

# SER 321 A Session

**SI Session**

**Sunday, September 17th 2023**

*6:00 - 7:00 pm MST*

# Agenda



Distributed Algorithms

Main Hurdles

Approaches

Consensus

This Assignment

# SI Session Expectations

Thanks for coming to the **SER 321** SI session. We have a packed agenda and we are going to try to get through as many of our planned example problems as possible. This session will be recorded and shared with others.

- If after this you want to see additional examples, please visit the drop-in tutoring center.
- We will post the link in the chat now and at the end of the session.
  - [tutoring.asu.edu](https://tutoring.asu.edu)
- Please keep in mind we are recording this session and it will be made available for you to review 24-48 hours after this session concludes.
- Finally, please be respectful to each other during the session.

# Interact with us:

## Zoom Features



### Zoom Chat

- Use the chat feature to interact with the presenter and respond to presenter's questions.
- Annotations are encouraged

# SER 321

## Assignment 4

How was assignment 4?

Any issues you didn't figure out?

# SER 321

## Protocol Organization

The more organized

The easier for you!

```
# Protocol #
<!-- TOC -->
* [Protocol](#protocol-)
  * [Leader Responses](...)
  * [Client Responses](...)
    * [General](#general-)
    * [Specifics](...)
      * [Connected Signal](...)
      * [Menu Signal](...)
      * [Borrowed Signal](...)
      * [Returned Signal](...)
      * [Error Signal](...)
  * [Node Responses](...)
    * [General](...)
    * [Specifics](...)
      * [Connected Signal](...)
      * [Exit Signal](...)
      * [Available Signal](...)
      * [Release Signal](...)
      * [Borrow Signal](...)
      * [Return Signal](...)
      * [Update Signal](...)
      * [Listing Signal](...)
  * [Client Requests](...)
  * [General](...)
  * [Specifics](...)
    * [Start Request](...)
    * [Login Request](...)
    * [Borrow Request](...)
    * [Return Request](...)
    * [Exit Request](...)
  * [Node Requests](...)
    * [General](...)
    * [Specifics](...)
      * [Connect Signal](...)
      * [Waiting Signal](...)
      * [Error Signal](...)
      * [Available Result](...)
      * [Release Result](...)
      * [Borrow Result](...)
      * [Return Result](...)
      * [Update](#update-)
<!-- TOC -->
```

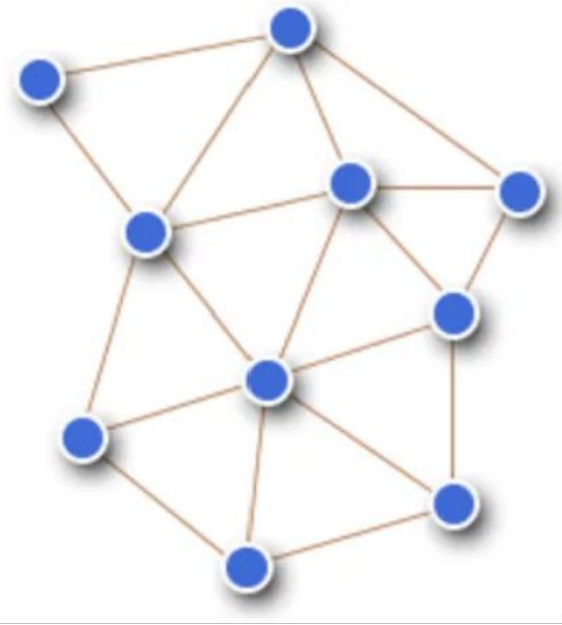
# SER 321

## Distributed Systems

Many nodes working together to form a **single system**

We can think of a node as an individual computer

Each node needs to communicate with the other nodes to form the single system as seen from the outside



# SER 321

## Distributed Systems

### Big Takeaways

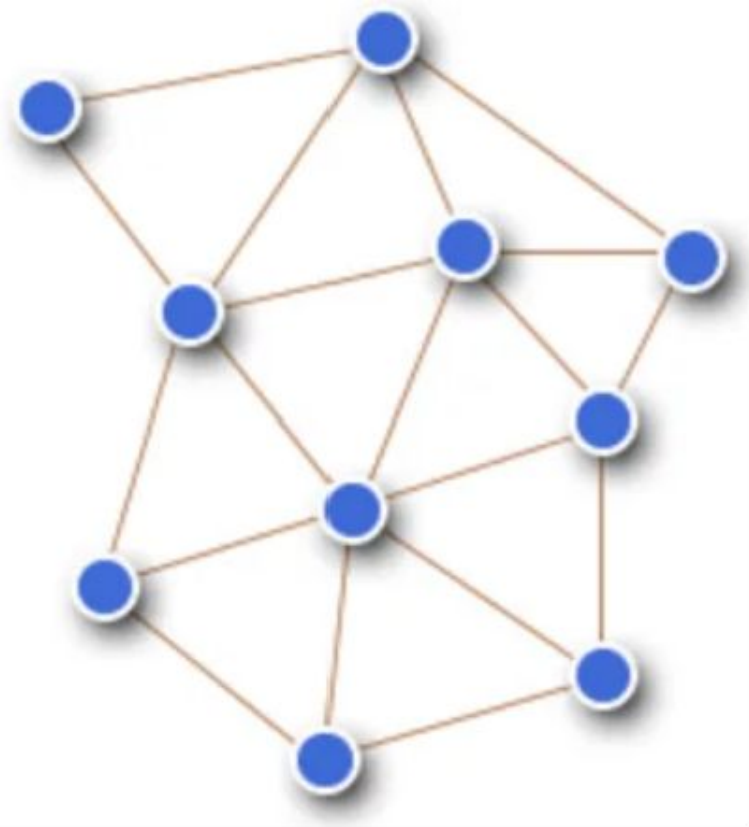
Nodes operate in *reality* not in a vacuum

- Nodes will fail

- The network will go down occasionally

- Sometimes a transmission will take forever

- Transmissions will not always take the same path



The nodes will experience problems and *we as the developers need to account for them*



# SER 321

## Distributed Systems

Distributed Algorithms can provide faster execution ***when done correctly***

Needs to be worth it!

- In terms of time/speed

- In terms of memory and overhead

Remember you also have to reconcile each result individually

- EX) sort array means you have to merge it all back together in the end!

# SER 321

## Distributed Systems

Maintain good programming practices to avoid common issues

- Handle Node failures
- Account for latency
- Account for network failures
- Protection of shared resources
- Prevention of deadlocks
- Execution safety - no errors or gross, bad stuff
- Ensuring liveness and that everyone eventually gets a turn

# SER 321

## Distributed Systems in Practice

So what do we do? How do we manage all this?

There are several ways of managing structure and addressing issues

- Main and Worker
- Peer to Peer
  
- Consensus algorithms
- Leader Elections

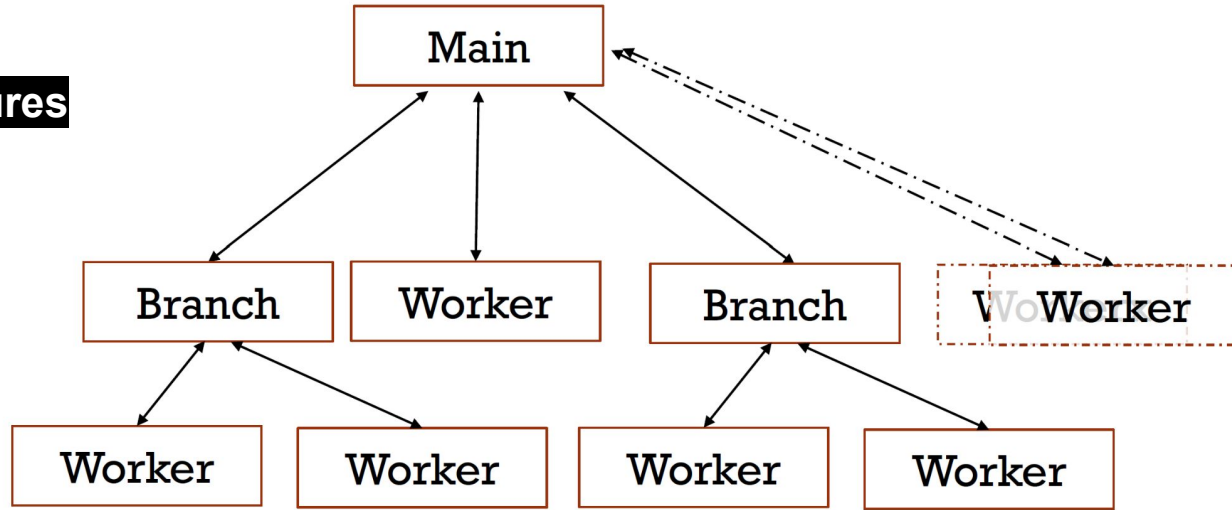
# SER 321

## Distributed Systems Structures

### Main and Worker

Pros & Cons

\_\_\_\_\_ construction!  
Communication is \_\_\_\_\_  
\_\_\_\_\_ scalable



Certain nodes have/are \_\_\_\_\_ than others  
\_\_\_\_\_ can be progressive and get out of \_\_\_\_\_

# SER 321

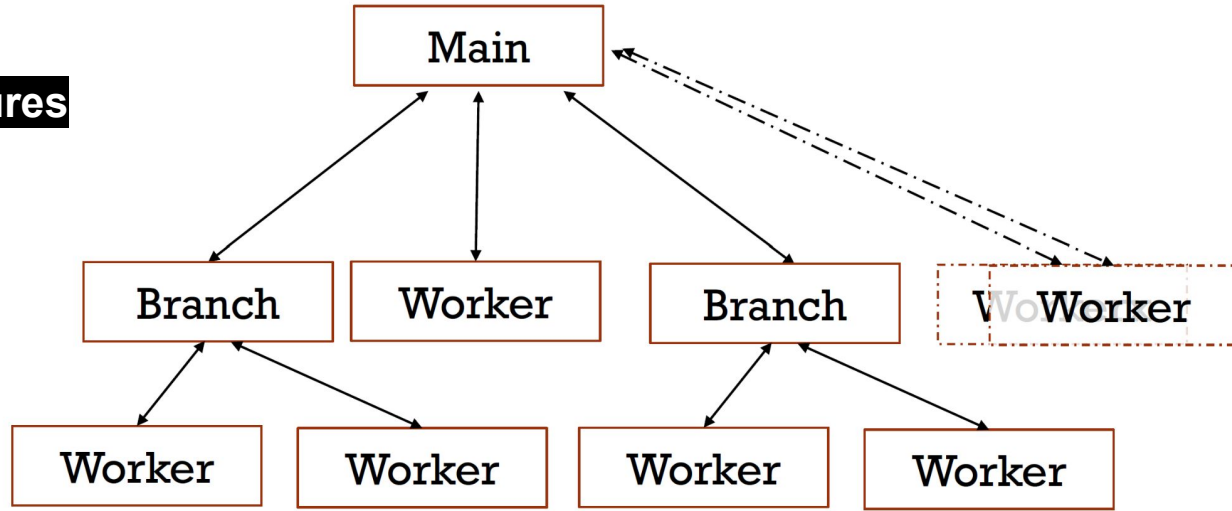
## Distributed Systems Structures

### Main and Worker

Pros & Cons

**Simple** construction!

Communication is \_\_\_\_\_  
\_\_\_\_\_ scalable



Certain nodes have/are \_\_\_\_\_ than others  
\_\_\_\_\_ can be progressive and get out of \_\_\_\_\_

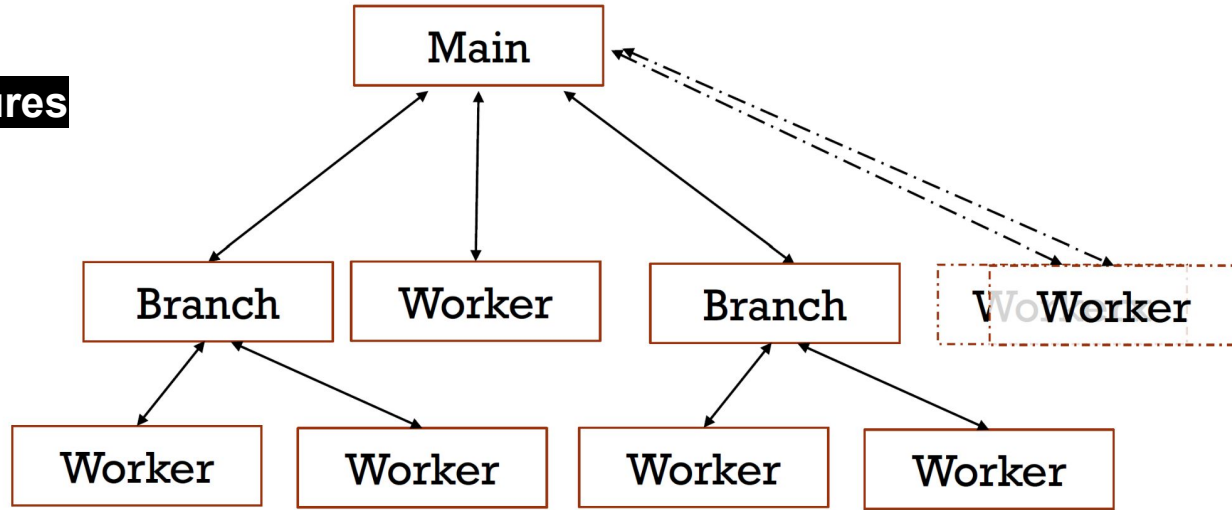
# SER 321

## Distributed Systems Structures

### Main and Worker

Pros & Cons

**Simple** construction!  
Communication is **linear**  
\_\_\_\_\_ scalable



Certain nodes have/are \_\_\_\_\_ than others  
\_\_\_\_\_ can be progressive and get out of \_\_\_\_\_

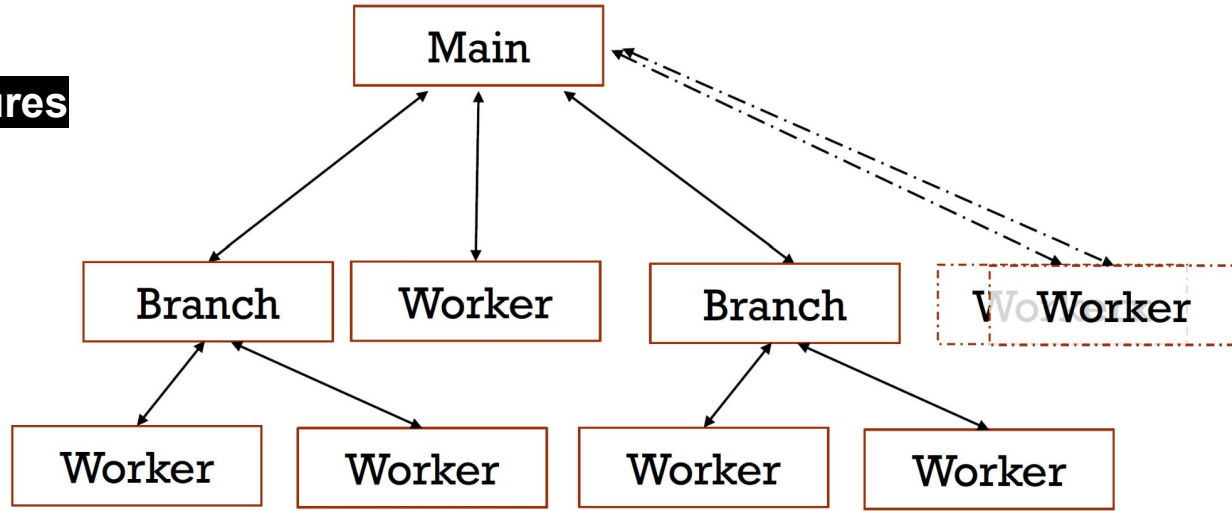
# SER 321

## Distributed Systems Structures

### Main and Worker

Pros & Cons

**Simple** construction!  
Communication is **linear**  
**Easily/Is** scalable



Certain nodes have/are \_\_\_\_\_ than others  
\_\_\_\_\_ can be progressive and get out of \_\_\_\_\_

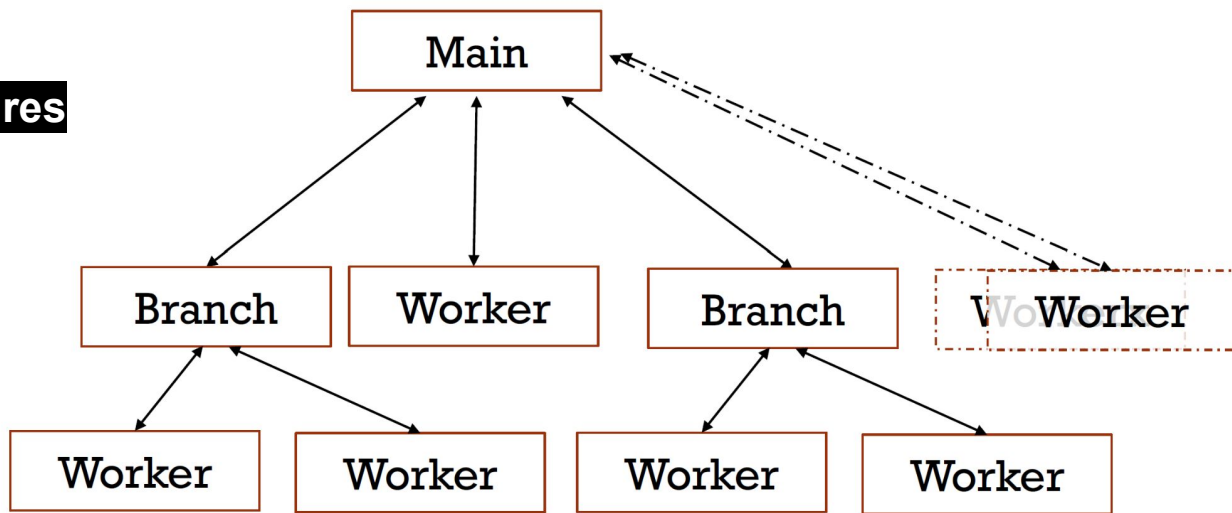
# SER 321

## Distributed Systems Structures

### Main and Worker

Pros & Cons

Simple construction!  
Communication is linear  
Easily/Is scalable



Certain nodes have/are more “value”/more important than others  
Failures can be progressive and get out of \_\_\_\_\_



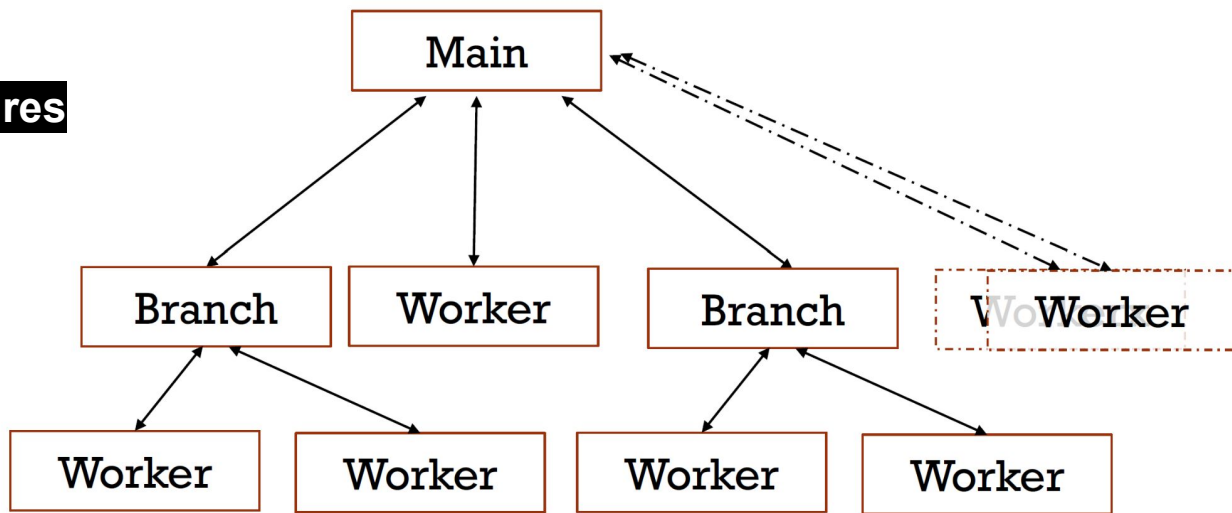
# SER 321

## Distributed Systems Structures

### Main and Worker

Pros & Cons

Simple construction!  
Communication is linear  
Easily/Is scalable



Certain nodes have/are more “value”/more important than others  
Failures can be progressive and get out of control

# SER 321

## Distributed Systems Structures

### Peer to Peer

Pros & Cons

All peers are \_\_\_\_\_

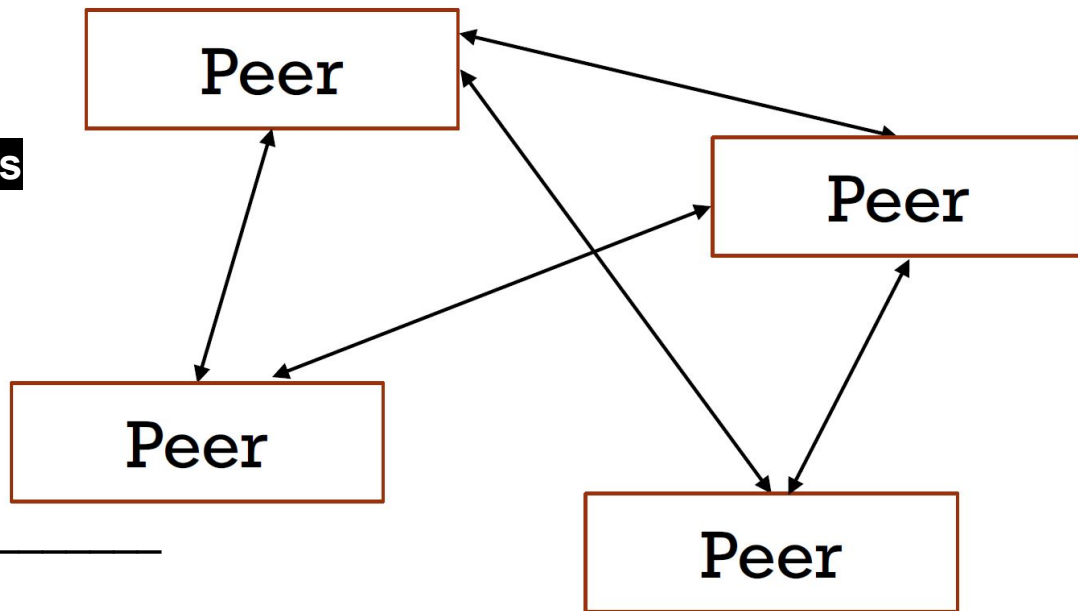
Failed peers are \_\_\_\_\_

Peers \_\_\_\_ join

\_\_\_\_ scalable

Communication is \_\_\_\_\_!

Handling \_\_\_\_ connections is \_\_\_\_\_



# SER 321

## Distributed Systems Structures

### Peer to Peer

Pros & Cons

All peers are **equal/connected**

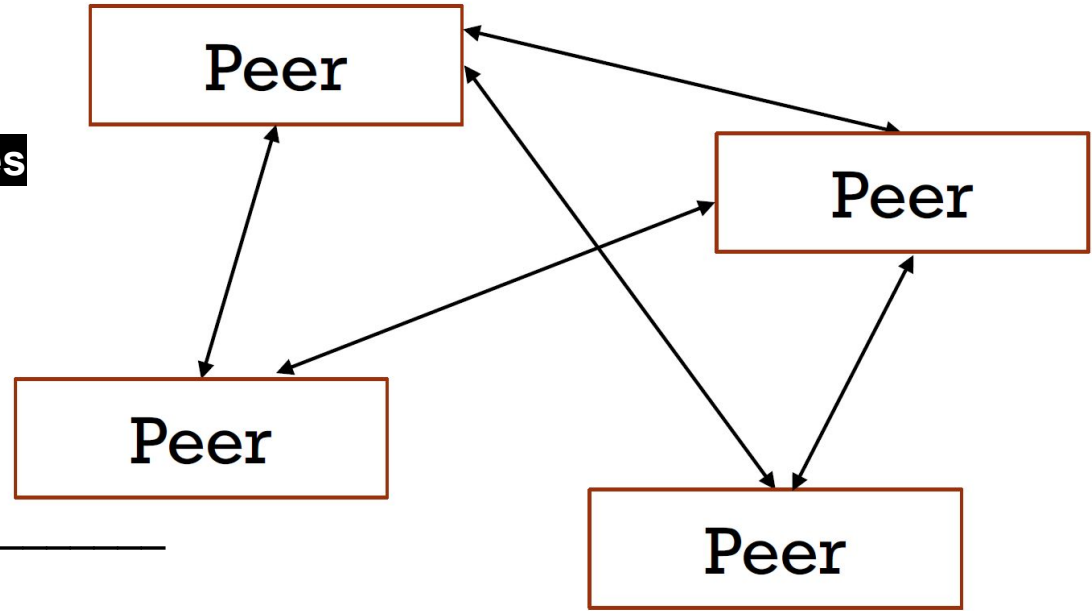
Failed peers are \_\_\_\_\_

Peers \_\_\_\_ join

\_\_\_\_ scalable

Communication is \_\_\_\_\_!

Handling \_\_\_\_ connections is \_\_\_\_\_



# SER 321

## Distributed Systems Structures

### Peer to Peer

Pros & Cons

All peers are **equal/connected**

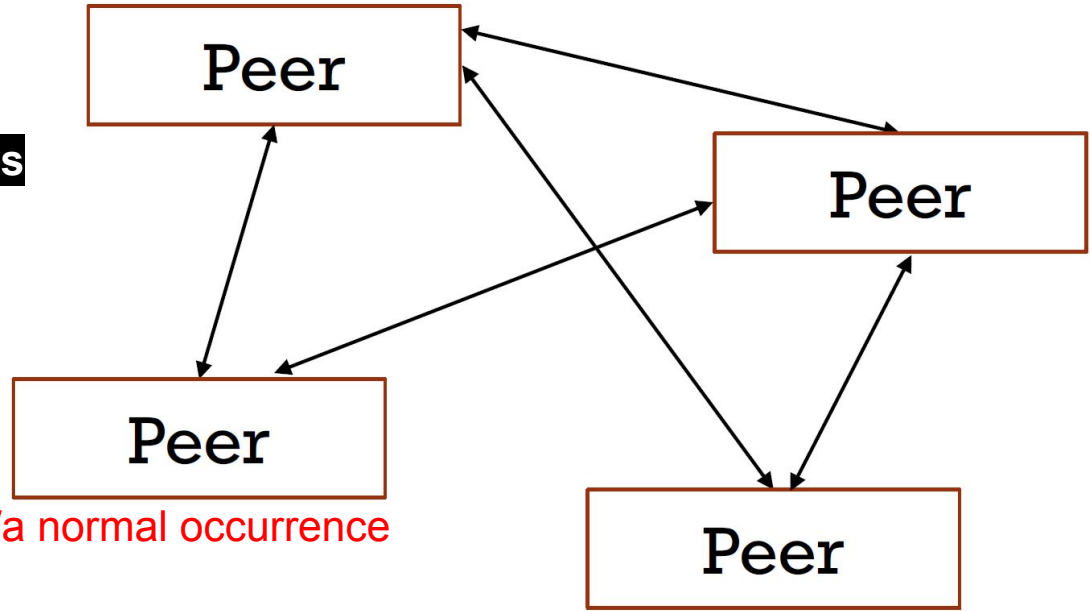
Failed peers are **easy to handle/a normal occurrence**

Peers \_\_\_\_ join

\_\_\_\_ scalable

Communication is \_\_\_\_\_!

Handling \_\_\_\_ connections is \_\_\_\_\_



# SER 321

## Distributed Systems Structures

### Peer to Peer

Pros & Cons

All peers are **equal/connected**

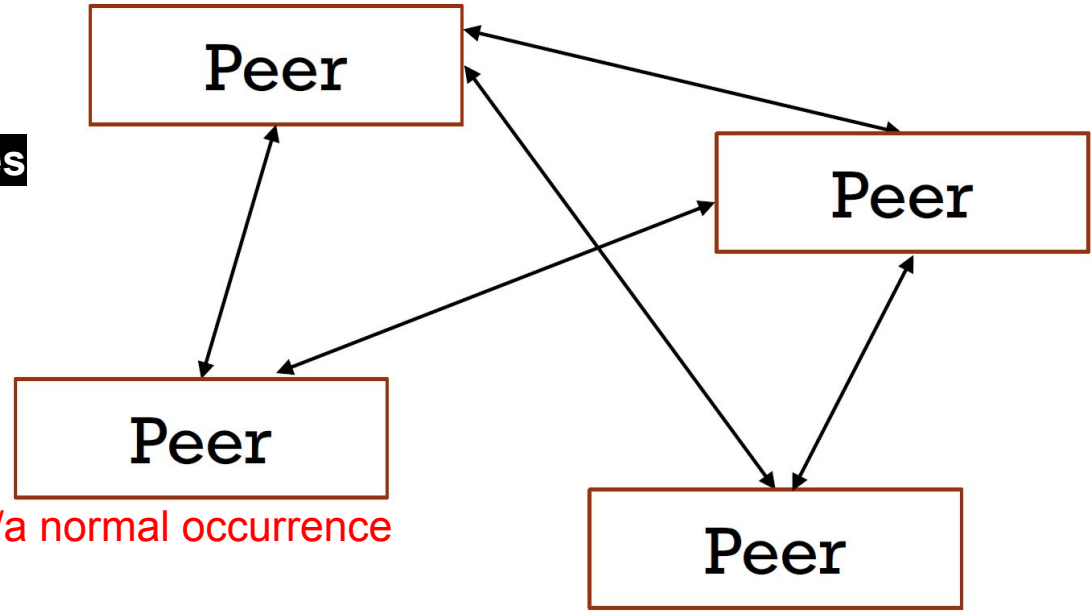
Failed peers are **easy to handle/a normal occurrence**

Peers **can** join

\_\_\_\_ scalable

Communication is \_\_\_\_\_!

Handling \_\_\_\_ connections is \_\_\_\_\_



# SER 321

## Distributed Systems Structures

### Peer to Peer

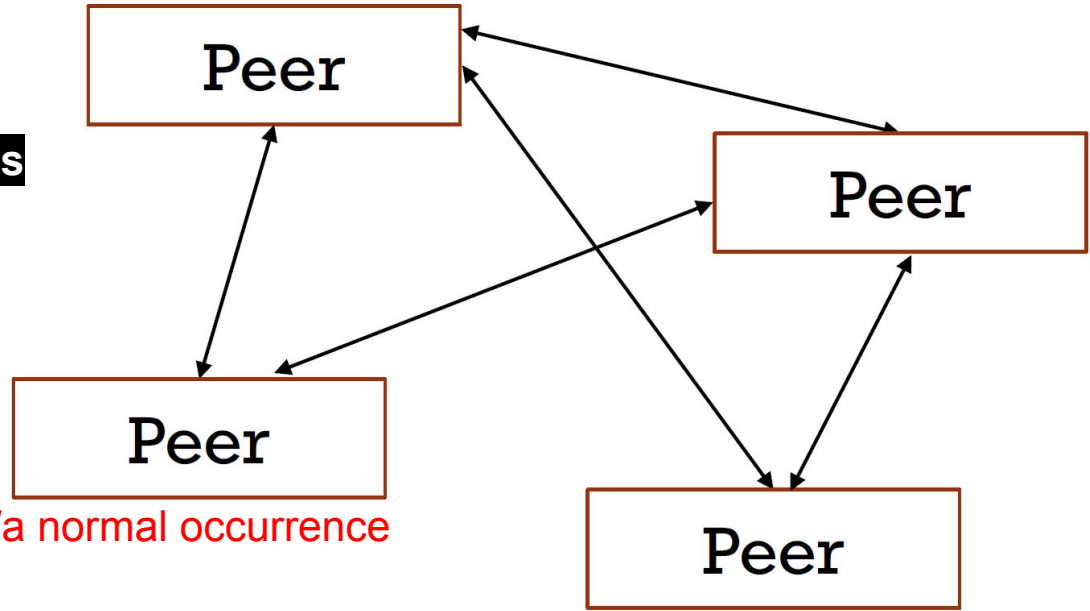
Pros & Cons

All peers are **equal/connected**

Failed peers are **easy to handle/a normal occurrence**

Peers **can** join

**Easily/Is** scalable



Communication is \_\_\_\_\_!

Handling \_\_\_\_\_ connections is \_\_\_\_\_

# SER 321

## Distributed Systems Structures

### Peer to Peer

#### Pros & Cons

All peers are **equal/connected**

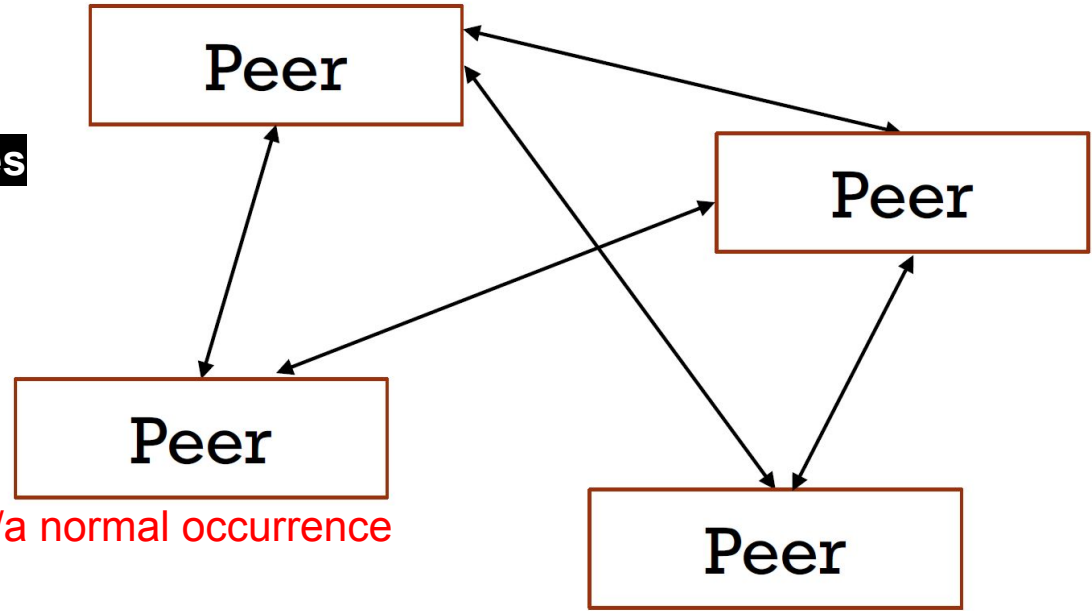
Failed peers are **easy to handle/a normal occurrence**

Peers **can** join

**Easily/Is** scalable

Communication is **complicated!**

Handling \_\_\_\_ connections is \_\_\_\_\_



# SER 321

## Distributed Systems Structures

### Peer to Peer

Pros & Cons

All peers are **equal/connected**

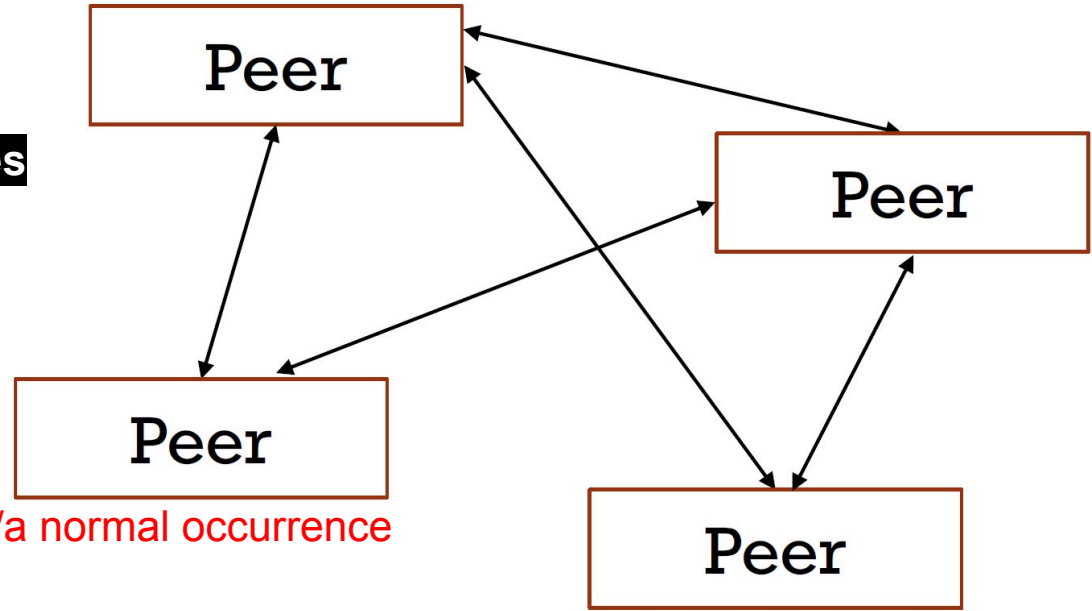
Failed peers are **easy to handle/a normal occurrence**

Peers **can** join

**Easily/Is** scalable

Communication is **complicated!**

Handling **client** connections is \_\_\_\_\_





# SER 321

## Distributed Systems Structures

### Peer to Peer

#### Pros & Cons

All peers are **equal/connected**

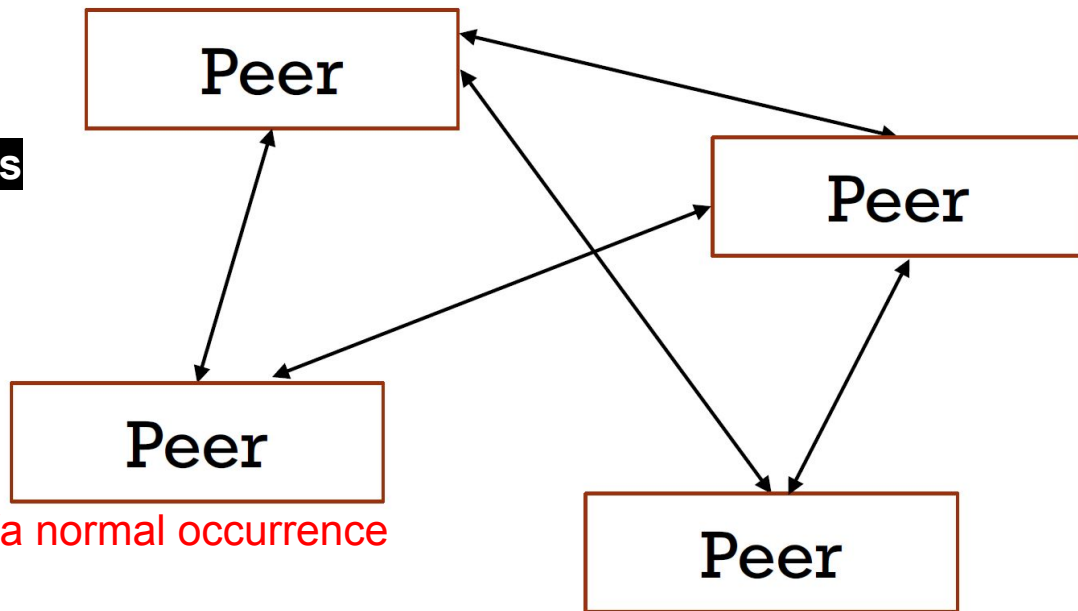
Failed peers are **easy to handle/a normal occurrence**

Peers **can** join

**Easily/Is** scalable

Communication is **complicated!**

Handling **client** connections is **different/more complex**



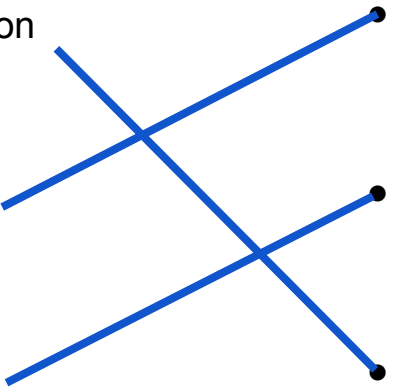
# SER 321

## Threading Pitfalls

- Race Condition
- Starvation
- Deadlock
- One thread is only able to acquire access to part its resources
- One thread never gets access to the resource it needs
- More than one thread accesses a single resource at one time

# SER 321

## Threading Pitfalls

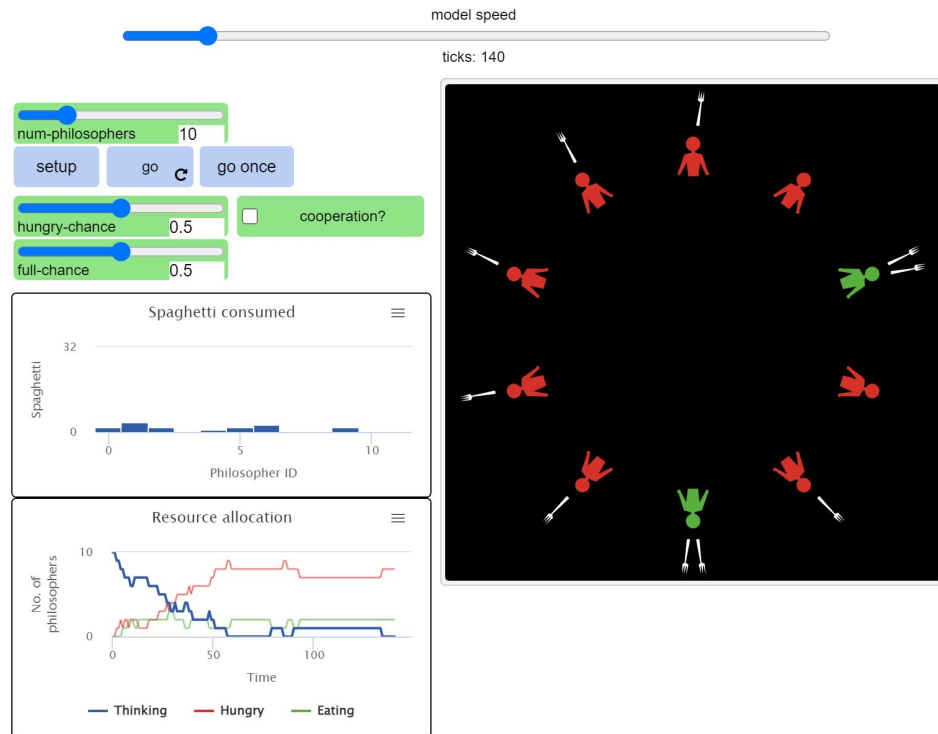
- Race Condition
  - Starvation
  - Deadlock
- 
- One thread never gets access to the resource it needs
  - One thread is only able to acquire access to part of its resources
  - More than one thread accesses a single resource at one time

# SER 321

## Dining Philosophers Again



## Interactive



# SER 321

## Consensus

Consensus is what keeps our nodes consistent

Allows a node to verify or validate something

Think of it like checking your work

I got  $x=42$ , what did you get?

Applies to all data for that node

Node *a/ways* checks with the other nodes before assuming a value is correct

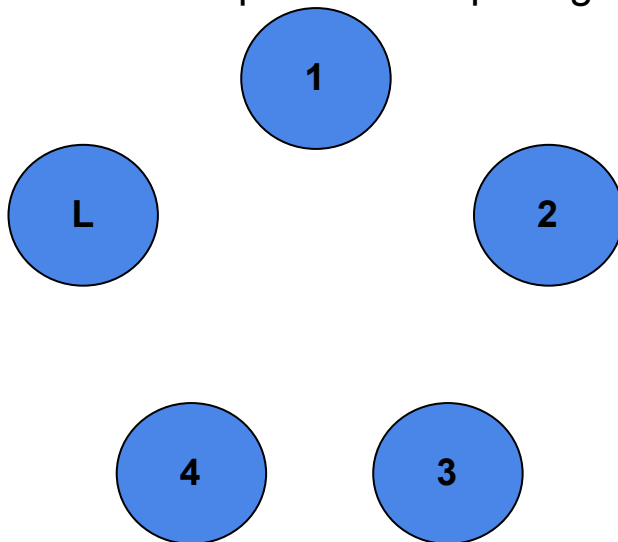
# SER 321

## What is a Leader

The **leader** is the node that is in charge

Leader Election algorithms exist with methods for choosing a leader

This leader is the node that is responsible for polling and checking all the other nodes



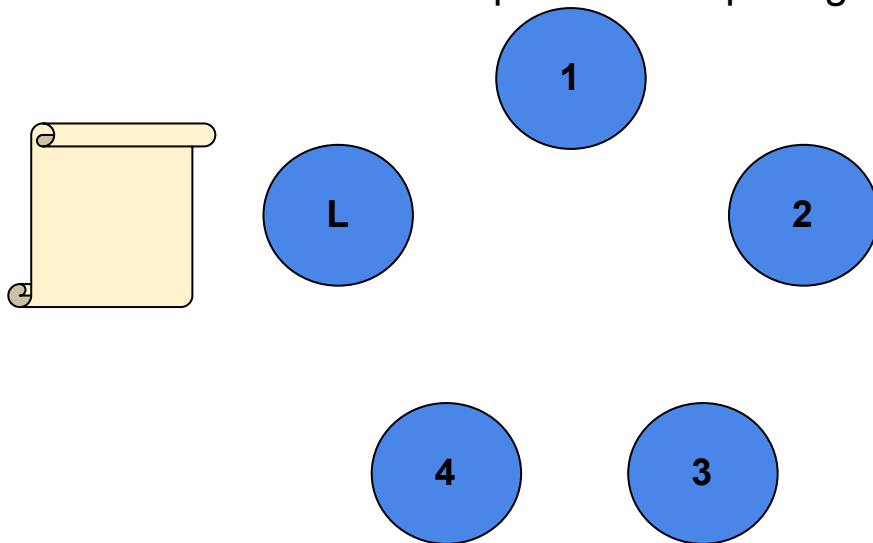
# SER 321

## What is a Leader

The **leader** is the node that is in charge

Leader Election algorithms exist with methods for choosing a leader

This leader is the node that is responsible for polling and checking all the other nodes

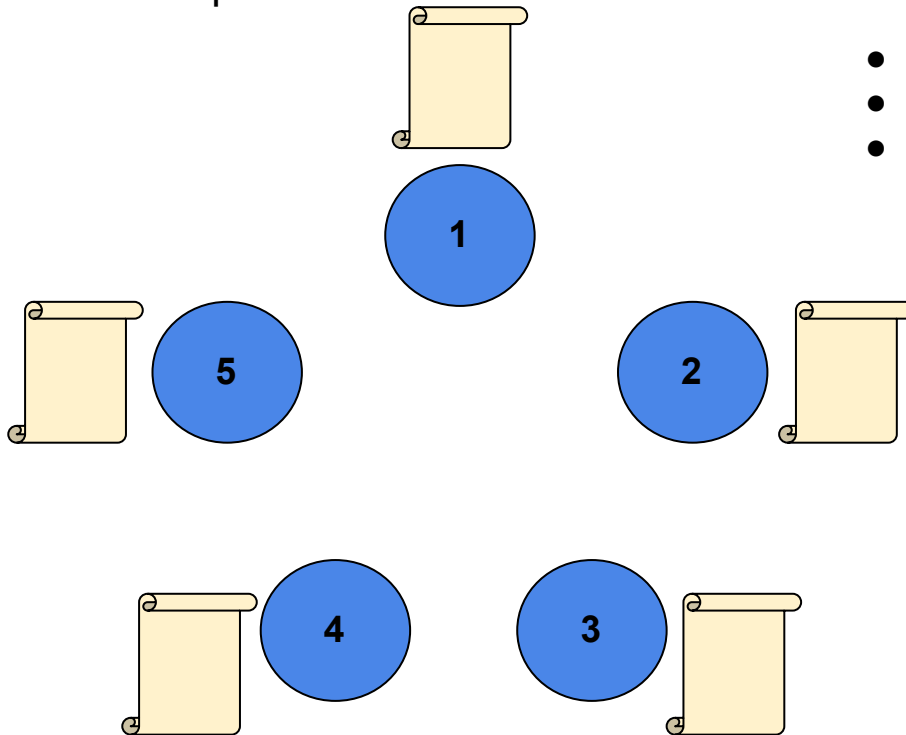


- Poll for consensus
- Determine verdict of consensus polls
- Verify node data matches leader data

# SER 321

## What about Peer to Peer with no leader?

Each node would be responsible its own data and consensus



- Poll other nodes for consensus
- Track poll responses
- Determine if data is valid

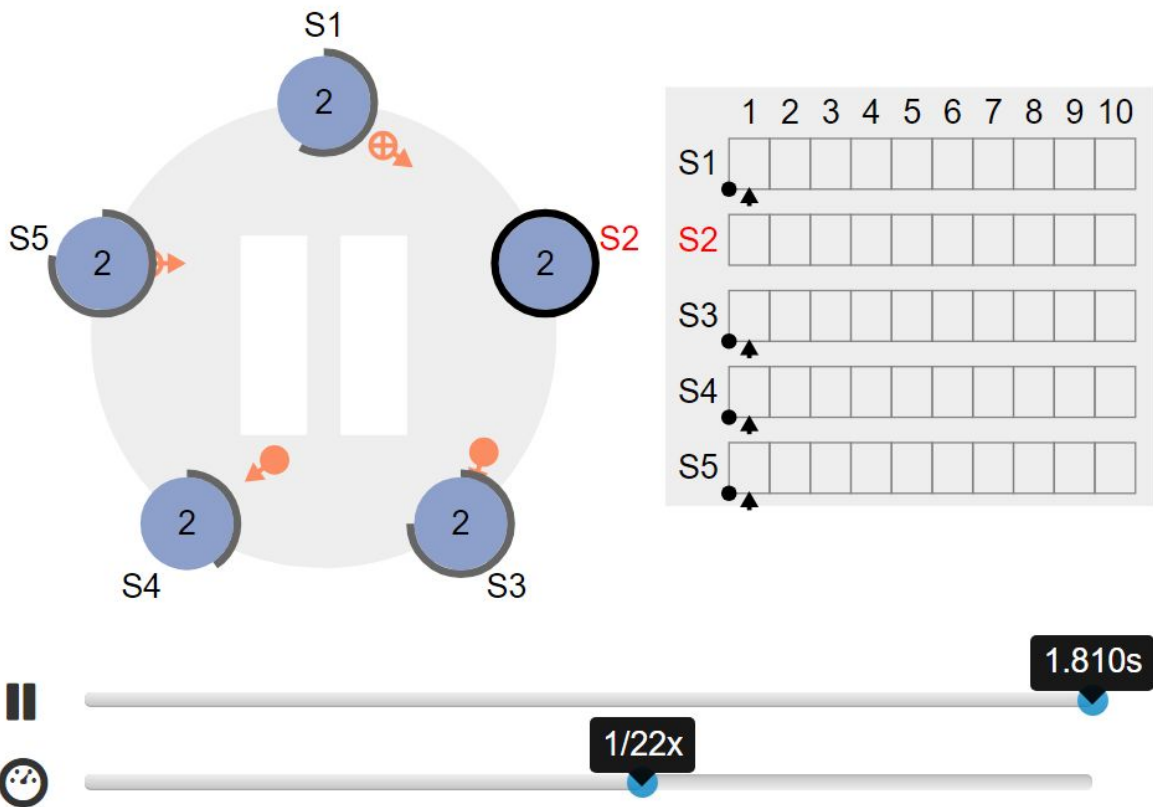


# SER 321

## Raft

Consensus Algorithm

Cool interactive [here](#)



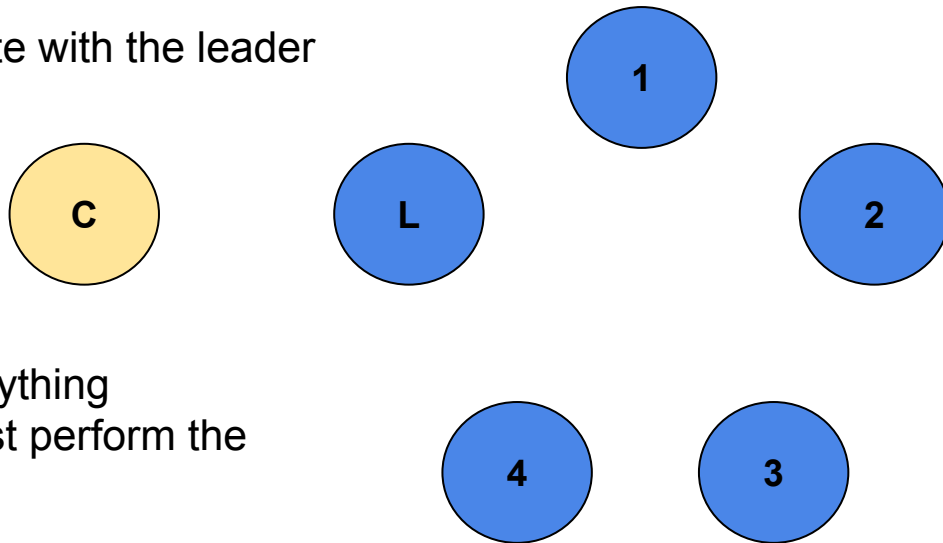
# SER 321

## Assignment

You are starting from scratch here - but don't be intimidated!

You are allowed to use code from the examples repo as a base to get you started

Your client will *only* communicate with the leader



The leader is in charge of everything

The nodes are workers that just perform the given task

# Questions?

## Survey:

[https://bit.ly/asn\\_survey](https://bit.ly/asn_survey)



## Upcoming Events

### SI Sessions:

- Tomorrow Monday September 18th 2023 6:00 pm MST

### Review Sessions:

- TBD

# More Questions?

Check out our other resources!

tutoring.asu.edu



## Academic Support

Academic Support Network (ASN) provides a variety of free services in-person and online to help currently enrolled ASU students succeed academically.

### Services



#### Subject Area Tutoring

Need in-person or online help with math, science, business, or engineering courses? Just hop into our Zoom room or drop into a center for small group tutoring. We'll take it from there.

[Need help using Zoom?](#)

[View the tutoring schedule](#)

[View digital resources](#)

Go to Zoom



#### Writing Tutoring

Need help with undergraduate or graduate writing assignments? Schedule an in-person or online appointment, access your appointment link, or wait in our drop-in queue.

[Access your appointment link](#)

[Access the drop-in queue](#)

Schedule Appointment



#### Online Study Hub

Join our online peer communities to connect with your fellow Sun Devils. Engage with our tools to search our bank of resources, videos, and previously asked questions. Or, ask our Tutorbot questions.

Now supporting courses in Math, Science, Business, Engineering, and Writing.

Online Study Hub

1-

Go to Zoom

2-

[Need help using Zoom?](#)

[View the tutoring schedule](#)

[View digital resources](#)



1. Click on 'Go to Zoom' to log onto our Online Tutoring Center.
2. Click on 'View the tutoring schedule' to see when tutors are available for specific courses.

# More Questions?

## Check out our other resources!

[tutoring.asu.edu/online-study-hub](https://tutoring.asu.edu/online-study-hub)

 **Academic Support Network**

 [Services](#)  [Faculty and Staff Resources](#) [About Us](#) 

[University College](#)

## Online Study Hub

Online peer communities for students and tutors, YouTube channels, and Tutorbots.



### What are online peer communities?

Individual courses have an online peer community that allows you to connect with your peers to post and answer questions and to develop study groups.



### How can tutoring center videos help?

Videos can help supplement the learning you're doing in and outside of class and include step-by-step methods for how to understand concepts.



### How does the Tutorbot work?

You can ask the Tutorbot questions about course concepts and the Tutorbot will recommend additional resources and examples to help address your questions.

Select a subject

- Any -

[Apply](#)



Academic Support Network



[Services](#) 

[Faculty and Staff Resources](#)

[About Us](#) 

[University College](#)

Select a subject

- Any -

[Apply](#)

Business

### ACC 231

Uses of Accounting Info I

 [Peer Community](#)


### ACC 241

Uses of Accounting Info II

 [Peer Community](#)

### CIS 105

Computer Applications and Information Technology

 [Peer Community](#)

Don't forget to check out the Online Study Hub for additional resources!

## Additional Resources

[Dining Philosophers Interactive](#)

[Raft Interactive](#)