Ensuring Reliable Networks TTTech

Fault-Tolerant Clock Synchronization

and thoughts on its use for
"Improved Grandmaster Changeover Time" in IEEE 802.1ASbt

Wilfried Steiner
Senior Research Engineer
wilfried.steiner@tttech.com

Overview

1. Introduction

2. Rationale for and use of fault-tolerant clock synchronization

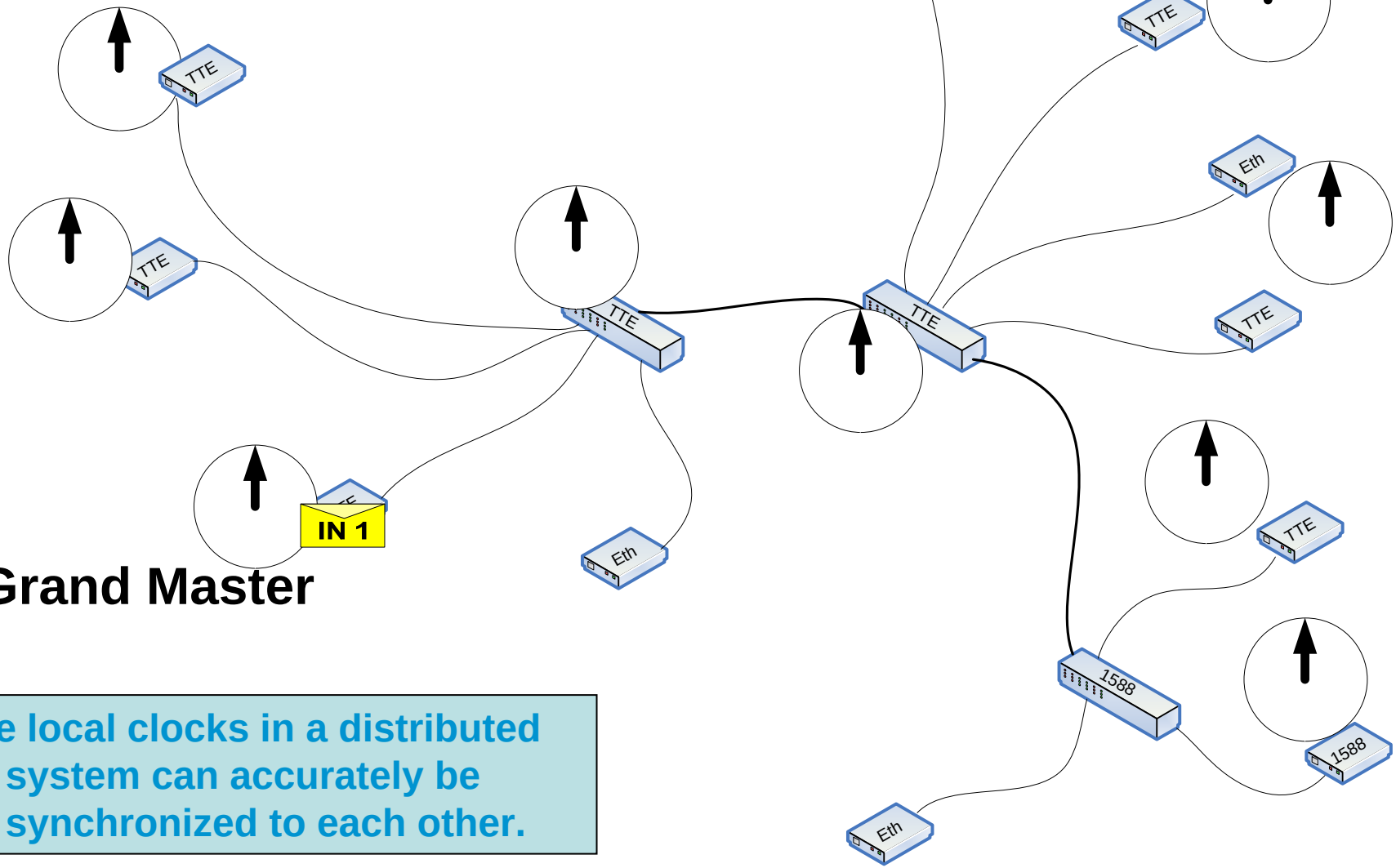
3. A short history on the development of fault-tolerant clock synchronization

4. Fault-tolerant clock synchronization and how it may be of benefit to IEEE 802.1AS

Overview

1. Introduction
2. Rationale for and use of fault-tolerant clock synchronization
3. A short history on the development of fault-tolerant clock synchronization
4. Fault-tolerant clock synchronization and how it may be of benefit to IEEE 802.1AS

Clock synchronization is a core building block of many RT Systems



The local clocks in a distributed system can accurately be synchronized to each other.

Our Understanding of Situation / Challenges

Background of IEEE work to date in clock synchronization

- Synchronization of clocks in a distributed system has several key benefits, e.g.,
  - distributed measurement of real-time durations
  - simultaneous activation of events
  - synchronized timestamps to reconstruct temporal order and to execute events in sequence/parallel
  - efficient utilization of shared resources, like the network itself
- Clock synchronization addresses phase synchronization and frequency synchronization.
- IEEE 802.1AS standardizes a master-slave clock synchronization algorithm with leader-election based on IEEE 1588.
- In the case of a disconnect of the master (named the Grand Master in 802.1AS) a new Grand Master is elected using the gPTP algorithm.

Key challenge in IEEE AS 802.1 regarding clock synchronization

- The changeover from one Grand Master to another Grand Master is not instantaneous and there is a possibility that the changeover causes non-continuous steps in the synchronized time, which may not be acceptable for certain applications.

Our Understanding on Current 802.1 Approach and Thinking on Solution

- One improvement to the “warm standby” strategy of 802.1AS is a “hot standby”.
- The system supports a primary and a secondary Grand Master, where both Grand Masters source synchronization messages.
- All slaves use the synchronization messages of the primary Grand Master.
- In the case of a failure of the primary Grand Master the slaves switch to the secondary Grand Master.
- While both primary and secondary masters are operational, the slaves can track the difference in their time and in the case of a changeover from the primary to the secondary, the slaves can apply the time difference gradually to avoid non-continuous steps in the synchronized time.

TTTech has long expertise in designing RT systems with Deterministic Clock Synchronization

Ensuring Reliable Networks TTTech

One idea of how to “attack” Grand Master changeover: Fault-Tolerant Clock Synchronization

Fault-Tolerant Clock Synchronization minimizes the changeover time between Grand Master clocks.

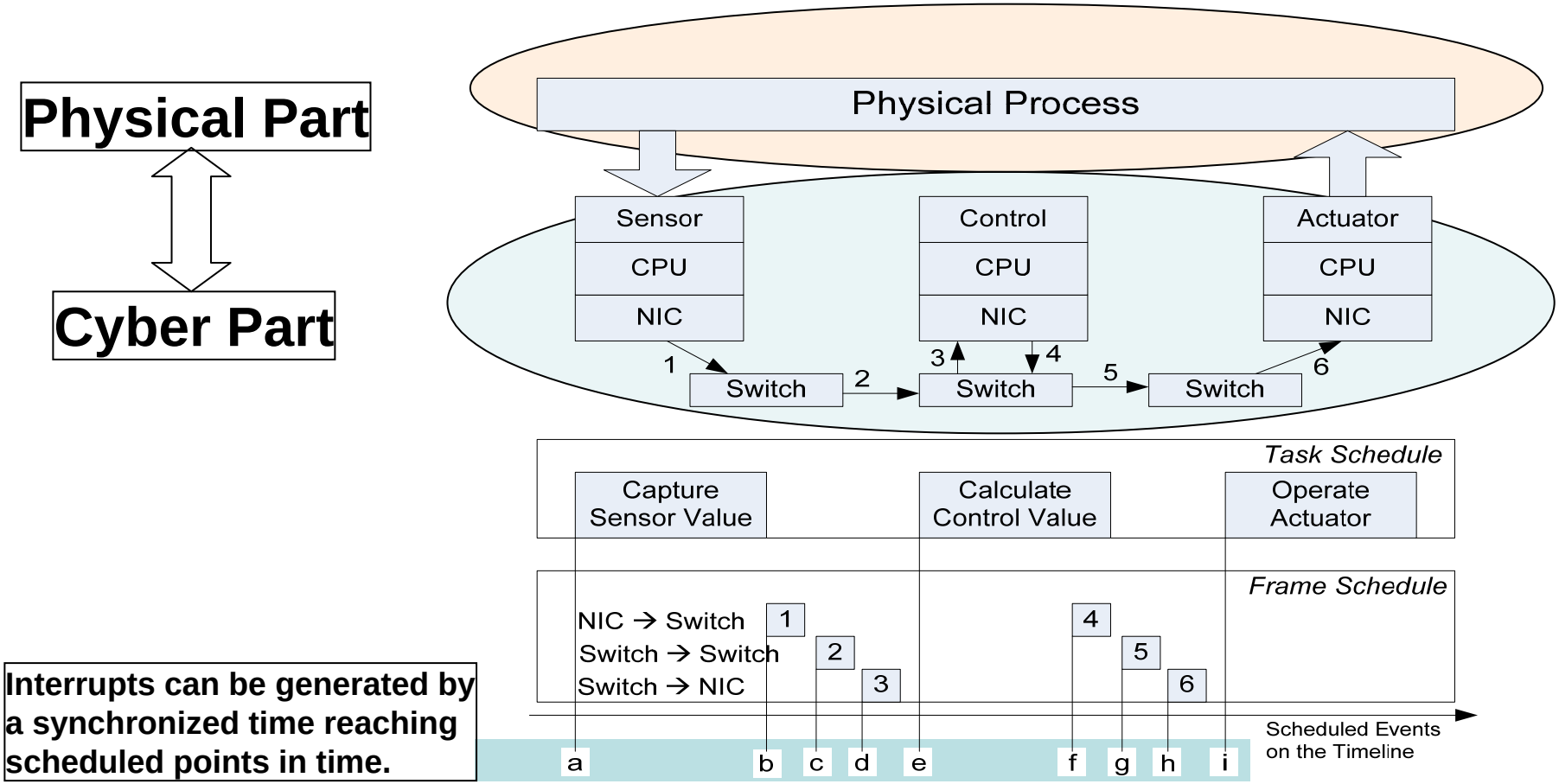
- →The synchronization of all Grand Master clocks is always taken into account in the synchronization process.
- →In case of a failure of a Grand Master clock there is no changeover at all.

Fault-Tolerant Clock Synchronization is the scope of this presentation.

Overview

1. Introduction
2. Rationale for and use of fault-tolerant clock synchronization
3. A short history on the development of fault-tolerant clock synchronization
4. Fault-tolerant clock synchronization and how it may be of benefit to IEEE 802.1AS

Distributed Cyber-Physical Systems

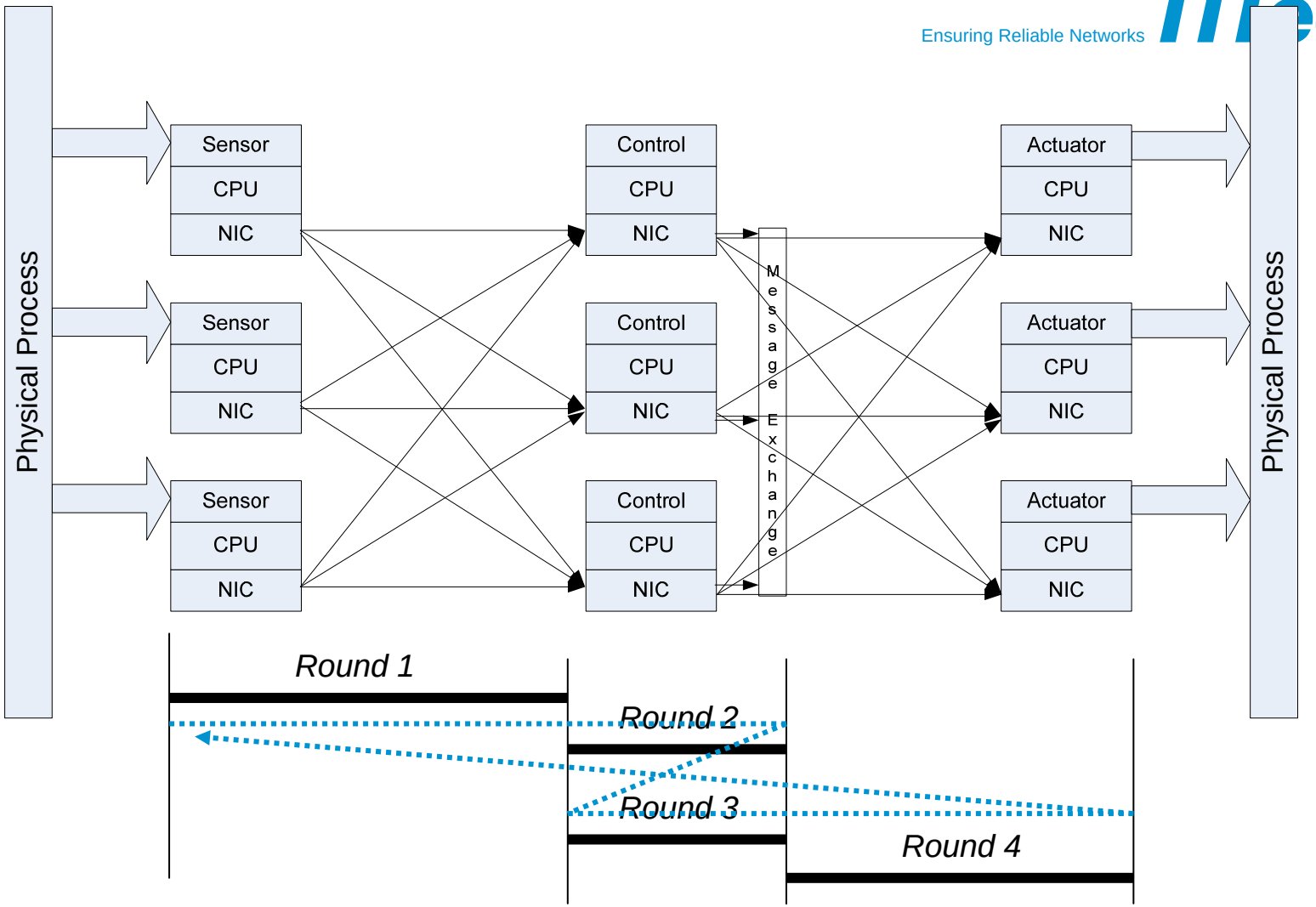


Interrupts can be generated by a synchronized time reaching scheduled points in time.

In several safety-relevant and safety-critical systems, synchronized time is a fundamental building block.

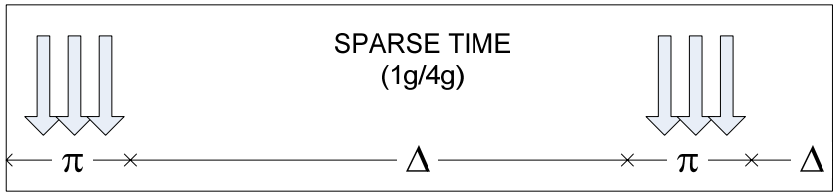
Fault-Tolerant Cyber Subsystem

Ensuring Reliable Networks TTTech



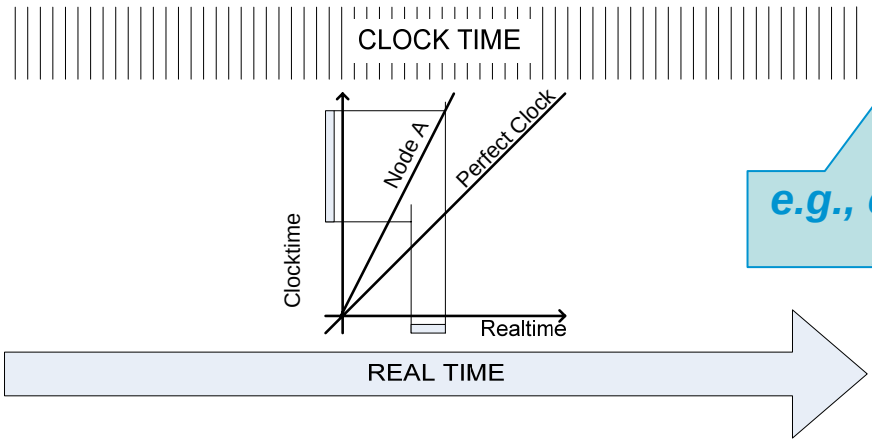
Synchronous Model of Computation (MoC)

Extended Application Interface for System Design



Sparse Time **is a design guideline according which a computer generates events only during pre-defined intervals.**

Global Time **groups a configurable number of ticks in Clock Time into a coarser tick granularity.**



Clock Time **is a simulation of Real Time inside a computer.**

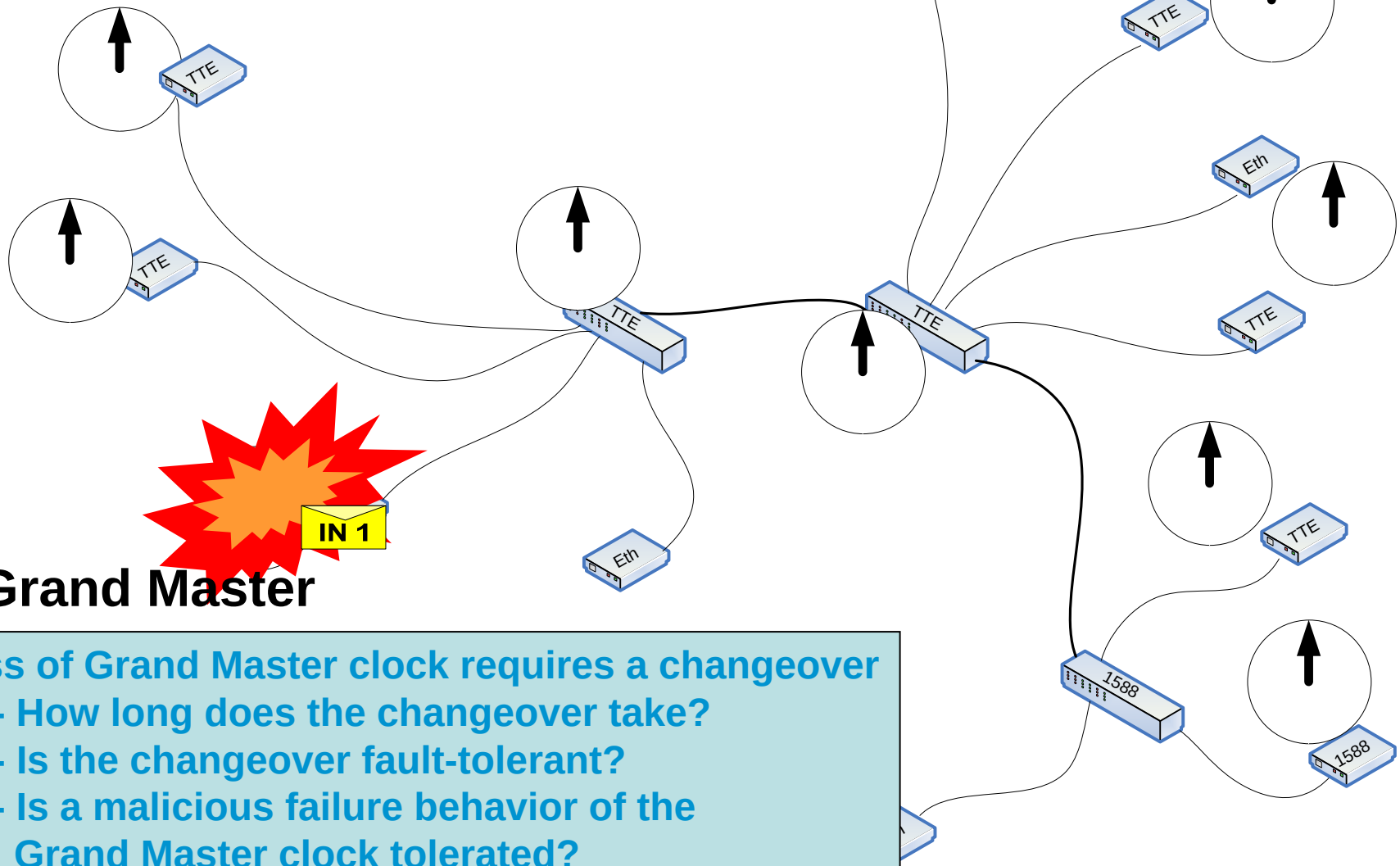
e.g., clockSlaveTime in 802.1AS

Real Time is Newtonian Time, a continuous entity.

Overview

1. Introduction
2. Rationale for and use of fault-tolerant clock synchronization
3. A short history on the development of fault-tolerant clock synchronization
4. Fault-tolerant clock synchronization and how it may be of benefit to IEEE 802.1AS

Basic Questions in Fault-Tolerant Clock Synchronization



Loss of Grand Master clock requires a changeover

- **How long does the changeover take?**
- **Is the changeover fault-tolerant?**
- **Is a malicious failure behavior of the Grand Master clock tolerated?**

Fault-Tolerant Clock Synchronization is not just electing a new Grand Master

Loss of Grand Master clock requires a changeover

- **How long does the changeover take?**
- **Is the changeover fault tolerant?**
- **Is a malicious failure behavior of the Grand Master clock tolerated?**

In fault-tolerant clock synchronization we also need to precisely specify

- **How many components may become faulty?**
- **What is the failure behavior (the failure mode) of a faulty component ?**
- **How many end stations and/or bridges are necessary to tolerate the specified failure mode of the faulty components?**
- **What is the proof that the fault-tolerant clock synchronization algorithm actually works?**

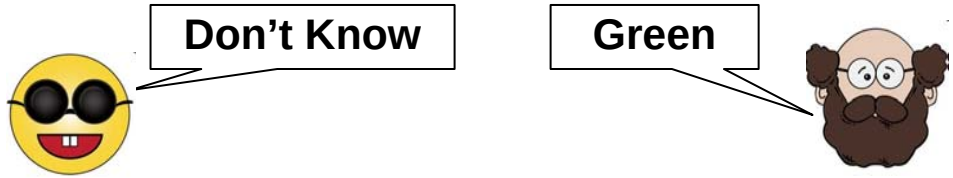
Fault-Tolerance through Redundancy

Situation:

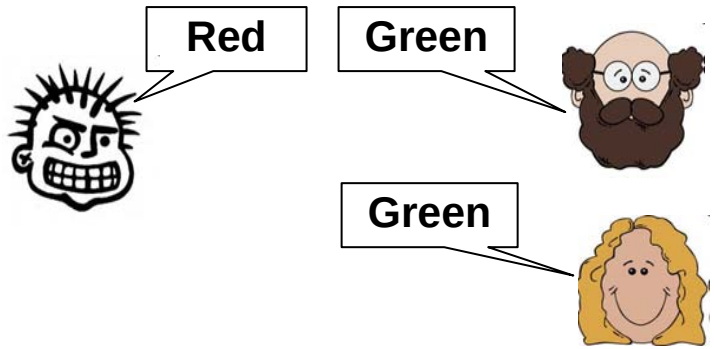
What is the color of the house?

No Failure

Fail-Silence Failure



Fail-Consistent Failure



Static vs. Dynamic Systems

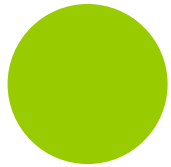
Situation:

What is the color of the house?

Static Situation – one Truth

Situation:

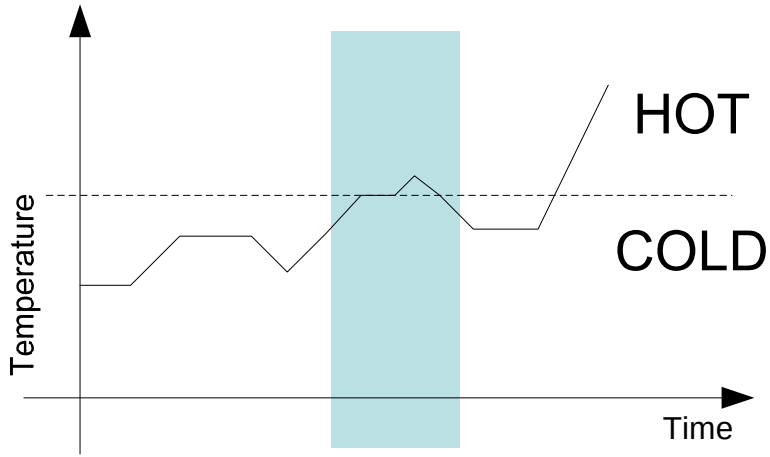
What is the color of the ball ?

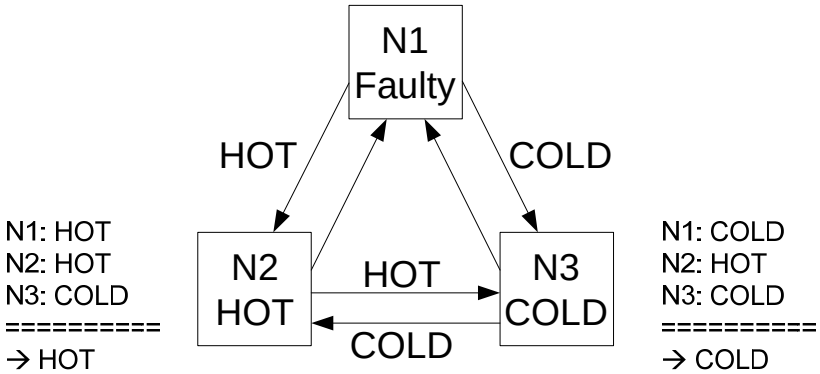


Dynamic Situation – >one Truth

Origins: Byzantine Failures

A distributed system that measures the temperature of a vessel shall raise an alarm when the temperature exceeds a certain threshold.
The system shall tolerate the arbitrary failure of one node.
How many nodes are required?
How many messages are required?

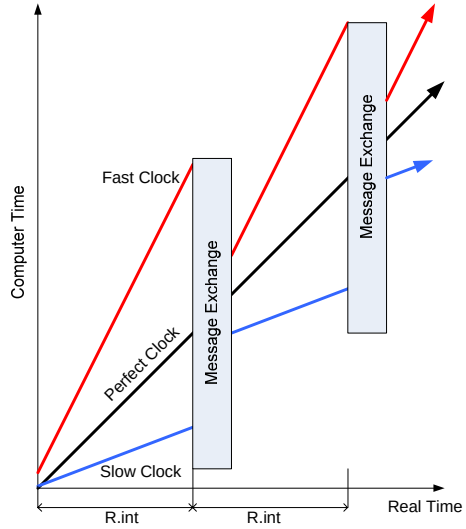


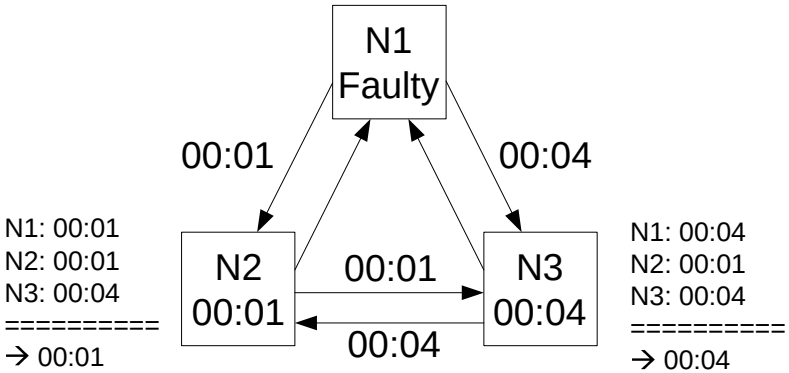


In general, three nodes are insufficient to tolerate the arbitrary failure of a single node.
The two correct nodes are not always able to agree on a value.
A decent body of scientific literature exists that address this problem of dependable systems, in particular dependable communication.

Byzantine Clocks

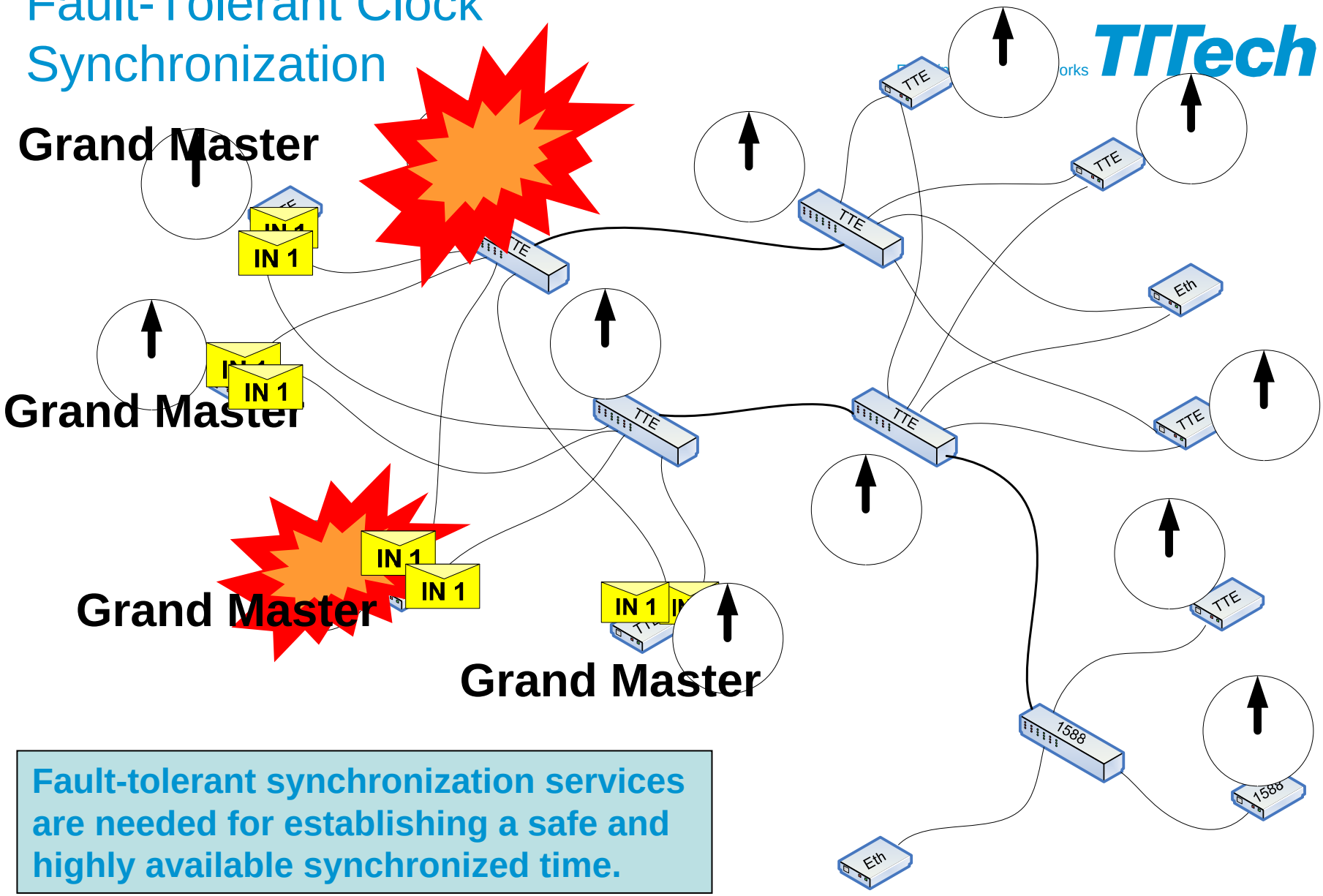
A distributed system in which all nodes are equipped with local clocks, all clocks shall become and remain synchronized.
The system shall tolerate the arbitrary failure of one node.
How many nodes are required?
How many messages are required?





In general, three nodes are insufficient to tolerate the arbitrary failure of a single node.
The two correct nodes are not always able to bring their clocks into close agreement.
A decent body of scientific literature exists that address this problem of fault-tolerant clock synchronization.

Fault-Tolerant Clock Synchronization



Fault-tolerant synchronization services are needed for establishing a safe and highly available synchronized time.

Academic Background

Time, Clocks and the Ordering of Events in a Distributed System,
L. Lamport, 1978

Using Time Instead of Timeout for Fault-Tolerant Distributed Systems,
L. Lamport, 1984

Synchronizing Clocks in the Presence of Faults,
L. Lamport and Michael Melliar-Smith, 1985

Understanding Protocols for Byzantine Clock Synchronization,
Fred B. Schneider, 1987

Event-Triggered versus Time-Triggered Real-Time Systems
H. Kopetz, 1991

Bus Architectures for Safety-Critical Embedded Systems
J. Rushby, 2001

TTA and PALS: Formally Verified Design Patterns for Distributed Cyber-Physical Systems
W. Steiner and J. Rushby, 2011

Examples of Industrial Applications of Fault-Tolerant Clock Synchronization

Aerospace Domain

- Boeing 787, C-Series, F-16 (TTP)
- Airbus A380 (TTP)

Space Domain

- NASA Orion (TTEthernet)

Automotive Domain

- Audi various models (FlexRay)
- BMW various models (FlexRay)
- Volkswagen various models (FlexRay)

Industrial Domain

- Wind turbine manufacturer (TTEthernet)

Overview

1. Introduction

2. Rationale for and use of fault-tolerant clock synchronization

3. A short history on the development of fault-tolerant clock synchronization

4. Fault-tolerant clock synchronization and how it may be of benefit to IEEE 802.1AS

Why is Fault tolerant Clock Sync relevant for 802.1AS in Safety-Relevant and Safety-Critical Systems?

For some safety-relevant/safety-critical systems 802.1AS is the solution.

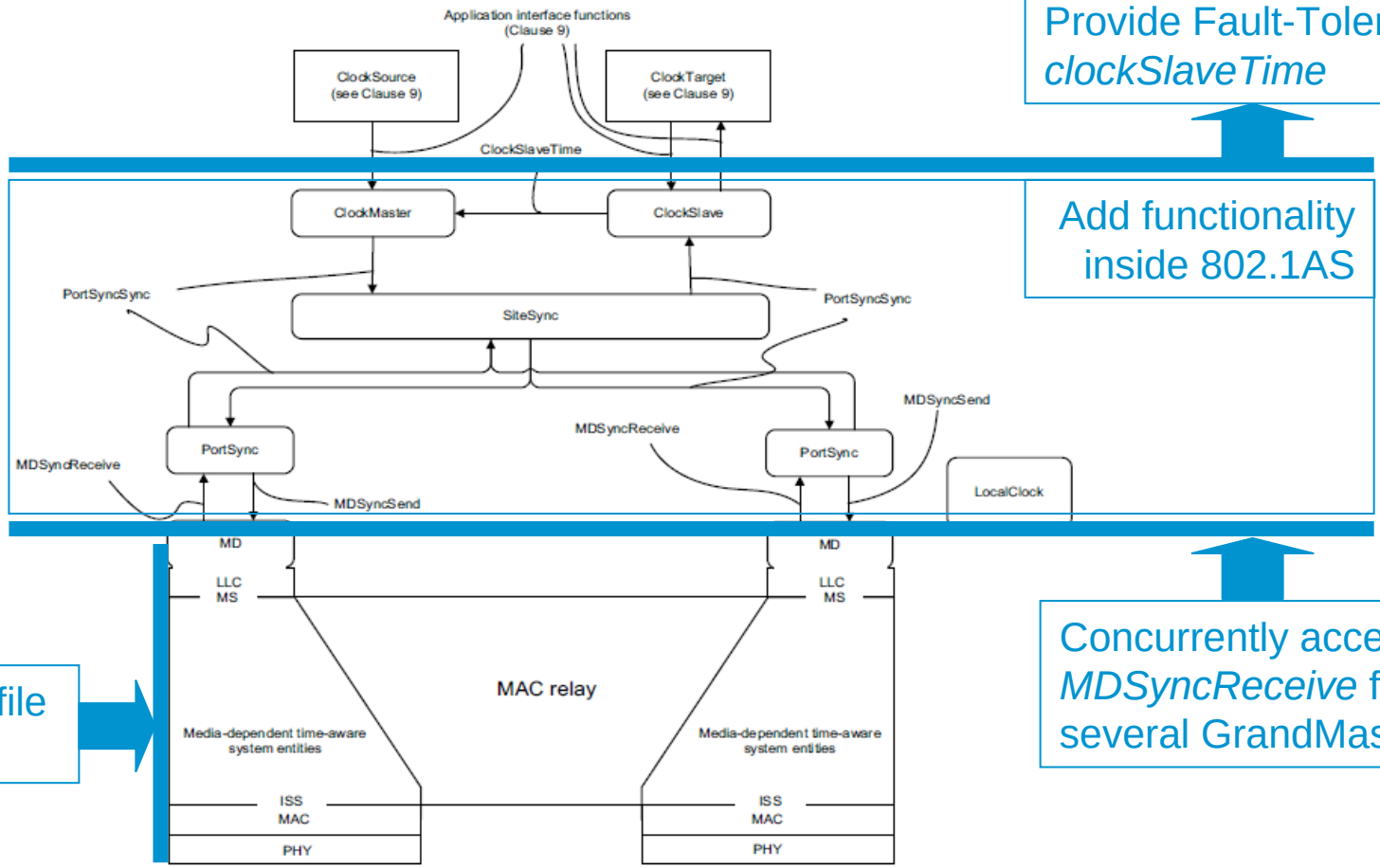
For full coverage in these application domains, additional fail-operational capabilities are required.

- Fail-operational systems like autonomous driving in automotive or flight management in aerospace require continuous operation of the network even in presence of failures.
- High availability
- ***It certainly <u>minimizes</u> the grandmaster changeover time.***

Fault-tolerant clock synchronization is understood and applied in safety-critical/safety-relevant applications.

Some Fail-Operational Options for 802.1AS (i)

Ensuring Reliable Networks TTTech



Provide Fault-Tolerant *clockSlaveTime*

Add functionality inside 802.1AS

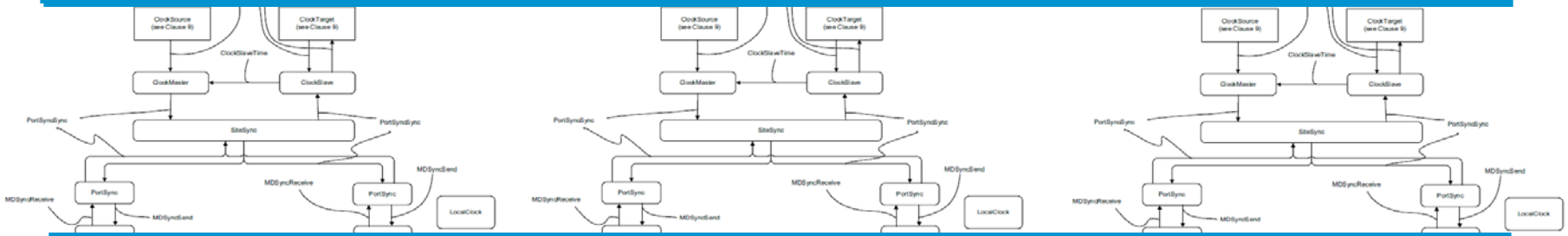
Concurrently accept *MDSyncReceive* from several GrandMasters

Specific Profile Required?

Some Fail-Operational Options for 802.1AS (ii)

Provide Fault-Tolerant *clockSlaveTime*

Fail-Operational Extensions



New MD Profile

Ensuring Reliable Networks

www.tttech.com

Wilfried Steiner
Senior Research Engineer
wilfried.steiner@tttech.com