ember

Release Notes

Product: AppFramework V2 GA for EmberZNet 4.5.2

Release date: June 27th, 2011.

1 Overview

This release contains the GA release of Application Framework V2 (AFV2). AFV2 must be used with EmberZNet 4.5.2 for the EM357. This must be installed at the same location as the stack it will be used with. This must be used with InSight AppBuilder 2.1 build 96 or greater.

Application Framework V2 (AFV2) improves on the V1 framework by adding multiple endpoint support, a callback structure that separates framework code from custom code, support for describing custom clusters in XML, tokenized ZCL attributes, external storage of ZCL attributes, sharing of attribute data across endpoints, and an improved command-line interface (CLI). Please see the "Features" section of the release notes for details.

When InSight AppBuilder configures V2 projects, these use the code contained in "app/framework" instead of "app/ha" (which is used for V1). Project files and configuration files generated by AppBuilder for AFV2 go into "app/builder/{projectName}"

2 Getting Started

The AppFramework V2 (AFV2) source code is bundled with the EmberZNet 4.5.0 installer

Once the EmberZNet 4.5.0 stack is installed, InSight AppBuilder needs to be restarted in order for the change in stack configuration data to be recognized. Once AFV2 has been installed, follow these steps to start an AFV2 project:

1) Start (or restart) InSight AppBuilder by selecting "Start → All Programs → Ember → InSight Desktop → InSight Desktop"

2) If the stack supporting AFV2 has not been added to InSight Desktop do that now. To check if the stack has been added or to add the stack select "File → Preferences" from the drop down menu. Then select "AppBuilder" from the left hand side and check the "available stacks" window. If the stack for AFV2 is not there, press the "add" button and select the directory where the stack is installed (for instance: "C:\Program Files\Ember\ EmberZNet4.5.0\ em250")

3) Select “File → New → ZigBee Application Configuration” from the drop down menu.

4) From the “Installed Stack Selection” dialog box that comes up, select the stack where the AFV2 has been added to, for instance: “EmberZNet 4.5.0 EM250”.

5) From the “Application Template selection” dialog box that comes up, select “ZCL Application Framework V2”.

2.1 Support

Development Kit customers are eligible for training and technical support. You can use the Ember web site http://www.ember.com to obtain information about all Ember products and services, and to sign up for product support.

You can contact Ember technical support at http://portal.ember.com.

3 Features

3.1 Existing AFV2 GA Features

- **Manufacturer Specific Clusters, Attributes and Commands –** The Application Framework and AppBuilder have been augmented to support manufacturer specific attributes and commands which overlap the id space of the existing ZigBee attributes and commands. A new chapter on this subject has been added to the Application Framework Developer Guide (120_7011_000_ApplicationFrameworkV2DeveloperGuide.pdf).
- **Plugins –** All of the Ember developed cluster code has been moved into plugins located in app/framework/plugin. The plugins can be optionally included in your application from within the AppBuilder interface. Any code which did nothing more than print out the incoming data has been removed. If you wish to print out incoming data and it is not handled by a plugin you may implement a command handling callback for the data you wish to print out. For more information on plugins please see the Application Framework Developer Guide. (120_7011_000_ApplicationFrameworkV2DeveloperGuide.pdf)
- **Events instead of Ticks –** The “tick” mechanism which drove the cluster’s timed events in the application framework has been removed**.** The tick mechanism did not work well for sleepy devices as the knowledge of when a certain action was going to take place was decentralized and only known by the cluster code. The event mechanism provides a centralized repository of all actions scheduled for the application. As a result, when a sleepy device is determining how long it can sleep for, it is able to query the event mechanism to find out when the next event is scheduled to fire. The device will only sleep for the returned duration so that it will be sure to wake up in time to fire the event. For more information on events please see the Application Framework Developer Guide. (120_7011_000_ApplicationFrameworkV2DeveloperGuide.pdf)

- **Default response handling –** The application framework no longer automatically sends a default response for command callbacks implemented by the application. It will send the default response for any command callbacks implemented by a plugin and for any command which is not implemented by the application. For more information on default response please see the Application Framework Developer Guide. (120_7011_000_ApplicationFrameworkV2DeveloperGuide.pdf)
- **Multiple endpoint support –** the V2 framework supports multiple endpoints which are configured through AppBuilder. This means multiple ZigBee devices can be supported with a single radio. An example of this is a device with three lights and one radio. Each light would be assigned a unique endpoint. The first endpoint listed in the AppBuilder GUI controls items such as security which must be set according to one profile (in the case where the project supports devices from different application profiles). Multiple endpoints are added in the "ZCL cluster configuration" tab within AppBuilder.
- **Callback structure / Code Separation**– the V2 framework provides callbacks for interesting events, such as: when an attribute changes, when a message is received, when a device wakes up, when a CLI (command line interface) command has been executed (in order to extend the CLI). This allows the framework code to be separate from any custom code that is written so upgrades to new versions of the framework are simple and straight-forward. Callbacks are configured through the "callback configuration" tab within AppBuilder. Library code that should not be modified is in the "app/framework" directory. Custom code that relates to a specific project is in the "app/builder/{projectName}" directory.
- **Support for custom clusters in XML –** This feature is described in further detail in Application Framework Developer Guide Chapter 16, (120_7011_000_ApplicationFrameworkV2DeveloperGuide.pdf).
    - An XML file that describes custom clusters can be supplied to AppBuilder. The custom clusters show up in the AppBuilder choices and the necessary code is added to your project. For clusters that support only attributes no additional code is required. For clusters that support receiving commands or want to execute code when they start (clusterInit) or at regular intervals (clusterTick), custom functions must be provided that are then called from the framework. An example of the XML files can be found in the "tool/appbuilder" directory. To add custom clusters to your project select "file → preferences → AppBuilder" from within InSight Desktop. Check the "enable custom clusters" box, add your custom XML file using the "New" button. Generating framework code from an XML schema files is different than generating an application project. Once you have loaded your XML schema file into AppBuilder preferences you need to regenerate a portion of the AppFramework by clicking on the green arrow on the AppBuilder preferences page (the same place that the new XML file is added). The framework code generated from the XML file must go into "app/framework/gen" (the default path) in order to work properly.
- **Tokenized ZCL attributes –** The framework now supports storing ZCL attributes in non-volatile storage (tokens). Attribute values are saved across reboots. To enable an

attribute to be stored in a token, find the attribute in the "attribute table" on the "ZCL cluster configuration" tab within AppBuilder and check the box in the column marked "F". Once attributes are tokenized, they are written into flash which has a limited number of writes for its lifetime. Use caution when choosing which attributes will be written into flash since each time the attribute changes, the token will be re-written. Future releases will provide a method of limiting the number of writes for an attribute into a token.

3.2 **Improved command-line interface (CLI)** – the CLI now contains online help. Typing an incorrect command shows all the possible top-level commands. Typing a top level commands shows the next level of command options. The CLI now takes integer arguments in two ways: as a hex number (0x1ABC) or as a decimal number (123). The CLI takes string arguments in two ways: as ascii text ("foo") or as raw hex values ({0A 1B 2C}). This feature is described in further detail in the Application Framework CLI documentation (120-3027-000)

4 Application Framework API Changes for the 4.5 release

4.1 APIs Deprecated / Removed

4.1.1 *emberAfSendRequest is deprecated*

- **Change description:** emberAfSendRequest is deprecated and will be removed in the next release.
- **Customer process:** Use emberAfSendCommandUnicast instead to unicast the message to a specific node or binding or address table index. If using the emberAfFillCommand macros to construct the outgoing message, the sequence number will be set automatically. Otherwise, emberAfNextSequence can be used to retrieve a new sequence number for the message.

4.1.2 *emberAfSetEndpointsForResponse is deprecated*

- **Change description:** emberAfSetEndpointsForResponse is deprecated and will be removed in the next release.
- **Customer process:** When an incoming message is received, the framework will prepare itself for sending a response. Customers sending synchronous responses (e.g., within IncomingMessageHandler) won't notice any changes. Customers who need to send asynchronous responses can prepare a response buffer and APS frame manually and call emberAfSendUnicast directly.

4.1.3 *emberAfSetReplyForInterPAN is deprecated*

- **Change description:** emberAfSetReplyForInterPAN is deprecated and will be removed in the next release.
- **Customer process:** When an incoming inter-PAN message is received, the framework will prepare itself for sending an inter-PAN response. Customers sending synchronous responses (e.g., within IncomingMessageHandler) won't notice any changes. Customers who need to send asynchronous responses can prepare a response buffer manually and call emberAfSendInterPan directly.

4.1.4 *emberAfSendInterpanUnicast is deprecated*

- **Change description:** emberAfSendInterpanUnicast has been deprecated and replaced by emberAfSendInterPan. emberAfSendInterpanUnicast could be used to send inter-PAN unicast, multicast, or broadcasts, so it was renamed to better reflect its use.
- **Customer process:** Replace calls to emberAfSendInterpanUnicast with emberAfSendInterPan.

4.1.5 *emberAfSendCommandInterpanUnicast is deprecated*

- **Change description:** emberAfSendCommandInterpanUnicast has been deprecated and replaced by emberAfSendCommandInterPan. emberAfSendCommandInterpanUnicast could be used to send inter-PAN unicast, multicast, or broadcasts, so it was renamed to better reflect its use.
- **Customer process:** Replace calls to emberAfSendCommandInterpanUnicast with emberAfSendCommandInterPan.

4.2 APIs Modified

4.2.1 *Update ZDO requests for service discovery to use proper app profile ID for cluster being discovered*

- **Change description:** Add emberAfFindDevicesByProfileAndCluster as improved service discovery API that allows caller to specify app profile ID; make emberAFindDevicesSupportingCluster into macro for this new API. Add EmberAfProfileId for convenience in passing app profiles as parameters. Update all sample apps and plugins to make use of this new API. Replace all TRUE/FALSE literals in emberAfFindDevicesXXX calls with friendlier EMBER_AF_CLIENT_CLUSTER_DISCOVERY / EMBER_AF_SERVER_CLUSTER_DISCOVERY aliases. Update "option discover" CLI command to use new API and update AFV2 CLI HTML documentation accordingly. Clean up Doxgen comments around emberAfFindDevicesXXX APIs. Update ota-client-test module to reflect API change. All afv2-tests passing at check-in time.

- **Customer process:** Shift from calling emberAFindDevicesSupportingCluster() (now deprecated but supported through a macro) to calling emberAfFindDevicesByProfileAndCluster() with a specific profile ID, such that the matching works correctly when using different profiles on different endpoints. Shift from using int16u to using EmberAfProfileId when referencing app profile IDs in code.

4.2.2 *Update emberAfSendXXX functions to properly set application profile ID in APS frame*

- **Change description:** More fixing of profile IDs in APS frames. The AF's emberAfSendUnicast primitive now determines the right app profile ID for the outgoing message and sets it using a new function: emberAfSetProfileIdFromEndpoint. Summary of changes are as follows... On the EM35x SoC platform, these changes save 56 bytes of code for HA targets and 20 bytes of code for SE targets:
- New emberAfSetProfileIdFromEndpoint function added to assist in setting app profile ID in an APS frame based on its source endpoint's descriptor.
- APS security option bit setting now determined in emberAfSendUnicast() instead of in higher-level emberAfSendCommandUnicast wrapper; this effectively moves all the AF outgoing message "magic" into emberAfSendUnicast() so that all the SendCommandXXX functions do is tie the message to a global command frame.
- emberAfDetermineIfLinkSecurityIsRequired() now takes an additional argument for application profile ID, allowing it to support the future use case of having SE clusters used (without APS security) in non-SE profiles.
- **Customer process:** No need for customer to set up the app profile anymore before calling emberAfSendUnicast or emberAfSendCommand unicast.

4.2.3 *Fix emberAfCopyString*

- **Change description:** The existing emberAfCopyString API was a simple macro wrapping a MEMCOPY. It did not correctly calculate the length of the source string or the size of the destination buffer. It took a buffer, an offset, and a source string. The new API take a destination buffer, source string, and the size of the destination buffer. It will copy as many bytes as will fit in the destination and correctly set the length of the resulting string. It correctly accounts for "invalid" source strings (i.e. length 0xFF).
- **Customer process:** Change calls to emberAfCopyString to pass the destination buffer, source string, and the size of the destination buffer.

4.2.4 *Move Key Establishment to a plugin*

- **Change description:** The key establishment functionality was moved to a new plugin called Key Establishment. To support this change, new callbacks were added: emberAfInitiateKeyEstablishmentCallback and emberAfPerformingKeyEstablishmentCallback.
- **Customer process:** Change calls to initiateKeyEstablishment and emberAfDeviceIsPerformingKeyEstablishment to

emberAfInitiateKeyEstablishmentCallback and emberAfPerformingKeyEstablishmentCallback respectively. If key exchange functionality is required, enable the new Key Establishment plugin or implement the new callbacks and functionality in the application. The functionality is required by all SE devices.

4.2.5 *Moved Partner Link Key Exchange to a plugin*

- **Change description:** The partner link key exchange functionality was moved to a new plugin called Partner Link Key Exchange. To support this change, new callbacks were added: emberAfInitiatePartnerLinkKeyExchangeCallback, emberAfPartnerLinkKeyExchangeRequestCallback, and emberAfPartnerLinkKeyExchangeResponseCallback.
- **Customer process:** Change calls to emberAfInitiatePartnerLinkKeyRequest to emberAfInitiatePartnerLinkKeyExchangeCallback. If partner link key exchange functionality is required, enable the new Partner Link Key Exchange plugin or implement the new callbacks and functionality in the application. Partner link key exchange functionality is required for SE devices that need to communicate with other devices that are not the trust center. The trust center itself does not need this functionality.

4.2.6 *Moved Partner Link Key Exchange to a plugin*

- **Change description:** The partner link key exchange functionality was moved to a new plugin called Partner Link Key Exchange. To support this change, new callbacks were added: emberAfInitiatePartnerLinkKeyExchangeCallback, emberAfPartnerLinkKeyExchangeRequestCallback, and emberAfPartnerLinkKeyExchangeResponseCallback.
- **Customer process:** Change calls to emberAfInitiatePartnerLinkKeyRequest to emberAfInitiatePartnerLinkKeyExchangeCallback. If partner link key exchange functionality is required, enable the new Partner Link Key Exchange plugin or implement the new callbacks and functionality in the application. Partner link key exchange functionality is required for SE devices that need to communicate with other devices that are not the trust center. The trust center itself does not need this functionality.

4.2.7 *emberAfProcessMessage no longer handles outgoing messages*

- **Change description:** emberAfProcessMessage previously processed incoming and outgoing messages, although it did little to actually handle outgoing messages. Outgoing messages are now handled in the internal emAfMessageSentHandler and passed to per-cluster MessageSent callbacks or the global emberAfMessageSentCallback from there. The outgoing-specific parameters "incoming" and "messageSentStatus" have been removed from emberAfProcessMessage.
- **Customer process:** Change calls to emberAfProcessMessage to remove the "incoming" and "messageSentStatus" parameters. The framework will automatically handle processing outgoing messages and emberAfProcessMessage can no longer be used for outgoing messages.

4.2.8 *emberAfPreCommandReceivedCallback changed*

- **Change description:** The isInterpan boolean parameter to emberAfPreCommandReceivedCallback has been removed. Instead, a pointer to the InterPanHeader data is included in the EmberAfClusterCommand structure. This allows applications to have information about the inter-PAN message.
- **Customer process:** Remove the isInterpan parameter from implementations of emberAfPreCommandReceivedCallback. If interPanHeader is NULL, the command was received via standard ZigBee messaging. Otherwise, the command was received via inter-PAN and interPanHeader contains additional information about message.

4.2.9 *Attribute change callbacks now take manufacturer code. [REQUIRES CHANGE TO CURRENT CUSTOMER CODE]*

- **Change description:** Attribute change callbacks now take a manufacturer code so that manufacturer specific attributes may be handled properly. If the passed attribute is not manufacturer specific, the passed manufacturer code is 0.
  - Callbacks affected:
    - emberAfAllowNetworkWriteAttributeCallback
    - emberAfPreAttributeChangeCallback
    - emberAfPostAttributeChangeCallback
    - emberAfExternalAttributeWriteCallback
    - emberAfExternalAttributeReadCallback
- **Customer process:** IN order to compile with the updated framework, all of the above callbacks must be modified to take the int16u manufacturerCode. If the attribute being passed is not manufacturerSpecific, the manufacturerCode will be 0.

4.2.10 *emberAfLocateAttributeMetadata, emberAfVerifyAttributeWrite now take a manufacturerCode [REQUIRES CHANGE TO CURRENT CUSTOMER CODE]*

- **Change description:** emberAfLocateAttributeMetadata and emberAfVerifyAttributeWrite functions now take a manufactureCode as an argument. If the attribute being saught or verified is not manufacturer specific the passed manufacturerCode should be 0.
- **Customer process:** Update calls to emberAfLocateAttributeMetadata and emberAfVerifyAttributeWrite to pass the manufacturerCode for manufacturer specific attributes, use 0 as a manufacturer code value if the attribute is not manufacturer specific.

4.2.11 *emberAfReadOrUpdateAttribute changed to take pointer to EmberAfAttributeSearchRecord [REQUIRES CHANGE TO CURRENT CUSTOMER CODE]*

- **Change description:** The generic read or update attribute function has been modified to take a pointer to an instance of the new struct EmberAfAttributeSearchRecord containing all of the data required to find the unique attribute identified in the attribute storage. The passed pointer to an instance of EmberAfAttributeSearchRecord is expected to be fully populated with all necessary fields to find the attribute in the attribute table, this includes: endpoint, clusterId, attributeId, clusterMask, manufacturerCode.

- **Customer process:** Modify any calls to emberAfReadOrUpdateAttribute to first populate an instance of the EmberAfAttributeSearchRecord and then pass a pointer to the declared and populated struct.

4.2.12 *Wrapper functions for emberAfReadOrUpdateAttribute modified to take manufacturer code. [REQUIRES CHANGE TO CURRENT CUSTOMER CODE]*

- **Change description:** Three wrapper functions declared in attribute-storage.h have been updated to take a manufacturer code. These include:
  - emberAfContainsAttribute
  - emberAfRetrieveAttribute
  - emberAfUpdateAttribute These functions are not currently exposed in af.h however they are used within the framework. These three functions are simply wrappers for the more complex emberAfReadOrUpdateAttribute. Because emberAfReadOrUpdateAttribute has changed to take an instance of EmberAfAttributeSearchRecord, they have been modified to take a manufacturer code.
- **Customer process:** If you used these functions in your code you should update the calls to them to pass a manufacturer code or 0 if the attribute being saught is not manufacturer specific.

4.2.13 *Custom attributes in xml files no longer need to include manufacturerSpecific="true" for manufacturer specific attributes.*

- **Change description:** Manufacturer specific Clusters, Commands and Attributes defined in an xml file loaded into the Application Configurator need to include a two byte manufacturer code in order to be considered manufacturer specific.
- **Customer process:** All manufacturer specific Clusters, Attributes and Commands used within a standard ZigBee profile **SHOULD** include a manufacturer code. If they do not include a manufacturer code they are assumed to be non-manufacturer specific. If they include a manufacturer code, the Attribute and Command ids may now overlap existing ZCL attributes and commands as well as other manufacturer specific attributes and commands. Manufacturer specific cluster ids must be inside the manufacturer specific id range 0xf000 - 0xffff.
- **EXAMPLE I:** For an example of a manufacturer specific attribute see MIRROR_IEEE_ADDRESS declared in ami.xml. This is an attribute used by the meter mirror to track the ieee address of the device it is mirroring.
- **EXAMPLE II:**

```
<clusterExtension code="0x0000">
   <attribute side="server" code="0x0000" define="EMBER_VERSION"
   type="INT16U" min="0x4000" max="0xFFFF" writable="true"
   default="0x4500" optional="true" manufacturerCode="0x79ab">Ember
   Version</attribute>
     <command source="client" code="0x00" name="ResetToEmberDefaults"
     optional="true" noDefaultImplementation="true"
     manufacturerCode="0x79ab">
```

```
            <description>Command that resets all attribute values to their
            Ember default values.</description>s
          </command>
     </clusterExtension>
```

4.3 APIs Added

4.3.1 Added struct EmberAfAttributeSearchRecord

- **Change description:** Added struct EmberAfAttributeSearchRecord to contain all data required to find an attribute in the attribute table.
- **Customer process:** Some attribute related APIs now take a pointer to an instance of this struct as an argument instead of many individual arguments.

4.3.2 Added struct EmberAfManufacturerCodeEntry to contain manufacturer code along with an index into the attribute or cluster tables

- **Change description:** Added a new struct EmberAfManufacturerCodeEntry as a manufacterer code table entry. The manufacter code table is generated by the appbuilder if the application contains manufacturer specific attributes or clusters.
- **Customer process:** No customer action required.

4.3.3 Added 8 new functions created for reading and writing attributes in the application framework. We will be moving the framework over to use these new functions internally in order to reduce code size.

- **Change description:** We have added 8 new functions for interacting with attributes. These include:
    - emberAfReadServerAttribute
    - emberAfWriteServerAttribute
    - emberAfReadClientAttribute
    - emberAfWriteClientAttribute
    - emberAfReadManufacturingSpecificServerAttribute
    - emberAfWriteManufacturerSpecificServerAttribute
    - emberAfReadManufacturingSpecificClientAttribute
    - emberAfWriteManufacturerSpecificClientAttribute. We found two things within the framework that were needlessly contributing to code size bloat. We were passing EMBER_CLUSTER_SERVER AND EMBER_CLUSTER_CLIENT masks to all of the read functions also in most instances where we were using the attribute read functions we were also passing NULL as the pointer to a data type. These new functions make the api more specific and should reduce code size (as we move the framework over to use them). The new read functions DO NOT take a pointer to data type for retrieval of the data type. In order to simply retrieve the data type use emberAfReadAttribute. **We will be moving the framework over to use the functions internally to reduce code size in our applications**.
- **Customer process:** In order to interact with manufacturer specific attributes you **MUST** use the manufacturer specific attribute read and write functions. All other attribute reads and writes **should** use the client and server specific functions in place of passing a cluster mask

into the existing functions emberAfReadAttribute and emberAfWriteAttribute for reasons of reducing code size.

5 Known Issues

5.1 Resolved Bugs

- **Case 12880** AFV2's emberJoinableNetworkFoundHandler doesn't work with Random (all-zero) extended PAN ID selection
- **Case 13554** KE incorrectly sends terminate key to wrong destination endpoint
- **Case 13546** DRLC Client incorrectly sends report-event-status for opt-out events
- **Case 13547** DRLC server incorrectly sends Default Response to Get Sceduled events
- **Case 13543** Key Establishment Cluster sent incorrect status code for too short certificate
- **Case 13496** Key Establishment sends terminate message to incorrect destination under certain conditions
- **Case 13548** Sending a status of Reject for duplicate DRLC load control events is not appropriate
- **Case 12243** cutom token file does not allow relative path
- **Case 12244** sometimes callback files are not being copied and renamed correctly
- **Case 12698** OTA cluster needs to handle 35x bootloaders with old dataflash drivers
- **Case 12689** [portalwatch] EZSP: Park Beta NCP image has issue with start scan
- **Case 12673** "zcl ota server notify" should take source and destination endpoints
- **Case 12694** AppBuilder Should Allow Default Values for Optional Fields in Commands
- **Case 12714** avr128-spi application image cannot be compiled - globalFragmentSourceRouteInfoValid undefined
- **Case 12637** CLI: Raw command is hard coded to 64 bytes, should allow larger

- **Case 12690** Disable Default response should be set for all OTA Server messages
- **Case 12713** DRLC Client Plugin Doesn't Always Respond to Cancel Commands
- **Case 12678** emberAfStartSearchForJoinableNetwork needs to use emAfExtendedPanId for network search
- **Case 12656** ISD - appbuilder upload application memory is out of sync
- **Case 12679** Joining state machine in network-find.c is prone to failure
- **Case 12640** OTA bootloading simple storage partial storage file support
- **Case 12674** OTA client always uses the same ZCL sequence number of the last message
- **Case 12703** Seg fault -357-ezsp uart with SW flow control
- **Case 12691** Source route should not be set when it cannot be found
- **Case 12658** Source routing needs to be enabled for concentrators (including Trust Centers)
- **Case 12489** Update dummy signature to match SE 1.1 changes
- **Case 12401** [portalwatch] DRLC cluster should send NOT_FOUND Default Response for LCEs with non-matching criteria
- **Case 11322** [SE 1.1] OTA client sent two upgrade requests at the end of downloading a file
- Case **12709** "print drlc" should go to DRLC print area
- **Case 12700** "zcl sprice publish" is broken
- **Case 12696** alternateCost Fields of Publish Price Command Need an introducedIn Tag
- **Case 12545** Build failure for UART host: smart-energy-registration.o
- Case **12715** Discover Attributes Never Reports "Complete"

- Case **12704** DRLC Client Table Size Should be at Least 3
- **Case 12697** DRLC: No report event status sent on receipt of cancel for non-existent event
- **Case 12651** em250 target - qa-mobile-node unknown file type
- **Case 12260** ha switch and combined interface sample apps keep looking for light after rejoin
- **Case 12650** [PortalWatch] Help command response only disply 3 commands with EmberZNet 4.2.0
- **Case 11275** SLEEPY: Modification of zclTick() for end devices.
- **Case 12434** Sync Framework with Latest HA Spec (053520r27)

5.2 Open Bugs / Features

- **Case 11958** [portalwatch] [AFV2] DRLC Cancel command can cause new events to be scheduled incorrectly
- **Case 12626** Commissioning cluster: Address and PAN attributes need to be updated when stack status changes
- **Case 12627** Commissioning cluster: Default value of StartupControl attribute should be 3 (not 2)
- **Case 12864** Framework Should Use ezspUtilInit to Configure NCP
- **Case 12875** OTA client state machine should fall back to discovery state at some point
- **Case 12994** zaTrustCenterTick is never called in AFV2; NWK key switch never happens
- **Case 13064** OTA - add plugin option to deal with deletion of downloaded images
- **Case 13182** Need declarations for RX statistics variables when EMBER_AF_ENABLE_STATISTICS defined
- **Case 13220** High-RAM concentrators should send source-routed ACKs with setSourceRoute, sendReply
- **Case 12554** [Commissioning cluster] security bitmask not set in emberAfCommissioningClusterRestartDeviceCallback()
- **Case 12639** Reporting Should Not Poll for Attribute Changes

- **Case 12667** AppBuilder Should Make Coordinators Concentrators by Default

Intended Behavior

- Please see http://portal.ember.com/not-a-bug for an explanation of behaviors that are intended but have caused confusion.