# Summary of CAP Twelve Years Later: How the "Rules" Have Changed

*By S. Xiao Fernández Marín*

## 1 Summary

The CAP theorem says that a DS can have at most two of these three properties: *Consistency*, having a single up-to-date copy among nodes, where everyone sees the same values and changes are atomic, not in between states (not the same as ACID consistency); *Availability* of the data, the system will always respond to r/w request and there is no state where the system is unavailable; and *Partition tolerance*, any message loss can be modeled as partitioning (between a node and the rest or group of nodes), it is assumed to be temporary (if not, its considered two different systems) and unavoidable effect of networking, it is impossible to avoid completely.

Two out of three: **Consistent and available:** Classic DBMS model, where ignored networking and it is correct for centralised systems, the partitions are always possible in a DS as a general model of message delivery errors. the CA is possible most of the times as partitioning is realy rare.

**Consistent and partition tolerant:** Refuse all requests so availability is not guaranteed and it is R-only mode, it the state is consistent then the partitioning occurs, The requests are delayed until the system is no longer partitioned.

**Available and partition tolerant:** The point of CAP theorem. it has weaker consistency models as the conflict resolves when the partitioning ends and it is popular in large-scale services as the more nodes, the higher probability of network errors and some nodes or network links will always be down

**In case of P** (If a message sent over the network doesn't arrive) **weaken C and/or A**:
1. Cancel and sacrifice A, Refuse/hold back some operations
2. Proceed and potentially sacrifice C, Weaken consistency levels for some operations

**Non-binary consistency**
Weak vs. strong consistency
– Reduce isolation level, read uncommitted values
– Session inconsistency

• Eventual consistency
– Temporary inconsistency

• Include (in)consistency information in interface
– The client may be able to deal with it

• Consistency is maintained unless there are updates

**Non-binary availability**
• Reduced availability
– Read-only
– Only certain types of updates

• How long can the response time be?
– Maybe the partitioning ends before the service has to reply

• Delay operations
– Record changes/actions but execute later

• Availability in some partitions
– Majority of nodes
– Majority/all of the nodes for one data object

**Non-binary partition tolerance**

- Latency can be between 0 and infinity
  - Where's the line between high latency and partitioning?

- Partition detection
  - How many retries?
  - How long timeout?

- Partitioning vs. response time
  - Can partitioning occur before the response time has been exceeded?

**ACID vs. BASE**

- Atomicity, consistency, isolation, durability
  - Traditional DBMSs
  - Focus on consistency over availability

- Basically available, soft state, eventually consistent
  - Typical for large-scale network services
  - Popular among NoSQL systems
  - Focus on availability over consistency: Applications/users have to handle inconsistencies