Multi-Option Device Support

Pat Telljohann
Rockwell Automation

Technical Track
Agenda

- Introduction
- Objectives
- Multi-Option Device Support Definition
  - Identity Object
  - Keying
  - Device Class Information
- Use cases
  - Online
  - Offline
Agenda

- Introduction
- Objectives
- Multi-Option Device Support Definition
  - Identity Object
  - Keying
  - Device Class Information
- Use cases
  - Online
  - Offline
Multi-option Device definition

- A device that presents more than one logical view 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
  - 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.)
Multi-option Device definition

- The Multi-option Device definition is not meant to be used to represent a modular system where each module can be addressed using a port segment.

- Some complex devices will be both modular and Multi-option.
Devices with multiple options

- Currently devices with multiple options require a unique identity (product code) for each unique combination of options
- This results in many identities
  - A device with 6 options, each with 3 choices would result in $3^6$ identities (729)
- And many EDS files (729 in example above)
Agenda

- Introduction
- Objectives
  - Multi-Option Device Support Definition
    - Identity Object
    - Keying
    - Device Class Information
  - Use cases
    - Online
    - Offline
Objectives

- Uniquely identify each logical view presented by option values
- Provide the ability to verify the option values supported by a specific device
- Limit the number of files required to represent “device class” information (EDS)
- CIPSE-001-189 defines Multi-option support, currently being reviewed by CIP System Architecture SIG
Agenda

- Introduction
- Objectives
- Multi-Option Device Support Definition
  - Identity Object
  - Keying
  - Device Class Information
- Use cases
  - Online
  - Offline
Identity Object and Keying

- The multi-option definition allows a single identity (Vendor ID, Device Type, Product Code) for the base device
  - Options are extensions to the base device
- New attribute defined to identify options that are present
- New information defined in Get_Attributes>All response data (bit 1 in Status attribute) to indicate device supports additional identity information
  - New Get_Additional_Attributes service defined that returns additional identity attributes, including the options attribute
- New electronic key segment type defined to key option values
Device Class information

- One EDS file is required for each Vendor ID, Device Type, Product Code and Major Revision
- One ODS (Option Data Sheet) is required for each option value. The ODS file is identified by Vendor ID, Option Type and Option Choice
- ODS file definition is very similar to EDS file definition
- A Multi-option device family with 6 options types, each with 5 options choices would require 1 EDS file and 30 ODS files - without the Multi-option device definition, 15,625 EDS files would be required
Device Class information

- The EDS defines the device class information for the base device plus the supported option types
- The ODS defines the device class information for an option value for an option type
- Typical information in an ODS
  - Catalog number
  - Attributes
  - Connections
  - Ports
  - Sub-options
- EDS file and ODS files combined and result in the full set of device class information for the device
Options

- EDS defines the supported options, which defines which ODS files to use
- One ODS per option value
- Example EDS and ODS files:

**EDS file snippet**

```markdown
[Device]
VendCode = 65535;
VendName = "Widget-Works, Inc.";
ProdType = 768;
ProdTypeStr = "Option type device";
ProdCode = 1;
MajRev = 1;
MinRev = 1;
ProdName = "Option Device";
Option1 = 1, "Frame Size",
  1, "Large",
  2, "Small";
Option2 = 5, "Overload Type",
  1, "Alloy",
  2, "Bi-metal";
Option3 = 25, "Control Power",
  1, "24 VDC",
  2, "110 VAC";
```

**ODS file snippet**

```markdown
[Option]
VendCode = 65535;
VendName = "Widget-Works, Inc.";
OptionType = 1;
OptionTypeName = "Frame Size";
OptionChoice = 1;
OptionChoiceName = "Large";
```

**ODS file snippet**

```markdown
[Option]
VendCode = 65535;
VendName = "Widget-Works, Inc.";
OptionType = 1;
OptionTypeName = "Frame Size";
OptionChoice = 1;
OptionChoiceName = "Large";
```

**ODS file snippet**

```markdown
[Option]
VendCode = 65535;
VendName = "Widget-Works, Inc.";
OptionType = 1;
OptionTypeName = "Frame Size";
OptionChoice = 2;
OptionChoiceName = "Small";
```
Catalog Number

- EDS defines the base catalog number
- Each ODS optionally defines an addition to the base catalog number
- The combination is a catalog number for the specific device
- Example EDS and ODS files and result:

**EDS file snippet**

```
[Device]
    VendCode = 65535;
    VendName = "Widget-Works, Inc.";
    ProdType = 768;
    ProdTypeStr = "Option type device";
    ProdCode = 1;
    MajRev = 1;
    MinRev = 1;
    ProdName = "Option Device";
    Catalog = "3255-OptDev%1";  
```

**ODS file snippet**

```
[Option]
    VendCode = 65535;
    VendName = "Widget-Works, Inc.";
    OptionType = 1;
    OptionTypeName = "Frame Size";
    OptionChoice = 1;
    OptionChoiceName = "Large";
    Catalog = "Ex";  
```

**Resulting device class information**

```
[Device]
    VendCode = 65535;
    VendName = "Widget-Works, Inc.";
    ProdType = 768;
    ProdTypeStr = "Option type device";
    ProdCode = 1;
    MajRev = 1;
    MinRev = 1;
    ProdName = "Option Device";
    Catalog = "3255-OptDevEx";  
```
Attributes

- EDS defines the attributes that exist in the base device via ParamN and AssemN keywords
- Each ODS defines attributes that exist for the option via ParamN and AssemN keywords
- Examples:

**EDS snippet**

```
[Params]
Param1 =
0,
6,"20 1D 24 01 30 06",
0x0002,
0xC7,
2,
"FilterOffOn",
"ms",
"Input OFF-to-ON Filter.
","
0,16000,1000,,,,,,,,;
```

**ODS snippet**

```
[Params]
Param2 =
0,
6,"20 1D 24 01 30 08",
0x0002,
0xC7,
2,
"Connection Config",
"",
"Some Config"
",
0,,10,,,,,,,,;
```

**Resulting device class information**

```
[Params]
Param1 =
0,
6,"20 1D 24 01 30 06",
0x0002,
0xC7,
2,
"FilterOffOn",
"ms",
"Input OFF-to-ON Filter.\n"
",
0,16000,1000,,,,,,,,;
```

```
Param2 =
0,
6,"20 1D 24 01 30 08",
0x0002,
0xC7,
2,
"Connection Config",
"",
"Some Config"
",
0,,10,,,,,,,,;
```
Connections

- EDS defines the connections that exist in the base device
- Each ODS defines connections that exist for the option
- If an option adds to previous I/O assemblies the ODS defines an AssemN entry that matches a previous AssemN
Connection added by option

**EDS snippet**
```
[Connection Manager]
Connection1 =
0x04020002,
0x66240405,
,,0,,
,,0,,
,,
,, "Simple short cut path", 
"", 
"20 04 24 66 2C 23 2C 69";
```

**ODS snippet**
```
[Connection Manager]
Connection2 =
0x04020002,
0x66240405,
,,0,,
,,0,,
,,
,, "Single short cut path", 
"", 
"20 04 24 66";
```

**Resulting device class information**
```
[Connection Manager]
Connection1 =
0x04020002,
0x66240405,
,,0,,
,,0,,
,,
,, "Simple short cut path", 
"", 
"20 04 24 66 2C 23 2C 69";
Connection2 =
0x04020002,
0x66240405,
,,0,,
,,0,,
,,
,, "Single short cut path", 
"", 
"20 04 24 66";
```
Connection extended by option

**EDS snippet**

```
[Connection Manager]
  Connection1 =
    0x04020002,
    0x66240405,
    "Assem1",
    0,
    "Assembly example",
    "20 04 24 66 2C 23 2C 69";

[Assembly]
  Assem1 =
    "Input",
    2,
    "Param1",
    "Param2",
    "Param3";
```

**ODS snippet**

```
[Connection Manager]
  Connection1 =
    0x04020002,
    0x66240405,
    "Assem1",
    0,
    "Assembly example",
    "20 04 24 66 2C 23 2C 69";

[Assembly]
  Assem1 =
    "More input",
    4,
    "Param2",
    "Param3";
```

**Resulting device class information**

```
[Connection Manager]
  Connection1 =
    0x04020002,
    0x66240405,
    "Assem1",
    0,
    "Assembly example",
    "20 04 24 66 2C 23 2C 69";

[Assembly]
  Assem1 =
    "Input",
    6,
    "Param1",
    "Param2",
    "Param3";
```
EDS defines the ports that exist in the base device

If the option adds ports then the ODS defines PortN entries

Examples:

**EDS snippet**

```
[Port]
  Port1 = TCP,
  "Port A",
  "20 F5 24 01",2;
```

**ODS snippet**

```
[Port]
  Port2 = ControlNet,
  "Port B",
  "20 F0 24 01",3;
```

**Resulting device class information**

```
[Port]
  Port1 = TCP,
  "Port A",
  "20 F5 24 01",2;
  Port2 = ControlNet,
  "Port B",
  "20 F0 24 01",3;
```
Sub-options

- An ODS can define supported sub-options for this specific option value (similar to EDS defining the supported options for the base device)
- The ODS defines the device class information for an option value for a sub-option
- Example option ODS file and sub-option ODS files:

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

  **Sub-option ODS file snippet**
  ```
  [Option]
  VendCode = 65535;
  VendName = "Widget-Works, Inc.";
  OptionType = 987;
  OptionTypeName = "Heating Element";
  OptionChoice = 10;
  OptionChoiceName = "FLC 11-20 amps";
  ```

  **Sub-option ODS file snippet**
  ```
  [Option]
  VendCode = 65535;
  VendName = "Widget-Works, Inc.";
  OptionType = 987;
  OptionTypeName = "Heating Element";
  OptionChoice = 3;
  OptionChoiceName = "FLC 1-10 amps";
  ```
Agenda

- Introduction
- Objectives
- Multi-Option Device Support Definition
  - Identity Object
  - Keying
  - Device Class Information
- Use cases
  - Online
  - Offline
Offline usage

- User adds multi-option device to configuration (same as adding other devices)
- This results in a default set of options choices
- User changes option choices
- The EDS and ODS files are “combined” to produce device class information for device
- User configures multi-option device (same as configuring other devices)
Online usage

- The bus is browsed - CIP Identity object
  Get_Attributes_All - for each possible link address
- The Get_Attributes_All response indicates more identity is available for the device
- Get_Additional_Attributes is used to read the option types and values
- When device is added to configuration, the options are pre-selected
- When communicating with the configured device the new key segment to verify options is used
Questions?