MICHIGAN STATE UNIVERSITY

5. Service Oriented Architecture/.NET

CSE 498, Collaborative Design

Brian Loomis
Microsoft Corporation

Department of Computer Science and Engineering
Michigan State University

Spring 2006

Overview

- Intro to web services
  - What is an XML Web service?
  - Consuming web services
  - WS standards
  - Demo of a simple web service
- How is SOA different?
  - Demo of Indigo

5-2

What Is An XML Web Service?

A programmable application component accessible via standard Web protocols

- **UDDI - Provides a directory of services on the Internet**
- **WSDL - XML Web services are defined in terms of the formats and ordering of messages**
- **SOAP - XML Web service consumers can send and receive messages using XML (wire format)**
- **Built using open Internet protocols**

UDDI
Universal Description, Discovery and Integration

WSDL
XML Web services Description Language

SOAP

XML and HTTP

5-3

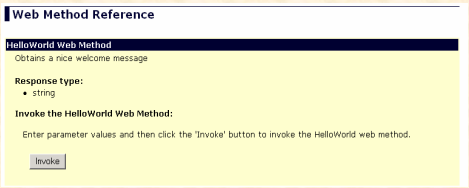
What Does An XML Web Service Do?

- XML Web services
  - Expose functionality as a service and allow applications to share data
  - Can be called across platforms and operating systems and regardless of programming language
  - "A function over HTTP"
- Web Services are part of the ASP.NET application model
  - Web Services must be URL accessible via IIS
  - Web Services have full access to ASP.NET object model (Request, Session, Application, etc.)
- The web service emits no UI
  - The web browser is not the intended client

5-4

Simple Web Service Code

```
[WebMethod(Description="Obtains a nice welcome message")]
public string HelloWorld()
{
   return "Hello World";
}
```



5-5

Protocols for calling web services

- HTTP-Get
  - Provide a URI to the web method, navigate to it, and the method returns values as XML
- HTTP-Post
  - Provide a URI as the Action for the Post
  - Provide method parameters s via Name-Value pairs (Hidden, text fields, etc.)
    - EX: Browse to an ASMX page
- SOAP
  - Use proxy class generated from WSDL
  - Supports more control and flexibility than HTTP-Get/Post

5-6

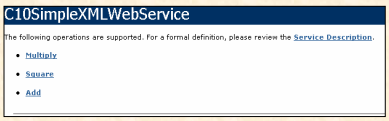
Building a consumer or client

- WinForms application
  - VB, C#
    1. Add Web Reference
    2. ws = new YourServer.ServiceName;
  - Services, too…
    - getStockPrice.asmx
- WebForms application
  - ASP.NET
    - create a proxy class (re-generate if WSDL changes)
    - just reference like any other .NET assembly
- Business applications
  - Custom SOAP
  - Other wire protocols (Winsock, Wininet)
  - With WebServiceUtil.exe (***WSDL.exe***)
    - Creates a proxy class in C# or VB.NET from a WSDL file

5-7

Show me the code!!!

- Basic web service
- Build a simple smart client



5-8

MICHIGAN STATE UNIVERSITY

Design aspects of web services

Things to Think About… Design Requirements

- How do I increase throughput (reduce latency)? (Performance)
- What if a Web Service is down? (Availability)
  - Does the client just die/timeout?
  - Do I allow for graceful fail-over? If so, how?
- Do your Web Services need to know about state? If so, where does the state go? (Scalability)
  - Application state (and methods)
  - Session state
  - In the message itself? SOAP headers, cookies, etc.
  - Should I use remoting instead?
- How do I guarantee a web service? (Reliability)

5-10

Designing an XML Web Service Design Guidelines

- Web Services are even more coarse grained than components
  - Don't send unnecessary data (such as an image) when you can send a URL
  - Cache data from the service where possible, rather than requesting the same data 100 times
  - Be efficient about the number of requests for dynamic data - collapse multiple web service methods into one (chunky versus chatty)
  - Group related operations and data into one method
- Decision points:
  - Know and understand the supported data types to determine arguments and return data
  - Whether to use existing classes or create new ones
  - How do I handle security?
  - Define your input and output data structures using XSD schemas or ADO DataSets

5-11

Differences from Component Design

- Build the service to be asynchronous if the potential exists to block other work
  - Separate XML Web service from underlying business logic
- Handle client errors when the server is unavailable
  - Use output caching to improve performance
- Exposing existing apps as web services may reveal new bottlenecks
  - Think about prototyping
  - Unmanaged/managed code transitions
- Watch out for…
  - Machine affinities (clustering for scalability)
  - Single points of failure (DTC, MSMQ, etc.)

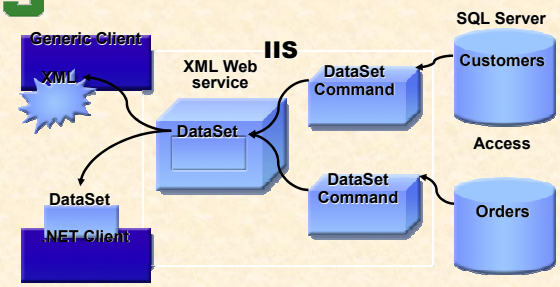
5-12

Serialization

- Class and Struct marshalling is done via XML Serialization
  - Serialization ONLY occurs on public fields and properties
  - Objects utilizing private fields may transfer incorrectly if they don't have a corresponding public get/set property
  - Standard Methods/Functions in your Biz Class will NOT serialize
  - To maintain Methods afterwards you must maintain the same type definition residing on both sides (e.g. DataSet)
- Web Services are designed as a data transport, not a distributed object system
  - Types that serialize:
    - string, Boolean, int64, uint64, guid, xmlqualifiedname, xmlnode, char, int16, uint16, single, decimal, class (only public properties), DataSet, byte, int32, uint32, double, DateTime, struct
- Web Service designers need to keep the XML Schema and Payload size in mind
- If you need to pass fully functional business objects use .NET Remoting; how do I pass other types of objects (Serialization)
  - This becomes more important on bigger systems

5-13

Returning Data From An XML Web Service



MICHIGAN STATE UNIVERSITY

Service Oriented Architecture

Now that you know the building blocks

Service Oriented Architecture

- An architectural approach to creating systems built from autonomous services
  - Integration as a fore-thought rather than an after-thought
- A service is a program you interact with via message exchanges
  - Services are built to last
  - Availability and stability are critical
- A system is a set of deployed services cooperating in a given task
  - Systems are built to change
  - Adapt to new services after deployment

5-16

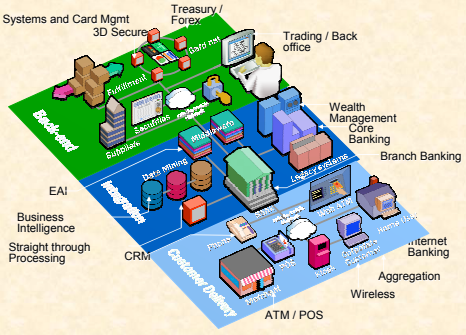
What is SOA?

Service oriented architecture

1. A method of architecting systems across organizations/companies and assuming different underlying technologies, using generally a web service interface (policies and practices)
   - "A service oriented architecture (SOA) significantly elevates the abstraction level for code re-use; it allows applications to bind to services that evolve and improve over time without requiring modification to the applications that consume them"
2. Specific technology extensions/interfaces which extend the WS model to support this integration (frameworks)

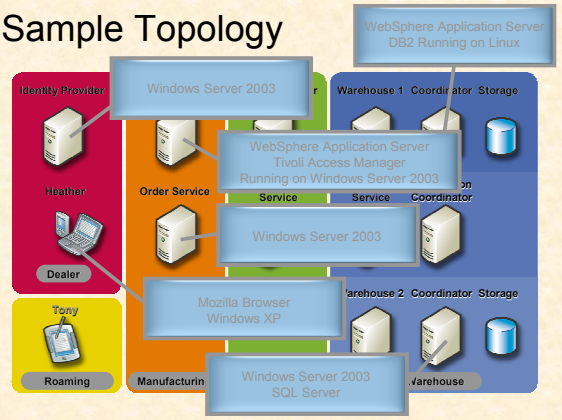
5-17

Connected systems across organizations



5-18

Sample Topology



The architecture piece

- At the highest level, this is "how" we tie sets of web services (grouped as applications) together
  - Need agreement across all applications on the protocol and dependencies
  - E.g., Integrating apps inside my company (CRM, shop floor, AP/AR), and maybe I have a third party web services which I buy for computing tax on shipments or finding a parts vendor
- Tradeoff of speed versus cross-platform compatibility
  - We can't assume that both ends use the same technology
    - Relatively closed solutions to distributed computing have been tried in the past (DCOM, RMI, CORBA, etc.)
  - We can't assume each end even knows the other
    - Discovery mechanisms add metadata discovery layer

5-20

The Four Tenets of Service-Orientation

Boundaries are Explicit	Developers opt-in to consuming, exposing, and defining public-facing service façade.
Services are autonomous	Services and consumers are independently versioned, deployed, operated, and secured.
Share schema & contract, not class	Data never includes behavior; Objects with data and behavior are a local phenomenon.
Compatibility based on policy	Capabilities and requirements represented by a unique public name; Used to establish service suitability.

The technology toolkit of SOA

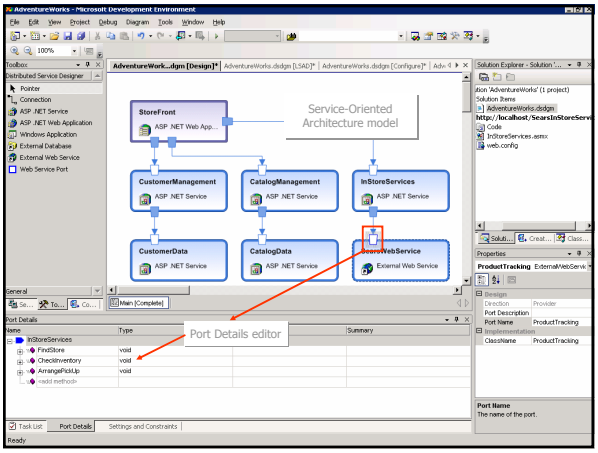
- Business process standards
  - XLANG, OASIS, BPEL and many others
    - Business processes are seldom "generic" and need customization for specific usage
- WS-E includes
  - WS-AtomicTransaction
  - WS-BusinessActivity
  - WS-Coordination, WS-Addressing
  - WS-ReliableMessaging
  - WS-Security, WS-Trust
  - WS-Routing, WS-Policy
  - WS-Referral
  - WS-Attachment

5-22

The technology piece on the MS stack

- It depends on what you're doing… i.e., what sort of distributed computation you're performing
  - Remoting (custom code, very fast)
  - WS-E in VS 2005 with MSMQ
    - Custom code may be applicable if you already have an app
    - Good for bridging and adding logic between services
  - Windows Workflow and Communications Service (lightweight, OS service for coordination of less structured communications)
  - BizTalk (high performance server for regular and high-throughput communication patterns)
  - SQL Server with web services (fast intranet access to data with low processing factor)
- All support transactions and open standards
- All have different cost of ownership/implementation
- You may use several of these in an SOA system

5-23



Productivity

Using Visual Studio 2005 and Indigo

```
[ServiceContract(SecureChannel, SecurityMode ="Windows")]
[Reliability(Guarantees.ExactlyOnce | Guarantees.InOrder)]
[ServiceContract]
class HelloService
{
 [ServiceOperation(TransactionFlowAllowed = true)]
 String Hello(String Greeting)
 {
  return Greeting;
 }
}
```

1 line security

1 line reliable messaging

1 line transactions

Total lines 3

5-25

Show me the code!!!

- Model designers
- WS-AtomicTransaction

Demo

5-26

Resources

- WS-I and specs - http://www.ws-i.org/ and http://www.uddi.org/
- Microsoft web services dev center
  - WSE 3.0 - http://www.microsoft.com/downloads/details.aspx?familyid=6bafe8a6-cdc9-4ae6-9625-e6260ccdef24&displaylang=en and http://msdn.microsoft.com/webservices/default.aspx?pull=/library/en-us/dnwebsrv/html/wsacoord.asp
  - Web services - www.microsoft.com/net/xmlwebservices.asp and downloads at http://msdn.microsoft.com/webservices/downloads/default.aspx
  - http://msdn.microsoft.com/webservices/default.aspx?pull=/library/en-us/dnwse/html/newwse3.asp
- Microsoft architecture and SOA guidance center
  - http://msdn.microsoft.com/architecture/
  - http://msdn.microsoft.com/architecture/soa/default.aspx
  - http://msdn.microsoft.com/webservices/default.aspx?pull=/library/en-us/dnbda/html/soadesign.asp
  - http://msdn.microsoft.com/webservices/default.aspx?pull=/library/en-us/dnbda/html/soa_vb-1.asp
  - http://msdn.microsoft.com/msdnmag/issues/04/01/indigo/default.aspx (Don Box)
  - http://msdn.microsoft.com/msdntv/episode.aspx?xml=episodes/en/20030827SOAPDB/manifest.xml (his webcast)
  - Connected systems intro - http://www.microsoft.com/windowsserversystem/overview/benefits/cstrial.mspx
- Microsoft Windows Workflow and BizTalk
  - http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnlong/html/WWFIntro.asp
  - http://www.microsoft.com/biztalk/default.mspx
- The "indy" press
  - http://www.enterprise-soa.com/ and http://www.developer.com/java/web/print.php/2207371
  - http://www.service-architecture.com/
- The hard way
  - http://www-128.ibm.com/developerworks/library/ws-statefulws5/?ca=dnt-536
  - http://www.oracle.com/technology/tech/webservices/index.html

5-27