Agenda

- Dependability and Fault Tolerance
- System Modelling
- Replication by Software
- Reliable Group Communication
- Distributed Agreement
- Consistent Database Replication

Agenda

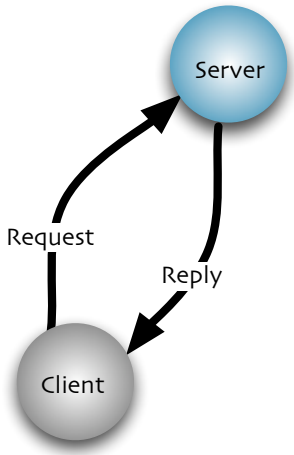
- Software based replication
  - Introduction
  - Consistency criteria
  - Active replication
  - Passive replication

Introduction

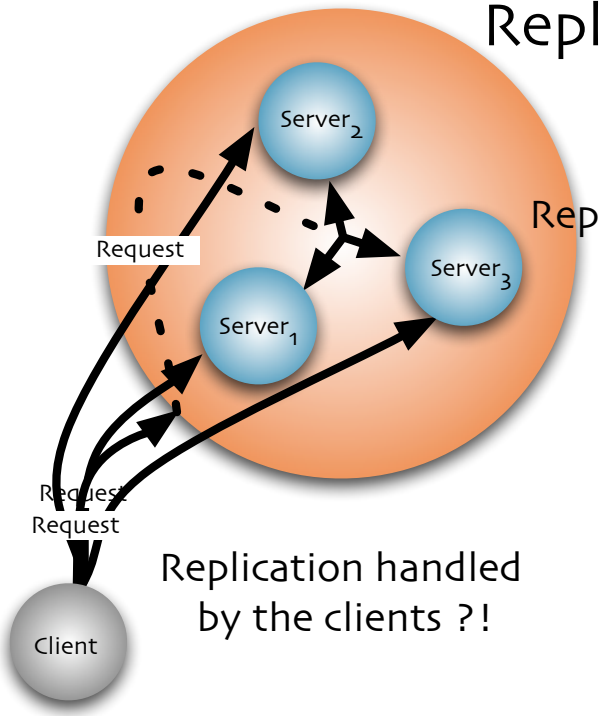
- Replication is the solution to achieve fault tolerance.
- Software based replication is an economically appealing technology to provide generic fault-tolerant services.
- Replication is usually expected to be transparent to the user. However, transparency impacts on performance and it's not always desirable.
- Services are replicated to increase their dependability and often their performance. However, these goals are often conflicting.
- Replica consistency is a key issue. Consistency enables reliability, availability and performance trade-offs.

Introduction

Non-replicated server

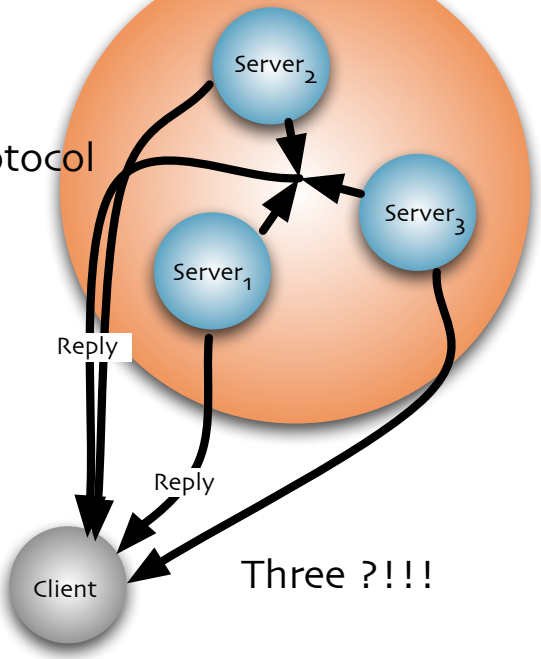


Replicated server



Replication handled by the clients ?!

Replication protocol



Three ?!!!

- How consistent shall the replicas be?
- What aspects shall the replication protocol favor?
- What fault model shall be considered?

Consistency criteria

- How consistent shall the replicas be?
- How to express replica consistency?
- The requirement of strictly consistent replicas could be stated as:

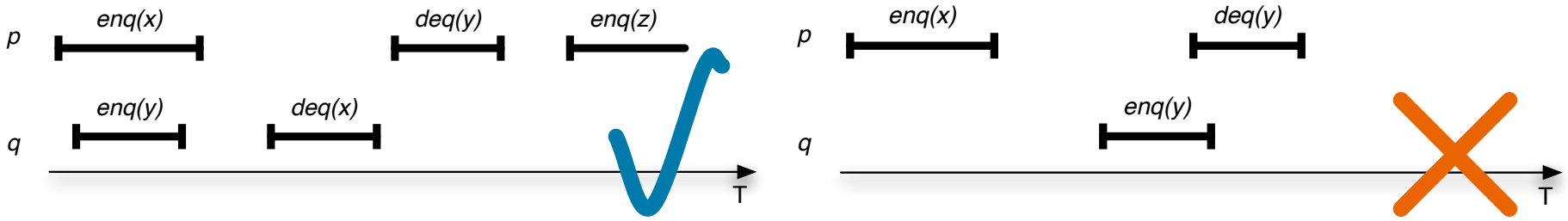
> Any read operation returns the value corresponding to the most recent written value

  - Relies on absolute global time. Cannot be achieved in a distributed system

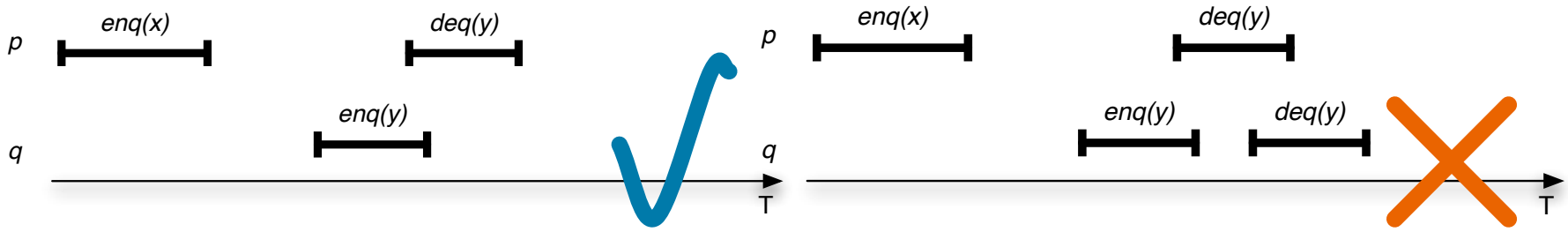
- Consistency criteria are usually stated as a set of predicates over system´s executions

Consistency criteria

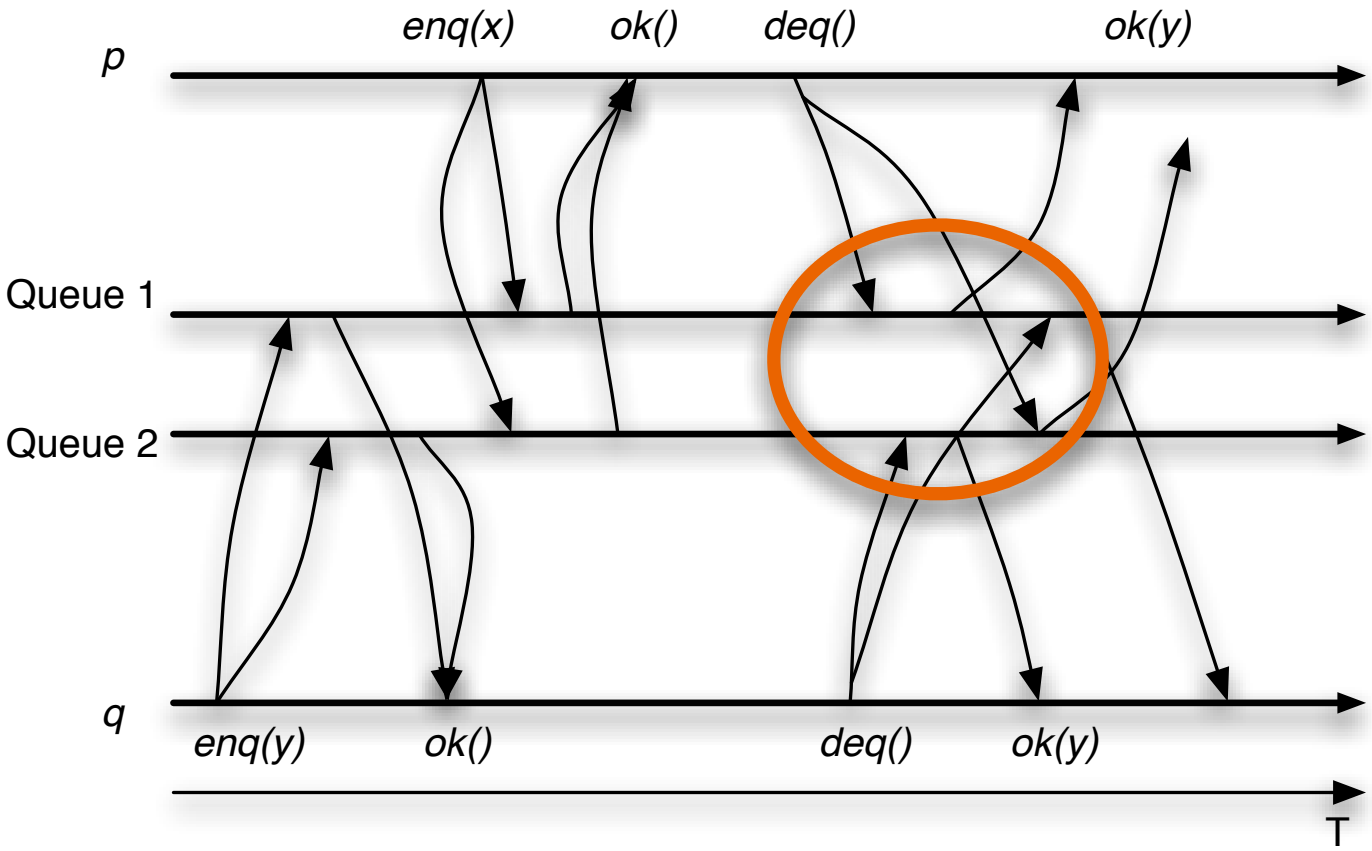
- A replicated service is said to be Linearizable if for every execution E there is some interleaved sequence S of all operations of E such that the following properties are satisfied:
  - S satisfies the semantics of the non replicated service
  - The realtime order of non-concurrent operations in E is preserved in S.
- Consider a FIFO queue:



- A replicated service is said to be Sequentially Consistent if for every execution E there is some interleaved sequence S of all operations of E such that the following properties are satisfied:
  - S satisfies the semantics of the non replicated service
  - The local order of operations in E is preserved in S.
- Consider a FIFO queue:



- In a replicated system (even) a non sequentially consistent execution can be easily produced even in the absence of faults:

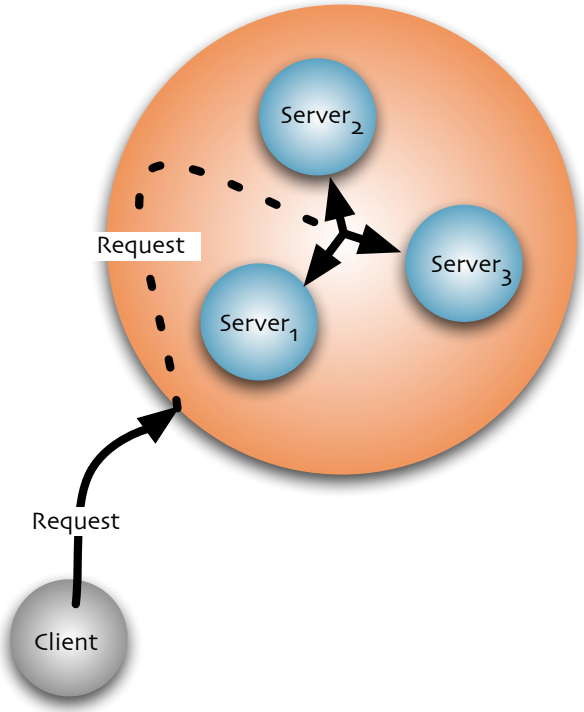


- To satisfy Linearizability two conditions are necessary and sufficient:

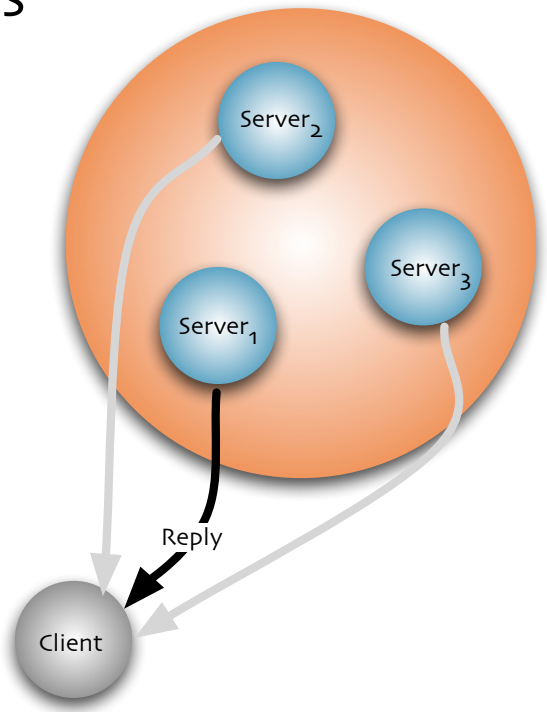
Order: Given invocations *op(arg)* and *op'(arg')* on replicated server *x*, any two replicas of *x* that handle both operations do so in the same order.

Atomicity: Given invocation *op(arg)* by client *p* on replicated server *x*, if one replica of *x* handles the operation, then every non faulty replica of *x* also handles *op(arg)*.

Active Replication



2. Each replica processes the request, updates its own state, and returns the response to the client



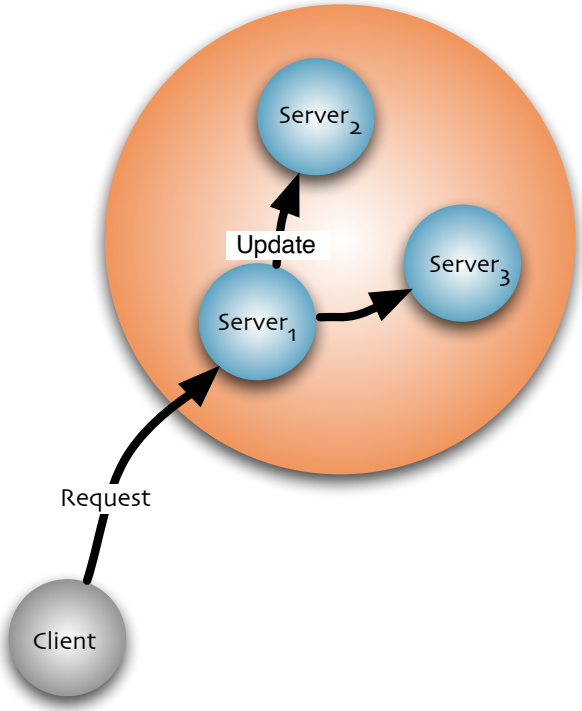
1. Request goes to all the replicas

3. Client waits until it receives the first response (or a majority of identical responses)

Active Replication

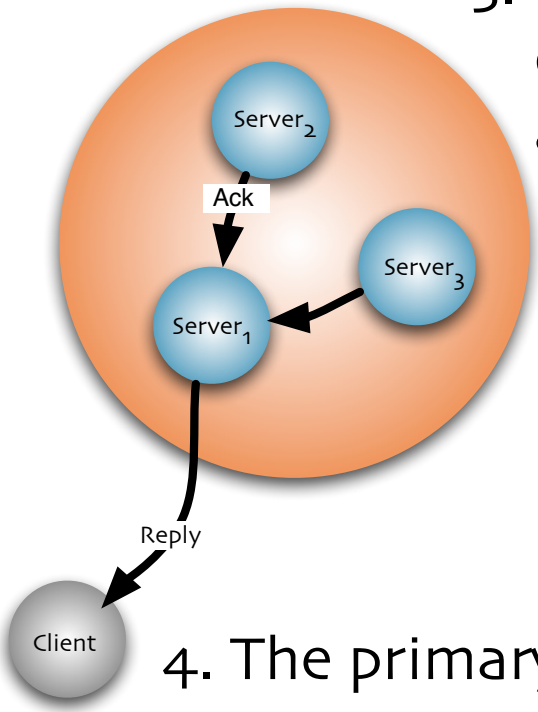
- Active replication requires request execution to be deterministic.
- The failure of a replica tends to be transparent for the clients.
- The failure and reintegration of the replicas has however to be dealt by the replication protocol.
- The active replication protocol is based on a communication primitive available to clients that ensures the required order and atomicity properties.
- This primitive is called total-order multicast or atomic multicast

Passive Replication



2. The primary processes the request, updates its own state, and sends a state update message to all other replicas.

3. Each replica updates its own state, and sends an acknowledgment to the primary.

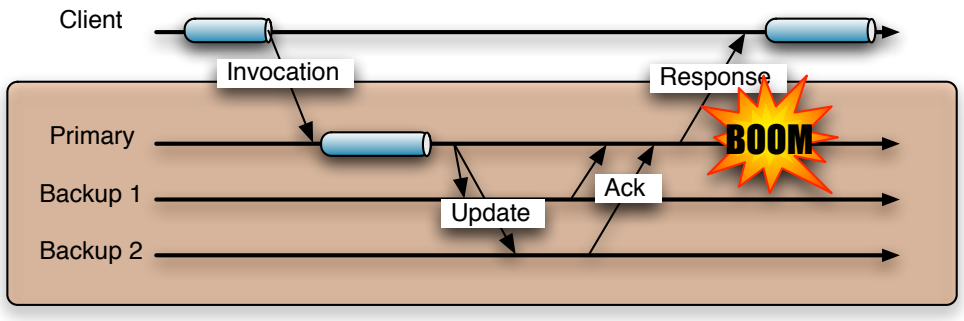


1. Request goes to a distinguished replica - the primary

4. The primary returns the response to the client.

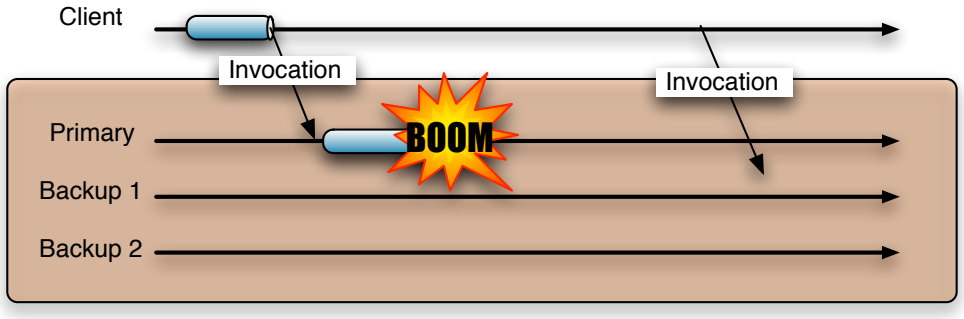
Passive Replication

A



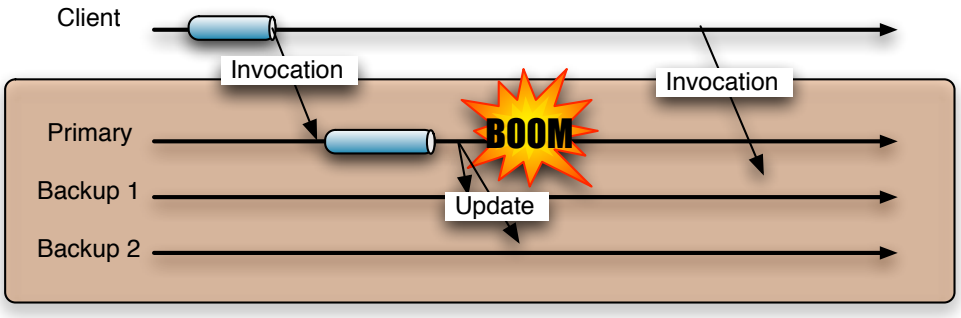
Transparent to the client

B



Client reissues invocation

C



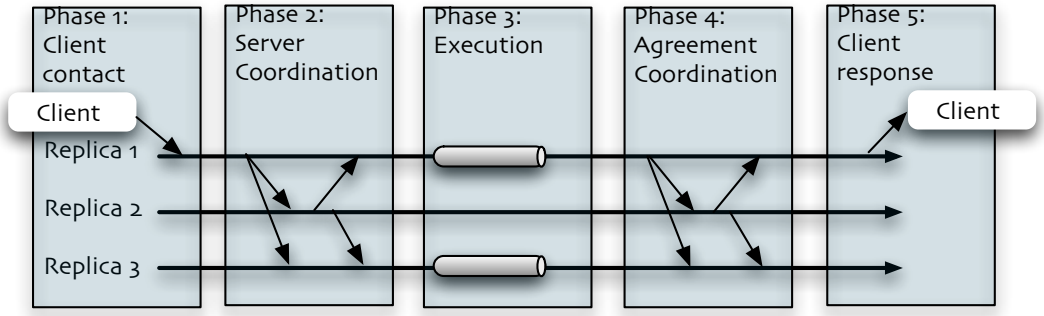
Which backups received updates?

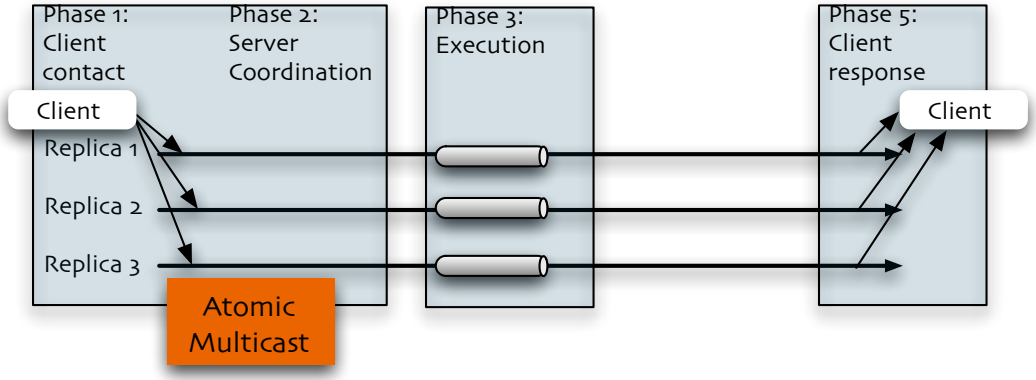
How does the new primary process the reissued invocation?

Passive Replication

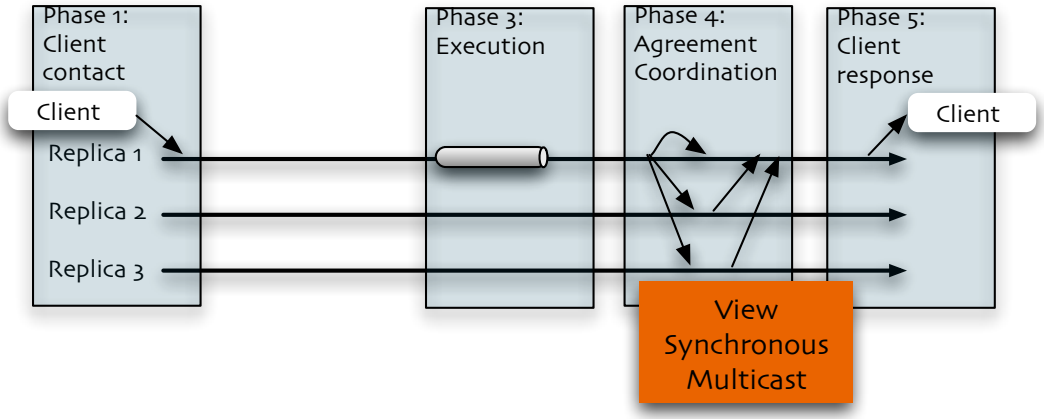
- Passive replication is asymmetric: Only one replica actually processes the requests. Execution does not need to be deterministic.
- The failure of the primary may not be transparent to the clients.
- The choice of the primary, as well as the failure and reintegration of the replicas has to be dealt by the replication protocol.
- The passive replication protocol is based on a group membership service and on the view-synchronous multicast communication primitive.

Functional model of replication





Active Replication



Passive Replication