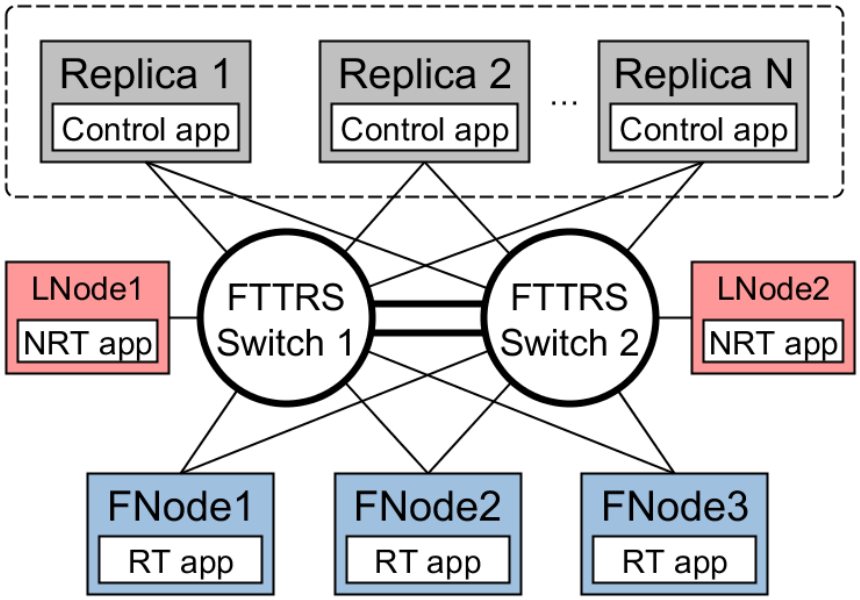
FT4FTT prototyping (aspects to improve)

Kick-off meeting of the DFT4FTT project

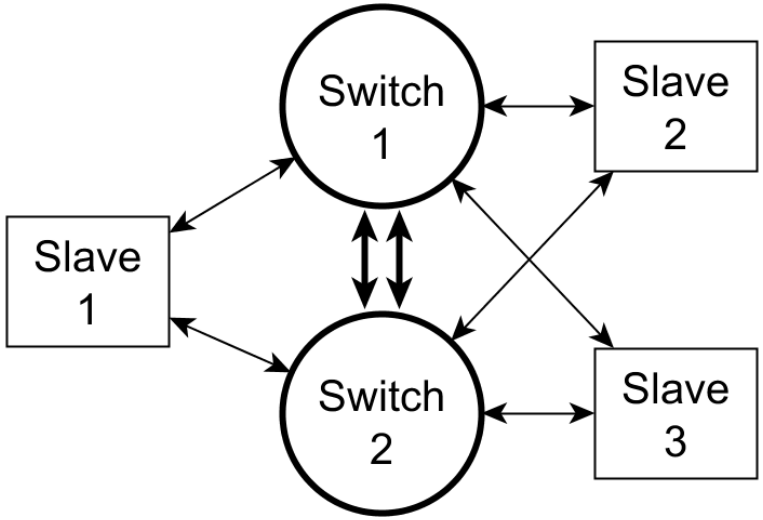


Alberto Ballesteros

Introduction

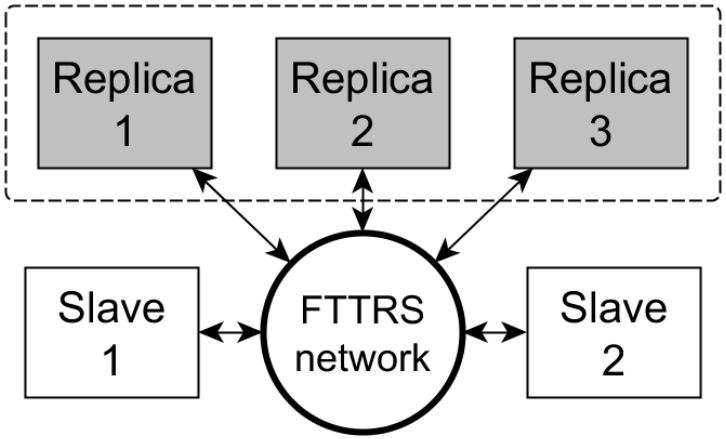
Fault tolerance to faults affecting the **network**

- Flexible Time-Triggered Replicated Star (FTTRS)



Fault tolerance to faults affecting the **nodes**

- Node Replication scheme



Outline

- Faults affecting the network
- Faults affecting the nodes
- Prototyping
- Issues

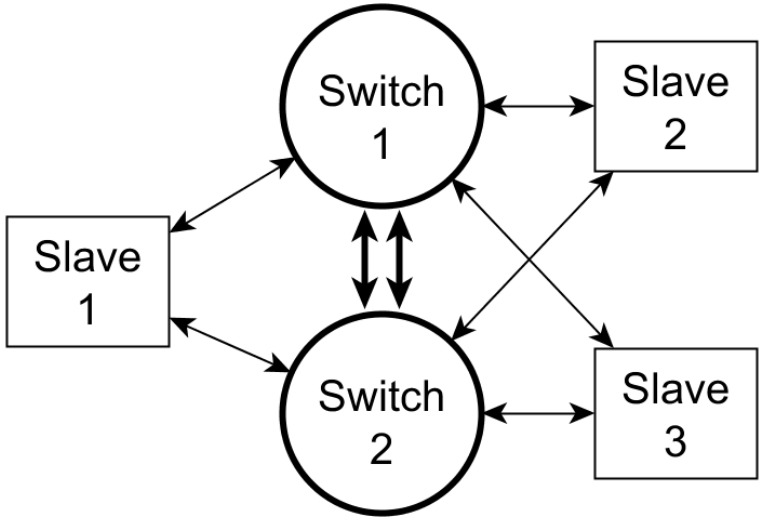
Outline

- **Faults affecting the network**
- Faults affecting the nodes
- Prototyping
- Issues

Introduction

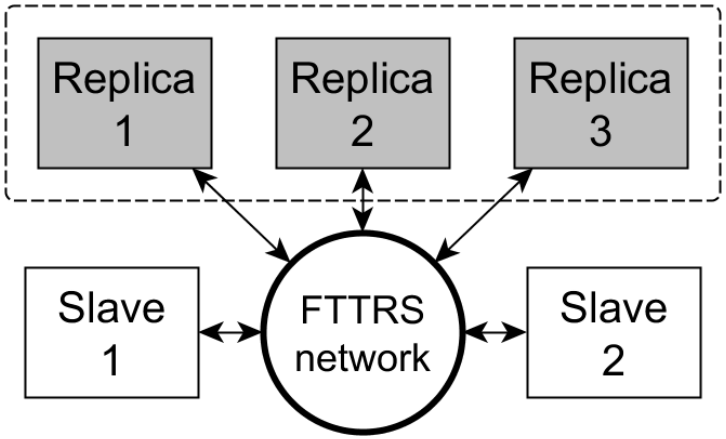
Fault tolerance to faults affecting the **network**

- Flexible Time-Triggered Replicated Star (FTTRS)



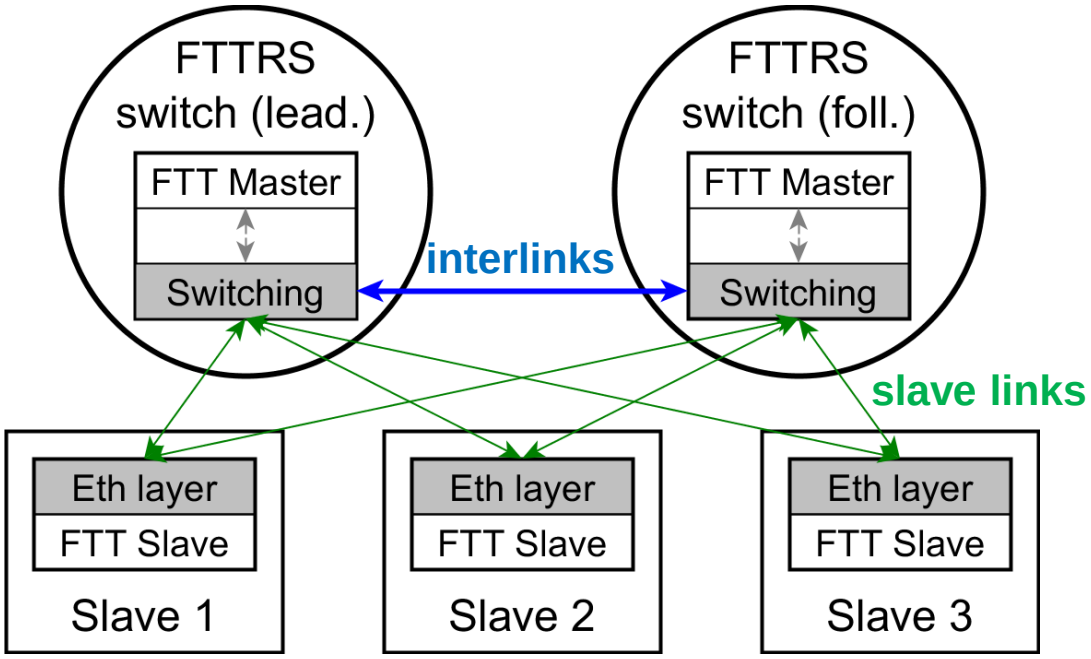
Fault tolerance to faults affecting the **nodes**

- Node Replication scheme



Faults affecting the network

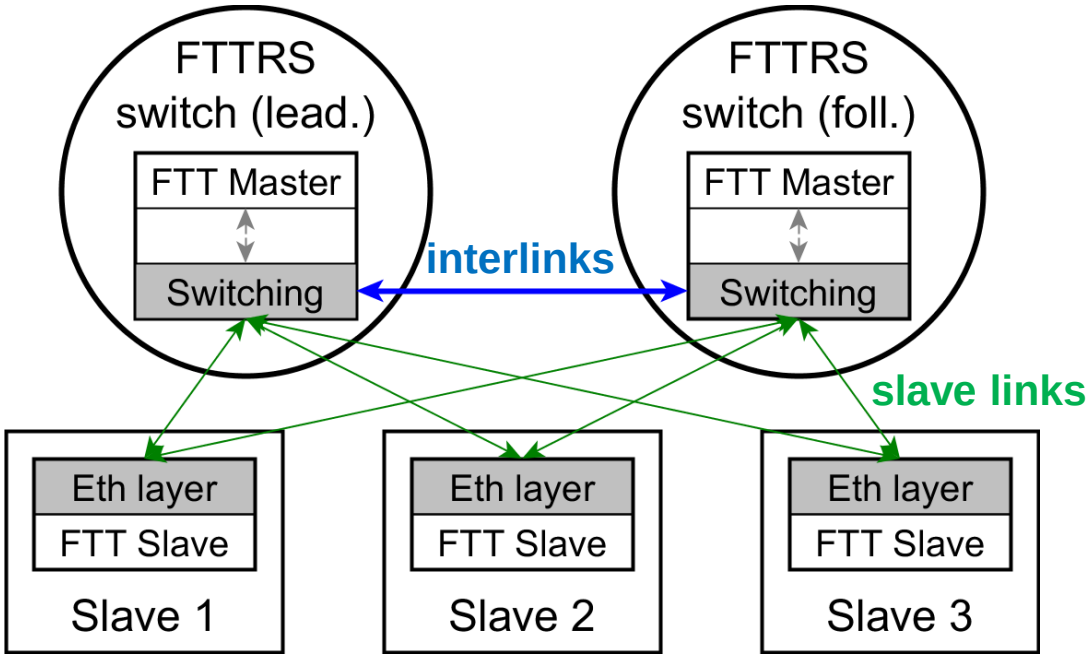
Flexible Time-Triggered Replicated Star for Ethernet (FTTRS)



- Active replicated switches and masters
- Replicated slave links

Faults affecting the network

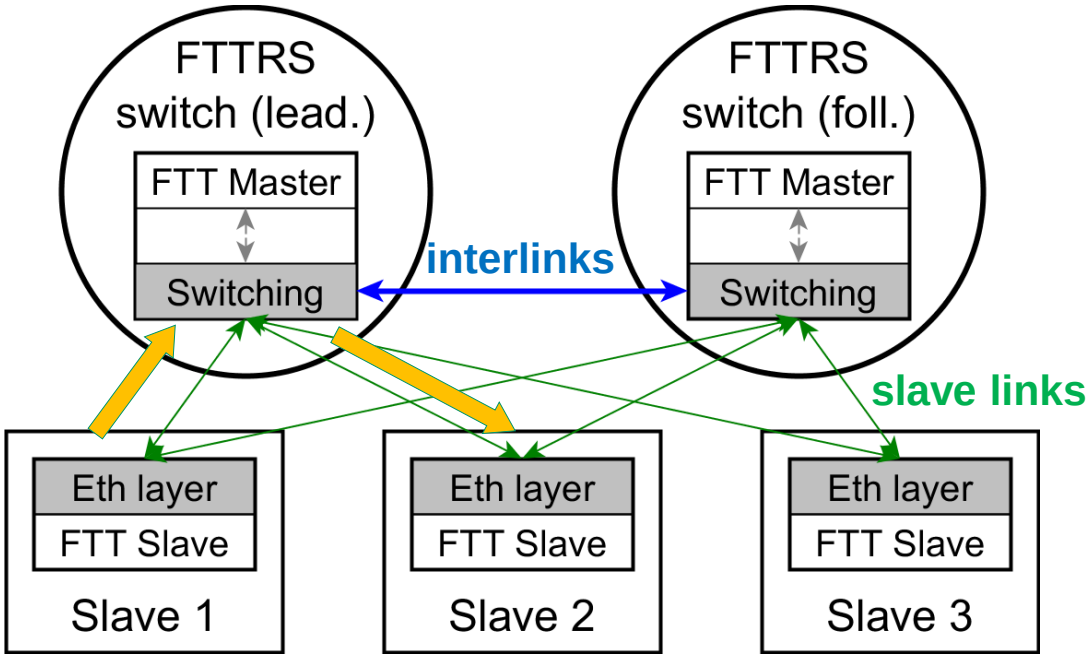
Flexible Time-Triggered Replicated Star for Ethernet (FTTRS)



- Active replicated switches and masters
- Replicated slave links
- Manage replica radiation

Faults affecting the network

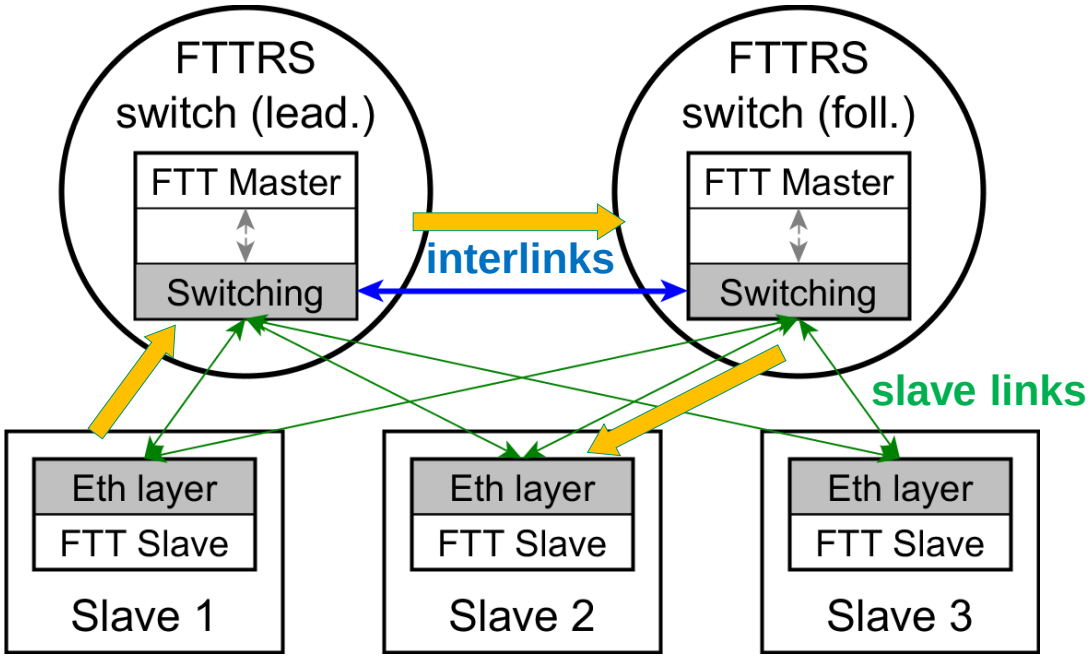
Flexible Time-Triggered Replicated Star for Ethernet (FTTRS)



- Active replicated switches and masters
- Replicated slave links
- Manage replica radiation

Faults affecting the network

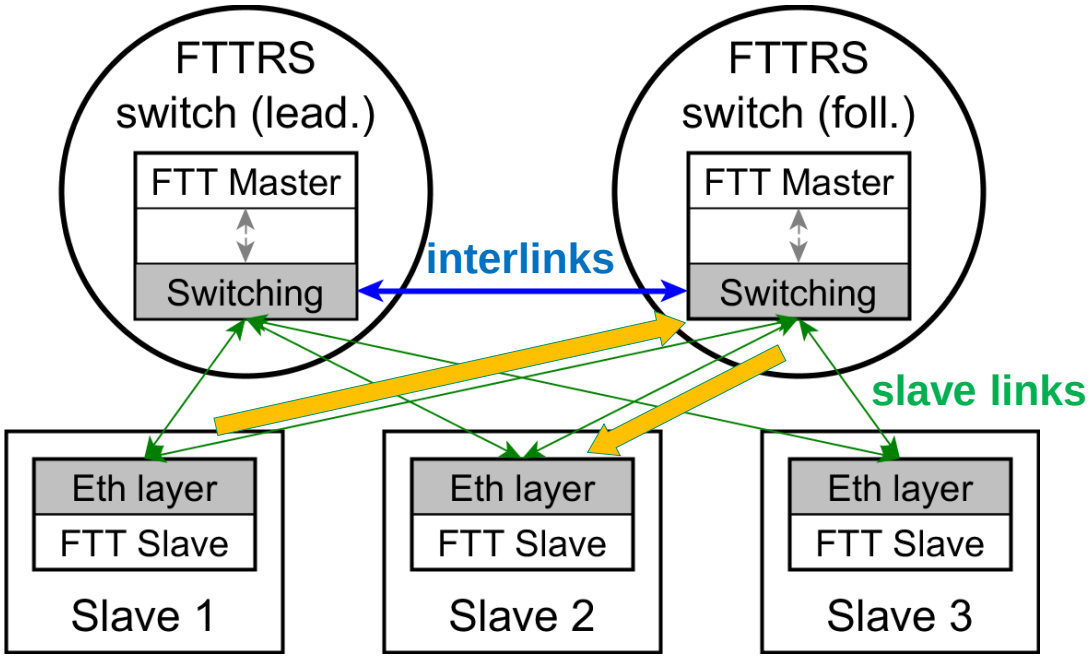
Flexible Time-Triggered Replicated Star for Ethernet (FTTRS)



- Active replicated switches and masters
- Replicated slave links
- Manage replica radiation

Faults affecting the network

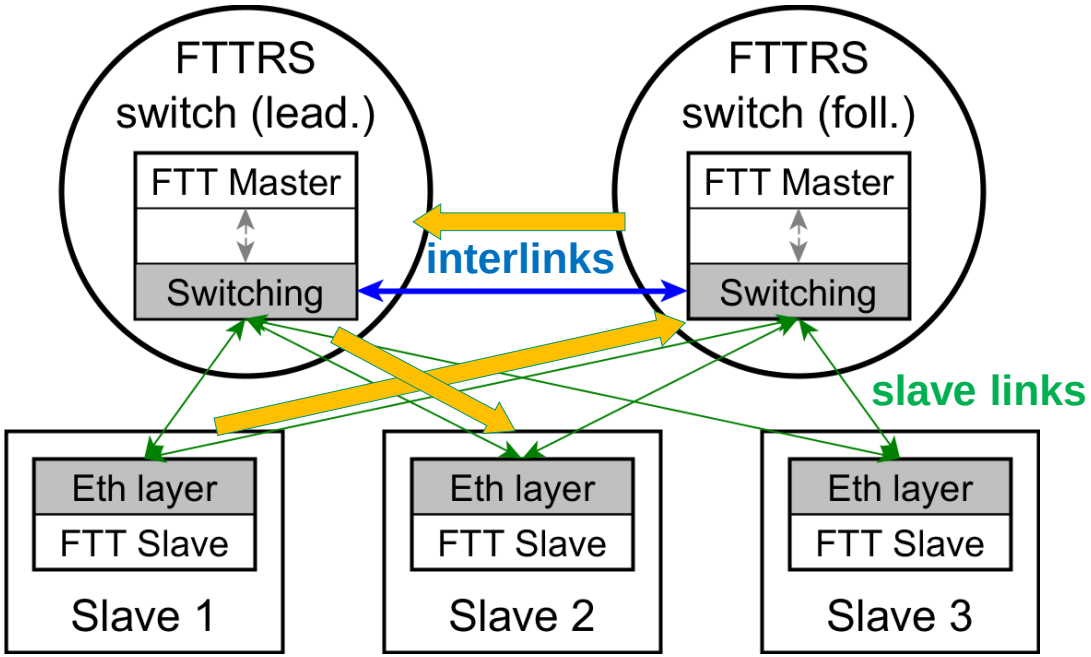
Flexible Time-Triggered Replicated Star for Ethernet (FTTRS)



- Active replicated switches and masters
- Replicated slave links
- Manage replica radiation

Faults affecting the network

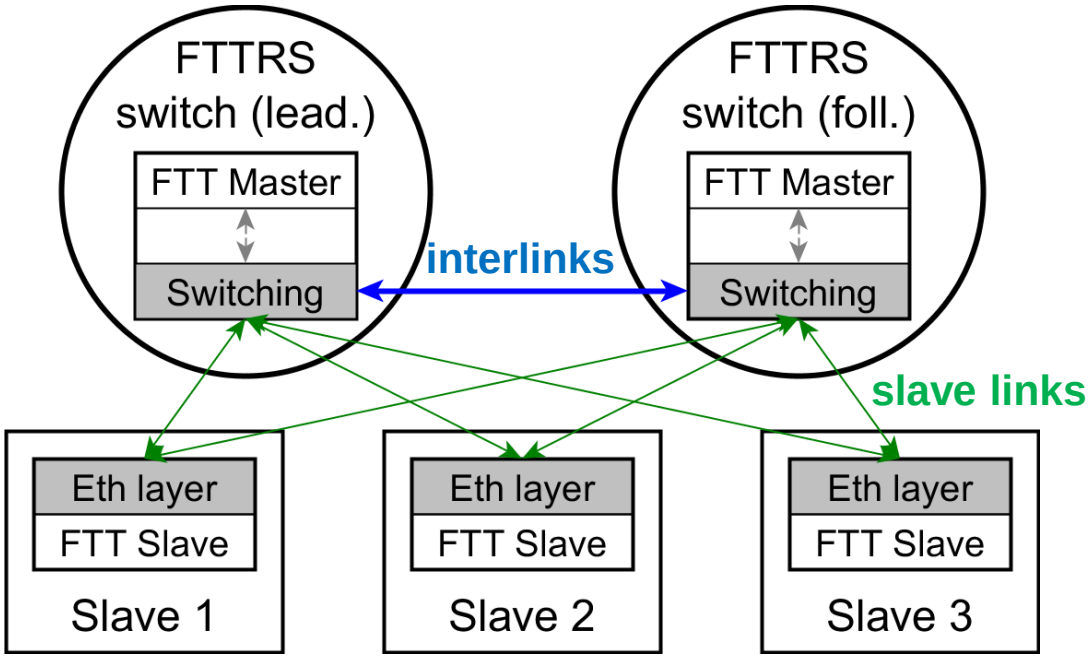
Flexible Time-Triggered Replicated Star for Ethernet (FTTRS)



- Active replicated switches and masters
- Replicated slave links
- Manage replica radiation

Faults affecting the network

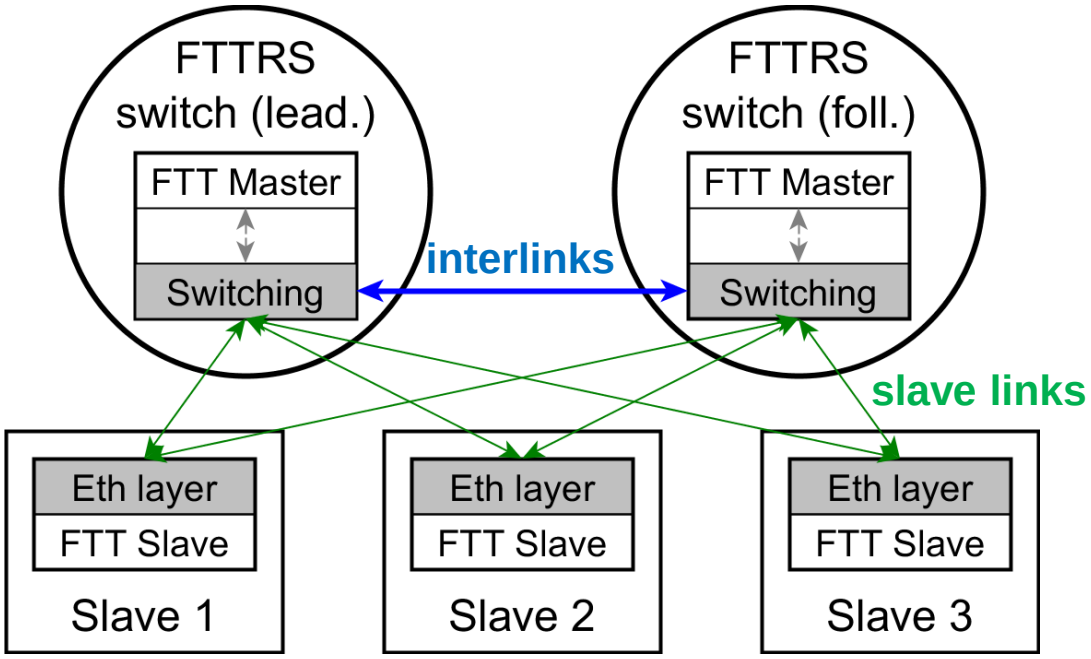
Flexible Time-Triggered Replicated Star for Ethernet (FTTRS)



- Active replicated switches and masters
- Replicated slave links
- Manage replica radiation

Faults affecting the network

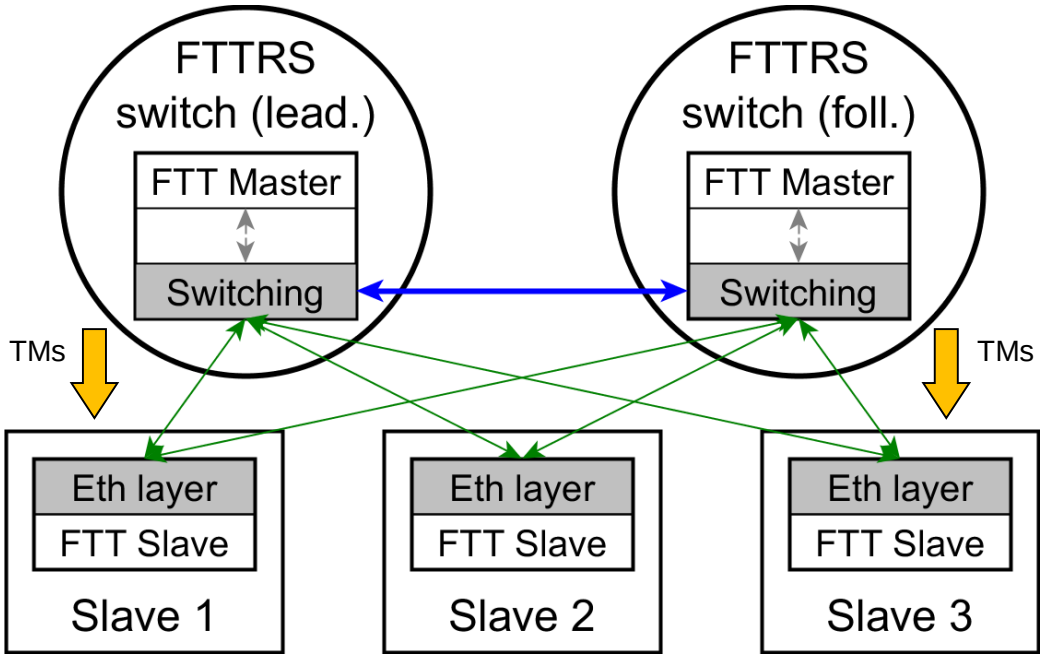
Flexible Time-Triggered Replicated Star for Ethernet (FTTRS)



- Active replicated switches and masters
- Replicated slave links
- Manage replica radiation
- Enforce replica determ.

Faults affecting the network

Flexible Time-Triggered Replicated Star for Ethernet (FTTRS)

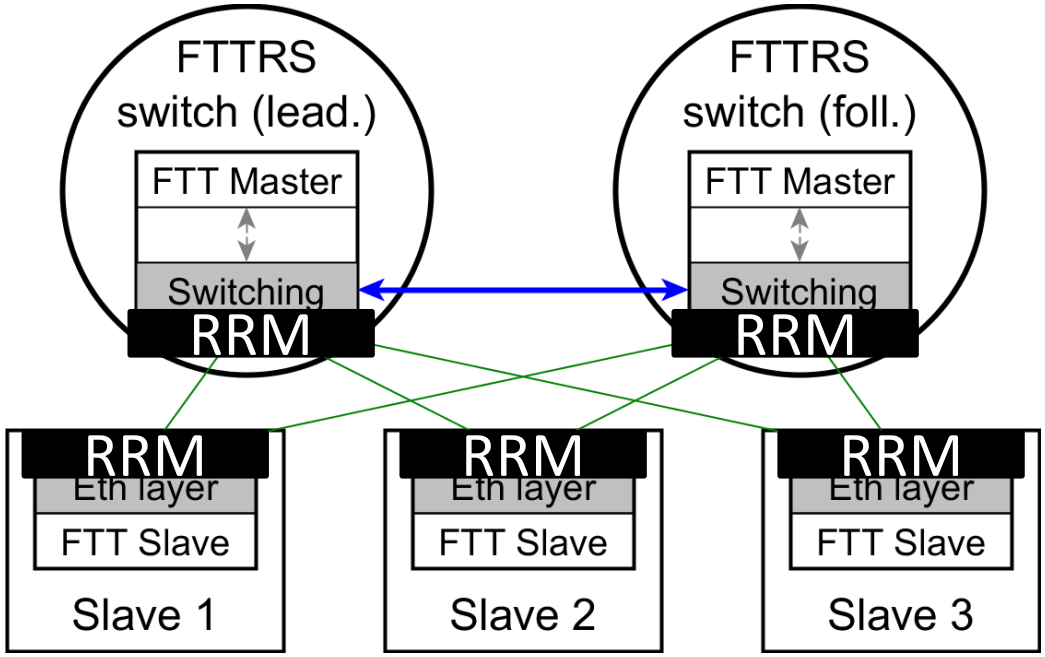


- Active replicated switches and masters
- Replicated slave links

- Manage replica radiation
- Enforce replica determ.

Faults affecting the network

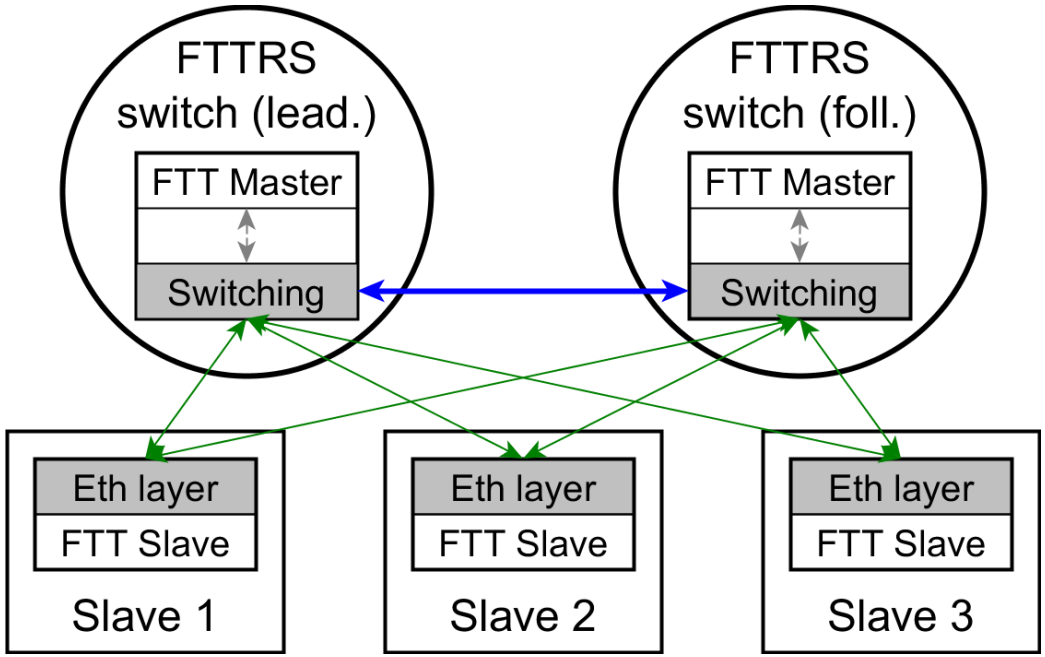
Replica radiation managers detect duplicated messages



Faults affecting the network

Replica determinism in the **time domain**

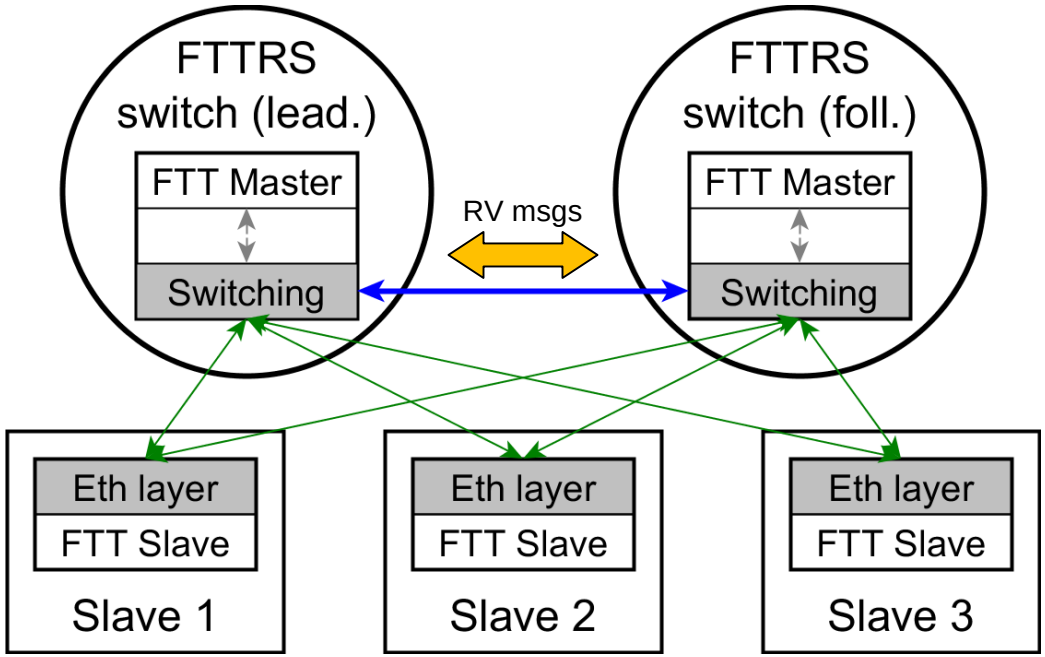
- For a given EC **TMs** must be transmitted **at the same** time by both masters
- Synchronization mechanism



Faults affecting the network

Replica determinism in the **time domain**

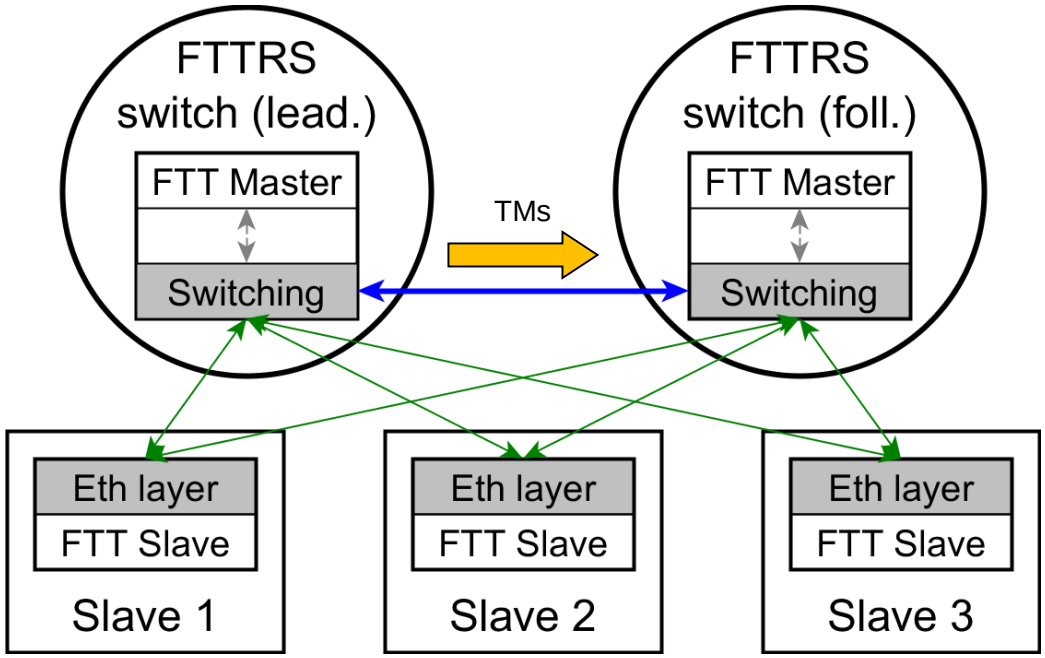
- Level 1 – Initialization: Both masters start their execution together



Faults affecting the network

Replica determinism in the **time domain**

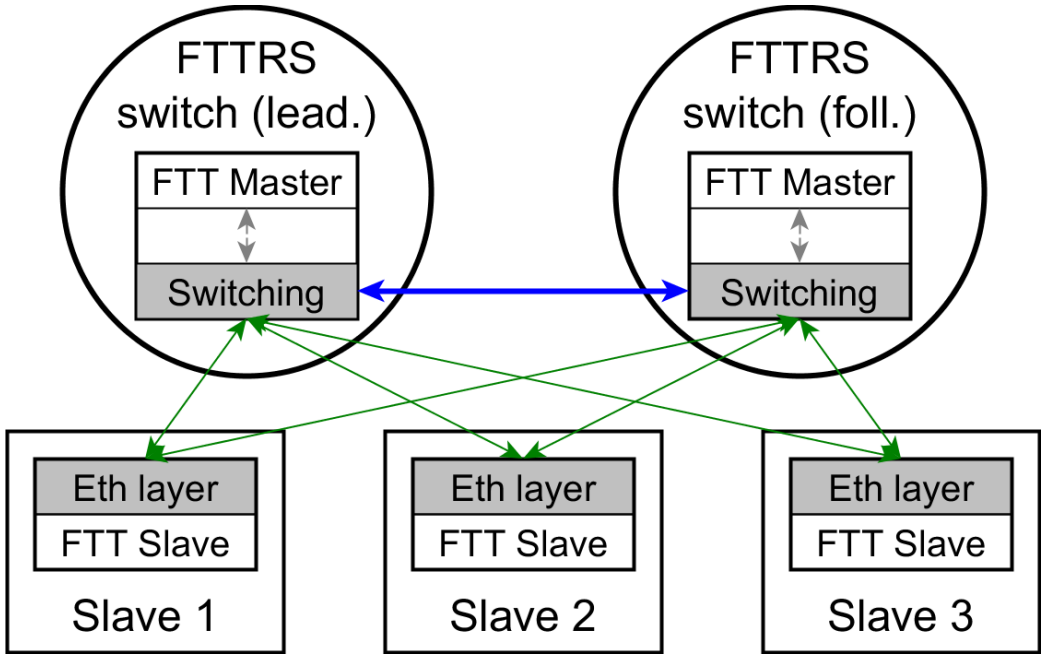
- Level 1 – Initialization: Both masters start their execution together
- Level 2 – Regular operation: The follower periodically syncs with the leader



Faults affecting the network

Replica determinism in the **value domain**

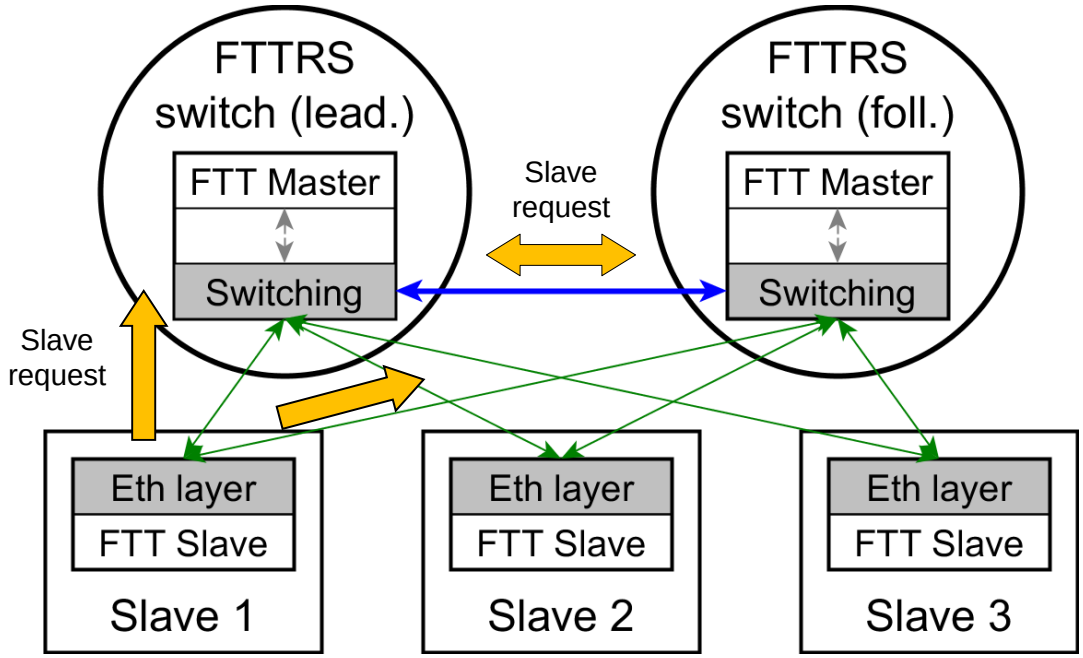
- For a given EC **TMs** must contain the **same EC-schedule**
- System Requirement Tables (SRTs) must be **identical**



Faults affecting the network

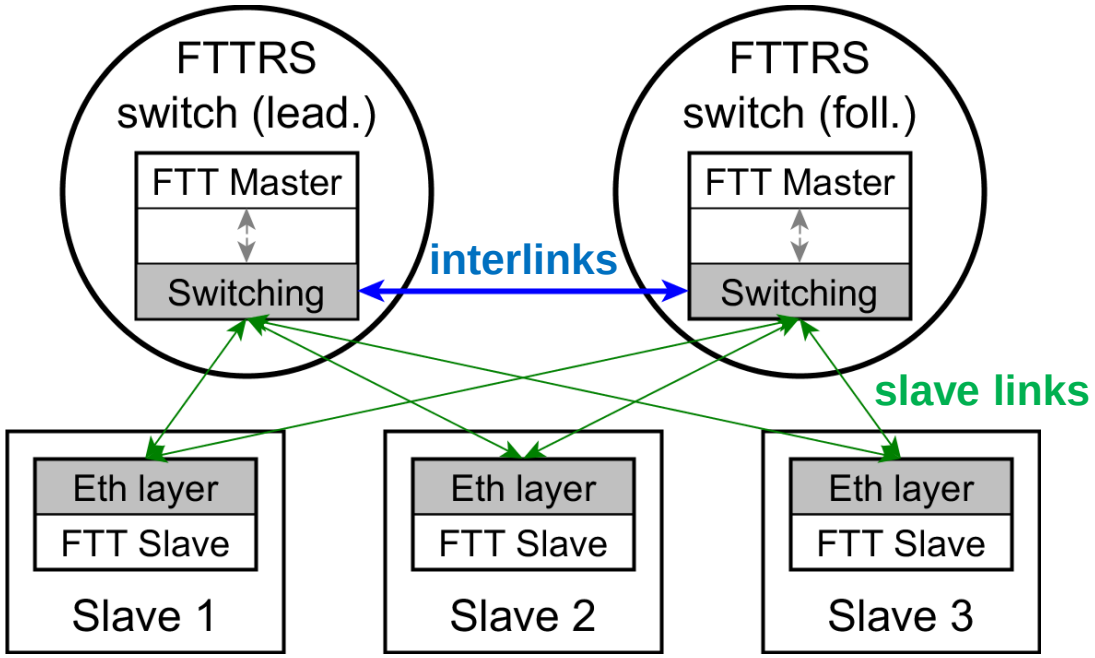
Replica determinism in the **value domain**

- For a given EC **TMs** must contain the **same EC-schedule**
- System Requirement Tables (SRTs) must be **identical**



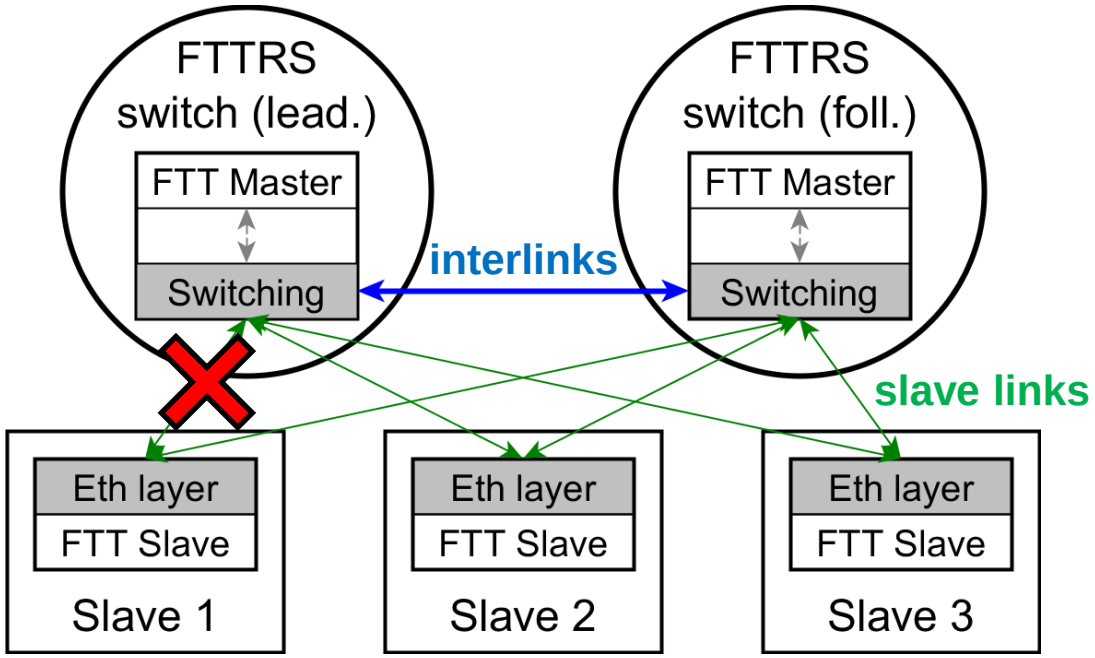
Faults affecting the network

Now we have a communication subsystem that can tolerate **permanent faults**



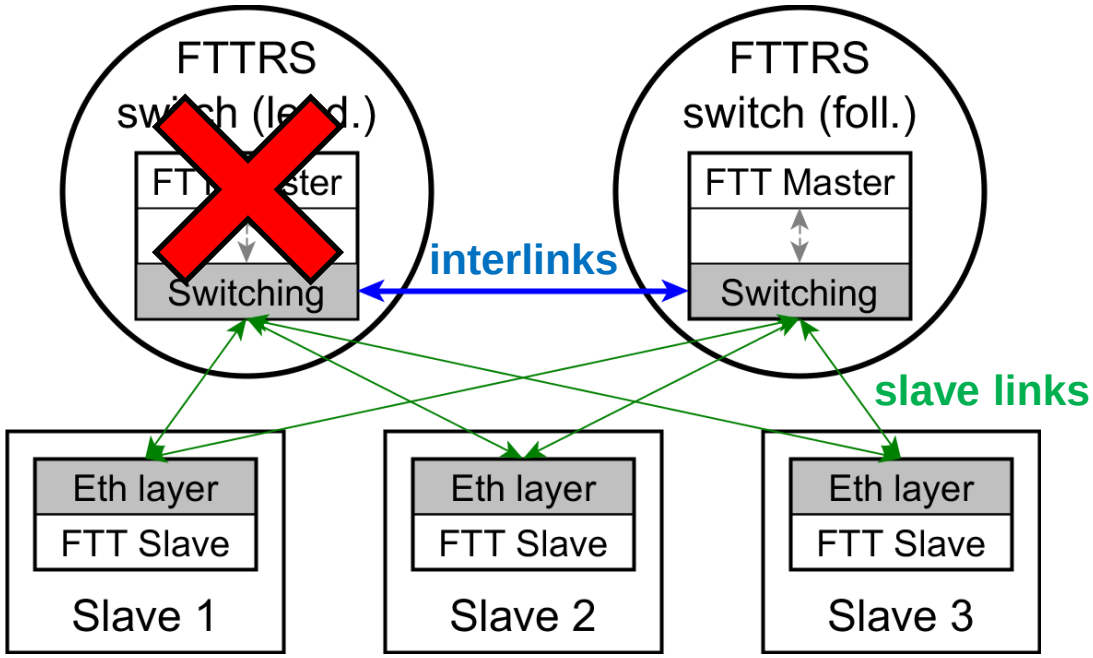
Faults affecting the network

Now we have a communication subsystem that can tolerate **permanent faults**



Faults affecting the network

Now we have a communication subsystem that can tolerate **permanent faults**



Faults affecting the network

Temporary errors

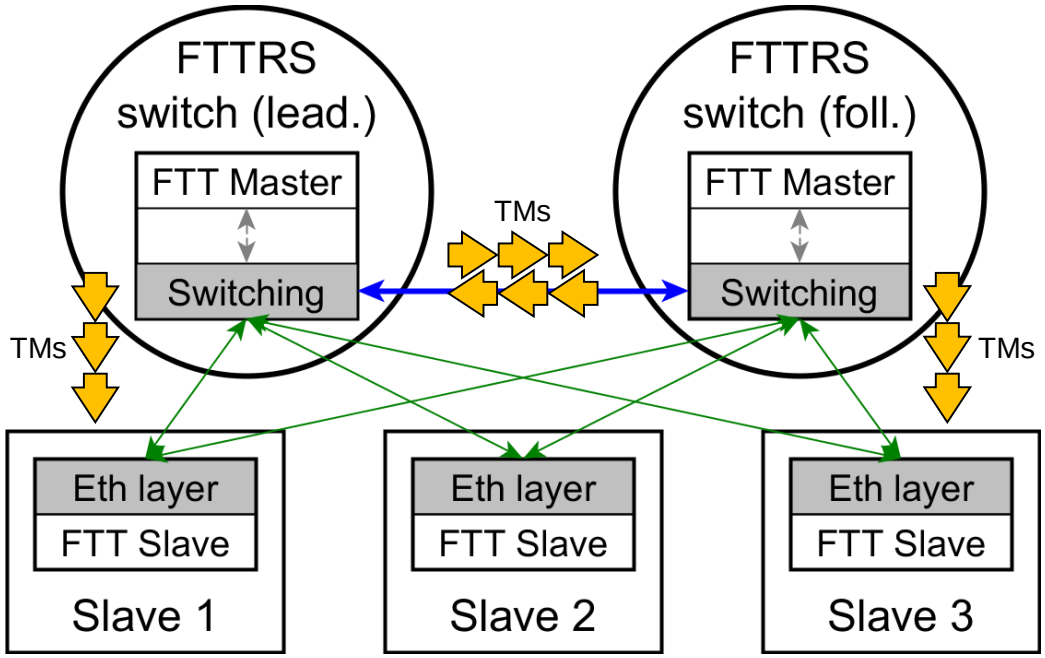
- Faults provoking **message omissions**
- TMs, application messages and control messages

Use **proactive transmission**

Faults affecting the network

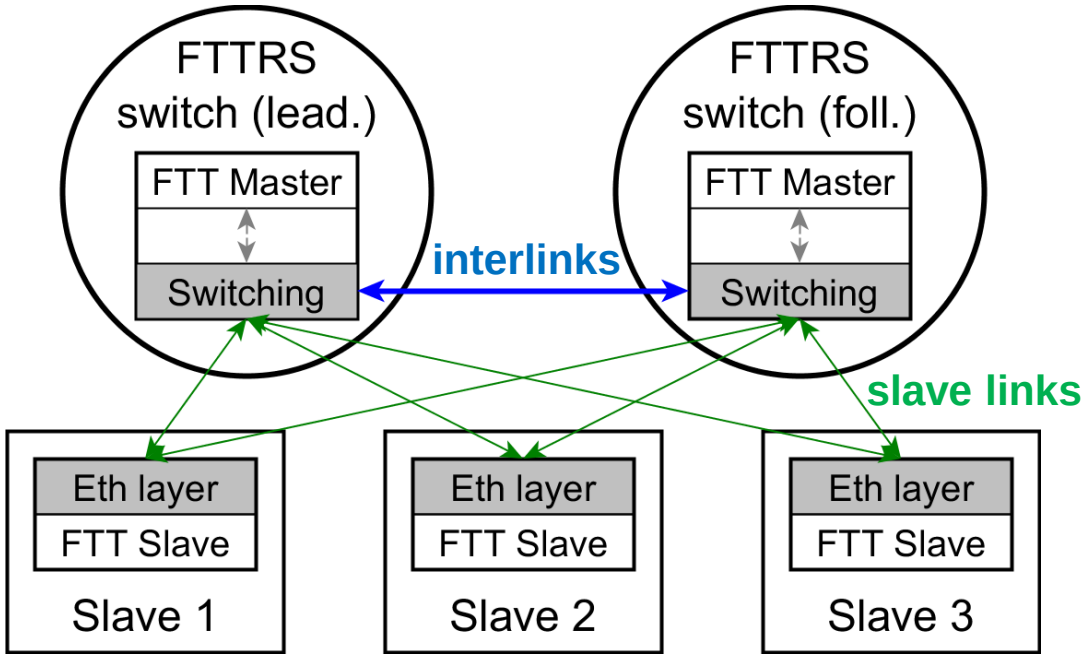
Transmission of TMs

- Masters transmit the TM *k* times
- Take into account for the **synchronization** of the **slaves**
- Take into account for the **synchronization** of the **master**



Faults affecting the network

Now we have a communication subsystem
that can tolerate both **permanent** and **temporary faults**



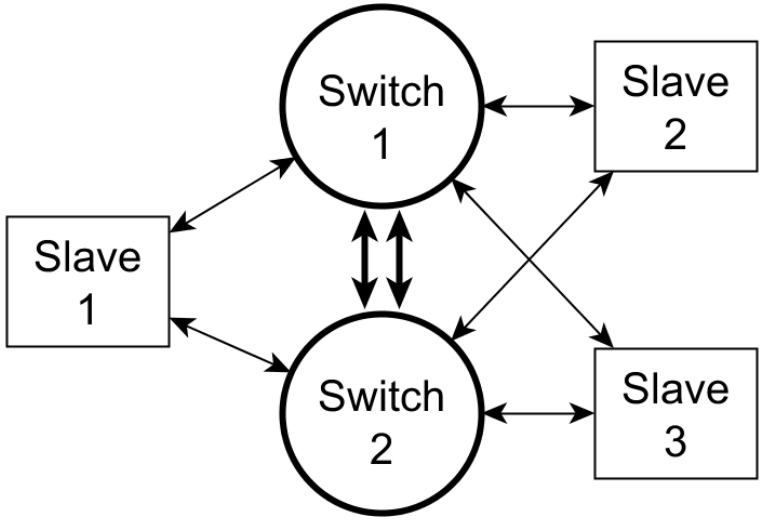
Outline

- Faults affecting the network
- **Faults affecting the nodes**
- Prototyping
- Issues

Faults affecting the nodes

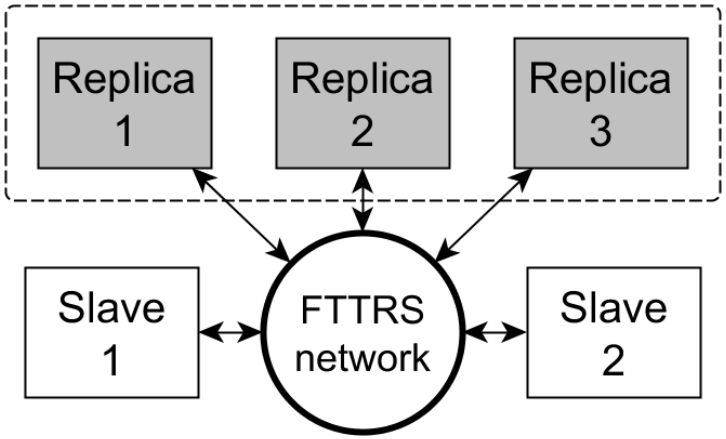
Fault tolerance to faults affecting the **network**

- Flexible Time-Triggered Replicated Star (FTTRS)



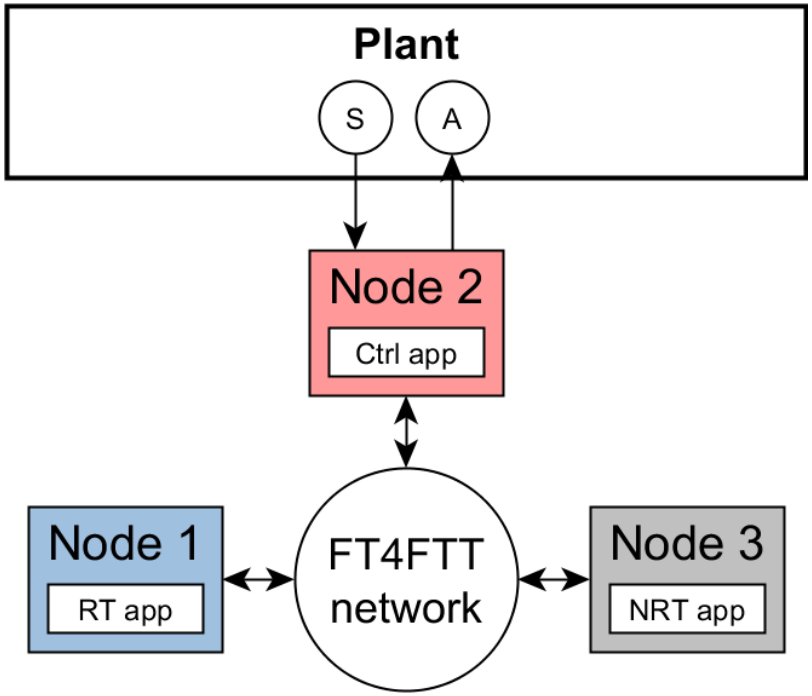
Fault tolerance to faults affecting the **nodes**

- Node Replication scheme



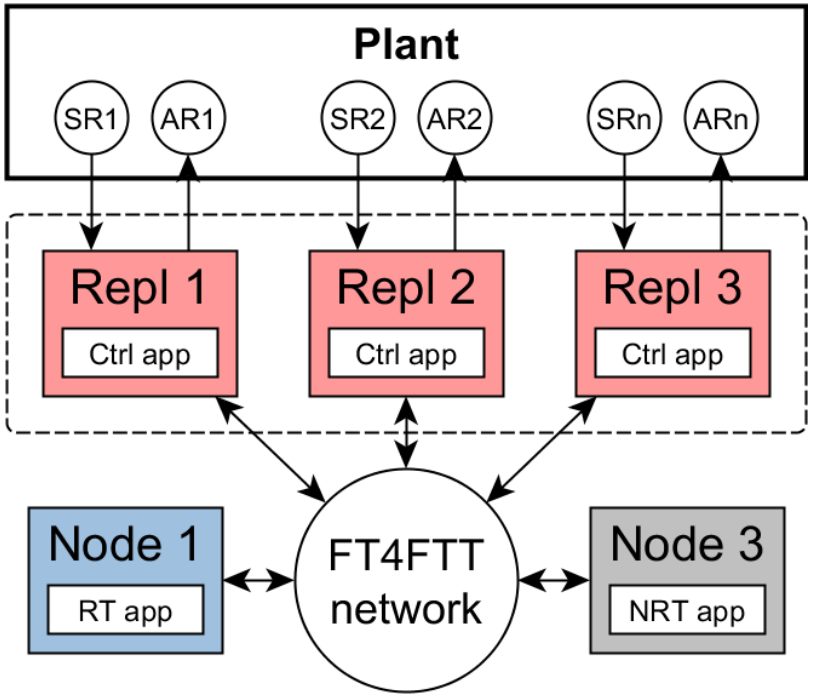
Faults affecting the nodes

Active replication – Physical replication



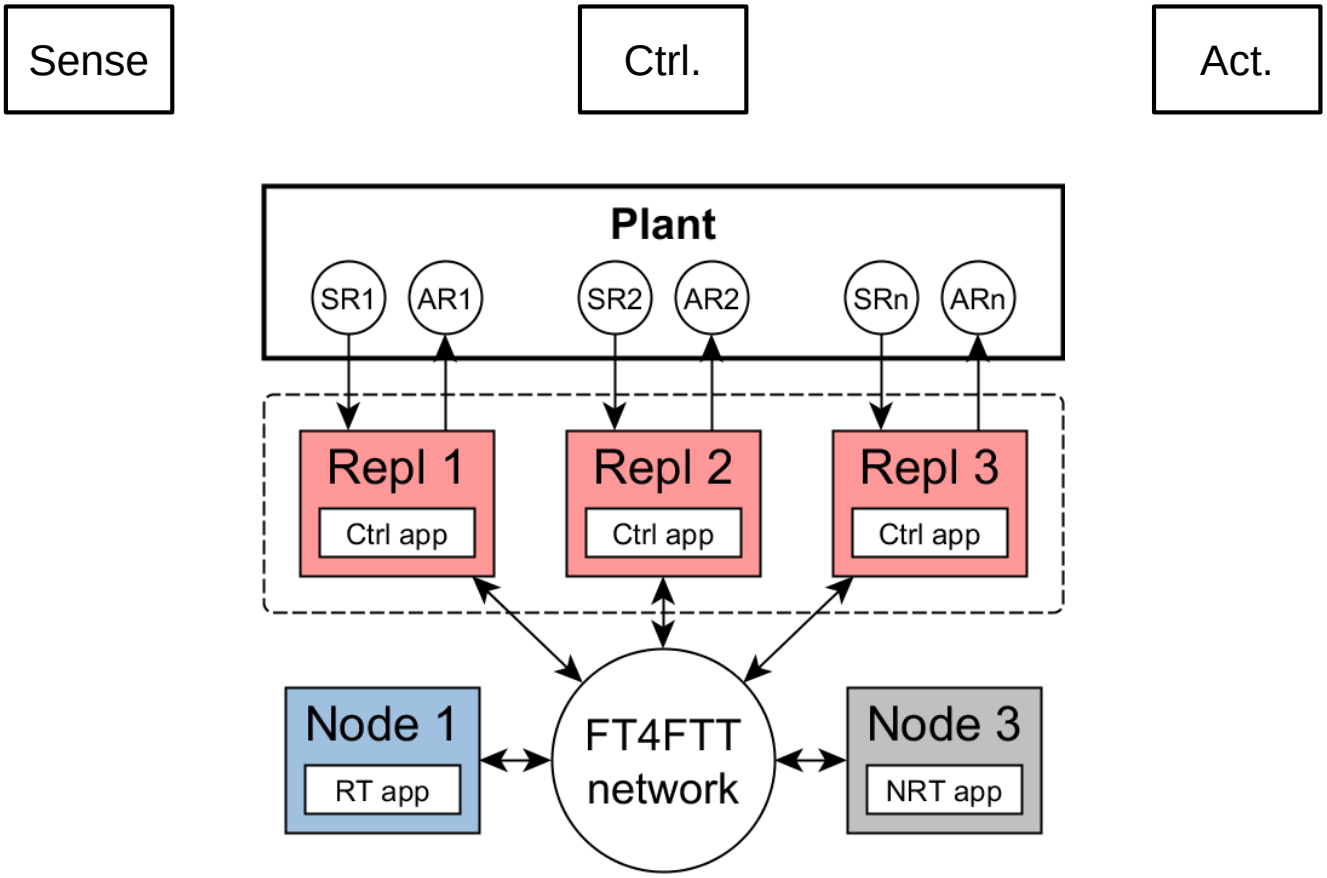
Faults affecting the nodes

Active replication – Physical replication



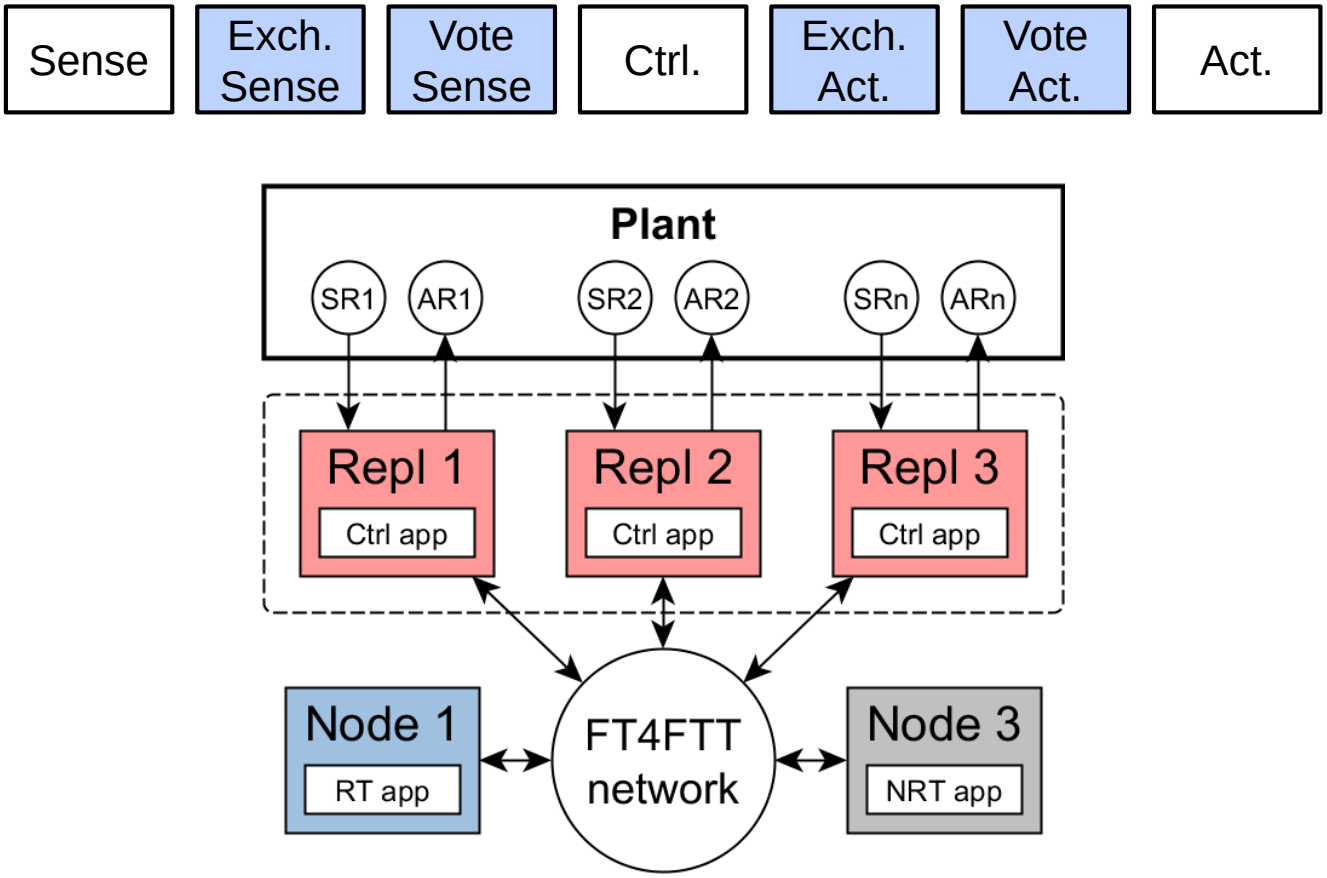
Faults affecting the nodes

Active replication – Majority voting



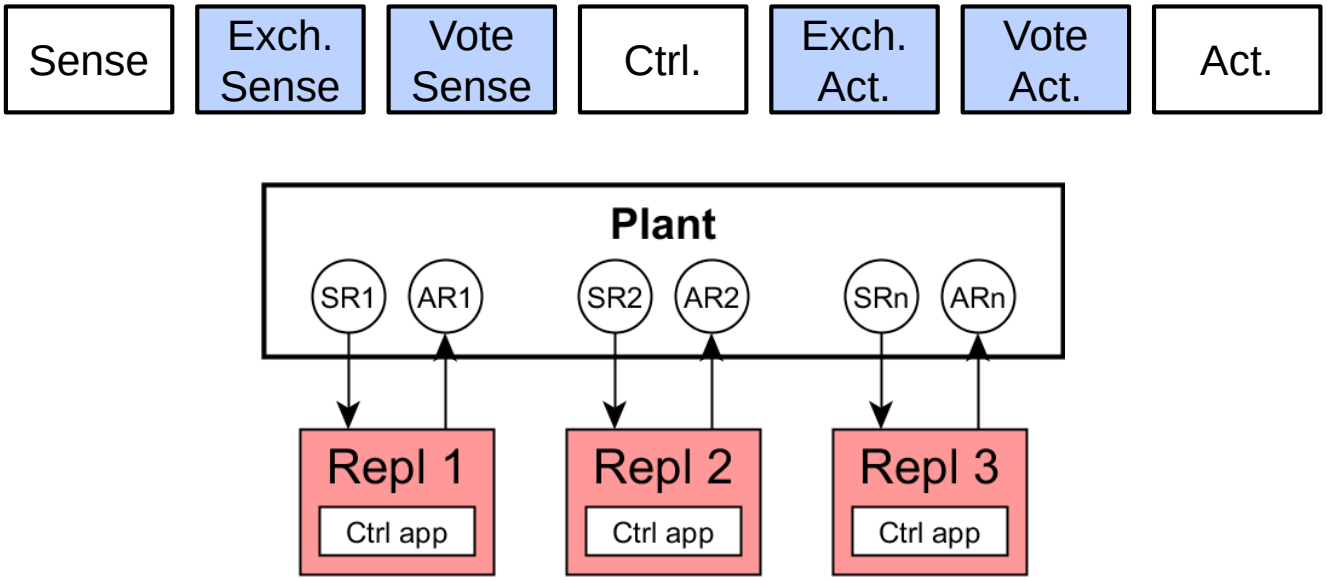
Faults affecting the nodes

Active replication – Majority voting



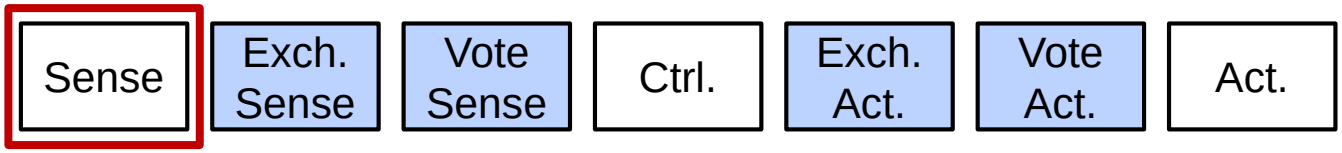
Faults affecting the nodes

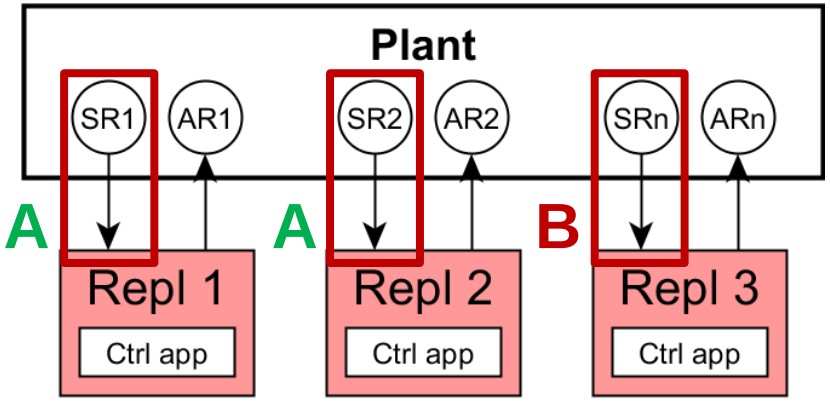
Active replication – Majority voting



Faults affecting the nodes

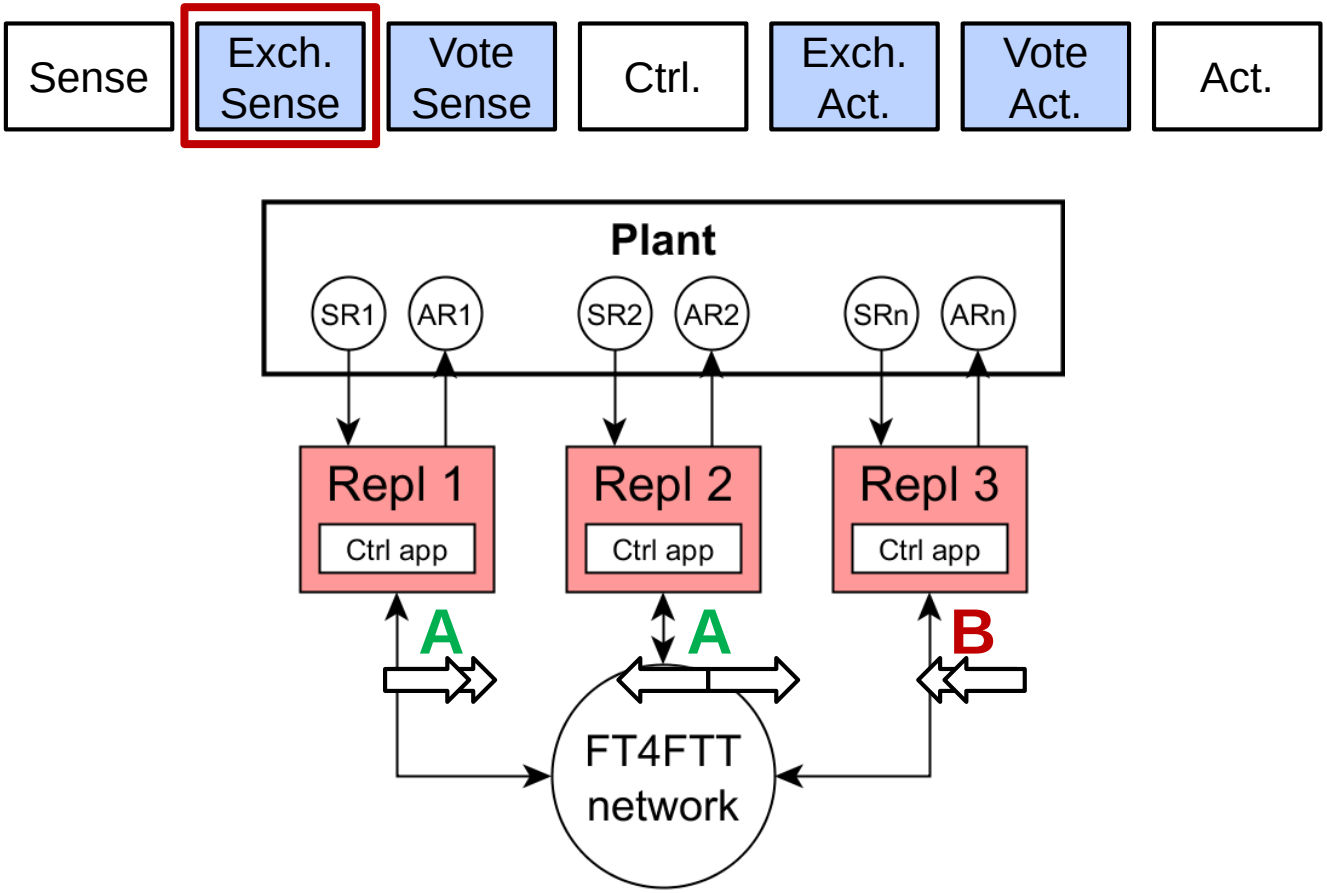
Active replication – Majority voting





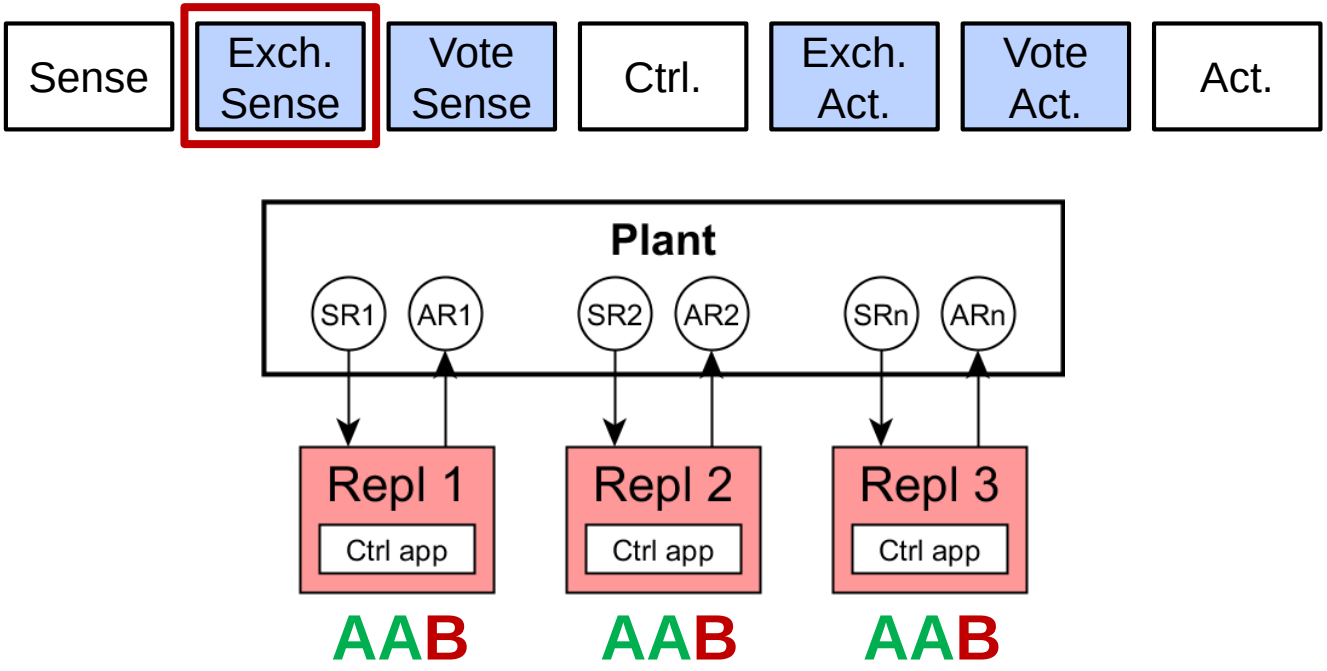
Faults affecting the nodes

Active replication – Majority voting



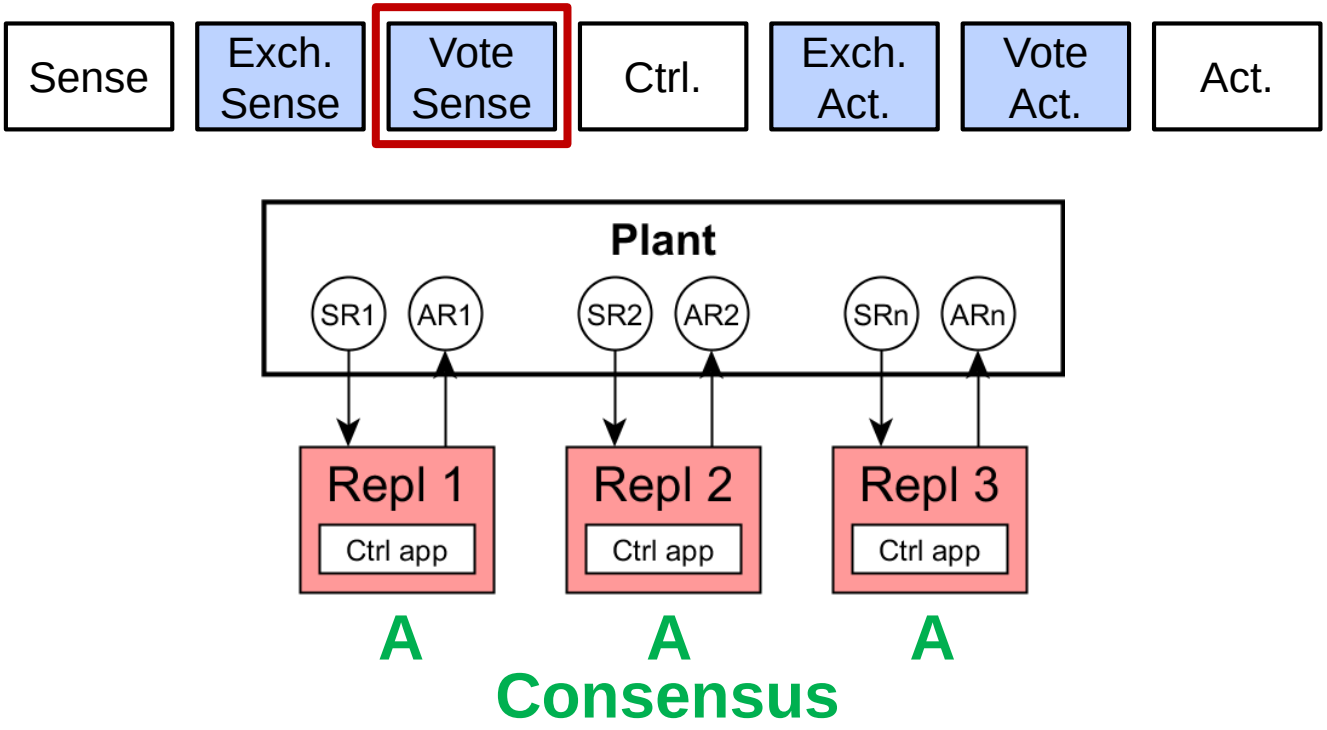
Faults affecting the nodes

Active replication – Majority voting



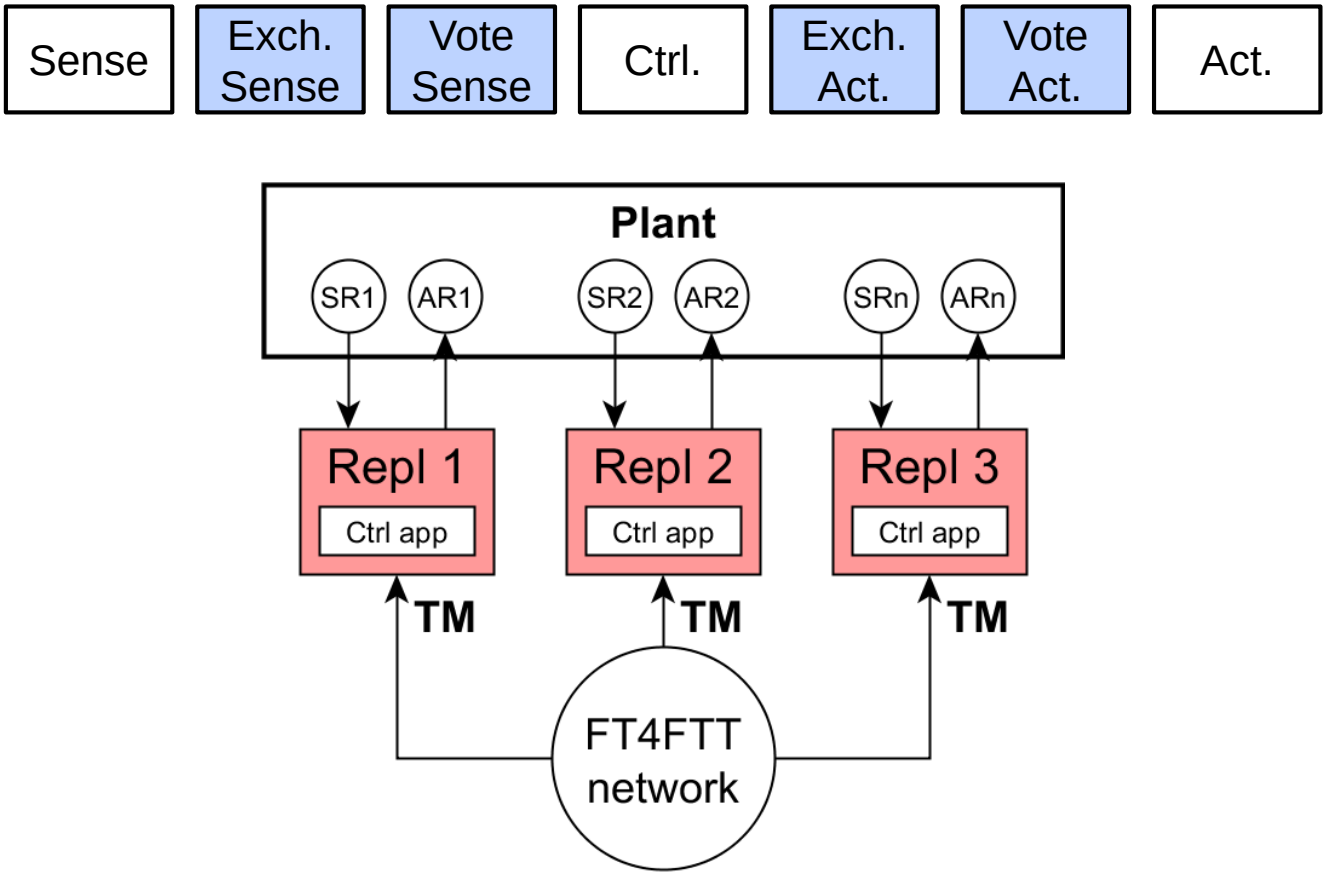
Faults affecting the nodes

Active replication – Majority voting



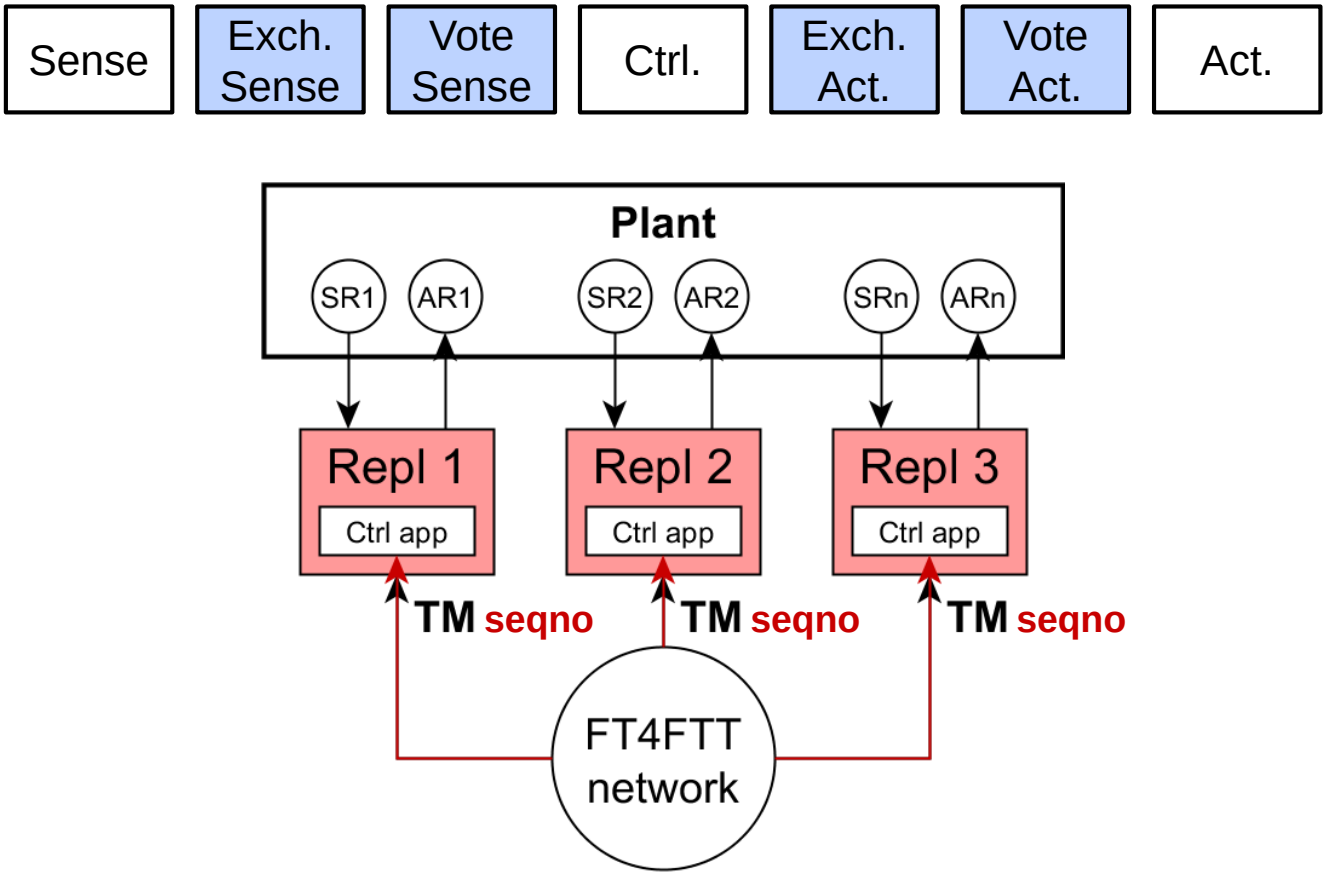
Faults affecting the nodes

Coordinated triggering of phases



Faults affecting the nodes

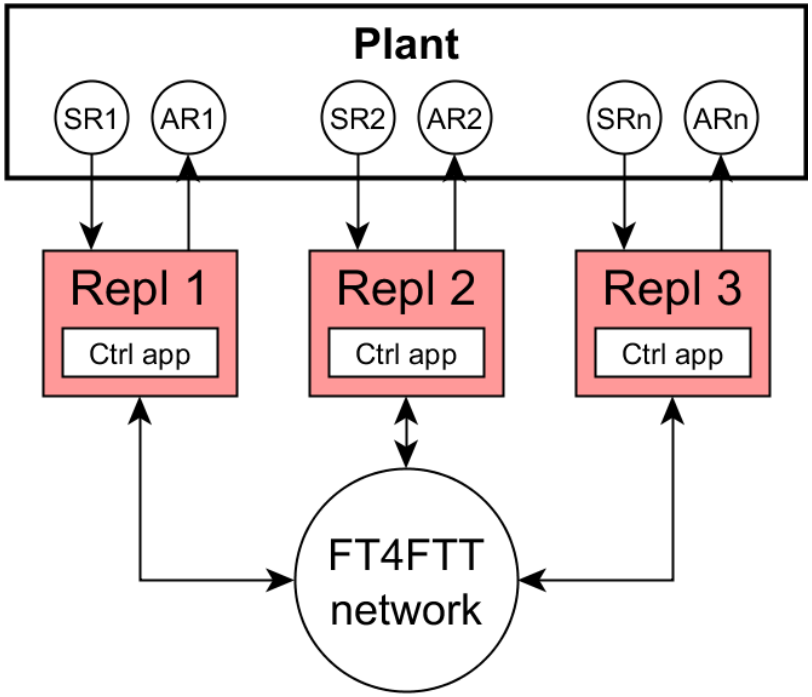
Coordinated triggering of phases



Faults affecting the nodes

Benefits

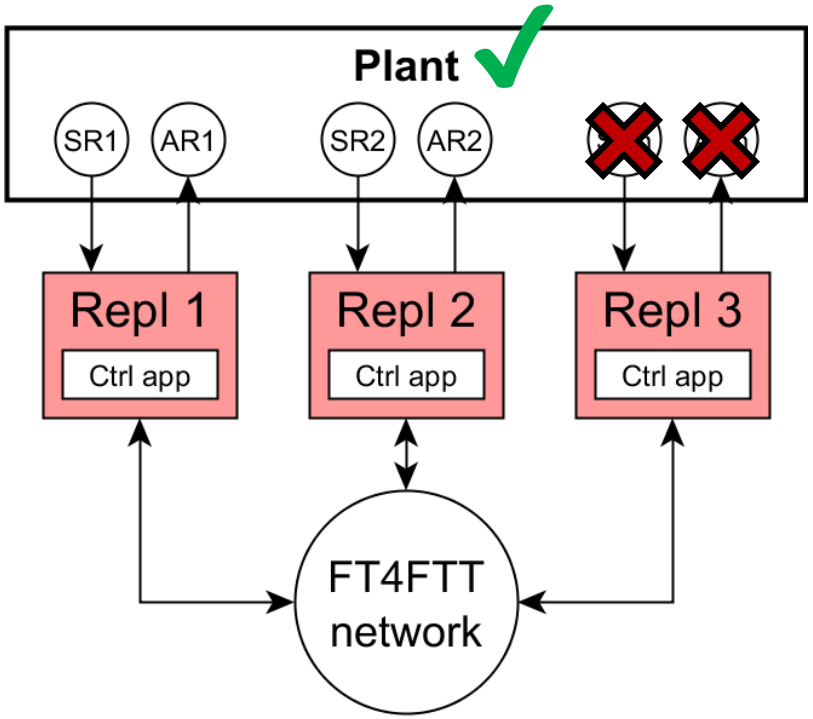
- **Tolerance** to **permanent** faults
- **Tolerance** to **temporary** faults



Faults affecting the nodes

Benefits

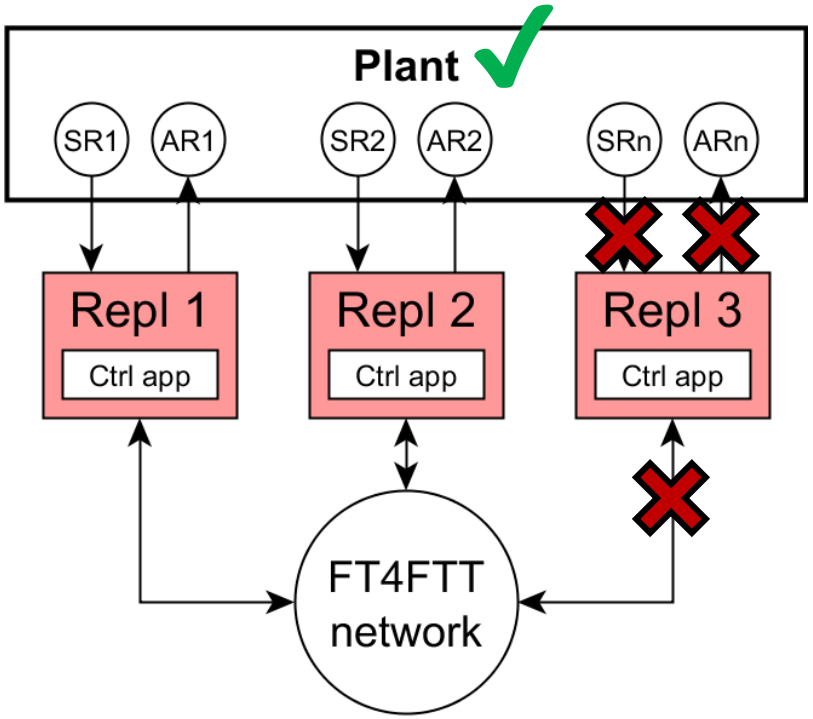
- **Tolerance** to **permanent** faults
- **Tolerance** to **temporary** faults



Faults affecting the nodes

Benefits

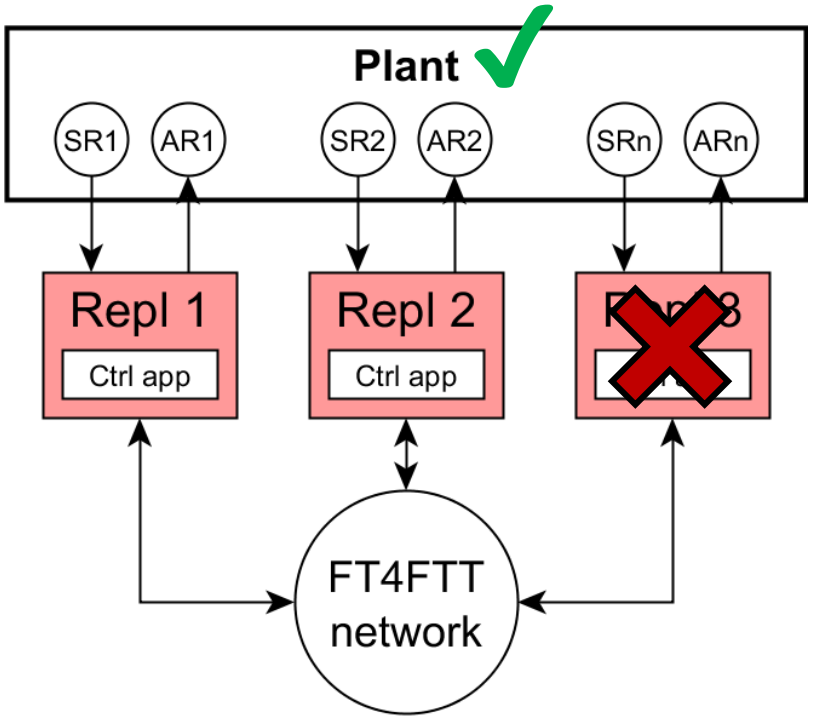
- **Tolerance** to **permanent** faults
- **Tolerance** to **temporary** faults



Faults affecting the nodes

Benefits

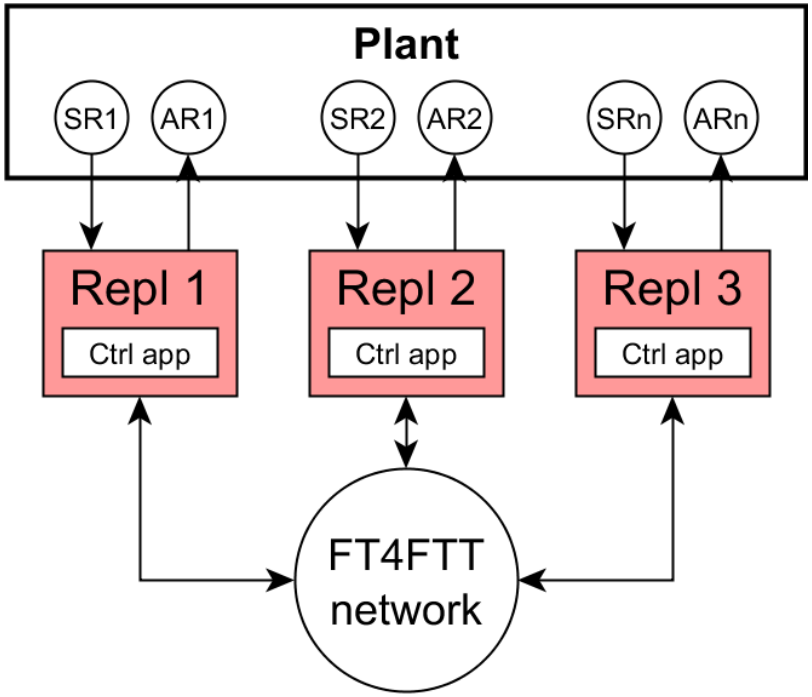
- **Tolerance** to **permanent** faults
- **Tolerance** to **temporary** faults



Faults affecting the nodes

Benefits

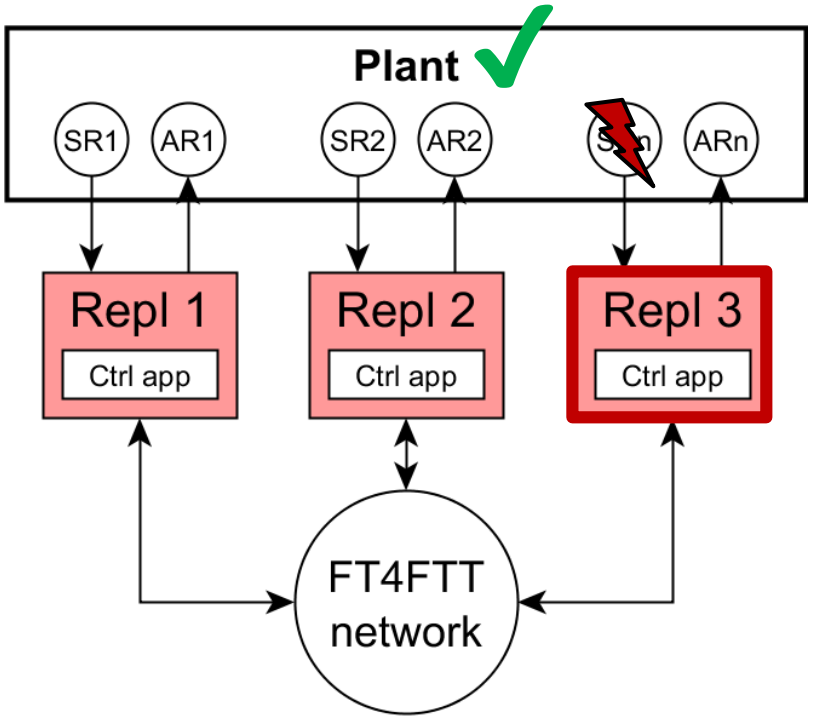
- **Tolerance** to **permanent** faults
- **Tolerance** to **temporary** faults



Faults affecting the nodes

Benefits

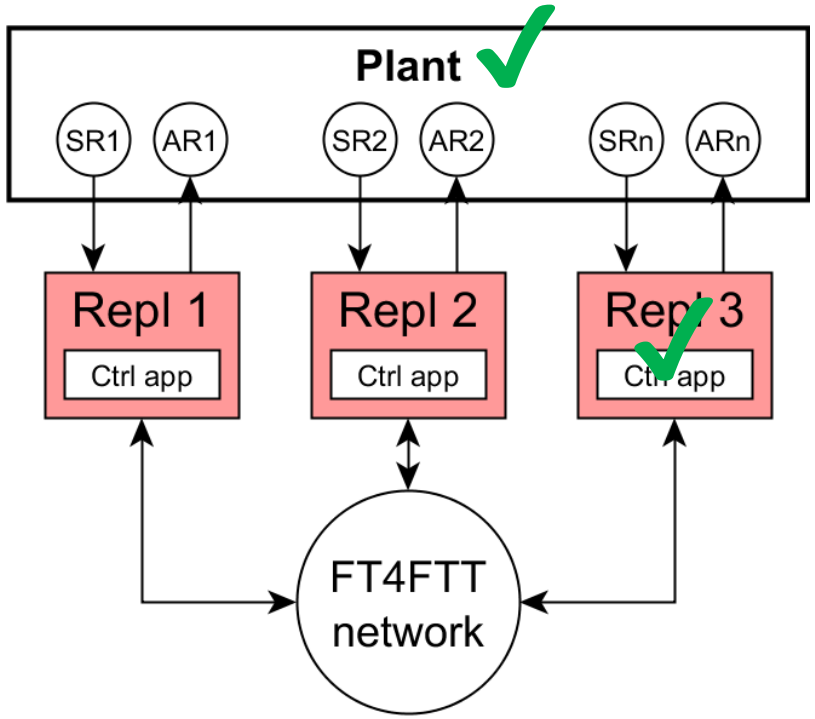
- **Tolerance** to **permanent** faults
- **Tolerance** to **temporary** faults



Faults affecting the nodes

Benefits

- **Tolerance** to **permanent** faults
- **Tolerance** to **temporary** faults

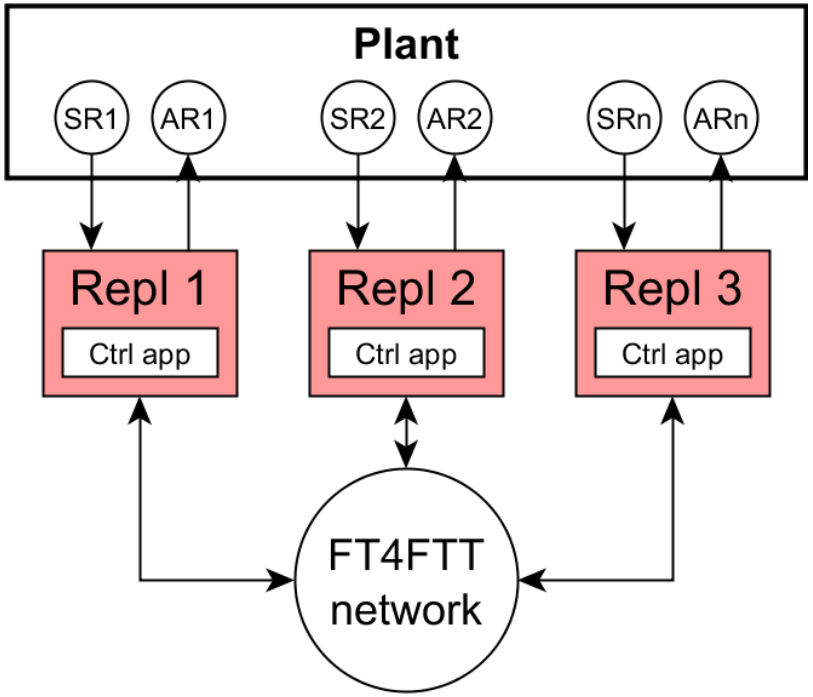


If replica 3 **can vote**
it **can recover**

Faults affecting the nodes

Some temporary faults

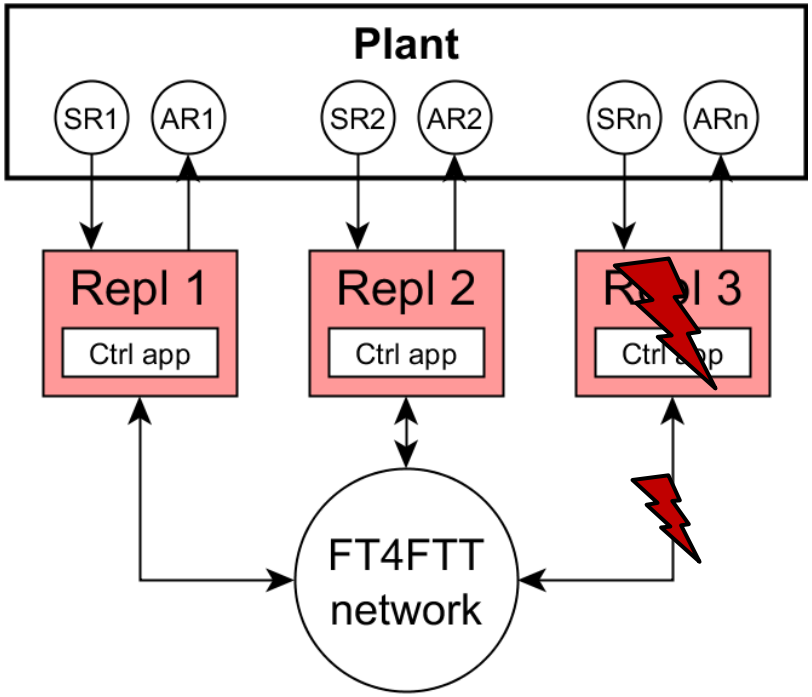
can make the **replica lost from then on**



Faults affecting the nodes

Some temporary faults

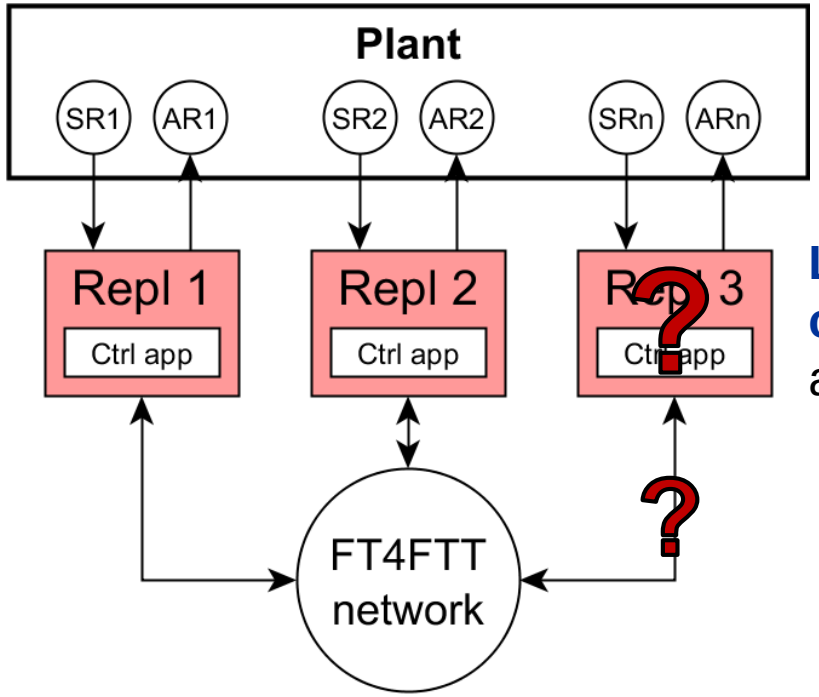
- **Communication** capabilities
- **Internals** of the replica



Faults affecting the nodes

Some temporary faults

- **Communication** capabilities
- **Internals** of the replica

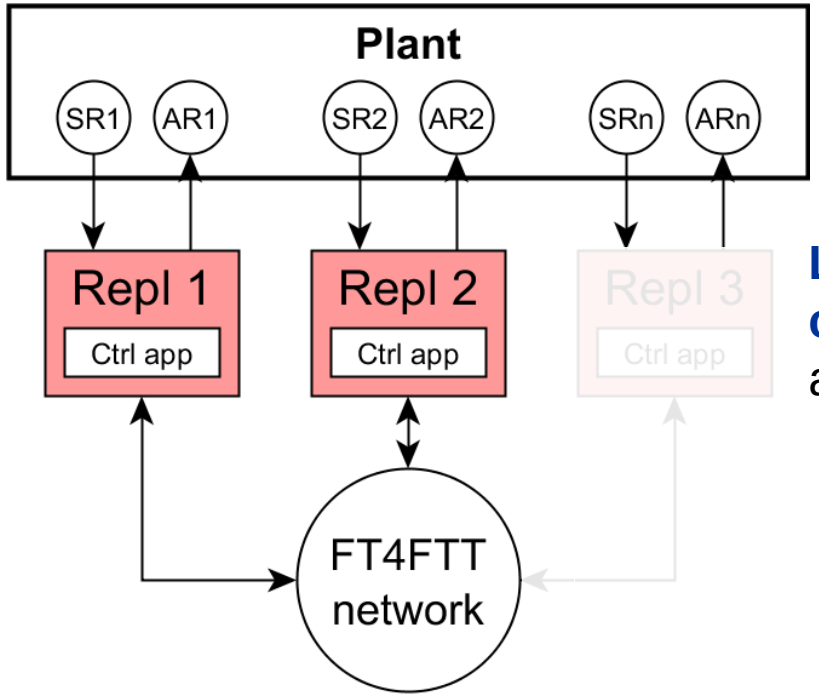


Loss of coordination
communication level
and/or **application** level

Faults affecting the nodes

Some temporary faults

- **Communication** capabilities
- **Internals** of the replica



Loss of coordination communication level and/or **application** level

Replica permanently disabled

Faults affecting the nodes

Temporary faults are **more probable** than **permanent ones**

if we do not **prevent** this **attrition** of the redundancy

we are **not taking full advantage** of the **redundancy investment**

Faults affecting the nodes

Diagnosis and **reintegration** of **discoordinated replicas**

- **Diagnose** discoordinated nodes
- **Reintegrate** them
- As **soon** as possible

Dependability Evaluation - Prism

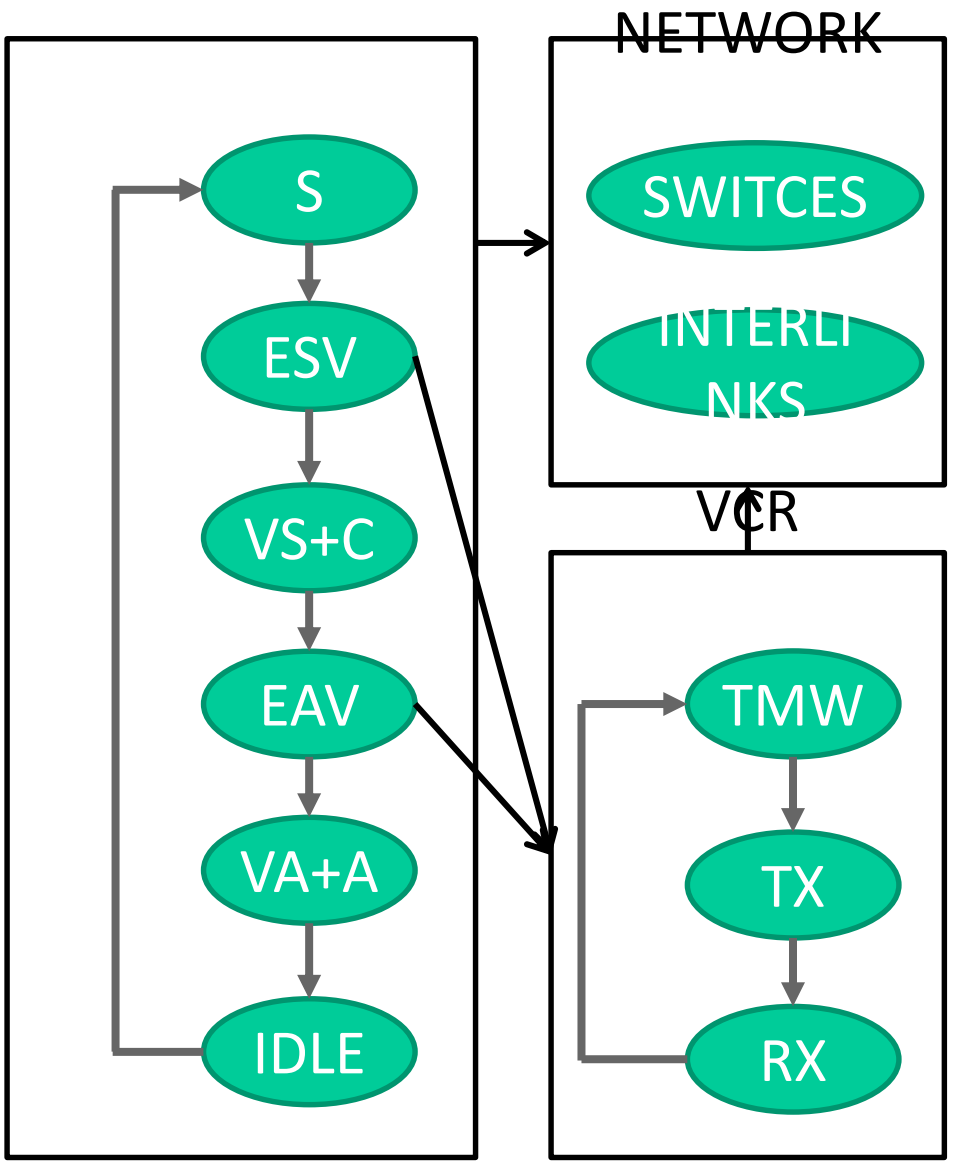
- Prism
  - Probabilistic model checker
    - modeling and analysis
    - systems exhibiting random or probabilistic behaviour
  - Build a model of the system using the prism modeling language
  - Exhaustively check whether the model meets a desired specification expressed in the prism property specification language
  - Why?
    - facilitates scripting – change FT parameters easily
    - state space explosion - symmetry reduction techniques applied on node replicas

Dependability Evaluation - Prism

- Goal
  - Model our system with all the FT mechanisms
    - Node FT in more detail
    - Switch FT in less detail
  - Measure the probability that the system will fail for a desired mission time (1h) - reliability
  - Compare the reliability achieved by adding different FT mechanisms
    - Replication
    - Reintegration
  - Perform sensitivity analysis to determine which FT mechanisms affect the achieved reliability the most and what are the optimal settings for some of the FT parameters, e.g. the number of node replicas, the number of retransmissions, the number of messages' replicas in a single EC, …

Dependability Evaluation - Prism

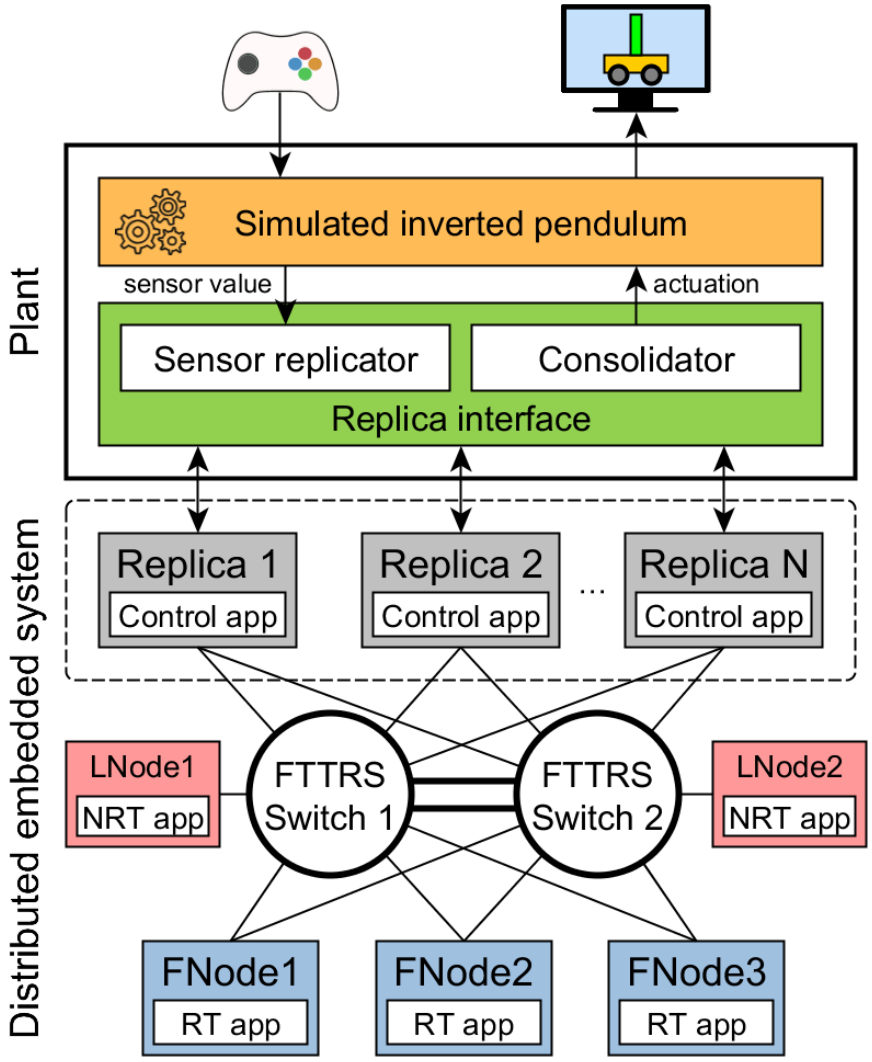
- Faults
  - Permanent
    - Replicas
    - Links
    - Switches
    - Interlinks
  - Temporary
    - Replicas
    - Links – LLFT, PFL
- Modules
  - Replica -> P(system fails)
  - VCR
    - P(cc-vector lost) for all the combination of links and switches faults
  - Network



Outline

- Faults affecting the network
- Faults affecting the nodes
- **Prototyping**
- Issues

Prototyping



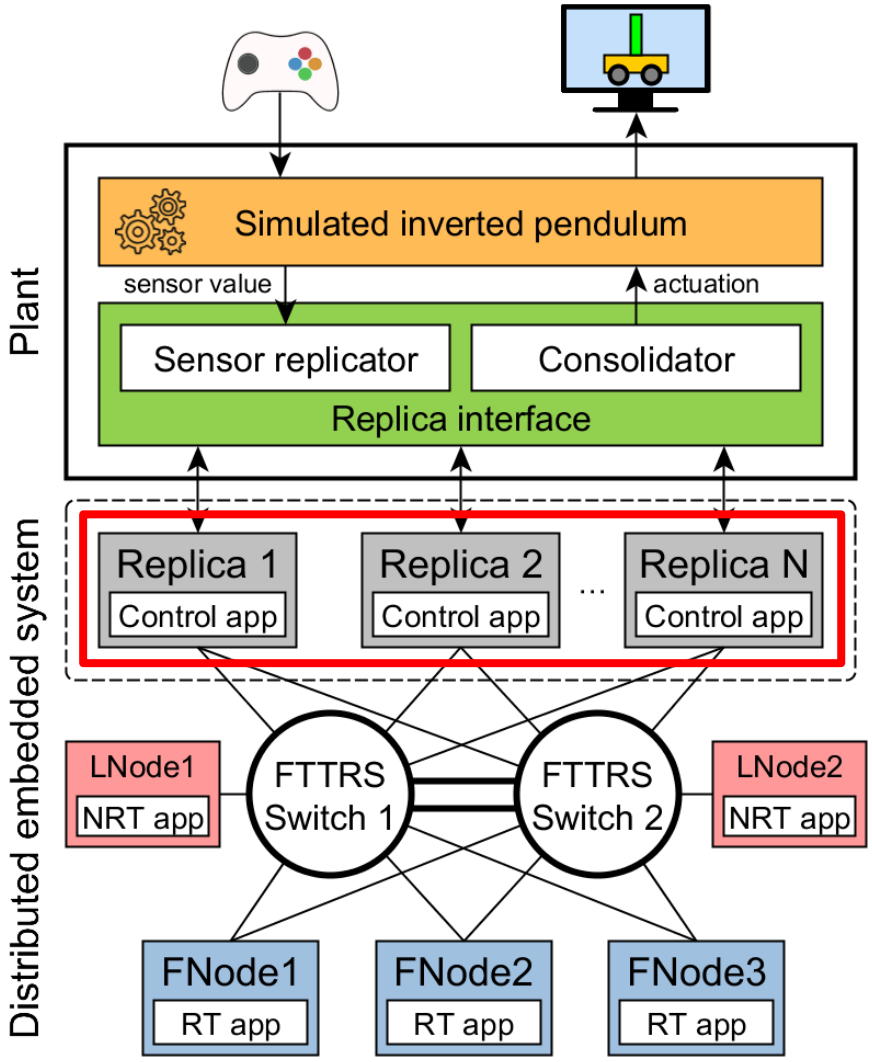
Distributed System

- Control application (RT + reliable)
- Periodic exchange (RT + flex)
- Video stream (non-RT)

Simulated plant

- (Final year project collaboration)
- Hardware-in-the-loop technique
- Inverted pendulum in Simulink
- Replica interface

Prototyping



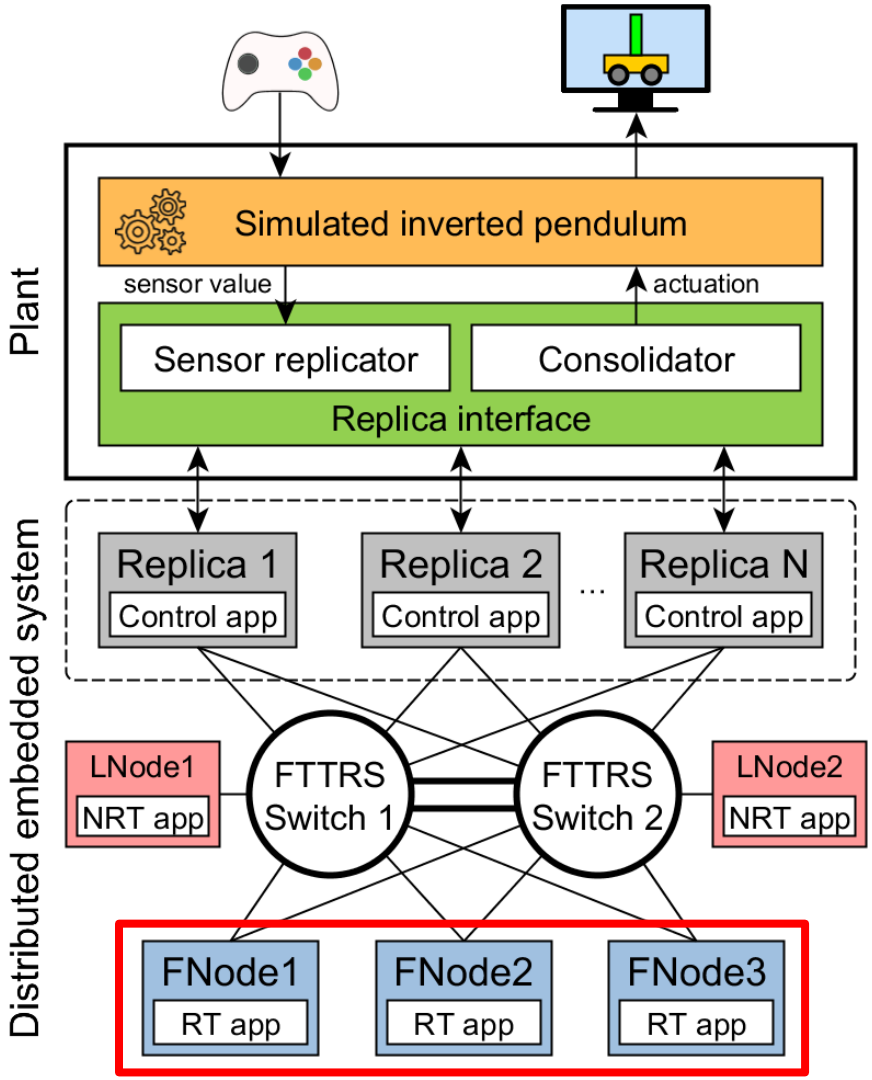
Distributed System

- Control application (RT + reliable)
- Periodic exchange (RT + flex)
- Video stream (non-RT)

Simulated plant

- (Final year project collaboration)
- Hardware-in-the-loop technique
- Inverted pendulum in Simulink
- Replica interface

Prototyping



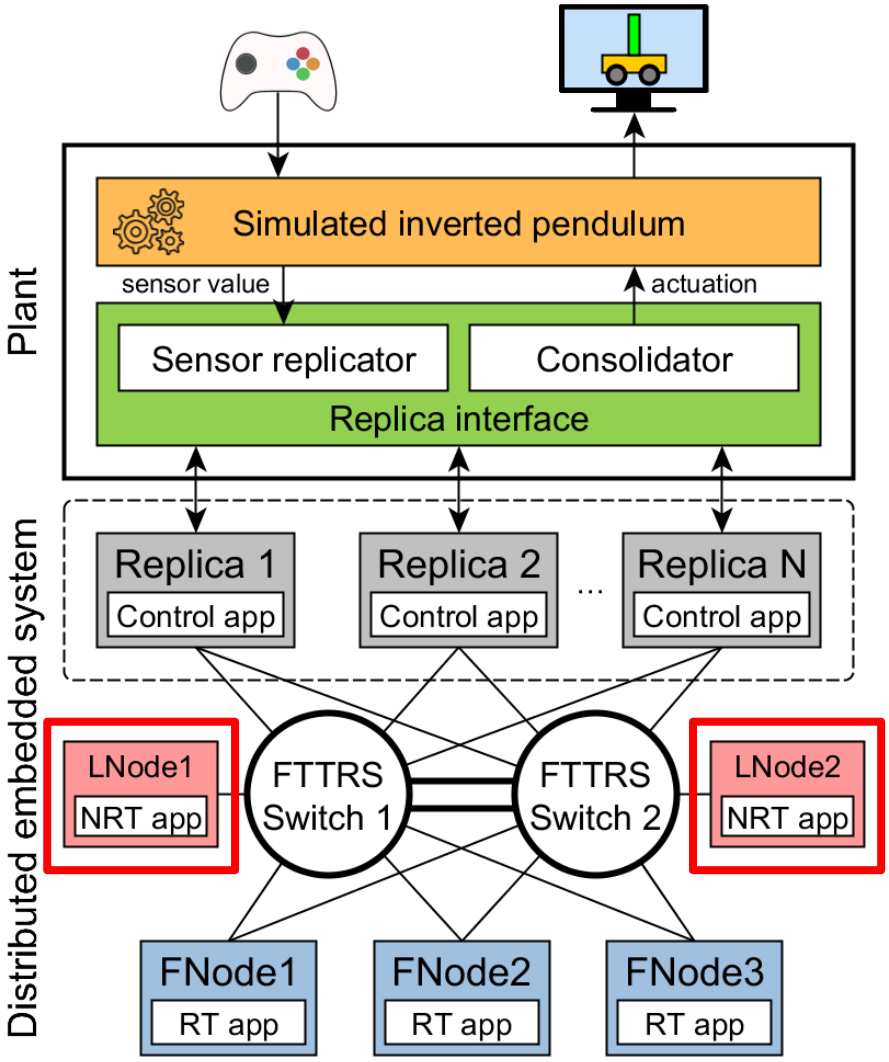
Distributed System

- Control application (RT + reliable)
- Periodic exchange (RT + flex)
- Video stream (non-RT)

Simulated plant

- (Final year project collaboration)
- Hardware-in-the-loop technique
- Inverted pendulum in Simulink
- Replica interface

Prototyping



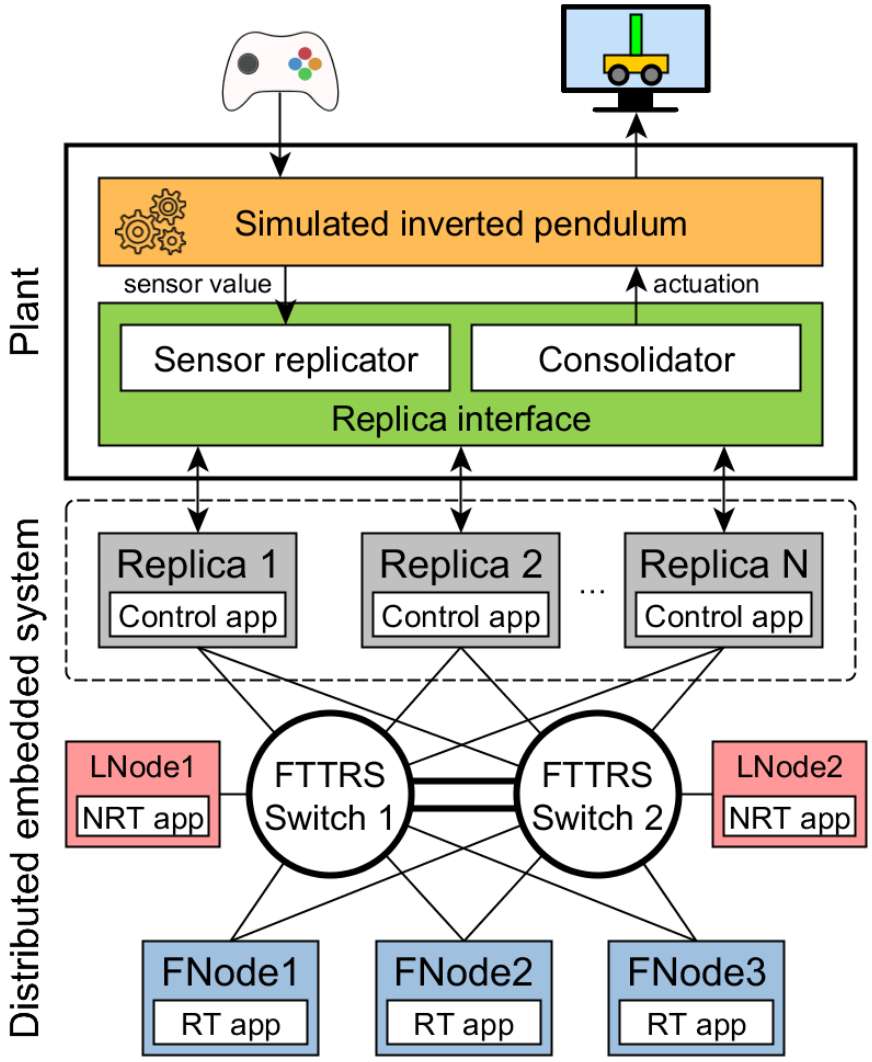
Distributed System

- Control application (RT + reliable)
- Periodic exchange (RT + flex)
- Video stream (non-RT)

Simulated plant

- (Final year project collaboration)
- Hardware-in-the-loop technique
- Inverted pendulum in Simulink
- Replica interface

Prototyping



Distributed System

- Control application (RT + reliable)
- Periodic exchange (RT + flex)
- Video stream (non-RT)

Simulated plant

- (Final year project collaboration)
- Hardware-in-the-loop technique
- Inverted pendulum in Simulink
- Replica interface

Video demo

Outline

- Faults affecting the network
- Faults affecting the nodes
- Prototyping
- **Issues**

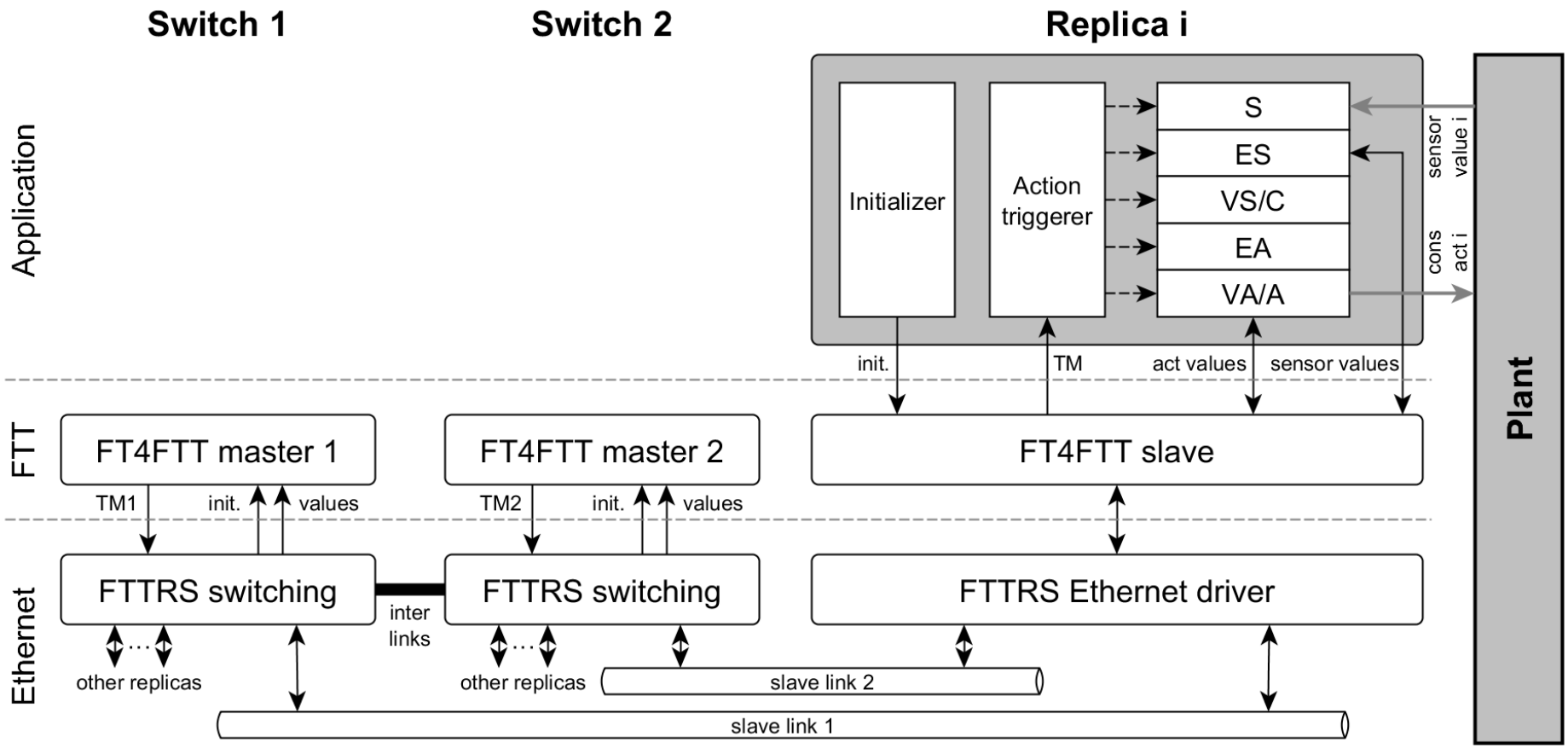
Issues

Software implementation

- **Very flexible**
- **Latency** and **jitter**
- **Uncontrolled growing**
  - FTT-SE to HaRTES
  - FT mechanisms
  - Node replication

Issues

Latency and jitter

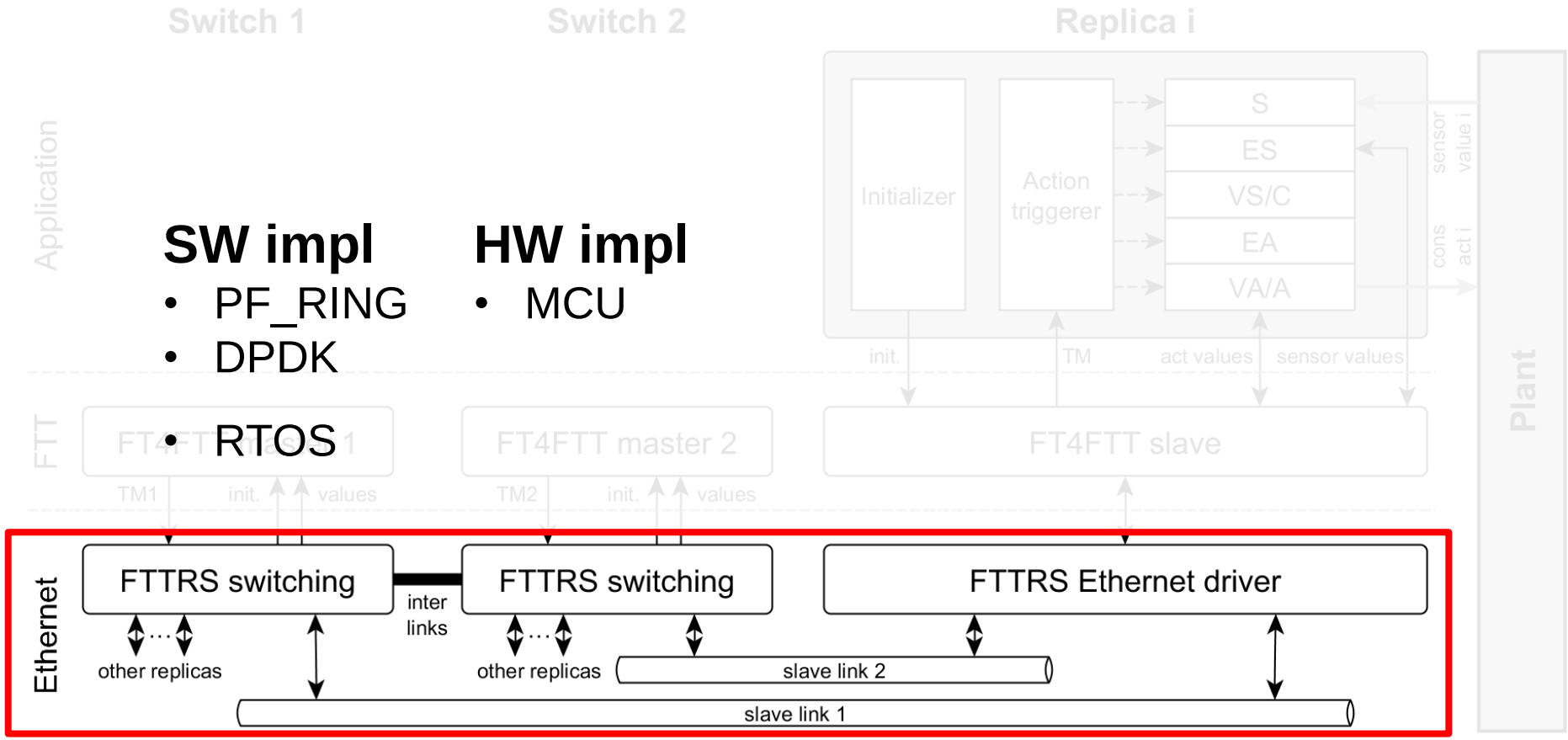


Issues

Latency and jitter

SW impl
- PF_RING
- DPDK
- RTOS

HW impl
- MCU



Issues

Uncontrolled growing

- Define a **software structure** that takes into account the **network** and the **fault tolerance services**
  - David and Ignasi (network and FT perpective)
  - FTT (software perspective)
- Pere and Alberto → new software structure

Issues

How we should proceed in the new project??

- Since we have something **different from FTT** maybe it is a **good idea** to implement **some parts from scratch**
  - We have a general view of the system
  - We now have more knowledge about the technologies
  - Remove what is not needed

  - ➢ Easier implementation → Easier for us and for students
  - ➢ Better RT behavior
  - ➢ Introduce new software paradigms like middleware and SDN

Issues

How we should proceed in the new project??

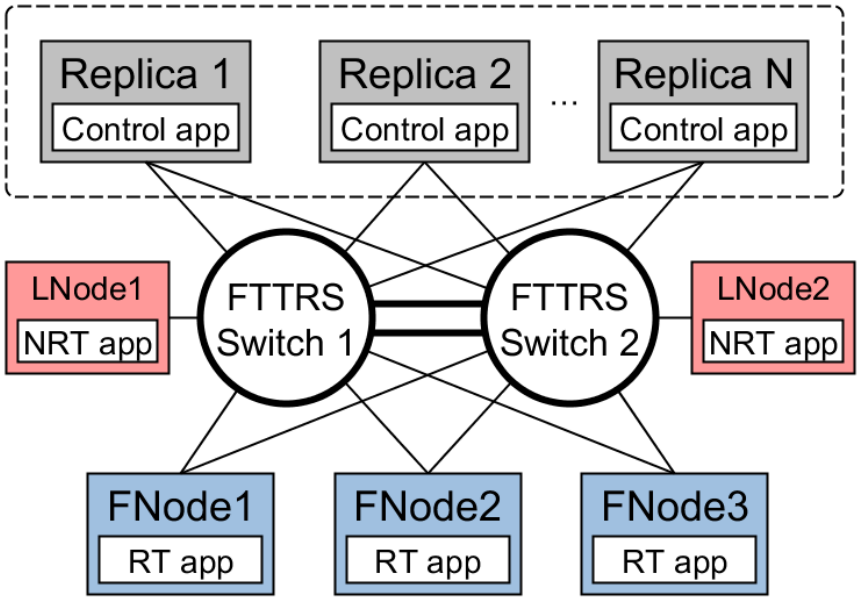
- Since we have something **different from FTT** maybe it is a **good idea** to implement **some parts from scratch**
  - We have a general view of the system
  - We now have more knowledge

New trends in Aveiro and/or Porto?

- Easier implementation → Easier for us and for students
- Better RT behavior
- Introduce new software paradigms like middleware and SDN

FT4FTT prototyping (aspects to improve)

Kick-off meeting of the DFT4FTT project



Alberto Ballesteros