UNICO 5G I+D

TIMING

Towards a smart and efficient telecom infrastructure meeting current and future industry needs (TIMING-SP1)
(Ref. TSI-063000-2021-145)

Deliverable D1.1

Preliminary design of the TIMING control architecture

Editor J. Vidal (UPC)

Contributors (TID), (Ikerlan), (Orolia), (E-lighthouse), (UPC), (ABB)

Version 2.1

Date March, 2023

Distribution PUBLIC (PU)

DISCLAIMER

This document contains information, which is proprietary to the TIMING (Towards a smart and efficient telecom infrastructure meeting current and future industry needs) consortium members.

Neither this document nor the information contained herein shall be used, copied, duplicated, reproduced, modified, or communicated by any means to any third party, in whole or in parts, except with prior written consent of the TIMING consortium members. In such case, an acknowledgement of the authors of the document and all applicable portions of the copyright notice must be clearly referenced. In the event of infringement, the consortium members reserve the right to take any legal action it deems appropriate.

This document reflects only the authors' view. Neither the TIMING consortium members as a whole, nor a certain TIMING consortium member warrant that the information contained in this document is suitable for use, nor that the use of the information is accurate or free from risk, and accepts no liability for loss or damage suffered by any person using this information.

The information in this document is provided as is and no guarantee or warranty is given that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability.

REVISION HISTORY

Revision	*Date*	*Responsible*	*Comment*
2.1	March, 02, 2023	Josep Vidal	Final reviewed version
2.0	February, 28, 2023	Josep Vidal	Complete version
1.0	October, 7, 2022	Josep Vidal	UPC contributions completed
ToC	June, 14, 2022	Josep Vidal, Luis Velasco	Table of contents

LIST OF AUTHORS

Partner ACRONYM	*Partner FULL NAME*	*Name & Surname*
UPC	*Universitat Politècnica de Catalunya*	*Josep Vidal, Javier Villares, Marga Cabrera, Olga Muñoz, Fernando Agraz, Salvatore Spadaro, Jaume Comellas, Davide Careglio, Marc Ruiz, Luis Velasco*
TID	*Telefonica I+D*	*Luis Miguel Contreras*
IKL	*Ikerlan*	*Oscar Seijo*
ORO	*Orolia*	*Jorge Sánchez, Trinidad García*
ELI	*E-lighthouse*	*José Manuel Martínez Caro, Enrique Fernández Sánchez, Pablo Pavon*
ABB	*Asea Brown Boveri*	*Laura González*

GLOSSARY

Abreviations/Acronym	*Descriptions*
5GPPP	5G Public Private Partnership
AI	Artificial Intelligence
AP	Access Point
API	Application Programming Interface
ARM	Advanced RISC Machines
BE	Best Effort
CM	Connectivity Manager
CNC	Central Network Configuration
CSI	Channel State Information
CSMA/CA	Carrier Sense Multiple Access/Collision Avoidance
DetNet	Deterministic Networks
DL	Downlink
DMA	Direct Memory Access
DT	Digital Twin
E2E	End-to-End
ENP	E-lighthouse Network Planner
ETSI	European Telecommunication Standards Institute
FPGA	Field-Programmable Gate Array
FRER	Frame Replication and Elimination for Reliability
GCL	Gate Control List
GUI	Graphical User Interface
gRPC	High Performance Remote Procedure Calls
HTTP	Hypertext Transfer Protocol
i4.0	Industry 4.0
IEEE	Institute of Electrical and Electronic Engineers
IETF	Internet Engineering Task Force
IIOT	Industrial Internet of Things
IP	Internet Protocol
KPI	Key Performance Indicators
LAN	Local Area Networks
LSP	Label Switched Paths
MAC	Multiple Access Control
MCS	Modulation and Coding Scheme
MEC	Multi-Access Edge Computing
MM	Monitoring Manager
MPLS	Multiprotocol Label Switching
NBI	Northbound Interface

NFV	Network Function Virtualization
NSC	Network Slice Controller
OFDMA	Orthogonal Frequency Division Multiple Access
OLS	Open Line System
OSS/BSS	Operations Support System and Business Support System
OTN	Open Transport Network
PCS/PMA	Physical Coding Sublayer/Physical Medium Attachment
PHC	PTP Hardware Clock
PTP	Precision Time Protocol
QoS	Quality of Service
RAN	Radio Access Network
RT	Real Time
SBI	Southbound Interface
SDN	Software Defined Networks
SDP	Slice Demarcation Points
SLA	Service Level Agreement
SLO	Service Level Objectives
SNMP	Simple Network Management Protocol
STA	Station in a WiFi network
STF	Short Training Field
TCP	Transmission Control Protocol
TDD	Time Division Duplexing
TDMA	Time Division Multiple Access
TE	Traffic Engineering
TFS	Teraflow SDN
TIF	Telecom Infra Project
TM	Topology Manager
TSN	Time Sensitive Networks
TSU	Time Stamping Units
UL	Uplink
URLLC	Ultra-reliable Low-latency Communications
VLAN	Virtual Local Area Network
VPN	Virtual Private Network
WLAN	Wireless Local Area Network

EXECUTIVE SUMMARY

We are witnessing the transition into the 4th industrial revolution, also termed Industry 4.0 (i4.0) or the industrial Internet-of-Things (IIoT). Its promise is to integrate IoT and cyber-physical systems into the industrial domain and to boost the productivity of industrial verticals thanks to a radical automation of all the phases of production, ranging from design and manufacturing to operation, supply chain, delivery management and service maintenance. Advancements in artificial intelligence (AI) and robotics are creating many opportunities in this space.

Communications are key to enable i4.0, but are subject to the stringent requirements of automated applications in terms of availability (>99,999% of locations and time), reliability (data loss rate <10-6), low latency (<1 ms), low jitter, safety, integrity, scalability, maintainability and positioning accuracy (<1 cm) of all kind of assets. Overall, these concepts have been grouped under the name of *determinism*: the ability of a network to exchange data in a precise a predictable manner with those defined requirements, which contrasts with Ethernet's *best effort* (BE) principle.

Towards this vision, the 5GPPP has recognized i4.0 as a key vertical, identifying five main use case families: i) control of time-critical processes inside the factory, ii) non-time critical in-factory communications, iii) remote control, iv) intra-inter enterprise communications, and v) connected goods. As the requirements are difficult to be reached with current wireless technologies, factories have relied primarily on Ethernet connectivity, with its inherent limited flexibility. Two complementary sets of standards have focused on providing deterministic network behavior:

- The IEEE 802.1 standards, collectively named Time Sensitive Networking (TSN), focusing on making Ethernet deterministic by design in Local Area Networks (LANs). With TSN capabilities, the same network can be shared by time-sensitive and other (best-effort) services. TSN is the pillar to provide ultra-reliable low-latency communications.
- The IETF DetNet Working Group2 is working to extend the scope of determinism to IP Layer 3 routed segments using a Software-Defined Networking (SDN) layer.

On its turn, a wirelessly connected factory enables novel mobile robots, easy reconfiguration of assembly lines or migration of embedded control functions to the virtually infinite computational/cache resources and flexibility of edge clouds. However, delivering wire-equivalent reliable communications with time guarantees remains a significant challenge. In fact, although some efforts exist (e.g., Intel Wireless TSN initiative) and significant progress has been made in extending accurate time synchronization over the air (e.g., 802.1AS profile for IEEE 802.11/Wi-Fi) there is no standard in this area yet. The availability of Wi-Fi TSN will accelerate its adoption in Industry 4.0 scenarios. The support of TSN over Wi-Fi is challenging because of the intrinsic variability of the wireless access and the latency introduced on network interfaces, so some delay budget needs to be reserved for the network interface out of the allowed e2e latency one. As a result, the i4.0 ecosystem has been identified as an opportunity for the wireless community and has become one of the key targets of the next generation mobile networks.

At its technical core, the development of deterministic networks entails a thorough rethinking of communication networks which are usually designed on average quantities e.g., average throughput, average delay and average response time. Indeed, techniques and architectures

require a paradigm shift from reactive and centralized networks towards massive, ultra-reliable and proactive networks that may operate in wide areas and remote scenarios, with thousands of devices, across several wireless and wired domains, uncertainty, high dynamics, rare events and unpredictable interference.

TIMING targets to develop these concepts by prototyping a demonstrator of e2e TSN services in intra and inter -TSN domains with an accurate reference timing delivery and QoS/scheduling enhancements. From an e2e perspective, TSN services entail the support of operators' transport networks. Traffic of different nature is commonly encapsulated at Layer 2 for traffic engineering purposes and transmitted over an optical transport network. As soon as operators' networks provide support to TSN traffic, scheduling and traffic shaping will guarantee the committed QoS of TSN flows and allow for the coexistence of TSN and BE traffic on the same network infrastructure.

In this document we set the foundations of the activities in TIMING, by defining the preliminary SW/HW architecture and its components for a deterministic network, that will enable end-to-end reliable TSN services for mixed traffic types in an industrial scenario, by integrating TSN Ethernet, WiFi TSN, and the components that enable the TSN network control.

TABLE OF CONTENTS

1 INTRODUCTION

TIMING aims at the design of a solution that enables end-to-end reliable TSN services for Industry 4.0 supported by the operators' infrastructures that are currently relying on a best-effort traffic management basis. TIMING subproject 1 analyzes the TSN support in the Ethernet and Wi-Fi segments and identify the enhancements to be made to support ultra-reliable, low latency communications (URLLC), with high reliability, high availability and low jitter for TSN services in coexistence with non-TSN (e.g., best-effort) ones. Scheduling solutions to support these enhancements will be devised. Moreover, solutions to automate the deployment of e2e TSN services with assured performance will be designed, relying on a hierarchical control plane and a digital twin QoS estimation tool that includes accurate TSN traffic models.

A high-level description of the project components is depicted in Figure 1, along with the name of partners involved in each component. The scenario considered is composed of two factory premises interconnected through a transport network. The factory floors will be equipped with TSN-enabled Wi-Fi and Ethernet network devices, and interconnected by means of a transport network able to cope with the TSN requirements imposed by the end-to-end services to be conveyed over the infrastructure. Hence, the scenario is composed of two types of control domains (i.e., the factory floor and the transport network) each one implementing its own SDN-based control, and all of them connected to the orchestration layer residing on top.

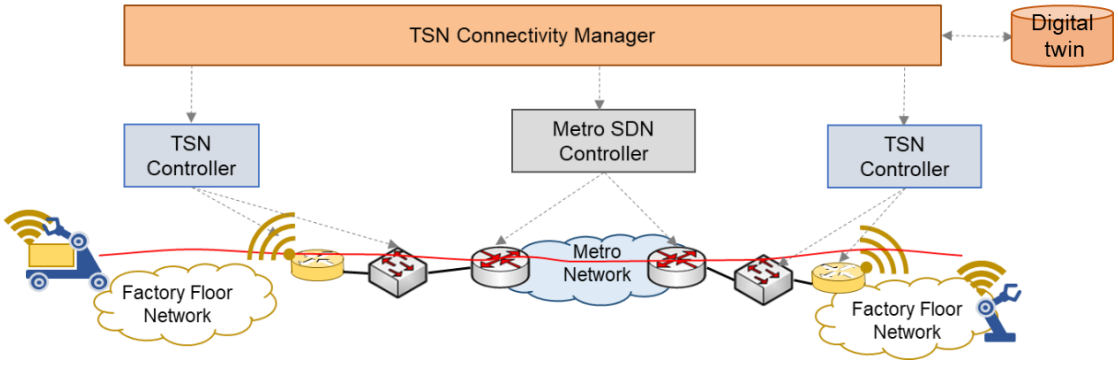

Figure 1. Block diagram of the TIMING general architecture including wireless and Ethernet segments, controllers, connectivity manager and digital twin.

This document describes the components included in the TIMING architecture, namely:

- Access segment (placed in the factory floor in Figure 1), in section 4
- Ethernet TSN segment (placed in the factory floor as well), in section 5
- SDN controller for TSN and metro networks, in section 6
- Connectivity manager, in section 7
- Digital twin, in section 8, and
- Communication interfaces, in section 9.

Besides, it will serve to build a common understanding and as a guide for the coordination of the TIMING activities.

2 PROJECT COMPONENTS

This section includes a brief description of the project components depicted in Figure 1, which are further detailed in sections 4 through 9.

2.1 ACCESS NETWORK AND SDN CONTROL

The access network refers to the devices that provide wireless connectivity between the mobile robots and the TSN network backhaul. From the robot / backhaul perspective, it is seen as a transparent bridge that seamlessly interconnects the devices located at its edges.

Physically, the access network is a set of Wi-Fi TSN nodes, each of them located either on a robot (STA) or on the network side (AP). All the Wi-Fi TSN nodes include at least one physical Ethernet TSN interface which is used to communicate with the network infrastructure, and one wireless interface to communicate between APs and STAs.

The Access Network must support mixed-criticality traffic (i.e., time critical / best-effort traffic), which refers to the support of traffic classification and QoS provisioning to each type of traffic, as well as precise time synchronization to enable the devices coordination to the same base time. These features are provided through three planes: the data plane, the synchronization plane, and the control plane. These planes are provided through four virtual interfaces set up over the Wireless / Ethernet interfaces:

- A synchronization application will be the interface to receive/transmit synchronization frames and will manage the synchronization plane.
- A TSN controller interface is required to manage the control plane (quasi-static TSN configuration). An additional scheduling interface is required to interconnect the wireless modem low-level scheduling with the scheduling algorithms developed in the project.
- The data plane will be provided by a reconfigurable, hardware interconnection between the wireless and the Ethernet interfaces.

The SDN-based control and configuration of the access network will be managed by the SDN-TSN Controller, which will be designed and developed in the project. To do this, the controller will interface the access network devices to (1) compose the access network topology, (2) configure the access points to cope with the services requested from the orchestration layer and (3) enable the monitoring of the access network. The SDN-TSN Controller is described in section 6.1.

2.2 ETHERNET TSN NETWORK AND SDN CONTROL

Orolia's contribution to the TSN Metro segment of the project will consist of the integration of its TSN-capable Ethernet switches to implement the interfaces between the different network segments of the TIMING network scenario supplied by other participants of the project. Specifically, our Ethernet-based Ethernet switches will interface with the core telecommunication network supplied by TID and the wireless TSN devices of IKERLAN. Hence, the role of our devices will be twofold. On the one hand, our Ethernet-based switches will handle the transport and forwarding of critical data between the wireless segment of the network and

its subsequent transport over the 5G/Telecommunication network. To this end, our switches will feature the necessary adaptations and customizations to successfully integrate with the telecommunications network, such as the implementation of TSN translation modes at the boundaries with the telecommunication that allow the successful injection of TSN streams over the core carrier network of TID. In addition, we will also supply TSN-compliant Ethernet interfaces to reach IKERLAN's wireless TSN bridges.

On the other hand, our TSN switches will also have to be included in the general control and resource management architecture of TIMING that allow the joint orchestration of critical data transport over the different segments of the hybrid wireless and Ethernet-based scenario of TIMING. This will imply the provision of the necessary data management interfaces to supply the appropriate configuration and operation parameters to our TSN switches and the deployment of a central network configuration (CNC) module that will interface with the TSN controller to receive optimized operational settings to govern the operation of the wired, Ethernet-based segment of the TIMING TSN network supplied.

Thus, our data plane implementation will be based on our custom TSN designs featuring

- compliance for advanced time-aware traffic shaping (802.1Qbv) with preemption (802.1Qbu & 802.3br),
- redundant data forwarding (802.1CB),
- time-based ingress policing (802.1Qci), and
- low-latency data transmission modes.

Network-wide synchronization of the TSN system will then be provided using our in-house implementation of the PTP synchronization protocol with the possibility of using advanced telecommunication profiles for enhanced interoperability. Moreover, the PTP synchronization will be tied to an accurate GNSS time reference supplied using a specialized network time server (SecureSync). The operational parameters of the synchronization component are be accessible to the common network orchestration framework of TIMING, for the harmonized management of synchronization across the different network segments of the project.

Lastly, the control plane will be supported by an implementation of the centralized management paradigm of TSN defined in 802.1Qcc provided over a custom REST API interface.

As in the access network case, the ethernet TSN network will be managed by means of the SDN-TSN Controller. Such entity will act as a unified SDN-based controller for the whole TSN network segment, which is composed by the access (Wi-Fi) network and the wired ethernet TSN one introduced in this section.

2.3 METRO NETWORK AND SDN CONTROL

The transport network interconnecting the TSN-enabled islands is a conventional packet-switched network offering programmatic, standard interfaces that can be leveraged by an SDN controller for properly providing desired configurations.

The packet-switched network elements, in the form of routers and switches, can be interconnected by means of optical transmission networks. Furthermore, such packet-switched elements could incorporate specialized technologies allowing to provide high performance

behavior to some of the traffic flows. Examples of those technologies are sophisticated queueing mechanisms, such as Hierarchical Quality of Service (H-QoS), or novel data plane solutions like Flexible Ethernet.

Moreover, in the packet-switched domain the usage of Traffic Engineering (TE) policies could be considered to provide differentiated behavior based on the nature of the flows.

All this variety of options and solutions, even if technically possible, cannot be fully used in any situation. It depends of course on the functional capabilities offered by the available equipment, but also of the level of control that can be enabled in a certain scenario. In the more extreme case, the actor providing the connectivity will have full control of the network elements underneath, then making possible a fine granular control of the traffic flows. On the opposite case, the actor providing the connectivity will only have the possibility of creating overlay connectivity (e.g., VPN) on top of an infrastructure which cannot be controlled at all, being only possible to act on the connectivity ends. In the case of TIMING, it is expected to explore different situations for understanding the implications and impact.

As per the SDN Controller, the initial proposal is to leverage on the TeraFlow SDN (TFS) controller. This is a new type of open source, secure cloud-native SDN controller originally proposed in the Teraflow project (https://www.teraflow-h2020.eu/) attempting to specify a reference architecture for an advanced SDN controller. Such concept of SDN Controller has been successfully transferred to the industry in the open source initiative in ETSI, the main Standardization Development Organization in Europe. An Open Source Group has been established around TeraFlow (https://tfs.etsi.org/) guaranteeing the sustainability of the effort beyond its seminal project.

The TFS controller will integrate the standard NFV and MEC frameworks, aiming also to develop features for flow management (service layer), and integrating packet-switched, optical and microwave network equipment (infrastructure layer) via the interaction of standard data models. TFS targets as well the development of a Carrier Grade SDN controller for B5G and 6G networks, providing extremely high levels of reliability, redundancy, and security.

TFS intends to cover a wide variety of network types, ranging from distributed edge-computing, through a transport backhaul (including optical and microwave solutions), to the network core. Thus, it is suitable choice for the overall goals of TIMING.

2.4 TSN CONNECTIVITY MANAGER

As mentioned, TIMING considers different concerns in regards control network capabilities. On one hand, TIMING requires the precise control of the TSN-enabled areas representing the manufacturing production premises. On the other hand, TIMING also considers the presence of intermediate conventional transport network that could allow for some centralization of the manufacturing processes while guaranteeing some of the TSN service expectations (that is, determinism and reliability).

Those independent control network concerns should interwork properly to allow the operation end-to-end. The element responsible of orchestrating all the control components is the TSN Connectivity Manager. This element will interact with both the TSN Controllers and the Metro

SDN Controller in order to provision and enforce the connectivity end-to-end with the necessary guarantees as needed by the final service.

To do this, the TSN Connectivity Manager will rely on standard-based mechanisms or interfaces that will also be supported by the underlying controllers. Such kinds of interfaces will be properly identified and extended, if needed, during the execution of the project.

2.5 TSN MODELS AND DIGITAL TWIN

The Digital Twin (DT) is the component in charge of evaluating the feasibility of the candidate solutions computed and provided by the TSN CM at the provisioning time of a new TSN service. This evaluation is provided as the expected performance in terms of the target KPIs and metrics of both the new TSN connection request and the already established connections, regardless of their type. Moreover, this evaluation is performed by combining static data (e.g., network/topology configuration) with monitoring/telemetry data (e.g., actual traffic) that is provided via TSN CM interface each time a new computation is needed. The outcome of the DT for each performance evaluation will contain the necessary content and in the agreed format to facilitate autonomous decision making of any TSN CM provisioning action.

The engine of DT is based on continuous fluid-flow queue models that are solved by means of ordinary differential equation solvers. Thus, the target network, established connections, and new TSN connection request under evaluation are modelled as a queue system and solved to produce the necessary outcome. Besides algorithms and queue models needed to compose and solve the target queue system, flow traffic models for TSN and other types of traffic will be loaded to generate traffic flows that are propagated through the queue systems and compute relevant KPIs and metrics.

Note that monitoring/telemetry traffic data can be used, among other functionalities, to tune traffic generators aiming at reproducing a virtual network instance as similar as possible to the physical system on the current moment and conditions of the TSN connectivity request evaluation.

3 NETWORK REQUIREMENTS

3.1 PROVISIONING

The TSN technology is a native layer-2 technology, based on IEEE 802.1 standards. When considering the extension of the connectivity further to the boundaries of a manufacturing premises, several options could be considered. For instance, connections by means of layer-2 VPN (L2VPN), Ethernet VPN (EVPN), layer-3 VPN (L3VPN), etc. All these solutions have pros and cons in terms of functionality, overhead, flexibility, etc.

More recently, with the advent of 5G technologies, the concept of network slicing has been generalized. Through network slicing the network provider is expected to provision a connectivity service end-to-end with guarantees in terms of Service Level Objectives (SLOs).

In the case of the solutions being proposed in IETF, the northbound interface (NBI) that is being defined for Network Slicing is assuming a technology-agnostic approach. This implies that a new component, called Network Slice Controller (NSC), will process the slice request deciding how to perform the slice realization, either by using conventional services (i.e., L2VPN, L3VPN, EVPN) or more sophisticated mapping (e.g., OTN, specific resource allocation such as in Flexible ethernet, etc.).

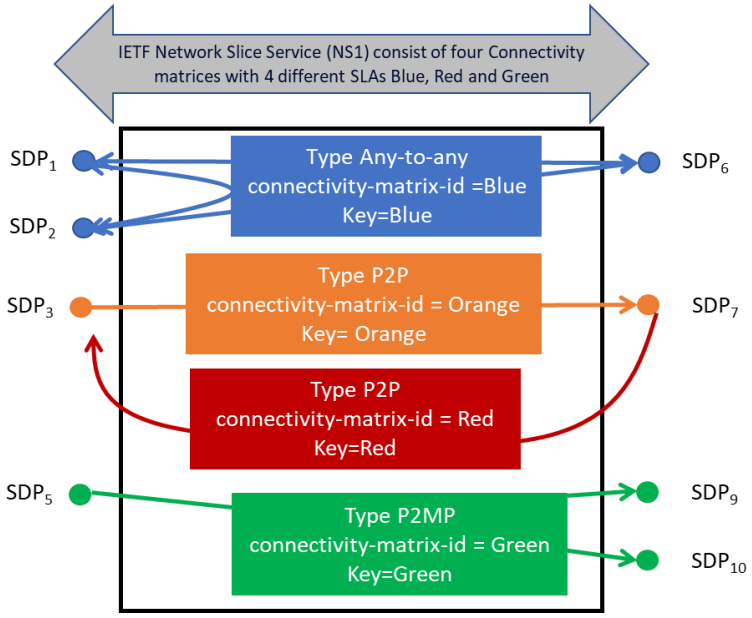

Figure 2. The main principles of the IETF NBI for network slicing.

The NBI permits the definition of several Slice Demarcation Points (SDPs) as defined by the customer requesting the slice, expressing the way as well in which the customer expects those SDPs to be connected (i.e., point-to-point, any-to-any, point-to-multipoint, etc.) as well as the associated SLOs. Additional information about traffic matching to specific slices could be required (e.g., which VLAN is matched to a particular slice).

As said, the NSC will receive the slice request and will be in charge of determining which per-technology network controller (i.e., optical, packet-switched, etc.) activate for provisioning the service.

In the case of TIMING, the requirements will be analyzed to understand if the service can be supported directly for a conventional service (L2 or L3VPN) or if it should be provided in the form of a network slice.

3.2 NETWORK MEASUREMENTS

To properly obtain the status of the network, it will be necessary to retrieve information (including alarms and events) collected directly from network elements through multiple protocols, such as NETCONF/YANG. Alternatively, streaming telemetry (gRPC) can be used. In case that monitoring or telemetry information could not be extracted from the network elements, it can be considered the utilization of external probes that can collect status

information by analyzing traffic flows injected in the network. The received data shall be stored, analyzed, and retrieved to perform additional closed-loop automation and big data analysis.

The way in which data will be measured in TIMING is yet under analysis, and will depend on the final capabilities of the equipment involved. If this is a limiting factor, external active probes will be considered.

3.3 KEY PERFORMANCE INDICATORS AND REQUIREMENTS

The main Key Performance Indicators (KPIs) to consider within TIMING are related to the specific characteristics of TSN technologies. These are:

- **Latency**. Determinism is not simply about low latency, but about deterministic latency, that is, to guarantee that the packets are delivered before a given latency deadline (i.e. in-time service) or within some time range (on-time service), this is why we include:
- **Jitter**, as variation of latency, that must be below a threshold in most cases necessary for industrial processes and devices. In all these goals of deterministic latency and bounded jitter, the control plane can smartly configure the network for providing the required guarantees.
- **Reliability**. Different approaches can be taken in this respect, such as the enablement of protection paths or the replication of information between source and destination.

In the case of TIMING the challenge resides on extending such KPIs particular to TSN technology to connectivity schemes not conceived initially with such stringent requirements. One approach in this direction is the technical solutions being defined in the IETF DetNet working group, which intends to port this kind of requirements to IP/MPLS based networks. This approach can be of relevance for TIMING as well.

4 ACCESS 802.11 TSN SEGMENT

This section focuses on the design of a Wi-Fi TSN node that can fulfill the wireless communication requirements established in the TIMING project. This section provides: an overview of the 802.1 (TSN) standards for wireless access, a detailed description of the HW/SW architecture of the node including the external interfaces for the data, control, and synchronization planes, the description of the PHY/MAC layers of the custom Wi-Fi protocol, and a preliminary design of the architecture to deploy scheduling algorithms on top of the Wi-Fi TSN node.

4.1 802.1 TSN SOLUTIONS FOR WIRELESS ACCESS

There are two standard wireless technologies that are aiming to provide Wireless TSN-like connectivity: 5G and IEEE 802.11. Through this section, we review their current status and major performance objectives.

5G is one of the major technologies to enable Wireless TSN. 5G is designed to support Real Time (RT) applications using the Ultra-Reliable and Low Latency Communications (URLLC) traffic profile [Dur16]. Both simulations and real measurements have demonstrated that 5G Radio Access Network (RAN) can provide a worst-case latency in the range of 1 ms [Sac18] and a cycle

time of 2-3 ms for specific configurations [Jia19]. Therefore, 5G can provide the QoS required by several industrial applications, though it is out of the most demanding ones, which require sub-ms latencies. Regarding the clock synchronization in the 5G radio access network, the 3GPP has recently defined a procedure to enable clock synchronization within the 5G system (5GS) [God20], and some authors have also studied in parallel the performance of such synchronization [Hu21][Sch21].

The second main wireless TSN candidate is IEEE 802.11, a widely used WLAN technology (commercially known as WiFi). IEEE 802.11 already supports precise clock synchronization using the time/fine time measurement schemes [Ibr18]. In addition to that, the last amendment, IEEE 802.11ax [WLAN21], introduces some mechanisms, such as the trigger frame, that enhance the network efficiency and that partially enable time-aware scheduling. However, 802.11ax performance is still not enough for the targeted QoS in some use cases [Cav19]. The recently released amendment, IEEE 802.11be [Ada21], of which devices will be briefly released, include more enhancements to reduce the worst-case latency, such as multilink operation. Finally, next Wi-Fi releases are expected to target ultra-reliable communications and even lower latency and efficiency for short packets, thanks to the introduction of advanced scheduling techniques [Ada21].

As indicated previously, it is worth mentioning here again that TIMING is focusing on Wi-Fi TSN solution and not on 3GPP TSN. Therefore, in the following section, we present the details of the Wi-Fi TSN technology that will be used in this project.

4.2 HW AND SW NODE ARCHITECTURE

Figure 3 displays the hardware/software architecture of the Wi-Fi TSN node and highlights its functionalities, interfaces, and associated subsystems. The hardware integrates the physical communication interfaces (Ethernet TSN and Wi-Fi TSN), an RT traffic translation entity to exchange frames between the Ethernet TSN and Wi-Fi TSN, and a PTP Hardware Clock (PHC).

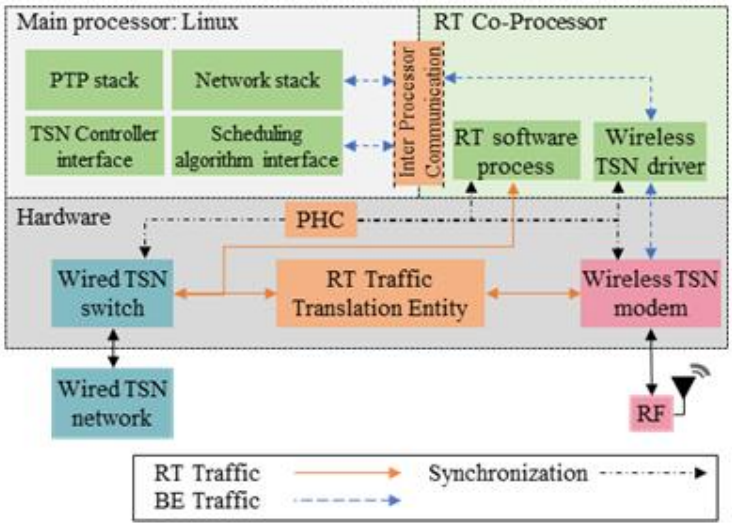

Figure 3. Block diagram of the Wi-Fi TSN node internal HW/SW architecture.

The Ethernet TSN is a standard Ethernet TSN interface and will include at least one physical port, and IEEE 802.1AS (PTP) capabilities to synchronize the local time of the Wi-Fi TSN node to the

Ethernet TSN network [TSNTG], and IEEE 802.1Qbv to support Scheduled Traffic and mixed-criticality traffic. On the other hand, the Wi-Fi TSN modem uses a custom Wi-Fi PHY/MAC layer to support TSN requirements such as time-triggered communication and time synchronization [Sei20]. The design of the PHY/MAC layers of the Wi-Fi TSN modem can be found in subsection 4.3.

The software layer includes two main subsystems: the main processor, which runs a general-purpose operating system, and an RT co-processor, which runs a BareMetal application. The main processor provides the main control application, a set of drivers to gain access to the hardware configuration and control, a standard TCP/IP network stack, and a set of applications that are required to operate the Wi-Fi TSN node, including:

- **a PTP stack** to implement IEEE 802.1AS on top of the Wi-Fi TSN modem / Ethernet TSN interface,
- **the TSN controller interface,** which will be used by an external TSN controller entity to exchange the Wi-Fi TSN node capabilities and the TSN configurations, and
- **the scheduling algorithm interface** to communicate with different wireless scheduling approaches.

The scheduling algorithms are implemented locally at the Wi-Fi TSN node main processor and are further described in section 4.4. On the other hand, the RT co-processor implements the driver of the Wi-Fi TSN modem and the RT software process. The RT software process controls the aspects of the Wi-Fi TSN modem that require real time operation, whereas the Wireless TSN driver exposes an interface to send/receive frames to/from the Wi-Fi TSN modem, as well as provides a control interface that will be used by the scheduling algorithm and by the TSN controller.

The architecture defined here presents different platform units that have been placed in specific SW/HW layers but that can be moved to a different layer according to the application needs.

4.3 PHY AND MAC LAYER DESCRIPTION

4.3.1 PHY layer

The PHY layer of the Wi-Fi TSN node has been co-designed with its MAC layer taking into consideration the targeted traffic profile scheduled by IEEE 802.1Qbv (short packets periodically generated that require bounded low latency). It includes two waveforms (DL and UL) based on the legacy IEEE 802.11 OFDM PHY combined with frame aggregation. The DL waveform is used to transmit frames from the AP to the STAs and the UL waveform is used to transmit frames from the STAs to the AP. Both waveforms use the same basic OFDM symbol structure and the same (de)coder (interleaver, convolutional coding, modulator, etc.), which are based on the IEEE 802.11g standard. Figure 4 displays an example of two frames using the DL and UL waveforms respectively.

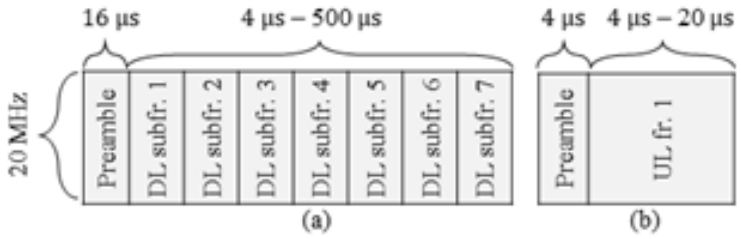

Figure 4. Examples of (a) downlink frame transmitted by a Wi-Fi TSN AP, (b) uplink frame transmitted by a Wi-Fi TSN STA.

The DL waveform has two key components: a long preamble structure like the IEEE 802.11 legacy preamble, and the use of PHY frame aggregation that reduces the preamble overhead. The preamble duration is 16 µs and it has two fields: the Short Training Field (STF), and the Long Training Field (LTF), as in IEEE 802.11g. The modulation and coding scheme (MCS) and length of each subframe inside a DL frame are decided by upper layers and thus the frame does not need to carry any extra parameter to be decodable by each of the users. This is translated in very low control overhead compared to IEEE 802.11, which always includes several PHY parameters through the SIGNAL field.

The UL waveform is used by the STAs to transmit frames to the AP. The UL Waveform allows the transmission of only one subframe (compared to multiple subframes of the DL), and the use of a short preamble to enhance efficiency compared to conventional Wi-Fi.

4.3.2 MAC layer

The MAC layer of the Wi-Fi TSN modem follows a flexible TDMA structure based on the periodic transmission of superframes with a configurable duration, as shown in Figure 5. The superframes are conformed by 3 fundamental periods that provide different functionalities: the RT Downlink period (RT DL), the RT Uplink period (RT UL), and the BE period (BE). The first two periods are used to transmit TSN frames with RT needs, whereas the BE period is used to transmit frames that do not have any RT need. The RT DL and RT UL periods are scheduled to avoid inter-device interferences, whereas the BE period uses the IEEE 802.11 CSMA/CA access scheme.

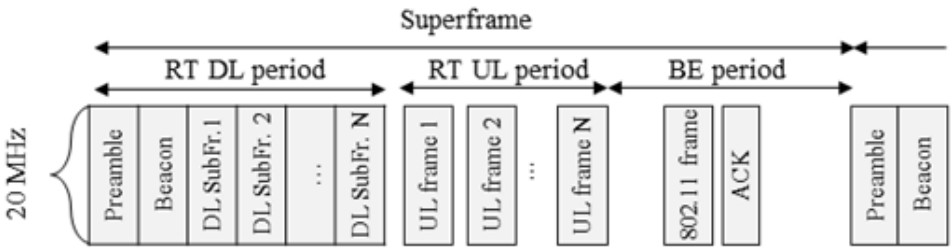

Figure 5. Example of superframe structure of the Wi-Fi TSN network.

Each superframe can comprise an indefinite number of RT periods with arbitrary order, and it must start with a RT DL period with a beacon frame used for synchronization purposes. Afterward, it presents the period to transmit UL information from the STAs to the AP. Finally, the superframe ends with a BE period used to transmit non-RT frames.

4.4 DATA PLANE

The Wi-Fi TSN node forwards the BE traffic through software using the TCP/IP network stack included at the SO of the main processor, an Ethernet TSN driver, and the inter-processor communication mechanism to access the Wi-Fi TSN modem. Both the Ethernet TSN driver and the Wi-Fi TSN modem exposes a standard network interface (layer 2), and the network stack is used to route the frames between the interfaces according to the selected rules.

4.4.1 Hardware-based forwarding

The Wi-Fi TSN node architecture includes a hardware RT Traffic translation entity that enables the RT traffic exchange between the Ethernet TSN and Wi-Fi TSN modem interfaces [Sei22]. As displayed in Figure 6 the translation entity includes two channels to quickly process and generate frames in the format required by each specific interface and two translation tables to keep the network scheduling information structured and accessible from the software when needed. The data translation is done by hardware means, ensuring minimum and guaranteed RT traffic translation latency.

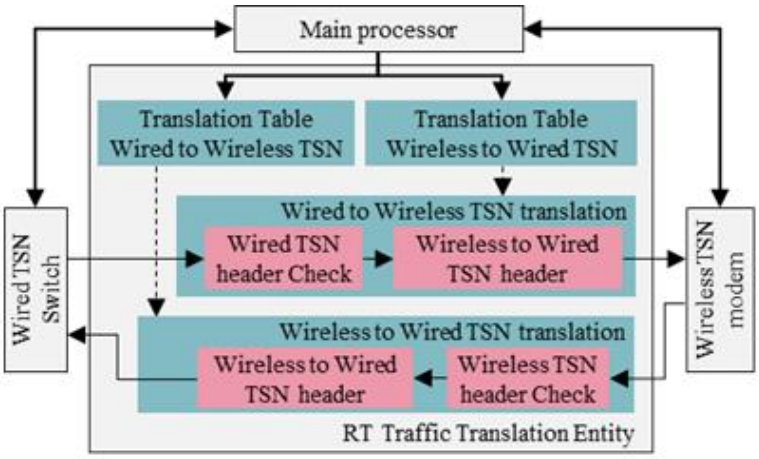

Figure 6. Block diagram of the RT traffic translation entity.

The operation is as follows. When the Wired to Wireless TSN translation receives a TSN frame from the Wired TSN switch, it compares the Ethernet header with the information stored in the translation table entries. When the channel finds the header in the table, it uses the entry information to build the Wireless TSN header. Then, it writes the frame in the corresponding Wireless TSN queue as indicated in the translation table. The opposite direction has identical behaviour. The Wireless TSN modem transmits a frame to the RT Traffic Translation Entity. Then, it searches its header in the Translation Table. If it finds the header, the channel replaces the Wireless TSN header with the Wired TSN header and sends the frame to the switch.

4.5 SYNCHRONIZATION PLANE

The Wi-Fi TSN node synchronization plane is configured to act as a boundary clock. A boundary clock includes one interface that will be configured as a PTP slave (either the Ethernet TSN or Wi-Fi TSN modem interfaces), and the remaining interfaces as PTP masters. The timing translation between the Ethernet and Wi-Fi TSN modem interfaces is performed through the PHC of the Wi-Fi TSN node.

The purpose of the PCH is twofold. On the one hand, it is used as the source time for the timestamping unit of the communication interfaces, and it also accepts modification of its internal time (offset/drift) to adjust the node time to the common base time of the network. The sync plane design allows the time extension in both directions (Ethernet TSN to Wi-Fi TSN modem, or Wi-Fi TSN modem to Ethernet TSN). Since the grand master clock in the TIMING architecture is located at the AP side, the Ethernet TSN interface of the AP is configured as PTP slave, whereas its Wi-Fi TSN modem is configured as slave. The STA, in return, has the opposite configuration. The hardware is supported by a PTP stack software that receives the timestamps from the slave interface and that computes the time synchronization error. This software process also applies the time synchronization error to the PHC to correct the internal node time.

4.6 CONTROL PLANE

The control plane of the Wi-Fi TSN node is used to configure the scheduling of the RT frames along the Wi-Fi TSN node. As in an Ethernet TSN switch, the Wi-Fi TSN node must include a control interface exposed to the TSN network controller. The control interface is accessible from both the Ethernet TSN interface, and the Wi-Fi TSN modem interface and is used to:

1) transmit the capabilities of Wi-Fi TSN node to the TSN controller (e.g., bandwidth, latency capabilities, list of devices connected to the Wi-Fi TSN node, etc.),
2) receive the specific TSN configuration that must be applied to the Wi-Fi TSN node, including superframe period, number of TSN flows and, for each TSN flow: direction (DL/UL), frame size, superframe period, transmission/reception instant of each frame, expected reliability, and maximum residence time of the frames of each flow within the Wi-Fi TSN node.

This information is then translated to a low-level register configuration structure that is applied to both the Wi-Fi TSN modem and to the Ethernet TSN interface.

4.7 UL AND DL SCHEDULING FOR MIXED TRAFFIC TYPES

The scheduler architecture proposed in TIMING is shown in Figure 7 for an illustrative simplified case where two access points (AP) serve some users each one. The architecture consists of multiple APs with overlapping coverage areas that are interconnected and can exchange information with sufficiently large rate and reduced latency to implement coordinated spatial-reuse and multi-AP scheduling (also called coordinated OFDMA) [Ada21]. We consider time division duplexing (TDD) full-duplex communications with per-frame dynamic design of the UL and DL intervals which are selected by an external control loop that takes into consideration the dynamicity of UL and DL traffic.

The studied scenario consists of multiple user terminals that are connected to one of the APs with multiple flows per user. These terminals are expected to exchange mixed traffic (TSN, best-effort, and others) with the network. TSN flows will convey isochronous periodic traffic from control devices (e.g., sensors and actuators) and aperiodic bursty traffic from other time-sensitive applications such as AR/VR. On the other hand, best-effort flows carry aperiodic random traffic with different (or even null) requirements in terms of latency, jitter and/or throughput. To cope with the randomness of incoming traffic and wireless channel capacity, we assume there are independent FIFO queues per flow and per link (UL & DL).

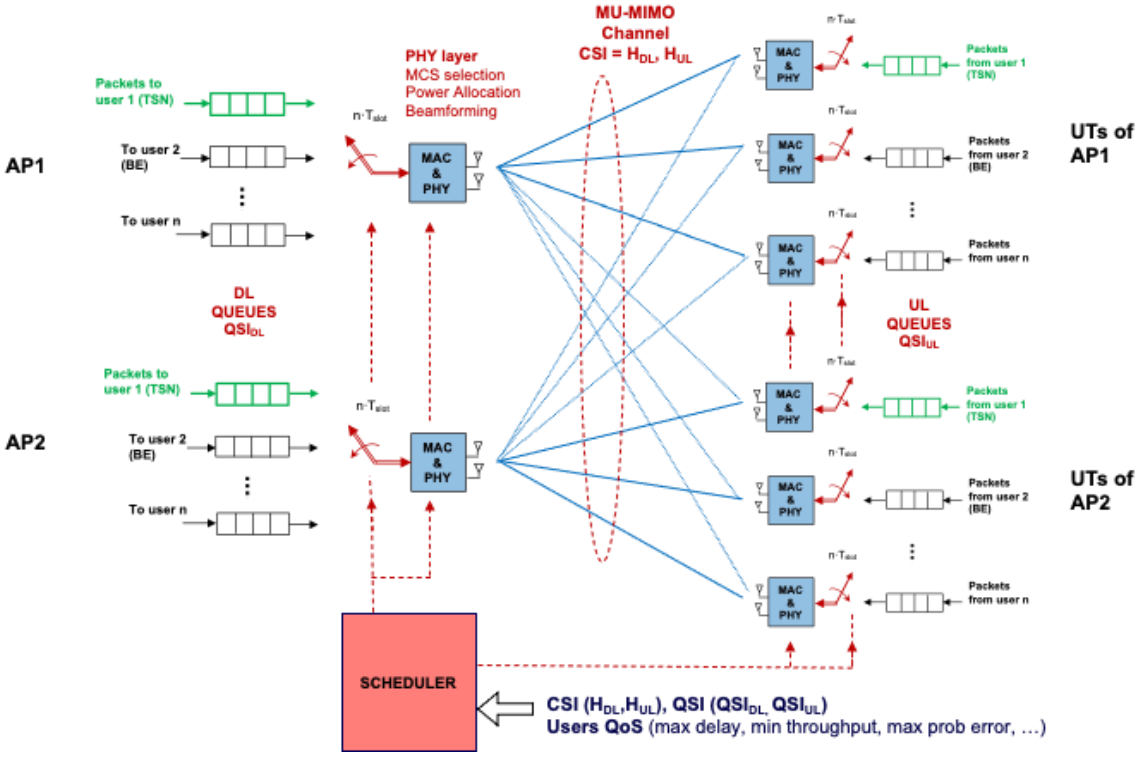

Figure 7. Architecture of the scheduler.

The adopted multi-user medium access mechanism is time division multiple access (TDMA). TDMA frames are slotted using a clock signal that is shared across the whole TSN network as standardized by 802.11as. At the beginning of the frame, the scheduler dynamically assigns the available radio resource blocks (RBs) under a slot-by-slot basis among the active flows and decides the MCS, power and beamforming spatial precoders of transmitting users. The scheduler takes decisions based on the QoS requirements (latency, jitter, rate, etc.) of flows and the collected information about channel state information (CSI) and queues state information (QSI) for both the UL and DL. Other relevant information for scheduling such as the delay and jitter of the latest queued packets will also be gathered.

In order to enhance the wireless access performance, APs are envisaged to be equipped with multiple antennas allowing beamforming and spatial multiplexing (MU-MIMO). The extension to coordinated multi-AP beamforming [Ada21] will be studied as well. As shown in Figure 8, the exchange of quantized information among the AP with the controller through the appropriate interface will allow coordinating interference through the management of radio resources [Muñ12].

As already introduced in 802.11ax, we consider a coordinated access based on OFDMA where the whole bandwidth is divided into different subbands (blocks of consecutive subcarriers) that the scheduler can assign to different flows. In the UL case, all users must be received at the AP with the same power and MCS to avoid near-far effects associated with gaining control and finite precision of analog-to-digital converters.

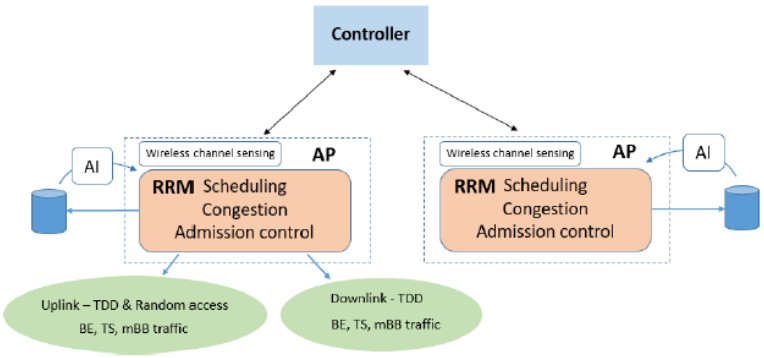

Figure 8. Multi-AP coordination and interfacing through the AP TSN controller.

We distinguish two scheduling modes. Critical isochronous traffic is expected to be scheduled according to the synchronous mode whereas other classes of time-sensitive traffic according to the asynchronous mode:

- **Synchronous mode**: frames are divided into orthogonal temporal windows (slices) assigned to every class of traffic (compatible with IEEE 802.1Qbv). Windows are defined by the orchestrator during provisioning and can be reconfigured during operation to maintain the QoS of all TSN flows. In this mode, TSN and best-effort flows are scheduled separately. TSN traffic is scheduled during the TSN window by considering the different QoS requirements of active TSN flows and the available CSI and QSI. During the best-effort window, best-effort flows could be scheduled in the same way as TSN traffic (without QoS constraints) or, alternatively, transmitted directly using the contention mechanism in legacy WiFi (CSMA/CA).

- **Asynchronous mode**: Following the philosophy of IEEE 802.1Qcr (Asynchronous Traffic Shaping), multiple classes of traffic (e.g., TSN and best-effort) are jointly scheduled giving maximum priority to TSN flows to satisfy the negotiated QoS. This approach has the following benefits with respect to synchronous mode: a) TSN packets arriving during the best-effort window do not have to wait in queue until the next TSN window (lower latency); b) the time assigned to TSN and best-effort traffic is not predefined but it is adjusted dynamically based on the requirements of TSN flows and the availability of radio resources; c) guard bands and preemption is not required provided that slots are sufficiently short and best-effort packets can be conveniently fragmented at L2.

The previous set of features will be first implemented in a simulator, that will be used to test the schedulers and to measure KPIs. Later, for the integration of the scheduler into the IKL hardware platform and further end-to-end demonstration, some of the scheduler features will be selected by considering their potential gain in terms of QoS (delay, jitter, throughput, etc.) as well as the feasibility of integrating them into the IKL platform.

5 ETHERNET TSN SEGMENT

To support the interconnection with the telecommunication carrier network, Orolia will contribute to the architecture of TIMING with its Ethernet-based TSN switches. These switches deliver a platform composed of a series of elements that operate together: a time synchronization subsystem to supply the time base throughout the system, a basic 1/10 Gb/s Ethernet subsystem that provides the underlying Ethernet communication service, a switching

subsystem for sending Ethernet frames between ports, a TSN subsystem for sending data in a deterministically and robustly, and a monitoring subsystem that collects data and statistics on the condition of the network and the TSN system. See Figure 9 for a description of the control plane solution.

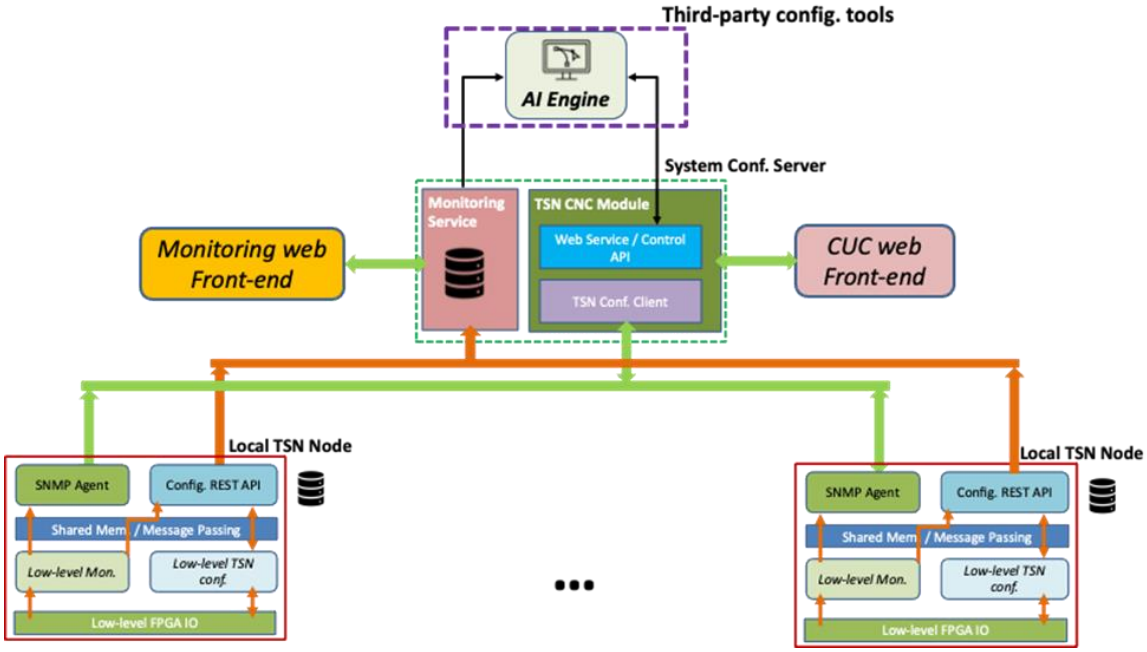

Figure 9. Generic overview of the control plane for the Ethernet-based TSN solution supplied for TIMING.

These elements are implemented using specialized FPGA logic. The proposal is based on other research projects and an intense internal development process. Within this framework, the different components of the solution have been developed on platforms based on the Xilinx Zynq-7000 [ZYNQ] family (Z-7030, Z-7015), although they have also subsequently been ported to other platforms with higher levels of performance and resources, such as Zynq-MPSoC architectures, UltraScale+, or even Alveo acceleration cards.

For the use case of TIMING, we have selected our TSN implementation for a multi-port platform based on the Xilinx Zynq-7000 devices, which feature a programmable FPGLA logic (PL) section and a processing system with an embedded ARM processor. As seen in Figure 10, the FPGA logic contains our designs for the main components supplying the TSN, Ethernet data transport, message switching, and network monitoring functionalities of the nodes.

Their corresponding management and operation are handled through the appropriate drivers, software stacks, and configuration utilities of the embedded Linux operating system running on the ARM processor of the processing system. These components will be described in greater detail in the following subsections.

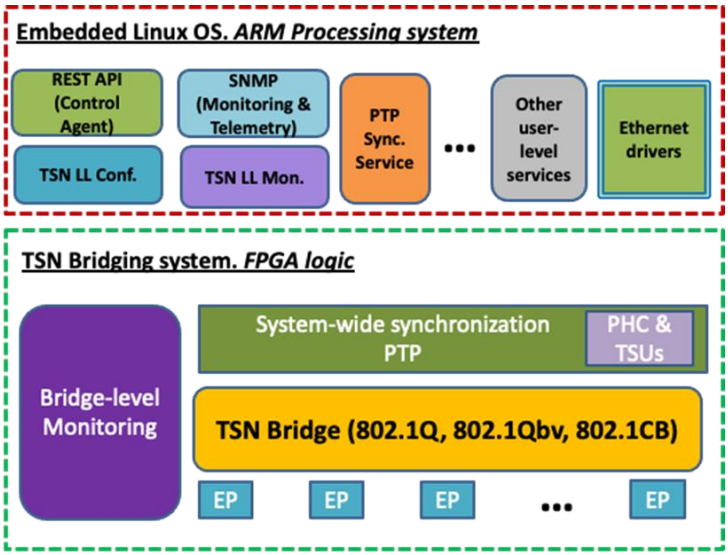

Figure 10. Generic block diagram of the TSN Ethernet switch solution delivering a multi-port TSN solution with data switching, network status monitoring, and advanced PTP synchronization over high-capacity Gigabit Ethernet links.

5.1 HW AND SW DESCRIPTION

As seen in Figure 10, the Ethernet TSN nodes feature a modular architecture whereby the operation of the system is supported by the simultaneous and cooperative use of various FPGA and software modules. The FPGA modules of the design consist mainly of a series of intellectual property modules ("*IP cores*") and FPGA coprocessors that implement the corresponding services for Ethernet data transport, time and frequency distribution, as well as the TSN processing of network data flows. In addition, these are complemented with a set of software elements of the embedded Linux operating system residing in the ARM processor of the device, which manage and govern the operation of the FPGA modules through custom drivers, libraries, or low-level APIs. These elements are reviewed in greater detail in the following sections.

In addition, our solution is delivered using the WR-Z16 node [WRZ16], which represents one of the flagship appliances for high-accuracy timing distribution in high-performance networks (see Figure 11).

Figure 11. Orolia's WR-Z16 node is supplied to support the Ethernet TSN switching functionality of the Ethernet segment of TIMING.

5.1.1 HW description

As seen in Figure 10, the architecture of the TSN switch is supplied using an integration of proprietary TSN designs into the FPGA logic of the WR-Z16 node. Hence, this design is highly modular and combines the joint actions of different main subsystems to deliver the functionality of a TSN-capable Ethernet data switch. These components are enumerated in the points below.

- **A timing subsystem**. This is responsible for supporting the operation and execution of the PTP timing distribution service through FPGA implementations of components such as a PTP hardware clock (PHC) and time-stamping units (TSUs). This system is discussed in greater detail in the corresponding section for the synchronization component of our design.
- **A TSN subsystem**. This is tasked with supplying the deterministic and robust data forwarding functionalities of our design through custom FPGA implementations of the TSN modules for time-aware traffic shaping (802.1Qbv) with preemption (802.1Qbu), seamless redundancy (802.1CB), traffic identification and data forwarding (802.1Q), and per-stream ingress filtering (802.1Qci).
- **An Ethernet data subsystem**. This supplies the essential Ethernet communication service of our TSN switches and is supplied in the form of specialized FPGA cores implementing 802.3-compliant Gigabit Ethernet MAC modules, Physical Coding Sublayer/Physical Medium Attachment (PCS/PMA) blocks, GTx transceivers, and direct memory access (DMA) engines for data transfers towards the embedded processor.
- **An Ethernet switching subsystem**. This implements the TSN bridging functionality of the nodes by allowing the forwarding of TSN data frames between the different ports of the TSN switch. This functionality is supported through the integration of specific FPGA interconnects and crossbar modules steered through the TSN system.
- **A monitoring subsystem**. This supplies the design of the network status reporting system that will be used to monitor the condition of the network in real-time and optimize its operation accordingly. In our design, this is supplied through the implementation of dedicated FPGA probing modules for gathering statistics related to the congestion of the network, bandwidth usage, or the time-stamping of select data frames to monitor their end-to-end latency.

5.1.2 SW description

The operation and management of the different subsystems and FPGA modules of the Ethernet TSN switch is managed using the appropriate utilities, APIs, interfaces, kernel drivers, and libraries from an embedded Linux operating system residing in the ARM processor of the WR-Z16 node. These elements, which provide all the necessary support to govern the operation of the different subsystems of the device and allow the provision of the deterministic data forwarding service over Ethernet of our devices with advanced synchronization, can be broken down into several main groups tasked with overseeing the operation of each main subsystem of the device. These are enumerated below:

- **The embedded operating system**. This provides the main execution framework for the operation and configuration of the TSN node. In our design, this will be supplied as an embedded Linux kernel running on the ARM-based processing system of the Zynq-7000 device that our design is based on. Thus, the role of the operating system would be to provide the essential foundation for all the necessary software components and kernel

modules supporting the operation of the PTP synchronization service and the effective data transport over the TSN network.

- **The PTP synchronization service**. This supplies the system-wide network time reference that will be used for steering the operation of the TSN subsystem and thus deliver the expected deterministic data handling performance of the solution. This is provided as a built-in service for the operating system based on the IEEE 1588 specification with support for different profiles, such as the default or the G.8265.1 telecom profiles. Thus, the PTP service is responsible for the execution of the different protocol state machines to compensate offset between the master and slave nodes of the network. This is achieved by processing the hardware time stamps retrieved at the TSUs of each port to periodically supply offset correction values to the internal time counter of the PHC module.
- **The Ethernet network drivers**. This provides the necessary operating system-level support to enable the transmission of Ethernet data frames over the Ethernet data ports of our solution. Our solution uses a custom Ethernet driver design capable of handling Gigabit Ethernet data over the FPGA-based Ethernet subsystem of the FPGA and of retrieving hardware-based time stamps from the TSUs for the PTP synchronization component.
- **A node-level configuration service**. This provides the implementation of a REST API service to allow the reception of configuration parameters for the TSN subsystem on the FPGA logic and to adjust the operational parameters of the PTP synchronization service. Hence, the REST API will allow for the specification of data routing preferences for TSN streams, gate control list (GCL) policies for the preemptable TAS modules, packet identification settings, and message filtering policies. In addition, it will also allow for the specification of settings related to the PTP synchronization service, such as the *clock class*, *clock ID*, …
- **A monitoring agent**. This supplies the implementation of an SNMP monitoring agent for propagating system-related status information to the network management system. In our solution, the local SNMP agent of our devices will be used to propagate status information related to the operation of the TSN service, such as the bandwidth usage of the system, the occurrence of congestion, or the ingress/egress times of select user flows for monitoring their end-to-end latency in a distributed system. These indicators will then be supplied to the network optimizers in the upper layers of the TIMING architecture to compute optimized settings for the different network elements of the project.
- **A low-level TSN configuration utility**. This implements a low-level configuration utility that interfaces directly with the TSN subsystem on the FPGA. Thus, the utility is meant to handle the optimized configuration parameters supplied to the local TSN switch through the REST API, carry out their translation to the data format expected at the IP cores of Orolia, and then apply their corresponding settings.
- **A low-level TSN monitoring utility**. This implements a low-level monitoring utility that interfaces directly with the TSN monitoring subsystem of the FPGA. Thus, it is tasked with retrieving network operation indicators from the FPGA and then perform their submission to the local SNMP agent to allow their propagation to the network management service and the network optimizer elements of TIMING.

5.2 802.1 TSN SOLUTIONS FOR THE TRANSPORT NETWORK

The following section contains an overview of the different TSN components of that have been selected for their inclusion in the Ethernet TSN switch solution supplied for TIMING. These are implemented as the specialized FPGA modules whose operation is supported through the corresponding software elements discussed in the previous section.

5.2.1 The traffic identification module (802.1Q)

Identification of the different traffic classes that will be processed in the system. This component is also responsible for applying the appropriate VLAN-tagging and encapsulation settings to the different time-sensitive data streams handled over the TSN network. These flows are identified by their unique tuples of VLAN priority (VPRIO) and VLAN ID (VID) values, which are used to process the flows on a hop-by-hop basis throughout the network and route them to their intended destination.

5.2.2 The enhancements for seamless redundancy (802.1CB)

Identification and handling of the highly critical TSN flows that will be transmitted using the additional protection mechanism of redundant transmissions. This effectively increases the robustness and reliability of the network by ensuring the delivery of the more critical flows in TIMING using redundant data forwarding routes.

5.2.3 The time-aware traffic shaper (TAS - 802.1Qbv)

Execution of cyclic data forwarding schedules synchronously to the time reference of the network. The TAS module allows the enforcement and allocation of bandwidth reservation capacity to the different flows of the network through the execution of a cyclic forwarding schedule (gate control list – GCL). GCL policies allow the definition of dedicated time slots reserved for the transmission of different flows in accordance with their respective priority levels and are thus one of the main drivers of the deterministic performance of our solution.

5.2.4 The enhancements for preemption (802.1Qbu & 802.3br)

This mechanism represents an extension of the deterministic data forwarding capabilities of the TAS module by allowing the designation of the different flows traversing the network as either *express* (extremely critical data) or *preemptable* (lower priority data). Hence, this allows the effective prioritization of the express messages of the system to the detriment of interfering preemptable traffic to further reduce the attainable jitter of the express flows and thus increase the overall determinism of the solution. This is supported through the implementation of targeted modifications to the original specification of the TAS module (802.1Qbu) with the application of specialized extensions to the Ethernet MAC module allowing the fragmentation of the preempted lower-priority frames (802.3br). It is expected that the integration of the Ethernet solution with the carrier network of timing will leverage this mechanism to further reduce the end-to-end latency of the different control flows propagated through the demonstrators of TIMING.

5.2.5 Ingress packet-based policing (802.1Qci)

This allows the definition of time-based packet filtering policies to enforce the reception of the TSN flows propagated through the network at the different intermediate switches and endpoints within their allocated time slot, and thus reject any traffic in breach of its allocated capacity. This mechanism can help preserve the integrity and quality of the network in the event

of the occurrence of anomalous behavior or a misconfiguration of any of the nodes in the system using time-based filters capable of rejecting traffic received outside of its allocated slot. Furthermore, the implementation of this component will allow the generation of statistics related to rejected/accepted traffic to complement the network status information supplied to the monitoring system of the network.

5.3 DATA PLANE

The data plane is responsible for the transmission and handling of the different network flows propagated through the TSN network. In the Ethernet TSN switches, this functionality is provided through the combination of different FPGA-based subsystems for TSN data processing and Ethernet data communication. These are enumerated in the following sections.

5.3.1 The Ethernet subsystem

This module provides the fundamental, high-capacity data transport of our Ethernet TSN switches through the instantiation of different specific FPGA components on a per-port basis.

- **A Xilinx DMA engine**. This allows the high-capacity transmission and reception of Ethernet data frames at the embedded ARM processor of the switch.
- **A Gigabit Ethernet MAC**. This supplies the medium access functionality for the Ethernet implementation of our device. Our design also features compliance with the 802.3br specification for interspersing express traffic and frame preemption.
- **A PCS/PMA module**. This implements the line coding scheme of the Ethernet interface and supplies the medium adaptation layer to the SFP-based optical fiber ports used in our devices.
- **A high-capacity serial data transceiver**. This provides the serialization and data transmission functionality of the serialized Ethernet data over the optical interfaces of the design with the use of the GTx transceiver primitives from Xilinx.

5.3.2 The Ethernet switching subsystem

This provides the data forwarding capability between the different ports of our Ethernet TSN solution. This functionality is in turn supplied through the instantiation of an FPGA interconnect module that serves all the Ethernet ports in the design and that is steered through the TSN subsystem, hence allowing the forwarding of TSN data frames between the different ports of the solution.

5.3.3 The TSN subsystem

This implements a robust and deterministic data forwarding service to ensure the integrity and timely delivery of the critical data streams handled in TIMING. The functionality is supplied through the instantiation of the different FPGA modules for the TSN subsystem outlined in section 5.2 on a per-port basis.

5.4 CONTROL, SYNC AND MONITORING PLANES

5.4.1 Control Plane

The control and management architecture of our devices adheres to the centralized management and configuration paradigm outlined in the 802.1Qcc specification whereby all the

elements of the network are managed from a centralized point of control. The solution supports this approach with the provision of specific APIs, management components, and configuration distribution utilities. The generic control architecture of the solution can be examined in Figure 9, and its components are described in greater detail next.

- **A centralized network configurator (CNC)**. This will be supplied as a standalone device that will allow the centralized management of all the Ethernet TSN switches offering in the context of TIMING. Our CNC will be supplied as a customization of a proprietary network management solution with REST APIs to allow the connectivity with the different TSN switches of the network. Likewise, our CNC will also feature customized APIs to allow the integration of our CNC solution with the system-wide orchestrators and software-defined networking techniques deployed by the other members of TIMING. The design of the CNC component is modular enough to allow its deployment alongside that of the centralized network monitoring service in the same device.
- **A node-level configuration service**. This will implement the REST API component of each local Ethernet TSN switch described in section 5.1.2. In our design, this will be supplied as a proprietary API service tasked with receiving configuration parameters from the CNC component. The local configuration service of each node will then interface with the low-level configuration utility of our TSN switches to propagate the optimized settings received from the CNC all the way down to the FPGA TSN subsystem.

  The REST API of the configuration service has been built using Swagger [SWAG] for streamlined usability and includes self-generated user documentation. The API is accessible on port 8201 of our devices and allows the inspection and visualization of the self-generated documentation through a web browser. The parameters and functionalities related to TSN are accessible in the area for "*TSN: metrics and configuration*", which contains the definitions of the appropriate endpoints to designate the different TSN-related resources that can be accessed and managed through the API. To that end, the users of the API may resort to the different HTTP methods for manipulating and upgrading the different resources accessible through the API, such as GET, POST, PATCH, PUT, and DELETE. The methods POST, PATCH and PUT represent the three main methods of configuring the unit, i.e., *save*, *apply*, and *save and apply*, respectively.

- **A low-level TSN configuration utility**. This is tasked with receiving the optimized settings delivered from the CNC module through the REST API of the local configuration service of the node. This component has already been described in section 5.1.2, and is supplied as a proprietary software module tightly integrated with the REST API service and the FPGA TSN subsystem.

5.4.2 Synchronization Plane

The synchronization plane is responsible for ensuring the distribution of an accurate time reference throughout the network. This will be used to steer the operation of the TSN subsystem and ultimately deliver a deterministic data transmission service. In our design, this functionality is provided through the combination of several of the proprietary blocks and modules that make the PTP synchronization solution. These are enumerated next.

- **The FPGA timing subsystem**. This supplies the proprietary implementation of the timing synchronization stack on the FPGA logic of our devices. As indicated in section 5.1.1, this includes specialized components for elements such as the PTP hardware clock (PHC) or the time-stamping units (TSUs), which provide the necessary hardware-level support to deliver a high-accuracy implementation of the PTP synchronization protocol.
- **The PTP synchronization service**. This supplies the proprietary implementation of the software service for the PTP protocol and its corresponding synchronization algorithm. As explained in section 5.1.1, the PTP synchronization service is executed as a standalone component in the proprietary Linux image of our devices and is tightly integrated with the FPGA timing subsystem and the Linux network drivers to ensure the propagation of a high-accuracy time reference throughout the network.

5.4.3 Monitoring Plane

The monitoring plane of the solution is responsible for supplying network status and condition data to the network monitoring service. Our solution will contemplate the possibility of including network statistics related to the occurrence of data congestion, bandwidth usage, or the evolution of the end-to-end latency of select flows. These indicators could in turn be provided to the network optimizers of the other members of TIMING to derive optimized settings for the Ethernet TSN switches of our solution. The monitoring plane offered for TIMING is supported through the components enumerated below.

- **A centralized network monitoring platform**. This is supplied as a proprietary monitoring solution integrating time-series data bases, an SNMP exporter, and a custom web GUI. The centralized monitoring service will aggregate all the TSN-related statistics and telemetry produced at the local monitoring services of the TSN switches forwarded over SNMP. In addition, the monitoring service will implement a specific query interface, such as PromQL, to allow its integration with the network optimizer of the other members of TIMING. Moreover, the implementation of the centralized monitoring service is flexible enough to allow its deployment alongside that of the CNC in the same device.
- **The SNMP agent of the Ethernet TSN switches**. This supplies the proprietary implementation of the SNMP agent of the TSN switches tasked with uploading network status information to the centralized monitoring platform. As explained in section 5.1.1, this is one of the services of the embedded Linux image of our devices and is tasked with collecting all the data sources of the node, including TSN-related data, to allow their propagation to the centralized monitoring platform.
- **The low-level TSN monitoring utility of the Ethernet TSN switches**. This delivers a software utility implementing a low-level interface between the SNMP agent and the FPGA monitoring subsystem of our devices. As seen in section 5.1.1, it is supplied as a standalone service of our design and is therefore tasked with fetching low-level indicators from the monitoring modules of the FPGA relating to bandwidth usage, congestion, or time-stamped events.

6 SDN BASED CONTROL FOR TSN AND METRO NETWORKS

One of the objectives of the TIMING project is to enable the TSN services deployment automation by means of a control and orchestration framework. In this regard, the Software

Defined Networking (SDN) paradigm [ONF] has become a de facto standard to enable network programmability and it has been applied to different data plane technologies [Cor19][Guo21][Pag23]. In brief, the Controller is the main entity in the SDN model and is responsible for collecting the connectivity requests coming from the upper layers of the architecture and configuring the underlying data plane to satisfy the requirements of such requests. To do this, the Controller implements an NBI which is used by the upper layers (e.g., the orchestrator) to request parametrized connectivity services. Such requests are then translated by the Controller into a set of configurations over the data plane, which are sent to the devices through the Southbound Interface (SBI), thus enabling the so-called programmability of the network. In addition, monitoring capabilities are implemented in the Controller that collects information about the devices through the SBI and exposes it to the upper layers by means of the NBI.

Considering the above, this section presents the SDN-based control solutions that will be designed, developed and deployed in TIMING. As illustrated in section 1, the scenario under consideration is composed of two factory floor premises, which are interconnected through a transport network. The factory floors will be equipped with TSN-enabled Wi-Fi and Ethernet network devices, and interconnected by means of a transport network able to cope with the TSN requirements imposed by the end-to-end services to be conveyed over the infrastructure. Hence, the scenario is composed of two kinds of control domains (i.e., the factory floor and the transport network) each one implementing its own SDN-based control, and all of them connected to the orchestration layer residing on top. The sections below describe the functional architecture and communication interfaces of both the TSN-aware SDN Controller that will reside in the factory floors, and the SDN Controller of the transport network.

6.1 TSN-AWARE SDN CONTROLLER

This section depicts the preliminary design of the SDN Controller for the TSN domain. As said, the TSN domain is composed of a heterogeneous data plane where two different transport technologies, namely Wi-Fi and wired Ethernet, coexist. Hence, the two major features that the controller will implement are the ability to configure TSN connectivity services, and the capacity to control and monitor heterogeneous data plane flows in a unified way.

6.1.1 SDN-TSN Controller functional architecture

The SDN-TSN Controller is depicted in Figure 12 and presents a modular architecture where different functional components operate to configure TSN-aware connectivity services.

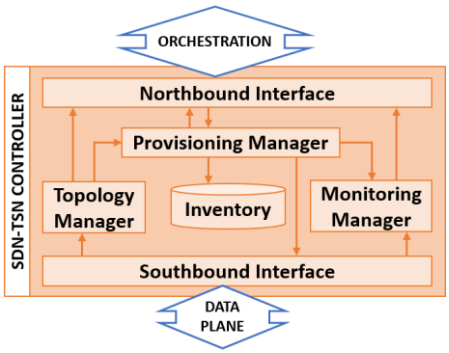

Figure 12. SDN-TSN controller architecture

Following a bottom-up description, the SBI implements the set of communication protocols needed to enable the control, configuration and monitoring of the data plane devices. In this regard, such communication protocols need to provide the means to:

1) discover the data plane topology and the capabilities of the devices,
2) configure TSN data flows in the devices, and
3) collect information about the performance and state of the devices.

Above the SBI, three main components implement the functional capabilities of the controller. First, the Topology Manager (TM) is responsible for building the graph that abstracts the data plane topology, also considering the availability of resources. Second, the Provisioning Manager (PM) is the entity that manages the connectivity service configuration process. To this end, it collects the requests coming from the upper layers (e.g., the Connectivity Manager) through the NBI, uses the information provided by the TM to compute the data plane devices that need to be configured to fulfill the requirements posed in the request, and configures the data plane accordingly by means of the SBI. The active connectivity services are stored in the Inventory. The third functional entity is the Monitoring Manager (MM) that takes care of managing the collection of monitoring data associated with the established TSN flows, processing and formatting it, and finally sending the resulting information to the upper layers through the NBI. Residing on top of the controller functional architecture, the NBI implements the communication interface between the controller and the orchestration layer, that is the Connectivity Manager. Such interface implements the messages needed to request connectivity services and to collect monitoring data that will in turn feed the Digital Twin (DT).

6.1.2 SDN-TSN Controller communication interfaces

The SDN-TSN controller communicates with the CM through the NBI and with the TSN-capable devices through the SBI. The next sub-sections provide a high-level description of such interfaces and a more detailed preliminary specification is provided in section 9.

6.1.2.1 Northbound Interface

The NBI exposes the operations required by the orchestration layer to provide the connectivity service requested by the client. In brief, the NBI has to implement the means to provide the topology of the underlying data plane, in this case, the TSN network topology. In addition, this interface has to provide operations to enable the configuration and management of the data plane that will support the requested service. Finally, the NBI needs to implement operations to expose monitoring information about the established connections to the upper layer.

6.1.2.2 Southbound Interface

The SBI of the SDN-TSN controller is used to manage and configure the data plane devices of the TSN network, which are the Wi-Fi nodes and the wired ethernet TSN ones. In particular, the functionalities this interface has to implement include the configuration of the devices and interfaces associated to a requested connectivity service, and the monitoring of such devices, that is, providing information about the status of the devices, interfaces, queues, etc. This information has to be properly formatted at the SDN-TSN controller and sent upwards to the upper layers of the system.

6.2 METRO SDN CONTROLLER

As described in section 2.3, the Metro Controller initially considered is the TeraFlow SDN (TFS) Controller.

TFS follows a cloud native architecture conceiving the SDN controller as composed of multiple micro-services. Those microservices interact one with each other using a common integration fabric. Since the controller is not an isolated component in an operational system, TFS can interact with other network elements, such as NFV and MEC orchestrators, as well as OSS/BSS. TFS is in charge of controlling and managing the underlying transport network infrastructure, including network elements of multiple technologies such as optical, microwave links, IP routers, etc., as well as compute nodes at edge or public cloud infrastructures.

Due to the fact of following a cloud-native architecture, TFS benefits from a high application resiliency, motivated by the continuous monitoring of microservices which are restarted in case of detecting whatever misbehavior. Another benefit inherited by the cloud native approach is scalability, since it is possible to accommodate a highly increasing number of requests (i.e., load), simply with the deployment of new microservice instances as needed.

TFS functionalities are divided into core and netApps functionalities, each one based on a single or multiple micro-service(s). The TFS core micro-services are tightly interrelated and collaborate to provide a complete smart connectivity service. On the other hand, TFS netApps consume TFS core micro-services. The TFS netApps provide the necessary carrier-grade features with a dedicated focus on load-balancing, cybersecurity, auto-scaling, self-healing, and inter-domain smart connectivity services.

6.2.1 Northbound Interface (NBI)

Regarding the NBI, TFS considers IETF L2/L3 network models, Traffic Engineering (TE) tunnels, and IETF Transport Network Slices.

The L2/L3 network models are used to provide an intermediate level of abstraction between the connectivity service that the customer requires, and the configuration implemented in the underlying network nodes. It provides the notion of a network service, which really is a set of configurations in the devices that happen to realize the customer needs. It is important to highlight that the services of the network models only exist in the controller. The L3NM model provides a network-centric view of L3VPN services, while the L2NM provides representation of the Layer 2 VPN Service from a network standpoint. In the case of Traffic engineering, the data model is used for the configuration and management of TE tunnels supporting Label Switched Paths (LSPs).

Finally, it is expected to support as well the NBI for slicing. A transport network slice will consist of a set of endpoints (e.g., SDPs), a connectivity matrix between subsets of these endpoints service level behaviors requested for each sender on the connectivity matrix. The connectivity between the endpoints might be point-to-point, point-to-multipoint or multipoint-to-multipoint. Often these slices will be used to satisfy network behavior defined in a Service Level Agreement (SLA).

6.2.2 Southbound Interface (SBI)

As SBI, TFS intends to support the following interfaces:

- ONF Transport API, to interact with Open Line System (OLS) Controllers in charge of managing underlying optical transport networks.
- ONF TR-532, to interact with a SDN Microwave (MW) Domain Controller, which has a network-level view of the underlying MW domain. The SDN MW controller interacts with microwave network elements using ONF TR-532 data models.
- OpenConfig, for the configuration and control of IP router parameters that are implemented in NETCONF server.
- ONOS P4, to manage next-generation SDN whiteboxes based on the P4 paradigm.

7 CONNECTIVITY MANAGER

TSN Connectivity Manager (CM) optimizes the e2e flows, analyzing the performance impact when traversing TSN systems, and no-TSN systems in the network. This module, as shown in Figure 13, interacts with TSN and Metro SDN controllers over different network domains, and connects with DT, which evaluate the e2e performance. The proposed solution distinguishes wired and wireless devices in a single-AP and multi-AP environments.

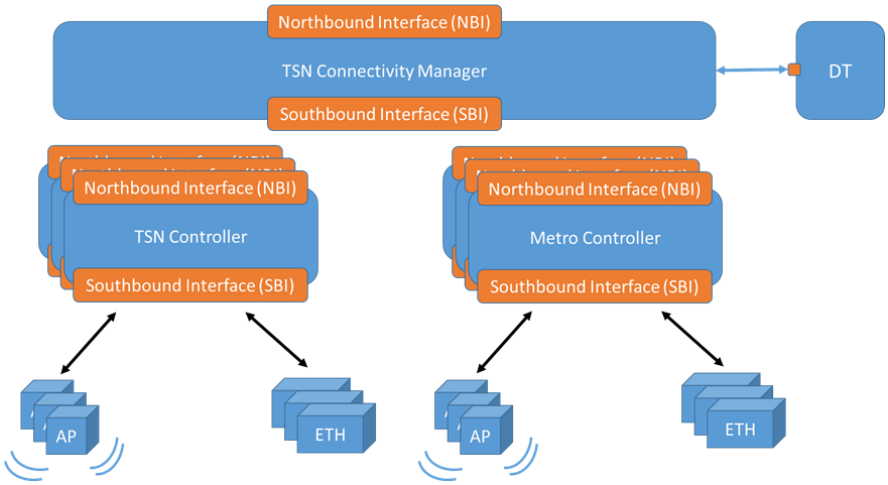

Figure 13. TIMING Project architecture

The objective of this section is (i) to introduce the exposed interfaces by the TSN CM (NBI, SBI), (ii) to discuss the importance of the data model to exchange the information by the interfaces, and finally (iii) to describe the details of the GUI exposed to display the most valuable information.

7.1 FUNCTIONALITIES

The TSN CM module is an entity, located on the top of the architecture, that centrally manages the network calculating the TSN flow paths from the moment of its request. The goal is to ensure the e2e QoS requirements through dynamic allocation of network resources.

This component allows the deployment, configuration, management and optimization of the TSN network resources acting on the underlaying SDN controllers (TSN and Metro), within a unique TSN domain or traversing different TSN domains separated by the operator's transport

network. It simplifies the management process of a complex network infrastructure with low-latency requirements.

The required optimization algorithms in the TIMING project are included in this component, extending the commercial software E-lighthouse Network Planner (ENP) [ENP] as a basis software. This solution allows quick and highly efficient resolution of complex optimization problems. The main tasks are:

- Network resources and capacity discovery
- Network provisioning and deployment via SBI
- Export network data and representation

The TSN CM is responsible of assessing if the underlying communications are meeting QoS values in the candidate-paths through performance evaluation in the DT. If required, the TSN CM can replan the new or established routes to acquire the best performance in the TSN flows without affecting to existing ones [Pav18].

7.2 APIs

The target networks in the TIMING project can potentially use varied technologies and have different constraints (latency, jitter, throughput, etc.). These heterogeneous environments can cover large areas, under the domain of different lower level SDN controllers, but leveraging a centralized manager to orchestrate it, such as TSN CM. The method to operate with the TSN CM is via its open APIs, that will be accessed by different other elements in the TIMING architecture. Nowadays, the use of the APIs is essential on any non-trivial application for the interoperability between its different components, including a set of protocols and tools that access and manipulate the functionality or data. APIs come in many different forms, but they typically provide a standard way for one system to request information or services from another, and to receive responses in a predictable format. In this case, the API is based on the REST paradigm [Son19], a common choice in the industry given its simplified structure, that results in simpler APIs, easier to develop and maintain.

These interfaces can also play a critical role in the research tasks providing a way for developers to extend the functionalities of existing products and services. The exposed architecture (see Figure 14) is composed of three main APIs, the TSN CM Northbound Interface (NBI), the TSN CM Southbound Interface (SBI), and the Digital Twin (DT) Interface.

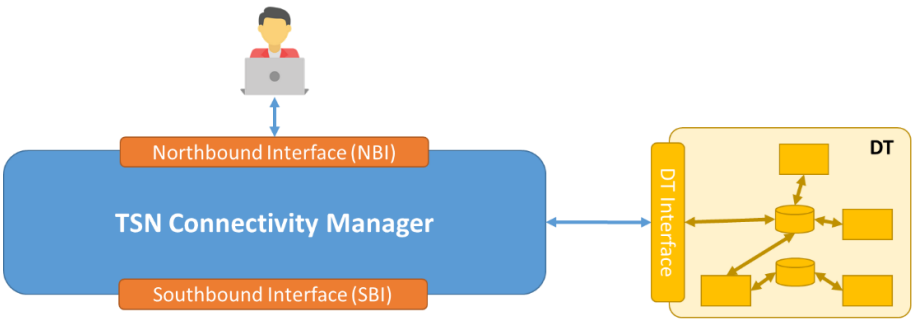

Figure 14. Interconnection of TSN CM and DT

7.2.1 Northbound Interface (NBI)

The NBI interface of the TSN CM is the northernmost interface of the architecture and exposes the service provided to the user, or higher-level systems, facilitating the exchange of information, configuration, and control signals to abstract the complexity of underlaying network devices and technologies. This interface provides a unique and standardized method that defines rules, protocols and paths making network management tasks and integrations easier in TSN architecture. Using this interface, the network operator can perform tasks in both domains (TSN and Metro) and observe performance metrics. The initial approach of the top NBI is detailed in section 9.2.4.1.

7.2.2 Southbound Interface (SBI)

The TSN CM SBI is connecting with the exposed NBIs of the underlying TSN-aware and non-TSN Metro SDN controllers. Target Metro SDN controllers in TIMING, can be both commercial (e.g. Cisco NSO) or non-commercial (e.g., ETSI TFS) SDN controllers for IP networks. The key use cases where TIMING project will concentrate are i) topology discovery, and ii) connection provisioning.

Additionally, section 9.2.3 analyses the standard approaches and best practices widely adopted by the industry to expose the network design and connections.

7.2.3 DT Interface

Once the TSN CM uses the optimization algorithms to estimate the candidate-paths, this module sends to the DT the candidate-paths via DT interface to calculate the performance for each route. Already the KPIs for the candidate-paths are processed, and the DT module returns to the TSN CM the results. The best performance, the selected path. The implementation design of the DT is included in section 8, while the details of the DT interface are included in section 9.2.4.2.

7.3 GRAPHICAL USER INTERFACE (GUI)

The resource management and operations are easier and more intuitive using a Graphical User Interface (GUI) with available data exposed by the NBI of the TSN CM. The objective is to create a prototype GUI allowing the interaction with the physical elements of the network, with minimalist appearance and design according to the project requirements, but will include:

- Visualization and analysis of network components (nodes, links, flows, etc.) and statistics.
- Provides information based on the trace, provision of new connections and internal calls to the DT REST API.

E-lighthouse will draw on its prior experience in creating graphical interfaces for network visualization and analysis tools, such as the ENP [ENP] tool. This will ensure access to the tool for visualizing network elements, retrieving network performance data, and making changes to the network.

8 DIGITAL TWIN

In this section, we briefly describe the main functionalities and requirements of the DT. Then, a preliminary architecture with several modules is proposed. Finally, details about the initial models for TSN traffic to be considered in the DT are presented.

8.1 FUNCTIONALITIES AND REQUIREMENTS

First, it is worth recalling that the TSN connectivity manager interfaces the DT for QoS analysis purposes. Specifically, every time a new TSN flow provisioning is requested and the TSN connectivity manager computes a candidate route to support such request, the DT is responsible for computing the expected performance of such new routes. The expected performance is measured in terms of features and/or KPIs to be defined.

For each of the identified feature/KPIs, two type of measures will be provided:

- For the new flow, the expected values of their features/KPIs.
- For the rest of flows (already established TSN and BE flows), the expected impact on their features/KPIs.

8.2 PRELIMINARY ARCHITECTURE

Figure 15 sketches the preliminary architecture envisioned for the DT, that will be based on the architecture presented in [Ve23]. It contains the following basic modules: *i*) a manager module configuring and supervising the operation of the rest of the modules; *ii*) a few modules that include algorithm, models, and the interface with the TSN connectivity manager; and *iii*) a Redis DB that is used in publish-subscribe mode to communicate the different modules among them.

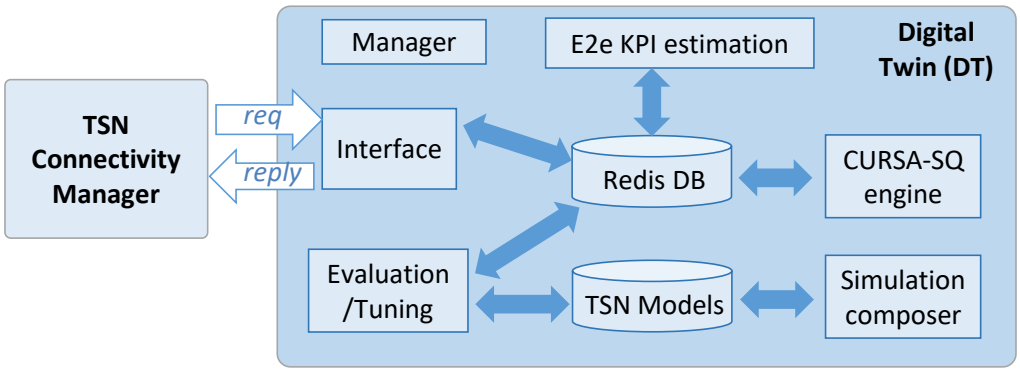

Figure 15. DT preliminary architecture

Table 8-1 presents a preliminary description of the data to be included in the interface between the TSN connectivity manager and the DT each time a new computation is requested. In addition, Table 8-2 shows the reply to be provided by the DT after finalizing computations.

Table 8-1: Interface definition (req)

Block	Attribute	Definition
Network (*G*)	*N*	Set of nodes, each with the required parameters e.g., switching/ routing capacity.
	E	Set of currently active edges (adjacencies) between nodes in *N,* each with the required parameters e.g., distance, maximum capacity
	IF	Set of network interfaces *IF*, each with the required parameters e.g., edge assignment, speed
Established TSN and BE flows (*F*) (for each flow $f \in F$)	*type*	Type of service/traffic
	route	Current route, defined as a sequence of network interface
	maxTraffic	Maximum actual traffic
	KPI_if	List of KPI metrics per interface of the flow
New TSN flow request (*r*)	*type*	Type of service/traffic
	route	Candidate route to evaluate, defined as a sequence of network interfaces
	maxTraffic	Maximum expected traffic

Table 8-2: Interface definition (reply)

Block	Attribute	Definition
Request (*r*)	*KPI_e2e*	Expected e2e KPI
Flows (*F*) (for each flow $f \in F$)	*KPI_if*	List of expected KPI metrics per interface if request *r* was established

Regarding the CURSA-SQ engine, we adopt the efficient flow-based approach was proposed by the authors in [Ru18] based on a continuous queuing model for network flows analysis, named CURSA-SQ. The CURSA-SQ queue model is a continuous G/G/1/k model with a First-In-First-Out (FIFO) discipline based on the logistic function. CURSA-SQ can be applied for a wide range of scenarios, such as generating realistic data for Machine Learning training purposes [Ra18] and for accurate, scalable, and predictive near real-time estimation of end-to-end KPIs in fixed and converged fixed-mobile networks [Be20].

The evaluation/tuning and simulation composer modules work together to translate the characteristics and parameters received in the incoming request into a simulation plan that needs to be solved by means of calls to the CURSA-SQ engine. Once the results of such simulations are available in the Redis DB, the e2e KPI estimation module aggregates and post-processes such results to compose the e2e KPIs metrics needed to provide as reply of the received request.

8.3 CURSA-SQ MODELS FOR TSN

CURSA-SQ extensions to model network interfaces supporting both BE traffic and TSN flows simultaneously under different TSN standards are presented next. Specifically, the following two approaches are initially considered:

1. **Synchronous (*sync*) TSN model**: aligned with IEEE 802.1Qbv standard [802.1Qbv], time slices are reserved to TSN traffic flows, whereas BE traffic is transmitted in between consecutive protected TSN slices. Note that this option, although ensures the QoS of TSN flows, might limit that of the BE traffic in the case that the protected time slices are not fully consumed by the TSN traffic.
2. **Asynchronous (*async*) TSN model**: aligned with IEEE 802.1Qcr standard [802.1Qcr], TSN flows use exactly the transmission time that they need, and inter-packet gaps can be filled with BE frames of suitable size, which maximizes BE throughput and reduces its latency.

For the sake of clarity, eq. (8.1) presents the compact basic differential equation that models the dynamics of the state of a capacitated continuous queue system in CURSA-SQ, denoted as $q(t)$ (bytes), supporting *a single flow* (b/s) (see [Ru18] for further details). Let $\hat{X}(\cdot)$ be the amount of input flow received and stored in the queue at time t (i.e., it depends on the state of the queue) and let $Y(\cdot)$ be the flow (b/s) leaving the queue, which depends on the state of the queue, as well as on the fixed server rate μ:

$$\frac{d}{dt}\big(q_i(t)\big) = q'(t) = \hat{X}(q(t), t) - Y(q(t), \mu) \tag{8.1}$$

However, to model the mix of both TSN and BE, we assume the scheme of the network interface detailed in Figure 16a, where n individual TSN input flows (X_{TSN}) that arrive conveniently shaped, are combined with an aggregated input BE flow (X_{BE}).

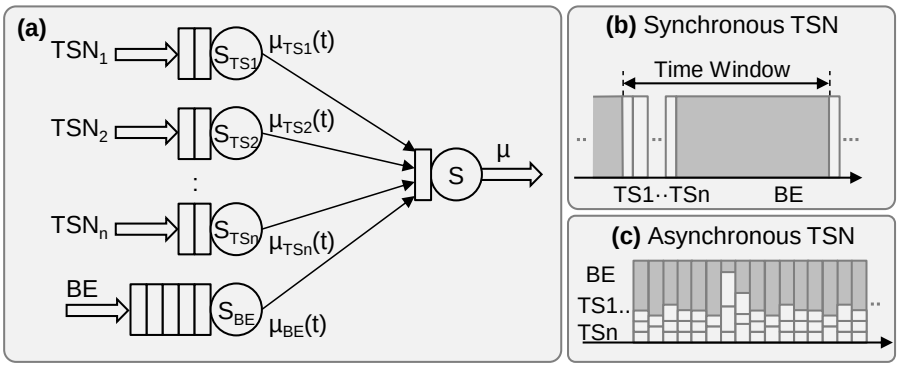

Figure 16. Interface model

Each individual flow i is associated to one continuous capacitated queue system, where its state $q_i(t)$ depends on the input flow and on a server rate $\mu_i(t)$ variable with time; all the individual queue systems access the network interface characterized by a fixed server rate μ. Then, the extended differential equation that characterizes every individual flow in the interface is formally described in eq. (8.2):

$$q'_i(t) = \hat{X}(q_i(t), t) - Y\big(q_i(t), \mu_i(t)\big), \forall i \in TSN \cup \{BE\} \tag{8.2}$$

At this point, the key difference between BE and TSN flows is in terms of the instant server rate $\mu_{(\cdot)}(t)$; while the server rate for TSN flows is a function $f_i(\cdot)$ that depends on their own configuration parameters ϑ_i only (eq. (8.3)), that for the BE flow depends on the remaining

resources available after assigning resources to each TSN flow *i* (eq. (8.4)), as resources are reserved for TSN flows at provisioning time:

$$\mu_i(t) = f_i(t;\theta_i), \forall i \in TSN \quad (8.3)$$

$$\mu_{BE}(t) = \mu - \sum_{i \in TSN} f_i(t|\theta_i) \quad (8.4)$$

To model the *sync* approach, a time window of fixed length (*T*) is defined and time slices for every TSN flow are reserved; the rest of the time window that remains unassigned can be used to BE traffic (Figure 16b). Thus, configuration parameters ϑ_i in the *sync* model are defined in terms of time. Equation (8.5) describes the server rate of every flow under the sync TSN model, where π(·) is the rectangular function that is 1 if the input value is in [+1/2, -1/2] and 0 otherwise, t_i is the center, and T_i the length of the time slice assigned to the flow.:

$$f_i^{sync}(t;t_i,T_i) = \mu \cdot \pi\left(\frac{(t\%T - t_i)}{T_i}\right), \forall i \in TSN \quad (8.5)$$

For the TSN flows, T_i is assigned to serve the expected input traffic rate X_{TSN} in *T*. Then, the summation of $f_i(\cdot)$ in eq. (8.4) results in μ if one of the TSN flows is served, which prevents the BE flow to be served at time *t*, and 0 otherwise, which gives the BE flow the whole capacity of the interface at that time.

On the other hand, in the *async* model, the TSN flows have higher priority than the BE one. Time is processed in small fragments of fixed size, where at every fragment the state of the queues is evaluated and served according to their priority (Figure 16c). To avoid the recurrence caused by evaluating such queues state, f_{TSN} is conveniently approximated to the current input traffic rate X_{TSN} (eq. (8.6)):

$$f_i^{async}(t) \cong X_i(t), \forall i \in TSN \quad (8.6)$$

This decreases the available server capacity for the BE flow.

9 WORKFLOWS AND COMMUNICATION INTERFACES

This section provides a first description of the workflow that will be followed for the provisioning of TSN services over the TIMING infrastructure, and a summary of the communication interfaces identified in the TIMING architecture that means to set the base for their formal specification.

9.1 TSN SERVICES PROVISIONING WORKFLOW

The main goal of the TIMING project is to design, develop and prototype a network architecture able to support TSN services in coexistence with non-TSN (e.g., best effort) ones. To this end, the project will demonstrate an advanced provisioning mechanism that will leverage on a Digital Twin of the network to configure TSN services with assured SLA compliance. In this section we describe the high-level design of the provisioning workflow (depicted in Figure 17), that will be implemented throughout the project. It can be summarized as follows:

- The Client requests a TSN service with a set of requirements through the GUI of the Connectivity Manager (CM) (step 1 in the figure).
- The CM uses the (rich) network topology (collected from the SDN controllers) to compute a set of candidate paths.
- The CM sends the list of candidates and other information (e.g., information about the existing services that use the links of the candidate paths) to the Digital Twin (DT) (step 2).
- The DT returns the path that fulfills the service requirements to the CM (step 3).
- The CM sends the path to be established to the involved SDN controllers (step 4).
- Each SDN controller is responsible for configuring the data plane under its control (step 5).

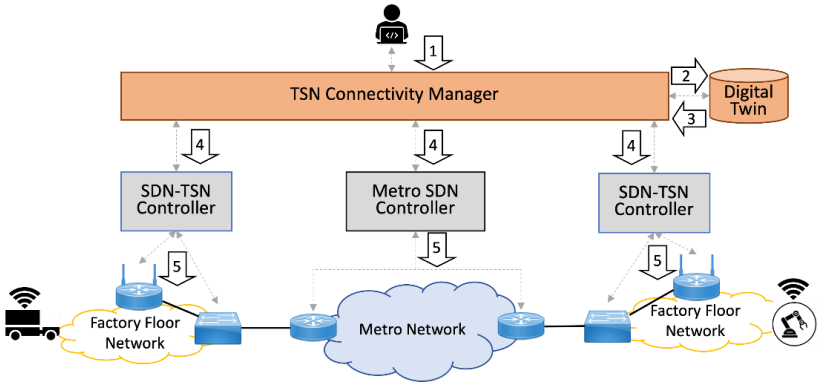

Figure 17. High-level design of the TIMING provisioning workflow

9.2 COMMUNICATION INTERFACES

9.2.1 Data Plane to Control Plane Interfaces

The sub-sections below provide a more detailed description of the interfaces that communicate the SDN-TSN Controller and the data plane components it oversees, that is, the Southbound interface of the SDN-TSN controller that also corresponds to the control and monitoring interfaces exposed by the data plane components.

9.2.1.1 Wi-Fi TSN to SDN-TSN Controller

The control of the Wi-Fi TSN node can be split into two main parts. The first one is related to the SDN-based control, configuration and monitoring of the node as described in section 4.6. The functionalities that enable such control can be listed as:

- *Advertisement of the node capabilities and interfaces*, so the upper layers of the architecture can compose an enriched network topology and compute paths that fulfill the requested service requirements. Such capabilities currently include available bandwidth, supported latency and connected devices. Further ones may be included according to the needs of the project.
- *Configuration of the device* to support the requested connectivity services. This configuration requires a set of low-level parameters including the number of TSN flows and, for each one, the direction (DL/UL), frame size, superframe period, transmission/reception instant of each frame, expected reliability, and maximum residence time of the frames of each flow within the Wi-Fi TSN node.

The second main part of the interface between the Wi-Fi TSN nodes and the SDN-TSN controller is devoted to communicating the controller with the scheduler of the Wi-Fi TSN node, which is described in section 4.7. In this regard, besides low-level information of the AP (like the occupation of the queues, channel information, etc. as described in section 4.7), the scheduler of the Wi-Fi AP requires information coming from the upper layers of the system to operate. Such information is provided by the SDN-TSN controller through this interface. In brief, the scheduler uses information about the active flows of the AP and their characteristics to assign radio resources to them according to their QoS requirements. A first list of the information per active flow needed to this end is provided next:

- *Flow identifier*, which allows to differentiate the different flows that are configured and active in the AP.
- *Flow QoS class*, according to the different traffic classes stated in the proposal (e.g., isochronous TSN, best-effort, and possible others).
- *Flow QoS requirements list for the current node*, which will include, among others, maximum delay, maximum jitter, minimum rate and maximum packet error rate.

In addition, the scheduler will provide monitoring information through this interface. In particular, the scheduler will send the actual values of the QoS parameters (i.e., delay, jitter, packet error rate, etc.) obtained for each active flow. This information will be further used by the orchestration layer to both confirm that the SLAs for the active services are fulfilled and provide feedback to the DT so it can update the models and improve its accuracy.

9.2.1.2 Wired TSN to SDN-TSN Controller

To control the ethernet TSN devices, the SDN-TSN Controller will implement in its SBI a client to the CNC platform offered by the ethernet TSN segment, which will allow the seamless management and control of the different Ethernet TSN switches composing such segment, and the definition of the operational parameters of their corresponding TSN components. Hence, the SDN-TSN Controller will access the REST API exposed by the CNC to supply configuration settings to the TSN components of each switch in the network. To do this, the API will expose configuration parameters related to the operation of the main TSN components in the design, such as those of the GCL schedules of the TAS module, seamless redundancy, VLAN-tagging and traffic identification, or the definition of frame-based ingress policies. A tentative enumeration of some of these parameters follows next; and will be completed in subsequent deliverables on the design of the system.

9.2.1.2.1 Configuration of the VLAN & FRER module for traffic identification, data routing, and redundancy

The API will expose the following parameters to configure the operation of this module. Please note that this specification is preliminary and may be subject to change.

- ***"addr":*** The destination MAC address supplied for the routing of the TSN stream once the corresponding traffic class has been identified.
- ***"vlan_id":*** The VLAN ID value applied to the TSN stream once the corresponding traffic class has been identified.
- ***"vlan_prio":*** The VLAN Priority field applied to prioritize the TSN stream once the corresponding traffic class has been identified.

- ***"filter_en":*** Discard received duplicates if the FRER extensions are enabled for the current TSN stream.
- ***"mac_addr"***: The destination MAC address used to identify a TSN traffic class.
- ***"proto":*** The protocol number used to identify a traffic class.
- ***"ip":*** The destination IP address used to identify a traffic class.
- ***"vlan_id_cfg":*** Unused.
- ***"port":*** The port number used to identify a traffic class. Used in combination with the *"ds"* field.
- ***"vlan_prio_cfg":*** Unused.
- ***"dscp":*** The value of the DSCP Ethernet header field used to identify a traffic class.
- ***"dest":*** The destination port number for a redirected TSN stream.
- ***"has_dest":*** Enable port redirection (i.e., data switching between the Ethernet ports of the device) for the current TSN stream.
- ***"ds":*** Specifies whether the Port number used to identify a traffic class is either the source ('1') or the destination ('0') port.
- ***"redundant":*** Enables the use of FRER data redundancy for the transmission of the current TSN stream.
- ***"red_dest":*** Specifies the egress Ethernet port that will be used for sending the duplicates when the redundant FRER transmissions are enabled.
- ***"red_handle":*** Specifies the handle number of the FRER component for each TSN traffic class to generate consistent values for the sequence number. The handle for each redundant TSN traffic class must always be a unique value.

9.2.1.2.2 Configuration of the TAS module with frame preemption for deterministic data transmission

The API will expose the following parameters to configure the operation of this module. Please note that this specification is preliminary and may be subject to change.

- ***" base_time_tv_sec ":*** The base time offset to start the operation of the TAS module (in seconds).
- ***" base_time_tv_nsec ":*** The base time offset to start the operation of the TAS module (in nanoseconds).
- ***"tas_table_no":*** The number of slots considered for the definition of the GCL policy.
- ***"framePreemptionStatusTable":*** An 8-bit variable with the positional coding of the preemption status of each one of the 8 queues of TAS module. This indicates whether the corresponding queue is designated either as preemptable ('1') or express ('0').
- ***"interval_time":*** The duration of the time slot of the GCL schedule that is currently being configured.
- ***"gate_cfg":*** An 8-bit positional coding variable indicating the gate control status for the slot of the GCL schedule that is currently being configured. An open queue is designated with a '1' for the associated bit position in the vector, whereas a closed queue is marked with a '0'.

These fields can be queried and updated using the HTTP commands of the REST API described in section 5.4.1. A simple example showing how the values of the VLAN module could be updated directly by invoking "curl", in Figure 18 for convenience.

Figure 18. A sample query to the TSN REST API of the TSN Ethernet switches using "*curl*" produced with the Swagger web GUI of our devices.

9.2.2 Metro network devices to Metro SDN Controller

As explained in section 2.3, the metro network segment will be composed of commercial non-TSN devices which will be controlled by the Metro controller through standard interfaces, which have already been described in section 6.2.2.

9.2.3 Control Plane to Orchestration Plane Interfaces

A first specification of the requirements of the SBI of the CM is provided in this section. It is worth noting here that, although different SDN controller implementations may be used for each segment of the TIMING data plane (e.g., TSN and Metro), a set of common functionalities is needed at the SBI of the CM for the orchestration layer to properly operate the end-to-end. Hence, a first set of the main common functionalities that need to be provided through this interface are:

- *Topology discovery and network representation:* This functionality will allow the CM to build the complete end-to-end view of the network, so it is able to compute the candidate paths to be analyzed by the DT.
- *Connection provisioning:* Once the connectivity service has been computed and validated at the orchestration level, the CM manager will be responsible for contacting each SDN controller involved in the service to configure each segment of the network.
- *List of active flows:* In addition to the topology view that each SDN controller will expose from its network domain, it shall also return information about the flows already established. For each flow, among others, the set of parameters to be pushed will include a list of devices, interfaces and links, to estimate the used and available capacity in the traversed network elements for optimal flow routing. Based on this information, different strategies will be studied to explore possible candidate-paths at the CM. After the evaluation of these candidate-paths by the DT, the flow configuration that obtains the best KPIs without disturbing the already established flows will be chosen. After the

selection, the CM will build connection requests to the SDN controllers to reserve the necessary resources in the underlying devices of each involved segment.

For the design and implementation of the SDN/TSN controllers NBI the TIMING project aims at leveraging the best practices and standard approaches most widely adopted by the industry. Two main options are being considered as the most promising:

1. Transport API (TAPI) [May17] [TAPI] is an NBI specification for Transport SDN controllers, that is growing popular in the optical domain as it simplifies access to real-time and historical transport layer data to make the user's tasks easier. TAPI is designed to enable network operators to implement SDN over a multi-layer, muti-domain, and multi-vendor transport infrastructure.
2. The standard defined by IETF [Cle18] is significantly adopted by SDN controllers in the IP domain. In the IETF model, the root element is the class *ietf-network*, which is a list of network nodes, supporting the concept of network hierarchy or network stack. The *ietf-network-topology* class defines a generic topology model including nodes, links and termination points used to augment the ietf-network class. This data model can integrate multiple network topologies for different layers, and overlayed topologies, enabling relationships between them (e.g., nodes in layer 2 could be supporting nodes of a layer 3 connection). The Telecom Infra Project (TIP) [TIP] proposal divides the topology into four layers:
   - Layer 1 provides a close view of the physical representation of network elements and applies to various L1 technologies with different TE topology models.
   - Layer 2 represents Link-layer or Ethernet connections built using various L2 technologies.
   - Layer 3 displays the topology elements located at the network and IGP layer and can be enhanced to provide more details about L3 unicast topology.
   - Layer 4 adds the service attachment points where network services such as L2VPN or L3VPN can be deployed.

These options will be thoroughly analysed in the next steps of the TIMING project and a decision will be taken towards the adoption of the most suitable model, which will be extended to cope with the TSN requirements. The key aspects to consider in the extension to devise, is the accommodation of the TSN-related information into the model. For that, the proposed approach will be the definition of TSN-specific attributes for (i) nodes, (ii) links and (iii) termination points. The specific form of these attributes will be elaborated along the project.

9.2.4 Orchestration Plane Interfaces

9.2.4.1 Top NBI of the Connectivity Manager

The NBI interface of the CM, which has been described in section 7.2.1, will be implemented by means of a REST API. This API is meant to provide a configuration endpoint to upper-level components of the architecture, such as the GUI used to operate the network or external software components like service or NFV orchestrators, etc. The main functionalities that this API will expose are:

- *Connectivity service provisioning and management:* The client side of this interface will use this operation to request parametrized connectivity services. A first enumeration of requirements to be conveyed in such request (which will be completed throughout the

project) includes: source and destination, type of connectivity service (e.g., best effort or TSN), maximum latency and jitter requirements, bitrate, etc.

- *Topological information:* Through this operation the topology of the data plane will be exposed.
- *Connectivity service monitoring:* Monitoring information associated to the active connectivity services will be provided to check the fulfillment of the established KPIs, SLAs, etc.

From an implementation perspective, the development and deployment of this ad-hoc interface will follow the best-practices using REST APIs specifications. The design provides a standard, open, easily accessible interface, and programmatic access to different use cases to be demonstrated. This solution avoids using multiple proprietary APIs for different network domains and devices. The details of this API will be defined later in the project appropriately reported, but the following policy have been established:

- REST-based APIs are accessible through open documentation frameworks, with OpenAPI being the preferred option. The exposed data will follow, and augment, if required, the different data models used by the TSN and Metro controllers.

In conclusion, the proposed solution plays a crucial role in current and future network management and software deployment providing a powerful and flexible tool for managing and configuring networks.

9.2.4.2 Connectivity Manager to Digital Twin

The interface between the CM and the DT enables the computation of the connectivity services deployment over the TIMING infrastructure. Such computation is composed of two phases. In the first one, the CM, which is the entity that receives the service request, uses the data plane information collected from the SDN controllers to compute a set of candidate paths for the service. Then, the CM sends those candidate paths with further information about the status of the network to the DT. In the second phase, the DT computes the expected fulfillment of the requirements or KPIs established for the service. This output is finally sent back to the CM that chooses the most appropriate candidate and starts the configuration. Hence, the preliminary version of this interface between the CM and the DT is expected to have two main operations:

- *The candidate path feasibility request*, which conveys three main groups of information:
  - The service request parameters including the type of the service (e.g., TSN or best effort), the candidate path to evaluate and the maximum expected traffic.
  - Information about the established flows that share some portion of the route with the candidate path. This information will include the type of flow, the route, the amount of conveyed traffic, and the list of KPIs that will help the DT to compute the ones that will achieve the candidate path. This list will be defined during the progress of the project.
  - The graph that represents the portion of the network associated to the candidate and established paths.
- *The estimation of the resulting KPIs* associated to the candidate path and the new KPIs estimated for the existing ones.

10 CONCLUSION AND OUTLOOK

This document contains a preliminary description of the TIMING project elements, where all partners have contributed. The foundations of the project demonstrator are identified and set, by describing the preliminary SW/HW architecture and its components for a deterministic network, that will enable end-to-end reliable TSN services for mixed traffic types in an industrial scenario, by integrating TSN Ethernet and WiFi domains, and all aspects that enable the TSN network control. A more detailed specification of the network components will be found in the forthcoming document D2.1, that will include the Wi-Fi and Ethernet TSN nodes, the TSN controllers, the connectivity manager, scheduling algorithms, TSN models for the digital twin and detailed interfaces.

11 REFERENCES

[802.1Qbv] “IEEE Standard for Local and metropolitan area networks -- Bridges and Bridged Networks - Amendment 25: Enhancements for Scheduled Traffic”, [Online] https://standards.ieee.org/ieee/802.1Qbv/6068/, 2016.

[802.1Qcr] “IEEE Standard for Local and Metropolitan Area Networks -- Bridges and Bridged Networks - Amendment 34: Asynchronous Traffic Shaping”, [Online] https://standards.ieee.org/ieee/802.1Qcr/7420/, 2020.

[8021CM] "IEEE Standard for Local and metropolitan area networks -- Time-Sensitive Networking for Fronthaul," in IEEE Std 802.1CM-2018 , vol., no., pp.1-62, 8 June 2018, doi: 10.1109/IEEESTD.2018.8376066.

[8023BR] "IEEE Standard for Ethernet Amendment 5: Specification and Management Parameters for Interspersing Express Traffic," in IEEE Std 802.3br-2016 (Amendment to IEEE Std 802.3-2015 as amended by IEEE Std 802.3bw-2015, IEEE Std 802.3by-2016, IEEE Std 802.3bq-2016, and IEEE Std 802.3bp-2016), vol., no., pp.1-58, 14 Oct. 2016, doi: 10.1109/IEEESTD.2016.7592835.

[Ada21] T. Adame, M. Carrascosa, and B. Bellalta, “Time-sensitive networking in IEEE 802.11be: On the way to low-latency WiFi 7,” *Sensors*, 2021.

[Be20] A. Bernal *et al.*, “Near Real-Time Estimation of End-to-End Performance in Converged Fixed-Mobile Networks,” Elsevier Computer Communications, vol. 150, pp. 393-404, 2020.

[Cav19] D. Cavalcanti, J. Perez-Ramirez, M. M. Rashid, J. Fang, M. Galeev, and K. B. Stanton, “Extending Accurate Time Distribution and Timeliness Capabilities over the Air to Enable Future Wireless Industrial Automation Systems,” *Proc. IEEE*, vol. 107, no. 6, pp. 1132–1152, Jun. 2019, doi: 10.1109/JPROC.2019.2903414.

[Cle18] A. Clemm, J. Medved, R. Varga, R. Bahadur, N. Ananthakrishnan, X. Liu “A YAND Data Model for Network Topologies”, IETF RFC 8345, 2018.

[Cor19] E. Coronado, S. N. Khan and R. Riggio, "5G-EmPOWER: A Software-Defined Networking Platform for 5G Radio Access Networks," in IEEE Transactions on Network and Service Management, vol. 16, no. 2, pp. 715-728, June 2019, doi: 10.1109/TNSM.2019.2908675

[Dur16] G. Durisi, T. Koch, and P. Popovski, “Toward Massive, Ultrareliable, and Low-Latency Wireless Communication with Short Packets,” *Proc. IEEE*, vol. 104, no. 9, pp. 1711–1726, 2016, doi: 10.1109/JPROC.2016.2537298.

[ENP] “*E-lighthouse Network Solutions*.” [Online] https://e-lighthouse.com/ (accessed February 02, 2023).

[God20] I. Godor *et al.*, “A Look Inside 5G Standards to Support Time Synchronization for Smart Manufacturing,” *IEEE Commun. Stand. Mag.*, vol. 4, no. 3, pp. 14–21, Sep. 2020.

[Guo21] X. Guo *et al.*, "Experimental Demonstration of SDN-enabled Reconfigurable Disaggregated Data Center Infrastructure," *2021 European Conference on Optical*

Communication (ECOC), Bordeaux, France, 2021, pp. 1-4, doi: 10.1109/ECOC52684.2021.9605888

[Hu21] B. Hu and H. Gharavi, "A Hybrid Wired / Wireless Deterministic Network for Smart Grid," no. June, pp. 1–6, 2021.

[Ibr18] M. Ibrahim *et al.*, "Verification: Accuracy evaluation of WiFi fine time measurements on an open platform," *Proc. Annu. Int. Conf. Mob. Comput. Networking, MOBICOM*, pp. 417–427, 2018.

[Jia19] X. Jiang, M. Luvisotto, Z. Pang, and C. Fischione, "Latency Performance of 5G New Radio for Critical Industrial Control Systems," *IEEE Int. Conf. Emerg. Technol. Fact. Autom. ETFA*, vol. 2019-Septe, pp. 1135–1142, 2019.

[May17] Mayoral, A. et al. "*Control Orchestration Protocol: Unified Transport API for Distributed Cloud and Network Orchestration*" Journal of Optical Communications and Networking, (2017). Vol. 9, Issue 2, pp. A216-A222, DOI: 10.1364/JOCN.9.00A216

[Muñ12] O. Muñoz, A. Agustín, J. Vidal, "MCS and Sub-Band Selection for Downlink Interference Coordination in LTE-A Femtocells," 2012 IEEE Vehicular Technology Conference (VTC Fall), Quebec City, Canada 2012.

[ONF] "SDN Architecture", Issue 1.1, ONF TR-521, 2016

[Pag23] A. Pagès, F. Agraz, S. Spadaro, SDN-based band-adaptive quality assurance scheme in support of heterogenous B5G services over sliceable multi-band optical networks, Optical Switching and Networking, vol. 47, February 2023.

[Pav16] P. Pavon Marino, "*Optimization of computer networks: modeling and algorithms: a hands-on approach*", John Wiley & Sons, Inc. (2016)

[Pav18] P. Pavon Marino, M. Garrich, Francisco Javier Moreno-Muro. "*The Role of Open-source Network Optimization Software in the SDN/NFV World*". Optical Fiber Communication Conference (2018), PP. Th1D.1, DOI: 10.1364/OFC.2018.Th1D.1.

[Ra18] D. Rafique, L. Velasco, "Machine Learning for Optical Network Automation: Overview, Architecture and Applications," IEEE/OSA J. Optical Comm. and Netw. (JOCN), vol. 10, pp. D126-D143, 2018.

[Ru18] M. Ruiz, F. Coltraro, L. Velasco, "CURSA-SQ: A Methodology for Service-Centric Traffic Flow Analysis," IEEE/OSA J. Optical Comm. and Netw. (JOCN), vol. 10, pp. 773-784, 2018.

[Sac18] J. Sachs, G. Wikstrom, T. Dudda, R. Baldemair, K. Kittichokechai, "5G Radio Network Design for Ultra-Reliable Low-Latency Communication," *IEEE Netw.*, vol. 32, no. 2, pp. 24–31, 2018.

[Sch21] M. Schungel, S. Dietrich, D. Ginthor, S.-P. Chen, and M. Kuhn, "Heterogeneous Synchronization in Converged Wired and Wireless Time-Sensitive Networks," 2021 17th IEEE International Conference on Factory Communication Systems (WFCS), pp. 67–74, 2021, doi: 10.1109/wfcs46889.2021.9483592.

[Sei20] Ó. Seijo, J. A. López-Fernández, 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 Trans. Ind. Informatics*, 2020.

[Sei22] Ó. Seijo, X. Iturbe and I. Val, "Tackling the Challenges of the Integration of Wired and Wireless TSN With a Technology Proof-of-Concept," in *IEEE Transactions on Industrial Informatics*, vol. 18, no. 10, pp. 7361-7372, Oct. 2022, doi: 10.1109/TII.2021.3131865.

[Son19] Soni, A., Ranga, V. "*API features individualizing of web services: REST and SOAP*". International Journal of Innovative Technology and Exploring Engineering, (2019). 8 (9 Special Issue), 664–671. https://doi.org/10.35940/ijitee.I1107.0789S19

[SWAG] Swagger IO. API development platform. API Development for Everyone

[TAPI] "*Transport API*" [Online] https://wiki.opennetworking.org/display/OTCC/TAPI (accessed February 09, 2023)

[TIP] Telecom Infra Project, "Open Transport SDN Architecture Whitepaper", 2021

[TSNTG] "Time-Sensitive Networking Task Group", [Online] http://www.ieee802.org/1/pages/tsn.html.

[Ve23] L. Velasco, P. Gonzalez, and M. Ruiz, "An Intelligent Optical Telemetry Architecture," accepted in Optical Fiber Communication Conference (OFC), 2023.

[WLAN21] "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 Ame," *IEEE Std 802.11ax-2021 (Amendment to IEEE Std 802.11-2020)*, 2021.

[ZYNQ] Zynq-7000 SoC Data Sheet: Overview. Xilinx. [Online]. Accessed on 07/04/2021. URL: https://www.xilinx.com/support/documentation/data_sheets/ds190-Zynq-7000-Overview.pdf