

CPSC 474

Distributed Computing Using Web Services and .NET Remoting

Fall 2008

Prerequisite: CPSC 473, Web Programming and Data Management

Since CPSC 473 is a prerequisite, I presume that you are comfortable with C# or VB.NET and with creating web applications in ASP.NET. If you are not, you may find it difficult to keep up in this course. In order to allow you to catch up or review the relevant topics, the first programming assignment asks you to create a simple web application using ASP.NET.

Course Description

This course introduces the concepts of distributed computing and Web services, the applications of XML and Web services, distributed applications development techniques with Web services and .NET Remoting.

ASP.NET XML (ASMX) Web Services and .NET Remoting are now “legacy” communication technologies (where “legacy” means “widely deployed but no longer the cutting edge”). With the release of .NET 3.0 in late 2006, the ASMX and Remoting models were combined with other .NET communication technologies into Windows Communication Foundation (WCF).

This course will briefly cover ASMX Web Services and Remoting because you are still very likely to see them deployed in existing systems. The majority of class time will be devoted to WCF.

Instructor

Kenytt D. Avery, M.S.

Office: CS-542

Office Hours: Monday, 6:00-7:00pm, and by appointment

Phone: (714) 327-3568

E-mail: kavery@fullerton.edu

Hours and Location

Section 1 meets Monday, 7:00-9:45pm in CS-102B.

There will be a 10-minute quiz at 7:00pm and a 10-minute break at approximately 8:20pm.

Textbook

Required: Bustamante, Michele Leroux, *Learning WCF*, O'Reilly and Associates, 2007. ISBN 0-5961-0162-7.

Recommended

- MacDonald, Matthew, *Microsoft .NET Distributed Applications: Integrating XML Web Services and .NET Remoting*, Microsoft Press, 2003. ISBN 0-7356-1933-6.

This is the textbook from previous semesters, and it still provides good background information on distributed applications, ASMX Web services, and .NET Remoting. It is currently available used on Amazon.com for under \$10.

- Ryan, Bill, Horn, Shannon, and Blomsma, Mark, *MCTS Self-Paced Training Kit (Exam 70-529): Microsoft .NET Framework 2.0 Distributed Application Development*, Microsoft Press, 2006.

This is a terrible book, riddled with errors, nonsensical explanations, and non-working code. Nevertheless, if you're looking to update the material from the MacDonald book or pass the MCTS examination, it's the only game in town. The most positive thing I can say about it is that at least you won't be paying full price – it's currently listed on Amazon.com for under \$5.

- Petzold, Charles, *.NET Book Zero: What the C or C++ Programmer Needs to Know About C# and the .NET Framework*. Freely available from <http://www.charlespetzold.com/dotnet/>.

Some advanced aspects of the .NET Framework and the C# language will be reviewed in class, but in general I presume that you are comfortable with .NET and C# at the level of this book.

- Albahari, Joseph, *Threading in C#*. Freely available from <http://www.albahari.com/threading>.

I presume that you know something about threading in general and C# threading in particular. Although it is not a prerequisite, having taken a course in Operating Systems such as CPSC 351 will be helpful. If you need to brush up or supplement the coverage in Chapters 4 and 5 of the textbook, start here.

Additional reading will be assigned as topics are addressed in lecture. Consult <http://csuf.kenytt.net/> for details.

MSDN Webcasts

The author of *Learning WCF* delivered a 15-part series of webcasts for the Microsoft Developer Network called *Windows Communication Foundation Top to Bottom*. If you are a visual learner and would benefit from seeing demonstrations of Visual Studio, I encourage you to visit the author's web page at <http://www.dasblonde.net/WCFWebcastSeries.aspx> to register for the webcasts.

Episodes of the series are listed at the appropriate points in the Course Outline. The webcasts are free, but you will need a Windows Live ID or Microsoft Passport account to register.

.NET 3.5 and Visual Studio 2008

The textbook covers .NET 3.0 and Visual Studio 2005. Since publication, however, Microsoft has released .NET 3.5 and Visual Studio 2008. We will take the following steps in order to update the material:

1. Download the Visual Studio 2008 version of the code from the author's web site at <http://www.thatindigogirl.com/LearningWCFCode.aspx>.
2. When you find that something in Visual Studio 2008 does not work as described in the textbook, check errata at the author's web site at [http://www.thatindigogirl.com/CategoryView.category.Labs%2B\(VS%2B2008%2BErrata\).aspx](http://www.thatindigogirl.com/CategoryView.category.Labs%2B(VS%2B2008%2BErrata).aspx)
3. I will assign graduate students two MSDN Virtual Labs covering some of the new features of WCF available in .NET 3.5. Virtual Labs allow you to use a copy of Visual Studio 2008 hosted on Microsoft's servers to work through a lab exercise. You will need a Windows Live ID or Microsoft Passport account to register for the Virtual Labs.

If you do not yet have a copy of Visual Studio 2008, I suggest that you contact the Computer Science Department to obtain one as soon as possible. Copies of Visual Studio 2008 as well as other Microsoft software are available to students through the Microsoft Academic Alliance.

Class Work

Quizzes

To encourage you to keep up with the assigned reading, there will be a short quiz at the beginning of class each week consisting of a few multiple-choice or fill-in-the-blank questions. Quizzes will begin at 7:00pm and last exactly 10 minutes. If you miss a quiz, it cannot be made up.

Quiz scores account for 10% of your total grade. Because life is unpredictable, I will only count the 10 highest quiz scores toward your grade (dropping the lowest 3).

Lab Exercises

Students will perform 20 Lab Exercises drawn from the textbook throughout the semester. Graduate students will complete two additional Lab Exercises on new features of .NET 3.5. These are hands-on exercises working with Visual Studio and WCF, covering the essential skills you will need for the Programming Assignments.

I expect you to keep a notebook with a record of the work you perform for each Lab.

Your Lab Notebook does not need to be a physical notebook; it may be kept electronically. (In fact, this may be the easiest way to include screenshots and code.) Examples of information that should be recorded for each lab include:

- A summary of the Lab
- Notes on where the lab material could be useful (e.g. at work or on Programming Assignments)
- Problems you run into while attempting the Labs
- Differences between the behavior of .NET 3.0 / VS2005 described in the textbook and the behavior you observe in .NET 3.5 / VS2008
- Mistakes in the sample code (if any)
- Troubleshooting steps that you perform
- MSDN and Google searches that you perform while working on the Lab and their results

You may work alone on the Lab Exercises, but you are encouraged to work together with your classmates. The work recorded in your Lab Notebook should include the names of the students you worked with.

Lab Checkpoints

There will be three checkpoints throughout the semester at which you should submit your Lab Notebook for grading:

- Checkpoint 1 on September 22nd will focus on the content of your Lab Notebook and make suggestions for improvement.
- Checkpoint 2 on October 27th will focus on making sure that you are keeping up with the Lab Exercises.
- Checkpoint 3 on December 17th will assign a final grade to the Lab Notebook.

Exams

There will be a Midterm Exam at 8:30pm on Monday, October 27th, covering lecture material, Chapters 1-4 of the textbook, and other assigned reading.

There will be a comprehensive Final Exam at 7:30pm on Monday, December 15th.

Exams will consist of several short questions requiring a few written paragraphs, some calculations, or (in some cases) a drawing or diagram.

If you will miss a scheduled exam, you must do the following:

1. Contact me **prior** to the exam. You may call, e-mail, or inform me in person. If you cannot contact me directly, you may contact the Computer Science Department. If you wait until after the exam has begun to contact me or the Department, you will not be allowed to make up the exam.

2. Under ordinary circumstances, you will have a 72-hour window to take the exam, beginning the day before the scheduled exam day. This means that you may take the exam the day before, the day of, or the day after the scheduled day.
3. If you need to take the exam at a time outside the 72-hour window (e.g., because you will be out of town), be prepared to provide proof of your absence. For example, you may provide a letter from your employer, on company letterhead, stating that your presence is required on a business trip.
4. You may only take the exam during the regular Computer Science Department office hours (8:00am-5:00pm Monday through Friday.)
5. It is your responsibility to call the Computer Science Department office and arrange to take the exam. You will need 2 hours. When you have scheduled an appointment, you must inform me of the time.

Programming Assignments

There will be 6 Programming Assignments throughout the semester, spaced at two- or three-week intervals. Undergraduates only need to complete the first 5, while graduate students will need to complete all 6. Details for each assignment will be posted on the web at least two weeks prior to the due date.

You may work alone or in a group of up to three other students. If you choose to work with other students, all team members will receive the same grade. Choosing a partner is not a semester-long commitment; you may choose to work with other students on subsequent assignments.

The textbook and lectures will use C#, but you may use any .NET CLI programming language. That said, choosing anything other than C#, VB.NET or C++/CLI makes it very likely that you will spend more time struggling with tools and translating example code than you will getting your work done.

All submissions must be documented. This includes not only appropriate comments, but also written documentation describing the purpose of the program, its design, the structure of its associated artifacts (e.g., source code files, project files, and compiled assemblies), and a brief demonstration of how it is to be used. Screenshots are encouraged.

Programming Assignments are due by **midnight** on the assigned date. If you run into problems, this will give you enough time re-submit after class. Under no circumstances will late work be accepted. If you are working on a team, only one copy of your project needs to be submitted. Make sure, however, to include the names of all members of your team.

In order to submit an assignment, e-mail your work as attachments to the assignment submission e-mail address given in class. Within a few minutes you should receive an auto-reply confirming receipt of your e-mail and listing the attached files. If you do not receive a reply, do the following:

1. Check your e-mail “Spam” folder
2. Check for a “bounce” message indicating that your message was not delivered. Read the envelope carefully before sending me a message that the auto-responder is broken – under ordinary circumstances the auto-responder accepts all incoming mail. In particular, if your message bounces because of a “dangerous” file extension, the problem is on your end, not mine. Keep reading.
3. Rename the attached files to use another extension (e.g., “Assignment1.zip” as “Assignment1.harmlessExtension”) and re-send. In the body of your message, include instructions to rename the file back to the appropriate extension.
4. Try sending from another e-mail address. Specifically, if you are using a free e-mail service such as Hotmail, Yahoo! Mail, or Gmail, try sending again from work, Titan, or your student.fullerton.edu account.

Do not send ordinary correspondence to the assignment auto-responder; I may not read it. The same applies to other e-mail addresses; use kavery@fullerton.edu for correspondence.

Additional work for Graduate Students

The University Regulations (UPS 411.100) state:

Graduate students enrolled in 400-level courses will be expected to:

1. Complete at least one additional assignment beyond that required of undergraduate students in the same course.
2. Demonstrate, in their written and oral performance in the course, quality higher than that expected of an undergraduate.
3. Demonstrate competence in areas required by a graduate-level course...

Requirement (1) is met by Programming Assignment 6 and Lab Exercises 21 and 22.

Requirements (2) and (3) mean that an exam answer that may be worth 4 points to an undergraduate may be worth only 3 points to a graduate student. Expect correspondingly higher standards for code and documentation as well.

Grading

Quiz and Exam answers and Lab Exercises will be graded on a 4-point scale:

4	Correct, and in sufficient detail
3	Substantially correct, with minor errors or missing details
2	Mostly incorrect
1	Completely misses the point of the question
0	Blank

Programming Assignments will be graded as follows:

A	Works as assigned and well-documented
B	Mostly works, but with minor errors in functionality or documentation
C	Does not work, or working but not documented
D	Mostly empty code, or re-submitted previous assignment
F	No assignment submitted

Grades will be weighted as follows:

Quizzes	10%
Lab Checkpoint 1	1%
Lab Checkpoint 2	4%
Lab Checkpoint 3	10%
Programming Assignments	30%
Midterm Exam	20%
Final Exam	25%

Grading Scale

Grades will be assigned on the standard plus/minus scale:

A+	97%	B-	80%	D	63%
A	93%	C+	77%	D-	60%
A-	90%	C	73%	F	< 60%
B+	87%	C-	70%		
B	83%	D+	67%		

I reserve the right to adjust the grading scale in your favor.

Extra Credit

None. Don't bother asking. If you want credit, do the assigned work.

Attendance

You are not required to attend class, but missing too many quizzes would be a bad idea.

Academic Dishonesty

The University Regulations (UPS 300.021) state:

Academic dishonesty includes such things as cheating, inventing false information or citation, plagiarism, and helping someone to commit an act of academic dishonesty. It usually involves an attempt by a student to show possession of a level of knowledge or skill which he/she in fact does not possess.

Cheating is defined as the act of obtaining or attempting to obtain credit for work by the use of any dishonest, deceptive, fraudulent, or unauthorized means. Examples of cheating include, but are not limited to, the following: using notes or aids or help of other students on tests and examinations in ways other than those expressly permitted by the instructor,

plagiarism, as defined below, tampering with grading procedures, and collaborating with others on any assignment where such collaboration is expressly forbidden by an instructor. Violation of this prohibition on collaboration shall be deemed an offense for the persons collaborating on the work, in addition to the person submitting the work.

Plagiarism is defined as the act of taking the work of another and offering it as one's own without giving credit to that source. When sources are used in a paper, acknowledgment of the original author or source must be made through appropriate references and, if directly quoted, quotation marks or indentations must be used.

In a word, **don't**. You will receive an **F** in the course, and I won't even feel sorry for you. You have been warned.

Accommodating Disabilities

The University requires students with disabilities to register with the Office of Disabled Student Services (DSS), located in UH-101 and at (714) 278-3112, in order to receive prescribed accommodations appropriate to their disability. Students requesting accommodations should inform the instructor during the first week of classes about any disability or special needs that may require specific arrangements/accommodations related to attending class sessions, completing course assignments, writing papers or quizzes/tests/examinations.

Withdrawal

The last day to drop the course without a grade of **W** is September 8.

The last day to drop with a grade of **W** for serious and compelling reasons is November 14. Note that in this context, poor grades are neither serious nor compelling.

Class Disruptions

If you need to discuss something with your classmates, please wait until the break or take the discussion outside.

Please turn off cell phones or pagers, or set them to vibrate.

You are welcome to use a laptop during class, but please type quietly. Be advised that if you are using a laptop, I may call on you to look something up (e.g. on Wikipedia or MSDN) and share it with the class during the lecture.

Tentative Course Outline

Note that this schedule is subject to change.

Date	Topics	Reading	Labs	WCF Webcast
8/25	Introduction Distributed Applications Microsoft Development Technologies	Chapter 1: Hello Indigo	1	Part 1 of 15: Overview
9/1	Labor Day			
9/8	ASMX Web Services .NET Remoting Service Oriented Architecture	TBD: ASMX and .NET Remoting	2, 3	
9/15	SOAP, WSDL, and UDDI Interoperability and the WS-I Basic Profile WS-* Services Assignment 1 due	Chapter 2: Contracts	4, 5	Part 2 of 15: Contracts
9/22	XML Namespaces DTDs and XML Schema Definitions Serialization and Marshaling Lab Checkpoint 1	TBD: XML and XSD	6, 7	Part 3 of 15: Contract Versioning
9/29	Networking Protocols TCP and Named Pipes Assignment 2 due	Chapter 3: Bindings	8, 9	Part 5 of 15: Bindings
10/6	Delegates, Events, and Callbacks One-Way and Duplex Communication		10, 11	Part 7 of 15: Messaging Patterns
10/13	SOAP Faults Fault Contracts	Chapter 8: Exceptions and Faults	19, 20	Part 4 of 15: Exceptions and Faults
10/20	Threading and User Interfaces Windows Services Hosting Services in IIS Assignment 3 due	Chapter 4: Hosting	12	Part 6 of 15: Hosting
10/27	Service Instancing Modes Lab Checkpoint 2 Midterm Exam, Chapters 1-3 and 8	Chapter 5: Instancing and Concurrency	13	Part 8 of 15: Instancing Modes
11/3	Sessions Concurrency Modes Thread Synchronization	<i>Threading in C#</i>	14, 15	Part 9 of 15: Concurrency, Throughput, and Throttling
11/10	Reliable Sessions RSS, Atom, and Syndication Assignment 4 due	Chapter 6: Reliability	16, 21	Part 12 of 15: Reliable Messaging
11/17	Message Queueing	TBD: Syndication	17	Part 14 of 15: Queued Messaging
11/24	Fall Recess			
12/1	Role-Based Access Control Public-Key Encryption Certificates Assignment 5 due	Chapter 7: Security	18	Part 10 of 15: Security Fundamentals
12/8	RESTful Web Services AJAX and JSON	TBD: REST, AJAX, and JSON	22	
12/15	Assignment 6 due Lab Checkpoint 3 Final Exam, 7:30-9:20pm			

Lab Exercises

Lab	Location	Title
1	Chapter 1, pp. 21-25	Creating Clients and Services Programmatically
2	Chapter 1, pp. 32-40	Using Tools to Generate Clients and Services
3	Chapter 1, pp. 51-56	Creating an IIS Host and Browsing Metadata
4	Chapter 1, pp. 61-71	Hosting Multiple Services and Sharing Types
5	Chapter 2, pp. 86-90	Designing a Service Contract
6	Chapter 2, pp. 103-110	Working with Data Contracts
7	Chapter 2, pp. 126-130	Controlling Message Serialization with Message Contracts
	Chapter 2, pp. 145-149	Working with Raw Messages
8	Chapter 3, pp. 165-170	Exposing Web Services with BasicHttpBinding and WSHttpBinding
9	Chapter 3, pp. 180-186	Distributing Calls with NetNamedPipeBinding and NetTcpBinding
10	Chapter 3, pp. 199-206	Duplex Communications over TCP and HTTP
11	Chapter 3, pp. 211-214	Using MTOM to Handle Large Messages with Binary Data
	Chapter 3, pp. 214-217	Enabling Streaming for Large File Transfers
12	Chapter 4, pp. 245-251	Working with a Windows Form Host
13	Chapter 4, pp. 263-269	Creating a Windows Service Host
14	Chapter 5, pp. 309-314	Controlling Service Instancing
15	Chapter 5, pp. 322-328	Protecting Shared Resources from Concurrent Access
16	Chapter 6, pp. 351-355	Working with Reliable Sessions
	Chapter 6, pp. 375-381	Configuring WCF Transactions
17	Chapter 6, pp. 399-403	Creating a Queued Service
	Chapter 7, pp. 426-430	Authenticating and Authorizing Windows Credentials
18	Chapter 7, pp. 436-445	Authenticating and Authorizing UserName Credentials
	Chapter 7, pp. 471-478	Creating a Custom Authorization Policy
	Chapter 7, pp. 486-491	Working with WSFederationHttpBinding
19	Chapter 8, pp. 503-505	Working with Uncaught Exceptions
20	Chapter 8, pp. 511-513	Throwing Faults and Declaring Fault Contracts
	Chapter 8, pp. 521-523	Intercepting Uncaught Exceptions with IErrorHandler
21	http://go.microsoft.com/?linkid=8662146	MSDN Virtual Lab: Syndication using Windows Communication Foundation
22	http://go.microsoft.com/?linkid=8662145	MSDN Virtual Lab: Building AJAX/JSON Services using WCF