Tutorial on Web Services

Alberto Manuel Rodrigues da Silva

Prof. DEI/IST/UTL, Portugal

Contents

- Overview
- Architecture
  - Microsoft .NET framework and ASP.NET
- Creating Web Services
- Using Web Services
  - Invoking and consuming Web Services
- Advanced Web Services
  - State Management, Security, and Transactions
- Web Services Enhancements

Section 1: Overview

- “Looking Back ...“
- What are Web Services?
- Distributed Web applications

Looking Back ...

- Traditional distributed computing
  - Client/server model
  - Distributed object model
    - Components: packaging and interoperability
    - Remoting: remote method invocation
    - COM, CORBA, Java RMI and EJB
- Microsoft Windows DNA
  - Distributed interNet Application Architecture
  - DHTML, COM, ASP, Message Queuing
- J2EE (Java 2 Enterprise Edition
  - RMI, JAXM,JAX-RPC, JAXR, …, EJB

What’s Wrong with That?

- Distributed object models don‘t scale to the Internet
  - Tightly coupling service and consumer
    - Need for homogeneous infrastructure
    - Versioning problems
- Limited COM support on non-Windows platforms
- CORBA is a remoting architecture
  - CORBA Component Model
  - Server object implementation not portable
- EJB—Enterprise JavaBeans
- The purely interactive Web

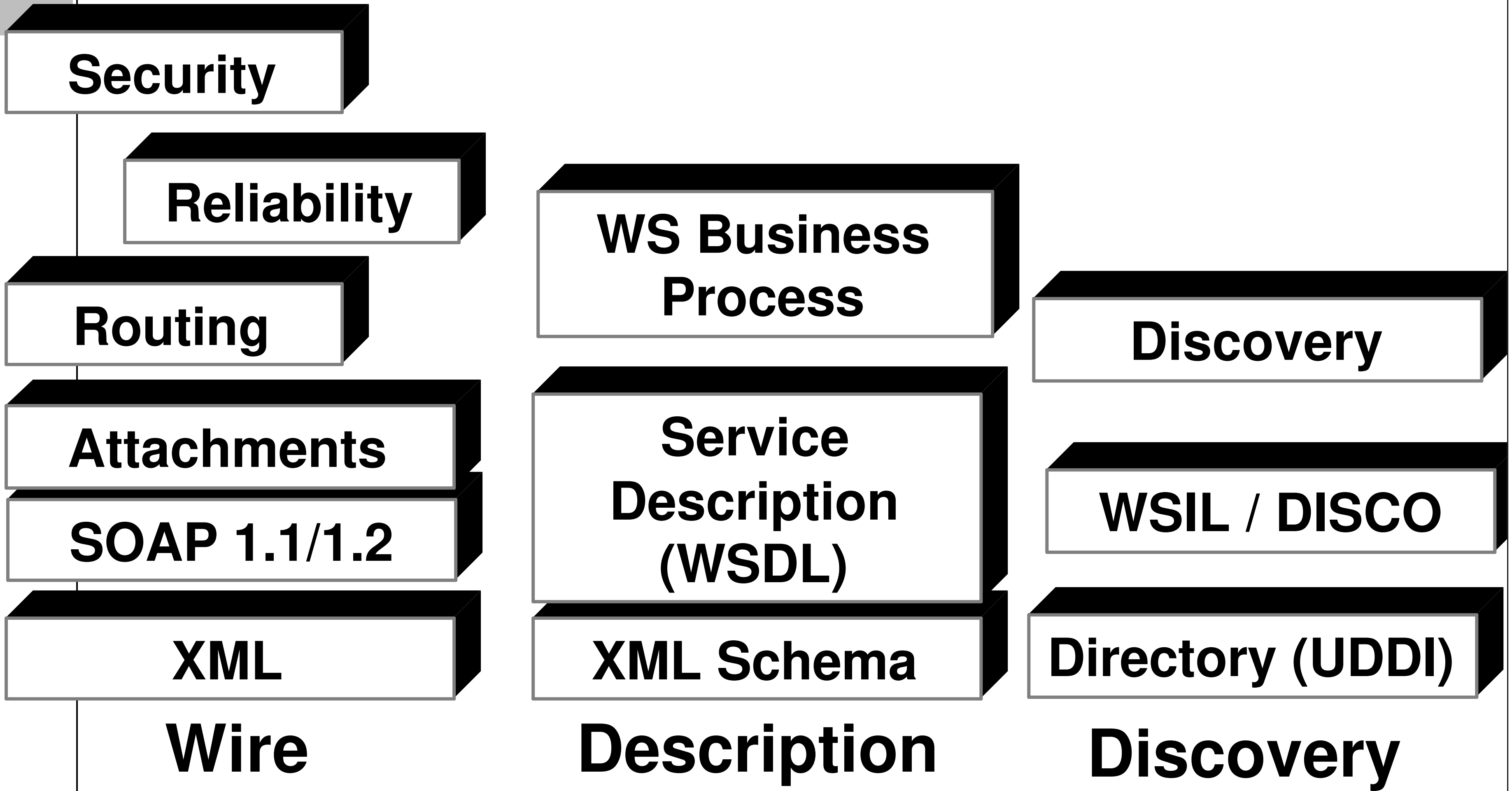
Web Services—Basics

- Expose services to other processes
  - Internet or intranet
- Black boxes
  - Component-like, reusable
- Supported by Produtive Framework
  - ASP.NET Web Services model
  - J2EE
- Based on open standards
  - HTTP, XML, and SOAP

Web Services—Basics

- Interconnect
  - Applications
  - Different clients
  - (M)any device
- Distribution and integration of application logic
- Web Services are loosely coupled
- Enable the programmable Web
  - Not just the purely interactive Web

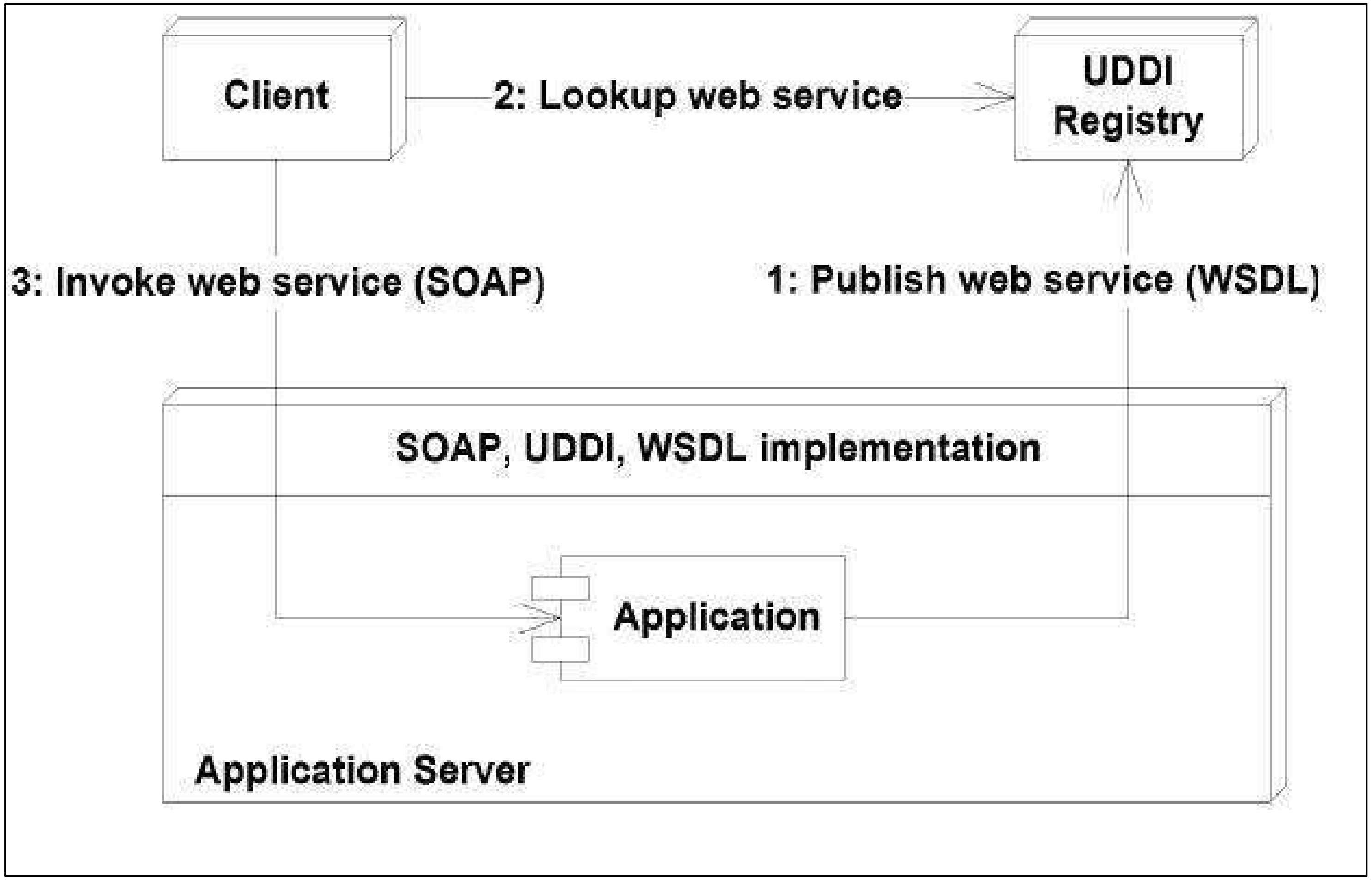
Web Services **Infrastructure**



Web Services **Infrastructure**

Publish, Find, Use Services:	**UDDI**
Service Interactions:	**SOAP, WSDL**
Universal Data Format:	**XML**
Communications Channel:	**Internet**

Distributed Web Applications


Client
2: Lookup web service
UDDI Registry
3: Invoke web service (SOAP)
1: Publish web service (WSDL)
SOAP, UDDI, WSDL implementation
Application
Application Server

SOAP (Simple Object Access Protocol)

✍ W3C specification

✍ SOAP is a lightweight protocol for exchange of information in a decentralized, distributed environment.

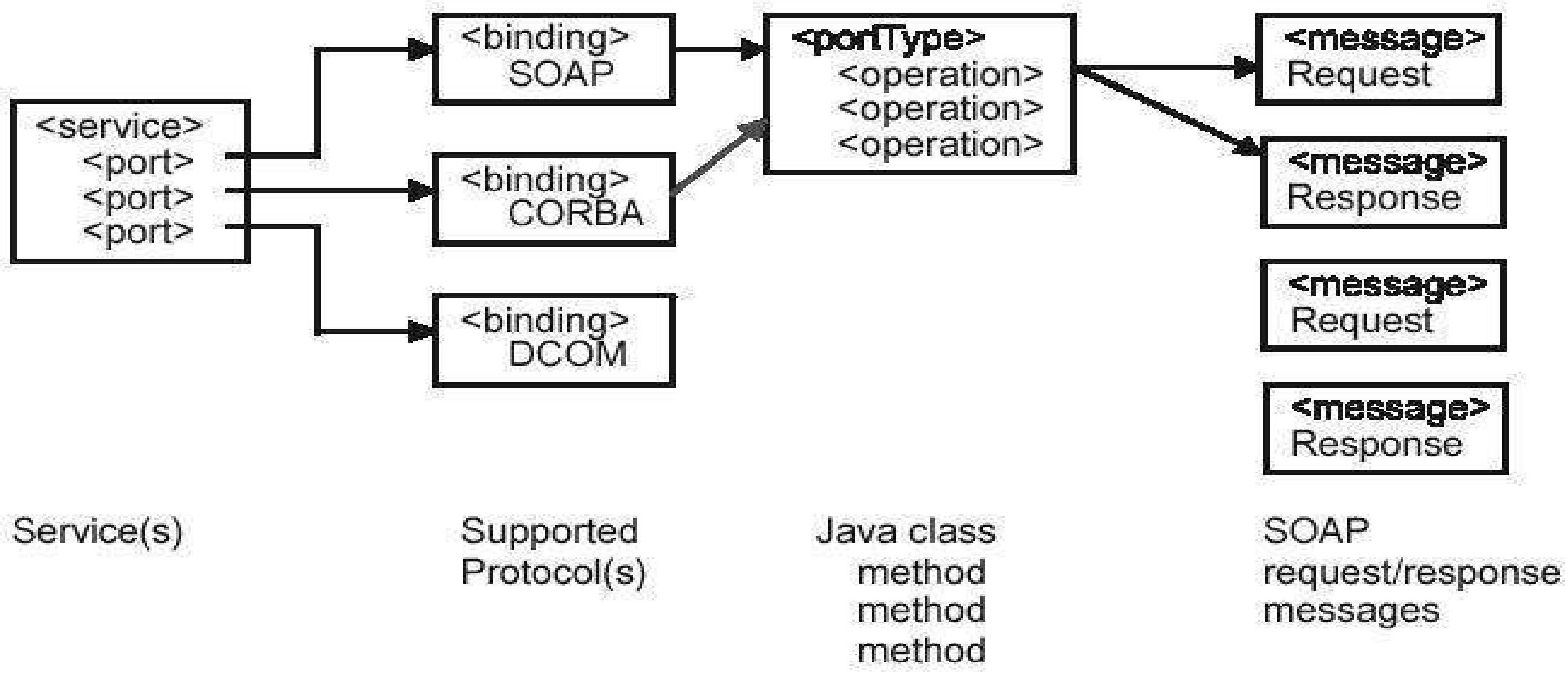
✍ It is an XML based protocol that consists of three parts:

- An envelope that defines a framework for describing what is in a message and how to process it
- A set of encoding rules for expressing instances of application-defined datatypes
- And a convention for representing remote procedure calls and responses

WSDL (Web Services Description Language)

- ✍ Standard for defining Web Services
  - Defines an abstract interface and bindings to particular message formats and protocols
  - Defines how to locate the service (URLs for HTTP)
  - Extensible, but only SOAP and HTTP extensions are defined
  - Written in XML
- ✍ Tooling support:
  - Generate WSDL from a number of Java service implementations (Java class, bean)
  - Generate client binding code from WSDL (proxy)
- ✍ Used to publish services in UDDI

WSDL Element Relationships



WSDL is an XML-based interface definition language that provides **operational information** about a service, such as the service interface, implementation details, access protocol, and contact endpoints.

What is UDDI?

Universal, Description,

Discovery, andIntegration

- A project to speed interoperability and adoption for Web services
  - Standards-based specifications for service description and discovery
  - Shared operation of a web based business registry
  - Partnership among industry and business leaders (more than 300 companies)

UDDI Registry

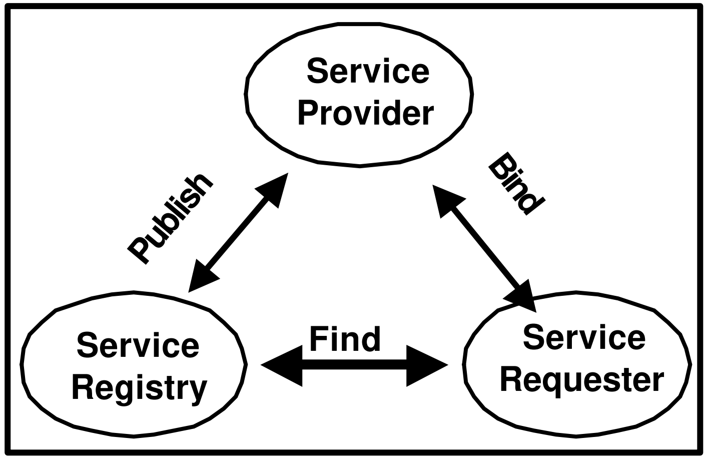
- Business registry has three components:
  - White pages
    - Information about the business (address, contacts, etc.)
  - Yellow pages
    - Categorization of the business and its services
  - Green pages
    - Technical information about services provided by a business
- Free, public, interconnected UDDI servers are deployed today
- Private UDDI Registry is available today for enterprise integration

UDDI Roles and Operations

- ✍ Service Requester
  - – FINDS required services via the Service Broker
  - – BINDS to services via Service Provider

- ✍Service Provider
  - ✍provides e-business services
  - ✍**PUBLISHES** availability of these services through a registry

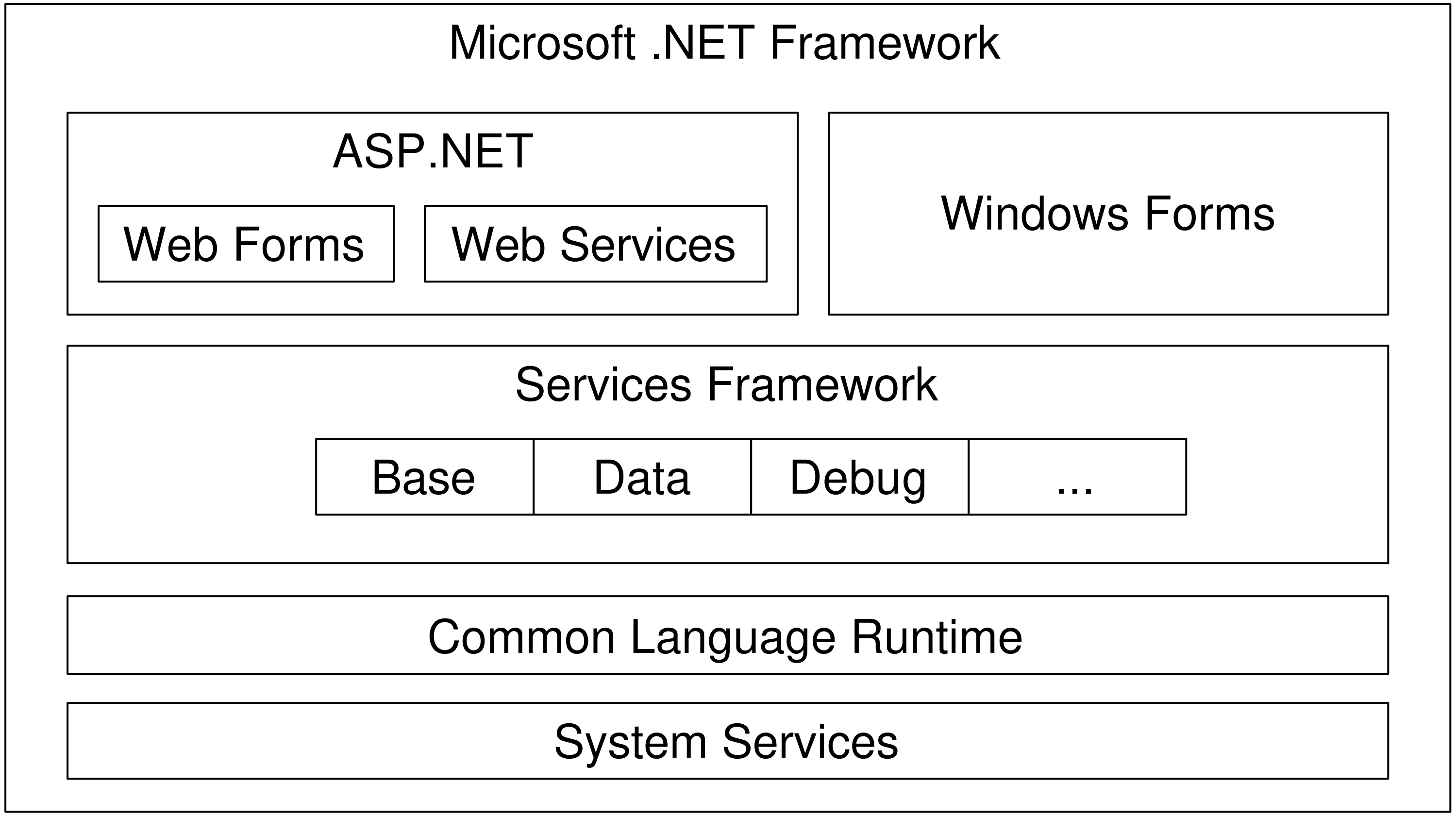
- ✍Service Registry
  - ✍provides support for publishing and locating services
  - ✍like telephone yellow pages



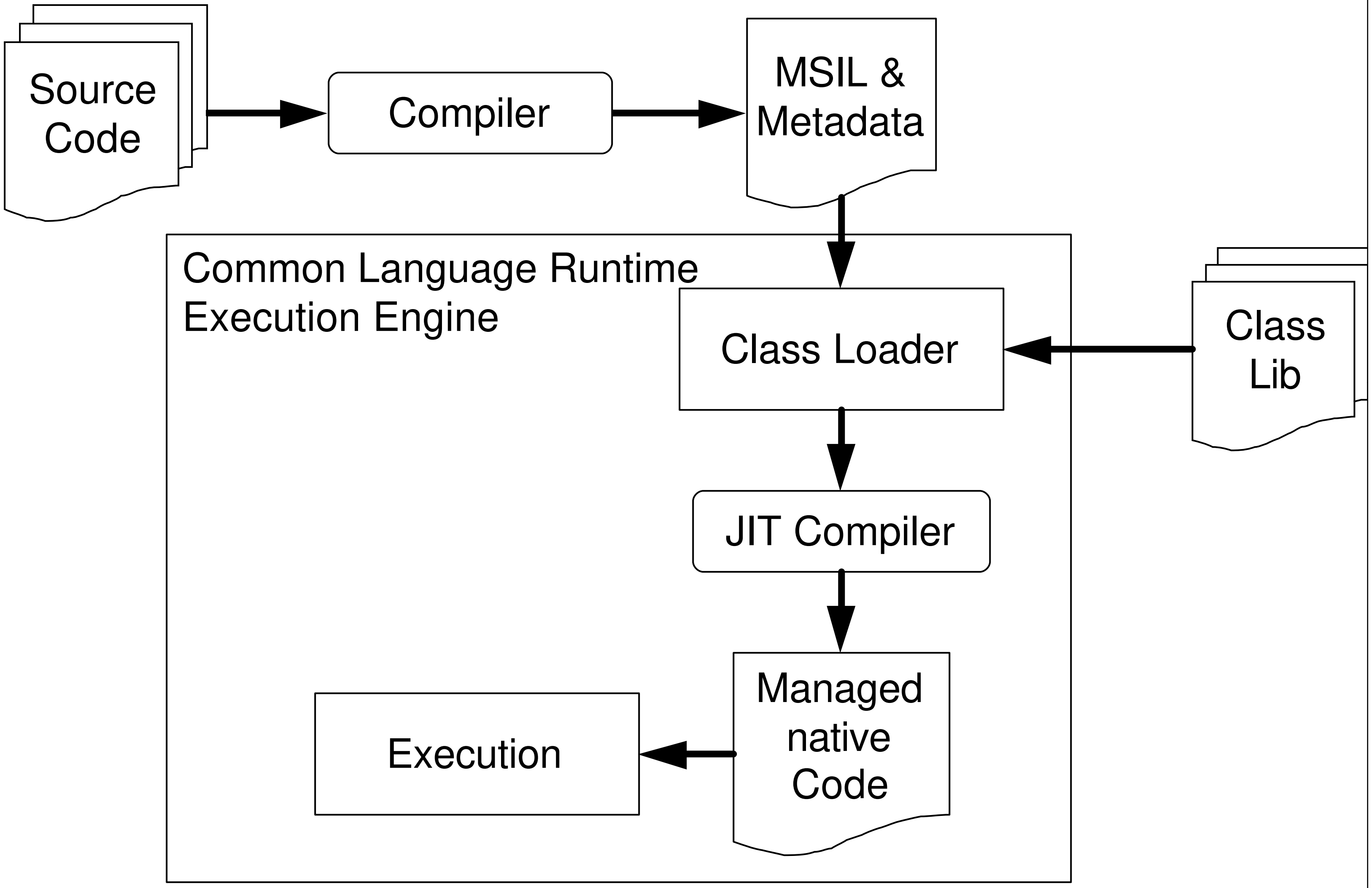
Section 2: Architecture

- The .NET Framework Architecture
- Programming Model
- Configuration

The .NET Framework Architecture

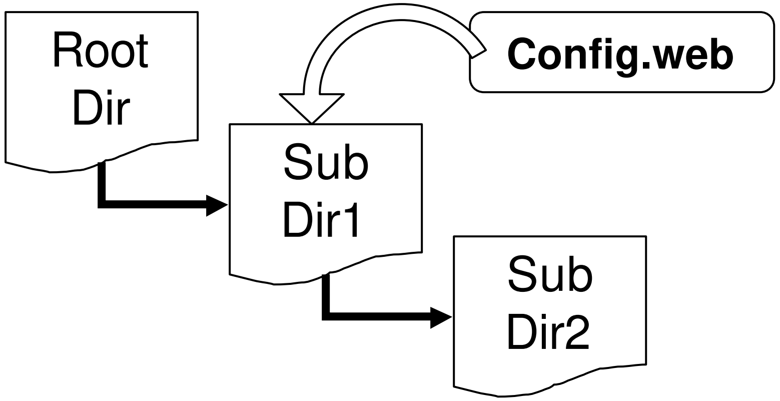


Programming model



Configuration 1/2

- Concepts and architecture
  - Config.web file
  - Hierarchical configuration architecture
    - Influence on the actual directory and all subdirectories



Configuration 2/2

- Config.web file
  - XML based
  - File is kept within the application directory
  - Default and custom configuration
    - Customized config.web file
    - Customized configuration section handler
- **WebServicesConfiguration** class
  - Contains configuration information
  - **<webservices>** section in Config.web

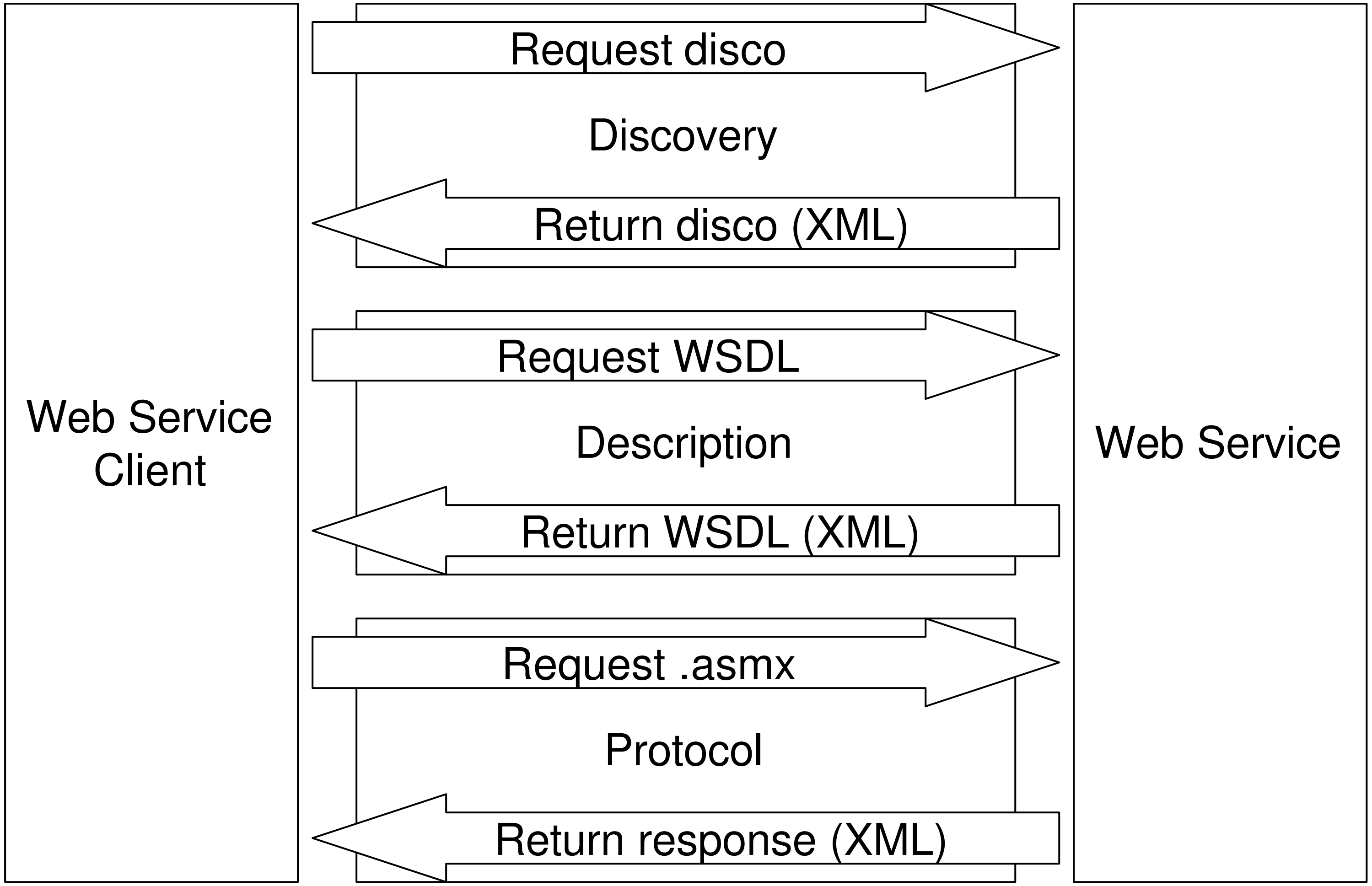
Section 3: Creating Web Services

- Basics of Creating Web Services
- Web Services Infrastructure
- Code and Syntax
- Web Services Namespace
- Publishing
- Discovery

Creating Web Services—Basics

- .asmx file
  - Virtual path of ASP.NET Web application
  - Stand-alone or part of an existing solution
- Web Services infrastructure
  - Discovery, description, and wire format
- Microsoft Visual Studio.NET
  - Microsoft Visual Basic.NET, C#, and Managed C++

Web Services Infrastructure



Web Service Client

Request disco

Discovery

Return disco (XML)

Request WSDL

Description

Return WSDL (XML)

Request .asmx

Protocol

Return response (XML)

Web Service

Code and Syntax

- **WebService** Directive
  - Settings for ASP.NET compilers

```
<%@ WebService Language=value Class=value %>
```

- **WebMethod** Attribute

- Code Declaration Syntax
  - Outline

```
<%@ WebService Class=“MyClass.MyWebService“ %>
```

  - Inline (in C#)

```
<%@ WebService Language=“C#“
               Class=“MathService“ %>
using System.Web.Services;
public class MathService : WebService {
   [ WebMethod ]
...
```

Sample .asmx file

```
<%@ WebService Language=“C#“ Class=“MathService“ %>

using System;
using System.Web.Services;

public class MathService
{
    [WebMethod]
    public int Subtract(int a, int b)
    {
        return a - b;
    }

    public int Subtract_vs(int a, int b)
    {
        return b - a;
    }
}
```

System.Web.Services Namespace 1/2

- **WebService**
  - Base class for Web Services
  - Provides base functionality
    - For example, *WebService.Session*
- **WebServiceAttribute**
  - Optional class to add additional information

System.Web.Services Namespace 2/2

- **WebServicesConfiguration**
  - Contains configuration information
  - <webservices> section in config.web
- **WebServicesConfigurationSectionHandler**
- **WebMethodAttribute**
  - **[WebMethod]**
  - Makes a method a Web Service method

Publishing a Web Service

- Expose Web Service and Web Service methods
- Create a Web Service proxy and an assembly
  - Generate proxy with **WebServiceUtil** tool
  - Create an assembly
  - Enables developers to program against Web Services
- Publish WSDL contract and HTML description
- Web Service clients
  - Can be Web applications or browsers

Discovery of Web Services

- .disco file
  - XML-based file
    - containing links to resources for retrieving WSDL
  - Stored in the server‘s root directory
  - Access via URL and dynamic discovery document
  - Start discovering with the **WebServiceUtil** tool
  - Automatically created by Visual Studio.NET

- Global directory of Web Services through UDDI
  - Universal Discovery, Description, and Integration (http://www.uddi.org)

WebServiceUtil.exe

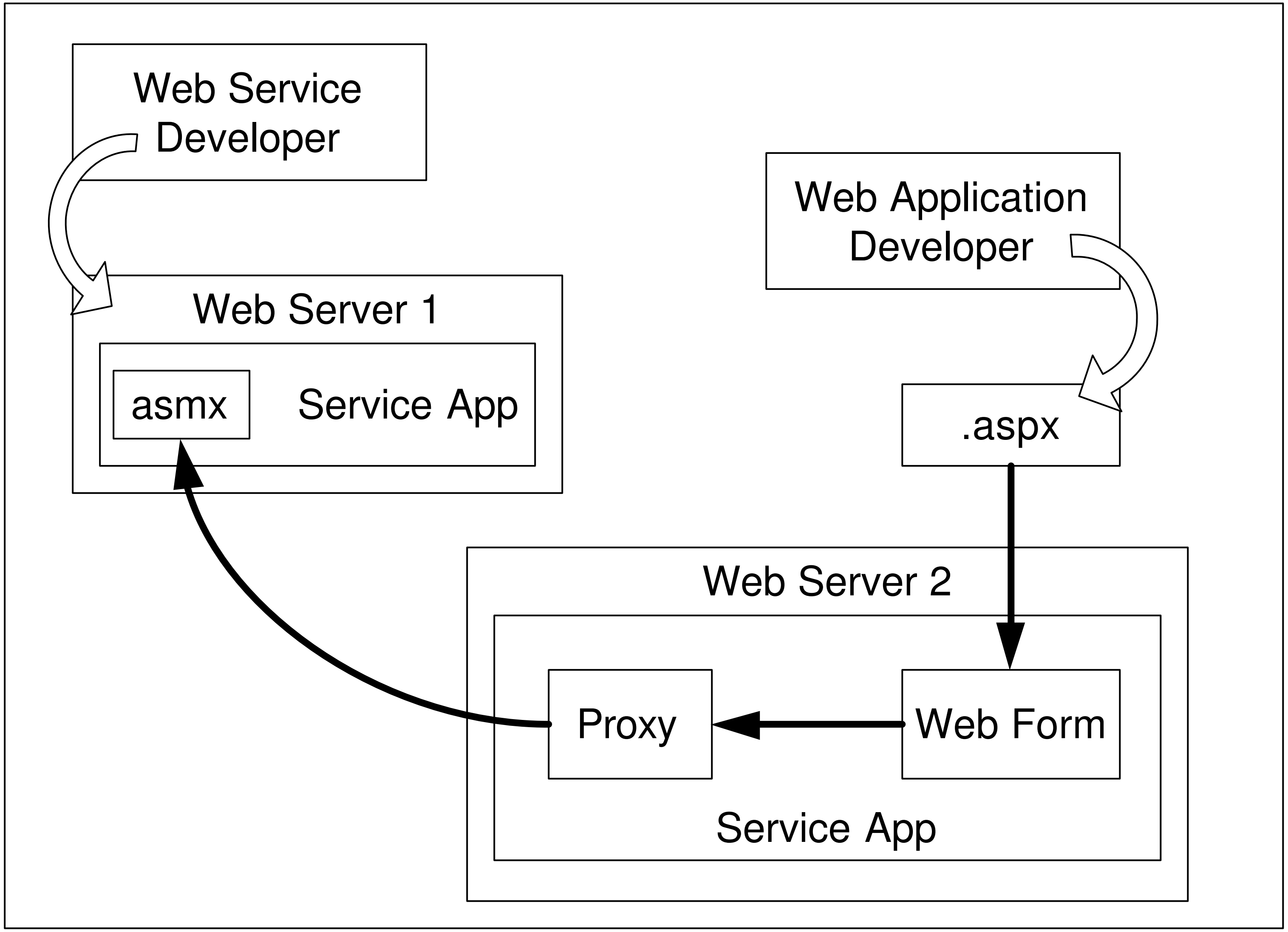
- Create client proxy class
- Input:
  - WSDL contract
  - Proxy language and protocol
- Output:
  - Single source file in specified language, containing:
    - Proxy class
    - Code for network invocation and marshalling
- Command-line example:

```
webserviceutil /c:proxy /l:C#
   /namespace:MathServiceSpace
 MathService.sdl
```

Section 4: Using Web Services

- Application Model
- Invoking Web Services
- Consuming Web Services
- Web Services Description Language (WSDL)

Application Model



Invoking Web Services

- Web Services are URL addressable
  - HTTP request
- Protocols
  - HTTP-GET
    - Method name and arguments in URL
  - HTTP-POST
    - Method name and arguments in POST body
  - HTTP-SOAP
    - XML grammar for
      - Addressing the Web Service
      - Returning results

Invoking: HTTP-GET and HTTP-POST

http://*server*/*appl*/*service*.asmx/*method*?*param*=*value*

- Standard HTTP-GET
  - Method name = PATHINFO
  - Method arguments = URL query string
    - Query string key = parameter name
    - Multiple parameters
    - Only primitive .NET runtime data types
  - Result is an XML document
    - Any .NET data type

- HTTP-POST
  - Similar to GET, but with arguments in the form body

Invoking: HTTP-SOAP

- XML grammar for
  - Web Service method, method parameters, results
- Supports all standard .NET data types and value classes
  - Additionally: classes, structs, datasets
- Class and struct marshalling
  - Serialization in XML format

Consuming Web Services

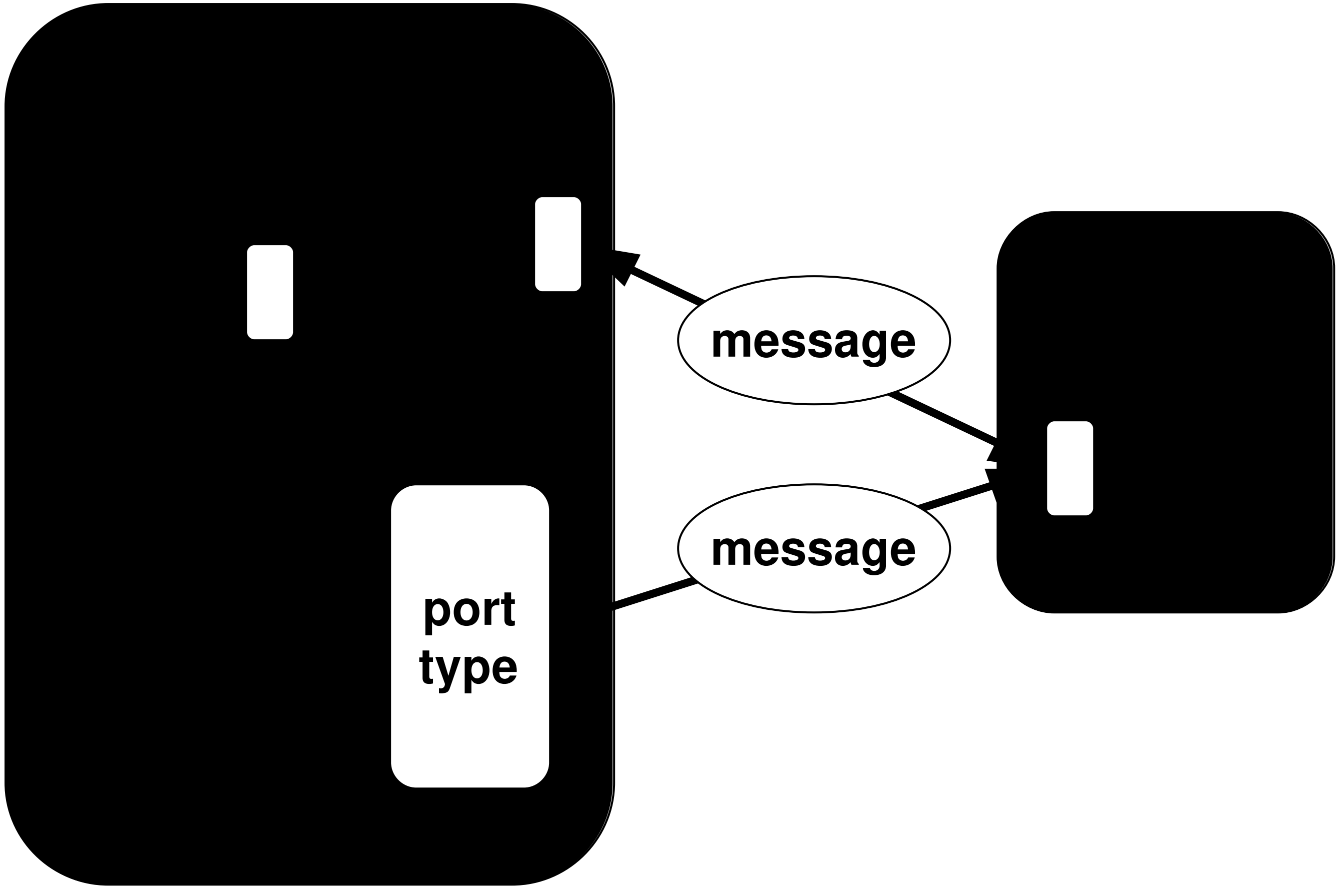
- Request without method name and parameters
  - HTML description of Web Service
  - Service capabilities, methods, protocols
- Web Service can return WSDL
  - HTTP-GET, HTTP-POST, and HTTP-SOAP
- Request with parameter “?SDL”
  - Formal WSDL description of Web Service
  - XML-based grammar
  - Can be used as input for WebServiceUtil.exe

WSDL 1/2

- XML grammar, defining:
  - Services and ports that communicate via messages
  - Binding
    - Specify a protocol or a data format for a message or a port
    - Extensions for SOAP 1.1, HTTP GET/POST, and MIME
- Public description of a Web Service and its content
  - WSDL contract
- Core Elements of WSDL
  - service, port, and portType
  - operations and messages

WSDL 2/2

- An abstract illustration of WSDL elements



Sample WSDL file

```
<definitions name=“serviceName“>
 <import namespace=“http://namespacePath“
         location=“http://path/fileName.wsdl“>

 <portType name=“portNameType“>
   <operation name=“opName“>
      <input message=“msgNameInput“ />
      <output message=“msgNameOutput“ />
   </operation>
 </portType>

 <binding>
    <soap:operation soapAction=“http://...“ />
 </binding>

 <service name=“serviceName“>
   <port name=“portName“ binding=“bindingName“>
      <soap:address location="http://..." />
   </port>
 </service>

</definitions>
```

Section 5: Advanced Web Services

- ✍ State Management
- ✍ Security
- ✍ Transactions
- ✍ Execution Model
- ✍ Distributed Web Applications

State Management

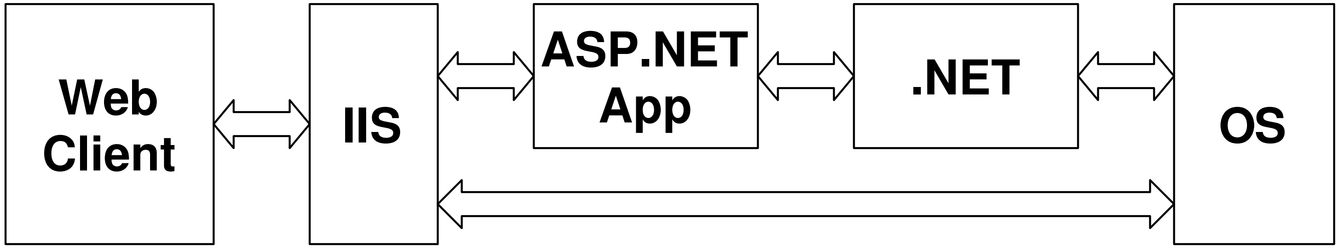
- Web Services are stateless
- Use, for example, ASP.NET session state
  - What is a session?
    - Restricted to a logical application
    - Context in which a user communicates to a server
  - Functionality
    - Request identification and classification
    - Store data across multiple requests
    - Session events
    - Release of session data
  - .NET State Server

Security Model

- Reasons for Security
  - Prevent access to areas of your Web server
  - Record and store secure relevant user data

- Security Configuration

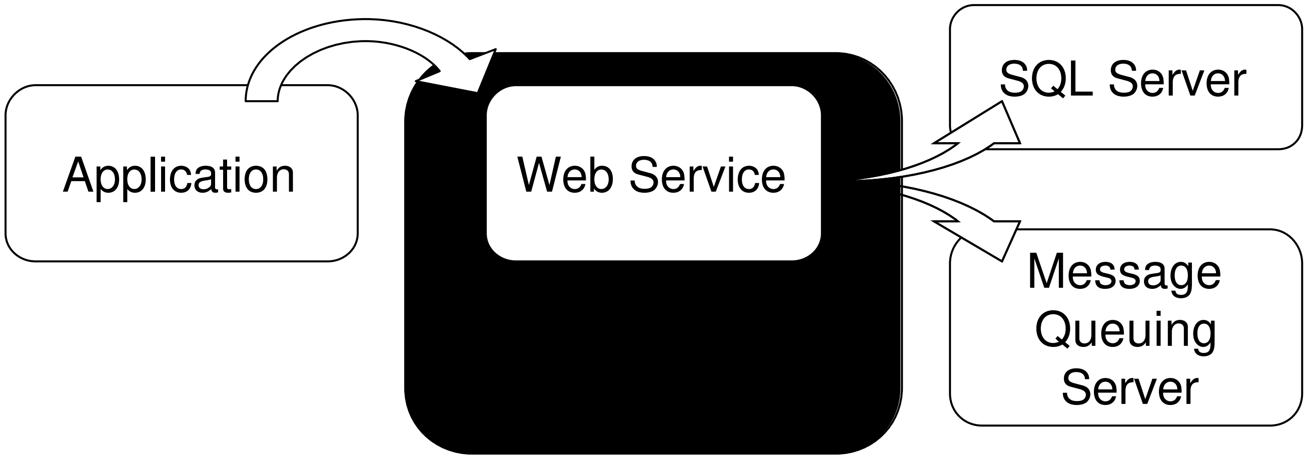
- Authentication, Authorization, Impersonation



- Code Access Security
  - Walks the call stack to check authorization

Transactions 1/2

- Like ASP.NET Web Forms
- COM+ Services
  - COM+ automatic transactions
  - Atomic, consistent, isolated, durable (ACID)



Transactions 2/2

- **TransactionMode** Property on **WebMethod** Attribute:

```
[WebMethod(TransactionMode= TransactionMode.Required)]
```

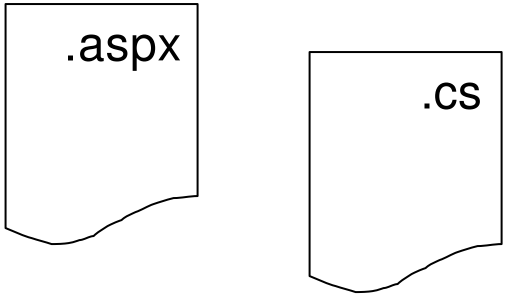
- Transaction Modes
  - Supported
  - NotSupported
  - Required
  - RequiresNew

Execution Model

- Synchronous
  - Like any other call to class methods
- Asynchronous
  - Split the method into two code blocks
    - Begin*MethodName*
    - End*MethodName*
  - Runtime determines if operation has finished

Using Web Services from UI

- Completely separate layout and processing logic
  - Two (or more) files: .aspx and .cs or .vb



- Files for designers and files for programmers
  - Easy maintainability of your application

Sample

- .aspx

```
<%@ Import Namespace=“MathServiceSpace“ %>

<script language=“C#“ runat="server">
    public void Submit_Click(Object S, EventArgs E) {
        service.Add(operand1, operand2);
    ...

</script>


...

<input OnServerClick=“Submit_Click“ runat="server"
...>
```

- .asmx file implements method “Add”
- WSDL file, returned by the ASP.NET runtime

Sample

- ✍ C# proxy class, generated by WebServiceUtil.exe

```
[System.Web.Services.Protocols.
    SoapMethodAttribute(“http://tempuri.org/Add“)]

public int Add(int a, int b)
{
    object[] results =
        this.Invoke(“Add“, new object[] {a, b});

    return (int)(results[0]);
}
```

Summary

- .NET Architecture Overview
- Web Services and ASP.NET
- Create and Publish Web Services
- Invoke and Consume Web Services
- WSDL and Proxy Classes
  - Program Against Web Services

WSE v2.0 Technical Preview

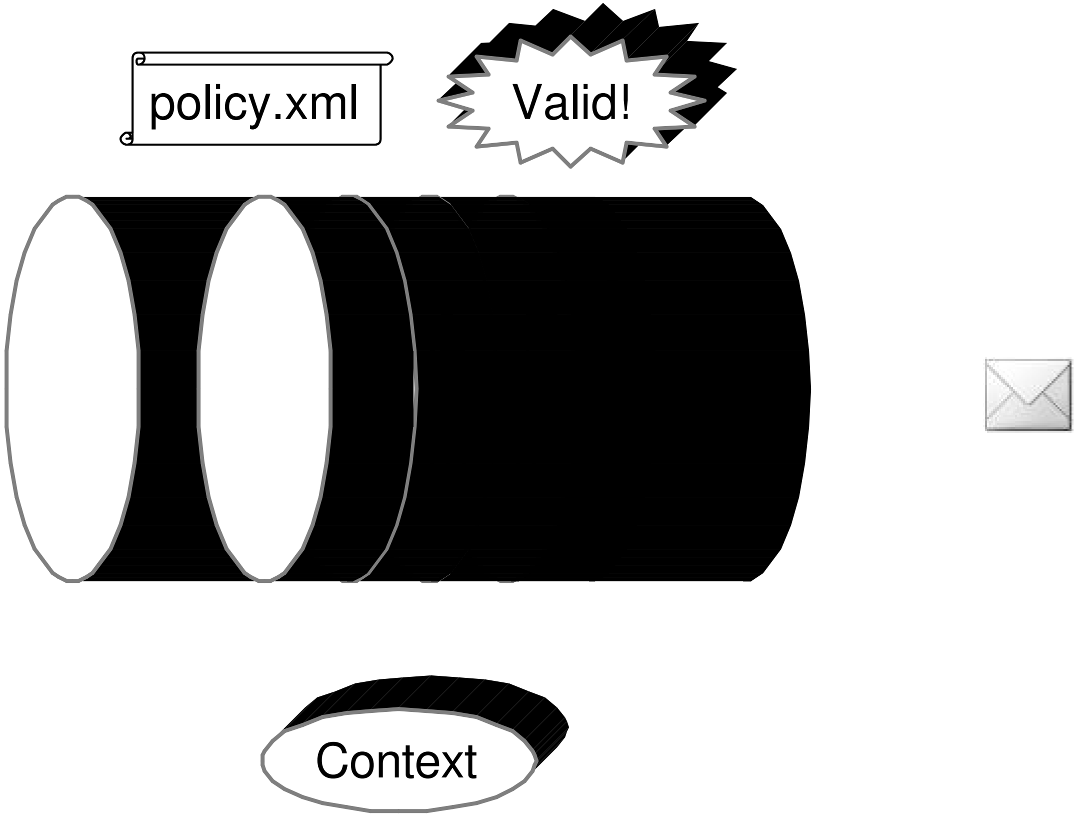
Overview

- Web Services Enhancements for Microsoft .NET 1.0 (WSE)
- Supports advanced XML Web services specifications
- Microsoft.Web.Services.dll
- WSE v1.0 released December, 2002
- WSE v2.0 TP released July, 2003

WSE Architecture

- Pipeline
  - May be hosted independently of the ASP.NET runtime
- Filters
  - Custom I/O filters
- SoapContext
  - Communication channel between application (i.e. ASMX) and infrastructure (i.e. filters)

WSE Pipeline



("Powerhouse" by Raymond Scott)

Specifications

- WS-Addressing
- WS-Attachments with Direct Internet Message Encapsulation (DIME)
- WS-SecureConversation
- WS-Security
- WS-SecurityPolicy
- WS-Policy
- WS-Referral
- WS-Routing
- WS-Trust

Addressing

- WS-Addressing
  - Network virtualization
- Endpoint references
  - Address, PortType, ReferenceProperties, ServiceName, Policy
- Message information headers
  - Address, FaultTo, From, Recipient, MessageID, RelatesTo, ReplyTo To

Addressing

```
<soap:Envelope xmlns:soap="..."
 xmlns:wsa="http://schemas.xmlsoap.org/ws/2003/03/addressing">
 <soap:Header>
  <wsa:ReplyTo>
   <wsa:Address>http://www.xyzzy.com/foobar</wsa:Address>
  </wsa:ReplyTo>
  <wsa:To>http://ws.ineta.org/SpeakerService.ashx</wsa:To>
  <wsa:Action>http://ws.ineta.org/2003/GetSpeakers</wsa:Action>
 </soap:Header>
 <soap:Body>
  ...
 </soap:Body>
</soap:Envelope>
```

Security

- WS-Security
  - Quality of protection through confidentiality and integrity
  - Defines mechanisms for associating security-related claims with a message
  - XML Encryption, XML Signature

Security Tokens

- Username/password
- x509 certificates
- Kerberos tickets
- XML-based tokens; XrML and SAML
- Security context tokens

Trust

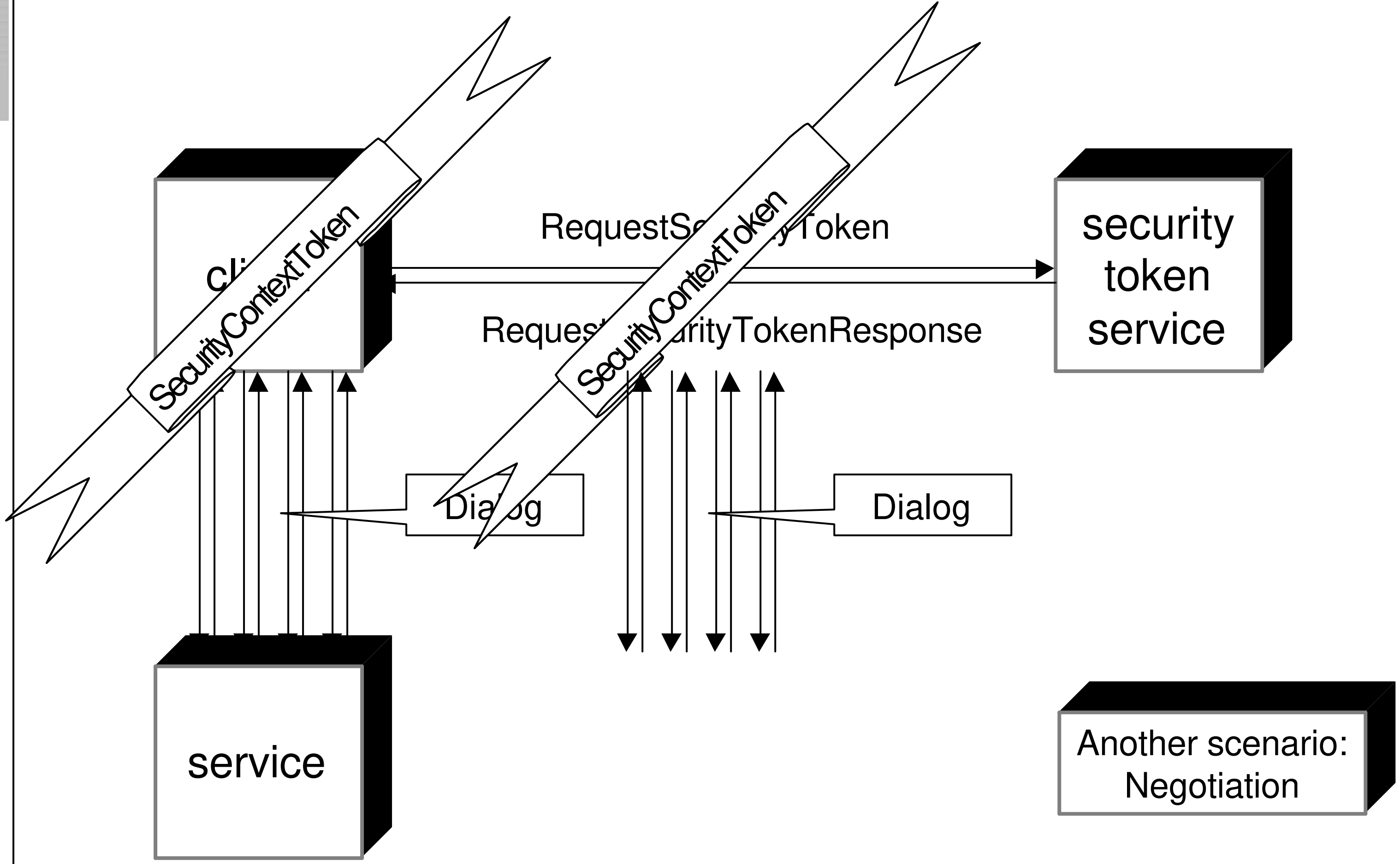
- ✍ WS-Trust
  - – Token exchange, issue, and validation
  - – RequestSecurityToken[Response]
  - – SecurityTokenService[Client]

Secure Conversation

- WS-SecureConversation
  - SecurityContextToken[Service]

```
<soap:Envelope>
 <soap:Header>
  <wsse:Security>
   <wsse:SecurityContextToken wsu:Id="Foo">
    <wsu:Identifier>uuid:...</wsu:Identifier>
   </wsse:SecurityContextToken>
   <!-- signature -->
  </wsse:Security>
 </soap:Header>
 <soap:Body>
  ...
 </soap:Body>
</soap:Envelope>
```

Secure Conversation



Policy

- WS-Policy
  - Extensible language for expressing the requirements, capabilities, and preferences of a service
  - Assertions
    - Usage: Ignored, Observed, Optional, Rejected, Required
  - Expressions
    - Operators: All, ExactlyOne, OneOrMore

Assertions

- Represent an individual preference, requirement, or capability

- WS-PolicyAssertions
  - TextEncoding, Language, SpecVersion, MessagePredicate
  - Defines XPath expressions and message part selection functions (i.e. wsp:Body())

- WSE provides an interception-based mechanism of assertion enforcement or validation

Security Policy

- WS-SecurityPolicy

Initial set includes:

- Confidentiality
- Integrity
- MessageAge
- SecurityHeader
- SecurityToken
- Visibility

Policy Mapping

```
<policyDocument>
 <mappings>
  <map to="...">
   <action name="..." policy="#Reference" />
   <default policy="#Reference" />
  </map>
  <mapDefault policy="#Reference" />
 </mappings>
 <policies>
  <wsp:Policy>
   <Integrity wsp:Usage="wsp:Required">
    ...
   </Integrity>
   <Confidentiality wsp:Usage="wsp:Required">
    ...
   </Confidentiality>
  </wsp:Policy>
 </policies>
</policyDocument>
```

Messaging

- Models
  - Dialogs, pub/sub, multicast, queues, etc.
- Communication
  - Request/Response, One Way, Dialog
- Asynchronous/synchronous
- Transfer/Transport protocols
  - HTTP/S (http[s]://) and TCP (soap.tcp://)
  - Extensible (i.e. soap.msmq://)

Miscellaneous

- SoapEnvelope (DOM)
- SoapService.GetDescription
- SecurityTokenCache
- Replay attacks (ReplayCacheManager)
  - IReplayCache
- Improved x509 certificate store support

Interoperability

- IBM Web Services Toolkit (WSTK)
- IBM WebSphere SDK for Web Services (WSDK)
- IBM Emerging Technologies Toolkit (ETTK)
- BEA WebLogic Server
- VeriSign Trust Gateway

Other Efforts…

Other Efforts…

- ✍ MDA (from OMG)
- ✍ ebXML, RosettaNet, OASIS, ...

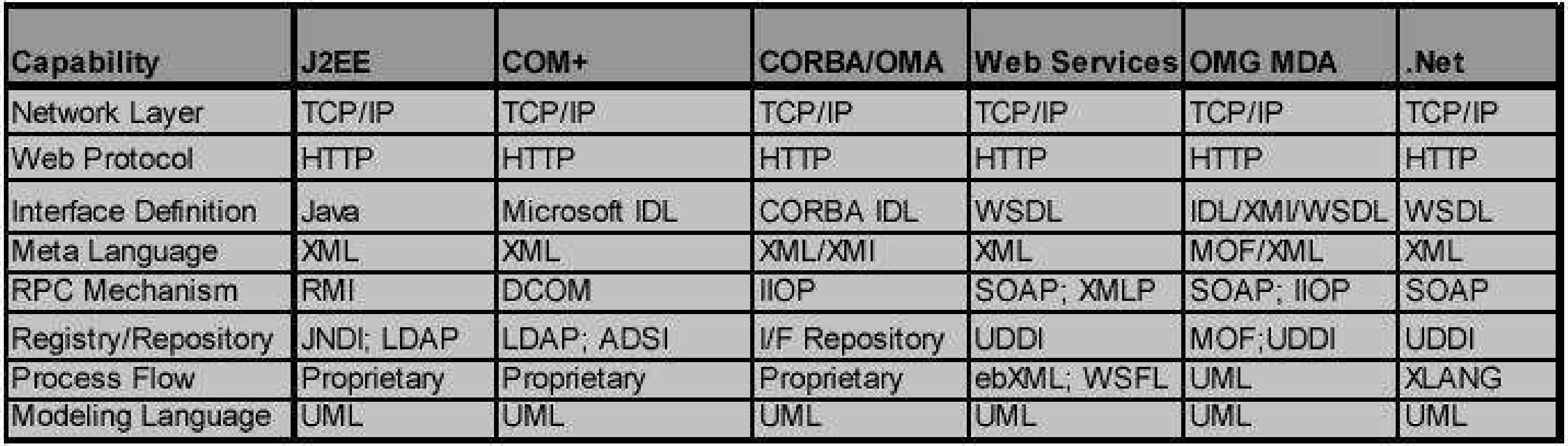
- ✍ UML Meta-models

- ✍ WS-based Workflows and Orchestration Languages
  - – **WSFL** (IBM)
  - – **Xlang** (Microsoft)
  - – **BPEL** (Microsoft + IBM)
  - – **WSCI** (Sun, BEA, Intalo, Sap),
  - – **XPDL**(WfMC)
  - – **BPSS**(ebXML)

  http://tmitwww.tm.tue.nl/research/patterns/patterns.htm

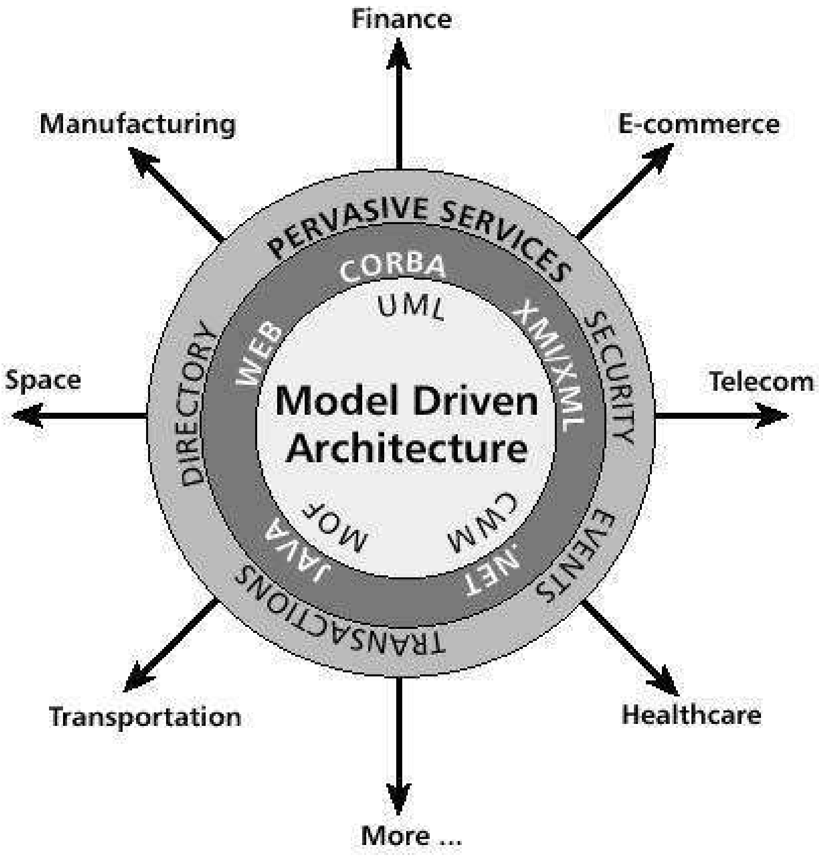
Other Efforts…

- Why Model Driven Development?



Capability	J2EE	COM+	CORBA/OMA	Web Services	OMG MDA	.Net
Network Layer	TCP/IP	TCP/IP	TCP/IP	TCP/IP	TCP/IP	TCP/IP
Web Protocol	HTTP	HTTP	HTTP	HTTP	HTTP	HTTP
Interface Definition	Java	Microsoft IDL	CORBA IDL	WSDL	IDL/XMI/WSDL	WSDL
Meta Language	XML	XML	XML/XMI	XML	MOF/XML	XML
RPC Mechanism	RMI	DCOM	IIOP	SOAP; XMLP	SOAP; IIOP	SOAP
Registry/Repository	JNDI; LDAP	LDAP; ADSI	I/F Repository	UDDI	MOF;UDDI	UDDI
Process Flow	Proprietary	Proprietary	Proprietary	ebXML; WSFL	UML	XLANG
Modeling Language	UML	UML	UML	UML	UML	UML

MDA (www.omg.org/mda)



"Design once, build it on any platform"

MDA (www.omg.org/mda)

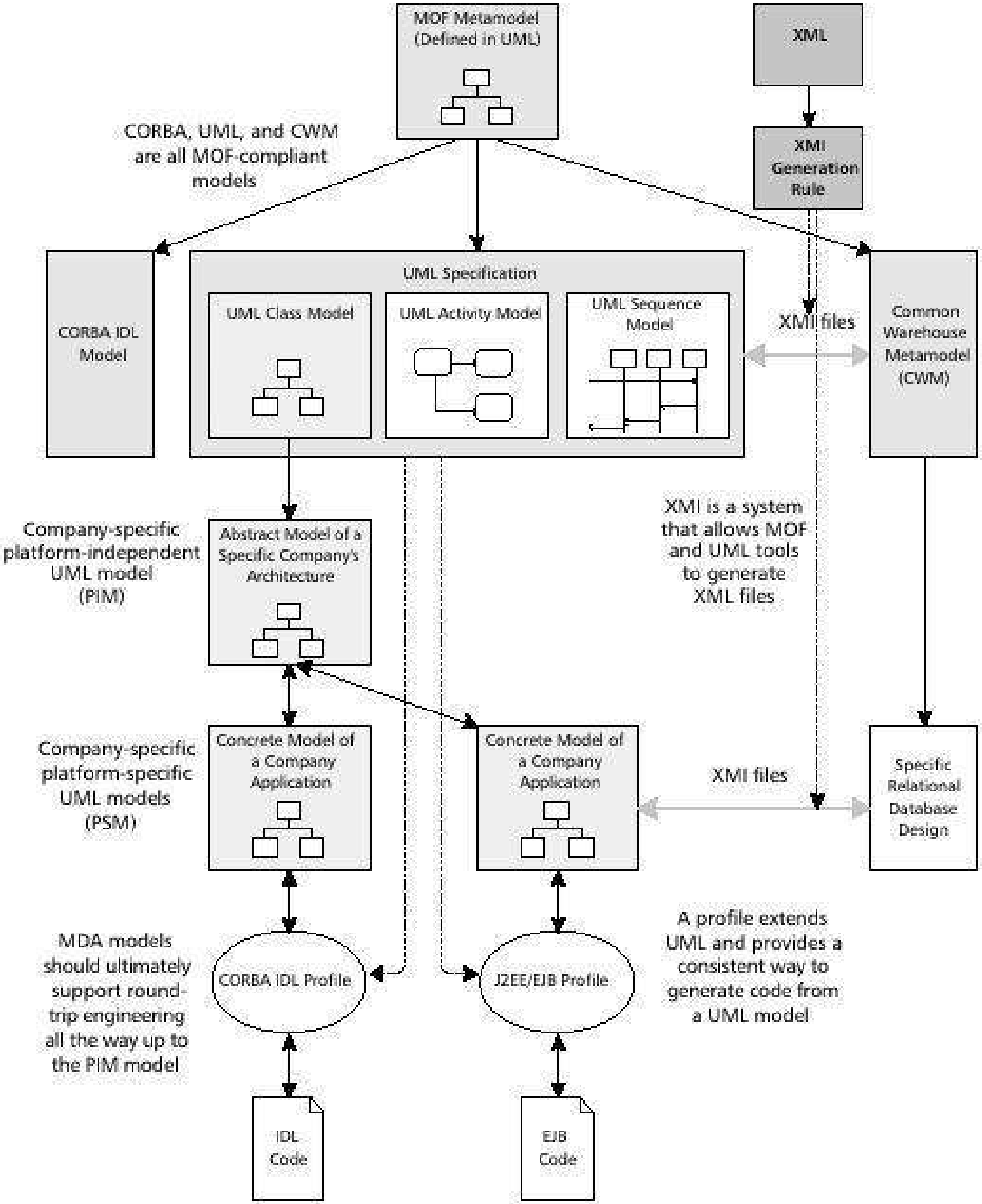
MDA’s main models

PIM...

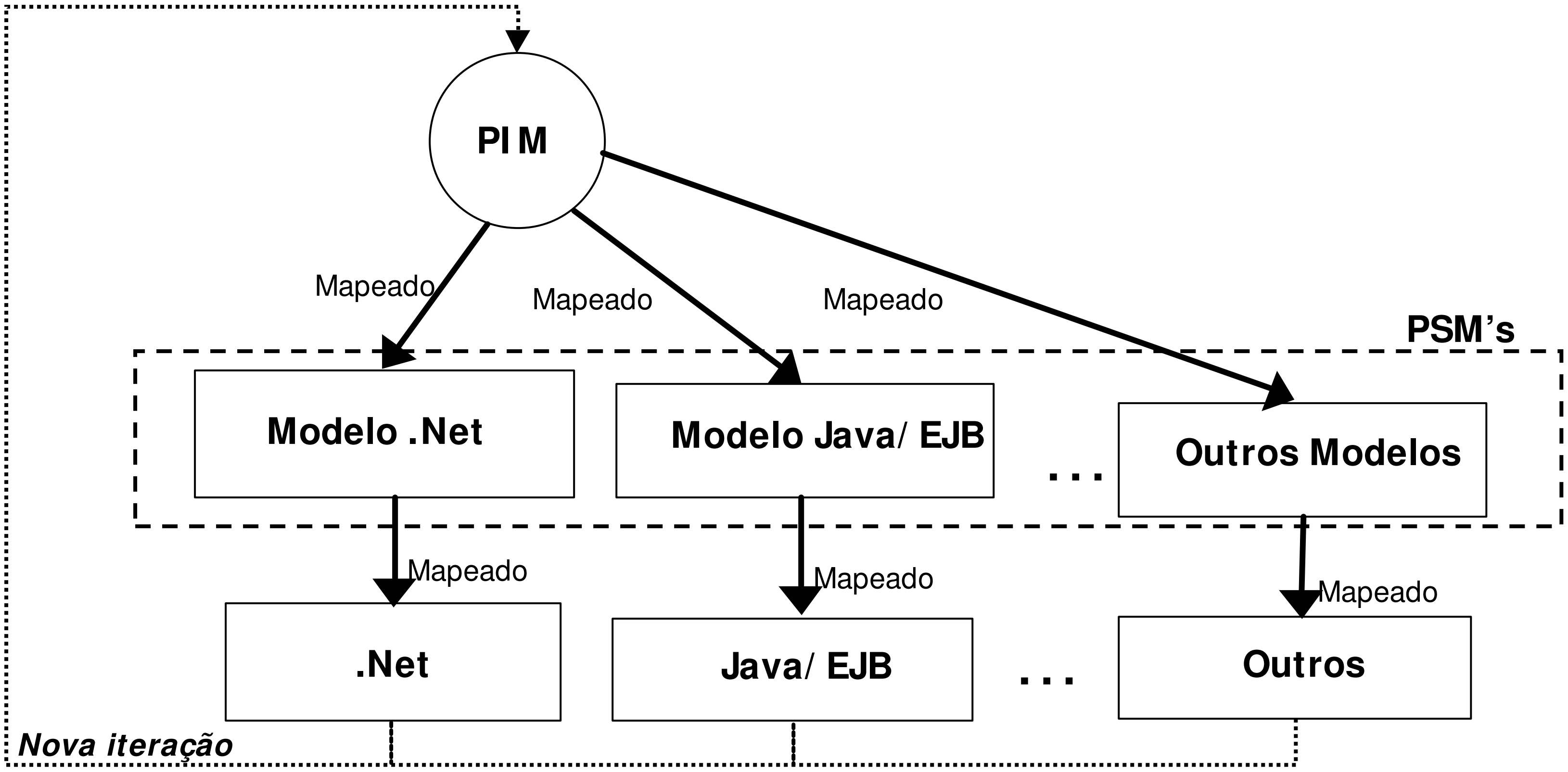
PSM...

4M’s

Models, Metadata, Mapping and Middleware of software.


MOF Metamodel (Defined in UML)
XML
XMI Generation Rule
CORBA, UML, and CWM are all MOF-compliant models
CORBA IDL Model
UML Specification
UML Class Model
UML Activity Model
UML Sequence Model
XMI files
Common Warehouse Metamodel (CWM)
Company-specific platform-independent UML model (PIM)
Abstract Model of a Specific Company's Architecture
XMI is a system that allows MOF and UML tools to generate XML files
Company-specific platform-specific UML models (PSM)
Concrete Model of a Company Application
Concrete Model of a Company Application
XMI files
Specific Relational Database Design
MDA models should ultimately support round-trip engineering all the way up to the PIM model
CORBA IDL Profile
J2EE/EJB Profile
A profile extends UML and provides a consistent way to generate code from a UML model
IDL Code
EJB Code

MDA (www.omg.org/mda)



ebXML (www.ebxml.org)

- ebXML is an electronic business standard developed by the United Nations Centre for Trade Facilitation and Electronic Business (UN/CEFACT) and the Organization for the Advancement of Structured Information Standards (OASIS). It is an adjunct to the UN's EDIFACT EDI standard.

- The ebXML Architecture involves describing abstract information and service models in UML, and defining mappings that support automatic generation of XML-based artifacts from the model.

- A UML profile called UN/CEFACT Modeling Methodology [UMM] is driving this work.

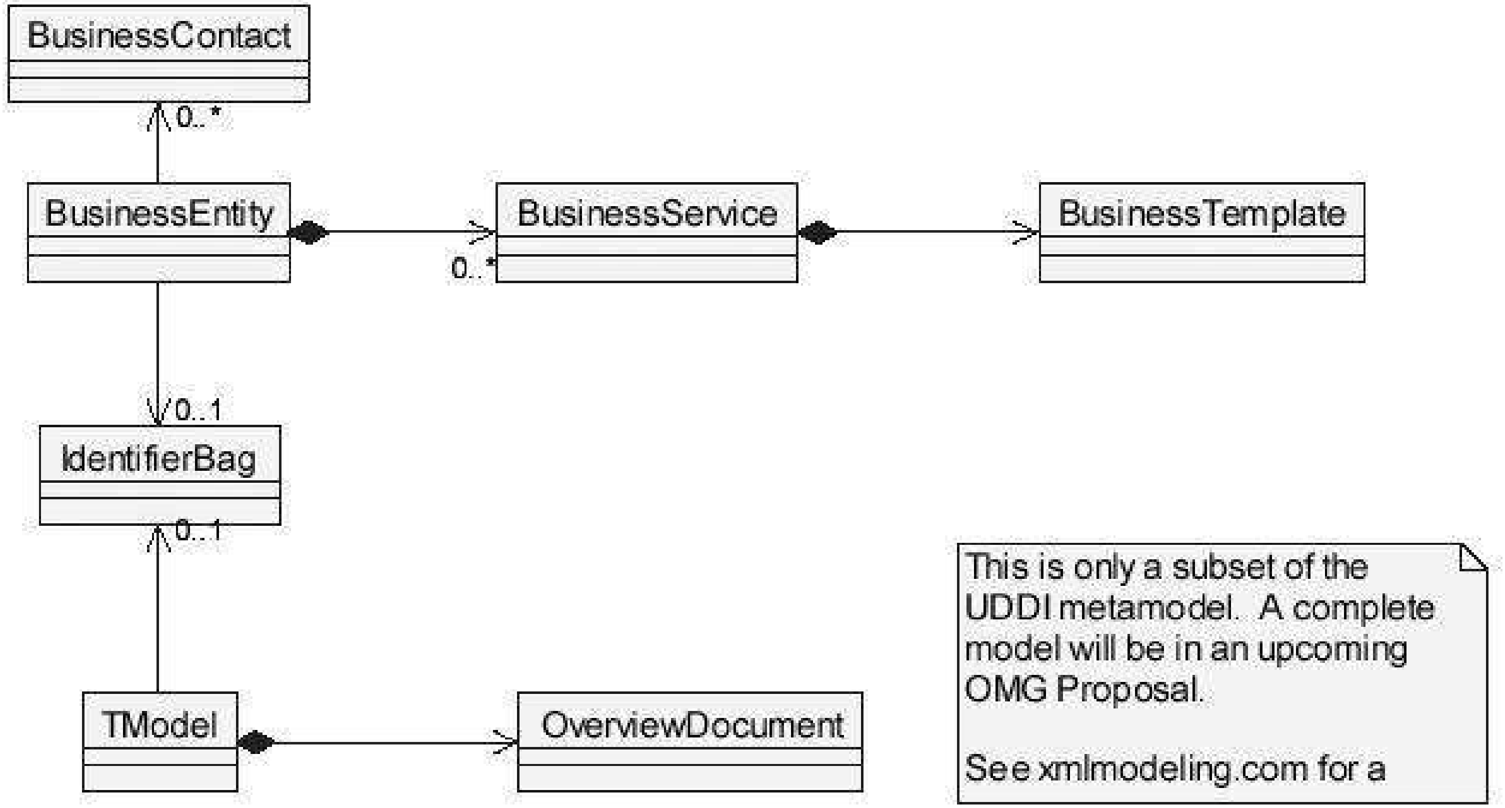
  www.ebtwg.org/projects/documentation/bioreference/

RosettaNet (www.RosettaNet.org)

- It is a consortium companies also defining standards for B2B integration.

- It is gradually moving to a UML-based approach with automatic mappings to generate XML-based artifacts.

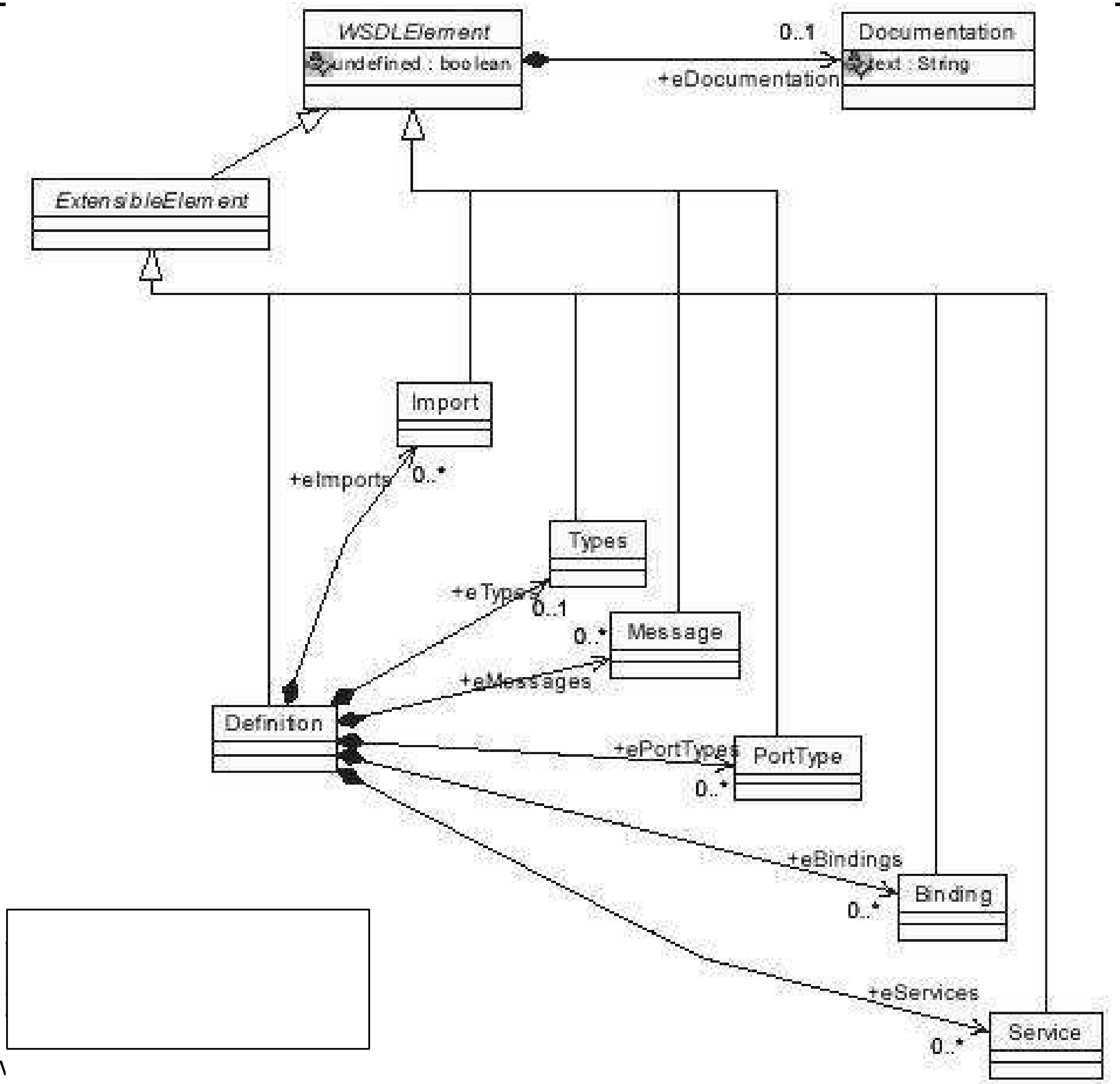
UML Metamodels, for UDDI



This is only a subset of the UDDI metamodel. A complete model will be in an upcoming OMG Proposal.

See xmlmodeling.com for a

UML Metamodels, WSDL Fragment



Resources

- Best practices/standards (www.w3.org/2002/ws/, www.openapplications.org/)
- Portal (www.webservices.org/, http://www.UDDI.org)
- Tutorials (www.w3schools.com/, java.sun.com/webservices/docs/1.0/tutorial/index.html)
- ASP.NET Forums (www.asp.net/forums)
- MSDN (msdn.com/webservices)
- Microsoft Newsgroups (news.microsoft.com)
- Weblogs (blogs.gotdotnet.com, weblogs.asp.net)
- Java Web Services Developer Pack 1.2 (java.sun.com/webservices/webservicespack.html)

Thanks!!!

- ✍ The interested reader is invited to

  contact the author [alberto.silva@acm.org] or

  visit his web site [http://berlin.inesc-id.pt/alb/]