Journal of
Sensor and Actuator Networks

MDPI

Review

Recent Advances in Time-Sensitive Network Configuration Management: A Literature Review

Boxin Shi [1,†], Xiaodong Tu [1,*,†], Bin Wu [2,*] and Yifei Peng [1,†]

[1] School of Information and Communication Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China; shiboxin@std.uestc.edu.cn (B.S.); pengyifei@std.uestc.edu.cn (Y.P.)
[2] The 34th Research Institute of China Electronics Technology Group Corporation, Guilin 541004, China
* Correspondence: xdtu@uestc.edu.cn (X.T.); wubin_cetc34@163.com (B.W.)
† These authors contributed equally to this work.

Abstract: At present, many network applications are seeking to implement Time-Sensitive Network (TSN) technology, which not only furnishes communication transmission services that are deterministic, low-latency, highly dependable, and have ample bandwidth, but also enables unified configuration management, permitting different network types to function under a single management system. These characteristics enable it to be widely used in many fields such as industrial sensor and actuator networks, in-vehicle networks, data center networks, and edge computing. Nonetheless, TSN's configuration management faces numerous difficulties and challenges related to network deployment, automated operation, and maintenance, as well as real-time and safety assurance, rendering it exceedingly intricate. In recent years, some studies have been conducted on TSN configuration management, encompassing various aspects such as system design, key technologies for configuration management, protocol enhancement, and application development. Nevertheless, there is a dearth of systematic summaries of these studies. Hence, this article aims to provide a comprehensive overview of TSN configuration management. Drawing upon more than 70 relevant publications and the pertinent standards established by the IEEE 802.1 TSN working group, we first introduce the system architecture of TSN configuration management from a macro perspective and then explore specific technical details. Additionally, we demonstrate its application scenarios through practical cases and finally highlight the challenges and future research directions. We aspire to provide a comprehensive reference for peers and new researchers interested in TSN configuration management.

Keywords: Time-Sensitive Network; real-time Ethernet; deterministic network; network configuration management

Citation: Shi, B.; Tu, X.; Wu, B.; Peng, Y. Recent Advances in Time-Sensitive Network Configuration Management: A Literature Review. *J. Sens. Actuator Netw.* **2023**, *12*, 52. https://doi.org/10.3390/jsan12040052

Academic Editor: Guangjie Han

Received: 11 May 2023
Revised: 17 June 2023
Accepted: 30 June 2023
Published: 6 July 2023

1. Introduction

1.1. Background and Related Work

The emergence of advanced technologies, such as automatic driving, industrial internet of things (IIoT), telemedicine, and Augmented Reality/Virtual Reality(AR/VR), has increased the demand for real-time services that require network transmission services with millisecond delays and microsecond jitters. Time-Sensitive Network (TSN), which is an Ethernet extension, offers deterministic low-latency transmission services for layer-2 networks and is therefore gaining widespread attention across various industries. For instance, the IEEE P802.1DG working group is investigating the feasibility and technical aspects of using TSN in vehicle networks, while the IEEE P802.1DP working group is focusing on applying TSN to avionics systems. In the field of industrial automation, the deployment of a large number of sensors and actuators on industrial networks is one of the challenges of TSN network configuration [1].

Currently, the standards for TSN's data plane are mostly mature, enabling TSN devices from different manufacturers to interconnect and interoperate. However, the relevant standards for TSN's control plane are still being enhanced. The TSN configuration management function operates on the TSN control plane and it has a significant impact on multiple aspects, including network deployment, configuration efficiency, routing and scheduling, network convergence, fault recovery, business safety of upper-layer applications, and network security, as shown in Figure 1. An efficient configuration management system plays a crucial role in the operational efficiency, reliability, safety, and robustness of the entire TSN network.

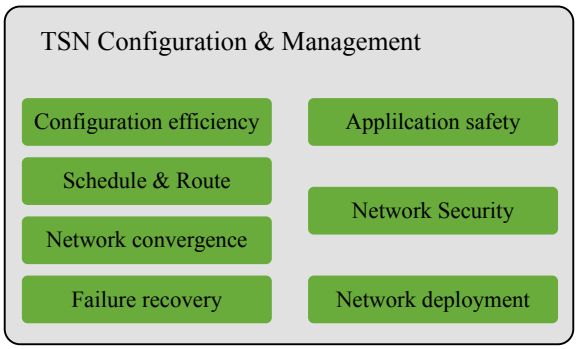


Figure 1. Configuration management has an impact on all aspects of TSN.

IEEE 802.1Qcc [2] introduces three configuration management models and provides a rough specification of network architecture. However, the protocol does not specify any specific network management behavior or protocols, making it impossible to configure and manage TSN devices produced by different manufacturers uniformly through the same system. When the network scale is large or the traffic mode is complex, TSN must provide a fast dynamic automatic configuration function to cope with the QoS requirements of different traffic types and potential network failures, which means it needs to modify the network configuration promptly during network operations without affecting the flow being transmitted in the network.

Therefore, efficient management of the entire TSN network's behavior and resources in a unified, fast, and flexible manner is a critical research topic. This article offers a comprehensive overview of TSN configuration management from several perspectives, including system architecture, key technology research, application scenarios, and future research directions.

1.2. Content and Structure

This article first introduces different TSN configuration management models from a macro perspective, then delves into specific technical details at the micro level, describes application scenarios, and finally discusses future research directions. The main content of this article is as follows:

- **TSN Configuration Management Models:** Various types of networks require different forms of configuration management. For Time-Sensitive Networking (TSN), three distinct models for configuration management have been abstracted to cater to diverse network requirements. These models comprise the fully distributed model, centralized network/distributed user model, and fully centralized model. This academic article provides a comprehensive description of the architectural design and operational workflow of each of these configuration management models, and highlights their individual strengths and limitations.
- **Key Technologies of TSN Configuration Management:** The management of Time-Sensitive Networking (TSN) configuration encompasses a wide array of technical intricacies. The present article concentrates on the areas of clock synchronization management, network topology discovery, configuration management patterns, fault

detection and recovery, as well as reconfiguration. Our analysis explores the critical role that these technologies play in the TSN configuration management system, whilst also identifying the current challenges and obstacles faced in their implementation.

- **Application of TSN Configuration Management in Different Scenarios:** At present, Time-Sensitive Networking (TSN) is a popular technology utilized in in-vehicle and industrial network scenarios. The present article aims to explicate the crucial role of TSN configuration management within these two domains, drawing on specific cases for illustration. To this end, we have cataloged several system architectures frequently deployed in these contexts, with the goal of inspiring relevant researchers and technicians in their pursuit of effective system design and application development.
- **Future Research Directions:** This article delves deeper into the potential research avenues for Time-Sensitive Networking (TSN) configuration management. These areas include the standardization of control planes, real-time dynamic reconfiguration, cross-domain TSN implementation, and wireless TSN. Each of these directions has the potential to contribute significantly to the advancement of TSN configuration management and therefore warrants further investigation and exploration.

The structure of this article is as follows: the first section provides a brief overview of the current state of TSN configuration management, along with the research scope of the article. Section 2 aims to introduce readers to the foundational knowledge of TSN, enabling them to quickly comprehend the basic terminology and related technologies involved in TSN network configuration management. In Section 3, we provide a detailed exposition of the three models of TSN configuration management and review the pertinent research. In Section 4, we scrutinize the relevant technical details pertaining to the configuration management model. Section 5 discusses the application of TSN configuration management in vehicle and industrial scenarios. Section 6 identifies the primary challenges currently faced by TSN configuration management and outlines future research directions. Finally, Section 7 offers a summary of this article.

2. TSN and Related Toolsets

This section provides a brief introduction to TSN itself and related toolsets, which will help readers quickly get started and better understand the subsequent content.

2.1. TSN

The Ethernet Audio/Video Bridging (AVB) task force was established by the IEEE 802.1 working group in 2005 to address the challenge of data synchronization transmission in audio and video networks. Over time, the task force expanded its scope and in 2012 was renamed as the IEEE 802.1 TSN task group, which is focused on researching a wider range of time deterministic networks. A variety of comprehensive literature reviews on TSN are available for interested readers, including references [3–7].

The TSN task group has focused its efforts on four key areas, including time synchronization, end-to-end bounded latency, high reliability, and network management [8], and has developed a range of protocols [9]. We summarize the objectives and contents of the main protocols in the literature [10] as shown in Figure 2, which lists the protocols for the data plane including IEEE 802.1AS [11], IEEE 802.1Qbv [12], IEEE 802.1Qci [13], IEEE 802.1Qbu [14], IEEE 802.1CB [15], IEEE 802.1Qch [16], and IEEE 802.1Qcr [17]. Since the data plane is not the focus of this article, it will not be repeated here. On the control plane, IEEE 802.1Qcc [2] improves the Stream Reservation Protocol (SRP) proposed in IEEE 802.1Qat [18] and proposes three network models to preliminarily define the network form of TSN. IEEE 802.1Qcp [19] introduces the basic YANG model for TSN configuration management and the YANG model related to specific functions is distributed in the corresponding protocol. IEEE 802.1Qca [20] provides explicit path control for TSN flows, with the ability to reserve bandwidth and set redundant routing. IEEE 802.1CB [15] relies on the redundant path provided by IEEE 802.1Qca to carry TSN flows over disjoint paths from sender to receiver in the network. IEEE P802.1Qdd [21] is developing a Resource

Allocation Protocol (RAP) that dynamically reserves network resources for TSN flows in the distributed model, as shown in Section 3.2. IEEE P802.1Qdj, which is under development, will further enhance the CUC and CNC modules introduced in IEEE 802.1Qcc.

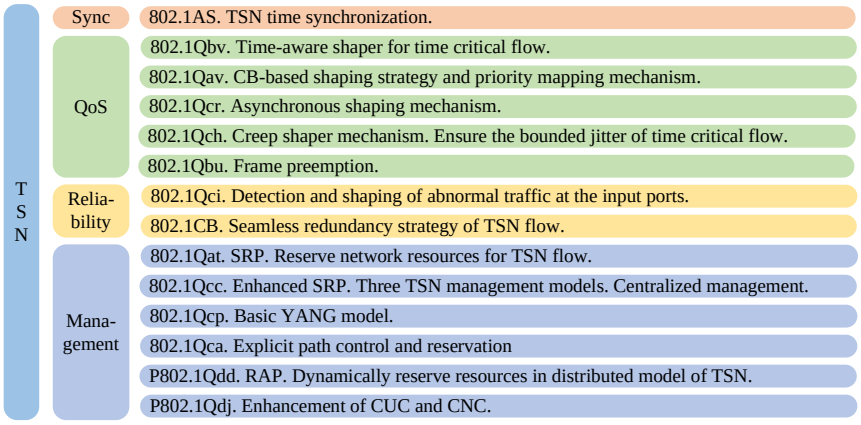


Figure 2. Overview of the TSN protocols [10].

2.2. YANG and NETCONF

In 2002, the Internet Engineering Task Force (IETF) established rigorous requirements for the new generation of network management protocols in the RFC3535 document. These requirements included the provision of programmable APIs for network configuration, support for network devices from various manufacturers and models, and utilization of a consistent and easily understood modeling language, as well as complete error detection and rollback functionalities. Building upon these specifications, the IETF developed the NETCONF protocol in 2006 and subsequently issued RFC6020 [22] in 2010, which established version 1.0 of the YANG language. The IETF also released version 1.1 of the YANG language in 2016, which remains in use today and is detailed in the RFC7950 [23] document.

As a unified modeling language (UML), YANG enables the modeling of configuration data, control data, and status data, thereby facilitating unified configuration management. NETCONF is a network configuration management protocol that leverages the Secure Shell Protocol (SSH) to offer secure and reliable operations such as retrieving, uploading, and distributing configuration data. The YANG model is closely integrated with the NETCONF protocol, and can be conveniently translated into general data formats such as XML and JSON. These data formats can then be transmitted via NETCONF to effect configuration editing, deletion, and copying operations.

As an autonomous protocol standard independent of manufacturers, Time-Sensitive Networking (TSN) requires a unified configuration management paradigm. The combination of the YANG model and NETCONF protocol can fulfill this requirement. IEEE 802.1Qcp [19] integrates the YANG model and NETCONF protocol into TSN configuration management, utilizing YANG as the modeling language for various configuration parameters and employing NETCONF to disseminate configuration information to network devices. This integration significantly enhances the standardization level of TSN, and promotes high compatibility and portability of TSN configuration.

2.3. SDN

Software Defined Networking (SDN) technology is a network management concept that separates the control plane and data plane of a network to achieve centralized network management, improve efficiency and flexibility, and support dynamic modification of network configuration. This architecture is divided into three planes: data, control, and application. The data plane consists of the network infrastructure and forwarding operations are carried out based on forwarding. The control plane, on the other hand, is managed by a centralized network controller, which uses a southbound interface to manage network devices and interfaces with business requirements from the application plane through a

northbound interface. OpenFlow is a widely used protocol for SDN southbound interfaces, which defines "flow table" operations that can control the forwarding rules of network switches. However, there is currently a lack of unified standards for northbound interfaces, resulting in inconsistent interfaces provided by suppliers. Mainstream open source SDN controllers include OpenDaylight, Ryu, ONOS, and Floodlight, among others.

IEEE 802.1Qcc proposes three Time-Sensitive Networking (TSN) configuration management models, among which the fully centralized model bears a high degree of similarity to the aforementioned SDN architecture. Moreover, some SDN controllers' southbound interfaces support the NETCONF protocol, such as OpenDaylight and Ryu. OpenDaylight even directly uses the YANG model to define its own data structure. As a result, many studies directly use common SDN controllers as TSN managers. In particular, introducing SDN controllers in TSN can improve network programmability, and resource management efficiency and flexibility. These benefits are discussed in detail in [24].

2.4. OPC UA

OPC UA (OPC Unified Architecture) is an information modeling method developed by the OPC Foundation, which is platform-agnostic and supports security features such as encryption and verification. This modeling method can be utilized to define complex information. OPC UA comprises two data interaction modes, namely, client–server mode and publish–subscribe mode (PubSub).

The trend of integrating OPC UA with TSN is gaining significant attention. TSN facilitates a deterministic data interaction process, while OPC UA provides a standardized information model for data, which defines the semantics of the data. The OPC UA client–server mode can function as a user-specific protocol for configuring and managing the TSN end system (as stated in [25,26]). Additionally, OPC UA PubSub can function as a data stream for transmission over the TSN network (as stated in [27]). According to reference [28], the combination of OPC UA and TSN is expected to be a solution for industrial 4.0 network data interaction and proposes a solution for industrial TSN network configuration management using OPC UA. Furthermore, D. Bruckner et al. believe that the combination of OPC UA and TSN will become the first and only mature solution for a new generation of industrial communication systems from sensors to the cloud. They have outlined the role of OPC UA TSN in the OSI reference model and conducted research on the application of OPC UA TSN in real-time industrial communication, as cited in [29,30].

3. Research on the Architecture of TSN Configuration Management System

This section focuses on the architecture model of TSN configuration management, and the main work is as follows:

- Provide an overview of the main content of the TSN configuration management model in the IEEE 802.1Qcc [2];
- Summarize distributed and centralized configuration management models and the latest research progress;
- Summarize the workflow of distributed and centralized models;
- Analyze the advantages and disadvantages of distributed and centralized models.

3.1. TSN Configuration Management Model

IEEE 802.1Qcc [2] introduces three TSN configuration management models, namely the fully distributed model, centralized network/distributed user model, and fully centralized model, as shown in Figure 3. The UNI (User/Network Interface) in the figure represents the interface between the end user and the TSN network, used to interact with user requirements and configuration related information, which is modeled in the form of U/NCI (User/Network Configuration Information) and does not rely on specific protocols.

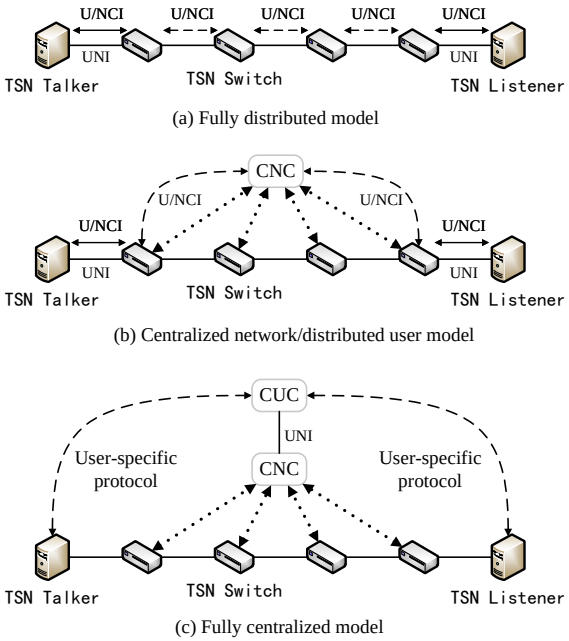


Figure 3. Three models of TSN configuration management.

- **The fully distributed model** does not have any entities for centralized management (Figure 3a). U/NCI hops from the Talker to the Listener along the path given by the spanning tree, and the bridges along the way perform admission control based on U/NCI and their own resource situation. UNI is located between the end user and their direct network bridge. The black solid line arrow indicates that the end user interacts with the direct bridge U/NCI through UNI. The black dashed arrow indicates the transfer of U/NCI between bridges.
- **The centralized network/distributed user model** introduces Centralized Network Configuration (CNC) as the centralized manager of the network bridge on the basis of complete distribution (Figure 3b). The main function of CNC is to discover network topology, acquire the capabilities and resources of the bridge, calculate based on user needs, provide feedback to users, and configure the network. Similar to fully distributed, the UNI of the network centralized/user distributed model is located between the end user and its direct network bridge. End users can directly send U/NCI (black solid line arrow) to the direct bridge through UNI. The difference is that, in the network concentration/user distribution model, the direct bridge acts as an agent for CNC, so U/NCI no longer transmits hop by hop, but directly forwards between the end user's direct bridge and CNC (black dashed arrow). CNC manages the network bridge through some remote management protocol (black dotted arrow), such as SNMP, NETCONF, and RESTCONF.
- **The fully centralized model** further builds on the network centralized/user distributed model by introducing Centralized User Configuration (CUC) as the centralized manager for end users (Figure 3c). It can meet the demand for direct configuration management of terminal devices. CUC discovers terminals through user specific protocols (black dashed arrows), obtains the capabilities and user requirements of terminal devices, and configures them. Compared to the first two models, in the fully centralized model, UNI is located between CNC and CUC. CNC acts as the proxy for the bridge, while CUC acts as the proxy for the terminal device, and U/NCI only interacts directly between the two. CNC also manages the network bridge through some remote management protocol (black dotted arrow).

The literature on the centralized network/distributed user model is limited and it exhibits certain characteristics of the fully centralized model. Therefore, we will henceforth refer to the fully centralized model and the centralized network/distributed user model

collectively as the ***centralized model***, while the fully distributed model will be referred to as the ***distributed model***. Currently, the majority of relevant literature focuses on the centralized model, leaving little room for discussion on the distributed model in the following discourse. As a result, this paper will primarily summarize the centralized model.

3.2. Distributed Model

The TSN distributed model has evolved from numerous previous studies. Initially, the IEEE 802.1AVB task group developed IEEE 802.1Qat [18] based on the IEEE 802.1ak [31]. Its main content was the Stream Reservation Protocol (SRP) protocol, which was used to reserve network resources for specific audio and video streams, and avoid resource competition between audio and video streams and other streams. SRP operates in a publish–subscribe mode and supports a one-to-many mode. In addition, SRP allows modifications to existing reservation situations during runtime to meet changing requirements. IEEE 802.1Qcc [2] has improved the SRP protocol (SRPv1) to support more stream reservations.

The deep binding between the upper and lower layers of SRP protocols is inseparable, which is one of its drawbacks. The Resource Allocation Protocol (RAP) defined in IEEE P802.1Qdd is a dynamic resource reservation protocol for unicast and multicast, used to replace SRP while also providing compatibility with the original SRP protocol [21]. Although both are distributed protocols based on hop-by-hop transmission, RAP is an independent protocol that is architecturally separate from the underlying protocols, better meeting the new characteristics of TSN.

3.2.1. Workflow

At present, there is no relevant literature that fully describes the workflow of distributed models within our research scope. We have analyzed and summarized a relatively reasonable distributed model configuration management process based on the relevant description of IEEE 802.1Qcc [2] (as shown in Figure 4):

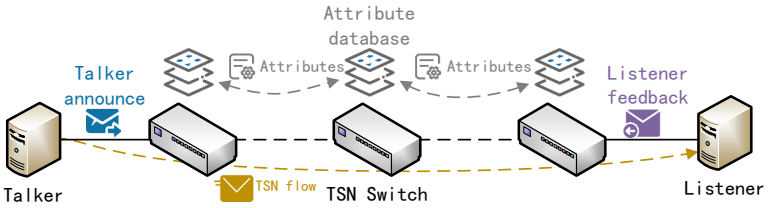


Figure 4. Process of distributed model.

- **Path selection:** All participants in the TSN network (including end users and bridges) establish a tree logical topology through a certain protocol, such as the Spanning Tree Protocol (STP). After the tree logical topology is successfully established, the paths from the Talker to all its Listeners are determined accordingly.
- **Talker announce:** The Talker sends declaration messages to the network using specific protocols such as SRP or RAP. The declaration message records the attribute information of the stream, including stream ID, period, rate, and frame length. This attribute information is exchanged between the databases of the bridge in a hop-by-hop propagation manner.
- **Listener feedback:** After receiving the declaration message from the Talker, the Listener determines whether to receive the stream based on the stream attribute information. If so, it sends a feedback message requesting subscription to the stream. The feedback message returns to the Talker hop by hop along the original path, and each bridge along the way determines whether it can provide services for the flow based on its own resource situation and attaches the judgment result to the feedback message. If it is allowed to provide services for this stream, local devices will also be configured to reserve bandwidth resources for this stream.

- **Sending TSN stream:** If the Talker receives any feedback message from the Listener indicating permission to send, it will send the TSN stream to the network and the stream will be forwarded along the original path of completing bandwidth resource reservation to the Listener subscribing to the stream.
- **Exit mechanism:** During the TSN stream transmission process, both the Talker and Listener can choose to exit the TSN stream they are in. When a stream's Talker or all Listeners exit, all bridges on the stream path will release the bandwidth resources reserved for them.

3.2.2. Research Overview

Reference [32] proposed a method based on SRPv1 to improve the fault-tolerant performance of redundant flow registration using redundancy mechanism. The research of reference [33] suggested that the current fully distributed model overestimates the delay boundary, which will lead to a decrease in the schedulability of the network. So, reference [33] proposed a flow reservation mechanism based on TDMA, which can obtain accurate delay boundaries for each flow. The authors of reference [34] provided a detailed description of the workflow of the RAP protocol in a fully distributed model in a specific network topology. The authors of reference [35] studied the dynamic configuration of TAS (IEEE 802.1Qbv [12]) based on RAP in a ring network topology.

3.3. Centralized Model

The most significant difference between the centralized model and the distributed model is the introduction of a centralized controller, which allows the data transmission and network management to operate on separate planes. Combining the figures, we begin with the basic architecture of the centralized model, introduce its data plane and control plane, and then provide a summary and description of the general workflow of a single-domain centralized model. These references are discussed in detail at the end of this section.

3.3.1. Basic Architecture

We will introduce the advantages and disadvantages of adopting a centralized model in Section 3.4, as well as the issues that may arise from introducing a centralized controller. Overall, the introduction of centralized models in TSN configuration management remains the main research direction at present. We have summarized the overall architecture of the TSN centralized model by referring to relevant research (Section 3.3.3), as shown in Figure 5. Overall, it is divided into two parts: the data plane and the control plane:

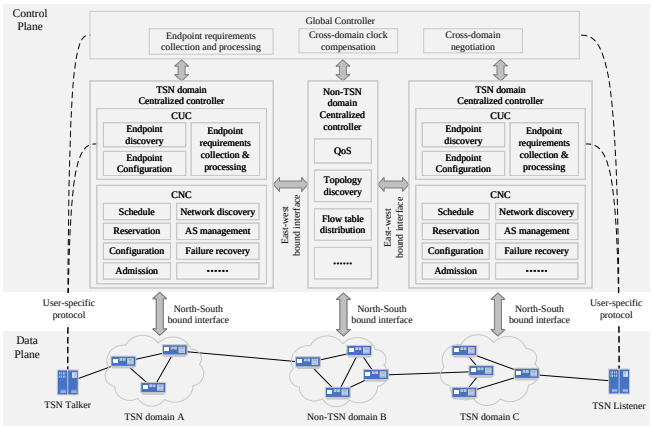


Figure 5. Architecture of centralized model.

- **Data Plane:** The behavior of the TSN domain on the data plane is mainly regulated by many protocols such as IEEE 802.1AS [11] (Timing and Synchronization), IEEE 802.1Qbv [12] (Time-Aware Shaper), IEEE 802.1Qci [13] (Per-Stream Filtering and Policing), IEEE 802.1Qbu [14] (Frame Preemption), IEEE 802.1CB [15] (Frame Repli-

cation and Elimination), IEEE 802.1Qch [16] (Cyclic Queuing and Forwarding), and IEEE 802.1Qcr (Asynchronous Traffic Shaping) [17]. Within the scope of this study, interested readers can refer to relevant standards. In addition, in heterogeneous multi-domain networks, there are network domains formed by non-TSN devices on the data plane. These non-TSN devices may be industrial legacy devices, ordinary IP bridges, or SDN switches that do not possess a range of capabilities specified in the TSN protocol cluster.

- **Control Plane:** The control plane consists of various controllers. The TSN domain centralized controller is the main carrier of the TSN configuration management function, divided into CNC and CUC parts. We have provided a preliminary introduction in Section 3.1. Among them, CUC includes modules such as terminal discovery, requirement collection and processing, and terminal configuration. CNC includes modules such as topology discovery, routing scheduling, admission control, resource reservation, AS management, NETCONF interface, and YANG database. Reference [36] provided a relatively comprehensive description of the functional requirements of centralized controllers, including support for multiple network management protocols, integrated SDN architecture, functional upgrades and extensions, control plane scalability, support for multiple configuration strategies, cross-domain connections, real-time dynamic automatic configuration, and handling uncertainty information, among others. In addition, support for terminal discovery, collection, and processing of end-user requirements, as well as terminal configuration and management, and support for network topology discovery are also included in the functional requirements [37]. Readers can refer to these two articles for detailed information.

In heterogeneous multi-domain networks, there are both TSN domains and non-TSN domains. On the data plane, TSN streams may need to traverse non-TSN domains. A non-TSN domain controller is used to manage devices in the corresponding non-TSN domain, which includes modules such as QoS policy, topology discovery, and flow table distribution. Different domain controllers negotiate through east–west interfaces to jointly manage the configuration of heterogeneous multi-domain networks. In addition, unified negotiation and management of various domain controllers can also be achieved through global controllers. The global controller can directly or indirectly collect and process end-user requirements. The global controller also needs to perform cross-domain clock compensation to ensure consistent clocks across different TSN domains.

3.3.2. Workflow

Single-domain centralized management is the most common TSN configuration management mode and is currently the main research object. Referring to Figure 5, the following is a brief introduction to the general workflow of the TSN single-domain centralized model in conjunction with Figure 6.

- **Topology Discovery:** Centralize the controller (CNC) for network topology discovery and obtain a global view of the network by using a specific protocol such as LLDP.
- **User Request:** The user sends an admission control request to the CUC through the user-specific protocol. The request information is directly forwarded to the CUC through a direct connection bridge, rather than being transmitted hop by hop. CUC aggregates all users' request information and submits it to CNC via UNI.
- **Configure Network:** After the CNC collects all the requests from all users, it calculates the routing and scheduling scheme according to the global view, and then converts the calculation results into profiles and distributes them to bridges using network management protocols such as SNMP, NETCONF, and RESTCONF. At the same time, the CNC provides feedback on the admission control status to the user and configures the user through the CUC. When the above steps are successfully completed, the Talker starts using TSN streams for data transmission.
- **Network Monitoring and Fault Recovery:** During operation, when a network fault occurs, the neighboring nodes of the faulty device actively report the detected abnor-

mal information to the CNC or the CNC detects that some devices are unavailable through the network topology discovery protocol. After collecting the fault information, the controller re-plans the traffic based on the global view, redistributes the configuration to the corresponding devices, and completes fault isolation or recovery.

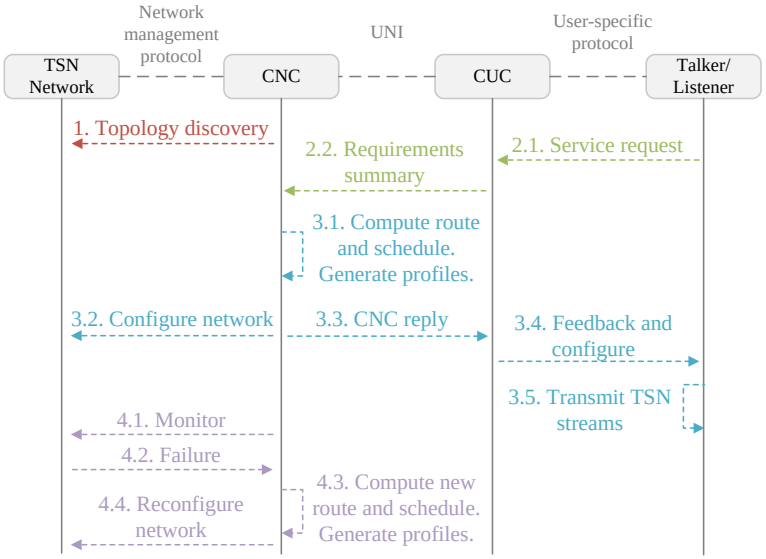


Figure 6. Process of centralized model. The four different colored fonts correspond to the four steps described in the text in order of serial numbers.

3.3.3. Research Overview

Currently, the majority of research conducted on centralized models is centered on a single TSN domain scenario. This approach simplifies the model, making it more intuitive and easier to implement. However, some studies also examine complex multi-domain scenarios. In the subsequent content, we have chosen a selection of relevant literature on both single- and multi-domain scenarios, which are presented in Table 1.

Table 1. References of centralized model.

Single/Multi-Domain	References
Single-Domain	[28,34,36–48]
Multi-Domain	[49–53]

- **Single-domain:** In the research on single-domain TSN, some literature has improved and expanded distributed protocols (RAP and SRP) to achieve faster convergence speed. Another part of the literature adopts a purely centralized approach for configuration management of single-domain TSN. The vast majority of the literature adopts SDN-based solutions, mainly for reasons we have explained in Section 2.3 and we will not elaborate on them here. Time-Sensitive Software Defined Network (TSSDN) is a relatively complete system architecture in SDN-based solutions. Below, we will review the above research:
  - **Improve and extend distributed protocols:** Reference [34] extended the RAP protocol to make it applicable to centralized models. The specific approach is to implement a RAP-related interface in CUC for receiving and processing RAP information from terminals. Ref. [40] used an SDN controller to enhance the SRP protocol. The bridge forwards SRP messages to the SDN controller for processing and then sends them back to the bridge, which then sends them to the next hop. In this scheme, SRP messages are processed in the SDN controller instead of the bridge, but still transmitted hop by hop. Ref. [42] was simplified accordingly. After forwarding SRP messages to the SDN controller, they are

directly forwarded to the destination and no longer sent back to the bridge for hop-by-hop transmission. Ref. [46] more systematically proposed the concept and architecture of Software Defined Flow Reservation (SDFR), which utilized SDN OpenFlow for SRP operations such as flow registration.

  - **Pure centralized method:** Ref. [38] explored the feasibility of introducing SDN in real-time Ethernet, analyzed the advantages and disadvantages of introducing SDN, and proposed a model architecture for network configuration management using SDN. Ref. [37] evaluated the feasibility of SDN OpenFlow protocol as a TSN network management protocol. Ref. [36] summarized the requirements and challenges of TSN configuration management and, based on this, proposed a flexible and scalable TSN centralized configuration management architecture combined with SDN. Ref. [39] used an SDN controller to accelerate the convergence process of clock synchronization. Ref. [41] explored the configuration management problem of wireless TSN networks based on a centralized model. Ref. [28] proposed a solution for industrial TSN configuration management using SDN and OPC UA technologies.
  - **TSSDN:** Reference [45] proposed the concept of TSSDN in 2016 and described its functions, including global attempts, routing, and scheduling. TSSDN is a relatively complete system solution that has been borrowed and adopted by many subsequent research institutes. The authors of [40] provided their understanding of the TSSDN architecture (as shown in Section 5.1.2). However, Ref. [48] further combined the fully centralized model described in IEEE 802.1Qcc-2018 [2] (as shown in Figure 3c), to divide the TSSDN architecture into more detailed module functions, and also introduces a unified control layer. Ref. [47] combed the challenges and future research directions of TSSDN from five aspects of reliability, performance, scalability, security, and interoperability. Ref. [44] studied the problems and challenges related to flow scheduling in TSSDN, and used integer linear programming (ILP) to solve the scheduling problem of new traffic. Ref. [43] proposed a method called $TSN\mu$ The centralized configuration management architecture is similar to TSSDN and compared with TSSDN architecture.

- **Multi-domain:** The main challenge faced by multi-domain TSN is clock synchronization and cross-domain negotiation between different TSN domains. The approach of [49,50] was similar, introducing a higher-level controller as the negotiation management module in the control plane (global controller in Figure 5), which centrally manages TSN domain controllers and indirectly manages multi-domain TSNs, enabling cross-domain clock compensation and traffic scheduling. And in the subsequent work of [49], a higher precision cross-domain clock compensation method was proposed in [52]. Cross-domain TSN flow reservation is also a challenge faced by multi-domain TSNs. Ref. [53] utilized a distributed approach by adding an east–west bound interface to the TSN domain controller. Through this interface, TSN controllers can communicate on the control plane to negotiate resource reservation for cross-domain TSN flows. However, there is no explanation in [53] on how to use east–west interfaces for cross-domain clock synchronization.
- **Heterogeneous network:** In the research on the multi-domain problem of TSN, some have involved the problem of heterogeneous TSN. The so-called heterogeneity refers to the connection between TSN networks and non-TSN networks, and the TSN flow needs to pass through the non-TSN network. This phenomenon is common in industrial legacy networks, where some devices in the network support TSN protocol clusters while others still do not possess TSN characteristics. In addition, sometimes the TSN flow from one industrial field also needs to traverse other networks to reach the other end of the industrial field. This brings a lot of uncertainty to latency and poses new challenges in configuration management. The multi-domain TSN architecture proposed by [49,50] is also used in heterogeneous networks. And Ref. [51] utilizes the idea of distributed models to solve the configuration management prob-

lem of industrial heterogeneous networks by introducing east–west interfaces in SDN controllers.

3.4. Advantages and Disadvantages of Distributed and Centralized Models

Distributed models can be quickly deployed in small networks but their disadvantage is slow network convergence, making it difficult to cope with large networks and complex user needs. Currently, most relevant literature focuses on centralized models. Deterministic Network (DetNet) requires support for network management using a centralized controller [54]. Compared to distributed models, centralized models have many advantages. Reference [38] provided a relatively comprehensive description of this, covering multiple aspects such as centralized control, standardization, global view, redundancy and fault handling, multiple networks sharing a set of devices, dynamic load balancing, fault node detection and isolation, and efficient multicast.

The centralized model also has flaws. Ref. [35] believed that introducing a centralized controller in a small TSN network is not worth the loss. In a network with only one centralized controller, if the centralized controller fails, the entire network will maintain its current operating status at most, unable to add new traffic, and unable to cope with network topology changes or handle network failures. Secondly, there are many computing programs running on the centralized controller and network performance is constrained by the computing performance bottleneck of the centralized controller [55]. Especially in TSN networks, centralized controllers need to sense network topology changes in a timely manner, quickly calculate routing and scheduling plans, and configure devices. And as the network scale expands, the performance requirements for centralized controllers are also increasing. Drawing on the ideas of traditional SDN networks, introducing multiple centralized controllers to serve as hot backup and share computing tasks may become a solution, but it also needs to face the impact of hot backup switching and collaboration between multiple controllers on network operation.

4. Research on Key Technologies of TSN Configuration Management

TSN configuration management has completed preliminary standardization work in terms of system architecture (IEEE 802.1Qcc [2]). We introduced it into distributed models and centralized models in Section 3. Whether it is a distributed model or a centralized model, the workflow of the system can be summarized as follows:

- Firstly, the TSN network needs to perform clock synchronization;
- Next, discover network topology and explore network resources;
- Then, according to user needs, adopt a certain network configuration mode for network configuration;
- Monitor the operation status of the network and promptly identify faults;
- Reconfigure the network for fault recovery.

There are many specific technical details that need to be studied, including clock synchronization management, network topology discovery, network configuration mode, fault detection, and safety and real-time performance of reconfiguration. We have summarized the relevant references, as shown in Table 2. Below, we will provide a detailed introduction to the meaning of the above content and corresponding research.

Table 2. References on key technologies of TSN configuration management.

Content	References
Clock Synchronization Management	[8,39,49,52,56]
Topology Discovery	[57,58]
Network Configuration Patterns	[39,59–63]
Fault Detection and Recovery	[55,61,62]
Safety and Real-time Performance of Reconfiguration	[57,61,62,64,65]

4.1. Clock Synchronization Management

Accurate clock synchronization on all devices is the foundation of TSN networks. IEEE 802.1AS [11] has developed a series of algorithms for clock synchronization, including Master Clock Election (BMCA), link delay measurement, synchronous message forwarding, and so on. These technologies are designed based on a distributed approach and have the problem of slow network convergence. Some studies have improved this by utilizing centralized models. Ref. [39] integrated SDN controllers into CNC based on a centralized model. All network devices also run BMCA, but the difference is that the SDN controller sends the pre-calculated configuration file containing the highest clock priority to the terminal, specifying it as the master clock. In the subsequent research of [39], Ref. [8] added a dynamic reconfiguration function for AS. When the network changes (such as the main clock device disconnects or a new device with a higher priority clock is connected), the SDN controller can automatically recognize and reconfigure the network without manual intervention. Ref. [56] added a timestamp to the TLV field of the LLDP message, which measures the link delay while conducting topology discovery. There is no need to send the link delay measurement message specified in IEEE 802.1AS separately, which helps to reduce network bandwidth overhead.

In Section 3.3.1, we mentioned that clock compensation is required in cross-domain TSN networks to ensure that they all have consistent clocks. Refs. [49,52] adopted a centralized time compensation method for time synchronization in cross-domain TSN, which can reduce the clock error of 1000 km TSN cross-domain network to sub-microseconds.

4.2. Topology Discovery

For distributed models, due to the lack of a centralized manager, the bridge needs to determine its own path for time-sensitive flows to ensure that information from the Talker can be smoothly forwarded to all its Listeners. Section 3.2.1 mentions that a certain protocol (such as Spanning Tree Protocol, STP) can be used to build a tree logical topology and messages from a Talker can reach all its Listeners along the path in the tree. The process of end users using a certain resource reservation protocol (such as SRP or RAP) to request services is to explore whether network resources are sufficient and available.

For centralized models, topology discovery and resource information collection can provide the centralized controller with a global network view, facilitating access control and traffic planning. The Link Layer Discovery Protocol LLDP developed by IEEE 802.1AB [66] adopted the distributed idea. Each network device publishes the locally stored network information to the adjacent nodes. In this interaction process, the topology information of the entire network is spread to each device, and finally all network members reach a consensus on the network topology. After introducing SDN into the centralized model, LLDP is combined with OpenFlow to jointly complete network topology discovery, named OpenFlow Discovery Protocol (OFDP). In addition to the LLDP protocol, the common spanning tree protocol can also serve as a tool for network topology discovery. For example, reference [57] used the Rapid Spanning Tree Protocol (RSTP) to detect network topology changes. By utilizing the remote network management protocol mentioned in Section 3.1, the centralized controller can directly collect network resource information. NETCONF's event notification mechanism can also be used to subscribe to certain events, such as configuration changes, network failures, and changes in data plane performance parameters [58].

4.3. Network Configuration Patterns

Based on the research in Table 2, we have summarized and divided the network configuration mode into manual configuration, static redundancy configuration, and dynamic automatic configuration. The configuration management mode adopted by the network is influenced by multiple factors such as network size, traffic characteristics, fault tolerance requirements, and costs.

4.3.1. Manual Configuration

Configuration is usually considered a one-time action during network initialization [59], where network administrators develop planning plans based on current traffic patterns (such as cycle, source and destination, traffic constraints, etc.), device characteristics (such as bandwidth, device processing delay, queue size, etc.), and link characteristics (such as bit rate, propagation delay, etc.), without modification during network operation. When new streams or devices need to be added and when there is a network failure, the network needs to be shut down for manual reconfiguration or processing. In situations where the network size is small and the traffic mode is relatively simple, manual reconfiguration can be addressed [39].

4.3.2. Static Redundancy Configuration

In order to improve fault tolerance and achieve timely or even seamless recovery in the event of network failures, redundancy mechanisms need to be introduced. Static redundancy configuration can be divided into hardware redundancy configuration and path redundancy configuration. Hardware redundancy configuration runs two or more identical networks simultaneously (including bridges, links, etc.), and the same data is transmitted simultaneously in different networks. This method is the most direct and reliable, but the cost is relatively high [61]. Path redundancy configuration refers to the transmission of data backups on two or more disjoint paths [62], commonly used in ring topology or mesh topology with a certain scale. The frame replication and cancellation (FRER) strategy developed by IEEE 802.1CB [15] adopts this strategy. This strategy saves some hardware costs, but requires sufficient redundancy in the processing power and link bandwidth of the device to cope with the worst-case load [61].

4.3.3. Dynamic Automatic Configuration

Manual configuration is very cumbersome, prone to errors, and is difficult to use to handle complex network and traffic patterns [60]. In the absence of faults, static redundancy configuration will lead to serious waste of network resources [62]. Therefore, dynamic automatic configuration as an alternative solution has become a research focus in TSN network management. The goal of dynamic automatic configuration is to avoid manual interference throughout all stages of network operation:

- During the initialization phase, automatically collect network topology information and distribute the initialization configuration to the network bridge;
- Monitor the network operation status during normal network operation and provide admission control for newly added flows;
- Search for and use redundant paths after a fault occurs, including recalculating routing and scheduling, and issuing new configuration information.

Many studies have proposed different solutions. Ref. [67] proposed a novel approach for automatic configuration based on SDN. Newly added flows can be directly forwarded on the default path, and then migrated to the optimal path by the SDN controller based on traffic characteristics and network resource conditions. Ref. [59] monitored network changes and reconfigured the network accordingly by introducing a proxy entity into the centralized model. Ref. [68] proposed a method to add new streams while the network is running by reconfiguring TAS [12]. Ref. [63] proposed a dynamic path reconfiguration algorithm based on SDN, which can seamlessly migrate TSN streams being transmitted in the network without generating additional latency.

4.4. Fault Detection and Recovery

Many application scenarios of TSN have real-time requirements for fault recovery, such as the AVnu alliance requiring that the fault recovery time (from fault occurrence to fault resolution [65]) of the vehicle network be less than 100 ms [69]. The first step in fault recovery is to discover the fault, which is exactly what this section will explore.

We mentioned in Section 4.2 that network topology can be sensed through network topology discovery related protocols. When there is a device disconnection or link disconnection in the network, this information will gradually spread to the entire network, thus sensing that the network has malfunctioned. For example, reference [57] adopted the Rapid Spanning Tree Protocol (RSTP) to detect changes in network topology. However, using such methods cannot achieve rapid and timely detection of faults. For example, in the LLDP protocol, the Time To Live (TTL) of neighbor information is usually a few seconds or even tens of seconds; coupled with the time for network reconvergence, fault detection will be a lengthy process. For other protocols, such as STP, RSTP, and MSTP, the same issue exists [55,62]. Ref. [57]'s research also indicated that using RSTP to detect topology changes cannot meet the real-time requirements of safety-critical applications. The above AVnu alliance's needs cannot be met either.

Some studies have proposed the use of centralized models to address the aforementioned issues. A. Kostrzewa et al. proposed a method in two consecutive works, refs. [61,62], by installing a monitor module on each device (including switches and terminals) to monitor the status of local devices and their direct links. When a malfunction occurs, the monitor reports the information to the centralized controller through a specific protocol. Ref. [70] proposed an architecture called DRTSN, which utilizes an SDN controller to actively instruct device agents deployed on all network nodes during normal network operation, explaining the actions that need to be performed when certain specific events occur. In this way, the device agent can automatically take measures to handle faults without the intervention of the SDN controller in the event of a fault. As shown in Figure 7, this method greatly reduces the time of fault recovery compared with the conventional fully centralized based fault recovery (CR). Ref. [55] also utilized SDN controllers for fault detection, reducing fault recovery time, and solves the routing and scaling issues of flow affected by faults by introducing source routing technology (encoding the forwarding path into the data packet header to avoid maintaining the forwarding table in the bridge).

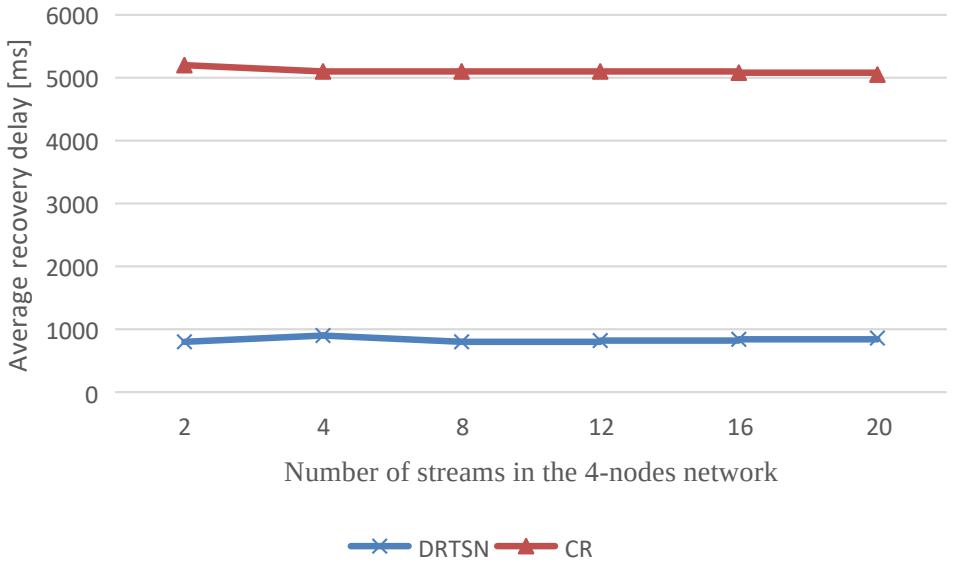


Figure 7. In a four-node network, the fault recovery time corresponding to different numbers of flows [70].

4.5. Safety and Real-Time Performance of Reconfiguration

The reconfiguration process may have an impact on the safety of running services in the TSN network, resulting in transmission timeout and even packet loss. Some TSN application scenarios require high execution efficiency in the reconfiguration process, which can cause the network to re-converge in a short period of time. Below, we will discuss the performance of these two aspects of reconfiguration.

4.5.1. Safety of Reconfiguration

The reconfiguration process may bring safety issues, as shown in Table 3. A. Kostrzewa et al.'s research in [61,62] and K. Weijiang et al.'s research in [64] used network pattern transformation to solve this problem. Each network mode corresponds to a set of network configurations, a set of supported flows, and a series of behaviors of the centralized controller. After the reconfiguration process caused by a fault is triggered, the affected parts of the network enter transition mode. In this mode, the affected devices perform simplified network functions, while the centralized controller reconfigures the affected devices in a safe order. If the reconfiguration is successful, the network will return to normal mode, otherwise it will enter fault mode.

Table 3. Safety issues during reconfiguration process [61].

Issues	Descriptions
Order	Some nodes complete reconfiguration before others and the network undergoes a series of intermediate states. When upstream nodes start contracting while downstream nodes are still busy configuring, it may lead to frame loss.
Buffer	After reconfiguration, the queue is adjusted, emptied, or blocked, resulting in frame loss.
Priority inversion	After reconfiguration, the originally high-priority flow has changed to a low-priority flow, resulting in an increase in the latency of the flow and affecting the original business.
Forwarding incorrectly	After reconfiguration, the forwarding rules of the bridge change, which may result in packets received before reconfiguration being discarded.

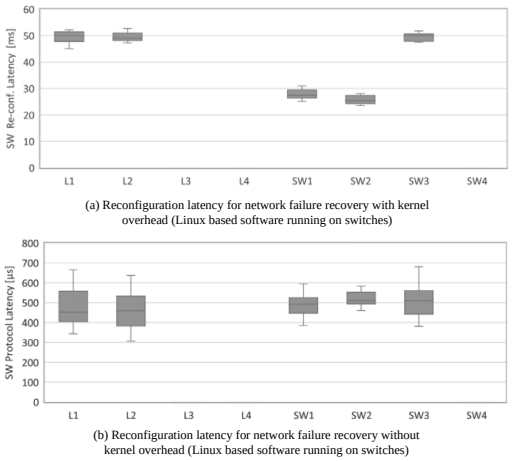
4.5.2. Real-Time Performance of Reconfiguration

We introduced the definition of fault recovery time in Section 4.4, which mainly includes the time required to discover faults and the time required to resolve faults. Section 4.4 has studied the former, while the latter is the real-time issue of reconfiguration to be explored in this section.

Many studies on centralized models have used SDN controllers and OpenFlow as the network configuration management protocol. Ref. [65] demonstrated through formal analysis that the worst-case configuration delay for TSN using SDN is much less than 50 ms, which can meet the real-time reconfiguration requirements of TSN. In the study of [65], OpenFlow is based on UDP (instead of the commonly used TCP) for transmission, eliminating the latency caused by TCP handshake and flow control mechanisms. Ref. [57] found through simulation that using BE streaming to transmit configuration management messages cannot meet real-time requirements. Therefore, reconfiguring related messages should be sent with the highest priority, which helps to reduce latency. In addition, using static routing for configuration management messages can also enhance real-time performance. Ref. [61] tested the reconfiguration delay based on SDN in a typical car network shown in Section 5.1.2. The results are shown in Figure 8, where (Figure 8a) represents the reconfiguration delay caused by using a software protocol stack based on Linux kernel and (Figure 8b) represents pure protocol overhead, excluding the overhead of Linux kernel compared to (Figure 8a). The test results indicate that the majority of the overhead (95%) comes from the processing time of the Linux kernel when reconfiguring the network stack (such as routing tables) but, even with the software implementation of the configuration protocol, the total recovery time in the worst-case scenario is within 50 ms.

Although the above research has proven that using SDN can meet the real-time requirements of TSN reconfiguration, ref. [37]'s research indicated that OpenFlow itself can only achieve partial TSN configuration management functions and requires collaboration with other extensions or accompanying protocols. And using the YANG model meets the protocol requirements, while OpenFlow does not support the YANG model. The NETCONF protocol, which is closely integrated with the YANG model, is the optimal choice for the southbound interface (Section 2.2). However, the NETCONF protocol was not designed with real-time requirements in mind. Ref. [70]'s research indicated that

the SSH and NETCONF session establishment operations of the NETCONF protocol are very time-consuming, and this has been optimized to establish SSH and NETCONF connections in advance between the central controller and the faulty node, reducing the entire fault recovery time from 823 ms to 254 ms. However, this still cannot meet the 100 ms fault recovery delay required by the AVnu alliance. Therefore, improving the real-time performance of NETCONF protocol is still a problem worth further in-depth research.



(a) Reconfiguration latency for network failure recovery with kernel overhead (Linux based software running on switches)

(b) Reconfiguration latency for network failure recovery without kernel overhead (Linux based software running on switches)

Figure 8. Reconfiguration delay of in-vehicle network based on SDN [61].

5. Application Scenarios of TSN Configuration Management

Many studies on TSN configuration management have combined specific application scenarios, including medical [71], data center [72], fog computing [73,74], and other fields. In-vehicle network and industrial network are currently the main application directions, with many related studies. We will introduce these two fields in the following content, combined with the specific references in Table 4.

Table 4. References on TSN configuration management in in-vehicle and industrial scenarios.

Scenario	Reference
In-vehicle	[39,40,42,61,62,75]
Industry	[28,51,73,76–79]

5.1. In-Vehicle Scenario

With the continuous development of Advanced Driving Assistance System (ADAS) and Internet of Vehicles (IoV), the intelligence and interconnectivity of vehicles are constantly improving, and the devices and traffic in the vehicle network are also gradually increasing. TSN can be used as the backbone network of the on-board network, connecting various modules (including the power system, body sensors, ADAS, entertainment, etc.) together.

5.1.1. Characteristics and Requirements Analysis of In-Vehicle Scenario

How to effectively manage increasingly complex in-vehicle TSN networks is an important research topic and the IEEE P802.1DG working group is currently committed to solving this problem. The characteristics of car networks include:

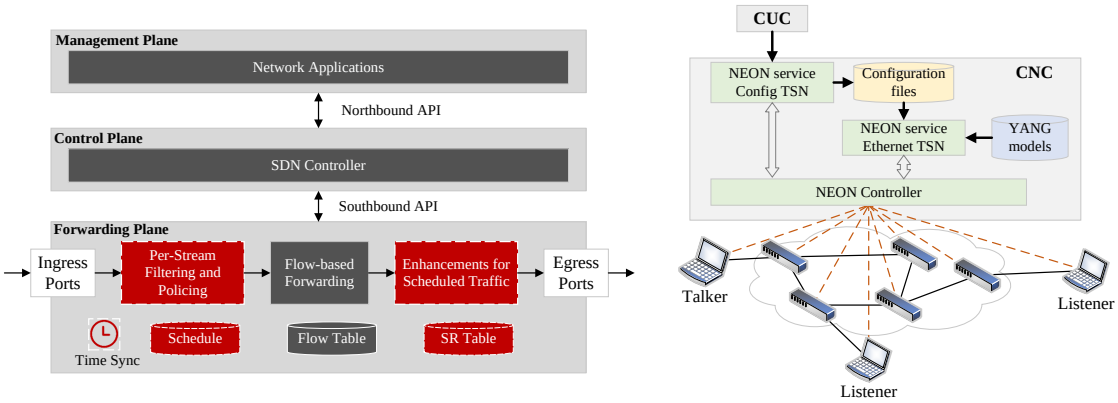
- Network size and relatively limited number of streams to be supported [64].
- Most of the traffic is already set before leaving the factory [32].
- The automotive control network requires a very fast startup time [32].
- Some businesses require dynamic configuration.

For the first three characteristics, it is achieved by directly loading configuration information from non-volatile memory into network devices after the car starts. This helps to minimize configuration, shorten startup time, and provide a secure and stable

communication environment for control data. In addition, there is almost no need for additional management protocols to support configuration. However, with the increase in Internet of Vehicles applications, some businesses may be offloaded to edge nodes, which increases the demand for dynamic configuration.

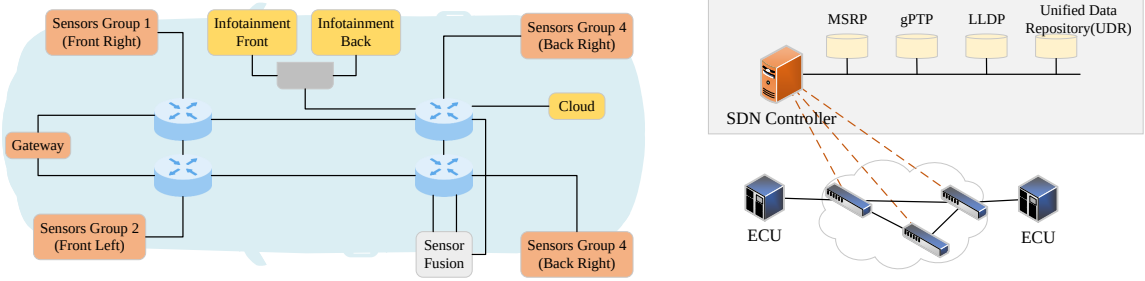
5.1.2. Case Study of In-Vehicle Scenario

(1) Expanding OpenFlow to meet TSN configuration requirements: Reference [40] suggested that the topology of the vehicle network is relatively stable, and the SDN controller can effectively determine routing to prevent link overload. It can also calculate and verify link latency at runtime. In addition, the author extended the OpenFlow configuration management table entry, as shown in Figure 9a. The SDN controller can configure the scheduling table, SR table, and flow based forwarding table of the TSN switch through OpenFlow, verifying the feasibility of TSSDN (Section 3.3.3).



(a) TSSDN switch architecture.

(b) NEON integration in IEEE 802.1Qcc fully centralized model.



(c) An exemplary Ethernet-based network for autonomous vehicles with ring topology assuring redundancy.

(d) Simplified MSRP over SDN Architecture.

Figure 9. Cases of TSN configuration management in vehicle scenario.

(2) Profile and parameter mapping: Ref. [39] proposed the use of SDN for configuration management of vehicle networks to cope with dynamic changes in business and failures. As shown in Figure 9b, the Config TSN component is used to generate appropriate configuration files for each device in the network and match these configuration files with the UUIDs of each device. The Ethernet TSN component is used to convert configuration files into YANG models for each device. Finally, the NEON SDN controller is called to complete the configuration distribution operation.

(3) Reconfiguration: A. Kostrzewa et al. focused their research on reconfiguring in vehicle networks [61,62], believing that reconfiguration operations must be carried out in the correct order to prevent deadlocks or livelocks during the reconfiguration phase (for example, incorrect order of switch reconfiguration may prevent access to certain paths or even to certain parts of the network). On this basis, a centralized car network reconfiguration architecture based on NMU was proposed, as shown in Figure 9c. The

NMU unit is essentially an SDN controller used to sense network failures and maintain network status information. In addition, the network reconfiguration process based on NMU was elaborated in detail and deployment suggestions for NMU were provided.

(4) Using SDN to enhance SRP protocol: Reference [75] studied the possibility of using Multiple Stream Registration Protocol (MSRP) [18] for resource reservation in vehicle networks and focused on the configuration management problem in scenarios of inconsistent link bandwidth. A TSN profile for vehicle network configuration was provided. Reference [42] suggested that the original MSRP protocol is distributed and requires support from Multiple VLAN Registration Protocol (MVRP) [18] and Multiple MAC Registration Protocol (MMRP) [18], which increases network complexity. Therefore, ref. [42] proposed a simplified version of SRP protocol based on SDN controllers for in-vehicle network. As shown in Figure 9d, LLDP protocol is deployed in the SDN controller to sense network status. All dynamic services are registered by the SDN controller uniformly, and the SDN controller converts the registration results into configuration table items and distributes them to various switches.

5.2. Industrial Scenario

Traditional industrial communication mainly relies on fieldbus and industrial Ethernet (such as PROFINET and EtherCAT), which are relatively closed, independent, and incompatible with each other. TSN provides high reliability and low latency traffic scheduling capabilities required for industrial scenarios, and is based on a standard Ethernet architecture, providing a clear and feasible solution for industrial interconnection.

5.2.1. Characteristics and Requirements Analysis of Industrial Scenario

The IEC/IEEE 60802 working group jointly established by IEC and IEEE is promoting the application of TSN in the interconnection and interworking of industrial equipment. In industry network:

- The network needs to have seamless reconfiguration capability and support plug-and-play;
- Generally, it is a heterogeneous network, where there are many legacy devices and the network scale is relatively large;
- There are challenges in adapting to industrial Ethernet applications with different QoS requirements and transmitting BE traffic on shared networks [76].

Considering the above characteristics, industrial networks are currently paying special attention to three technologies: TSN, SDN, and OPC UA. At present, the research hotspots of TSN configuration management in industrial scenarios mainly include system architecture, legacy device agents, configuration management side protocols (NETCONF, OpenFlow), configuration parameter mapping, application layer protocol mapping, scheduler research, and dynamic automatic configuration, etc.

5.2.2. Case Study of Industrial Scenario

(1) System architecture: Reference [51] studied the TSN configuration management architecture for heterogeneous industrial networks, as shown in Figure 10a. Overall, a fully centralized model was adopted. The architecture is divided into four network domains: machine, workshop, factory, and internet. All the bridges displayed in this architecture are TSN bridges. All TSN bridges in each network domain are configured and managed by a local SDN controller (CNC) through a southbound interface, while CNCs in different network domains are connected through east–west interfaces. In addition to CUC and CNC (Section 3.1), this study also introduces SDN agents in the machine domain, consisting of PN bridges (SDN bridges with PROFINET gateway capabilities), for integrating industrial legacy devices into the TSN network. Industrial legacy equipment communicates with PLC integrated with CNC through PN (PROFINET) in the TSN network. Sensors and cameras on industrial fields use OPC UA to transmit information to each other or to applications on the internet cloud.

(2) Embedded OPC UA: With the continuous improvement of processor and memory performance in industrial field devices, embedding OPC UA server functions into industrial field devices has gradually become a trend. This method can directly provide data access services to external OPC UA clients but it also has drawbacks: if there are multiple sets of OPC UA servers and communication requirements between clients, it requires multiple manual settings of the connection between external OPC UA clients and embedded OPC UA servers, which will bring a huge workload and result in a mesh connection between the clients and servers in the system. To address this issue, reference [73] proposed an aggregation architecture that can connect OPC UA servers within the on-field industrial control subsystem. Based on this, reference [77] studied the aggregated OPC UA server architecture for on-field TSN network control in the factory area.

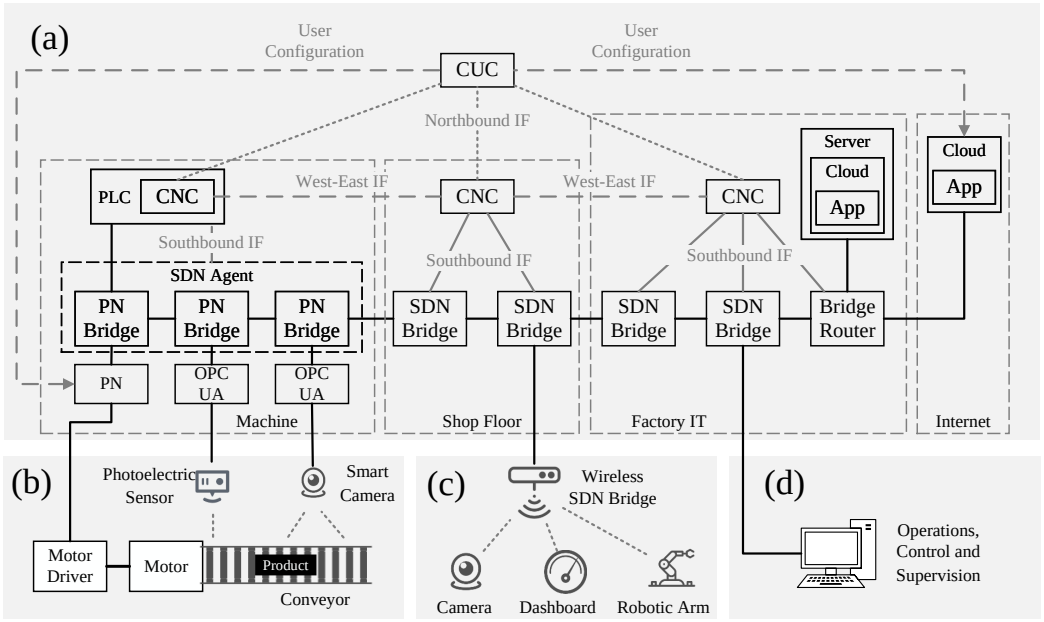


Figure 10. Cases of TSN configuration management in industrial scenario. Subfigure (**a**) portrays the TSN configuration management architecture for heterogeneous industrial networks. Subfigure (**b**) shows a quality inspection system. Subfigure (**c**) shows a case of industrial wireless TSN. And subfigure (**d**) represents a module for operating, controlling and monitoring.

(3) Industrial field equipment: Reference [28] more specifically studied an application case of TSN configuration management in industrial scenarios. Figure 10b shows a quality inspection system. As the photoelectric sensor of TSN Talker, the product position information mapping on the conveyor belt is mapped to the OPC UA stream and then transmitted to the monitor as the TSN Listener through the TSN network. The motor of the conveyor belt is connected to the TSN network through the PROFINET gateway. Reference [78] studied the feasibility of wireless TSN industrial networks and built a prototype system as shown in Figure 10c. Industrial devices, including cameras, instrument panels, and robotic arms, were wirelessly connected to the TSN network, and the behavior of all industrial devices was managed by the upper control plane. Reference [79] also proposed a similar wireless industrial TSN network, where all industrial TSN devices, whether wired or wireless, are managed by the same set of control plane devices. In addition, Ref. [79] also designed modules specifically for operating, controlling, and monitoring the entire industrial TSN network, as shown in Figure 10d.

6. Research Directions and Challenges

6.1. Standardization of Control Plane

We mentioned in Section 1.1 that the standardization of TSN control plane needs to be improved, which can be started from the following aspects:

- **CUC and CNC:** IEEE 802.1Qcc [2] introduced CNC and CUC as centralized control units, but did not clearly define the specific functions of CNC and CUC, nor how the communication interface (UNI) between them interacts with information. The IEEE P802.1Qdj being developed will address this issue, further clarify the functions of CNC and CUC, and stipulate that the two communicate using protocols based on the YANG model (such as RESTCONF).
- **Configuration management interface:** The TSN standard does not unify the interface protocol between CUC and end users, as well as the interface protocol between CNC and the network bridge. Ref. [28] proposed using the client server mode of OPC UA as the interface between network controllers and end users in industrial scenarios, which may be a potential solution. In Section 4.5.2, we discussed the interface protocol between CNC and the bridge, namely the TSN remote configuration management protocol. The improvement of standardization and real-time performance in this area is still a problem worth further in-depth research.
- **YANG model:** We mentioned in Section 2.2 that the YANG model plays a role in promoting unified management in TSN. However, the current standard YANG model still needs to be improved. In addition, various device manufacturers will introduce custom YANG models when improving and extending standard protocols, while these private YANG models are not publicly disclosed. Therefore, there are currently many difficulties in implementing global unified configuration management in a TSN network composed of products from different manufacturers.

6.2. Real-Time Dynamic Reconfiguration

In Section 4.3, we described the importance of dynamic automatic configuration in centralized configuration management but there are still some important issues that need to be addressed:

- **Process specification:** Dynamic automatic configuration itself is a process completed by the network itself, but at present there is no relevant standard [59] for it and there is no unified specification for its complete process, which is not conducive to the formation of an interoperable, vendor-independent TSN configuration management model.
- **Real time:** In Sections 4.4 and 4.5, we mentioned that fault recovery requires reconfiguration of the network, which includes the time required to discover and resolve faults. Reference [57] demonstrated that using spanning tree protocols to detect topology changes is time-consuming and cannot meet real-time requirements. The research in reference [70] indicated that the NETCONF protocol lacks real-time performance. Improving the real-time performance of dynamic reconfiguration is a topic that involves the design of real-time operating systems and network protocols, and further research is needed.

6.3. Cross-Domain TSN

We introduced the literature related to cross-domain TSN in Section 3.3, but these articles lack research on and description of many specific technical details:

- **East–west interface:** In the absence of a global controller (i.e., a distributed control plane), domain controllers need to use an east–west interface for communication, but there is currently no research on or evaluation of specific east–west interface protocols in relevant literature. In addition, how to use east–west interfaces for cross-domain negotiation and clock compensation between domain controllers is also a challenge, especially when there are a large number of domain controllers; the distributed nature can bring significant complexity to the design of east–west interface protocols.
- **Global controller:** When there is a global controller in the network (i.e., the control plane is centralized), similar problems can also be encountered:
  - There is currently a lack of research on the interface between the global controller and the domain controller;

- The functional division between the global controller and the domain controller is currently unclear, such as which controller should be responsible for collecting and processing end-user requirements;
- The collaboration mechanism between the global controller and the domain controller needs to be explored, such as how the two share information (including user terminal information, network topology information, etc.), how to negotiate management, and how to compensate the network clock.

Overall, on the control plane of cross-domain TSN, whether introducing east–west interfaces or introducing a higher-level global controller, it will bring many problems and challenges worth studying.

6.4. Wireless TSN

The Avnu Alliance has released a white paper on wireless TSN [80], outlining a roadmap for integrating wireless technology into industrial TSN systems. The first challenge to overcome in the roadmap is centralized model based configuration management for wireless TSN devices. Intel Lab is currently committed to expanding IEEE 802.11 (Wi-Fi) into the field of wireless TSN [81]. They outlined the potential applications, requirements, and challenges of expanding TSN features on wireless networks in [79], and have built an industrial robot test bench that supports wireless TSN [78]. TSN wireless devices are configured with the same CUC and CNC as wired devices to achieve seamless expansion from wireless domain to wired domain. At present, the following work needs to be considered:

- **Introducing More Wireless Technologies:** The aforementioned Intel laboratory research used IEEE 802.11 (Wi-Fi) and reference [41] configured terminal sensors by integrating DECT ULE wireless technology in CUC. Expanding the TSN configuration management model to other wireless networks requires more practical validation, such as cellular network technology and common wireless technologies in the industrial field.
- **TSN Distributed Configuration Management Model in Wireless Domain:** The above research is focused on centralized wireless TSN and currently there is no attempt to extend the TSN distributed configuration management model (Section 3.2) to wireless domains.
- **Configuration Management Challenges Brought by Wireless Networks:** Currently, there is no literature considering using configuration management entities to address interference issues (such as network overlap) in wireless network planning and deployment. In addition, due to interference, distance, and other factors, the wireless channel capacity is unstable, so it is particularly important to use the TSN configuration management entity to monitor and manage the wireless channel.

7. Conclusions

As a critical technology for the development of Time-Sensitive Networking, TSN configuration management currently faces many difficulties and challenges. Although TSN's functionality in the data plane has been increasingly improved, much work still needs to be done in the standardization of the control plane. This article provides a comprehensive summary of system architecture and various key technologies related to Time-Sensitive Network configuration management. It also introduces the application cases of TSN configuration management in two popular fields, namely vehicular networks and industrial networks. Finally, this article analyzes the challenges faced by TSN configuration management and outlines the next research directions. It is hoped that this literature review in this field can provide references and assistance to relevant researchers.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Varis, P.; Leyrer, T. *Time-Sensitive Networking for Industrial Automation*; Texas Instruments: Dallas, TX, USA, 2020.
2. *IEEE Std 802.1Qcc-2018*; IEEE Standard for Local and Metropolitan Area Networks—Bridges and Bridged Networks Amendment 31: Stream Reservation Protocol (SRP) Enhancements and Performance Improvements. IEEE: Piscataway, NJ, USA, 31 October 2018.
3. Messenger, J.L. Time-Sensitive Networking: An Introduction. *IEEE Commun. Stand. Mag.* **2018**, *2*, 29–33. [CrossRef]
4. Finn, N. Introduction to Time-Sensitive Networking. *IEEE Commun. Stand. Mag.* **2018**, *2*, 22–28. [CrossRef]
5. Cai, Y.; Yao, Z.; Li, T. A Survey on Time-Sensitive Networking: Standards and State-of-the-Art. *Chin. J. Comput.* **2021**, *44*, 1378–1397.
6. Seol, Y.; Hyeon, D.; Min, J.; Kim, M.; Paek, J. Timely Survey of Time-Sensitive Networking: Past and Future Directions. *IEEE Access* **2021**, *9*, 142506–142527. [CrossRef]
7. Deng, L.; Xie, G.; Liu, H.; Han, Y.; Li, R.; Li, K. A Survey of Real-Time Ethernet Modeling and Design Methodologies: From AVB to TSN. *ACM Comput. Surv.* **2022**, *55*, 1–36. [CrossRef]
8. Thi, M.; Said, S.B.H.; Boc, M. SDN-Based Management Solution for Time Synchronization in TSN Networks. In Proceedings of the 25th IEEE International Conference on Emerging Technologies and Factory Automation, ETFA 2020, Vienna, Austria, 8–11 September 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 361–368. [CrossRef]
9. Farkas, J.; Bello, L.L.; Gunther, C. Time-Sensitive Networking Standards. *IEEE Commun. Stand. Mag.* **2018**, *2*, 20–21. [CrossRef]
10. Peng, Y.; Shi, B.; Jiang, T.; Tu, X.; Xu, D.; Hua, K. A Survey on In-vehicle Time Sensitive Networking. *IEEE Internet Things J.* **2023**, *Early access*. [CrossRef]
11. *IEEE Std 802.1AS-2020*; IEEE Standard for Local and Metropolitan Area Networks—Timing and Synchronization for Time-Sensitive Applications. IEEE: Piscataway, NJ, USA, 19 June 2018.
12. *IEEE Std 802.1Qbv-2015*; IEEE Standard for Local and Metropolitan Area Networks—Bridges and Bridged Networks Amendment 25: Enhancements for Scheduled Traffic. IEEE: Piscataway, NJ, USA, 18 March 2016.
13. *IEEE Std 802.1Qci-2017*; IEEE Standard for Local and Metropolitan Area Networks—Bridges and Bridged Networks–Amendment 28: Per-Stream Filtering and Policing. IEEE: Piscataway, NJ, USA, 28 September 2017.
14. *IEEE Std 802.1Qbu-2016*; IEEE Standard for Local and Metropolitan Area Networks—Bridges and Bridged Networks—Amendment 26: Frame Preemption. IEEE: Piscataway, NJ, USA, 30 August 2016.
15. *IEEE Std 802.1CB-2017*; IEEE Standard for Local and Metropolitan Area Networks—Frame Replication and Elimination for Reliability. IEEE: Piscataway, NJ, USA, 27 October 2017.
16. *IEEE Std 802.1Qch 2017*; IEEE Standard for Local and Metropolitan Area Networks—Bridges and Bridged Networks—Amendment 29: Cyclic Queuing and Forwarding. IEEE: Piscataway, NJ, USA, 28 June 2017.
17. *IEEE Std 802.1Qcr 2020*; IEEE Standard for Local and Metropolitan Area Networks—Bridges and Bridged Networks Amendment 34: Asynchronous Traffic Shaping. IEEE: Piscataway, NJ, USA, 6 November 2020.
18. *IEEE Std 802.1Qat-2010*; IEEE Standard for Local and Metropolitan Area Networks—Virtual Bridged Local Area Networks Amendment 14: Stream Reservation Protocol(SRP). IEEE: Piscataway, NJ, USA, 30 September 2010.
19. *IEEE Std 802.1Qcp-2018*; IEEE Standard for Local and Metropolitan Area Networks—Bridges and Bridged Networks–Amendment 30: YANG Data Model. IEEE: Piscataway, NJ, USA, 14 June 2018.
20. *IEEE Std 802.1Qca 2015*; IEEE Standard for Local and Metropolitan Area Networks—Bridges and Bridged Networks Amendment 24: Path Control and Reservation. IEEE: Piscataway, NJ, USA, 3 September 2015.
21. Feng, C. *Resource Allocation Protocol (RAP) Based on LRP for Distributed Configuration of Time-Sensitive Streams*; IEEE: Piscataway, NJ, USA, 2017.
22. *RFC6020*; YANG—A Data Modeling Language for the Network Configuration Protocol (NETCONF). Bjorklund Companies: Cambridge, MA, USA, October 2010.
23. *RFC7950*; The YANG 1.1 Data Modeling Language. Bjorklund Companies: Cambridge, MA, USA, August 2016.
24. Silva, L.; Pedreiras, P.; Fonseca, P.; Almeida, L. On the adequacy of SDN and TSN for Industry 4.0. In Proceedings of the 2019 IEEE 22nd International Symposium on Real-Time Distributed Computing (ISORC), Valencia, Spain, 7–9 May 2019; pp. 43–51. [CrossRef]
25. Tian, S.; Hu, Y. The Role of OPC UA TSN in IT and OT Convergence. In Proceedings of the 2019 Chinese Automation Congress (CAC), Hangzhou, China, 22–24 November 2019; pp. 2272–2276. [CrossRef]
26. Zhou, Z.; Shou, G. An Efficient Configuration Scheme of OPC UA TSN in Industrial Internet. In Proceedings of the 2019 Chinese Automation Congress (CAC), Hangzhou, China, 22–24 November 2019; pp. 1548–1551. [CrossRef]
27. Arestova, A.; Martin, M.; Hielscher, K.S.; German, R. A Service-Oriented Real-Time Communication Scheme for AUTOSAR Adaptive Using OPC UA and Time-Sensitive Networking. *Sensors* **2021**, *21*, 2337. [CrossRef]
28. Kobzan, T.; Blöcher, I.; Hendel, M.; Althoff, S.; Gerhard, A.; Schriegel, S.; Jasperneite, J. Configuration Solution for TSN-based Industrial Networks utilizing SDN and OPC UA. In Proceedings of the 2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), Vienna, Austria, 8–11 September 2020; Volume 1, pp. 1629–1636. [CrossRef]
29. Bruckner, D.; Blair, R.; Stanica, M.P.; Ademaj, A.; Skeffington, W.; Kutscher, D.; Schriegel, S.; Wilmes, R.; Wachswender, K.; Leurs, L.; et al. *OPC UA TSN A New Solution for Industrial Communication*; B&R: Eggelsberg, Austria, 2018.

30. Bruckner, D.; Stanica, M.; Blair, R.; Schriegel, S.; Kehrer, S.; Seewald, M.G.; Sauter, T. An Introduction to OPC UA TSN for Industrial Communication Systems. *Proc. IEEE* **2019**, *107*, 1121–1131. [CrossRef]
31. *IEEE Std 802.1ak-2007*; IEEE Standard for Local and Metropolitan Area Networks—Virtual Bridged Local Area Networks—Amendment 07: Multiple Registration Protocol. IEEE: Piscataway, NJ, USA, 22 March 2007.
32. Kleineberg, O.; Fröhlich, P.; Heffernan, D. Fault-tolerant Audio and Video Bridging (AVB) Ethernet: A novel method for redundant stream registration configuration. In Proceedings of the Proceedings of 2012 IEEE 17th International Conference on Emerging Technologies & Factory Automation (ETFA 2012), Krakow, Poland, 17–21 September 2012; pp. 1–8. [CrossRef]
33. Chuang, C.C.; Shih, Y.Y.; Chen, J.C.; Pang, A.C. Time-Aware Stream Reservation for Distributed TSN. In Proceedings of the 2021 22nd Asia-Pacific Network Operations and Management Symposium (APNOMS), Tainan, Taiwan, 8–10 September 2021; pp. 190–195. [CrossRef]
34. Osswald, L.; Lindner, S.; Wüsteney, L.; Menth, M. RAP Extensions for the Hybrid Configuration Model. In Proceedings of the 2021 26th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), Vasteras, Sweden, 7–10 September 2021; pp. 1–8. [CrossRef]
35. Nasrallah, A.; Balasubramanian, V.; Thyagaturu, A.; Reisslein, M.; ElBakoury, H. Reconfiguration Algorithms for High Precision Communications in Time Sensitive Networks. In Proceedings of the 2019 IEEE Globecom Workshops (GC Wkshps), Waikoloa, HI, USA, 9–13 December 2019; pp. 1–6. [CrossRef]
36. Chahed, H.; Kassler, A.J. Software-Defined Time Sensitive Networks Configuration and Management. In Proceedings of the 2021 IEEE Conference on Network Function Virtualization and Software Defined Networks, NFV-SDN 2021, Heraklion, Greece, 9–11 November 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 124–128. [CrossRef]
37. Böhm, M.; Ohms, J.; Kumar, M.; Gebauer, O.; Wermser, D. Dynamic Real-Time Stream Reservation for IEEE 802.1 Time-Sensitive Networks with OpenFlow. In Proceedings of the 8th International Conference on Applied Innovations in IT, (ICAIIT), Kothen, Germany, 10 March 2020. [CrossRef]
38. Du, J.L.; Herlich, M. Software-defined Networking for Real-time Ethernet. In Proceedings of the Proceedings of the 13th International Conference on Informatics in Control, Automation and Robotics (ICINCO 2016) —Volume 2, Lisbon, Portugal, 29–31 July2016; Gusikhin, O., Peaucelle, D., Madani, K., Eds.; SciTePress: Setubal, Protugal, 2016; pp. 584–589. [CrossRef]
39. Said, S.B.H.; Truong, Q.H.; Boc, M. SDN-based configuration solution for IEEE 802.1 time sensitive networking (TSN). *ACM SIGBED Rev.* **2019**, *16*, 27–32. [CrossRef]
40. Häckel, T.; Meyer, P.; Korf, F.; Schmidt, T.C. Software-Defined Networks Supporting Time-Sensitive In-Vehicular Communication. In Proceedings of the 89th IEEE Vehicular Technology Conference, VTC Spring 2019, Kuala Lumpur, Malaysia, 28 April–1 May 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 1–5. [CrossRef]
41. Nsiah, K.A.; Alkhouri, K.; Sikora, A. Configuration of TSN Networks. In Proceedings of the 2020 IEEE 5th International Symposium on Smart and Wireless Systems within the Conferences on Intelligent Data Acquisition and Advanced Computing Systems (IDAACS-SWS), Dortmund, Germany, 17–18 September 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 1–5. [CrossRef]
42. Nam, S.; Kim, H.; Min, S. Simplified Stream Reservation Protocol Over Software-Defined Networks for In-Vehicle Time-Sensitive Networking. *IEEE Access* **2021**, *9*, 84700–84711. [CrossRef]
43. Balasubramanian, V.; Aloqaily, M.; Reisslein, M. An SDN architecture for time sensitive industrial IoT. *Comput. Netw.* **2021**, *186*, 107739. [CrossRef]
44. Nayak, N.G.; Dürr, F.; Rothermel, K. Incremental Flow Scheduling and Routing in Time-Sensitive Software-Defined Networks. *IEEE Trans. Ind. Inform.* **2018**, *14*, 2066–2075. [CrossRef]
45. Nayak, N.G.; Dürr, F.; Rothermel, K. Time-sensitive Software-defined Network (TSSDN) for Real-time Applications. In Proceedings of the Proceedings of the 24th International Conference on Real-Time Networks and Systems, RTNS 2016, Brest, France, 19–21 October 2016; ACM: New York, NY, USA, 2016; pp. 193–202. [CrossRef]
46. Gerhard, T.; Kobzan, T.; Blöcher, I.; Hendel, M. Software-defined Flow Reservation: Configuring IEEE 802.1Q Time-Sensitive Networks by the Use of Software-Defined Networking. In Proceedings of the 24th IEEE International Conference on Emerging Technologies and Factory Automation, ETFA 2019, Zaragoza, Spain, 10–13 September 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 216–223. [CrossRef]
47. Haur, N.K.; Chin, T.S. Challenges and Future Direction of Time-Sensitive Software-Defined Networking (TSSDN) in Automation Industry. In Proceedings of the Security, Privacy, and Anonymity in Computation, Communication, and Storage: 12th International Conference, SpaCCS 2019, Atlanta, GA, USA, 14–17 July 2019; Springer: Berlin/Heidelberg, Germany, 2019; Volume 11611; pp. 309–324. [CrossRef]
48. Boehm, M.; Ohms, J.; Kumar, M.; Gebauer, O.; Wermser, D. Time-Sensitive Software-Defined Networking: A Unified Control-Plane for TSN and SDN. In Proceedings of the Mobile Communication—Technologies and Applications; 24. ITG-Symposium, Osnabrueck, Germany, 15–16 May 2019; pp. 1–6.
49. Guo, M.; Shou, G.; Xue, J.; Hu, Y.; Liu, Y.; Guo, Z. Cross-domain Interconnection with Time Synchronization in Software-defined Time-Sensitive Networks. In Proceedings of the 2020 Asia Communications and Photonics Conference (ACP) and International Conference on Information Photonics and Optical Communications (IPOC), Beijing, China, 24–27 October 2020; pp. 1–3.
50. Xue, J.; Shou, G.; Li, H.; Liu, Y. Enabling Deterministic Communications for End-to-End Connectivity with Software-Defined Time-Sensitive Networking. *IEEE Netw.* **2022**, *36*, 34–40. [CrossRef]

51. Schriegel, S.; Kobzan, T.; Jasperneite, J. Investigation on a distributed SDN control plane architecture for heterogeneous time sensitive networks. In Proceedings of the 14th IEEE International Workshop on Factory Communication Systems, WFCS 2018, Imperia, Italy, 13–15 June 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 1–10. [CrossRef]
52. Zhang, X.; Shou, G.; Xue, J.; Li, H. An Error Compensation Method of Time Synchronization for Cross-domain Interconnection in SD-TSN. In Proceedings of the Optical Fiber Communications Conference and Exhibition, OFC 2022, San Diego, CA, USA, 6–10 March 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 1–3.
53. Böhm, M.; Ohms, J.; Wermser, D. Multi-Domain Time-Sensitive Networks - An East-Westbound Protocol for Dynamic TSN-Stream Configuration Across Domains. In Proceedings of the 24th IEEE International Conference on Emerging Technologies and Factory Automation, ETFA 2019, Zaragoza, Spain, 10–13 September 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 1363–1366. [CrossRef]
54. Deterministic Networking Problem Statement. Available online: https://www.rfc-editor.org/info/rfc8557 (accessed on 24 May 2023).
55. Kumar, G.N.; Katsalis, K.; Papadimitriou, P.; Pop, P.; Carle, G. Failure Handling for Time-Sensitive Networks using SDN and Source Routing. In Proceedings of the 2021 IEEE 7th International Conference on Network Softwarization (NetSoft), Milan, Italy, 27 June–1 July 2022; pp. 226–234. [CrossRef]
56. Mohammadi, S.; Colle, D.; Tavernier, W. Latency-aware Topology Discovery in SDN-based Time-Sensitive Networks. In Proceedings of the 8th IEEE International Conference on Network Softwarization, NetSoft 2022, Milan, Italy, 27 June–1 July 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 145–150. [CrossRef]
57. Pahlevan, M.; Schmeck, J.; Obermaisser, R. Evaluation of TSN Dynamic Configuration Model for Safety-Critical Applications. In Proceedings of the 2019 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking, ISPA/BDCloud/SocialCom/SustainCom 2019, Xiamen, China, 16–18 December 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 566–571. [CrossRef]
58. Garbugli, A.; Bujari, A.; Bellavista, P. End-to-end QoS Management in Self-Configuring TSN Networks. In Proceedings of the 17th IEEE International Conference on Factory Communication Systems, WFCS 2021, Linz, Austria, 9–11 June 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 131–134. [CrossRef]
59. Gutiérrez, M.; Ademaj, A.; Steiner, W.; Dobrin, R.; Punnekkat, S. Self-configuration of IEEE 802.1 TSN networks. In Proceedings of the 22nd IEEE International Conference on Emerging Technologies and Factory Automation, ETFA 2017, Limassol, Cyprus, 12–15 September 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 1–8. [CrossRef]
60. Houtan, B.; Bergström, A.; Ashjaei, M.; Daneshtalab, M.; Sjödin, M.; Mubeen, S. An Automated Configuration Framework for TSN Networks. In Proceedings of the 22nd IEEE International Conference on Industrial Technology, ICIT 2021, Valencia, Spain, 10–12 March 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 771–778. [CrossRef]
61. Kostrzewa, A.; Ernst, R. Achieving safety and performance with reconfiguration protocol for ethernet TSN in automotive systems. *J. Syst. Archit.* **2021**, *118*, 102208. [CrossRef]
62. Kostrzewa, A.; Ernst, R. Fast Failover in Ethernet-Based Automotive Networks. In Proceedings of the 23rd IEEE International Symposium on Real-Time Distributed Computing, ISORC 2020, Nashville, TN, USA, 19–21 May 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 134–139. [CrossRef]
63. Bülbül, N.S.; Ergenç, D.; Fischer, M. Towards SDN-based Dynamic Path Reconfiguration for Time Sensitive Networking. In Proceedings of the 2022 IEEE/IFIP Network Operations and Management Symposium, NOMS 2022, Budapest, Hungary, 25–29 April 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 1–9. [CrossRef]
64. Kong, W.; Nabi, M.; Goossens, K. Run-Time Recovery and Failure Analysis of Time-Triggered Traffic in Time Sensitive Networks. *IEEE Access* **2021**, *9*, 91710–91722. [CrossRef]
65. Thiele, D.; Ernst, R. Formal analysis based evaluation of software defined networking for time-sensitive Ethernet. In Proceedings of the 2016 Design, Automation & Test in Europe Conference & Exhibition, DATE 2016, Dresden, Germany, 14–18 March 2016; IEEE: Piscataway, NJ, USA, 2016; pp. 31–36.
66. *IEEE Std 802.1AB-2009*; IEEE Standard for Local and Metropolitan Area Networks—Station and Media Access Control Connectivity Discovery. IEEE: Piscataway, NJ, USA, 11 September 2009.
67. Bülbül, N.S.; Ergenç, D.; Fischer, M. SDN-based Self-Configuration for Time-Sensitive IoT Networks. In Proceedings of the 46th IEEE Conference on Local Computer Networks, LCN 2021, Edmonton, AB, Canada, 4–7 October 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 73–80. [CrossRef]
68. Gärtner, C.; Rizk, A.; Koldehofe, B.; Guillaume, R.; Kundel, R.; Steinmetz, R. On the Incremental Reconfiguration of Time-sensitive Networks at Runtime. In Proceedings of the 2022 IFIP Networking Conference (IFIP Networking), Catania, Italy, 13–16 June 2022; pp. 1–9. [CrossRef]
69. Takeuchi, J. Requirements for Automotive AVB System Profiles. 2011. Available online: https://avnu.org/wp-content/uploads/2014/05/Contributed-Automotive-Whitepaper_April-2011.pdf (accessed on 24 May 2023).
70. Sambo, N.; Fichera, S.; Sgambelluri, A.; Fioccola, G.; Castoldi, P.; Katsalis, K. Enabling Delegation of Control Plane Functionalities for Time Sensitive Networks. *IEEE Access* **2021**, *9*, 136151–136163. [CrossRef]
71. Rother, B.; Kasparick, M.; Schweißguth, E.; Golatowski, F.; Timmermann, D. Automatic Configuration of a TSN Network for SDC-based Medical Device Networks. In Proceedings of the 16th IEEE International Conference on Factory Communication Systems, WFCS 2020, Porto, Portugal, 27–29 April 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 1–8. [CrossRef]

72. Zhao, Y.; Wang, W.; Zhang, J. Time-sensitive software-defined networking (TS-SDN) control architecture for flexi-grid optical networks with data center application. *Photonic Netw. Commun.* **2014**, *28*, 82–91. [CrossRef]
73. Raagaard, M.L.; Pop, P.; Gutiérrez, M.; Steiner, W. Runtime reconfiguration of time-sensitive networking (TSN) schedules for Fog Computing. In Proceedings of the IEEE Fog World Congress, FWC 2017, Santa Clara, CA, USA, 30 October–1 November 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 1–6. [CrossRef]
74. Pop, P.; Raagaard, M.L.; Gutierrez, M.; Steiner, W. Enabling Fog Computing for Industrial Automation Through Time-Sensitive Networking (TSN). *IEEE Commun. Stand. Mag.* **2018**, *2*, 55–61. [CrossRef]
75. Lee, J.; Park, S. Time-Sensitive Network Profile Service for Enhanced In-Vehicle Stream Reservation. In Proceedings of the 4th International Conference on Control, Robotics and Cybernetics, CRC 2019, Tokyo, Japan, 27–30 September 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 133–136. [CrossRef]
76. Metaal, M.A.; Guillaume, R.; Steinmetz, R.; Rizk, A. Integrated Industrial Ethernet Networks: Time-sensitive Networking over SDN Infrastructure for mixed Applications. In Proceedings of the 2020 IFIP Networking Conference, Networking 2020, Paris, France, 22–26 June 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 803–808.
77. Li, Y.; Jiang, J.; Lee, C.; Hong, S.H. Practical Implementation of an OPC UA TSN Communication Architecture for a Manufacturing System. *IEEE Access* **2020**, *8*, 200100–200111. [CrossRef]
78. Sudhakaran, S.; Montgomery, K.; Kashef, M.; Cavalcanti, D.; Candell, R. Wireless Time Sensitive Networking for Industrial Collaborative Robotic Workcells. In Proceedings of the 17th IEEE International Conference on Factory Communication Systems, WFCS 2021, Linz, Austria, 9–11 June 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 91–94. [CrossRef]
79. Cavalcanti, D.; Perez-Ramirez, J.; Rashid, M.; Fang, J.; Galeev, M.; Stanton, K. Extending Accurate Time Distribution and Timeliness Capabilities Over the Air to Enable Future Wireless Industrial Automation Systems. *Proc. IEEE* **2019**, *107*, 1132–1152. . [CrossRef]
80. Bush, S.F.; Mantelet, G. *Industrial Wireless Time-Sensitive Networking: RFC on the Path Forward*; Avnu Alliance: Beaverton, OR, USA, 2018.
81. Cavalcanti, D. *Wireless Time-Sensitive Networking (WTSN)*; Intel Lab: Santa Clara, CA, USA, 2020.