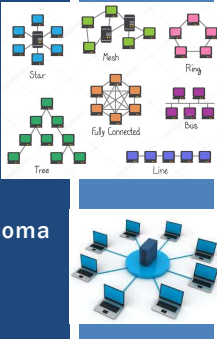


TCSS 558:  
APPLIED DISTRIBUTED COMPUTING

Introduction

Wes J. Lloyd  
Institute of Technology  
University of Washington - Tacoma



OBJECTIVES

- Course demographics survey
- Syllabus
- Chapter 1 - What is a distributed system?
- Design goals of distributed systems:
  - Resource sharing / availability
  - Distribution transparency
  - Openness
  - Scalability
- Activity: Design goals of distributed systems
- Research directions


January 7, 2019

TCSS558: Applied Distributed Computing [Winter 2019]  
Institute of Technology, University of Washington - Tacoma

L1.2

DEMOGRAPHICS  
SURVEY

SEE LINK AT:  
<http://faculty.washington.edu/wlloyd/courses/tcss558/announcements.html>




January 7, 2019

TCSS558: Applied Distributed Computing [Winter 2019]  
School of Engineering and Technology, University of Washington - Tacoma

L1.3

TCSS 558 B – Winter 2019

- 558 PRIME TIME section: 5:50-7:50pm
  - Jan 7: civil twilight - 5:11pm
  - Mar 13: civil twilight - 7:42pm
- Winter Quarter '19 - Going green...
  - 10% reduction of carbon footprint
- 18 class meetings
  - In contrast to usual 20
  - 2 Monday holidays: Jan 21, Feb 18
- Saves commuting time
  - Less fuel expenses
- Easier to achieve perfect attendance
- Convenient for TCSS 598
- Final exam Monday March 18<sup>th</sup>



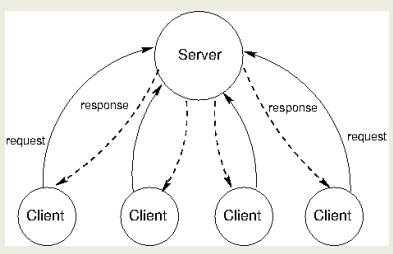
TCSS 558 B  
PRIME TIME  
WINTER  
2019

January 7, 2019

TCSS558: Applied Distributed Computing [Winter 2018]  
School of Engineering and Technology, University of Washington - Tacoma

L1.4

WHAT IS A DISTRIBUTED SYSTEM?

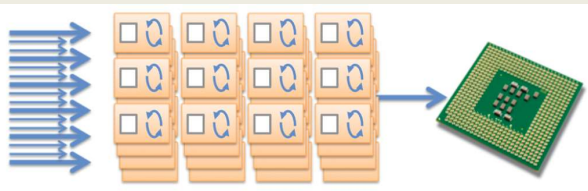


January 7, 2019

TCSS558: Applied Distributed Computing [Winter 2019]  
Institute of Technology, University of Washington - Tacoma

L1.5

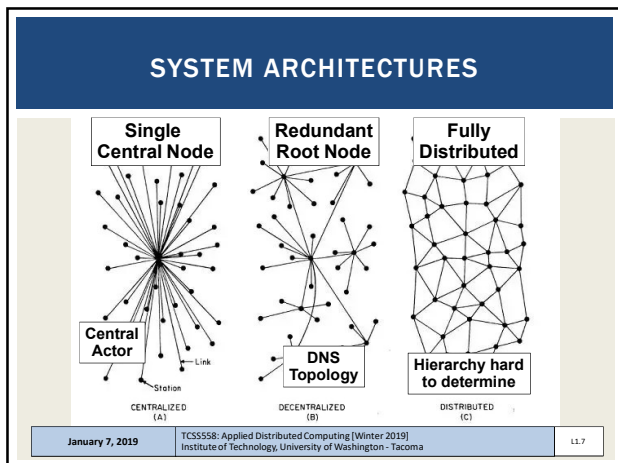
CLIENT/SERVER



January 7, 2019

TCSS558: Applied Distributed Computing [Winter 2019]  
Institute of Technology, University of Washington - Tacoma

L1.6



### WHAT IS A DISTRIBUTED SYSTEM?

- **Definition:**
- A **collection of autonomous computing elements** that appears to users as a single coherent system.
- How nodes collaborate / communicate is **key**
- **Nodes**
  - Autonomous computing elements
  - Implemented as hardware or software processes
- **Single coherent system**
  - Users and applications perceive a single system
  - Nodes collaborate, and provide "abstraction"

January 7, 2019 TCSS558: Applied Distributed Computing [Winter 2019]  
Institute of Technology, University of Washington - Tacoma L1.8

### CHARACTERISTICS OF DISTRIBUTED SYSTEMS - 1

- **#1: Collection of autonomous computing elements**
  - Node synchronization
  - Node coordination
  - Overlay networks – enable node connectivity
- **#2: Single coherent system**

January 7, 2019 TCSS558: Applied Distributed Computing [Winter 2019]  
Institute of Technology, University of Washington - Tacoma L1.9

### C1: AUTONOMOUS COMPUTING ELEMENTS NODE SYNCHRONIZATION

- Nodes behave/operate independently
- Maintain separate clocks (notion of time)
  - There is no global clock
- Nodes must address synchronization and coordination

Node synchronization and coordination...

- Subject of chapter 6

January 7, 2019 TCSS558: Applied Distributed Computing [Winter 2019]  
Institute of Technology, University of Washington - Tacoma L1.10

### C1: AUTONOMOUS COMPUTING ELEMENTS NODE COORDINATION

- Must manage **group membership**
- Nodes can join/leave the group
- Authorized vs. unauthorized nodes
- Open group: any node is allowed to join the distributed system
- Closed group: communication is restricted to the group
  - Admission control: supports mechanism to enable nodes to join/leave the group

January 7, 2019 TCSS558: Applied Distributed Computing [Winter 2019]  
Institute of Technology, University of Washington - Tacoma L1.11

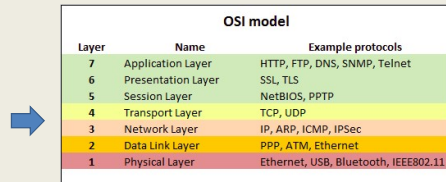
### C1: AUTONOMOUS COMPUTING ELEMENTS OVERLAY NETWORKS

- Simply implies "on top of" another network
- Typically the internet,
- Nodes in a collection communicate only with other nodes in the system
- The set of neighbors may be dynamic, or may even be known only implicitly (i.e., requires a lookup).
- Structured: each node has a well-defined set of neighbors with whom it can communicate (tree, ring).
- Unstructured: each node has references to randomly selected other nodes from the system.
- Always connected, communication paths are available

January 7, 2019 TCSS558: Applied Distributed Computing [Winter 2019]  
Institute of Technology, University of Washington - Tacoma L1.12

## C1: AUTONOMOUS COMPUTING ELEMENTS OSI MODEL

- Open systems interconnect:
- Standardization of the functionalities in a communication system via abstract layers



January 7, 2019

TCCS558: Applied Distributed Computing [Winter 2019]  
Institute of Technology, University of Washington - Tacoma

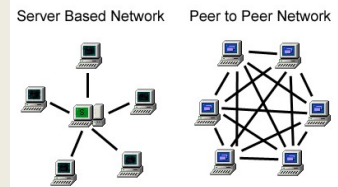
L1.13

## C1: AUTONOMOUS COMPUTING ELEMENTS PEER-TO-PEER NETWORK

- Distributed systems leverage the ability of computing systems to collaborate and aggregate resources across many nodes providing potential for great scale and robustness than centralized client-server models

- How can **fault tolerance** be provided in the client/server model?

- How can **fault tolerance** be provided by the peer-to-peer model?



January 7, 2019

TCCS558: Applied Distributed Computing [Winter 2019]  
Institute of Technology, University of Washington - Tacoma

L1.14

## CHARACTERISTIC 2: SINGLE COHERENT SYSTEM

- Collection of nodes operates the same, regardless of where, when, and how interaction between a user and the system takes place
- Distribution transparency
- From the user's perspective, they can't discern how the distributed system is implemented
- What are some examples of transparent distributed systems that you frequently use?

January 7, 2019

TCCS558: Applied Distributed Computing [Winter 2019]  
Institute of Technology, University of Washington - Tacoma

L1.15

## C2: SINGLE COHERENT SYSTEM DISTRIBUTION TRANSPARENCY

- An end user cannot tell where a computation takes place
- Where data is stored is abstracted (hidden)
- State of data replication is abstracted (hidden)
  - Is data consistent?
- Devices accessing services deployed on "The Cloud" is one example of distributed transparency



January 7, 2019

TCCS558: Applied Distributed Computing [Winter 2019]  
Institute of Technology, University of Washington - Tacoma

L1.16

## C2: SINGLE COHERENT SYSTEM DISTRIBUTION TRANSPARENCY - 2

- Partial failures: when part of a distributed system fails
- Hiding partial failures and their recovery is challenging
- Leslie Lamport, a distributed system is:
  - ".. one in which the failure of a computer you didn't even know existed can render your own computer unusable"

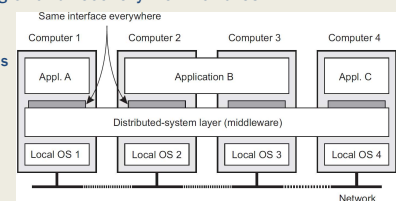
January 7, 2019

TCCS558: Applied Distributed Computing [Winter 2019]  
Institute of Technology, University of Washington - Tacoma

L1.17

## C2: SINGLE COHERENT SYSTEM MIDDLEWARE

- The OS of distributed systems:
- Facilities for inter-node communication
- Security, user account services (authentication, access control)
- Reliability: masking of and recovery from failures
- Service protocols
- Transaction support: "atomic" transactions all-or-nothing



January 7, 2019

TCCS558: Applied Distributed Computing [Winter 2019]  
Institute of Technology, University of Washington - Tacoma

L1.18


## DESIGN GOALS OF DISTRIBUTED SYSTEMS

- **Accessibility:** support for sharing resources
- **Distribution transparency**
- **Openness:** avoiding vendor lock-in
- **Scalability**

January 7, 2019
TCS558: Applied Distributed Computing [Winter 2019]  
Institute of Technology, University of Washington - Tacoma
L1.19

## ACCESSIBILITY: RESOURCE SHARING

- Easy for users (and applications) to share remote resources
  - Storage, compute, networks, services, peripherals, ...
- Field programmable arrays (FPGAs) "as a service":
 



Amazon EC2 F1 Instances

Run Customizable FPGAs in the AWS Cloud

  - <https://aws.amazon.com/ec2/instance-types/f1/>
- Nearly any resource can be shared

January 7, 2019
TCS558: Applied Distributed Computing [Winter 2019]  
Institute of Technology, University of Washington - Tacoma
L1.20

## DISTRIBUTION TRANSPARENCY

- Types of transparency
- Object is a resource or a process

Transparency	Description
Access	Hide differences in data representation and how an object is accessed.
Location	Hide where an object is located
Relocation	Hide that an object may be moved to another location while in use
Migration	Hide that an object may move to another location
Replication	Hide that an object is replicated
Concurrency	Hide that an object may be shared by several independent users
Failure	Hide the failure and recovery of an object

January 7, 2019
TCS558: Applied Distributed Computing [Winter 2019]  
Institute of Technology, University of Washington - Tacoma
L1.21

## DISTRIBUTION TRANSPARENCY - 2

- Why would we want **location transparency**?
  - Uniform resource locator (URL) ...
  - Where is it?
- **Relocation transparency:**
  - May feature temporary loss of availability
  - Cloud application migrates from one data center to another
- **Migration transparency:**
  - There is no loss of availability
  - e.g. cell phones roaming networks

January 7, 2019
TCS558: Applied Distributed Computing [Winter 2019]  
Institute of Technology, University of Washington - Tacoma
L1.22

## DISTRIBUTION TRANSPARENCY - 3

- **Replication transparency:**
  - Hide the fact that several copies of a resource exist
  - What if a user is aware of, or has to interact with the copies?
- **Reasons for replication:**
  - Increase availability
  - Improve performance
  - Fault tolerance: a replica can take over when another fails

January 7, 2019
TCS558: Applied Distributed Computing [Winter 2019]  
Institute of Technology, University of Washington - Tacoma
L1.23

## DISTRIBUTION TRANSPARENCY - 4

- **Concurrency transparency:**
  - Concurrent use of **any** resource requires synchronization via locking
  - Transactions can be used
- **Failure transparency:**
  - Masking failures is one of the hardest issues in dist. systems
  - How do we tell the difference between a failed process and a very slow one?
  - When do we need to "fail over" to a replica?
  - Subject of chapter 8...

January 7, 2019
TCS558: Applied Distributed Computing [Winter 2019]  
Institute of Technology, University of Washington - Tacoma
L1.24

- Full distribution transparency may be impractical
- Communication latencies cannot be hidden
- Completely hiding failures of networks and nodes is impossible
  - Difference between slow computer and failing one
  - Transactions: did operation complete before crash?
- Full transparency will lead to slower performance:
  - Performance vs. transparency tradeoff
- Synchronizing replicas with a master requires time
- Immediately commit writes in fear of device failure

L1.25

- Abstracting location when user desires to interact intentionally interact with local resources / systems
- May be good
  - Location-based-services (find nearby friends)
  - Help a user understand what's going on
  - When a server doesn't respond for a long time – is it far away?
  - Users in different times zones?
- Can you think of examples where distribution is not hidden?
  - Eventual consistency
  - Many online systems no longer update instantaneously
  - Users are getting accustomed to delays

L1.26

- System with components that are easily used by, or integrated into other systems
- **Interfaces**: provide general syntax and semantics to interact with distributed components
- Services expose interfaces: functions, parameters, return values
- Semantics: describe what the services do
  - Often informally specified (via documentation)
- General interfaces enable alternate component implementations


L1.27

- **Interoperability**: ability for components from separate systems to work together (different vendors?)
  - Though implementation of a common interface
  - How could we measure interoperability of components?
- **Portability**: degree that an application developed for distributed system A can be executed without modification on distributed system B
  - How could we evaluate portability of a component?


L1.28




- **Extensible:** easy to reconfigure, add, remove, replace components from different developers
- Example: replace the underlying file system of a distributed system
- To be open, we would like to **separate policy from mechanism**
- Policy may change
- Mechanism is the technological implementation
- Avoid coupling policy and mechanism
- Enables flexibility

11.29




**CLOUD AND DISTRIBUTED SYSTEMS LAB**  
WES LLOYD, WLLOYD@UW.EDU,  
HTTP://FACULTY.WASHINGTON.EDU/WLLOYD



- **Serverless Computing (FaaS):**
  - *How should cloud native applications be composed from microservices to optimize performance and cost? Code structure directly influences hosting costs.* 
  - Service composition, performance and cost optimization/modeling/analytics, Application migration, Mitigation of Platform limitations, Influencing infrastructure, Lambda@Edge
- **Containerization (Docker):**
  - *How should containers and container platforms be leveraged and managed to optimize performance, reduce costs, and maximize server utilization?* 
  - Containers, container orchestration frameworks, resource allocation, checkpointing
- **Infrastructure-as-a-Service (IaaS) Cloud:**
  - *How should applications and workloads be deployed to optimize performance and cost? There are many "knobs", configuration options to consider.* 
  - Application/workload deployment, performance and cost optimization/modeling/analytics, infrastructure management, resource contention detection/mitigation, HW heterogeneity

QUESTIONS



January 7, 2019

TCCS558: Applied Distributed Computing [Winter 2019]  
Institute of Technology, University of Washington - Tacoma

L1.32