基于验证的数据复制

数据库管理组
刘小成

Agenda

- Database Replication Techniques
- Certification Based Replication
- Percona Xtradb Cluster
- Reference

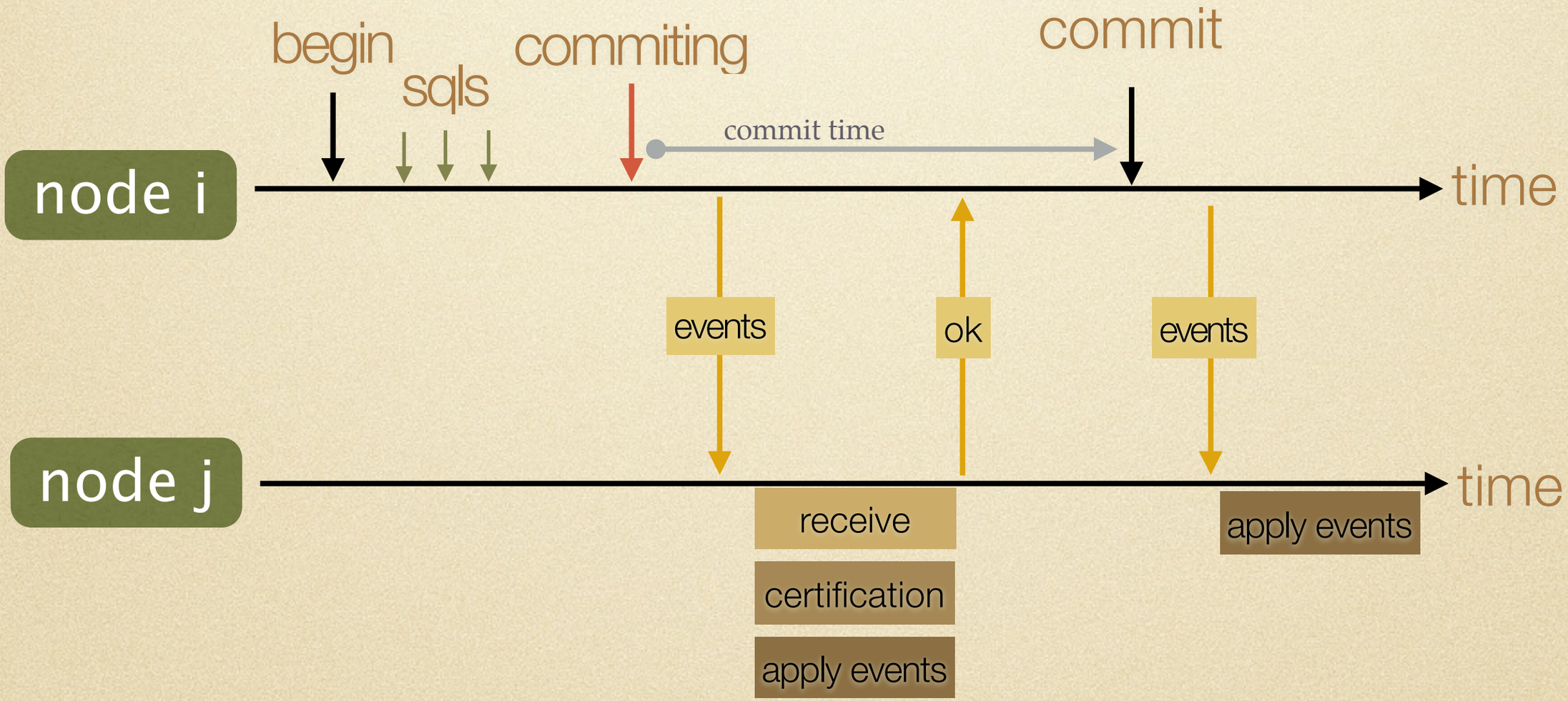
Database Replication

Failover / Load balance / Backup / Offline Report / …

Latency / Data Loss / Inconsistency /
Performance / …

Database Replication

Synchronous ? Asynchronous



Database Replication

	Eager Replication	Lazy Replication
Group Ownership	N transaction N Database	1 transaction N Database
Master Ownership	N transaction 1 Database	1 transaction 1 Database

Database Replication

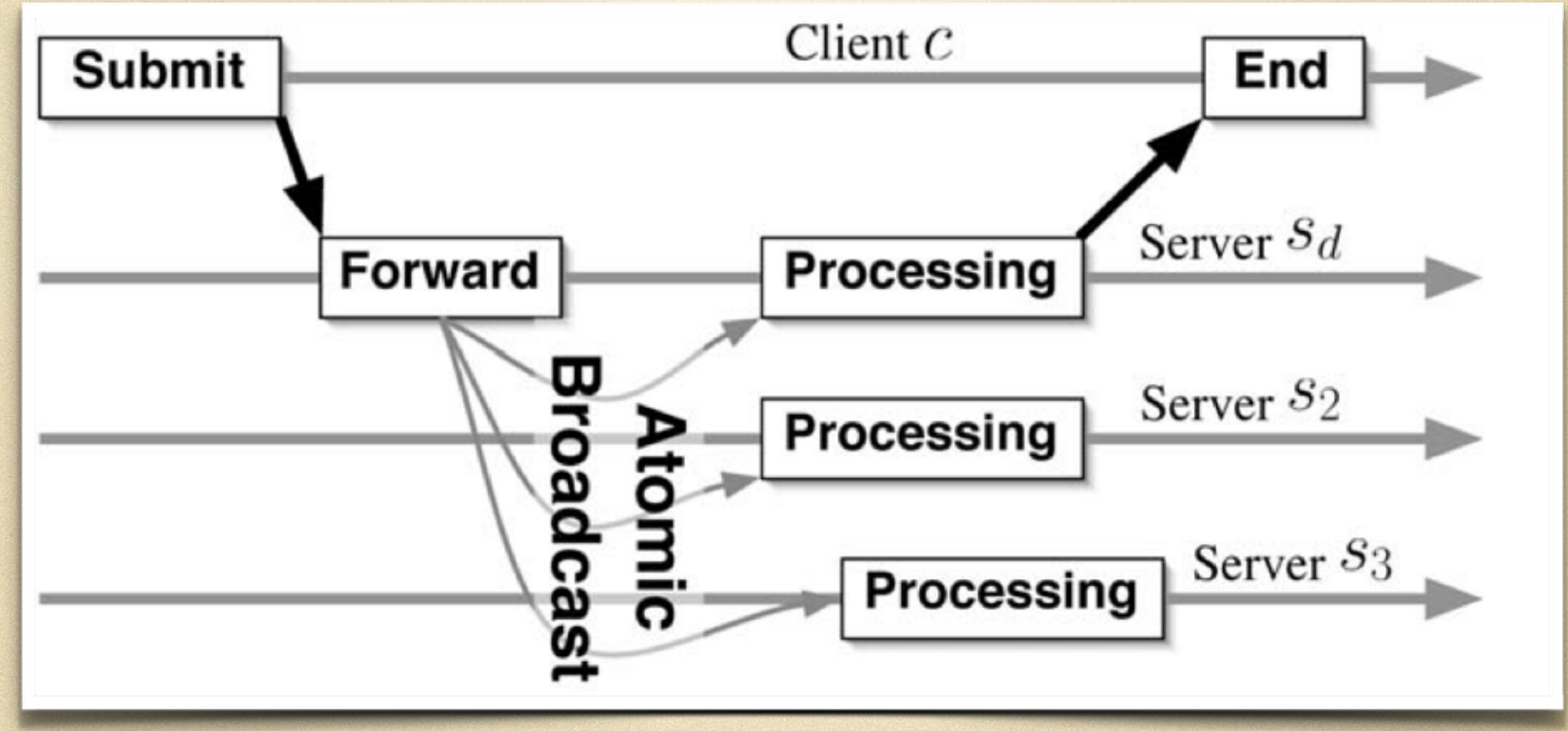
Total Order Broadcast:

- Active Replication
- Certification Based Replication
- Weak Voting Replication
- ....

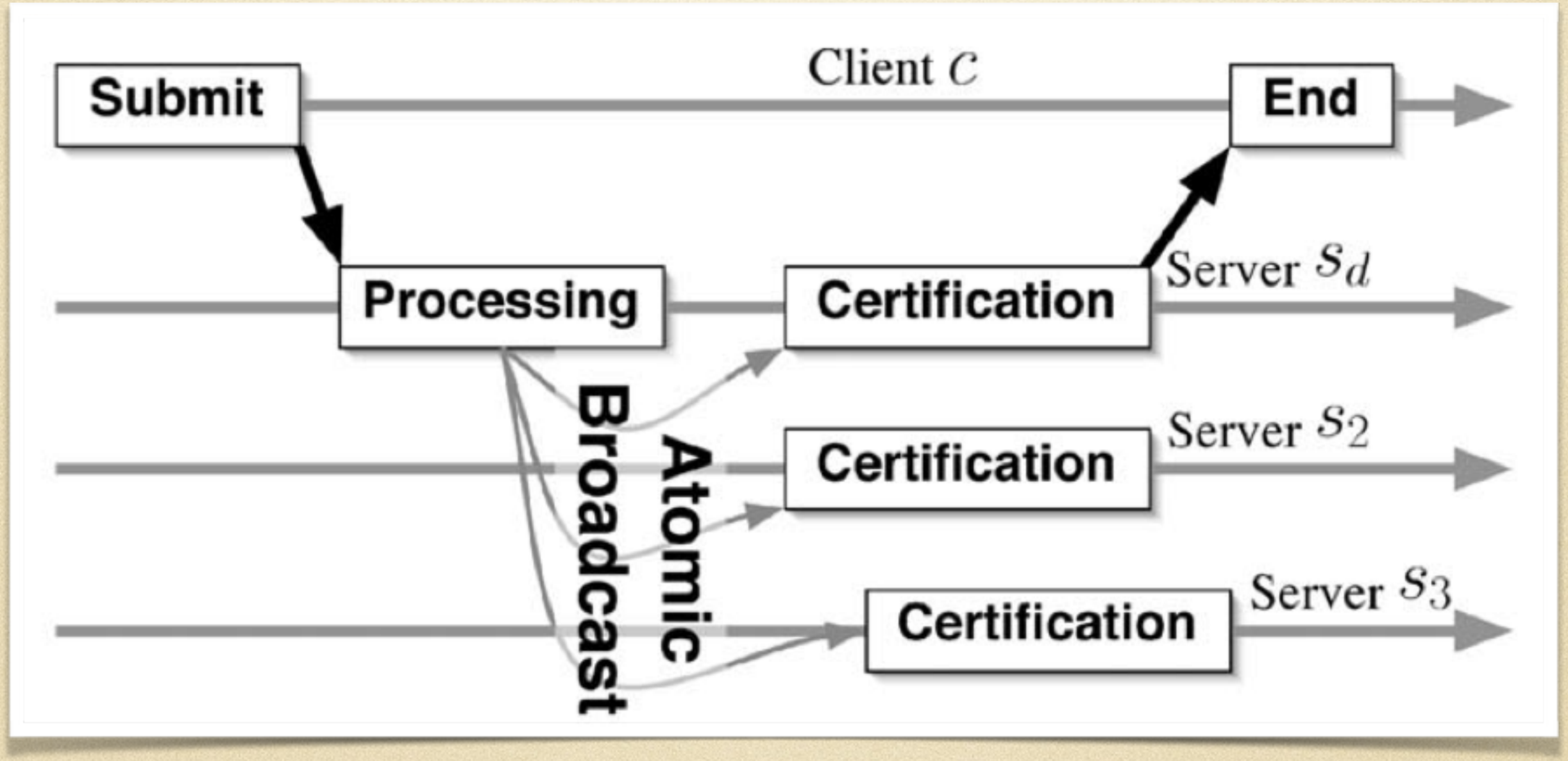
Primary copy replication

Lazy replication

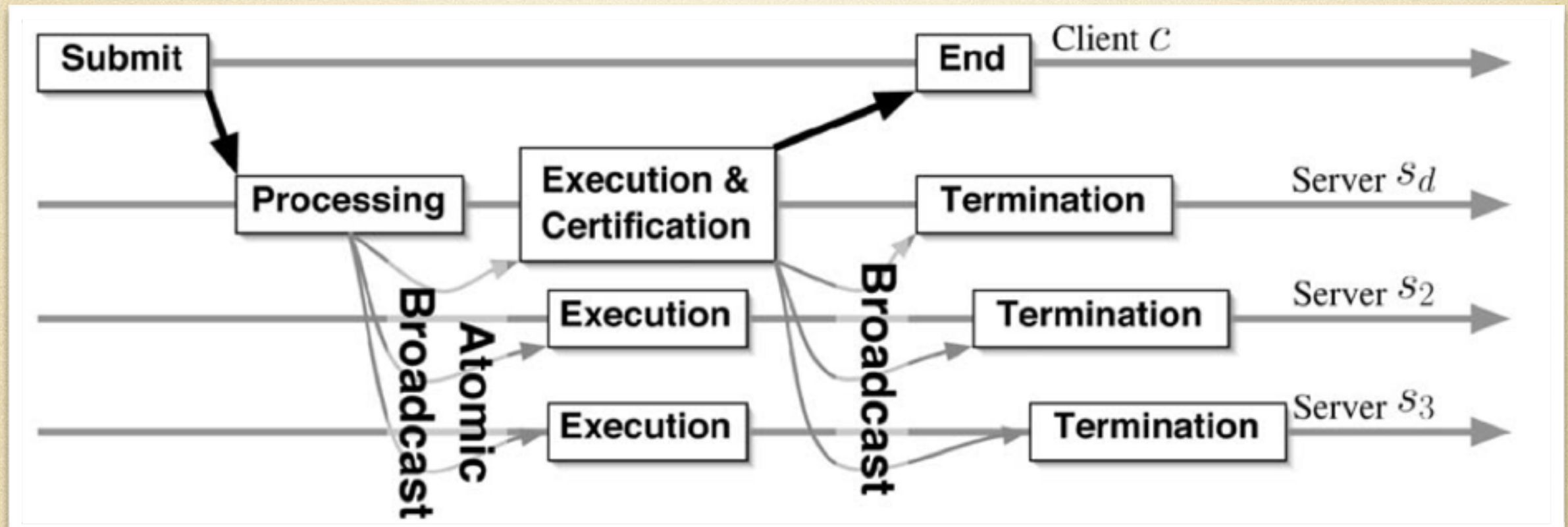
Active Replication



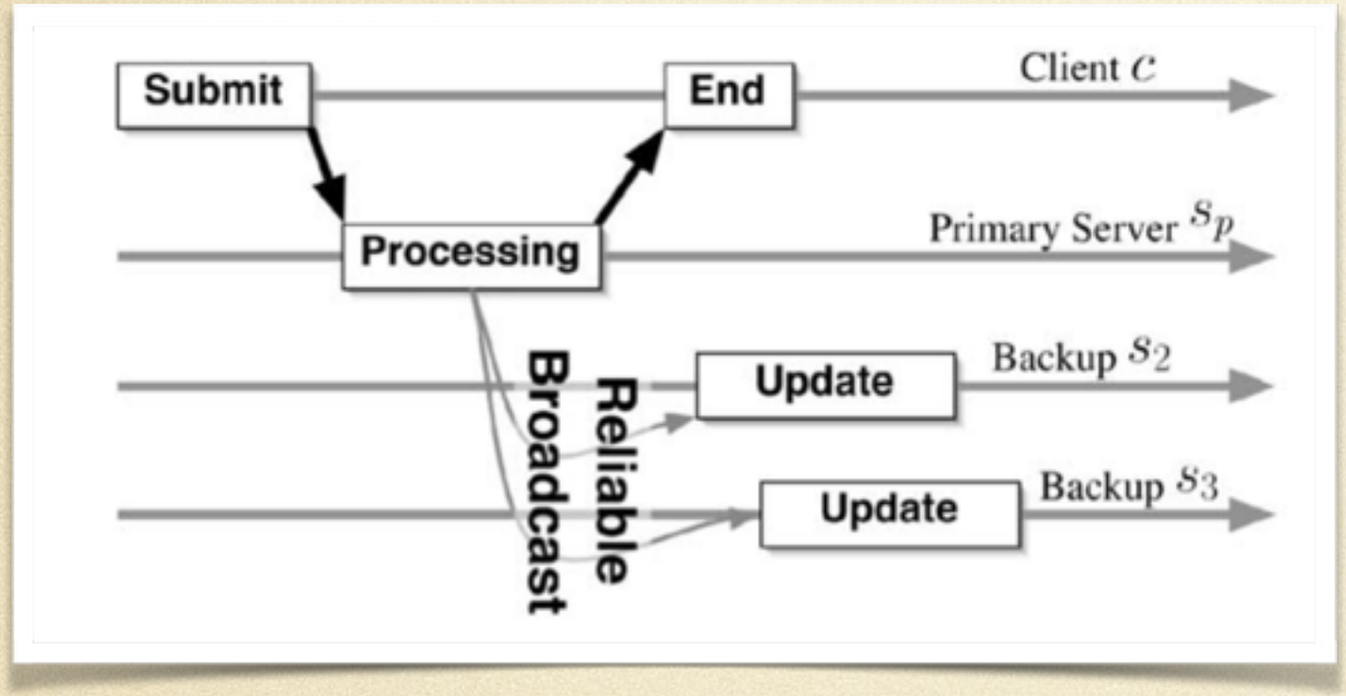
Certification-based Replication



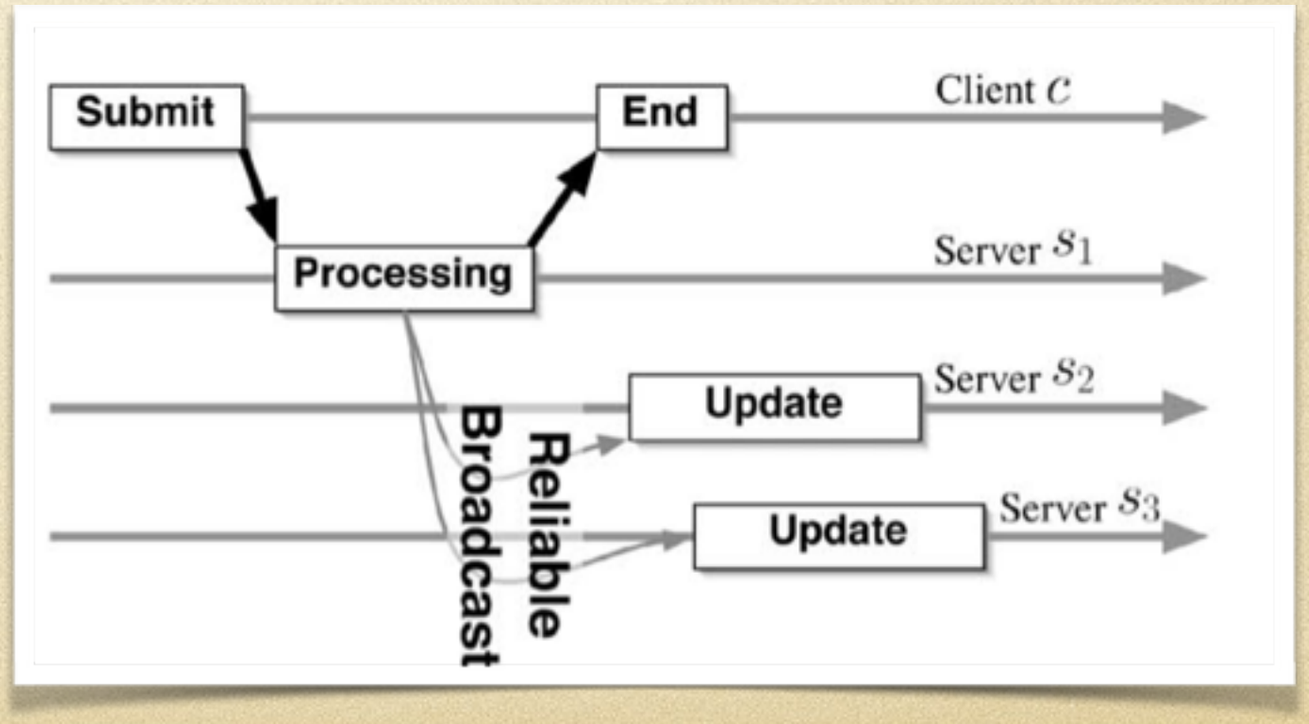
Weak Voting Replication



Primary Copy Replication



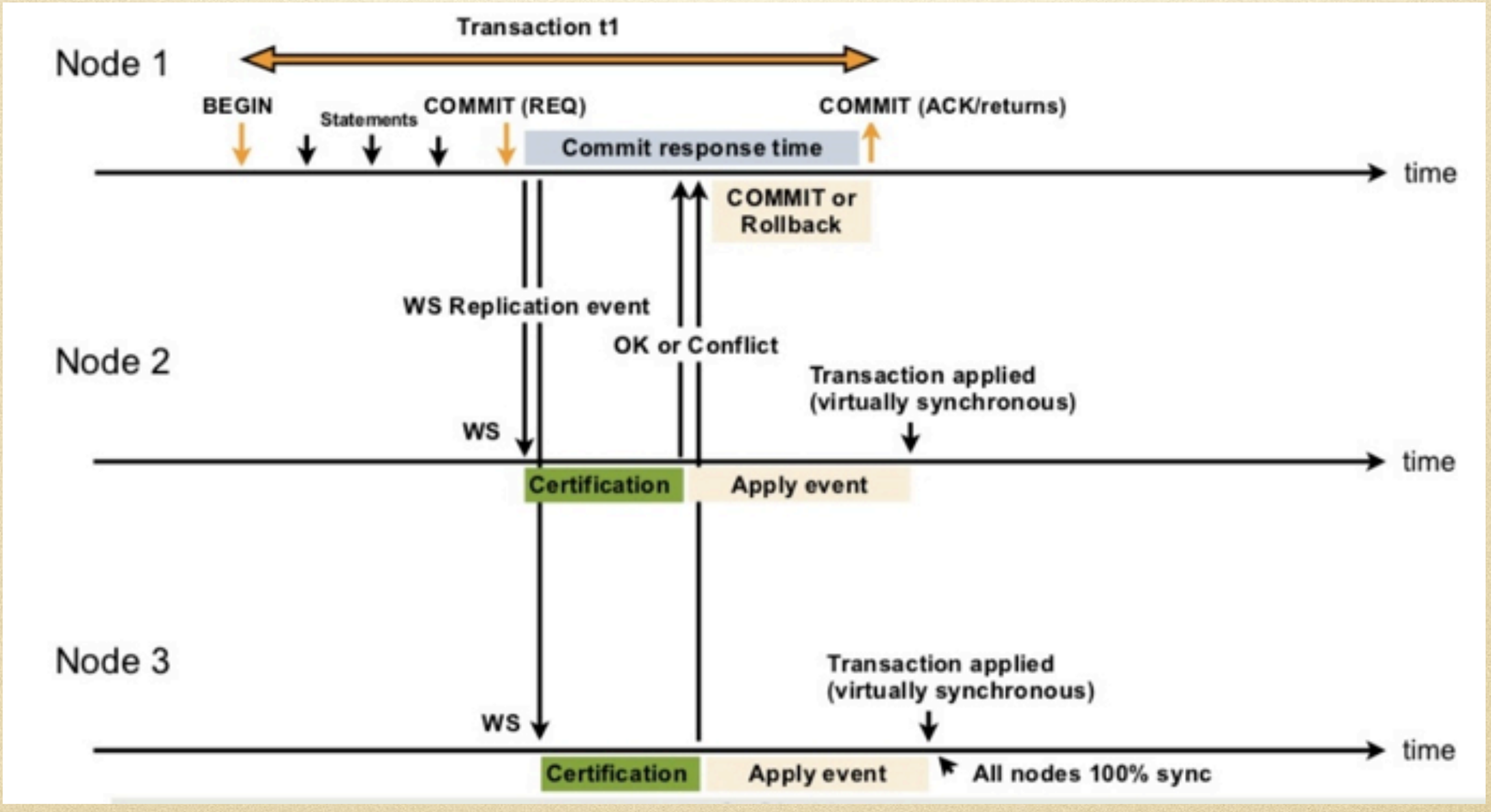
Lazy Replication



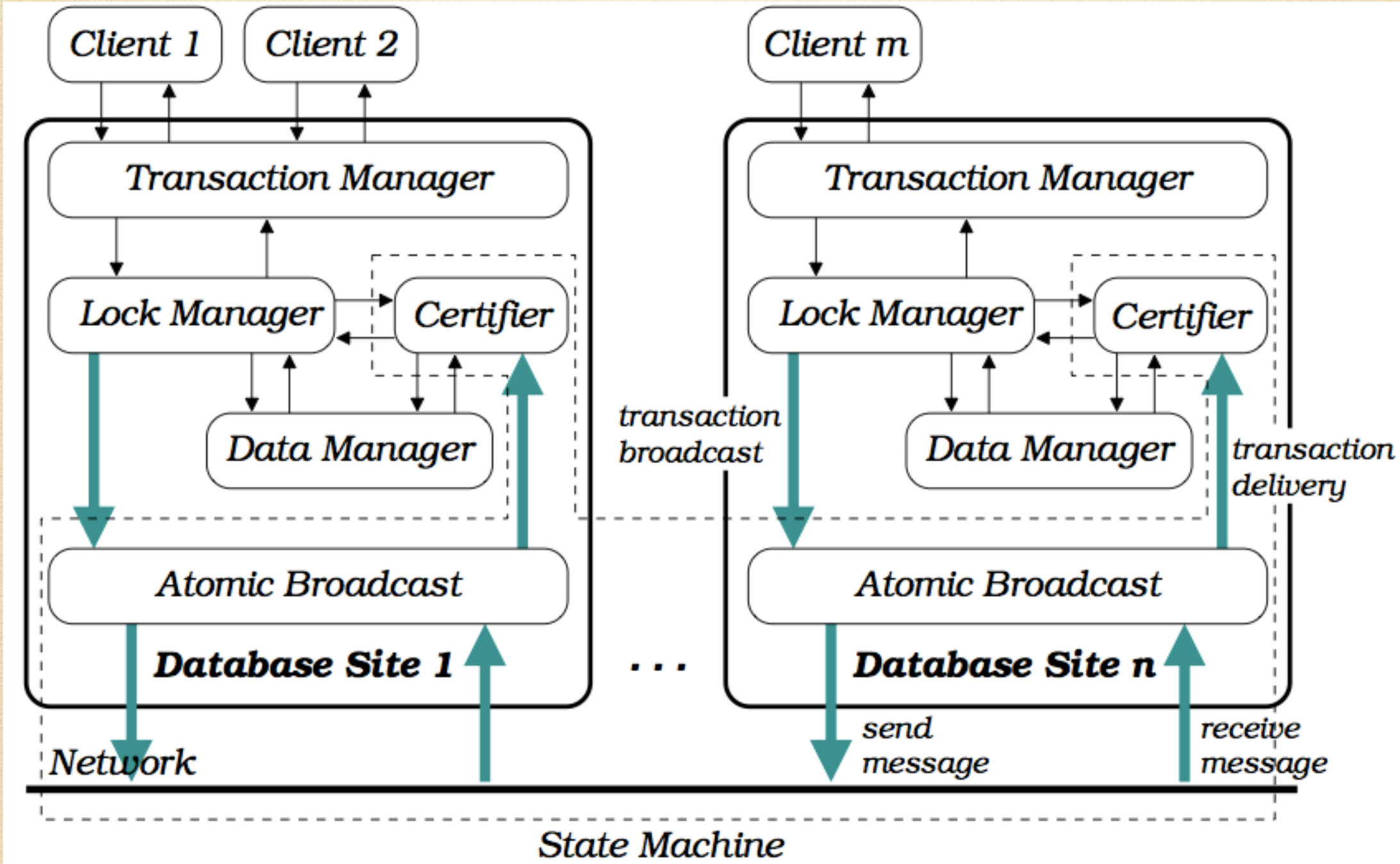
Database Replication

- Every no-faulty replica receives every request
  - 每个正常运行的节点正确接收所有事务
- Process request in the same relative order
  - 以相同的相对顺序执行事务

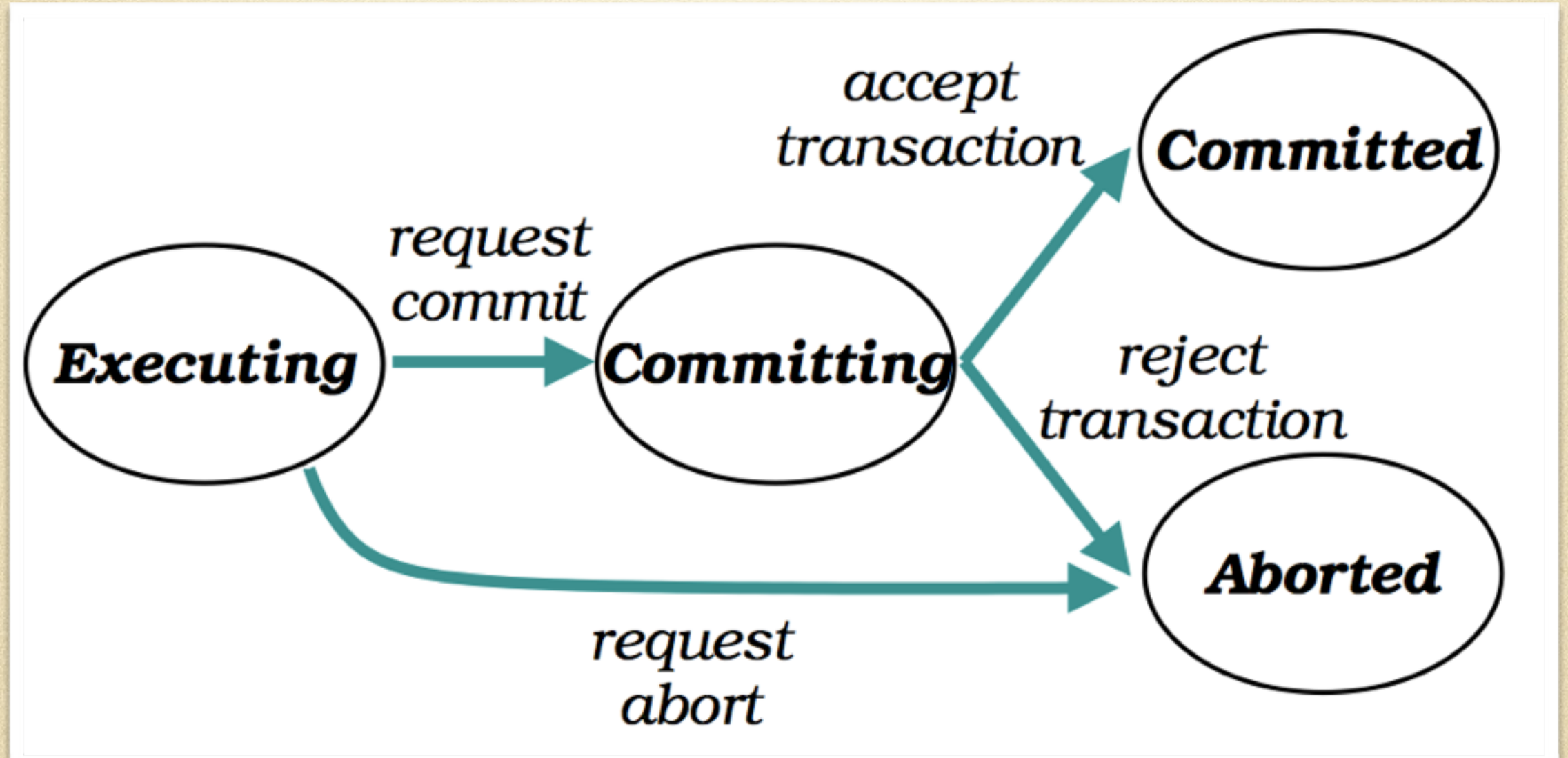
Certification Based Replication



Database State Machine



Transaction states



Certification Test

Ta: update t set a=2 where id=1;

Tb: update t set a=3 where id=1;

Order

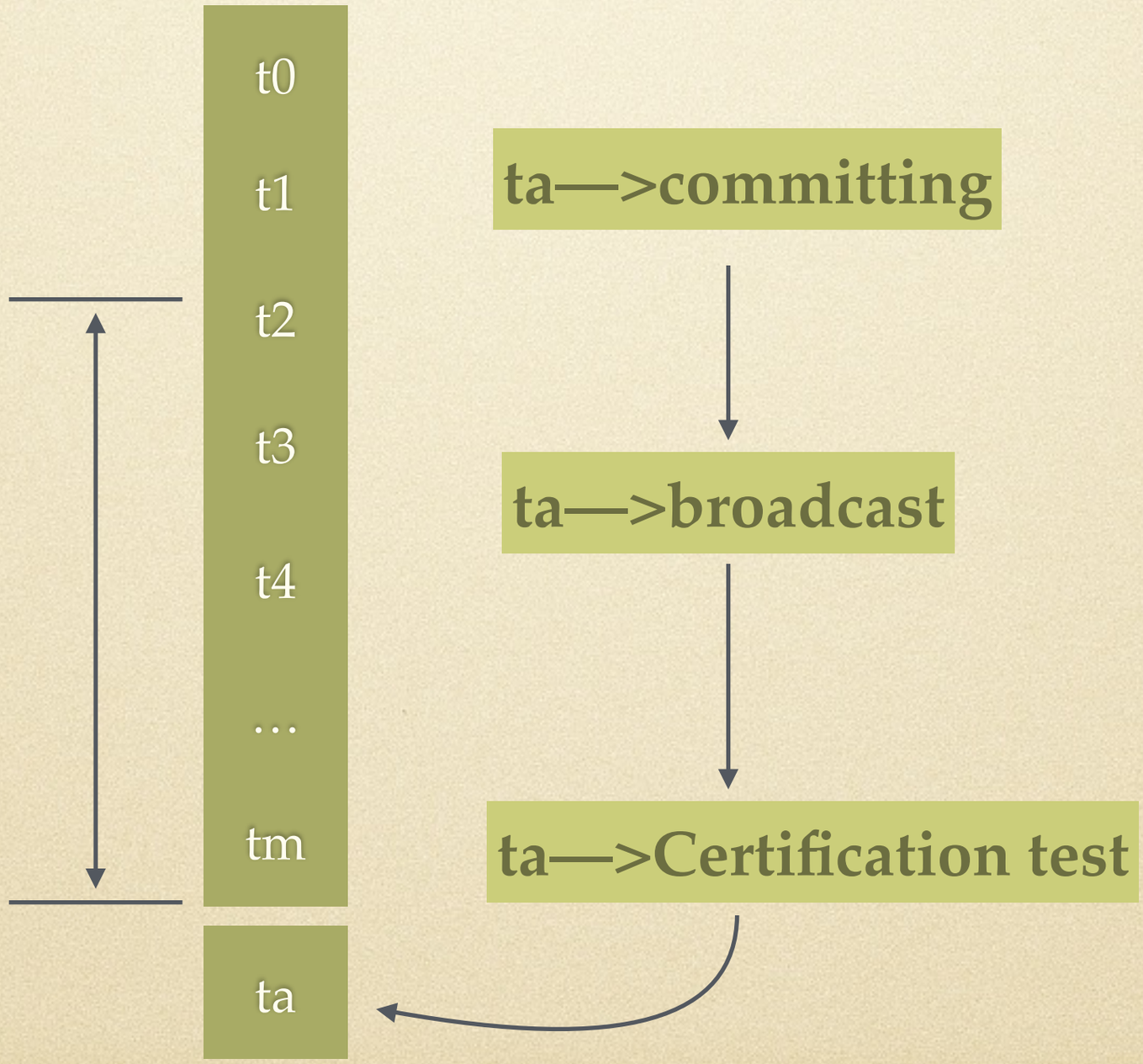
Conflicts

Ta： update t set a=2 where id=1 and a<2;

Tb： update t set a=3 where id=1 and a<2;

Certification Test

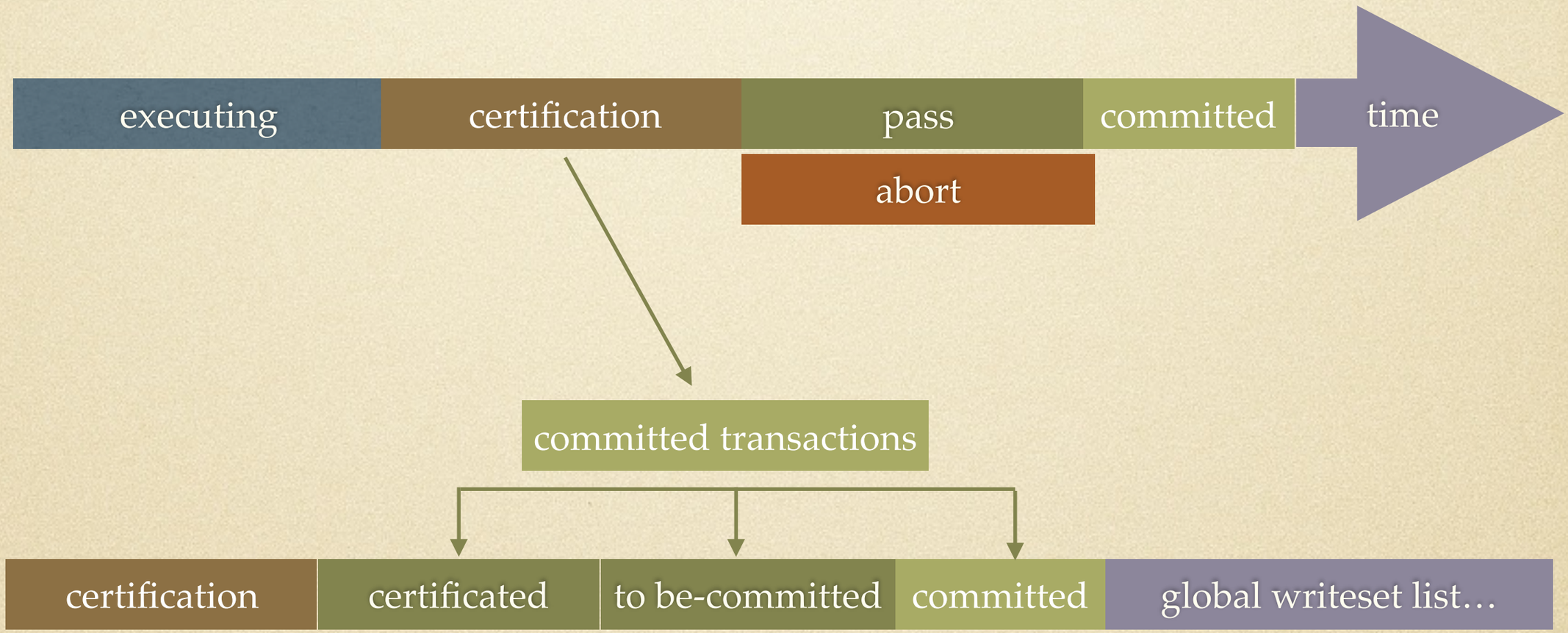
State update:



Certification Test

Conflicts : **Row(write-read)** *read-write write-write*

Transaction states:



Certification Test

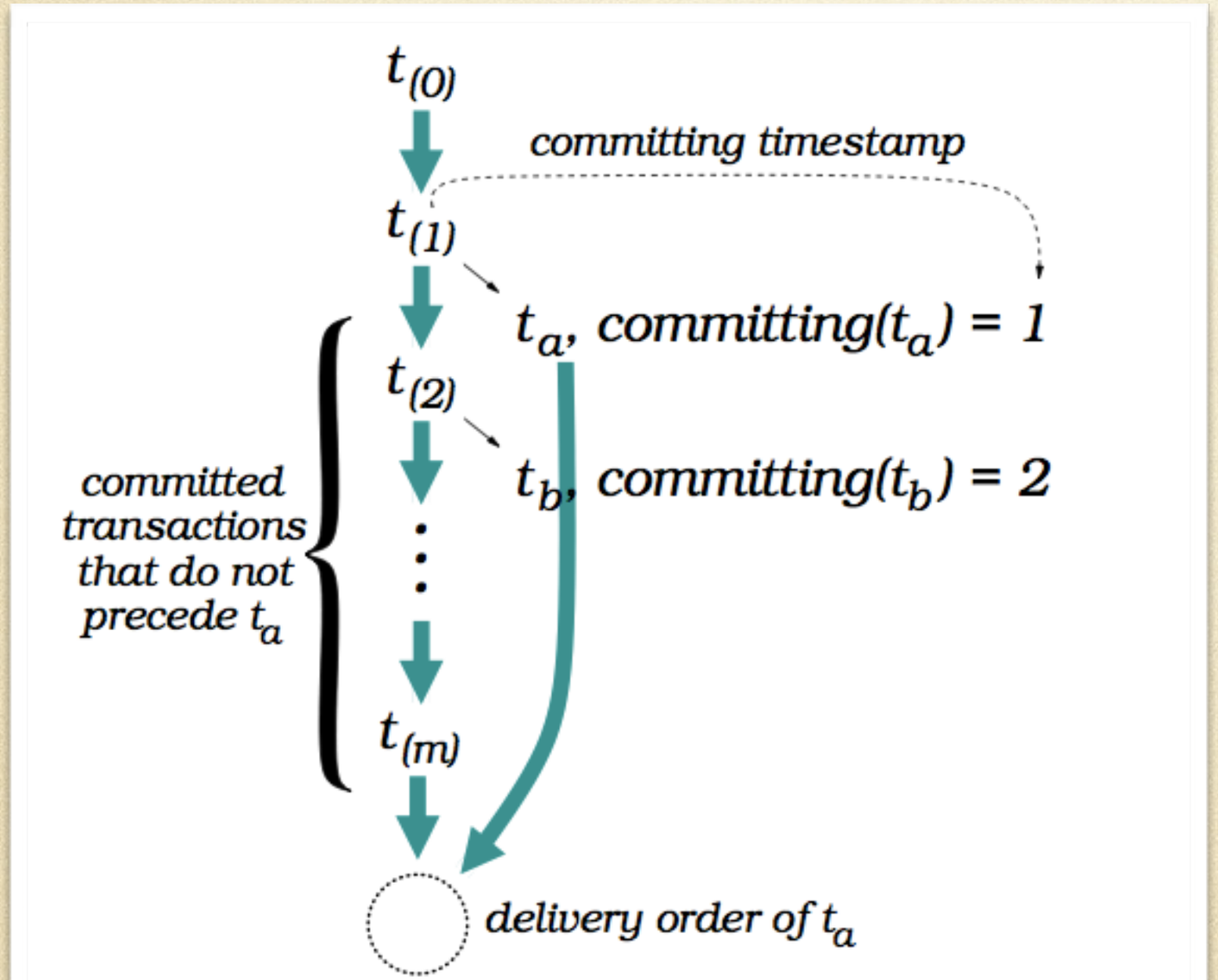
Total order timestamp(unique):

- On the same node:
  committing(tb) ⟶ committing(ta)
- On different node:
  commited(tb)node(ta) ⟶ committing(ta)

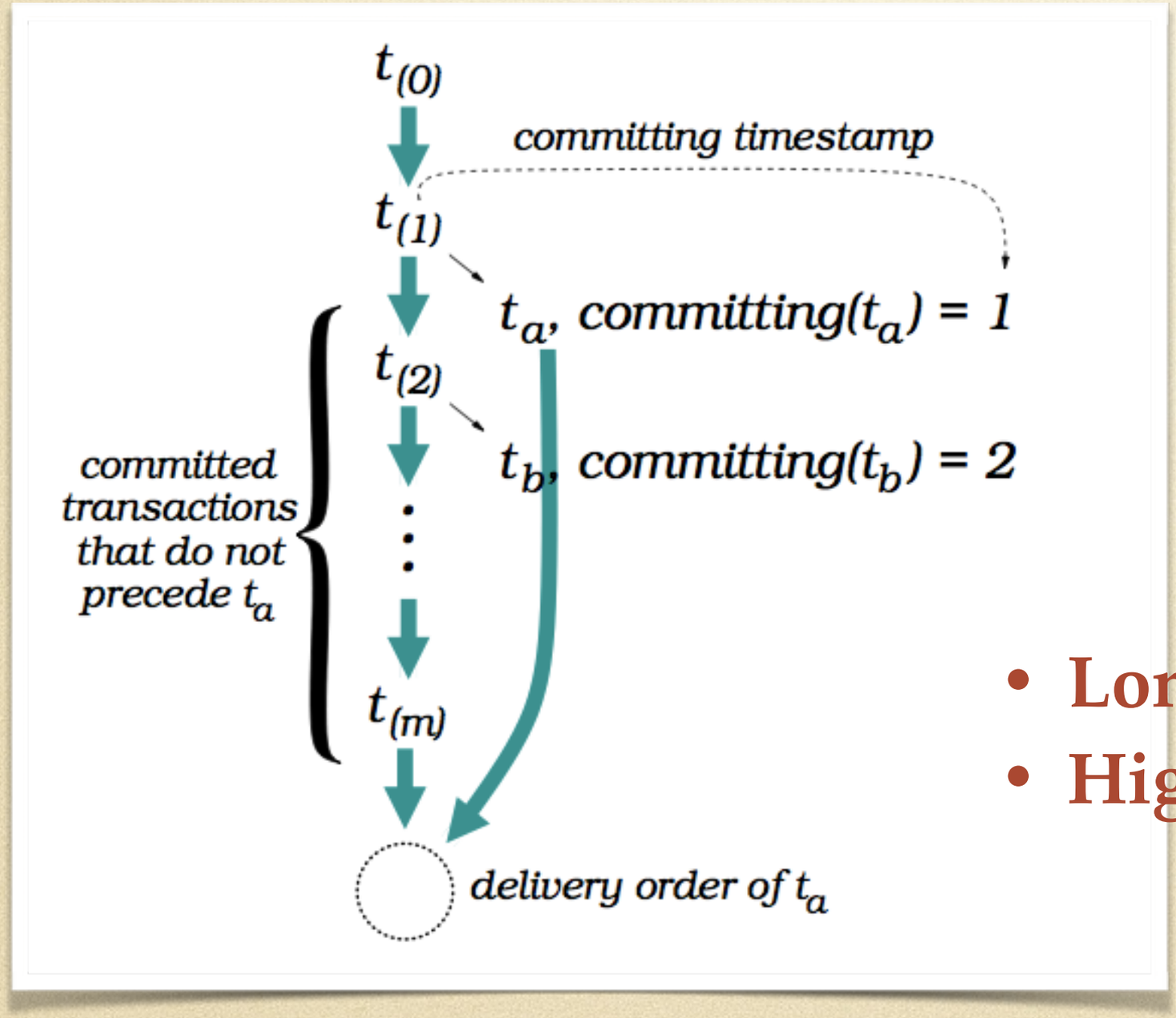
All sites certify a transaction using a same set of committed transactions

Certification List

Transactions with greater committed timestamp than ta

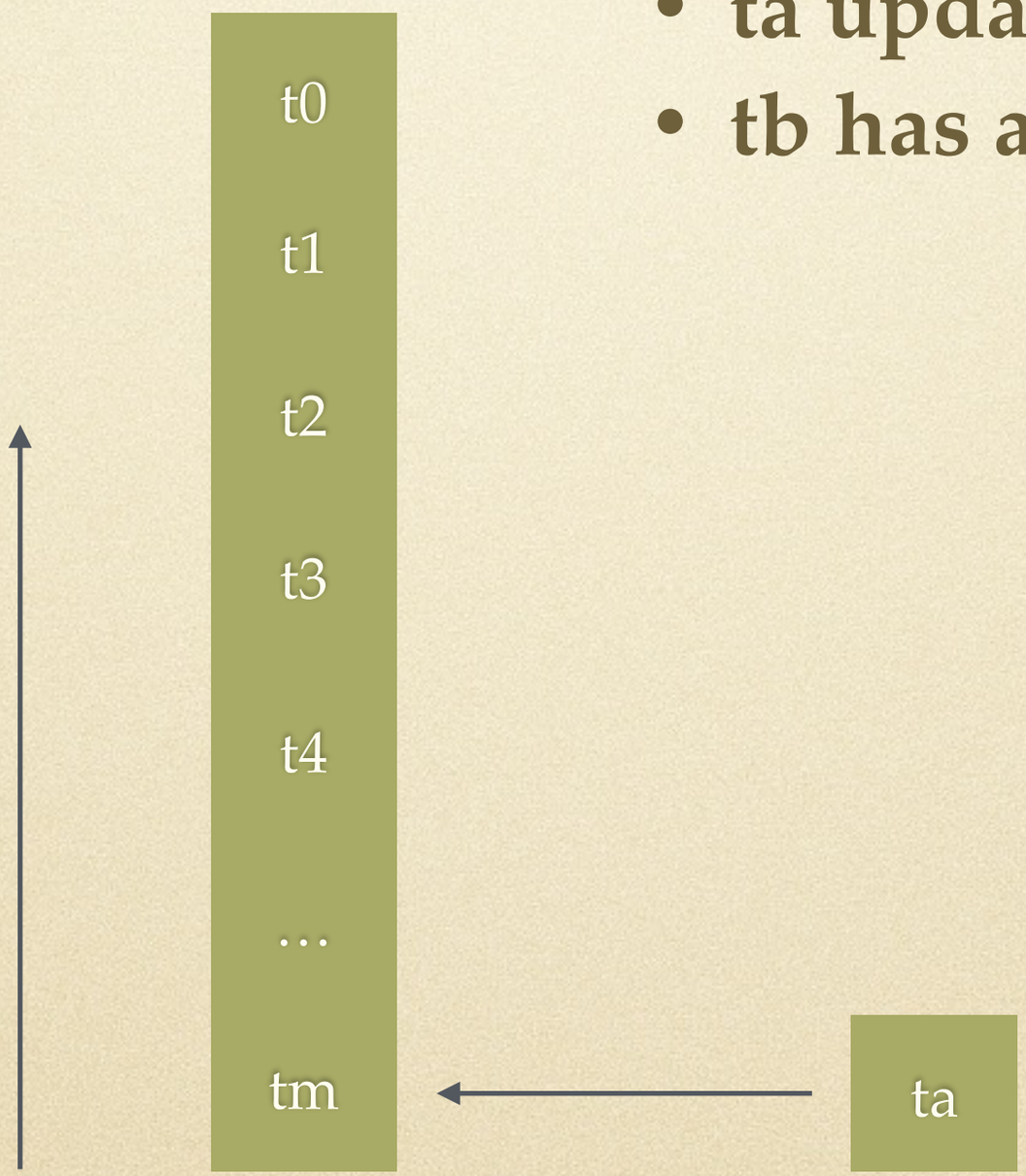


Certification Test



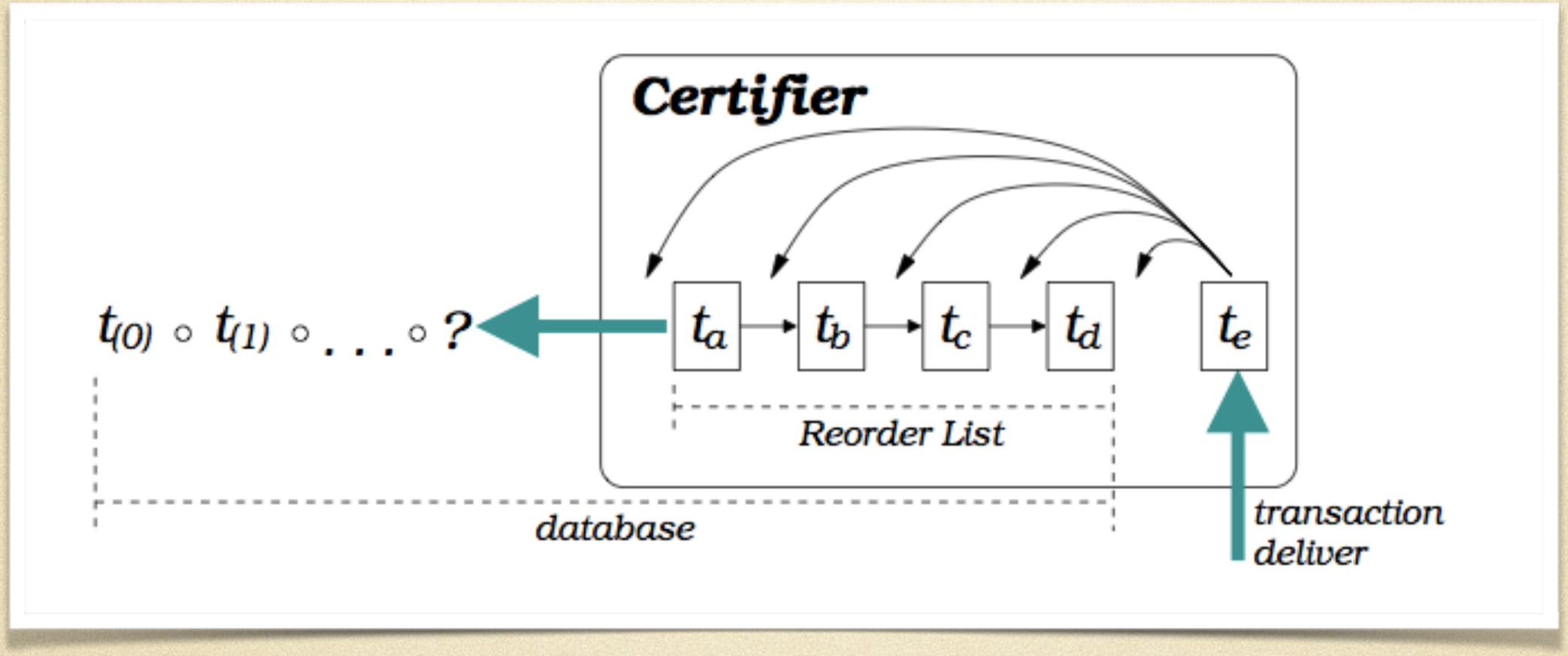
- Long Certification List
- High Abort Rates

Reorder Technique



- **ta updates tb's read**
- **tb has applied**

Reorder Technique



- 验证的消耗增加
- 驻留时间变长

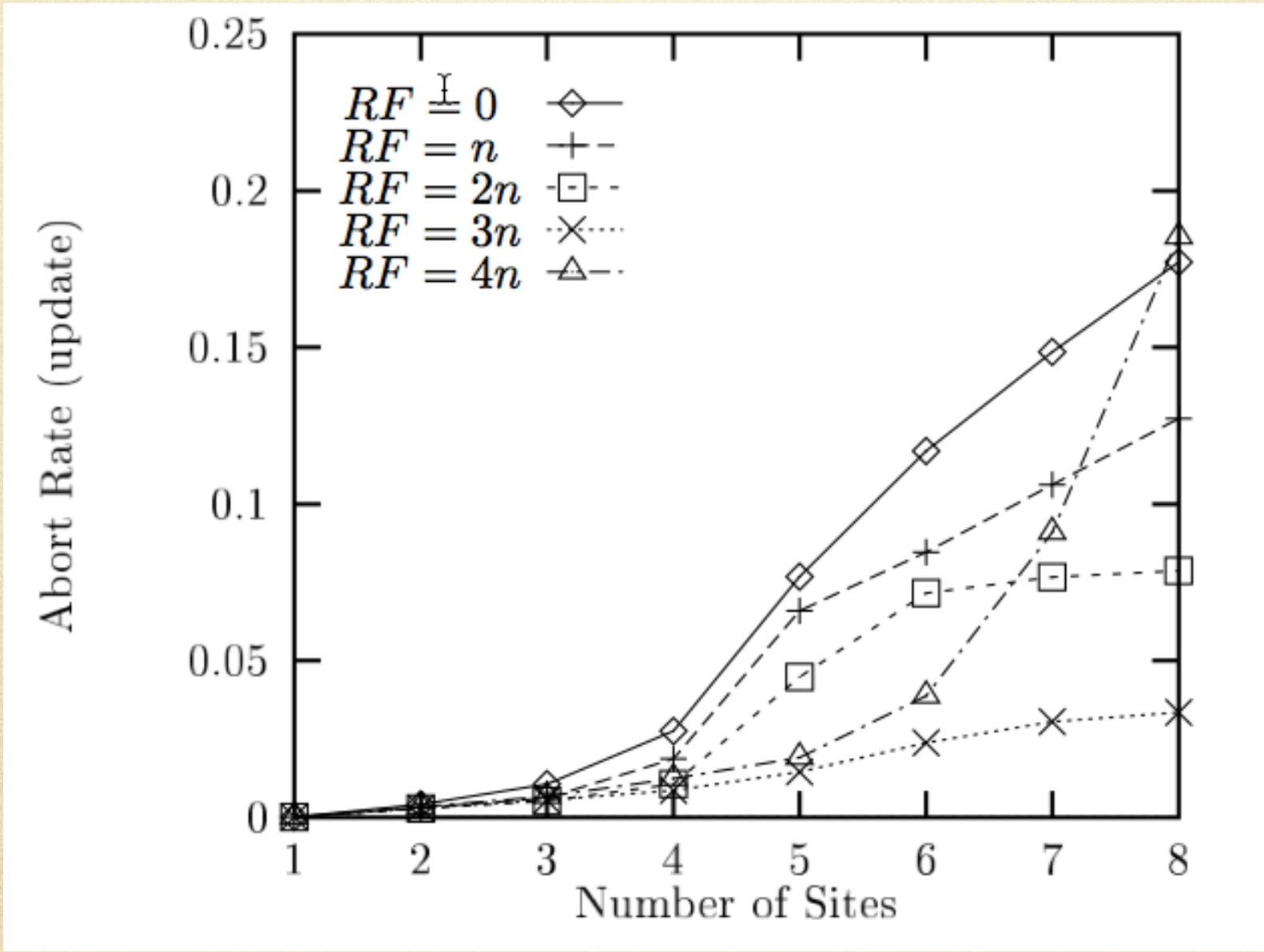
- 短队列
- ta更早提交

Commitment

All transaction on the right of p shifted to right

The left one be removed from the list and apply updates

Reorder Technique



Commitment

Get lock locally:

executing transactions

committing transactions

committed transactions

Apply updates

Pessimisic locking

session 1:

```
mysql> select * from t;
+----+-----+
| id | idd |
+----+-----+
|  1 |   0 |
|  2 |   0 |
|  3 |   0 |
+----+-----+
3 rows in set (0.00 sec)

mysql> set autocommit=0;
Query OK, 0 rows affected (0.00 sec)

mysql> update t set idd=2 where id=1;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings:
0

mysql> commit;
Query OK, 0 rows affected (0.00 sec)

mysql>
```

session 2:

```
9 rows in set (0.00 sec)

mysql> select * from t;
+----+-----+
| id | idd |
+----+-----+
|  1 |   0 |
|  2 |   0 |
|  3 |   0 |
+----+-----+
3 rows in set (0.00 sec)

mysql> set autocommit=0;
Query OK, 0 rows affected (0.00 sec)

mysql> update t set idd=1 where id=2;
ERROR 1205 (HY000): Lock wait timeout
 exceeded; try restarting transaction
mysql>
Query OK, 0 rows affected (0.00 sec)

mysql>
```

optimistic locking

Node1:

```
mysql> select * from t;
+----+------+
| id | idd  |
+----+------+
|  1 |    2 |
|  2 |    0 |
|  3 |    0 |
+----+------+
3 rows in set (0.00 sec)

mysql> set autocommit=0;
Query OK, 0 rows affected (0.00 sec)

mysql> update t set idd=2 where id=2;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> commit;
ERROR 1213 (40001): Deadlock found when tr
ying to get lock; try restarting transacti
on
mysql>
```

Node2:

```
affiliates. Other names may be trademarks of
owners.

Type 'help;' or '\h' for help. Type '\c' to c

mysql> use test;
Reading table information for completion of t
You can turn off this feature to get a quicke

Database changed
mysql> set autocommit=0;
Query OK, 0 rows affected (0.00 sec)

mysql> update t set idd=1 where id=2;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> commit;
Query OK, 0 rows affected (0.00 sec)

mysql>
```

Unilateral Abort

- Two Phase Commit
- Site Failure

```
 1: Execute t.
 2: On t commit request:
 3:   TO-bcast(R, ⟨wset(t), r_i⟩)
 4: Upon ⟨wset(t), r_d⟩ reception:
 5:   mutex.lock
 6:   status_t ← certify(wset(t), wslist_i)
 7:   if (status_t = commit) then
 8:     append(wslist_i, wset(t))
8a:   mutex.unlock
 9:     if (r_i ≠ r_d) then
10:       DB.apply(wset(t))
11:     DB.commit(t)


12:   else DB.abort(t)
13:   mutex.unlock
14:   if (r_i = r_d) then
15:     send(c, status_t)
```

a) SI CBR protocol.

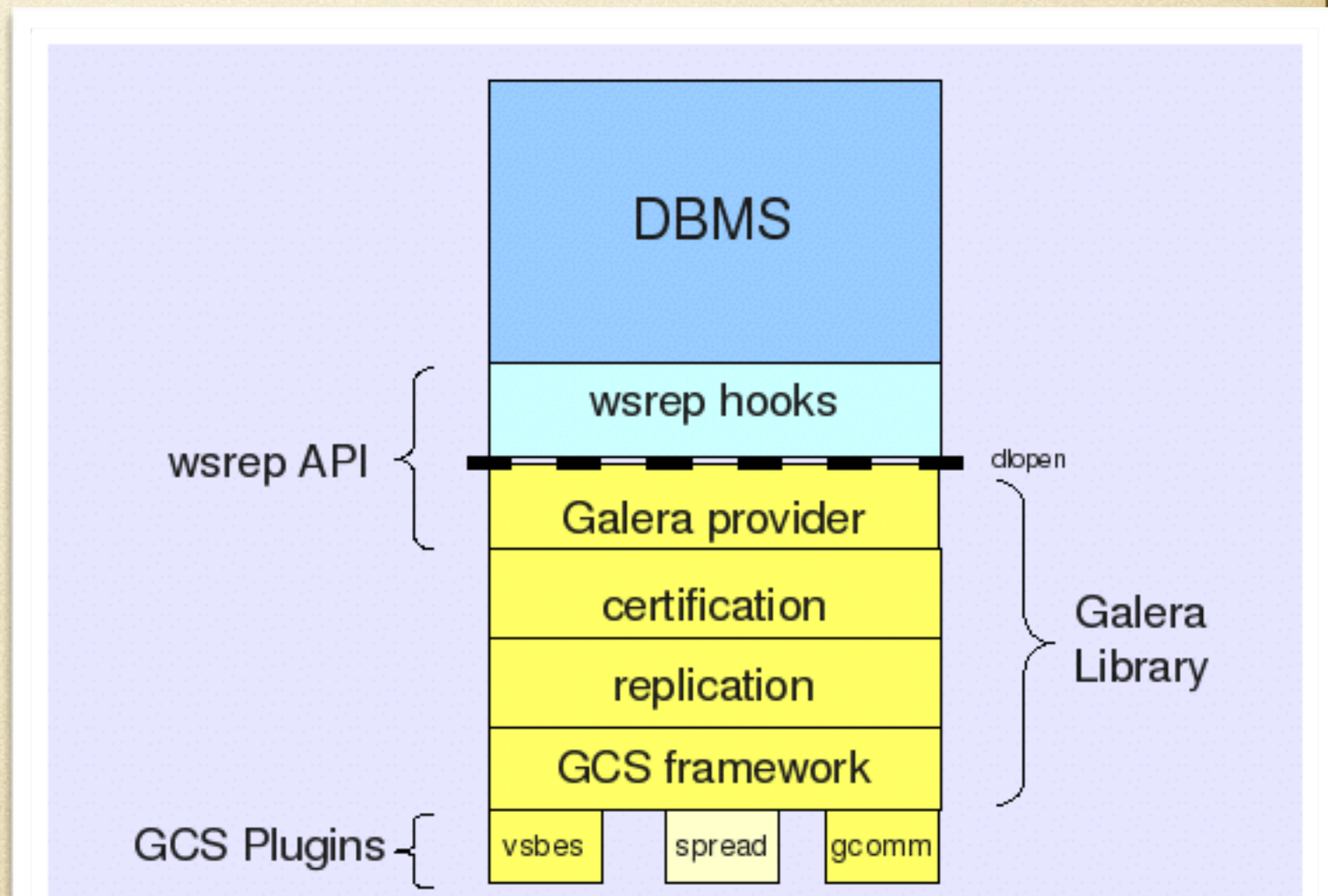
```
 1: Execute t.
 2: On t commit request:
 3:   TO-bcast(R, ⟨wset(t), r_i⟩)
 4: Upon ⟨wset(t), r_d⟩ reception:
 5:   mutex.lock
 6:   status_t ← certify(wset(t), wslist_i)
 7:   if (status_t = commit) then
 8:     append(wslist_i, wset(t))

 9:     if (r_i ≠ r_d) then
10:       DB.apply(wset(t))
11:     status_t ← DB.commit(t)
11a:    if (status_t = abort) then
11b:      remove(wslist_i, wset(t))
12:   else DB.abort(t)
13:   mutex.unlock
14:   if (r_i = r_d) then
15:     send(c, status_t)
```

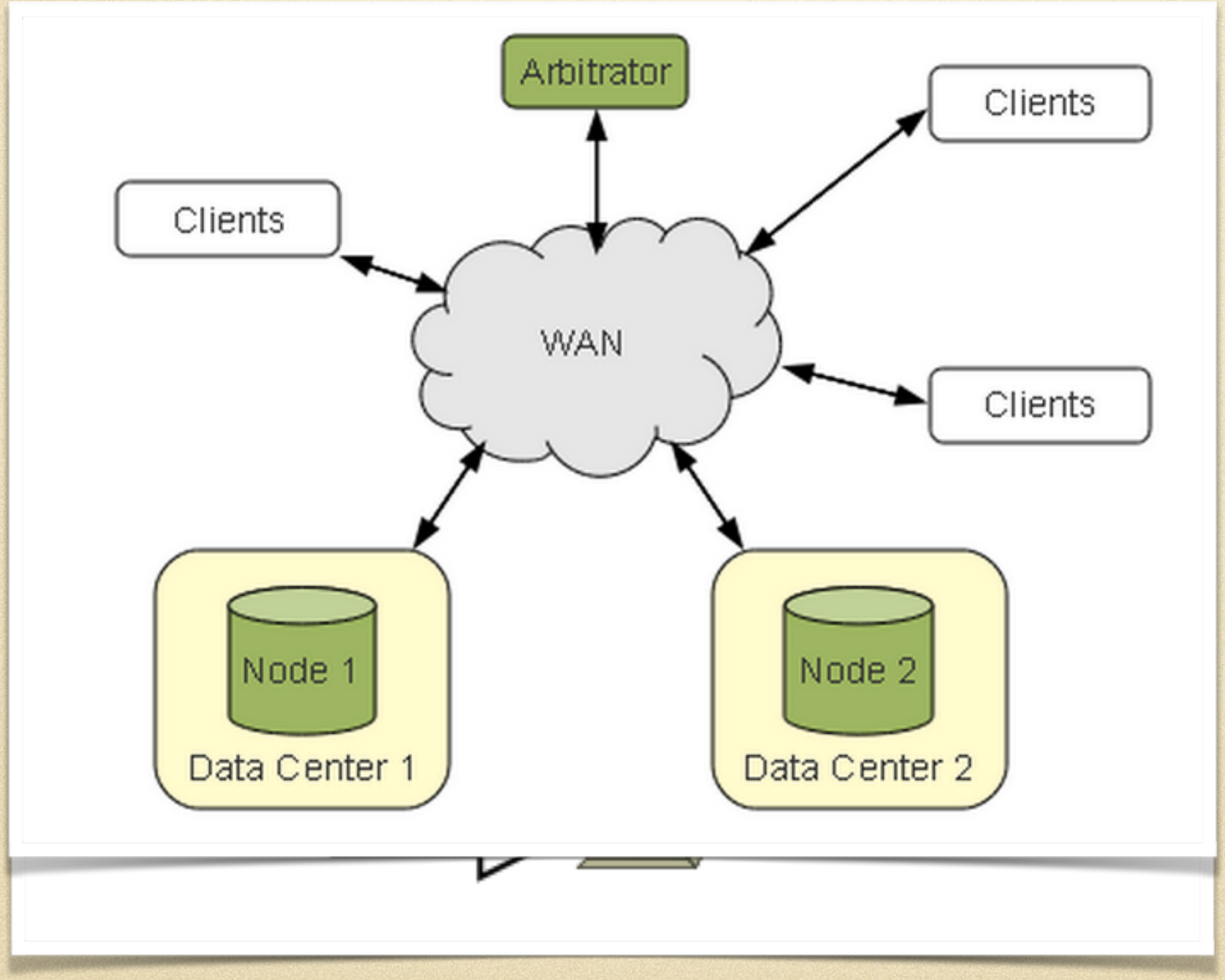
b) Extended SI CBR protocol.

Percona xtradb cluster

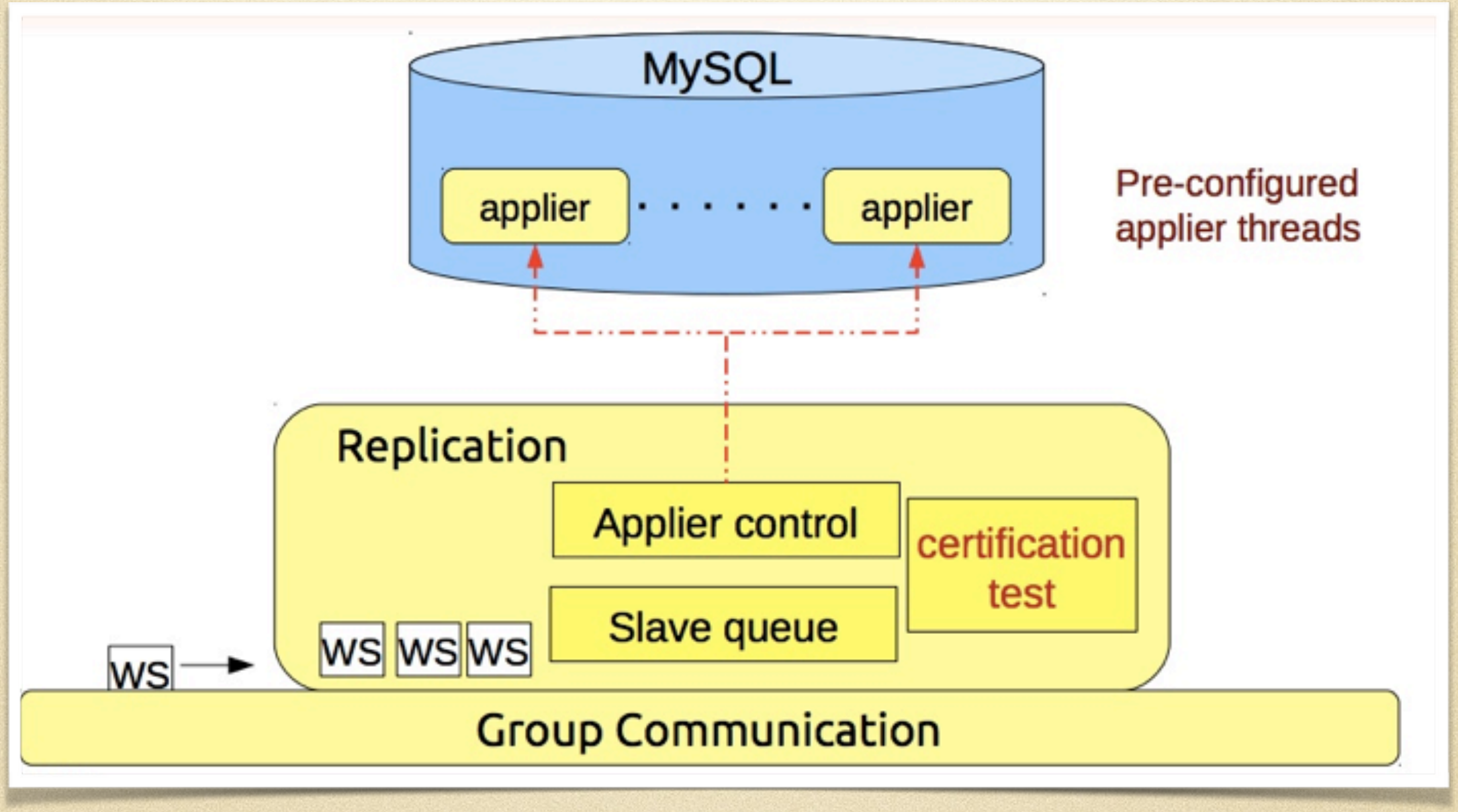
- Percona XtraDB Server
- WSREP API (write set replication patches)
- Galera (galera library)



Group Communication



Group Communication



GTID

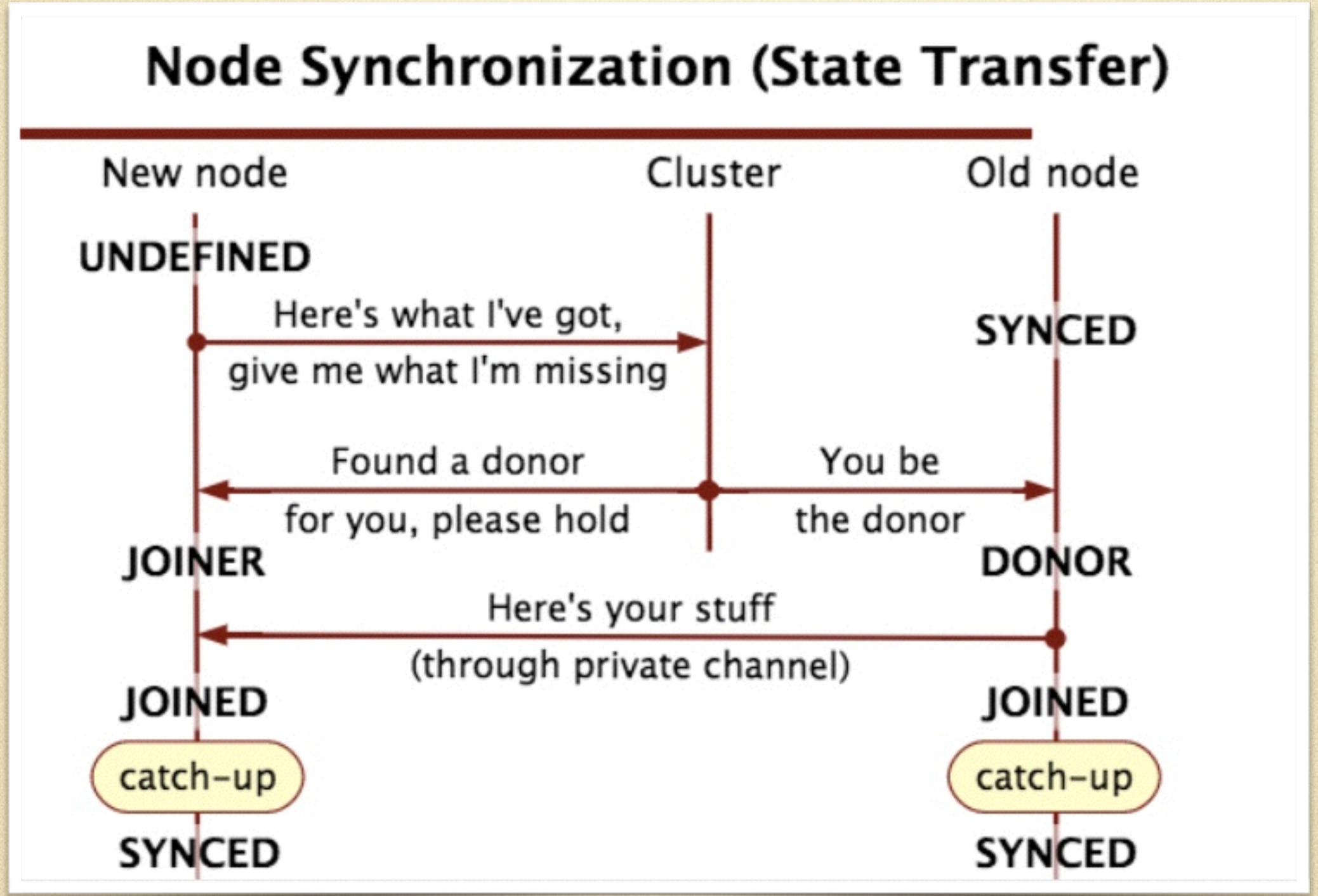
- UUID : state and the sequence of changes

  0e3bc86a-0014-11e3-a876-e7050fefda45

- GTID: UUID +Offset position

  0e3bc86a-0014-11e3-a876-e7050fefda45:17121

State Transfer



State Transfer

- IST: Incremental State Transfer
- SST: State Snapshot Transfer

method	speed	blocks the donor	can be done on live node?	logical/physical	requires root access to MySQL server?
mysqldump	slow	yes	yes	logical	both donor and joiner
rsync	fastest	yes	no	physical	none
xtrabackup	fast	for a very short time	no	physical	donor only

Writeset Cache

- Memory
- Disk:128M

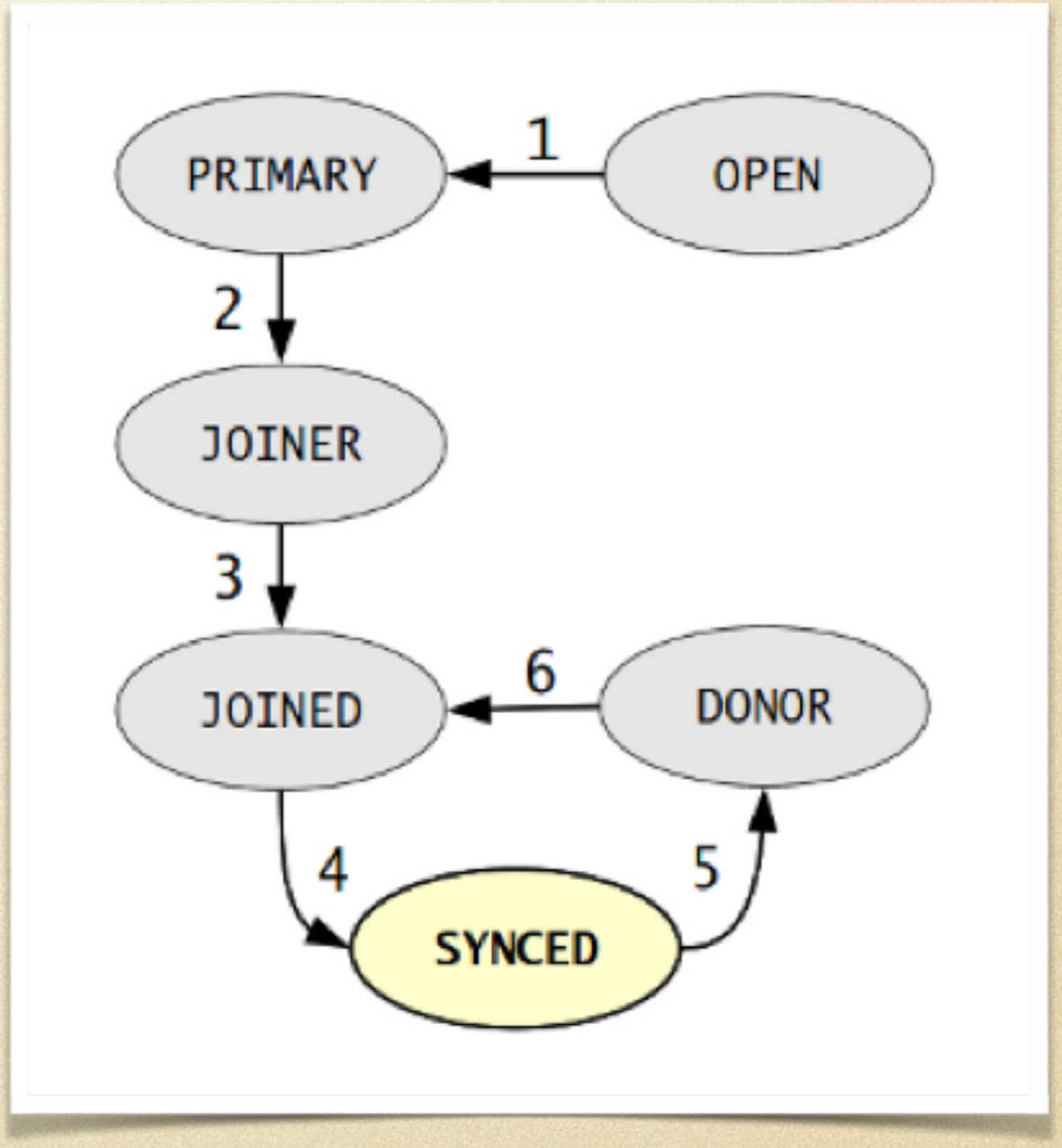
  ring buffer

  on-demand

gcache.size=10G

Node States

- New node
- Node failure
- IST,SST



Node Manage

Startup

- First node：

  service mysql start --wsrep-cluster-address=gcomm:/ /

  service mysql bootstrap-pxc

- Assign donor： service mysql start --wsrep-sst-donor=node3

- Normal： service mysql start

Node Manage

Add Node

wsrep_cluster_address=gcomm://10.10.58.168:4030,10.10.58.232:403

wsrep_sst_receive_address=10.10.58.209:4020

wsrep_cluster_name=PXCS_10-10-57-2

wsrep_provider_options=ist.recv_addr = tcp://10.10.58.209:4031;

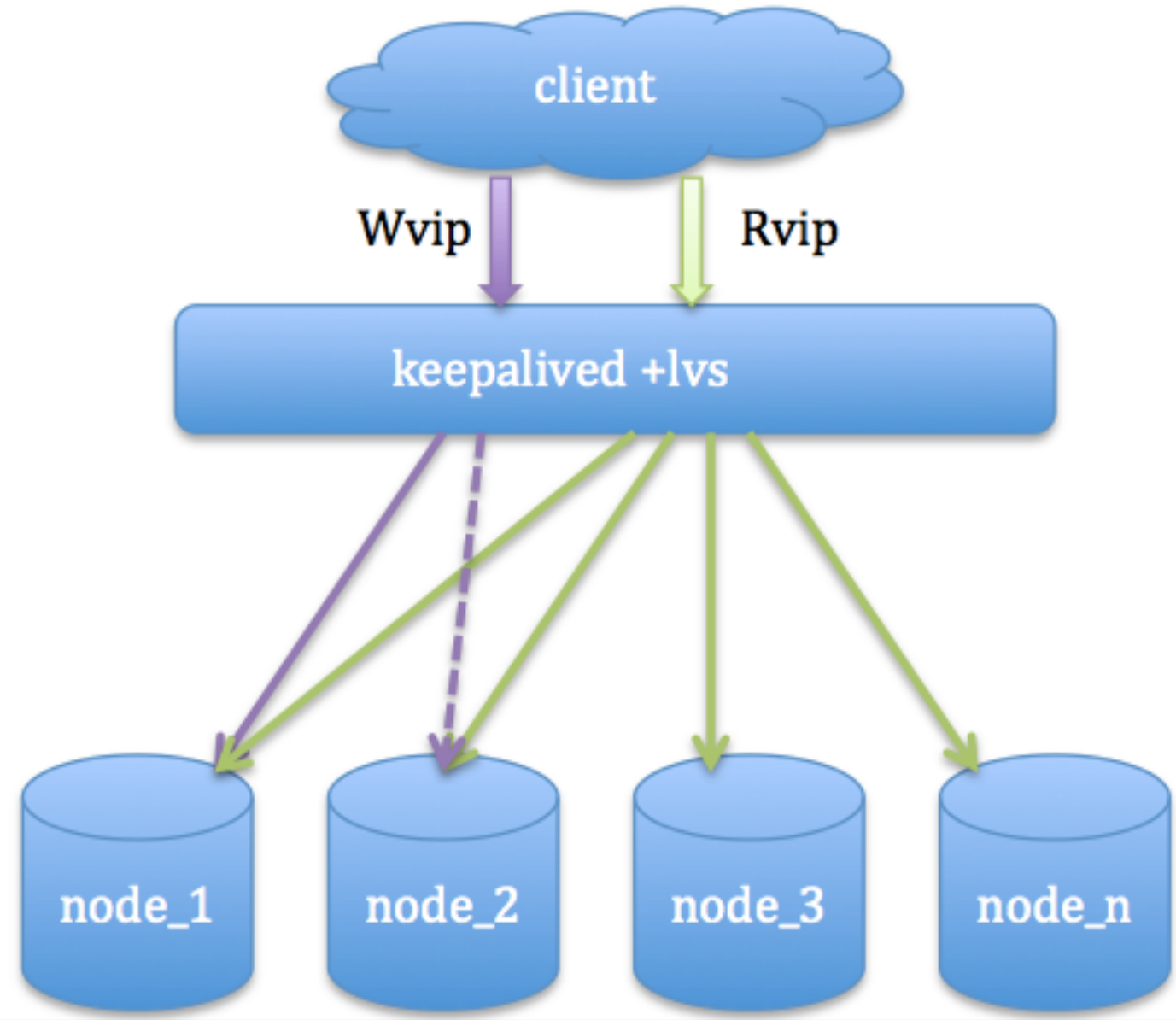
wsrep_node_name=PXCN_10-10-58-209

wsrep_sst_method=xtrabackup

wsrep_sst_auth=user:pwd

Cluster Manage

keepalived 实现 vip 代理和切换。基于 db 分配读写 vip，lvs 实现负载均衡。



Rolling Schema Upgrade

- TOI：Total Order Isolation
- RSU：Rolling Schema Upgrade

Failure Detection

evs.keepalive_period <=
evs.inactive_check_period <=
evs.suspect_timeout <=
evs.inactive_timeout <=
evs.consensus_timeout

Weighted Quorum

$$(\text{sum}(p_i * w_i) - \text{sum}(l_i * w_i))/2 < \text{sum}(m_i * w_i)$$

- p_i - members of the last seen primary component
- l_i - members that are known to have left gracefully
- m_i - current component members
- w_i - member weights

Monitor

wsrep_cluster_status	wsrep_cluster_status
wsrep_cluster_size	3
wsrep_cluster_state_uuid	e2c9a15e-5485-11e0-0800-6bbb637e7211
wsrep_incoming_addresse	10.10.58.168:3306,10.10.58.209:3306,10.10.58.2
wsrep_local_state_comment	Synced
wsrep_local_state	4
wsrep_ready	ON
wsrep_local_state_uuid	e2c9a15e-5485-11e0-0800-6bbb637e7211

Monitor

wsrep_replicated	16109
wsrep_received	17831
wsrep_local_cert_failures	333
wsrep_local_bf_aborts	960
wsrep_local_send_queue_avg	0.145
wsrep_local_recv_queue_avg	3.348452
wsrep_flow_control_paused	0.184353
wsrep_cert_deps_distance	23.88889
wsrep_commit_window	0

Inconsistency

root@vm-10-10-58-232:~

```
Database changed
mysql> select * from t;
+----+------+
| id | idd  |
+----+------+
|  1 |    0 |
|  2 |    0 |
|  3 |    0 |
|  4 |    0 |
|  5 |    0 |
|  6 |    0 |
+----+------+
6 rows in set (0.00 sec)

mysql> select * from t;
ERROR 2013 (HY000): Lost connection to MySQL server during query
mysql>
```

root@vm-10-10-58-168:~

```
4 rows in set (0.01 sec)

mysql> insert into t values(4,0);
Query OK, 1 row affected (0.01 sec),

mysql> select * from t;
+----+------+
| id | idd  |
+----+------+
|  1 |    0 |
|  2 |    0 |
|  3 |    0 |
|  4 |    0 |
|  6 |    0 |
+----+------+
5 rows in set (0.00 sec)
```

```
131028 13:21:10 [ERROR] WSREP: Failed to apply trx: source: d8195058-3
f8b-11e3-9132-cafb3361d482 version: 2 local: 0 state: APPLYING flags:
1 conn_id: 1747 trx_id: 142117 seqnos (l: 14, g: 14190, s: 14189, d: 1
4189, ts: 13829376703567839888)
131028 13:21:10 [ERROR] WSREP: Failed to apply app buffer: seqno: 1419
0, status: WSREP_FATAL
         at galera/src/replicator_smm.cpp:apply_wscoll():52
         at galera/src/replicator_smm.cpp:apply_trx_ws():118
131028 13:21:10 [ERROR] WSREP: Node consistency compromized, aborting.
..
```

Inconsistency

```
7 rows in set (0.00 sec)

mysql> select * from t;
+----+------+
| id | idd  |
+----+------+
|  1 |    0 |
|  2 |    0 |
|  3 |    0 |
|  4 |    2 |
|  5 |    2 |
|  6 |    2 |
|  7 |    2 |
+----+------+
7 rows in set (0.00 sec)

mysql>
mysql> update t set idd=2 where id=2;
Query OK, 1 row affected (0.03 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql>
```

```
mysql> select * from t;
+----+------+
| id | idd  |
+----+------+
|  1 |    0 |
|  3 |    0 |
|  4 |    2 |
|  5 |    2 |
|  6 |    2 |
|  7 |    2 |
+----+------+
6 rows in set (0.01 sec)

mysql> select * from t;
ERROR 2006 (HY000): MySQL server has gone away
No connection. Trying to reconnect...
ERROR 2002 (HY000): Can't connect to local MySQL server through socket
/DATA/mysql/mysql.sock' (111)
ERROR:
Can't connect to the server

mysql>
```

Inconsistency

SET wsrep_on=0;

- **binlog_format=ROW**
- **log-bin = mysql-bin**
- **writeset collection**

Tips

- Primary key
- InnoDB
- No myisam
- No forien key

Tips

- Small transaction
- innodb_flush_log_at_trx_commit=2
- wsrep_slave_threads=32
- query_cache_size=0
- innodb_doublewrite_buffer=1

Reference

1.CAP：http://en.wikipedia.org/wiki/CAP_theorem

2.The database state machine and group communication issues

http://infoscience.epfl.ch/record/32566/files/EPFL_TH2090.pdf

3.Integrity Dangers in Certification-Based Replication Protocols：

http://web.iti.upv.es/~fmunyoz/research/pdf/TR-ITI-ITE-0813.pdf

4.Comparison of Database Replication Techniques Based on Total Order Broadcast

5.Galera :http://www.codership.com/wiki/doku.php?id=info

6.PXC 运维总结

7.PXC 问题追踪http://wiki.no.sohu.com/index.php/PXC%E9%97%AE%E9%A2%98%E8%BF%BD%E8%B8%AA