Interoperability Between Java™ EE Technology and .NET Applications

Laurence Moroney

Ray Lai

Marina Fisher

TS-4611

Connecting Systems Efficiently

It Should Be Second Nature!

Learn how and when to use the right technologies to meet different interoperability goals

Agenda

Exploring Interoperability

Web Services—Software as a Service

Bridging Middleware

- Bridging with .NET Remoting
- Messaging
- The Enterprise Service Bus

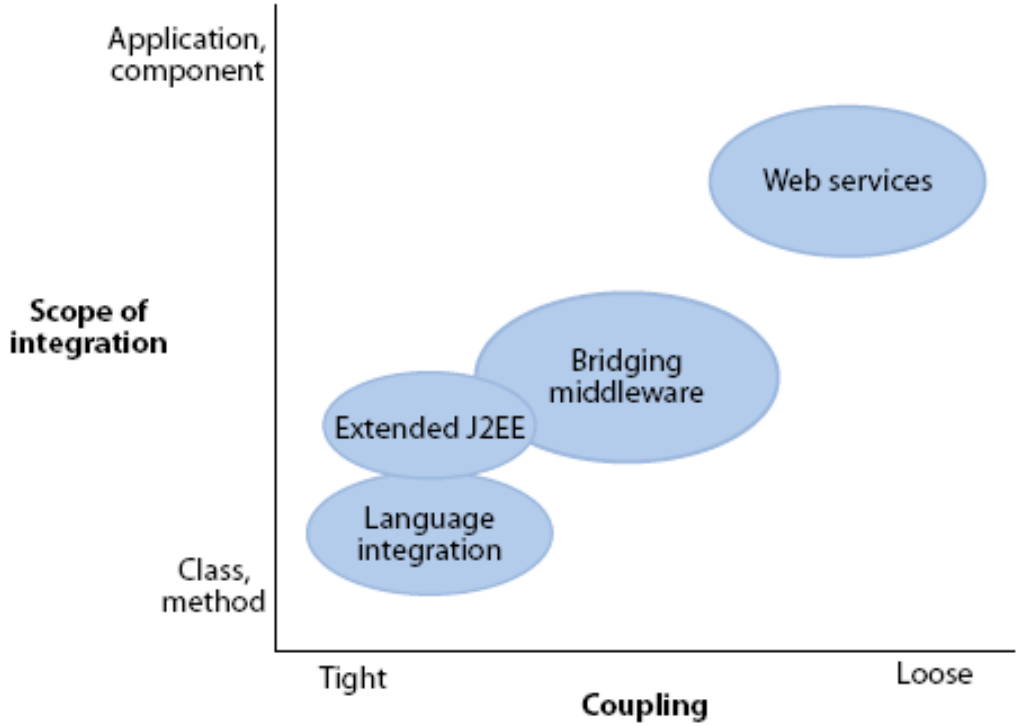
Extending Java EE: Platform Unification

Exploring Interoperability

- What is it?
  - A set of techniques and tools that allow software applications on different technology bases to communicate
  - The ability for software to share data regardless of the representation, retrieval or runtime technology used to manage the data
  - The ability for disparate software, runtime platform and operating systems to logically form a single virtual application platform

Exploring Interoperability

Interoperability Options (How)



Source: Forrester Research, Inc.

Source: Forrester Research: Choosing the best option for Java/.NET Interoperability, John Rymer January 31, 2006

Agenda

Exploring Interoperability

Web Services—Software as a Service

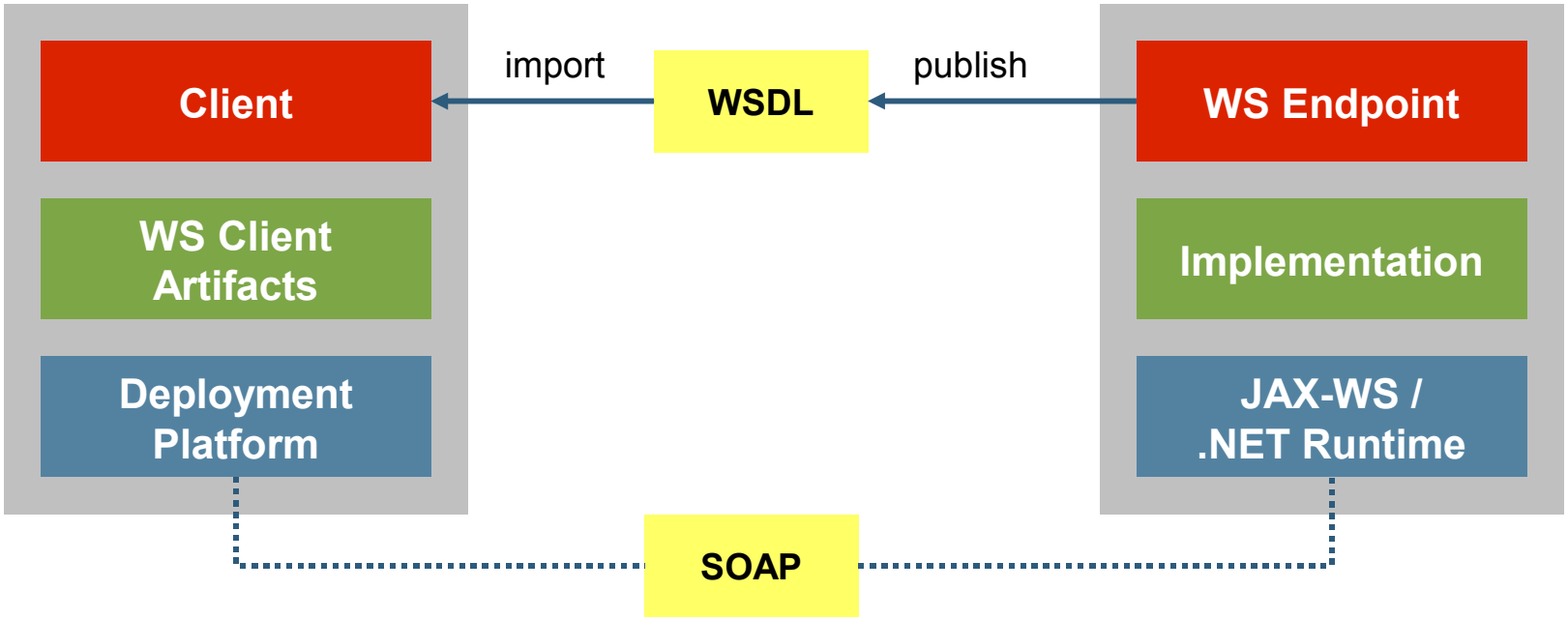
Bridging Middleware

- Bridging with .NET Remoting
- Messaging
- The Enterprise Service Bus

Extending Java EE: Platform Unification

Web Services—Software as a Service

Web Services Interoperability Architecture



Technologies

- WSIT (Web Services Interoperability Technology)
- JWSDP (Java Web Services Developer Pack) 2.x
- WSE (Web Services Enhancement) 3.0
- WCF (Windows Communication Foundation)

Standards

- Web Services Interoperability (WS-I) Organization

Web Services—Software as a Service

Strengths and Weaknesses

Strengths

- Technology agnostic
- Standards driven
- XML used for definition, remote procedure calls and data transfer

Weaknesses

- Performance overhead
- Designed for request/response, not real time or streaming information
- Problems with data mismatch caused by different interpretations of the standards by different vendors
- Managing states is complex, and could be problematic

Web Services—Software as a Service

Interoperability Challenges

- Troubleshooting
  - Inferring the root cause, e.g. configuration, anomalies
- Schema and data types
  - Data types mismatch, e.g. service was defined in Java or .NET before the schema is defined, incompatible or unsupported data types
  - WSI compliance
- Performance and scalability
- Versioning and backward compatibility
  - Tools versioning, e.g. JWSDP 1.x/2.0 interoperating with WSE 2.0/3.0 and WCF

Example #1—Troubleshooting (Cont.)

Sample Troubleshooting Methodology

- Using tools such as TCPSpy
  - Shows data types and exception details in the SOAP messages
- SOAP versions and “signature”
  - SOAP 1.1, and WS-I compliant—not an issue on SOAP versions
- Logical/runtime errors and warnings
  - The SOAP response doesn’t indicate any error
  - No SOAPFault—no obvious syntax or logical errors
- Schema and data types
  - Is the schema validated on both ends?
  - Data type compatibility (e.g. any, anyType)?
  - Further testing show:
    - addNumbers(1.40000000, 2.00000000 - 1.30000000) = 2
    - addNumbers(2, 3) = 5
    - addNumbers(1.4, 2.6) = 4

Example #1—Troubleshooting (Cont.)

Analysis

This poor example converts any number/ object into integers, and adds them. If the numbers have invalid format, return 0;

```
...
    [WebMethod]
    public int addNumbers(Object x, Object y)
    {
        try
        {
            int z = Convert.ToInt16(x) + Convert.ToInt16(y);
            return z;
        }
        catch (System.Exception)
        {
            Console.WriteLine("Cast integer exception in the
add operation");
            return 0;
        }
    }
}
```

Findings

- Data type conversion issue (Object to Int16)
- Exceptions handling—return when errors encountered
- “5432l” (long), not 54321 (integer)

Web Services—Software as a Service

Some Best Practices

Design	Development	Deployment
Schema / Data Types pp. 83-89	**Asynchronous Integration** pp. 169-245	**Scalability** pp. 505-507
Error Handling	**Messaging** pp. 247-319	**Performance** pp. 505-507
…	**Shared Data / Session** pp. 495-496, pp. 323-350	…
	Security pp. 444-481	
	…	

Note: Page numbers refer to the book [Marina et al.]

Web Services—Software as a Service

Best Practices—Development Stage

- Define schema first
- Keep types simple
  - Avoid advanced XML schema for data representation
  - Keep it simple for speed and stability
- Adhere to WS-I Profiles

Web Services—Software as a Service

Pitfalls

- Making assumptions about the service/source
  - Assume data types are verified
  - Assume client/server use the same SOAP version
  - Expect the use of synchronous Web services (e.g. RPC-style) should work for all business scenarios
- Making assumptions that different versions of WS tools are compatible or backward compatible
  - This underestimates the complexity of versioning and tools compatibility
- Use of RPC encoding
  - This is not WS-I compliant
- When to use Web Services

DEMO

Sun Java™ Studio Creator with SAP Business One™ Integration Based on Web Services

Agenda

Exploring Interoperability

Web Services—Software as a Service

Bridging Middleware

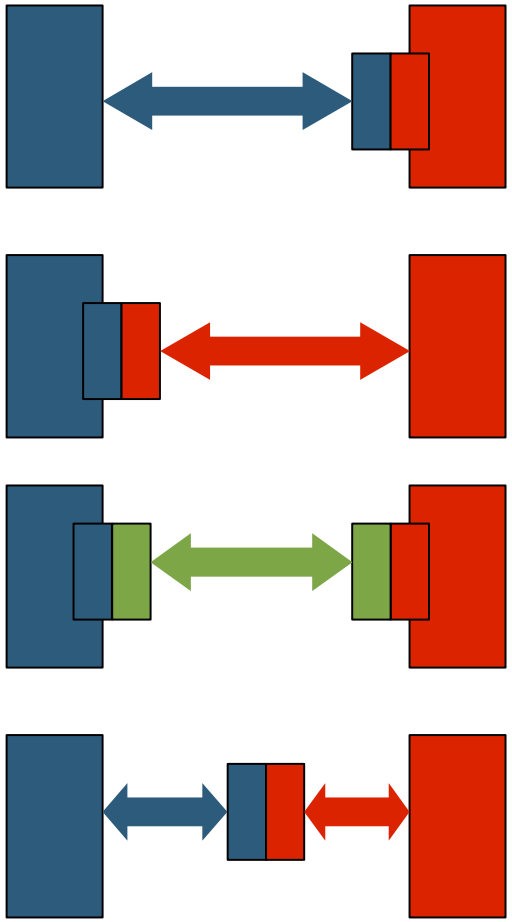
- Bridging with .NET Remoting
- Messaging
- The Enterprise Service Bus

Extending Java EE: Platform Unification

Bridging Middleware

What Is It?

- A wrapper at the service layer that speaks the language of the client layer
- A wrapper at the client layer that speaks the language of the service layer
- Wrappers at each end which manages communication between them
- A third layer in between the service and client that speaks both languages



Bridging with .NET Remoting

Solutions That Enable Bridging

- IIOP.NET
  - Open source framework; incorporates CORBA/IIOP into .NET using .NET Remoting
- Borland Janeva
- Intrinsyc's J-Integra for .NET
- JNBridgePro
- Codemesh JuggerNet

Bridging Middleware Interoperability

Strengths and Weaknesses

Strengths

- Highly interoperable
- Strong in tightly coupled scenarios
- Perform very well
- Secure (SSL)

Weaknesses

- Mostly proprietary
- Brittleness when extending application
- Requires specialized skill set
- Increases system complexity and maintenance

Agenda

Exploring Interoperability

Web Services—Software as a Service

Bridging Middleware

- Bridging with .NET Remoting
- Messaging
- The Enterprise Service Bus

Extending Java EE: Platform Unification

Interoperability with Messaging

Messaging Bridge

Adapter

Web Services Messaging

Internet Email for Asynchronous Messaging

Messaging Middleware Interoperability

Strengths and Weaknesses

Strengths

- Secure, reliable, asynchronous communication
- Leverage existing infrastructure
- Reliability with some of the strategies: WS with messaging, Internet email server
- Failover

Weaknesses

- Troubleshooting can be difficult between MSMQ and JMS messaging products
- WS Façade may become a bottleneck and impact reliability

Agenda

Exploring Interoperability

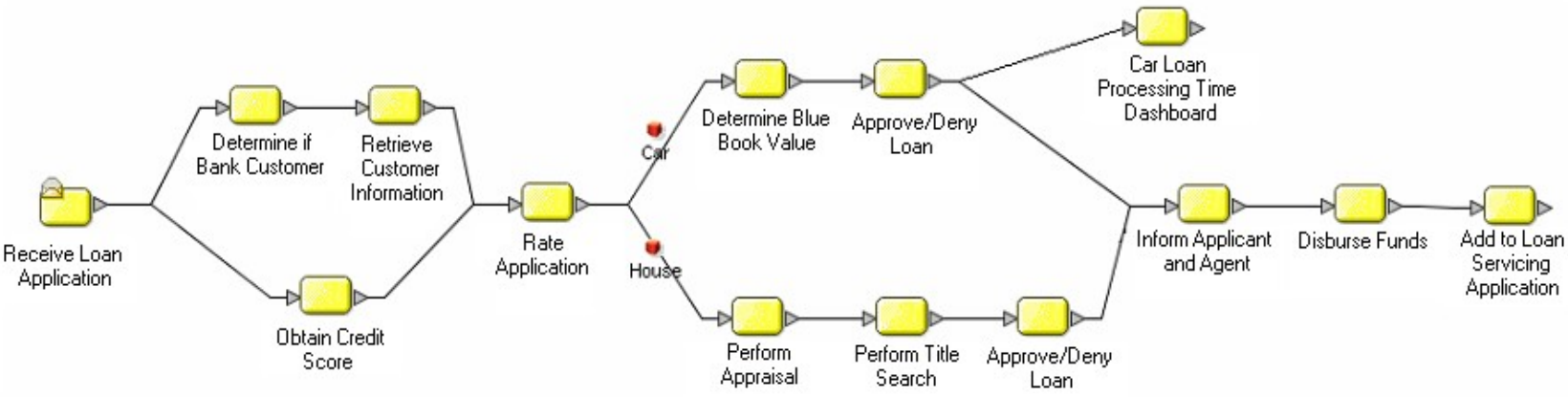
Web Services—Software as a Service

Bridging Middleware

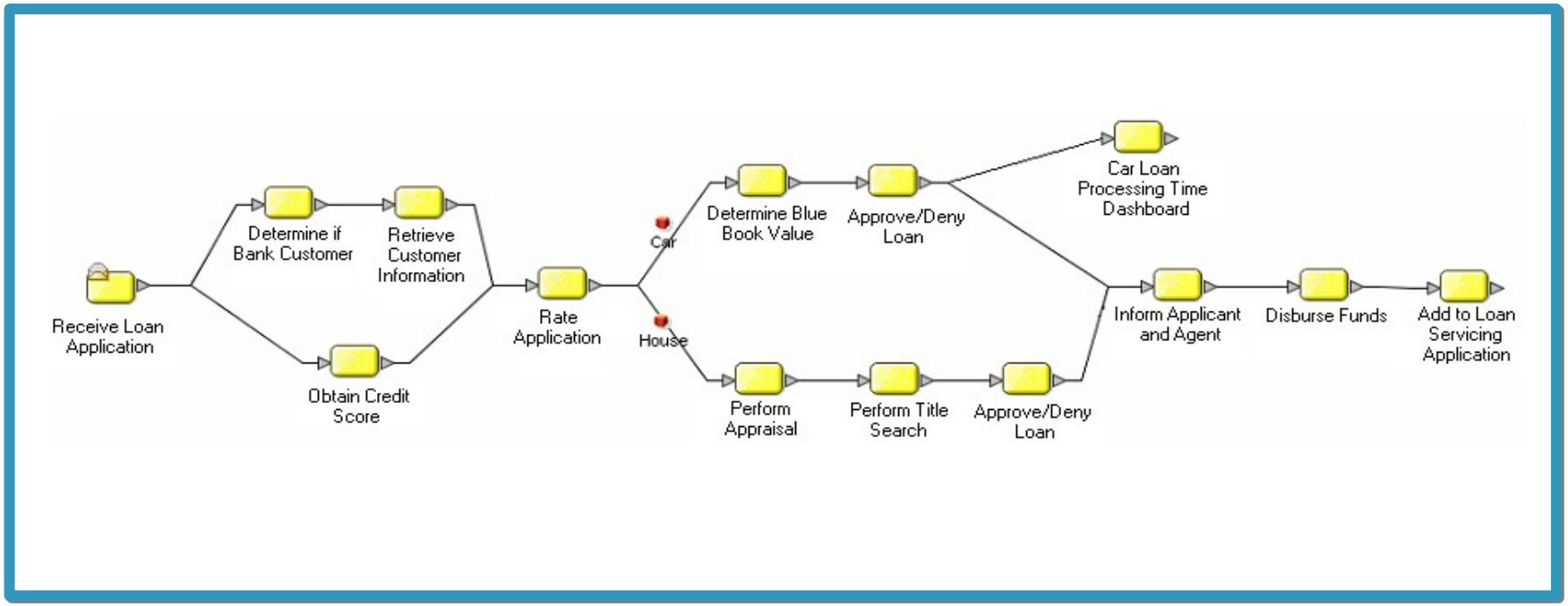
- Bridging with .NET Remoting
- Messaging
- The Enterprise Service Bus

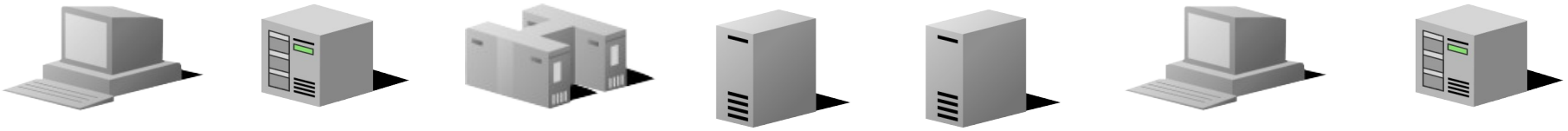
Extending Java EE: Platform Unification

ESB for Composite Applications



Composite Applications Are Leveraging Existing Systems





Car Dealer Systems

Estate Agent Systems

Credit Reporting Applications

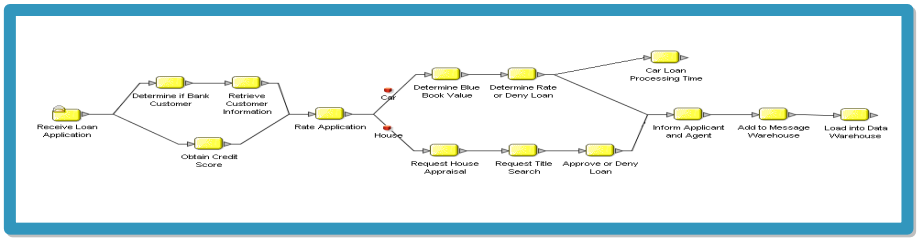
Internal Banking Systems

Car Valuation Application

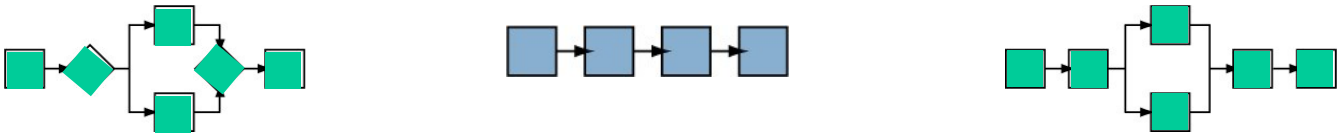
Loan Applicant Rating

Funds Disbursement

Composite Applications and SOA

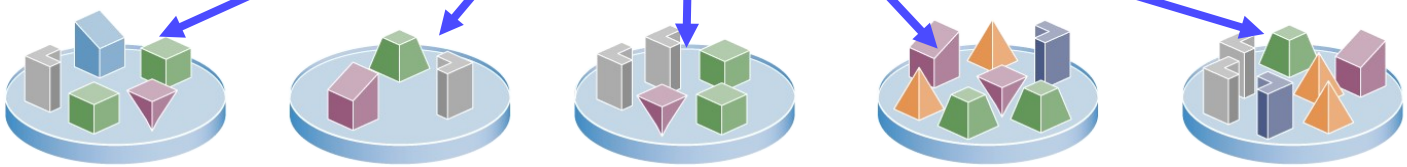


Order Processing Composite Application



Composed Business Service

Installation Scheduling | Process Customer Order | Bill Presentment/Payment

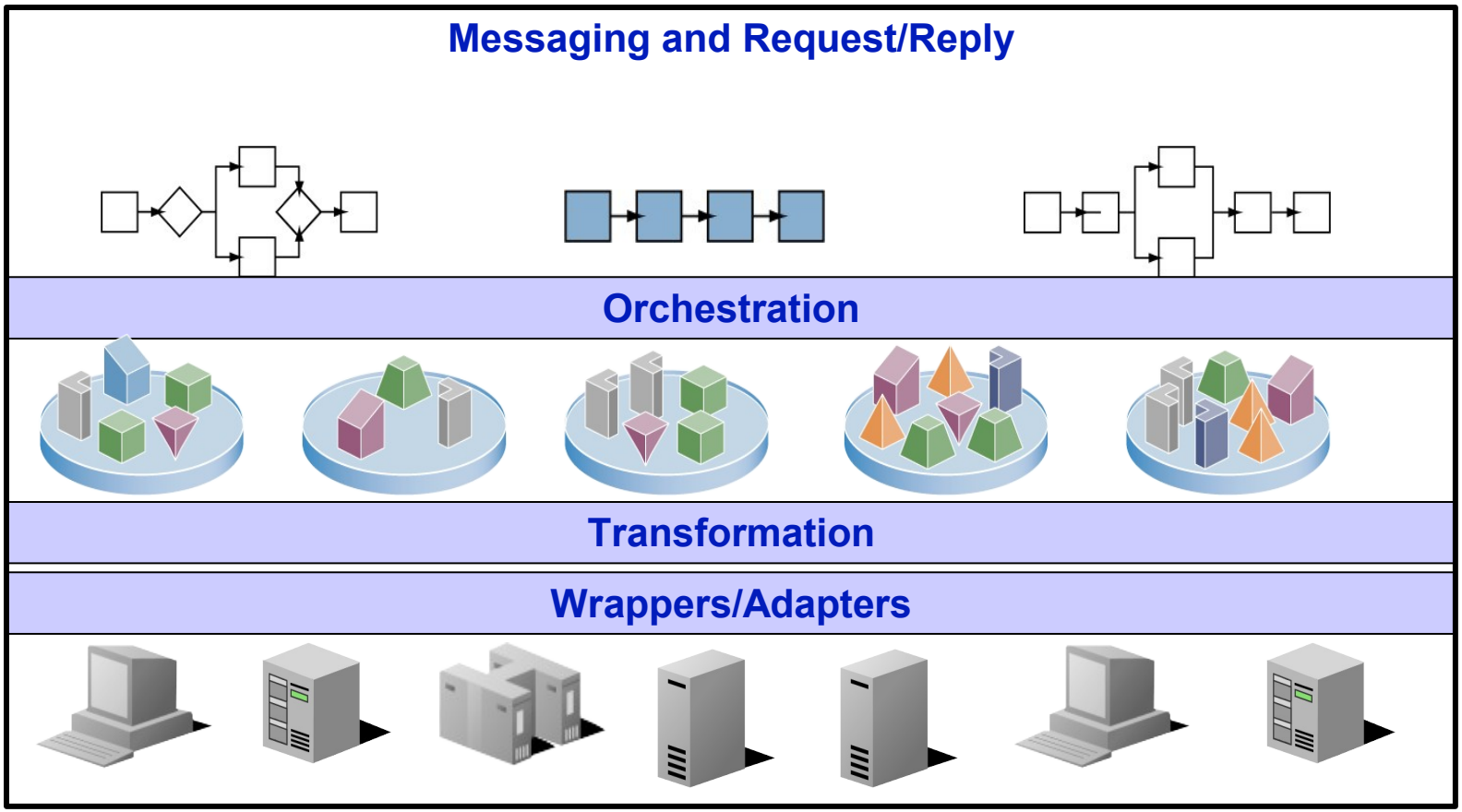


Elemental Business Services

Check Customer Status | Verify Customer Credit | Look-up Customer Discount | Determine Product Availability | Calculate Shipping Charges

Distribution Systems | Materials Management Systems | Credit Assessment Applications | Manufacturing Scheduling Systems | Materials Procurement Application | Logistics Management System | Order Processing Application

ESB: Adapters, Transformations, Orchestration, and Communication



Enterprise Service Bus

Strengths and Weaknesses

Strengths

- Canonical message format
- Management/toolkit
- Scalability
- Reliability

Weaknesses

- License fee
- Evolving standards (SCA, JBI)
- Utilize emerging technologies (WS-*)

The Enterprise Service Bus

ESB Solutions

Commercial

- Sun Java Composite Application Platform Suite
- Sonic ESB
- Fiorano ESB
- TIBCO ESB
- IBM WebSphere ESB
- Oracle ESB
- BEA AquaLogic Service Bus
- Cape Clear ESB

Open Source

- Sun Open ESB
- Mule
- IONA Celtix
- ObjectWeb ESBi
- LogicBlaze FUSE
- Apache ServiceMix
- Apache Synapse

The Enterprise Service Bus

Best Practices—Analyzing ESB Interoperability Features

Features	Product A	Product B
Architecture	Peer-to-peer	Centralized
Web container (a.k.a. server)	Tomcat	Java EE application server
JBI container	Sun JBI RI	Proprietary
Workflow integration support	BPEL (PXE)	BPEL (home-grown), proprietary
Client Development Model	Send events to Event Manager	Use proprietary APIs or BPEL scripting
Client language support	Java, .NET (C#, via Web services)	Java
Framework (e.g. MVC) support	Spring	Struts
Messaging	JMS, SMTP, POP, TCP	Proprietary middleware
Connector support	25+ connectors	120+ connectors
Web Services Support	WS-I compliant, WS*	WS* (a.k.a. WSIT)
Scalability and Performance	750 messages per second	500 messages per second

Agenda

Exploring Interoperability

Web Services—Software as a Service

Bridging Middleware

- Bridging with .NET Remoting
- Messaging
- The Enterprise Service Bus

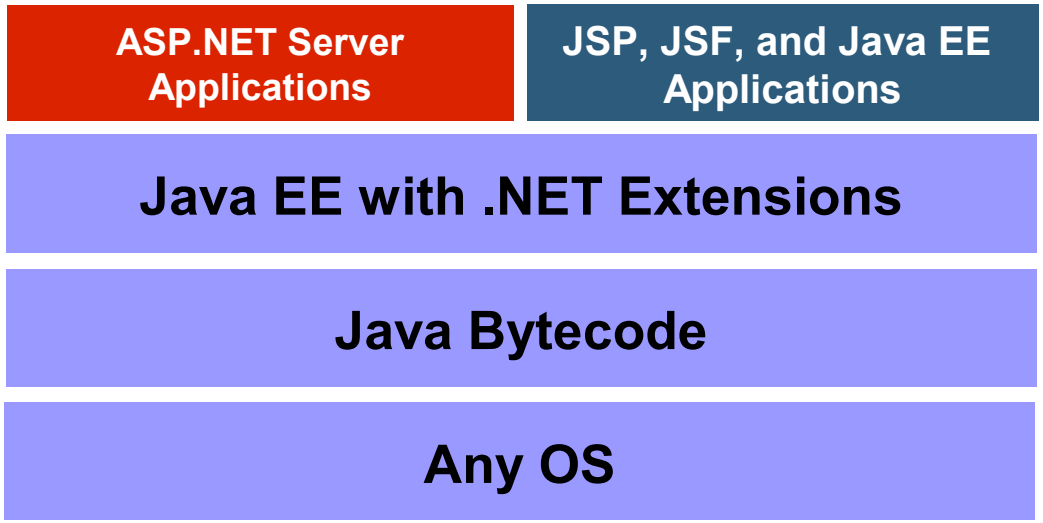
Extending Java EE: Platform Unification

Extending Java EE: Platform Unification

What Is It?

- Extensions to Java EE that allow .NET applications to run on an application server
- Compile-time translation layer that converts MSIL into Java byte code

Applications execute on the same VM, thus they 'understand' each others data types and can interoperate cleanly



Extending Java EE: Platform Unification

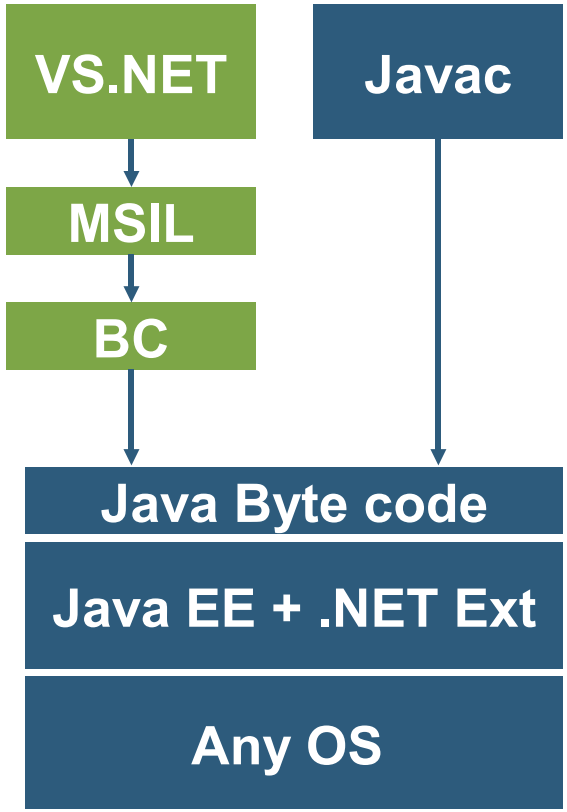
Impact on Interoperability

- .NET applications run on the same JVM as Java EE apps, thus they do not need a communication ‘layer’ between them
- .NET applications can consume Java resources, including EJBs, directly and vice versa
- Supported on WebSphere, WebLogic, JBoss, Tomcat

Extending Java EE: Platform Unification

How Does It Work?

- Visual MainWin™ for J2EE from Mainsoft
  - Plug-in for Visual Studio.NET allowing developers to use C# or VB.NET
  - Binary compiler that converts MSIL into Java Byte code
  - Java version of .NET framework class libraries



DEMO

Re-Hosting a .NET Application on Java EE

Overall Interoperability Strategy Checklist

Requirements	Web Services	Bridging	Messaging	ESB	J2EE Extension
Granularity	Coarse	Fine	Fine	Fine	Application Specific
Coupling	Loose	Mildly loose	Mildly loose	Mildly loose	Loose
Performance and Scalability	Reasonable	High (e.g. IIOP)	Reasonable	Reasonable	Reasonable
Manageability (e.g. fault management, configuration management)	Maturing	Good	Reasonable	Maturing	Good
Security	WS*	Varies	MSMQ/JMS security	Varies	Java EE platform security
Legacy Requirements	Good for integrating legacy systems by opening up their interfaces in web services	Depends	Good when existing messaging infrastructures use MSMQ and JMS	Good for integrating legacy systems	Application Specific
Delivery lead time	Fast	Varies	Varies	Fast	Fast

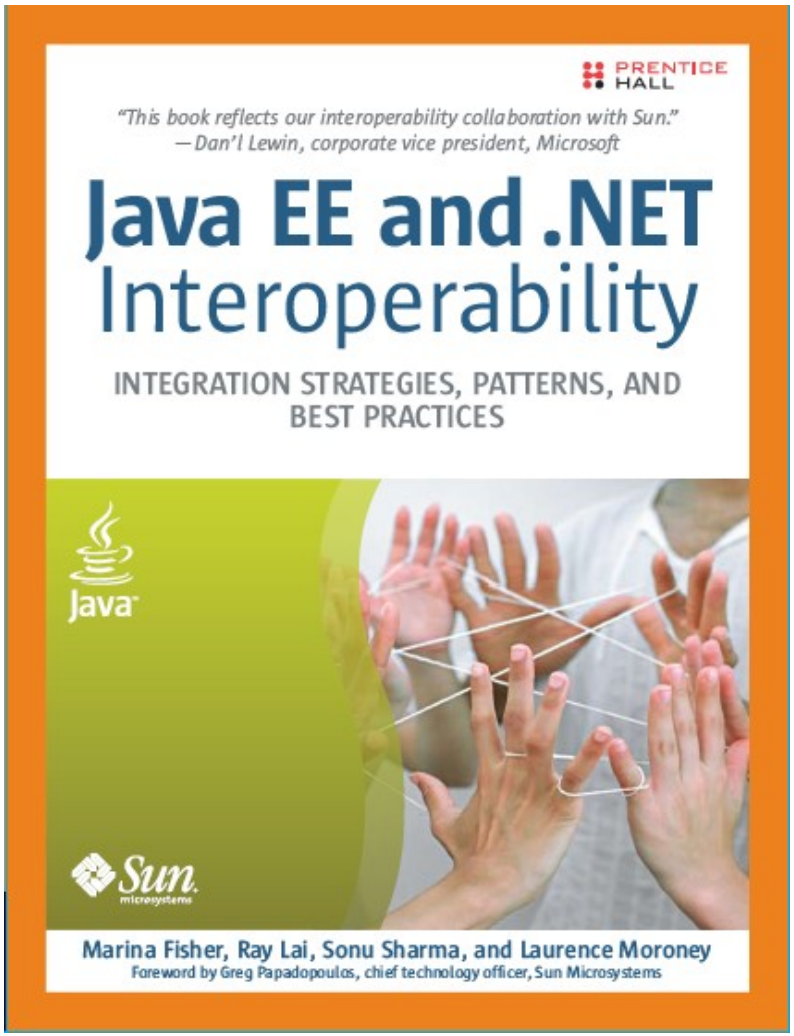
[Marina et al. pp. 546-549]

Summary

- Many interoperability options!
- Consider them in terms of
  - Synchronous or asynchronous integration
  - Performance, reliability and other systemic qualities
  - Impact on manageability through runtime complexity
  - Technology in-house expertise
  - Development cost
- Stick with standards where possible—using WS-I is a must for Web Service interoperability

For More Information

- Java EE and .NET Interoperability [Marina et al.]
  - www.prenhallprofessional.com/title/0131472232
  - javanetinterop.dev.java.net
- www.ws-i.org
- www.mainsoft.com
- iiop.sourceforge.net
- j-integra.intrinsyc.com/
- jnbridge.com



Q&A

Interoperability Between Java™ EE Technology and .NET Applications

Laurence Moroney

Ray Lai

Marina Fisher

TS-4611