IEEE P802.11  
Wireless LANs

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| MIB Attributes Analysis | | | | | |
| Date: 2015-01-10 | | | | | |
| Author(s): | | | | | |
| Name | Affiliation | Address | Phone | email |
| Mark Hamilton | Spectralink, Corp | 2560 55th St  Boulder, CO 80301 USA | +1-303-441-7553 | mark.hamilton@spectralink.com |
|  |  |  |  |  |

Abstract

This document is a work-in-progress, of evaluating MIB attributes that might fit the “\*Implemented” or “\*Activated” usage patterns. The intent is to build a list of such patterns of use, as a foundation for creating a Design Pattern for such attributes.

Revision History

R0 – First revision.

R1 – Investigated many pairs of \*Implemented and \*Activated (where both are in the MIB for the same feature), and commented on lack of perceived value for this. Proposed first pattern “Pattern A”, which seems to cover these cases, with only one or the other.

R2 – Proposed “Pattern B”. Started one-by-one review of the listed attributes starting on page 2; commentary on each

R3 – Added more discussion and questions at the end (“To do” section)

R4 – Added some background from SNMP RFCs, and from 11-09/533r1, in response to questions raised in the last face-to-face (at the end of the document, in black text in response to red questions)

***MIB attributes of type “TruthValue”, with a start at analysis:***

* dot11CFPollable – Definitely an “Implemented/Activated” type of thing, but not spelled like one. Only by implication though (never stated) that this attribute is true in “CF-Pollable STA”s. Of course, this is really old, and we’ve changed style since then.
* dot11PrivacyOptionImplemented
* dot11PrivacyInvoked – These are really WEP only. dot11PrivacyOptionImplemented is described as a capability, although actually if it is true, the feature is operating. dot11PrivacyInvoked is described as a control attribute. It controls frame encryption, while dot11PrivacyOptionImplemented controls key management
* dot11MultiDomainCapabilityImplemented
* dot11MultiDomainCapabilityActivated – No value evident in having both of these. This is only one of many examples where the \*Implemented attribute does not seem to have any added value, and it is not clear why the \*Activated attribute would ever change during operation.
* dot11SpectrumManagementImplemented
* dot11SpectrumManagementRequired - Definitely an “Implemented/Activated” type of thing, but not spelled like one
* dot11RSNAOptionImplemented
* dot11RSNAPreauthenticationImplemented
* dot11OperatingClassesImplemented
* dot11OperatingClassesRequired
* dot11QosOptionImplemented
* dot11ImmediateBlockAckOptionImplemented
* dot11DelayedBlockAckOptionImplemented
* dot11DirectOptionImplemented
* dot11APSDOptionImplemented
* dot11QAckOptionImplemented
* dot11QBSSLoadImplemented
* dot11QueueRequestOptionImplemented
* dot11TXOPRequestOptionImplemented
* dot11MoreDataAckOptionImplemented
* dot11AssociateInNQBSS - Definitely an “Implemented/Activated” type of thing, but not spelled like one. Never used outside the MIB.
* dot11DLSAllowedInQBSS - Definitely an “Implemented/Activated” type of thing, but not spelled like one. Usage (and relation to QoS) is pretty confusing.
* dot11DLSAllowed - Definitely an “Implemented/Activated” type of thing, but not spelled like one.
* dot11RadioMeasurementImplemented
* dot11RadioMeasurementActivated
* dot11RMLinkMeasurementActivated
* dot11RMNeighborReportActivated
* dot11RMParallelMeasurementsActivated
* dot11RMRepeatedMeasurementsActivated
* dot11RMBeaconPassiveMeasurementActivated
* dot11RMBeaconActiveMeasurementActivated
* dot11RMBeaconTableMeasurementActivated
* dot11RMBeaconMeasurementReportingConditionsActivated
* dot11RMFrameMeasurementActivated
* dot11RMChannelLoadMeasurementActivated
* dot11RMNoiseHistogramMeasurementActivated
* dot11RMStatisticsMeasurementActivated
* dot11RMLCIMeasurementActivated
* dot11RMLCIAzimuthActivated
* dot11RMTransmitStreamCategoryMeasurementActivated
* dot11RMTriggeredTransmitStreamCategoryMeasurementActivated
* dot11RMAPChannelReportActivated
* dot11RMMIBActivated
* dot11RMMeasurementPilotTransmissionInformationActivated
* dot11RMNeighborReportTSFOffsetActivated
* dot11RMRCPIMeasurementActivated
* dot11RMRSNIMeasurementActivated
* dot11RMBSSAverageAccessDelayActivated
* dot11RMBSSAvailableAdmissionCapacityActivated
* dot11RMAntennaInformationActivated
* dot11FastBSSTransitionImplemented
* dot11GASPauseForServerResponse – Does affect protocol behavior, in a sense similar to an “\*Implemented”, but not spelled like one.
* dot11TxAntennaImplemented – Never used except in DSSS and ERP PHY characteristics.

***MIB attributes of the form “\*Implemented” that are not of type TruthValue and some notes on them. They may reflect more information, such as “how much/many of X is implemented?”***

* dot11RSNAConfigPairwiseCipherImplemented OCTET STRING
* dot11RSNAConfigPairwiseCipherSizeImplemented Unsigned32
  + Above used in dot11RSNAConfigPairwiseCiphersTable, which is never referenced. The MIB claims this is used by an external management entity: “The pairwise cipher suite list in the RSNE is formed using the information in this table.” How this is accomplished appears to be missing.
* dot11TVHTMUMaxUsersImplemented – Never used except in the MIB
* dot11WEPKeyMappingLengthImplemented – OK, this is an implementation imposed limit and discussed in text, but doesn’t affect over-the-air signaling.
* dot11NumberSupportedPowerLevelsImplemented – Never used except in the DSSS, HR and HT PHY characteristics

***MIB attributes of type “TruthValue” still to be checked into:***

* dot11LCIDSEImplemented
* dot11LCIDSERequired
* dot11DSERequired
* dot11ExtendedChannelSwitchActivated
* dot11RSNAProtectedManagementFramesActivated
* dot11RSNAUnprotectedManagementFramesAllowed
* dot11HighThroughputOptionImplemented
* dot11RSNAPBACRequired
* dot11PSMPOptionImplemented
* dot11TunneledDirectLinkSetupImplemented
* dot11TDLSPeerUAPSDBufferSTAActivated
* dot11TDLSPeerPSMActivated
* dot11TDLSChannelSwitchingActivated
* dot11OCBActivated
* dot11WirelessManagementImplemented
* dot11RMCivicMeasurementActivated
* dot11RMIdentifierMeasurementActivated
* dot11RM3rdPartyMeasurementActivated
* dot11InterworkingServiceImplemented
* dot11InterworkingServiceActivated
* dot11QosMapImplemented
* dot11QosMapActivated
* dot11EBRImplemented
* dot11EBRActivated
* dot11ESNetwork
* dot11SSPNInterfaceImplemented
* dot11SSPNInterfaceActivated
* dot11EASImplemented
* dot11EASActivated
* dot11MSGCFImplemented
* dot11MSGCFActivated
* dot11MeshActivated
* dot11RejectUnadmittedTraffic
* dot11QMFActivated
* dot11QMFReconfigurationActivated
* dot11RobustAVStreamingImplemented
* dot11MultibandImplemented
* dot11DynamicEIFSActivated
* dot11VHTOptionImplemented
* dot11OperatingModeNotificationImplemented
* dot11TVHTOptionImplemented
* dot11ChannelScheduleManagementActivated
* dot11ContactVerificationSignalActivated
* dot11NetworkChannelControlActivated
* dot11RLSSActivated
* dot11WhiteSpaceMapActivated
* dot11GeolocationCapabilityActivated
* dot11GDDActivated
* dot11AuthenticationAlgorithmsActivated
* dot11WEPKeyMappingWEPOn
* dot11PrivacyInvoked
* dot11ExcludeUnencrypted
* dot11RSNAActivated
* dot11RSNAPreauthenticationActivated
* dot11RSNAConfigGroupRekeyStrict
* dot11RSNAConfigPairwiseCipherActivated
* dot11RSNAConfigAuthenticationSuiteActivated
* dot11FastBSSTransitionActivated
* dot11FTOverDSActivated
* dot11FTResourceRequestSupported
* dot11RegLocAgreement
* dot11RegLocDSE
* dot11DependentSTA
* dot11NDelayedBlockAckOptionImplemented
* dot11STBCControlFrameOptionImplemented
* dot11LsigTxopProtectionOptionImplemented
* dot11PCOOptionImplemented
* dot11HTControlFieldSupported
* dot11RDResponderOptionImplemented
* dot11SPPAMSDUCapable
* dot11SPPAMSDURequired
* dot11FortyMHzOptionImplemented
* dot11LocationActivated
* dot11FMSImplemented
* dot11FMSActivated
* dot11EventsActivated
* dot11DiagnosticsActivated
* dot11MultiBSSIDImplemented
* dot11MultiBSSIDActivated
* dot11TFSImplemented
* dot11TFSActivated
* dot11WNMSleepModeImplemented
* dot11WNMSleepModeActivated
* dot11TIMBroadcastImplemented
* dot11TIMBroadcastActivated
* dot11ProxyARPImplemented
* dot11ProxyARPActivated
* dot11BSSTransitionImplemented
* dot11BSSTransitionActivated
* dot11QoSTrafficCapabilityImplemented
* dot11QoSTrafficCapabilityActivated
* dot11ACStationCountImplemented
* dot11ACStationCountActivated
* dot11CoLocIntfReportingImplemented
* dot11CoLocIntfReportingActivated
* dot11MotionDetectionImplemented
* dot11MotionDetectionActivated
* dot11TODImplemented
* dot11TODActivated
* dot11TimingMsmtImplemented
* dot11TimingMsmtActivated
* dot11ChannelUsageImplemented
* dot11ChannelUsageActivated
* dot11TriggerSTAStatisticsActivated
* dot11SSIDListImplemented
* dot11SSIDListActivated
* dot11MulticastDiagnosticsActivated
* dot11LocationTrackingImplemented
* dot11LocationTrackingActivated
* dot11DMSImplemented
* dot11DMSActivated
* dot11UAPSDCoexistenceImplemented
* dot11UAPSDCoexistenceActivated
* dot11WNMNotificationImplemented
* dot11WNMNotificationActivated
* dot11UTCTSFOffsetImplemented
* dot11UTCTSFOffsetActivated
* dot11FineTimingMsmtImplemented
* dot11FineTimingRespActivated
* dot11FineTimingInitActivated
* dot11LciCivicInNeighborReport
* dot11RMFineTimingMsmtRangeRepImplemented
* dot11RMFineTimingMsmtRangeRepActivated
* dot11RMRqstParallel
* dot11RMRqstEnable
* dot11RMRqstRequest
* dot11RMRqstReport
* dot11RMRqstDurationMandatory
* dot11RMRqstTrigdQoSAverageCondition
* dot11RMRqstTrigdQoSConsecutiveCondition
* dot11RMRqstTrigdQoSDelayCondition
* dot11RMNeighborReportSecurity
* dot11RMNeighborReportCapSpectrumMgmt
* dot11RMNeighborReportCapQoS
* dot11RMNeighborReportCapAPSD
* dot11RMNeighborReportCapRM
* dot11RMNeighborReportCapDelayBlockAck
* dot11RMNeighborReportCapImmediateBlockAck
* dot11RMNeighborReportKeyScope
* dot11RMNeighborReportMobilityDomain
* dot11RMNeighborReportCapHT
* dot11RMNeighborReportHTLDPCCodingCap
* dot11RMNeighborReportHTSupportedChannelWidthSet
* dot11RMNeighborReportHTGreenfield
* dot11RMNeighborReportHTShortGIfor20MHz
* dot11RMNeighborReportHTShortGIfor40MHz
* dot11RMNeighborReportHTTxSTBC
* dot11RMNeighborReportHTDelayedBlockAck
* dot11RMNeighborReportHTMaxAMSDULength
* dot11RMNeighborReportHTDSSCCKModein40MHz
* dot11RMNeighborReportHTFortyMHzIntolerant
* dot11RMNeighborReportHTLSIGTXOPProtectionSupport
* dot11RMNeighborReportHTTxMCSSetDefined
* dot11RMNeighborReportHTTxRxMCSSetNotEqual
* dot11RMNeighborReportHTTxUnequalModulationSupported
* dot11RMNeighborReportHTPCO
* dot11RMNeighborReportHTCSupport
* dot11RMNeighborReportHTRDResponder
* dot11RMNeighborReportHTImplictTransmitBeamformingReceivingCap
* dot11RMNeighborReportHTReceiveStaggeredSoundingCap
* dot11RMNeighborReportHTTransmitStaggeredSoundingCap
* dot11RMNeighborReportHTReceiveNDPCap
* dot11RMNeighborReportHTTransmitNDPCap
* dot11RMNeighborReportHTImplicitTransmitBeamformingCap
* dot11RMNeighborReportHTExplicitCSITransmitBeamformingCap
* dot11RMNeighborReportHTExplicitNonCompressedSteeringCap
* dot11RMNeighborReportHTExplicitCompressedSteeringCap
* dot11RMNeighborReportHTAntSelectionCap
* dot11RMNeighborReportHTExplicitCSIFeedbackBasedTxASELCap
* dot11RMNeighborReportHTAntIndicesFeedbackBasedTxASELCap
* dot11RMNeighborReportHTExplicitCSIFeedbackBasedCap
* dot11RMNeighborReportHTAntIndicesFeedbackCap
* dot11RMNeighborReportHTRxASELCap
* dot11RMNeighborReportHTTxSoundingPPDUsCap
* dot11RMNeighborReportHTInfoSTAChannelWidth
* dot11RMNeighborReportHTInfoRIFSMode
* dot11RMNeighborReportHTInfoNonGreenfieldHTSTAsPresent
* dot11RMNeighborReportHTInfoOBSSNonHTSTAsPresent
* dot11RMNeighborReportHTInfoDualBeacon
* dot11RMNeighborReportHTInfoDualCTSProtection
* dot11RMNeighborReportHTInfoSTBCBeacon
* dot11RMNeighborReportHTInfoLSIGTXOPProtectionSup
* dot11RMNeighborReportHTInfoPCOActive
* dot11RMNeighborReportHTInfoPCOPhase
* dot11RMNeighborReportExtCapPSMPSupport
* dot11RMNeighborReportExtCapSPSMPSup
* dot11WNMRqstColocInterfAutoEnable
* dot11MeshAcceptingAdditionalPeerings
* dot11MeshConnectedToMeshGate
* dot11MeshSecurityActivated
* dot11MeshForwarding
* dot11MeshGateAnnouncements
* dot11MBCAActivated
* dot11MCCAImplemented
* dot11MCCAActivated
* dot11DMGOptionImplemented
* dot11RelayActivated
* dot11REDSActivated
* dot11RDSActivated
* dot11MultipleMACActivated
* dot11ClusteringActivated
* dot11GCRActivated
* dot11AdvancedGCRImplemented
* dot11AdvancedGCRActivated
* dot11SCSImplemented
* dot11SCSActivated
* dot11QLoadReportActivated
* dot11AlternateEDCAActivated
* dot11GCRGroupMembershipAnnouncementActivated
* dot11PublicHCCATXOPNegotiationImplemented
* dot11PublicHCCATXOPNegotiationActivated
* dot11ProtectedHCCATXOPNegotiationImplemented
* dot11ProtectedHCCATXOPNegotiationActivated
* dot11ProtectedQLoadReportImplemented
* dot11ProtectedQLoadReportActivated
* dot11MeshGCRImplemented
* dot11MeshGCRActivated
* dot11VHTControlFieldOptionImplemented
* dot11VHTTXOPPowerSaveOptionImplemented
* dot11RIFSMode
* dot11PSMPControlledAccess
* dot11DualCTSProtection
* dot11LSIGTXOPFullProtectionActivated
* dot11NonGFEntitiesPresent
* dot11PCOActivated
* dot11FortyMHzIntolerant
* dot112040BSSCoexistenceManagementSupport
* dot11EDCATableMandatory
* dot11QAPEDCATableMandatory
* dot11AntennaSelectionOptionImplemented
* dot11TransmitExplicitCSIFeedbackASOptionImplemented
* dot11TransmitIndicesFeedbackASOptionImplemented
* dot11ExplicitCSIFeedbackASOptionImplemented
* dot11TransmitIndicesComputationASOptionImplemented
* dot11ReceiveAntennaSelectionOptionImplemented
* dot11TransmitSoundingPPDUOptionImplemented
* dot11AntennaSelectionOptionImplemented
* dot11TransmitExplicitCSIFeedbackASOptionImplemented
* dot11TransmitIndicesFeedbackASOptionImplemented
* dot11ExplicitCSIFeedbackASOptionImplemented
* dot11TransmitIndicesComputationASOptionImplemented
* dot11ReceiveAntennaSelectionOptionImplemented
* dot11TransmitSoundingPPDUOptionImplemented
* dot11TxAntennaImplemented
* dot11RxAntennaImplemented
* dot11DiversitySelectionRxImplemented
* dot11FiveMHzOperationImplemented
* dot11TenMHzOperationImplemented
* dot11TwentyMHzOperationImplemented
* dot11OFDMCCAEDImplemented
* dot11OFDMCCAEDRequired
* dot11ShortPreambleOptionImplemented
* dot11ShortSlotTimeOptionImplemented
* dot11ShortSlotTimeOptionActivated
* dot11FortyMHzOperationImplemented
* dot11FortyMHzOperationActivated
* dot11HTGreenfieldOptionImplemented
* dot11HTGreenfieldOptionActivated
* dot11ShortGIOptionInTwentyImplemented
* dot11ShortGIOptionInTwentyActivated
* dot11ShortGIOptionInFortyImplemented
* dot11ShortGIOptionInFortyActivated
* dot11LDPCCodingOptionImplemented
* dot11LDPCCodingOptionActivated
* dot11TxSTBCOptionImplemented
* dot11TxSTBCOptionActivated
* dot11RxSTBCOptionImplemented
* dot11RxSTBCOptionActivated
* dot11BeamFormingOptionImplemented
* dot11BeamFormingOptionActivated
* dot11TxMCSSetDefined
* dot11TxRxMCSSetNotEqual
* dot11TxUnequalModulationSupported
* dot11ReceiveStaggerSoundingOptionImplemented
* dot11TransmitStaggerSoundingOptionImplemented
* dot11ReceiveNDPOptionImplemented
* dot11TransmitNDPOptionImplemented
* dot11ImplicitTransmitBeamformingOptionImplemented
* dot11ExplicitCSITransmitBeamformingOptionImplemented
* dot11ExplicitNonCompressedBeamformingMatrixOptionImplemented
* dot11LowPowerSCPHYImplemented
* dot11LowPowerSCPHYActivated
* dot11VHTShortGIOptionIn80Implemented
* dot11VHTShortGIOptionIn80Activated
* dot11VHTShortGIOptionIn160and80p80Implemented
* dot11VHTShortGIOptionIn160and80p80Activated
* dot11VHTLDPCCodingOptionImplemented
* dot11VHTLDPCCodingOptionActivated
* dot11VHTTxSTBCOptionImplemented
* dot11VHTTxSTBCOptionActivated
* dot11VHTRxSTBCOptionImplemented
* dot11VHTRxSTBCOptionActivated
* dot11VHTSUBeamformeeOptionImplemented
* dot11VHTSUBeamformerOptionImplemented
* dot11VHTMUBeamformeeOptionImplemented
* dot11VHTMUBeamformerOptionImplemented
* dot11TVHTShortGIOptionIn4WImplemented
* dot11TVHTShortGIOptionIn4WActivated
* dot11TVHTLDPCCodingOptionImplemented
* dot11TVHTLDPCCodingOptionActivated
* dot11TVHTTxSTBCOptionImplemented
* dot11TVHTTxSTBCOptionActivated
* dot11TVHTRxSTBCOptionImplemented
* dot11TVHTRxSTBCOptionActivated
* dot11TVHTSUBeamformeeOptionImplemented
* dot11TVHTSUBeamformerOptionImplemented
* dot11TVHTMUBeamformeeOptionImplemented
* dot11TVHTMUBeamformerOptionImplemented
* dot11NonAPStationAuthHCCAHEMM
* dot11NonAPStationAuthSourceMulticast
* dot11NonAPStationAuthDls

**Patterns of Usage**

This section has a start at some structure for “Patterns of Usage for MIB attributes indicating support or activation”.

Start with a reminder of key concepts, terminology and recommendations from 2009 review. See 11-09/533r1, especially slides 4, 5, 12, 13, 14, 15, 17.

Per discussion during August face-to-face meeting, Patterns should be based on overall system behaviour. A particular type of usage then maps to a collection of one or more MIB attributes and recommended naming conventions, as well as text description of behaviour for each component in the system, to realize the overall pattern.

*Starting to collect examples, below. Discuss if these have the right level of granularity, and if the right descriptive elements are included.*

**Pattern A**

Simple – Static feature that involves frames or information that is optionally exchanged (one-way or two-way) between peers.

Recommended pattern:

* MIB attribute: dot11<feature>Implemented, and a capability attribute, usually. Could be dot11<feature>Activated (and a control attribute) if the feature is enabled by an external entity even though static during operation (for example, set at initialization). Care should be given to the latter case, to ensure that there is justification that this attribute is changeable, and not simply a statement of implementation capability and therefore a dot11<feature>Implemented attribute.
* Results in information included in all or a subset of Beacon, Probe Request and Response, (Re)Association Request and Response, and Mesh Peering frames.
* Information can be conditionally present in primitives as appropriate to support its inclusion in the frames above.
* Indicated with a bit in Capability elements, or by inclusion of optional element(s) in the above frames.
* The feature can also include additional frame (or Action frame) types transmitted. These could be unilaterally sent if the transmitter is using the feature (and the receiver will ignore it if it’s not understood).or could be sent only if both peers have indicated the feature is being used.
* A STA with the feature in use can be referred to as a “<feature> STA.”

Examples: dot11WirelessManagementImplemented, dot11ExtendedChannelSwitchActivated, dot11QMFActivated

Notes: There is no need or value in both dot11<feature>Implemented and dot11<feature>Activated MIB attributes; only one should be used. (See dot11InterworkingServiceImplemented, dot11QosMapImplemented and many more as examples that should be eliminated.)

**Pattern B**

Dynamic, slow changing – Dynamic feature, but that only changes when a BSS is terminated, or before a STA joins a BSS. Generally, involves frames or information that is exchanged (one-way or two-way) between peers.

Recommended pattern:

* MIB attribute: dot11<feature>Activated. Care should be given to ensure that there is justification that this attribute is changeable, and not simply a statement of implementation capability and therefore a dot11<feature>Implemented attribute.
* Written by an external management entity (a control attribute)
* Results in information included in all or a subset of Beacon, Probe Request and Response, (Re)Association Request and Response, and Mesh Peering frames.
* Information can be conditionally present in primitives as appropriate to support its inclusion in the frames above.
* Indicated with a bit in Capability elements, or by inclusion of optional element(s) in the above frames.
* The feature can also include additional frame (or Action frame) types transmitted. These could be unilaterally sent if the transmitter is using the feature (and the receiver will ignore it if it’s not understood).or could be sent only if both peers have indicated the feature is being used.
* A STA with the feature in use can be referred to as a “<feature> STA.”

Examples: x, y, z

Notes: There is no need or value in both dot11<feature>Implemented and dot11<feature>Activated MIB attributes; only one should be used. (See x, y, z, and many more as examples that should be eliminated.)

**To Do:**

No examples found yet of needing both \*Implemented and \*Activated for the same feature.

*Are there three concepts, and are all three necessary/useful/relevant to the scope of the Standard: “hard-wired/manufactured ‘capable’”, “’enabled’, by something/someone, at say, power on”, and “’activated’ dynamically”*

*Find examples of \*Activated. There are examples where the change takes effect “as soon as practical”, and examples where the change takes effect at the next MLME-Start or MLME-JOIN. (And, examples where it is not specified at all… ☹ ) Are both the first two types actually meaningful/useful in the Standard?*

*Finish discussion: If there is only dot11XxxActivated, can the external management entity try to read it, and determine if it is implemented by whether that read returns with an error or not? How about if it tries to write to it – same thing?*

From RFC 1157 (SNMP):

(1) if said variable is defined in the MIB with "Access:" of

"none," it is unavailable as an operand for any operator;

(2) if said variable is defined in the MIB with "Access:" of

"read-write" or "write-only" and the access mode of the

given profile is READ-WRITE, that variable is available

as an operand for the get, set, and trap operations;

(3) otherwise, the variable is available as an operand for

the get and trap operations.

Upon receipt of the GetRequest-PDU, the receiving protocol entity

responds according to any applicable rule in the list below:

(1) If, for any object named in the variable-bindings field,

the object's name does not exactly match the name of some

object available for get operations in the relevant MIB

view, then the receiving entity sends to the originator

of the received message the GetResponse-PDU of identical

form, except that the value of the error-status field is

noSuchName, and the value of the error-index field is the

index of said object name component in the received

message.

The SetRequest-PDU has a very similar rule, also showing the return of a noSuchName error.

It seems that, yes, we can assume a dot11XxxActivated attribute will return an error upon either read (get) or write (set) operations, if the device does not implement the Xxx feature.

*“MAX\_ACCESS” in MIB: what does this mean? Access given to which entity (any entity other than the “owner”)? Do the SNMP RFCs give any guidance?*

From RFC 2578 (one of the SMIv2 RFCs):

Mapping of the MAX-ACCESS clause:

The MAX-ACCESS clause, which must be present, defines whether it

makes "protocol sense" to read, write and/or create an instance of

the object, or to include its value in a notification.

The value of the MAX-ACCESS clause for objects with a SYNTAX clause

value of Counter32 is either "read-only" or "accessible-for-notify".

In general, the SMI and SNMP RFCs seem to use the “ACCESS” (and “MAX-ACCESS” and “MIN-ACCESS”) clause in reference to the type of access provided by the “agent”, and the “agent” is generally the device (or that portion of the device) that provides SNMP access to the MIB.

*Check on this from 11-09/533r1: Status: dot11XxxActivated (capability that is enabled). (Status is written by the entity itself, not an external entity) However, most (all?) examples are written by the SME or other external entity. Did we change our minds on this name? Should we?*

11-09/533r1 used the category Control attribute for anything written by an external entity to control the operation of a device. By extension, this would apply to externally writing an attribute to turn on (“enable”) a capability. However, this usage (for a feature capability) was not explicitly anticipated by 11-09/533r1. dot11XxxActivated was defined in that document as a Status attribute – that is, it would indicate to the external entity that a feature had been turned on, but did not support the external entity writing to it, so the assumption must be that this is a dynamic state (otherwise it would be a Capability) but one which is changed internally to the entity.

The current usage in the MIB is for “Activated” to cover both cases.

There is very little usage of “Enabled” in the MIB (really, only one, at is a different usage: dot11RMNeighborReportRMEnabledCapabilities).

The case of the entity internally turning on a feature seems much less common. It could be imagined for some features that are dependent on a peer, like Short Slot Time, or DSE, that a “dot11XxxEnabled” MIB attribute might be useful, which indicates that the feature is currently in operation based on logic within the MAC/PHY/MLME directing that it should be (probably after protocol exchange with a peer).

Then, “dot11XxxActivated” could be used only for those features which are controlled by an external entity.

With this scheme, there is no need for “do11XxxImplemented” along with “dot11XxxActivated” for the same Xxx feature.

However, “dot11XxxImplemented” might be useful to have in pair with “dot11XxxEnabled” attributes, so an external entity can tell if the feature is supported but not currently in operation.

And, yes, this is a change to the decisions in 11-09/533r1, although it is really a refinement, having recognized a gap in the logic of that document.

*Is there a difference between an externally set, at initialization, control over the activation/enablement of a feature, and an externally reported but not necessary for interoperability piece of information? Example, supported rates versus 11k counters.*