DISTRIBUTED AND CLOUD COMPUTING

ENCT 411

Lecture : 3 **Year: IV**
Tutorial : 1 **Part: I**
Practical : 1.5

Course Objectives:
The objective of this course is to provide fundamental concepts, principles and design techniques of distributed and cloud computing systems. It focuses on architectural and communication models, synchronization, replication, consistency and consensus, distributed storage and data processing, cloud service models, virtualization, containers and cloud-native systems, and fault tolerance.

1 Introduction to Distributed Systems (4 hours)

1.1 Distributed systems: definition, characteristics, Importance and Applications
1.2 Types of distributed systems: high-performance computing, information and pervasive systems
1.3 Properties and Design Goals: transparency, heterogeneity, openness, reliability and scalability
1.4 Architectural models: client-server, peer-to-peer, multi-tier and event-based
1.5 Fundamental models: interaction, failure and security models
1.6 Design challenges and fallacies of distributed computing; overview of real-world distributed systems

2 Communication in Distributed Systems (6 hours)

2.1 Distributed object concepts
2.2 Middleware system concepts
2.3 Remote Method Invocation (RMI) and Remote procedure call (RPC)
2.4 Modern remote invocation: RESTful web services and gRPC
2.5 Data serialization and marshalling: XML, JSON, Protocol Buffers
2.6 Message-oriented middleware: message queues and publish-subscribe systems (RabbitMQ, MQTT)

3 Synchronization and Coordination (6 hours)

3.1 Need for synchronization; physical clock synchronization: Cristian's algorithm, Berkeley algorithm, NTP
3.2 Logical clocks: Lamport clocks and vector clocks, event ordering and causality
3.3 Global state and distributed snapshots: Chandy-Lamport algorithm
3.4 Distributed mutual exclusion: centralized, Ricart-Agrawala, token ring and Maekawa algorithms
3.5 Election algorithms: bully and ring
3.6 Coordination services: ZooKeeper and etcd

4 Replication, Consistency and Consensus (6 hours)

4.1 Replication: motivation, Passive, Active, multi-leader and leaderless replication
4.2 Data-centric consistency models: linearizability, sequential and causal consistency
4.3 Client-centric consistency models: eventual consistency, read-your-writes, monotonic reads
4.4 Quorum-based protocols and replica management
4.5 Distributed transactions: two-phase commit and three-phase commit
4.7 Consensus problem, Paxos

5 Distributed Storage and Data Processing (5 hours)

5.1 Storage models: file, block and object storage
5.2 Distributed file systems: design issues, NFS, Google File System (GFS), Hadoop Distributed File System (HDFS)
5.3 Software-defined storage: Ceph; copy-on-write and snapshots (ZFS)
5.4 Concurrency control approach: Lock, optimistic concurrency control, timestamp-based concurrency control
5.5 Distributed data processing: Overview of MapReduce programming model, Hadoop YARN, Apache Spark

6 Cloud Computing (5 hours)

6.1 Cloud computing: definition, essential characteristics and reference architecture
6.2 Service models (IaaS, PaaS, SaaS, FaaS) and deployment models (public, private, community, hybrid)
6.3 Cloud data center architecture: regions, availability zones, software-defined networking and storage
6.4 Elasticity and scalability: load balancing, horizontal and vertical scaling, auto-scaling
6.5 Cloud security and governance: shared responsibility model, identity and access management (IAM), data encryption, compliance
6.5 Service level agreements (SLA), pricing models, cloud economics and vendor lock-in

7 Virtualization, Containers and Cloud-Native Systems (5 hours)

7.1 Virtualization: full, para- and hardware-assisted virtualization
7.2 Hypervisors: type 1 and type 2 (KVM, Xen, ESXi); server, storage and network virtualization
7.3 Containers: namespaces, cgroups, containers versus virtual machines
7.4 Kubernetes: architecture, pods, deployments, services, scaling and self-healing
7.5 Microservices architecture: API gateway, service discovery and service mesh
7.6 Cloud-native application design: twelve-factor app, DevOps, CI/CD and infrastructure as code

8 Fault Tolerance and Reliability (4 hours)

8.1 Dependability concepts: availability, reliability, safety, maintainability, MTTF and MTTR

8.2 Fault, error and failure; failure models: crash, omission, timing and Byzantine fault tolerance
8.3 Failure detection and process resilience: process resilience, redundancy techniques
8.4 Checkpointing, message logging and rollback recovery
8.5 High availability and disaster recovery: failover, geo-redundancy, RPO and RTO

9 Edge Computing, Emerging Trends and Case Studies (4 hours)

9.1 Edge and fog computing: architecture and the IoT-edge-cloud continuum
9.2 Task offloading, migration and edge-cloud collaboration models
9.3 AI-driven cloud optimization (ML-based auto-scaling) and distributed systems for large-scale AI training and inference
9.4 Hybrid and multi-cloud strategies
9.6 Case studies: Google (GFS, Bigtable, MapReduce, Spanner), Amazon (Dynamo, AWS global infrastructure), Netflix microservices architecture

Tutorial (15 hours)

1. Comparison of architectural models (client-server, peer-to-peer, multi-tier, event-based) with examples; types of transparency
2. Comparison of RPC, REST, gRPC and message-oriented communication; RPC message flow with stubs and marshalling; JSON versus Protocol Buffers
3. Clock synchronization problems: Cristian's, Berkeley and NTP offset and delay calculations
4. Lamport and vector timestamps on event diagrams, happened-before relation, Chandy-Lamport snapshot
5. Tracing mutual exclusion (Ricart-Agrawala, token ring, Maekawa) and election (bully, ring) algorithms with message complexity
6. Identification of consistency models from execution histories; CAP and PACELC classification of real systems
7. Two-phase and three-phase commit message flow with failure scenarios; Raft leader election and log replication
8. Consistent hashing, data partitioning, replica placement and quorum (N, R, W) calculations
9. MapReduce job design (word count, inverted index); HDFS read and write flow
10. Selection of cloud service and deployment models for given scenarios; SLA and cost estimation
11. Comparison of virtual machines and containers; decomposition of a monolithic application into microservices and Kubernetes objects
12. Availability calculations for series and parallel systems, RPO/RTO, and design of a fault-tolerant, highly available edge-cloud architecture

Practical (22.5 hours)

1. Client-server communication using sockets (TCP/UDP, multi-threaded server) and RPC (gRPC or Java RMI)
2. Implementation of Lamport logical clocks and vector clocks across multiple processes

3. Simulation of distributed mutual exclusion (Ricart-Agrawala or token ring) and election (bully or ring) algorithms
4. Message-oriented middleware and coordination: publish-subscribe using Kafka, RabbitMQ or MQTT; leader election and distributed locks using ZooKeeper or etcd
5. Setting up a multi-node HDFS cluster and running MapReduce and Spark jobs
6. Containerization: building Docker images, multi-container applications using Docker Compose and private registries
7. Deployment of a microservice application on Kubernetes (minikube, kind or equivalent): deployments, services, scaling, rolling updates and self-healing
8. Cloud deployment on a public cloud (AWS, Azure or GCP free tier) or OpenStack: provisioning virtual machines, storage and serverless functions using infrastructure as code (Terraform), with monitoring, auto-scaling and fault-injection testing

Final Exam

The questions will cover all the chapters in the syllabus. The evaluation scheme will be as indicated in the table below:

Chapter	Hours	Marks distribution*
1	4	6
2	6	6
3	6	8
4	6	10
5	5	6
6	5	6
7	5	6
8	4	6
9	4	6
Total	**45**	**60**

* There may be minor deviations in marks distribution.

References

1. Coulouris, G., Dollimore, J., Kindberg, T., Blair, G. (2011). Distributed systems: Concepts and design (Latest ed.). Addison-Wesley.
2. van Steen, M., Tanenbaum, A.S. (2023). Distributed systems (Latest ed.). distributed-systems.net.
3. Hwang, K., Fox, G.C., Dongarra, J.J. (2012). Distributed and cloud computing: From parallel processing to the internet of things. Morgan Kaufmann.
4. Buyya, R., Broberg, J., Goscinski, A. (2011). Cloud computing: Principles and paradigms. Wiley.
5. Kleppmann, M. (2017). Designing data-intensive applications. O'Reilly Media.
6. Erl, T., Cope, R., Naserpour, A. (2015). Cloud computing design patterns. Prentice Hall.
7. Burns, B., Beda, J., Hightower, K., Evenson, L. (2022). Kubernetes: Up and running (Latest ed.). O'Reilly Media.