Outline

• Introduction
  – what are web services?
  – why you should care?

• Course organization, policies and guidelines
  – topics
  – workload and grading
  – collaboration policy

• Assessment of student background

What are Web Services?

• Depends on whom you ask …
  – A revolutionary new way of building distributed applications
  – The natural evolution of distributed programming APIs
    • Sockets → RPC → Distributed Objects → Web Services
    • To simplify application integration and interoperability

• Main ideas
  – Applications structured as lightweight components, which expose services
    • Example: A Weather component, which offers a GetTemperature service
      – Input parameter: Zip code
      – Output response: An integer that represents the temperature
  – Services discovered, described, and interacted with using standard protocols
    • UDDI, WSDL, SOAP, all of which make heavy use of XML

• Goal: Provide a simple application-to-application interface just like the
  web has provided a simple human-to-application interface
  – Specifications such as HTML and HTTP, servers and browsers

An Example Distributed Application – NYUHome

HTTP
GET, POST

NYUHome
server

Albert
Mail
Bobst

Horoscopes
News
Movies

webscopes.com
moreover.com
INSPEC
ERIN...

HTML
Building NYUHome Using Web Services

Potential benefits:
- **Easier integration of services**
  - Standard as opposed to per-service protocols
  - Higher-level programming of interaction
    - Assembly vs. HL programming
- **Extensibility**
- **Flexibility for improved user experience**
  - Custom portals
  - Filtering
  - Caching
  - Localization
  - …

HTTP
GET, POST
HTML
WML
NYUHome
server
Albert
Mail
Bobst
Horoscopes
News, Movies
webscopes.com
moreover.com
INSPEC
ERIN …

Why Study Web Services?

- Most new applications have a distributed component
  - Accessible across wide-area networks
  - Themselves need to access other services across networks
- Need to understand what is involved in writing distributed programs
  - Both from the point of view of what is in use/proposed earlier …
    - Sockets, Remote Procedure Calls, Distributed Objects
    - Standard protocols, data formats such as HTTP and XML
- … As well as how these approaches have evolved
  - Web services architecture
    - Protocols such as SOAP, WSDL, UDDI, …
    - Easier to construct, more interoperable
    - Are we giving up anything …

What This Course is About

- Understanding the **general issues** that must be addressed while constructing **distributed applications** from component pieces
  - **Discovery**: How do components learn about each other?  **UDDI**
  - **Description**: How does a component learn about another’s interface?  **WSDL**
    - What services does it provide?
    - **Interaction**: How are these services invoked?  **SOAP**
      - How are messages transferred from one component to another?  **HTTP, TCP**
      - How do we encode service parameters, return values in these messages?  **XML**
      - How do the components at either end know which encoding to use?
  - Understanding how different approaches deal with these issues
    - Sockets, Remote Procedure Calls, Distributed Objects, Web Services
  - Getting some practical experience in using these approaches
    - C# language, .NET framework class libraries, Visual Studio.NET IDE (tools)

What This Course is **Not About**

- Learning how to program in C#
- Learning how to use the Visual Studio.NET environment
  - You will need to learn whatever is required on your own
- Learning everything there is to know about the .NET framework
  - The .NET framework is huge
  - We will be covering only a very small part
  - We will be focusing on **general concepts** (beyond the .NET framework), not all of the ways in which you can do ‘X’ in the .NET framework
- Understanding how to build **scalable** and **secure** web services
- Understanding the implications of choices made in the web services architecture (e.g., XML, SOAP, UDDI, …)
  - This is an introductory course, so we can only cover so much
  - Coverage of these topics in Professor Grimm’s graduate course
Microsoft’s .NET Framework

- C#
- VB.NET
- C++.NET
- Other

Common Language Specification

Framework Class Library
- ASP.NET
- Web Services
- ADO.NET
- XML
- Network

- Windows Forms
- Controls
- Drawing

- Threading
- IO

- Security
- Diagnostics
- Remoting

Common Language Runtime
- Memory Management
- Common Type System
- Lifecycle Monitoring

Operating System

Visual Studio .NET

Prerequisites

- Official
  - V22.0202: Computer Systems Organization II

- Would prefer
  - you to have taken the Honors section of V22.0201
  - you to have experience with
    - programming in Java and/or C++
    - working in a Windows environment
      - Logging in, creating folders, launching applications …
      - Both with the GUI and using command-line tools

- Will expect you to pick up required expertise on your own
  - Two tutorial lectures on using the Visual Studio.NET environment
  - Suggest learning things as required (too complicated otherwise)
  - Course web page has links to online resources

Tentative Course Schedule

Lecture 1: Introduction, Course Organization
Lecture 2: Tutorial: Visual Studio.NET familiarization
Lectures 3-11: Precursors/Alternatives to Web Services
  Networking fundamentals, Sockets, RPC, Distributed Objects
Lecture 12: Web Services Architecture – Overview
Midterm Exam: October 14, 2003
Lecture 13: Tutorial: Building/Deploying/Using Web Services
Lectures 14-21: Web Services Architecture – Details
  SOAP, WSDL, UDDI, State Management, Intermediaries, GXA
Lectures 21-26: Course Project: Design and Review Meetings
  Project Demos

Workload and Grading

- First time I am teaching this course
  - Will not be as well-organized as the core courses

- Workload
  - Classes
    - Five C#/.NET programming projects: 50%
    - each due 12-15 days after it is handed out
    - BSD Sockets, XML-RPC, .NET-Remoting, Web Services (2)
  - Midterm exam 20%
  - Course project 30%
    - A larger web-services based application

- Grading will be on the curve
  - Relative to overall class performance
  - I will use subjective factors to decide whether to push those on the border to the next higher grade
Course Project

- A mock-up of the NYUHome service
- Done in groups of 12-15 students each
  - 4-5 subgroups of 3 students each
    • Everybody in a subgroup will get the same grade for the project
- Three aspects
  - Designing/deploying component web services (for each channel)
  - Integrating client components of these services into one or more portals
  - Suggesting/implementing innovative uses of web service intermediaries
    • Customizing portal behavior for classes of users
    • Improving performance
    • Localizing information by connecting to local services …
- No lectures after November 13th: Project design and review meetings
  - Demos in the last week of classes
- Additional details will be provided as we progress in the semester

Computing Resources

- I am providing: Three development servers
  - netserver{1,2,3}.pdsg.cs.nyu.edu: one dual-Xeon, two Pentium 4’s
  - Windows Server 2003 (Enterprise Edition)
    • Application Server (IIS w/ ASP.NET) for deploying, accessing web services
    • Terminal Server for logging in using Remote Desktop clients
      - You will get a Windows desktop as if you were logging in into your own machine
  - Visual Studio.NET 2003 (Professional)
    • C# compiler, .NET libraries, and miscellaneous utilities
    • Documentation
    • Cygwin tools, Emacs-NT
- You will need: Remote desktop clients (supporting 128-bit RC4)
  - Windows: pre-installed on XP, downloadable for rest
  - UNIX: source code available from www.rdesktop.org
  - Macs: available for OS X 10.2.3 and higher
  - Am assuming that each of you will have access to at least one of the above

Course Resources

- Textbook(s)
  - No required text: Could not find any with appropriate coverage of material
  - You may find it useful to have books for
    • Programming in C#
    • Using the .NET Framework’s Class Libraries
    • Working with the Visual Studio.NET IDE
  - None of these are required: Dynamic help feature in VS.NET
  - Pointers to online resources
- Course web page:
  - http://www.cs.nyu.edu/courses/fall03/V22.0480-004/index.htm
- Class mailing list: v22_0480_004_fa03@cs.nyu.edu
  - send questions of general interest here
- E-mail: vijayk@cs.nyu.edu, TAs when assigned

Comparison w/ Professor Grimm’s Course

- G22.3033-008: Web Services and Applications
  - Targets a graduate audience
  - More research-oriented: technical papers + discussion
  - Performance, security, scalability issues for large-scale distributed applications
  - Assumes familiarity/experience with network programming
  - Java-based programming assignments
    • Exposes all of the underlying mechanisms/machinery
- This course
  - Targets an undergraduate audience
  - Assumes no prior exposure to distributed programming
  - No discussion of performance, security, scalability issues
  - C#/.NET-based programming assignments
    • VS.NET projects will hide some of the ‘gory’ details