What's New in AES70

Changes in AES70 for 2018

Background

- AES70 is the AES standard for device control.
- Publication history
  - Recommended specification developed by the OCA Alliance 2011-2014
  - Specification passed to AES for standardization; AES70-2015 published in January of 2015
- AES70-2015 is substantially complete, but requires a few improvements.
- Improvements were drafted by the OCA Alliance in 2017 and 2018, and passed to the AES as recommended revisions to AES70-2015.
- The revisions are now approved internally by AES and are currently in a six-week public comment period that ends on November 9.
- After all comments (if any) have been integrated, AES70-2018 will become an official standard.

Process

- AES70-2018 is not a major upgrade to AES70-2015, but it does contain several important changes.
- Most of the changes have been developed in response to the experiences of early adopters of AES70-2015.
- For AES70-2018, the version numbers of all AES70-2018 control classes have been incremented. This will allow controllers to distinguish between AES70-2015 and AES70-2018 implementations.

Quick List

1. Connection Management, version 3 (CM3)
2. Improved support for clocking and time
3. WebSocket and UDP Protocol support
4. Improved object search features
5. Upgraded ClassID format to support proprietary classes more completely
6. Reusable blocks
7. Better support for proprietary volume types in OcaLibrary
8. Support for nonexclusive locking of objects
9. The OcaPhysicalPosition class
10. New Task mechanism

1 Connection Management v3 ("CM3")

- "Connection management" means setting up, configuring, controlling, monitoring, and taking down media stream connections between devices.
  - The new connection management scheme of AES70-2018 is called "Connection Management 3" or CM3.
  - The scheme in AES70-2015 is Connection Management 2 (CM2).
  - CM1 was a developmental scheme that was never standardized.
- In comparison to CM2, CM3 has lower storage requirements, better management of connection parameters, much better management of clocking, better and more flexible codec support, and is generally easier to implement in both devices and controllers.
- AES70-2018 still includes the specification of CM2, although CM2 classes are now deprecated. AES70-2018-compliant devices can support coexisting CM2 and CM3 implementations with no conflicts.

2 Improved support for clocking and time

- The new classes **OcaTimeSource** and **OcaMediaClock3** provide improved ways of describing and controlling external time references and media clocks.

3 WebSocket and UDP protocol support

- AES70-2018 will run over WebSocket and UDP links, as well as the TCP links used heretofore. Features and restrictions of these new link types are as follows:
  - Encryption and authentication are supported only over TCP links, not WebSocket or UDP links.
  - WebSocket and UDP versions use the same protocol data unit formats as the TCP version.
  - WebSocket implementations need implement only a few webserver functions, not entire webservers.
  - UDP links are for small, noncritical applications on single IP subnets. The UDP version uses fewer device resources, but is inherently less reliable.
  - For UDP functions, the rules for supervision (“keepalive”) functions are a little different from the other versions.

4 Improved object search features

- AES70-2018 includes features to help controllers locate objects by their **Role** properties.
- **Role** is a property of every AES70 object. It is designed to be a text description of the object's function in the device.
- **Role** is analogous to the text printed adjacent to a control knob, button, or indicator on the front-panel of a conventional manually-controlled device.
- **Roles** are fixed at object creation time, which means for most devices they will be set a time of manufacture.
- Using **Roles** to find objects means that controllers will not need to have prior knowledge of a device's set of object numbers, but instead can discover the object numbers at runtime by searching for the objects desired. This improves interoperability and makes object number management simpler.

4 Improved object search features, continued

- The new **Role** features in AES70-2018 are:
  - Method **GetPath(...)** in class **OcaWorker** and class **OcaAgent**.
- These methods return the **Rolepath** of an object. The **Rolepath** is the ordered list of the **Roles** of all an object's nested containing **OcaBlocks** followed by the **Role** of the object itself.
- The following methods in class **OcaBlock**:

  FindObjectsByRole(...)

  FindObjectsByRoleRecursive(...)

  FindObjectsByPath(...)

  FindObjectsByLabelRecursive(...)

- These methods find objects within blocks based on various search criteria.

5 Upgraded ClassID format

- Format of the ClassID has been revised (compatibly) to do a better job of supporting proprietary classes (i.e., classes manufacturers add to the standard class tree for special purposes).
- The previous scheme, in AES70-2015, allowed proprietary ClassID values, but did not prevent clashes if proprietary classes from two different sources were combined in one device.
- The new scheme includes a unique company ID (an IEEE OUI or CID) in proprietary ClassID values, and therefore allows free mixing of such classes with no conflicts.
- The new ClassID scheme is upwards-compatible with the AES70-2015 scheme, except that the class index value 65,535 = FFFF16 is now reserved.

6 Reusable OcaBlocks

- A manufacturer or other design source may now assign unique identifiers (global block identifier) to specific OcaBlock configurations. An OcaBlock so identified is termed a ***reusable block***. Reusable blocks may be instantiated in multiple products, where controllers will recognize them by their global block identifiers, and invoke corresponding common controller codes.

- It is anticipated that companies using AES70 will develop libraries of reusable block specifications for deployment across their product lines. Global block identifiers include unique company identifiers (IEEE OUI or CID), which ensures that companies can allocate identifier values without fear of clashing with others.

7 Better support for proprietary volume types in OcaLibrary

- The agent object **OcaLibrary** is designed to store large binary objects("volumes") of various types, including both standard types and proprietary types.
- Each volume type is identified by a unique code named **OcaLibVolType**.
- Under AES70-2015, it was possible for proprietary **OcaLibVolType** values to clash with proprietary libraries from multiple companies in the same device.
- In AES70-2018, the structure of **OcaLibVolType** now includes an IEEE OUI or CID value that uniquely identifies the source. Clashes are now impossible.
- For standard (i.e. non-proprietary) **OcaLibVolType** values, AES70-2015 and AES70-2018 are the same.
- If proprietary classes are used, the AES70-2018 scheme is incompatible with the AES70-2015 scheme.

8 Support for nonexclusive locking of objects

- AES70-2018 now supports an object locking option that allows read-only access to locked objects.
- Previously, AES70-2015 supported only exclusive locking, in which properties of locked objects could be neither retrieved nor changed by controllers other than the lock holder.
- AES70-2018 now supports two locking options: LockTotal, which is identical to the exclusive lock defined in AES70-2015, and LockReadonly, which prevents change of locked objects, but allows retrieval of their properties' values.
- This change is implemented compatibly. When an AES70-2015-compliant controller locks an AES70-2018 device via the `Lock()` method that is defined in AES70-2015, the resulting lock state is LockTotal, i.e. exclusive lock.

9 The OcaPhysicalPosition class

- The new OcaPhysicalPosition class defines an agent that can report and, depending on implementation, change a physical position.
- This class has several expected usecases, including:
    1. Reporting physical position of active loudspeakers and microphones that can sense their locations and orientations;
    2. Allowing manipulation of object-based audio program entities;
    3. Controlling position and/or orientation of automated mechanical devices;
    4. Supporting geographically-aware devices.
- To support these and other purposes, OcaPhysicalPosition has options for working in three kinds of coordinate systems: (a) six-axis robotic coordinates; (b) object-based audio coordinates of several types; and (c) world geographic coordinates such as are used in GPS.

10 The AES70 **Task** mechanism

- AES70-2018 defines a new architectural concept called the AES70 Task mechanism. This mechanism allows management and control of transient processes within a device. Examples of such processes include:
  - Execution of predefined, prepackaged real-time actions. Such sequences are called presets, cues, or edits in other contexts.
  - Execution of actions that have been scheduled sometime in the past.
  - Execution of timed control operations such as fades, crossfades, timed pans, and other segues.
  - Execution of predefined system configuration changes.
  - Execution of emergency procedures.
  - Media playback.
  - Execution of mechanical operations.

10 The AES70 **Task** mechanism

- The Task mechanism is based on two concepts:
  1. The ***Program***, a predefined entity that defines action(s) to be performed;
  2. The ***Task***, a container that executes the program.
- In AES70-2018, Programs are stored as OcaLibrary volumes whose OcaLibVolType is Program, and Tasks are defined in data structures collected by a new Manager class, OcaTaskManager.
- OcaTaskManager provides methods for defining Tasks, for assigning Programs to them, and for starting, stopping, and monitoring them.
- AES70-2018 does not define a method for encoding the specific actions of Programs - conditions tested, steps executed, notifications generated, et cetera.