Avnu Alliance™ Best Practices

Theory of Operation for TSN-enabled Systems

Applied to Industrial Markets

Revision 1.0

Author

Eric Gardiner

Contributors

Paul Brooks, Sundeep Chandhoke, Rodney Cummings, Paul Didier, George Ditzel, René Hummen, Albert Mitchell, Bogdan Tenea, Todd Walter, Tom Weingartner, Jordon Woods, and Steve Zuponcic

Avnu Alliance

Prologue

The primary goal of this document is to describe the theory of operation for a system architecture comprising TSN-enabled end stations and bridges in support of a variety of markets and their applications. Presently, this document focuses on Industrial markets, and the architecture applies a specific set of TSN technologies to meet the needs of a subset of Industrial applications. In the future, this document will include additional technologies in support of a broader set of applications and markets.

1 Introduction and Scope

Recent work by IEEE 802, the Internet Engineering Task Force (IETF), and other standards groups has extended the number of applications that Ethernet networks can serve. These applications include professional audio/video, automotive, and industrial, all of which can experience system malfunction when application requirements exceed the path delay variation and/or latency capabilities of the underlying network. These standards, driven primarily by the IEEE 802.1 Time-Sensitive Networking (TSN) task group, define new mechanisms for creating distributed, synchronized, real-time systems using standard Ethernet technologies.

For industrial markets specifically, these standards support the ability to simplify development and deployment of distributed and synchronized control systems commonly found in a variety of industrial applications including machine control, factory automation, power generation and distribution, oil and gas exploration, etc. Many current generation of control applications are deployed using nonstandard network interfaces and infrastructure. But TSN and related efforts, including IETF projects such as DetNet, provide mechanisms to solve these applications using standard Ethernet technologies in a manner that enables convergence between Operational Technology (OT) and Information Technology (IT) data on a common physical network. This increases connectivity to industrial devices and enables a faster path to new business possibilities including the Industrial Internet of Things (IIoT), big data analytics, and smart connected systems and machines.

The Industrial Internet of Things, much like the consumer Internet of Things, encapsulates the capture, storage, contextualization, analysis, and presentation of data generated in production processes. While capture of data is frequently from existing plant devices, that data is likely unused, or only partially used in the real-time operation of the production processes; it may also be from new instrumentation specifically deployed for IIoT applications. Storage of data is typically using either public or private cloud techniques and typically consolidates multiple plants into an enterprise or fleet view. Contextualization and analysis aim to provide new insights into plant and equipment operation, primarily enabling shorter new product introduction times, higher equipment efficiency, and predictive maintenance; these analytics are increasing being provided using an as-a-service business model. Presentation must be contextualized for individual stakeholders – from plant floor operators and maintainers to operations managers and executives to machine builders and value chain partners. All these data flows put new demands on networking throughout the enterprise and demand network convergence right into the heart of machinery.

Standards groups including IEEE 802.1 and IETF define foundational mechanisms that enable a variety of technical options in support of diverse market requirements. But these standards do not generally specify the set of technical options a given market should adopt to ensure interoperable use of a shared network between individual products in a system. For example, IEEE 802.1 defines multiple mechanisms for configuring traffic shapers within network infrastructure (that is, Ethernet bridges/switches) to ensure quality of service. But the 802.1 standards do not mandate the use of any particular configuration mechanism, leaving it to device manufacturers or outside industry groups to select from the set of available options. It is important to select the relevant options for a given market; otherwise, applications within that market may make assumptions about the underlying TSN-enabled network services that other solution providers do not deliver, resulting in failure to coexist on a shared network.

To enable interoperable use of a shared TSN-enabled network, Avnu, an organization composed of leading silicon, network infrastructure, and end station suppliers, selects and certifies a set of underlying mechanisms to meet a given market's requirements. Avnu's focus is to create a common network foundation so that multiple industry group or vendor applications and protocols can share a TSN-enabled network, as shown in Figure 1. In addition to selecting the appropriate mechanisms to support foundational network services for a given market, Avnu also offers open source software, hardware reference designs, test plans, and certification services to develop and verify the correct operation of TSN-enabled products.

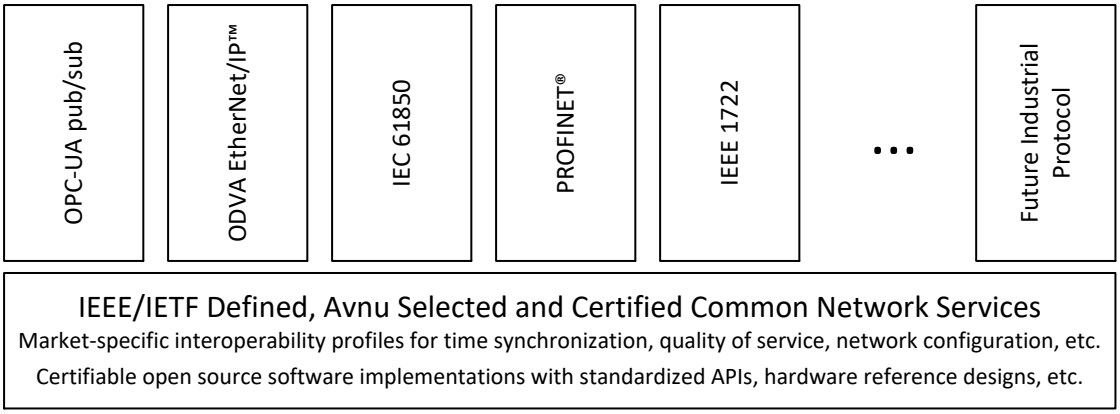

Figure 1: Multiple Applications Using Avnu Common Network Services

This document's primary purpose is to describe a system architecture, including reference architectures for the services comprising network infrastructure and Industrial End Stations in the system, for enabling multiple industry group and vendor applications and protocols to share a TSN-enabled network. The system architecture details are described in section 4.

To provide real-world application context for the system architecture, section 2 describes several industrial use cases, including machine control, power generation and monitoring, power transmission and distribution, and oil and gas exploration. Section 2 also summarizes the requirements for each application. As industrial applications have a comparatively long lifespan compared to enterprise IT infrastructure, the ability to support existing, non-TSN devices is critical for success. Future revisions of this document will more completely address

migration and integration approaches to enable and integrate non-TSN technologies, especially those already deployed in operational plants.

Prior to describing the detailed system architecture, section 3 introduces the fundamental mechanisms that the system architecture is built on, including time synchronization, quality of service using scheduled transmission, and network configuration using a centralized approach informed by software-defined networking (SDN) concepts. In addition to describing the fundamental mechanisms comprising the system architecture, section 3 also describes the associated IEEE, IETF, and other standards defining each mechanism.

As noted above, this document's primary purpose is to describe a system architecture for enabling multiple industrial applications and protocols to share a TSN-enabled network. Section 4 describes this system architecture in detail, applying the fundamental mechanisms described in section 3, and meeting the application requirements motivated by section 2. Section 4 covers the following topics:

- How time is synchronized in the system
- The logical entities comprising a TSN-enabled industrial system, including bridges, end stations (including bridged stations), and software controllers for configuring the system
- How the system is configured to meet an industrial application's requirements, including the flow of configuration information between the logical entities comprising the system, and the interfaces between those logical entities

As Avnu's work evolves to comprehend additional industrial market requirements, the set of underlying mechanisms and corresponding standards this architecture supports will also evolve, eventually including emerging mechanisms such as frame preemption, redundancy, ingress policing, and security. In addition, the architecture this document describes will evolve to support additional capabilities, including support for multiple IEEE 1588 profiles, guidelines for scaling to very large network architectures, and aggregation/composition of multiple networks into a single TSN-enabled network domain.

While this document describes a way to configure a TSN-enabled network so that multiple vendor and industry group applications and protocols can share it, the system architecture, entity and service reference architectures, and interface descriptions are *informative* in nature. A companion document, the *Avnu Industrial Interoperability Specification*, using this document for context, defines *normative* requirements for bridges and end stations to ensure a given vendor's products can interoperate (that is, share a TSN-enabled network). Using the conformance statements derived from the *Avnu Industrial Interoperability Specification*, Avnu will define test plans and certification services for Industrial products, including bridges, end stations, and bridged end stations.

1.1 Intended Audience

The primary audience of this document is developers of products used to solve industrial control problems. Such product types include bridges, end stations, and bridged end stations. By applying the concepts described in section 4, developers can ensure that their products can coexist in a shared TSN-enabled network, regardless

of the set of applications and protocols running on the network. This document specifically describes architectural reference models for a given product type (bridges, end stations, or bridged end stations), informing product developers of the types of services, including the responsibilities of each service, that a given product type should implement.

Another key audience of this document is contributors to industry-specific consortia, who can apply the network configuration mechanisms described in section 4.2, especially the end station reference model described in section 4.3, to carry existing application protocol data on a TSN-enabled network.

1.2 Avnu's Relationship to AVB/TSN

Avnu is an industry alliance to foster, support and develop an ecosystem of vendors providing interoperable Audio Video Bridging (AVB) and Time Sensitive Networking (TSN) devices for wide availability to consumers, system integrators and other system-specific applications. Avnu provides interoperability guidelines and compliance testing for device manufacturers, as well as technical guidance such as this document for implementing AVB/TSN systems. For more information, visit our website.

Contents

1.3 Terms, Definitions, and Abbreviations Used in this Document

This section defines terms and abbreviations used in this document.

1.3.1 Definitions from IEEE 802.1

For convenience, this section includes definitions of terms specified in IEEE 802.1 that are relevant to this document. Consult the *IEEE Standards Dictionary: Glossary of Terms & Definitions* for definitions not included in this section.

Bridge: A system that includes Media Access Control (MAC) Bridge or Virtual Local Area Network (VLAN) Bridge component functionality and that supports a claim of conformance to Clause 5 of IEEE Std 802.1Q-2014 for system behavior.

Centralized Network Configuration (CNC): A centralized component that configures network resources on behalf of TSN applications (users).

Centralized User Configuration (CUC): A centralized component that discovers and configures application (user) resources in end stations. The CUC exchanges information with the CNC in order to configure TSN features on behalf of its end stations.

End station: A device attached to a local area network (LAN) or metropolitan area network (MAN), which acts as a source of, and/or destination for, traffic carried on the LAN or MAN.

Gate-close event: An event that occurs when the transmission gate associated with a queue transitions from the Open state to the Closed state, disconnecting the transmission selection function of the forwarding process from the queue and preventing it from selecting frames from that queue.

Gate-open event: An event that occurs when the transmission gate associated with a queue transitions from the Closed state to the Open state, connecting the transmission selection function of the forwarding process to a queue and allowing it to select frames from that queue.

Gate control list: An ordered list of gate operations, associated with a given Port.

Gating cycle: The period of time over which the sequence of operations in a gate control list repeats.

Grandmaster: The time-aware system that contains the best clock, as determined by the best master clock algorithm (BMCA), in the generalized precision time protocol (gPTP) domain.

Listener: The end station that is the destination, receiver, or consumer of a stream.

Spanning tree: A simply and fully connected active topology formed from the arbitrary physical topology of connected Bridged Network components by relaying frames through selected Bridge Ports and not through others. The protocol parameters and states used and exchanged to facilitate the calculation of that active topology and to control the Media Access Control (MAC) relay function.

Stream: A unidirectional flow of data from a Talker to one or more Listeners.

Synchronized time: The synchronized time of an event is the time of that event relative to the grandmaster.

Synchronized time-aware systems: Two time-aware systems are synchronized to a specified uncertainty if they have the same epoch and their measurements of the time of a single event at an arbitrary time differ by no more than that uncertainty.

Syntonized time-aware systems: Two time-aware systems are syntonized if the duration of the second is the same on both, which means the time measured by each advances at the same rate. They can but need not share the same epoch.

Talker: The end station that is the source or producer of a stream.

Time-sensitive stream: A stream of data frames that are required to be delivered with a bounded latency.

Traffic class: A classification used to expedite transmission of frames generated by critical or time-sensitive services. Traffic classes are numbered from zero through N-1, where N is the number of outbound queues associated with a given Bridge Port, and 1 <= N <= 8, and each traffic class has a one-to-one correspondence with a specific outbound queue for that Port. Traffic class 0 corresponds to non-expedited traffic; nonzero traffic classes correspond to expedited classes of traffic. A fixed mapping determines, for a given priority associated with a frame and a given number of traffic classes, what traffic class will be assigned to the frame.

Transmission Gate: A gate that connects or disconnects the transmission selection function of the forwarding process from the queue, allowing or preventing it from selecting frames from that queue. The gate has two states, Open and Closed.

1.3.2 Industrial TSN Definitions

This section defines terms relevant to industrial systems built on TSN and related standards.

Bridged End Station: An end station that operates with an IEEE 802.1 bridge embedded within or alongside it.

Working Clock Timescale: The timescale within an Industrial TSN Domain that provides a continuous and monotonically increasing clock to applications for performing scheduled network transmission and application input/output functions. In this context, continuous and monotonically increasing means that the clock does not decreases in time, and it does not jump ahead in time by more than a specified amount.

Global Traceable Timescale: Synchronized time in a network that is traceable to a known source. In the context of an Industrial TSN Domain, this applies to a second, optional timescale that Industrial End Stations can use to correlate the Working Clock Timescale in one Industrial TSN Domain with a second Industrial TSN Domain.

Industrial End Station (IES): A TSN-enabled end station that implements the features necessary to send or receive time-sensitive streams within an Industrial TSN Domain.

Industrial TSN Domain: A logical group of Industrial End Stations and IEEE 802.1 bridges that form a network with their own grandmaster that does not propagate time outside the group.

Stream Object: A data structure that holds configuration information for a time-sensitive stream on an Industrial End Station; it is shared between the Industrial End Station's network Interface, application, and the Centralized User Configuration (CUC).

1.4 References

Standard	Description
IEEE 1588-2008	IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems https://standards.ieee.org/findstds/standard/802.1AS-2011.html
IEEE 802.1AS-2011	Timing and Synchronization for Time-Sensitive Applications in Bridged Local Area Networks http://standards.ieee.org/getieee802/download/802.1AS-2011.pdf
IEEE 802.1AS-Rev	Timing and Synchronization for Time-Sensitive Applications in Bridged Local Area Networks (*draft*) http://www.ieee802.org/1/pages/802.1AS-rev.html
IEEE 802.1Q-2014, Clause 34	Forwarding and Queuing Enhancements for Time-Sensitive Streams (FQTSS) https://standards.ieee.org/findstds/standard/802.1Q-2014.html
IEEE 802.1Qbv-	IEEE Standard for Local and Metropolitan Area Networks -- Bridges and Bridged Networks - Amendment 25: Enhancements for Scheduled Traffic

2015	https://standards.ieee.org/findstds/standard/802.1Qbv-2015.html
IEEE 802.1Q-2014, clause 35	Stream Reservation Protocol (SRP) https://standards.ieee.org/findstds/standard/802.1Q-2014.html
IEEE 802.1Qcc	Stream Reservation Protocol (SRP) Enhancements and Performance Improvements (*draft*) http://www.ieee802.org/1/pages/802.1cc.html
IEEE 802.1Qci	Per-Stream Filtering and Policing http://www.ieee802.org/1/pages/802.1ci.html
IEEE 802.1Qcp	Bridges and Bridged Networks Amendment: YANG Data Model (*draft*) http://www.ieee802.org/1/pages/802.1cp.html
IETF RFC 6020	YANG – A Data Modeling Language for the Network Configuration Protocol (NETCONF) https://tools.ietf.org/html/rfc6020
IEEE 802.1AB	IEEE Standard for Local and Metropolitan Area Networks - Station and Media Access Control Connectivity Discovery https://standards.ieee.org/findstds/standard/802.1AB-2016.html
IETF DetNet	Deterministic Networking (DetNet) (*draft*) https://datatracker.ietf.org/wg/detnet/documents/

1.5 Revision History

Version	Description
1.0	Initial Release

2 Industrial Use Cases and Requirements

Industrial markets encompass a wide range of applications from motion control, to machine control, to electrical power generation, to power distribution, to process control. While the specific requirements for these applications vary, they share a common need for distributed sensing, logic, and actuation. Additionally, there is often a need to merge equipment designed for one industrial vertical market into another application, so it is useful to look at all industrial applications together when defining foundational elements for communications and connectivity. This section describes several industrial use cases and the networking requirements that result from each use case.

2.1 Industrial Automation

Figure 2 shows the representative architecture of an industrial control application and associated network design.

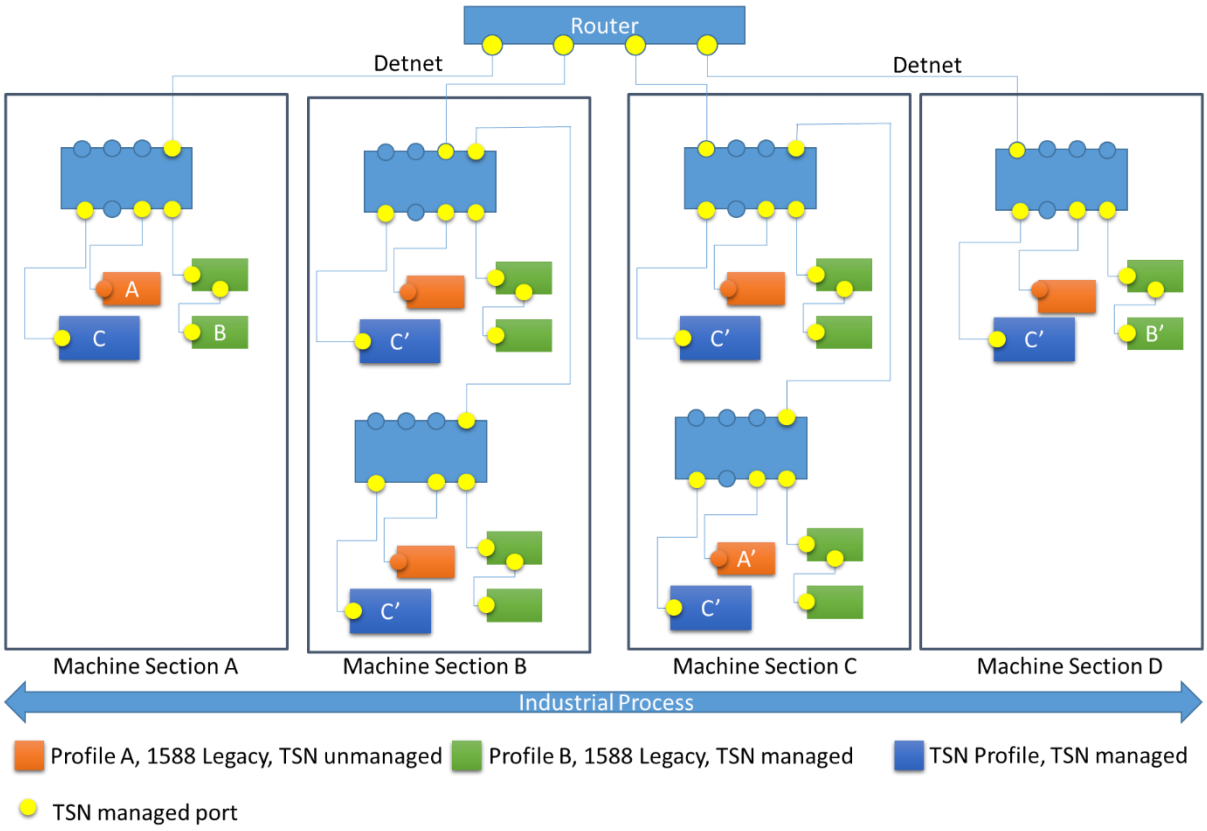

Figure 2: Industrial Process Control Use Case Diagram

In this system, a single machine, which consists of four different sections, controls an industrial process. Different OEMs deliver each section of machinery to the process, with each OEM specializing in the portion of the process it controls. In this example, the end user has seven manufacturing sites around the globe. There are fifteen of these machines per site. Each machine's IP addressing scheme is identical to the other machines in

the same manufacturing facility, which are in turn identical to the addressing schemes of the other facilities. Moreover, each machine section is a subnet with a unique VLAN so that the equipment can be constructed modularly to assist functional organization. All sections of the machine are synchronized and coordinated to produce the final product, and relevant events are timestamped so that data on the manufacturing floor can be correlated against data in the Manufacturing Execution System (MES) and data from the supply chain. The entire manufacturing facility uses the same understanding of absolute time, and all events are related back to the common notion of globally-traceable "wall clock time."

Additionally, this use case includes products and technologies that use a variety of IEEE 1588 profiles for synchronizing time, the underlying technology for which is discussed in section 3.1.1. In this example, Component A in Machine Section A may communicate with Component A' in Machine Section C. Component B in Section A may communicate with Component B' in Machine Section D. Finally, Component C in Machine Section A produces data that C' components in sections B, C, and D consume. To meet the requirements of this application, Ethernet switches support concurrent operation of or translation between multiple IEEE 1588 profiles, providing a migration path for existing products and technologies to be included in the wider TSN value proposition. In this use case, Components A and A' and B and B' would require time gateway functionality, discussed in section 4.1.2. Presently, such time gateway functionality is not specified in any standards communities, and individual suppliers can develop time gateway functionality to address market need.

This use case also shows the need for communication solutions at both layer-2 (switching) and layer-3 (routing). Today, the IEEE 802.1 TSN task group defines solutions at layer-2 in the architecture, creating the mechanisms needed to build a common network foundation in support of diverse markets and applications using standard Ethernet technology. The Internet Engineering Task Force (IETF) is defining layer-3 functionality as part of the "DetNet" project, extending time-sensitive networking capabilities, including those built on layer-2 TSN mechanisms, to routers.

Avnu's initial focus is on layer-2 Ethernet solutions, reflecting the progress of each standards effort, and this document focuses on creating a common network foundation using layer-2 technologies. Future versions of this architecture will expand in scope to include layer-3 technologies, including DetNet.

In addition, when considering network requirements for industrial automation systems, consider a typical industrial control room or collaboration center, as shown in Figure 3.

Figure 3: Typical Industrial Automation Control Room[1]

In the control room shown in Figure 3, multiple systems deliver a variety of time-sensitive data streams, including the following:

- A video system carries multiple camera surveillance video streams to a number of screens
- A sound system broadcasts either automated alarm messaging or control room audio to plant floor workers
- An automation information protocol (for example, OPC-UA pubsub) delivers real-time updates to monitors
- An automation control protocol (for example, EtherNet/IP) monitors pushbuttons and emergency stops

In this example, a single network carries all these data streams, and the allocation of bandwidth must be centrally managed. However, common tools do not configure these streams – each system has its own engineering tool for stream configuration. Therefore, multiple engineering tools must provide stream requirements to a single network configuration controller that must resolve all the requirements into a single

[1] Photo Credit: VGB Power Tech GmbH, CC BY-SA 3.0

network schedule. This schedule must then be communicated to both bridges and end nodes. This configuration process is introduced in section 3.4.2 and described in detail in section 4.2.

2.2 Machine Control

Companies creating industrial machinery used for fabrication, assembly, and testing are constantly upgrading their machines' capabilities. Increasingly, these upgraded capabilities depend on free-flowing data both within the machine and between the machines, from the machine to the user's enterprise, and on to the remote machine builder's remote facilities. This flow of information reduces the cost of machine customization, enables increases in machine availability, and can provide the end user with simpler integration into their facility and logistics systems. It allows the machine builder to take greater ownership of operating efficiency, support servicing, and consumables management.

For reference, consider equipment used for semiconductor production. Semiconductor production machines perform multiple chemical and photo lithographic steps where electronic circuits are created on a wafer of silicon. This wafer is then tested, cut into individual chips, and then the chips are packaged to provide electrical contact points and to allow for thermal management. The cost of producing and processing a silicon wafer is very high. To minimize the cost per chip, designers optimize the size of chips and distance between the chips on the wafer. This, coupled with smaller semiconductor processes, is allowing designers to pack many chips onto a single wafer. The step where the wafer is cut to produce multiple chips is called wafer dicing. To handle the tightly packed chips, the wafer dicing machine needs to make very precise cuts with precision measured in 1/1000 of a millimeter. Figure 4 shows the subsystems comprising a wafer dicing machine.

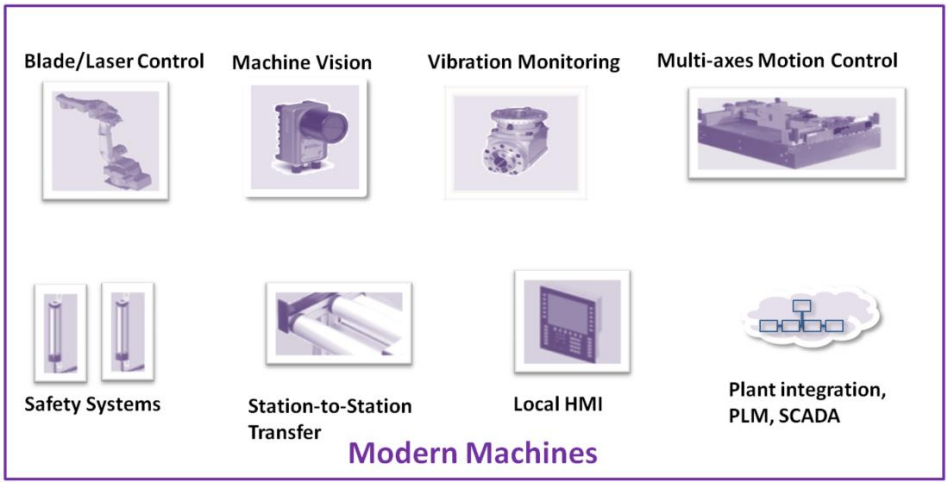

Figure 4: Functions on a Modern Production Machine

To achieve the required dicing accuracy, a typical implementation uses precise dicing blades or lasers to etch and cut the wafer. This cutting is controlled with high-performance motion control axes often using fast-

responding mechanisms such as voice coil motors or multi-axis galvanometers for laser control. These axes get inputs from high-resolution absolute position feedback mechanisms. Machine vision is now sometimes also used in the main control loop to assure proper position. These performance axes are sometimes also augmented with vibration feedback to adjust for small movements of the system. The controller must read the sensors, which are used as inputs into sophisticated control algorithms, and the cutting motion adjusted many thousands of times per second. This control loop must be deterministic and precisely timed. Around this core function of the machine, there are additional tasks such as coordinated multi-axis motion to load the wafer and unload the cut chips. Many machines use vibration or power quality measurements as part of predictive maintenance. The system normally has a local user interface where an operator can interact with the machine and has connections to the manufacturing logistics and enterprise systems.

To meet the stability and reliability requirements of high-speed closed loop motion control, a network needs to consistently deliver the control packets between the drives/sensors and the controller with a latency of <100 µs. The motion axes also need to be coordinated, requiring time synchronization between the nodes of <1 µs. Without TSN enhancements, achieving these performance requirements with standard, shared Ethernet requires a high level of technical competence and can result in prescriptive network architectures. Even then, it cannot guarantee this performance over the lifetime of the machine as new IIoT solutions are applied that were not designed at the outset. To enable ease of use and design, these fast control loops frequently use isolated proprietary buses, with modified, nonstandard hardware in the nodes and bridges to eliminate queuing and provide latency bounding, even when they run on an infrastructure connected using an Ethernet physical layer and Category 5 cabling. This proprietary bus often also services other discrete sensors and actuators.

Typically, these buses cannot accommodate IIoT applications; for example, cameras require very high bandwidth and can cause congestion problems on a shared media. Typically, these cameras must run on a dedicated standard Ethernet link or a USB 3.0 connection, or may use a vision-specialized serial bus in parallel to the proprietary motion bus. The controllers and the HMI are also connected using standard Ethernet, and this same network bus may be used to integrate the machine into the larger manufacturing process, where it directly communicates with neighboring machines. This connection also provides integration into the overall plant MES system and mechanisms for remote connection to the original machine builder's maintenance and service systems.

To meet these application requirements, existing systems frequently use multiple bus layers, leading to a situation where Ethernet is common in control applications but where a typical machine supports multiple versions of Ethernet, each optimized to meet the requirements of a specific task. This technical approach leads to challenges for the machine designer, including bandwidth limitations, integration and configuration complexity, and limited remote access. To address these challenges, the IEEE 802 TSN task group and Avnu are enhancing standard Ethernet to provide bounded, low latency, shared synchronized time, and support for high bandwidth. With these updates, standard, TSN-enabled Ethernet can offer convergence, performance, and cost optimizations when compared to existing solutions. Figure 5 shows some of these benefits, enabling converged IIoT applications.

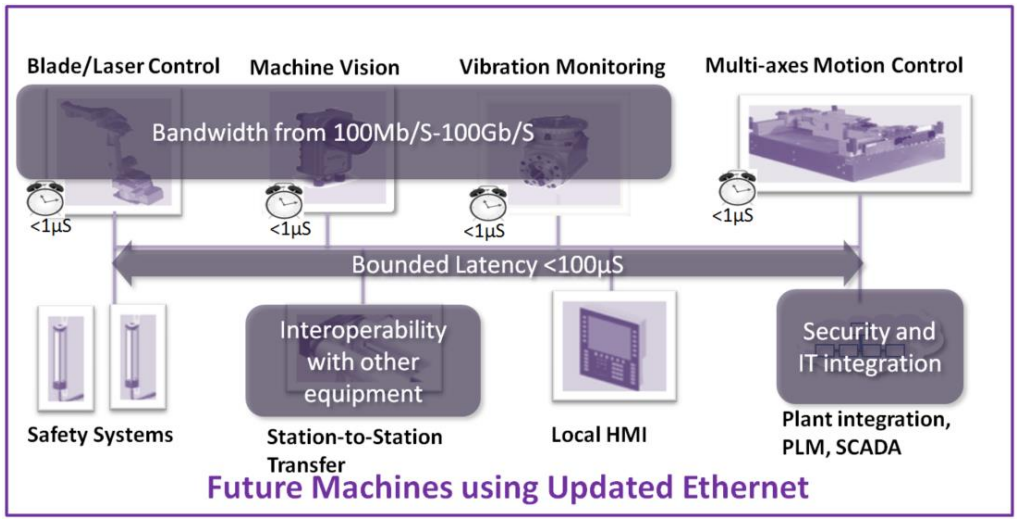

Figure 5: Converged, TSN-Enabled Ethernet Enabling IIoT Applications

2.3 Industrial Networking Requirements

Ethernet is a desired communication mechanism for all these applications due to its bandwidth, flexibility, ubiquitous use, and ability to provide clean access into IT infrastructure. The enhancements coming to the Ethernet standard, such as TSN and DetNet, are poised to improve on the ability for Ethernet to provide these mechanisms in a high-performance, consistent, scalable fashion. Table 1 summarizes the requirements for using Ethernet to solve the industrial use cases discussed in sections 2.1 and 2.2.

Industrial Network Requirement	Benefit
Time synchronization	Enables common clock for transmission scheduling, correlated I/O, etc.
High packet rates	Enables many servers (or similar) to communicate small data payloads with few clients (or similar)
Predictable low/guaranteed latency	Enables actuation within a guaranteed latency from completion of computation and computation within a guaranteed latency from sensing
Low cycle times	Enables scalable and user-selectable cyclic update rates that can meet or exceed legacy Industrial Ethernet solutions
Available bandwidth	Enables applications to operate predictably in the presence of network congestion
Redundancy and policing	Enables fault tolerance due to component failures, misbehaving components, etc.
Scalability	Can grow from small systems to large systems, including multimachine and multnetwork integration at a plant level
Converged network	Enables coexistence with best effort traffic, supports

	multiple industrial protocols, and integrates with plant IT systems
Composable system design	Enables secure functional isolation of different domains within the same physical network, aligned with existing IT standards like VLANs and subnets
Support across large network infrastructure	Enables coordinated actuation between machine segments on different subnets (for example: 192.168.0.0/255 to 192.168.1.0/255 through connected routing, 192.168.0.0/255 to 192.168.0.0/255 through network address translation via 192.168.1.0/255, 192.168.0.0/255 to 10.10.0.0/254 through dynamic routing, etc.)
Topology flexibility	Enables common industrial network topologies including line, ring, and tree
Offline prediction	Enables system design to be assured before purchase of components or installation and integration with other systems at end user premises
Security	Enables protection of critical assets and processes while also providing integration into IT infrastructure and remote management
Backwards compatibility	Includes mechanisms to integrate any existing IEEE 802.3 standard protocols, equipment, and installations
Augmented in runtime	Enables devices to be added to the network without interfering with runtime operation of any other device on the network; enables devices to be removed from the network without impacting any data flows in which they are not directly engaged
Consistent reconfiguration and restart	Ensures that scheduled communications between devices can be locked in time/phase through system reconfiguration and restart to ensure consistent operation of commissioned devices through the system lifecycle
Ability to migrate	Ensures that TSN-enabled infrastructure can replace legacy infrastructure without affecting relationships between existing and remaining legacy end nodes; ensures that end nodes can be replaced one at a time or subsystem by subsystem without impacting remainder of legacy end nodes
Ability to duplicate	Allows identical operation of machines deployed in different geographies and different years; allows offline reproduction of network configuration (for example, for system fault rectification)

Table 1: Industrial Network Requirements and Benefits

3 TSN Fundamentals

This section describes the foundational TSN mechanisms used to build distributed, real-time industrial control systems including those discussed in section 2. These foundational mechanisms include the following:

- Time synchronization
- Quality of service
- Stream identification
- Network configuration

As new foundational mechanisms are introduced that benefit industrial systems, this system architecture will evolve to comprehend those mechanisms, including redundancy, frame preemption, ingress policing, and security.

3.1 Time Synchronization

Industrial control systems composed of multiple distributed controllers, each with associated sensors and actuators, often require precise, synchronized time to facilitate event coordination and data correlation. For example, an industrial control application may need to synchronize distributed motion controllers to automate a manufacturing process. To achieve precise time synchronization, controllers in the system must have direct access to timing signals from a common clock source, or the controllers must synchronize their individual clocks to a common time base.

Because enabling direct access to a common clock source can be costly or infeasible in distributed environments, distributed clock synchronization mechanisms exist for aligning local clocks to a common time base. Examples of distributed clock synchronization include synchronizing to a Global Positioning System (GPS) satellite, synchronizing a controller's internal clock to a Network Time Protocol (NTP) time server, or a group of controllers synchronizing to a common time source using the IEEE 1588 Precision Time Protocol (PTP). Instead of sharing timing signals directly, controllers periodically exchange synchronization information, adjusting their local timing sources to match each other.

3.1.1 IEEE 1588-2008

IEEE 1588 provides a standard mechanism for synchronizing clocks connected via multicast-capable networks, including Ethernet. Released as a standard in 2002 and revised in 2008, IEEE 1588 was designed to provide fault tolerant synchronization between heterogeneous networked clocks requiring little network bandwidth overhead, processing power, and configuration management. IEEE 1588 provides this by defining a protocol known as the *Precision Time Protocol (PTP)*. Using PTP, all participating clocks are synchronized to the highest quality clock in the network.

IEEE 1588 defines a distributed algorithm, called the *Best Master Clock Algorithm* (BMCA), to select the clock source that participants in the system will synchronize to. Using the BMCA, participants advertise their clock's

capabilities so that clocks can be ranked against each other. If a participant observes that higher-ranking clock is present on the network, the participant will cease advertising its own clock, ultimately enabling participants to converge on the single clock each will synchronize to. This highest-ranking clock is called the *grandmaster* clock, used to synchronize all other *slave* clocks. If the grandmaster clock is removed from the network, or if its characteristics change such that it is no longer the highest-ranking clock, the BMCA enables the participating clocks to automatically converge on a new highest-ranking clock, which becomes the new grandmaster.

Slave clocks synchronize to the grandmaster using bidirectional multicast communication as shown in Figure 6. The grandmaster clock periodically issues a *sync* packet containing a timestamp reflecting when the packet left the grandmaster clock. The grandmaster may also issue a *follow up* packet containing the timestamp for the sync packet. The use of a separate follow up packet allows the grandmaster to accurately timestamp the sync packet on networks where the departure time of a packet cannot be known accurately in the initial *sync* message. Clocks that use a separate follow up packet are known as *two-step* clocks. Some master clocks, known as *one-step* clocks, can provide an accurate timestamp in the *sync* message itself, avoiding the need for a second *follow-up* message.

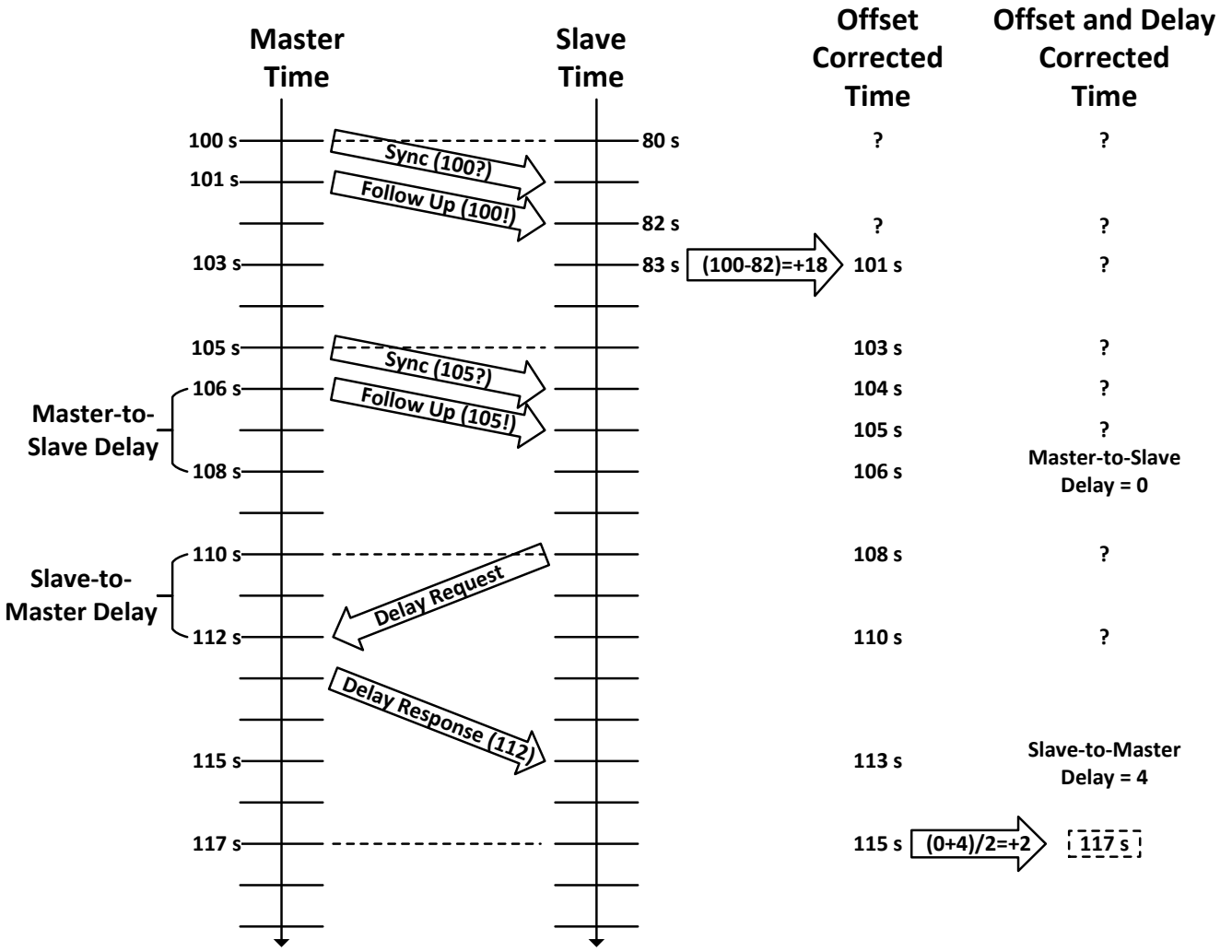

Figure 6: IEEE 1588 Synchronization Packet Flow and Clock Adjustment Example

A slave clock receives the grandmaster's sync packet and timestamps the packet's arrival time using its own clock. The difference in the sync packet's departure timestamp and the sync packet's arrival timestamp is the combination of the slave clock's offset from the master and the network propagation delay. By adjusting its clock by the offset measured at this point, the slave clock can reduce the time difference between it and the master to the network propagation delay only.

IEEE 1588 operates under the assumption that the network propagation delay is symmetrical. By making this assumption, the slave can discover and compensate for the propagation delay. The slave clock accomplishes this by issuing a *delay request* packet that is timestamped on departure from the slave. The master clock receives and timestamps this delay request packet, and the arrival timestamp is sent back to the slave clock in a *delay response* packet. The difference between these two timestamps is used to calculate the network propagation delay. In addition to this *delay request-response* mechanism, which can measure the end-to-end path delay between the master and slave clocks and can operate over paths that include non-time aware switches, IEEE 1588 also defines a *peer delay* mechanism that measures the path delay between two directly connected network ports. When all paths between a master and slave clock are time aware, including switches, the peer delay mechanism offers some advantages, including the ability to scale more easily as the number of slave clocks increases.

By sending and receiving these synchronization packets, a slave clock can accurately measure the offset between its local clock and the grandmaster clock. The slave can then adjust its clocks by this offset to match the time of the grandmaster, resulting in distributed, synchronized clocks.

While IEEE 1588 defines the foundational mechanisms for distributed time synchronization, the standard leaves it to industry groups and system integrators to select 1588-defined configurable attribute values and optional features so that time can be synchronized in an interoperable manner. As a result, individual *profile* specifications exist for specific market segments, defining how to use IEEE 1588 to synchronize time for a given type of application. Without agreement on a specific profile of IEEE 1588, time synchronization interoperability is not possible, and solutions such as time gateways must be used to synchronize time between clock profiles.

3.1.2 IEEE 802.1AS-2011

Originally developed by the IEEE 802.1 Audio/Video Bridging (AVB) task group to optimize distributed time synchronization in professional audio/video systems, IEEE 802.1AS-2011 specifies a formal profile of IEEE 1588, defining an interoperable mechanism for synchronizing time in the context of standard Ethernet. As a profile of IEEE 1588, 802.1AS-2011 configures and selects from available IEEE 1588 options, including the following:

- Use of modified Best Master Clock Algorithm (BMCA), enabling fast convergence on a grandmaster clock source
- Use of peer delay mechanism for measuring network path delay
- Transport of time synchronization messages over IEEE 802.3 (Ethernet), 802.11 (Wi-Fi), and Coordinated Shared Network (CSN) networks

- Mandatory participation by bridges, including participation in best master clock selection
- Inclusion of measurement of nearest-neighbor frequency offset in end stations and bridges for better synchronization accuracy and faster grandmaster convergence
- Specification of a performance requirement such that any two time-aware systems separated by seven or fewer hops will be synchronized to within 1 µs of each other

Building on the IEEE 1588-2008 Precision Time Protocol (PTP), the time synchronization protocol defined in 802.1AS-2011 is known as the *generalized Precision Time Protocol (gPTP)*. Since its release in 2011, gPTP has been deployed in numerous AVB networks and can serve as a foundation for markets adopting TSN capabilities, including industrial control markets, where other PTP profiles have historically been deployed.

3.1.3 IEEE 802.1AS-Rev

Following the completion of 802.1AS-2011, the IEEE 802.1 TSN task group began work on a revision to enable application of gPTP to a broader set of markets, including industrial control. Whereas 802.1AS-2011 was optimized for use in professional audio/video systems, 802.1AS-Rev adds features needed in other market segments, including the following:

- Support for redundant grandmaster clocks and time-synchronization paths, leveraging new work in IEEE 1588
- Support for multiple concurrent timescales, using multiple gPTP domains, to enable synchronization or correlation to multiple simultaneous time sources

The best master clock algorithm defined in 802.1AS-2011 and IEEE 1588-2008 supports selecting a new grandmaster clock if the current grandmaster disappears, enabling a basic level of fault tolerance. But transitioning to a new grandmaster is not a seamless operation. For some systems, especially industrial control systems requiring high uptime and depending on a continuously-available synchronization clock, the ability to seamlessly switch over to a new grandmaster can be desired or required. With this requirement in mind, 802.1AS-Rev, building on new capabilities in IEEE 1588, adds the ability to seamlessly switch to a backup grandmaster if the primary grandmaster fails.

Similarly, the best master clock algorithm in 802.1AS-2011 establishes a single time-synchronization spanning tree, resulting in a single network path between the grandmaster clock and any given slave clock. If a network path between a slave clock and the grandmaster clock is broken, the best master clock algorithm will establish a new time-synchronization spanning tree. But the switchover is not seamless, and synchronization can suffer. As with seamless grandmaster redundancy, 802.1AS-Rev adds mechanisms for seamless time-synchronization network path redundancy, enabling a higher degree of fault tolerance for industrial control systems.

In addition to grandmaster and network path redundancy, some industrial control systems can benefit from concurrent use of multiple timescales. For example, if an industrial control system uses time-aware scheduling (described in section 3.2.2) to reserve bandwidth and guarantee latency, it requires a timescale that avoids snaps or jumps (but can be based on an arbitrary epoch) for controlling a bridge's scheduled traffic shaping

function. At the same time, the higher-level application may require a timescale based on a globally-traceable source, such as GPS, for data correlation or logging. For a more detailed description of the use of multiple timescales in an industrial control system, refer to section 4.1.1. Because the underlying network transmission mechanisms and higher-level application may have conflicting timescale requirements, it can be useful to operate multiple concurrent timescales on the same physical network. Whereas 802.1AS-2011 supported only one timescale, 802.1AS-Rev adds support for multiple timescales, using distinct gPTP domains. With the ability to run multiple timescales on a common network, industrial control applications can enable predictable scheduled transmission of packets while also synchronizing a higher-level application to a globally-traceable time source, all using mechanisms defined in the context of standard Ethernet.

Some of the features of 802.1AS-Rev, especially seamless grandmaster and synchronization path redundancy, require new configuration mechanisms. For example, the best master clock algorithm cannot configure redundant synchronization paths. For such cases, 802.1AS-Rev adds the ability to configure time synchronization parameters using managed objects, via a remote network management protocol, enabling configuration paradigms in addition to the BMCA. This *external port role configuration* capability is itself a foundational mechanism for network configuration, as explained in section 3.1.3.

3.2 Quality of Service

To ensure the proper operation of time-sensitive systems, a bridged network connecting Talkers and Listeners must ensure that high-priority traffic can predictably meet bandwidth and latency requirements, especially in the presence of same-priority or best-effort traffic. To meet the requirements of these systems, the IEEE 802.1 AVB/TSN task group developed a variety of traffic-shaping mechanisms, beginning with a credit-based traffic shaper optimized for audio/video systems, and evolving to a time-aware, scheduled traffic. This section describes the operation and evolution of these quality of service mechanisms, explaining how time-aware, scheduled traffic is highly suited to meet the requirements of hard real-time industrial control systems.

3.2.1 IEEE 802.1Qav-2009

Originally developed to meet the bandwidth requirements of audio/video streams while preserving bandwidth for best effort traffic, IEEE 802.1Qav-2009, known as *Forwarding and Queueing for Time-Sensitive Streams (FQTSS)* and later integrated into 802.1Q-2011 as Clause 34, defines a credit-based traffic shaper for reserving bandwidth for A/V streams in a bridged network. In a system using FQTSS, after acquiring a reservation from the network using mechanisms discussed in section 3.4.1, end stations sourcing streams (that is, Talkers) evenly transmit frames based on quality of service parameters. These parameters establish the bandwidth reservation for the stream, governing their bandwidth usage consistent with the reservation established with the network.

In bridges, reserved A/V stream traffic is prioritized over best effort traffic as long as sufficient credits are available to transmit the higher-priority A/V data. As with end stations sourcing stream data, the credit-based traffic shaper results in stream data distributed evenly across a bridged network over time. This smooths out delivery times so that “bunching” of prioritized A/V traffic is reduced through the bridged network, bounding the latency through the network, and allowing for reduced buffer sizes in bridges. While FQTSS’ credit-based

shaper allows bandwidth reservation for high-priority traffic, it also ensures that best effort traffic will continue to flow, as the maximum interference from high-priority traffic is limited and known.

FQTSS meets the quality of service requirements for many types of applications. But the credit-based shaper can still allow interference in some circumstances, resulting in higher worst-case latency than some applications can tolerate, including some hard real-time industrial control applications. To meet the requirements of applications requiring more predictable or lower latencies, the 802.1 TSN task group developed a new traffic shaper based on time-aware scheduling, discussed in the next section.

3.2.2 IEEE 802.1Qbv-2015

As described in section 2, some industrial control systems require predictable, very low latency and cycle-to-cycle variation to meet hard real-time application requirements. In these systems, multiple distributed controllers commonly synchronize their sensor/actuator operations with other controllers by scheduling these operations in time, typically using a repeating control cycle. Traditionally, the required network bandwidth between controllers may be small because the control data needed to coordinate operations is itself small. But dedicated and costly engineered networks are used between controllers to ensure that the network does not introduce unbounded latency and path delay variation, especially if competing traffic may be present. Further, traditional network prioritization methods and the AVB credit-based shaper can help ensure quality of service of control data through a network. But these alone are sometimes not adequate, as best effort or low-priority traffic may begin to transmit just before higher-priority control data. This results in latencies up to the maximum frame size, potentially occurring at each network hop. To ensure that applications requiring very low, predictable latency can meet their quality of service requirements, the 802.1 TSN task group developed time-aware traffic scheduling, specified in 802.1Qbv-2015.

Time-aware traffic scheduling operates using a set of time-aware *gates* that immediately precede a bridge's transmission selection function as shown in Figure 7. Each *transmission gate* corresponds to a traffic class associated with a specific queue, with potentially multiple queues associated with a given port. An individual transmission gate can be either on or off, allowing transmission of a traffic class to be switched on or off depending on the requirements of the application. A configurable *gate control list* of *gate-open* and *gate-close* events determines whether or not a traffic class' gate is open or closed at any point in time, and the gate control list is executed over a configurable cycle time. The gate control mechanism is itself a time-aware PTP application operating within a bridge or end station port. This allows the execution of the gate control list to be precisely coordinated across the network, enabling scheduled transmission for a given class of traffic across a TSN-enabled network.

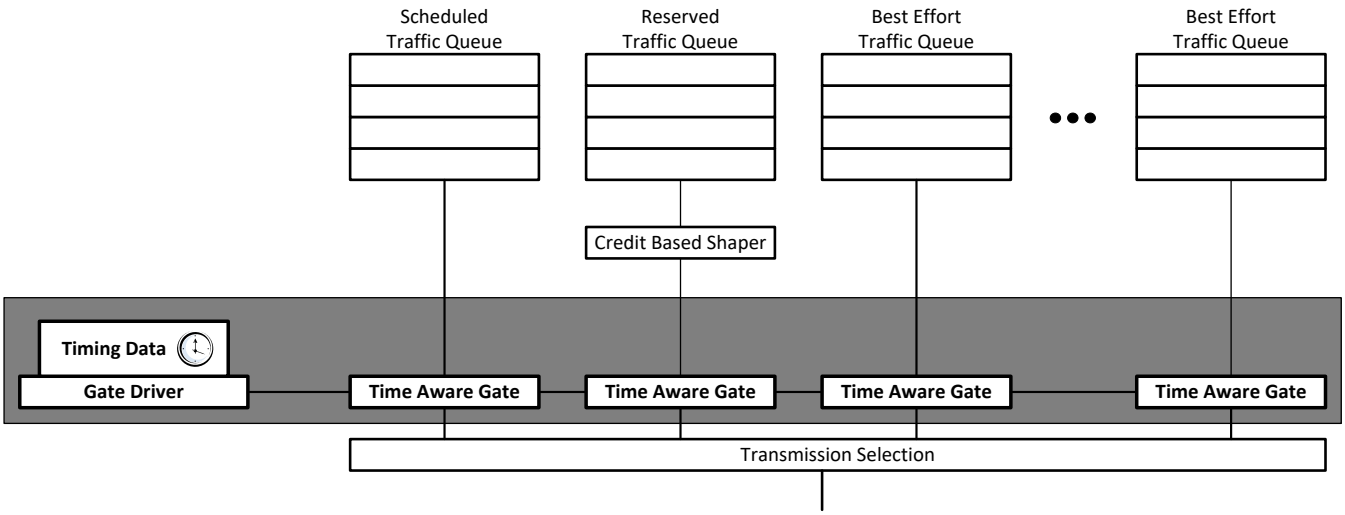

Figure 7: Transmission Selection Using the Time-Aware Traffic Scheduler

Assume bridges and end stations implement the time-aware traffic scheduler on multiple queues, including sufficient gate operations to enable on/off transmission access for each queue on a given time period. If so, an industrial control application can minimize latency and path delay variation for control traffic by configuring the gate control list so that high-priority control traffic will transmit at known time intervals. Similarly, gate operations for lower-priority or best effort traffic can be allocated in the gate control list to time slots not used by high-priority control traffic, ensuring that all traffic types will flow. Further, assume the time synchronization mechanism used to control the time-aware traffic scheduler in a bridged network is also used to schedule execution of the industrial application itself. If so, controllers distributed across a network can very tightly synchronize their sensor/actuator operations, resulting in very fast industrial control cycle times.

As the time-aware traffic scheduler is itself a time-aware application executing within a bridge or end station port's transmission selection function, it requires a PTP-based time synchronization mechanism (such as those described in section 3.1) to operate the gate control list on a port. The timing parameters associated with a port's gate control list all operate relative to this PTP time, and the time-aware scheduling mechanisms defined in 802.1Qbv assume the availability of PTP time. While 802.1Qbv does not mandate a specific PTP profile, for interoperable operation of time-aware traffic scheduling in a bridged network, an application must agree on what PTP profile to use. As 802.1AS-2011 is a profile commonly implemented in AVB/TSN-capable switches and specified as part of standard Ethernet, the architecture described in this document uses 802.1AS-2011 (and in the future, 802.1AS-rev) to operate the network schedule, as described in section 4.1.

The 802.1Qbv standard defines managed objects for configuring the parameters associated with a transmission schedule, assuming the use of a remote network management protocol to configure these parameters on each bridge in a TSN-enabled network. The next section describes a fully-centralized configuration model for configuring bridge schedules using remote network management.

3.3 Stream Differentiation

To ensure quality of service for time-sensitive streams, bridges and end stations must differentiate time-sensitive streams from other flows. For example, to ensure that a scheduled time-sensitive stream is forwarded to a bridge's scheduled traffic queue as described in section 3.2.2, the bridge must differentiate the Ethernet frames comprising the scheduled stream from other types of traffic. This section describes the fundamental mechanisms used to differentiate traffic within bridge and end stations.

IEEE 802.1Q-2014 describes the fundamental mechanisms used to identify and differentiate time-sensitive streams from other types of traffic. To support traffic differentiation, 802.1Q specifies a *Virtual Local Area Network (VLAN) tag* field, shown in Table 2.

Tag Protocol Identifier (TPID)	**Tag Control Information (TCI)**		
0x8100 (16 bits)	Priority Code Point (PCP) (3 bits)	Drop Eligible Indicator (1 bit)	VLAN Identifier (VID) (12 bits)

Table 2: IEEE 802.1Q VLAN Tag Field

The VLAN tag field is added to the header of standard layer-2 Ethernet frame, as shown in Table 3.

Layer-2 Ethernet Header					**Payload**	**CRC**
Preamble (8 octets)	Destination Address (DA) (6 octets)	Source Address (SA) (6 octets)	802.1Q VLAN Tag (4 octets)	EtherType (2 octets)	(46-1500 octets)	(4 octets)

Table 3: 802.1Q VLAN Tag in Layer-2 Ethernet Frame

IEEE 802.1Q uses *traffic classes* to differentiate traffic. The standard allows up to eight traffic classes per port, with each traffic class associated with a dedicated queue. The *Priority Code Point (PCP)* field of the VLAN Tag determines the traffic class (that is, queue) for a given Ethernet frame. Specifically, bridges are configured to map PCP values to traffic classes, and as frames arrive, the bridge directs them to the appropriate queue based on the value of the frame's PCP field and the configured PCP-to-traffic class mapping. When bridge ports implement the traffic shaping or scheduling mechanisms described in section 3.2 on their queues, an application can ensure that its stream data receives differentiated treatment by setting the PCP field to target the appropriate queue. More detail about how the PCP value is provided to an application is described in section 4.3.2.

Because the PCP field directs layer-2 Ethernet frames to queues, it is possible for end stations not participating in time-sensitive stream exchange to interfere with time-sensitive streams. For example, if a TSN-enabled network is configured to direct Ethernet frames with the PCP value of 7 to a scheduled queue, any end station,

including malicious or misconfigured end stations, can direct traffic to a bridge's scheduled queue by setting its PCP value to 7, potentially breaking the application. To protect against malicious or misconfigured end stations, bridges can enable per-stream filtering and policing, specified in IEEE 802.1Qci, to ensure non-time-sensitive streams cannot use TSN resources within a bridge, including scheduled queues. A future version of this document will include more detail on how to apply per-stream filtering and policing.

In addition to the PCP field, other fields of the Ethernet frame are also useful for differentiating time-sensitive streams. The VLAN Identifier (VID), for example, can be used to create a group of end stations that will exchange time-sensitive streams with each other. Specifically, end stations can tag Ethernet frames with a common VLAN ID, and bridges can forward the frame exclusively to end stations sharing that VLAN ID, creating a logically-isolated domain of AVB/TSN Talkers and Listeners.

3.4 Network Configuration

As described in previous sections, the fundamental elements of AVB/TSN systems, including time synchronization and traffic shaping, require dynamic configuration based on application requirements. This section describes the operation and evolution of AVB and TSN configuration mechanisms, explaining how the fully-centralized configuration model is optimized for use with TSN-enabled industrial control applications.

3.4.1 IEEE 802.1Qat

Developed as a bandwidth reservation mechanism for AVB systems using the credit-based traffic shaper described in section 3.2.1, the Stream Reservation Protocol (SRP), specified in IEEE 802.1Qat and later integrated into IEEE 802.1Q as Clause 35, defines a plug-and-play configuration mechanism to set up and tear down stream reservations.

Using SRP, stream sources (Talkers) declare bandwidth requirements prior to transmitting stream data. These bandwidth requirements are communicated to the network using *talker advertise* messages that describe stream quality of service requirements including traffic class, data rate, and accumulated worst-case latency. A talker advertise message is propagated through the bridged network toward potential Listeners as long as the bandwidth requirements can be met in bridges along the path. As the talker advertise message propagates, the accumulated latency is updated at each hop so that Listeners know the worst-case latency. If a bridge cannot support the stream's bandwidth requirement, it sends a *talker failed* message, and the stream reservation fails.

Listeners receiving talker advertise messages indicate their intent to receive stream data by sending *listener ready* messages back to the Talker. As listener ready messages propagate through the bridged network, bridges lock down the resources needed to deliver the stream data according to its quality of service requirements. When the Talker receives a listener ready message, it can begin transmitting the stream.

Using SRP, a bridged network can be configured in a distributed, plug-and-play manner, resulting in straightforward, low-maintenance network management for AVB/TSN systems. As new TSN capabilities have emerged, network configuration requirements have evolved, necessitating new configuration models to supplement SRP. For example, in a network with multiple paths between a Talker and Listener, if a spanning-

tree path is too congested to support a stream's bandwidth requirements, SRP cannot allocate resources along an alternate path, though one may exist. Similarly, configuration of network schedules, in systems using the time-aware traffic scheduling, is difficult and relatively costly to support using a distributed protocol when compared to centralized configuration options. With these new capabilities in mind, TSN is evolving to support several new configuration models as described in the next section.

3.4.2 IEEE 802.1Qcc

To meet the needs of markets beyond professional audio/video, the IEEE 802.1 TSN task group is defining new configuration models, the first of which are specified in IEEE 802.1Qcc. As described in the previous section, many new TSN capabilities, especially redundancy and traffic scheduling using time-aware traffic scheduling, are best configured using a centralized architecture, where central software controllers gather application requirements and dynamically configure the network to meet those requirements. Whereas SRP uses a distributed configuration approach, where network bandwidth reservations are established by propagating requests and responses through the entire network, 802.1Qcc adds a *fully centralized* configuration model that allows all-knowing, centralized software controllers to receive stream requirements from Talkers and Listeners and to directly configure the relevant bridges to meet those requirements. Figure 8 shows the 802.1Qcc fully centralized configuration model.

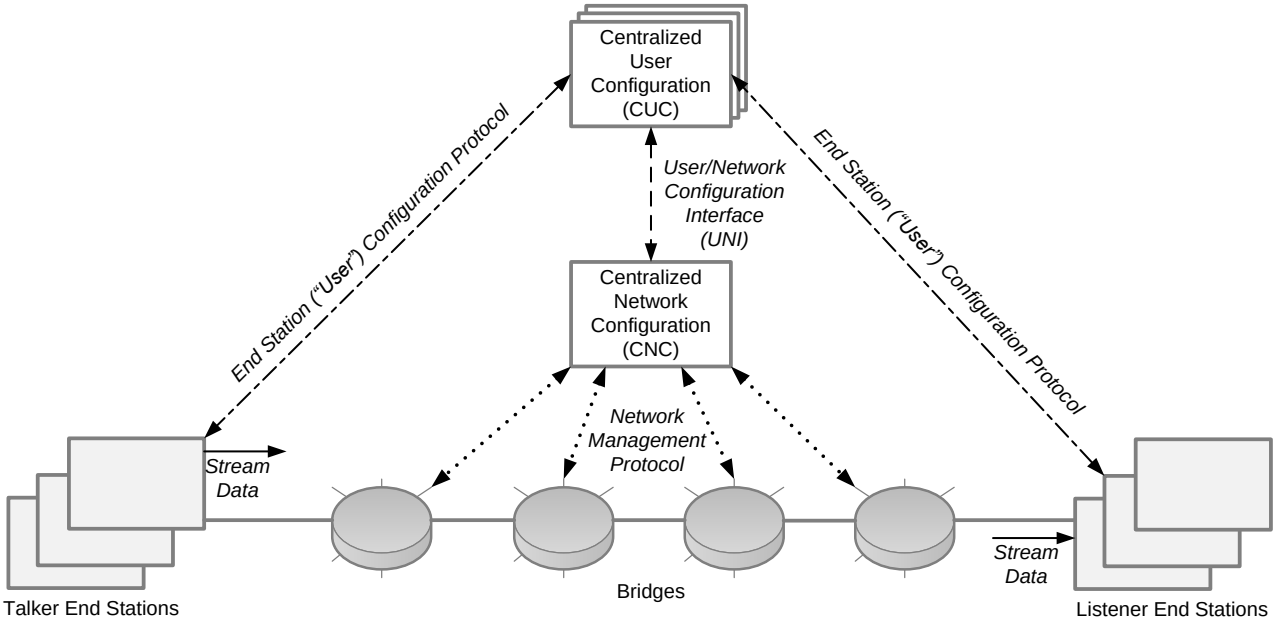

Figure 8: 802.1Qcc Fully Centralized Configuration Model

As shown in Figure 8, and common to all AVB/TSN-enabled networks, Talkers send stream data to Listeners over a bridged network. In the fully centralized configuration model, stream connections between Talkers and Listeners are established as follows:

1. Talkers and Listeners communicate their stream quality of service requirements to a *Centralized User Configuration (CUC)* entity using an end-station specific configuration protocol.

2. The CUC communicates stream quality of service requirements, on behalf of all Talker/Listener groups for its associated application, to a *Centralized Network Configuration (CNC)* entity using a *User/Network Configuration Interface* protocol.

3. The CNC performs necessary calculations to meet stream quality of service requirements in the bridged network, including calculating transmission schedules, determining data paths, etc.

4. If the CNC can satisfy the end stations' stream quality of service requirements, it performs the following actions:

    a. The CNC configures bridges in the TSN network to meet end stations' quality of service requirements by setting the appropriate bridge managed objects using a *Network Management Protocol*.

    b. The CNC returns relevant end station stream transmission information to the CUC using the User/Network Configuration Interface protocol.

    If the CNC cannot meet stream quality of service requirements, it returns an error to the CUC.

5. If the CNC indicated it could support the end stations' quality of service requirements, the CUC configures Talkers and Listeners for stream transmission, and communication begins.

As explained in the steps above, in the fully centralized model, to configure stream connections, Talkers and Listeners send their stream quality of service requirements to a Centralized User Configuration (CUC) entity using a configuration protocol specific to (and optimized for) the application running on the TSN-enabled network. For example, in an application using OPC-UA's publish/subscribe mechanism, Talkers and Listeners could communicate their requirements to a CUC using an OPC-UA-specific stream configuration protocol. These application-specific configuration protocols are defined outside of IEEE 802, enabling vendors and protocol organizations to flexibly migrate existing solutions to TSN-enabled networks, in addition to optimizing new protocol solutions for use with a TSN-enabled network.

With knowledge of its application's end station stream quality of service requirements, a CUC next communicates those requirements to a Centralized Network Configuration (CNC) entity. Whereas the CUC is responsible for configuring "users" of the network (that is, Talkers and Listeners), the CNC is responsible for configuring the TSN-enabled network itself. With knowledge of the entire network's stream requirements, including the requirements of potentially multiple applications represented by multiple CUCs, the CNC performs calculations to determine if stream quality of service requirements for a given application can be met in the TSN-enabled network, and if so, how to meet them. For example, in a system using time-aware scheduling, a CNC calculates transmission schedules such that all stream flows can meet their quality of service requirements.

The CUC requests end station quality of service requirements from the CNC using a User/Network Configuration Interface (UNI) protocol. To communicate these requirements, 802.1Qcc specifies Talker and Listener groups, enabling a CUC to join Talkers with Listeners, including the desired rate of data exchange between them. Similarly, for returning Talker/Listener transmission details from the CNC to the CUC, 802.1Qcc specifies Status groups, including parameters for stream identification and transmission scheduling, so that end station Talkers and Listeners can be configured to transmit and receive in accordance with the network's schedule.

It is important to remember that the CUC and CNC are logical functions in a network and are not intended to imply any specific hardware or software components or product embodiments. The talker end station vendor may choose to integrate the CUC function into that device such that all talkers autonomously advertise their own streams to the CNC. Similarly, a bridge manufacturer could integrate the CNC function into its bridges; in this case, network configuration will require that one bridge be defined as the CNC for that network. The remainder of this section describes the standards, data models, and protocols that have been developed to enable interoperability between bridges in this configuration.

While 802.1Qcc specifies the contents of the messages exchanged between the CUC and CNC, it does not specify the protocol used to exchange those contents. Because both the configuration message contents and the protocol must be specified for application CUCs to interoperate with a TSN-enabled network's CNC, Avnu will select or specify an appropriate User/Network Configuration Interface protocol in the *Avnu Industrial Interoperability Specification*.

After performing calculations to determine how to configure a TSN-enabled network to support a CUC's application requirements, the CNC uses a Network Management Protocol to communicate with bridges to configure the relevant *Managed Objects* on each bridge. For example, in a system using time-aware scheduling to meet quality of service requirements, the CNC uses a Network Management Protocol to configure 802.1Qbv gate control lists, described in section 3.2.2. Section 3.4.3 gives more details about the Network Management Protocol.

While 802.1Qcc depicts the CNC entity as a discrete logical entity, the CNC is a software service that can be integrated with system functionality in a variety of product embodiments. For example, the CNC can be an end station or a bridge, and it can exist remotely on a separate subnet. Similarly, while bridges are shown as discrete logical entities, a bridging function can be integrated with an end station in a product. In the 802.1Qcc fully centralized architecture, the bridge function of a bridged end station is considered part of the TSN-enabled network, configured by the CNC, and the end station is configured by the CUC.

The 802.1Qcc fully-centralized configuration model assumes a single CNC for configuring network services between any application's Talkers and Listeners. This single CNC can support configuration requests from an arbitrary number of CUCs, where each CUC is responsible for its application's stream requirements as described above. For example, consider a TSN-enabled network where two applications are sharing network resources. The first application includes Talkers and Listeners exchanging data using OPC-UA's pub/sub protocol, and the second application exchanges data using ODVA's Common Industrial Protocol (CIP). In this example system, the OPC-UA pub/sub application's CUC makes requests of the CNC on behalf of its Talkers and Listeners, and the

ODVA CIP application's CUC independently makes requests of the CNC for its Talkers and Listeners. As long as the network can support the stream requirements of both applications, the CNC will allocate network resources, including paths and bandwidth, to each CUC using the configuration process described above. Also, CUCs can request network resources from the CNC at any time; CUCs do not need to coordinate their activity. In this way, the network can dynamically support requests from multiple CUCs, as long as the underlying network resources, including the network's transmission schedule, can accommodate new stream requirements.

By separating the responsibility of configuring end stations from the responsibility of configuring a bridged TSN-enabled network, the 802.1Qcc fully centralized configuration architecture enables a *separation of concerns*, abstracting the details of network configuration from users of the network. This ability to separate the concern of network configuration from the end station application is a key benefit of *Software Defined Networking (SDN)*, an architectural model that heavily informs the 802.1Qcc fully centralized configuration model. End stations can consider the TSN-enabled network as an opaque cloud, focusing on application configuration details instead of network configuration details.

By viewing the network as an opaque cloud, end stations and the applications coordinating their communication can use TSN mechanisms, include time-aware scheduling, time synchronization, and redundancy, without direct knowledge of how to configure those mechanisms in network bridges. As new TSN mechanisms are added, such as ingress filtering and policing or additional traffic shaping mechanisms, the network can provide these services without applications needing to comprehend how to configure each new mechanism. In all cases, end stations and their applications are concerned only with their application-specific stream configuration and not with the details of network configuration.

The TSN-enabled industrial system architecture described in this document applies the 802.1Qcc fully centralized configuration model, as detailed in section 4.2. As the IEEE TSN task group adds new configuration mechanisms appropriate for industrial systems, Avnu will update this document to describe the application of those mechanisms for network configuration in TSN-enabled industrial systems.

3.4.3 IETF YANG, 802.1Qcp, and Network Management Protocols

As described in section 3.4.2, IEEE 802.1Q-2014 and its amendments define bridge managed objects to enable configuration of bridge features, including TSN features. To enable a variety of managed object data model and configuration protocol options, 802.1Q describes managed objects using multiple data modeling languages; the standard does not prescribe the use of any specific data modeling language for bridge managed objects. For example, 802.1 defines managed objects using both the *Management Information Base (MIB)* data format and the *YANG* data format. Similarly, 802.1Q assumes the use of a Network Management Protocol to remotely configure bridge managed objects, but the standard does not mandate the use of any specific management protocol, enabling a variety of protocol options for remote network management, including the *Simple Network Management Protocol (SNMP)*, *NETCONF*, and *RESTCONF*.

Because both the managed object data modeling language and Network Management Protocol must be specified for a CNC to configure TSN-enabled bridges in an interoperable manner, and the IEEE 802.1Q does not

mandate the use of any specific data model or protocol, Avnu will select appropriate data models and Network Management Protocols for bridge configuration, documenting the selections in the *Avnu Industrial Interoperability Specification.*

The Internet Engineering Task Force (IETF) currently recommends the *YANG* data modeling language for bridge managed objects, and YANG-friendly Network Management Protocols, such as *NETCONF* and *RESTCONF*, are recommended for new network management designs. In support of this effort, the IEEE 802.1 TSN task group is defining foundational mechanisms for describing bridge managed objects using YANG, specified in IEEE 802.1Qcp. Using these foundational YANG modules for 802.1Q managed objects, future 802.1 projects will define YANG modules for TSN-specific bridge features, including time synchronization, time-aware scheduling, and other configuration parameters. Once these TSN-specific YANG modules are defined, Avnu and other clients can use a YANG-friendly Network Management Protocol to configure TSN features in bridges in an interoperable manner.

4 Industrial TSN System Architecture

This section describes a system architecture and entity reference models for a TSN-enabled industrial system, applying the foundational mechanisms described in section 3 to meet the application requirements enumerated in section 2.

This architecture defines two high-level concepts for describing the system:

- *Industrial End Station (IES):* A TSN-aware end station that implements the features necessary to send or receive time-sensitive streams, shown generically as *Talker End Stations* and *Listener End Stations* in Figure 8.
- *Industrial TSN Domain*: A logical group of Industrial End Stations and IEEE 802.1 bridges which form a network with their own Working Clock Timescale (described in section 4.1.1).

Time-sensitive streams are exchanged only between Industrial End Stations that are part of the same Industrial TSN Domain, and each Industrial End Station in the network can implement multiple Talkers and/or Listeners. As described in section 3.4.2, the Industrial End Stations (with the exception of bridging functions on bridged end stations) need not be concerned with the details of the IEEE 802.1 bridged network, essentially viewing the network as an opaque cloud guaranteed to meet the requirements for their time-sensitive stream data. Future improvements, including mechanisms defined by the IETF DetNet group, will allow merging multiple Industrial TSN Domains to compose larger systems.

While the architecture described in this section does not prescribe the use of specific configuration protocols, the *Avnu Industrial Interoperability Specification* will select and/or specify the relevant protocols to ensure a variety of applications can share foundational TSN services on a common network, consistent with the goals described in section 1.

4.1 Time Synchronization Overview

As described in section 2, synchronized time is a foundational mechanism for ensuring quality of service and application synchronization in industrial systems. Specifically, synchronized time is used for scheduling time-sensitive stream communication within an Industrial TSN Domain, ensuring quality of service for streams, and Industrial End Stations also use it to perform data acquisition and control application functions such as physical input and output operations. This section describes how to apply the time synchronization mechanisms described in section 3.1 to enable industrial applications using TSN and related technologies.

4.1.1 Working Clock and Globally Traceable Timescales

As explained in sections 3.1.3 and 3.2.2, to coordinate transmission through a TSN network using time-aware scheduling, the PTP clock driving bridges' time-aware traffic schedulers must use a reliable, continuous, and monotonically increasing timescale. This PTP timescale, also called the *Working Clock Timescale*, can be based on an arbitrary epoch and is only loosely connected to a traceable source of time.

The Working Clock timescale is the primary timescale used within an Industrial TSN Domain. The Working Clock timescale provides a continuous and monotonically increasing clock to applications for performing application functions (such as input/output operations) and time-based transmission scheduling. Consistent with the Working Clock requirements stated above, the PTP grandmaster of an Industrial TSN Domain must not produce a discontinuity in time after time-sensitive data transfer within the Industrial TSN Domain starts. This may require configuring the grandmaster to defer synchronizing to a globally traceable time source (like GPS) that could go away or lose connection. For example, the grandmaster of an Industrial TSN Domain could synchronize the Working Clock Timescale to a globally traceable time source at system start-up, but it would only frequency lock (that is, syntonize) to the global traceable time source on subsequent reconnections, recording the epoch offset without adjusting it, thereby preventing a discontinuity in time.

The Working Clock timescale propagates time only within an Industrial TSN Domain. If multiple Industrial TSN Domains are used on a shared network, each will have its own independent PTP grandmaster.

If an Industrial End Station's application requires globally traceable time, the IES can implement a second timescale called the *Global Traceable Timescale*. This timescale is managed independently of the Working Clock timescale and can be implemented using a second PTP domain number, allowing the Globally Traceable timescale to operate concurrently with the Working Clock timescale on the same network, using the same PTP profile. To correlate the Working Clock timescale to the Global Traceable timescale, the Industrial End Stations should maintain the offset of the Working Clock timescale from the Global Traceable timescale at all times. The Global Traceable timescale can also be used to correlate Working Clock timescales from multiple Industrial domains, as shown in Figure 9:

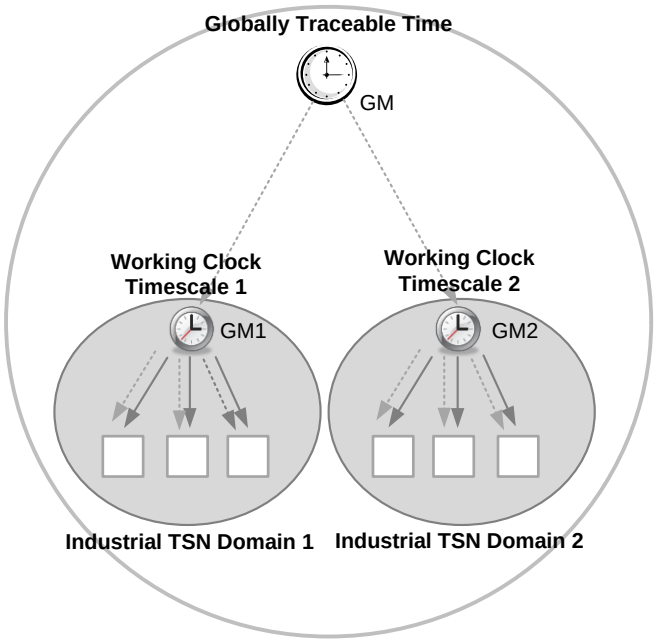

Figure 9: Correlating Working Clock Timescales Across Multiple Domains Using Globally Traceable Time

To ensure Working Clock interoperability in an Industrial TSN Domain, Industrial End Stations and TSN-capable bridges must use a common PTP profile and domain number configuration for the Working Clock timescale. To ensure interoperability for the Working Clock timescale, Avnu will select and document an appropriate Working Clock timescale configuration in the *Avnu Industrial Interoperability Specification*. As 802.1AS-2011 and 802.1AS-Rev are optimized for use with TSN networks, this architecture uses gPTP as the time synchronization mechanism for implementing an Industrial TSN Domain Working Clock timescale.

4.1.2 Support for Additional PTP Profiles

As discussed in the previous section, this architecture uses gPTP, specified in 802.1AS-Rev, as the Working Clock timescale for interoperable time synchronization in an Industrial TSN-enabled system. Though gPTP will provide time synchronization services for the network, driving the operation of time-aware traffic scheduling, Industrial End Stations may still use additional PTP profiles, including the IEEE 1588 default profile, if they desire to do so. This section describes mechanisms for supporting additional PTP profiles on Industrial End Stations while concurrently using gPTP for time synchronization in a TSN-enabled network.

One way to support concurrent PTP profiles is to run both gPTP and a second PTP service (corresponding to the additional PTP profile) on an Industrial End Station, using unique domain numbers for each PTP profile. In this configuration, an Industrial End Station uses gPTP to coordinate stream transmission and reception with the TSN-enabled network, and it uses the additional PTP profile for application-level time coordination or correlation. The application can correlate between the two timescales as the application requires. This configuration also requires the TSN-capable bridges comprising the network to support multiple PTP domains.

Another way to support concurrent operation of gPTP and another PTP profile is to use a gateway service to translate between 802.1AS gPTP in the TSN-enabled network and the additional PTP profile on the Industrial End Station. In this configuration, an Industrial End Station runs only the additional PTP profile, and the gateway service translates gPTP time to the PTP profile's time. Gateway services are discussed in more detail in section 4.7.

4.2 System Configuration Overview

As described in section 2, Industrial markets need emerging TSN capabilities to meet application requirements, including traffic scheduling and redundancy, and these TSN mechanisms require new centralized configuration models to optimize their configuration. As a result, this architecture applies the fully centralized configuration model specified in 802.1Qcc and described in section 3.4.2.

As explained in section 3.4.2, in the fully centralized configuration model, the Centralized User Configuration (CUC) entity configures end station Talkers and Listeners while the Centralized Network Configuration (CNC) entity configures bridges. In industrial applications, the presence of a CUC is common given that the applications executing on the end stations generally need coordinated configuration. For example, there is often an engineering tool that configures the rate of tasks producing and consuming data and simultaneity in I/O conversion times. To support operation over a TSN-enabled network, the fully centralized model extends existing CUC functionality to support determining the bandwidth and latency requirements of streams

exchanged between the end stations on a converged network, in addition to specifying relational constraints between end stations and streams.

Figure 10 shows the flow of configuration information between a CUC and CNC when configuring an Industrial TSN Domain for scheduled communication. Consistent with the operation of the fully centralized configuration model as described in section 3.4.2, the CUC makes requests using a User/Network Configuration Interface protocol, and the CNC replies using the same protocol.

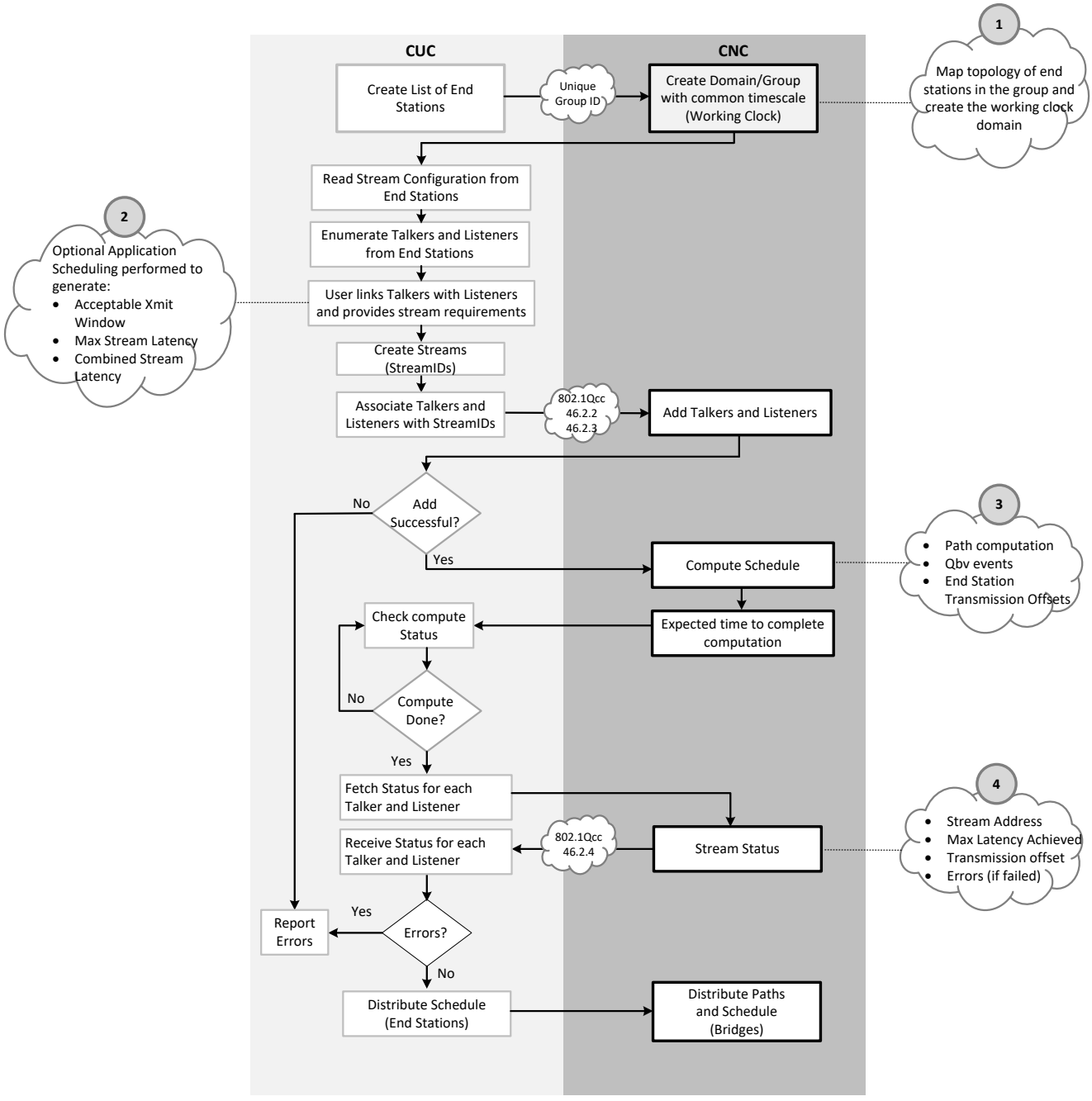

Figure 10: CUC<-->CNC Interactions for Industrial TSN Domain Configuration

Step 1: Industrial End Stations are first enumerated by a CUC using discovery mechanisms specific to the IES application. The CUC then creates an Industrial TSN Domain by grouping the Industrial End Stations that intend to exchange time-sensitive streams, and it provides this grouping to the CNC. The CNC then configures the bridges connecting these end stations such that each group has its own Working Clock. For example, when gPTP is used as the time synchronization mechanism providing the Working Clock, the *external port role configuration* mechanism defined in 802.1AS-Rev is used to configure the boundaries of the time domain and thus the Industrial TSN Domain.

Figure 11 shows an example of two Industrial TSN Domains created by the CNC. The CUC provides a unique group ID to distinguish between the groups of end stations. The CNC ensures that the bridge port that is at the edge of the group is disabled from sending the PTP *sync* and *follow-up* messages for that domain, thereby preventing the group's grandmaster's time from propagating outside the domain. This allows for groups to be totally independent of each other while still being managed by a common CNC.

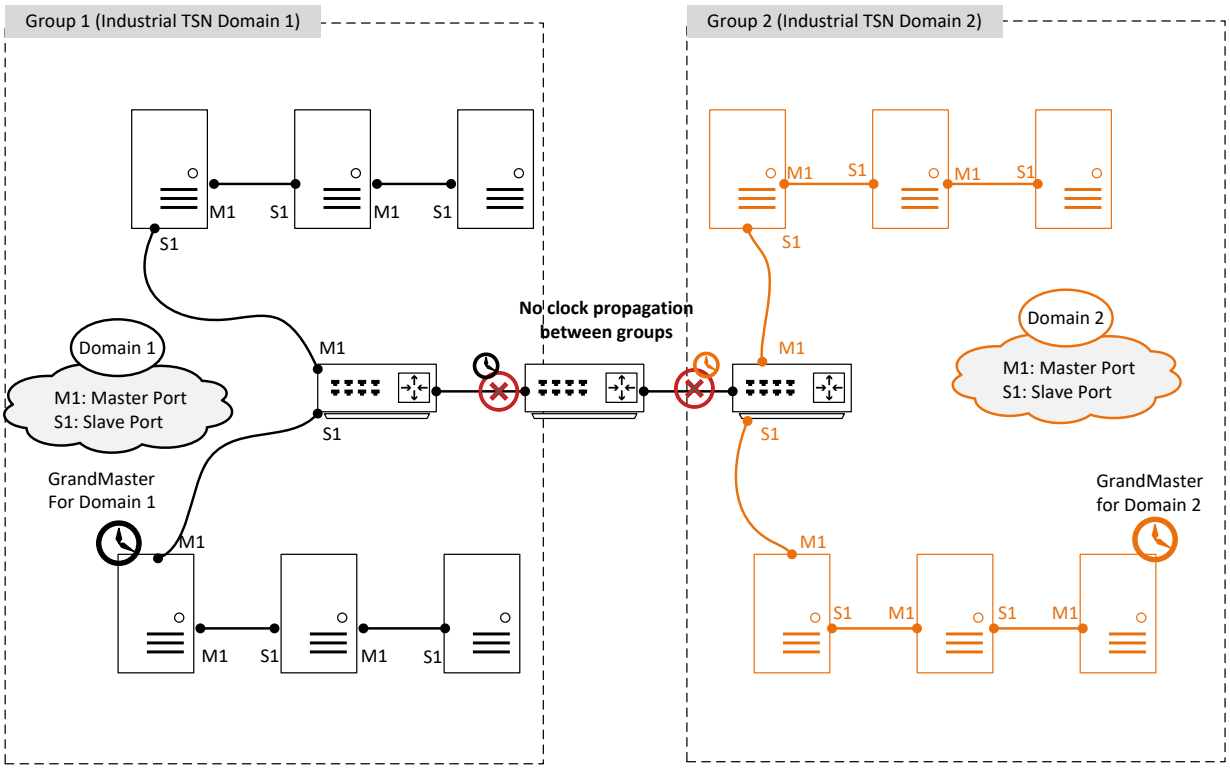

Figure 11: Multiple Working Clock Domains Configured by a Common CNC

Step 2: After establishing the Industrial TSN Domain, the CUC determines stream requirements for the Industrial End Stations in the domain. To determine stream requirements, the CUC performs the following actions:

1. The CUC reads the stream data information from Industrial End Stations, including the payload size and metadata describing the data format.
2. The CUC reads the Industrial End Station network interface configuration, including the MAC address of the network interface the end station desires to transmit/receive streams on. If an Industrial End Station has multiple ports and prefers the CNC to pick one, it publishes a list of available MAC addresses. The network interface configuration information also includes maximum transmission delay variation that the end station may experience when transmitting a stream.

The user (for example, a system designer or an automated tool) interacts with this stream data information and creates the stream requirements as follows:

1. The user links a Talker to one or more Listeners, thereby specifying the routing requirements for a stream.
2. Based on the physical process the system is controlling, the user picks a rate of data exchange (period/interval).

This user-provided information is used to create Talker and Listener groups, specified in 802.1Qcc clauses 46.2.2 and 46.2.3, respectively. The Talker and Listener groups are then provided to the CNC.

Step 3: Once all the Talkers and Listeners are added to the CNC, the CUC requests the CNC to perform path and schedule computation for the streams. The path computation is comprised of computing the route taken by the stream from the Talker to the Listener, taking into account the stream's bandwidth and latency requirements. The schedule computation comprises of calculating the bridge gate events, specified in 802.1Qbv clause 12.29, the Scheduled Offset for the Talkers, specified in 802.1Qcc clause 46.2.4.3, and the Accumulated Latency for the Listeners, specified in 802.1Qcc clause 46.2.4.2. Finally, the CNC allocates parameters to uniquely identify each stream, including the destination MAC address, VLAN ID, and PCP that the Talker should use when transmitting the stream. These stream identification parameters are part of the *InterfaceConfiguration* Status group as specified in 802.1Qcc clause 46.2.4.3.

Step 4: The stream identification and scheduling parameters allocated and computed in step 3 are returned to the CUC via a Status group, specified in 802.1Qcc clause 46.2.4. On receiving this information, the CUC checks for errors, and if satisfied, the CUC distributes the scheduled transmission offset for each stream to Talkers and the Accumulated Latency to Listeners so all can configure their network interfaces and applications. The CUC also requests the CNC to configure network paths by statically configuring the forwarding tables for the ports on each bridge between the Talkers and Listeners, and it requests the CNC to apply the computed schedule to the bridged network.

4.3 Industrial End Station Architectural Model

To support configuration using the 802.1Qcc fully centralized configuration model, an Industrial End Station must implement a variety of services and interfaces. This section describes an architectural model for Industrial End Stations, enabling support for the configuration model described in section 4.2, shown in Figure 12.

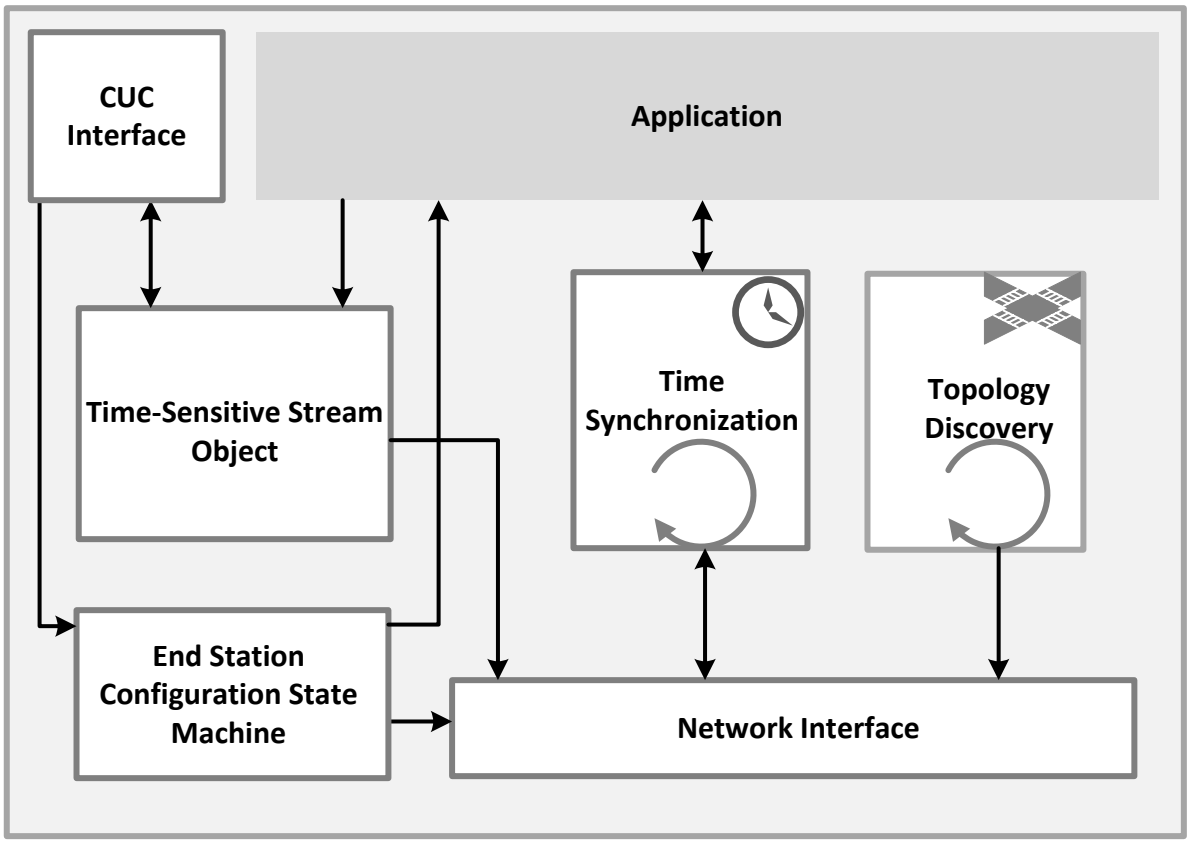

Figure 12: Industrial End Station Architectural Model

4.3.1 Centralized User Configuration (CUC) Interface

In the 802.1Qcc fully centralized configuration model, Talkers and Listeners expose an interface to their application's Centralized User Configuration (CUC) entity so that the CUC can configure the end station for stream transmission or reception. This CUC-to-Industrial End Station communication interface typically uses a push model, where the CUC initiates communication, and the end station responds with information.

Using this interface, a CUC can query the end station's stream requirements, in addition to configuring the end station's stream parameters. As the fully centralized configuration model relies on configuration states, the CUC interface is also used to control the state of the end station throughout the configuration process.

As described in sections 3.4.2 and 4.2, the protocol used to communicate between a CUC and Industrial End Station is not specified by IEEE 802.1 or other TSN-related standards, allowing vendor and application protocol groups migrate their existing CUCs (and associated protocols) for use with TSN-enabled networks.

4.3.2 Time-Sensitive Stream Object

The time-sensitive *Stream Object* is a repository of information characterizing and controlling the Industrial End Station's stream, and there is one object per Talker or Listener on the end station. The time-sensitive Stream Object is used as follows:

1. The application on the end station writes stream information to the Stream Object.
2. The CUC reads stream information from the Stream Object and provides it, along with other system requirements, to the CNC so that the CNC can schedule streams on the TSN-enabled network.
3. After receiving network transmission parameters from the CNC, the CUC writes the stream schedule (that is, Scheduled Offset and Accumulated Latency) and the stream's Period back to the Stream Object.
4. For a stream configured as a Talker, the network interface in the Industrial End Station reads the Scheduled Offset and Period information from the Stream Object and configures the transmission layer to transmit the stream.
5. For a stream configured as a Listener, the application running on the Industrial End Station reads the Accumulated Latency from the Stream Object and configures the read function in the application.

The parameters of the Stream Object are described in Table 4.

Parameter Name	Access	Description
Type	CUC: Read-only (RO) Application: Read/Write (RW) Network Interface: RO	An enumerated value indicating the type of stream. Possible values include Talker and Listener. The application running on the Industrial End Station sets this parameter.
Associated Port (MAC) Address	CUC: RW Application: RW Network Interface: RO	The physical port to serve as the egress port for the stream on the Talker and the ingress port for the stream on the Listener. The application running on the Industrial End Station can set this parameter, or the CUC can set it. If the End Station sets it, the CUC will always assign the stream to the port specified. The Industrial End Station can request the CUC to set this parameter by setting its value to zero.
Port (MAC) Address Array	CUC: RO Application: RO Network Interface: RO	A list of physical ports capable of serving as the egress port streams on a Talker or the ingress port streams on a Listener. If the Industrial End Station supports multiple physical network interfaces for time-sensitive stream

		transmission, it reports the corresponding port addresses using this parameter. Providing a list of port addresses allows a CNC flexibility in determining how to fulfill stream requests.
Stream Address	CUC: RW Application: RO Network Interface: RO	A unique ID identifying the stream within a TSN network. The CNC sets this parameter, which corresponds to the group address (that is, 48-bit multicast MAC address) used as the destination MAC Address, VID, and PCP in the time-sensitive stream frame.
Period	CUC: RW Application: RO Network Interface: RO	The cyclic transmission interval for the stream, in nanoseconds. The CUC sets this parameter based on the system requirements.
Scheduled Offset (Talker), Accumulated Latency (Listener)	CUC: RW Application: RO Network Interface: RO	For Talkers, this parameter is the offset within the period, in nanoseconds, when the stream is transmitted, corresponding to the Scheduled Offset that the CNC calculated for the stream. For Listeners, this parameter is the latest time the stream is expected to arrive within a period of the stream, in nanoseconds, corresponding to the Accumulated Latency that the CNC calculates for the stream.
Payload Size	CUC: RO Application: RW Network Interface: RO	Maximum size of stream data, in octets. This parameter is determined and set by the application layer. The payload size does not include the Layer 2 Ethernet headers, but it does include the IP based headers if IP-based flows are used for this stream.
Transmission Jitter	CUC: RO Application: RO Network Interface: RW	The maximum difference in time between a Talker's intended transmission time (configured using the Scheduled Offset parameter) and the actual transmission time, in nanoseconds. If the Industrial End Station supports multiple network

		interfaces for time-sensitive stream transmission, this parameter is set based on the port selected or the maximum value of all ports available.
Multi-Frame Stream	CUC: RO Application: RO Network Interface: RW	A Boolean value indicating whether the data mapped into this stream uses multiple Ethernet frames.
Multi-Frame Stream Data Sizes	CUC: RO Application: RO Network Interface: RW	If the stream is composed of multiple frames, this array indicates the size of data in each frame, in octets.
Data Access Type (Information Model)	CUC: RO Application: RW Network Interface: RO	An enumerated value defining the Ethernet layer the application expects to use for data access. Possible values include: L2 TSN: direct access to the Ethernet layer (Meta data). IP-based flow: The application is using the IP socket API and expects the Industrial End Station's network interface to intercept the IP flow specified (that is, Source IP Address, Source Port#, Destination IP address, Destination Port#, and Protocol type). These access models are discussed in section 4.3.3.

Table 4: Industrial End Station Time-Sensitive Stream Object

4.3.3 Network Interface

An Industrial End Station's network interface is responsible for transmitting time-sensitive application data on a TSN-enabled network. The network interface accepts stream data from the application running on the Industrial End Station, and it transmits according to parameters specified in the Stream Object described in section 4.3.2.

To support time-sensitive communication, the network interface must support enhancements for TSN, including Stream Translation and Time-Sensitive Queueing functions, as shown in Figure 13 and described below.

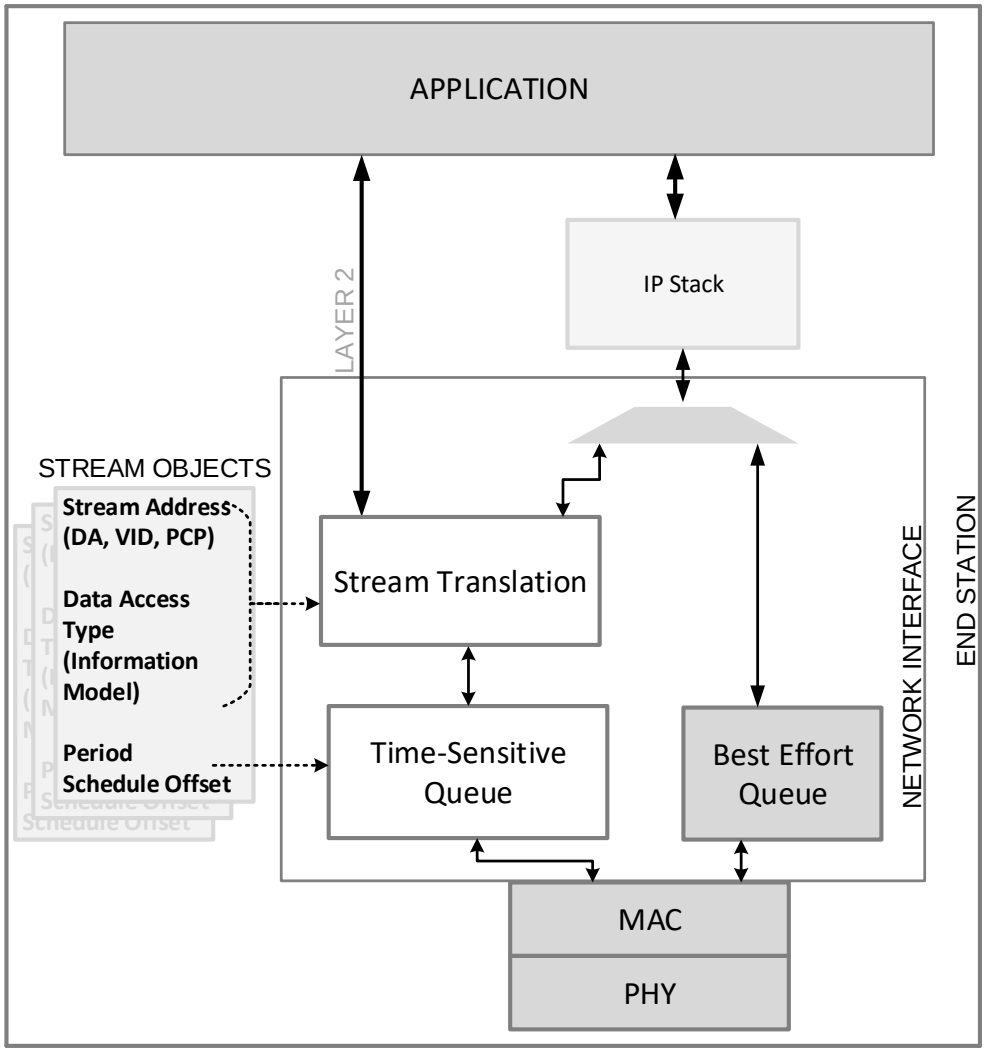

Figure 13: Network Interface Enhancements for Industrial End Stations

4.3.3.1 Stream Translation

To support time-sensitive transmission for both Layer-2 and IP-encapsulated (that is, Layer-3) stream data, an Industrial End Stations' network interface must support a Stream Translation function, described in 802.1Qcc clause 46.1.4.

The Data Access Type parameter of the Stream Object configures the Stream Translation function, which provides two types of data access to the Industrial End Station's application:

1. **Direct Layer 2 Access**: In this data access model, the application provides payload data that is inserted directly into the Ethernet frames for a stream. The application is responsible for segmenting and structuring the payload. The Stream Translation function inserts the Ethernet headers for the frames using the

Destination Address, VLAN ID and PCP in the Stream Address parameter of the Stream Object for the given stream and forwards it to the network interface's time-sensitive queue.

2. **IP Interception**: In this data access model, the application uses standard socket layer APIs and IP-based flows. The Data Access Type parameter of the Stream Object is configured by the application with the 5-tuple (that is, Source IP Address, Source Port #, Destination IP address, Destination Port #, and Protocol type) defining the IP-based flow that needs to be transmitted or received using time-sensitive streams. For transmit, the Stream Translation function intercepts this flow from the higher-level IP layer, configures it with the Stream Address (DA, VID, and PCP) in the Stream Object, and forwards it to the network interface's time-sensitive queue. On receive, the Stream Translation function reads the stream data from the time-sensitive queue, removes the Ethernet header, and forwards it to the higher-layer IP stack.

4.3.3.2 Time-Sensitive Queue

An Industrial End Station's network interface also includes a time-sensitive queue to provide a mechanism for transmitting application data according to a schedule. The time-sensitive queue is configured by the Period and Scheduled Offset parameters of the Stream Object, providing the queue with the information it needs to transmit the stream data autonomously, without software intervention.

Unlike the best-effort queue, where the application needs to write a doorbell register to begin transmission, the time-sensitive queue transmits on the configured schedule every period. If the application does not provide the stream data in time, the queue can either skip transmission and flag an error, or it can use the payload sent in the previous period, retransmitting stale data.

On the receive side, the functionality is similar to the best effort queue, where the stream data is forwarded to the application.

4.3.4 End Station Configuration State Machine

Because the fully centralized configuration model uses a multistage configuration process, including gathering requirements from end stations, calculating paths and schedules in the network, configuring streams on bridges and end stations, and starting stream transmission, Industrial End Stations must implement a state machine so that a CUC can coordinate the configuration process. This section describes an example End Station Configuration State Machine. The state machine includes support for coordinating application execution (for example, conversion of physical inputs and outputs) with time-sensitive stream transmission.

The operation of the End Station Configuration State Machine is shown in Figure 14.

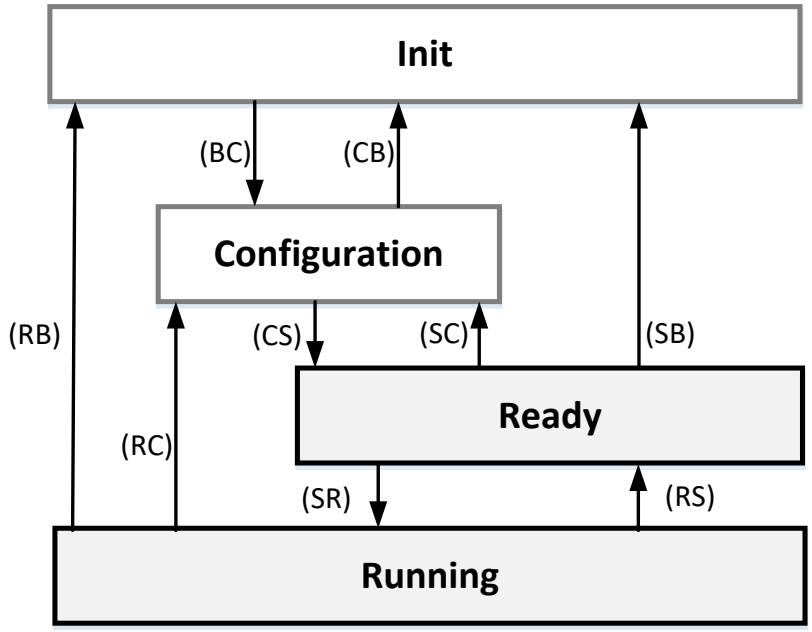

Figure 14: Industrial End Station Configuration State Machine

Table 5 describes the End Station Configuration State Machine's states and their operation.

State Name	Description
Init	This is the starting state for the communication relationship between the CUC and an Industrial End Station. In this state, the Stream Object is under the application's control, and the CUC is not yet allowed to access it. Industrial End Stations use this state for internal application initialization, gathering time-sensitive stream data requirements, and updating the Stream Object.
Configuration	In this state, the CUC is granted access to the Industrial End Stations' Stream Object so that it can send them to the CNC for schedule and path computation. Once the CUC receives schedule and path information from the CNC, it uses this state to write these parameters

	back to the Stream Object. This state assumes that the Industrial End Station's application has finished writing stream data information into its Stream Object. Because the system is being configured while in this state, time-sensitive stream transmission has not yet started. Clock synchronization procedures may have started but need not have completed.
Ready	In this state, an Industrial End Station has successfully configured its Network Interface for transmitting or receiving time-sensitive streams using the schedule and path information computed and written to the Stream Object during the Configuration state. The Industrial End Station enters this state only after the clock it uses for time-sensitive stream transmission is synchronized to the grandmaster. In this state, neither the Industrial End Station's application nor the CNC can alter the contents of the Stream Object. Time-sensitive stream transmission has not yet started.
Running	In this state, time-sensitive stream transmission begins. If an Industrial End Station's time-sensitive stream is configured to start at an explicit start time, the stream is delayed from starting until a future-time event, sent by the CUC, is received. The CUC coordinates stream transmission between Industrial End Stations comprising an application by sending the same future-time event to all Industrial End Stations after all have successfully transitioned to the Running state.

Table 5: End Station Configuration State Machine States and Descriptions

The CUC typically requests state changes, and Industrial End Stations respond by performing the necessary operations to transition to the requested state. For example, the CUC may request an Industrial End Station to transition to the Running state when it is in the Configuration state, causing the Industrial End Station to first transition to Ready and then to Running. If the requested state change fails, the Industrial End Station should respond to the CUC with an error indicating the failure.

State transitions, including state management functions that execute on the Industrial End Station as a result of a requested transition, are described in Table 6.

State Transition	Industrial End Station State Management Functions	Description
BC	Begin Configuration	The Industrial End Station enters the Configuration state, and the Stream Object is locked from further updates by the application so that the CUC can read the stream requirements.
CB	Stop Configuration	The Industrial End Station enters the Init state, and its application is free to update the Stream Object.
CS	Apply Configuration	The Network Interface takes control of the Stream Object, reading schedule and path parameters and configuring itself for scheduled transmission or reception. On this transition, the Network Interface can allocate the buffers needed for the Industrial End Station's

		application to access stream data.
SC	Disable Configuration	The Network Interface releases control of the Stream Object. On this transition, the Network Interface may deallocate any memory buffers used for the Industrial End Station's application to access stream data.
SR	Enable Stream Transmit and Receive	The Industrial End Station enables transmission and reception of time-sensitive streams.
RS	Disable Stream Transmit and Receive	The Industrial End Station disables transmission and reception of time-sensitive streams.
RC	Disable Stream Transmit and Receive, Disable Configuration	The Industrial End Station performs the actions corresponding to Disable Stream Transmit and Receive and Disable Configuration, given above.
SB	Disable Configuration, Stop Configuration	The Industrial End Station performs the actions corresponding to Disable Configuration and Stop Configuration, given above.
RB	Disable Stream Transmit and Receive, Disable Configuration, Stop Configuration	The Industrial End Station performs the actions corresponding to Disable Stream Transmit and Receive, Disable Configuration, and Stop Configuration, given above.

Table 6: End Station Configuration State Machine State Transitions and Descriptions

After transitioning all Industrial End Stations to the Running state, the CUC can provide a future-time event to serve as a synchronization point for starting time-sensitive stream transmission and execution of physical I/O operations. To determine time to use for the future-time event, the CUC can read the current time from one Industrial End Station (after it is in Running state), adding an offset, and distributing the current time plus offset to each Industrial End Station in the domain.

Similarly, to stop a distributed application, the CUC can provide a future-time event to as a synchronization point for stopping time-sensitive stream transmission and execution of physical I/O operations.

4.3.5 Time Synchronization

As motivated in sections 2 and 3.1, Industrial End Stations often require time synchronization services to coordinate network transmission, enabling quality of service on a network, and coordinating measurement and control operations for sensors and actuators associated with the Industrial End Station's application. In this architecture, Industrial End Stations use the generalized Precision Time Protocol (gPTP), standardized in IEEE 802.1AS and described in section 3.1.2, to synchronize time.

Specifically, gPTP time is used to schedule transmission through an Industrial End Station's Network Interface, coordinating traffic with the network to ensure quality of service. gPTP time is also used to drive the sample clock used to operate measurement and control functions on the Industrial End Station, synchronizing operation of sensors and actuators with the rest of the system.

Stated another way, gPTP time provides the Working Clock for the Industrial End Station as described in section 4.1.1. If an application requires a second timescale based on globally-traceable time, it can use a second, distinct gPTP domain number, so that both timescales can operate concurrently on a network.

If an application desires to use a different PTP profile, such as the IEEE 1588 default profile, to coordinate time between applications running on Industrial End Stations, it can do so as long as the PTP domain number used is distinct from the gPTP domain number used to implement the Working Clock. In this way, gPTP enables an interoperable Working Clock, controlling time-aware, scheduled network transmission, and a different PTP profile enables application coordination, all operating on the same network. Similarly, gateway services can translate time between IEEE 802.1AS and other PTP profiles. Gateway services are discussed in more detail in section 4.7.

To ensure interoperability with respect to the Working Clock, the *Avnu Industrial Interoperability Specification* will specify the use of gPTP domain number so that there are not conflicts between the Working Clock and other synchronization clocks.

With respect to state management, on receiving a request to transition from the Configuration state to the Ready state, the Industrial End Station verifies that its clock is synchronized to the grandmaster before successfully transitioning.

4.3.6 Physical Topology Discovery

To compute network paths and transmission schedules, a CNC needs to understand the network's physical topology, including the MAC addresses for ports on Industrial End Stations and bridges comprising an Industrial TSN Domain. To convey this physical topology information to a CNC, this architecture uses the Link Layer Discovery Protocol specified in IEEE 802.1AB. Section 4.5.4 provides more information about how a CNC uses LLDP to collect topology information.

To support the CNC topology discovery algorithm described in section 4.5.4, end stations run an LLDP service configured as described in Table 7. An Industrial End Station supports LLDP for each physical network port participating in time-sensitive stream communication, in cases where an IES supports multiple network ports.

LLDP Configuration Option	Value	Description
Operating Mode	Transmit Only	Industrial End Stations need only support transmitting LLDP packets. Transmission is required so that edge bridges can know the MAC address of the Industrial End Station.
Destination Address	"Nearest bridge" group MAC address (that is, 01-80-C2-00-00-0E)	This parameter indicates the LLDP packet does not propagate beyond the immediate link partner (that is, the Industrial End Stations' edge bridge).
Source Address	The Industrial End Station's sending MAC address	This parameter indicates the MAC address of the Industrial End Station's sending port.

Chassis ID Subtype	"MAC address" enumeration value (that is, 4)	This value indicates the Industrial End Station is conveying its MAC address to its edge bridge link partner.
Port ID Subtype	"Interface name" enumeration value (that is, 5)	This value indicates that the Industrial End Station should convey an RFC 2863-conformant interface name to its edge bridge link partner.
System Name TLV	Required	The Industrial End Station must support sending its fully-qualified domain name. This is helpful to display topology information to end users.

Table 7: LLDP Configuration for Industrial End Stations

To enable an interoperable discovery mechanism, the *Avnu Industrial Interoperability Specification* will specify requirements for a CNC detecting the physical topology connecting Industrial End Stations and bridges in a TSN-enabled network.

4.4 Centralized User Configuration (CUC) Architectural Model

As explained in section 3.4.2, a Centralized User Configuration (CUC) entity is responsible for collecting Talker and Listener stream requirements so that a TSN-enabled network can be centrally-configured to meet those requirements. This section defines an architectural model for a Centralized User Configuration entity.

Centralized engineering tools are commonly used to configure distributed industrial applications, and these engineering tools can perform the role of the CUC. To perform the role of a CUC for configuring Industrial End Stations, existing engineering tools can be augmented with the services and functionality shown in Figure 15.

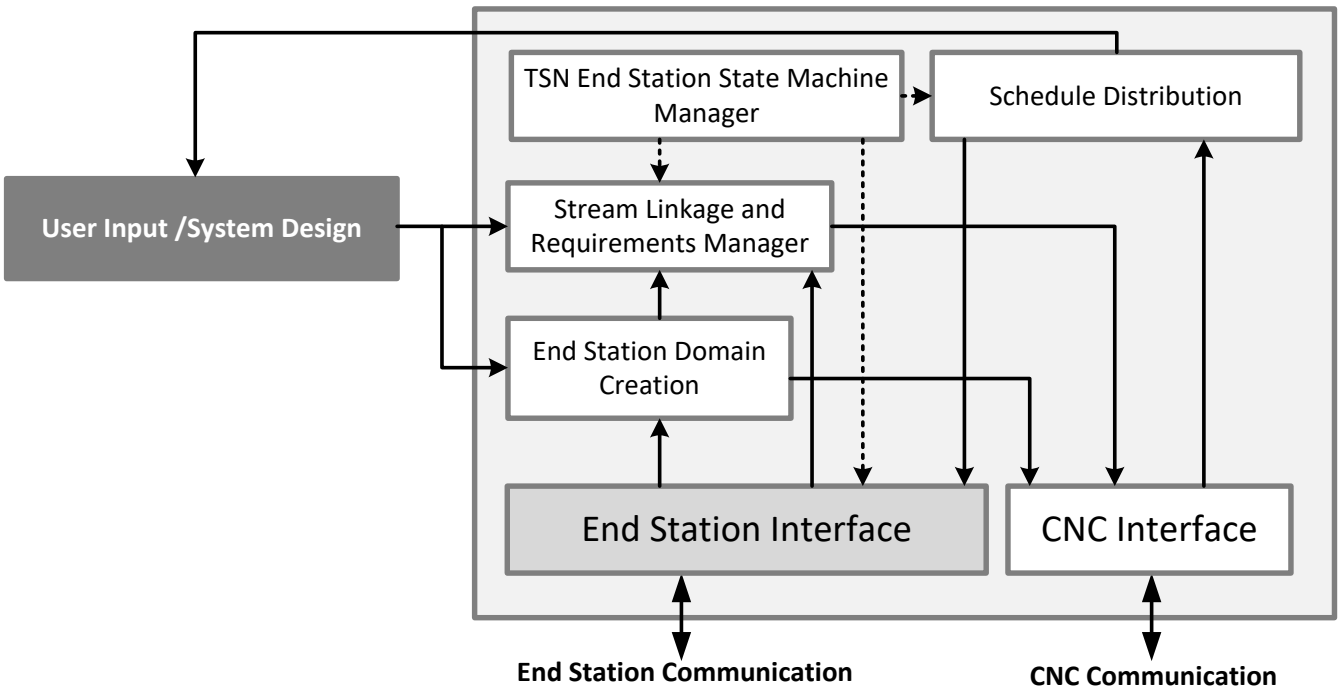

Figure 15: Centralized User Configuration Services for TSN

4.4.1 CNC Interface

As described in section 3.4.2 and shown in Figure 8, a CUC implements an interface for communicating with a CNC so that the CNC can fulfill the CUC's stream requirements on behalf of its Industrial End Stations. The 802.1Qcc standard defines the configuration data that a CUC and CNC use to communicate, and Avnu, to enable interoperability between any vendors CUC and CNC, will select an appropriate CUC←→CNC interface protocol, to be documented in the *Avnu Industrial Interoperability Specification*.

4.4.2 End Station Domain Creation

Consistent with the configuration process described in section 4.2, one role of a CUC is to group Industrial End Stations into an Industrial TSN Domain. To perform this function, the CUC takes input from the user (for example, a system designer) to specify the set of Industrial End Stations that will exchange time-sensitive streams. After the user provides this grouping, the CUC sends the group information to the CNC so that the CNC can configure a common Working Clock for the domain, thereby completing the creation of the Industrial TSN Domain. Section 4.5.3 provides more information about the CNC's domain-creation process.

4.4.3 Stream Linkage and Requirements Manager

Another responsibility of a Centralized User Configuration entity is to establish stream connections between Talkers and Listeners, linking Industrial End Stations for time-sensitive data exchange. Specifically, the user (for example, a system designer) provides input to the CUC specifying desired connections between Talkers and

Listeners, including desired application latency and bandwidth, and the CUC reads Stream Object data from the corresponding Industrial End Stations.

Using input from the user and Stream Objects, the CUC creates Talker and Listener groups, specified in 802.1Qcc clauses 46.2.2 and 46.2.3, respectively. The CUC then sends these Talker and Listener groups to the CNC so that the network can be configured to support the Industrial End Stations' stream requirements. Section 4.5.5 provides more information about the CNC's path and schedule computation process, based on the input provided from the CUC's Stream Linkage and Requirements Manager service.

4.4.4 End Station Configuration State Machine Manager

A CUC's End Station Configuration State Machine Manager allows the CUC to manage Industrial End Station state during that configuration process. Figure 16 describes the operation of a CUC's End Station State Machine Manager as it configures Industrial End Stations for time-sensitive stream exchange. Section 4.3.4 provides more information about the Industrial End Station Configuration State Machine.

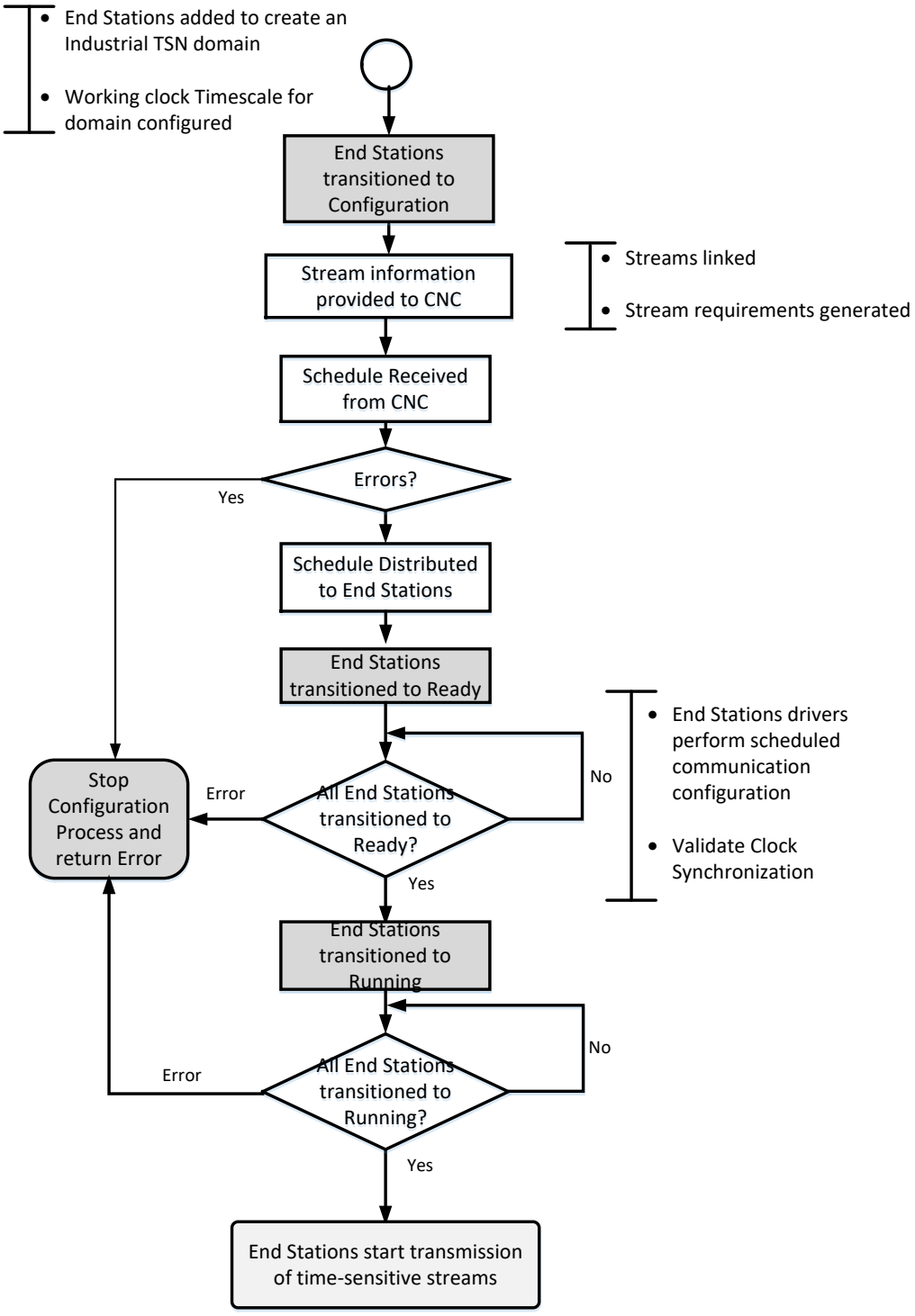

Figure 16: End Station State Machine Manager Operation During Configuration Process

4.4.5 Schedule Distribution

Another CUC responsibility is to distribute schedules to Industrial End Stations after scheduling information is received from the CNC. The CNC returns schedule information to the CUC using Status groups, defined in 802.1Qcc clause 46.2.4. If the CNC could not compute a schedule, it reports scheduling errors to the CUC, and the CUC reports the errors to both Industrial End Stations and the user.

Like all communication between a CUC and its Industrial End Stations, the CUC uses a protocol specific to and optimized for communication with its end stations, shown as the *End Station (User) Configuration Protocol* in Figure 8.

4.5 Centralized Network Configuration Architectural Model

As explained in section 3.4.2, a Centralized Network Configuration (CNC) entity is responsible for collecting Talker and Listener stream requirements, sent from a CUC on behalf of Industrial End Stations, and configuring TSN-capable bridges to meet these requirements. This section defines an architectural model for a Centralized Network Configuration entity.

The services comprising a Centralized Network Configuration entity are shown in Figure 17.

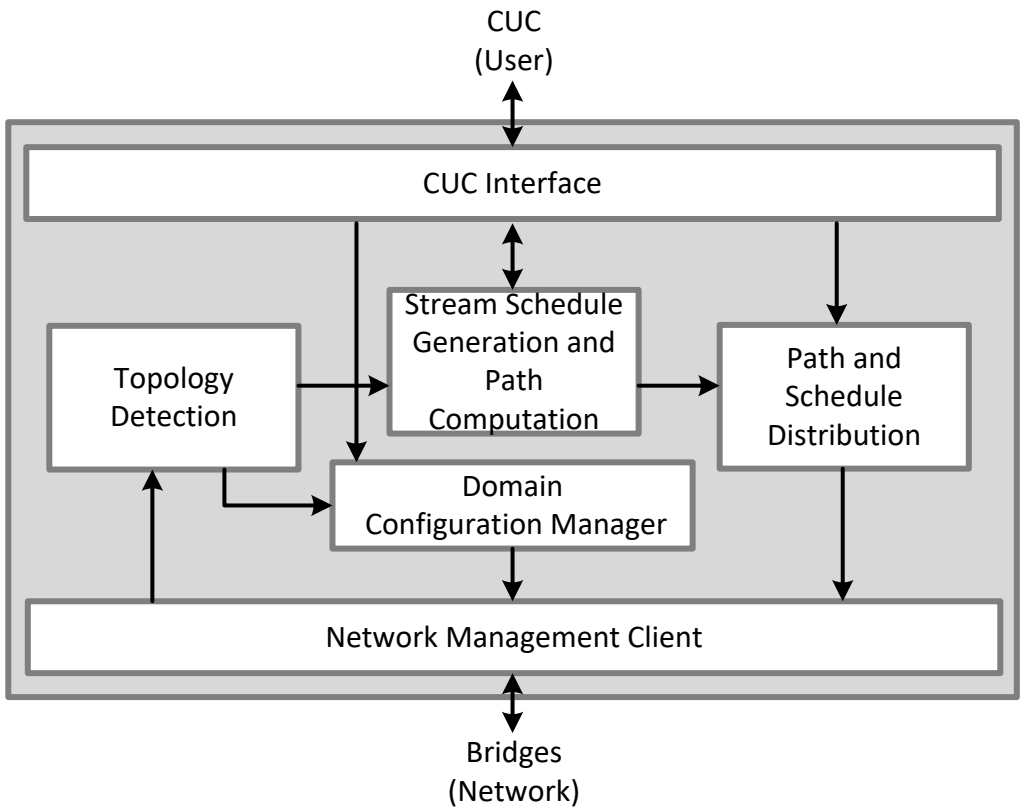

Figure 17: Centralized Network Configuration (CNC) Architectural Model

4.5.1 CUC Interface

As described in section 3.4.2 and shown in Figure 8, a CNC implements an interface for communicating with a CUC so that the CNC can fulfill the CUC's stream requirements on behalf of its Industrial End Stations. The 802.1Qcc standard defines the configuration data that a CUC and CNC use to communicate, while the protocol used to carry the configuration data is left to individual vendors or application protocol organizations to select or define.

To enable interoperability between any vendor's CUC and CNC, Avnu will select an appropriate CUC←→CNC interface protocol, to be documented in the *Avnu Industrial Interoperability Specification*.

4.5.2 Network Management Client

As described in section 3.4.2 and shown in Figure 8, a CNC configures features of TSN-capable bridges using a Network Management protocol. In this context, the CNC is the client of the Network Management protocol, and a given bridge is the server.

To enable interoperability for any CNC to remotely manage TSN-capable bridges, Avnu will select an appropriate Network Management protocol, to be documented in the *Avnu Industrial Interoperability Specification*.

4.5.3 Domain Configuration Manager

As shown in step 1 of the configuration process described in section 4.2, a CNC is responsible for receiving a group of Industrial End Stations from a CUC and configuring an Industrial TSN Domain for the end stations, including creation of a Working Clock domain. To create an Industrial TSN Domain, a CNC's Domain Configuration Manager does the following:

1. The CNC receives a group of Industrial End Stations from the CUC. The CUC communicates the list of Industrial End Stations comprising the group using the User/Network Configuration Interface protocol.
2. For the Industrial End Stations comprising the group received in step 1, the CNC verifies that the physical topology can support the creation of a gPTP time domain between the Industrial End Stations.
3. For the edge bridges comprising the Industrial TSN Domain, the CNC disables clock propagation outside of the domain. For example, the CNC sets the *ptpPortEnabled* IEEE 802.1AS Managed Objects to False, disabling propagation of gPTP synchronization messages for ports leading outside the Industrial TSN Domain.

As noted in section 3.4.2, IEEE 802.1Qcc generally defines the contents of configuration information exchanged between a CUC and CNC. The format of some configuration information used in this architecture, however, including the list of Industrial End Stations comprising a domain, is not defined in 802.1Qcc. To enable interoperable exchange of configuration information between a CUC and CNC, where the format of configuration information is not defined, Avnu will define the format, documenting it in the *Avnu Industrial Interoperability Specification*.

4.5.4 Physical Topology Discovery and Verification

To compute network paths and transmission schedules, a CNC needs to understand the network's physical topology, including the MAC addresses for ports on Industrial End Stations and bridges comprising the network. This section describes how a CNC determines the physical topology.

To enable a CNC to determine the physical topology, this architecture uses the Link Layer Discovery Protocol (LLDP), specified in IEEE 802.1AB. Using LLDP, link partners exchange port connection information with each other, including their port MAC addresses.

Figure 18 shows the LLDP-based topology discovery process.

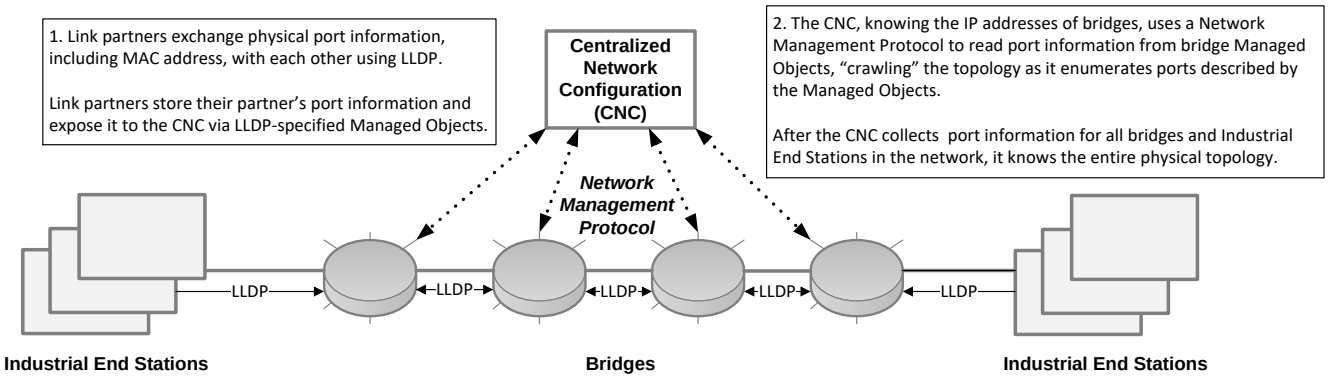

Figure 18: Physical Topology Discovery Using LLDP

LLDP operates over a single point-to-point network connection, and each link partner stores information about its peer, exposing this information using Managed Objects defined in IEEE 802.1AB. These Managed Objects can be read from bridges to build a comprehensive view of the physical topology. Applying these concepts to this architecture, a CNC determines the physical topology of network as follows:

1. Industrial End Stations and bridges run LLDP for each port involved in time-sensitive stream exchange. The precise LLDP requirements for Industrial End Stations and bridges are described in sections 4.3.6 and 4.6.3, respectively.
2. Using LLDP, link partners, especially bridges, learn the connection information for their immediate network peers. This connection information is stored in bridges and made available using Managed Objects that can be read using a Network Management protocol.
3. For each bridge in the network, and for each port on a bridge, the CNC uses a Network Management protocol to read Managed Objects describing bridge connections. Using this information, the CNC can "crawl" the entire network, building a view of the physical topology for the network.

While the CNC is crawling the network, it can also gather performance metrics and capabilities for each network path by reading bridge Managed Objects, including the 802.1Qcc standard's Bridge Delay and Propagation Delay Managed Objects. The CNC's Schedule Generation and Path Computation function described in section 4.5.5 can use this information.

This function of a CNC is also responsible for verifying the physical topology of the network prior to distributing a computed schedule. This verification step is useful in a variety of scenarios, including the following:

- The network configuration changes between the time the CNC computes a schedule and distributes the schedule to bridges
- A schedule was computed for a specific network configuration so that the schedule can be deployed to multiple instances of that network configuration, but a given instance of the network configuration does not match that which was used to compute the schedule

A future revision of this document will describe the details of physical topology verification.

4.5.5 Stream Schedule Generation and Path Computation

Given the stream requirements from Industrial End Stations, provided by the end stations' CUC, the CNC computes a transmission schedule and network paths.

As shown in Figure 10 and described in section 4.2, a CNC receives Talker and Listener groups, specified in 802.1Qcc clauses 46.2.2 and 46.2.3, from the CUC. It also receives physical topology information and performance metrics for the network from the Physical Topology Discovery function described in section 4.5.4.

Using this information, the CNC computes the optimal path for each stream, including the schedule for bridge gate-control events for all bridges along the stream's path. The CNC also computes the maximum latency for each stream and the transmission offset for the Talker. Finally, the CNC allocates the group Destination MAC Address, VLAN ID, and PCP for each stream.

The schedule and stream identification parameters for each stream are returned to the CUC via an 802.1Qcc Status group, and the CUC can then use this information to configure the schedule on the end stations.

4.5.6 Path Configuration and Schedule Distribution

After computing the transmission schedule as described in section 4.5.5, a CNC configures paths between Talkers and Listeners and distributes schedules to bridges. As with all communication between the CNC and bridges, the CNC uses a Network Management Protocol to configure bridge Managed Objects. To configure paths and distribute schedules, the CNC performs the following operations:

1. The CNC, using the stream identification parameters computed as described in section 4.5.5, configures bridge forwarding tables, statically nailing down the path for each stream.
2. The CNC configures bridge gate events, specified in 802.1Qbv clause 12.29, based on the schedule computed as described in section 4.5.5.

Once paths and schedules in bridges, the TSN-enabled network is ready to service streams between Talkers and Listeners.

4.6 Bridge Requirements

As shown in Figure 8 and described in section 3.4.2, bridges connect Talkers and Listeners and comprise the network in the fully centralized configuration model. The 802.1 standards, especially 802.1Q-2014 and its amendments, specify an architectural model and behavioral policies for TSN-enabled bridges. This section enumerates the 802.1 and related bridge requirements necessary for a bridge to interoperate in the architecture described in this document.

4.6.1 Time Synchronization

As described in section 4.1.1, bridges support 802.1AS (that is, gPTP) to implement the Working Clock Timescale. For applications that need a second timescale, such as a Global Traceable Timescale, bridges can support a second PTP domain, as described in section 4.1. Alternatively, for timescales other than the Working Clock Timescale, in addition to 802.1AS, a bridge can support other PTP profiles as described in section 4.1.2.

4.6.2 Traffic Scheduling

As described in section 3.2.2, this architecture uses time-aware traffic scheduling to enable quality of service for time-sensitive stream communication between Talkers and Listeners. To enable time-aware scheduling, bridges support the mechanisms defined in IEEE 802.1Qbv-2015, including the following:

- At least two traffic queues per port, allowing traffic differentiation between high-priority traffic and low-priority or best-effort traffic
- At least two gate-control events per queue, allowing a queue to be switched on or off according to a schedule, using the bridge's gate control list

4.6.3 Physical Topology Discovery

To support the CNC topology discovery algorithm described in section 4.5.4, bridges use LLDP, configured as described in Table 8. A bridge supports LLDP for each physical network port participating in time-sensitive stream communication.

LLDP Configuration Option	Value	Description
Operating Mode	Transmit and Receive	Bridges support both transmitting and receiving LLDP packets.
Destination Address	"Nearest bridge" group MAC address (that is, 01-80-C2-00-00-0E)	This parameter indicates that the LLDP packet does not propagate beyond the immediate link partner.
Source Address	The Industrial End Station's sending MAC address	This parameter indicates the MAC address of the bridge's sending port.

Chassis ID Subtype	"MAC address" enumeration value (that is, 4)	This value indicates that the bridge is conveying its MAC address to its edge bridge link partner.
Port ID Subtype	"Interface name" enumeration value (that is, 5)	This value indicates that the bridge should convey an RFC 2863-conformant interface name to its edge bridge link partner.
System Name TLV	Required	The bridge must support sending its fully-qualified domain name. This is helpful to display topology information to end users.

Table 8: LLDP Configuration for Bridges

To enable an interoperable discovery mechanism, the *Avnu Industrial Interoperability Specification* will specify requirements for a CNC detecting the physical topology connecting Industrial End Stations and bridges in a TSN-enabled network.

4.6.4 Multiple Spanning Tree Protocol (MSTP)

While a CNC statically configures the network paths for time-sensitive streams as described in section 4.5.6, a bridge must also prevent loops in network paths for best-effort communication. To ensure the network is free of loops for best-effort traffic, bridges use the Multiple Spanning Tree Protocol (MSTP), specified in IEEE 802.1Q. In addition, bridges should support the Traffic Engineering Multiple Spanning Tree Instance Identifier (TE-MSTID) so that the VLAN IDs assigned to time-sensitive streams are not under control of either a spanning tree protocol or IS-IS.

4.6.5 Network Management

As described in section 3.4.3, bridges expose configuration options using Managed Objects, and a CNC configures bridge Managed Objects using a Network Management Protocol. Consistent with the requirements stated above, a bridge must support Managed Objects for the following mechanisms:

- Time synchronization Managed Objects, specified in IEEE 802.1AS-Rev
- Traffic scheduling Managed Objects, specified in IEEE 802.1Qbv-2015
- Physical topology discovery Managed Objects, specified in IEEE 802.1AB
- Bridge performance metrics, specified in IEEE 802.1Qcc

Consistent with section 3.4.3, bridges represent Managed Objects using the YANG data modeling language. To expose YANG-based Managed Objects for configuration using a CNC, a bridge also implements a Network Management Protocol server.

To enable interoperability for any CNC to remotely manage TSN-capable bridges, Avnu will select an appropriate Network Management protocol, to be documented in the *Avnu Industrial Interoperability Specification*.

4.7 Gateway Services

As described in section 2, gateway services play an important role when integrating existing industrial devices and systems with TSN-enabled networks. Specifically, to enable a migration path for existing devices and systems, TSN-enabled systems can use time synchronization gateways to translate between an 802.1AS-based Working Clock or Universal Clock, used in the TSN-enabled network and described in section 4.1, and other IEEE 1588 PTP profiles, used in existing industrial devices and infrastructure. Similarly, gateway services can also provide a way for existing industrial application data, used in non-TSN-enabled industrial networks, to receive quality-of-service benefits from a TSN-enabled network by translating the existing industrial application data for use in a TSN-enabled scheduled network.

A future version of this document will describe time synchronization and data translation gateway services.

Appendix A: Configuration of Example TSN-enabled Industrial Systems

This section describes the use of the system architecture described in section 4 on example industrial control systems.

A.1 Scheduling Example

This section describes how to configure an example scheduled TSN network, including the following:

- How Stream Objects are configured on Industrial End Stations (described in section 4.3.2)
- How a CUC creates 802.1Qcc Talker and Listener groups (described in section 3.4.2 and specified in 802.1Qcc clauses 46.2.2 and 46.2.3)
- How a CNC can compute a schedule based on application requirements specified in 802.1Qcc Talker and Listener Groups

This example focuses on the following interactions:

- Talker and Listeners communicating their stream quality of service requirements to a CUC using Stream Objects, described in section 4.3.2
- The CUC requesting stream quality of service requirements from the CNC, described in section 4.4.3
- The CNC calculating a network schedule and returning the relevant Talker/Listener transmission parameters to the CUC, described in section 4.5.5
- The CUC configuring Talker and Listeners via the Stream Object, described in section 4.4.5

Future versions of this document will augment this example to illustrate other aspects of system configuration, including bridge configuration.

A.1.1 Controller-to-Controller Communication using TSN

In this example, three PLCs, each with its own dedicated, non-TSN fieldbus connecting I/O devices, controls different aspects of a distributed industrial application. The PLCs connect to each other using TSN-enabled Ethernet, and the system is configured using the fully centralized configuration model as described in section 4.2. PLCs fulfill the role of Industrial End Stations, described in section 4.3.

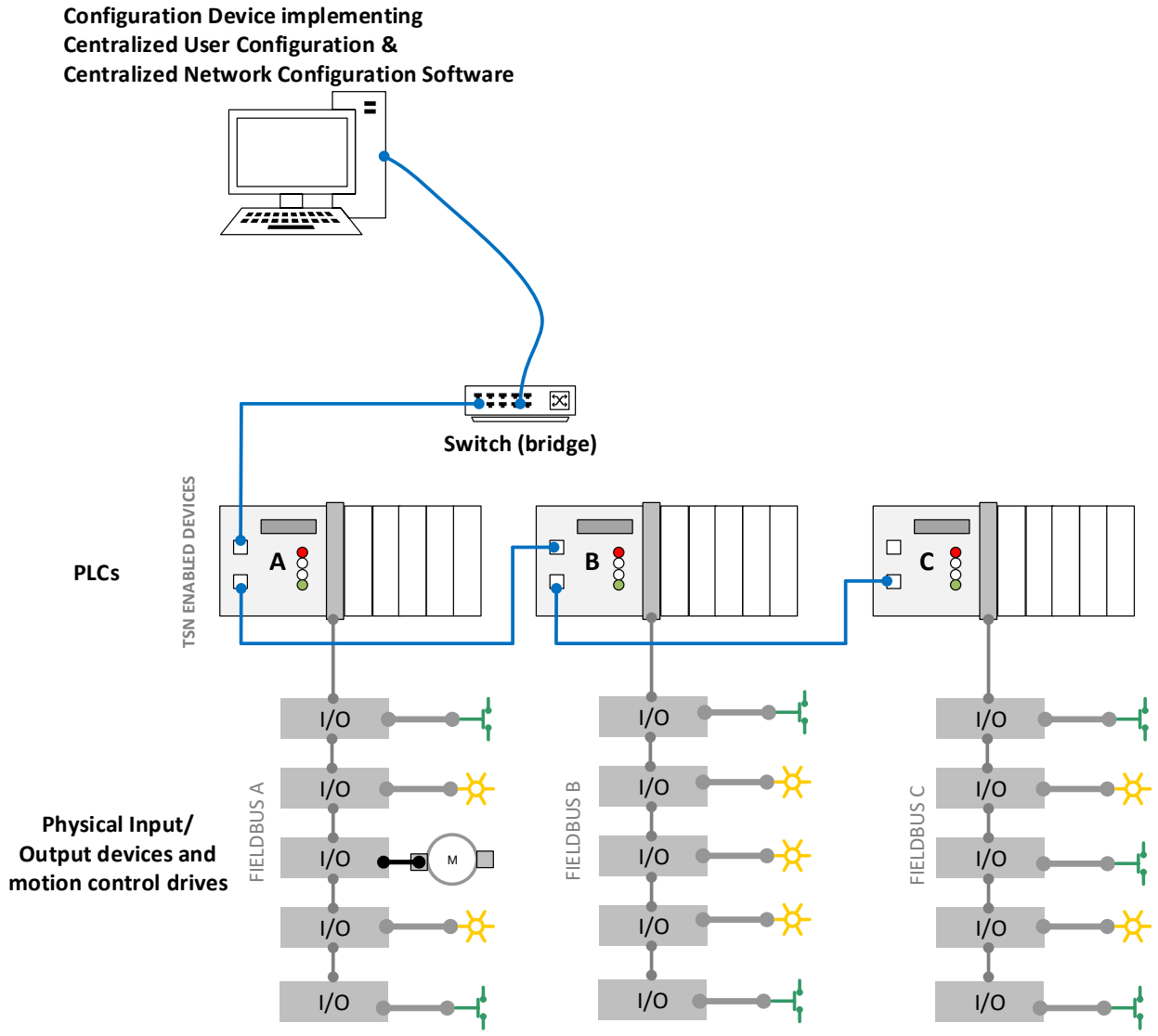

Figure 19: Control-Level Example System

In this example, PLC A samples inputs from its fieldbus system and provides those inputs to PLC B via a time-sensitive stream (Stream A->B). PLC B uses these input values along with additional inputs from its own fieldbus to compute an output value that is sent to PLC C using another time-sensitive stream (Steam B->C). Finally, PLC C writes this output value to an I/O device connected to its own fieldbus.

A.1.1.1 PLC Stream Configuration

This section describes the configuration of each PLC in the example controller-to-controller application. Specifically, this section describes how each PLC configures its Stream Object, described in section 4.3.2, so that the CUC can create 802.1Qcc Talker and Listener groups to communicate stream requirements to the CNC. Note that, for clarity, the Stream Objects tables listed in this section contain only the parameters set by the

Application and Network Interface functions of the PLC itself, to be consumed by the system's CUC. Stream Object parameters set by the CUC, after a schedule is computed by the CNC, are described in section A.1.1.4.

This example assumes the following:

- The application executing on each PLC uses the traditional scanning architecture, where the data from the I/O devices on the fieldbus and data from the time-sensitive streams are updated before executing the control logic (program/tasks). This IO Scan function runs in the beginning of every PLC cycle, as shown in Figure 20
- PLCs communicate using layer-2 TSN frames for time-sensitive streams, adding an overhead of 22 bytes to the data payload

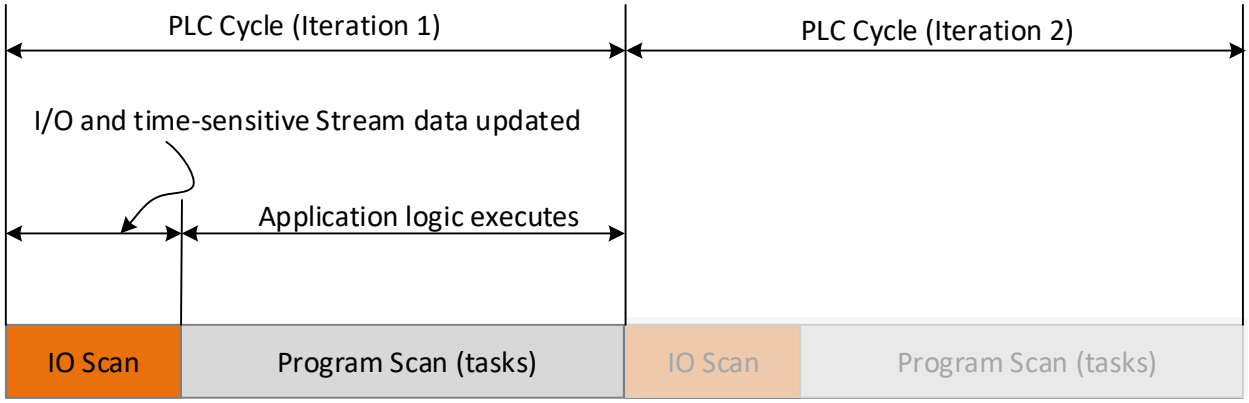

Figure 20: PLC Execution Model

To communicate its input values to PLC B, the application on PLC A uses a 30-byte data payload, configuring a Talker Stream Object, as shown in Table 9.

Parameter Name	Value
Type	Talker
Associated Port (MAC) Address	34-17-EE-C9-F5-F6
Payload Size	42 bytes (30 payload + 22 overhead)
Transmission Jitter	2
Multi-Frame Stream	No

Data Access Type (Information Model)	L2 TSN

Table 9: Talker Stream Object on PLC A for Stream A→B

To receive PLC A's input values, the application on PLC B configures a Listener Stream Object, as shown in Table 10.

Parameter Name	Value
Type	Listener
Associated Port (MAC) Address	32-17-EE-C9-F6-F6
Payload Size	42 bytes (30 payload + 22 overhead)
Multi-Frame Stream	No
Data Access Type (Information Model)	L2 TSN

Table 10: Listener Stream Object on PLC B for Stream A→B

To communicate its input values to PLC C, the application on PLC B uses a 50-byte payload, configuring a Talker Stream Object, as shown in Table 11.

Parameter Name	Value
Type	Talker
Associated Port (MAC) Address	32-17-EE-C9-F6-F6
Payload Size	72 bytes (50 payload + 22 overhead)
Transmission Jitter	2
Multi-Frame Stream	No

Data Access Type (Information Model)	L2 TSN

Table 11: Talker Stream Object on PLC B for Stream B→C

To receive PLC B's input values, the application on PLC C configures a Listener Stream Object, as shown in Table 12.

Parameter Name	Value
Type	Listener
Associated Port (MAC) Address	31-17-EE-C9-F4-F6
Payload Size	72 bytes (50 payload + 22 overhead)
Multi-Frame Stream	No
Data Access Type (Information Model)	L2 TSN

Table 12: Listener Stream Object on PLC C for Stream B→C

A.1.1.2 CUC Stream Configuration

This section describes the operation of the CUC to request time-sensitive communication on behalf of the PLC streams configured in section A.1.1.1.

The CUC reads parameters from Stream Objects described in section A.1.1.1 to create 802.1Qcc Talker and Listener groups to send to the CNC. These CNC uses these Talker and Listener groups to compute network paths and a transmission schedule.

This example assumes that a cycle time (that is, interval) of 1 ms is sufficient to solve the distributed application. Further, this example assumes that streams will be transmitted and received in the same cycle, resulting in the maximum latency for a stream to be the cycle time (that is, 1 ms). These assumptions produce results in less than a seven-cycle input to output reaction time for the example distributed application.

To request stream requirements for Stream A→B, the CUC sends a Talker group to the CNC, as shown in Table 13.

Parameter Name	Value		
StreamIdentifier	34-17-EE-C9-F5-F6-00-01		
StreamRank	True		
EndStationInterfaces	PortAddress		34-17-EE-C9-F5-F6
TrafficSpecification	Interval		1/1000
	MaxFramesPerInterval		1
	MaxFrameSize		42
	TransmissionSelection		0
	Scheduled	EarliestTransmissionOffset	0
		LatestTransmissionOffset	1000000
		Jitter	2000
NetworkRequirements	NumSeamlessTrees		1
	MaxLatency		1000000
InterfaceCapabilities	VlanTagCapable		True
	CB-Capable		True

Table 13: Talker Group for Stream A→B

To request stream requirements for Stream A→B, the CUC sends a Listener group to the CNC, as shown in Table 14.

Parameter Name	Value	
StreamIdentifier	34-17-EE-C9-F5-F6-00-01	
EndStationInterfaces	PortAddress	32-17-EE-C9-F6-F6
NetworkRequirements	NumSeamlessTrees	1

	MaxLatency	1000000
InterfaceCapabilities	VlanTagCapable	True
	CB-Capable	True

Table 14: Listener Group for Stream A→B

To request stream requirements for Stream B→C, the CUC sends a Talker group to the CNC, as shown in Table 15.

Parameter Name	**Value**		
StreamIdentifier	32-17-EE-C9-F6-F6-00-02		
StreamRank	True		
EndStationInterfaces	PortAddress		32-17-EE-C9-F6-F6
TrafficSpecification	Interval		1/1000
	MaxFramesPerInterval		1
	MaxFrameSize		72
	TransmissionSelection		0
	Scheduled	EarliestTransmissionOffset	0
		LatestTransmissionOffset	1000000
		Jitter	2000
NetworkRequirements	NumSeamlessTrees		1
	MaxLatency		1000000
InterfaceCapabilities	VlanTagCapable		True
	CB-Capable		True

Table 15: Talker Group for Stream B→C

To request stream requirements for Stream B→C, the CUC sends a Listener group to the CNC, as shown in Table 16.

Parameter Name	Value	
StreamIdentifier	32-17-EE-C9-F6-F6-00-02	
EndStationInterfaces	PortAddress	31-17-EE-C9-F4-F6
NetworkRequirements	NumSeamlessTrees	1
	MaxLatency	1000000
InterfaceCapabilities	VlanTagCapable	True
	CB-Capable	True

Table 16: Listener Group for Stream B→C

A.1.1.3 CNC Stream Configuration

The CNC uses the Talker and Listener groups described in section A.1.1.2, in addition to topology information and performance metrics gathered from the network's bridges, to create the schedule for each stream. The CNC returns the stream schedule information to the CUC using the Status groups described in this section. Note that the details of how the CNC computed the schedule are not included in this example.

To communicate the schedule for Stream A→B, the CNC returns a Listener group to the CUC, as shown in Table 17.

Parameter Name	Value	
StreamIdentifier	34-17-EE-C9-F5-F6-00-01	
StatusInfo	TalkerStatus	Ready
	ListenerStatus	Ready
	FailureCode	0
AccumulatedLatency	210000	
InterfaceConfiguration.InterfaceList[0]	DestinationMacAddress	ab:ad:ba:be:03:e8

	VlanId	3000
	PriorityCodePoint	5
	ScheduledOffset	10000

Table 17: Status Group for Stream A→B

To communicate the schedule for Stream B→C, the CNC returns a Listener group to the CUC, as shown in Table 18.

Parameter Name	**Value**	
StreamIdentifier	31-17-EE-C9-F4-F6-00-02	
StatusInfo	TalkerStatus	Ready
	ListenerStatus	Ready
	FailureCode	0
AccumulatedLatency	210000	
InterfaceConfiguration.InterfaceList[0]	DestinationMacAddress	ab:ad:ba:be:03:e9
	VlanId	3000
	PriorityCodePoint	5
	ScheduledOffset	10000

Table 18: Status Group for Stream B→C

A.1.1.4 PLC Schedule Configuration

After receiving the Status groups described in section A.1.1.3 from the CNC, the CUC can configure the schedule on each PLC using the PLC's Stream Object. This section describes how each PLC's Stream Object is updated by the CUC to configure the stream's schedule. Specifically, the CUC updates the Stream Object's Stream Address, Period, Scheduled Offset, and Accumulated Latency parameters, using the corresponding values from 802.1Qcc Status Groups, as shown in the tables below. Note that, for clarity, the Stream Objects tables listed in this section contain only the parameters set by the CUC. Stream Object parameters set by the PLC's Application and Network Interface functions are described in section A.1.1.1. The complete Stream Object definition is described in section 4.3.2.

To configure the Talker schedule on PLC A for stream A→B, the CUC configures a PLC A's Talker Stream Object, as shown in Table 19.

Parameter Name	**Value**	
Stream Address	Group MAC Address	ab:ad:ba:be:03:e8
	VlanId	3000
	PriorityCodePoint	5
Scheduled Offset	10000	
Period	1000000	

Table 19: Talker Stream Object on PLC A for Stream A→B

To configure the Listener's schedule on PLC B for stream A→B, the CUC configures a PLC B's Listener Stream Object, as shown in Table 20.

Parameter Name	**Value**	
Stream Address	Group MAC Address	ab:ad:ba:be:03:e8
	VlanId	3000
	PriorityCodePoint	5
Accumulated Latency	210000	
Period	1000000	

Table 20: Listener Stream Object on PLC B for Stream A→B

To configure the Talker's schedule on PLC B for stream B→C, the CUC configures a PLC B's Talker Stream Object, as shown in Table 21.

Parameter Name	**Value**	
Stream Address	Group MAC Address	ab:ad:ba:be:03:e9
	VlanId	3000

	PriorityCodePoint	5
Scheduled Offset	10000	
Period	1000000	

Table 21: Talker Stream Object on PLC B for Stream B→C

To configure the Listener's schedule on PLC C for stream B→C, the CUC configures a PLC C's Listener Stream Object, as shown in Table 22.

Parameter Name	**Value**	
Stream Address	Group MAC Address	ab:ad:ba:be:03:e9
	VlanId	3000
	PriorityCodePoint	5
Accumulated Latency	210000	
Period	1000000	

Table 22: Listener Stream Object on PLC B for Stream B→C

A.1.1.5 System Schedule Generation

After each PLC's Stream Objects are configured as described in section A.1.1.4, the network and PLCs are ready to start time-sensitive stream communication, and the distributed application can begin. This section describes the schedule and operation of the distributed application.

Given the assumptions described in section A.1.1.2, the distributed application produces its output after seven iterations (cycles). Each iteration is described below:

- **Iteration 1**: PLC A's IO Scan function copies the input data, read its fieldbus, to PLC A's Network Interface.
- **Iteration 2**: PLC A transmits, and PLC B receives, the input data from Iteration 1.
- **Iteration 3**: PLC B copies the input data from Iteration 2 to application memory, and PLC B's application uses this input data to compute a new output data value to send to PLC C.
- **Iteration 4**: PLC B copies the output data value computed in Iteration 3 to its Network Interface.
- **Iteration 5**: PLC B transmits, and PLC C receives, the output data from Iteration 4.
- **Iteration 6**: PLC C copies the output data from Iteration 5 to its fieldbus memory.
- **Iteration 7**: PLC C's fieldbus writes the output data from Iteration 6 to the physical output.

The distributed application's system schedule, including the seven iterations described above, are shown in Figure 21.

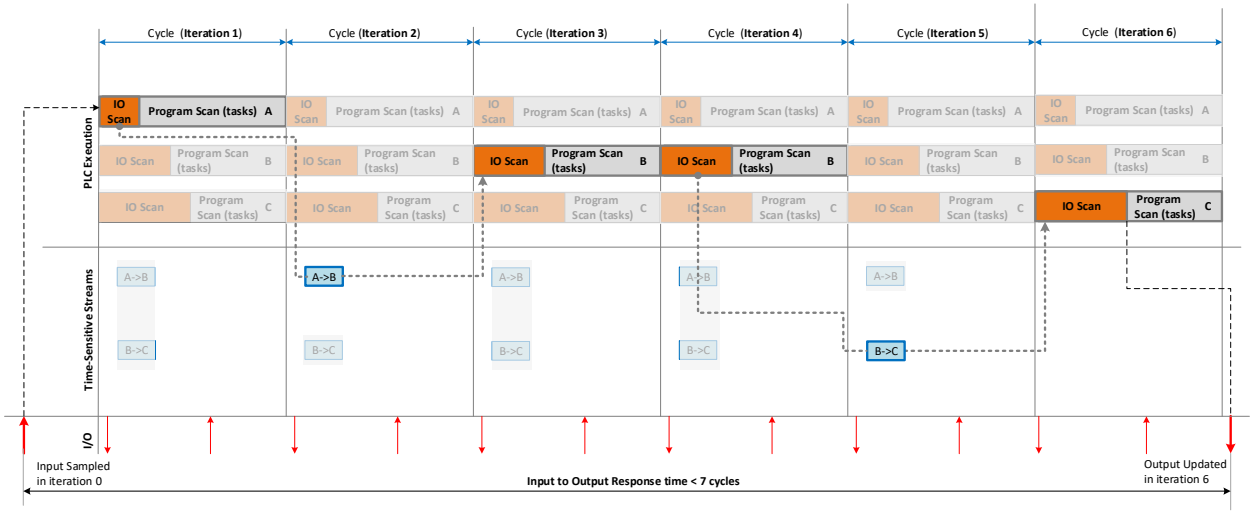

Figure 21: Distributed Application System Schedule