# Clocks, order of events

Source:

- IB Distributed System
- y2014p5q7

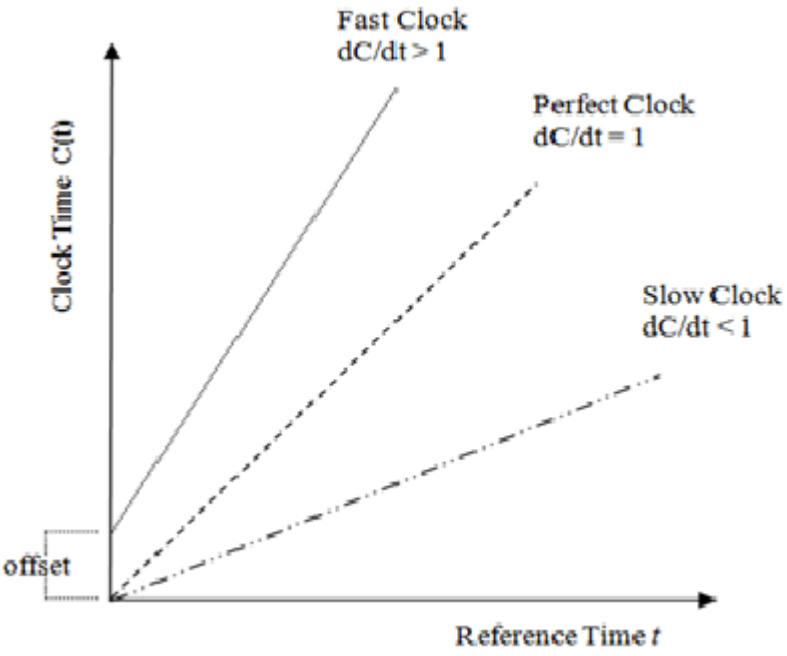
| Clock | Physical | Logical |
|---|---|---|
| measure | seconds | events with causality |
| example | analogue/mechanic<br>digital: Quartz (drift)<br>Atomic, GPS | Lamport<br>Vector |

## Physical Clock

### Time-of-day and Monotonic

| Physical Clock | Real Time | Monotonic |
|---|---|---|
| since | a fixed date time | arbitrary point (start-up) |
| correction | $slew \implies step$ | always slew *forward* |
| behaviour | human readable;  compare ts among nodes if sync | measure elapsed time on a single node |
| usage | certificate time | measure intervals / timeouts |

### Synchronization



The time of a clock in a machine $p$ is $C_p(t)$, frequency/rate of a clock is $C'_p(t)$

- perfect clock $\Leftrightarrow C_p(t) = t$

Clock skew / offset

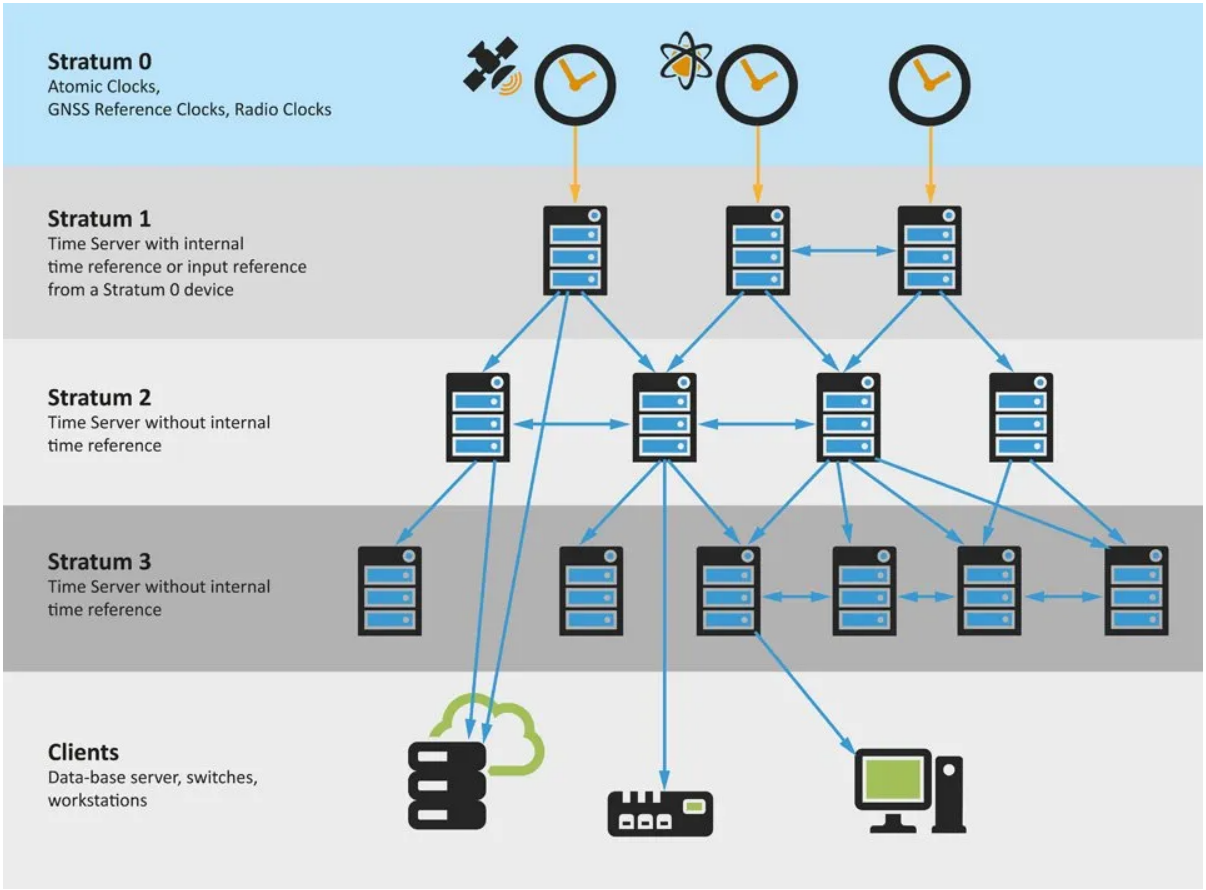
- the difference between the time on two clocks
- skew $\Delta_s = C_a(t) - C_b(t)$ (ms)
- measure: RTT $\delta$, Cristian's Algorithm wiki
    - assumption:
        - symmetric latency
        - not consider the derivative of the clock (i.e. drift) or higher derivatives
- correction
    - as $\Delta_s$ increases, $slew \implies step \implies panic$

Clock drift

- the difference of clock rate of oscillations / ticks
- drift $\Delta_d = C'_a(t) - C'_b(t) = \Delta_s(t_1) - \Delta_s(t_2)$ (ms/day, parts per million)
    - affected by temperature, etc.
- measure: Cristian's Algorithm twice
    - assumption
        - symmetric latency
        - not considering the second or higher derivatives of the clock

NTP / PTP ( Stratum 0-2 )

- Less accurate synchronization
    - Time source (higher stratum)
    - Assumption of Cristian's Algorithm




## Logical Clock

- causal / happen-before dependency $e_1 \to e_2$
  - $e_1$ and $e_2$ occurred at the same node, different by execution time
  - $e_1$ is sending message to $e_2$
  - transitivity, $\exists e_3.(e_1 \to e_3) \wedge (e_3 \to e_2) \implies e_1 \to e_2$.
  - (strict) partial order, asymmetric, undefined when race condition has occurred $e_1 || e_2$
- logical clock timestamp is consistency with causal dependency
  - But lamport may not get causal dependency of events back from logical timestamps.

$$e_1 \to e_2 \implies T(e_1) < T(e_2)$$

| | **Lamport** | **Vector** |
|---|---|---|
| format | $(N(e), L(e))$<br>$(i, Seq)$ | $\langle N_1, ..., N_n \rangle$<br>$V(e) = \langle t_1, ..., t_n \rangle$ |
| order | $total \prec$ | $partial < \|\|$ |
| timestamp | scalar | vector |
| | $\implies$ | $\iff$ |
| initial | $(i, 0)$ | $\langle 0, ..., 0, ..., 0 \rangle$ |
| event occur | $(i, t) \to (i, t+1)$ | $T_V[i] := T_V[i] + 1$ |

| | Lamport | Vector |
|---|---|---|
| $receive(t'/T', m)$ | $t := max(t, t') + 1$ | $T_V := max_j(T_V, T')$<br>$T_V[i] := T_V[i] + 1$ |
| broadcast | FIFO of each sender | Causal |