CS 580 Client-Server Programming
Spring Semester, 2007
Doc 19 XML, SOAP
April 19, 2007

Computers changed how to produce

Books
Manuals
Forms
Presentations

How to represent a document?

Imbed commands or tags in the text

What should the commands do?

Format output

<bold> See the cat run </bold>

Structure the document

<ChapterHeader>See the cat run</ChapterHeader>

<header>Short History of Tags</header>

GenCode

- Late 1960s
- Used descriptive content tags (commands)

Generalized Markup Language (GML)

- Developed by IBM

Standard Generalized Markup Language (SGML)

- 1983 ANSI Standard
- System for developing tags for documents
- Used to create books, manuals, forms, etc
- Widely used by IBM, IRS, DOD etc.
- Not well known in computer industry

HTML

Markup language for WWW

Wide spread use

Fixed set of tags

Some tags are presentational

<CENTER> <B>

Web Browsers permit poorly formed HTML

```
<A NAME="WhichOne"></a>
<b><center>Hello World</b></CENTER><Br>
<A NAME="WhichOne"></A>
```

These problems with HTML restrict Web functionality

XML

XML creators wanted
- Flexibility of SGML
- Simplicity of HTML

Key differences from HTML
- Presentation is separate from document description
- Error Checking
- Unambiguous Structure

XML is about

Document structure

Describing data

```
<?xml version="1.0" ?>
<CATALOG>
        <CD>
                <TITLE>Empire Burlesque</TITLE>
                <ARTIST>Bob Dylan</ARTIST>
                <COUNTRY>USA</COUNTRY>
                <COMPANY>Columbia</COMPANY>
                <PRICE>10.90</PRICE>
                <YEAR>1985</YEAR>
        </CD>
        <CD>
                <TITLE>Hide your heart</TITLE>
                <ARTIST>Bonnie Tyler</ARTIST>
                <COUNTRY>UK</COUNTRY>
                <COMPANY>CBS Records</COMPANY>
                <PRICE>9.90</PRICE>
                <YEAR>1988</YEAR>
        </CD>
</CATALOG>
```

Parts of The XML Universe

Basic Syntax

XML 1.0 spec, XML 1.1

XLinks

Namespaces

Markup Languages

XHTML

MathML

SMIL

VoiceXML

Data Addressing & Query

XPath

XPointer

XML Query Language (XQL)

Document Modeling

Document Type Definitions (DTDs)

XML Scheme

Presentation and Transformation

XML Stylesheet Language (XSL)

XSL Transformation Language (XSLT)

Cascading Style Sheets (CSS)

Extensible Stylesheet Language for Formatting Objects (XSL-FO)

Levels of XML

Well-formed

XML document that satisfies basic XML structure

Valid

XML document that is well-formed and

Specifies which tags are legal

A Document Type Definition (DTD) is use to specify

- Legal tags
- Correct tag nesting

Well-Formed XML Documents

Optional Prolog
Root Element

```
<?xml version="1.0" ?>
<!-- A comment -->
<greetings>
      Hello World!
</greetings>
```

```
<?xml  version="1.0"  encoding='iso-8859-1' standalone=no ?>
<!-- A comment -->
<facts>
      <statement>We know 5 &lt; 10</statement>
      <statement><![CDATA[We know 5 <10]]></statement>
</facts>
```

Valid XML Documents

XML document that is well-formed and

Specifies which tags are legal

A Document Type Definition (DTD) is use to specify

- Legal tags
- Correct tag nesting

```
<?xml version="1.0" ?>
<!DOCTYPE greetings [
    <!ELEMENT    greetings   ( from, to, message, date?)>
    <!ELEMENT    from        ( name )>
    <!ELEMENT    to          ( name )>
    <!ELEMENT    message     ( #PCDATA )>
    <!ELEMENT    date        ( #PCDATA )>
    <!ELEMENT    name        ( #PCDATA )>
]>
<greetings>
    <from><name>Roger</name></from>
    <to><name>World</name></to>
    <message>Hi</message>
</greetings>
```

XML Namespaces

An XML namespace is a group of Elements & Attributes

XML namespaces allows mixing of elements from different DTDs

```
<?xml version="1.0" ?>
<whitney:greetings  xmlns:whitney=”http://www.eli.sdsu”
                    xmlns:godot=”http://www.waiting.com”>
    <whitney:from>
            <godot:firstname>Roger</godot:firstName>
    </whitney:from>
    <whitney:to>
            <godot:firstname>John</godot:firstName>
    </whitney:to>
    <whitney:message>Hi</whitney:message>
</whitney:greetings>
```

Schemas

A way to define XML documents

Allow element content to have a type
Allow element content to be restricted

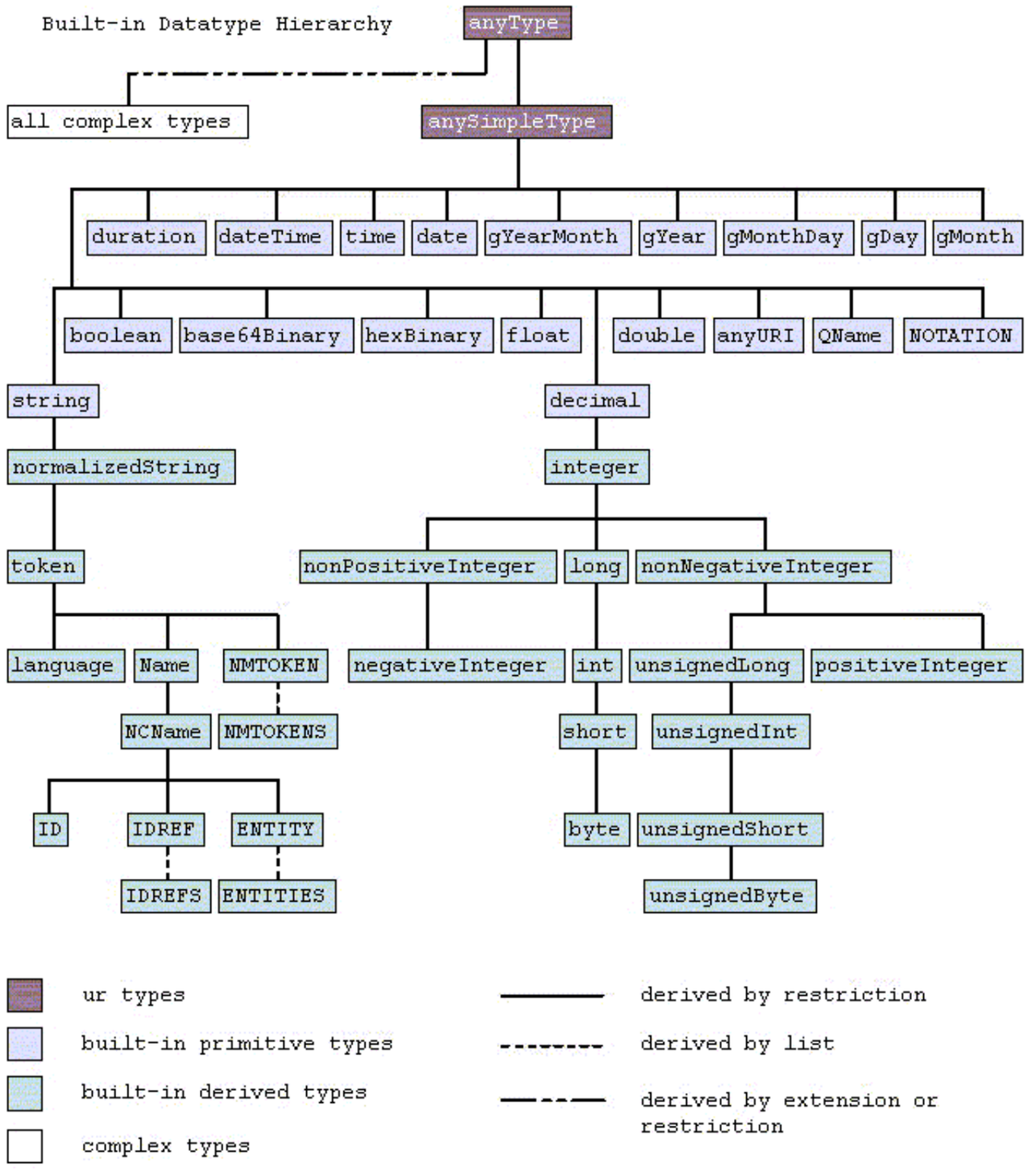
Sample Schema parts

```
<datatype name="Price">
    <scalar datatype="float" decimals="2"/>
</datatype>

<datatype name="MovieTicketPrice">
    <scalar datatype="Price" digits="1"
        maxvalue="9.50" minvalue="1.50"/>
</datatype>

<elementtype name="MovieTicket">
    <model>
        <element name="MovieTitle" type="string"/>
        <element name="TicketPrice" type="MovieTicketPrice">
    </model>
</elementtype>
```

XML Using Schema

```
<MovieTicket>
    <MovieTitle>Gone With the Wind</MovieTitle>
    <TicketPrice>6.50</TicketPrice>
</MovieTicket>
```

Basic Types in XML Schema



http://www.w3.org/TR/xmlschema-2/

Web Services

SOAP – Simple Object Access Protocol

WSDL – Web Services Description Language

UUDI – Universal Description, Discovery and Integration of Web Services

UDDI

Registry for businesses worldwide to list themselves on the Internet

UDDI business registration consists of:

- White Pages — address, contact, and known identifiers;
- Yellow Pages — industrial categorizations based on standard taxonomies;
- Green Pages — technical information about services exposed by the business.

2005 - 70% of Fortune 500 companies plan to use UDDI

2006, January

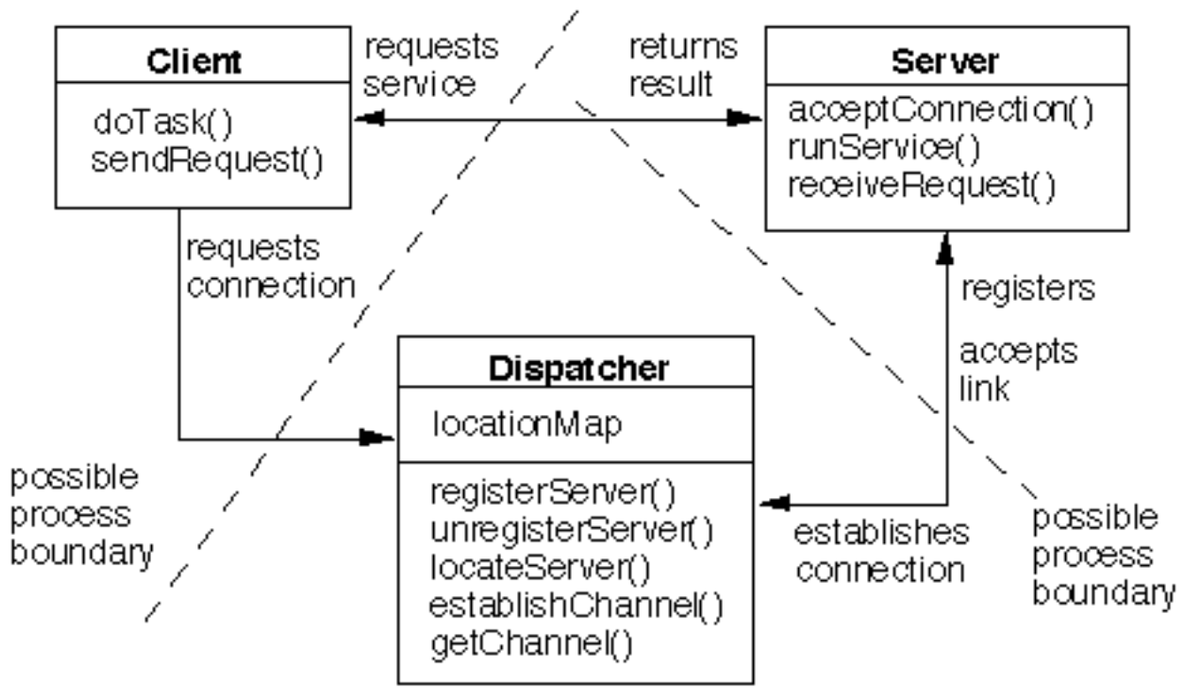
IBM, Microsoft shut down root UDDI servers

http://en.wikipedia.org/wiki/UDDI

UDDI Reborn

Plays the role of Dispatcher

Not clear how wide spread use is



WSDL

Description of how to interact with a Soap Server

Tools exist to

- Generate WSDL from a Server class
- Generate Server or client stub classes from WSDL

Sample Server

```
public class HelloServer
{
	public String hello(String aName)
		{
		return  "Hello to: " + aName;
		}
}
```

WSDL - Namespaces & Service

```
<definitions
    targetNamespace="urn:http://www.eli.sdsu.edu/cs580"
    xmlns:tns="urn:http://www.eli.sdsu.edu/cs580"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
    xmlns="http://schemas.xmlsoap.org/wsdl/">

<service name="SoapServer">
    <port name="HelloWorld" binding="tns:HelloWorld">
        <documentation>A simple Soap Example</documentation>
        <soap:address location="http://localhost:4920/HelloWorld"/>
    </port>
</service>
```

WSDL - binding

```
<binding name="HelloWorld" type="tns:HelloWorld">
    <soap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/>
    <operation name="Hello">
        <soap:operation soapAction="urn:http://www.eli.sdsu.edu/cs580#Hello" style="rpc"/>
        <input>
            <soap:body use="literal" namespace="urn:http://www.eli.sdsu.edu/cs580"/>
        </input>
        <output>
            <soap:body use="literal" namespace="urn:http://www.eli.sdsu.edu/cs580"/>
        </output>
    </operation>
</binding>
```

WSDL - PortType

```
<portType name="HelloWorld">
    <operation name="Hello" parameterOrder="yourName">
        <documentation>Returns a greeting Input: your name</
documentation>
        <input message="tns:HelloSoapIn"/>
        <output message="tns:HelloSoapOut"/>
    </operation>
</portType>
```

WSDL - Message signatures

```
<message name="HelloSoapIn">
      <part name="yourName" type="xsd:string"/>
</message>
<message name="HelloSoapOut">
      <part name="return" type="xsd:string"/>
</message>
```

SOAP

Exchanging XML messages over computer network

Message Exchange Patterns
- RPC - Remote procedure call
- Document - one way message

Transport
- HTTP, HTTPS, SMTP

Data types Supported
- All base Schema types
- Struct
- Array

Struct Example

Book as Struct

```
<e:Book>
   <author>Henry Ford</author>
   <preface>Prefatory text</preface>
   <intro>This is a book.</intro>
</e:Book>
```

Book Class

```
public class Book {
      String author;
      String preface;
      String intro;
```

Schema Defining Book

```
<element name="Book">
<complexType>
  <element name="author" type="xsd:string"/>
  <element name="preface" type="xsd:string"/>
   <element name="intro" type="xsd:string"/>
</complexType>
</e:Book>
```

Soap structs are used to send objects in messages

http://www.w3.org/TR/2000/NOTE-SOAP-20000508/

Sample Java Client

```
import org.apache.soap.rpc.*;
import org.apache.soap.Constants;

import java.util.Vector;
import java.net.URL;

class HelloSoapClient {
    public static void main(String[] args) throws Exception {
        Call call = new Call();
        call.setTargetObjectURI("urn:http://www.sdsu.edu/cs580");
        call.setMethodName("Hello");
        call.setEncodingStyleURI(Constants.NS_URI_SOAP_ENC);

        //creating a parameter list
        Vector params = new Vector();
        params.addElement(new Parameter("yourName", String.class, "Roger", null));
        //adding the parameter(s) to the Call object
        call.setParams(params);

        //invoke the soap method
        Response res = call.invoke(new URL("http://localhost:4920/HelloWorld"), "");
        System.out.println(res);
    }
}
```

Sample Ruby Client

```
require 'soap/wsdlDriver'

proxy = SOAP::WSDLDriverFactory.new("http://www.eli.sdsu.edu/courses/spring07/cs580/Hello.wsdl").createDriver
puts proxy.Hello('Roger')
```

The Examples don't work

Soap is complex

Soap has problems working between venders

Some Performance

	Connect time	Send String 21,000 Chars	Send 5,000 integers	Server LOC	Message size sending 100 integers
socket	0.002242	0.001377	6.71	25	85,863
Corba	0.000734	0.004601	1.52	18	27,181
XML-RPC	0.007040	0.082755	100.34	17	324,989
SOAP	0.000610	0.294198	1,324.30	10	380,288

Time in seconds

Code written in Python

http://www-128.ibm.com/developerworks/webservices/library/ws-pyth9/