ETHERNOVIA

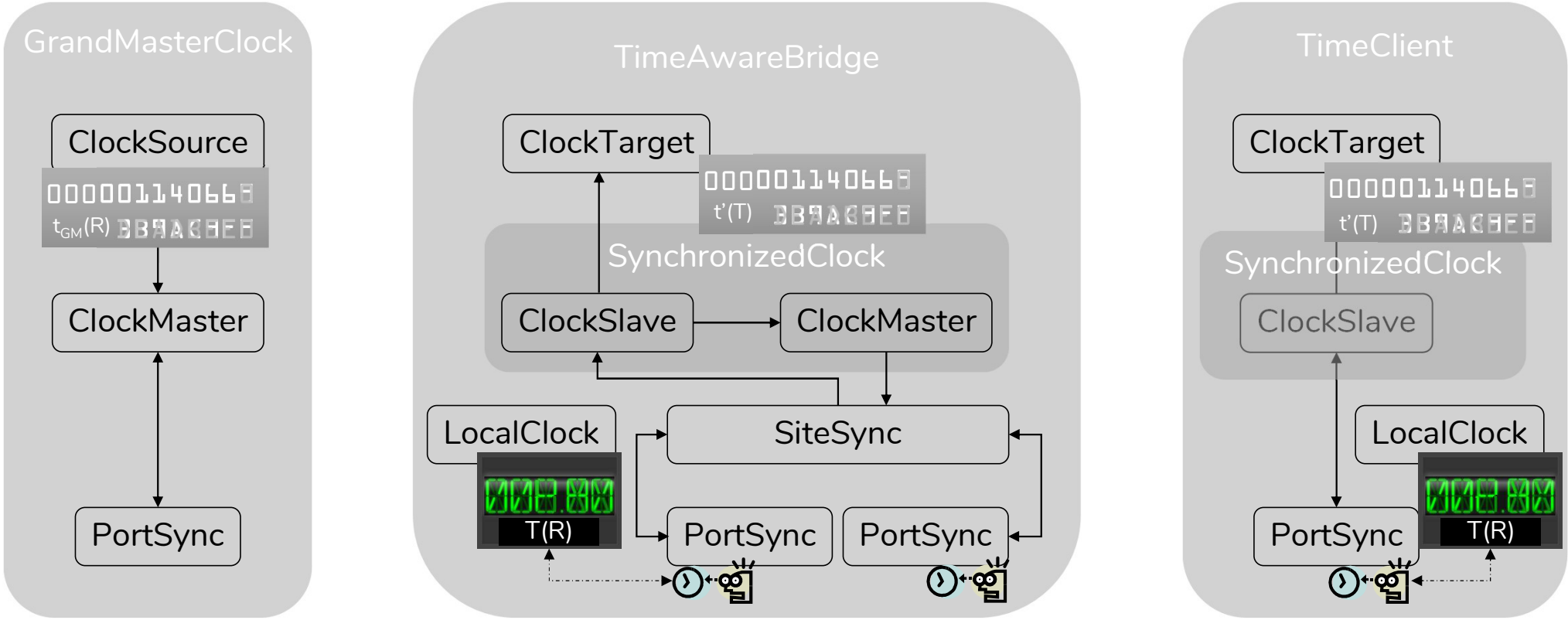
TRANSFORMING HOW CARS OF THE FUTURE ARE BUILT

IEEE802.1DG – GPTP | 2020-09-23

IEEE contribution

ETHERNOVIA
TRANSFORMING HOW CARS OF THE FUTURE ARE BUILT

Symbols and Nomenclature

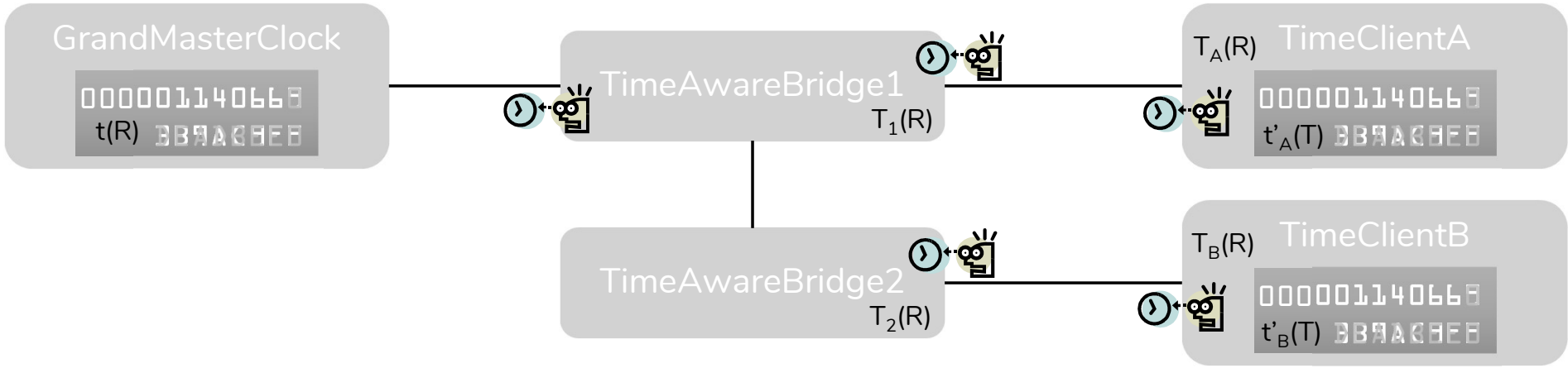


T ... LocalClock (free running)

t' ... SynchronizedClock (syntonous with GM)

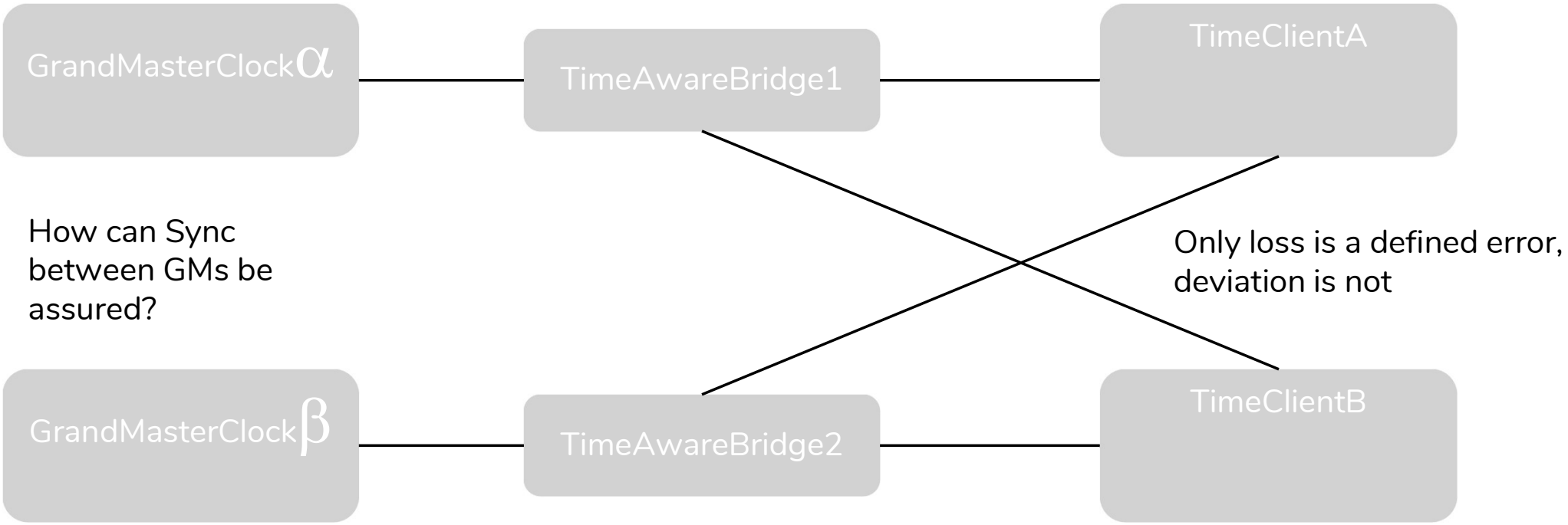
ETHERNOVIA
TRANSFORMING HOW CARS OF THE FUTURE ARE BUILT

Problem Description



- gPTP-Sync-Messages (t') sent from GM to the Bridges/Clients
- pDelay-Messages based on Local-Clock (T) information
- How can we ensure – in safety terms – at any one point in Real-Time (R), the GM and all (relevant) clients have synchronized to the same clock-counter value, within an accepted accuracy?
  $t(R) \approx t'_A(R) \approx t'_B(R)$

Redundant GM/Client Connections



How can Sync between GMs be assured?

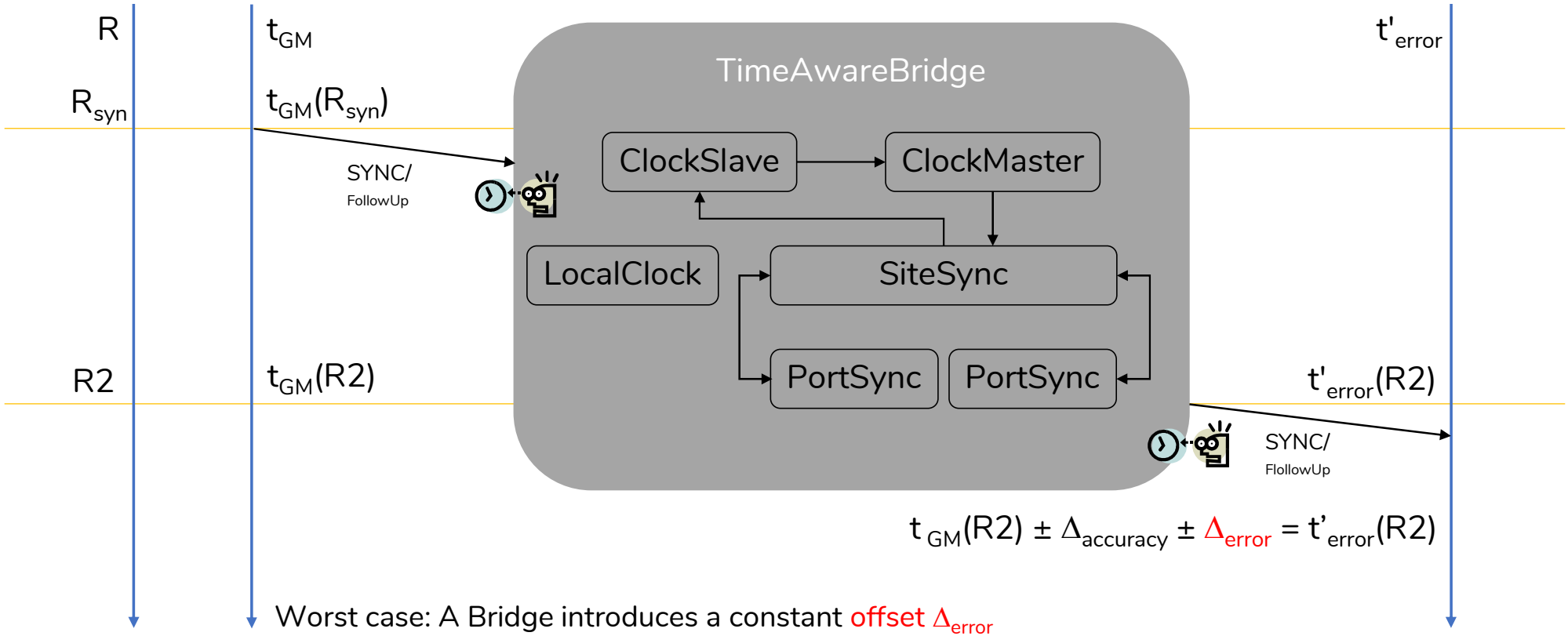
Only loss is a defined error, deviation is not

System can only start up, if <u>both</u> GMs are operational (and synchronized) -> reduced availability

Bridges can still introduce constant offset to one domain

Clients can compare domains locally but cannot compare each other

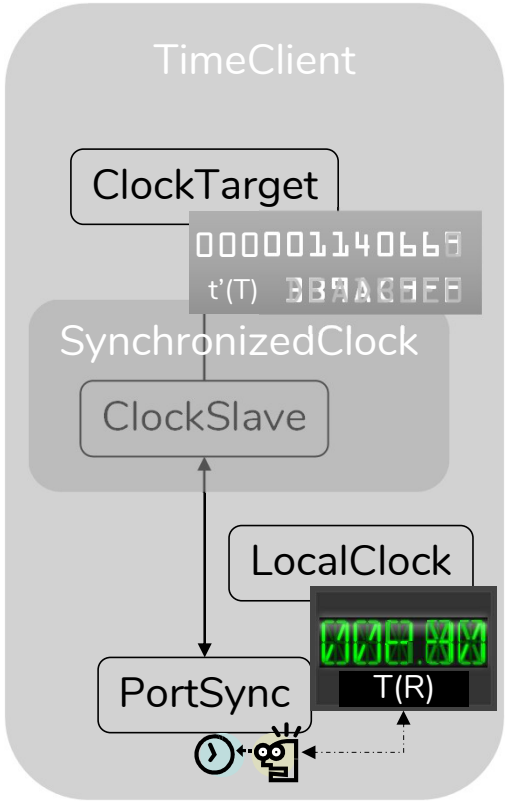
The “Byzantine” Bridge Error



Worst case: A Bridge introduces a constant offset Δ_{error} from the start-up (not a transient change)

ETHERNOVIA
TRANSFORMING HOW CARS OF THE FUTURE ARE BUILT

Delay within the Timing-Stack

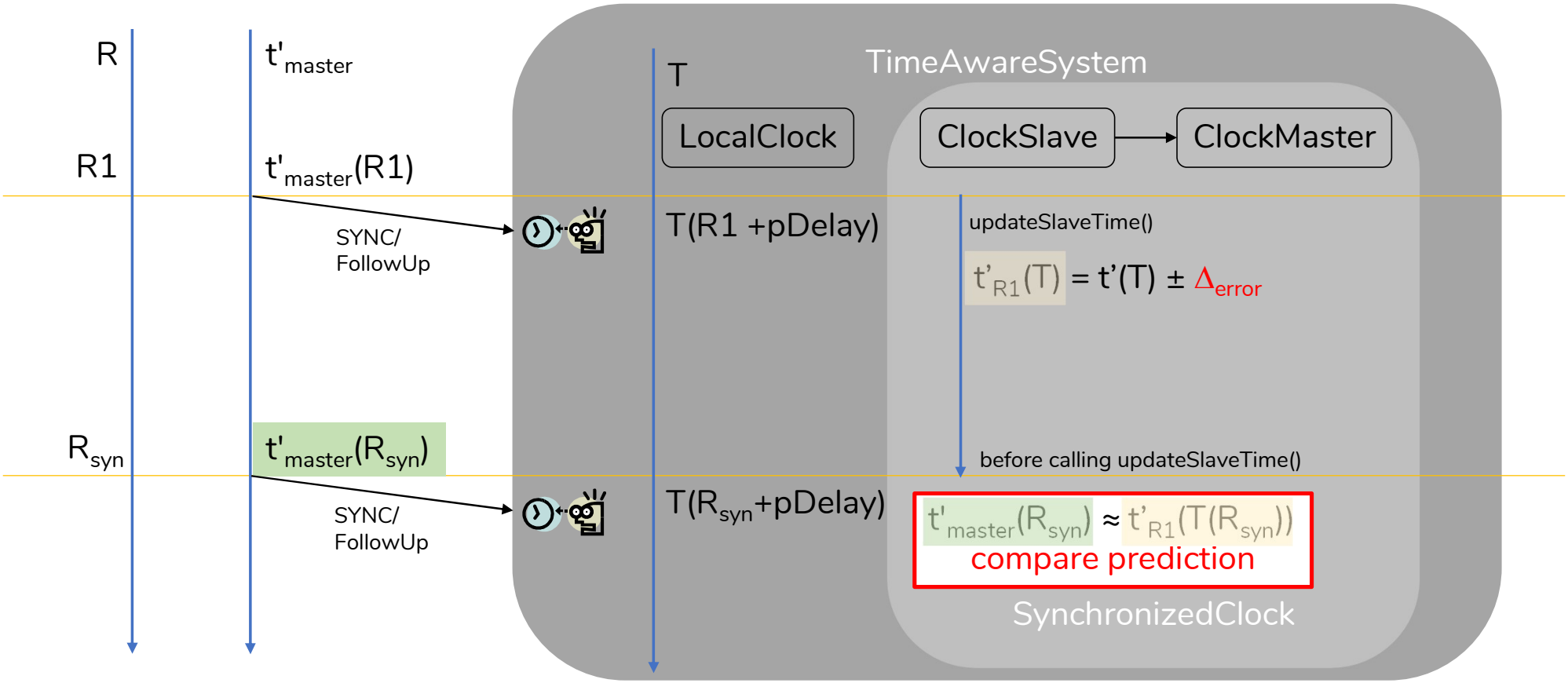


Does the application (ClockTarget) actually get the correct time?

Stack can introduce a delay

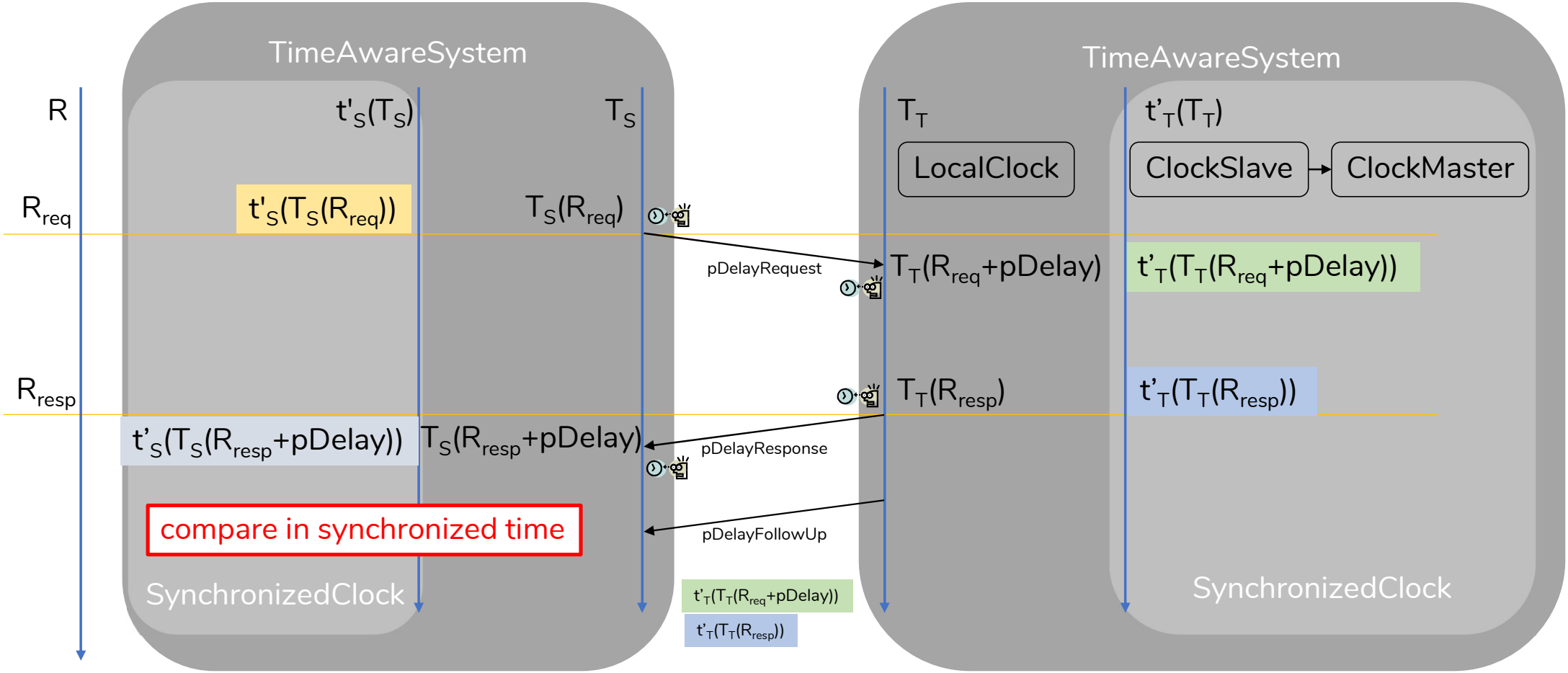
Local-Clock time-stamping can introduce a delay

Predict next originTimestamp



ETHERNOVIA
TRANSFORMING HOW CARS OF THE FUTURE ARE BUILT

Use pDelay to probe SynchronizedClock

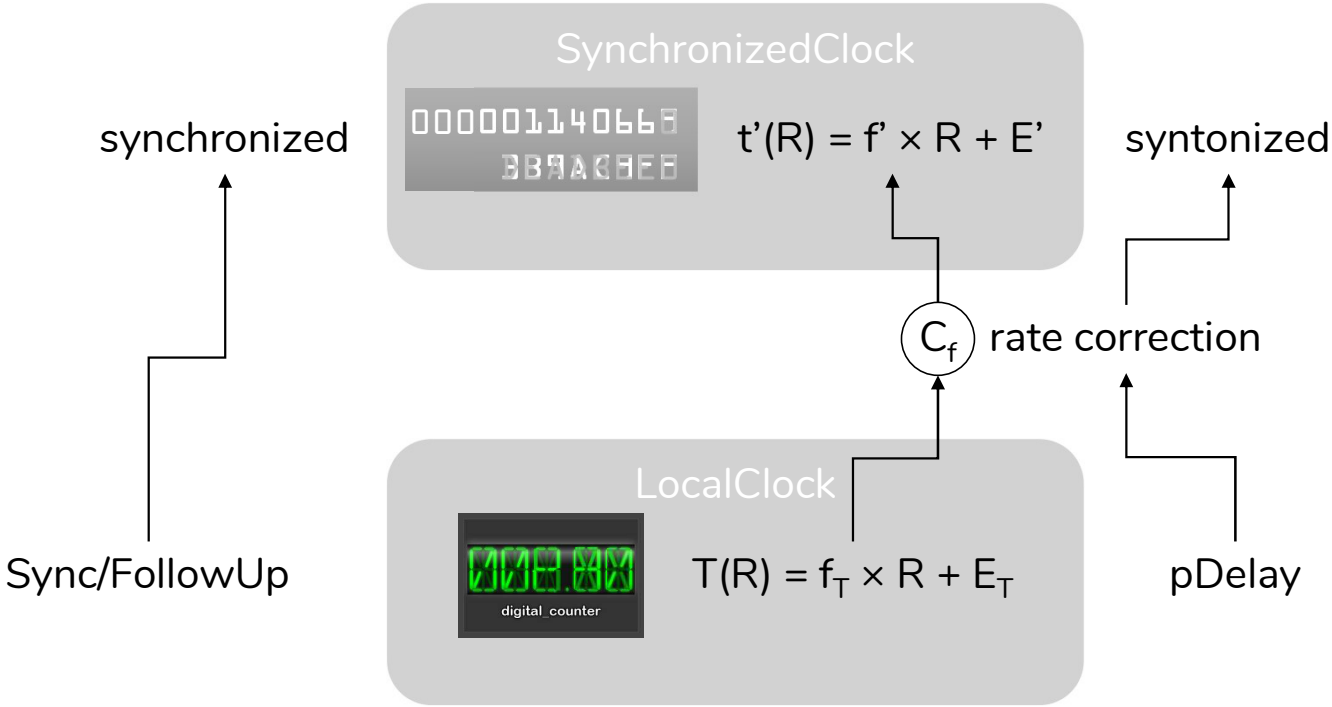


pDelay Source

pDelay Target

ETHERNOVIA
TRANSFORMING HOW CARS OF THE FUTURE ARE BUILT

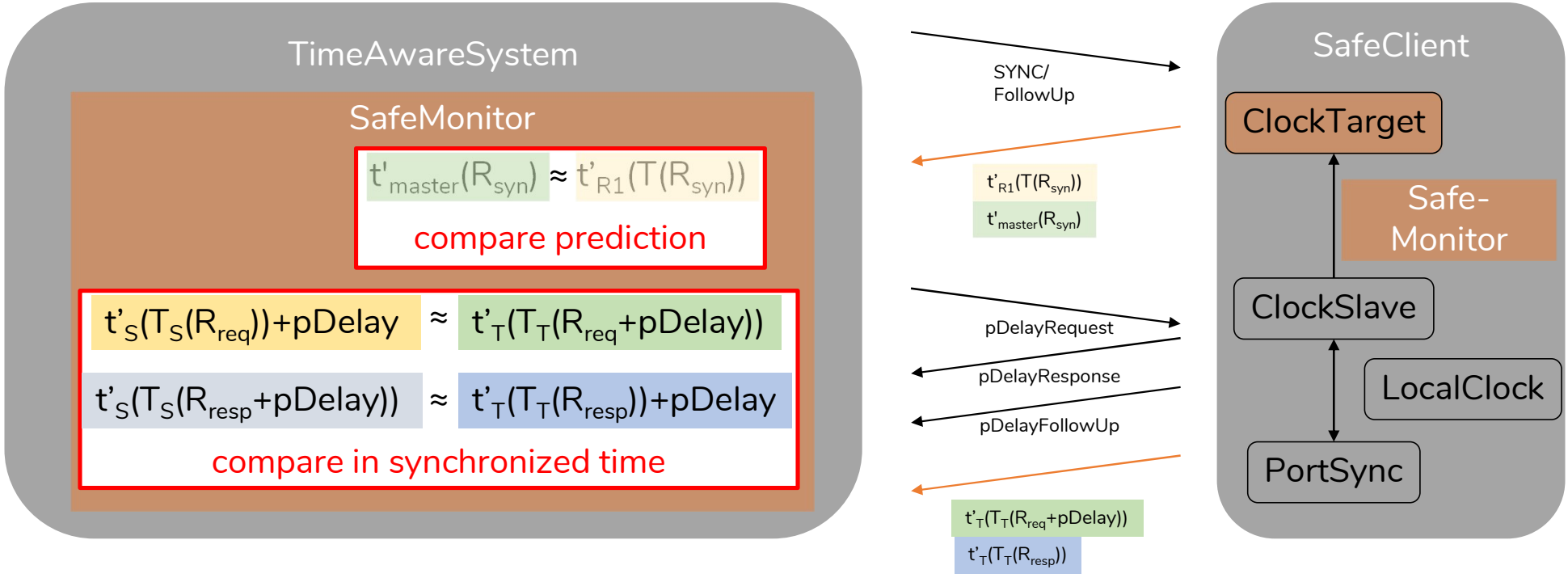
Why neighbourRateRatio from pDelay?



Improved Start-Up time:

- Can start on each link after link-up
- pDelay and neighbourRateRatio are already known when first Sync-Message arrives

Stations on a Link can monitor each other



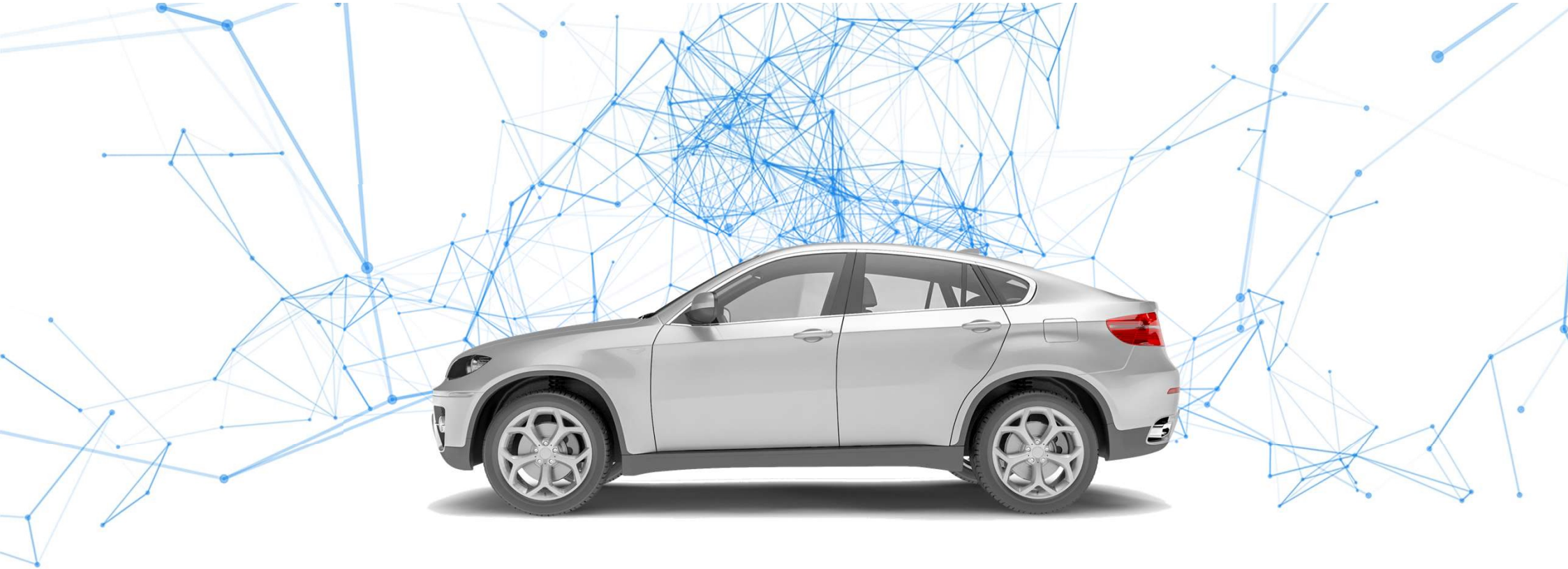
ETHERNOVIA
TRANSFORMING HOW CARS OF THE FUTURE ARE BUILT

List of Data-Points per Link

Event Message	Transmit time-stamp	Receive time-stamp
Sync-Message	$T_{master}(R_{sync})$ $t'_{master}(T(R_{sync}))$ [also in FollowUp-Msg]	$T_{client}(R_{sync}+pDelay)$ $t'_{client}(T(R_{sync}+pDelay))$
pDelayRequest-Message	$T_S(R_{req})$ $t'_S(T_S(R_{req}))$	$T_T(R_{req}+pDelay)$ [also in pDelayFollowUp-Msg] $t'_T(T_T(R_{req}+pDelay))$
pDelayResponse-Message	$T_T(R_{resp})$ [also in pDelayFollowUp-Msg] $t'_T(T_T(R_{resp}))$	$T_S(R_{resp}+pDelay)$ $t'_S(T_S(R_{resp}+pDelay))$

Autosar has **add**ed **interfaces** to record these time-tuples, should IEEE802.1AS and IEEE1588 follow?
The knowledge of Synchronized Time at each port requires Boundary Clocks
Do NOT specify use of the time-tuples in IEEE!

Boundary Clock: create new OriginTimeStamp for transmitted Sync-Message
Transparent Clock: keep incoming OriginTimeStamp, add CorrectionField in Local-Time to transmitted Sync-Message



THANK YOU

ETHERNOVIA | max.turner@ethernovia.com