This document provides a detailed example of the nature of the reorganization of clause 8.6.5 suggested by my ballot comments on D1.0.

In the process of figuring out what detailed text changes would best satisfy those comments, and following some discussion with Johannes (who has helped me by providing starting material for this document, but has not been involved in its production other than as described here) it seemed best to attempt a traditional top down description of flow metering and classification. The aim is to start with a high level decomposition of each stage of the problem, using ideas that are hopefully already familiar to someone who has read early editions of the published 802.1Q standard, but has not been in the room while P802.1Qcr amendment is being discussed. The challenge put to me was to introduce that top-down explanation without making use of excessive forward references.

I have endeavored to solve the problem that I encountered when attempting to read P802.1Qcr/D1.0 of rapidly becoming lost in the detail, referencing back and forth with stack overflow. Whether the suggested form of the result works for others only they can tell. I have tried to avoid any technical changes to the D1.0 (except for changes that can be used as a larger effort to clarify how to claim conformance

Clearly this is a work in progress (if any further progress is to be made in the direction it attempts to set out). I am certainly not suggesting that resolution of my comments is dependent on accepting all this replacement text. There will be many places in this document where I intend to make improvements/have not yet made changes that I consider obviously necessary. It is of course missing the changes that will have been accepted as part of D1.0 ballot resolution. However I wanted to make it available while there was still some time for discussion during the May 2019 interim.

In preparing this text I have used a number of editing stylistic approaches to keep things clear while remaining within a style and use of language that has been found acceptable to IEEE staff or their contract editors. I have also tried to keep the use of language consistent with early editions of 802.1Q, though undoubtedly the effeorst of many editors has already introduced some variety.

Mick Seaman, 23rd May 2019

8.6.5 Flow classification and metering

The Forwarding Process can apply flow classification and metering to frames that are received on a Bridge Port and have one or more potential transmission ports. Bridge ports and end stations may support Per-Stream Filtering and Policing (PSFP, 8.6.5.1), Asynchronous Traffic Shaping (ATS) Filtering and Eligibility Time Assignment (8.6.5.1), or the general flow classification rules specified in 8.6.5.1.

NOTE—The general flow classification and metering specification was added to this standard by IEEE Std 802.1Qxx-20YY, PSPF by IEEE Std 802.1Q-20YY, and ATS by IEEE Std 802.1Qcr-2020.

PSFP and ATS share common classification and metering elements, as shown in Figure 8-12. The stream identification function specified in IEEE Std 802.1CB is used to associate each received frame with a set of stream parameters that can also identify an applicable SDU size filter, a stream gate, and a flow meter (for PSFP) or a transmission eligibility time scheduler (for ATS).

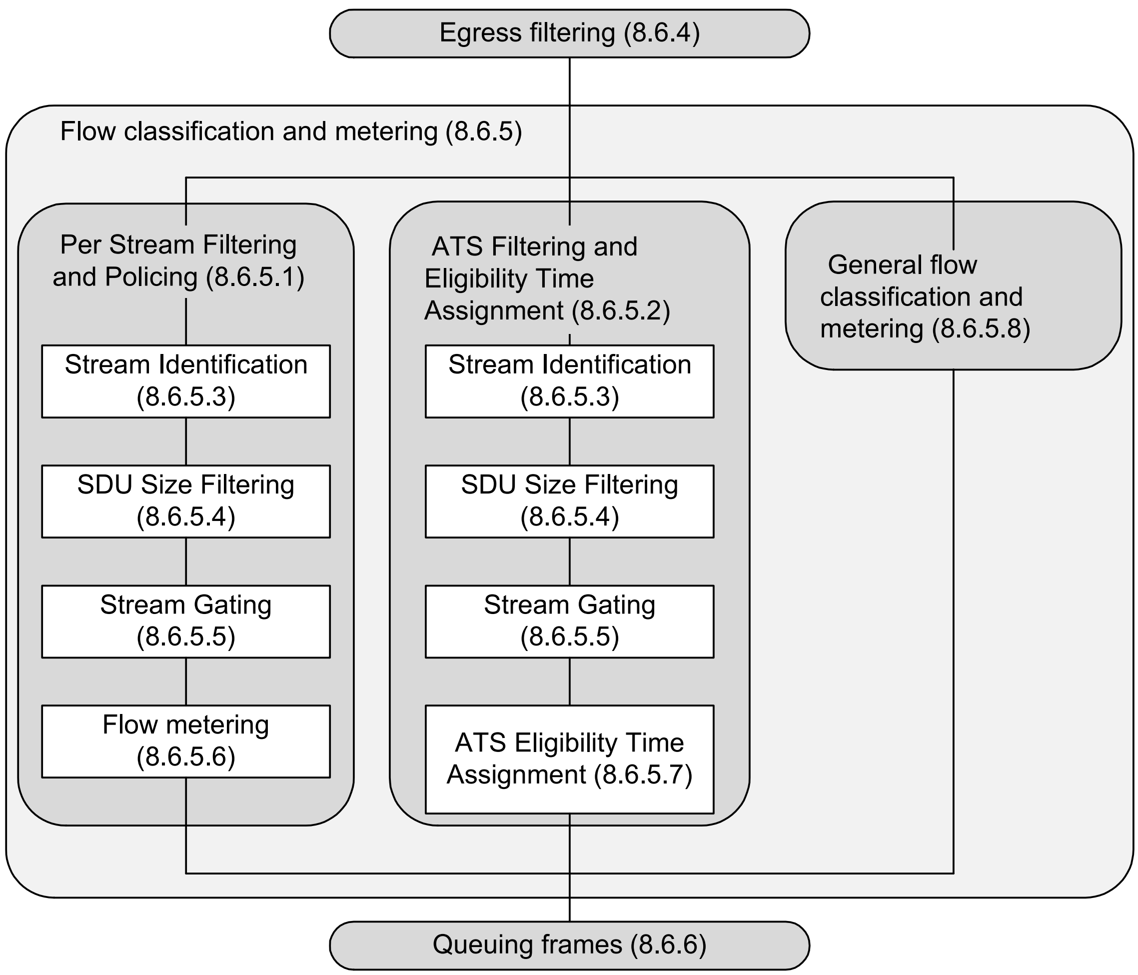


Figure 8-12—Flow classification and metering

8.6.5.1 Per-Stream Filtering and Policing (PSFP)

Per-Stream Filtering and Policing (PSFP) makes filtering and policing decisions for received frames, and supports subsequent queuing decisions (8.6.6.1), as follows:

a) Each received frame is associated with a stream filter, as specified in 8.6.5.3.

   If no matching stream filter is found, the frame is queued for transmission as specified in 8.6.6, without further frame classification and filtering processing. Wildcard stream filters can be configured to match and discard frames not associated with a specified stream.

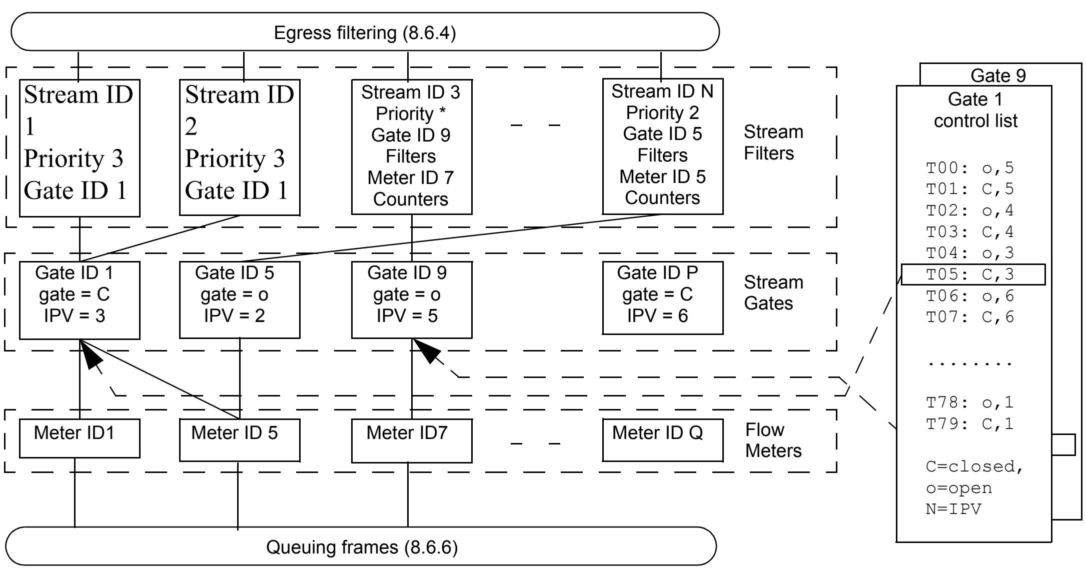
b) The frame is subject to Maximum SDU Sizing Filtering (8.6.5.4), using parameters specified by the stream filter.

c) The frame is processed by the Stream Gate (8.6.5.5) specified by the stream filter, potentially discarding the frame if there is excess traffic for the stream and mapping the frame's priority to an internal priority value (IPV) that can influence subsequent queuing decisions (8.6.6).

   A stream filter can be configured without a Stream Gate, allowing the PSFP controls to provide simpler behavior if desired, with an IPV equal to the frame's priority.

d) The frame is processed by the Flow Meter (8.6.5.5) specified by the stream filter, potentially discarding the frame or marking it as drop eligible. A stream filter can be configured without a Flow Meter.

The relationship between stream filters, Stream Gates, and Flow Meters is illustrated by Figure 8-14.



KEY
Stream ID: stream filter instance identifier (8.6.5.3)
Gate ID: stream gate instance identifier (8.6.5.3, 8.6.5.5)
Meter ID: flow meter instance identifier (8.6.5.3, 8.6.5.6)

Figure 8-14—Per-stream filtering and policing

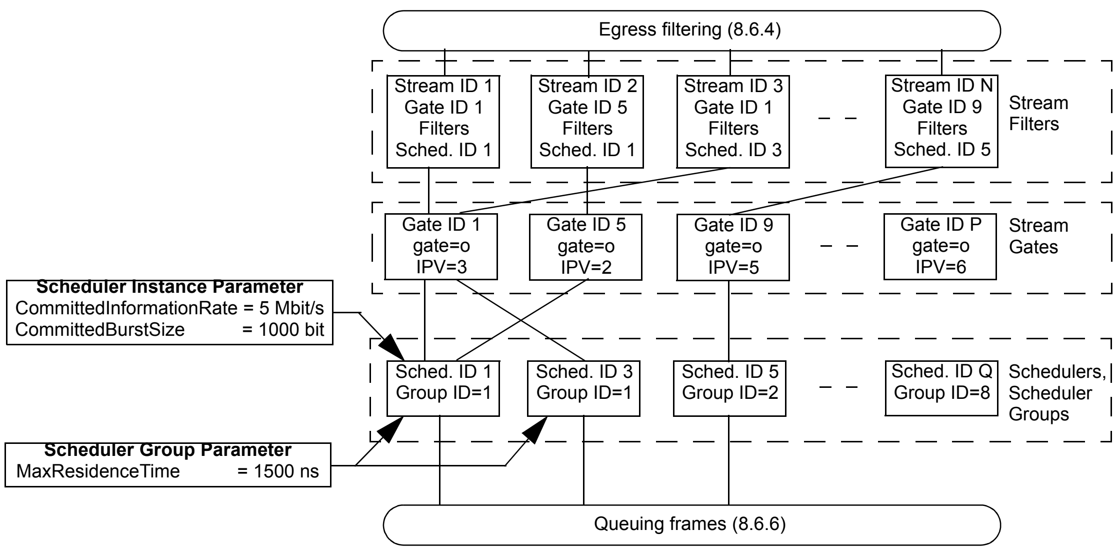
Each Bridge component or end station that implements PSPF, supports stream identification, Maximum SDU Size Filtering, Stream Gates, and Flow Meters, with a single *Stream Filter Instance Table* (8.6.5.3), a single *Stream Gate Instance Table* (8.6.5.5), and a single *Flow Meter Instance Table* per Bridge component or end station.

8.6.5.2 Asynchronous Traffic Shaping (ATS) Filtering and Assignment Functions

Asynchronous Traffic Shaping (ATS) makes filtering and policing decisions for received frames, and supports subsequent queuing decisions (8.6.6.2), as follows:

a) Each received frame is associated with a stream filter, as specified in 8.6.5.3.
   If no matching stream filter is found, the frame is queued for transmission as specified in 8.6.6, without further frame classification and filtering processing. Wildcard stream filters can be configured to match and discard frames not associated with a specified stream.
b) The frame is subject to Maximum SDU Sizing Filtering (8.6.5.4), using parameters specified by the stream filter. The ATS scheduler state machine operation (8.6.11) assumes that the sizes of frames that it processes are less than or equal to the associated CommittedBurstSize parameter (8.6.11.3.5).
c) The frame is processed by the Stream Gate (8.6.5.5) specified by the stream filter, potentially discarding the frame if there is excess traffic for the stream and mapping the frame’s priority to an internal priority value (IPV) that can influence subsequent queuing decisions (8.6.6).
d) The frame is processed by the ATS Scheduler (8.6.5.5) specified by the stream filter, potentially discarding the frame or marking it as drop eligible.

The relationship between stream filters, Streeam Gates, and ATS Schedulers is illustrated by Figure 8-15.



KEY
Stream ID: stream filter instance identifier (8.6.5.3)
Gate ID: stream gate instance identifier (8.6.5.3, 8.6.5.5)
Sched. ID:ATS scheduler instance identifier (8.6.5.3, 8.6.5.7)

Figure 8-15—ATS filtering and assignment functions

8.6.5.3 Stream Identification

Each received frame is associated with a stream filter using the frame's *stream_handle* and *priority* parameters. The *stream_handle* is a sub-parameter of the connection_identifier parameter of the ISS (6.6), provided by the stream identification function specified in Clause 6 of IEEE Std 802.1CB-2017.

Each stream filter comprises the following:

a) An integer *stream filter identifier*.
b) A *stream_handle* specification, either:
   1) A single value, as specified in IEEE Std 802.1CB.
   2) A wildcard, that matches any *stream_handle*.
c) A *priority* specification, either:
   1) A single priority value.
   2) A wildcard value that matches any priority value.
d) Maximum SDU Size Filtering information, comprising:
   1) An integer *Maximum SDU size*, in octets.
   2) A boolean *StreamBlockedDueToOversizeFrameEnable* parameter.
   3) A boolean *StreamBlockedDueToOversizeFrame* parameter.
e) An integer *stream gate identifier*. A value of 0 for this parameter indicates that a Stream Gate is not associated with this stream filter.
f) A set of zero or more *stream filter specifications*.

and the following counters for frames associated with the stream filter:

g) *MatchingFramesCount*: all frames associated with that stream filter.
h) *PassingFrameCount*: frames passing the associated stream gate (8.6.5.5).
i) *NotPassingFrameCount*: frames not passing the stream gate (8.6.5.5).
j) *PassingSDUCount*: frames passing the Maximum SDU size filter (8.6.5.6).
k) *NotPassingSDUCount*: frames not passing the Maximum SDU size filter (8.6.5.6).
l) *RedFramesCount*: frames discarded by the flow meter (8.6.5.6).

The *stream filter identifier* uniquely identifies the stream filter, and acts as an index into a *Stream Filter Instance Table* of up to *MaxStreamFilterInstances* stream filters. Each received frame is associated with the stream filter with the lowest *stream filter identifier* whose *stream_handle* and *priority* specification match the frame's parameters, and the *MatchingFramesCount* is incremented for that filter. A *stream filter identifier* value of 0 is reserved to indicate the lack of any match.

NOTE 1—The use of stream_handle and priority, along with the wild-carding rules previously stated, allow configuration possibilities that go beyond the selection of individual streams, as implied by the sub-clause title; for example, per-priority filtering and policing, or per-priority per-reception Port filtering and policing can be configured using these rules.

NOTE 2—If it is desired to discard frames that do not match any other stream filter, rather than such frames being processed without filtering, this can be achieved by placing a stream filter at the end of the table, in which the stream_handle and priority are both wild-carded (set to the null value), and where the stream gate instance identifier points at a stream gate that is permanently closed.

8.6.5.4 Maximum SDU Size Filtering

If the SDU size of a frame exceeds the value of the associated stream filter's Maximum SDU size parameter, the frame is discarded and that stream filter's NotPassingSDUCount is incremented. If the stream filter's StreamBlockedDueToOversizeFrameEnable parameter is configured to be TRUE, the StreamBlockedDueToOversizeFrame parameter is set to TRUE and all subsequent frames will be discarded until StreamBlockedDueToOversizeFrame is administratively reset to FALSE.

Otherwise, the stream filter's PassingSDUCount is incremented (see 8.6.5.3). The default configuration od both StreamBlockedDueToOversizeFrameEnable and StreamBlockedDueToOversizeFrame is FALSE.

NOTE—The Maximum SDU size is defined per stream filter and can therefore differ from the queueMaxSDU specified in 8.6.8.4. As queueMaxSDU is applied after the flow classification and metering, it is possible that a frame that passes the Maximum SDU size filter will later be discarded because its SDU size exceeds queueMaxSDU.

8.6.5.5 Stream Gates

Stream gates determine whether frames are discarded or passed for further processing by the remaining forwarding process. Stream gates also map the frame's priority to an internal priority value (IPV) that is used to make subsequent queuing decisions. The IPV co-exists with the received frame's priority parameter, and does not replace that priority.

NOTE 1—The co-existence of the IPV specification and the priority value allows subsequent queuing decisions (8.6.6) to be based on the IPV, while the priority value is retained for transmission. A use case of IPV for ATS is the adjusting of per-hop delay bounds to satisfy end-to-end delay requirements in a specific network. Another use case for the ability to assign IPVs can be found in Annex T (CQF).

Each stream gate comprises the following:

a) An integer *stream gate instance identifier*.
b) An administrative and an operational *stream gate state* parameter.
   The operational *stream gate state* an take one of two values:
   1) Open: Frames are permitted to pass through the stream gate.
   2) Closed: Frames are not permitted to pass through the stream gate.
c) An administrative and an operational *internal priority value specification*.
   The operational *internal priority value specification* can be one of the following:
   1) Null, in this case the received frame's priority parameter is used as the IPV.
   2) A specific IPV for the frame.
d) An administrative and an operational *stream gate control list*.
e) A boolean *GateClosedDueToInvalidRxEnable* parameter.
f) A boolean *GateClosedDueToInvalidRx* parameter.
g) A boolean *GateClosedDueToOctetsExceededEnable* parameter.
h) A boolean *GateClosedDueToOctetsExceeded* parameter.

The *stream gate instance identifier* uniquely identifies the stream gate, and acts as an index into a *Stream Gate Instance Table* of up to *MaxStreamGateInstances* stream gates.

An instance of the stream gate control state machine (8.6.10) determines the operational values of the *stream gate state* and the *internal priority value specification* [b) and c) above] by the cyclical execution of the control operations (see Table 8-4) specified in the stream gate's *stream gate control list* [d) above]. The administrative *stream gate state* and *internal priority value specification* parameters are used to determine the initial values of the corresponding operational parameters, and the administrative *stream gate control list* parameter allows configuration a new control list prior enabling its use in a running system.

If a frame is passed by a stream gate, the *PassingFrameCount* of the stream filter (8.6.5.3) associated with that frame is incremented. The *NotPassingFrameCount* is incremented if the frame is discarded.

Stream gates are able to permanently discard frames and thus effectively override the operational gate state (i.e., the stream gate behaves as if the operational stream gate state is Closed). This capability is provided by the GateClosedDueToInvalidRx and GateClosedDueToOctetExceed functions:

i) The GateClosedDueToInvalidRx function is enabled if the GateClosedDueToInvalidRxEnable parameter is TRUE, and disabled if this parameter is FALSE. If the function is enabled and any

Table 8-4—Stream gate control operations

Operation name	Parameter(s)	Action
SetGateAndIPV	StreamGateState, IPV, TimeInterval, IntervalOctetMax	The StreamGateState parameter specifies a desired state, *open* or *closed*, for the stream gate, and the IPV parameter specifies a desired value of the IPV associated with the stream. On execution, the StreamGateState and IPV parameter values are used to set the operational values of the stream gate state and internal priority specification parameters for the stream. After *TimeInterval* ticks (8.6.9.4.16) has elapsed since the completion of the previous stream gate control operation in the stream gate control list, control passes to the next stream gate control operation. The optional IntervalOctetMax parameter specifies the maximum number of MSDU octets that are permitted to pass the gate during the specified TimeInterval. If the IntervalOctetMax parameter is omitted, there is no limit on the number of octets that can pass the gate.

frame is discarded because the stream gate is in the closed state, then the GateClosedDueToInvalidRx parameter is set to TRUE, and all subsequent frames are discarded as long as the GateClosedDueToInvalidRxEnable and GateClosedDueToInvalidRx parameters are TRUE.

j) The GateClosedDueToOctetsExceeded function is enabled if the GateClosedDueToOctetsExceededEnable parameter is TRUE, and disabled if this parameter is FALSE. If the function is enabled and any frame is discarded because there are insufficient IntervalOctetsLeft (8.6.10.8), then the GateClosedDueToOctetsExceeded parameter is set to TRUE, and all subsequent frames are discarded as long as the GateClosedDueToOctetsExceededEnable and GateClosedDueToOctetsExceededEnable parameters are TRUE.

Per default, the GateClosedDueToInvalidRx and GateClosedDueToOctetExceeded functions are disabled and all associated parameters have the default value FALSE.

NOTE 2—The GateClosedDueToInvalidRx and GateClosedDueToOctetsExceeded functions allow the detection of incoming frames during time periods when the stream gate is in the closed state and exceptionally large ingress bursts to result in the stream gate behaving as it is in a permanently closed state, until such a time as management action is taken to reset the condition. The intent is to support applications where the transmission and reception of frames across the network is coordinated such that frames are received only when the stream gate is open with a limited overall amount of ingress octets. Hence, frames received by the stream gate when it is in the closed state and unexpected amounts of ingress octets represent invalid receive conditions.

8.6.5.6 Flow Meters

The Flow meters specified by this clause (8.6.5.6) implement the parameters and algorithm specified in *Bandwidth Profile Parameters and Algorithm* in MEF 10.3 with the additions described in this clause.

Each Bridge component provides a *Flow Meter Instance Table* with parameters and variables of up to *MaxFlowMeterInstances* flow meter instances. Each flow meter instance is associated with the following parameters and variables:

a) An integer *Flow meter instance identifier.*
b) An integer *Committed information rate (CIR)*, in bits per second (MEF 10.3).
c) An integer *Committed burst size (CBS)*, in octets (MEF 10.3).
d) An integer *Excess Information Rate (EIR)*, in bits per second (MEF 10.3).
e) An integer *Excess burst size (EBS) per bandwidth profile flow*, in octets (MEF 10.3).
f) A *Coupling flag (CF)*, which takes the value 0 or 1 (MEF 10.3).
g) A *Color mode (CM)*, which takes the value *color-blind* or *color-aware* (MEF 10.3).
h) A boolean *DropOnYellow* parameter.

i) A boolean *MarkAllFramesRedEnable* parameter.
j) A boolean *MarkAllFramesRed* parameter.

NOTE 1—Envelope and Rank, as defined in MEF 10.3, are not used by the flow meters described in this clause; i.e., the reduced functionality algorithm described in 12.2 of MEF 10.3 is used.

The flow meter instance identifier uniquely identifies the flow meter instance, acts as an index to the Flow Meter Instance Table, and associates stream filter instances (8.6.5.3) with the flow meter instance.

The DropOnYellow parameter indicates whether frames marked yellow by the MEF 10.3 algorithm are discarded or marked as drop eligible:

k) A value of TRUE indicates that yellow frames are discarded.
l) A value of FALSE indicates that the drop_eligible parameter of yellow frames is set to TRUE.

NOTE 2—Changing the value of the drop_eligible parameter may change the contents of the frame, depending on how the frame is tagged when transmitted, which may then require updating the frame_check_sequence. Mechanisms for conveying information from ingress to egress that the frame_check_sequence may require updating are implementation dependent.

Flow meters are able to permanently discard all frames after an initial frame has been discarded. This capability is provided by the MarkAllFramesRed function. The function is enabled if the MarkAllFramesRedEnable parameter is TRUE, and disabled if this parameter is FALSE. If the function is enabled and the flow meter discards a frame, then the MarkAllFramesRed parameter is set to TRUE, and all subsequent frames are discarded as long as the MarkAllFramesRedEnable and MarkAllFramesRed parameters are TRUE. Per default, the MarkAllFramesRed function is disabled and both associated parameters have the default value FALSE.

Each time a flow meter discards a frame, the RedFramesCount counter of the originating stream filter (8.6.5.3) is increased.

8.6.5.7 ATS Schedulers

Asynchronous Traffic Shaping (ATS) Schedulers assign eligibility times to frames which are then used for traffic regulation by the ATS transmission selection algorithm (8.6.8.5).

NOTE 1—Contrary to the clause name, ATS Schedulers only realize the computational part of the overall traffic shaping operation of ATS. The complete operation is provided by the combination with the ATS transmission selection algorithm, which uses the assigned eligibility times to regulate the traffic for transmission.

ATS Schedulers are organized in *ATS Scheduler Groups*. There is one ATS scheduler group per reception Port per upstream traffic class, where the latter refers to the transmitting traffic class in the device connected to the given reception Port. All ATS scheduler instances that process frames from a particular reception Port and a particular upstream traffic class are in the respective ATS scheduler group.

NOTE 2—The organization of ATS scheduler instances into groups results in a non-decreasing ordering of eligibility times of successive frames associated with a single ATS scheduler group. This permits frames of one group to be queued in a FIFO order.

Each Bridge component provides an *ATS Scheduler Instance Table* with parameters and variables of up to *MaxSchedulerInstances* ATS scheduler instances, an *ATS Scheduler Group Instance Table* with parameters and variables of up to *MaxSchedulerGroupInstances* ATS scheduler group instances, and an *ATS Port Parameter Table* with parameters and variables shared by all ATS scheduler instances associated with a reception Port.

Each ATS Scheduler instance is associated with the following parameters and variables:

a) An integer *scheduler instance identifier*.
b) An integer *scheduler group instance identifier*.
c) An integer *CommittedBurstSizeParameter* parameter, in bits (8.6.11.3.5).
d) An integer *CommittedInformtionRate* parameter, in bits per second (8.6.11.3.6).
e) An internal *bucket empty time* state variable, in seconds (8.6.11.3.3).

Each ATS Scheduler group instance is associated with the following parameters and variables:

f) An integer *scheduler group instance identifier*.
g) An integer *MaximumResidenceTime* parameter, shared by all ATS scheduler instances in a scheduler group, in nanoseconds (8.6.11.3.13).
h) An internal *group eligibility time* state variable, shared by all ATS scheduler instances in a scheduler group, in seconds (8.6.11.3.10).

Each Port is associated with the following variable for ATS schedulers:

i) An integer *DiscardedFramesCount* counter for frames that were discarded by the associated ATS scheduler instances.

The scheduler instance identifier uniquely identifies the ATS scheduler instance, acts as an index to the ATS Scheduler Instance Table, and associates stream filter instances (8.6.5.3) with a particular ATS scheduler instance. The scheduler group instance identifier uniquely identifies a scheduler group instance and establishes the relationship between ATS scheduler instances and ATS scheduler group instances.

NOTE 3—Whether ATS-scheduler instances, ATS scheduler group instances, the scheduler instance table, and the scheduler group instance table are located in reception ports or transmission ports is implementation specific.

Each ATS scheduler instance assigns eligibility times to the associated frames, and discards frames in exceptional situations. The underlying operations are performed by an ATS scheduler state machine (8.6.11) associated with an ATS scheduler instance. This state machine updates the associated bucket empty time and group eligibility time state variables based on the CommittedBurstSize parameter, the CommittedInformationRate parameter, the MaxResidenceTime parameter, the frame arrival times, and the frame lengths (including media-specific overhead).

If an ATS scheduler instance discards a frame, the DiscardedFramesCount counter of the associated Port is increased.

8.6.5.7.1

For ATS support in bridges, stream gate state transitions based on stream gate control lists (8.6.5.5) are optional. That is, it is sufficient that stream gates permanently reside either in the state open or in the state closed, and IPVs are assigned on a per-gate basis.

NOTE 2—For bridges with support for ATS, and without support for Scheduled Traffic and PSFP, stream gates of ATS traffic will never close. In this case, stream gates are permanently open and only used for IPV assignment.

NOTE 3—If the stream gate state value Closed is supported and used (e.g., for dedicated traffic classes for scheduled traffic, co-existent with dedicated traffic classes for ATS traffic), the asynchronous behavior of ATS traffic can require simultaneous stream gate state changes of multiple stream gates associated to ATS traffic.

ATS support in end stations is provided by a modified variant of the ATS Filtering and Assignment Functions, as specified in clause 49.

8.6.5.8 General flow classification and metering

Bridges that implement general flow classification and metering can identify subsets of traffic (frames) each of which is subject to the same flow metering and forwarding. Classification rules may be based on

a) Destination MAC address
b) Source MAC address
c) VID
d) Priority

Item c), specifying a VID value, is not applicable to VLAN-unaware MAC Relays.

Frames classified using the same set of classification rules are subject to the same flow meter. The flow meter can change the drop_eligible parameter associated with each frame and can discard frames on the basis of the following parameters for each received frame and previously received frames, and the time elapsed since those frames were received:

e) The received value of the drop_eligible parameter
f) The mac_service_data_unit size

The flow meter shall not base its decision on the parameters of frames received on other Bridge Ports, or on any other parameters of those Ports. The metering algorithm described in the Metro Ethernet Forum (MEF) Technical Specification 10.3 (MEF 10.3) should be used.

NOTE 1—Changing the value of the drop_eligible parameter may change the contents of the frame, depending on how the frame is tagged when transmitted, which may then require updating the frame_check_sequence. Mechanisms for conveying information from ingress to egress that the frame_check_sequence may require updating are implementation dependent.

NOTE 2—The flow meter described here can encompass a number of meters, each with a simpler specification. However, given the breadth of implementation choice permitted, further structuring to specify, for example, that frames can bypass a meter or are subject only to one of a number of meters provides no additional information.

NOTE 3—Although flow metering is applied after egress (Figure 8-11), the meter(s) operate per reception Port (see first sentence of 8.6.5), not per potential transmission Port(s).