Multi-option device support

Pat Telljohann
Software Engineer
Rockwell Automation

Presented at the ODVA
2014 Industry Conference & 16th Annual Meeting
March 11-13, 2014
Phoenix, Arizona, USA

Abstract:

Some complex devices have a number of options with each option having a number of choices. The number of option combinations for these devices can be large. These choices are typically made by the user at purchase time. Representing each combination with a unique Product Code is often not practical because the number of combinations can be in the 1000s. Multi-option support defines a mechanism that allows devices to have a unique identity for each combination of options, allows for keying along with a mechanism to define each option separately and how the options can be combined without the need for 1 EDS file for each combination of options.

Keywords:

Multi-Option Device, EDS, Device Identity, Electronic Key

Definition of terms:

Device Class information
A description of a device's identity and capabilities. This information includes how the device is configured, I/O connections supported by the device, the set of objects supported by the device, etc. For CIP devices, the Electronic Data Sheet (EDS) holds this information.

EDS
Electronic Data Sheet.

Electronic Key
A logical segment used to verify a device has the expected logical view.

Multi-Option Device
A device identified with a vendor, device type and product code that can support different logical views depending on which device options that are selected/present for the device.

Introduction
A multi-option device is a device that presents more than one logical view on the network depending on the option value for each option. These options are typically items the user can choose either when the device is ordered or by adding pluggable components to the device.

A car is an analogy for multi-option device. The customer can order a model of a car and then select options like body style (coupe, convertible, hatchback, station wagon, etc.), engine type (4 cylinder, 6 cylinder, electric, etc.), wheel style (steel, aluminum, etc.), transmission (automatic, manual, etc.).

Currently multi-option devices must be supported using a unique vendor, device type and product code for each combination of option values and also a unique EDS.

Multi-option device support goals
The goals for supporting multi-option devices are as follows:

1. Define a mechanism for a device to identify which option types it supports.
2. Define a mechanism for an originator to verify that a device has the option types expected.
3. Define a compact mechanism to describe the device class information.
4. Define a mechanism that allows options to be used in multiple device types developed by a vendor.

Identity Object
The Identity Object will be enhanced to support new conditional attributes (required for multi-option devices, not allowed otherwise) that defines the number of options, the option type and values for each option type.

To clue a browsing tool in on the fact the device may be a multi-option device, bit 1 of the Status Attribute will be defined to indicate additional identity information is available. (Browsing tools typically perform a Get_Attributes_All service request to Identity Object Instance 1 and the Status Attribute is included in the response).

A new conditional service will be defined that will return additional device identity attributes. The response data will include meta-data that will allow this service to be extensible for other purposes (like returning option information).

Electronic Key Segment
The Electronic Key logical segment is the mechanism used to verify a client is communicating with the expected device. Electronic Key format 4 is used to verify the Vendor, Device Type, Product Code and Revision of a device. A new Electronic Key format will be defined to support verification to the option types and values.

Electronic Data Sheet
The Electronic Data Sheet (EDS) is the mechanism used to define device class information (all the details about the logical view the device presents). The EDS definition will be enhanced to provide a mechanism to describe option class information in an Option Data Sheet (ODS), one Option Data Sheet per option type and value. Going back to the car analogy, an EDS file would exist for the car and it would include a list of the option types and option type values available for the car. An ODS file would exist for each option type/value. Depending on the set of option values the EDS and ODS files will be combined to provide the device class information for the specific device. The ODS file is similar to a header file in computer programming.

In addition, an Option value might be able to support sub-options (for instance if the option value is convertible, then there might be options for electronic or manual operation). These sub-options would be listed in the specific ODS file they apply to. Option nesting can be performed for multiple levels.

An example of a multi-option device that has 3 options (Frame Size, Overload Type and Control Power) with 1 Alloy Overload Type sub-option (Heating Element) might look something like this:

<table>
<tr>
<td>Multi-option Motor Overload device</td>
<td>Option Type 1
(Frame Size)
1. Option Choice 1
(Large)
2. Option Choice 2
(Small)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Option Type 5
(Overload Type)
1. Option Choice 1
(Alloy)

2. Option Choice 2
(Bi-metal)</td>
<td>Option Type 987
(Heating Element)
1. Option Choice 3
(FLC 1-10 amps)
2. Option Choice 10
(FLC 1-10 amps)</td>
</tr>
<tr>
<td></td>
<td>Option Type 25
(Control Power)
1. Option Choice 1
(24 VDC)
2. Option Choice 2
(110 VAC)</td>
<td></td>
</tr>
</table>

The EDS snippet for this device might look like:

```
[Device]
    VendCode = 65535;
    VendName = "Widget-Works, Inc.";
    ProdType = 3;
    ProdTypeStr = "Motor Overload";
    ProdCode = 1;
    MajRev = 1;
    MinRev = 1;
    ProdName = "Deluxe Overload";
    Catalog = "3255-OptDev%1%2%3";
    Option1 = 65535, 1, “Frame Size”,
            1, “Large”,
            2, “Small”;
    Option2 = 65535, 5, “Overload Type”,
            1, “Alloy”,
            2, “Bi-metal”;
    Option3 = 65535, 25, “Control Power”,
            1, “24 VDC”,
            2, “110 VAC”;
```


The ODS snippet for the Option Type “Overload Type”, Option Value “Alloy” might look like (other ODS files would be similar and are not included):

```
[Option]
    VendCode = 65535;
    VendName = "Widget-Works, Inc.";
    OptionType = 5;
    OptionTypeName = “Overload Type”;
    OptionChoice = 1;
    OptionChoiceName = “Alloy”;
    Catalog = "1%1";
    Option1 = 65535, 987, “Heating Element”
        3, “FLC 1-10 amps”,
        10, “FLC 11-20 amps”;

[Connection Manager]
    Connection1 =
        0x04020002,
        0x66240405,
        ,0,,
        ,0,,
        ,,
```

Use cases

To show how multi-option devices might be supported in a system online and offline use cases are examined below.

Online

When online a configuration tool might browse a network to discover the set of CIP devices that are present. This browsing is performed by sending a Get_Attributes_All request to Identity Object Instance 1.

If the Status attribute value returned has bit 1 set, an Extended_Get_Attributes_All request to Identity Instance 1 would be performed.

At this point the configuration tool knows the specific option types and values for the device. The configuration tool would then allow the user to configure the device based on the base device capabilities and options. This would include configuration of I/O connections between a master device and the multi-option device, application configuration of the target device, calibration, etc.

If the configuration tool performs any direct communications with the multi-option device (for example for calibration), it might do so by establishing a class 3 connection. This class 3 connection might include electronic keys to verify Vendor, Device Type, Product Code and Revision in addition to the options.

When the class 0 or 1 I/O connection between the master device and the multi-option device is established, the connection might include electronic keys to verify Vendor, Device Type, Product Code and Revision in addition to the options.

Offline

When offline a configuration tool might allow the user to select the type of device they want to configure. If this device is a multi-option device, the user would be shown the top level option types and be able to select an option value for each option type (if nested options exist, the user would be shown this option information under a specific option choice value).

At this point the configuration tool knows the specific option types and values for the device. The configuration tool would then allow the user to configure the device based on the base device capabilities and options. This would include configuration of I/O connections between a master device and the target device, application configuration of the target device, calibration, etc. When the configuration is complete it would be downloaded to the originator device.

When the class 0 or 1 I/O connection between the originator device and the multi-option device is established, the connection might include an electronic key to verify Vendor, Device Type, Product Code and Revision in addition to a new electronic key to verify to the options (see Electronic Key Segment section above).

Updates required for multi-option support

To support multi-option changes are required in the following areas.

Multi-option device

The multi-option device will need to support the new Identity object definition, including the new multi-option attribute and support the new service to read additional identity information.

The multi-option device will need to support the new electronic key segment to verify its options.

Originator device

An originator device will need to be able to include the new electronic key segment in the services where keying is typically used (like Forward_Open).

Configuration tool

The configuration tool will need to be updated to browse for additional Identity object information if additional identity information is indicated.

The configuration tool will need to be updated to provide the user with an interface for choosing options (when offline).

The configuration tool will need to be updated to support combining the multi-option EDS file with selected option ODS files.

Summary

There is a specification enhancement for multi-option device support definition (CIPSE-001-189) working through the CIP System Architecture SIG approval process. There is also a patent pending for this multi-option device support concept.