CORA Cork Open Research Archive
Cartlann Taighde Oscailte Chorcaí

Title	Resilient time-sensitive networking for industrial IoT: Configuration and fault-tolerance evaluation
Authors	Seliem, Mohamed;Pesch, Dirk;Roedig, Utz;Sreenan, Cormac
Publication date	2025-07-15
Original Citation	Seliem, M, Pesch, D, Roedig, U & Sreenan, C 2025 'Resilient time-sensitive networking for industrial IoT : Configuration and fault-tolerance evaluation' IEEE International Conference on Emerging Technologies and Factory Automation, ETFA, IEEE, pp. 1-8. https://doi.org/10.1109/ETFA65518.2025.11205659
Type of publication	Article (preprint)
Link to publisher's version	https://www.scopus.com/pages/publications/105021807204 - 10.1109/ETFA65518.2025.11205659
Rights	© 2025, IEEE. This is the author's version of a paper accepted for presentation at the IEEE ETFA 2025 conference. The final version will appear in the conference proceedings. For the purpose of Open Access, the author has applied a CC-BY public copyright license to any Author Accepted Manuscript version arising from this submission - https://creativecommons.org/licenses/by/4.0/
Download date	2026-10-23 05:55:08
Item downloaded from	https://hdl.handle.net/10468/18650

UCC
University College Cork, Ireland
Coláiste na hOllscoile Corcaigh

Resilient Time-Sensitive Networking for Industrial IoT: Configuration and Fault-Tolerance Evaluation

Mohamed Seliem, Dirk Pesch, Utz Roedig, and Cormac Sreenan
School of Computer Science and Information Technology
National Universities of Ireland, University College Cork, Cork, Ireland
MSeliem@ucc.ie, Dirk.pesch@ucc.ie, u.roedig@ucc.ie, Cormac.Sreenan@ucc.ie

arXiv:2507.11250v1 [cs.NI] 15 Jul 2025

***Abstract*—Time-Sensitive Networking (TSN) is increasingly adopted in industrial systems to meet strict latency, jitter, and reliability requirements. However, evaluating TSN's fault tolerance under realistic failure conditions remains challenging. This paper presents IN2C, a modular OMNeT++/INET-based simulation framework that models two synchronized production cells connected to centralized infrastructure. IN2C integrates core TSN features—including time synchronization, traffic shaping, per-stream filtering, and Frame Replication and Elimination for Redundancy (FRER)—alongside XML-driven fault injection for link and node failures. Four fault scenarios are evaluated to compare TSN performance with and without redundancy. Results show that FRER eliminates packet loss and achieves sub-millisecond recovery, though with 2–3× higher link utilization. These findings offer practical guidance for deploying TSN in bandwidth-constrained industrial environments.**

***Index Terms*—TSN, Industrial Ethernet, OMNeT++, Deterministic Networking, Fault Tolerance, Network Resilience.**

I. Introduction

Ensuring deterministic, low-latency, and fault-tolerant communication is vital for industrial automation and control systems. The convergence of Information Technology (IT) and Operational Technology (OT), coupled with the rise of cyber-physical systems, imposes stringent performance and reliability requirements on network infrastructures. While traditional Ethernet remains widespread, it lacks the timing guarantees, traffic isolation, and failure resilience required by real-time industrial applications.

Time-Sensitive Networking (TSN), a suite of IEEE 802.1 standards, augments Ethernet with features such as precise time synchronization, Generalized Precision Time Protocol (gPTP), time aware shaping (TAS), per-stream filtering and policing, VLAN-based prioritization, and Frame Replication and Elimination for Redundancy (FRER). These capabilities collectively enable deterministic behavior, low jitter, and high availability, making TSN a key enabler for industrial Internet of Things (IIoT) and smart manufacturing networks. However, assessing TSN's effectiveness under real-world failure conditions remains an open research challenge.

This paper presents a modular OMNeT++/INET-based simulation framework that models a realistic, multi-cell industrial network equipped with TSN features. Our simulation framework includes configurable traffic patterns, Virtual Local Area Network (VLAN)-based stream prioritization, gPTP-based synchronization, per-stream filtering, and stream redundancy using FRER. Crucially, the framework integrates XML-based failure scenarios to simulate both link and cell-level disruptions and evaluates TSN's recovery behavior with and without redundancy. The key contributions of this work are:

- **Realistic Industrial TSN Modeling:** A full-scale industrial network with two production cells [1] is implemented in OMNeT++, integrating TSN mechanisms such as time synchronization, stream filtering, traffic shaping, and redundancy.
- **Scenario-Based Fault Injection:** Link and node failures are modeled using timed XML scripts, enabling the evaluation of FRER and failover mechanisms under controlled disruptions.
- **Resilience Evaluation:** The framework provides detailed performance insights—including latency, jitter, and recovery time—under both baseline and redundancy-enhanced TSN configurations.

The remainder of this paper is structured as follows: Section II reviews related work on TSN for industrial systems. Section III outlines the system architecture. Section IV details the simulation setup, including topology, traffic modeling, and fault injection. Section V presents results across the four scenarios. Section VI concludes and outlines future directions.

II. Background and Related Work

TSN has emerged as a key enabler of deterministic Ethernet communication for industrial automation, automotive, and other latency-sensitive domains. Defined by the IEEE 802.1 working group, TSN incorporates core features such as time synchronization (IEEE 802.1AS), time-aware scheduling (IEEE 802.1Qbv), per-stream filtering and policing (IEEE 802.1Qci), and Frame Replication and Elimination for Redundancy (FRER, IEEE 802.1CB) to ensure low-latency, bounded jitter, and high availability of critical traffic flows [2]. These mechanisms allow Ethernet to meet the stringent requirements of Industrial Internet of Things (IIoT) applications, replacing traditional fieldbus and proprietary real-time solutions [3].

Extensive research has confirmed the benefits of TSN in reducing jitter and improving deterministic behavior in mixed-criticality environments [4]. Simulation-based studies—particularly those using OMNeT++ and the INET frame-

This publication has emanated from research conducted with the financial support of Research Ireland under Grant number 13/RC/2077 P2. For the purpose of Open Access, the author has applied a CC-BY public copyright license to any Author Accepted Manuscript version arising from this submission

work—have validated the effectiveness of TSN for traffic prioritization, scheduling, and reliability in industrial scenarios [5]. Fault tolerance, a cornerstone of industrial communication, is typically achieved in TSN using redundant streams and fast fail-over techniques such as FRER [6]. These features are especially relevant in factory networks where uptime and predictable performance are critical.

While TSN has gained traction in automotive networking [7], its deployment in industrial automation—particularly under realistic fault conditions—remains an open research challenge. Recent work has examined TSN's integration with legacy protocols such as PROFINET and EtherCAT [8], and proposed new scheduling strategies for optimizing latency and bandwidth usage [9]. Beyond scheduling, several studies have focused on fault-tolerance and redundancy in TSN and wireless TSN systems. Sudhakaran et al. [10] introduced a dynamic scheduling approach enabling zero-delay roaming for mobile robots using wireless TSN redundancy, showing that FRER can uphold real-time guarantees under mobility. Jover et al. [11] analyzed the trade-off between reliability and redundancy cost in TSN networks through simulation-based configuration insights. On the wireless side, Cena et al. [12] proposed seamless redundancy techniques for high-reliability Wi-Fi, leveraging Wi-Fi 7 Multi-Link Operation (MLO) to implement FRER-like duplication and de-duplication. Those complement our study by tackling advanced fault-handling approaches. However, most existing efforts focus on static performance or isolated failure events, overlooking complex failure injection scenarios with synchronized industrial traffic. Our work addresses this gap by introducing a scenario-driven simulation framework (IN2C) that models industrial production cells, real-time traffic, and coordinated failure events. It evaluates four distinct failure configurations, each with and without FRER, to quantify TSN's fault-recovery capabilities in terms of latency, jitter, and recovery time. The novelty of this work lies in:

- Comprehensive fault injection across layered industrial traffic flows, using XML-triggered scenarios and real-world-inspired production logic;
- Comparative evaluation of TSN with and without redundancy, enabling quantitative insights into the impact of FRER and per-flow protection under failure;
- Multi-role traffic modeling (sensor-actuator-PLC-SCADA-Edge unit-HMI), reflecting realistic IIoT interactions rarely combined at this level of detail in existing TSN studies.

These contributions provide a deeper understanding of TSN's resilience in industrial environments and offer a reusable simulation testbed for future TSN extensions and industrial scheduling strategies.

III. System Model and Design Considerations

The industrial TSN network designed in this work models a realistic smart factory with two production cells and centralized infrastructure, incorporating sensing, actuation, control, inspection, SCADA, HMI, and edge computing units over wired TSN links. The section outlines the network architecture, traffic models, node configurations, fault injection methods, and performance metrics used to evaluate system behavior under both normal and failure conditions.

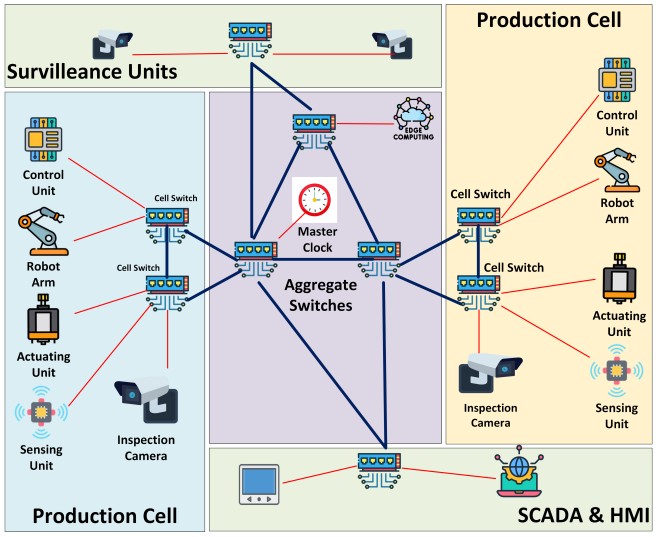

Fig. 1. Industrial TSN Network with Two Production Cells

TABLE I
Industrial Network Components

Component	Function
PLC (Control Unit)	Central logic controller coordinating sensing, actuation, and robotic tasks.
Sensing Unit	Detects object arrival, triggers inspection, and sends metrics to PLC/Edge unit.
Inspection Camera	Captures high-frequency images and sends them to Edge unit for quality analysis.
Robot Arm	Handles objects, sends status updates to PLC and SCADA.
Actuating Unit	Performs reject/eject operations based on PLC commands.
HMI	Interfaces with operators; receives visual feeds and status information.
SCADA	Supervisory unit for monitoring and control across cells and infrastructure.
Edge unit	Receives and processes data from inspection and sensing units. Sends logs to SCADA.
Cameras (Surveillance)	Broadcast high-rate video streams to SCADA and HMI.
Master Clock	Root clock that distributes time via gPTP across all TSN-enabled devices.

A. Industrial Network Architecture and Configuration

The industrial network follows a modular, hierarchical architecture designed to emulate realistic industrial automation environments. It comprises two independent production cells, each capable of autonomous operation, and a centralized infrastructure responsible for supervision, monitoring, and edge processing. The network topology includes cell-level TSN switches, central aggregation switches, and interconnecting links configured for redundancy and time synchronization.

Each production cell integrates five key functional units: a sensing unit, actuating unit, robotic arm, inspection system,

and a control unit (PLC). These units, detailed in Table I, are dual-homed through two TSN-capable switches (SwitchA_X and SwitchB_X, where X denotes the cell index), forming a fault-tolerant local sub-network. This dual-switch design allows for redundant forwarding paths toward central infrastructure via centralSwitch_1, centralSwitch_2, and centralSwitch_3. The backbone consists of 1-Gbps Ethernet links interconnecting switches at the core and aggregation layers. These include:

- $centralSwitch_1$: Core switch connecting both cells.
- $centralSwitch_2$ and $centralSwitch_3$: Interconnect with SCADA, HMI, Edge unit, and quality-control switches ($qcSwitch_1$, $qcSwitch_2$).

End devices (e.g., PLCs, sensors, actuators) are connected via 100-Mbps links, representing edge connections commonly found in factory deployments. Figure 1 illustrates this architecture. All switches and hosts are configured with TSN capabilities under the TSNBase configuration. Table II summarizes the TSN features implemented in the network, as configured in the INET-based .ini file.

TABLE II
TSN FEATURES AND THEIR IMPLEMENTATION

TSN Feature	Implementation Details
Time Synchronization	IEEE 802.1AS via a dedicated master clock module (masterClock).
Traffic Scheduling	Always-open gate schedule across all egress ports.
Credit-Based Shaping	Enabled per-port on all TSN switches.
Stream Identification	Per-stream filtering and Priority Code Point (PCP)-based tagging (e.g., 6 for control, 5 for sensing, 2 for cameras).
Redundancy (FRER)	Frame replication and elimination enabled via StreamRedundancyConfigurator in selected scenarios.
Per-Stream Filtering	Each TSN switch performs stream classification and rate metering.
Network Speed	Backbone: 1 Gbps for all inter-switch links; Edge: 100 Mbps for all device links.

This architectural design provides a robust and flexible testbed for evaluating TSN performance in a realistic industrial setting, enabling detailed investigation of latency, jitter, and recovery under fault conditions.

B. Traffic Characterization and Fault Resilience

The industrial network models a diverse set of traffic streams that reflect the communication requirements of industrial automation systems. These streams are classified by source-destination roles, communication patterns (periodic vs. event-driven), packet size, and criticality level. TSN mechanisms—including stream identification and Priority Code Point (PCP)-based prioritization—are configured to ensure that high-priority traffic receives deterministic service, even under network stress or failure. Table III summarizes key traffic types modeled in the simulation, along with their associated PCP values and timing constraints.

All streams are tagged and encoded using PCP values, as defined by per-stream mapping in the configuration. TSN switches perform ingress classification, policing, and shaping per stream to enforce bandwidth and burst control.

To evaluate the fault resilience of the TSN network, two types of failure scenarios are introduced:

- S1A1/S1A2 simulate a mid-cell link failure (SwitchA_1 - SwitchB_1),
- S2A1/S2A2 simulate cell disconnection (centralSwitch_2 - Edge unit and SCADA).

Table IV summarizes these fault scenarios and their recovery logic. The following metrics are used to evaluate network behavior during fault events:

- **End-to-End Latency:** Transmission delay from source to destination.
- **Jitter:** Variability in packet delay over time.
- **Packet Loss:** Number of packets dropped during normal and faulted operation.
- **Recovery Time:** Time required to reestablish stream delivery after a failure.

Algorithm 1 Initialization of IN2C TSN Simulation

```
BEGIN Initialization
Load IN2C topology, TSN configuration, and fault scenario script.
for each node do
    Configure clock with drift model.
    Set stream encoder/decoder mappings.
    Load UDP application traffic profile (size, interval, PCP).
end for
Establish topology using TSN switches and assign gPTP roles.
END Initialization
```

IV. FRAMEWORK ARCHITECTURE AND WORKFLOW

The IN2C simulation framework is implemented in OMNeT++ using the INET framework, extended with TSN-specific modules to emulate industrial communication behavior. The simulation models two synchronized production cells and centralized infrastructure, capturing interactions across sensors, PLCs, robot arms, inspection units, SCADA, Edge unit, and HMI systems. INET's TSN extensions—including time-aware switches, gPTP clocks, per-stream filtering, and traffic shaping—enable deterministic and fault-tolerant communication modeling.

All nodes are built from INET's TsnDevice and StandardHost templates, depending on their role. Each functional unit (e.g., Sensing_1, Control_1, Inspection_2) includes:

- A local clock module with oscillator drift,
- User Datagram Protocol(UDP) traffic generators and sinks,
- Stream coders with PCP-tag mapping,
- Bridging and scheduling logic for TSN operation.

Cell switches (SwitchA_X, SwitchB_X) and aggregation switches (centralSwitch_X, qcSwitch_X) use the TsnSwitch

TABLE III
REPRESENTATIVE INDUSTRIAL TRAFFIC STREAMS

Traffic Type	Source - Destination	PCP	Typical Timing
PLC Commands	Control_X - Actuating_X / RobotArm_X	6	5-10 ms cyclic
Sensor Data	Sensing_X - Control_X	6	5-10 ms cyclic
Object Detection	Sensing_X - Control_X	5	10–50 ms event-driven
Inspection Images	Inspection_X - Edge unit	5	33–100 μs periodic
Barcode/RFID Scans	Inspection_X - Control_X	3	10–50 ms event-driven
Robot Status Feedback	RobotArm_X - Control_X	5	5–20 ms periodic
Camera Surveillance	Camera_X - SCADA/HMI	2	30 FPS (330 μs)
SCADA Reports	Control_X - SCADA	4	10–100 ms periodic
Operator Input	HMI - SCADA	3	Sporadic / user-driven
Edge unit Logs	Edge unit - SCADA	1	1–10 sec periodic

TABLE IV
FAULT INJECTION AND RECOVERY STRATEGIES

Fault Type	**Description**	**Recovery Mechanism**
Link Failure (S1A1/S1A2)	Disconnection between SwitchA_1 and SwitchB_1 at 4s.	FRER (S1A2) provides alternate paths using stream replication.
Cell Disconnection (S2A1/S2A2)	centralSwitch_2 disconnects from SCADA and Edge unit at 2s.	Redundant inter-switch paths restore communication (S2A2).
No Redundancy (S1A1/S2A1)	Baseline behavior without FRER or protection.	Traffic interruption occurs with no failover.

type, with FRER, stream filtering, and shaping enabled as configured.

The master clock (masterClock) operates as a global time reference, distributing synchronization via gPTP across the topology. Each node aligns its local clock to this source, ensuring coherent transmission timing for time-sensitive streams.

The workflow follows an event-driven model:

- Initialization: Node clocks are configured, traffic applications instantiated, stream encoders applied, and synchronization established.
- Traffic Generation: Application modules produce cyclic or event-driven UDP streams with preconfigured packet size, interval, and destination.
- Scheduling: Time-aware and credit-based shaping modules manage queueing and dispatch under bounded latency constraints.
- Failure Handling: XML-based ScenarioManager scripts trigger link/node failures (e.g., linkfailure.xml or cellfailure.xml).
- Redundancy/Recovery: If redundancy is enabled, FRER-based failover automatically engages. Otherwise, communication may stall.
- Metrics Collection: Throughout the simulation, OMNeT++ records delay, jitter, packet drops, and fault recovery durations.

Synchronization is driven by IEEE 802.1AS. Each switch forwards sync messages downstream to maintain precise offset and frequency alignment. TSN traffic scheduling uses always-open TAS gates and per-stream credit-based shaping. This configuration is intentionally chosen to isolate the impact of shaping and redundancy mechanisms by avoiding the additional complexity introduced by time-aware gate scheduling.

Algorithm 2 Time Synchronization via gPTP

```
BEGIN TimeSync
for each TSN node do
    Receive gPTP sync from upstream switch.
    Adjust local oscillator phase and frequency.
    Propagate time to downstream nodes.
end for
END TimeSync
```

Streams are filtered and metered based on source and PCP. Faults are introduced at simulation time using scheduled disconnect/connect events. Redundant configurations trigger alternate paths using FRER trees and packet duplication. This modular, event-driven structure enables controlled experimentation with traffic shaping, synchronization, stream redundancy, and recovery behavior under realistic factory network conditions.

V. RESILIENCE EVALUATION UNDER NETWORK FAILURES

This section analyzes the performance of the IN2C TSN network under fault conditions affecting link and cell connectivity. Four scenarios are evaluated to examine the impact of redundancy on latency, jitter, packet loss, and recovery time.

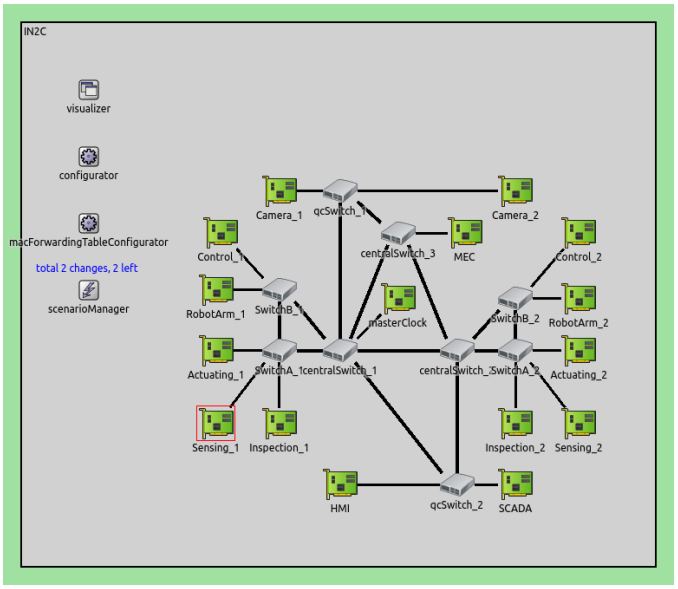

Fig. 2. TSN based Industrial Network of Two Production Cells (IN2C).

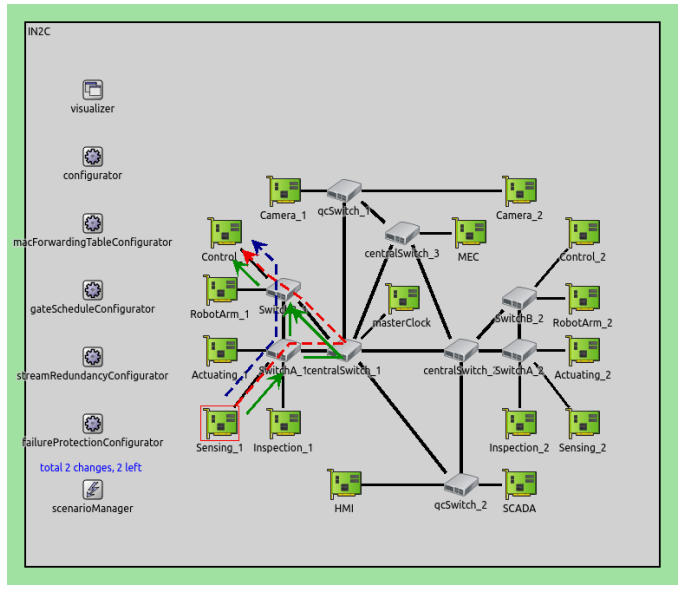

Fig. 3. Active FRER to provide link failure protection.

Results reflect how TSN mechanisms perform when subject to realistic disruptions in smart manufacturing environments.

Algorithm 3 TSN Traffic Scheduling and Forwarding

```
BEGIN Scheduling
for each output queue do
    Filter packets per PCP and stream ID.
    Apply shaping (idle slope, burst control).
    Dispatch if time gate is open.
end for
END Scheduling
```

Algorithm 4 Fault Injection and Recovery

```
BEGIN FaultEvent
if link/node failure triggered then
    Disconnect affected modules via gate logic.
    if FRER enabled then
        Activate alternate stream path.
    end if
    Start recovery timer.
end if
if repair triggered then
    Reconnect modules and synchronize clocks.
    Log recovery metrics.
end if
END FaultEvent
```

A. Experimental Setup

The evaluation is based on the IN2C simulation framework [13], developed in OMNeT++ using the INET-TSN extension. The network, Figure 2, consists of two TSN-enabled production cells connected to centralized infrastructure (SCADA, Edge unit, HMI) through a 1-Gbps backbone. End devices communicate via 100-Mbps links. Each simulation captures behavior across initialization, steady-state, failure, and recovery phases. In this context, four scenarios are defined:

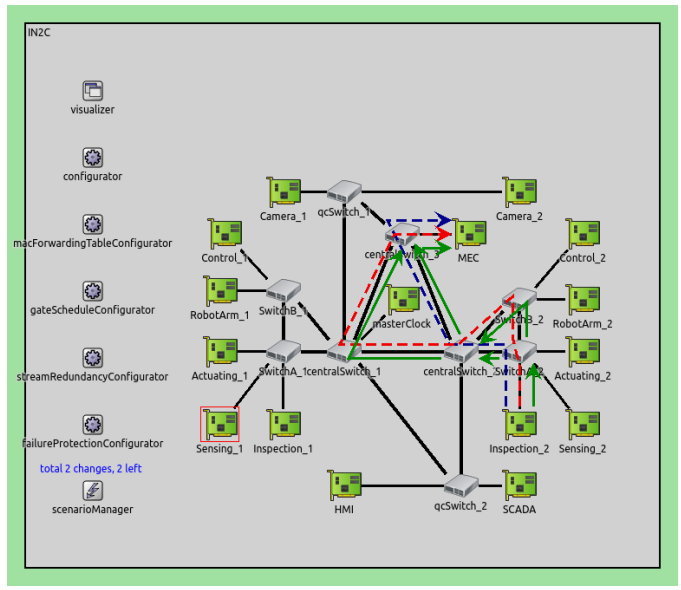

Fig. 4. Active FRER to provide cell failure protection.

- **S1A1 / S1A2:** Simulate a link failure between `SwitchA_1` and `SwitchB_1` at 4 seconds. S1A2, Figure 3, enables FRER; S1A1 does not. Runtime: $10s$.
- **S2A1 / S2A2:** Simulate cell isolation by disconnecting `centralSwitch_2` from both `Edge unit` and `SCADA` at 2 seconds. S2A2, Figure 4, uses FRER for redundancy. Runtime: $5s$.

Failures are introduced using XML-based `ScenarioManager` scripts, which explicitly disconnect and

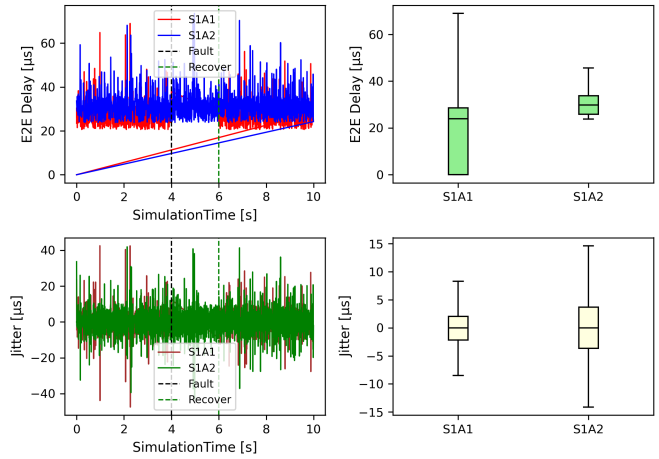

Fig. 5. End to End latency and Jitter, Sensing_1/Control_1 stream, comparison while handling link failure.

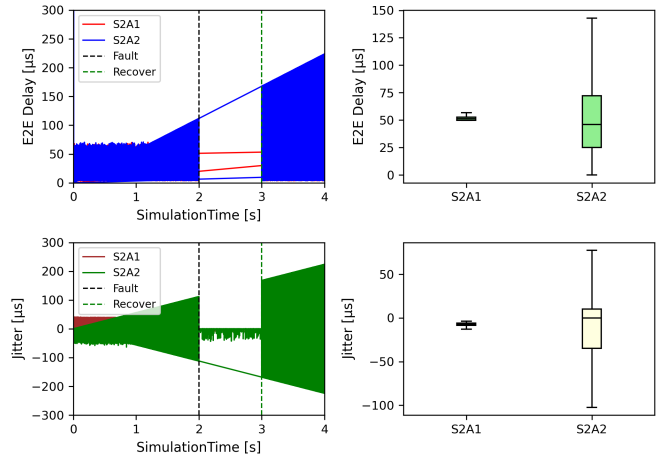

Fig. 6. End to End latency and Jitter, Inspection_2/Edge unit stream, comparison while handling cell failure.

later restore Ethernet gates between modules. No node-level failures are used; instead, link-level disruptions emulate practical fault cases such as cable damage or aggregation switch malfunction.

Redundancy mechanisms, including Frame Replication and Elimination for Redundancy (IEEE 802.1CB), are activated only in S1A2 and S2A2 via `StreamRedundancyConfigurator`. Gate control and clock synchronization are implemented using `EagerGateScheduleConfigurator` and IEEE 802.1AS-compliant gPTP via a centralized `masterClock`. The full simulation source code, scenarios, and TSN configurations are publicly available at [13].

B. Latency and Jitter Under Failure

Figure 5 shows the performance of the S1 stream under a link failure between `SwitchA_1` and `SwitchB_1` injected at $t = 4$ s and resolved at $t = 6$ s. In the non-redundant setup (S1A1), latency becomes highly variable post-failure, with a median of 24.01μs and a wide spread (mean 18.93μs). Jitter increases significantly, with a standard deviation of 6.29μs due to unstable retransmissions and path recovery effects. Despite the existence of physical alternate paths, their absence from the active stream configuration renders the stream vulnerable to the failure.

In contrast, the redundant configuration (S1A2) uses FRER to pre-distribute stream replicas over disjoint paths. This eliminates the need for dynamic rerouting or recovery signaling. As expected, latency remains consistently bounded despite the fault, though its median rises slightly to 29.81μs. This increase stems from buffering delay introduced by duplicate frame handling. The trade-off yields a more stable mean delay (27.43μs) and enhanced fault tolerance. Jitter remains controlled (std 8.13μs) as the FRER mechanism ensures that the first valid frame at the receiver is delivered without delay, masking the failure entirely.

Figure 6 presents the S2 stream under a cell-level fault, where `centralSwitch_2` is disconnected from the Edge and SCADA units at $t = 2$ s and reconnected at $t = 3$ s.

In the non-redundant case (S2A1), latency spikes sharply, with a median of 51.28μs and peaks over 90 μs. Jitter remains moderate (std 17.32μs), reflecting consistent recovery behavior once the fault is resolved.

In contrast, S2A2 with FRER exhibits a more complex profile. While the median latency improves slightly to 45.99μs, the mean increases to 63.27μs, and the tail becomes pronounced—latency reaches up to 879μs, and jitter swings from –224 to +224 μs. This indicates that while FRER masks the failure itself, it introduces significant delivery variability even under normal operation. The congestion between `centralSwitch_1` and `centralSwitch_3`, likely due to duplicate stream forwarding, amplifies path imbalance and leads to large jitter fluctuations. These results reveal that FRER is not universally beneficial. While it provides failover protection, it may degrade performance for bursty or high-bandwidth streams like inspection traffic.

The right-hand boxplots in both figures confirm these observations statistically. In S1, the non-redundant configuration exhibits outliers and jitter spread during failure, while FRER (S1A2) achieves consistently low variance, validating its effectiveness for critical, low-volume streams. However, in S2, FRER (S2A2) displays a heavy-tailed distribution and wide jitter range, in contrast to the more stable profile of the non-redundant setup (S2A1). These results highlight that while FRER prevents packet loss and ensures continuity, it can introduce delivery unpredictability under normal load, particularly for high-throughput or bursty traffic.

C. Packet Loss and Recovery Analysis

To quantify the resilience of each configuration, Table V summarizes packet transmission, loss, and recovery characteristics for all scenarios.

TABLE V
ESTIMATED PACKET LOSS, RECEPTION, AND LOSS RATE PER STREAM

Scenario	Stream	Sent	Lost	Received	Loss Rate	Redundancy	Recovery
S1A1	Sensing_1 → Control_1	1,337	270	1,067	20.2%	No	2.0 s
S1A2	Sensing_1 → Control_1	1,331	0	1,331	0.0%	FRER	$<$ 1 ms
S2A1	Inspection_2 → Edge unit	75,096	19,776	55,320	26.3%	No	1.0 s
S2A2	Inspection_2 → Edge unit	75,126	0	75,126	0.0%	FRER	$<$ 1 ms

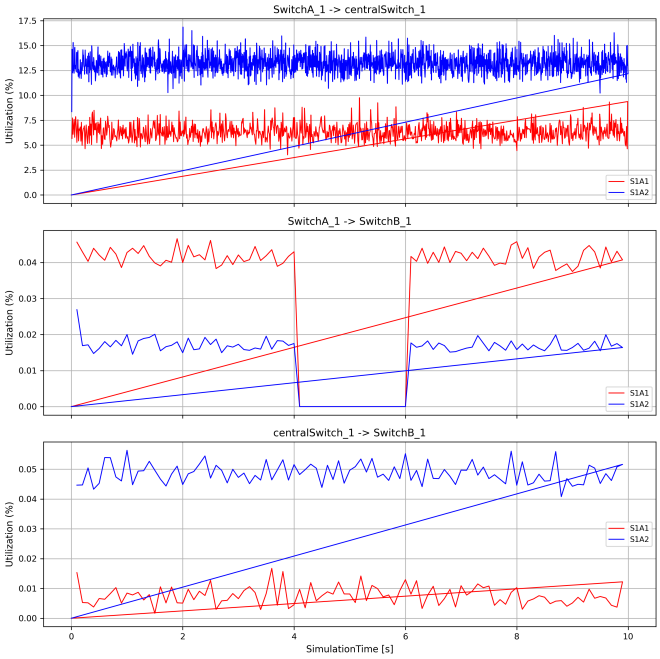

Fig. 7. Links utilization for the `Sensing_1` → `Control_1`.

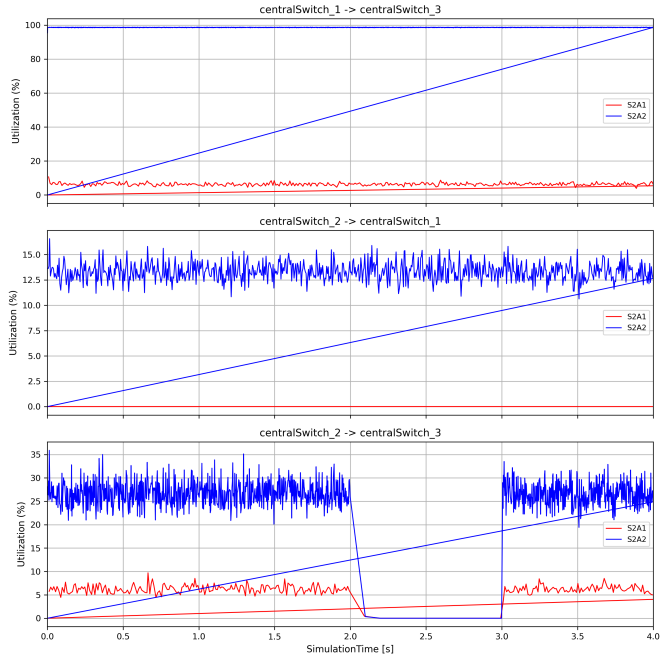

Fig. 8. Links utilization for the `Inspection_2` → `Edge unit`.

For the `Sensing_1` → `Control_1` stream (S1), which transmits every 7.5 ms on average, the S1A1 configuration experiences 270 lost packets during the 2-second link failure—yielding a 20.2% loss rate. Recovery is purely reactive and delayed until the link is restored. In contrast, FRER-enabled S1A2 maintains continuous delivery across disjoint paths, with no packet loss and instantaneous failover.

The `Inspection_2` → `Edge unit` stream operates at a high rate (66 μs intervals), and the S2A1 setup suffers 19,776 dropped packets during a 1-second isolation, corresponding to a 26.3% loss rate. Despite available physical paths, the absence of redundancy prevents their use. S2A2, leveraging FRER, avoids loss by using duplicate streams; however, as discussed earlier, this comes at the cost of significant jitter and latency variability.

This study compares FRER against a non-redundant baseline, excluding traditional protection schemes like Spanning Tree Protocol (STP), Rapid STP (RSTP), and Ethernet Ring Protection Switching (ERPS), which require tens to hundreds of milliseconds to recover—too slow for Time-Sensitive Networking (TSN) traffic with sub-millisecond deadlines.

FRER enables zero recovery time via data-plane replication over disjoint paths but introduces bandwidth overhead and possible jitter due to path imbalance. Alternatives include FRER with Scheduled Traffic (FRER-ST), which improves determinism but needs centralized scheduling; Packet Duplication and Elimination Function (P-DEF) in 5G, which is wireless-specific; and reactive options like Multi-Protocol Label Switching - Transport Profile (MPLS-TP) and IEEE 802.1Qca, which offer fast failover at the cost of added complexity. While FRER is ideal for critical flows, selective or hybrid use is advised for high-bandwidth or bursty traffic.

D. Key Observations

While FRER eliminates packet loss and enables seamless failover, it introduces measurable bandwidth overhead. Figure 7 shows link utilization over time for the `Sensing_1` → `SwitchB_1`, resulting in steady utilization (4–5%) until the link failure at $t = 4$ s. Utilization then drops to zero, confirming complete disconnection, and only resumes after recovery at $t = 6$ s. The lack of alternative forwarding results in data loss.

In S1A2, FRER replicates traffic across disjoint paths—`SwitchA_1` → `centralSwitch_1` link consistently reaches 13–15%, while `centralSwitch_1`→ `SwitchB_1` carries 5%. Meanwhile, the direct path remains active with a reduced load (2%). This validates continuous delivery of duplicate frames, confirming the overhead introduced by FRER.

Figure 8 captures the impact of FRER in the S2 scenario, which involves high-bandwidth streaming from `Inspection_2`→ `centralSwitch_3` to near saturation (100%). Even during fault-free operation, these links remain heavily loaded due to constant duplication. This congestion not only increases latency and jitter, as shown earlier, but also reduces available headroom for other traffic.

These findings demonstrate that while FRER offers fault resilience, it comes at the cost of substantial link overhead. This trade-off must be carefully considered—especially for high-rate or bursty streams—where redundancy may amplify congestion risks in shared core segments.

Recommendation: In bandwidth-constrained or converged TSN networks, FRER use must be selective and strategic, prioritizing critical control flows with strict latency and loss constraints. Our evaluation (Section V) shows that indiscriminate replication can increase congestion and jitter, particularly for high-bandwidth streams like inspection video, which experience marginal benefit from FRER unless provisioned with excess capacity. These observations are consistent with prior analyses on redundancy overhead and congestion risk in TSN systems [11], [12]. To mitigate these risks and maintain deterministic performance, we recommend the following:

- Topology-aware stream allocation to prevent redundant paths from overlapping at congestion-prone links.
- Per-link capacity planning that accounts FRER duplication overhead under both nominal and failure conditions (e.g., up to 2× bandwidth increase).
- Flow-aware shaping and policing to bound burst sizes before replication and avoid queue buildup.
- VLAN-based isolation to confine traffic classes and avoid cross-stream contention during failover events.
- Adaptive redundancy levels, where redundancy is enabled dynamically based on flow criticality, system load, or link health indicators (as explored in [10]).

These measures, informed by both our simulation results and recent studies, help ensure that FRER delivers its intended benefits—lossless failover and zero recovery time—without destabilizing the broader TSN network.

VI. Conclusion and Future Work

This paper introduced a scenario-driven simulation framework for evaluating TSN resilience under fault conditions in industrial networks. By modeling synchronized production cells with stream-level redundancy, we assessed the impact of link and cell failures on latency, jitter, packet loss, and recovery time.

Results show that FRER achieves zero-loss failover and uninterrupted operation during faults. However, this reliability comes at the cost of significantly increased bandwidth usage—up to 2–3× in some cases—due to redundant traffic. These findings underscore the need for selective, traffic-aware application of redundancy in bandwidth-constrained environments.

Future work will enhance the framework with dynamic reconfiguration capabilities, enabling networks to autonomously adapt to faults via gate schedule updates, stream rerouting, and flow prioritization. We also plan to simulate multi-fault and transient failure scenarios to assess TSN's robustness under more realistic industrial conditions.

References

[1] M. Seliem, A. Zahran and D. Pesch, "Delay Analysis of TSN Based Industrial Networks With Preemptive Traffic Using Network Calculus," 2023 IFIP Networking Conference (IFIP Networking), Barcelona, Spain, 2023, pp. 1-9, doi: 10.23919/IFIPNetworking57963.2023.10186400.

[2] J. Farkas, L. L. Bello and C. Gunther, "Time-Sensitive Networking Standards," in IEEE Communications Standards Magazine, vol. 2, no. 2, pp. 20-21, JUNE 2018, doi: 10.1109/MCOMSTD.2018.8412457. keywords: Special issues and sections;Sensitivity analysis;Time measurement;Telecommunication traffic;Mission critical systems,

[3] L. Lo Bello and W. Steiner, "A Perspective on IEEE Time-Sensitive Networking for Industrial Communication and Automation Systems," in Proceedings of the IEEE, vol. 107, no. 6, pp. 1094-1120, June 2019, doi: 10.1109/JPROC.2019.2905334.

[4] Libing Deng, Guoqi Xie, Hong Liu, Yunbo Han, Renfa Li, and Keqin Li. 2022. A Survey of Real-Time Ethernet Modeling and Design Methodologies: From AVB to TSN. ACM Comput. Surv. 55, 2, Article 31 (February 2023), 36 pages. https://doi.org/10.1145/3487330

[5] H. -H. Liu et al., "Improving TSN Simulation Accuracy in OMNeT++: A Hardware-Aligned Approach," in IEEE Access, vol. 12, pp. 79937-79956, 2024, doi: 10.1109/ACCESS.2024.3410109.

[6] B. Varga, J. Farkas, F. Fejes, J. Ansari, I. Moldován and M. Máté, "Robustness and Reliability Provided by Deterministic Packet Networks (TSN and DetNet)," in IEEE Transactions on Network and Service Management, vol. 20, no. 3, pp. 2309-2318, Sept. 2023, doi: 10.1109/TNSM.2023.3284590.

[7] Z. Zhou, J. Lee, M. S. Berger, S. Park and Y. Yan, "Simulating TSN traffic scheduling and shaping for future automotive Ethernet," in Journal of Communications and Networks, vol. 23, no. 1, pp. 53-62, Feb. 2021, doi: 10.23919/JCN.2021.000001.

[8] Trifonov, H. and Heffernan, D. (2023), "OPC UA TSN: a next-generation network for Industry 4.0 and IIoT", International Journal of Pervasive Computing and Communications, Vol. 19 No. 3, pp. 386-411. https://doi.org/10.1108/IJPCC-07-2021-0160

[9] G. Li, S. Wang, Y. Huang, T. Huang, Y. Cui and Z. Xiong, "Optimizing Fault-Tolerant Time-Aware Flow Scheduling in TSN-5G Networks," in IEEE Transactions on Mobile Computing, vol. 24, no. 4, pp. 3441-3455, April 2025, doi: 10.1109/TMC.2024.3510604.

[10] S. Sudhakaran et al., "Zero-Delay Roaming for Mobile Robots Enabled by Wireless TSN Redundancy," 2023 IEEE 19th International Conference on Factory Communication Systems (WFCS), Pavia, Italy, 2023, pp. 1-8, doi: 10.1109/WFCS57264.2023.10144124.

[11] M. Jover, M. Barranco, J. Naranjo, J. Proenza and A. Ballesteros, "Characterizing the Tradeoff between Fault Tolerance and Cost of Redundant TSN Networks," 2024 IEEE 29th International Conference on Emerging Technologies and Factory Automation (ETFA), Padova, Italy, 2024, pp. 1-4, doi: 10.1109/ETFA61755.2024.10710768.

[12] G. Cena, S. Scanzio, D. Cavalcanti and V. Frascolla, "Seamless Redundancy for High Reliability Wi-Fi," 2023 IEEE 19th International Conference on Factory Communication Systems (WFCS), Pavia, Italy, 2023, pp. 1-4, doi: 10.1109/WFCS57264.2023.10144228.

[13] M. Seliem, "IN2C: Industrial Network with TSN-enabled Cells," GitHub, 2024. [Online]. Available: https://github.com/MohamedSeliem/IN2C. [Accessed: June. 19, 2025].