COMP 533 - DISTRIBUTED SYSTEMS: COURSE INFO

Instructor: Prasun Dewan (FB 150, dewan@unc.edu)
DISTRIBUTED PROGRAM?

A program “involving” multiple computers

Specific computers must be bound at run time

→ Program can run on a single computer

Definition involves processes
Program

Program is execution instance of program, associated with program and memory.

Execution instance

Process is execution instance of program, associated with program and memory.

Processes are independent activities that can interleave or execute concurrently.

Thread is also an independent activity, but within a process, associated with a process and a stack.

Same program can result in multiple processes.

Thread

Process

Same program can result in multiple processes.
Different processes can execute on different (distributed) computers.

A single process executes on one machine.
Distributed Program

Is the shell a distributed program?

Connected process pair: Some computation of a process can be influenced by or influence computation of the other process.

Connected process group: Each process is coupled to at least one other process in the group.

Graph created by creating pair-wise dependency links is not partitioned—every node reachable from every other node.
DISTRIBUTION VS. CONCURRENCY

Distribution, no fine-grained concurrency

Concurrency, not distribution

Distribution and fine-grained concurrency (typical)
NON-DISTRIBUTED VS. DISTRIBUTED PROGRAM

Non-Distributed

- Creates a single process unconnected to any other process
- Must deal with sequential and possibly concurrency issues

Distributed

- Creates a pair or larger group of connected processes
- Must also deal with distribution and usually concurrency issues
EXAMPLES OF PRACTICAL DISTRIBUTED APPLICATIONS

Distributed applications?

Non distributed applications?

In today’s world, what is or should not be distributed?
EXAMPLES

- Distributed Repositories (Files, Databases)
- Remotely Accessible Services (Printers, Desktops)
- Collaborative Applications (Games, Shared Desktops)
- Distributed Sensing (Disaster Prediction)
- Computation Distribution (e.g. Simulations, Big data)
**Prereqs?**

**Distributed Systems**

**OS**
- Interrupt Management
- Thread Communication
- Thread Synchronization
- Thread Management

**Networking**
- UDP
- TCP/IP
- IP
- Link-Level Communication
- Physical Communication
EXISTING MVC-BASED DISTRIBUTED SIMULATION

- Java NIO
- Asynchronous Broadcast
- Custom IPC/RPC
- Custom Serializer
- Replicated Objects
- GIPC IPC/RPC
- Paxos
- Two-Phase Commit
- Atomic Broadcast

DISTRIBUTED COMPUTING

- Operating Systems
- Compilers
- Distributed Computing
- Toolkits, Databases, APIs
- Basic Data Structures and Algorithms
- Applications (Data Centers, Mobile, IoT, Collaboration)
- Programming Languages

IPC: Design, Analysis, and Implementation

Consistency: Design, Analysis, and Implementation

UNDERGRAD COURSE ON DISTRIBUTED SYSTEMS?
UNDERGRAD COURSE ON DISTRIBUTED SYSTEMS?

Distributed Consensus-based Simulation with Consensus on Consensus Mechanism
- Paxos
- Two-Phase Commit
- Atomic Broadcast
- Asynchronous Broadcast
- Replicated Objects
- Custom IPC/RPC
- GIPC IPC/RPC
- Custom Serializer
- Java NIO
- Atmosphere Cloud-Based Infrastructure (Cyberinfrastructure)

IPC: Design, Analysis and Implementation

Consistency: Design, Analysis, and Implementation

- Distributed Computing
- Toolkits, Databases APIs
- Basic Data Structures and Algorithms
- Programming Languages
- Applications (Data Centers, Mobile, IOT, Collaboration)
- Compilers
- Operating Systems
The sending client trace shows that the input string is set, then the client locally processes, and then it sends to the server.

This time the clients process locally, so the middle takes and the left moves before the server can relay the commands. This causes an inconsistency, as the left simulation moves first, so when it receives the take command after resuming past the breakpoint it cannot take candy as it is out of position.
The sending client trace in atomic mode shows that it receives the inputstring, and then sends the command to the server without local processing. It then waits for the server to send the command back before processing.
In summary the local loop took 11196 ms to complete, the atomic 5233 ms, and the non atomic 24823 ms. The local does not send to the server. The non atomic takes longer to complete writing, because it does local processing and writes. The atomic doesn’t do any local processing, so the time for the main thread to write to socket is quicker since it’s processing after it reads from the socket. However total time for both is comparable.
**Semi-Automatic Grading**

**Grading Assistant**

<table>
<thead>
<tr>
<th>Feature</th>
<th>Max</th>
<th>Score</th>
<th>Manual</th>
<th>Extra</th>
<th>Graded</th>
<th>Full Credit</th>
<th>Validate</th>
<th>Selected</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 ServerTagged</td>
<td>5.0</td>
<td>5.0</td>
<td>□ Manual</td>
<td>□ Extra</td>
<td>✓ Graded</td>
<td>✓ Full Credit</td>
<td>□ Validate</td>
<td>□ Selected</td>
</tr>
<tr>
<td>2 StaticArguments</td>
<td>0.0</td>
<td>0.0</td>
<td>□ Manual</td>
<td>□ Extra</td>
<td>✓ Graded</td>
<td>✓ Full Credit</td>
<td>□ Validate</td>
<td>□ Selected</td>
</tr>
<tr>
<td>3 OneClientThreadsNonAtomic</td>
<td>5.0</td>
<td>5.0</td>
<td>□ Manual</td>
<td>□ Extra</td>
<td>✓ Graded</td>
<td>✓ Full Credit</td>
<td>□ Validate</td>
<td>□ Selected</td>
</tr>
<tr>
<td>4 OneClientConnection</td>
<td>20.0</td>
<td>20.0</td>
<td>□ Manual</td>
<td>□ Extra</td>
<td>✓ Graded</td>
<td>✓ Full Credit</td>
<td>□ Validate</td>
<td>□ Selected</td>
</tr>
<tr>
<td>5 OneClientThreadsAtomic</td>
<td>5.0</td>
<td>2.5</td>
<td>□ Manual</td>
<td>□ Extra</td>
<td>✓ Graded</td>
<td>✓ Full Credit</td>
<td>□ Validate</td>
<td>□ Selected</td>
</tr>
<tr>
<td>6 OneClientReadWrite</td>
<td>0.0</td>
<td>0.0</td>
<td>□ Manual</td>
<td>□ Extra</td>
<td>✓ Graded</td>
<td>✓ Full Credit</td>
<td>□ Validate</td>
<td>□ Selected</td>
</tr>
<tr>
<td>7 ClientTagged</td>
<td>5.0</td>
<td>5.0</td>
<td>□ Manual</td>
<td>□ Extra</td>
<td>✓ Graded</td>
<td>✓ Full Credit</td>
<td>□ Validate</td>
<td>□ Selected</td>
</tr>
<tr>
<td>8 OneClientReadWriteNonAtomic</td>
<td>70.0</td>
<td>70.0</td>
<td>□ Manual</td>
<td>□ Extra</td>
<td>✓ Graded</td>
<td>✓ Full Credit</td>
<td>□ Validate</td>
<td>□ Selected</td>
</tr>
</tbody>
</table>

**Auto Notes:**

-- Couldn't find client connection info in traces.

**Manual Notes:**

**Transcript**

```
INPUT(StaticArguments)
g 0

MAIN ARGS(StaticArguments)
[localhost:9000,Generic Client.true](OUTPUT(StaticArguments))
****Tracer: showinfo = true
```
Locally Executable Checks

Debug and your programs and evaluate your progress.

Debug our tests!

Sufficient for evaluating work?

Abstraction and Algorithm Analysis?
**Regular Credit**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Midterm</td>
<td>25%</td>
</tr>
<tr>
<td>Final</td>
<td>25%</td>
</tr>
<tr>
<td>Regular Collaborative Open-Book Quizzes</td>
<td>10%</td>
</tr>
<tr>
<td>Assignment: Correctness</td>
<td>25%</td>
</tr>
<tr>
<td>Assignment: Video</td>
<td>15%</td>
</tr>
</tbody>
</table>
## Extra Credit

<table>
<thead>
<tr>
<th>Extra Credit</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class participation: Reports of In-Class Answers by You</td>
<td>5%</td>
</tr>
<tr>
<td>Class participation: Reports for In-Class Answers by You or Others</td>
<td>5%</td>
</tr>
<tr>
<td>Extra credit serialization assignments</td>
<td>10%</td>
</tr>
<tr>
<td>Extra credit Paxos assignment</td>
<td>5%</td>
</tr>
<tr>
<td>Extra credit quizzes</td>
<td>???</td>
</tr>
</tbody>
</table>

Many ways to succeed!

EC > Final exam
### Why Class Participation and Diaries

Instructor needs feedback

<table>
<thead>
<tr>
<th>Learn to speak in public (I will ask questions every one can answer such as what was interesting in the material you studied)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oral “exam” over whole semester</td>
</tr>
<tr>
<td>Retain material better, logical vs physical attendance</td>
</tr>
<tr>
<td>Recruiters regularly ask about oral and verbal skills</td>
</tr>
<tr>
<td>NSF Survey asked what was being done to improve these skills</td>
</tr>
</tbody>
</table>
Learning Resources

No textbook!

Alternatives?
First-year grads and several undergrads took this course
Comp 533: Distributed Systems

Course Overview

This course will provide a practical overview of distributed systems. It will be driven by a series of implementation-based projects.

We will cover the design, implementation, performance, and applications of abstractions for sharing information among distributed processes. These sit between the network layer and application and include: distributed shared memory, byte and object communication, remote procedure call, and broadcast/multicast sessions. Some of the general issues we will address are naming, synchronization, extensibility, and routing and consistency of broadcast/multicast messages.

We will overview but not focus on any of the specific applications of distributed abstractions such as the Web, distributed data mining, distributed databases, file systems, and distributed collaborative systems, which are covered in other courses. Our focus will be on foundational concepts applicable to all of these application areas. These concepts will be introduced as layers that sit above the OS and networks.

The assignments and lectures will have two related goals. The first is to expose you to standard implementation of the covered abstractions in a Java environment and other distributed systems and discuss how these abstractions can be used in a realistic programs. The assignments addressing this goal will involve the use of only Java abstractions. The second is to expose you to the implementation of these abstractions. I have developed a Java-based teaching/research testbed, called GIPC (for Generalized Interprocess communication), that provides implementation of these abstractions. In the assignments addressing this goal, you will replace parts of this extendible system to understand implementation concepts. The result of the replacement exercised will be a system that is more sophisticated in many ways than the state of the art, and you are likely to use it for programming future distributed programs.

At the end of the course, you will have a basic understanding of how distributed software works, the potential uses of this software, the design and implementation space of distributed abstractions; the performance of alternative distributed abstractions and how to run experiments to measure performance. As a distributed program is also a parallel program, you will also sharpen your understanding of threads and thread synchronization. Because of the emphasis on assignments, you will gain practice with the use and implementation of advanced software engineering concepts such as layers, generic types, factories, and abstract factories.
ISSUES IN MESSAGE PASSING

- Reliable message delivery?
- In-order message delivery?
- Port access (and message routing)?
- Operations?
- Synchronous vs asynchronous?
- Blocking vs non-blocking?
- Buffering of messages at queue?
- Location of communicating threads

Process/Thread

Message Queue

message

message

message

Port (Mailbox)

Process/Thread

RELIABLE VS. UN-RELIABLE
POWERPOINT OF SLIDES

ISSUES IN MESSAGE PASSING

Process/Thread

Message Queue

Port (Mailbox)

message\(^1\)

message\(^2\)

message\(^3\)

Reliable message delivery?

In-order message delivery?

Port access (and message routing)?

Operations?

Synchronous vs asynchronous?

Blocking vs non-blocking?

Buffering of messages at queue?

Location of communicating threads?
Can escape out into unsynchronized or no audio mode (WPS Office on Android will play synchronized audio)
PowerPoint Slides With Unsynchronized Recordings and Media Control

Issues in Message Passing

- Reliable message delivery?
- In-order message delivery?
- Port access (and message routing)?
- Operations?
- Synchronous vs asynchronous?
- Blocking vs non-blocking?
- Buffering of messages at queue?
- Location of communicating threads?
Recorded YouTube Videos

Issues in Message Passing

Play 2X, rewind, pause, fast-forward to match understanding pace

Youtube video generated from PPT Recordings, does not allow slide-based browsing

PPT modes allow slide-based browsing but requires downloading PPT
Long pauses, you may know the answer

Cannot hear student answer

Audio is not the fastest way to get information, specially when studying for an exam

Recordings of live lectures with q/a rather than 15 minute lessons

Can fast forward

You can get a clue from my answer

VIDEO NATURE
WRITTEN DOCUMENTS

- Interprocess Communication
  - Shared Memory
  - Software Interrupt
  - Message Passing
    - Reliability of Messages
    - Order
    - Access
    - Remote Assignment vs Procedure Call
    - Synchronous vs Asynchronous
    - Buffering of Messages
    - Pipes
    - Selectivity of Receipt
    - Integration with Programming Language
    - Integration with I/O
  - Xinu Low-Level Message Passing
    - Semantics
    - Implementation

- Communication across a Network

Incomplete  Concise
DETAILED ASSIGNMENTS

Consult this section only if you are stuck trying to implement some aspect of the assignment. You will, of course, learn more by figuring out the implementation structure on your own.

You essentially have to extend the “coupled simulation” program using the distributed pattern illustrated in the “meaning of life” distributed program.

The trace of the coupled simulation program illustrates how it couples the two simulation models.

The following steps occur before any interaction occurs:

```java
I***{main}(AddedPropertyChangeListener) EvtSrc(AHalloweenCommandProcessor)
I***{main}(AddedPropertyChangeListener) EvtSrc(AHalloweenCommandProcessor)
I***{main}(AddedPropertyChangeListener) EvtSrc(AHalloweenCommandProcessor)
coupledSims.ASimulationCoupler@3445341
I***{main}(AddedPropertyChangeListener) EvtSrc(AHalloweenCommandProcessor)
coupledSims.ASimulationCoupler@14cc1f5c
```

The first two steps are ObjectEditor registering view observers (whose toString() methods return the empty strings) with each of the two coupled models described by the class,
WHAT EXACTLY DO YOU NEED TO KNOW

You need to know enough to know how to answer the quizzes and do the assignments.

Not all questions can be answered based on posted material, google, think, and talk to others!
WHAT DO WE DO IN CLASS?

Without existing resources

With existing resources

Live lectures introducing a quiz-based section

Do quiz collaboratively or use class time for other 533-based work

Live discussion with field levelled
MORE COURSE DESCRIPTION

Piazza for announcements: use onyen

Course Page:
COLLABORATION

- Can discuss solutions with each other at a high level
- Not at the code level
- Sharing of code or diary text is honor code violation
- Can help each other with debugging as long as it does not lead to code sharing
- Assignments may contain solution in English (read only if stuck)