A UML model for link aggregation management

Mick Seaman

This note is a continuation of work begun during the early development of IEEE Standards 802.1Qcp and 802.1Xck. The aim is to help in the review of P802.1AX-Rev[1].

In an ideal world P802.1AX-Rev would incorporate a UML model. This is probably too ambitious, and it would be better to let that project complete without adding further work.

1. Summary

Section 3 describes some important aspects of the current management specification and an initial mapping to a UML/YANG model. Section 4 suggests a reassignment of link aggregation attributes to interfaces to remove duplication and simplify configuration, reducing the scope for errors and temporary inconsistencies between parameters associated with separate interfaces. Figure 2 is a draft UMLish model incorporating that suggestion. Section 2 describes some tentaiive conclusions drawn from the work so far.

2. Tentative conclusions

The current organization of management in 802.1 documents in general (which originated back in 1987) of having management objects in a supposedly management protocol independent clause, separate from the specification (in text, state machines, and formal variables) of that which is to be managed, is cumbersome and error prone. While this seemed to be working when we were dealing with differences between pre-standard management protocols, GDMO, and early MIBs, it has lead to the generation of massive translation tables that are hard to maintain and even harder to review.

A better approach would have been to include the use of all management controls, and updating of counters etc. within the basic operational specification, using the descriptive techniques natural to that specification and making sure each control and counter in that specification could be unambiguously referred to by a clause number and a name.

An SNMP MIB, for example, would then reference that specification directly instead of having an additional layer of translation and repetition (e.g. as in the subclauses of .1AX/D0.4) where the repetition of "ATTRIBUTE", "APPROPRIATE SYNTAX", and "BEHAVIOR DEFINED AS" overlaps with what is in the MIB, does not have unambiguous references to behavioral specification, and in any case leads to a further[2] 4 page mapping table (Table D.1) enroute to the MIB.

At this stage, I don’t believe that there is much to be gained by attempting to repair the SNMP MIB situation. Product MIBs bearing greater or lesser resemblance to the standard MIB and the standard’s state machines and procedures have already been written and deployed. The aim should be to minimize the amount of work involved in revising the standard.

We don’t need to fall into the same trap of translation table upon translation table when it comes to YANG. A UML model is incredibly useful in describing the management attributes and operations, not least by making it possible to get the entire picture on one page and thus avoid overloading the natural limitations of human short-term memory. At the same time considerations of alternate models convinces me that the most useful UML is not protocol-independent either. While a protocol-ignorant UML model can be very useful as a way of depicting what is involved in managing a particular protocol or product feature, management frameworks (such as that underlying YANG) make sweeping assumptions about how the management information should be structured. Extensive compromises, such as duplicating information so the vast majority of it can be presented as per-interface attributes, may be made.

So for YANG we probably want to develop a UML model that: (a) directly references .1AX Clauses 6 and 9, adding or clarifying text in those clauses if necessary to allow an unambiguous reference to be

[1] In general (though not particularly in .1AX-Rev/D0.4) the description of management objects is so verbose as to discourage all but the most dedicated of reviewers. A typical managed object definition/declaration takes about 6 lines of text, only part of which is a more or less accurate repetition of what has to be (and is already) said elsewhere in a standard. In contrast C++ can fit the declaration of up to half-a-dozen or so such objects on a single line (many objects being of a common type and supporting the same set of operations, and the name being a sufficient reference to the existing description), and UML can provide an even greater information density - to the point where it is actually possible to see what is going on in a module of significant size.

[2] In addition to the clause 7.3 subclauses we have already had 5 pages of Table 7-1.

made; and (b) is deliberately designed to make the final translation to YANG as mechanical as possible. This should facilitate review of the management specification by those who can't or won't plow their way through the YANG code, and help ensure that the YANG actually manages what is in Clauses 6 and 9 and not some alternative specification that seems better to management section developers.

3. 802.1AX-2014 management

The management specification for Link Aggregation in Clause 7 of 802.1AX–2014 is basically that initially standardized in Clause 30 of 802.3ad–2000.[3] Management attributes for Aggregation Ports complement those already specified for the 802.3 MAC, while attributes for Aggregators do replicate those of the MAC. The idea was to make each Aggregator look exactly like a MAC. The 802.3ad specification effort was completed just prior to that for the Interfaces Group MIB (RFC 2863). 802.1AX-2014 added references to 2863, in a set of cross-reference tables that explained the equivalence of some of the definitions involved.[4] The MIBs for other 802.1 standards that make use of the ISS (including that for the 802.1AE MAC Security Entity shim) use 2863 and assume their management relationship to that of the underlying service is independent of the entity providing that service.

Clause 7 does not include cross-references to the state machines, variables, and procedures used in specification of link aggregation in Clauses 6 and DRNI in Clause 9. However Clause 6 does provide forward cross-references to Clause 7. The MIB is cross-referenced to Clause 7.

Clause 7 divides link aggregation attributes (with the exception of some for DRNI) in two: Aggregator (port) attributes; and Aggregation Port attributes. This note provides a summary in Table 1 through Table 5, organized by P802.1AX/D0.4 clause number, with a tentative mapping of each to one of five sets of ietf-interfaces/interface/ objects:

a) Aggregator basic interface attributes (e.g. if-index, description, name);

b) Aggregator basic interface statistics (e.g. octetstx, octetsrx, out-unicast-pkts);

c) Aggregator specific attributes, including those related to LACP operation, such as partner system identity;

d) Attributes that are already associated with other likely Aggregator interface augmentations, such as bridge-port-statistics, and should not be duplicated for link aggregation;[5]

e) Aggregation Port basic interface attributes, (if-index, description, name etc.);

f) Aggregation Port basic interface statistics;

g) Aggregation Port specific attributes, including those related to LACP.

Omitting deprecated attributes, there are 22 attributes associated with an Aggregator [item c) above] and 26 associated with Aggregation Port [item g)]. There are a few items missing (e.g. the Wtr_Revertive control) and a few more may be desirable.

The LACP parameters are split between the Aggregators and the Aggregation Ports, and 8 of the Aggregation Port attributes are duplicates of Aggregator attributes. This makes LACP configuration more complex than necessary, with the possibility of windows of inconsistency as changes to one interface are made before changes to another. The proposed UML for YANG model (see 4. below) reassigns attributes to remove the duplicates and possible inconsistencies.

4. UML/YANG for basic LACP and CSCD

Figure 1 reproduces Figure 6-15 of .1AX/D0.4, which illustrates the recommended default operation of the aggregator Selection Logic.

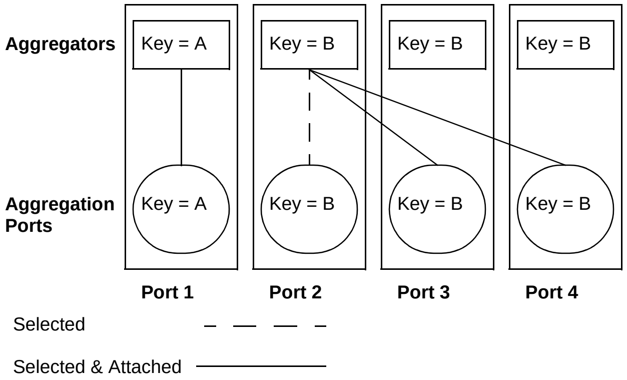


Figure 1—Selection of Aggregators

[3] DRNI had been added, but the initial text of the clause describing its structure, purpose, and assumptions remains unchanged.

[4] 802.3-2000 Clause 30 and 802.1AX-2008 both reference RFC 2233 (The Interfaces Group MIB using SMIv2) which was obsoleted by RFC 2863.

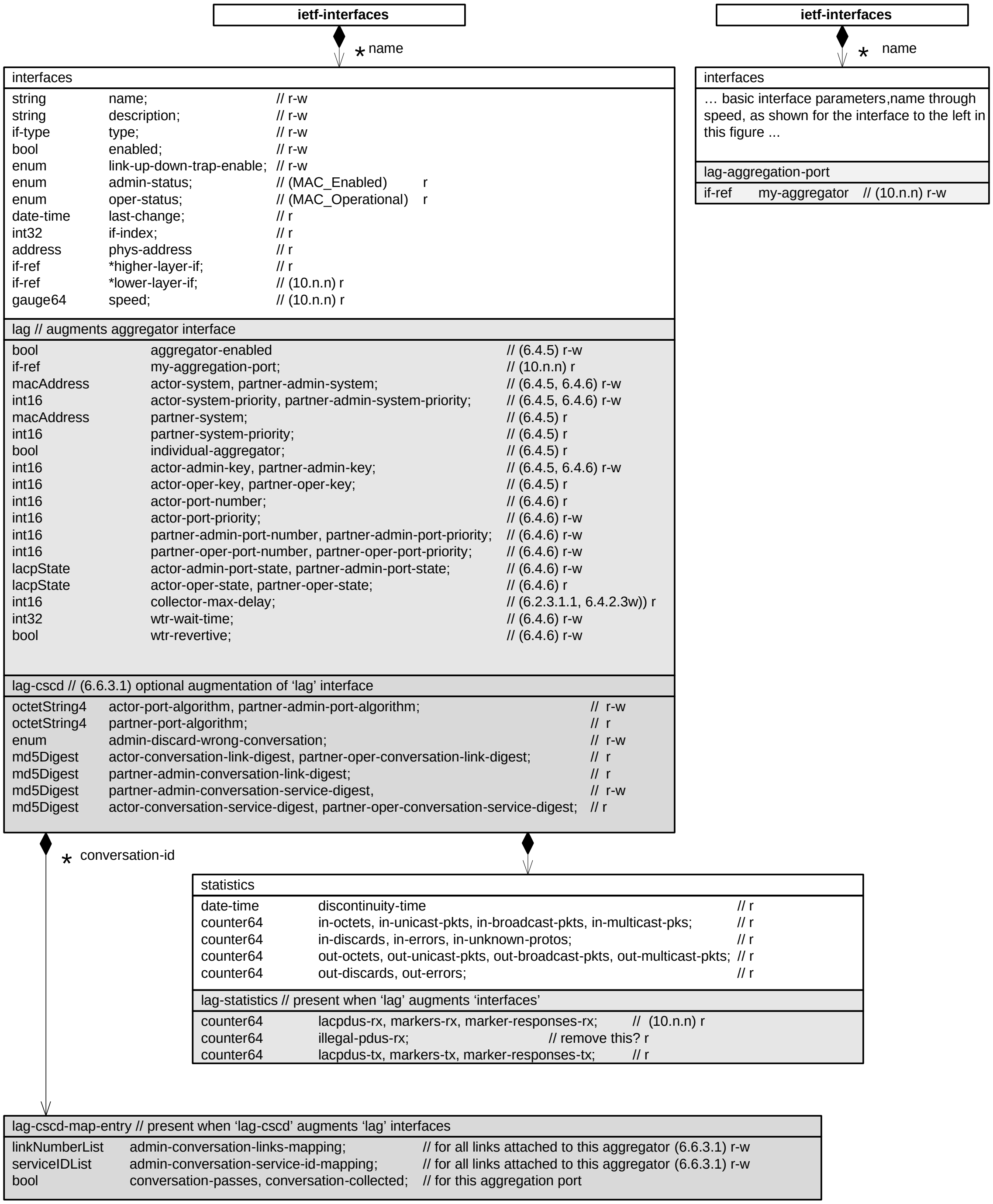
[5] At present I have only identified two of these: aAggFramesTxOK and aAggFramesRxOK. They don't seem to map to the IETF interface packet statistics—because the latter are broken out by unicast, multicast, and broadcast—but the bridge-port-statistics specified in 802.1Qcp include the consolidated parameters.

A UML model for link aggregation management

Some well known points bear repeating. Each Aggregation Port is paired with an Aggregator (its ‘home Aggregator’), with each member of the pair having the same actor and administratively assigned partner values for System Priority, SystemID, and Key. If its home Aggregator has been selected, they have same values for the actor and partner operational variables they have in common. If an Aggregation Port has selected another Aggregator, then its home Aggregator has not been selected by any other Aggregation Port. We can take advantage of these characteristics to remove duplication of information between Aggregator and Aggregation Port interfaces, and thus lessen the potential for misconfiguration and the difficulty involved in synchronizing changes. Further, we can associate all the LACP configuration information with one of the pair (the Aggregator is the obvious choice). This further simplifies configuration.[6]

Figure 2 is a UMLish model along these lines. For the present I hope it can speak for itself. UML/YANG attribute names have been aligned with those used in Clauses 6 and 9 (with the exception of seemingly inevitable differences in case stropping/hyphenation/underscore conventions) rather with the Clause 7 variants.

[6] These changes do not prevent the use of the model with other Selection Logic choices: if fewer Aggregator than Aggregation Ports are required, each surplus Aggregator interface can be marked as disabled, with its parameters still supporting its Aggregation Port; if operational practice requires dynamic identification of physical links (which can be plugged into physical ports in no particular order) while each Aggregator has permanently configured attributes (associated with other interface augmentations, such as those for a Bridge Port), the Aggregation Port to Aggregator pairing can be changed.



interfaces

string	name;	// r-w
string	description;	// r-w
if-type	type;	// r-w
bool	enabled;	// r-w
enum	link-up-down-trap-enable;	// r-w
enum	admin-status;	// (MAC_Enabled) r
enum	oper-status;	// (MAC_Operational) r
date-time	last-change;	// r
int32	if-index;	// r
address	phys-address	// r
if-ref	*higher-layer-if;	// r
if-ref	*lower-layer-if;	// (10.n.n) r
gauge64	speed;	// (10.n.n) r

lag // augments aggregator interface

bool	aggregator-enabled	// (6.4.5) r-w
if-ref	my-aggregation-port;	// (10.n.n) r
macAddress	actor-system, partner-admin-system;	// (6.4.5, 6.4.6) r-w
int16	actor-system-priority, partner-admin-system-priority;	// (6.4.5, 6.4.6) r-w
macAddress	partner-system;	// (6.4.5) r
int16	partner-system-priority;	// (6.4.5) r
bool	individual-aggregator;	// (6.4.5) r
int16	actor-admin-key, partner-admin-key;	// (6.4.5, 6.4.6) r-w
int16	actor-oper-key, partner-oper-key;	// (6.4.5) r
int16	actor-port-number;	// (6.4.6) r
int16	actor-port-priority;	// (6.4.6) r-w
int16	partner-admin-port-number, partner-admin-port-priority;	// (6.4.6) r-w
int16	partner-oper-port-number, partner-oper-port-priority;	// (6.4.6) r-w
lacpState	actor-admin-port-state, partner-admin-port-state;	// (6.4.6) r-w
lacpState	actor-oper-state, partner-oper-state;	// (6.4.6) r
int16	collector-max-delay;	// (6.2.3.1.1, 6.4.2.3w)) r
int32	wtr-wait-time;	// (6.4.6) r-w
bool	wtr-revertive;	// (6.4.6) r-w

lag-cscd // (6.6.3.1) optional augmentation of 'lag' interface

octetString4	actor-port-algorithm, partner-admin-port-algorithm;	// r-w
octetString4	partner-port-algorithm;	// r
enum	admin-discard-wrong-conversation;	// r-w
md5Digest	actor-conversation-link-digest, partner-oper-conversation-link-digest;	// r
md5Digest	partner-admin-conversation-link-digest;	// r
md5Digest	partner-admin-conversation-service-digest,	// r-w
md5Digest	actor-conversation-service-digest, partner-oper-conversation-service-digest;	// r

interfaces

… basic interface parameters,name through speed, as shown for the interface to the left in this figure ...

lag-aggregation-port

if-ref	my-aggregator	// (10.n.n) r-w

statistics

date-time	discontinuity-time	// r
counter64	in-octets, in-unicast-pkts, in-broadcast-pkts, in-multicast-pks;	// r
counter64	in-discards, in-errors, in-unknown-protos;	// r
counter64	out-octets, out-unicast-pkts, out-broadcast-pkts, out-multicast-pkts;	// r
counter64	out-discards, out-errors;	// r

lag-statistics // present when 'lag' augments 'interfaces'

counter64	lacpdus-rx, markers-rx, marker-responses-rx;	// (10.n.n) r
counter64	illegal-pdus-rx;	// remove this? r
counter64	lacpdus-tx, markers-tx, marker-responses-tx;	// r

lag-cscd-map-entry // present when 'lag-cscd' augments 'lag' interfaces

linkNumberList	admin-conversation-links-mapping;	// for all links attached to this aggregator (6.6.3.1) r-w
serviceIDList	admin-conversation-service-id-mapping;	// for all links attached to this aggregator (6.6.3.1) r-w
bool	conversation-passes, conversation-collected;	// for this aggregation port

lacpdus-rx: 6.4.8 recordPDU()
markers-rx: 6.5.4
marker-responses: 6.5
lacpdus-tx: 6.4.8 transmitLACPDU()
markers-tx, markers-responses-tx: 6.5

Figure 2—Link Aggregation management (draft)

Table 1—Aggregator [port] attributes (from P802.1AX D0.4 7.3.1.1.)

#[1]	D0.4 Clause 7 name	ietf-interfaces/interface/[2]	
1	aAggID	if-index	
2	aAggDescription-r	description	Is YANG attribute defined as read-only?
3	aAggName-rw	name	Check -rw for all IETF YANg attributes
		type	Set to reflect top of i/f stack type?
4	aAggActorSystemID -rw[3]	**aggregator**/actor-system-id	
5	aAggActorSystemPriority-rw	**aggregator**/actor-system-priority	Is this really independently settable per aggregator?[4]
6	aAggAggregateOrIndividual-r	**aggregator**/aggregate-or-individual[5]	
7	aAggActorAdminKey-rw	**aggregator**/actor-admin-key	Description not explicit on rw.
8	aAggActorOperKey-r	**aggregator**/actor-oper-key	
9	aAggMACAddress-r	phys-address	
10	aAggPartnerSystemID-r	**aggregator**/partner-system-id	Don't we need partner admin values?
11	aAggPartnerSystemPriority-r	**aggregator**/partner-system-priority	Partner admin value?
12	aAggPartnerOperKey-r	**aggregator**/partner-oper-key	Partner admin value?
13	aAggAdminState-rw	enabled[6]	Check relation to bridge port?
		admin-status	Read-only. Equivalent to ISS MAC_Enabled (per .1Xck)
14	aAggOperState-r	oper-status	Equivalent to ISS MAC_Operational (per .1Xck)
15	aAggTimeofLastOperchange-r	last-change	Check IETF defn for this and all interface attributes
16	aAggDataRate-r	speed	
17	aAggOctetsTxOK-r	statistics/octetstx	
18	aAggOctetsRxOK-r	statistics/octetsrx	
		statistics/out-unicast-pkts	
		statistics/in-unicast-pkts	
19	aAggFramesTxOK-r	**bridge-port-statistics**/frame-tx[7]	Total frame count, unicast, multicast, broadcast?
20	aAggFramesRxOK-r	**bridge-port-statistics**/frame-rx	Total frame count, unicast, multicast, broadcast?
21	aAggMulticastFramesTxOK-r	statistics/out-multicast-pkts	
22	aAggMulticastFramesRxOK-r	statistics/in-multicast-pkts	
23	aAggBroadcastFramesTxOK-r	statistics/out-broadcast-pkts	
24	aAggBroadcastFramesRxOK-r	statistics/in-broadcast-pkts	
25	aAggFramesDiscardedOnTx-r	statistics/out-discards	
26	aAggFramesDiscardedOnRX-r	statistics/in-discards	
27	aAggFramesWithTxErrors-r	statistics/out-errors	
28	aAggFramesWithRxErrors-r	statistics/in-errors	
29	aAggUnknownProtocolFrames-r	statistics/in-unknown-protos	As defined don't believe this belongs here at all.
30	aAggPortList-r	*lower-layer-if	
31	aAggLinkUpDownNotificationEnable-rw	link-up-down-trap-enable	Check IETF YANG definition
32	aAggCollectorMaxDelay-r	**aggregator**/collector-max-delay	2 octets in LACPDU but MIB says int32
33	aAggPortAlgorithm-rw	**aggregator**/actor-port-algorithm[8]	Is algorithm negotiation possible?
34	aAggPartnerAdminPortAlgorithm-rw	**aggregator**/partner-admin-port-algorithm	
35	aAggConversationAdminLink[]-rw	**aggregator**/actor-admin-conv-link[][9],[10]	aAgg name has Admin in unusual place
36	aAggPartnerAdminPortConversationListDigest-rw	**aggregator**/partner-admin-conv-link-digest	
37	aAggAdminDiscardWrongConversation-rw	**aggregator**/actor-admin-discard-wrong-conv	
38	aAggAdminServiceConversationMap[]-rw	**aggregator**/actor-admin-conv-service-map[]	D0.4 calls this a Service Conversation Map ...
39	aAggPartnerAdminConvServiceMappingDigest-rw	**aggregator**/partner-admin-conv-service-digest[11]	... but this a Conv[ersation] Service Digest
40	aAggOperDiscardWrongConversation	**aggregator**/actor-oper-discard-wrong-conv[12]	
41	aAggConvLinkDigest-r	**aggregator**/actor-conv-link-digest	
42	aAggConvServiceDigest-r	**aggregator**/actor-conv-service-digest	
43	aAggPartnerPortAlgorithm-r	**aggregator**/partner-port-algorithm	Did not add '-oper' after 'partner'
44	aAggPartnerConvLinkDigest-r	**aggregator**/partner-conv-link-digest	
45	aAggPartnerConvServiceDigest-r	**aggregator**/partner-conv-service-digest	

[1] 7.3.1.1. subclause number.
[2] Initial/working name suggestions.
[3] Might allow multiple sets of potentially aggregatable ports within a single system or indeed within a single component. See NOTE on page 46 of D0.4Otherwise dependent on bridge port.
[4] Is this really independently settable per aggregator? If not how is it set?
[5] No strong opinion about suggested name
[6] What is the relationship to the IETF YANG interface admin-status (which is read-only) See also 1AC defn. of MAC_Enabled which is read only?
[7] .1Qcp has 'frametx' in Figure 48-10, which is surely wrong.
[8] Note name. For consistency with actor-admin-key I have added 'actor-'.
[9] Name reordered for consistency
[10] I am not sure how this list should be represented.
[11] Left out 'Mapping' from the name to align with aAggConvServiceDigest.
[12] Added 'actor-' for consistency with other 'oper' attributes, e.g.aAggActorOperKey/actor-oper-key.

Table 2—Aggregator notifications (from P802.1AX D0.4 7.3.1.2.)

#[1]	D0.4 Clause 7 name	ietf-interfaces/interface/[2]	
1	nAggLinkUpNotification	link-up-down-trap-enable	
2	nAggLinkDownNotification		Not separately controlled from Up notification?

[1] 7.3.1.2 subclause number.
[2] Initial/working name suggestions.

Table 3—Aggregation Port attributes (from P802.1AX D0.4 7.3.2.1.)

#[1]	2	D0.4 Clause 7 name	ietf-interfaces/interface/[3]	
1		aAggPortID	if-index	
2			description	Description and name missing in 7.3.2, was assumption ..
3			name	.. these were in ‘real Ethernet’ attributes?
			type	Set to reflect top of i/f stack type?
2	5	aAggPortActorSystemPriority -rw[4]	**aggPort**/actor-system-priority	This and next reordered from 7.3.1.
3	4	aAggPortActorSystemID -r	**aggPort**/actor-system-id	Independently settable per aggregation port?[5]
4	7	aAggPortActorAdminKey-rw	**aggPort**/actor-admin-key	Description not explicit on rw.
5	8	aAggActorOperKey-r	**aggPort**/actor-oper-key	
			phys-address	MAC Address in ‘real Ethernet’ attributes?
6	—	aAggPortPartnerAdminSystemPriority-rw	**aggPort**/partner-admin-system-priority	This and SystemID reordered from 7.3.1.
7	11[6]	aAggPortPartnerOperSystemPriority-r	**aggPort**/partner-oper-system-priority	
8	—	aAggPortPartnerAdminSystemID-rw	**aggPort**/partner-admin-system-id	
9	10[7]	aAggPortPartnerOperSystemID-rw	**aggPort**/partner-oper-system-id	
10	—	aAggPortPartnerAdminKey-rw	**aggPort**/partner-admin-key	
11	12[8]	aAggPortPartnerOperKey-r	**aggPort**/partner-oper-key	
12	—	aAggPortSelectedAggID-r	**aggPort**/selected-aggID	Trying to track a fleeting state?
13	—	aAggPortAttachedAggID	**aggPort**/attached-aggID	Read-only
14	—	aAggPortActorPort-r	**aggPort**/actor-port	Note: read-only ...
15	—	aAggPortActorPortPriority-rw	**aggPort**/actor-port-priority	... but this is read/write
16	?—	aAggPortPartnerAdminPort-rw	**aggPort**/partner-admin-port	
17	—	aAggPortPartnerOperPort-r	**aggPort**/partner-oper-port	Check IETF defn for this and all interface attributes
			speed	No estimate of equivalent speed?
18	?—	aAggPortPartnerAdminPortPriority-rw	**aggPort**/partner-admin-port-priority	
19	—	aAggPortPartnerOperPortPriority-rw	**aggPort**/partner-oper-port-priority	
20	—	aAggPortAdminState-rw	enabled[9]	
			admin-status	Read-only. Equivalent to ISS MAC_Enabled (per .1Xck)
21	—	aAggPortActorOperState-r[10]	oper-status	Equivalent to ISS MAC_Operational (per .1Xck)
22	—	aAggPortPartnerAdminState-rw	**aggPort**/partner-admin-enabled[11]	
23	—	aAggPortPartnerOperState-rw	**aggPort**/partner-enabled[12]	Some naming confusion here?
24	6	aAggPortAggregateOrIndividual-r	**aggPort**/aggregate-or-individual[13]	Repeat of aggregator information.
25	—[14]	aAggPortOperConversationPasses-r	**aggPort**/conversation-passes	
26	—"	aAggPortOperConversationCollected-r	**aggPort**/conversation-collected	
27	—	aAggPortAdminLinkNumberID[15]		Deprecated. Issue: scope uniqueness
28	—	aAggPortPartnerAdminLinkNumberID		Deprecated
29		aAggPortWTRTime-rw	**aggPort**/wtr-wait-time[16]	
			aggPort/wtr-revertive	Control Wtr_Revertive (Fig 6-16) is missing
30		aAggPortEnableLongPDUXmit		Deprecated
31		aAggOctetsPortOperLinkNumberID		Deprecated

[1] 7.3.2.1. subclause number.
[2] Entries in this column refer to 7.3.1.1 clauses (for Aggregator attributes) that duplicate the 7.3.2.1 clauses (for Aggregation port attributes) shown in the first column.
[3] Initial/working name suggestions.
[4] See note to Table-1 aAggSystemID. Clauses for SystemID and SystemPriority presented in reverse order to those for Aggregator.
[5] Is this really independently settable per aggregator? If not how is it set?
[6]
[7]
[8]
[9] What is the relationship to the IETF YANG interface admin-status (which is read-only) See also 1AC defn. of MAC_Enabled which is read only?
[10] Name is inconsistent with Aggregator’s OperState, which does not include ‘Actor.
[11] What is the relationship to the IETF YANG interface admin-status (which is read-only) See also 1AC defn. of MAC_Enabled which is read only?
[12] What is the relationship to the IETF YANG interface admin-status (which is read-only) See also 1AC defn. of MAC_Enabled which is read only?
[13] No strong opinion about suggested name
[14] Does this duplicate information obtainable from the Aggregator.
[15] Unclear why ‘Admin’ appears in this name. Don’t believe all writable attributes are named ‘admin’.
[16] Name chosen to match Figure 6-16

Table 4—Aggregation Port extension attributes (from P802.1AX D0.4 7.3.2.2.)

#[1]	D0.4 Clause 7 name	ietf-interfaces/interface/[2]	
1	aAggPortProtocolDA	aggPort/lacp-group-da[3]	

[1] 7.3.2.2. subclause number.
[2] Initial/working name suggestions.
[3] Just using ‘protocol-da’ seems a little obscure.

Table 5—Aggregation Port Statistics attributes (from P802.1AX D0.4 7.3.3.1.)

#[1]	D0.4 Clause 7 name	ietf-interfaces/interface/[2]	
1	aAggPortStatsID		The if-index for the Aggregation Port
2	aAggPortStatsLACPDUsRx	**agg-port-statistics**/lacpdus-rx	
3	aAggPortStatsMarkerPDUsRx	**agg-port-statistics**/markers-rx[3]	
4	aAggPortStatsMarkerResponsePDUsRx	**agg-port-statistics**/marker-responses-rx	
5	aAggPortStatsUnknownRx	statistics/in-unknown-protos	
6	aAggPortStatsIllegalRx	**agg-port-statistics**/not-legal-rx[4]	QA testing in the field? 50 per sec max?
7	aAggPortStatsLACPDUsTx	**agg-port-statistics**/lacpdus-tx	
8	aAggPortStatsMarkerPDUsTx	**agg-port-statistics**/markers-tx	
9	aAggPortStatsMarkerResponsePDUsTx	**agg-port-statistics**/marker-responses-tx	

[1] 7.3.3.1. subclause number.
[2] Initial/working name suggestions.
[3] Unlike ‘LACPDU’, ‘MarkerPDU’ is not one word, and this is just as clear leaving out ‘PDU’.
[4] Using ‘Illegal’ can be a problem as may typefaces do not clearly distinguish ‘I’ and ‘l’.