COMP 790-063
COURSE OVERVIEW

Instructor: Prasun Dewan (FB 150, help401@cs.unc.edu)

Course page:
http://www.cs.unc.edu/~dewan/290/f14/index.html
Comp 790-063: Distributed Collaborative Systems

Course Overview

Distributed collaborative systems allow users from different computers to interact with each other. Thus, they are layered on top of two base technologies: user-interface and distributed abstractions. They include shared desktop/window systems, instant messaging, shared editors and spreadsheets, distributed presentations systems, and of course, games. In this course, we will look at issues in the design, implementation, testing, and evaluation of these systems.

At the end of the course, you will have a basic understanding of how state of the art collaboration software available works, the potential uses of this software, the design space of collaborative applications and infrastructures, and the two base technologies: user-interface and distributed abstractions.

The topics covered will include collaboration architectures, consistency of replicated objects, collaborative user-interfaces, application and system taxonomies, session-based application-level multicast, performance, causality, operation transformation, and concurrency and access.

Linked from my home page (google my name to find it) and UNC course pages
### Class Material

#### Tentative Schedule

<table>
<thead>
<tr>
<th>Unit (Start Date)</th>
<th>Slides</th>
<th>Chapters</th>
<th>Assignment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Application-level Multicast and</td>
<td></td>
<td>Implement IM and Editing User Interface</td>
<td></td>
</tr>
<tr>
<td>Architecture of a Basic Collaborative Application</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Awareness</td>
<td></td>
<td>Implement basic IM/Editing Using Application-Level Multicast</td>
<td></td>
</tr>
<tr>
<td>Jitter Recovery</td>
<td></td>
<td></td>
<td>Implement Replicated AWT/Swing Windows support in project</td>
</tr>
<tr>
<td>Response Times</td>
<td></td>
<td></td>
<td>Implement telepointers with jitter recovery</td>
</tr>
</tbody>
</table>

No book

Each unit will have PPT slides (with recordings), sometimes Word document, and assignment.
COLLABORATION
COMPUTER SUPPORTED COLLABORATION
TECHNIQUES AND ABSTRACTIONS FOR SUPPORTING COLLABORATIVE APPLICATIONS
ABSTRACTIONS

Reusable hardware or software component that can be used by some large set (domain) of applications

Array, class, file, database, textfield
Techniques

Concept applicable to a large set of applications that may or may not be converted to an abstraction

User-interface (graphical, ...); algorithm (bubble sort, ..), design pattern (observer, ..); architecture (client-server, p2p, ...)
COLLABORATIVE APPLICATION

User inputs “draw red circle”

A user’s output can be influenced by the input of at least one other user

Or a user’s input can influence the output of at least one other user

Why collaborative applications?
TOWARDS BEING THERE

Face-to-face Interaction

Computer Supported Interaction
GESTURECAM: REMOTE SURROGATE

Control GestureCam in another location
Both users see what the GestureCam is “looking” at on TV screens
Controller can “zoom in” visually and physically to point at an item
TOWARDS BEING THERE

Face-to-face Interaction

Computer Supported Interaction

Computer Supported < Face to Face

Second best – used to overcome time and cost constraints
BENEFIT: BEYOND BEING THERE

Face-to-face Interaction

Computer Supported Interaction

Hollan & Stornetta

Computer Supported > Face to Face

Use it even when face to face option available
MEETING VIDEO BROWSING

Basic Video Controls:
Play, Pause, etc.

Beyond Being There

Video Processing
- Bookmark Video
- Pause Removal
- Time Compress Without Changing Pitch
- Automatic TOC Generation
- Automatic Slide Summaries
COMBINING BOTH GOALS

Chat Rooms

- Feel Transported to a Common Virtual World
- Avatar Selection
- Anonymity

Toward Being There

Beyond Being There
**Base Technologies**

**Application**

- Interactive → User-Interface
- U1
- Distributed → Inter-process Communication
- U2
- Multi-Threaded

Coupling
BASE TECHNOLOGIES

Collaborative Software

User-Interface Toolkit (Swing/AWT/SWT)  Inter-Process Communication Primitives (RMI/Sockets)

Will learn to apply these technologies and not learn in depth the principles behind them

These technologies are not easy to use!
CLASS ABSTRACTIONS

Collaborative Software

Class Library

User-Interface Toolkit (Swing/AWT/SWT)

Class Library

Inter-Process Communication Primitives (RMI/Sockets)

Will provide Java libraries to allow you to focus on the what rather than the how

Java/Object-Oriented Programming!
COLLABORATIVE ABSTRACTIONS VS APPLICATIONS

Collaborative Applications

Collaborative Abstractions/Toolkit

Class Library

User-Interface Toolkit (Swing/AWT/SWT)

Class Library

Inter-Process Communication Primitives (RMI/Sockets)

Middleware

Challenges?
Design of coupling UI, algorithms, design patterns, architectures and abstractions

Other functions?

Application

Coupling

U1

Coupling

U2

Coupling

Design of coupling UI, algorithms, design patterns, architectures and abstractions
## ISSUES: SEMANTIC

<table>
<thead>
<tr>
<th>Collab. Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Session Management</td>
<td>How do distributed users create, destroy, join, and leave collaborative sessions?</td>
</tr>
<tr>
<td>Coupling</td>
<td>What is the remote feedback of a user command and when is it given?</td>
</tr>
<tr>
<td>Awareness</td>
<td>How are users made aware of uncoupled commands of their collaborators, that is, “out of band” commands?</td>
</tr>
</tbody>
</table>
## ISSUES: SEMANTIC

<table>
<thead>
<tr>
<th>Collab. Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access Control</td>
<td>How do we ensure that users do not execute unauthorized commands?</td>
</tr>
<tr>
<td>Concurrency Control</td>
<td>How do we ensure that concurrent users do not enter inconsistent authorized commands?</td>
</tr>
</tbody>
</table>
## ISSUES: SEMANTIC

<table>
<thead>
<tr>
<th>Collab. Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Merging</td>
<td>How do we merge conflicting commands entered by different users?</td>
</tr>
<tr>
<td>Undo/Redo</td>
<td>What are the semantics of undo/redo operations in a collaboration session?</td>
</tr>
</tbody>
</table>

Many more!
# ISSUES: IMPLEMENTATION

<table>
<thead>
<tr>
<th>Issue</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Objects</td>
<td>What kind of objects are composed in a collaborative application?</td>
</tr>
<tr>
<td>Collaboration Awareness</td>
<td>Which of these objects are collaboration aware? How are these objects integrated with existing collaboration-unaware objects?</td>
</tr>
<tr>
<td>Concurrency</td>
<td>How is the application decomposed into concurrent threads?</td>
</tr>
<tr>
<td>Distribution</td>
<td>How are the application objects placed on different hosts?</td>
</tr>
</tbody>
</table>
## ISSUES: IMPLEMENTATION

<table>
<thead>
<tr>
<th>Issue</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Replication/Migration</td>
<td>Which of these objects are centralized (in a server) and which are replicated (on peers)?</td>
</tr>
<tr>
<td>Performance</td>
<td>What kind of services are provided to overcome jitter and latency?</td>
</tr>
</tbody>
</table>

Many more!
Layered Assignments = Project

Assignments will build on each other to create a semester project.

Due dates normally separated by two weeks (holidays, exams can cause more separation)
GETTING HELP

- Can discuss solutions with each other at a high level
- Not at the code level
- Sharing of code is honor code violation
- Can help each other with debugging as long as it does not lead to code sharing
- Will give algorithms for assignments
Getting Help and Class Discussion

We will be using Piazza for class discussion and getting help. The system is highly catered to getting you help fast and efficiently from classmates, the TA, and myself. Rather than emailing questions to the teaching staff, I encourage you to post your questions on Piazza. If you do not get a response within a day or two on Piazza, please send mail to help401@cs.unc.edu. But try Piazza first. Do not send mail to an individual instructor, as that can overwhelm him - such mail will be ignored.

Before posing a question, please check if this question has been asked before. This will reduce post clutter and reduce our burden. Repeat questions will be ignored by the instructors.

Piazza allows anyone to respond. So if you see a question that you think you can respond to, please do so, as that will reduce our burden and help you "teach" your fellow students.

This will be a form of class participation that will be noted when I allocate my fudge points!

Hope it works well

If you have any problems or feedback for the developers, email team@piazza.com.

Signup Link: piazza.com/unc/fall2014/comp790063

Class page: piazza.com/unc/fall2014/comp790063/home
## Grade Distribution

<table>
<thead>
<tr>
<th>Component</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exams (Two midterms, no final)</td>
<td>40%</td>
</tr>
<tr>
<td>Assignments (Home work)</td>
<td>60%</td>
</tr>
<tr>
<td>Fudge Factor (Class/piazza participation, other distinguishing factors), particularly useful for borderline cases</td>
<td>10%</td>
</tr>
</tbody>
</table>

*Project Heavy but No final!*
PROJECTS

Demo of Projects