What is this course about?

• Overview of the concepts, systems and techniques of distributed systems
  ▪ Basic concepts and principles of distributed systems, which are useful in many real-world applications/projects
  ▪ Lectures/readings, discussions, case studies, extensive hands-on exercises/homework
  ▪ By the end of this course, you will have a good technical understanding of many distributed system related technologies

• NOT a programming class
  ▪ We will use several languages (XML, JavaScript, XQuery, etc.), only the basics for exercises/homework
  ▪ No real programming. Only need to understand programs and make some changes

• More technical than MIS, more application-oriented than CS
Why this course is important?

• People are using distributed systems everyday
  ▪ Web sites: Gmail, Facebook, ...
  ▪ Distributed databases: MySQL Cluster, Hbase, ...
  ▪ Distributed file systems: NFS, AFS, ...
  ▪ Distributed scientific software: MPI, OpenMP, ...

• Knowing the knowledge might help your future career
  ▪ One of most actively evolving topic: Cloud, Big Data, Mobile, GPU, ...
  ▪ Design/implement a new distributed system
  ▪ Running data analytics on a distributed system
  ▪ Managing a distributed database
  ▪ ...
Grading

• 10 discussions: 10*0.5=5 points for 5%
  ▪ Often open questions, no exact one correct answer
  ▪ Exchange ideas with your neighbors during discussion
  ▪ Write your final brief answer at blue book (a new page for each discussion)
  ▪ Return your blue book every class to be graded
  ▪ You can get full point as long as your answer is reasonable
  ▪ You are also welcome to write your feedback/suggestions

• 1 group case study: 10 points for 10%
  ▪ Form a group of 3-4 students: more on it later
  ▪ Select a topic and find a related paper or project interesting to your group
  ▪ Present your work at class

• 7 homework: 10*5=35 points for 35%
  ▪ Submit before deadline (the end of Sunday), it will be graded Monday morning

• 3 exams: 15+20+15=50 points for 50%
How to Study Well?

• Good participation in class: question and discussion
• Good group collaboration for your case study
• Start exercise and homework early
• Read the requirements carefully
• Read book chapters, especially before exams
• If needed, ask for help early
  ▪ If you are experiencing any problems that affect your performance in this class, please contact instructor immediately
• Study hard from the beginning of the semester
Learning Outcomes

• After learning chapter 1, you should be able to
  ▪ Understand the basics of distributed systems and service oriented architecture (SOA)
  ▪ Understand the layered architecture of network
  ▪ Can access and navigate linux server (gl.umbc.edu)
Communication

• Slack (is651-spring-2018.slack.com)
  ▪ You can use it through either website or smartphone app
  ▪ Channels for each chapter
  ▪ You can also send direct message
  ▪ Send to instructor on private issues
    ▪ Better informed and faster responses

• Instructor will try to reply within 24 hours
  ▪ Ask your questions early, not right before the deadline
Course Prerequisites

• Programming with systems analysis and design
• Networking
• Databases
Distributed Systems

• A distributed system consists of multiple autonomous computers that communicate through a computer network. The computers interact with each other in order to achieve a common goal.

• Definition: A distributed system is a collection of independent computers that appears to its users as a single coherent system.
Service-Oriented Architecture (SOA)

• SOA is an evolution of distributed computing

• SOA defines how two computing entities, such as programs, interact in such a way as to enable one entity to perform a unit of work on behalf of another entity.

• Service interactions are defined using a description language. Each interaction is self-contained and loosely coupled, so that each interaction is independent of any other interaction.

• This is an abstract architectural concept and no specific technology is assumed
SOA is still Very Popular in Real World

• Examples:
  • Amazon Web Services
  • Cloud computing follows XaaS architecture
  • Microservices
  • Service oriented Manufacturing
  • MEAN (MongoDB, Express.js, AngularJS, and Node.js) stack for web application
  • ...

IS 651: Distributed Systems
SOA Concepts

- Top-down
- Bottom-up
- Loosely-coupled
- Interface, not application
XML Web Service

- XML Web Services are the current technology most associated with SOA.
- A Web Service (XML Web Service) is a unit of code that can be activated using HTTP requests. Stated another way, a Web Service is an application component (in any language) that can be remotely callable using standard Internet Protocols such as HTTP and XML.
- Web Services came into existence to deliver distributed computing over the Internet.
- A major advantage of the Web services architecture is, it allows programs written in different languages on different platforms to communicate with each other in a standards-based way.
Traditional B2B Scenario

• To support the company's inventory query and supply ordering with no web services, the enterprise architecture requires that four systems be connected using three proprietary interfaces.

• Across organization communication.

B2B Scenario with SOA

• SOA opens up a number of new possibilities for conducting B2B commerce without significant reworking of the underlying systems.

• In addition to eliminating the proprietary interfaces, the SOA makes it easily possible for the first plant to check directly with the second plant and place orders without going through the HQ computer.
B2B Scenario with More SOA

• The manufacturer now wants to institute an electronic competitive bidding system for its orders.

• The suppliers who want to bid on the opportunity to win business from the manufacturer can connect to the bidding system through a Web service.
Networking - OSI Seven Layer Model

<table>
<thead>
<tr>
<th>Media/Host</th>
<th>Data Unit</th>
<th>Layer</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Host</td>
<td>Data</td>
<td>7. Application</td>
<td>HTTP</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6. Presentation</td>
<td>Representation</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5. Session</td>
<td>Dialogue</td>
</tr>
<tr>
<td>Segment</td>
<td></td>
<td>4. Transport</td>
<td>End-to-end</td>
</tr>
<tr>
<td>Media</td>
<td>Datagram</td>
<td>3. Network</td>
<td>Routing</td>
</tr>
<tr>
<td></td>
<td>Frame</td>
<td>2. Data Link</td>
<td>MAC address</td>
</tr>
<tr>
<td></td>
<td>Bit</td>
<td>1. Physical</td>
<td>Signals</td>
</tr>
</tbody>
</table>
Networking - Peer-to-Peer (P2P) Communication

• Peer-to-peer communications through protocol data unit (PDU) headers
• The data unit of one layer is part of data unit of its underlying layer
Networking - TCP/IP

TCP/IP model | Protocols and services | OSI model |
---|---|---
Application | HTTP, FTP, Telnet, NTP, DHCP, PING | Application |
Transport | TCP, UDP | Presentation |
Network | IP, ARP, ICMP, IGMP | Session |
Network Interface | Ethernet | Transport |

IS 651: Distributed Systems
Networking - TCP/IP Headers

<table>
<thead>
<tr>
<th>Source port #</th>
<th>Destination port #</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sequence #</td>
<td>Acknowledgement #</td>
</tr>
<tr>
<td>Other headers and options</td>
<td>TCP Header</td>
</tr>
</tbody>
</table>

TCP/IP header encapsulation

From [http://www.tenouk.com/Module42a.html](http://www.tenouk.com/Module42a.html)
Distributed Systems Topics

• **Architecture**: how distributed systems are put together to provide a single system abstraction from many separate parts.

• **Fault-tolerance**: how distributed systems can continue to provide service when some parts have failed to provide availability until they are recovered.

• **Consistency**: how distributed systems can maintain logical coherency when data is distributed.

• **Scalability**: how a distributed system can grow to meet demand in an efficient and effective way.

• **Performance**: how to optimize response time when components are distributed.

• **Security**: how to ensure data integrity and confidentiality is a distributed environment.
Eight Fallacies for Distributed Systems

• The network is reliable.
• Latency is zero.
• Bandwidth is infinite.
• The network is secure.
• Topology doesn't change.
• There is one administrator.
• Transport cost is zero.
• The network is homogeneous.
Unix/Linux Command-line

• Use of an SSH client
  • Mac: Terminal command line
  • Windows: Putty, Bitvise SSH Client
• Use of an SCP client
  • Mac: Terminal command line, cyberduck
  • Windows: cyberduck, Bitvise SSH Client, WinSCP
• Use of a small number of unix commands
  • ls, mkdir, cd, pwd, man, more, cat, wget, ...
• Edit a text file on a unix machine
  • pico, vi, nano, emacs, ...
• Navigation of your student account on gl.umbc.edu
Exercise/Homework Notes

• Exercises are demonstrated by instructor in class
  ▪ No grading for exercises

• You should do homework by yourself
  ▪ No communication before homework submission

• Same homework code from multiple students will be investigated
  ▪ No cheating

• Make sure you submit your homework by the deadline (End of Sunday)
  ▪ It will be graded the next morning

• Any time a homework asks you to put information into a file with a .txt extension, it must be a plain text file
  ▪ Never use a word processor format
  ▪ Most submissions are just urls.
Demonstration for exercise #0