2020

CSCI 49378: Lecture 2: Distributed Systems - Key Concepts & Techniques

Bonan Liu
CUNY Hunter College

NYC Tech-in-Residence Corps

Distributed Systems

Key Concepts & Techniques

Bonan Liu
Tech-In-Residence Member, Hunter College, CUNY
Spring 2020

Disclaimer

The content of this presentation is being provided for educational and informational purposes only. The views, thoughts, and opinions expressed in this presentation belong solely to the author, and not necessarily to the author's employer.

The content of this presentation is not endorsed by the author's employer.

Agenda

- Distributed systems basics
- Basic elements in distributed systems
- Communications in distributed systems
- Common techniques in distributed systems

What is a distributed system

A distributed system is a system whose components are located on different networked computers, which communicate and coordinate their actions by passing messages to one another.[1]

[1] M. van Steen and A.S. Tanenbaum, Distributed Systems, 3rd ed., distributed-systems.net, 2017.

Why do we need distributed systems

- Imagine you are a retail store owner and want to develop a ecommerce site.
- You have a server running standard LAMP environment.
- What are you going to do when the business grows?

Why do we need distributed systems

Scalability is the property of a system to handle a growing amount of work by adding resources to the system.[2]

- Vertical Scaling: adding more resources to a single node
- Horizontal Scaling: adding more nodes in the system

[2] Bondi, André B. (2000). Characteristics of scalability and their impact on performance. Proceedings of the second international workshop on Software and performance.

Why do we need distributed systems

- Fault Tolerance — a distributed system with multiple computing nodes is more fault-tolerant than a single machine.
- Low Latency — The time for a network communication is physically bounded by physics.

Challenges in distributed systems

The fundamental challenge is the asynchronous nature of distributed systems.

- Consensus Problem
- Fault Tolerance Problem
- Clock Problem
- ...and more

Examples of distributed systems

- Distributed Applications/API Frontend
- Distributed Data Storage System
- High-throughput Data Pipelines
- Blockchains

Could you please give an example of distributed systems?

Basic elements in distributed systems

Hardware Layer:

- Machine
- Rack
- Data Center

Software Layer:

- Thread
- Process
- Web Service

Basic elements in distributed systems

Logical Elements

- Nodes/Cells/Clusters
- Zones/Regions
- Tasks/Jobs
- Servers/Ports

Basic elements in distributed systems

Review the tiny URL example.

What are the basic elements in the example?

Communications in distributed systems

- RESTful API
- Streaming API
- Remote Procedure Call (RPC)
- Inter-Process Call (IPC)
- Multi-Thread Communication

Communications in distributed systems

Review the tiny URL example.

What are the communication methods in the example?

CAP Theorem

It is **impossible** to provide the more than two of the following three guarantees:

- Consistency: always read the latest data
- Availability: always receive a non-error response (the data in the response may or may not be latest)
- Partition tolerance: the system can continue process requests despite the failures

Performance evaluation of distributed systems

- Requests/Queries per second: RPS/QPS
- Availability
  - 3-nine: 8.7 hrs downtime per year
  - 4-nine: 52 mins downtime per year
  - 5-nine: 5 mins downtime per year
- Latency
- Throughput

Primary/Replica model (Master/Slave model)

One node (primary) serves as the source of the truth while the other nodes (replicas) keep copying from the primary node

Master/Worker model

One special node (task master) allocates the work while the other nodes (worker) complete the actual work.

Sharding

Sharding is a technique to break up the large amount of data or large amount of computation requests into smaller pieces and store or process them separately.

Consistent Hashing

- Map the hash space into a loop (ring)
- Hash the target data/request
- Hash the node
- Always allocate the target data/request to the nearest node

Two-Phase Commit

Assumption:

- Each node can stabelly write logs in local storage
- A node serves as the coordinator of the system

Commit-Request Phase:

- Coordinator send queries to each participants
- Participants vote

Commit Phase:

- Participants act per Coordinator’s second request

Readings

The Role of Distributed State, J. Ousterhout, 1991.

Scale and performance in a distributed file system, Howard, Kazar, Menees, Nichols, Satyanarayanan, Sidebotham, and West. ACM Transactions on Computer Systems (Feb 1988).

Homework

- Sign up Google Cloud Platform (strongly recommended) or the cloud service you preferred.
- Complete before 10/7/2019.