FACULTY OF ENGINEERING AND ARCHITECTURE

Time-Sensitive Networking without Wires: Deterministic Communication over Wi-Fi

Pablo Esteban Avila Campos

Doctoral dissertation submitted to obtain the academic degree of
Doctor of Electrical Engineering

Supervisors
Prof. Jeroen Hoebeke, PhD - Prof. Ingrid Moerman, PhD
Department of Information Technology
Faculty of Engineering and Architecture, Ghent University

June 2025

GHENT UNIVERSITY

ISBN 978-94-6355-989-8
NUR 986, 975
Wettelijk depot: D/2025/10.500/49

Members of the Examination Board

Chair

Prof. Joris Degroote, PhD, Ghent University

Other members entitled to vote

Dave Cavalcanti, PhD, Intel Labs, USA
Spilios Giannoulis, PhD, Ghent University
Prof. Johann Marquez-Barja, PhD, Universiteit Antwerpen
Prof. Wouter Tavernier, PhD, Ghent University

Supervisors

Prof. Jeroen Hoebeke, PhD, Ghent University
Prof. Ingrid Moerman, PhD, Ghent University

To my mother, who saw the spark in my curiosity for taking toys apart and always believed in me.

Para mi mamá, que vio la chispa en mi curiosidad por desarmar juguetes y siempre creyó en mí.

Acknowledgement

This dissertation is the realization of a long-awaited dream—one that would not have been possible without the help and support of many people, including my advisors, colleagues, friends, and family.

First and foremost, I would like to express my deepest gratitude to my main advisor, Prof. Jeroen Hoebeke, for his continuous support, patience, and guidance throughout my Ph.D. journey. His insightful feedback, endless ideas, and constant encouragement have been invaluable during these years.

I also want to sincerely thank my advisor, Prof. Ingrid Moerman, who placed her trust in me from the very beginning and welcomed me into IDLab even during the challenging times of COVID-19.

Also, I am deeply grateful to my colleague and friend, Dr. Jetmir Haxhibeqiri, for always being willing to discuss ideas, offer technical help, and provide constructive criticism that continuously pushed my work forward.

I am grateful to the examination board members, Prof. Joris Degroote, Prof. Johann Marquez-Barja, Dr. Spilios Giannoulis, Prof. Wouter Tavernier, and Dr. Dave Cavalcanti, for agreeing to be part of my Ph.D. committee. I appreciate them for taking the time to read my work and for fitting me into their schedules.

My time at the office was an immensely enriching experience thanks to the international team of researchers at IDLab. I extend my heartfelt gratitude to all my colleagues and friends, including Abderraouf Yamani, Adnan Shahid, Andy Van Maele, Ben Van Herbruggen, Colvin Chen, Dries Naudts, Hojjat Navidan, Ihtisham Khalid, Irfan Jabandžić, Jaron Fontaine, Jorg Wieme, Mathias Baert, Merkebu Girmay, Mohammad Cheraghinia, Mostafa Naseri, Özgur Özkaya, Ramyashree Bhat, Robbe Gaeremynck, Stijn Luchie, Vasilis Maglogiannis, Wei Liu, Xianjun Jiao, and many others. Your insights, support, and camaraderie made our workplace not only intellectually stimulating but also genuinely enjoyable. I am truly thankful for the opportunity to have shared this journey with you all.

I would also like to extend my heartfelt thanks to my friends, both in Ecuador and Belgium. Even while pursuing something I am passionate about, being far from home sometimes felt like an emotional rollercoaster. Your support and encouragement helped me stay grounded through the highs and lows, and for that, I am deeply grateful.

I also wish to acknowledge those who, at different times along the way, shared moments of this journey with me. Each encounter, in its own way, contributed to the person I became while writing these pages.

Finally, I owe a special debt of gratitude to my family, my aunts, uncles, cousins, father, and especially my mother Gloria, not only for teaching me that “the best things in life require patience and effort” but also for their unconditional support throughout these years. Their belief in me has been my greatest source of strength.

Ghent, June 2025
Pablo Esteban Avila Campos

Table of Contents

List of Figures

List of Tables

List of Acronyms

0-9

3GPP 3rd Generation Partnership Project
5G Fifth Generation of wireless cellular technology
5GS 5G System

A

AC Access Categories
ACM Access and Mobility Management Function
ADAS Advanced Driver Assistance Systems
AF Application Function
AIML Artificial Intelligence Machine Learning
ANC Ultra Wideband Anchor
ANN Artificial Neural Network
AP Access Point
AR Augmented Reality
AVB Audio Video Bridging

B

BBU Base Band Units
BSS Basic Service Set
BSSID Basic Service Set Identifier

BSTM Basic Service Set Transition Management

C

CBS Credit-Based Shaper
CCA Clear Channel Assessment
CDF Cumulative Density Function
CMI Class Measurement Interval
CNC Centralized Network Controller
CRC Cyclic Redundancy Check
CSMA Carrier Sense Multiple Access
CSMA/CA Carrier Sense Multiple Access Collision Avoidance
CUC Centralized User Controller
CW Contention Window
C-RAN Cloud - Radio Access Network

D

DA Destination Address
DCF Distributed Coordination Function
DIFS DCF Interframe spacing
DL Downlink
DS-TT Device-Side Translation Function
DT Digital Twin

E

ECU Electronic Control Unit

EDCA Enhanced Distributed Channel Access
EIFS Extended Inter-Frame Space
EK WTSN Evaluation Kit
ELBD Early/Late Beacon Detection
ERP Enterprise Resource Planning

F

FCS Frame Check Sequence
FPGA Field Programmable Gate Array
FTM Fine Time Measurement

G

GB Guard Band
GCL Gate Control List
GM Grand Master
gPTP generalized Precision Time Protocol

H

HAM Handover Agent Module
HCM Handover Controller Module
HO Handover

I

IEEE Institute of Electrical and Electronics Engineers

IETF Internet Engineering Task Force

INT In-band Network Telemetry

IT Information Technology

ITU-T International Telecommunication Union Telecommunication Standardization Sector

J

JSD Jensen-Shannon Divergence

K

KPI Key Performance Indicator

L

LAN Local Area Network

LSTM Long Short-Term Memory

M

MAC Medium Access Control

MAE Mean Absolute Error

MCS Modulation and Coding Scheme

MES Manufacturing Execution Systems
ML Machine Learning
MQTT Message Queuing Telemetry Transport
MRP Media Redundancy Protocol
MSE Mean Squared Error
MU Multi-User

N

NAV Network Allocation Vector
NETCONF Network Configuration Protocol
NPCA Non-Primary Channel Access
NUC Next Unit Computing
NW-TT Network-Side Translation Function

O

OFDMA Orthogonal Frequency-Division Multiple Access
OT Operational Technology

P

PCF Policy Control Function
PCP Priority Code Point
PDF Probability Density Function
PER Packet Error Rate
PHY Physical Layer
PI Proportional Integral
PID Proportional–Integral–Derivative

PLC Programmable Logic Controller
PMK Pairwise Master Key
PRACH Physical Random-Access Channel
PTP Precision Time Protocol

Q

QoS Quality of Service

R

REM Radio Environmental Map
RF Radio Frequency
ROS Robot Operating System
RPI Raspberry Pi
RRH Remote Radio Heads
RSSI Received Signal Strength Indicator
RST Residual Service Time
RTSP Rapid Spanning Tree Protocol

S

SA Source Address
SCADA Supervisory Control and Data Acquisition
SDR Software Defined Radio
SMF Session Management Function
SNMP Simple Network Management Protocol
SNR Signal-to-Noise Ratio
STA Station/Client

T

TAG Ultra Wideband Tag/Client
TAS Time-Aware Shaper
TCP Transmission Control Protocol
TDMA Time Division Multiple Access
TFDMA Time-Frequency Division Multiple Access
TS Time Slot
TSF Timestamp Synchronization Function
TSN Time-Sensitive Networking
TU Time Units
TWT Target Wake Time

U

UDP User Datagram Protocol
UL Uplink
URLLC Ultra Reliable Low Latency Communications
USB Universal Serial Bus
UWB Ultra-Wideband

V

VLAN Virtual Local Area Network

W

Wi-Fi Wireless Fidelity

WIA-FA Wireless networks for Industrial Automation-Factory Automation Protocol

WTSN Wireless Time-Sensitive Networking

X

XR Extended Reality

Samenvatting

Stel je een Amazon-magazijn voor tijdens de piekuren: robots verplaatsen goederen van schappen naar verpakkingsstations en verwerken duizenden bestellingen per minuut. Ze gebruiken realtime data en een draadloos netwerk om hun locatie, snelheid en taakstatus te delen, allemaal beheerd door een centraal systeem. Plotseling veroorzaakt een kleine storing in het draadloze netwerk een vertraging in de gegevensoverdracht, waardoor één robot een update mist. Als gevolg hiervan wijkt de robot van zijn pad af en blokkeert de andere robots. Binnen enkele ogenblikken vormt er zich een opstopping, waardoor de operaties stilvallen. Wat een naadloos proces had moeten zijn, wordt een tijdrovend probleem voor het team. Voor industrieën zoals Amazon, waar efficiëntie essentieel is, kunnen deze verstoringen leiden tot kostelijke vertragingen. Mijn onderzoek pakt deze uitdaging aan door draadloze systemen te ontwikkelen die lage vertragingen en betrouwbare communicatie garanderen, waardoor robots, sensoren en werknemers kunnen functioneren op een perfect gesynchroniseerde manier, zelfs in dynamische omgevingen.

Hoewel het scenario in het Amazon-magazijn de directe effecten van draadloze verstoringen benadrukt, is het verre van uniek. Tijdgevoelige communicatie is cruciaal in sectoren zoals produktie, logistiek, gezondheidszorg, automobielindustrie, multimedia en zelfs communicatienetwerken in de ruimte. Echter, huidige draadloze technologieën, waaronder Wi-Fi, 5G en andere, zijn niet ontworpen om de deterministische prestaties te leveren die vereist zijn voor kritieke toepassingen. Er is reeds vooruitgang geboekt in de transformatie van bekabelde systemen tot tijdsgevoelige netwerken - ook wel Time-Sensitive Networking (TSN) genoemd - maar de integratie daarvan in draadloze netwerken is beperkt door uitdagingen zoals signaalinterferentie, lagere capaciteit, variabele linkkwaliteit en onvoorspelbare transmissievertragingen.

Bestaand onderzoek heeft zich voornamelijk gericht op het vergroten van de draadloze capaciteit, maar er blijft een cruciale kloof bestaan: de afwezigheid van betrouwbare oplossingen met gegarandeerde lage vertragingen die TSN effectief integreren in dynamische draadloze omgevingen. De huidige draadloze systemen voldoen niet aan de strikte eisen inzake timing van industriële systemen, waardoor ze onbetrouwbaar en ongeschikt zijn voor gebruik. De uitdaging gaat verder dan alleen het verhogen van de doorvoersnelheid; het gaat erom te garanderen dat gegevens altijd op tijd aankomen, ondanks fluctuerende draadloze kanaalomstandigheden.

Het dichten van deze kloof is meer dan alleen een technische stap vooruit; het is een game changer voor industrieën die afhankelijk zijn van besluitvorming in ware tijd. Door draadloze netwerken net zo betrouwbaar te maken als bekabelde

netwerken, kunnen we het volledige potentieel van slimme productie, autonome systemen en grootschalige industriële automatisering realiseren. Op grotere schaal zou dit onderzoek slimme produktie, gezondheidszorgsystemen en zowat elke sector waar beslissingen in een oogwenk dienen genomen te worden. kunnen hervormen. Deze systemen zouden dan niet alleen efficiënter, maar ook veiliger en veerkrachtiger worden.

Om deze kloof aan te pakken, is het in dit werk mijn doel om te onderzoeken hoe Wi-Fi-gebaseerde draadloze netwerken kunnen worden aangepast en geoptimaliseerd om betrouwbare communicatie met gegarandeerde lage vertragingen te leveren voor tijdgevoelige toepassingen. Specifiek richt mijn onderzoek zich op:

- Het onderzoeken van de impact van verkeersplanning op tijdgevoelige verkeersstromen in omgevingen met variabele linkkwaliteit en interferentie.
- Het ontwikkelen van efficiënte associatiemechanismen om nieuwe apparaten snel en naadloos te integreren, zonder de lopende operaties te verstoren.
- Het verkennen van mobiliteitsoplossingen om ervoor te zorgen dat tijdgevoelig verkeer onaangetast blijft tijdens overgangen tussen toegangspunten in industriële omgevingen.
- Het voorstellen van een beheerkader dat realtime dataverzameling combineert met voorspellende modelleringstechnieken, zoals digitale tweelingen, om het netwerk proactief te beheren en deterministische prestaties te waarborgen.

Door deze doelstellingen aan te pakken, streeft mijn onderzoek ernaar de eerste stappen te zetten om van draadloze TSN een levensvatbaar alternatief voor bekabelde netwerken te maken voor realtime toepassingen. Om de onderzoeksvraag te verkennen en de gestelde doelen te bereiken, heb ik gebruikgemaakt van een combinatie van experimentele, theoretische en op simulatie gebaseerde methodologieën.

Experimentele validatie was een belangrijk onderdeel van mijn onderzoek. Met behulp van openwifi[1] hardware met verbeterde TSN-functies, zoals een mechanisme om pakkettransmissies in de tijd in te plannen en kloksynchronisatie met nauwkeurigheden in de orde van microseconden, en flexibiliteit om elke laag van de Wi-Fi-stack aan te passen, kon ik ieder van de uitdagingen die in dit werk werd aangepakt ook valideren middels een labopstelling. Deze experimentele opstellingen omvatten zowel statische als dynamische mobiliteitsscenario's om de uitdagingen in industriële omgevingen te repliceren. De opstellingen werden gebruikt om de prestaties van verschillende netwerkconfiguraties te evalueren en de praktische effectiviteit van de voorgestelde oplossingen te bevestigen.

Deze experimentele validatie was gebaseerd op theoretische modellen die ontworpen zijn om het gedrag van draadloze TSN voor verschillende doeleinden te begrijpen en te voorspellen. Bijvoorbeeld, bij het ontwikkelen van de modellen om het verkeer in te plannen, speelde het creëren van een reeks logische beperkingen, gecombineerd met optimalisatie-algoritmen zoals Satisfiability Modulo Theories

[1] https://github.com/open-sdr/openwifi

(SMT), een cruciale rol om, de optimale end-to-end transmissieconfiguraties te bepalen van de gatesystemen die de tijden van de datatransmissies controleren. Theoretische modellen behandelden ook de synchronisatie van tijdgevoelige apparaten in een draadloos netwerk. Markov-modellen werden gebruikt om de verandering van toegangspunt in draadloze netwerken te modelleren, rekening houdend met factoren zoals signaalsterkte en apparaatmobiliteit. Daarnaast werden digitale tweelingenmodellen, gebaseerd op feedbackcontroltheorie en voorspellingen gemaakt met behulp van machinaal leren, ontworpen om het netwerk proactief te beheren en lage vertragingen en jitter van tijdskritische communicatie te waarborgen.

Een van de belangrijkste bevindingen van dit werk is een effectief planningsmechanisme voor zowel bekabelde als draadloze Time-Sensitive Networks. Deze ontwikkeling slaagde erin de eisen van tijdgevoelig verkeer in balans te brengen met de inherente variabiliteit van draadloze kanalen, rekening houdend met factoren zoals gedeelde en individuele transmissietijdssloten, verwerkingsvertragingen en fluctuerende transmissiesnelheden. Het planningsmechanisme toonde aan dat het mogelijk is een betrouwbaar communicatieframework op te zetten voor kritieke waretijdstoepassingen.

Ik introduceerde een nieuwe pre-synchronisatie- en pre-planningsmethode voor het opstarten van apparaten in WTSN. Door beacon-protocollen te wijzigen en tijdsynchronisatiegegevens in de broadcasts van toegangspunten te incorporeren, verkortte ik de tijd en complexiteit die nodig is voor nieuwe apparaten om zich bij het netwerk aan te sluiten. Deze oplossing loste een kritiek probleem op in draadloze TSN, waar traditionele Wi-Fi-associatieprocessen tijdgevoelig verkeer konden verstoren.

Het onderzoek ontdekte ook dat modellen gebaseeerd op machinaal leren, vooral kunstmatige neurale netwerken, de prestaties in mobiele omgevingen aanzienlijk konden verbeteren. Door gebruik te maken van data in ware tijd, zoals signaalsterkte en locatie, voorspelden de modellen optimale momenten om van toegangspunt te veranderen, waardoor verstoringen van kritisch verkeer werden geminimaliseerd. Deze innovatie werd gedemonstreerd in zowel eendimensionale als tweedimensionale mobiliteitsscenario's, waar het vertragingen en pakketverlies verminderde.

Verder toonde ik het potentieel aan van Digital Twin (DT)-gestuurd netwerkbeheer, waarbij realtime gegevens over de aankomst van datapakketjes uit het netwerk werden gebruikt om verkeerspatronen te voorspellen en planningsbeslissingen proactief aan te passen. Deze benadering verminderde de vertragingen over de draadloze verbinding met maximaal 96% op het 90e percentiel in vergelijking met traditionele niet-beheerde systemen. De integratie van Digitale Tweelingen in WTSN-beheer bood een robuust framework om beslissingen in ware tijd te nemen, dat zich bovendien kon aanpassen aan veranderende omstandigheden van de verkeerspatronen in het netwerk.

Hoewel het onderzoek met succes oplossingen heeft ontwikkeld en getest voor belangrijke uitdagingen in draadloze Time-Sensitive Networks, zijn er nog steeds gebieden die verdere exploratie vereisen. Een beperking is de schaalbaarheid van de voorgestelde oplossingen naar grotere netwerkomgevingen. Hoewel de metho-

den effectief bleken in kleinschalige netwerken, kunnen complexere topologieën met honderden apparaten nieuwe uitdagingen met zich meebrengen op het gebied van verkeersopstoppingen, synchronisatie, verwerking en efficiëntie om van toegangspunt te veranderen bij mobiliteit. Toekomstig onderzoek zou niet alleen communicatie, maar ook dataverwerking kunnen onderzoeken, als een reeks onderling afhankelijke gebeurtenissen die gesynchroniseerd en samen gepland moeten worden.

Een ander gebied voor toekomstig werk is de integratie van opkomende draadloze technologieën, zoals 5G en andere. Deze netwerken bieden hogere doorvoersnelheden en lagere vertragingen, zelfs voor adoptie in private netwerken, maar de *bridge*-benadering van TSN door deze technologieën kan nieuwe complexiteiten introduceren bij het handhaven van het beloofde determinisme. Het onderzoeken hoe de voorgestelde oplossingen kunnen worden aangepast of geïntegreerd met deze toekomstige draadloze standaarden zal cruciaal zijn voor het bevorderen van industriële draadloze netwerken.

Daarnaast, hoewel de Digital Twin-benadering significant potentieel toonde in het proactief beheren van netwerkprestaties, is er ruimte voor verdere verfijning. Toekomstige studies zouden gedistribueerde beheertopologieën en meer geavanceerde algoritmen voor machinaal leren, zoals transfer- of hybride leren die verklaarbaarheid introduceren, kunnen verkennen om de nauwkeurigheid, voorspelbaarheid en responsiviteit te verbeteren. Dergelijke vooruitgang zou het systeem toelaten zich nog sneller aan te passen aan fluctuerende netwerk- en verkeersomstandigheden.

Het hier gepresenteerde onderzoek legt een solide basis voor de toekomst van draadloze communicatie in tijdgevoelige toepassingen. Naarmate industrieën steeds meer slimme produktie, autonome systemen en realtime datagedreven operaties adopteren, zal de vraag naar betrouwbare, draadloze netwerken met gegarandeerde lage vertragingen blijven stijgen. Door kritieke uitdagingen zoals planning, associatie, mobiliteit en netwerkbeheer in draadloze Time-Sensitive Networks aan te pakken, zorgt mijn werk ervoor dat realtime communicatie in dynamische, industriële omgevingen zowel betrouwbaar als voorspelbaar is.

Terugkomend op het scenario in het Amazon-magazijn, waar een lichte verstoring in het draadloze netwerk een kostbare vertraging veroorzaakte in de operaties van autonome robots, biedt dit onderzoek de eerste oplossingen. Net zoals een naadloze overname een robot in staat stelt zonder onderbreking doorheen de magazijnvloer te navigeren, legt dit werk de basis voor betrouwbare draadloze communicatie met gegarandeerde lage vertragingen, die ervoor zorgt dat industrieën soepel, efficiënt en zonder onderbreking draaien. Deze scriptie helpt een potentieel knelpunt om te vormen tot een krachtig hulpmiddel, waardoor industrieën wereldwijd volledig op draadloze netwerken in tijdgevoelige toepassingen kunnen vertrouwen.

Summary

Imagine an Amazon warehouse during peak hours: robots move goods from shelves to packing stations, processing thousands of orders every minute. They use real-time data and a wireless network to share their location, speed, and task status, all managed by a central system. Suddenly, a small hiccup in the wireless network causes a delay in data transmission, causing one robot to miss an update. As a result, it deviates from its path, blocking others. Within moments, a bottleneck forms, stopping operations. What should have been a seamless process becomes a time-consuming issue for the team. For industries like Amazon warehouse, where efficiency is essential, these disruptions can lead to costly delays. My research addresses this challenge by developing wireless systems that ensure low-latency, and reliable communication, allowing robots, sensors, and workers to function in perfect synchronization, even in dynamic environments.

While the scenario in the Amazon warehouse highlights the immediate effects of wireless disruptions, it is far from being unique. Time-sensitive communication is crucial across industries such as manufacturing, logistics, healthcare, automotive, multimedia, and even deep space networks. However, current wireless technologies, including Wi-Fi, 5G, and others, were not designed to deliver the deterministic performance required for critical applications. While advancements in Time-Sensitive Networking (TSN) are transforming wired systems, their integration into wireless networks has been limited by challenges such as signal interference, lower capacity, variable link quality, and unpredictable transmission delays.

Existing research has primarily focused on enhancing wireless capacity, yet a critical gap remains: the absence of reliable, low-latency solutions that effectively integrate TSN into dynamic wireless environments. Current wireless systems fail to meet the strict timing demands of industrial systems, making them unreliable and unsuitable for use. The challenge extends beyond simply boosting network speed, it is about guaranteeing that data arrives on time, every time, despite fluctuating wireless channel conditions.

Closing this gap is not just a technical advancement; it is a game changer for industries that rely on real-time decision-making. By making wireless networks as reliable as wired ones, we can fully realize the potential of smart manufacturing, autonomous systems, and large-scale industrial automation. On a broader scale, this research could reshape how we approach smart manufacturing, healthcare systems, and any sector where split-second decisions are crucial, making these systems not only more efficient but also safer and more resilient.

To address this gap, in this work, I aim to explore how to adapt and optimize

Wi-Fi-based wireless networks to deliver reliable, low-latency communication for time-sensitive applications. Specifically, my research focuses on:

- Investigating the impact of traffic scheduling on time-sensitive traffic flows in environments with variable link quality and interference.
- Developing efficient association mechanisms to integrate new wireless devices quickly and seamlessly, without disrupting ongoing time sensitive operations.
- Exploring mobility solutions to ensure that time-sensitive traffic can be maintained during handovers between access points in industrial environments.
- Proposing a management framework that combines real-time data capture with predictive modeling techniques, such as digital twins, to proactively manage the network and ensure deterministic performance.

By addressing these objectives, my research seeks to take the first steps in making wireless TSN a viable alternative to wired networks for real-time applications. To explore the research question and achieve the stated goals, I utilized a combination of experimental, theoretical, and simulation-based methodologies.

Experimental validation was a key component of my research. Using openwifi[2] hardware with enhanced TSN features, such as a packet transmission gating system and microsecond-accurate time synchronization, along with the flexibility to adjust every layer of the Wi-Fi stack, allowed me to provide real-world validation for each challenge addressed in this dissertation. These experimental setups included both static and dynamic mobility scenarios to replicate the challenges faced in industrial environments. The setups were used to evaluate the performance of different network configurations and confirm the practical effectiveness of the proposed solutions.

This experimental validation was grounded in theoretical models designed to understand and predict the behavior of wireless TSN for various purposes. For example, in Traffic Scheduling Models, the creation of a set of logical constraints, combined with optimization algorithms such as Satisfiability Modulo Theories (SMT), played a crucial role in determining optimal end-to-end transmission configurations for the gating systems. Theoretical models also addressed the synchronization of time-sensitive devices in a wireless network. Markov models were employed to represent the handover process in wireless networks, taking into account factors like signal strength and device mobility. Additionally, digital twin models, grounded in feedback control theory and machine learning-based predictions, were designed to proactively manage network performance and ensure low-latency, and low-jitter real-time communication.

Among the key findings of this work is an effective scheduling mechanism for both wired and wireless Time-Sensitive Networks. This development successfully

[2] https://github.com/open-sdr/openwifi

balanced the demands of time-sensitive traffic with the inherent variability of wireless channels, taking into account factors such as shared and dedicated transmission time slots, processing delays, and fluctuating transmission rates. The scheduling mechanism demonstrated that it is possible to establish a reliable communication framework for critical real-time applications.

I introduced a novel pre-synchronization and pre-scheduling method for bootstrapping devices in WTSN. By modifying beacon protocols and embedding time synchronization data into access point beacons, I reduced the time and complexity required for new devices to join the network. This solution addressed a critical gap in wireless TSN, where traditional Wi-Fi association processes could disrupt time-sensitive traffic.

The research also found that machine learning models, especially artificial neural networks, could significantly enhance handover performance in mobile environments. By utilizing real-time data, such as signal strength and localization, the models predicted optimal handover moments, minimizing disruption to critical traffic. This innovation was demonstrated in both one-dimensional and two-dimensional mobility scenarios, where it reduced handover delays and packet loss.

Furthermore, I demonstrated the potential of Digital Twin (DT)-enabled network management, where real-time traffic arrival data from the network was used to predict traffic patterns and proactively adjust scheduling decisions. This approach reduced link latency by up to 96% at the 90th percentile compared to traditional non-managed systems, by leveraging predictive models to optimize network behavior. The integration of Digital Twins into WTSN management provided a robust framework for real-time decision-making, capable of adapting to changing conditions in dynamic traffic generation patterns.

While the research successfully developed and tested solutions for key challenges in wireless Time-Sensitive Networks, there are still areas that require further exploration. One limitation is the scalability of the proposed solutions in large network environments. Although the methods proved effective in small-scale networks, more complex topologies with hundreds of devices may introduce new challenges related to traffic congestion, synchronization, processing, and handover efficiency. Future research could investigate not only communication but also processing, as a series of interdependent events that must be synchronized and scheduled together.

Another area for future work is the integration of emerging wireless technologies, such as 5G and beyond. These networks offer higher throughput and lower latency, even for private development, but the *bridge* approach of TSN towards these technologies may introduce new complexities in maintaining deterministic behavior promises. Investigating how the proposed solutions can be adapted or integrated with these next-generation wireless standards will be crucial for advancing industrial wireless networking.

Additionally, while the Digital Twin approach demonstrated significant promise in proactively managing network performance, there is potential for further refinement. Future studies could explore distributed management topologies and more advanced machine learning algorithms, such as transfer or hybrid learning intro-

ducing explainability, to improve prediction accuracy and responsiveness. These advancements would enable the system to adapt even more swiftly to fluctuating network and traffic conditions.

The research presented here establishes a solid foundation for the future of wireless communication in time-sensitive applications. As industries increasingly adopt smart manufacturing, autonomous systems, and real-time data-driven operations, the demand for reliable, low-latency wireless networks will continue to rise. By addressing critical challenges such as scheduling, association, mobility, and network management in wireless Time-Sensitive Networks, my work ensures that real-time communication in dynamic, industrial environments is both reliable and predictable.

Returning to the scenario in the Amazon warehouse, where a slight disruption in the wireless network caused a costly delay in the operation of autonomous robots, this research provides the first solutions. Just as a seamless handover enables a robot to navigate the warehouse floor without disruption, this work lays the foundation for reliable, low-latency wireless communication that ensures industries run smoothly, efficiently, and without interruption. This thesis helps transform a potential bottleneck into a powerful asset, enabling industries worldwide to fully trust wireless networks for time-sensitive applications.

1 Introduction

"Science is still only a candle glimmering in a great pitch-dark cavern."

–Mario Vargas Llosa

This chapter situates the conducted research work. It begins with a brief historical overview of how industrial automation evolved and led to the development of TSN. Next, it presents key applications and core components. Then, it describes how the shift from wired to wireless TSN is becoming a reality. After that, it summarizes the main challenges and contributions before outlining the dissertation's structure. Finally, it provides an overview of the publications and patents authored during the research period.

1.1 The Evolution of Industrial Networks: From 3.0 to 4.0 and further

Historical Context

If you walked through a factory floor a few decades ago, you would see production lines full of rows of machines, controlled by technicians carefully monitoring blinking lights and connecting endless cables. This period marked the early phase of digitalization, also called Industry 3.0. During that time, mechanical processes started to rely on electronics and computer systems. In those days, engineers sought a systematic way to coordinate production, and thus the pyramidal approach was born.

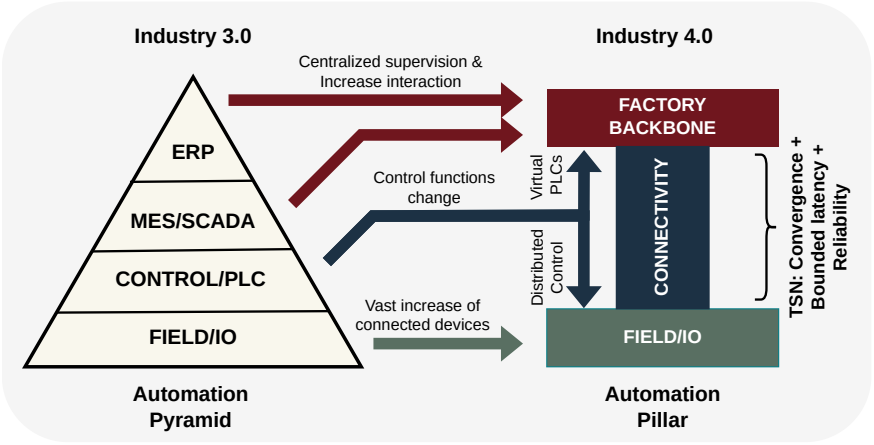


Figure 1.1: Moving from the automation pyramid to the automation pillar

As shown in Figure 1.1, the so-called automation pyramid placed production systems, typically managed by programmable logic controllers (PLCs), at its base. Above these systems, supervisory control and data acquisition (SCADA) and Manufacturing Execution Systems (MES) provided layered oversight. At the top, enterprise resource planning (ERP) software handled broader resource management and business operations [1]. This structure brought order to the rising complexity of factories. However, it was originally designed for the more static, less data-driven environment of Industry 3.0.

With the rise of Industry 4.0, manufacturing transformed yet again. More sensors, tighter coordination, and an insatiable hunger for data demanded flexible, scalable, and real-time communication throughout the entire production environment. The rigid, layered structure of the automation pyramid could not handle these new demands. Engineers and managers realized they needed a system less hierarchical, a pillar-like design that would enable seamless communication from the sensor floor all the way up to the enterprise level.

Hence, the traditional controller layer started to change: safety functions and some control algorithms were pushed down to smart field devices, while the "brains" of production, virtual PLCs, migrated upward to a centralized factory backbone. This new "pillar" requires a connectivity layer that is not just fast, but deterministic: it must offer bounded latency, high reliability, and the ability to converge time-critical and best-effort traffic without interference. Naturally, Ethernet was the main candidate to fill this position [2].

When Ethernet was introduced nearly 40 years ago, few could have predicted the remarkable success it would achieve. Initially designed as a standardized solution

for local networks, it has significantly advanced in both technical capabilities and applications. Starting from local computer networks, Ethernet has become the de facto standard for field buses, widely used in protocols such as Profinet, Ethernet/IP, EtherCAT, and more. While these proprietary solutions have addressed one of Ethernet's most significant limitations, the absence of real-time communication support, they ended up fragmenting the market, creating a tangled web of non-interoperable solutions [3].

Surprisingly, the spark for the next evolution of Ethernet did not come from factory floors, but from the realm of professional multimedia. In 2007, high-end audio and video productions relied heavily on a chaotic nest of analog wiring. Early digital approaches like Audio over Ethernet and Audio over IP eased some frustration but lacked true interoperability and rigorous Quality of Service (QoS) guarantees [4]. In response, the Audio Video Bridging (AVB) Task Group was created to standardize media transmission over Ethernet, starting in a new era of deterministic performance and potentially lighting the path toward a unified connectivity layer for modern factories [5].

The AVB group introduced several key standards, such as IEEE 802.1AS [6], which improved network time synchronization, IEEE 802.1Qat for stream Reservation, and IEEE 802.1Qav to control forwarding and queuing for time-sensitive flows. Finally, with rising connectivity demands in both professional multimedia and a broader Industry 4.0, the AVB group evolved into the Time-Sensitive Networking (TSN) Task Group [1], using AVB standards as the basis for TSN [7].

Key Use Cases: Real-world applications highlighting the need for TSN

With the rise of technologies like extended reality (XR), which merges the virtual and real worlds, the concept of motion-to-photon latency has become critical. This latency measures the time it takes for user movements to be reflected on a display, such as an augmented reality (AR) headset. If motion-to-photon latency exceeds 20 milliseconds, the virtual environment feels less immersive [8]. Prolonged exposure to high latency can even cause motion sickness and disorientation [9].

Consider a scenario where multiple AR headsets connect to a central computer with ample processing power to deliver a seamless user experience. In such cases, the network connecting the headsets to the computer must be highly reliable, allow convergence of traffic with different priorities, and keep communication latency below the motion-to-photon threshold.

Currently, these requirements can only be fulfilled by TSN. However, TSN was not created only for multimedia traffic. The TSN task group developed several profiles that illustrate TSN's applications and help to focus on specific features for

[1] http://www.ieee802.org/1/tsn

better interoperability and easier deployment [10]:

- IEEE Std 802.1CM TSN for Fronthaul. In next-generation mobile networks, separating radio hardware from centralized baseband processing (as in Cloud-RAN) can significantly improve scalability and efficiency. However, transmitting large volumes of baseband data poses strict demands on the fronthaul network. TSN helps meet these performance requirements by providing reliable, low-latency connectivity in high-bandwidth mobile environments. [11].
- IEEE P802.1DP TSN Profile for Aerospace Onboard Ethernet Communications. Modern aircraft rely on a wide range of communication systems, many of which are decades old. This diversity increases complexity, weight, and maintenance costs. By transitioning to a unified, TSN-based Ethernet network, the aerospace industry can streamline onboard communications, reduce infrastructure overhead, and improve overall efficiency, even within long product life cycles. [12].
- IEEE P802.1DG TSN Profile for Automotive In-Vehicle Ethernet Communications. As vehicles evolve into complex, software-defined platforms, a zonal architecture—where different vehicle areas handle specific electronic functions—becomes crucial. TSN enables reliable, time-critical data transfers among these zones, ensuring that systems such as infotainment, ADAS, and powertrain can seamlessly operate together in real-time [13].
- IEC/IEEE 60802 TSN Profile for Industrial Automation addresses a wide range of industrial scenarios. In industrial environments that involve robotics, automated guided vehicles, and diverse field-bus standards, consistency and predictability in data exchange are vital. TSN provides a unified solution that simplifies communication across multiple industrial protocols, making it one of the most impactful application domains for time-sensitive, high-reliability networking. [14].

Figure 1.2 highlights more specific TSN use cases in industrial automation, audio, video, and AR/VR with wearable and spatial interaction. It also shows their latency and bandwidth requirements. Note that some of these scenarios not only require TSN but also benefit from wireless mobility.

1.2 Core Components of TSN

TSN seeks to enhance traditional networks by introducing features that ensure deterministic behavior. The IEEE TSN Task Group defines these features in four categories: synchronization, reliability, latency, and resource management. Figure

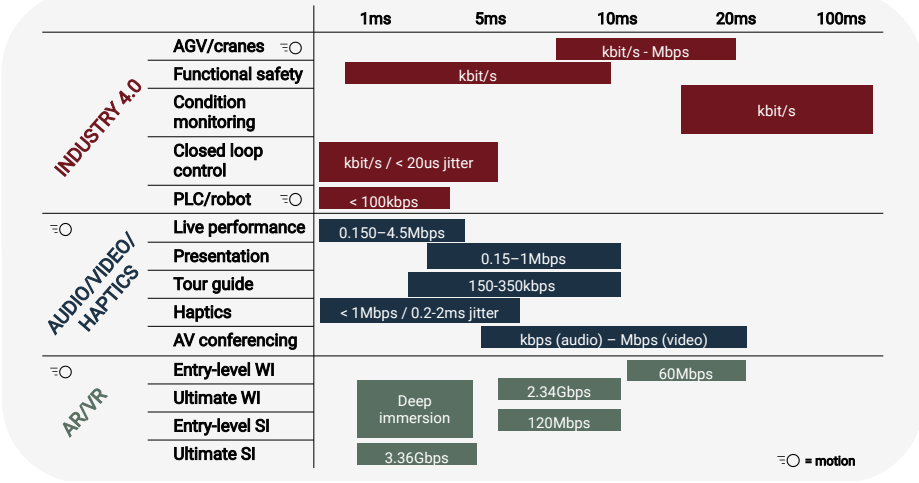


Figure 1.2: Requirements for industrial and professional multimedia applications

1.3 shows each category which includes multiple standards, some of which are still under development. With this broad picture of the TSN framework, the next subsections will delve into the specific standards related to my research.

Prioritization: Handling critical data with appropriate urgency

A TSN network consists of two types of components: end points and bridges or switches. The end points act as frame generators (talkers) or as sinks (listeners). Figure 1.4 shows how they link through one or more bridges. These bridges manage frame forwarding between the end points. They perform two main tasks. First, they switch frames by choosing the correct output port. Second, they shape traffic by selecting which frame goes first when multiple frames are ready, then serializing their transmission.

Each bridge uses information in the Ethernet frame to manage switching and shaping. For switching, the media access control (MAC) destination and the virtual local area network (VLAN) identifier are used to choose the outgoing ports for the frame. For traffic shaping, it relies on the priority code point (PCP) in the VLAN tag. This three-bit PCP field indicates the Ethernet frame's priority. By assigning values from 0 to 7, Ethernet switches can define eight levels of priorities. The relation between each incoming port and outgoing port, as well as the specific traffic-shaping mechanism, is set locally but managed by the configuration model, which will be described later in this section.

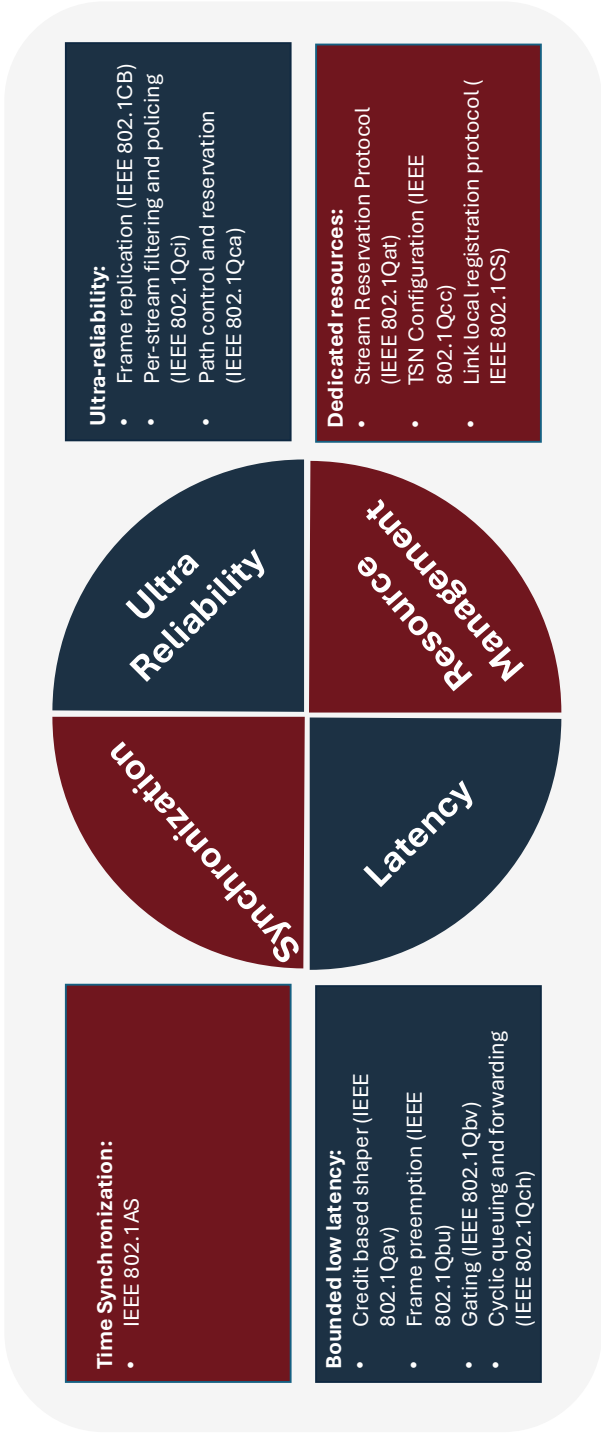


Figure 1.3: TSN protocols summary

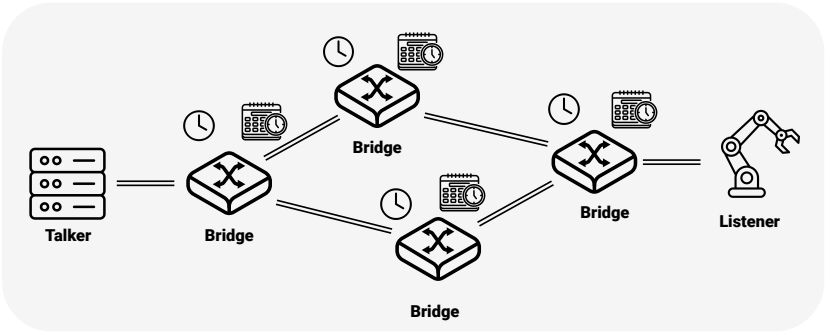


Figure 1.4: An example of a physical topology of a TSN

1.2.1 Time Synchronization: Ensuring precise timing across the network

As shown in Figure 1.4, all the bridges require time synchronization. The goal of time synchronization is to align local clocks in bridges so that, at any moment, two functioning clocks at two different switches display nearly identical times. The clock synchronization protocol used in TSN follows the IEEE 802.1AS standard, which closely aligns with IEEE 1588. IEEE 802.1AS defines a traditional master-slave clock synchronization system where one component, called the master clock, serves as the reference for synchronizing all other clocks. The master clock, known as the grandmaster, forms the root of a synchronization spanning tree connecting all local clocks [6, 15].

In summary, IEEE 802.1AS defines protocols for:

- Selecting a *grandmaster* from a group of eligible masters.
- Periodically synchronizing clocks to the *grandmaster*.
- Measuring forwarding delays in bridges.
- Measuring communication delays across links.

For grandmaster selection, IEEE 802.1AS employs the Best Master Clock Algorithm (BMCA). Masters capable of serving as grandmasters announce their status through announcement messages. Receiving slaves compare these messages and select the best candidate using a standardized evaluation method. The master that sent the top-ranked announce message becomes the grandmaster and the root of the synchronization tree.

Once selected, the grandmaster regularly distributes its local time (referred to as the *preciseOriginTimestamp*) through the network using two message types:

Sync and *Follow_up*. The *Sync* message is time-critical, and its transmission and reception are precisely timestamped at each component (including the grandmaster and bridges). The transmission and forwarding delays of the *Sync* message are conveyed in the *Follow_up* message. As the *Follow_up* message travels down the synchronization tree, each component updates its payload with the forwarding delay, calculated as the time difference between receiving and transmitting the previous *Sync* message, using the *correctionField*.

In addition to forwarding delays, the network periodically measures communication delays across links through the peer delay mechanism. This process occurs on each link individually and involves the following steps:

1. A slave sends a $Pdelay\_Req$ message to the grandmaster, storing the transmission time as t1.
2. The grandmaster timestamps its reception of the $Pdelay\_Req$ message as t2.
3. At time t3, the grandmaster replies with a $Pdelay\_Resp$ message containing t2.
4. The slave records the time t4 when it receives the $Pdelay\_Resp$ message.
5. The grandmaster sends a $Pdelay\_Resp\_Follow\_Up$ message to inform the slave of time t3.
6. The slave calculates the link delay using the formula: $\frac{(t4-t1)-(t3-t2)}{2}$.

Intermediate bridges in synchronization links can operate in two modes: transparent clock (TC) or boundary clock (BC). In TC mode, the bridge does not take part in the master-slave hierarchy. It simply forwards synchronization messages while updating them with its own residence time to account for forwarding delay. In BC mode, the bridge acts as a slave to a grandmaster on one port, synchronizes its own clock, and then functions as a grandmaster to other slaves connected through its ports.

While IEEE 802.1AS delivers excellent synchronization quality suitable for various applications, such as automotive and industrial applications, the TSN task group is developing enhancements to improve reliability. One proposed mechanism involves using redundant grandmasters in a hot or warm standby configuration. Instead of relying on a single grandmaster, at least two operate simultaneously: a primary and a secondary. In normal conditions, all components, including the secondary grandmaster, synchronize with the primary. If the primary fails, the secondary seamlessly takes over synchronization.

This fault-tolerant approach, though simple, significantly improves reliability and broadens TSN applicability. However, safety-critical systems often demand advanced fault-tolerant synchronization methods. For TSN to meet such needs,

standardized interfaces for highly dependable clock synchronization protocols are essential.

1.2.2 Traffic Shaping: Managing data flows for optimal performance

Typically, multiple applications or traffic flows must share a single network. Often, messages from different sources become ready for forwarding within a specific bridge to the same communication link. This bridge must then decide which message to transmit next. Additionally, many industries such as the vehicular, are moving toward converged networks, where applications that were previously isolated are now integrated into the same network. As systems become increasingly integrated, determining the transmission order of messages grows more complex. When the network must also guarantee end-to-end delays for critical messages, the serialization of messages within bridges becomes even more challenging [1].

To address this issue, TSN has introduced several traffic-shaping mechanisms, including the credit-based shaper and the time-aware shaper.

Credit-Based Shaper

The credit-based shaper (CBS) starts by defining two traffic classes A and B. Ethernet frames are then classified based on their Priority Code Point (PCP) in the VLAN tag. Additionally, the CBS specifies the Class Measurement Interval (CMI), which limits the number of frames the talker may send. In practice, this means that a generator typically sends one Ethernet frame of configurable size per CMI. The CMI for class A messages is standardized at 125 μs, while for class B messages, it is set to 250 μs. The execution of CMIs is not synchronized between generators and bridges. As a result, class A frames from different generators can still accumulate in the outgoing port of a bridge. An example of how a bridge shapes class A traffic is illustrated in Figure 1.5 [16–18].

Figure 1.5 shows the operation of a single outgoing port in a bridge. Specifically, it depicts two outgoing queues for the port: one queue handles class A frames (A1–A4), while the other serves other traffic (M1–M2). Since class A frames have higher priority, a bridge using a strict priority-driven traffic-shaping policy would send all class A frames first, followed by the remaining traffic. This approach, however, could cause excessive delays for lower-priority frames. To address this, the CBS ensures that the bridge selects a class A frame only if its current credit is nonnegative. Credit is a local variable that evolves over time according to the following rules:

- If no class A frame is in the queue, the system sets credit to zero.

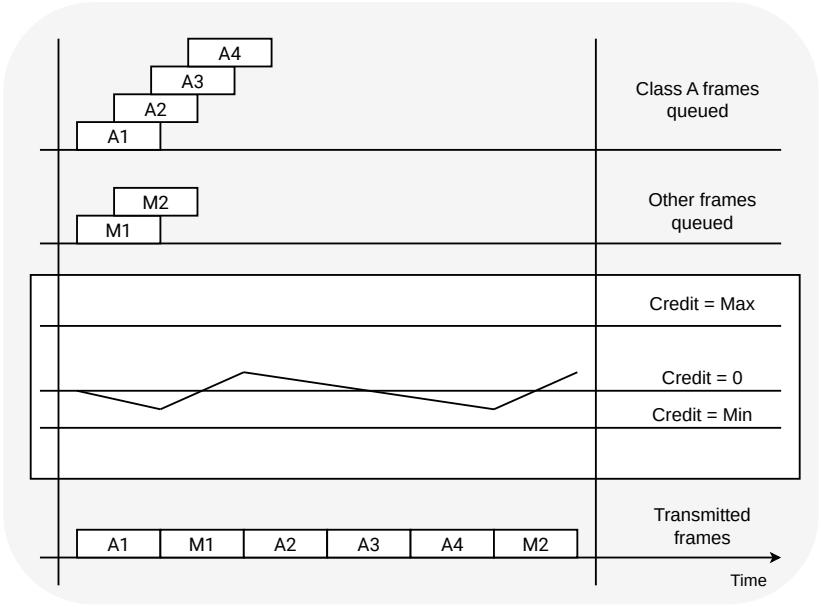


Figure 1.5: Example of Credit-Based Shaper Functionality

- If a class A frame is transmitted, the system decreases credit over time using the configurable rate *sendSlope*.

- If at least one class A frame is in the queue but not sent (because credit is negative or another frame is in transit), the system increases credit over time. It does so using the configurable rate *idleSlope*.

The scenario in Figure 1.5 begins with a credit of zero. Class A frames and other frames then become ready for transmission. As shown, the system first transmits frame A1, which drives the credit below zero. Thus, it sends M1 next. Because the *sendSlope* is usually lower than the *idleSlope*, the system then transmits several class A frames (A2–A4). As a result, the credit again becomes negative. Finally, the system ends the sequence with a non-class A frame (M2). The CBS handles class B frames in a manner similar to class A frames. However, it maintains a separate local variable to keep track of the AVB class B credit.

The original goal of the credit-based shaper was to achieve a 2 ms latency for highly time-critical messages across a maximum of six bridges. Although it has been shown that this goal may not always be met in certain network topologies, the CBS still proves valuable in multiple applications.

Time-Aware Shaper

The time-aware shaper (TAS) is one of the most important mechanisms for coordinating communication in TSN. It is defined in the IEEE 802.1Qbv standard and introduces time-based scheduling to ensure that critical messages arrive at predefined intervals. As with CBS, it classifies Ethernet frames using the PCP, which carries their priority. However, unlike CBS, TAS aims to reduce latency and especially jitter for high-priority traffic by keeping time resources continually available for it [19]. It achieves this through the following approach:

- TAS splits time into repeating segments of equal length, known as cycles.
- In each cycle, the scheduler allocates specific time slots for frames with different priority levels.
- The TSN manager configures the cycle time and the time slots in each cycle. This configuration must meet the requirements of the applications using the network.
- The TSN manager can also assign different priority levels to each time slot.

In Figure 1.6, TAS uses a gating system to generate the cycle and define the transmission time slots for each traffic class. It then sends each class's frames during its assigned time slot. One can grant higher priority to a traffic class by adjusting the size of its time slot, overlapping, or by assigning more than one time slot to that class within a cycle. Normally, the best effort time slot is shared among best-effort traffic flows or queues.

The requirement of synchronized clocks in switches and end devices plays a key role in TAS. There is only one transmission medium, so different traffic priorities must access it at separate times. This setup ensures that time-critical traffic finds an empty medium exactly when it must send data. At that moment, the switches open their time-aware gates together for the high-priority frames, preventing interruptions along the path. Meanwhile, the traffic generators must know when the gates open, so they can send high-priority frames at just the right time [1].

Guard Bands: Reserving time slots to prevent data collision

Transmission queues often exhibit unpredictability, especially when the generation of frames is not time-synchronized with the TSN. Sometimes, a frame is sent late in its time slot or is too large for that slot. As a result, the frame overlaps into the next time slot, which might be allocated to higher-priority traffic. This situation interrupts the higher-priority flow. To avoid this problem, the TAS proposes a guard band that blocks the oversized frame from entering the next slot. Consequently, the frame is deferred to its next available time slot. Guard bands are vital for protecting

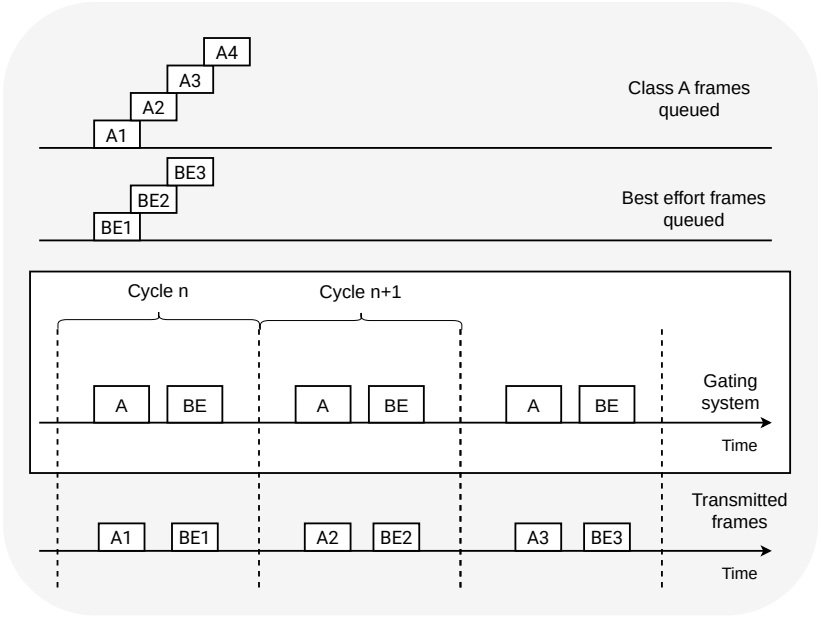


Figure 1.6: Example of Time-Aware Shaper Functionality

latency guarantees of mission-critical traffic. However, they also block frames from lower-priority queues, which results in wasted bandwidth [1].

Frame Pre-emption: Interrupting lower priority traffic for critical messages

Frame preemption was originally introduced to improve transmission latency, but it also promotes more efficient bandwidth use. It involves splitting a frame into smaller "framelets," sometimes as small as 64 bytes. IEEE 802.1Qbu describes this technique as enabling a bridge to preempt a frame in transit and intersperse another frame. The goal is to reduce latency for the inserted frame and allow the preempted frame to resume once the inserted one finishes.

Hence, instead of blocking a frame from transmitting entirely, as with guard bands, TSN sends most of the frame and only defers the portion that does not fit in the current time slot. That portion then shifts to the next available time slot. Frame preemption requires two cooperating devices: the transmitting device that splits the frame and the receiving device that reassembles it [20].

1.2.3 Redundancy: Ensuring continuous operation

Each network may lose frames due to flaws in its operating environment. For example, electromagnetic emissions may cause bit flips on communication links. This error can invalidate the frame check sequence of an Ethernet frame, which is its cyclic redundancy check (CRC). A typical office network addresses such losses by using higher-layer retransmission protocols, such as TCP/IP. However, real-time networks often cannot tolerate the delay caused by retransmissions. Therefore, they require alternative methods.

In safety-related networks, it is common to use redundant communication paths that send parallel copies of frames. This parallel redundancy boosts the reliability of frame transmission because at least one copy is likely to arrive. At the destination, or at selected bridges, these duplicate frames must be removed so the information is used only once. Redundancy protocols generally fall into two categories: seamless redundancy and no-seamless redundancy. In seamless redundancy, all network paths operate in parallel. As a result, if a disruption arises, traffic remains protected. In contrast, no-seamless redundancy recovers from a failed path by quickly switching to a new route [17].

Seamless redundancy

Seamless redundancy is essential for continuous operation when a fault arises. In these situations, TSN can use redundancy methods defined in the TSN standards, including IEEE 802.1CB-2017. This standard transmits frames at the same time over disjoint paths. Each time a mission-critical frame is sent, TSN duplicates it and routes the copies through at least two redundant paths. When the duplicates reach their destination, one copy is used while the other is dropped. Seamless redundancy provides the highest level of fault tolerance in Ethernet networks. If one path fails, the other continues the application without pause. Moreover, IEEE 802.1CB-2017 can support an unlimited number of parallel paths, constrained only by the physical network topology [21].

Non-seamless redundancy

Not all TSN applications require seamless redundancy. Many could use non-seamless protocols such as the Rapid Spanning Tree Protocol (RSTP) [22] or the Media Redundancy Protocol (MRP) [23]. The key question is whether the latency and jitter created by switching paths during a fault remain acceptable for the application. In TSN, the network recovery time adds extra latency and jitter because it delays frame delivery. The end device sees this as varying latency, as long as lost frames are compensated by retransmissions. If the redundancy protocol's recovery time stays below the maximum tolerated latency, it can protect the running application in a TSN setting.

1.2.4 Configuration models: managing a TSN

At the beginning of this section, we explained that TSN comprises a set of standards and mechanisms that address different requirements for deterministic data transmissions. To use them together in a network and configure them for various participants from different manufacturers, a standardized configuration method is necessary. This method must enable TSN mechanisms, such as Ethernet frame preemption or seamless redundancy according to IEEE 802.1CB, to be activated on network devices as needed. Moreover, traffic-shaping features like TAS require consistent parameterization to work properly. This includes setting cycle times, assigning traffic class priorities, and defining time slots for time-prioritized transmissions.

Three models for TSN configuration have been developed under IEEE 802.1Qcc: a centralized, a distributed, and a hybrid approach [24]. All three require largely automated configuration to simplify TSN usage. End devices should announce their supported communication mechanisms, such as TAS, and their data transmission requirements. Network devices should then be configured automatically based on these needs and their capabilities.

The general TSN configuration sequence proceeds as follows. First, the network's TSN mechanisms are identified and activated as needed. Next, the transmitting end device (the talker) announces details about its data stream, including cycle time and bandwidth requirements. End devices interested in receiving this stream (the listeners) use that information to register for the specific data flow. Once registered, they receive the corresponding data packets.

These three configuration models differ in how they communicate and process device and stream requirements. In the centralized model, talkers and listeners communicate directly with a logically centralized configuration instance called the Centralized User Configuration (CUC), as shown in Figure 1.7.

The Centralized User Configuration (CUC) combines data stream requirements from the talker and the listener and forwards them to the Centralized Network Configuration (CNC). The CNC, acting as the logical network configuration instance, calculates the time slots for new data streams. It does this by considering the network topology and existing resource reservations. It then configures network participants, such as switches, accordingly. Application-specific protocols like OPC UA facilitate communication between the talker and listener.

Switches are configured using established management protocols, including Simple Network Management Protocol (SNMP) or YANG models via protocols like NETCONF. To connect the CNC with the CUC, IEEE 802.1Qcc specifies basic structures using a YANG model or RESTCONF.

In contrast, the distributed approach propagates end device requirements across the network, as shown in Figure 1.8. Each device determines a configuration for TSN mechanisms using locally available information. The Stream Reservation Protocol (SRP), originally designed for AVB, works on this principle.

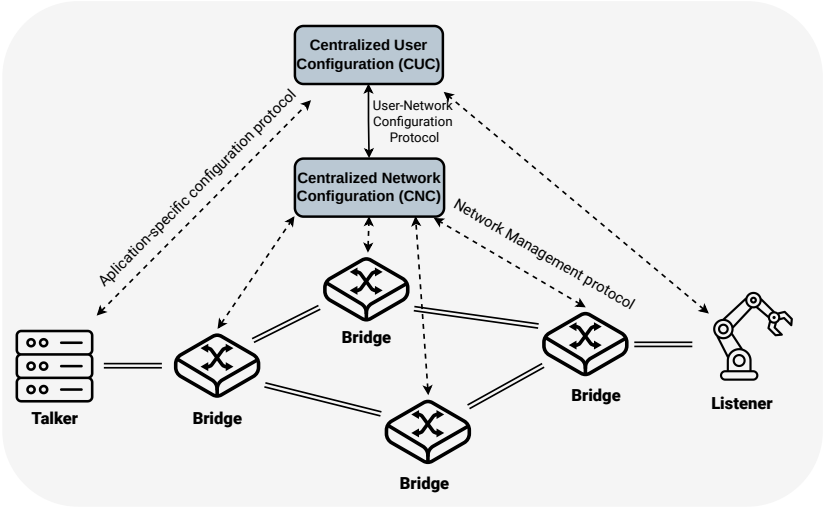


Figure 1.7: Centralized TSN configuration approach

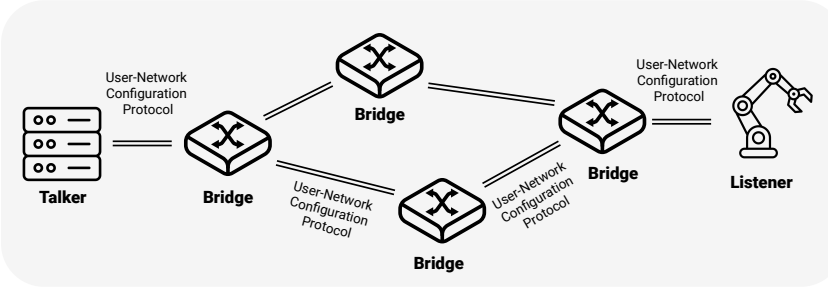


Figure 1.8: Decentralized TSN configuration approach

The hybrid approach combines centralized and decentralized methods. Like the distributed approach, end devices announce their requirements through a distributed protocol. However, as shown in Figure 1.9, the TSN configuration itself is performed centrally. This approach allows end devices to use a single configuration protocol while the network operator can choose between centralized or decentralized management. SRP in IEEE 802.1Qcc has been extended to support this hybrid model.

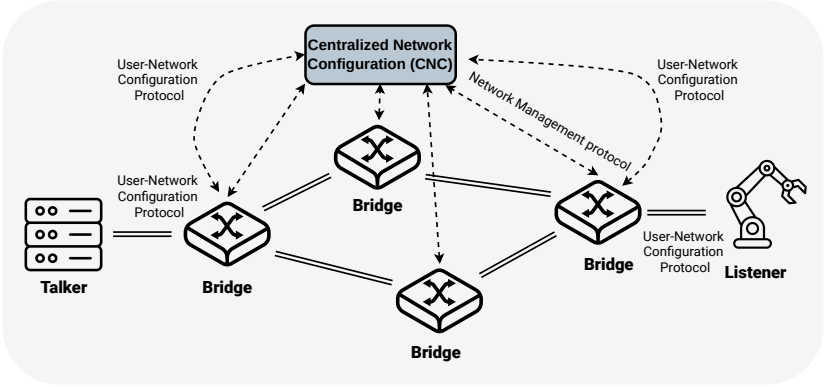


Figure 1.9: Hybrid TSN configuration approach

1.3 The Present and the Future: Transitioning from Wired to Wireless TSN

1.3.1 The Rise of Wireless TSN

Wireless communication technologies have transformed data transmission by removing the need for physical connections between devices. By using electromagnetic waves these technologies enable data transfer over distances without cables. This flexibility has driven the extensive use of wireless systems across multiple fields.

The role of wireless communication in our daily lives is undeniable. They enable cellular networks for phone calls, support Wi-Fi at home for internet access, enable Bluetooth for music, and fuel RFID for contactless payments. Wireless systems seem to be everywhere. However, this is not the case in industrial settings, where approximately 92% of communications still rely on wired systems [25].

Despite this lower adoption rate, wireless communication offers several advantages in industry [26]:

- **Flexibility and Scalability:** Wireless connections simplify the reconfiguration of production lines, allowing fast adaptation to new layouts or processes without rewiring.
- **Lower Installation Costs:** Running cables can be expensive, especially in large or hard-to-reach facilities. Wireless solutions reduce these expenses.
- **Better Accessibility:** In hazardous or confined areas, it is safer and easier to set up wireless nodes than wired equipment.
- **Enhanced Mobility:** Wireless networks support mobile robots, portable

sensors, and handheld devices, which give operators real-time data on the factory floor.

- **Reduced Downtime:** Removing physical cabling lowers the risk of cable failures or accidental disconnections that can stop production.

Nevertheless, significant challenges still obstruct broader adoption. These include signal interference, lower capacity, variable latency, security, coverage, and more, which lead to skepticism from industry [27]. As a result, TSN has been proposed to bring trust to wireless networks in industrial settings. However, the TSN standards have been designed for wired networks, so major adaptations are required to work with wireless. This is a shared effort that began before my work and it has now been consolidated by many academic and industrial participants. On the academic side, European projects such as DETERMINISTIC6G[2] and PREDICT6G[3] are exploring the necessary adaptations for a multi-domain deterministic communication. Meanwhile, alliances like AVNU[4] represent the industrial perspective by contributing to these ongoing efforts.

1.3.2 State of the Art: Current advancements in WTSN technology

Among the various wireless technologies, the 3rd Generation Partnership Project (3GPP) Release 16 (5G) and IEEE 802.11 (Wi-Fi) stand out as the main candidates for integrating TSN capabilities.

Within 3GPP, the main approach for integrating TSN is a black box model in which the 5G system (5GS) serves as a TSN bridge. In Release 16, the 5GS bridge includes three translation functions: (i) a TSN application function (AF) in the control plane, which connects the 5G control plane to the CNC, (ii) a device-side TT (DS-TT) on the user equipment (UE), and (iii) a network-side TT (NW-TT) on the user plane function (UPF) [28].

In this release, the time synchronization grandmaster only works on the network side. However, Release 17 would support multiple grandmasters, including those at the user equipment (UE) side. Current research covers a broad set of topics, such as synchronization errors caused by translation functions, scheduling and resource allocation methods, QoS mapping algorithms, and techniques for latency and reliability. Many of these approaches build on ultra-reliable low-latency communication (URLLC) enhancements but adapt them for TSN [29].

Although significant efforts aim to adapt 5GS for TSN, fundamental differences in QoS profiles, time synchronization, and management architecture limit a seamless transition of frames between Ethernet and 5G.

[2] https://deterministic6g.eu/
[3] https://predict-6g.eu/
[4] https://avnu.org/

On the other hand, Wi-Fi, as an 802 LAN transport technology, can integrate TSN protocols more naturally [30]. This integration eliminates the need for additional translation functions, making Wi-Fi a simpler and more cost-effective option for deterministic communication. Wi-Fi provides key advantages, including reduced infrastructure complexity and lower latency, especially in short-range industrial applications. Additionally, its ease of deployment, flexibility, and lower cost compared to 5G make Wi-Fi an appealing choice for many wireless TSN use cases. However, despite Wi-Fi has the potential to support TSN, its expansion into Wi-Fi networks remains at an early stage.

1.3.3 Deterministic Wi-Fi enablers: Integrating TSN with Wi-Fi

Openwifi + TSN: An open-source approach to wireless TSN

The European Union's Horizon 2020 program funded openwifi, which was developed in 2009 under the Orchestration and Reconfiguration Control Architecture (ORCA) project[5]. Openwifi aimed to provide experimentation facilities and testbed environments for wireless innovation by replacing proprietary Wi-Fi devices [31].

In Linux, the mac80211 subsystem, part of Linux wireless, defines a set of APIs (ieee80211_ops) to control a Wi-Fi chip driver behavior. The SoftMAC Wi-Fi chip driver implements some of these APIs, thereby enabling Linux to support numerous types of Wi-Fi chips. Therefore, by using this approach, openwifi is recognized by Linux as any standard commercial Wi-Fi interface.

As illustrated in Figure 1.10, in addition to accessing the mac80211 APIs, openWiFi provides userspace control through a NETLINK-based system. The project also includes code running on a field-programmable gate array (FPGA) as part of the software-defined radio and an SDR driver that interfaces with the SDR beneath mac80211.

As intended by the ORCA project, openwifi is open source[6]. It provides full access to Wi-Fi 4, enabling users to adjust parameters like CSMA, ACK, NAV, DIFS, EIFS, CW, TX power, RX gain, frequency, sensitivity, transmission rate, and others. Additionally, it offers statistics at both the FPGA such as the channel state information and driver levels. OpenWiFi also supports more than ten SDR development platforms.

Having this level of access to the Wi-Fi stack has allowed us to establish a robust research platform for the community and take the first steps toward developing a Wi-Fi-based wireless TSN solution. Currently, on top of the standard, the openwifi platform includes low-level features such as microsecond-accurate time synchronization and a Qbv-like gating system. Additionally, it incorporates

[5] https://www.orca-project.eu/

[6] https://github.com/open-sdr/openwifi

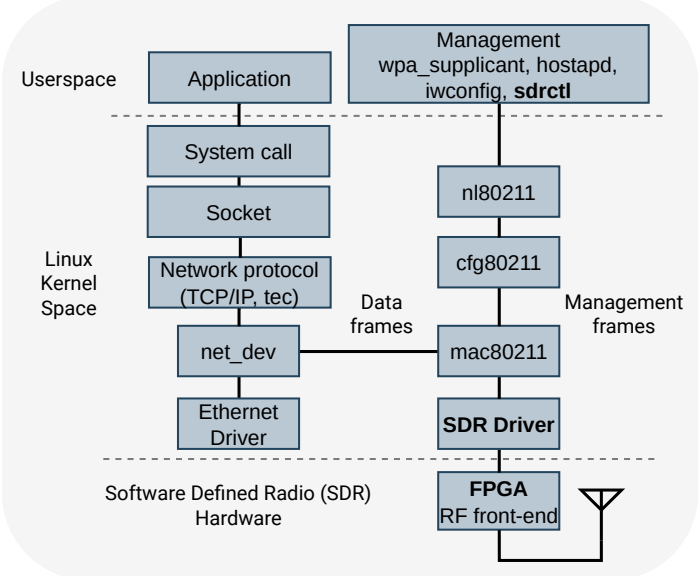


Figure 1.10: Block diagram of the openwifi design stack [32]

high-level features developed as part of this dissertation, including association, handover, management, and more.

Next generation Wi-Fi for wireless TSN

IEEE 802.11be (Extremely High Throughput), also known as Wi-Fi 7, is the latest wireless local area network technology, designed to meet the connectivity demands of Industry 4.0 and beyond. Since its introduction, several consumer-oriented commercial devices have already entered the market. Although Wi-Fi 7 does not explicitly incorporate wireless TSN, it includes several features that will be crucial in making Wi-Fi TSN a reality [33–35].

- **Multi-Link Operation (MLO):** Wi-Fi 7 introduces the capability to access multiple frequency bands (2.4GHz, 5GHz, and 6GHz) concurrently. While primarily aimed at increasing throughput by aggregating links, MLO represents a transformative technology for wireless TSN, enhancing reliability.

- **High Modulation Order:** With Orthogonal Frequency-Division Multiplexing (OFDM) supporting up to 4096-QAM, Wi-Fi 7 enables very high data rates. Although high bandwidth is often less critical for industrial systems, the ability to transmit short frames with low latency is essential. Faster transmission times and concurrent transmissions allow for smaller and shorter

transmission time slots, providing a significant advantage in traffic scheduling.

- **Enhanced OFDMA (Orthogonal Frequency Division Multiple Access):** Wi-Fi 7 builds on OFDMA from Wi-Fi 6 by introducing Multiple Resource Units (MRU), which allow several RUs to serve one client, improving bandwidth use. Adding a frequency dimension removes the time-axis limit in scheduling, but introduces new challenges in RU assignment and queueing delays.

- **Restricted Target Wake Time (r-TWT)**: Another significant feature is r-TWT, an enhanced scheduling mechanism aimed at reducing contention and improving reliability. It enforces stricter transmission control, ensuring uninterrupted scheduled service periods. This behavior closely aligns with the IEEE 802.1Qbv standard's Time-Aware Shaper (TAS), positioning r-TWT as a strong candidate for deterministic wireless communication.

The integration of Wi-Fi and TSN extends beyond merely introducing features that still prioritize bandwidth over determinism. The upcoming IEEE 802.11bn standard brings hope for changing this, as it is expected to focus on Ultra High Reliability (UHR) and incorporate TSN enablers. Some of the most interesting proposed features are [36, 37]:

- **Multi-AP Operation:** Wi-Fi 8 enables multiple access points to cooperate by sharing critical information, such as channel state information. This feature facilitates seamless handovers, extending coverage areas. However, further research is needed to align the management topology with TSN CNC requirements.

- **Non-Primary Channel Access (NPCA):** NPCA allows devices to switch to secondary channels when the primary one is busy. This enhances spectrum efficiency and reduces network congestion. As a result, it improves the determinism of WTSN, particularly when the spectrum is shared with non-TSN devices.

- **PHY and MAC Enhancements for determinism:** These enhancements propose to target two main areas. First, they extend Enhanced Distributed Channel Access (EDCA) by adding new priority classes and corresponding channel access parameters, such as backoff times. Second, they improve OFDMA by introducing resource unit (RU) reservation and allowing preemption. Together, these changes aim to reduce worst-case latency and jitter.

One key challenge that remains underexplored is network management. Due to the dynamic nature of the wireless channel, the TSN schedule requires ongoing adjustments. In this regard, hybrid configuration approaches, such as the model in subsection 1.2.4, combine a global CNC with local CNCs that manage the wireless segments, potentially through APs. This structure may offer a practical and scalable solution.

1.4 Challenges

The primary aim of current research is to develop mechanisms that transfer the deterministic advantages of TSN to wireless networks. This PhD dissertation addresses four key challenges identified in this context:

I Lack of understanding of the relationship between time-sensitive traffic and WTSN.

II Absence of a bootstrapping procedure for WTSN clients.

III Uncertainty in maintaining performance guarantees during mobility.

IV Unrealistic network management in WTSN.

1.4.1 Challenge 1: Lack of understanding of the relationship between time-sensitive traffic and WTSN

At the start of my research, most existing research focused on implementing standards in wired TSN networks and the limited publications on WTSN using Wi-Fi primarily addressed translator mechanisms for functions like time synchronization and basic scheduling, recognizing that Wi-Fi is not a natural extension of Ethernet. However, they lacked a crucial understanding of key factors, such as the relationship between transmission rates and time slot sizes, the behavior of retransmissions, and the constraints posed by channel variations, frame sizes, processing delays, and traffic generation. As critical applications increasingly require wireless guarantees, it is essential not only to develop solutions but also to first establish a solid foundation. This groundwork is vital to identifying the real problems that must be solved to enable a seamless transition from Ethernet to Wi-Fi.

1.4.2 Challenge 2: Absence of a bootstrapping procedure for WTSN clients

As described in the configuration models in Section 1.2, the first step for a new end device to operate in TSN, according to IEEE 802.1Qcc, is to announce its capabilities and requirements to the CUC. Then, the CNC will configure the different

nodes in order to support the requirements. This process is straightforward in wired TSN, where physical, and sometimes dedicated, one-to-one management links can be established between nodes and network managers. However, in WTSN, the medium is shared. This means that when a wireless prospective client announces its capabilities and requirements, it may cause delays in time-sensitive traffic or, in the worst case, collisions. These collisions would require retransmissions, negatively impacting already associated wireless nodes. Therefore, it is essential to design a mechanism that meets multiple requirements. It should enable fast and reliable association for prospective clients, remain transparent to higher-layer association mechanisms, and avoid disrupting other time-sensitive traffic.

1.4.3 Challenge 3: Uncertainty in maintaining performance guarantees during mobility

The next challenge arises once the wireless client is associated. After announcing its requirements to the CUC, the CNC configures the switches and APs along the traffic flow(s) path. This allows the client to benefit from TSN even while moving. However, in challenging environments, such as industrial settings, wireless signals often degrade over distance or encounter obstructions. To maintain the required RSSI and SNR levels across an area, multiple APs are typically necessary. Consequently, the associated client must switch between APs, and TSN must ensure guarantees even during these switching events, commonly referred to as handovers. When only one wireless interface is available and different dedicated channels are assigned to each AP, soft handovers are not feasible. Therefore, a procedure is essential to anticipate handover events, prepare resources in advance, and guide the client to the new AP while bounding delays. This ensures seamless transitions and preserves the performance guarantees expected from TSN.

1.4.4 Challenge 4: Unrealistic network management in WTSN

While dedicated TSN hardware now exists specifically for wired TSN, most research remains focused on traffic scheduling challenges, particularly from a theoretical perspective. These studies propose mechanisms capable of scheduling thousands of traffic flows in complex topologies. However, in wired TSN, once the configuration is set, updates are rarely needed apart from events like adding or removing a traffic flow. Unfortunately, this is not the case in WTSN.

As discussed in the previous section, channel variability is a fundamental characteristic of wireless communications. This continuous variability, linked to factors such as traffic generation, frame losses, processing delays, and transmission, demands new paradigms for TSN network management. It is crucial to understand that processing and communication operate as an integrated chain with inherent

variabilities. These variabilities must be compensated to maintain the required level of determinism.

1.5 Research contributions

This section outlines the specific research contributions detailed in the dissertation. A comprehensive list of the contributions made in each chapter is presented below.

- Time-Sensitive Traffic Support in WTSN (Chapter 2)
  - Identified key challenges in extending TSN features from Ethernet to wireless, such as synchronization, reliability, and seamless integration. Discussed the shared nature of wireless, interference issues, and variable link quality.
  - Conducted experimental validation of WTSN scheduling strategies. Empirically evaluated scheduling strategies (dedicated vs. shared time slots) using a Wi-Fi testbed.
  - Developed an SMT-based traffic scheduler that incorporates wireless-specific parameters, including variable transmission rates and contention. Analyzed scheduling effectiveness and reconfiguration delays, highlighting the challenges of adapting deterministic methods to dynamic wireless environments.
  - Presented a real-world demonstration using a distributed ball-balancing Proportional-integral—derivative (PID) machine control system over WTSN, illustrating the practical impact of shared versus dedicated time slots on latency and system performance.
- Impactless Association Methods for Wi-Fi based Time-Sensitive Networks (Chapter 3)
  - Developed an efficient association procedure fully compatible with TSN and optimized for Wi-Fi. Demonstrated the feasibility of seamless association by pre-scheduling time slots and embedding association offsets in AP broadcast beacons.
  - Proposed three innovative pre-synchronization methods to enable wireless association for prospective TSN clients. The first method assumes a fixed beacon transmission interval, while the second and third methods use a transmission schedule enforced at the AP.
  - Introduced a pre-scheduling mechanism that enables the AP and CNC to dynamically determine the association time slot using beacons.

  - Proposed an efficient encoding scheme to reduce the number of occupied bits in the beacons vendor-element.
  - This work presents a real-world implementation and test of the association system, which includes two STAs and one AP. Moreover, it achieves association delays shorter than one second, requires three beacons, and enables real-time changes in the pre-schedule.

- Unlocking Mobility for Wi-Fi-based Wireless Time-Sensitive Networks (Chapter 4)

  - Introduced a state machine-based approach for handover within Wi-Fi-based WTSN. Designed a centralized architecture incorporating a Handover Controller Module (HCM) and a Handover Agent Module (HAM) to manage and optimize the handover process.
  - Reduced handover delay to under 10 milliseconds through modifications in channel switching and userspace/driver-level communication optimizations, achieving significant improvements in uplink and downlink performance during handover events.
  - Proposed three AP candidate determination techniques: localization-based, coordinated beacon and scanning, and coordinated probing. Implemented the localization approach using high-accuracy UWB localization to eliminate the need for traditional channel scanning, ensuring low interruption to time-sensitive traffic during handover.
  - Developed and tested three one-dimensional spacial approaches for selecting the handover moment:
    * **Middle Point:** A simple threshold-based mechanism.
    * **Offline Learning:** Pre-trained Radio Environmental Map (REM) to optimize handover decisions based on RSSI values.
    * **Online Learning:** Dynamic adaptation of handover points using continuous RSSI monitoring and fitting of logarithmic models, enabling real-time adjustments to environmental changes.
  - Proposed two, two-dimensional techniques for environments with more than two APs.:
    * **Fixed Polygon-Based:** Dividing space into fixed regions with buffer zones to reduce ping-pong effects.
    * **Machine Learning Online-Based:** Leveraged artificial neural networks and continuous monitoring to dynamically predict RSSI values and optimize handover points, reducing unnecessary handovers and improving network performance.

  - Conducted extensive testing in one-dimensional and two-dimensional mobility scenarios, involving realistic network setups and rover mobility patterns. Demonstrated improvements in throughput, jitter, and frame losses.
  - Presented a real-world implementation and demonstration validating the effectiveness of combining openwifi and machine learning techniques for handover decisions.

- A Predictive Management Framework for Low-Latency and Low-Jitter Wireless TSN Systems (Chapter 5)
  - Proposed a Wi-Fi-based WTSN Digital Twin (DT) management reference framework element. Integrating the WTSN-DT framework with TSN centralized configuration elements (CNC and CUC) to ensure compliance with the IEEE 802.1Qcc standard. This proof of concept focused on reducing the residual service time (RST) of unsynchronized traffic generators.
  - Investigated three analytical and machine learning-based approaches: Linear Regression, Kalman Filtering, and Long Short-Term Memory Networks, for predicting real-world cyclic traffic patterns based on frame arrival time series.
  - Conducted a comprehensive evaluation of the proposed modeling techniques through simulations and real-world experiments. Highlighted trade-offs in accuracy, fitting/forecasting time, and latency reduction. Validated the proposed approach in single and multi-node configurations, demonstrating adaptability to different network topologies and traffic scenarios.
  - Implemented the proposed WTSN-DT management framework element and a low-latency-focused scheduler in a physical Wi-Fi WTSN testbed, demonstrating the feasibility of reducing wireless communication link jitter and end-to-end latency, achieving up to 96% latency reduction at the 90th percentile.

1.6 Outline

This dissertation brings together a number of publications that address the research challenges and findings explored during the study. The different research contributions are detailed in Section 1.5 and the complete list of publications that resulted from this work is presented in Section 1.7. In this section, we summarize the key insights from each chapter and explain how they connect. Figure 1.11 provides an outline of the dissertation's structure, showing the relationships between chapters.

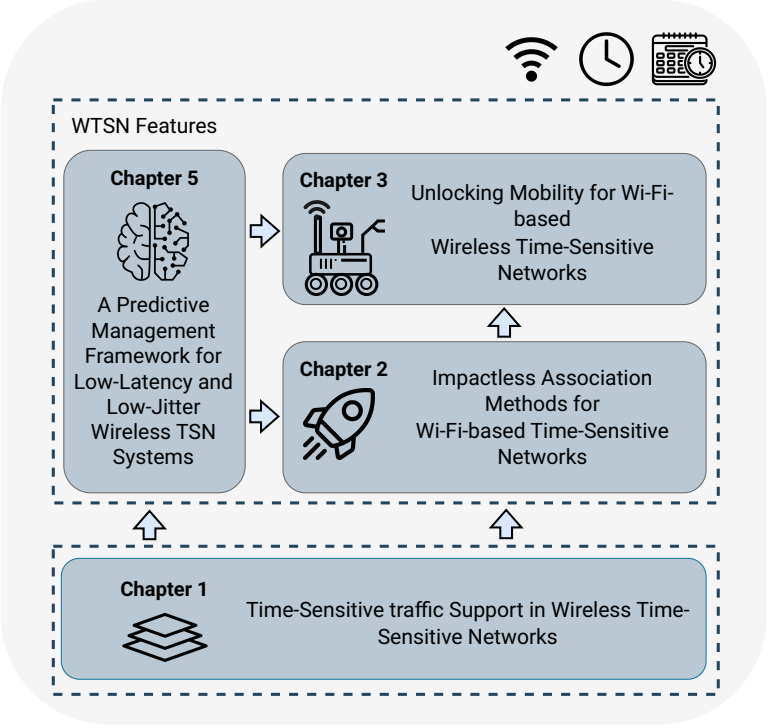


Figure 1.11: Outline and Chapter relationships in the dissertation

The journey begins in Chapter 2, which tackles the foundational challenge of transitioning TSN from wired to wireless networks. This transition introduces unique issues, such as shared medium access, variable channel conditions, and lower transmission rates, which make traffic scheduling particularly complex. By using openwifi, this chapter investigates scheduling features like varying transmission slot sizes and the distinction between shared and dedicated time slots. These experiments led to the development of an Satisfiability Modulo Theories (SMT)

based TSN scheduler that incorporates real-world parameters such as variable transmission rates and processing delays. The chapter also identifies the inefficiencies wireless introduces in hybrid wired-wireless setups. To contextualize these findings, a PID-based machine control application was tested, demonstrating how wireless TSN can handle time-sensitive traffic in industrial settings. By exploring the practical use case of a machine control loop this work not only illustrates the challenges but also lays the groundwork for solutions explored in subsequent chapters.

Once the challenges of WTSN were unpacked in Chapter 2, in Chapter 3 the journey progresses to the first critical step in deploying a wireless TSN: bootstrapping. A real-world wireless TSN implementation cannot assume already associated clients; a process must enable this. Furthermore, the default IEEE 802.11 association and authentication processes are unsuitable for WTSN. The problem was: how can time-sensitive features, such as synchronization and scheduling, be extended to a prospective client before it joins the network? The solution centers on beacons. Typically used by APs to announce presence and capabilities, beacons were adapted to: 1) pre-schedule prospective clients with information about association slots and 2) pre-synchronize prospective clients, allowing them to transmit association and authentication frames without disrupting time-sensitive traffic. The chapter addresses the complexities of this process, including the variable delays introduced by dynamic AP schedules, which were accounted for to ensure synchronization accuracy. Both pre-scheduling and pre-synchronization were implemented and tested in a real environment, where the CNC dynamically adjusted association slot positions and enforced AP transmission schedules.

Once clients are associated to the WTSN, Chapter 4 explores the most significant advantage of wireless TSN: mobility. Industrial environments often have obstacles and large coverage areas, necessitating multiple APs and, therefore, handovers. A key question arises: can time-sensitive guarantees be maintained during handovers? The answer, demonstrated in this chapter, is yes. A handover state machine procedure was developed, emphasizing that fast handover is not enough; it must also occur at the right moment. Fast handover was achieved through optimized communication of handover commands, while the right timing was determined using methods that included ultra-wideband-based localization, and machine learning. These solutions were implemented and tested in an industrial environment using a rover, affectionately named ”Roverto,” which successfully demonstrated seamless handovers in real-world scenarios.

Finally, Chapter 5 discusses network management. While management is an underlying theme in the previous chapters, this chapter consolidates the challenges. Traffic scheduling, as explored in Chapter 2, is one of the biggest obstacles to TSN adoption. Effective management must also handle association slots for prospective clients, as shown in Chapter 3, and prepare for handovers by reserving resources and triggering actions when needed. Although substantial progress has been made

Table 1.1: Overview of contributions per chapter in this dissertation

Contribution	Related Chapter			
	Ch. 2	**Ch. 3**	**Ch. 4**	**Ch. 5**
Lack of understanding of traffic behavior complexities in WTSN	•			
Absence of an bootstrapping procedure for WTSN clients		•		
Uncertainty in maintaining performance guarantees during mobility			•	
Unrealistic network management in WTSN				•

in management and specially in scheduling algorithms, including exact and machine learning-based approaches, they often oversimplify the complexities of real-world factors. These include variable delays in frame processing, channel propagation, transmission, time synchronization accuracy, and traffic generation jitter. To address this gap, this chapter proposes combining digital twins with wireless TSN. This innovative approach enables realistic decision-making using models trained on real-time data, allowing proactive responses to future scenarios. This integration serves as the culmination of the research, uniting insights from previous chapters and advancing the potential of wireless TSN.

This dissertation represents a comprehensive journey through the challenges and solutions in realizing WTSN. From addressing foundational issues such as scheduling and association, to exploring the intricacies of mobility and network management, each chapter has contributed critical insights toward making wireless TSN a practical reality. By combining theoretical advancements with practical implementations, the work bridges the gap between academic research and real-world industrial applications. The integration of digital twins in network management, as presented in the final chapter, not only consolidates the findings but also points toward a promising future for adaptive and robust wireless TSN systems. This research lays a strong foundation for further exploration and underscores the transformative potential of wireless TSN in enabling next-generation industrial and real-time communication systems. Table 1.1 shows the challenges that were highlighted in Section 1.4 and indicates chapters at which they are targeted.

1.7 Publications

The findings from this PhD research have been published in peer-reviewed scientific journals and shared at various international conferences. The following list

summarizes the publications produced during the course of this research.

1.7.1 Publications in international journals (listed in the Science Citation Index[7])

1. **P. Avila-Campos**, J. Haxhibeqiri, I. Moerman, X. Jiao, and J. Hoebeke, "Impactless association methods for wi-fi based time-sensitive networks," WIRELESS NETWORKS, 2024.

2. **Pablo Avila-Campos**, Jetmir Haxhibeqiri, Xianjun Jiao, Ben Van Herbruggen, Ingrid Moerman, and Jeroen Hoebeke. *Unlocking Mobility for Wi-Fi-based Wireless Time-Sensitive Networks*, IEEE ACCESS, vol. 12, pp. 30687–30699, 2024.

1.7.2 Publications in international conferences (listed in the Science Citation Index [8])

1. **P. Avila-Campos**, J. Haxhibeqiri, X. Jiao, I. Moerman, and J. Hoebeke, "Optimizing Handover in Time-Sensitive Wi-Fi Networks through Machine Learning," in 2024 IEEE 29th International Conference on Emerging Technologies and Factory Automation (ETFA), Padova, Italy, 2024, pp. 1–8.

2. J. Haxhibeqiri, **P. Avila-Campos**, I. Moerman, and J. Hoebeke, "Optimizing scheduling in wireless TSN utilizing genetic algorithms," in WiMob2024, the 20th International Conference on Wireless and Mobile Computing, Networking and Communications, Paris, France, 2024.

3. M. Girmay, **P. Avila-Campos**, V. Maglogiannis, D. Naudts, A. Shahid, and I. Moerman, "Intelligent spectrum sharing between LTE and Wi-Fi networks using muted MBSFN subframes," in 2023 IEEE Wireless and Microwave Technology Conference (WAMICON), Melbourne, FL, USA, 2023, pp. 13–16.

4. **P. Avila-Campos**, J. Haxhibeqiri, M. Girmay, I. Moerman, and J. Hoebeke, "Residual service time optimization for legacy wireless-TSN end nodes," in 2023 19th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), Montréal, Canada, 2023, pp. 466–471.

[7]The publications listed are recognized as 'A1 publications', according to the following definition used by Ghent University: A1 publications are articles listed in the Science Citation Index Expanded, the Social Science Citation Index or the Arts and Humanities Citation Index of the ISI Web of Science, restricted to contributions listed as article, review, letter, note or proceedings paper.

[8]The publications listed are recognized as 'P1 publications', according to the following definition used by Ghent University: P1 publications are proceedings listed in the Conference Proceedings Citation Index - Science or Conference Proceedings Citation Index - Social Science and Humanities of the ISI Web of Science, restricted to contributions listed as article, review, letter, note or proceedings paper, except for publications that are classified as A1.

5. J. Haxhibeqiri, X. Jiao, **P. Avila-Campos**, I. Moerman, and J. Hoebeke, "To update or not: dynamic traffic classification for high priority traffic in wireless TSN," in 2023 IEEE 19TH INTERNATIONAL CONFERENCE ON FACTORY COMMUNICATION SYSTEMS, WFCS, Pavia, Italy, 2023, pp. 1–8.

6. **P. Avila-Campos**, J. Haxhibeqiri, I. Moerman, and J. Hoebeke, "Beacon-based wireless TSN association," in IEEE INFOCOM 2022 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), New York, NY, USA, 2022.

7. **P. Avila-Campos**, J. Haxhibeqiri, I. Moerman, and J. Hoebeke, "Impactless beacon-based wireless TSN association procedure," in 18TH IEEE INTERNATIONAL WORKSHOP ON FACTORY COMMUNICATION SYSTEMS 2022 (WFCS 2022), Pavia, Italy, 2022, pp. 49–56.

8. J. Haxhibeqiri, **P. Avila-Campos**, I. Moerman, and J. Hoebeke, "Safety-related applications over wireless time-sensitive networks," in 2022 IEEE 27TH INTERNATIONAL CONFERENCE ON EMERGING TECHNOLOGIES AND FACTORY AUTOMATION (ETFA), Stuttgart, Germany, 2022.

1.7.3 Publications in other international conferences

1. **P. Avila-Campos** et al., "Periodic Control Traffic Support in a Wireless Time-Sensitive Network," in 2022 13th International Conference on Network of the Future (NoF), Ghent, Belgium, 2022.

2. **P. Avila-Campos**, J. Haxhibeqiri, X. Jiao, I. Moerman, and J. Hoebeke, "Removing the wires in time-sensitive networks," in 2022 61st FITCE International Congress Future Telecommunications: Infrastructure and Sustainability (FITCE), Rome, Italy, 2022.

1.7.4 Patents

1. J. Hoebeke, J. Haxhibeqiri, **P. E. A. Campos**, "WO2023198464A1 - Impactless associating in a wireless time sensitive network," Apr. 13, 2022. https://patents.google.com/patent/WO2023198464A1/en

2. J. Hoebeke, J. Haxhibeqiri, **P. E. A. Campos**, "WO2025068033 - Impactless hand-over in a wireless time-sensitive network," Apr. 3, 2025. https://patentscope.wipo.int/search/en/WO2025068033

References

[1] H. Chahed and A. Kassler. *TSN Network Scheduling—Challenges and Approaches*. Network, 3:585–624, 12 2023. Available from: https://urn.kb.se/resolve?urn=urn:nbn:se:kau:diva-98183, doi:10.3390/NETWORK3040026.

[2] *TSN Technology*. Available from: https://www.belden.com/solutions/tsn-technology.

[3] *Development solution for next generation Ethernet using TSN — Industrial Ethernet Book*. Available from: https://iebmedia.com/technology/industrial-ethernet/development-solution-for-next-generation-ethernet-using-tsn/.

[4] S. Potter. *Following Protocols: A Dive Into The World Of Audio Over Ethernet (AoE) - ProSoundWeb*, 10 2021. Available from: https://www.prosoundweb.com/following-protocols-a-dive-into-the-world-of-audio-over-ethernet-aoe/.

[5] *802.1BA-2011 IEEE standard for local and metropolitan area networks-audio video bridging (AVB) systems*. page 31, 2011.

[6] *IEEE 802.1 AS: IEEE Standard for Local and Metropolitan Area Networks-Timing and Synchronization for Time-Sensitive Applications*, 2020. Available from: https://standards.ieee.org/ieee/802.1AS/7121/.

[7] H.-T. Lim, D. Herrscher, M. J. Waltl, and F. Chaari. *Performance Analysis of the IEEE 802.1 Ethernet Audio/Video Bridging Standard*. 2012.

[8] M. Warburton, M. Mon-Williams, F. Mushtaq, and J. R. Morehead. *Measuring motion-to-photon latency for sensorimotor experiments with virtual reality systems*. bioRxiv, page 2022.06.24.497509, 6 2022. doi:10.1101/2022.06.24.497509.

[9] *What is Motion-To-Photon Latency?* Available from: http://www.chioka.in/what-is-motion-to-photon-latency/.

[10] J. Farkas. *IEEE 802.1 TSN Profiles*. 2 2021.

[11] S. Bhattacharjee, R. Schmidt, K. Katsalis, C. Y. Chang, T. Bauschert, and N. Nikaein. *Time-Sensitive Networking for 5G Fronthaul Networks*. IEEE International Conference on Communications, 2020-June, 6 2020. doi:10.1109/ICC40277.2020.9149161.

[12] H. Lambers, W. Fischer, and D. Kliem. *Cabin Electronics-Airbus Buxtehude TSN Applications in the Aircraft Cabin*. 2024.

[13] D. Pannell. *Use Cases-IEEE P802.1DG*. 10 2019.

[14] S. Brooks and E. Uludag. *Time-Sensitive Networking: From Theory to Implementation in Industrial Automation - White paper*.

[15] *IEEE 1588: IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems*, 2008. Available from: https://standards.ieee.org/ieee/1588/4355/.

[16] *IEEE Std 802.1Q-2014 : IEEE Standard for Local and metropolitan area networks-Bridges and Bridged Networks*. 2014.

[17] R. Zurawski. *Industrial Communication Technology Handbook, Second Edition*. 2017. Available from: https://books.google.be/books?id=RZ1BBAAAQBAJ&printsec=frontcover&redir_esc=y#v=onepage&q&f=false.

[18] L. Maile, D. Voitlein, A. Grigorjew, K. S. J. Hielscher, and R. German. *On the Validity of Credit-Based Shaper Delay Guarantees in Decentralized Reservation Protocols*. ACM International Conference Proceeding Series, pages 108–118, 6 2023. doi:10.1145/3575757.3593644.

[19] *IEEE 802.1Qbv-2015 - IEEE Standard for Local and Metropolitan Area Networks - Enhancements for scheduled traffic : bridges and bridged networks*. page 57, 2016.

[20] *IEEE 802.1Qbu-2016: Frame Preemption*. Available from: https://standards.ieee.org/ieee/802.1Qbu/5464/.

[21] *802.1CB-2017 - IEEE Standard for Local and metropolitan area networks–Frame Replication and Elimination for Reliability*. 2017.

[22] *Rapid Spanning Tree Protocol (RSTP) — Accuenergy*. Available from: https://www.accuenergy.com/support/reference-directory/rapid-spanning-tree-protocol-rstp/.

[23] A. Telesis and I. A. rights reserved. *Media Redundancy Protocol (MRP) Feature Overview and Configuration Guide*.

[24] *IEEE 802.1Qcc - Stream Reservation Protocol (SRP) Enhancements and Performance Improvements*. pages 1–236, 2018.

[25] *Industrial Ethernet & wireless networks growing — Industrial Ethernet Book*, 5 2023. Available from: https://iebmedia.com/news/tech-updates/industrial-ethernet-wireless-growing/.

[26] Z. Gdali. *Defining Wireless LAN: Key Benefits and Applications in Factory and Warehouse Automation - firecell.io*, 12 2024. Available

from: https://firecell.io/defining-wireless-lan-key-benefits-and-applications-in-factory-and-warehouse-automation/.

[27] J. Takacs. *Challenges of Wireless Communication in Manufacturing: Navigating Interference, Congestion, Security — ASSEMBLY*, 7 2024. Available from: https://www.assemblymag.com/articles/98653-challenges-of-wireless-communication-in-manufacturing-navigating-interference-congestion-security.

[28] 5G-ACIA. *Integration of 5G with Time-Sensitive Networking for Industrial Communications*, 2021.

[29] K. Zanbouri, M. Noor-A-Rahim, J. John, C. J. Sreenan, H. V. Poor, and D. Pesch. *A Comprehensive Survey of Wireless Time-Sensitive Networking (TSN): Architecture, Technologies, Applications, and Open Issues*. 12 2023. Available from: http://arxiv.org/abs/2312.01204, doi:10.1109/COMST.2024.3486618.

[30] D. Cavalcanti and S. Bush. *Wireless TSN-Definitions, Use Cases & Standards Roadmap - Avnu Alliance® White Paper*. 3 2020.

[31] *IDLab — Openwifi – wireless time-sensitive networking*. Available from: https://idlab.ugent.be/resources/openwifi-wireless-time-sensitive-networking.

[32] *GitHub - open-sdr/openwifi: open-source IEEE 802.11 WiFi baseband FPGA (chip) design: driver, software*. Available from: https://github.com/open-sdr/openwifi.

[33] K. Zanbouri, M. Noor-A-Rahim, J. John, C. J. Sreenan, H. V. Poor, and D. Pesch. *A Comprehensive Survey of Wireless Time-Sensitive Networking (TSN): Architecture, Technologies, Applications, and Open Issues*. 12 2023. Available from: http://arxiv.org/abs/2312.01204, doi:10.1109/COMST.2024.3486618.

[34] T. Adame, M. Carrascosa-Zamacois, and B. Bellalta. *Time-Sensitive Networking in IEEE 802.11be: On the Way to Low-Latency WiFi 7*. Sensors 2021, Vol. 21, Page 4954, 21:4954, 7 2021. Available from: https://www.mdpi.com/1424-8220/21/15/4954/htmhttps://www.mdpi.com/1424-8220/21/15/4954, doi:10.3390/S21154954.

[35] J. John, M. Noor-A-Rahim, A. Vijayan, H. V. Poor, and D. Pesch. *Industry 4.0 and Beyond: The Role of 5G, WiFi 7, and Time-Sensitive Networking (TSN) in Enabling Smart Manufacturing*. Future Internet 2024, Vol. 16, Page 345, 16:345, 9 2024. Available from: https://www.mdpi.com/1999-5903/16/9/345/htmhttps://www.mdpi.com/1999-5903/16/9/345, doi:10.3390/FI16090345.

[36] L. Galati-Giordano, G. Geraci, M. Carrascosa, and B. Bellalta. *What Will Wi-Fi 8 Be? A Primer on IEEE 802.11bn Ultra High Reliability*. 2023.

[37] E. Reshef and C. Cordeiro. *Future Directions for Wi-Fi 8 and Beyond*. IEEE Communications Magazine, 60:50–55, 10 2022. doi:10.1109/MCOM.003.2200037.

2

Time-Sensitive traffic Support in Wireless Time-Sensitive Networks

"To learn to doubt is to learn to think."

–Octavio Paz

The transition from wired to wireless TSN raises critical questions about the reliability of wireless systems in supporting time-sensitive traffic. It also highlights the relationship between transmission timeslot sizes and the variable transmission times inherent to wireless systems. These challenges become more pronounced when low-cycle traffic flows occur alongside poor wireless channel conditions. To address these issues and develop effective strategies, evaluating this behavior under real-world conditions is essential. This chapter provides an initial perspective on the challenges of wireless TSN. It introduces the complexities of TSN scheduling and sets the stage for demonstrating how a practical use case, such as a machine control loop, can be effectively supported by wireless TSN.

⋆ ⋆ ⋆

This Chapter is based on:
Removing the Wires in Time-Sensitive Networks

Presented in FITCE, Sep. 2022.
and:

Periodic Control Traffic Support in a Wireless Time-Sensitive Network

Presented in NoF, Oct. 2022.

⋆ ⋆ ⋆

Abstract

Applications ranging from machine control to multimedia and distributed real-time monitoring demand high determinism and low end-to-end latency. Time-Sensitive Networking (TSN), built on Ethernet, has emerged as the solution in recent years. This chapter examines some of the critical challenges that must be addressed to extend TSN's time-sensitive operations seamlessly into the wireless domain. We present a TSN-compatible design based on our open-source Wi-Fi implementation, with a specific focus on scheduling. Two experiments were conducted on our IEEE 802.11-based WTSN testbed, comparing contention between shared and dedicated time slots. The results reveal a smaller difference in the number of received packets between shared and dedicated slots tests at shorter time slots. Additionally, through a demonstration, this chapter showcases time synchronization and scheduling mechanisms in a wireless setting. A control loop scenario, where a ball is balanced in a canal, illustrates how time-critical traffic can maintain the required latency under scheduled operation.

2.1 Introduction

Historically, wireless technologies offering deterministic communication, such as WirelessHART, IEEE 802.15.4 TSCH, and ISA 100.11a [1, 2], have been developed, but their data rates were limited to hundreds of kilobits per second. In contrast, Wi-Fi has always served as a natural extension of Ethernet networks, offering lower management and operational complexity compared to cellular technologies like 5G. Although Wi-Fi operates in unlicensed spectrum and has traditionally focused on increasing data rates, scaling networks, and achieving medium access fairness, recent advances in Wi-Fi specifications and research on integrating TSN features position Wi-Fi as a cost-effective candidate for wireless TSN. This chapter explores Wi-Fi's potential in this context. We present a novel preliminary study measuring the influence of time slot size on cycle throughput in a wireless TSN, comparing dedicated and shared time slot scenarios to highlight the impact of contention. Additionally, we demonstrate the capability of WTSN to support time-sensitive traffic.

This chapter is structured as follows: Section 2.2 outlines the requirements for a Wi-Fi-based TSN and discusses the associated network challenges. Section 2.3 explains how openwifi facilitates the implementation of a WTSN. Section 2.4 presents a preliminary evaluation of scheduling strategies for uplink traffic in

WTSN. Section 2.5 presents an SMT-based traffic scheduler, while Section 2.6 describes the demo setup. Finally, Section 2.7 summarizes the chapter's findings.

2.2 The road to Wi-Fi TSN

Despite the recent advancements introduced by Wi-Fi 7 [3], achieving wire-equivalent dependable and secure wireless communication with time guarantees for usage in time-critical applications remains a major challenge. Moreover, to fully leverage end-to-end deterministic communication, it is important to have a seamless operation between wired and wireless TSN domains. As such, any upcoming wireless TSN features need to be fully compatible with wired TSN features and easy to manage by a unified network controller. As presented in Figure 2.1, a wired-wireless TSN that makes use of Wi-Fi, uses Ethernet as the communication backbone. This backbone interconnects network management devices such as the Central Network Controller (CNC), wired end-nodes, network switches, and Wi-Fi Access Points (AP), that should support TSN features for the wireless network part. From the network management perspective, CNC is the converging point for managing both wired and wireless TSN domains.

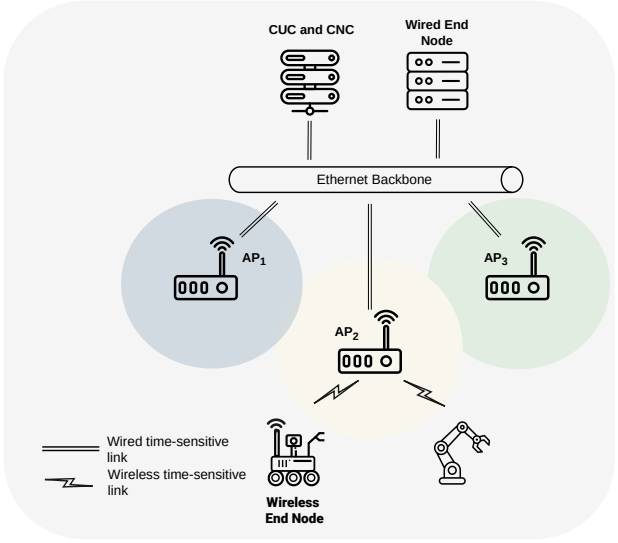


Figure 2.1: Typical architecture of Ethernet and Wi-Fi networks

The wireless network domain inherits its beacon-based time synchronization, which is currently only used to announce network presence and for control purposes. Utilization of such mechanism for time synchronization within the wireless domain

has been done in the past. However, either the accuracy is not sufficiently high [4] or the synchronization is not achieved end-to-end and is not unified with the wired TSN [5]. In order to support Precision Time Protocol (PTP) over wireless, an accurate timestamping mechanism of PTP-related packets needs to be implemented, which is missing in current Wi-Fi interface cards. To mitigate this, accurate timestamping of PTP packets is needed [6]. Such high-accuracy time synchronization can be further utilized for better channel access coordination.

The basic contrasts between wireless and wired communications, such as the variable capacity of wireless links and the Packet Error Rate (PER) being typically greater in wireless, are some of the issues to be addressed when mapping TSN features to wireless. Another essential factor to consider is the shared nature of wireless. On the one hand, it may allow for more devices to be reached with a single broadcast, but it is also more prone to interference. As a result, coordinated access, as well as interference resilience, are critical.

Another aspect to consider is the channel multiple access method used by Wi-Fi. Carrier sense multiple access/collision avoidance (CSMA/CA) tries to maximize the medium usage and lower the amount of collisions by checking whether traffic is present [7]. To move towards Wi-Fi TSN, channel access from different nodes needs to be organized and become deterministic. The IEEE 802.11 standard has introduced schemes to prioritize traffic such as Enhanced Distributed Channel Access (EDCA) in order to support certain Quality of Service (QoS). However, such approaches are not focused on maintaining determinism in channel access. In the past, some works such as hMAC [8], have tried to improve channel access determinism by introducing TDMA on top of COTS, by achieving determinism only for downlink communication. In [9] the w-SHARP technology is presented, achieving sub-millisecond real-time communication cycles. Such TSN features are promising, however, in terms of network management and wired-wireless TSN interoperability, there is still a way to go. Other solutions have evaluated TDMA on top of Wi-Fi in a network simulator showing the feasibility of using Wi-Fi for TSN as well [10]. As aforementioned, one of the TSN techniques to achieve reliability is to assign dedicated time slots (TS). Hence, in a fully controlled WTSN, a multiple access method such as CSMA/CA is no longer required for dedicated time slots. However, in a realistic scenario, time resources are limited, and nodes have different requirements and priorities which can lead to dedicated or shared time slot strategies that will be analyzed in Section 2.4.

Control traffic for managing the WTSN should not only coordinate the node's access to channel resources but also monitor the status of the links. This would inevitably lead to an increase in overhead in the network compared with non-TSN Wi-Fi. Opposite to cellular networks, in Wi-Fi only a single channel is used for all communications, which might create bottlenecks. For managing the scheduling mechanism in WTSN an approach would be to incorporate scheduling

information in existing Wi-Fi control traffic in the downlink. Further optimization of channel usage can be achieved by enriching the data packets with monitoring information [11].

2.3 Wireless TSN based on openwifi

Openwifi[1] is the first open-source Linux mac80211 compatible full-stack IEEE 802.11/Wi-Fi design based on Software Defined Radio (SDR) [12]. Currently, openwifi includes most of the 802.11a/g/n features together with the freedom to test features such as CCA, retransmissions, contention delays, and more. In addition, the platform's flexibility offers ample opportunities to implement and explore new features in view of wireless TSN, including the design of high-accuracy hardware timestamping for PTP synchronization and a time-aware gating system.

To have unified end-to-end time synchronization, PTP synchronization over wireless is enabled in openwifi. To achieve accurate time synchronization in wireless networks, PTP-related packets are timestamped using the Time Synchronization Function (TSF) timer along with hardware-based timestamping. This approach corrects both clock offset and clock skew [13]. Since the TSF is a hardware timer already used in Wi-Fi networks, relying on it also helps minimize the overall hardware footprint. As such, there is no need for a dedicated hardware clock to achieve synchronization. Based on the TSF timer resolution that is 1 μs, the achieved accuracy is in the range of ~ 1 μs [13]. Since the TSF timers of the devices are synchronized with each other via PTP, end-device transmission coordination can be controlled using such a clock.

openwifi supports four different hardware traffic queues. Each of the queues can support different channel access parameters such as contention window (CW) min, CW max, AFIS, and maximum transmission opportunity, as specified by IEEE 802.11e. Such channel access parameters can be dynamically changed at run time for faster access if required.

In addition to normal priority-based channel access, openwifi supports a gated channel access mechanism for all of the queues. The gate control mechanism is coordinated by the TSF timer. The parameters that are specified by the controller are the communication cycle length, time slot start and time slot end inside the cycle length for each queue as shown in Figure 2.2. Then the communication cycle is repeated periodically over time. At every TSF increment, the modulo operation of the TSF value and communication cycle length is checked and if the output is 0 then the new communication cycle starts. Then, from the communication cycle start, the time slot start and end are determined. Since the TSF counter accuracy is 1 μs, the smallest time slot that can be applied theoretically is 1 μs. This design is fully compatible with IEEE 802.1Qbv [14].

[1] https://github.com/open-sdr/openwifi

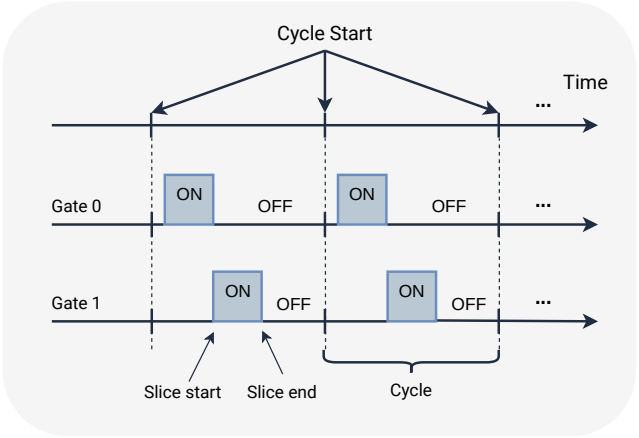


Figure 2.2: Transmission control gating system of the WTSN

Table 2.1: WTSN cycle and time slot scheduling mapping numerology

Cycle n	0	1	2	3	4	5	6
TS n							
0	4	8	16	32	64	128	256
1	2	4	8	16	32	64	128
2	1	2	4	8	16	32	64
3	0	1	2	4	8	16	32
4	0	0	1	2	4	8	16
5	0	0	0	1	2	4	8
6	0	0	0	0	1	2	4
7	0	0	0	0	0	1	2

One additional challenge is to distribute the schedule information from the CNC to the wireless end devices in a compact way, without increasing the control traffic. To achieve this, we determine a set of allowed cycle lengths that can be written in the format $2^n * 512\mu s$, where n is the value that is distributed to the end devices to reduce the amount of bits used. Similarly, the allowed time slot lengths to be used are written in the form $2^n * 128\mu s$ and is a submultiple of defined cycle length. Then the information to be sent to the end device includes the n of the time slot and the time slot ID, defining the position of the time slot inside the cycle. The numerology of such scheduling information is given in Table 2.1, where cycle and time slot n are given as well as the number of time slots per cycle.

2.4 Results

Using the wireless TSN building blocks presented in Section 2.3, a preliminary assessment of scheduling strategies for uplink traffic is performed.

As demonstrated by different works, TSN scheduling to guarantee bounded latency and high reliability is a complex problem [15, 16]. Considering every node has its own requirements, with diverging traffic flow priorities, plus a constantly changing environment and limited time resources, it is important to devise strategies to optimize time slot usage.

As not all traffic flows have high priority, and TSN aims for coexistence between time-sensitive and best-effort traffic, time slot sharing becomes a clear option to optimize resource usage. However, not only the traffic priority plays a role in the sharing/not-sharing decision. A wireless end node has variables such as the link data rate, application packet size, traffic types, assigned time slot size, and others that should also be taken into consideration.

In case of dedicated time slots, there is only a single node transmitting in the time slot and contention mechanisms like CW, NAV, $DIFS$, or $EIFS$ are not required anymore. Hence, equation 2.1 presents the necessary time for transmitting a 118 bytes packet at 65 Mbps link rate using IEEE 802.11n:

$$\begin{aligned} Total_D &= T_{DATA} + T_{SIFS} + T_{ACK} \\ Total_D &= 14.52\mu s + 13\mu s + 93.33\mu s \\ Total_D &= 120.85\mu s \end{aligned} \tag{2.1}$$

As shown in Equation 2.1, T_{ACK} presents the largest delay. This is because the ACK, as all control packets, is transmitted at the lowest data rate, in this case 6 Mbps.

Alternatively, a shared time slot requires the aforementioned contention mechanisms. Besides this, the variable nature of the channel or hidden node problem might introduce collisions, which lead to bigger delays due to re-transmissions and back-off.

$$\begin{aligned} Total_S &= T_{DIFS} + T_{DATA} + T_{SIFS} + T_{ACK} + CW \\ Total_S &= 28\mu s + 14.52\mu s + 13\mu s + 93.33\mu s + CW \\ Total_S &= 148.85\mu s + CW \end{aligned} \tag{2.2}$$

Equation 2.2 shows the delay of a single uplink packet transmission in a shared time slot. Again, IEEE802.11n is used with a packet of 118 bytes. In comparison with Equation 2.1, $DIFS$ and CW are added. In this Equation, CW represents back-off. From $Total_D$ and $Total_S$ we already notice the theoretical advantage of a dedicated time slot to fit more packets per cycle. Having introduce both scheduling options, the measurements comparing these two scenarios are presented in the following subsections.

2.4.1 Setup Description

As introduced in Section 2.3, the openwifi IEEE802.11/Wi-Fi baseband chip/FPGA design, along with the TSN extensions, is used on top of the ADRV9361-Z7035 SDR, which combines the Analog Devices AD9361 integrated RF Agile Transceiver and the Xilinx Z7035 Zynq®-7000. An Ethernet interface is added using the carrier card ADRV1CRR-BOB/FMC [12].

The infrastructure mode wireless network, as shown in Figure 2.3, consists of three nodes: one access point (AP) and two stations, all of which are connected to a PC acting as the CNC for the WTSN.

For the wireless environment, a Wi-Fi channel 36 with 20MHz bandwidth is used. No other nodes are using this channel or any overlapping channels during the experiments. Hence, no external interference is considered during the experiments.

2.4.2 Shared vs. Dedicated Time Slot Measurements

During the measurements all the network nodes are synchronized using PTP over Wi-Fi [13]. The AP is set to be the PTP master node, while the stations are PTP slave nodes. Furthermore, as shown in Figure 2.4, PTP traffic, UL and DL, is confined to a different time slot inside the communication cycle, not to interfere with the data traffic.

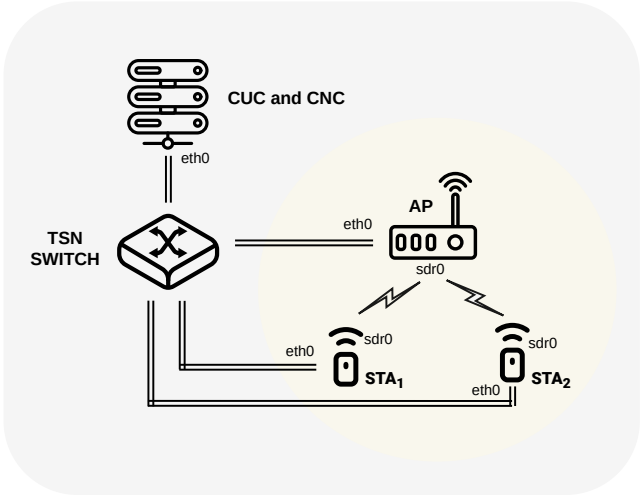
The selected cycle length is $65535\mu s$, and the time slot (TS) size ranges from $128\mu s$ to $1024\mu s$. In line with the theoretical calculations in Equations 2.1 and 2.2, a fixed data rate of 65 Mbps is used with IEEE802.11n. The injected traffic emulates one of the typical time-sensitive applications, namely machine control traffic. A total of one thousand UL packets are periodically generated every 1 ms for every experiment.

In the non-shared measurements, all contention mechanisms are disabled, hence once a TS is available and the client has a packet queued, it will transmit immediately. Figure 2.5 shows the results of mode values for both measurements.

In Figure 2.5, the purple bar represents the number of packets that arrived at the AP from a unique client not sharing its TS. As expected, the longer the TS, the higher the number of arrivals. The blue and orange bars represent the number of arrived packets at the AP when two clients, (C_1 and C_2), share the TS. Finally, the yellow bar represents the addition of both clients' arrivals in the shared TS experiment.

It is clear from Figure 2.5 how the contention produced by both clients sharing the TS, decreases the number of packets transmitted and thus received by the AP compared with the non-shared case. It is also noticeable how this difference becomes bigger when the TS size increases.

Complementary to this, it is also clear that Equations 2.1 and 2.2 certainly reflect the behavior captured with the experiments.



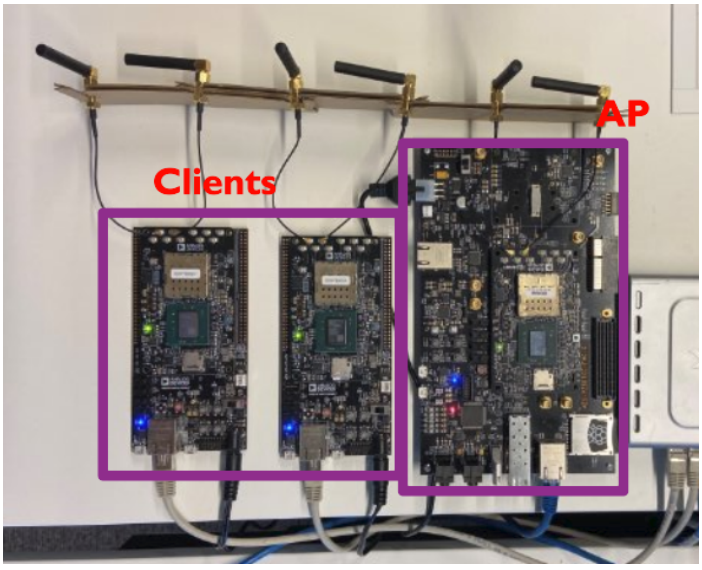


Figure 2.3: Hardware and topology of the WTSN testing setup

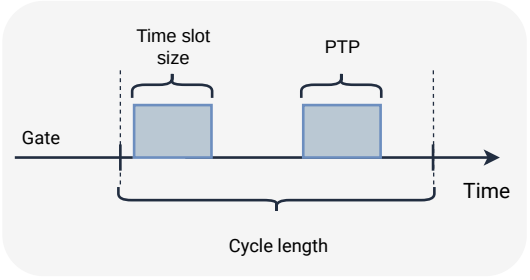


Figure 2.4: Measurements time slot size and PTP transmission schedule

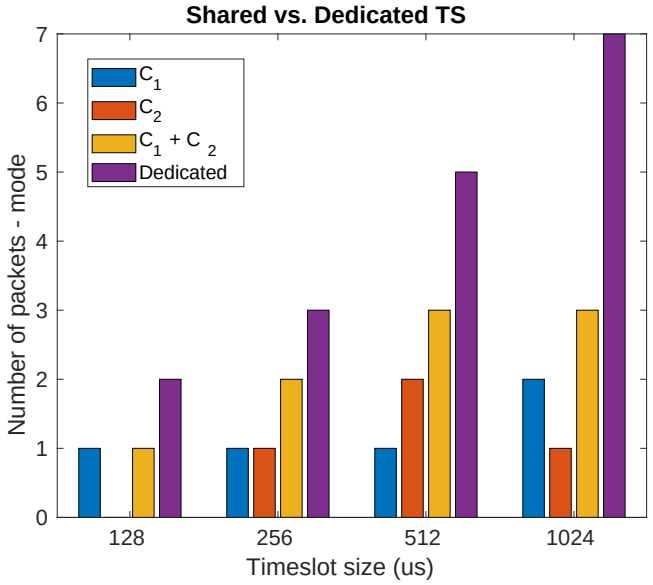


Figure 2.5: Comparison of the number of received packets per cycle, most frequent

Table 2.2 presents a comparison of cycle throughput (CT) for both experiments using the number of received packets at the AP - mode. CT was calculated considering UL packet sizes of 118 bytes and a cycle length of $65535\mu s$. As this is almost completely dependent on the number of packets, the tendency is the same as in Figure 2.5.

After examining the numbers of Table 2.2, it might be concluded that the CT is rather low compared with current wireless communication systems. However, the experiment's schedule represents an extreme case. In a normal scenario, higher data rates, bandwidths, and TS would be used, as well as smaller cycles and more TS per

Table 2.2: Cycle throughput measurement results with different time slot sizes

	Shared (bps)			Dedicated (bps)
TS size (us)	C_1	C_2	$C_1 + C_2$	
128	14.404	0	14.404	28.808
256	14.404	14.404	28.808	43.212
512	14.404	28.808	43.212	72.02
1024	28.808	14.404	43.212	100.828

cycle, increasing the achievable throughput. In this experiment the assigned time percentage ranged from 0.19% for TS 128 μs case to 1,56 % for the TS 1024 μs case. Such small percentage of air time clearly show that the achieved CT depends on the assigned schedule.

To describe how the transmission schedule relates to throughput, Equation 2.3 defines the link throughput. In this formula, TS denotes the timeslot size, DR the link data rate, and GCL_{Cycle} the cycle length.

$$TP_{TSN} = (TS * DR)/GCL_{Cycle} \tag{2.3}$$

2.5 SMT-Based Traffic Scheduling in TSN

Once the relationship between time slot sizes and frame sizes, as well as the advantages and disadvantages of shared and dedicated time slots, has been identified, the next step is to apply this knowledge to a wired-wireless TSN traffic scheduler. The goal is to better understand the behavior and limitations imposed by the wireless channel, considering available resources and application requirements.

2.5.1 Introduction to SMT Scheduling

Currently, there is a plethora of traffic scheduling solutions designed to achieve various objectives, such as enforcing tight latency boundaries, constructing constraints, ensuring requirement feasibility, and addressing traffic classes. These approaches can be categorized into approximate and exact methods. In the approximate category, heuristic methods, genetic algorithms, and artificial intelligence are the most commonly used. In contrast, exact methods include techniques such as integer linear programming (ILP), Satisfiability Modulo Theories (SMT), and constraint programming [17–19].

However, these solutions share a key limitation: They mainly address wired networks. Moreover, they fail to consider critical key performance indicators (KPIs). These indicators include reconfiguration delays, the effect of wireless time slot

sizes on integrated wired-wireless schedules, and the dynamic transmission rate impact.

Hence, we decided to implement a scheduler that considers these limitations based on SMT. As the name says, the SMT primary purpose is to determine whether a formula is satisfiable, meaning, whether there exists a set of variable assignments that make the formula true. Initially, SMT focused on Boolean equations. Over time, SMT has evolved to handle more complex data types, or ”theories,” such as arithmetic, arrays, vectors, and others. This versatility makes it applicable to a wide range of practical problems, including areas like automated planning, task scheduling, and formal verification tasks in software and hardware systems [20].

The basic approach of SMT solvers involves assigning truth values to variables, checking for conflicts, and systematically trying different assignments. The process continues until a solution is found or the solver proves that no solution exists, known as unsatisfiability.

2.5.2 Scheduler Implementation Description

Now that we know SMT is effective for scheduling problems, the next questions are: What exactly do we require from a scheduler, and how can SMT be applied to TSN scheduling problem? The general goal is to manage both wired and wireless TSN, where multiple traffic flows have specific requirements, such as low latency or guaranteed delivery. The scheduler's role is to determine the optimal timeslot allocation at each network node to meet these requirements. Since SMT operates with formulas, we must translate the network conditions and scheduling requirements into mathematical constraints.

For example, consider a traffic flow that must pass through three nodes with the lowest possible end-to-end latency. To achieve this, transmission timeslots must be allocated sequentially at each node. The SMT formula for the second timeslot, for instance, would depend on the position of the first timeslot and its duration. In the case of wireless networks, this duration includes not only the transmission time but also the acknowledgment (ACK) time and potential retransmissions.

Using Python's Z3 SMT library[2], the translation from network conditions to formulas was implemented, as shown in Figure 2.6. In this implementation, a graph defines the network structure, and the traffic flow paths are pre-defined. The following features were implemented:

- Describes a TSN with nodes and links, where each node has one or more wired or wireless interfaces and a processing delay.

- Defines traffic flows (ID, frame size, priority, maximum end-to-end latency).

[2]https://github.com/Z3Prover/z3

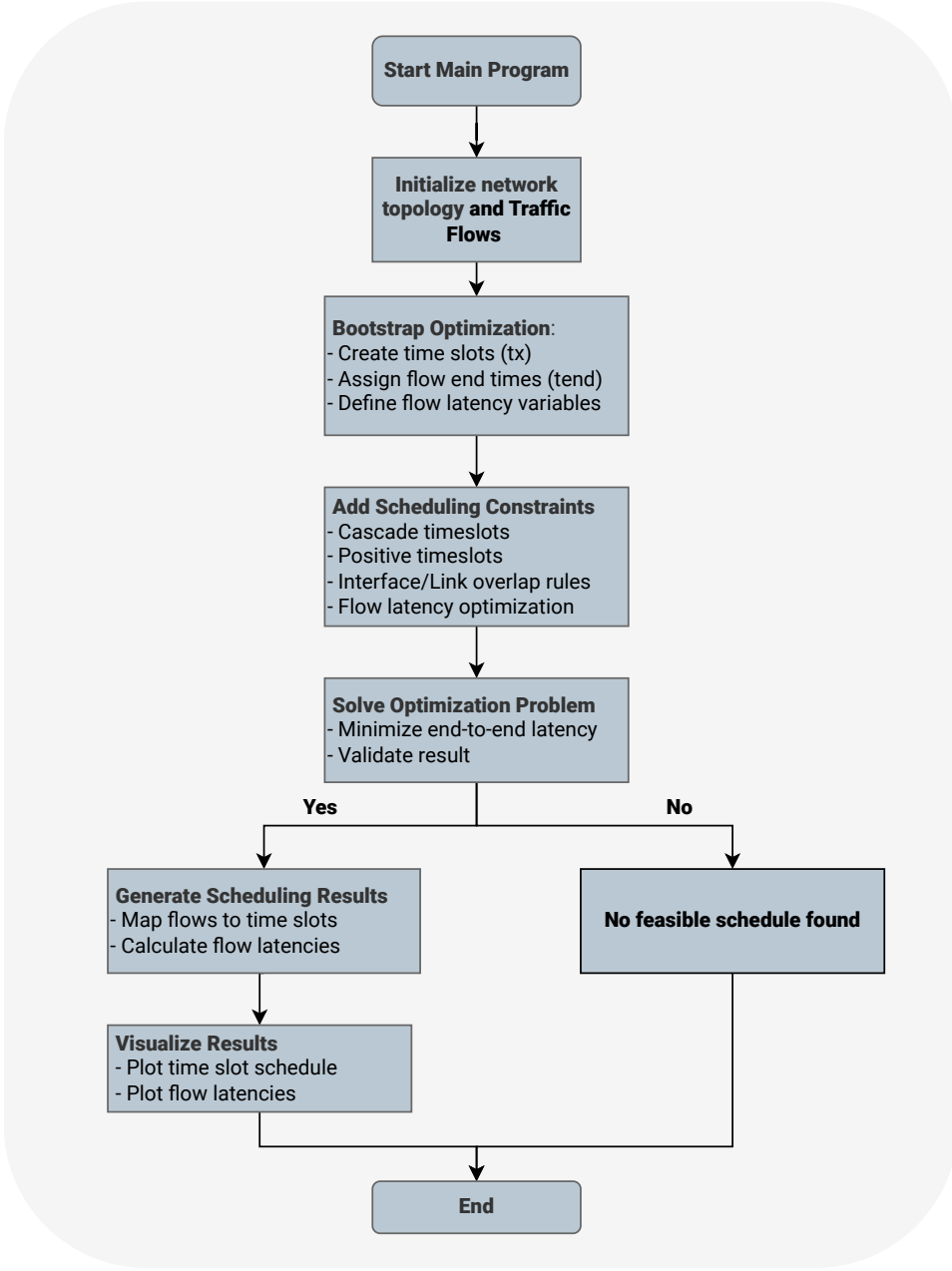


Figure 2.6: General code flow diagram of the SMT-based TSN scheduler

- Calculates time slot sizes using the flow frame size, interface transmission rate, processing delay, and interface type.

- Considers full duplex capability of wired links and half duplex in wireless links.

Once the SMT formulas are created, the optimization problem focuses on minimizing each flow's end-to-end latency, i.e., the time from the beginning of the first timeslot to the end of the last timeslot (t_{end}). The objective is also to reduce the t_{end} value relative to the start of the GCL cycle.

2.5.3 Results and Analysis

To test the described TSN scheduler, the network topology shown in Figure 2.7 was used. This topology represents a basic but common wired-wireless TSN, suitable for controlling an industrial process that requires mobility, such as a rover.

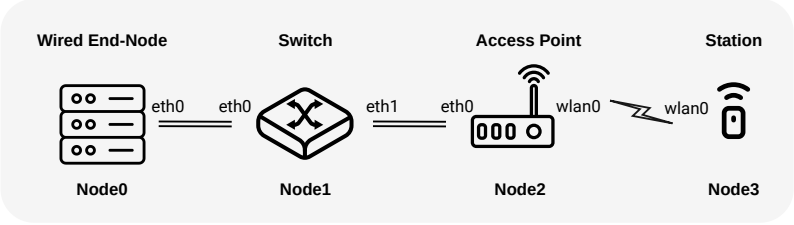


Figure 2.7: SMT scheduler testing with wired-wireless TSN topology

As shown in Figure 2.8, the initial test involved scheduling one uplink (UL) and one downlink (DL) flow in TSN. Each color represents a different traffic flow. The vertical axes show the Nodes and interfaces and the horizontal represents the time. A fixed frame size of 1000 bytes was used. For time slot size calculation, a 1 Gbps transmission rate was assumed for the wired network and 26 Mbps for the wireless part of the network. The node processing delay was considered for both cases. For the wireless part, a 13μs SIFS value was applied, and an acknowledgment (ack) duration of 93.33μs was used, based on the standard, assuming the lowest transmission rate of 6.5 Mbps for its transmission.

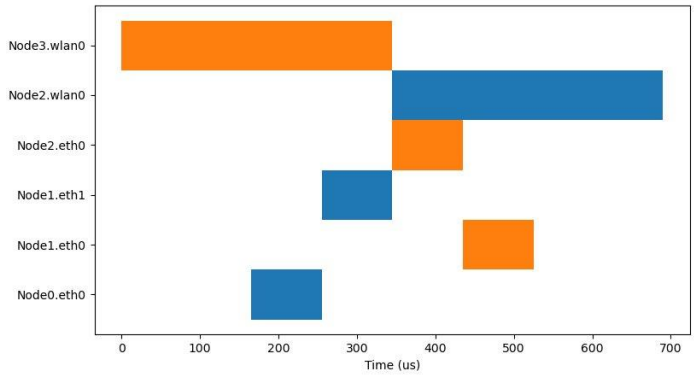


Figure 2.8: Schedule results for one uplink and one downlink traffic flow using the testing topology

2.5.3.1 Transmission Rate

Figures 2.9 and 2.10 represent tests with different transmission rates in the wireless part of the TSN. In the first case, a typical rate of 26 Mbps was used, while in the second case, a rate of 100 Mbps was applied. It is notable that the maximum t_{end} differs significantly between the two cases. In the first case, t_{end} is approximately 10 ms (from 0 to the last time slot), whereas in the second case, it is around 4 ms. Additionally, in the first case, many time resources in the wired part remain unused due to the lower transmission rate of the wireless part.

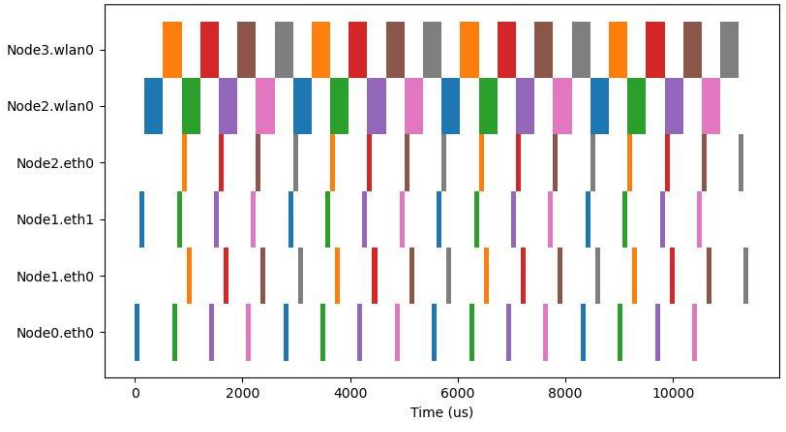


Figure 2.9: Schedule results with 28 flows (14 downlink and 14 uplink) at 26 Mbps in the wireless section using the testing topology

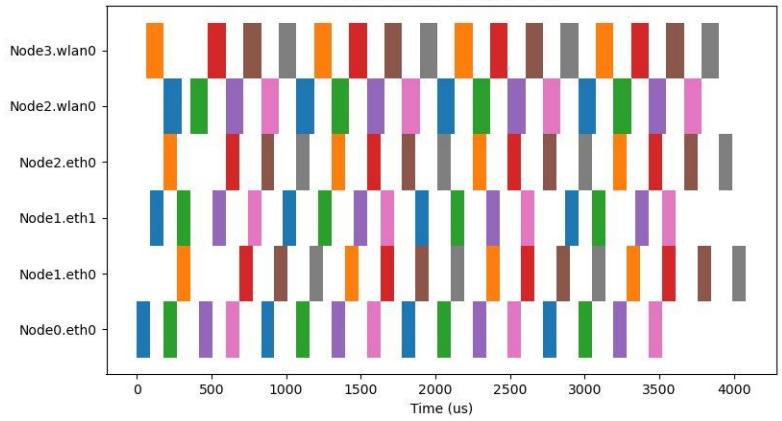


Figure 2.10: Schedule results with 28 flows (14 downlink and 14 uplink) at 100 Mbps in the wireless section using the testing topology

2.5.3.2 Reconfiguration Delay

Another important KPI is the reconfiguration delay. Depending on the application, the wireless channel conditions might change, altering the transmission rate of the wireless part. Thus, a key characteristic of a scheduler is its ability to quickly update the schedule. Figure 2.11 presents a measurement of the reconfiguration delay for the implemented SMT scheduler. The horizontal axis represents the number of flows, while the vertical axis shows the delay in seconds.

As shown, for 16 traffic flows, the delay increases to nearly 2 seconds. This result highlights the primary weakness of exact scheduling methods when addressing NP-hard problems.

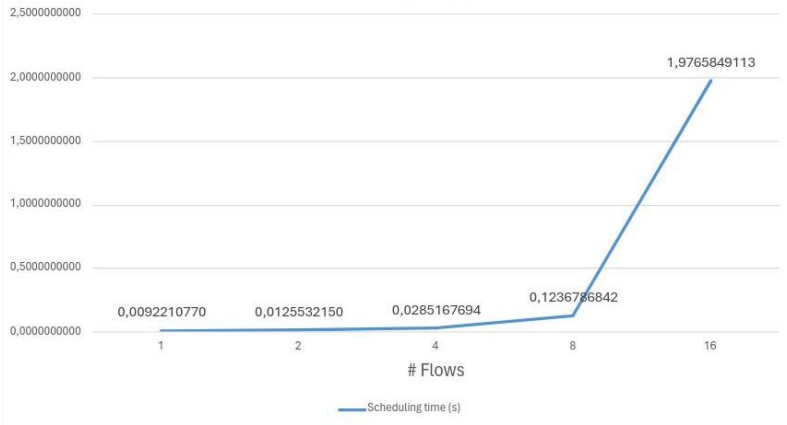


Figure 2.11: SMT scheduler reconfiguration delay versus number of traffic flows

2.6 Scheduling Demonstration

This section presents the demo's design and implementation. Its goal is to illustrate the advantage of time-sensitive traffic protection. To achieve this, it presents two traffic flows, a time-sensitive and a best effort. Then, scheduling them in dedicated time slots demonstrates WTSN's capabilities. However, when the time slot is shared, no guarantees are possible.

2.6.1 Data Generation

In this demo, machine control traffic is generated by means of a proportional integral derivative (PID) control loop feedback. The PID control aim is to balance a ball in the middle of a canal using an infrared distance sensor and a stepper motor as an actuator. A PID control is defined by three parts: the controller, the sensor, and the process. In this demo, the sensor and the controller are placed in two different Raspberry Pi (RPI). The first RPI is in charge of sampling measurements from the sensor every 10ms, which gives information about the ball position. Then, this information is included in a UDP packet and transmitted to the second RPI through the WTSN as shown in Figure 2.12. The second RPI receives the ball position, calculates the error, and applies a PID correction to the canal angle using a stepper motor.

Furthermore, a second traffic flow is produced between both RPI-s to account for other background traffic in the network. This dummy traffic is also UDP traffic, generated using $iperf$, whose function is to show the time multiplexing capabilities of the demo.

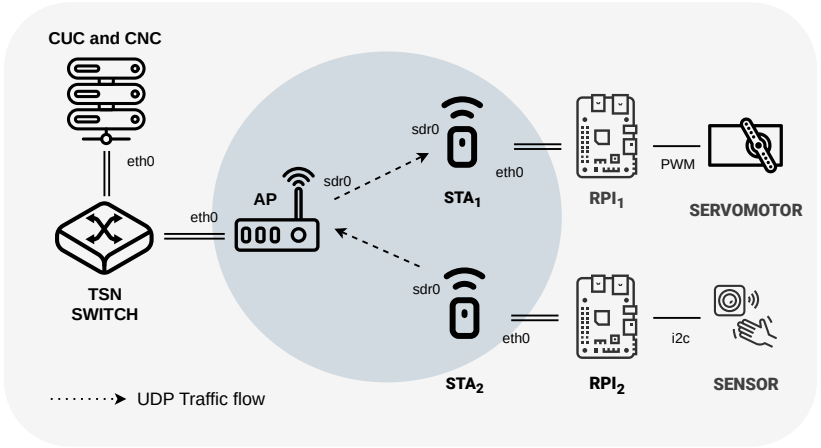


Figure 2.12: Topology of the balance ball demo setup

2.6.2 Data Forwarding

Time-sensitive networks are originally layer 2 networks where data is forwarded at layer 2. We kept the same concept for the WTSN, by offering the data forwarding on layer 2 between different wireless nodes. As such, the WTSN will behave as layer 2 bridge between the two RPI.

Each wireless node has a schedule for each queue. The schedule consists of a cycle length of 8.192 ms, with 16 time slots of 512 μs each. The distribution of the time slots inside the cycle for each queue in each node is shown in Figure 2.13. All PTP and wireless control traffic is assigned to queue 0. Time critical traffic (PID traffic) is assigned to queue 1, background traffic ($iperf$ traffic) is assigned to queue 2 while queue 3 is used for publishing the monitored information from the end nodes to the CNC. The PID control traffic is sourced at client 1 with destination client 2. Thus, the time slots of queue 2 on each node (client 1, AP, client 2) follow each other to give the shortest latency for the traffic. By giving the dedicated time slots for time critical traffic we expect to have the best latency performance for the time critical traffic. In the second case, time slots of the time critical traffic and best-effort traffic are shared, to show the performance in a normal shared Wi-Fi network.

2.6.3 Telemetry

Network performance monitoring is done using In-band Network Telemetry (INT) for wireless networks [21]. INT employs monitoring on a per flow basis, on per-hop and end-to-end fashion. Different wireless parameters are monitored on each

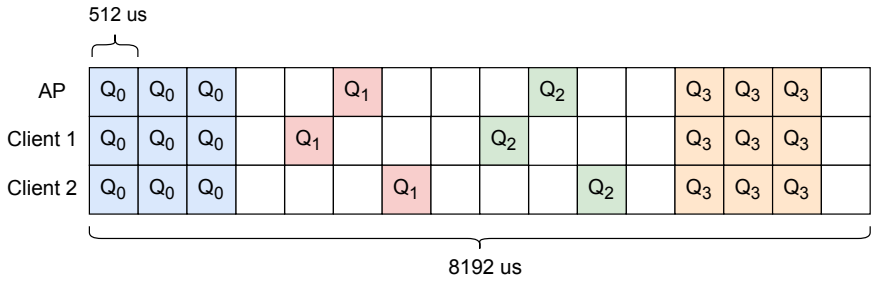


Figure 2.13: Transmission schedule applied in the balance ball demo

hope, such as: RSSI, SNR, channel used, data retransmission, data rate (DR) and Modulation and Coding Scheme (MCS) index. Most importantly, on each network hop the packet is timestamped enabling the calculation of accurate latency on each hop and end-to-end.

2.6.4 Demo Setup

The described demo is implemented on top of imec WTSN evaluation kit (EK). imec WTSN EK uses an FPGA-based Software Defined Radio (SDR) platform, $openwifi$ running on Xilinx ZC706 board and Xilinx ZedBoard for the access point (AP) and wireless clients, respectively. Figure 2.12 presents the demo setup architecture. Each RPI holds part of the PID control loop as described in Section 2.6.1, while the $openwifi$ clients serve as end nodes of the WTSN, receiving and transmitting application packets. Additionally, AP forwards packets between the clients and the network. Finally, the CNC is an essential element of a TSN whose function is to provide configuration data, such as the schedule, to TSN nodes in response to application requirements. In this demo, the CNC also receives, monitors, and presents the network telemetry data using the $Grafana$ dashboard. Finally, the complete demo picture is shown in Figure 2.14.

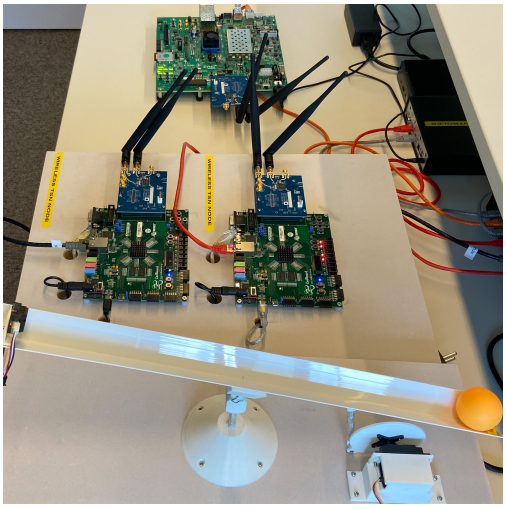


Figure 2.14: Picture of the balance ball demo setup

2.6.5 Demo Results

To demonstrate the capabilities of WTSN, two experiments are designed. The intention of such experiments is to display the potential of a WTSN compared to normal Wi-Fi to support time-sensitive traffic, specifically, PID control traffic.

2.6.6 Dedicated time slot

The first experiment uses the schedule shown in Figure 2.13. As described in Section 2.6.1, queue 1 (Q_1) is used by PID control traffic and queue 2 (Q_2) for $iperf$ traffic. With this schedule, both traffic flows have a transmission opportunity every 8192 μs, meeting the time-sensitive application latency requirements. Furthermore, management and telemetry traffic are scheduled in queues 0 and 3 respectively, leaving guard bands to avoid disturbing time-sensitive traffic. Figure 2.15 depicts the end-to-end latency of machine control traffic (red), and $iperf$ traffic (blue). As seen, both latencies keep under $\sim$ 10 ms. This is also visible by the stability of the ball in the middle of the canal.

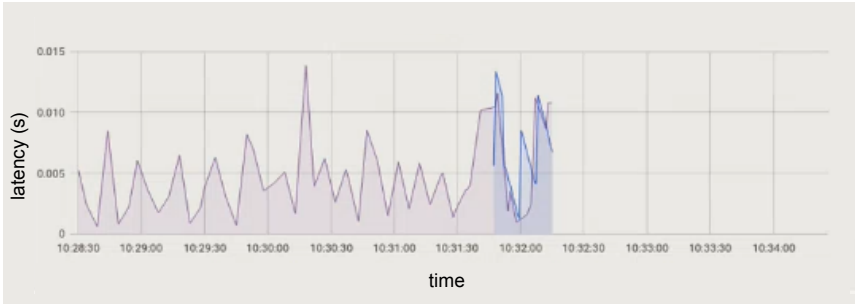


Figure 2.15: Grafana visualization of machine-control and iPerf traffic end-to-end latency with dedicated time slots

2.6.7 Shared time slot

In a non-WTSN case using CSMA/CA as IEEE802.11, all nodes compete for the channel before transmitting. When high traffic or high node density is present, best-effort networks do not offer latency guarantees. To represent this scenario, both machine control, and $iperf$ traffic are set to share Q_1. The result is shown in Figure 2.16. As both traffic flows compete for the same time resource, latency is unpredictable any more (rising up to 1.5 s). The effect is also visible in the balance of the ball. As control traffic is highly time-sensitive, packet delay makes it impossible to keep the ball balanced.

Figure 2.16: Grafana visualization of machine-control and iPerf traffic end-to-end latency with shared time slots

2.7 Conclusion

In this chapter, we presented some of the main challenges that need to be overcome to achieve a seamless Ethernet/Wi-Fi TSN network. Despite the industry is still skeptical that the same performance level in synchronization, latency, security, and

reliability can be achieved over wireless, TSN features on top of Wi-Fi are bringing wireless closer to this deterministic vision.

Still, additional research effort is required in features such as scheduling in a wireless environment. The variable nature of the channel together with the limited amount of resources requires adaptable mechanisms that are not only able to allocate time-sensitive but also best-effort traffic. Then, it is important to consider instruments such as time slot sharing for lower priority streams.

Furthermore, an initial SMT-based scheduler was evaluated, demonstrating its capability to schedule traffic flows in a wired-wireless TSN. However, the evaluation also highlighted significant challenges. These include the lower data rates of wireless networks compared to wired networks, the half-duplex nature of wireless communication, and the variable transmission rates that necessitate rapid reconfiguration, a requirement where solutions like SMT often fall short.

Using our TSN features built on top of openwifi, a small-scale WTSN testbed was built and measurements were conducted to compare shared and dedicated time slots. Together with a complementary theoretical analysis, it was demonstrated how contention produced by CSMA/CA in a shared time slot, directly impacts the received throughput.

References

[1] M. Nixon. *A Comparison of WirelessHART and ISA100.11a*. 2012.

[2] M. Chwalisz and A. Wolisz. *Towards efficient coexistence of IEEE 802.15.4e TSCH and IEEE 802.11*. IEEE/IFIP Network Operations and Management Symposium: Cognitive Management in a Cyber World, NOMS 2018, pages 1–7, 7 2018. doi:10.1109/NOMS.2018.8406146.

[3] T. Adame, M. Carrascosa-Zamacois, and B. Bellalta. *Time-Sensitive Networking in IEEE 802.11be: On the Way to Low-Latency WiFi 7*. Sensors, 21(15):4954, 7 2021. Available from: https://www.mdpi.com/1424-8220/21/15/4954, doi:10.3390/s21154954.

[4] J. Haxhibeqiri, X. Jiao, M. Aslam, I. Moerman, and J. Hoebeke. *Enabling TSN over IEEE 802.11: Low-overhead Time Synchronization for Wi-Fi Clients*. In Proceedings of the IEEE International Conference on Industrial Technology, volume 2021-March, pages 1068–1073. Institute of Electrical and Electronics Engineers Inc., 3 2021. doi:10.1109/ICIT46573.2021.9453686.

[5] A. Mahmood, R. Exel, and T. Sauter. *Performance of IEEE 802.11's Timing Advertisement Against SyncTSF for Wireless Clock Synchronization*. IEEE Transactions on Industrial Informatics, 13(1):370–379, 2 2017. doi:10.1109/TII.2016.2521619.

[6] I. V. Óscar Seijo, Jesús A. López-Fernández, and Manuel Vélez. *IEEE 1588 Clock Synchronization Performance over Time-Varying Wireless Channels*. 2018.

[7] D. Coleman and D. Westcott. *CWNA Certified Wirelesss Network Administrator ™ Official Study Guide*. Brandon A. Nordin, first edition, 2021.

[8] S. Zehl, A. Zubow, and A. Wolisz. *hMAC: Enabling Hybrid TDMA/CSMA on IEEE 802.11 Hardware*. (November), 2016. Available from: http://arxiv.org/abs/1611.05376.

[9] O. Seijo, J. A. Lopez-Fernandez, and I. Val. *W-SHARP: Implementation of a High-Performance Wireless Time-Sensitive Network for Low Latency and Ultra-low Cycle Time Industrial Applications*. IEEE Transactions on Industrial Informatics, 17(5):3651–3662, 2021. doi:10.1109/TII.2020.3007323.

[10] O. Seijo, J. A. Lopez-Fernandez, H. P. Bernhard, and I. Val. *Enhanced Timestamping Method for Subnanosecond Time Synchronization in IEEE 802.11 over WLAN Standard Conditions*. IEEE Transactions on Industrial Informatics, 16(9):5792–5805, 9 2020. doi:10.1109/TII.2019.2959200.

[11] J. Haxhibeqiri, X. Jiao, E. Municio, J. M. Marquez-Barja, I. Moerman, and J. Hoebeke. *Bringing Time-Sensitive Networking to Wireless Professional Private Networks: Filling Gaps and Bridging the Innovation*. Wireless Personal Communications, 121(2):1255–1271, 2021. doi:10.1007/s11277-021-09056-0.

[12] X. Jiao, W. Liu, M. Mehari, M. Aslam, and I. Moerman. *Openwifi: A free and open-source IEEE802.11 SDR implementation on SoC*. IEEE Vehicular Technology Conference, 2020-May, 5 2020. doi:10.1109/VTC2020-SPRING48590.2020.9128614.

[13] M. Aslam, W. Liu, X. Jiao, J. Haxhibeqiri, G. Miranda, J. Hoebeke, J. Marquez-Barja, and I. Moerman. *Hardware Efficient Clock Synchronization Across Wi-Fi and Ethernet-Based Network Using PTP*. IEEE Transactions on Industrial Informatics, 18(6):3808–3819, 6 2022. doi:10.1109/TII.2021.3120005.

[14] *IEEE 802.1Qbv-2015 - IEEE Standard for Local and Metropolitan Area Networks - Enhancements for scheduled traffic : bridges and bridged networks*. page 57, 2016.

[15] V. Gavrilut and P. Pop. *Scheduling in time sensitive networks (TSN) for mixed-criticality industrial applications*. IEEE International Workshop on Factory Communication Systems - Proceedings, WFCS, 2018-June:1–4, 7 2018. doi:10.1109/WFCS.2018.8402374.

[16] A. M. Kentis, M. S. Berger, and J. Soler. *Effects of port congestion in the gate control list scheduling of time sensitive networks*. Proceedings of the 2017 8th International Conference on the Network of the Future, NOF 2017, 2018-Janua:138–140, 7 2017. doi:10.1109/NOF.2017.8251236.

[17] Q. Li, D. Li, X. Jin, Q. Wang, and P. Zeng. *A Simple and Efficient Time-Sensitive Networking Traffic Scheduling Method for Industrial Scenarios*. Electronics 2020, Vol. 9, Page 2131, 9:2131, 12 2020. Available from: https://www.mdpi.com/2079-9292/9/12/2131/htmhttps://www.mdpi.com/2079-9292/9/12/2131, doi:10.3390/ELECTRONICS9122131.

[18] H. Chahed and A. Kassler. *TSN Network Scheduling—Challenges and Approaches*. Network, 3:585–624, 12 2023. Available from: https://urn.kb.se/resolve?urn=urn:nbn:se:kau:diva-98183, doi:10.3390/NETWORK3040026.

[19] L. D. Moura, B. Dutertre, and N. Shankar. *A Tutorial on Satisfiability Modulo Theories (Invited Tutorial)*. 2007. Available from: http://research.microsoft.com/leonardo/http://www.csl.sri.com/œ\{bruno,shankar\}/.

[20] A. Biere, M. Heule, H. V. Maaren, T. Walsch, C. Barrett, R. Sebastiani, S. A. Seshia, and C. Tinelli. *Handbook of Satisfiability Satisfiability Modulo Theories*. 2008.

[21] J. Haxhibeqiri, P. H. Isolani, J. M. Marquez-Barja, I. Moerman, and J. Hoebeke. *In-Band Network Monitoring Technique to Support SDN-Based Wireless Networks*. IEEE Transactions on Network and Service Management, 18(1):627–641, 3 2021. doi:10.1109/TNSM.2020.3044415.

3

Impactless Association Methods for Wi-Fi-based Time-Sensitive Networks

"In quoting others, we cite ourselves."

–Julio Cortázar

Although most standardization efforts focus on wired domains, recent advancements in wireless communications are extending time-sensitive capabilities to wireless networks. To achieve full integration with the wired domain, addressing wireless procedures such as association is critical. This chapter presents three beacon-based, low-overhead time synchronization mechanisms that include dynamic association time slot information. These mechanisms enable wireless client devices to join the network deterministically without disrupting time-sensitive traffic. A proof-of-concept implementation and experiments were conducted on an IEEE 802.11-based wireless testbed.

⋆ ⋆ ⋆

This Chapter is based on:
Impactless Association Methods for Wi-Fi based Time-Sensitive Networks

Published in Springer Wireless Networks, Feb. 2024.

⋆ ⋆ ⋆

Abstract

Current safety-critical applications, found in domains such as automotive networking, automated industries, and control systems, require deterministic and latency-bounded end-to-end communication, currently not supported by best-effort networks. Time-Sensitive Networking (TSN) is the most feasible approach to fulfill such requirements. Originating from the wired domain, their application to the wireless domain is now attracting attention. Besides some advances in time synchronization and scheduling, essential policies such as association with a wireless TSN (WTSN), have not been explored. This chapter analyses the necessary tools for a prospective client to accomplish association with the WTSN without affecting ongoing time-sensitive traffic. We introduce three beacon-based methods to pre-synchronize and pre-schedule nodes willing to associate with the network. On top of the first full-stack Software Defined Radio (SDR) based IEEE802.11/Wi-Fi design, the proposed approaches are developed and tested in a real-world environment. A delay analysis shows a worst-case association delay of up to 1.8s, and high accuracy synchronization on prospective clients' association frame receptions, with 100% of on-time arrivals, even with challenging scheduling timeslots as short as 128 μs.

3.1 Introduction

Apart from bringing existing TSN components to the wireless world, it is also imperative to consider primary wireless network processes such as bootstrapping. Bootstrapping refers to the process of establishing initial network connectivity, including association and configuration on a wireless device. It involves setting up the necessary parameters and protocols to enable wireless communication and access to the network. This process becomes particularly critical in a TSN environment, as the network must ensure uninterrupted traffic flow while the bootstrapping process takes place. In 802.11, prospect clients will use Clear Channel Assessment (CCA) before transmitting to avoid collisions; however, this is not fully compatible with WTSN. As it is shown in Figure 3.1, the WTSN has assigned time slots to the clients. In the case of a new client wanting to associate, two possibilities may arise. On the one hand, if the TSN Clients use CCA, it will delay its transmission, increasing its latency and hence jitter. On the other hand, if TSN Clients do not use CCA, a collision will take effect, thus retransmission will be required. In both cases, determinism is affected, and the network is no longer capable of providing bounded latency.

As previously mentioned, achieving high reliability and bounded latency are key objectives of TSN. Therefore, the network must offer compliant solutions for fundamental processes like bootstrapping to ensure these goals are met. Our strategy

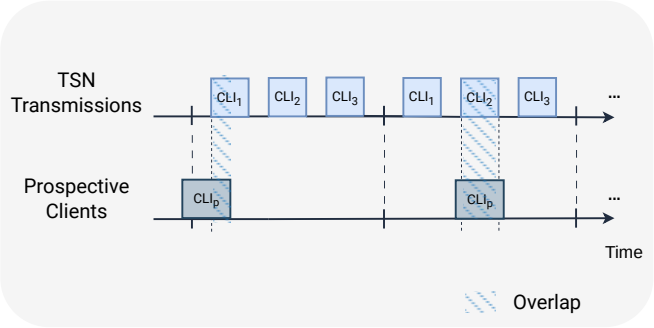


Figure 3.1: Representation of wireless association collision with TSN traffic

to solve this problem is to grant TSN features to clients even before they achieve a connection with the network. By using beacons, prospect clients are designated a time slot and a good enough time synchronization mechanism, avoiding overlaps on other client transmissions and keeping compatibility with IEEE802.11. As an extension of our previous work in [1] and [2], two new pre-synchronization mechanisms both compatible with Access Point scheduled transmissions have been implemented. Furthermore, information in beacons is encoded to reduce its size, and an analysis of the impact of WTSN association over ordinary association has been appointed. Also, a second prospective client has been included in the tests to clearly demonstrate the pre-schedule concept. To the best of our knowledge, we are the first to give a functional solution to the association in a Wi-Fi-based WTSN. This solution was developed and tested on top of a TSN-enhanced version of *openwifi*, a Linux mac80211 compatible full-stack IEEE802.11/Wi-Fi design based on Software Defined Radio (SDR) [3].

The chapters's specific contributions can be summarized as follows:

- We present a highly efficient association procedure that is fully compatible with TSN (Time-Sensitive Networking) and specifically designed for Wi-Fi networks.
- As part of this association procedure, we suggest utilizing beacon frames to enable scheduling and time synchronization for non-associated clients.
- To optimize functionality, we propose an efficient encoding scheme within the beacon frames, which allows prospective clients to determine their time slot for association.
- Lastly, we introduce three distinct time pre-synchronization methods that maintain time synchronization during the association procedure for prospective clients.

The remainder of the chapter is organized as follows. First, in section 3.2, related work on Wireless TSN association is presented, followed by a background introduction to association and the main TSN features required for association in section 3.3. Next, in section 3.4, the proposed association procedure and how it integrates into the current IEEE 802.11 association process are explained. In section 3.5, the developed beacon-based time synchronization methods are explained. In section 3.6, the results including synchronization accuracy, threshold analysis, schedule correctness, and association delays are shown. Finally, section 3.7 concludes this work.

3.2 Related Work

Before getting into the WTSN association, it is important to provide sufficient background regarding the current research in the scope of this work. Therefore, this section is divided into three parts. The first will introduce wireless broadband technologies suitable to provide TSN capabilities. Next, a review of recent WTSN implementations is given, focused on how every work tackles association. Finally, the recent advances in synchronization and scheduling with similar approaches are reviewed.

3.2.1 TSN Capable Wireless Technologies

Among the various requirements a wireless technology must include to support time-sensitive traffic, are traffic differentiation, low transmission latency, high precision in time synchronization, vendor neutrality, and reliability [4]. It is also important to consider that despite meeting the previous conditions, the biggest obstacle all wireless technologies have to overcome to properly deliver time-sensitive traffic is the variant nature of the wireless medium. Nevertheless, in this context, 5G and Wi-Fi are the clear candidates for realizing WTSN. Both technologies' standardization bodies have been continuously working towards some sort of Ultra Reliable Low Latency Communications (URLLC), directly related to the aforementioned TSN requirements.

On the 5G side, the 3GPP Release-16 introduces support to TSN applications demanding determinism and reliability. Here, for a centralized TSN controlling architecture, the 5G system becomes a transparent bridge, hiding the core and access network from the TSN. Then, control plane functions such as session management function (SMF), policy control function (PCF), and Access and Mobility Management Function (AMF), create QoS policies to allow the user traffic to be handled in a time-sensitive manner [5, 6]. Concerning the association of a new node, no big changes have been introduced in recent releases. The prospect client randomly selects a preamble to transmit association frames. The association process

takes place over the Physical Random-Access Channel (PRACH), dedicated to that purpose [7]. Hence, given the frequency domain separation, the effect of association on time-sensitive traffic may be considered non-existent.

At the IEEE802.11 or Wi-Fi side, mechanisms to better support time-sensitive traffic have long been lacking. As part of the IEEE802.11e and IEEE802.11aa amendments, the inclusion of time-critical Access Categories (AC), i.e. traffic differentiation, tries to improve the QoS of streams. However, the lack of flexible scheduling policies avoids guaranteeing time-sensitive features. Furthermore, the MAC layer of IEEE802.11 has always depended on a random access method with exponential back-off, which is unable to ensure a bounded latency because of its unpredictable nature [4, 8]. Nevertheless, with the advent of IEEE802.11ax or Wi-Fi 6, new features such as the introduction of OFDMA, the low-latency operation mode, and Target Wake Time (TWT) mechanisms provide significant latency reduction and improved efficiency [5, 9, 10]. The latest flavor, i.e. IEEE 802.11be or Wi-Fi 7, is still under definition. However, it will bring additional mechanisms such as traffic differentiation and prioritization, multi-Access Point operation, and the use of the 6 GHz band with wider 320 MHz channels to enhance transmission rates, bringing low latency and high reliability, enabling new scenarios such as TSN [11]. With respect to the association, Section 3.4 describes in detail the current procedure that has not changed nor been adapted to TSN requirements. As explained later, our work focuses on studying a TSN-capable association mechanism, that fills in the current gap.

3.2.2 WTSN Implementations

Currently, there is an important interest in implementing TSN features in the wireless world. However, most practical implementations do not consider basic processes such as association. On the contrary, most of them focus on the wired-wireless interface, synchronization, and scheduling [12–15]. There are several studies closely related to our research. For example, in the study conducted by Cruces et al. [16], they propose a TSN that incorporates a discovery and configuration stage based on Carrier Sense Multiple Access (CSMA), followed by a switch to a Time Division Multiple Access (TDMA) system during the operational stage. However, when compared to our work, their focus primarily lies on the operational stage rather than addressing the challenge of incorporating new clients into the WTSN. While their discovery stage provides a solution for the initial setup, it does not offer a comprehensive solution for integrating new clients who wish to join the WTSN during an operational stage.

Superframes are also implemented by [17] in W-SHARP, a wireless technology for real-time industrial applications. In this case, communication is performed using superframes, while time synchronization is achieved with beacons at the

beginning of the superframes. In this case, the stations estimate the time offset and pass it through a proportional-integral(PI) filter. Different from this work, we present three different means of synchronization that directly drop $late$ beacons. This helps to avoid the typical delays imposed by PI filters and leads to a faster association process. Moreover, even though the W-SHARP work has similarities with IEEE 802.11, it is not compatible with it.

In [18], the authors propose a Time-Frequency Division Multiple Access (TFDMA) based system called RETIS which implements different techniques to improve reliability and latency in wireless transmissions. Bringing the idea from the Wireless networks for Industrial Automation-Factory Automation (WIA-FA) protocol, beacons are used for time synchronization and scheduling. The authors briefly introduce how new nodes would use the information in the beacons to become part of the network, but contrary to our work there is not a clear description of how it will handle channel access delays for time synchronization or the compatibility with IEEE802.1Qbv, which defines traffic shaping in wired TSN.

3.2.3 Association-related Time Synchronization and Scheduling

As aforementioned, there are big efforts in research toward precise time synchronization. Here, works such as [19, 20] and [21] present and evaluate high-accuracy alternatives focused on wired-wireless architectures in line with 802.1AS. They explore different messaging alternatives, including beacons. However, compared with our work, they still lack solutions for when the client is still not part of the network or depends on complex and high resource consumption time-compensation algorithms.

In conclusion, several studies about time synchronization and traffic scheduling, particularly in TDMA-based systems, have been presented. However, those do not take into account an association phase that must be carried out without interfering with traffic in dynamically assigned time slots.

3.3 Wireless Association and TSN Background

This section aims to introduce the concepts related to wireless association, and TSN features necessary for association. Initially, it will explain the current IEEE 802.11 association procedure, providing a foundation for understanding the proposed modifications to make it TSN-friendly. Time synchronization and time-aware scheduling are particularly vital in a deterministic association process [22, 23]. Therefore, this section will also explore the fundamental concepts underlying these two features in both wired and wireless communications, offering a comprehensive understanding of their significance.

3.3.1 Current Association Procedure

The IEEE 802.11 standard, also known as Wi-Fi, defines a set of standards related to Local Area Networks (LAN), and protocols in Media Access Control (MAC), and Physical Layer (PHY). Among the services defined in the standard, the association is of importance at the earliest stage. Each station must enforce an association procedure to join an access point before it can communicate with other nodes in the network [24].

The first step a prospective client has to complete to associate is to select a target Access Point (AP). The selection is done by scanning the available channels. These channels depend on client configuration, hardware capabilities, and country regulations. IEEE 802.11 standard currently includes two scanning options: active and passive. For active scanning on every channel, the client first transmits a probe request frame and waits for a default dwell time of 500 ms for a probe response from APs in coverage. Opposite to this, passive scanning only listens for AP beacons on every channel for the dwell time [25].

Once the target AP has been selected, the association starts. As shown in Figure 3.2, there are three connection states in IEEE 802.11. Every state change requires a frame transmitted by the client to the AP and a response. Hence to get to State 3, the client will transmit authentication and an association request which needs to be replied to by the AP [25].

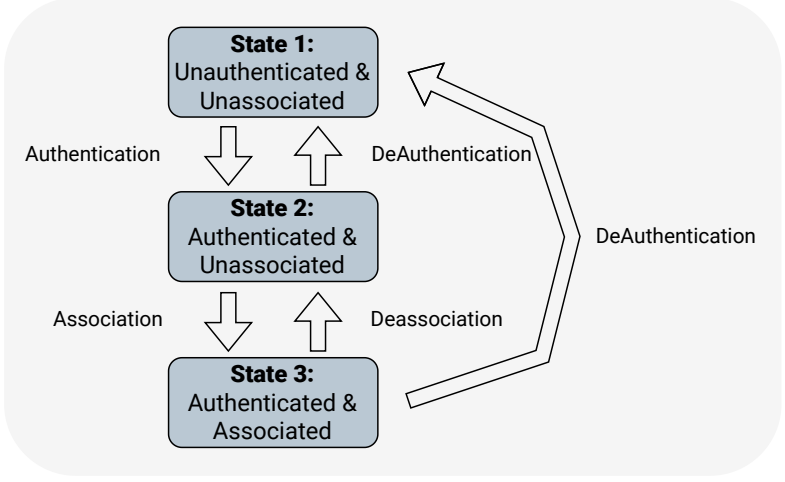


Figure 3.2: IEEE 802.11 association and authentication state machine

3.4 Beacon Based WTSN Bootstrapping

The central aspect of our proposal revolves around the utilization of beacons, which play a crucial role in introducing features such as scheduling and time synchronization to potential clients, effectively integrating them into the WTSN even before they establish a formal association. This section aims to provide an overview of how the beacons, once received by prospective clients, seamlessly integrate with the TSN during the association process. Additionally, to ensure the proper functioning of the beacons, a management system is an integral part of our solution. The role of the controller in defining the behavior of the beacons will be discussed, followed by an outline of how beacons carry pre-synchronization and pre-schedule information, which are key elements enabling the WTSN association for prospective clients.

3.4.1 IEEE 802.11 association integration to TSN

Keeping the procedure flow described in Section 3.3.1, our WTSN association method integrates with IEEE 802.11 by solely using passive scanning. This way early prospect client transmissions are avoided. Then, as shown in Figure 3.3, both authentication and association request frames are transmitted by the prospective client in a TSN-scheduled fashion, transparent to the higher layers. To achieve this, the prospective client first needs to obtain a schedule and time synchronization from the captured beacons. As these two TSN features are used only during the association stage we will call them pre-schedule and pre-synchronization from now on.

3.4.2 TSN Management for Bootstrapping

The IEEE 802.1Qcc standard [26] introduces three TSN management configuration models: centralized, decentralized, and hybrid. Among these models, our study focuses on the centralized model due to its numerous advantages, including simplified network management, enhanced resource allocation, flexibility, and scalability. In the centralized model, nodes within the network convey their application requirements to a Centralized User Controller (CUC). The CUC then communicates with a Centralized Network Controller (CNC) responsible for distributing the network resources among the nodes [27].

In terms of association, the CNC plays a crucial role by allocating association time slots to APs based on the available resources, as depicted in Figure 3.4 and implemented in this study. The association time slot is shared among all potential clients seeking to associate with an AP. Consequently, the size of the shared association time slot should be adjusted according to the anticipated number

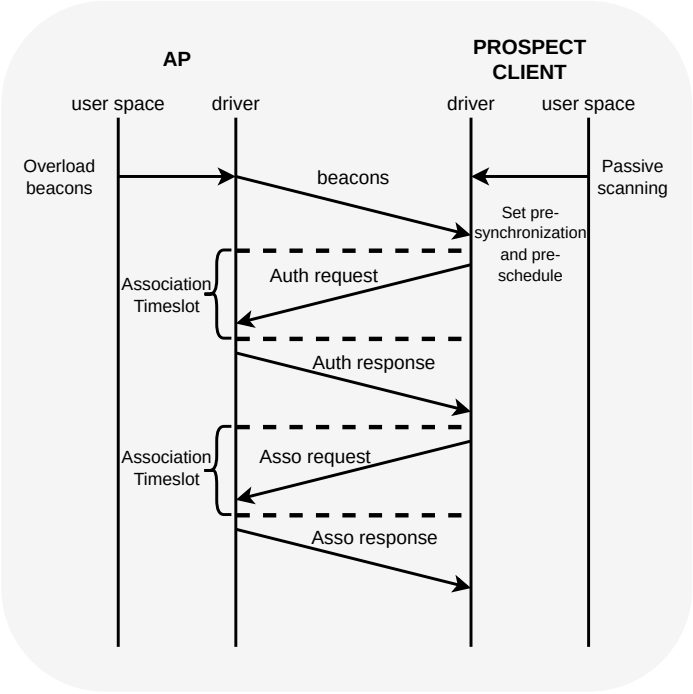


Figure 3.3: Complete timeline of the proposed association process

of associations. This ensures that sufficient time is allocated to accommodate the expected client associations effectively.

As part of the bootstrapping process, once the new client has been associated with the WTSN AP, the network will follow the typical process, providing an IP address and an enhanced time synchronization protocol such as gPTP, described in Section 1.2, will take over, ceasing pre-synchronization [28]. Lastly, the client and the network will negotiate a schedule depending on the application's needs and available resources.

3.4.3 Beacon overloading and encoding

Beacons are one of the management frames of IEEE 802.11. The AP periodically transmits beacons to announce its presence and help associated clients with time synchronization. Beacons are the only frames that are received by prospective clients and could be used to trigger an association. As shown in Figure 3.5, among other information, beacons carry the AP timestamp, which in a TSN describes the

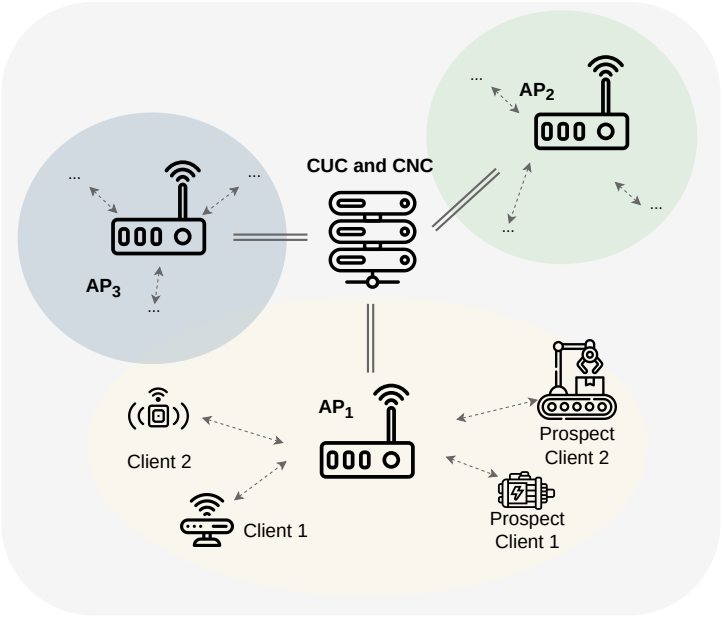


Figure 3.4: Topology for WTSN association management

network time. They also contain the beacon interval, typically configured to 100 Time Units (TU) with a TU being equal to 1.024 ms, so a beacon is expected to be transmitted every 102.4 ms. Further, capability information brings information about the type of network, support of features, and more. Finally, up to 2320 bytes of information elements can be included, part of which we will make use of [25].

As previously stated, the association requires two actors. Pre-synchronization and a pre-schedule or association time slot. Once the AP is provided with the pre-schedule, this is broadcasted to prospective clients in the vendor-element of beacons. The pre-schedule is described in four fields of the vendor element: cycle length index (j), time slot length index (k), time slot bitmap index start (w_n), and time slot bitmap index end (w_m) as seen in Figure 3.5.

Typically, manufacturers utilize the vendor element in beacons to incorporate customized information or facilitate unique functionalities within their devices. Consequently, the proposed modification to include schedule information aligns seamlessly with the standard specifications, ensuring compatibility across systems.

To reduce the number of occupied bits in the vendor-element, the pre-schedule is encoded using the following nomenclature: j and k, each with 3 bits, represent the cycle and time slot length respectively as seen in Equation 3.1. Then, the cycle length may vary from 512 to 65536 μs and the time slot duration from 128 to 16384

Table 3.1: Notation for pre-schedule encoding

Variable	Description
S	Pre-synchronization schedule carried by the beacon
j	Encoded cycle length value, where j = 0,..,7
k	Encoded time slot length value, where k = 0,..,7
w_n	Time slot position index start, where 0 $\leq w_n$ 0 $\leq (4 * 2^{j-k} - 1)$
w_m	Time slot position index end, where 0 $\leq w_m$ 0 $\leq (4 * 2^{j-k} - 1)$
C_L	Decoded cycle length value, where C_L = 512,..,65536μs
TS_L	Decoded time slot length value, where C_L = 128,..,16384μs
TS_{Start}	Decoded time slot start value
TS_{End}	Decoded time slot end value

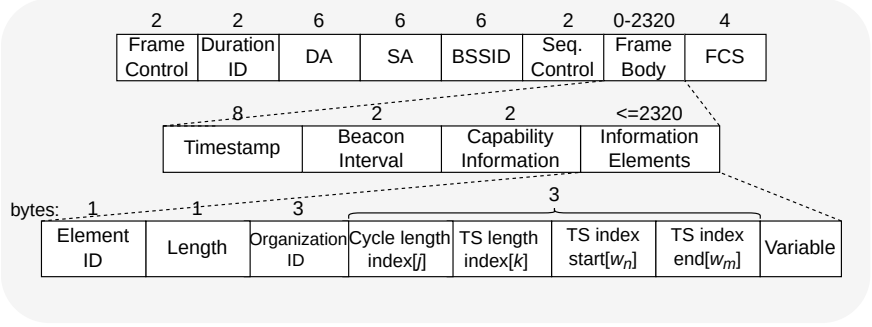


Figure 3.5: Beacon frame structure with beacon stuffing in the vendor proprietary element

μs as can be seen in Equations 3.2 and 3.3, respectively:

$$S = j[3bits] + k[3bits] + w_n[9bits] + w_m[9bits] \tag{3.1}$$

$$C_L = 512 * 2^j \ [\mu s] \tag{3.2}$$

where

$$j = 0, .., 7$$

$$TS_L = 128 * 2^k \ [\mu s] \tag{3.3}$$

where

$$k = 0, .., 7$$

with

$$TS_L \leq C_L$$

Next, the time slot position index start and end, are encoded as w_m and w_n with 9 bits each. Both indexes mark the beginning and the end of the period during which association is allowed as seen in Figure 3.6. As described by Equation 3.4, the start and end index may range from 0 to $4 * 2^{(j-k)} - 1$.

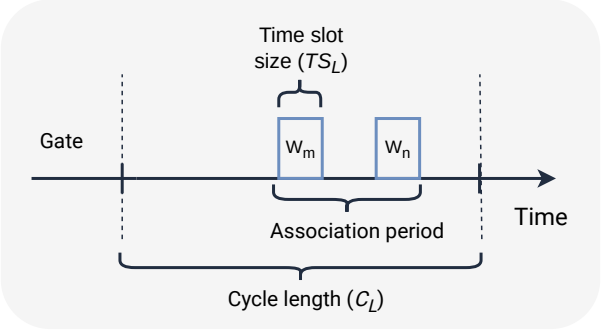


Figure 3.6: Diagram of the proposed pre-schedule bitmap encoding

$$0 \leq w_m \leq w_n \leq 4 * 2^{(j-k)} - 1 \tag{3.4}$$

where

$$k \leq j + 2$$

Finally, Equations 3.5 and 3.6 define the decoding operation done in the prospect client. Once w_m and w_n are received, the prospective client calculates on a time basis the transmission gate opening and closing as previously presented in the Chapter 1. The local time is ruled by the Time Synchronization Function (TSF), which is a timer with modulus 2^{64} counting in increments of microseconds.

$$TS_{Start} = TS_L * w_n \ [\mu s] \tag{3.5}$$

$$TS_{End} = TS_L * (w_m + 1)\ [\mu s] \tag{3.6}$$

3.5 Pre-synchronization methods

Providing accurate time synchronization between the network and prospective clients is required to ensure the correct usage of the association schedule. As this time synchronization is accomplished during association, it is referred to as pre-synchronization. To accomplish pre-synchronization, three different methods have been fashioned. They can be divided into two groups. The first considers that the AP beacon transmissions are not scheduled, hence, a new beacon will be generated according to a fixed interval, as described in subsection 3.4.3, and broadcasted from the AP to stations in coverage. On the other hand, the second considers a full WTSN scenario where beacons, as any other DL traffic is scheduled, following the mechanism described in the chapter 1.

The first method is called *Early/Late Beacon Detection (ELBD)* and it is part of the first group. The second group comprises the other two methods: *Slice-Based Cycle Detection (SBCD)* and *Follow-up based*. By different techniques, all methods attain time synchronization with prospective clients, making TSN association feasible.

As seen in Equation 3.7, beacon delay might be divided into a fixed delay (δ), and a variable delay (X). Beacons function as management frames and thus are exempt from the backoff mechanism. Instead, the variable delay (X) in beacon transmission primarily occurs due to Clear Channel Assessment (CCA). This delay occurs when there is an ongoing concurrent transmission, causing the beacon transmission to be postponed. In the subsequent proposed time synchronization methods, efforts are made to either filter or compensate for this variable delay. On the other hand, the fixed delay (δ) is a distinct aspect addressed in detail in the following section.

$$Delay = \delta + X \tag{3.7}$$

Beacon delay compensation

The received beacon presents fixed delays that the prospective client must adjust to achieve a correct pre-synchronization. Such delays are caused by channel access, beacon transmission time, and processing between the driver level and the physical layer at the transmitter side and vice-versa at the receiver side. As can be seen in Equation 3.8, δ is the overall corrected delay that is added to the received beacon time information at the driver level of the receiver side before setting it to the prospective client.

$$\begin{aligned}\delta &= DIFS + B_T + P \\ &= 28 + \frac{148 * 8}{6} + 19.7 \\ &= 28 + 197.33 + 19.7 \\ &= 245.03\ \mu s\end{aligned} \tag{3.8}$$

The DCF interframe space (DIFS) is the initial delay element, and it provides a technique for assigning priorities to frames in the distributed coordination function [25] of IEEE 802.11. It is the least amount of time a beacon frame must wait before being transmitted, and it varies depending on the standard used. The 802.11g is used in our implementation, therefore $DIFS$ takes 28 μs according to [29]. The beacon transmission time is B_T, and it is calculated using the beacon frame size of 148 bytes and the default data rate of 6 Mbps at which beacons are transmitted. Finally, the client processing time P was monitored and taken into account while processing and filtering beacon frames at the driver.

3.5.1 Early/late beacon detection (ELBD)

Due to the variable nature of the wireless channel, and the Clear Channel Assessment (CCA) mechanism that must be followed by the IEEE802.11 nodes, the beacon arrival time at the prospective clients is not always consistent. As a result, if the Timestamp field shown in Figure 3.5 is directly overwritten into the prospective client, this could result in synchronization inaccuracies and thus unsynchronized client transmissions.

The purpose of *ELBD* is to eliminate ineffective beacons. Relating to Equation 3.7, this method filters beacons affected by the variable delay X. As shown in Figure 3.8a, this filter uses the arrival time difference($\Delta_{arrival}$) of a pair of received beacons at the prospect's client driver, and checks if the difference is within a defined range (R) in a moving window manner. To define R, the received beacon frame's *Beacon Interval* - $bcn.int$ field is extracted, and an *error* deviation value is added, as indicated in Equation 3.9. If the received beacon pair matches the criteria, the client's timestamp is set using the last beacon's internal timestamp. A deep explanation of how the $error$ value is calculated may be found in Section 3.6.2.

$$R = bcn.int \pm error \tag{3.9}$$

This filter, unlike previous filtering approaches found in literature [17], does not require a convergence period, which is critical for a quick WTSN association operation. The algorithm 1 shows the operation of the filter. The term TSF stands for Time Synchronization Function, which represents the timestamp. First, when a

beacon is received, the current TSF is saved to calculate $\Delta_{arrival}$. If the $\Delta_{arrival}$ condition of two continuous beacon arrivals is met, the client's time is updated adding the delay compensation δ described in Equation 3.8, and lastly the $gate.flag$ will allow association packets to be transmitted.

3.5.2 Slice-based Cycle Detection (SBCD)

Besides channel access delays, having a schedule at the AP means that beacons will not follow a default transmission interval (102.4 ms), but instead will rely on the schedule as shown in Figure 3.8b. The $SBCD$ mechanism benefits from the cyclic operation of the schedule to achieve pre-synchronization.

To operate, apart from the pre-schedule, the variable part of the beacon's vendor element shown in Figure 3.5 carries the AP schedule. This way the prospective client knows when the beacon is expected to arrive. As it is not possible to predict how long the beacon had to wait before being transmitted, X in Equation 3.7, in this method, the timestamp in the beacon is not used. Instead, the client compares the $\Delta_{arrival}$ with the $ap.cycle$ of the AP schedule.

This comparison is performed by calculating the module operation between the $\Delta_{arrival}$ and $ap.cycle$, from two consecutive arrivals, as shown in Equation 3.10. In this manner, the prospect client filters beacons that are not being transmitted at the beginning of the AP time slot. Subsequently, once a pair of beacons meets the condition, as shown in the Algorithm 2, the client timestamp is set to 0 plus $ap.ts_{start}$, and the delay compensation described in Equation 3.8 is also added. $ap.ts_{start}$ is different from 0 when the AP time slot did not start with the cycle.

$$\Delta_{mod} = mod(\Delta_{arrival}, ap.cycle) \tag{3.10}$$

3.5.3 Follow-up based

The last pre-synchronization mechanism can be defined as the most generic as it supports non-scheduled and scheduled beacons. It exploits the fact that once a frame is transmitted, the physical layer notifies the driver and higher layers. The follow-up method timestamps every beacon transmission, overcoming the channel delay issue we described previously in Section 3.5.1, and loads the next beacon with this timestamp information ($bcn.tx.TSF$).

In contrast to previous pre-synchronization methods, this approach takes a different approach by compensating for the variable delay X from Equation 3.7. Instead of filtering out ineffective beacons, this method utilizes pairs of beacons, where one beacon carries information about the variable delay X of the previous beacon.

As shown in Algorithm 3, the first condition to calculate the current time is to confirm a pair of beacons are consecutive. For this, Δ_{tx} is compared with $bcn.int$,

which comes as a beacon field. Then, as seen in Figure 3.7, the $corrected.TSF$ can be calculated using Equation 3.11.

$$corrected.TSF = \Delta_{arrival} + bcn.tx.TSF[i-1] + \delta - DIFS \tag{3.11}$$

Equation 3.11, calculates the correct time by adding the $\Delta_{arrival}$, the transmission time, which is part of δ, and the timestamp of the moment the previous beacon was transmitted. $DIFS$ is already considered in the $bcn.tx.TSF$ hence needs to be subtracted from Equation 3.11.

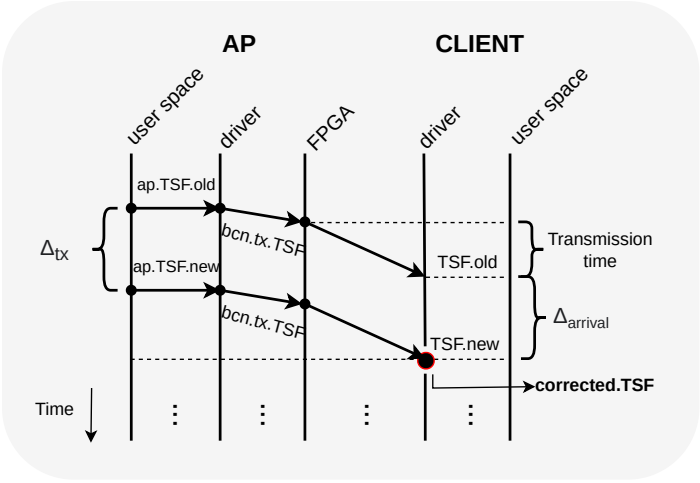
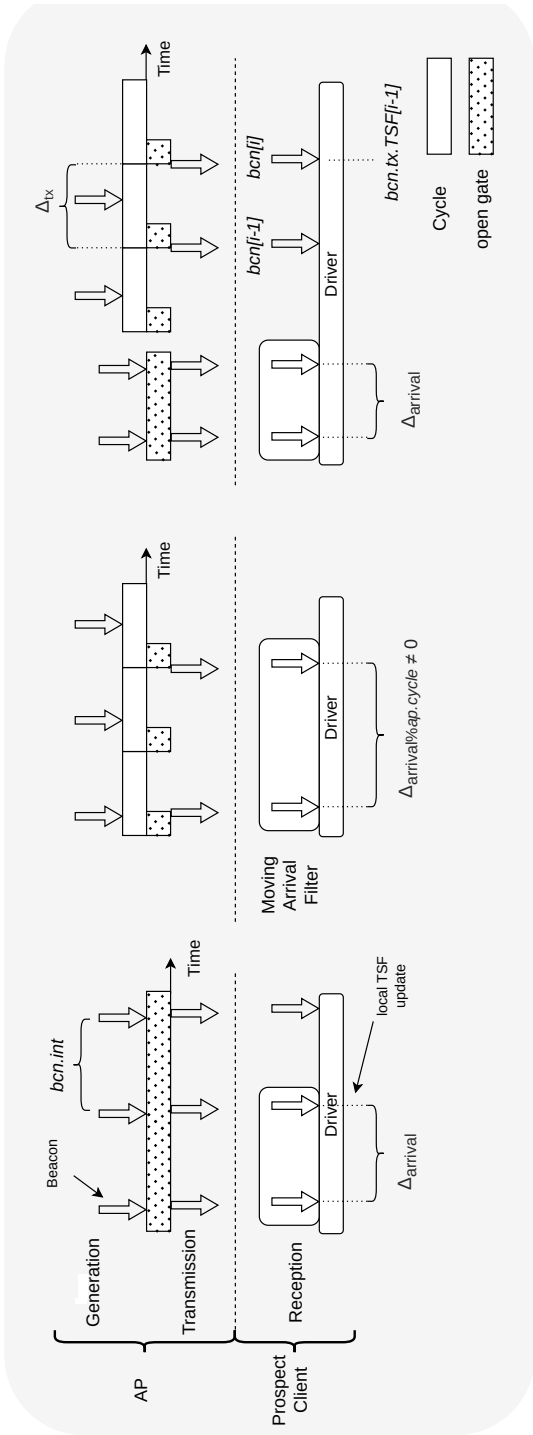


Figure 3.7: Timeline of the follow-up pre-synchronization process

After calculating the $corrected.TSF$, it is then set to the prospective client. This update in time is also applied to previous time information, hence more beacons can be used for pre-synchronization. A timeline diagram of the correction equation is shown in Figure 3.7. As it is shown, beacons are generated at the AP user space, transmitted from the FPGA, and finally processed at the client's driver.

Figure 3.8c, presents the advantage of the follow-up method over the previous ones as it can work for both scheduled and unscheduled AP beacon transmissions. This Figure also illustrates Δ_{tx} and $\Delta_{arrival}$ in the generation and reception of beacons.



(a) ELBD *(b) SBCD* *(c) Follow-up*

Figure 3.8: Overview of the three proposed pre-synchronization methods

Algorithm 1 ELBD pre-synchronization

Input: *new received beacon,* $error$
Output: *pre-synchronization*
Initialization:
Extract $ap.TSF$, $bcn.int$ {A beacon is received, and fields are extracted.}
Save $TSF.now$ {The beacon arrival TSF is saved.}
$\Delta_{arrival} = TSF.now - TSF.old$ {The TSF difference between two beacon arrivals is calculated.}
Moving Arrival Filter:
if $bcn.int - error \leq \Delta_{arrival} \leq bcn.int + error$ **then**
    {Check if $\Delta_{arrival}$ is within the filter limits.}
    $client.TSF = ap.TSF + \delta$ {The client TSF is updated.}
    $gate.flag = 1$ {The gate flag indicates the client was successfully pre-synchronized.}
end if
$TSF.old = TSF.new$
return $client.TSF, gate.flag$

Algorithm 2 SBCD pre-synchronization

Input: new received beacon, $error$
Output: pre-synchronization
Initialization:
Extract $ap.TSF$, $ap.cycle$, $ap.ts_{start}$ {A beacon is received, and fields are extracted.}
Save $TSF.now$ {The beacon arrival TSF is saved.}
$\Delta_{arrival} = TSF.now - TSF.old$ {The TSF difference between two beacon arrivals is calculated.}
$\Delta_{mod} = \text{mod}(\Delta_{arrival}, ap.cycle)$ {$\Delta_{arrival}$ is compared with the AP cycle using the module operation.}
Moving Arrival Filter:
if $0 \leq \Delta_{mod} \leq error$ **then**
    $client.TSF = 0 + ap.ts_{start} + \delta$ {The client TSF is updated considering the AP time slot position.}
    $gate.flag = 1$ {The gate flag indicates the client was successfully pre-synchronized.}
end if
$TSF.old = TSF.new$
return $client.TSF, gate.flag$

3.5.4 Implementation and Standard Compliance

The bootstrapping implementation covers mainly the driver of both the AP and prospective client. Table 3.2 provides a summary of the necessary changes in the

Algorithm 3 Follow-up pre-synchronization

Input: new received beacon
Output: pre-synchronization
Initialization:
Extract $ap.TSF.now$, $bcn.int$, $bcn.tx.TSF[i-1]$ {A beacon is received, and fields are extracted.}
Save $TSF.now$ {The beacon arrival TSF is saved.}
$\Delta_{arrival} = TSF.now - TSF.old$ {The TSF difference between two beacon arrivals is calculated.}
$\Delta_{tx} = ap.TSF.now - ap.TSF.old$ {The TSF difference between two beacon transmissions is calculated.}
Moving Arrival Filter:
if $\Delta_{tx} \leq 1.5 \cdot bcn.int$ **then**
    Save $TSF.current$ {To track current client TSF.}
    $corrected.TSF = \Delta_{arrival} + bcn.tx.TSF[i-1] + \delta - DIFS$ {Corrected TSF as in Eq. 3.11.}
    $TSF.old = TSF.now + corrected.TSF - TSF.current$ {Calculate the corrected old TSF at client.}
    $client.TSF = corrected.TSF$ {The client TSF is updated.}
    $gate.flag = 1$ {The gate flag indicates the client was successfully pre-synchronized.}
else
    $TSF.old = TSF.now$
end if
$ap.TSF.old = ap.TSF.new$
return $client.TSF, gate.flag$

driver for pre-synchronization and pre-scheduling. Notably, only the Follow-up method requires a driver enhancement at the AP which enriches the beacons with $bcn.tx.TSF[i-1]$ as detailed in Section 3.5.3 and Algorithm 3. Regarding driver adjustments to the prospective STA, each method necessitates distinct approaches. These varied approaches have been elaborated upon in the corresponding sections, namely 3.4 and 3.5. As shown in Section 3.4.1, the association procedure remains transparent to higher layers. Furthermore, it is important to emphasize that both WTSN and the proposed bootstrapping procedure are fully compliant with standards. However, it is worth noting that integrating TSN with legacy devices could potentially impact TSN's performance, as discussed in the Introduction of this document.

Table 3.2: Bootstrapping implementation: Pre-scheduling and pre-synchronization modifications in AP and STA

		AP	Prospective STA
Pre-scheduling		No modifications	Requires driver modifications
Pre-synchronization	**ELBD**	No modifications	
	SBCD	No modifications	
	Follow-up	Requires driver modifications	

3.6 Results

3.6.1 Setup description

The specified association mechanism has been implemented on an FPGA-based SDR platform utilizing IEEE 802.11. The openwifi IEEE802.11/Wi-Fi baseband chip/FPGA design was utilized for evaluating the proposed solution, which was implemented in the Software Defined Radio (SDR) ADRV9361-Z7035, which combines the Analog Devices AD9361 integrated RF Agile Transceiver™with the Xilinx Z7035 Zynq-7000. The carrier card ADRV1CRR-BOB/FMC was used to add an Ethernet interface [3].

Despite the 802.11g standard being employed for developing and testing, the association procedure has not been modified in newer iterations of Wi-Fi such as 802.11ax or 802.11be, hence the proposed association technique is still valid [30]. Even though using 802.11ax will also provide more transmission degrees of freedom when using OFDMA, during the association phase, however, resource use is constrained to the time domain. Apart from accurate time synchronization, UL OFDMA requires AP trigger frames. Hence for the association, the AP would need to realize whether there are one or more prospect stations willing to associate in order to trigger their transmissions in different OFDMA sub-carriers, which is far from ideal and might delay the process.

As shown in Figure 3.9, the infrastructure mode wireless network comprised two nodes: one access point (AP) and two clients/stations, all of which were connected to a PC functioning as a CNC and a Dashboard/MQTT Broker.

All the experiments presented next were performed in an office environment with neighbor channels in use and a fixed distance of 10cm between nodes. Also, a signal level lower than -50 dBm was ensured in all nodes. Despite the short testing distance and good signal level, as long as the nodes are in coverage, no pre-synchronization nor pre-scheduling will be altered as propagation delay is negligible for this procedure.

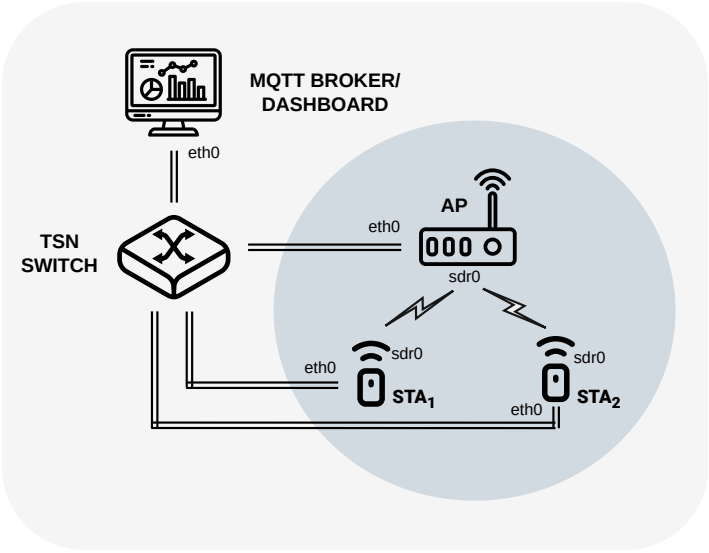


Figure 3.9: Testing topology for WTSN bootstrapping implementation

3.6.2 Threshold definition

In both $ELBD$ and $SBCD$ algorithms, an $error$ value is included in the main time update condition. This variable allows controlling the beacon filtering used for pre-synchronization. Thus, this $error$ value becomes a threshold allowing or avoiding pre-synchronization with the received beacons.

To define the optimal threshold value, two variables have been considered. First, the association delay is a key performance indicator of how transparent the WTSN association process is to higher layers. In [24], an in-depth study on the delays in setting up an IEEE802.11 link and how it affects user experience can be found. In that work, the association process (association+authentication) results in about 35% of the total set-up time which can take up to 30s, but in most cases takes 7s. Hence, 2.45s will be the baseline for our measurements.

The association delay values shown next, are measured from the reception of the first beacon until the association (association+authentication) is successful. Every value is the result of 20 successful association processes. Furthermore, a worst-case fixed association time slot of 128 μs within a cycle of 65535 ms was loaded in beacons. Also, in the $SBCD$ case, the AP was set to the same worst-case schedule to restrict beacon transmissions.

Tables 3.3 and 3.4 show different threshold levels as well as the median delay caused by association. In addition, from the tested successful association attempts,

we measured how many beacons were needed to achieve pre-synchronization from the reception of the first beacon. The most common, or modal, value observed in these captures is also reported.

Table 3.3: Measurement results of association delays (median) and number of beacons (mode) required with the ELBD resynchronization mechanism

Threshold(us)	Association Delay(s)	Beacons
2	1.284	3
3	1.159	5
5	1.029	3
10	1.039	3
20	1.039	3
50	1.038	4
100	1.037	3

The second variable to contemplate is the percentage of beacons used. Although this will highly depend on the channel access delays, it is helpful to infer the filtering means of the algorithm.

Looking into Figures 3.10 and 3.11, for both methods, as was expected, as the threshold level increases, the percentage of used beacons does as well. However, in the case of $SBCD$, the increase is sharper than in $ELBD$. Furthermore, as described in section 3.5, it is not possible to compare both approaches as they are based on different concepts.

Regarding the follow-up method, two tests were performed, one with and one without the worst-case schedule at the AP. The median association delays were 1.8 and 1.75 seconds, respectively. Additionally, the mode beacon count was 3 for both cases.

In conclusion, the introduction of pre-synchronization with worst-case schedules results in association delays of up to 1.284s for $ELBD$, 1.048s for $SBCD$, and 1.8s for the follow-up approach. These values remain lower than the 2.45s reported by [24]. This clearly demonstrates that the inclusion of pre-synchronization does

Table 3.4: Measurement results of association delays (median) and number of beacons (mode) required with the SBCD resynchronization mechanism

Threshold(us)	Association Delay(s)	Beacons
0	1.015	4
1	1.048	3
2	1.015	3
10	0.983	3

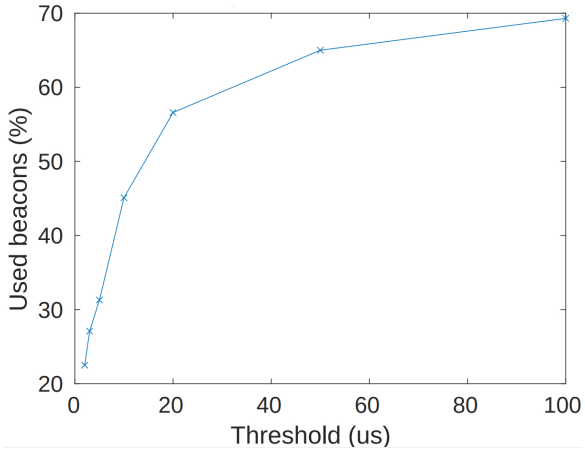


Figure 3.10: Results of used beacon percentage in the ELBD pre-synchronization method with different threshold values

not significantly delay the standard association process.

Additionally, it is important to note that both pre-synchronization methods, $ELBD$, and $SBCD$, allow for the definition of a threshold level that can be tailored to the percentage of used beacons in a given application. Although the number of beacons in this case is similar, resulting in a similar association delay, the threshold represents a trade-off between pre-synchronization accuracy and association delay. This trade-off depends on specific variables such as beacon generation frequency, AP schedule, and channel congestion.

Lastly, it is crucial to take into account scenarios in which multiple stations need to utilize the association time slot. Contention between stations might be another association delay source. However, an appropriate selection of the association time slot size and strategies such as grouping, used in IEEE 802.11ah will diminish such delays [31].

Furthermore, it is crucial to highlight the correlation between the possibility of simultaneous connections and the interaction between the WTSN schedule and the starting timing of the prospective client association procedure. Particularly noteworthy is the impact within shorter cycles and the consideration of the minimal packet prerequisites for association, which would lead to a reduction in the probability of concurrent connections.

3.6.3 Pre-synchronization

There are different methods to measure time synchronization accuracy. However, even though a driver-level implementation is done, due to the limitations of beacons,

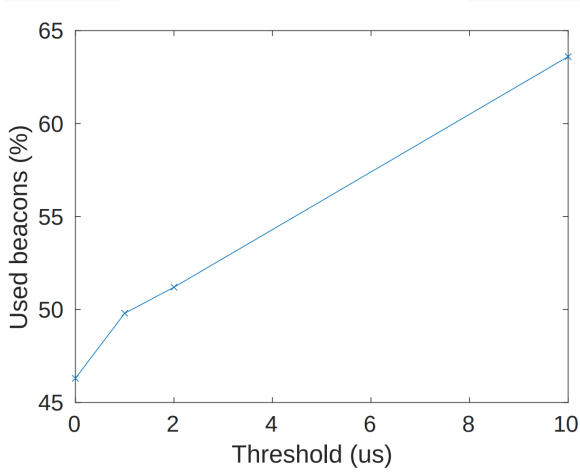


Figure 3.11: Results of used beacon percentage in the SBCD pre-synchronization method with different threshold values

time synchronization accuracy is limited. Thus, the aim of pre-synchronization is not high accuracy, but instead a good enough synchronization at small time slot sizes of 128 μs. Furthermore, the intention behind offering multiple pre-synchronization methods is to provide users with the flexibility to choose the approach that best suits their needs and aligns with the specific characteristics of their network environment.

Hence, to evaluate how good pre-synchronization is, the arrival time of ping request frames of similar size and the data rate of association frames were used. To elaborate, by using the architecture of Figure 3.9, once both clients were pre-synchronized, they transmitted ping request frames within the association time slot, which were logged at the AP. Equation 3.12 was used to analyze the arrival TSF of such frames into a cycle-based measurement.

$$T_{Of} = mod(T_{Ar}, C_L) \tag{3.12}$$

where T_{Of} represents the arrival time within the cycle in μs, T_{Ar} the frame arrival TSF, C_L the designated cycle length, and $mod()$ is the modulo operation. Once calculated, T_{Of} is pushed to the MQTT broker/Dashboard, which saves and presents the frame arrivals.

Figure 3.12, presents both clients' arrival offset to the AP using follow-up-based pre-synchronization. The horizontal red lines limit the time slots of 128 μs. The AP updates in real-time the pre-schedule, which then needs to be adopted by the prospective clients as well. As can be seen, both client receptions are always within the assigned time slot limits.

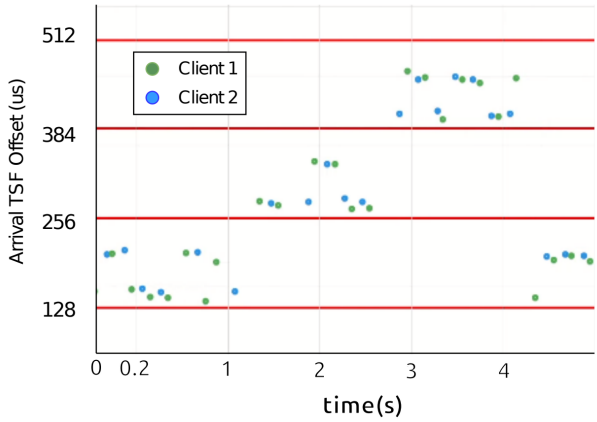


Figure 3.12: Arrival offset changes in the pre-schedule process

3.6.4 Pre-schedule

The correct behavior of the pre-scheduling mechanism is highly dependent on pre-synchronization. The time reference acquired by the time synchronization helps the prospective client to define the beginning of a cycle and thus, the opening and closing of the gates using the received pre-schedule. Figure 3.13 presents four different experiments where the prospective client changed its pre-schedule using the $ELBD$ pre-synchronization method. This Figure illustrates the arrival TSF at the AP across different association time slots, highlighting the pre-scheduling capabilities for adaptation. Each point on the graph corresponds to a specific TSF arrival time, demonstrating the ability to dynamically adjust and adapt within different association time slots.

3.7 Conclusion

By developing three pre-synchronization techniques to support the wireless association of prospective TSN clients, this study established that it is possible to achieve impactless association by providing TSN features to unconnected clients. Pre-scheduling is accomplished by loading AP broadcast beacons with the association time slot offset in time. The aforementioned pre-synchronization techniques grant the prospective client a time reference to transmit the association frames in the assigned time slots. The first pre-synchronization mechanism considers the AP transmitting beacons at a fixed interval, while the second and third include a schedule at the AP. For both, the prospective client had to compensate for delays related to contention, processing, and time slot access. The third pre-synchronization method outperforms the previous ones as it operates in a scheduled and non-scheduled

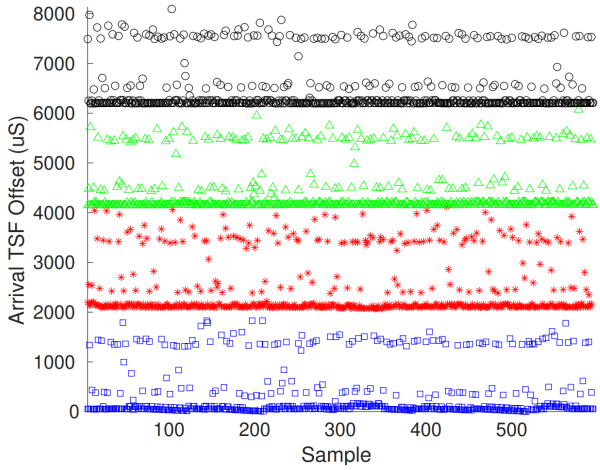


Figure 3.13: Arrival offset across different association time slots

scenario. Finally, it is important to note that the implementation and evaluation of these pre-synchronization solutions, as described in Section 3.6, were conducted under ideal conditions. Variations in the channel, caused by mobility or external interference, may lead to delays or, in some cases, failures in the association process. Addressing these issues requires further investigation.

References

[1] P. Avila-Campos, J. Haxhibeqiri, I. Moerman, and J. Hoebeke. *Impactless beacon-based wireless TSN association procedure*. 2022 IEEE 18th International Conference on Factory Communication Systems (WFCS), pages 1–8, 5 2022. doi:10.1109/WFCS53837.2022.9779186.

[2] P. Avila-Campos, J. Haxhibeqiri, I. Moerman, and J. Hoebeke. *Beacon-based wireless TSN association*. IEEE INFOCOM 2022 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), pages 1–2, 6 2022. doi:10.1109/INFOCOMWKSHPS54753.2022.9798105.

[3] X. Jiao, W. Liu, M. Mehari, M. Aslam, and I. Moerman. *Openwifi: A Free and Open-Source IEEE802.11 SDR Implementation on SoC*. IEEE Vehicular Technology Conference, 2020-May, 5 2020. doi:10.1109/VTC2020-SPRING48590.2020.9128614.

[4] A. B. Kinabo, J. B. Mwangama, and A. A. Lysko. *An evaluation of broadband technologies from an industrial time-sensitive networking perspective*. 2021 7th International Conference on Advanced Computing and Communication Systems, ICACCS 2021, pages 715–722, 2021. doi:10.1109/ICACCS51430.2021.9441748.

[5] M. K. Atiq, R. Muzaffar, O. Seijo, I. Val, and H. P. Bernhard. *When IEEE 802.11 and 5G meet time-sensitive networking*. IEEE Open Journal of the Industrial Electronics Society, 3(January):14–36, 2022. doi:10.1109/OJIES.2021.3135524.

[6] 5G-ACIA. *Integration of 5G with time-sensitive networking for industrial communications*. Technical report, 5G Alliance for Connected Industries and Automation (5G-ACIA), 2021.

[7] ETSI. *LTE; Evolved universal terrestrial radio access (E-UTRA); Physical channels and modulation (3GPP TS 36.211 version 12.5.0 Release 12)*. 2015. Available from: http://portal.etsi.org/tb/status/status.asp.

[8] T. Adame, M. Carrascosa-Zamacois, and B. Bellalta. *Time-sensitive Networking in IEEE 802.11be: On the Way to Low-Latency WiFi 7*. Sensors (Basel, Switzerland), 21(15), 8 2021. Available from: /pmc/articles/PMC8347193//pmc/articles/PMC8347193/?report=abstracthttps://www.ncbi.nlm.nih.gov/pmc/articles/PMC8347193/, doi:10.3390/S21154954.

[9] G. Kronauer, M. Seewald, M. Smith, and A. Regev. *Avnu Alliance® White paper in wireless TSN : market expectations, capabilities & certification*. (February), 2022.

[10] D. Cavalcanti. *White paper: wireless TSN – definitions , use cases & standards roadmap*. Avnu Alliance, pages 1–16, 2020.

[11] T. Adame, M. Carrascosa-Zamacois, and B. Bellalta. *Time-sensitive networking in ieee 802.11be: On the way to low-latency wifi 7*. Sensors, 21(15), 2021. doi:10.3390/s21154954.

[12] S. Sudhakaran, K. Montgomery, M. Kashef, D. Cavalcanti, and R. Candell. *Wireless time sensitive networking for industrial collaborative robotic workcells*. IEEE International Workshop on Factory Communication Systems - Proceedings, WFCS, 2021-June:91–94, 6 2021. doi:10.1109/WFCS46889.2021.9483447.

[13] S. Sudhakaran, V. Mageshkumar, A. Baxi, and D. Cavalcanti. *Enabling qos for collaborative robotics applications with wireless TSN*. 2021 IEEE International Conference on Communications Workshops, ICC Workshops 2021 - Proceedings, pages 0–5, 2021. doi:10.1109/ICCWorkshops50388.2021.9473897.

[14] O. Seijo, X. Iturbe, and I. Val. *SHARP: Implementation of a hybrid wired-wireless TSN network to enable flexible smart factories*. IEEE International Workshop on Factory Communication Systems - Proceedings, WFCS, 2021-June:95–98, 2021. doi:10.1109/WFCS46889.2021.9483597.

[15] O. Seijo, X. Iturbe, and I. Val. *Tackling the challenges of the integration of wired and wireless TSN with a technology proof-of-concept*. IEEE Transactions on Industrial Informatics, 3203(c):1–1, 2021. doi:10.1109/tii.2021.3131865.

[16] C. Cruces, R. Torrego, A. Arriola, and I. Val. *Deterministic hybrid architecture with time-sensitive network and wireless capabilities*. IEEE International Conference on Emerging Technologies and Factory Automation, ETFA, 2018-Septe:1119–1122, 2018. doi:10.1109/ETFA.2018.8502524.

[17] O. Seijo, J. A. Lopez-Fernandez, and I. Val. *W-SHARP: Implementation of a high-performance wireless time-sensitive network for low latency and ultra-low cycle time industrial applications*. IEEE Transactions on Industrial Informatics, 17(5):3651–3662, 2021. doi:10.1109/TII.2020.3007323.

[18] R. Cecil, V. Setka, D. Tolar, and A. Sikora. *RETIS-Real-Time sensitive wireless communication solution for industrial control applications*. IDAACS-SWS 2020 - 5th IEEE International Symposium on Smart and Wireless Systems within the International Conferences on Intelligent Data Acquisition and Advanced Computing Systems, Proceedings, 2020. doi:10.1109/IDAACS-SWS50031.2020.9297074.

[19] H. Baniabdelghany, R. Obermaisser, and A. Khalifeh. *Extended synchronization protocol based on IEEE802.1AS for improved precision in dynamic and asymmetric TSN hybrid networks*. 2020 9th Mediterranean Conference on Embedded Computing, MECO 2020, pages 8–11, 2020. doi:10.1109/MECO49872.2020.9134100.

[20] I. Val, O. Seijo, R. Torrego, and A. Astarloa. *IEEE 802.1AS Clock Synchronization Performance Evaluation of an Integrated Wired-Wireless TSN Architecture*. IEEE Transactions on Industrial Informatics, 18(5):2986–2999, 5 2022. doi:10.1109/TII.2021.3106568.

[21] A. M. Romanov, F. Gringoli, and A. Sikora. *A precise synchronization method for future wireless TSN networks*. IEEE Transactions on Industrial Informatics, 17(5):3682–3692, 5 2021. doi:10.1109/TII.2020.3017016.

[22] *802.1AS-2020 - IEEE Standard for Local and Metropolitan Area Networks-Timing and Synchronization for Time-Sensitive Applications*. Technical report, 2020.

[23] *IEEE 802.1Qav-2009 - IEEE standard for local and metropolitan area networks– virtual bridged local area networks - forwarding and queuing enhancements for time-sensitive streams*. Technical report, 2010.

[24] C. Pei, Z. Wang, Y. Zhao, Z. Wang, Y. Meng, D. Pei, Y. Peng, W. Tang, and X. Qu. *Why It Takes so Long to Connect to a WiFi Access Point*. Proceedings - IEEE INFOCOM, 2017. doi:10.1109/INFOCOM.2017.8057164.

[25] D. D. Coleman and D. A. Westcott. *CWNA: Certified Wireless Network Administrator Official Study Guide*. SYBEX Inc., USA, 4th edition, 2021.

[26] *IEEE P802.1Qcc - IEEE draft standard for local and metropolitan area networks–media access control (MAC) bridges and virtual bridged local area networks amendment: stream reservation protocol (SRP) enhancements and performance improvements*. pages 1–236, 2018.

[27] K. A. Nsiah, K. Alkhouri, and A. Sikora. *Configuration of wireless TSN networks*. IDAACS-SWS 2020 - 5th IEEE International Symposium on Smart and Wireless Systems within the International Conferences on Intelligent Data Acquisition and Advanced Computing Systems, Proceedings, pages 3–7, 2020. doi:10.1109/IDAACS-SWS50031.2020.9297066.

[28] M. Aslam, W. Liu, X. Jiao, J. Haxhibeqiri, G. Miranda, J. Hoebeke, J. Marquez-Barja, and I. Moerman. *Hardware efficient clock synchronization across Wi-Fi and ethernet-based network using PTP*. IEEE Transactions on Industrial Informatics, 18(6):3808–3819, 6 2022. doi:10.1109/TII.2021.3120005.

[29] *802.11 IEEE standard for information technology - telecommunications and information exchange between systems - local and metropolitan networks - specific requirements - Part 11 : Wireless LAN medium access control (MAC) and physical layer (PHY) specifications*. IEEE Std 802.11b-1999, pages 1–96, 2000.

[30] *open-sdr/openwifi: open-source IEEE 802.11 WiFi baseband FPGA (chip) design: driver, software*. Available from: https://github.com/open-sdr/openwifi.

[31] A. Seferagic, I. Moerman, E. De Poorter, and J. Hoebeke. *Evaluating the suitability of IEEE 802.11ah for low-latency time-critical control loops.* IEEE Internet Things J., 6(5):7839–7848, 10 2019. doi:10.1109/JIOT.2019.2916579.

4

Unlocking Mobility for Wi-Fi-based Wireless Time-Sensitive Networks

"Utopia is on the horizon. I move two steps closer; it moves two steps further away. I walk another ten steps and the horizon runs ten steps further away. As much as I may walk, I will never reach it. Then what is the purpose of utopia? It is there to make us keep walking."

–Eduardo Galeano

While integrating a system designed for determinism with inherently stochastic communication technology presents challenges, it also offers notable advantages, with mobility being a key benefit. As a result, addressing the handover procedure becomes a critical feature for WTSN systems. This chapter introduces a refined handover protocol for seamless TSN over Wi-Fi. It specifically addresses scenarios where Access Points operate on distinct channels, and Stations are limited to a single radio. We propose four conceptual methodologies for identifying potential candidate APs, designed to minimize disruptions to time-sensitive applications during the handover process. Additionally, we provide a practical implementation and comparative analysis of one-dimensional and two-dimensional UltraWideband location-based handover techniques.

⋆ ⋆ ⋆

This Chapter is based on:

Unlocking Mobility for Wi-Fi-based Wireless Time-Sensitive Networks

Published in IEEE ACCESS, Feb. 2024.
and:
Optimizing Handover in Time-Sensitive Wi-Fi Networks through Machine Learning

Presented in ETFA, Sep. 2024.

⋆ ⋆ ⋆

Abstract

Undoubtedly, mobility remains a fundamental asset in wireless communications. Conversely, time-sensitive networking (TSN) represents a vital technology that enables determinism and low latency, fundamental for time-sensitive applications. In our study, we marry these two concepts by introducing a pioneering procedure that facilitates seamless handover within a Wi-Fi-based Wireless Time-Sensitive Network (WTSN). Through extensive real-world development and testing, we assess various techniques for optimizing handover moment selection and reducing handover delay. Our findings demonstrate that an integrated approach combining Received Signal Strength Indication (RSSI) and location-based reduces the need for traditional channel scanning. Our findings demonstrate that our mechanism reduces handover delay below 10 milliseconds and optimizes the handover moment selection, leading to improvements in critical network parameters such as bandwidth and jitter.

4.1 Introduction

With the growing emphasis on technologies such as extended reality (XR), which aims to blur the line between the virtual and real world, the concept of motion-to-photon latency has emerged. This latency refers to the time it takes for user movements to be accurately reflected on a display screen, such as those found in an augmented reality (AR) headset. When the motion-to-photon latency exceeds 20 ms, it becomes difficult for your mind to believe that you are truly immersed in the virtual environment being portrayed [1]. In fact, prolonged exposure to high motion-to-photon latency can lead to unpleasant symptoms such as motion sickness and disorientation [2].

Imagine a scenario where multiple AR headsets are wirelessly connected to a central computer that possesses ample processing power, enabling a top-notch mobile user experience. In this context, the network connecting the headsets to the computer becomes vital. It should offer high reliability and minimize communication latencies below the motion-to-photon latency limit, even during the handover

time. This is one of the objectives that wireless time-sensitive networking (WTSN) should fulfill.

In addition to proposing a general handover procedure, our approach in this work focuses on combining localization within a dedicated network with online and offline link quality learning to refine handover decisions. This method eliminates interruptions to time-sensitive traffic and reduces the latency overhead typically caused by channel scanning.

The chapter's specific contributions can be summarized as follows:

- A near seamless handover procedure for Wi-Fi-based TSN between Access Points (AP) operating in different channels while Stations (STAs) have a single radio.
- Four different conceptual techniques to determine candidate APs mitigating the adverse impact on time-sensitive applications during handover time.
- The practical implementation and comparison of three one-dimensional and two two-dimensional Ultra-Wideband (UWB) localization-based handover techniques are presented, focusing on their impact on bandwidth, jitter, and frame loss. These techniques reduce the need for the STA to scan nearby APs.

The remainder of the chapter is organized as follows. First in Section 4.2, related work on WTSN handover is presented. Next, in Section 4.3, the proposed WTSN handover procedure is described in detail. In Section 4.4, the hardware handover implementation is explained. In Section 4.5, the results, regarding the handover delay in uplink (UL) and downlink (DL), and handover moment selection are shown. Finally, Section 4.6, concludes this chapter.

4.2 Background and Related Work

Before delving into WTSN handover, we will first establish the current research context. We will begin by examining general handover optimization standards within IEEE 802.11, followed by a focus on localization-based and link quality-driven optimization. Finally, we will explore research at the intersection of WTSN and mobility.

Several key standards address the challenges of efficient handover in Wi-Fi networks. IEEE 802.11k allows for the creation of a predefined list of potential target channels for STAs to streamline the channel selection process during handover, despite relatively lengthy scan times [3]. IEEE 802.11v introduces Basic Service Set (BSS) Transition Management (BSTM), proactively informing STAs of disassociation to seek better APs [4]. Pre-authentication, when used with methods like

Pairwise Master Key (PMK) Security Association, enables authentication through the existing AP [5]. IEEE 802.11r, or fast BSS transition, enhances handover through key caching and wired network routing of handshake messages [6]. Cisco's Intracontroller handover transfers the STA's entry from the old AP to the new one via a Wireless Local Area Network (LAN) Controller, typically taking 10 milliseconds, with STA involvement in decision-making [7]. IEEE 802.11be's Multilink Operation allows STAs to associate with new APs while maintaining data flow with the current AP, ensuring a seamless handover experience, but using two radios [8]. Available hardware incorporates some of these techniques to enhance handover capabilities, but it primarily targets specialized industrial-oriented hardware with closed, proprietary solutions.

In addition to the solutions previously discussed, handover optimization remains a focal point in wireless systems research, serving as a critical response to an acknowledged weakness in wireless. Localization, which capitalizes on the spatial information of network nodes to select the most suitable AP for an STA, stands out as a promising strategy. In [9], an approach to this centers on improving the handover's discovery phase by exclusively relying on localization data, coverage area, and resource metrics. However, a notable limitation in this work is the exclusion of key link quality metrics, such as Received Signal Strength Indicator (RSSI) and Signal-to-Noise Ratio (SNR), from the model. This omission leads to the assumption of ideal propagation conditions, diminishing the real-world applicability of the proposed solution.

The Artificial Intelligence Machine Learning (AIML) Topic Interest Group within the IEEE 802.11 working group explores the application of AIML in IEEE 802.11 systems and devices, assessing technical feasibility. In their latest report [10], handover recommendations include adding weights to AP candidates based on handover probability predictions to prioritize channels in the scanning stage and predicting RSSI values to determine the optimal handover moment, facilitating STAs in initiating the scanning process at the right time. Given the absence of WTSN conditions concerning channel scanning, these recommendations do not present a viable solution for time-sensitive applications. However, they underscore the importance of utilizing RSSI in the handover process.

In the study referenced in [11], the authors utilize RSSI zones around the AP to assess the need for handover. They also introduce interleaved scanning for channel assessment, which schedules traffic and scanning in time. However, it is important to note that still, the average scanning time in this approach is 18 milliseconds, during which the STA cannot transmit or receive data, potentially impacting a possible time-sensitive traffic.

An interesting approach known as SyncScan, detailed in [12], introduces continuous AP discovery by aligning scanning timeslots at the STA with AP transmissions. However, again, a notable limitation of this approach arises during the scanning

phase, where, although optimization reduces handover delay, it prevents the STA from receiving its own AP's traffic concurrently. It is important to note that this issue is mitigated within the context of TSN, as apart from an accurate network time synchronization, all nodes adhere to a transmission schedule that can accommodate a scanning timeslot. Consequently, the STA avoids receiving time-sensitive data during the scanning process, ensuring smooth operation in practice.

Presently, there exists a limited body of research that addresses the integration of handover and WTSN systems. Among these initiatives, the work by [13] is particularly noteworthy. Notably, this study diverges from our own approach through its adoption of a decentralized architecture. In this alternative framework, the responsibility for initiating handover decisions rests with the STA rather than being centralized at the Centralized User Configuration/Central Network Controller unit, thus introducing significant disparities when compared to our proposed handover procedure. It is worth noting that this approach employs a single radio at the STA, giving rise to a pivotal consideration. The feasibility of achieving a genuine soft handover is contingent upon the TSN schedule in relation to the speed of channel switching, a metric typically measured in the range of tens of milliseconds under ideal circumstances [14]. In practical terms, this constraint implies that the seamless execution of soft handover (i.e. with no disconnection as in cellular networks) still may not be practically attainable for real-world implementations and applications.

Another notable work that addresses the integration of mobility with WTSN can be found in [15]. In this study, the authors assess their zero-delay handover approach through a combination of simulations and real-world experiments. Similar to our own work, this research encompasses multiple phases aimed at achieving seamless handover. However, a key point of differentiation lies in the utilization of redundancy via multiple radios for handover, in contrast to our approach, which employs a single radio. This pivotal distinction introduces a trade-off between TSN scheduling complexity, the allocation of time resources to manage handover delays, and the handover delay. Nonetheless, it is essential to note that, with control over the hardware, as demonstrated in our research, it is possible to implement optimizations that effectively reduce this handover delay while using a single radio. This, in turn, simplifies the handover procedure, enhances efficiency in terms of time resources utilization, and has minimal impacts on the operation of time-sensitive applications.

One significant distinction between the proposed handover approach in this work and existing methods lies in the time-sensitive nature of the network, which now presents a crucial constraint. While most methods aim to optimize and reduce the time required to complete the handover process, none of the current methods address the critical need to maintain time synchronization and provide a transmission schedule. Furthermore, none of these methods support identifying a set of candidate APs for handover without impacting the time-sensitive traffic of the node performing handover or any other time-sensitive traffic handled by the candidate APs.

Finally, traditionally, the STA has typically been responsible for making the final handover decision. However, in the context of TSN, which is intrinsically driven by the pursuit of determinism and underscores enhanced network control guided by telemetry measurements, a transition towards centralized handover decisions aligns more naturally with the principles of WTSN.

4.3 WTSN Handover

WTSN, designed as an extension of TSN, must uphold the objectives of TSN, which are high reliability and tightly bounded latencies. This necessitates the avoidance of interference with both its own traffic and other time-sensitive data streams during operation as well as handover time. Considering these factors, we have developed a handover procedure that encompasses three critical aspects: i) near seamless transition, ii) minimal disruption, and iii) strategic handover.

Prior to describing the WTSN process, it is crucial to provide an overview of the devices, modules, and topology that play integral roles in this procedure.

4.3.1 WTSN Handover Architecture

To achieve the aforementioned objectives while ensuring compatibility with TSN, it is imperative to utilize the TSN architecture. Among these architectures, the centralized approach, depicted in Figure 4.1 and used in this study, proves to be particularly advantageous for handover scenarios, thanks to its comprehensive network perspective and control advantages. As such, the Centralized User Configuration (CUC) and the Central Network Controller (CNC) jointly manage the computation and distribution of schedules across all TSN nodes. The TSN switches form the core, responsible for handling and forwarding time-sensitive traffic flows. The APs signify the wired network's edge providing wireless communication for the STAs.

As an essential incorporation, we have included a Handover Controller Module (HCM) at the controller side and a Handover Agent Module (HAM) at the STA side. These modules play a pivotal role in the management of handover, as they detect the need for handover and initiate the process accordingly.

As illustrated in Figure 4.1, in addition to the wired and wireless time-sensitive links, a management and configuration network is presented. This network serves two primary objectives. Firstly, it facilitates the HCM in gathering information concerning the STA link status. Simultaneously, it is employed to provide the STA through the HAM with accurate details for near seamless handover between APs, as described in the following section.

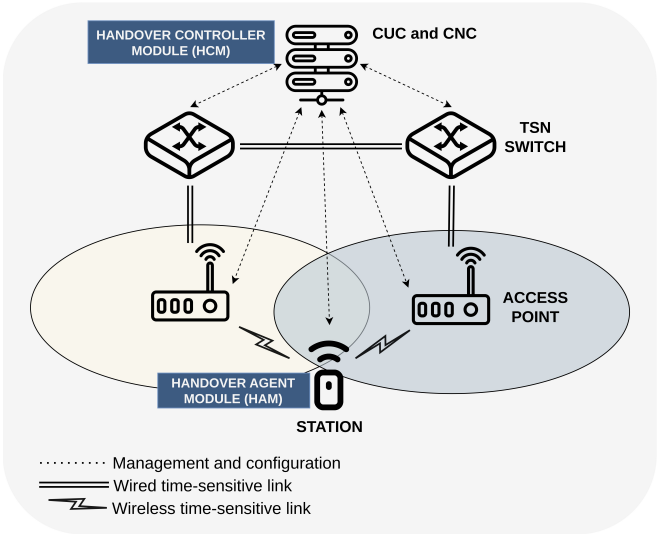


Figure 4.1: Centralized WTSN handover topology

4.3.2 Handover Process

To streamline the WTSN handover process, we have segmented it into four distinct states as shown in Figure 4.2. For clarity in understanding the procedure, we have chosen a two-part explanation. Initially, we will establish and define the proposed states, followed by a detailed description of each state transition. The states are as follows:

1. Before HO: In this state, the STA is connected to an AP, featuring an allocated transmission schedule tailored to meet its specific traffic flow demands. In this state, there is no need for a handover process.

2. AP candidates determined: in this state, the HCM together with the HAM, employ one of the time-sensitive, impactless methods—such as localization, coordinated beaconing, and coordinated probing, as detailed in the subsequent section. Through these techniques, the STA assesses the link quality to potential APs, and determines if it is necessary to perform handover.

3. Handover Prepared: In this state, the handover decision has been finalized, and in alignment with the chosen target AP, the resources along the new path, including the schedule, are prepared for utilization. At this stage, both the TSN and the STA are in a state of readiness, anticipating the handover trigger.

4. Handover Done: At this state, the STA has transitioned to the target AP and it is able to communicate with the TSN using its new transmission schedule.

Subsequently, the STA will revert to State 1.

The first state change, as seen in Figure 4.2, *AP Candidates Determination*, is dedicated to the identification of potential APs to which the handover could take place. In the second state shift, *Preparation*, the system assembles the essential elements required for a near seamless transition. Finally, in the third state change, *Trigger*, the handover process is initiated, ensuring a seamless and uninterrupted transition toward the selected AP.

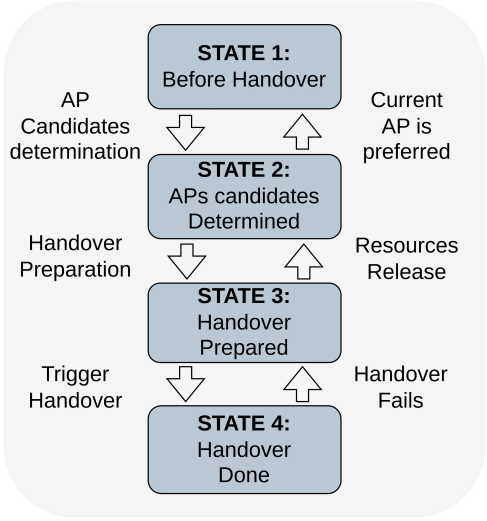


Figure 4.2: Proposed state diagram of the WTSN handover process

Let's delve into each of these stages change in the following subsections:

4.3.2.1 Determining candidate APs

When the need for handover is identified, for reasons such as signal strength deterioration, load distribution requirements among APs, or STA localization/mobility, the selection of candidate APs must be carried out without disrupting time-critical traffic. For this, we propose using one or a combination of the following methods:

- **Localization**: Through localization, the STA can pinpoint its exact location and inform the HCM of it via the present AP using the management and configuration link. Either a Wi-Fi-based localization system using fine timing measurement (FTM) or an external localization system such as UWB-based can be used. In the first case, to prevent affecting other time-critical traffic, the exchange of FTM requests must be planned within the overall WTSN. By leveraging these methods, the HCM can identify one or more candidate

APs that can be chosen as the target for handover based on the position of the STA, the location of the APs, and telemetry information indicating the signal quality for STA-AP pairs at particular locations.

- **Coordinated beaconing and scanning** Time-sensitive STAs can only scan for potential APs when they are not sharing any time-sensitive data in the UL or DL. STAs can be given the ability to scan for nearby APs while no time-sensitive data exchanges are occurring by appropriately scheduling beacon transmissions across APs at a specific time. In addition, this specific time should be known by all STAs. In this manner, the STA can gather data about potential APs and provide it to the HCM. Similar to current Wi-Fi handover protocols presented in Section 4.1, the STA can be informed about nearby APs and their operational channel to limit the number of channels that need to be scanned.

- **Coordinated probing** To gather data regarding potential APs, the STA would directly transmit probe requests in a fixed shared time slot across APs. This will prevent interfering with ongoing time-sensitive traffic. The information from probe requests that candidate APs receive is sent to the HCM to determine the group of candidate APs. As an alternative, probe responses could be returned to the STA, allowing it to gather the data on its own. Such probe responses can also be transmitted over the AP the end node is currently connected to, minimizing the amount of time the STA needs to remain on a different channel. As in the previous approach, by informing the end node about nearby APs and their operational channel, the number of channels to send probes on can be reduced.

- **Overhearing** STAs on different Basic Service Sets (BSS) operating on the same channel in dense installations can overhear one another, a phenomenon known as overlapping BSS. Information about the region or zone where an end station is located can be collected by listening to traffic and logging details such as the STA involved in the communication, received signal strength, etc. The HCM can then be informed of this information. This allows for the identification of possible APs on various channels inside or close to that region.

4.3.2.2 Handover Preparation

The HCM, after obtaining the set of candidate APs, will decide the target AP the STA must handover to and, optionally, the timing at which the handover should take place. To prepare for handover, the necessary resources must be set up in the target AP and along the new path via the TSN switches in order to prevent interfering with other time-sensitive flows handled by the target AP and guarantee the timings

of the new flow(s) from the STA that will wander to the AP. The HCM, knowing the STA's application requirements, will compute transmission schedules for the impending traffic flow path as part of this preparation. Such schedules concern the target AP, the STA performing handover, and the TSN switches along the new route. The CUC/CNC communicates the allocation of resources at the target AP and switches by using the management and configuration links, either to be executed immediately or at the predetermined time when the handover should be executed.

Also, resources that will be no longer required after the completion of the handover procedure should be considered. When the exact timing of handover has been defined, the release of resources can already be communicated along with timing, which takes into consideration sufficient margin to verify the successful completion of the handover or to cancel the release upon detection of failure of the handover. In case no exact timing of the handover has been defined, the communication to release resources will only take place upon successful completion of the handover procedure. Further, to avoid the overhead related to the association and authentication to a new AP, the HCM can already provide all necessary information about the STA to the new AP, hence performing the association and authentication on behalf of the STA.

Figure 4.3 illustrates a handover scenario within the TSN schedule framework. The diagram depicts schedules for both the initial and target APs, featuring designated time slots for connected devices, particularly the STA. Notably, during the handover preparation phase, an unused time slot appears in the target AP schedule, serving as a crucial element for a smooth transition. Upon the handover trigger, the STA shifts from the initial AP's allocated slot to the corresponding slot in the target AP. Despite potential variations in cycle lengths between the initial and target schedules, strict adherence to the STA's time-sensitive traffic flow requirements is required. Post-handover success, resources are liberated at the initial AP. More details on the handover process are described in the next section. This approach ensures that the STA smoothly transitions from the initial AP to the target AP, mitigating potential traffic contention.

4.3.2.3 Handover Trigger

Once the handover process advances to State 3, the handover manager module will initiate handover for the STA via the current AP. This initiation encompasses important information, such as the new AP's channel, transmission schedule, whether pre-association has already been carried out at the target AP, and, if specified, the timing for executing the handover. In the absence of precise timing instructions, the receiving STA will promptly commence the handover procedure.

The process involves the following steps at the STA:

1. In order to mitigate undesired UL transmissions throughout the procedure,

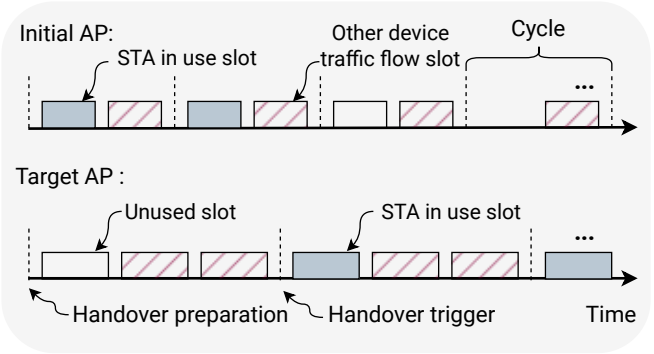


Figure 4.3: Example of WTSN handover transmission schedule

the STA will enact a transmission gating system and systematically queue frames. While a comprehensive elucidation of the gating system is beyond the scope of this chapter, a detailed exposition can be referenced in [16].

2. The new transmission schedule is set up and the STA switches to the new channel.

3. The transmission gates are reopened at the STA according to the new schedule.

4. Once the gates are opened, the STA can immediately continue to perform time synchronization and time-sensitive communication via the target AP using the provided schedule. For time synchronization, a profile of the Precision Time Protocol (gPTP) defined within IEEE1588, is the preferred time approach in TSN and transparent and boundary clocks are the most relevant clock modes in PTP [17]. For both cases, TSN handover will work as follows.

    - With a transparent clock, the STA can continue executing the time sync procedure with the end-to-end transparent clock.
    - With a boundary clock, the STA can proceed with the sync procedure, but now with the new boundary clock, which is synced with the old boundary clock in the old AP. As part of the trigger, the IP address or MAC address of the target AP can be communicated, such that the STA can anticipate the new clock it will interact with.

5. In case of association with the target AP has not yet taken place, the association process with the new AP will be performed using the provided schedule.

4.3.2.4 Handover Failure

From a theoretical point of view, the occurrence of a handover failure is dependent on specific conditions, such as the inability to establish the target link. This may be attributed to factors like a failed wireless association or unfavorable channel conditions. Nevertheless, from a practical perspective, incorporating handover failure mechanisms becomes imperative to safeguard time-sensitive traffic flows. Detecting such failures is straightforward, as both the HCM and the HAM anticipate the initiation of traffic to and from the new link within a defined timeframe.

In response to this scenario, two alternative approaches are proposed. First, the STA attempts to revert to the previous Access Point (AP), retaining its time resources as described in Section 4.3.2.2 and resetting the handover process to re-enter State 1, as shown in the state machine in Figure 4.2. Second, if this initial approach fails, the STA initiates a bootstrapping process. Importantly, this process must be designed to avoid disrupting existing time-sensitive flows for clients already associated, as outlined in our earlier work [18].

4.4 WTSN Handover Practical Implementation

In this section, we will focus on the practical implementation and testing of the described methods in Section 4.3. The objective is to realize a seamless WTSN handover experience, with particular attention to two key aspects: i) swift and efficient switching between APs by optimizing delays in network and STA processes that deal with AP switching, and ii) making handover decisions on optimal space/time moment considering wireless channel dynamics. These two aspects will be described in the next subsections. For this implementation, we have been focused on the wireless part of the WTSN Evaluation Kit, which can be seen in Figure 4.4 [19].

The Evaluation Kit offers a range of topology options for developing and testing both wired and wireless TSN. It is capable of accommodating an increased number of APs, as demonstrated later in our handover development and testing. Furthermore, the kit integrates essential TSN features, including PTP-based time synchronization mechanisms, a TSN gating system within the wireless nodes, and comprehensive management implementations.

4.4.1 Handover Delay Optimization

Handover delay can be defined as the time elapsed from the reception of the last frame from the old AP to the receipt of the first frame from the target AP with no other traffic [20]. Minimizing this delay requires attention to two distinct components: the wired part, responsible for updating the traffic flow path, and the wireless part, which handles the channel update at the STA. In this section,

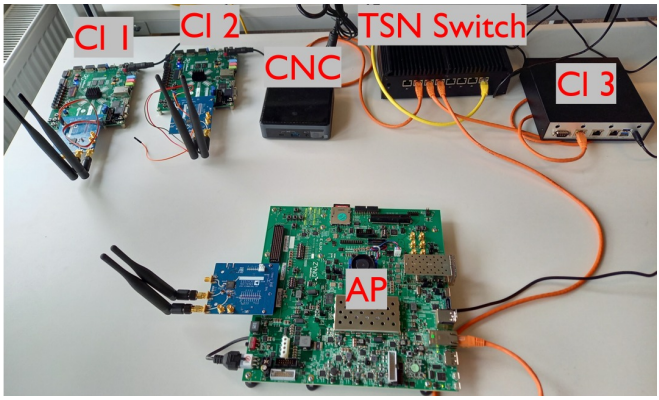


Figure 4.4: Evaluation kit for wired-wireless TSN

our primary focus will be on the latter component, as it tends to consume more time [12].

Once the HAM within the STA receives instructions from the controller's HCM, it initiates the handover process using the user-space command *Sdrctl*. *Sdrctl* is a user-space utility designed as an nl80211 test mode command, facilitating communication with the openwifi[1] driver. This tool enables control over FPGA and transceiver settings, including channel selection, clear channel assessment (CCA) threshold, transmission schedule, data rate, and various other parameters.

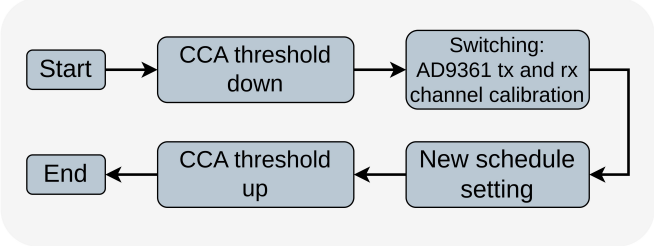


Figure 4.5: Flow diagram of the WTSN STA handover process

Therefore, as illustrated in Figure 4.5, HAM leverages *Sdrctl* to execute a sequence of actions. Initially, it reduces the CCA threshold below the channel noise level, hence prompting the STA to perceive the channel as occupied. The CCA threshold helps devices decide whether a channel is considered busy or idle. As such, reducing the CCA threshold will virtually prevent the STA from transmitting during handover time. Subsequently, the new frequency is configured for the transceiver, next, the new transmission schedule is set, concluding with reversing the CCA

[1] https://github.com/open-sdr/openwifi

threshold. Both the new frequency and new transmission schedule, are received from the HCM as described in Section 4.3.2.3. By completing the entire handover operation with just a single instance of netlink communication between the user space and the driver, we effectively optimize the processing delay associated with this operation. This optimization, as we will demonstrate in the Results Section 4.5, leads to a reduction in the overall handover delay.

4.4.2 Handover Moment Selection in One Dimension

Now that we have introduced an optimal channel switching procedure, the next critical aspect is to execute the handover at the precise spatial moment. This directly correlates with the techniques detailed in Section 4.3.2.1. Traditionally, achieving an optimal handover location involved setting an RSSI threshold to trigger channel scanning, a process that may be limited by the controller based on network configuration. However, channel scanning can be incompatible with TSN, as it temporarily disrupts the STA's ability to transmit or receive time-critical traffic from the current AP as demonstrated with the related works in Section 4.1. To circumvent this issue, alternative mechanisms described in Section 4.3.2.1 have been proposed. In our exploration, we have chosen to investigate the use of localization as a means to avoid scanning and facilitate localization-dependent applications.

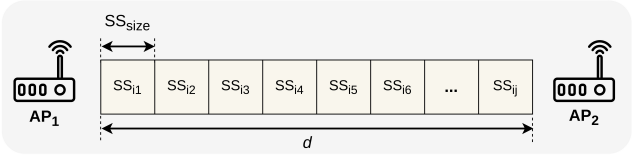


Figure 4.6: Implementation of handover moment selection with spatial slots (SS) distribution

Integrating an Ultra-Wideband (UWB) localization tag with the STA offers the ability to pinpoint the STA's location in relation to the surrounding APs. To correlate STA localization with channel quality, RSSI values are continually monitored and relayed as monitoring traffic to the handover module at the TSN controller. The controller leverages this data to construct a one-dimensional Radio Environmental Map (REM), partitioning the space between APs into Spatial Slots (SS) that encapsulate the recorded RSSI values as shown in Figure 4.6. Dividing the space into SS offers the advantage of mitigating the impact of localization system accuracy limitations. Consequently, the SS_{size} is chosen in consideration of both the localization system accuracy and the anticipated spatial coherence of the environment. The RSSI values on each SS are updated every t_s seconds as in Equation 4.1:

$$\mathrm{SS_{ij}} = \alpha \cdot \mathrm{RSSI_{ij}^{NEW}} + (1 - \alpha) \cdot \mathrm{RSSI_{ij}^{OLD}} \tag{4.1}$$

where α will control the importance of new RSSI values, i represents the AP and j is the SS index.

Utilizing the REM methodology, the network controller determines the optimal AP for the STA to connect to. This approach is not constrained by a single dimension; instead, it can be expanded into a two-dimensional framework if the need arises. This approach has given rise to three distinct flavors of localization-based WTSN handover: i) middle point, ii) offline learning, and iii) online learning.

- **Middle point:** This handover method was developed as a baseline reference approach. It relies on the STA's position, where it defines a geometric middle point between the APs and establishes a threshold zone to prevent the well-known "ping-pong" effect when STA is equally served by both APs. Consequently, when the STA crosses this middle threshold zone, the HCM instructs the STA to switch AP. However, this method lacks a mechanism to account for the spatial variability in signal propagation. Given the inherent imperfections in signal propagation and the influence of environmental phenomena like reflection, refraction, and diffraction caused by obstacles, leading to multipath effects, the ideal handover point in space will likely not align with the geometric midpoint between the APs. As a result, the use of this method is restricted, and it is only applied in this study as a reference scenario.

- **Offline learning:** This handover method draws inspiration from REM network planning techniques and is structured into two distinct stages [21]:

  - The learning stage where the STA is required to traverse the environment while associating with each available AP. This process enables the controller's handover module to compile a comprehensive REM by collecting the RSSI values associated with each AP within every SS_{ij}.
  - The working stage where the network controller utilizes the acquired REM knowledge in conjunction with the current STA SS position to make informed decisions regarding STA handover, ensuring it connects to the most suitable AP.

  This method leverages offline learning to enhance handover decisions based on the learned radio environment. It also better accounts for the propagation variability as the best AP for each SS is considered. However, as a drawback, if the environment changes, while in the working stage, the learned REM might not be applicable anymore. This limitation not only affects handover but also poses challenges for effective network planning. Despite this drawback, this method is valuable in scenarios where significant environmental changes are not expected, making it a practical approach. This consideration led us to conduct testing and evaluation.

- **Online learning:** In contrast to the previous method, the online learning approach eliminates the need for a dedicated learning stage. Instead, the HCM maintains the REM of each AP, which is continually updated with RSSI values as the STA moves within its coverage area. As an initial handover point, the geometric middle is chosen. However, once enough Spatial Slots are filled within each AP's REM (SS_{ij}), the recorded values are utilized to iteratively fit an RSSI-distance logarithmic model. The Equation 4.2, shows the model, representing the two APs shown in Figure 4.7:

$$\mathrm{RSSI}_{\mathrm{AP_i}} = a_i - b_i \cdot \log_{10}(x) + X_{\theta i} \tag{4.2}$$

where i represents each AP with $i = \{1, 2\}$, x, the distance, a, and b represent the received signal power loss and are determined through the least-squares fitting procedure. The fitting process also yields the standard error X_θ, which is a zero mean Gaussian random variable that reflects the random variation in the path loss due to multipath. This standard error is used in calculating the 95% confidence interval, which is also considered when the optimal handover SS is selected.

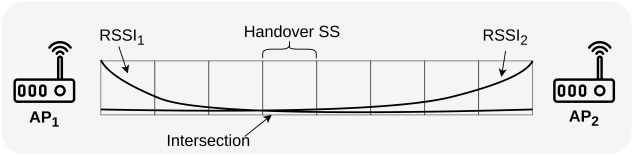


Figure 4.7: Online learning scheme for handover moment selection

The HCM then, by solving the given equation system, identifies the intersection point of the models as seen in Figure 4.7. Based on this intersection, the HCM iteratively selects the optimal handover SS for the STA, ensuring an informed and efficient handover decision. This online approach offers the distinct benefit of seamless adaptation to evolving environmental conditions for the constant update of the handover SS. Nevertheless, a drawback of this method is its similarity to the midpoint approach, where only one SS is chosen for the transition. This could overlook the intricacies of wireless multipath scenarios. Despite this limitation, its practical use is broad, especially in dynamic scenarios where both the STA and the environment are subject to frequent changes.

4.4.3 Handover Moment Selection in Two Dimensions

As discussed in the previous section, incorporating UWB-based localization into Wireless Time-Sensitive Networking (WTSN) eliminates the need to use time slots

for making handover decisions. However, to make this approach practical, it is essential to account for more than one spatial dimension. Following the process described earlier, the subsequent methods are formulated at the HCM. The HCM utilizes the received information to make decisions in two spatial dimensions, using one of the following specific algorithms:

- **Fixed polygon-based:** The first solution we propose does not fully consider the aforementioned challenges, however, it brings useful insights and a comparison point towards our next solution. Therefore, this approach involves dividing the space through which the STA navigates into fixed polygons, establishing well-defined handover borders. To mitigate the ping-pong effect, where the STA frequently switches between different APs along a border, a buffer zone is introduced. This buffer zone replaces the rigid boundary line with two distinct borders, effectively addressing the issue.

  As illustrated in Algorithm 4, this mechanism initiates by receiving the localization information of the STA. Subsequently, it verifies the belonging of the position with one of the polygons. Finally, taking into account the current AP and circumventing the buffer zone, the decision is made.

Algorithm 4 Fixed Polygons

Input: $localization\_coordinates(loc), polygons(pol_i), AP_{now}$
Output: AP_i

Initialization:
Extract loc {Localization coordinates are received at the HCM.}
if $loc \in pol_i$ **then**
{Check if the location point belongs to any of the provided polygons.}
$AP_i \leftarrow pol_i$ {Set the current polygon as the associated access point.}
else
$AP_i \leftarrow 0$ {Assign 0 when the location is in the buffer zone.}
end if
if $(AP_i \neq AP_{now})$ **and** $(AP_i \neq 0)$ **then**
{Ensure a handover only if outside the current AP and not in the buffer zone.}
$AP_{now} \leftarrow AP_i$ Perform handover to AP_{now} {Initiate handover using the HAM.}
end if
return AP_{now}

- **Machine Learning online-based:** Acknowledging the challenges mentioned within a two-dimensional environment, we propose a machine learning (ML)-based modeling approach. This method continuously learns from the captured values of localization and RSSI. The mechanism involves two distinct stages:

1. The initial offline stage involves capturing channel information in tuples (localization, RSSI) to train the models. This is achieved by connecting the STA to an AP_i and relocating the STA over a specified duration, denoted as t. Subsequently, the acquired channel information is utilized to train an artificial neural network (ANN) model. This model is designed to predict the RSSI value corresponding to a specific STA position in relation to an AP_i. Hence, the number of models corresponds to the number of present APs.
2. Following the completion of the initial model training, the subsequent phase involves the online stage, as illustrated in Figure 4.8 and described in detail in the next steps:
   (a) Initially, the HCM receives localization information along with the RSSI from AP_i.
   (b) Subsequently, the decision to update the current model depends on the absolute difference between the sampled RSSI and the RSSI predicted by the AP_i model as seen in Equation 4.3:

$$|RSSI - RSSI_{model_i}| \geq u\_th \tag{4.3}$$

   Where $RSSI_{model_i} = AP_i model(x, y)$.
   This discrepancy sets the update threshold u_th, a parameter that can be adjusted as required.
   (c) If the condition specified in Equation 4.3 is satisfied, the sample (AP_i;x,y; RSSI) is stored in an update cache of size u_ch. The purpose of the update cache is to promote batch updates of model(s) instead of individual updates, thereby minimizing the overall model update delay.
   The model is updated using the online gradient descent method with a constant learning rate. Where the parameters of the network (weights and biases) are updated following Equation 4.4:

$$\theta_{new} = \theta_{old} - \eta * \nabla_\theta Loss \tag{4.4}$$

   where, θ is the parameter, η is the learning rate and $\nabla_\theta Loss$ is the gradient of the loss with respect to the parameter [22].
   (d) If Equation 4.3 is not met for the given sample, the subsequent step involves examining whether the sample belongs to a buffer zone. The buffer zone is created by identifying the intersection lines of the models and its purpose is to avoid the ping-pong effect.
   (e) If the STA is not within a buffer zone, the subsequent step is to predict an RSSI value for each available model and compare them to identify the highest one.

(f) Finally, if the best AP model does not correspond to the current AP (AP_{now}), a handover command is sent to the HAM, specifying to which AP the STA should initiate the handover.

In summary, the outlined procedure shows adaptability to wireless environment fluctuations using STA localization and RSSI measurements. Importantly, enhancing the models could involve multiple STAs, not just a single one, which can improve accuracy. During the online stage, responsiveness depends on correctly setting η and u_th to manage RSSI changes effectively.

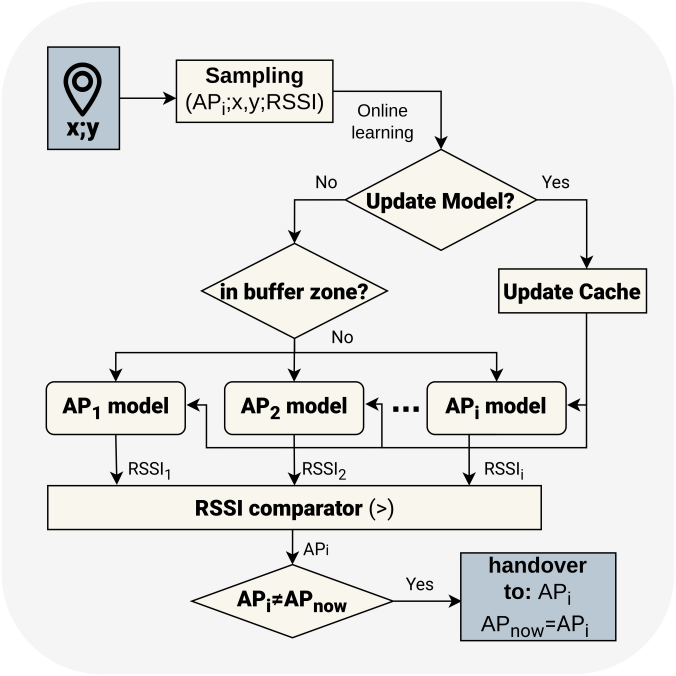


Figure 4.8: Flow diagram of the machine learning online stage for handover moment selection

4.5 Results

4.5.1 Setup Description

The proposed WTSN handover procedure has been implemented and tested using the openwifi IEEE802.11/Wi-Fi baseband chip/FPGA design. As shown in Figure 4.9, on the mobile wireless client side, we have employed a Xilinx Zedboard

equipped with the AD-FMCOMMS4-EBZ transceiver, serving as the STA. This is connected via Ethernet to an Intel Next Unit Computing (NUC) end node, which employs the Robot Operating System 2 (ROS 2) to drive a 4WD Rover Zero 3. Each AP in our setup utilizes a Xilinx ZC706 board along with an AD-FMCOMMS4-EBZ transceiver. These APs are connected through Ethernet to a Qotom Q818GE Linux TSN Switch, which in turn is connected via Ethernet to another NUC, functioning as the CNC/CUC.

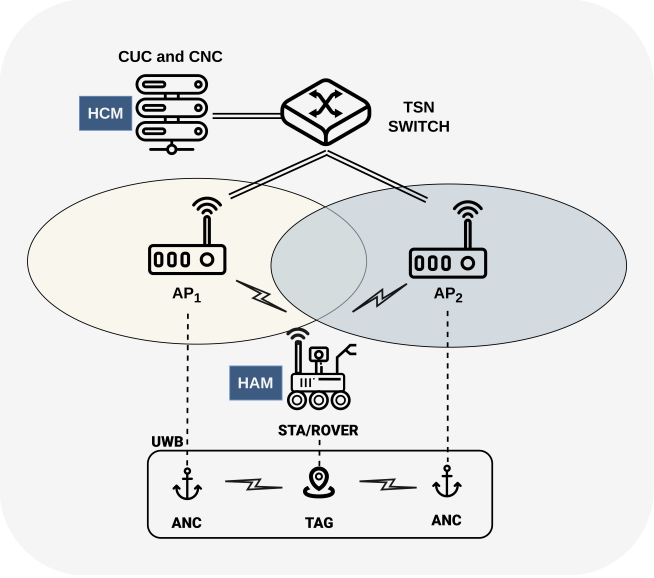


Figure 4.9: Topology of the WTSN handover testing setup

The UWB localization system comprises both Anchors (ANC) at known fixed positions and a mobile TAG, which is attached to the STA board via a Universal Serial Bus (USB) interface. The UWB hardware is built on the Wi-PoS platform, with localization precision within the 5-centimeter range. This system is further bolstered by a sub-GHz wireless infrastructure for seamless communication, as outlined in a study by [23]. This setup was developed and tested within a warehouse-style environment of 12x25 meters, featuring metallic shelving units but devoid of any obstructions between the APs.

Lastly, the management and configuration of the network is established using a ZeroMQ PUB/SUB model, illustrated in Figure 4.10, facilitating message exchange between the HCM and HAM. On one front, the HCM can initiate an AP handover when necessary. Conversely, the HAM disseminates messages to the HCM, encompassing its real-time coordinates and the prevailing RSSI value.

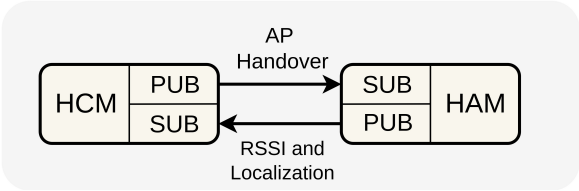


Figure 4.10: ZeroMQ model for handover communication

4.5.2 Handover Speed

To assess the improvements outlined in Section 4.4.1, we conducted an uplink traffic test. The experiment platform can be viewed in Figure 4.11. It is worth noting that we did not implement any specific TSN scheduling scheme in the gating system, ensuring that we measure only the handover delay and not any possible additional delay due to time slot access delays. For this measurement, the STA/Rover remained stationary at the central point between the APs. The distance separating the APs, 'd', is 6 meters. and to avoid external interference, the APs operated on Wi-Fi 6E channels (specifically, 5975 MHz and 5995 MHz). Additionally, we manually configured the handover triggers at 50-ms intervals and the traffic consisted of 100k pings generated every 1ms going from STA to CNC, where only ping requests are considered.

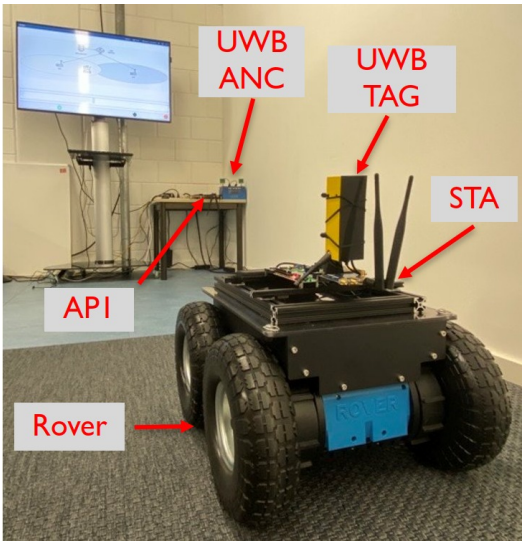


Figure 4.11: Photograph of the handover experiment platform

To avoid the need for time synchronization between the APs and the STA for

measuring handover delay, which could introduce additional traffic and require precise synchronization, we internally assessed the interframe arrival time of pings within the STA. The STA driver has two interrupt functions: one is called upon receiving a frame for transmission from higher layers, and the other notifies higher layers when a frame has been successfully transmitted.

By comparing the frame arrival times in both interrupt functions, we were able to measure the impact of handover delay on traffic. Figure 4.12 presents the results. The three curves in the graph represent the cumulative distribution function (CDF), of frame interarrival times for the following scenarios: no handover, slow handover (handover before the introduced modifications), and enhanced handover (handover with modifications). As reported in the literature, typical handover results in delays exceeding 100 ms [14]. However, in the case of enhanced handover, the frame interarrival time was reduced to as low as 13 ms.

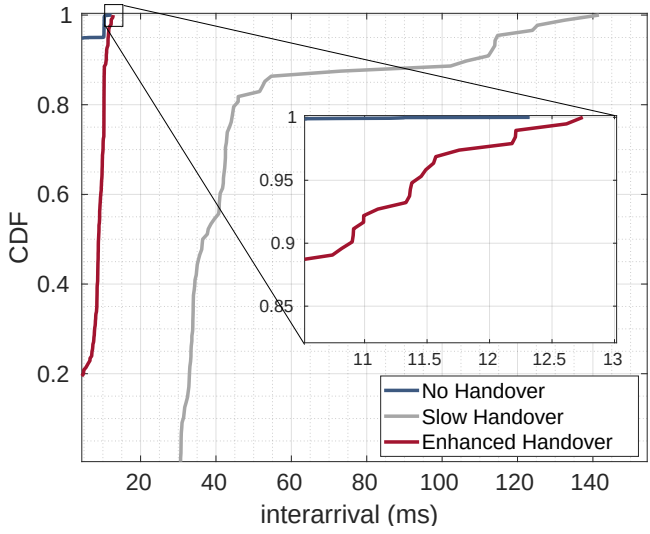


Figure 4.12: CDF of uplink handover delay results

To provide an application-focused perspective on the handover delay of the proposed enhanced handover solution, we conducted end-to-end tests. These tests encompassed both UL and DL scenarios, involving UDP traffic flowing between the CNC/CUC and the Rover in Figure 4.9. To quantify the handover delay, we employed the frame interval time, utilizing 100k UDP frames transmitted at a generation rate of 1 ms. Furthermore, we manually initiated handovers at 50-ms intervals, from the HCM, to update both the wired and wireless segments of the TSN. The results of these tests can be seen in Figure 4.13.

In Figure 4.13, we initially showcase two baseline measurements conducted without any handover (No HO) for both the DL and UL scenarios. Subsequently,

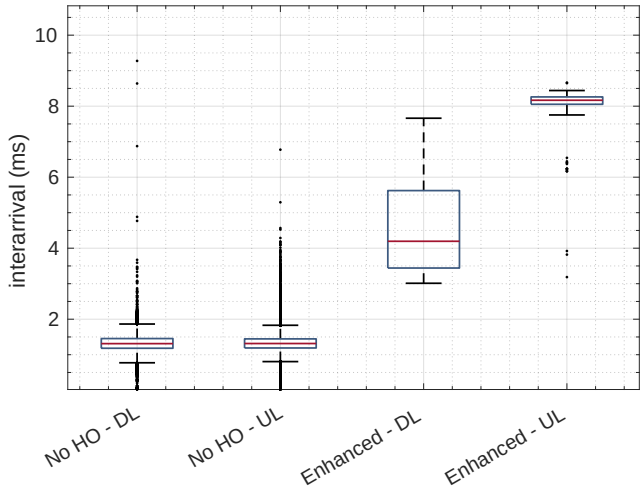


Figure 4.13: End-to-end Handover Delay

we present the enhanced measurements for DL and UL traffic under the handover condition, revealing median values of 4 and 8 ms, respectively. It is important to take into account that these results are influenced not only by handover delay but also by the inherent variable processing delays at various points in the network, including the STA, AP, switch, and CUC/CNC or Rover. In conclusion, while the significance of a low handover delay is evident, it is crucial to contextualize it within the traffic dynamics. Depending on the application frame generation rate, there is a possibility that a frame may need to be transmitted during handover. In the case of uplink (UL), illustrated in Figure 4.5, the CCA threshold mandates the frame to wait in the buffer, causing a delay in its transmission. Conversely, in the case of downlink (DL), the frame may face the risk of being lost. This underscores the imperative to minimize handover delay, particularly in scenarios where the timely transmission of frames is critical.

4.5.3 Handover Moment Selection in One Dimension

To evaluate and compare the handover moment selection methods outlined in Section 4.4.2, we opted for a realistic experimental approach. In this context, we established a specific movement pattern for the Rover, as illustrated in Figure 4.9. The Rover follows a predefined path, traversing between the APs at a constant speed for a duration of 150 seconds in each test run. The rest of the parameters are summarized in Table 4.1.

While the primary objective of this experiment is to assess handover perfor-

Table 4.1: Measurement parameters used in one-dimensional handover selection algorithms

Parameter	Value
Measurement time	150 s
APs distance (d)	6 m
APs frequencies	5975 & 5995 MHz
Data rate	26 Mbps
Bandwidth	20 MHz
iperf traffic	UL&DL at 5 Mbps
Rover speed	0.4 m/s
SS_{size}	30 cm
α	0.8
t_s	40 ms

mance, it is essential to consider the performance throughout the entire trajectory between the APs from an application perspective. Therefore, the following results measure various performance metrics, including throughput, packet loss, and jitter in both the UL and DL using end-to-end *iperf* traffic. To prevent automatic adjustments in the transmission rate made by the Linux Minstrel algorithm, we set the transmission rate to a constant 26 Mbps in all wireless nodes.

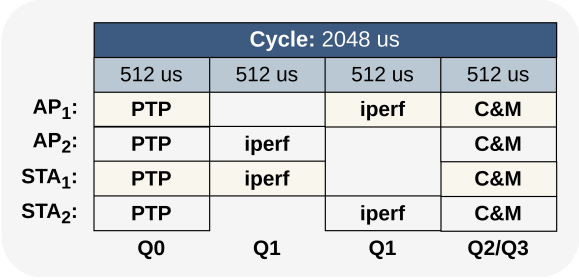


Figure 4.14: WTSN transmission schedule used in one-dimensional handover tests

The transmission schedule employed for the WTSN is detailed in Figure 4.14. This schedule consists of a cycle with a duration of 2048 μs, divided into time slots of 512 μs. The first time slot, assigned to queue 0 in all nodes, is dedicated to time synchronization traffic, which utilizes PTP, as previously described. If the handover delay, is small enough, there are no visible affections on time synchronization. The second and third time slots are allocated for UL/DL *iperf* UDP traffic between the STA connected to one of the APs, respectively, utilizing queue 1. The final time slot, which involves queues 2 and 3, is designated for configuration and management (C&M) traffic, as depicted in Figure 4.10.

Considering the 26 Mbps fixed physical data and the applied schedule, we

configured the *iperf* traffic for both the UL and DL to operate at a realistic 5 Mbps.

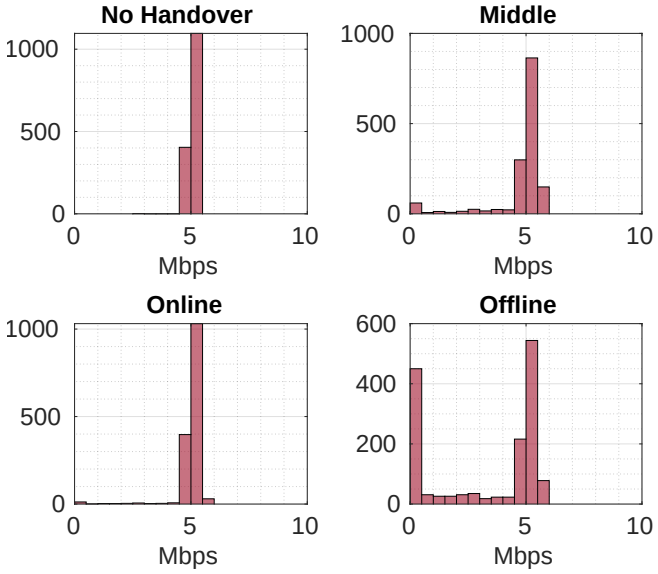


Figure 4.15: Downlink throughput distribution results during handovers

Figure 4.15 illustrates a DL throughput comparison across four examined scenarios. The initial scenario, *No handover*, serves as the reference case and involves a stationary STA associated with an AP. Subsequently, the following three scenarios, namely *Middle*, *Online*, and *Offline* are aligned with the mechanisms elaborated upon in Section 4.4.2. In this Figure, it is evident that the reference scenario, along with the *Online* approach, effectively maintains a higher level of requested bandwidth compared to the *Middle* and *Offline* approaches.

The results obtained for DL throughput align with the findings presented in Figures 4.16 and 4.17, depicting packet loss and jitter, respectively. It is evident that the *Online* method consistently outperforms the other approaches in these metrics as well.

As anticipated, the results pertaining to throughput, packet loss, and jitter in the DL exhibit a similar pattern to those observed in the UL as depicted in Figures 4.18, 4.19, and 4.20. Of particular note is the suboptimal performance of the *Offline* approach, which is elaborated upon in Section 4.4.2. The *Offline* mechanism initially constructs a REM using the best RSSI information from the APs' REMs. This approach results in multiple potential handover points within the general REM. Contrary to expectations, the outcome of this strategy, as indicated by the results, deteriorates overall link quality instead of enhancing it.

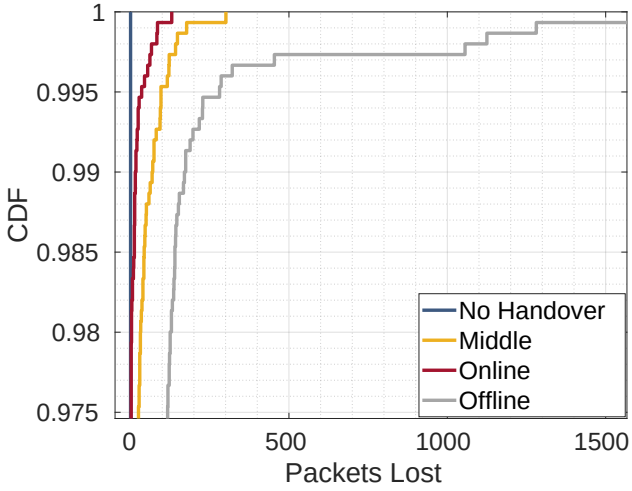


Figure 4.16: CDF results of downlink packet loss during handovers

4.5.4 Handover Moment Selection in Two Dimensions

To evaluate the handover moment selection algorithm described in 4.4.3, we utilized the topology illustrated in Figure 4.9 adding an extra AP. Table 4.2 provides a summary of the parameters employed for the testing. Figure 4.21 depicts the TSN schedule utilized, where STA_i corresponds to the STA connected to the AP_i. In this schedule, the first timeslot in queue 0 is allocated for time synchronization via precision time protocol (PTP). Subsequently, queue 1 is dedicated to iperf testing traffic, while queues 2 and 3 in the final timeslot are designated for management and configuration traffic.

Table 4.2: Measurement parameters used in two-dimensional handover selection algorithms

Parameter	**Value**
Measurement time	150 s
APs distance (d)	3 m
APs channels	5955, 5975 & 5995 MHz
Data rate	26 Mbps
Bandwidth	20 MHz
iperf traffic	UL&DL at 5 Mbps
Rover speed	0.4 m/s
t_s	40 ms

The ANN employed in our study comprised five layers with 64, 32, 16, 32, and

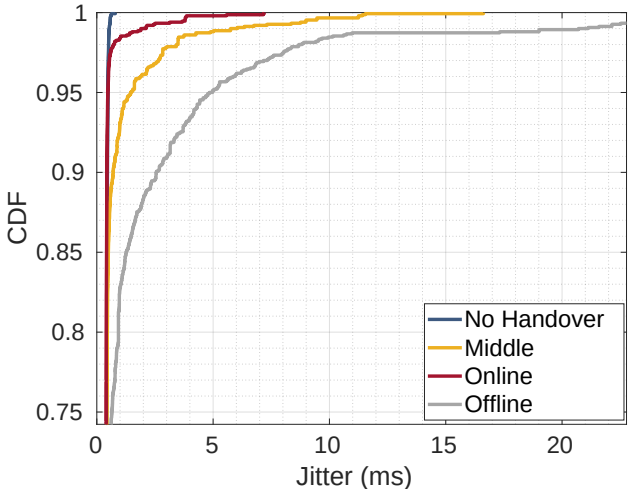


Figure 4.17: CDF results of downlink jitter during handovers

1 neurons, respectively, utilizing a rectified linear activation function, also an u_th of 10 dBm, and η of 0.0001 were used for the online training. Figure 4.22 illustrates the training measurements conducted with the mobile STA connected to each AP, along with the resultant models utilized in the offline component of the handover selection algorithm. As observed, the APs are positioned in a triangular formation. For the testing, the mobile STA was programmed to follow a fixed trajectory around the testing zone, ensuring that it crosses all three coverage zones. Another essential step in the algorithm's functionality is identifying the buffer zone during model updates. This is achieved through image processing tools, initially by delineating the edges of the APs' optimal coverage zones, as illustrated in Figure 4.23.

Figure 4.24 illustrates the throughput distributions of iperf traffic in three scenarios: no handover with the STA associated with only one AP, with the fixed polygon algorithm, and with the machine learning online algorithm for both UL and DL traffic. In the UL case, the distributions appear quite similar, whereas in the DL case, the online scenario seems to resemble the no handover and fixed polygon cases more closely.

Another crucial network variable, particularly for TSN, is jitter. Figures 4.26 and 4.25 present the jitter results of the aforementioned scenarios in both downlink (DL) and uplink (UL), respectively. It is evident the advantage of the online method compared to the fixed one. This advantage, observed in both throughput and jitter, stems not only from optimal handover moment selection but also from the reduction in the number of handovers due to updates in the APs' coverage areas. Specifically, in case of online learning method, the number of handover moments

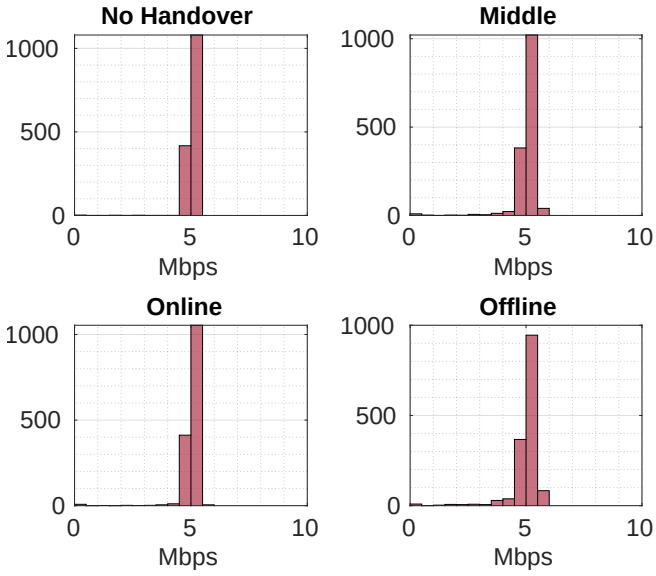


Figure 4.18: Uplink throughput distribution results during handovers

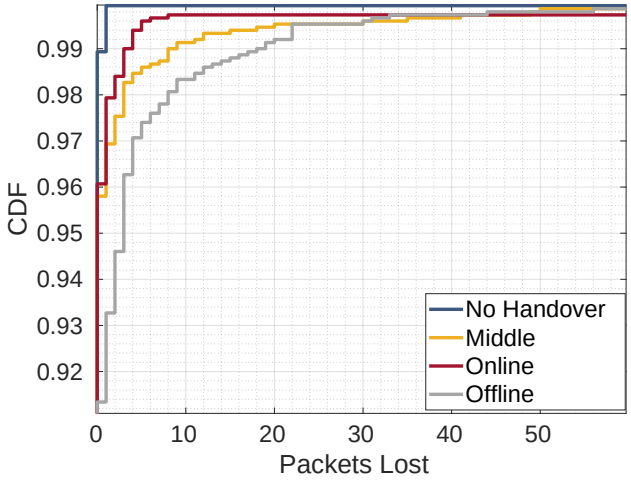


Figure 4.19: CDF results of uplink packet loss during handovers

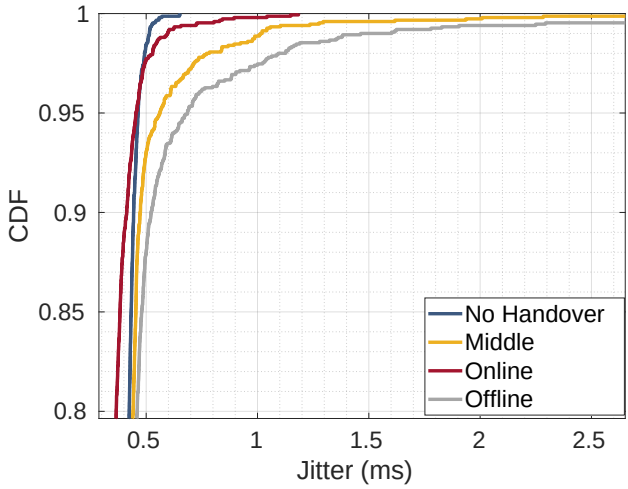


Figure 4.20: CDF results of uplink jitter during handovers

	Cycle: 2048 us			
	512 us	512 us	512 us	512 us
AP_1:	**PTP**		**iperf**	**C&M**
AP_2:	**PTP**	**iperf**		**C&M**
AP_3:	**PTP**		**iperf**	**C&M**
STA_1:	**PTP**	**iperf**		**C&M**
STA_2:	**PTP**		**iperf**	**C&M**
STA_3:	**PTP**	**iperf**		**C&M**
	Q0	**Q1**	**Q1**	**Q2/Q3**

Figure 4.21: WTSN transmission schedule used in two-dimensional tests

during one measurement decreased from 46 to 40 and from 46 to 38 for UL and DL measurements, respectively.

4.6 Conclusion

By developing a state machine procedure specifically designed for handover in Wi-Fi-based Wireless Time-Sensitive Networking, optimizing handover delays, and integrating a UWB machine-learning algorithm to predict RSSI variations across different locations, an efficient handover solution has been achieved. This approach ensures optimal handover timing while preventing interference with time-sensitive

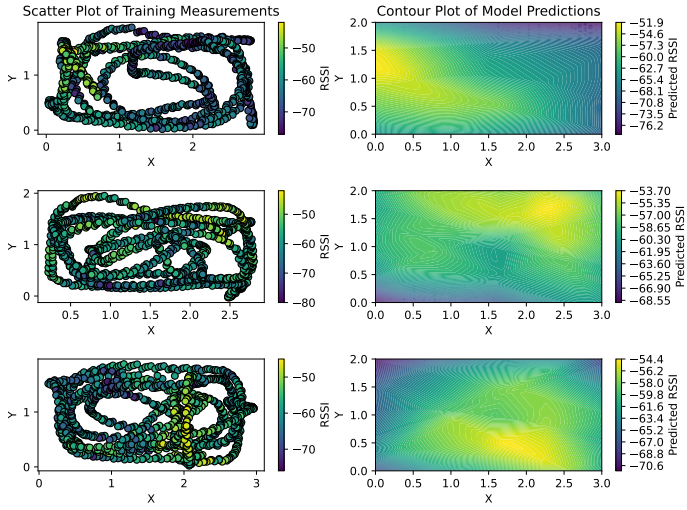


Figure 4.22: Graphical representation of measurements and model predictions during offline learning

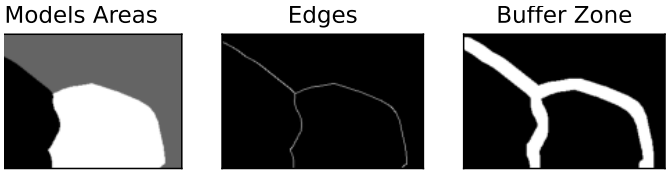


Figure 4.23: Example of buffer zone generation

traffic.

Empirical results demonstrate the effectiveness of combining a fast handover mechanism ($<$ 10 ms) enabled by openwifi with a self-learning algorithm for selecting the optimal handover moment. This combination reduces jitter and improves throughput, both essential for enhancing determinism in TSN.

Although the results establish a strong basis for the proposed solutions, it is essential to recognize their limitations. A primary constraint is the processing power required for real-time model updates. While strategies like the update cache help alleviate this, they may still introduce delays in highly dynamic environments, affecting responsiveness. Additionally, although RSSI proves effective in interference-free scenarios, its reliability diminishes under more complex conditions. This limits the method's broader applicability and highlights the need to include other metrics, such as packet error rate, signal-to-interference-plus-noise ratio, and channel state information. Another challenge lies in the REM models'

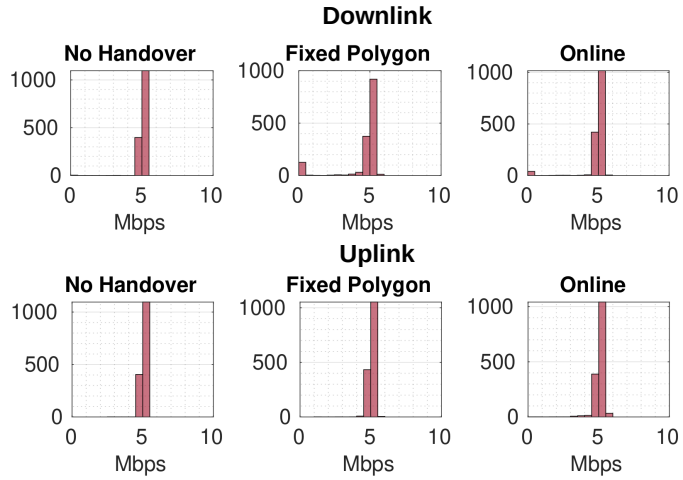


Figure 4.24: Downlink and uplink throughput distribution results during handovers in two dimensions

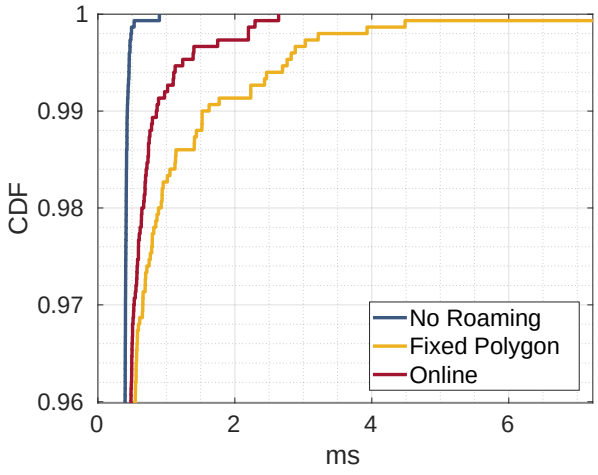


Figure 4.25: CDF results of uplink jitter during handovers in two dimensions

reliance on STA samples. When devices are static or operate within limited areas, model adaptability suffers. Addressing this would require external mechanisms to maintain the REM independently of STA activity.

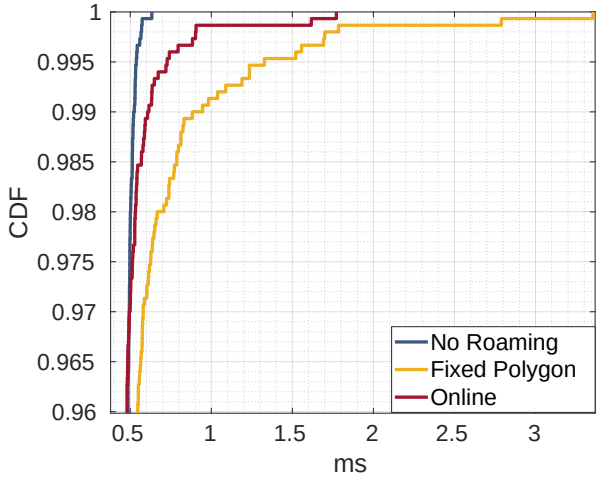


Figure 4.26: CDF results of downlink jitter during handovers in two dimensions

References

[1] M. Warburton, M. Mon-Williams, F. Mushtaq, and J. R. Morehead. *Measuring motion-to-photon latency for sensorimotor experiments with virtual reality systems*. bioRxiv, page 2022.06.24.497509, 6 2022. doi:10.1101/2022.06.24.497509.

[2] *What is Motion-To-Photon Latency?* Available from: http://www.chioka.in/what-is-motion-to-photon-latency/.

[3] *802.11k-2008 - IEEE Standard for Information technology– Local and metropolitan area networks– Specific requirements– Part 11: Wireless LAN Medium Access Control (MAC)and Physical Layer (PHY) Specifications Amendment 1: Radio Resource Measurement of Wireless LANs — IEEE Standard — IEEE Xplore*. Available from: https://ieeexplore.ieee.org/document/4544755.

[4] *802.11v-2011 - IEEE Standard for Information technology– Local and metropolitan area networks– Specific requirements– Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications Amendment 8: IEEE 802.11 Wireless Network Management — IEEE Standard — IEEE Xplore*. Available from: https://ieeexplore.ieee.org/document/5716530.

[5] *802.11-2007 - IEEE Standard for Information Technology - Telecommunications and Information Exchange Between Systems - Local and Metropolitan Area Networks - Specific Requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications — IEEE Standard — IEEE Xplore*. Available from: https://ieeexplore.ieee.org/document/4248378.

[6] *802.11r-2008 - IEEE Standard for Information technology– Local and metropolitan area networks– Specific requirements– Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 2: Fast Basic Service Set (BSS) Transition — IEEE Standard — IEEE Xplore*. Available from: https://ieeexplore.ieee.org/document/4573292.

[7] Cisco. *Understanding Wireless Roam and Location Services Instructor Materials CCNP Enterprise: Core Networking*. 2016.

[8] *P802.11be/D4.0, Jul 2023 - IEEE Draft Standard for Information technology– Telecommunications and information exchange between systems Local and metropolitan area networks–Specific requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*

Amendment 8: Enhancements for Extremely High Throughput (EHT) — IEEE Standard — IEEE Xplore. Available from: https://ieeexplore.ieee.org/document/10265800.

[9] L. Wisniewski, H. Trsek, I. Dominguez-Jaimes, A. Nagy, R. Exel, and N. Kerö. *Location-based handover in cellular IEEE 802.11 networks for factory automation*. In Proceedings of the 15th IEEE International Conference on Emerging Technologies and Factory Automation, ETFA 2010, 2010. doi:10.1109/ETFA.2010.5641296.

[10] W. Xiaofei. *IEEE802.11 AIML TIG Technical Report Draft*, 2023.

[11] A. Sarma, R. K. Gupta, and S. Nandi. *A zone based interleaved scanning technique for fast handoff in IEEE 802.11 wireless networks*. In I-SPAN 2009 - The 10th International Symposium on Pervasive Systems, Algorithms, and Networks, pages 232–237, 2009. doi:10.1109/I-SPAN.2009.108.

[12] I. Ramani and S. Savage. *SyncScan: Practical Fast Handoff for 802.11 Infrastructure Networks*. 2005. Available from: http://activecampus.ucsd.edu, doi:10.1109/INFCOM.2005.1497933.

[13] Z. Fernandez, O. Seijo, M. Mendicute, and I. Val. *Analysis and evaluation of a wired/wireless hybrid architecture for distributed control systems with mobility requirements*. IEEE Access, 7:95915–95931, 2019. doi:10.1109/ACCESS.2019.2927298.

[14] C. Pei, Z. Wang, Y. Zhao, Z. Wang, Y. Meng, D. Pei, Y. Peng, W. Tang, and X. Qu. *Why it takes so long to connect to a WiFi access point*. Proceedings - IEEE INFOCOM, 10 2017. doi:10.1109/INFOCOM.2017.8057164.

[15] S. Sudhakaran, I. Ali, M. Eisen, J. Perez-Ramirez, V. Frascolla, and D. Cavalcanti. *Zero-Delay Roaming for Mobile Robots enabled by Wireless TSN Redundancy*, 2023.

[16] P. Avila-Campos, J. Haxhibeqiri, M. Girmay, I. Moerman, and J. Hoebeke. *Residual Service Time Optimization for legacy Wireless-TSN end nodes*. In WiMob 2023 - The 19th International Conference on Wireless and Mobile Computing, Networking and Communications., 2023.

[17] M. Aslam, W. Liu, X. Jiao, J. Haxhibeqiri, G. Miranda, J. Hoebeke, J. Marquez-Barja, and I. Moerman. *Hardware Efficient Clock Synchronization Across Wi-Fi and Ethernet-Based Network Using PTP*. IEEE Transactions on Industrial Informatics, 18:3808–3819, 6 2022. doi:10.1109/TII.2021.3120005.

[18] P. Avila-Campos, J. Haxhibeqiri, I. Moerman, and J. Hoebeke. *Impactless Beacon-Based Wireless TSN Association Procedure*. In 18th IEEE International Conference on Factory Communication Systems, 2022.

[19] X. Jiao, W. Liu, M. Mehari, M. Aslam, and I. Moerman. *Openwifi: A free and open-source IEEE802.11 SDR implementation on SoC*. IEEE Vehicular Technology Conference, 2020-May, 5 2020. doi:10.1109/VTC2020-SPRING48590.2020.9128614.

[20] P. A. Shah, M. Yousaf, A. Qayyum, and H. B. Hasbullah. *Performance comparison of end-to-end mobility management protocols for TCP*. Journal of Network and Computer Applications, 35:1657–1673, 11 2012. doi:10.1016/J.JNCA.2012.05.002.

[21] R. K. Marjan, M. H. Aldulaimi, and R. S. H. Al-Naseri. *Design and evaluation of Wi-Fi Network Heat map generator*. In NICST 2019 - 1st Al-Noor International Conference for Science and Technology, pages 14–19. Institute of Electrical and Electronics Engineers Inc., 10 2019. doi:10.1109/NICST49484.2019.9043825.

[22] B. Mcmahan and Y. Huang. *The Online Gradient Descent with adaptive learning rate*. 2012.

[23] B. V. Herbruggen, B. Jooris, J. Rossey, M. Ridolfi, N. Macoir, Q. V. D. Brande, S. Lemey, and E. D. Poorter. *Wi-PoS: A Low-Cost, Open Source Ultra-Wideband (UWB) Hardware Platform with Long Range Sub-GHz Backbone*. Sensors 2019, Vol. 19, Page 1548, 19:1548, 3 2019. Available from: https://www.mdpi.com/1424-8220/19/7/1548/htmhttps://www.mdpi.com/1424-8220/19/7/1548, doi:10.3390/S19071548.

5

A Predictive Management Framework for Low-Latency and Low-Jitter Wireless TSN Systems

"According to Darwin's Origin of Species, it is not the most intellectual of the species that survives; it is not the strongest that survives; but the species that survives is the one that is able best to adapt and adjust to the changing environment in which it finds itself"

–Leon C. Megginson

Balancing traffic characteristics and requirements with available resources remains an unsolved NP-hard optimization problem in wired TSN. The inherent variability of wireless networks in WTSN adds further complexity to this challenge. While current technology cannot offer a universal solution, an application-specific and practical approach is within reach. Modern learning algorithms and paradigms, such as digital twins, might enable the modeling of wireless link behavior and processing delays. This capability supports proactive scheduling decisions, bringing us closer to a functional WTSN scheduler. Building on the residual service time problem, this chapter introduces a practical network management framework as a viable enabler for traffic scheduling in WTSN.

Abstract

Time-sensitive networking (TSN) ensures traffic delivery guarantees, making it essential for applications in industrial automation, multimedia, and automotive systems. While TSN's benefits are well-established in wired networks, wireless systems face unique challenges such as delays, interference, and unstable links. Extending TSN to wireless networks adds complexity, particularly in traffic scheduling, due to the variability of wireless conditions. This chapter proposes a data-driven solution using the digital twin (DT) concept to enhance WTSN management and address this variability. Part of the proposed WTSN-DT management framework is implemented and tested in a real IEEE 802.11 TSN-based setup. Our findings demonstrate that by applying various traffic pattern learning techniques, our framework effectively achieves TSN traffic protection benefits while reducing latency a 96% at 90th percentile, caused by residual service time.

5.1 Introduction

Traffic scheduling aims to balance traffic characteristics and requirements, available resources, and the inherent variability of wireless networks. Accurate WTSN scheduling must also account for factors like variable delays in frame processing, channel propagation, transmission, time synchronization accuracy, and frame generation in heterogeneous hardware [1]. Ignoring these factors reduces determinism, affecting end-to-end latency and jitter, and rendering the system unsuitable for time-sensitive applications [2].

A key reason for this lack of realism is the absence of a framework that accurately models network complexities without oversimplifying, as traditional approaches often do [3]. In this work, we propose a management framework to bridge this gap. To demonstrate its potential, we focus on the residual service time (RST) problem, as illustrated in Figure 5.1. The figure highlights how mismatches between frame generation offsets and variable frame processing delays at TSN nodes affect the frame time slot access delay, or RST. These mismatches increase end-to-end latency and its variability, resulting in jitter [4].

Network jitter refers to the variation in latency or delay of packet delivery over a network. Jitter challenges the TSN core goal of achieving predictable, low-latency communication. If not addressed, it can block critical traffic from accessing its reserved time slot, cause collisions, or introduce delays, thereby compromising the system's determinism. Therefore, minimizing jitter is not merely beneficial but essential for TSN to operate as intended [5].

Current scheduling demonstrations frequently showcase examples with short Gate Control List (GCL) cycle times to reduce RST, effectively masking the issue. However, this approach does not accurately reflect real-world conditions and more-

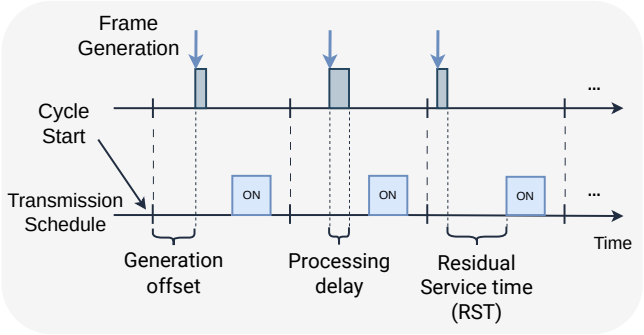


Figure 5.1: Representation of the residual service time (RST) challenge in a TSN node

over will not scale in large networks. Alternatively, the strategy proposed in 3GPP Release 18 holds frames in the buffer until they reach a defined maximum delay. While this reduces jitter, it increases end-to-end latency, limiting its usability in time-sensitive applications [6, 7].

To overcome these limitations, a data-driven, real-time management framework is essential. The Digital Twin (DT) concept has emerged as a promising solution for addressing this challenge [8, 9]. The IETF and ITU-T recognize Network-DT as a key technology for achieving closed-loop network automation, proposing a reference architecture that fully integrates DT within a general network controller framework. However, these efforts still lack specific guidance for TSN [10, 11].

In this chapter, we explore the feasibility of combining a IEEE 802.11-based WTSN with an element of a DT (WTSN-DT), enabling better traffic scheduling and addressing the critical challenge of reducing wireless communication link jitter and end-to-end latency introduced by RST. This is particularly important when variable processing delays from heterogeneous nodes and non-time-synchronized frame generators, such as non-TSN end devices, are present. Non-TSN end nodes are likely to be among the first to benefit from WTSN advantages. Therefore, solutions like the one proposed here aim to bridge this gap. Specifically, the contributions of this work are as follows:

- Propose a Wi-Fi-based WTSN-DT theoretical management reference framework that integrates real-time measurements, modeling, scheduling, and setting.
- Investigate three analytical and Machine Learning (ML)-based predictive modeling strategies for real-world cyclic traffic patterns based on frame arrival time series.
- Implement a WTSN-DT element and a low-link-latency-focused scheduler

in a real Wi-Fi-based WTSN setup using openwifi [12].

The remainder of the chapter is organized as follows. Section 5.2 presents related work on DT and WTSN. Section 5.3 describes the proposed integration of TSN control and DT in detail. Section 5.4 explains the proposed analytical and AI-based predictive traffic modeling strategies. Section 5.5 presents the setup and results from both the simulated and practical implementations. Finally, Section 5.6 concludes the chapter.

5.2 Related Works

Interest in introducing the DT concept into networking is not new. The Networking Digital Twin has been proposed as a powerful tool to connect physical and virtual digital spaces. According to ITU-T, it enables real-time management of complex networks by using physical models to replicate their states, providing high fidelity and seamless integration [10, 11, 13].

An example of this is found in [9], where the authors propose a DT data pipeline architecture that demonstrates an integrated structure for network modeling flows. They also introduce a feature extraction technique for traffic modeling to optimize learning time. Although not focused on TSN, this work highlights gaps in data flow integration when combining networking, digital twins, and time series modeling. These gaps are even more pronounced in the TSN environment.

Efforts to use DT for specific issues, such as network jitter, have been explored in works like [8]. This study highlights the need for solutions in wireless networks, particularly those aligned with 3GPP standards. It proposes training a deep learning agent to learn inter-packet arrival patterns and control buffer times to reduce jitter. While the approach effectively minimizes jitter, it increases end-to-end latency, making it less suitable for TSN, where bounded low latency is a critical requirement.

Another challenge in integrating DT with wireless networks is reliability. In the context of Wi-Fi 7 multi-link operation (MLO), the authors of [14] propose WiTwin, a centralized DT designed to enable seamless handovers. Handovers are essential for time-sensitive applications, as they must support mobility while maintaining performance guarantees. However, while the study addresses time-sensitivity, it does not focus on TSN-specific control elements or their interactions.

Closer to the current work, [15] presents a broader vision for an end-to-end control plane supporting multi-domain and multi-technology communication systems, including IEEE802.11, with their federated DT model sharing similar goals to this work for the IEEE802.11 case but still far from practical implementation. Similarly, other proposals explore integrating TSN and DT with future technologies like 6G, such as [16], which identifies DT as a core component for integrating TSN with 6G. While their theoretical approach addresses challenges like network complexity,

infrastructure, and data collection, both approaches share the common issue of lacking a detailed TSN DT data model, which is crucial for real-world adoption.

Regarding the specific issue of RST, existing research remains limited. Current WTSN implementations typically manage a small number of traffic flows with similar or identical generation cycles and often lack dedicated time slots for time synchronization or network management. When the GCL cycle is short, jitter exists but is less noticeable. However, as GCL cycle durations increase, jitter becomes a significant problem [17].

Close to the present work, in [18], the authors address TSN jittering from a missynchronization perspective in non-TSN end devices by introducing TALESS, a solution that compensates for drift between TSN and non-TSN devices by adjusting the transmission schedule. A key feature of their approach is a drift detector (DD) that continuously monitors synchronization drift at the reception port and calculates the drift percentage.

Our approach differs in several key aspects. First, we do not consider frame losses, as synchronization drift does not cause frame loss when the generation and GCL cycles are similar. Second, our focus on wireless communication introduces a different topology. While our capture mechanism is decentralized, it incorporates both generation and processing delays in the TSN node. This enables the network manager to accurately model traffic and schedule transmission time slot positions and sizes. Additionally, we do not assume that the drift probability distribution follows a fixed distribution. Instead, our approach dynamically models it using various techniques.

Finally, the present problem was partially addressed in our previous work from a theoretical perspective in [4]. In that study, using traffic frame arrival captures in a station (STA) and MATLAB simulations, we proposed three mechanisms, Q-learning, active update, and polynomial forecasting, to update the transmission time slot position of the STA and reduce link latency and jitter.

However, the current approach advances this earlier work in several significant ways. First, in architectural terms, the DT concept has been introduced and integrated with TSN control elements. This enabled the implementation of a distributed online data-capturing system. Furthermore, the previously proposed pseudo-synchronization mechanism for time synchronization between the end node and the STA has been removed, as it was deemed unrealistic. Finally, this work includes a real implementation of a Wireless TSN (WTSN), demonstrating the feasibility of the proposed concepts.

In conclusion, the DT concept is increasingly seen as a key enabler for network control across various levels, including general networking and TSN-specific applications. However, real-world implementations that integrate all required components and demonstrate their functionality remain scarce, both for 3GPP 5G/6G and IEEE802.11. Specifically, in the context of RST, existing implementations often

rely on unrealistic traffic schedules that do not account for jitter. Most advanced studies in this field focus on wired TSN and remain theoretical, oversimplifying the problem. They frequently model frame arrival distributions without considering the time component or assume normal distribution patterns, which fail to reflect real-world conditions.

5.3 WTSN-DT System Design

As shown in Figure 5.2, the proposed integration does not change the existing centralized configuration management defined by the TSN IEEE 802.1Qcc standard [19]. Where the Centralized User Configuration (CUC) retrieves end-node capabilities, handles application stream requirements, and communicates stream requests to the Centralized Network Configurator (CNC), which configures end-stations according to scheduling results. Rather than altering this structure, our integration introduces the DT as an additional component. The DT acts as a virtual replica of the WTSN, encompassing devices, communication paths, and control logic. It models the behavior of devices such as STAs or APs, control flows, and network communication, enabling testing and validation.

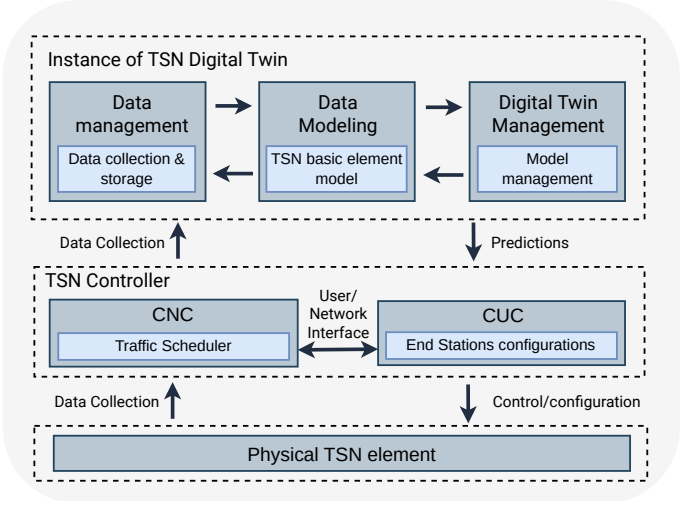


Figure 5.2: Proposed reference management architecture of a WTSN-DT

Currently, there is no standard definition of a DT for networks. However, the IETF specifies that a digital twin for networks should include four key elements: data, mapping, models, and interfaces [11]. Real-time data is collected from physical TSN elements, and a representation of this data is generated through a data model. Two interfaces are typically defined: one between the DT and the

network infrastructure, and another between the DT and applications. In WTSN, these interfaces are managed by the CNC and CUC.

In this context, the tasks of each component could be defined as follows:

- Digital Twin (DT): The DT serves as a key component for system optimization. It replicates the WTSN environment by modeling devices, control flows, and communication paths. This allows it to emulate system behavior and predict performance under varying traffic loads and potential failures. The DT also provides real-time feedback on the network's current state to the CNC and CUC. By comparing predicted behavior with real-time data, it identifies anomalies and assists in preventing issues before they occur.

- CNC: The CNC manages TSN configuration to meet time-sensitive communication requirements, such as latency and jitter. It uses DT predictions to optimize traffic schedules, time slots, and traffic shaping. Additionally, the CNC configures TSN switches and devices, monitors the network, and adapts schedules and routes in response to deviations or failures based on DT feedback.

- CUC: The CUC interfaces with end devices, such as sensors, actuators, and controllers, to collect application requirements and relay them to the CNC as well as to DT. It gathers communication needs, including cycle times, jitter tolerance, and critical paths—and synchronizes end device operations using DT predictions. The CUC also supplies real-time performance data from end devices to the DT for ongoing analysis and system improvement.

This architecture marks a transition toward a complete WTSN-DT system. While the CUC and CNC retain their roles as defined in the IEEE 802.1Qcc standard, tasks such as traffic scheduling are enhanced through DT predictions. At a later stage, a full WTSN-DT system will integrate simulations to enable what-if scenarios and emulate node issues. This will allow for the prediction and resolution of failures before they occur.

5.4 Traffic Pattern Modeling and Scheduling

Focused on reducing the described RST problem, yet general for network modeling, the proposed and implemented WTSN DT-based control loop is shown in Figure 5.3. As illustrated, a traffic monitoring agent in the WTSN node captures data, specifically frame arrival times, and sends it to the WTSN-DT controller. The data is then modeled and used to generate a transmission schedule, which is sent back to the WTSN node for implementation. Therefore, this centralized topology was chosen to leverage the additional resources typically available in the controller

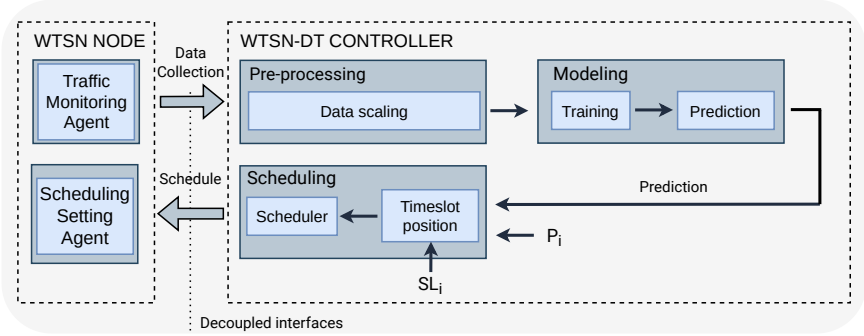


Figure 5.3: DT-based modeling and scheduling management control loop

rather than in the nodes. The following sections describe the capture, pre-processing, modeling, and scheduling blocks in detail.

5.4.1 Data Collection

In a WTSN, STAs and APs typically serve as the entry points for traffic flows. They act as the initial interface between non-aligned frames and the TSN transmission schedule. Therefore, our approach begins by capturing frame arrivals at the WTSN traffic entrance node, specifically within the driver, just before they reach the wireless PHY. This method accounts for both the frame generation delay and the network stack processing delay of the node itself as previously described in Figure 5.1. The traffic arrival timestamp of the k-th frame is represented by Equation 5.1:

$$\begin{aligned} t_{\text{arr}_k} &= k \cdot T_{\text{gen}} + \eta_{\text{gen}_k} + D_{\text{proc}_k} + \eta_{\text{proc}_k}, \\ t_{\text{arr}_k} &= k \cdot T_{\text{gen}} + D_{\text{proc}_k} + \eta_k \end{aligned} \tag{5.1}$$

where T_{gen} represents the traffic flow generation cycle, which is either the same as or a multiple of the GCL cycle, D_{proc} is the processing time that accounts for the network stack processing delay of the k-th frame. Finally, η_k is a random variable that captures both the generation time jitter and the variability in processing delay, $\eta_k = \eta_{\text{gen}_k} + \eta_{\text{proc}_k}$.

To reduce transmission overhead, the frames are captured and sent in batches. As shown in Equation 5.2, the M captured frame arrivals, generated every T_{gen} seconds and sampled every T_{s} seconds, are collected by the traffic monitoring agent.

$$M = \frac{T_s}{T_{\text{gen}}} \quad \text{subject to} \quad T_{\text{gen}} \leq T_s \tag{5.2}$$

Then, as shown in Equation 5.3, the vector $\mathbf{d}$ is created, containing the differences between the M captured frame arrival timestamps. This vector represents the

frame inter-arrival times. The first difference is calculated between the last timestamp of the previous batch and the first timestamp of the current batch. Hence, the vector $\mathbf{d}$ would contain M values. For the first captured batch, the initial difference is substituted with the average values of the subsequent differences.

$$\mathbf{d} = [t_{\text{arr}_1} - t_{\text{arr}_{M_{\text{prev}}}}, t_{\text{arr}_2} - t_{\text{arr}_1}, \ldots, t_{\text{arr}_M} - t_{\text{arr}_{M-1}}] \tag{5.3}$$

To reduce the data size while preserving the time component, the vector $\mathbf{d}$ and the timestamp t_{arr_1} are sent to the WTSN-DT controller.

5.4.2 Data Pre-processing

Once the batch of M inter-arrival times is received at the WTSN-DT controller, it is *scaled* using equation 5.4:

$$\mathbf{d}_{\text{sca}} = \frac{\mathbf{d} - \mathbf{d}_{\text{min}}}{\mathbf{d}_{\text{max}} - \mathbf{d}_{\text{min}}} \tag{5.4}$$

where, $\mathbf{d}_{\text{min}}$ and $\mathbf{d}_{\text{max}}$ are the minimum and maximum values of the vector $\mathbf{d}$, respectively. Scaling can be performed either at the node or at the controller. However, because processing time is critical, scaling is handled by the controller, which typically has higher computational resources.

5.4.3 Data Modeling and Prediction

The effect of η from equation 5.1 on t_{arr} in the context of the GCL cycle (T_{GCL}) can be seen in Figure 5.4. In this example, $T_{\text{gen}} + D_{\text{proc}} = T_{\text{GCL}}$, but due to the variable addition of η, a drift between the two cycles occurs. Therefore, the goal of the methods described in this section is to model this relationship and use it to predict the subsequent frame arrivals.

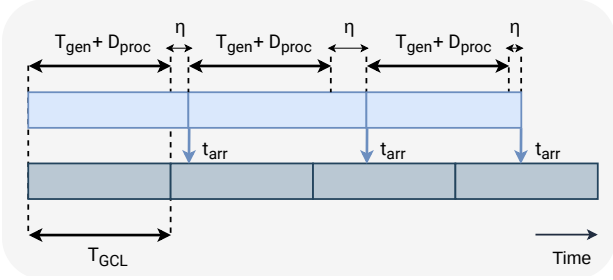


Figure 5.4: Frame generation and GCL cycle drift diagram

For all subsequent modeling methods, the input is the scaled differences vector $\mathbf{d_{sca}}$ and the output is the corresponding scaled predicted differences vector $\hat{\mathbf{d}}$. This

output is then used to compute the predicted frame arrival timestamps, as detailed in the following section. Note that each model includes internal variables that are either fixed or computed from the input.

5.4.3.1 Linear Regression

Starting with the simplest and most explainable approach, and assuming that most components of the captured vector $\mathbf{d}$ are linear, we propose a linear regression modeling approach. Based on the classical linear regression formulation, the matrix form of the model is expressed in Equation 5.5.

$$\hat{\mathbf{d}} = X\beta \tag{5.5}$$

Here, $\hat{\mathbf{d}}$ represents the predicted frame arrivals, while β are the learning coefficients. The design matrix X is:

$$X = \begin{bmatrix} 1 & 2 & 3 & \dots & M \end{bmatrix}^T \tag{5.6}$$

Representing X in this way simplifies the forecasting process. To estimate the coefficients of β, we use the well-known ordinary least squares estimator:

$$\hat{\beta} = (X^T X)^{-1} X^T \mathbf{d}_{\text{sca}} \tag{5.7}$$

Finally, the predicted values for the differences are:

$$\hat{d}_i = \hat{\beta}_0 + \hat{\beta}_1 \cdot i \tag{5.8}$$

Then, using the cumulative sum of the *de-scaled* predicted differences, the i-th predicted arrival timestamp is:

$$\hat{t}_{\text{arr}_i} = t_{\text{arr}_1} + \sum_{j=1}^{i-1} \hat{d}_j \tag{5.9}$$

5.4.3.2 Kalman Filter

The Kalman filter is a dynamic algorithm widely used in navigation and vehicle control tasks [20]. It provides an efficient computational framework for estimating the future state of a process, modeled by the following stochastic difference equation:

$$x_k = Ax_{k-1} + w_k \tag{5.10}$$

This estimation is based on measurements defined as:

$$d_k = Hx_k + v_k \tag{5.11}$$

In our specific case, the M-scaled interarrival times (d_{sca}) are introduced into the filter as d_k, which learns the dynamics and noise characteristics and predicts the next M system's states (x_k). Then, *de-scaling* the system's state, the $\hat{t}_{\text{arr}}$ arrival timestamps are calculated using equation 5.9.

Specific to the Kalman filter, the equations 5.11 and 5.11, w_k and v_k represent the process and measurement noise, respectively, with probability distributions given by:

$$\begin{aligned} p(w) &\sim N(0, Q), \\ p(v) &\sim N(0, R) \end{aligned} \tag{5.12}$$

where Q and R denote the covariances of the process and measurement noise. The matrix A relates the state at the previous time step $(k-1)$ to the current state (k), while H relates the state to the measurement.

The Kalman filter employs two types of equations: time update and measurement update equations. The time update (predictor) equations project the current state ($\hat{x}_{k|k-1}$) and error covariance ($P_{k|k-1}$) forward in time to generate $a\ priori$ estimates for the next step:

$$\begin{aligned} \hat{x}_{k|k-1} &= A\hat{x}_{k-1|k-1}, \\ P_{k|k-1} &= AP_{k-1|k-1}A^T + Q \end{aligned} \tag{5.13}$$

The measurement update (corrector) equations refine the $a\ priori$ estimates by incorporating new measurements to produce $a\ posteriori$ estimates. The Kalman gain (K_k) minimizes the $a\ posteriori$ error covariance, as shown below:

$$\begin{aligned} K_k &= \frac{P_{k|k-1}H^T}{HP_{k|k-1}H^T + R}, \\ P_{k|k} &= (I - K_kH)P_{k|k-1}, \\ \hat{x}_{k|k} &= \hat{x}_{k|k-1} + K_k(d_k - H\hat{x}_{k|k-1}) \end{aligned} \tag{5.14}$$

After each cycle of time and measurement updates, the previous $a\ posteriori$ estimates are used to project $a\ priori$ estimates for the next step.

In this specific case, the M-scaled interarrival times (d_{sca}) are introduced into the filter as d_k, which learns the dynamics and noise characteristics and predicts the next M system's states (x_k). Then, *de-scaling* the system's state, the $\hat{t}_{\text{arr}}$ arrival timestamps are calculated using equation 5.9.

5.4.3.3 Long Short-Term Memory (LSTM)

Time series data benefit from this ML approach. It learns patterns in sequential data over extended periods through gating mechanisms. Although simpler models such as GRU exist, LSTM remains the optimal starting point. Its architecture manages long-term dependencies in a virtually infinite dataset [21]. In our case, it captures

the relationships between frame arrivals with greater accuracy and improves as it receives additional data batches. The model processes the input sequence to predict differences between consecutive arrival timestamps in the next batch. The LSTM model is defined as follows:

$$\hat{\mathbf{d}} = [\hat{d}_1, \hat{d}_2, \ldots, \hat{d}_i] = \text{LSTM}(\mathbf{d}_{\text{sca}}, \theta) \tag{5.15}$$

where, $\mathbf{d}_{\text{sca}}$ is the scaled input sequence, θ represents the learnable parameters, and $\hat{\mathbf{d}}$ refers to the predicted frame arrivals. Then, given the predicted frame inter arrivals, the M predicted arrival timestamps are *de-scaled* and computed as shown in equation 5.9.

An important advantage of the three modeling described cases is that, even with inconsistent data collection, predictions can still be made for the future, as the pattern is captured by the model. While such predictions may be less accurate, they offer the benefit of decoupled interfaces, removing the need for constant data captures to make decisions on the traffic schedule.

5.4.4 Traffic Scheduling

Predicting the arrival of frames allows the TSN scheduler to select the optimal position for the WTSN traffic entrance node time slot, reducing RST and, consequently, link latency and jitter. This section describes the relationship between the forecasted arrivals and the TSN schedule, which is divided into two parts: time slot position generation and schedule generation.

5.4.4.1 Time Slot Position

Once the N predicted frame arrival timestamps (P_i) are received in the Scheduling block, the offset of these timestamps relative to the GCL cycle can be calculated using the modulo operation $\text{mod}(\hat{\mathbf{t}}_{\text{arr}}, T_{\text{GCL}})$. The time slot position is then determined using different percentile values of the sorted modulo results.

These percentile values correspond to the required traffic flow service levels (SL). Specifically, 90%, 70%, 50%, and 40% are assigned to SL 0, 1, 2, and 3, respectively. Therefore, a lower SL provides higher confidence in capturing frame arrivals but may not yield the lowest link latency and jitter. Conversely, a higher SL offers lower confidence but results in lower latency and jitter.

Furthermore, the relative frame arrival timestamps can be used to generate a histogram. The histogram standard deviation value, along with the transmitter Modulation Coding Scheme (MCS) value, could help define the time slot size. However, this is not considered in this work; therefore, a fixed time slot size is used.

5.4.4.2 Schedule Generation

As discussed in Section 5.1, the goal of a WTSN scheduler is to determine optimal time slot positions by considering traffic flow requirements, available resources, wireless channel conditions, and factors such as frame generation. However, since reducing RST is the focus of this work, the proposed traffic scheduler optimizes only the time slot positions to minimize link latency and jitter within a single WTSN Basic Service Set (BSS).

After the system calculates the traffic flow time slot ($timeslot\_start$) as described in the previous section, the scheduler generates the transmission schedule. The initial step is to identify the time slot index within the cycle. This process is detailed in Algorithm 5. The algorithm finds and returns the index closest to $timeslot\_start$ (w_m).

Algorithm 5 Timeslot Index Finder

Input: $timeslot\_start$, gcl_cycle_index, $timeslot\_size\_index$
Output: w_m

$gen\_cycle \leftarrow 512 \times 2^{gcl\_cycle\_index}$
$ts\_size \leftarrow 128 \times 2^{timeslot\_size\_index}$
$n\_timeslots \leftarrow \lfloor gen\_cycle / ts\_size \rfloor$
Initialize $start\_positions$ as an empty list
for $i \leftarrow 0$ **to** $n\_timeslots - 1$ **do**
    $start\_positions[i] \leftarrow i \times ts\_size$
end for
for $i \leftarrow 0$ **to** $n\_timeslots - 1$ **do**
    if $start\_positions[i] \geq timeslot\_start$ **then**
        $w\_m \leftarrow i$
        break
    end if
end for
return w_m

Because the system frequently updates w_m, time slot overlaps may occur. When two time slots index overlap, the scheduler compares their priorities to decide which one remains in place. It then shifts the other to the next available slot on the right to reduce latency and jitter. If the overlapping time slots have identical priority, the scheduler adopts a first come, first served rule. Finally, the scheduler sends the schedule to the Scheduling Setting Agent at the WTSN node, which then sets it.

Due to the dynamic nature of the time slots and other influencing factors, scheduling time becomes critical. However, modeling allows for the prediction of time slot positions further into the future, introducing the concept of *predictive scheduling*. This approach can be seamlessly integrated into both current and future

exact and approximate scheduling methods, such as integer linear programming, satisfiability modulo theories, or artificial intelligence-based techniques [3].

5.5 Results

The obtained results focus on evaluating the three proposed traffic pattern modeling algorithms. In the first stage, a real-world capture of frame arrivals is used to test various algorithm configurations within a simulation environment. In the second stage, the best-performing algorithm configurations are implemented and tested in a real WTSN.

Next, in subsection 5.5.1, we describe the setup used for both data collection in the simulation stage and the real implementation in the second stage. Subsection 5.5.2 presents the simulated results, while subsection 5.5.3 focuses on the implementation results.

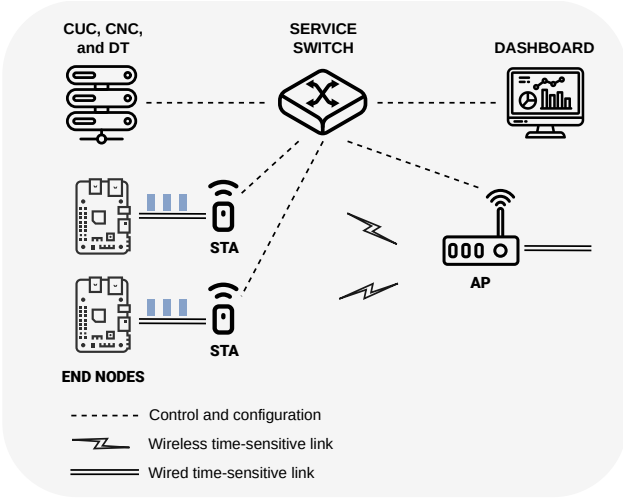


Figure 5.5: Hardware implementation topology for WTSN digital twin-based scheduling

5.5.1 Setup Description

The proposed WTSN Digital Twin-based scheduling method has been implemented and tested using our in-house WTSN Evaluation Kit (EK), based on the openwifi IEEE 802.11/Wi-Fi baseband chip and FPGA design [12]. As shown in Figure 5.5, two Xilinx Zedboards equipped with AD-FMCOMMS4-EBZ transceivers act as STAs for the wireless clients. The Access Point (AP) in our setup uses a Xilinx ZC706 board with an AD-FMCOMMS4-EBZ transceiver. Both the AP and STAs

are connected via Ethernet to a service switch, which provides separate control and configuration interfaces.

Additionally, two Intel Next Unit of Computing (NUC) nodes running Ubuntu are connected to the service switch. One node manages the CNC, CUC, and DT functions, while the other serves as a monitoring dashboard. The EK also integrates key TSN features, including a PTP-based time synchronization mechanism and a TSN gating system within the wireless nodes [22].

The network control and configuration links are established using a ZeroMQ PUB/SUB model with agents in all WTSN nodes userspace, serving three purposes: i) sending time-sensitive traffic arrival data to the WTSN controller, ii) setting the schedule determined by the WTSN scheduler, and iii) measuring the wireless link latency of the traffic flows. To reduce overhead, the schedule information is bitmap-encoded using the notation described in [23]. Furthermore, retransmissions are disabled in all nodes in order to avoid their impact on measurements.

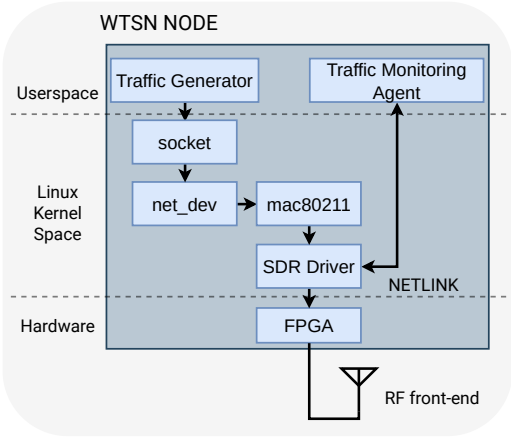


Figure 5.6: Data capture agent diagram for a WTSN node

Figure 5.6 presents a schematic of the traffic arrival data capture at the wireless interface of a WTSN node. The testing traffic flow, is generated by an unsynchronized traffic generator using the $time.perf\_counter()$ python function, enters the node's kernel space. Before reaching the FPGA, which controls the TSN gating system, the frame arrival timestamps are captured and sent via NETLINK to the Traffic Monitoring Agent in user space. Then, as described in Section 5.4.1, the data is transmitted in batches to the WTSN-DT Controller using the network control link.

If the traffic generator is located on an external node and frames enter the WTSN node via an Ethernet instance, for example, these frames would also be captured and timestamped at the SDR driver. This ensures the capture of both WTSN node

and generator delays.

5.5.2 Testing Results and Data Analysis

Using the setup described in Section 5.5.1, 100000 frame arrival data points were captured from a WTSN node. These data points were then used to test the Linear Regression, Kalman Filter, and LSTM approaches in an online fashion to identify the best methods for the next stage of implementation. For the LSTM, configurations with 2 and 3 layers of 32 neurons with 50 and 100 epochs were tested.

First, the Mean Squared Error (MSE) and Mean Absolute Error (MAE) were calculated, and the cumulative distribution function (CDF) of them is shown in Figures 5.7 and 5.8. The test compared and saved the output of N predicted elements from each model with the N actual values. In this study, we employ the CDF for most metrics, as it effectively displays achieved percentiles, a crucial factor for determinism.

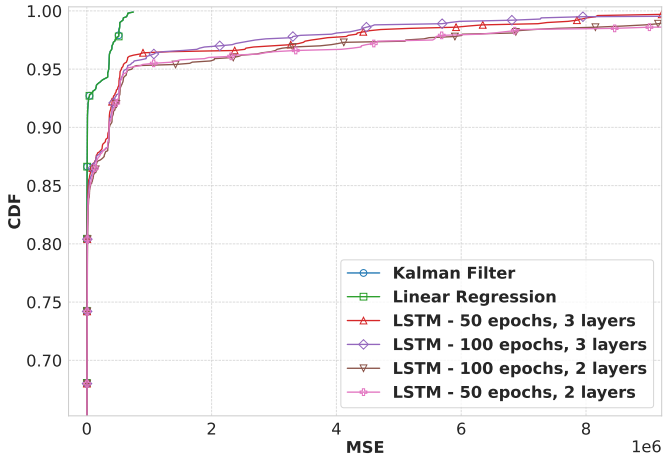


Figure 5.7: CDF of mean squared error (MSE) results with different controllers

The results in Figures 5.7 and 5.8 show that both the Kalman Filter and Linear Regression outperform the LSTM across different configurations. This suggests that the drift component between arrivals is primarily linear, based on these performance indicators.

Next, to measure the fitting quality between the predicted batch values and the real ones, the Jensen-Shannon divergence is used [24]. This method quantifies the difference between two distributions, in this case, the predicted and real distributions. It outputs a value between 0 and 1, with 0 indicating a perfect fit. As shown in Figure 5.9, the CDFs of the LSTM methods outperform the Linear and Kalman approaches. This was expected, as ML methods learn from the distribution shape.

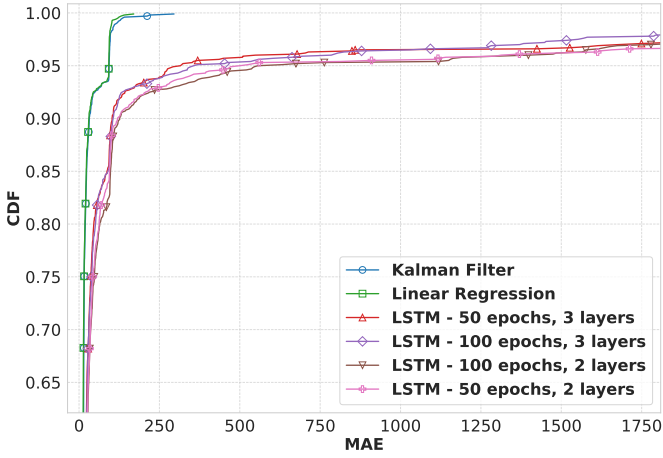


Figure 5.8: CDF of mean absolute error (MAE) results with different controllers

Specifically, for LSTM, the more epochs and layers used, the better the distribution shape is learned.

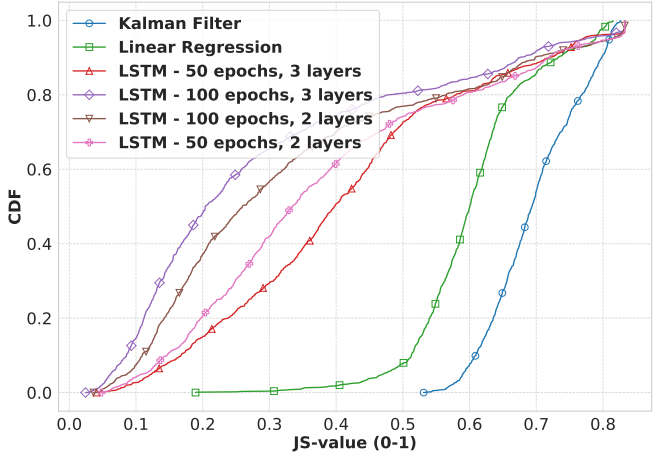


Figure 5.9: CDF of Jensen-Shannon divergence results with different controllers

Effective learning often comes at the cost of increased processing time. To evaluate this, the fitting and forecasting times for each iteration were measured and compared, as shown in Figures 5.10 and 5.11. As expected and shown in the resulting CDFs, the Linear and Kalman approaches perform similarly in terms of fitting time, while the Kalman approach is the worst for forecasting. In the case of LSTM, as anticipated, the more epochs and layers used, the higher the fitting time,

while for forecasting, their performance is similar.

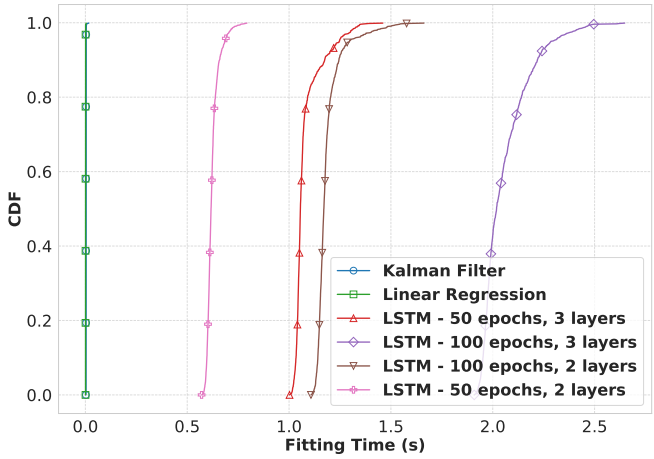


Figure 5.10: CDF of fitting time results with different controllers

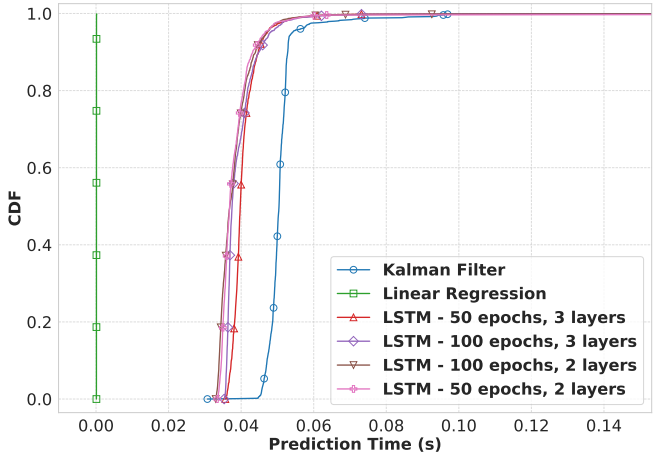


Figure 5.11: CDF of forecasting time results with different controllers

5.5.3 Implementation Performance Evaluation

Based on the results from the previous section, three algorithms were implemented in the real setup: Linear Regression, Kalman Filter, and LSTM with 2 layers and 50 epochs, which provides a good balance across the evaluated indicators. Using the topology described in Figure 5.5, the development and testing were carried

out in two stages: first, one traffic flow from a single STA to the AP was tested, followed by two simultaneous uplink traffic flows, each from a different STA to the AP. The TSN and sampling parameters used are summarized in Table 5.1. Figure 5.12 shows the real-time captured dashboard result of link latency before and after enabling the network controller, using the Linear Regression approach. As seen, once the controller is enabled, both latency and jitter are reduced significantly.

Table 5.1: TSN and sampling measurement parameters used for RST controllers

Parameter	Value
$T_{\text{gen}}, T_{\text{GCL}}$	8,192 ms
Time slot size	256 ms
Total samples	10 000
T_s	1 s
M	125
SL	0

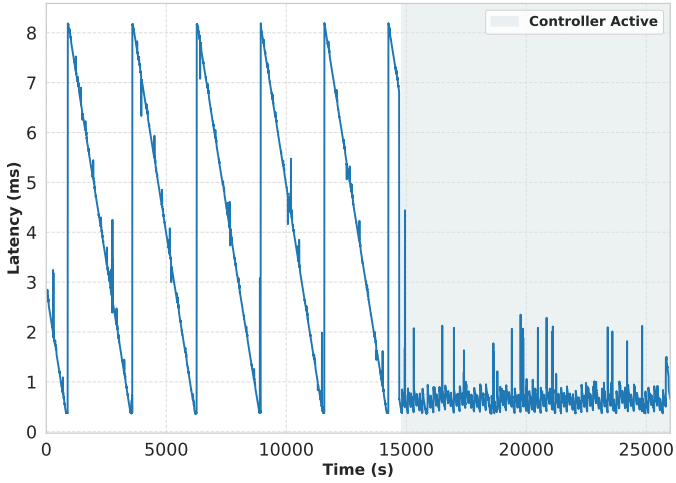


Figure 5.12: Real-time capture of WTSN link latency control using a linear regression controller

An important factor to consider when implementing dynamic TSN systems is the delay introduced during data collection, fitting, prediction, schedule calculation, and setting of the schedule in the TSN node. These delays must be accounted for in the prediction process within the WTSN-DT controller before being sent to the TSN node. Additionally, as mentioned previously, the advantage of having a model is that predictions can be made for the future, but this must also be incorporated into the prediction process. Therefore, leveraging the fact that all TSN nodes are

tightly synchronized, a new variable, t_{delay}, was added to the forecasting equation 5.9 to account for this compensation.

The t_{delay} value used in each iteration is approximated based on the previous one. However, since the WTSN-DT runs on a non-real-time operating system/kernel, t_{delay} may vary. Therefore, especially in the case with two STAs using LSTM, each traffic capture, fitting, and prediction is executed on an independent processor core to reduce this variability.

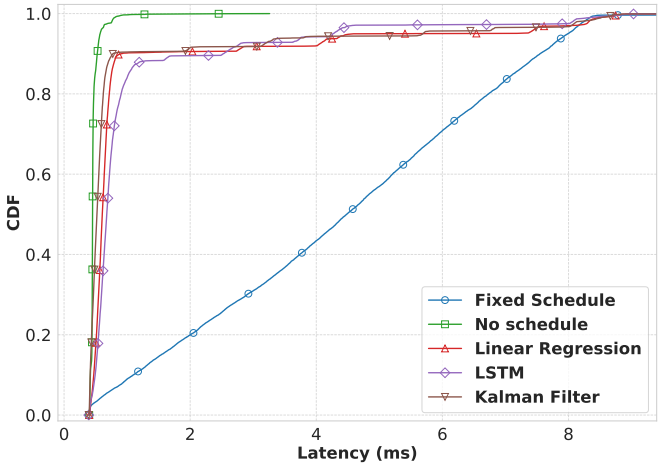


Figure 5.13: CDF of WTSN link latency measurements for different controllers

As shown in Figure 5.1, RST primarily occurs because frame generation is not aligned with the TSN schedule. Figure 5.13 demonstrates this phenomenon as measured in real life. As seen, in the *fixed schedule* case, the latency CDF, although bounded to 8,192 ms, introduces jitter. In contrast, in the *no schedule* case, the STA is allowed to transmit as soon as the frame is received (using CSMA/CA), resulting in minimal latency and jitter. However, this approach sacrifices the TSN benefits of traffic protection and bounded latency. In contrast, the three proposed methods achieve latency and jitter levels similar to the unscheduled case while preserving the advantages of TSN. Comparing them at the 90th percentile, a 96,7% latency reduction was achieved.

An important factor, particularly for t_{delay}, is presented in Figure 5.14, which shows measurements of the total fitting, prediction, and scheduling operations. As observed, although there are notable differences between the methods, consistent with the trends identified in the previous section, the values remain relatively constant. This consistency validates the use of the previous t_{delay} as a reliable approximation for delay compensation.

Finally, to evaluate the adaptability of the system, the link latency CDF results

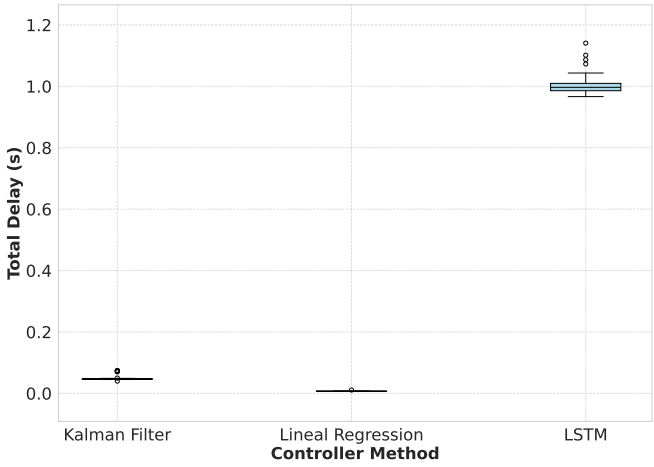


Figure 5.14: Measurement results of total delay (fitting, prediction, and scheduling) for different controllers

for two simultaneous traffic flows from two STAs are shown in Figure 5.15. Similar to the single STA case, all approaches successfully reduce latency and jitter. However, in this scenario, the Linear Regression approach performs the worst, while the Kalman Filter delivers the best overall performance.

The performance difference between methods with one and two STAs is related to the implementation. Although the capture, fitting, and prediction processes, which introduce the highest delay, are performed on separate processor cores to achieve processing delay determinism, the scheduling is handled by a single instance. This can lead to unexpected delays. While deterministic processing is beyond the scope of this study, it plays an important role and requires further investigation.

5.6 Conclusion

The wireless environment presents significant challenges to achieving the determinism required by time-sensitive networking (TSN). Tasks such as traffic scheduling need enablers that can address the unique characteristics of wireless networks in order to transition from wired to wireless systems. A concept, known as the digital twin (DT), has the potential to facilitate this transition. The ability to create virtual representations of network devices, links, and other components opens the door for concepts like predictive scheduling. However, before this can be fully realized, the interaction between DTs and TSN management must be clearly defined. To take the first steps toward these definitions, the current work proposes, implements, and

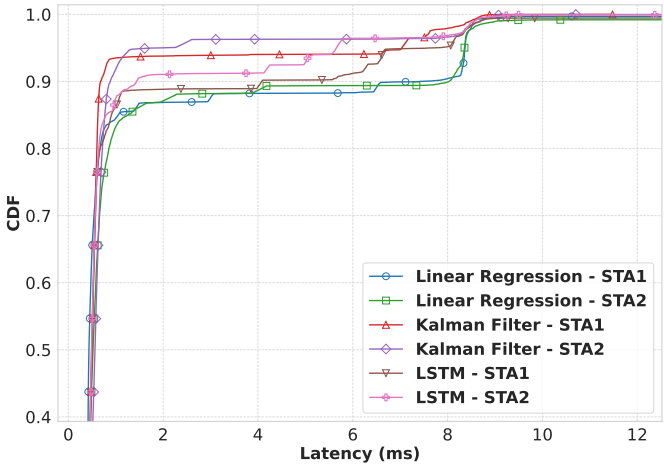


Figure 5.15: CDF of WTSN link latency measurement results for different controllers using two stations

tests a wireless time-sensitive management framework that includes a digital twin focused on reducing wireless link latency and jitter caused by residual service time.

Traffic arrival frames are captured and modeled centrally using linear regression, Kalman filtering, and a Long Short-Term Memory (LSTM) network to predict frame arrivals and optimize the allocation of transmission time slots, reducing link latency and jitter.

References

[1] C. Xue, T. Zhang, Y. Zhou, M. Nixon, A. Loveless, and S. Han. *Real-Time Scheduling for 802.1Qbv Time-Sensitive Networking (TSN): A Systematic Review and Experimental Study*. IEEE Real Time Technology and Applications Symposium, pages 108–121, 2023. doi:10.1109/RTAS61025.2024.00017.

[2] H. Ma, H. Li, L. Chen, D. Zhao, X. Zhao, K. Xia, and G. Shou. *Time Error Bound of Deterministic Transmission in Cross-Domain Time-Sensitive Networking*. 2024 Asia Communications and Photonics Conference (ACP) and International Conference on Information Photonics and Optical Communications (IPOC), pages 1–4, 11 2024. Available from: https://ieeexplore.ieee.org/document/10809464/, doi:10.1109/ACP/IPOC63121.2024.10809464.

[3] T. Stuber, L. Osswald, S. Lindner, and M. Menth. *A Survey of Scheduling Algorithms for the Time-Aware Shaper in Time-Sensitive Networking (TSN)*. IEEE Access, 11:61192–61233, 2023. doi:10.1109/ACCESS.2023.3286370.

[4] P. Avila-Campos, J. Haxhibeqiri, M. Girmay, I. Moerman, and J. Hoebeke. *Residual Service Time Optimization for legacy Wireless-TSN end nodes*. In WiMob 2023 - The 19th International Conference on Wireless and Mobile Computing, Networking and Communications., 2023. Available from: http://www.ieee802.org/1/tsn.

[5] T. Zhang, G. Wang, C. Xue, J. Wang, M. Nixon, and S. Han. *Time-Sensitive Networking (TSN) for Industrial Automation: Current Advances and Future Directions*. ACM Computing Surveys, 2 2024. doi:10.1145/3695248.

[6] *3GPP TS 23.501 version 16.6.0 Release 16 - System architecture for the 5G System (5GS)*.

[7] T. Fiori, F. G. Lavacca, F. Valente, and V. Eramo. *Proposal and Investigation of a Lite Time Sensitive Networking Solution for the Support of Real Time Services in Space Launcher Networks*. IEEE Access, 12:10664–10680, 2024. doi:10.1109/ACCESS.2024.3353466.

[8] M. M. Roselló and M. Lorenzo. *Smart De-jittering using Network Digital Twin*. 2024 IEEE International Mediterranean Conference on Communications and Networking, MeditCom 2024, pages 97–101, 2024. doi:10.1109/MEDITCOM61057.2024.10621358.

[9] H. Shin, S. Oh, A. Isah, I. Aliyu, J. Park, and J. Kim. *Network Traffic Prediction Model in a Data-Driven Digital Twin Network Architecture*. Electronics 2023, Vol. 12, Page 3957, 12:3957, 9 2023. Available from:

https://www.mdpi.com/2079-9292/12/18/3957/htmhttps://www.mdpi.com/2079-9292/12/18/3957, doi:10.3390/ELECTRONICS12183957.

[10] I.-T. T. S. S. O. ITU. *ITU-T Rec. Y.3090 (02/2022) Digital twin network – Requirements and architecture*, 2022. Available from: http://handle.itu.int/11.1002/1000/11.

[11] I. R. T. Force. *Digital Twin Network: Concepts and Reference Architecture*, 2023. Available from: https://datatracker.ietf.org/drafts/current/.

[12] X. Jiao, W. Liu, M. Mehari, M. Aslam, and I. Moerman. *Openwifi: A free and open-source IEEE802.11 SDR implementation on SoC*. IEEE Vehicular Technology Conference, 2020-May, 5 2020. doi:10.1109/VTC2020-SPRING48590.2020.9128614.

[13] A. Freedman. *Network Digital Twins: The Next Revolution in Network Management — Keysight Blogs*, 2024. Available from: https://www.keysight.com/blogs/en/tech/educ/2024/network-digital-twin.

[14] S. Scanzio, M. Rosani, G. Formis, D. Cavalcanti, V. Frascolla, G. Marchetto, and G. Cena. *Multi-Link Operation and Wireless Digital Twin to Support Enhanced Roaming in Next-Gen Wi-Fi*. IEEE International Workshop on Factory Communication Systems - Proceedings, WFCS, 2024. doi:10.1109/WFCS60972.2024.10540931.

[15] S. Robitzsch, A. de la Oliva, C. F. Chiasserini, C. E. Casetti, F. Agraz, G. Landi, L. M. Contreras, P. Szilagyi, P. G. Giardina, R. Rosales, S. Spadaro, , and V. Frascolla. *Standardisation Assessment of a Digital Twin-Based Multi-Domain Deterministic Communications System*. IEEE GLOBECOM 2024 Workshop - 6GArch, 2024. Available from: https://iris.polito.it/handle/11583/2992674.

[16] S. B. H. Said, M. T. Thi, M. Kellil, and A. Olivereau. *On the Management of TSN Networks in 6G: A Network Digital Twin Approach*. IEEE International Conference on Emerging Technologies and Factory Automation, ETFA, 2023-September, 2023. doi:10.1109/ETFA54631.2023.10275568.

[17] S. Sudhakaran, J. Perez-Ramirez, C. Hall, A. Regev, M. Smith, A. Stern, N. Georghiou, E. Oren, G. Kronauer, G. Venkatesan, and J. Huang. *Wireless TSN: Tests and Use Case Demonstrations with Wi-Fi*, 1 2024.

[18] D. B. Mateu, J. Proenza, A. V. Papadopoulos, T. Nolte, and M. Ashjaei. *TALESS: TSN With Legacy End-Stations Synchronization*. IEEE Open Journal of the Industrial Electronics Society, 5:807–822, 2024. doi:10.1109/OJIES.2024.3436590.

[19] *IEEE P802.1Qcc - IEEE Draft Standard for Local and metropolitan area networks–Media Access Control (MAC) Bridges and Virtual Bridged Local Area Networks Amendment: Stream Reservation Protocol (SRP) Enhancements and Performance Improvements*. pages 1–236, 2018.

[20] G. Welch and G. Bishop. *An Introduction to the Kalman Filter*. Department of Computer Science University of North Carolina at Chapel Hil, 7 2016. Available from: http://www.cs.unc.edu/~gb.

[21] P. Srivatsavaya. *LSTM vs GRU. LSTM (Long Short-Term Memory) and GRU… — by Prudhviraju Srivatsavaya — Medium*, 7 2023. Available from: https://medium.com/40prudhviraju.srivatsavaya/lstm-vs-gru-c1209b8ecb5a.

[22] M. Aslam, W. Liu, X. Jiao, J. Haxhibeqiri, G. Miranda, J. Hoebeke, J. Marquez-Barja, and I. Moerman. *Hardware Efficient Clock Synchronization Across Wi-Fi and Ethernet-Based Network Using PTP*. IEEE Transactions on Industrial Informatics, 18:3808–3819, 6 2022. doi:10.1109/TII.2021.3120005.

[23] P. Avila-Campos, J. Haxhibeqiri, I. Moerman, X. Jiao, and J. Hoebeke. *Impactless association methods for wi-fi based time-sensitive networks*. Wireless Networks, 30:2457–2475, 5 2024. Available from: https://link.springer.com/article/10.1007/s11276-024-03681-w, doi:10.1007/S11276-024-03681-W/FIGURES/14.

[24] P. Maurya. *Getting to Know Jensen-Shannon Divergence (JSD) — by Priyanshu maurya — Medium*, 8 2024. Available from: https://medium.com/@_prinsh_u/getting-to-know-jensen-shannon-divergence-jsd-5c96720a4434.

6

Conclusion and future work

"It is the possibility of having a dream come true that makes life interesting."

–Paulo Coelho

Throughout history, human survival has hinged on our ability to adapt in the face of uncertainty. In much the same way, wireless systems must be engineered to respond dynamically to ever-changing environments, marked by interference, fluctuations, and unexpected disruptions, to maintain seamless connectivity in a world that never stands still. Since their inception in the early 20th century, one of the persistent challenges has been enabling wireless networks to adapt to dynamic conditions while maintaining reliability. The introduction of the concept of determinism in wireless networks not only adds a new layer of complexity to this challenge but also opens a new world of possibilities. This dissertation explores a range of innovative techniques that empower wireless time-sensitive networks to not only adapt intelligently and ensure reliability but also build trust among industry players. By fostering adaptation driven by learning, these networks can effectively meet the demands of limited resources, environmental variability, and stringent application requirements. The findings presented in this work demonstrate how adaptability and learning can transform wireless time-sensitive networks into resilient, high-performing systems. The next section provides a summary of the key contributions and insights that advance the design and management of these networks.

6.1 Conclusions

As with any solid construction, a strong foundation is essential. In this work, the foundations are described in Chapter 2. This chapter examines the challenges addressed in later sections. It begins with traffic scheduling from different perspectives, focusing on the behavior of shared and dedicated transmission time slots. Empirical tests revealed that dedicated time slots provide higher throughput, especially when the CSMA system is disabled.

Building on these insights, a broader approach to traffic scheduling led to the development of an SMT-based wired and wireless scheduler. A TSN is represented as a graph with nodes and links, which can be wired or wireless. The model incorporates traffic flow priorities for dedicated and shared time slots, adjustable transmission rates, and various tests to assess the impact of traffic crossing wired and wireless sections of the network.

Additionally, a reconfiguration test exposed the main challenge of optimal schedulers: their difficulty in adapting to the dynamic nature of wireless channels. The results emphasize the importance of proactive strategies, such as pre-calculated schedules, oversized time slots with resource optimization techniques, and GCL cycle segmentation to protect time-sensitive traffic.

Finally, despite the challenges of wireless TSN, its advantages are already evident in real-world demonstrations. A distributed PID machine control system was used to illustrate a scenario where TSN protects machine control traffic. In contrast, a conventional Wi-Fi shared medium risks delaying frames.

Beyond identifying the challenges of wireless TSN, Chapter 2 also highlighted the need to improve fundamental wireless processes, such as association. Therefore, Chapter 3 focused on developing an efficient, TSN-compatible association procedure. The goal was to integrate TSN features like scheduling and time synchronization into wireless clients before they associate.

The proposed solution introduced new components that leveraged beacons as the primary enabler. Beacons were used for both pre-scheduling and pre-synchronization. In pre-scheduling, beacon stuffing embedded association time slot schedules into beacons, with the information encoded to optimize frame size. For pre-synchronization, various methods were proposed to allow prospective clients to achieve sufficient time synchronization with the network before initiating the association process.

By ensuring synchronization and scheduling before association, the mechanism prevents disruptions to time-sensitive traffic from already scheduled nodes. Extensive testing accompanied this development, evaluating association delays, the number of beacons required for pre-synchronization, and, ultimately, its performance in a real-world implementation.

Chapter 4 continues to address wireless process improvements in TSN, with a

particular focus on mobility. The ability of nodes to move while still supporting time-sensitive traffic is one of the most critical features of wireless TSN. This challenge is closely linked to the handover process, where a wireless client transitions between access points. The strategy was guided by two main objectives: (1) ensuring a fast handover and (2) executing it at the optimal time.

A key decision was shifting the handover control from the client to the CNC. This change allowed for better resource allocation in both the current and target access points. Additionally, improving the communication of the handover command from the controller to the client significantly reduced handover delay. A structured procedure was introduced to ensure a seamless transition from the moment the handover is triggered until it is completed.

The next challenge was determining the best time to initiate the handover. In standard Wi-Fi, this decision is based on active or passive channel scanning, where clients monitor signal strength from known access points. However, this approach is not compatible with TSN, as it requires long dwell periods during which the client cannot receive traffic from its current access point.

To address this limitation, four methods were proposed: localization, coordinated beaconing and scanning, coordinated probing, and overhearing. Each method provided a way to gather link information, but the localization approach was implemented. This solution used a UWB-based localization system to model the channel in terms of position and received power. The model was developed in both one and two dimensions, with continuous online updates to account for environmental changes. By combining these channel models, the system could determine the best candidate access point for handover.

As in previous chapters, this solution was evaluated through extensive testing. Performance was measured in terms of handover latency, jitter, throughput, and frame loss. Additionally, a real-world demonstration validated the implementation in a practical scenario.

Finally, complementing the previous chapters, Chapter 5 explores the integration of digital twins with TSN management. Much of the current research focuses on traffic scheduling, and while valuable ideas emerge, most solutions are not applicable to real TSN deployments. This is often due to oversimplified traffic characteristics, a lack of consideration for wireless channel variations, or network models based on perfectly time-synchronized nodes.

With our experience in wireless TSN, a key conclusion is that a universal scheduler for all applications is still not feasible with current technology. Instead, scheduling efforts must be application-specific and consider realistic variables such as variable traffic generation, processing delays, time synchronization, and transmission constraints.

Our proposed approach is generalizable but demonstrates its potential by modeling real unsynchronized traffic arrivals to reduce link latency and jitter. Three

different modeling methods were explored: a simple and explainable linear regression, a more complex system model using a Kalman Filter, and an advanced machine learning approach, LSTM, which is particularly suited for time-series data.

Performance evaluation compared these methods in terms of accuracy, fitting, and forecasting time, and link latency reduction in a real-world scenario with two clients acting as traffic generators and an access point.

The findings of this research have significant implications for real-world Wireless TSN applications, particularly in domains that require time-sensitive and highly reliable communication, such as industrial automation, autonomous systems, and mission-critical operations. By addressing key challenges in extending TSN to wireless, this work takes the first steps toward enabling industries to transition from wired Ethernet-based TSN to more flexible, scalable, and cost-effective Wi-Fi-based deployments without compromising deterministic guarantees.

Wireless TSN is at a pivotal stage. Its advantages, especially for industrial applications, are evident. However, key enablers like hardware and real implementations remain underdeveloped, which significantly hinders both research and commercial progress. The challenge goes beyond creating a broad WTSN solution; it also involves addressing the industry's lack of trust in wireless technologies. Encouragingly, recent developments suggest momentum is building. Several major European projects and vendors have begun investing in WTSN, accelerating its advancement.

At this early stage, with limited resources, the complexity of achieving determinism in a stochastic environment can be discouraging. To stay on course, it is essential to define clear, focused, and limited goals. Closer coordination between hardware and software development is key to enabling practical deployments. This effort should prioritize easing the transition from non-TSN systems. Just as crucial is identifying which network segments genuinely require strict determinism and which can rely on available capacity instead.

6.2 Future Work

This research advances wireless TSN and highlights numerous challenges that remain. In Chapter 2, limitations emerge from various perspectives, especially concerning time resources. A key issue is wireless transmission delay. Nevertheless, the future appears promising with technologies such as OFDMA, beamforming, MLO, C-SR, and higher data rates. These innovations can optimize the use of time resources. Specifically, OFDMA can add a frequency dimension to traffic scheduling. However, most of these technologies are opportunity-based. Therefore, to achieve deterministic network behavior, a combination of them is necessary to ensure bounded determinism even in the worst-case scenarios.

Research, both in my work and more broadly, focuses on time variables such as

latency and jitter to assess time sensitivity. However, this emphasis often overlooks a critical aspect of TSN: high reliability. Variables like packet loss rate, retransmission rate, and interference metrics deserve greater attention. Furthermore, technologies such as MLO promise significant improvements in reliability. Although MLO scheduling adds complexity, it provides valuable opportunities to enhance network resilience, making it worth pursuing in the future.

Research, both in this work and in the broader field, tends to prioritize time-related metrics like latency and jitter when evaluating time sensitivity. This focus, however, often neglects a key feature of TSN: high reliability. Metrics such as packet loss rate, retransmission rate, and interference should receive more attention. At the same time, technologies like MLO offer strong potential to improve reliability. While MLO scheduling introduces added complexity, it also brings important opportunities to strengthen network resilience. For this reason, it merits further exploration.

The mobility research demonstrated that real-time, adaptive radio environment modeling is feasible for decisions such as handover. Openwifi continues to provide opportunities, particularly by incorporating variables like channel state information to enhance modeling accuracy. Moreover, these models could integrate with management functions such as load balancing and wireless transmission schedules, which might be related to REM models.

Scheduling remains an open challenge, as previously stated, requiring both application-specific and proactive approaches. Rescheduling delay stands as the most significant obstacle in wireless TSN. To address this challenge, a flexible and distributed management system is essential, allowing local CNCs to oversee wireless segments of the network. These controllers must have the autonomy to make rapid and proactive decisions in response to network changes while maintaining deterministic guarantees.

The introduction of an element of a digital twin for TSN management addressed the lack of realism in scheduling. However, it also introduced new challenges related to processing delays and network architecture. This highlights the need for deterministic processing in communication decisions. This principle applies not only to network management but also to applications. TSN requires viewing networking not merely as data transmission between two points but as an interconnected system that starts with frame generation and includes processing delays through to delivery. This poses a challenge in a world that demands flexible virtualized resources. Future research could explore integrating real-time operating systems and developing a scheduler that coordinates both transmission and processing time slots.

Finally, TSN achieves determinism through stronger and stricter control. But how scalable is this? This prompts consideration of the actual network links that require TSN. TSN reaches its full potential near a link's capacity, where guarantees can be provided even when resources are limited. Making it a perfect

candidate to reduce network oversizing and even energy consumption reduction. Hence, compared to wired links, wireless links inherently have lower capacity and reliability, making them the primary candidates for TSN. Realizing this could significantly reduce the network control needed to ensure determinism and focus our research efforts.