Assignment No. 07

Semester	B.E. Semester VIII – Computer Engineering
Subject	Distributed Computing Lab
Subject Professor In-charge	Dr. Umesh Kulkarni
Assisting Professor	Prof. Prakash Parmar
Academic Year	2024-25

Student Name	Deep Salunkhe
Roll Number	21102A0014

Title: Trade-offs in replication policy

Replication in Distributed Systems

1. Introduction to Replication in Distributed Systems

Definition of Replication

Replication in distributed systems refers to the process of creating and maintaining multiple copies of data across different nodes to enhance system reliability, availability, and performance.

Necessity of Replication

Replication is crucial in distributed systems for the following reasons:

- **Fault Tolerance**: Ensures data availability even in case of node failures.
- **Availability**: Allows users to access data even if some nodes are unreachable.
- **Load Balancing**: Distributes read and write operations across multiple nodes, reducing bottlenecks and improving response times.

Types of Replication

1. **Full Replication**: Every node maintains a complete copy of the dataset.

2. **Partial Replication**: Only a subset of the data is replicated across nodes.
3. **Selective Replication**: Critical data is replicated more frequently than less essential data.

2. Replication Policies and Their Trade-offs

A. Synchronous vs. Asynchronous Replication

Synchronous Replication

- Ensures **strong consistency** by updating all replicas before confirming a transaction.
- **Trade-offs**:
    - **High consistency** but increases **latency**.
    - Susceptible to **network failures**, reducing availability.

Asynchronous Replication

- Updates replicas in the background, leading to **eventual consistency**.
- **Trade-offs**:
    - **Lower latency** but can result in **stale reads**.
    - Improves **fault tolerance** since operations proceed without waiting for acknowledgments.

B. Primary-Backup (Passive) vs. Multi-Primary (Active) Replication

Primary-Backup Replication

- A single **primary** node processes writes and propagates changes to backup nodes.
- **Trade-offs**:
    - **Easier consistency management**.
    - **Single point of failure** unless failover mechanisms are implemented.

Multi-Primary Replication

- Multiple nodes accept writes, requiring conflict resolution mechanisms.

- **Trade-offs**:
    - **Higher availability** but **increased complexity** in maintaining consistency.
    - More resilient to node failures but may introduce **data conflicts**.

C. Read-Only vs. Read-Write Replication

Read-Only Replication

- Data is replicated for read operations, reducing load on primary databases.
- **Use Case**: Content delivery networks (CDNs), caching systems.
- **Trade-offs**:
    - **Fast reads**, but **stale data** if updates are not synchronized quickly.

Read-Write Replication

- Supports both read and write operations across replicas.
- **Trade-offs**:
    - **Increases conflict resolution overhead**.
    - Higher complexity in maintaining consistency.

D. Quorum-Based Replication

- Uses read (R) and write (W) quorums with the constraint **R + W > N** (total replicas).
- **Trade-offs**:
    - **High availability** if R and W are chosen appropriately.
    - Can achieve **strong consistency** but may increase **latency**.

3. Case Study Analysis

Case Study: Amazon DynamoDB

Replication Policy

- Uses **asynchronous multi-primary replication** to ensure high availability.

- Employs a quorum-based approach for eventual consistency.

Trade-offs

- **Prioritizes availability over consistency** (AP in CAP theorem).
- **Conflict resolution** handled via vector clocks.
- **High scalability** but may result in **temporary stale reads**.

Justification

- Suitable for large-scale applications requiring high availability, such as e-commerce and real-time services.

4. Critical Evaluation

Best Scenarios for Replication Policies

- **Synchronous replication**: Best for banking systems where consistency is critical.
- **Asynchronous replication**: Ideal for social media where availability is more important than immediate consistency.
- **Primary-backup replication**: Suitable for applications requiring a clear leader, like transactional databases.
- **Multi-primary replication**: Works well in collaborative environments where multiple users modify data simultaneously.

Influence of Network Partitions (CAP Theorem)

- **Consistency vs. Availability**: Systems must choose trade-offs based on network reliability.
- **Partition-Tolerant Systems**: Often sacrifice strict consistency for availability (e.g., NoSQL databases like Cassandra).

Recommendations for Selecting Replication Policies

Application Type	Recommended Replication Policy
Banking Systems	Synchronous replication with primary-backup
Social Media	Asynchronous multi-primary replication

Application Type	Recommended Replication Policy
E-commerce	Quorum-based replication
Real-time Analytics	Read-only replication with caching

5. Conclusion

Key Findings

- Replication enhances **availability, fault tolerance, and load balancing**.
- Trade-offs exist between **consistency, performance, and failure handling**.

Reflection

Understanding replication trade-offs helps in designing **efficient and scalable** distributed systems tailored to specific application needs.