Distributed Systems Essay

Nuno Vivas
Francisco Santos

Master's Degree in IT Engineering, FCT – University of Algarve

February 21, 2026

CONTENTS

***Abstract*—This document provides the initial draft structure for a distributed systems essay, covering core topics from architectures to security. Each section is organized in a separate file to support collaborative writing and incremental refinement.**

I. Introduction

Distributed systems are a foundational pillar of modern computing, enabling applications to scale across multiple machines while serving users with high availability and low latency. This essay surveys essential concepts, design principles, and practical trade-offs in distributed systems.

A. Context and Motivation

Draft notes: Explain why distributed systems are critical in cloud platforms, data-intensive applications, and large-scale services.

B. Objectives and Scope

Draft notes: State the objectives of this essay and the boundaries of the discussion.

C. Document Structure

Draft notes: Briefly summarize what each major section contributes.

II. System Models and Transparency

This section introduces the assumptions and abstractions used to reason about distributed systems.

A. System and Failure Models

Draft notes: Define synchronous, asynchronous, and partially synchronous models, and relate them to failure assumptions.

B. Distribution Transparency

Draft notes: Discuss access, location, migration, replication, and failure transparency.

C. Heterogeneity and Openness

Draft notes: Explain interoperability concerns and the role of open interfaces and standards.

III. Architectures

This section introduces the main architectural styles used in distributed systems and compares their strengths and limitations in practice.

A. Client–Server and Multi-Tier Models

Draft notes: Describe the traditional client–server approach, n-tier decomposition, and typical use cases in enterprise systems.

B. Peer-to-Peer and Hybrid Architectures

Draft notes: Explain decentralized approaches, scalability implications, and trade-offs in control and reliability.

C. Microservices and Service-Oriented Architectures

Draft notes: Discuss loose coupling, independent deployment, and operational complexity in modern distributed applications.

IV. Processes

This section focuses on process models, concurrency, and lifecycle management in distributed environments.

A. Process Models and Threads

Draft notes: Define processes and threads, and compare performance and isolation considerations.

B. Process Creation and Management

Draft notes: Explain how distributed applications start, monitor, and terminate processes across nodes.

C. Scheduling and Resource Usage

Draft notes: Analyze how CPU, memory, and I/O constraints affect process behavior and system throughput.

V. Communication

This section presents communication paradigms and protocols that enable interaction among distributed components.

A. Synchronous and Asynchronous Communication

Draft notes: Compare request/response communication with message-based interaction and event-driven designs.

B. Remote Invocation and Messaging

Draft notes: Cover RPC/RMI/gRPC, message queues, and publish-subscribe mechanisms.

C. Latency, Throughput, and Reliability

Draft notes: Discuss communication overheads, bottlenecks, retries, and delivery guarantees.

VI. Coordination

This section studies how distributed processes cooperate to achieve correct global behavior.

A. Time, Ordering, and Logical Clocks

Draft notes: Introduce physical clocks, Lamport clocks, and vector clocks for reasoning about event order.

B. Mutual Exclusion and Leader Election

Draft notes: Present algorithms and trade-offs for controlling shared resources and selecting coordinators.

C. Consensus and Agreement

Draft notes: Explain the role of consensus protocols (e.g., Paxos/Raft) and their practical constraints.

VII. Naming

This section explores naming, discovery, and location transparency in distributed systems.

A. Name Spaces and Identifiers

Draft notes: Describe global vs local naming, unique identifiers, and hierarchical naming schemes.

B. Name Resolution and Directory Services

Draft notes: Explain DNS-like approaches, directory services, and service registries.

C. Mobility and Dynamic Binding

Draft notes: Discuss how systems handle changing locations of services and resources.

VIII. Consistency and Replication

This section analyzes why replication is used and how consistency guarantees are defined and implemented.

A. Replication Strategies

Draft notes: Compare active/passive replication, leader-based replication, and multi-leader models.

B. Consistency Models

Draft notes: Present strong, eventual, and causal consistency with practical examples.

C. CAP and Design Trade-offs

Draft notes: Discuss availability-partition tolerance trade-offs and implications for system design choices.

IX. Scalability and Performance

This section evaluates how distributed systems maintain efficiency as users, data, and workloads grow.

A. Dimensions of Scalability

Draft notes: Analyze size, geographic, and administrative scalability.

B. Performance Metrics

Draft notes: Define latency, throughput, response time percentiles, and utilization.

C. Optimization Techniques

Draft notes: Cover caching, load balancing, partitioning, and batching.

X. Fault Tolerance

This section examines how distributed systems detect, isolate, and recover from failures.

A. Failure Models

Draft notes: Define crash, omission, timing, and Byzantine failures in distributed contexts.

B. Detection and Recovery Mechanisms

Draft notes: Explain heartbeats, checkpointing, rollback recovery, and failover procedures.

C. Redundancy and Availability Engineering

Draft notes: Discuss replication, redundancy planning, and service-level objectives.

XI. Security

This section addresses key security concerns and controls for distributed systems.

A. Threat Model and Attack Surface

Draft notes: Identify common threats, trust boundaries, and security assumptions.

B. Authentication, Authorization, and Confidentiality

Draft notes: Cover identity management, access control, encryption in transit and at rest.

C. Integrity, Auditing, and Secure Operations

Draft notes: Discuss integrity checks, logging, monitoring, and incident response practices.

XII. Conclusion

Distributed systems require balancing competing goals such as performance, consistency, availability, and security. The concepts presented in this essay provide a foundation for evaluating real-world architectures and making informed engineering decisions.

A. Key Takeaways

Draft notes: Summarize the most important principles and trade-offs discussed.

B. Practical Implications

Draft notes: Relate the content to engineering decisions in production environments.

C. Future Work

Draft notes: Identify topics for deeper analysis (e.g., edge computing, data governance, and AI-driven operations).

References

[1] A. S. Tanenbaum and M. van Steen, *Distributed Systems: Principles and Paradigms*. Pearson, 2017.