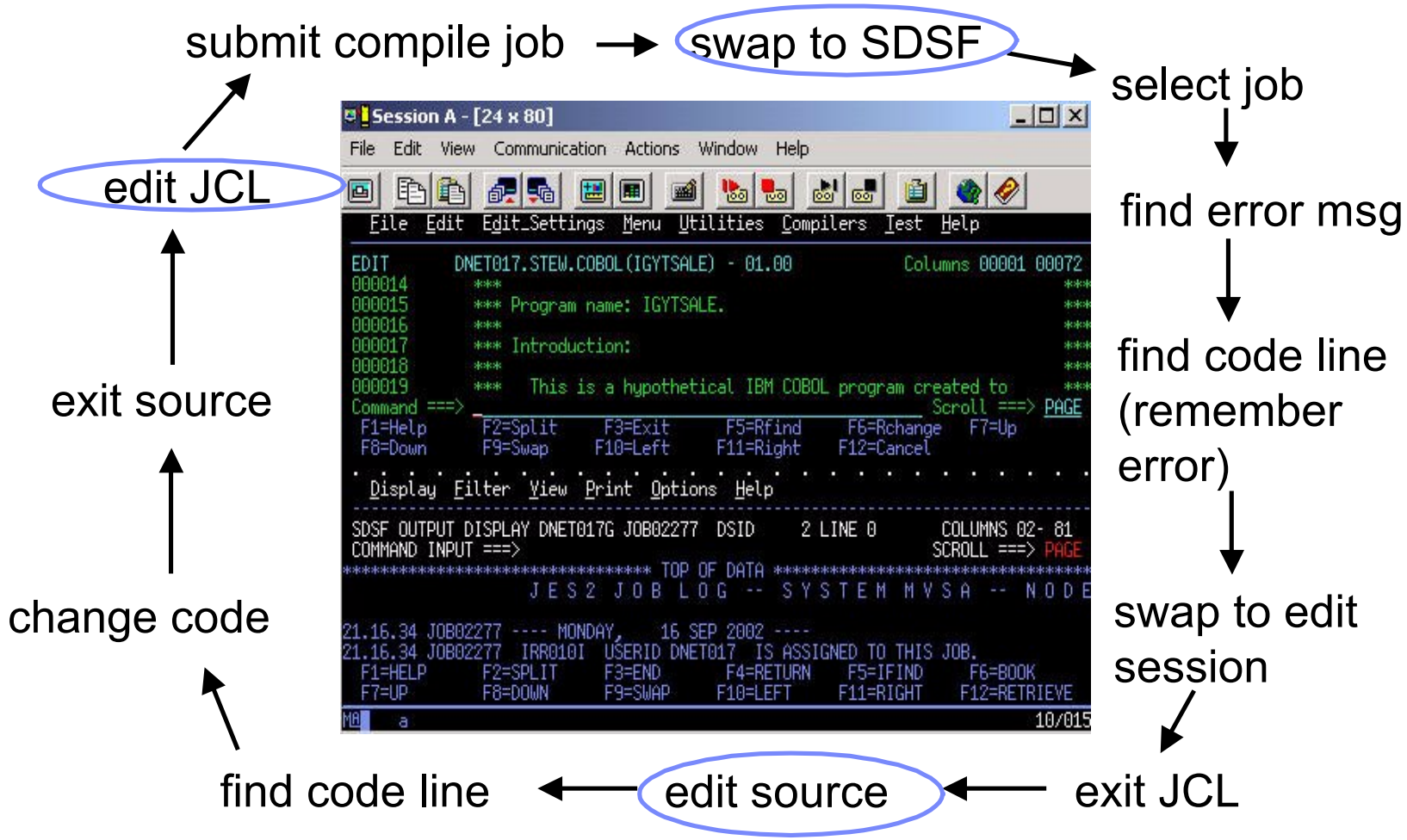
© 2007 IBM Corporation

Software Development Platform



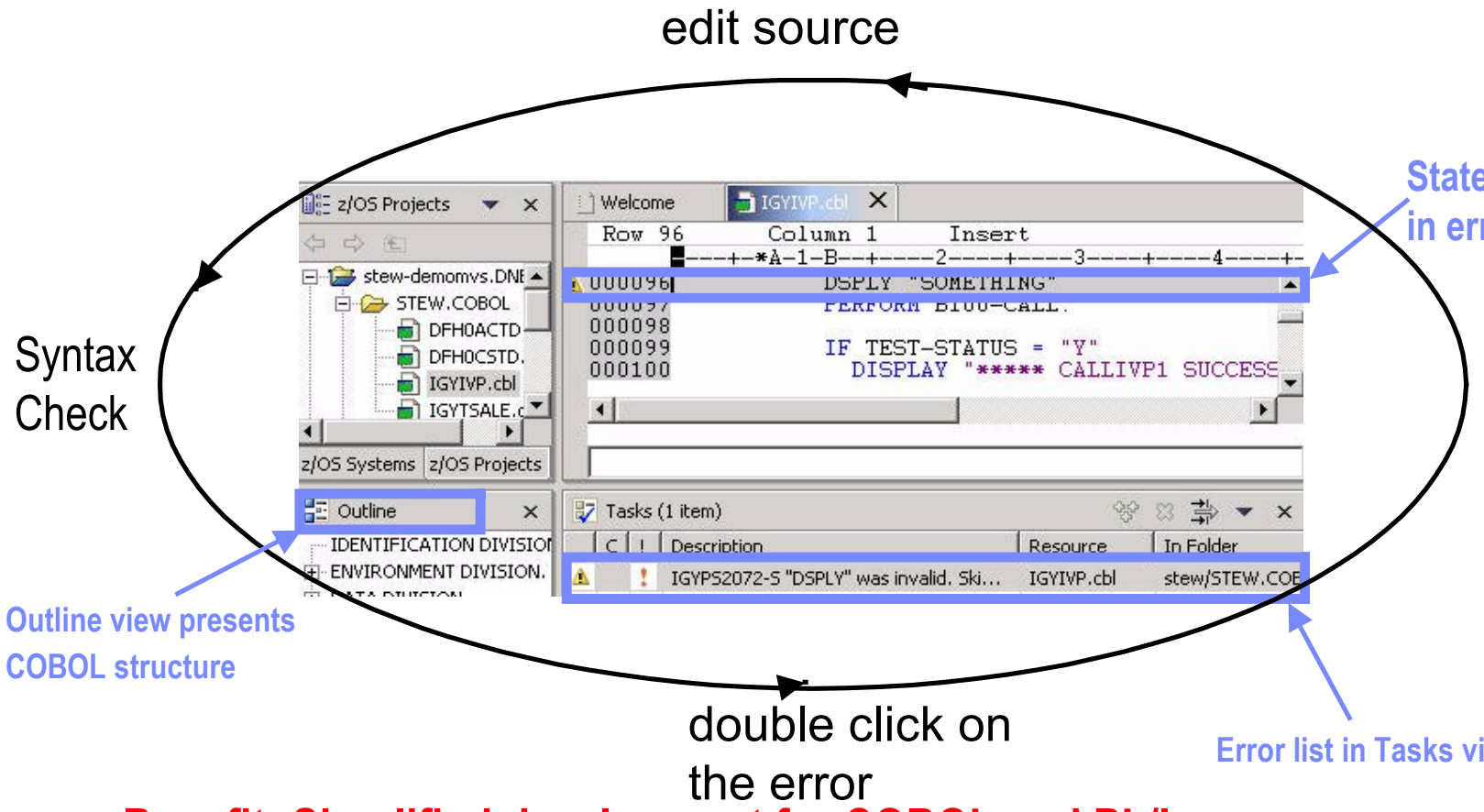
© 2007 IBM Corporation

ISPF based Development



© 2007 IBM Corporation

WebSphere Development



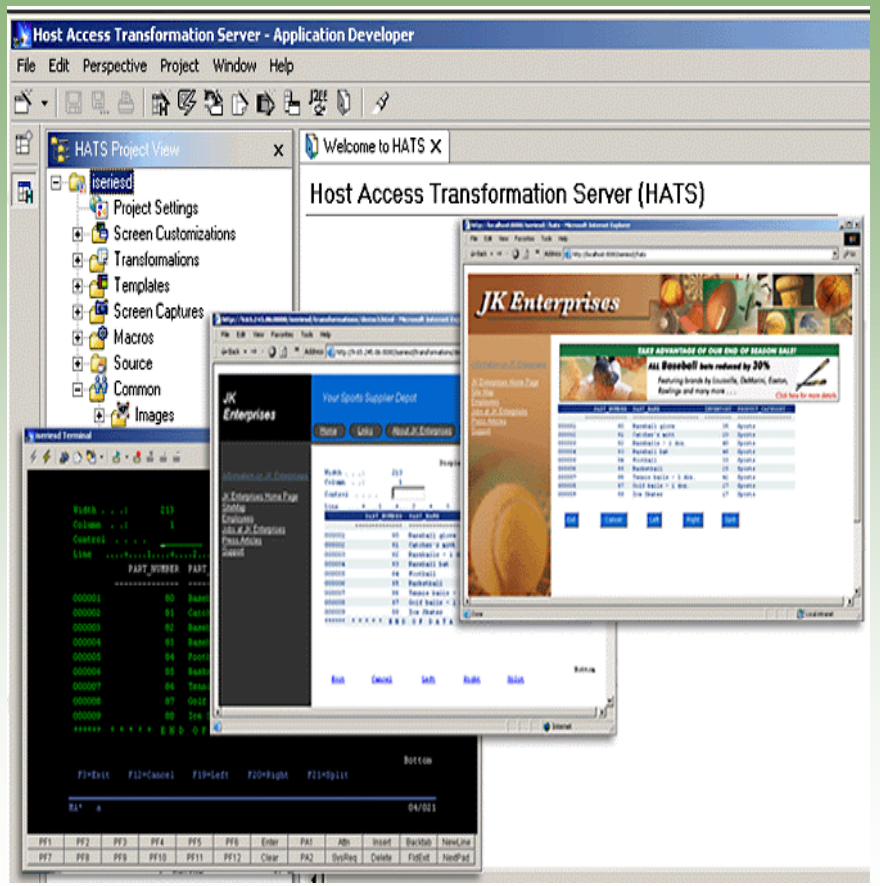
Benefit: Simplified development for COBOL and PL/I on a common development environment

© 2007 IBM Corporation

WebSphere Host Access Transformation Server

Extend business processing through existing interfaces

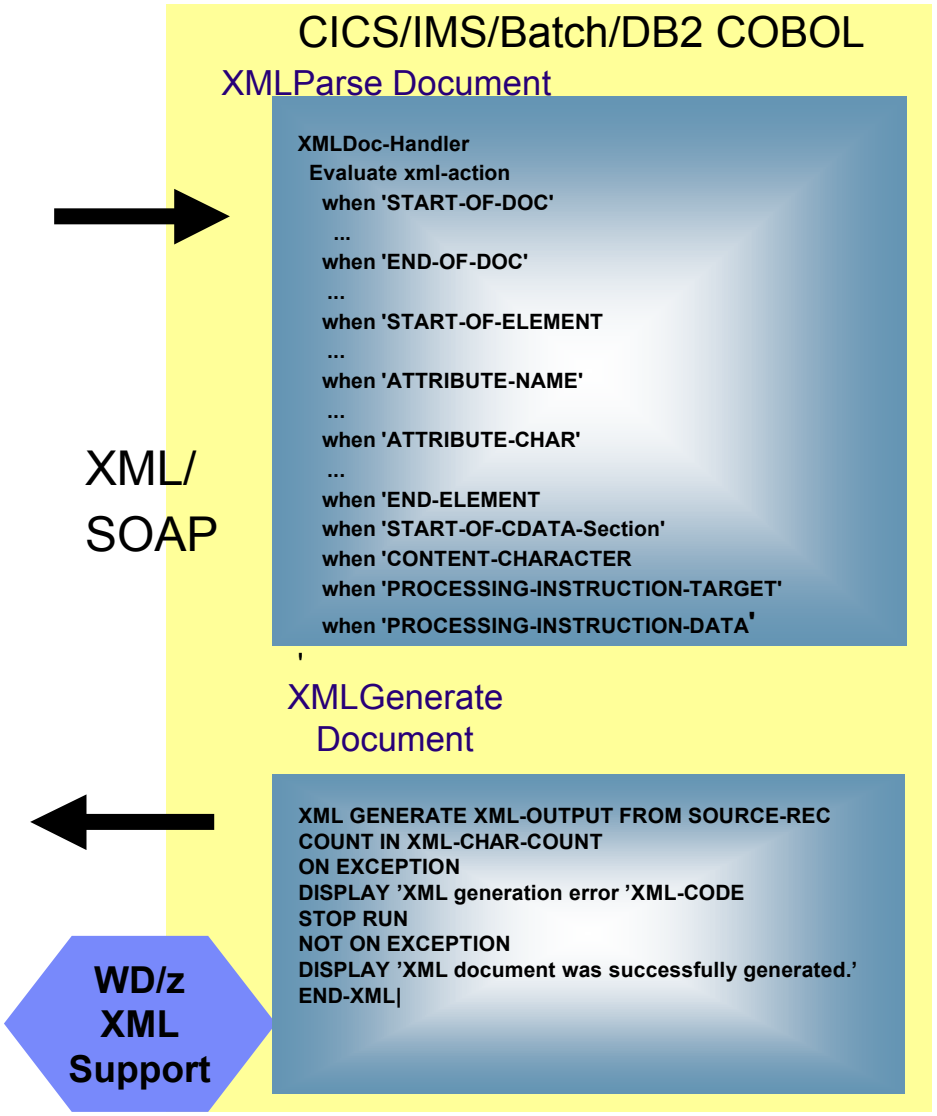
- Automatically transforms 3270 & 5250 green screen applications into HTML interfaces
- Extends terminal applications as Web Services
- Low skills requirement – no zSeries skills required
- Rules-based, highly customizable
- Iterative, eclipse-based development environment



© 2007 IBM Corporation

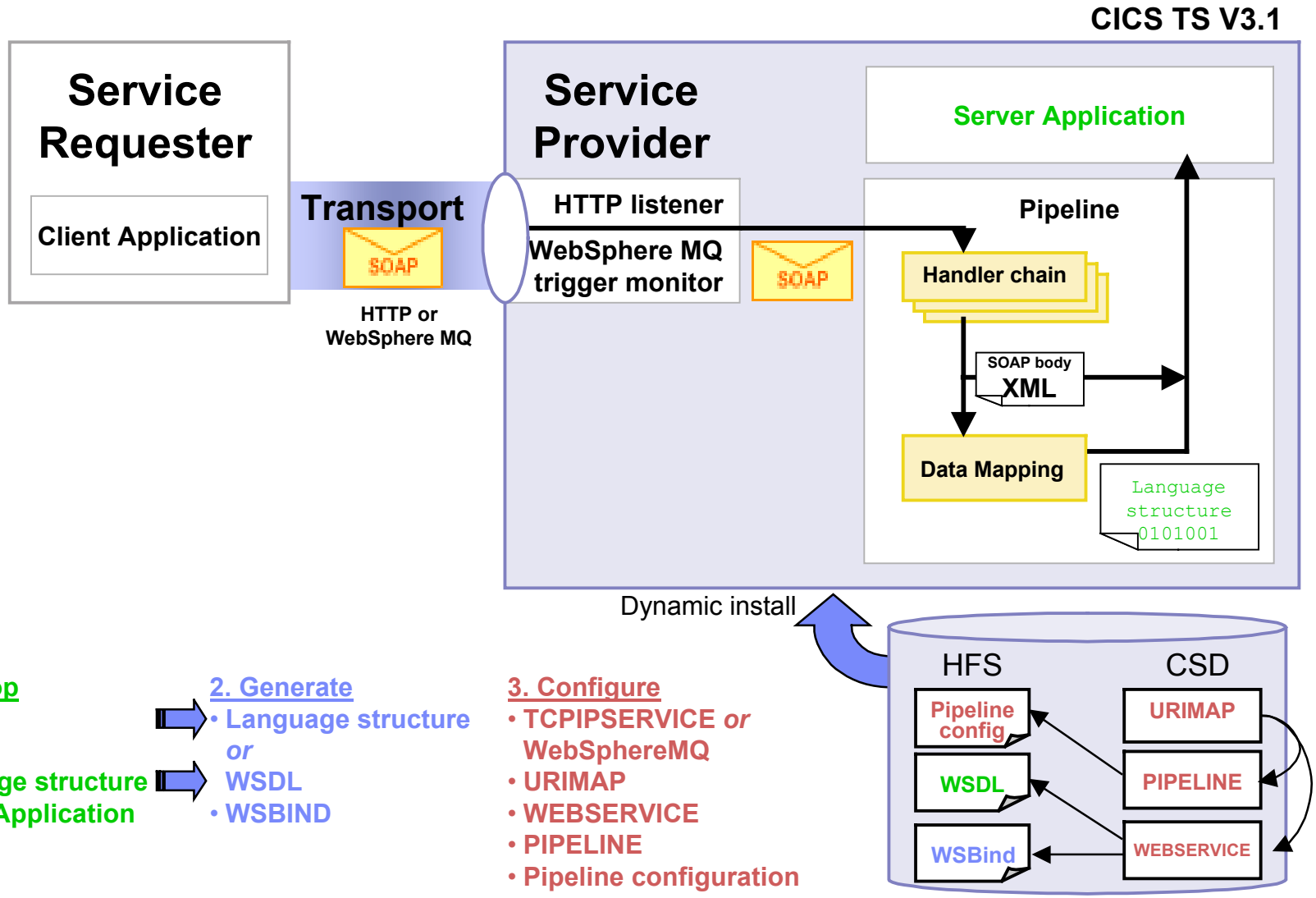
Using Enterprise COBOL to service-enable z/OS

- What's the latest…
  - XML Language based generation from COBOL data structure
  - WebSphere EJB support
  - DB2 V8 preprocessor
  - CICS preprocessor
- High speed XML Sax based parsing
- Object Oriented Support for Java COBOL Interoperability
- Unicode support
- Similar XML parsing support available in Enterprise PL/I
- CICS and DB2 integrated preprocessor
- Raise 16Mb COBOL data size limit



© 2007 IBM Corporation

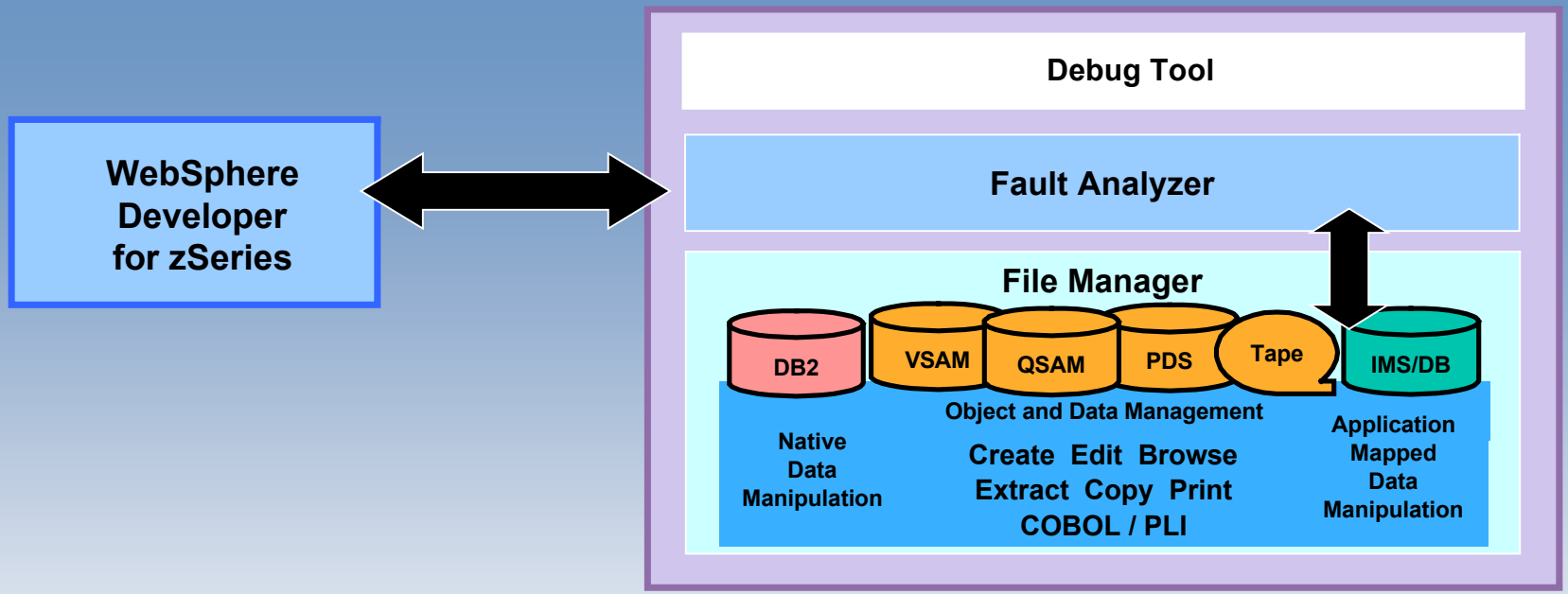
CICS come Web service provider



© 2007 IBM Corporation

Test and Problem Determination

Integration speeds time to market



Benefits:

- **Simplify development of zSeries test cases**
  - **Data creation for DB2, IMS/DB, VSAM, and QSAM**
  - **Extract and load**
- **Reduced deployment complexity**
  - **Production data validation and creation**
- **Common environment**
  - **Reuse of skills across e-bus and traditional applications**

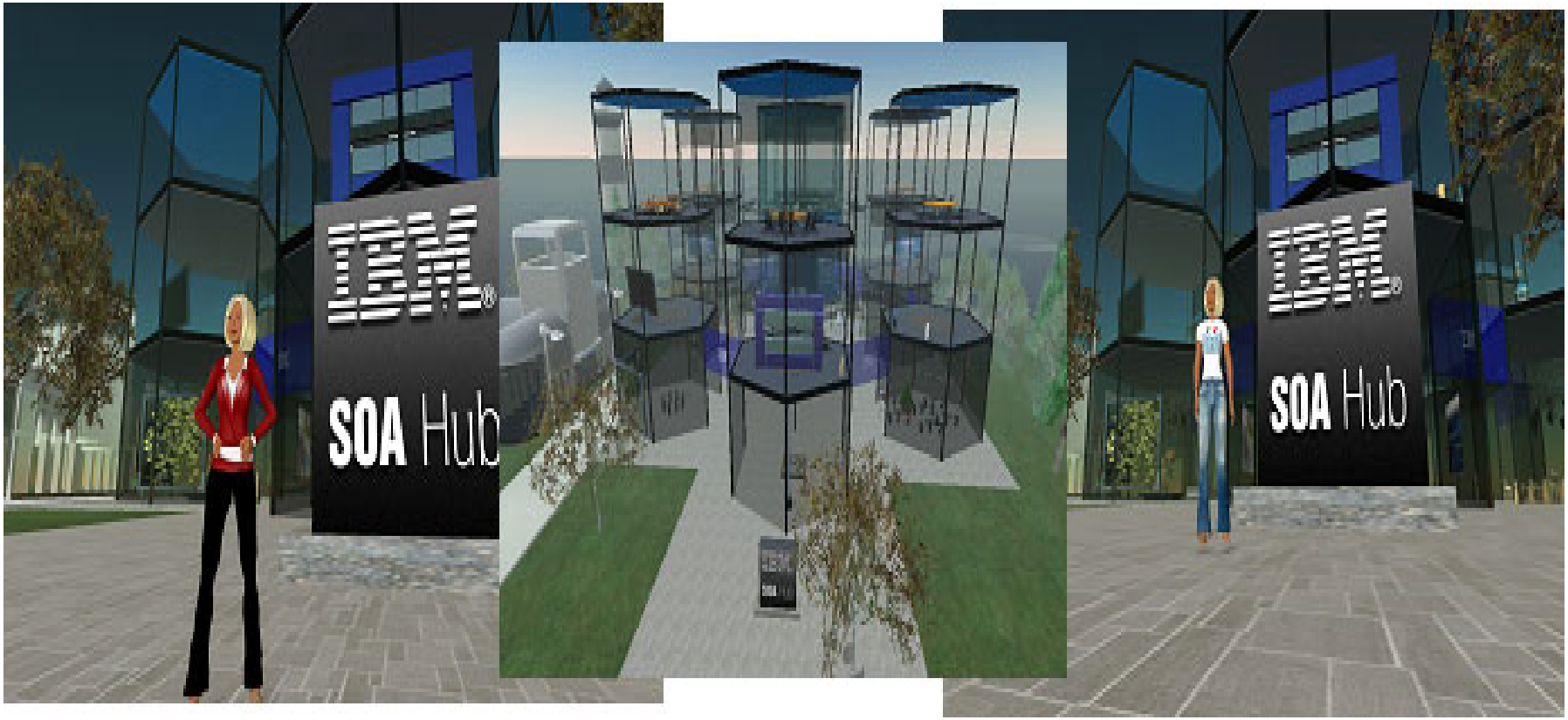
© 2007 IBM Corporation

Gartner: Best Practices for Mainframe SOA

- Act tactical, think strategic
- Evaluate tools that provide good microflow orchestration
- Create services that utilize function from across existing application boundaries.
- Build a reuse culture and technology infrastructure.
- Work with operations to create management/performance-monitoring support.
- Use code understanding/inventory/restructuring tools to improve service granularity.
- Define the role of the mainframe in future application architecture.

© 2007 IBM Corporation

Second Life ?



© 2007 IBM Corporation

SOA the next level : Web 2.0

Bridge between Web and Enterprise SOA

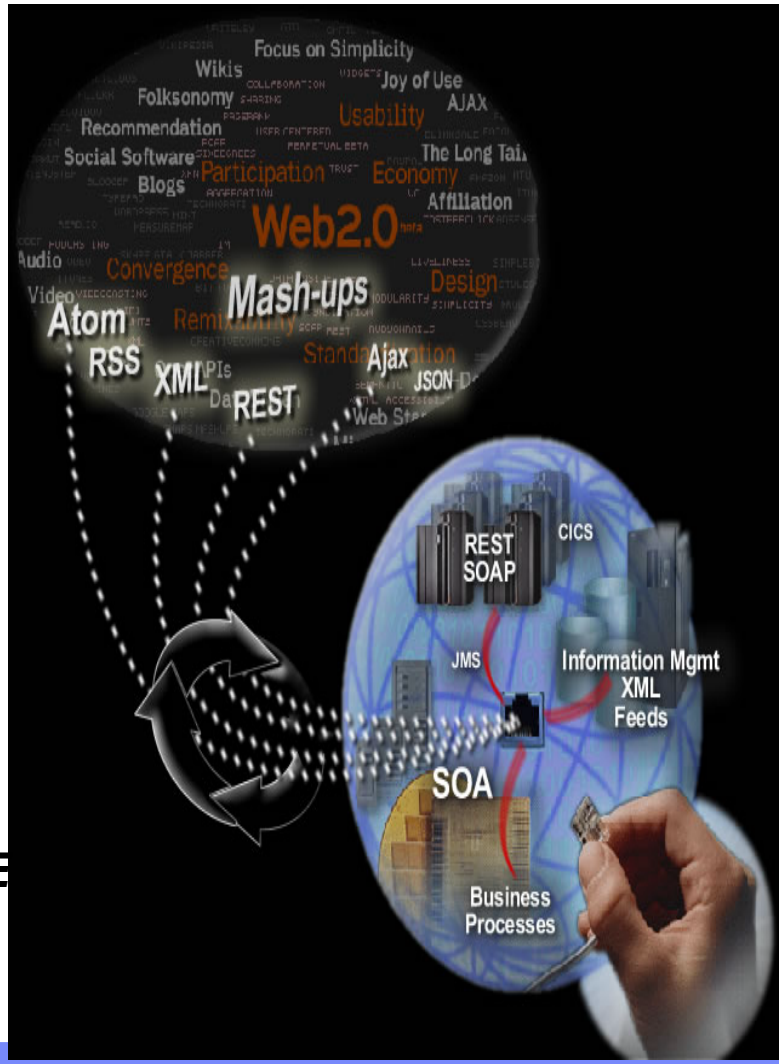
Expand SOA with Emerging Web 2.0 Technologies

- **The Web is about content :**
  - **Social Computing**
  - **Mash-Ups**
  - **Feeds**
  - **Rich User Experience (XML, AJAX, etc..)**

Continue Industry Web 2.0 Technology Collaborations

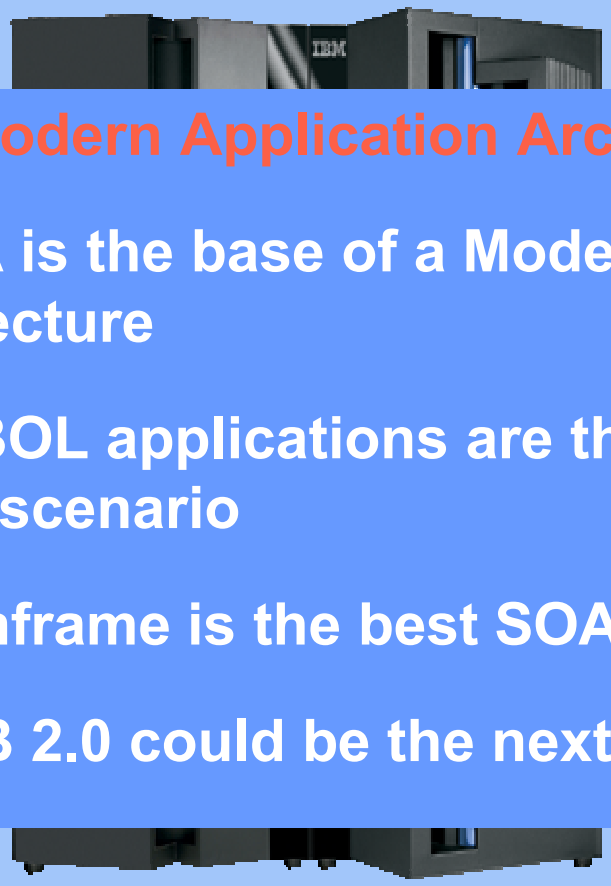
See video Web 2.0 on YouTube

http://www.youtube.com/watch?v=6gmP4nk0EOE



© 2007 IBM Corporation

Conclusion



Modern Application Architecture

- SOA is the base of a Modern Application Architecture
- COBOL applications are the key components in this scenario
- Mainframe is the best SOA Server
- WEB 2.0 could be the next step

© 2007 IBM Corporation