Replication and Consistency in Distributed Systems

49

Replication

- Multiple copies of data each on different nodes of a distributed system
- Reasons for replication
  - Reliability
  - Performance

50

Data-Centric Consistency Models

- Strict
- Sequential
- Causal
- FIFO
- Weak
- Release
- Entry

51

Sequential Consistency

The result of any execution is the same as if the (read and write) operations by all processes on the data store were executed in some sequential order and the operations of each individual process appear in this sequence in the order specified by its program.

52

Linearizability

The result of any execution is the same as if the (read and write) operations by all processes on the data store were executed in some sequential order and the operations of each individual process appear in this sequence in the order specified by its program. In addition, if ts_op1(x)<ts_op2(y), the op1(x) should precede op2(y) in this sequence

53

Causal Consistency

- Writes that are potentially causally related must be seen by all processes in the same order. Concurrent writes may be seen in a different order on different machines.

54

FIFO Consistency

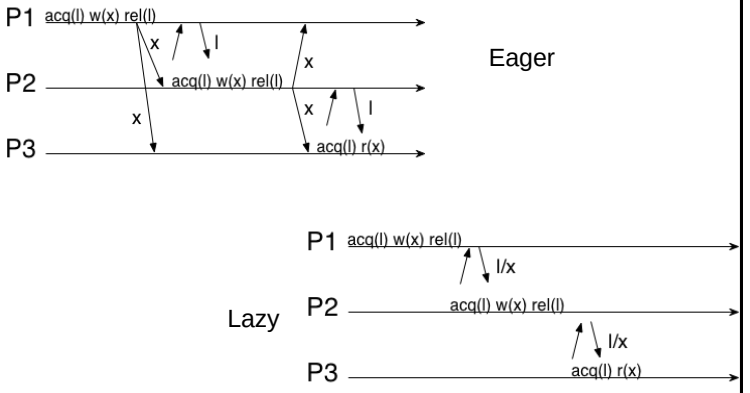
Writes by a single process are seen by all other processes in the order in which they were issued, but writed by different processes may be seen in a different order by different processes.

55

P1: W(x)a
P2: W(x)b
P3: R(x)b R(x)a
P4: R(x)a R(x)b

56

Release Consistency: Eager vs. Lazy



57

Entry Consistency

- Data associated with a critical section protected by a synchronization variable
- Only data associated with synchronization variable brought up-to-date on acquire
- Mutual exclusion required for modification of data

58

Client-Centric Consistency

- Eventual consistency: In the absence of updates, all replicas converge to the same values
- Monotonic reads
- Monotonic writes
- Read your own writes
- Writes follow reads

59

Types of Replicas

- Replica creation and placement:
  - Permanent
  - Server-initiated
  - Client-initiated

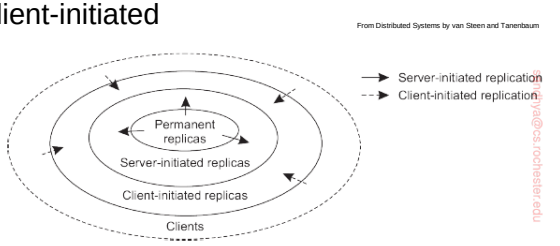


Figure 7.21: The logical organization of different kinds of copies of a data store into three concentric rings.

60

Keeping Replicas Consistent

- Primary-based
- Quorum-based

61

Quorum-Based Replication Protocols

- Replicate file on N servers
  - For update: contact majority (N/2+1) for agreement
  - For read: contact majority once again, read is successful if they have the same version
  - In general
    - $N_R+N_W > N$
    - $N_W > N/2$

62

Update Propagation

- What is propagated?
  - Invalidation
  - Update
  - active replication (move the computation)
- When is it propagated?
  - Pull versus push
  - Leases
  - Epidemic protocols

63

When is it Propagated?

- Push vs. Pull

Issue	Push-based	Pull-based
State at server	List of client replicas and caches	None
Messages sent	Update (and possibly fetch update later)	Poll and update
Response time at client	Immediate (or fetch-update time)	Fetch-update time

Figure 7.23: A comparison between push-based and pull-based protocols in the case of multiple-client, single-server systems.

From Distributed Systems by van Steen and Tanenbaum

- Leases – a compromise [Gray and Cheriton SOSP'89]
  - Adaptive leases
    - Age-based (lower frequency of writes→ higher lease)
    - Client request frequency based (higher freq → higher lease)
    - Server state space/load based (higher space → lower lease)

64

Epidemic protocols

- Infective: server that holds updates and is willing to spread it
- Susceptible: server that has not been updated yet
- Removed: server that is not willing or not able to spread its update

65

65

Epidemic protocols

- Try to “infect” all members in the group with new updates as fast as possible

66

66

Anti-entropy

- A server P picks up a server Q at random
  - P only *pushes* its own updates to Q
    - Spreads slowly
  - P only *pulls* in new updates from Q
    - Works better when most servers are infective
  - P and Q send updates to each other (*push-pull*)

67

67

Gossiping

- If P is recently updated with data item x, it will contact an arbitrary server Q and try to push the updates to Q
- If Q has already received the update, P will lose interest in spreading it further with some probability
  - No guarantee that all servers will be updated

68

68