LONMARK®
Interoperability Association

Understanding
LonMark
Self-Documentation

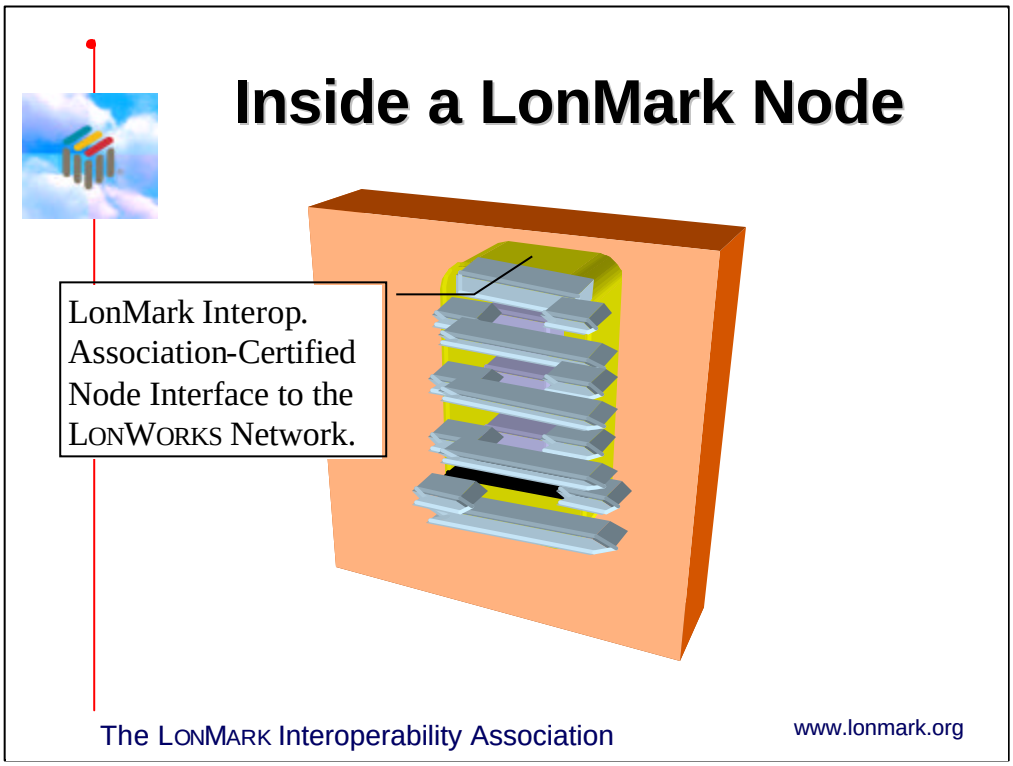
2002-05

The LONMARK Interoperability Association

www.lonmark.org

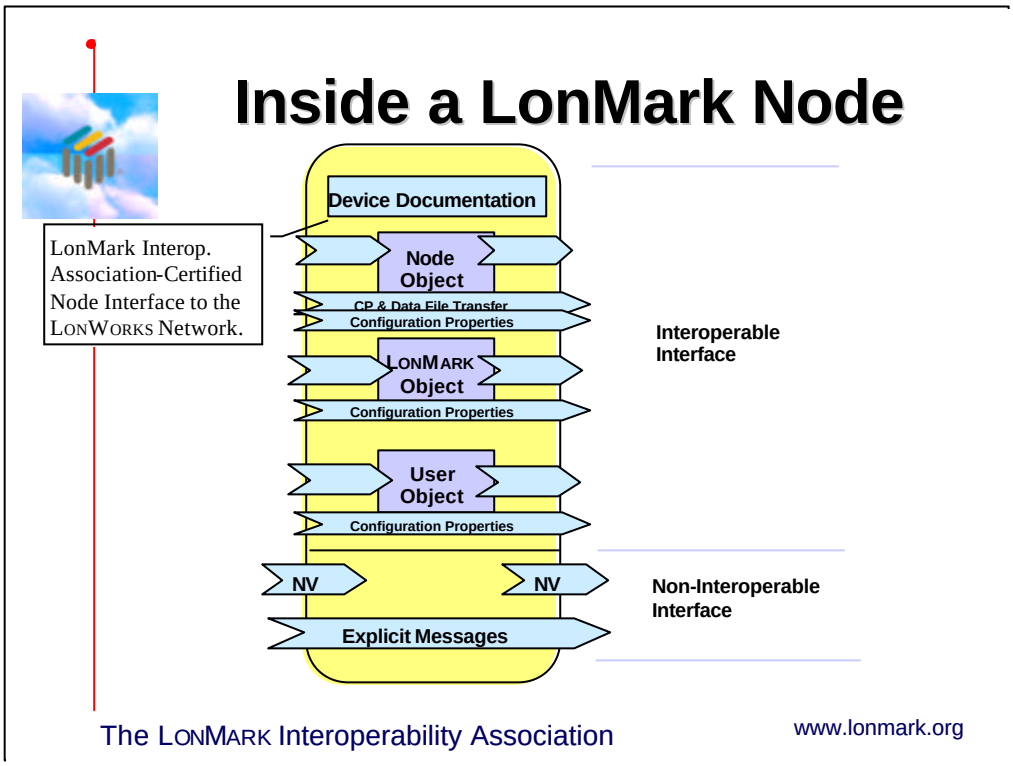
LONMARK® Interoperability Association:

Understanding LonMark Self-Documentation



Inside a LonMark Association-Certified device there exists an “external interface,” by which the node communicates with other nodes on a LonWorks network. The eXternal InterFace (XIF) can be extracted from a node, or a file can be created at the time the node is created. In the latter case, an XIF file is made. This allows installers to design a network without having to physically deal with the nodes.

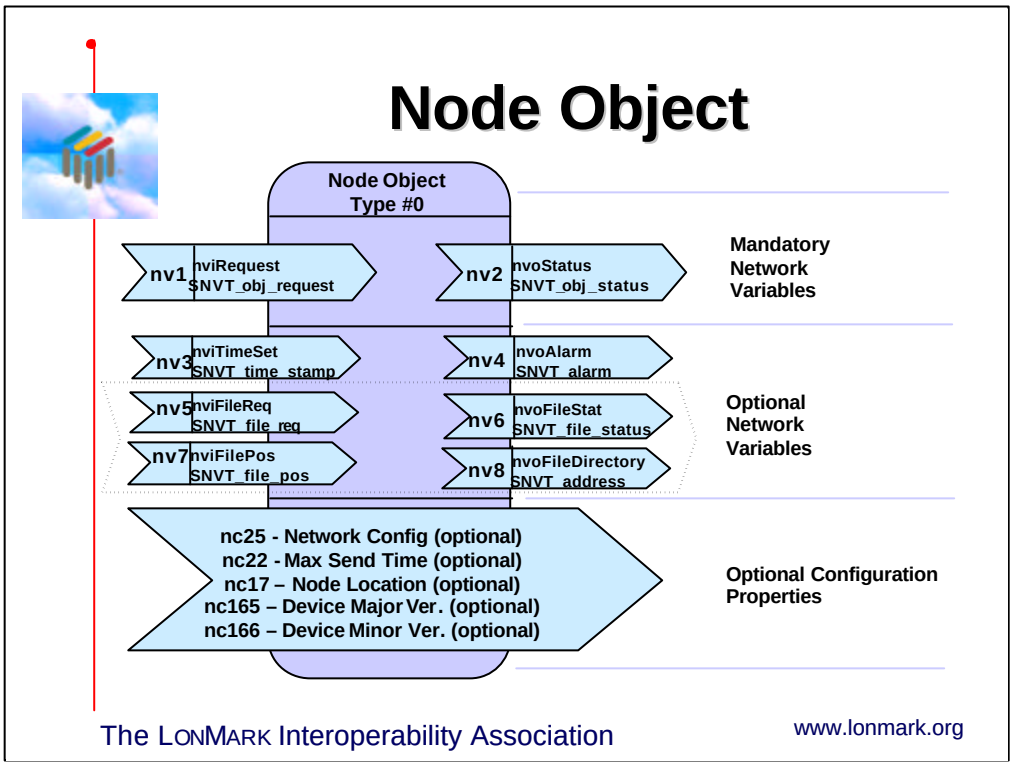
While we may know very little about the node itself from looking at the external interface, we can ascertain certain things about the node, and those will be discussed in this document.



This image is of the external interface of the document. It contains Objects (functional blocks). These functional blocks offer the interface to specific functions that are handled in the node.

If a node has more than one functional block, then a special LonMark Object is required to exist on the node: the Node Object. This functional block allows a network management tool to handle the node in its entirety, rather than the tool needing to know about each functional block on the node.

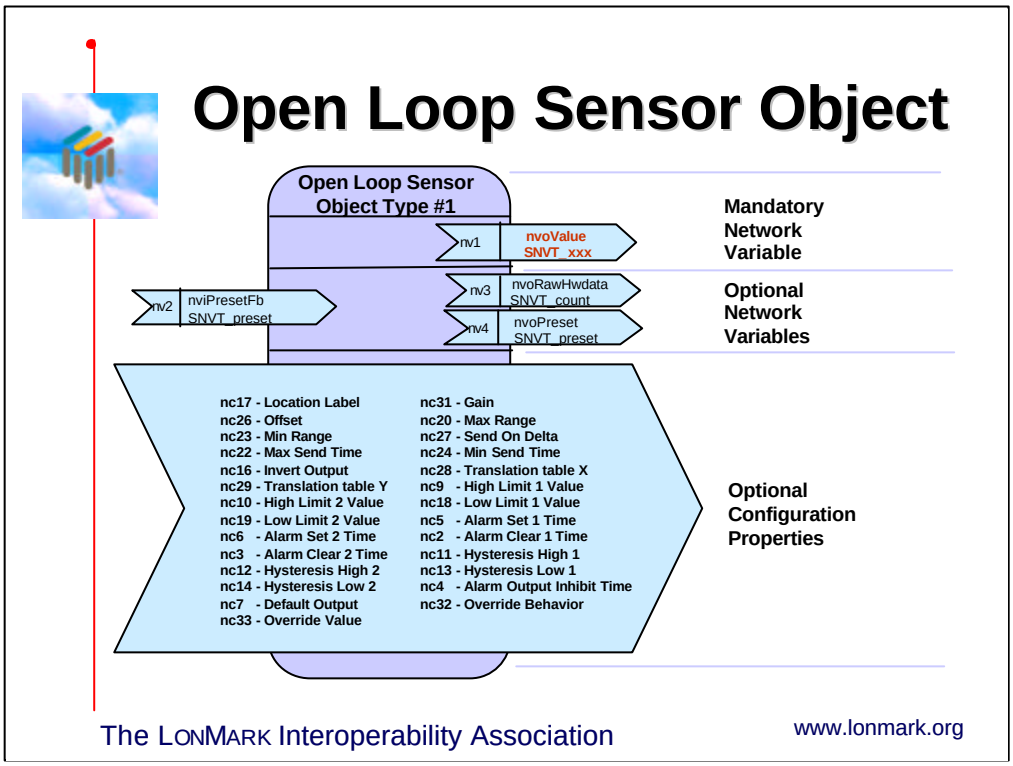
The Node Object also serves other purposes that are described more fully in the *LonMark Application Layer Interoperability Guidelines version 3.2* or earlier, or in the Node Object document, 0000_10.pdf.



Concentrating for a moment on the Node Object, we see it is made-up of Mandatory and Optional Network Variables.

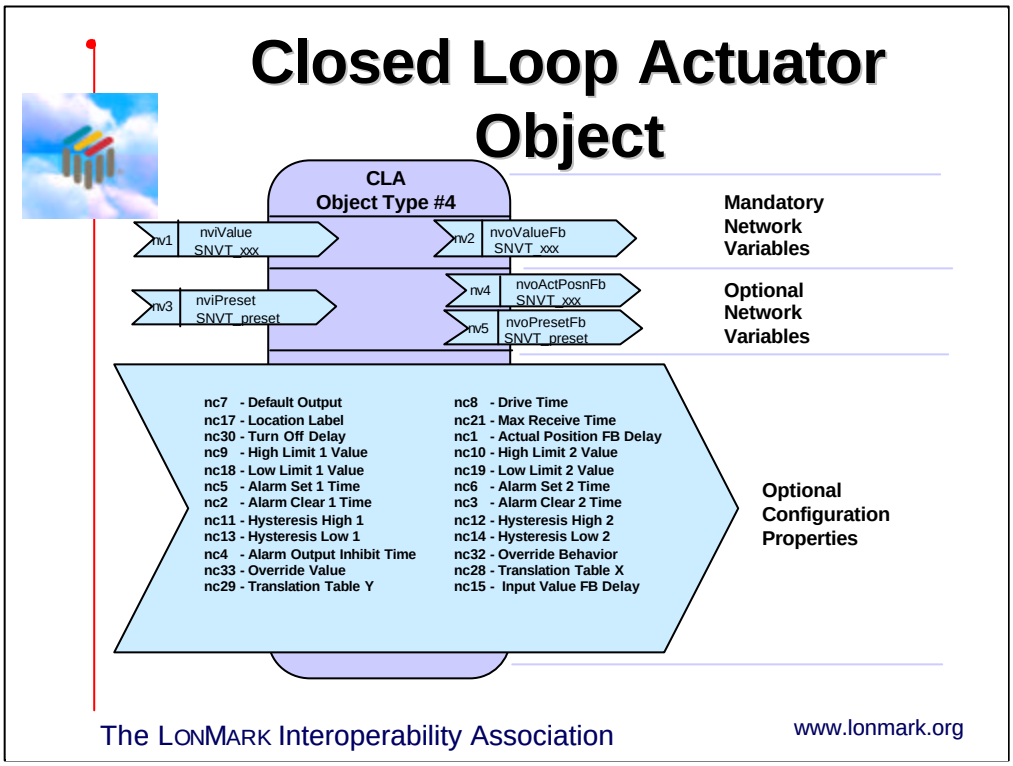
In the “Optional Network Variables” section, there exists in this drawing a dashed area around four network variables. These network variables are used for handling both the LonMark Interoperable File Transfer Protocol and the Direct-Memory Read/Write methods of applying configuration properties to a node. Those methods can be used for other things as well, but that is outside the realm of this document.

The “Configuration Properties” section (in this case optional), shows the supported configuration properties for that functional block (in this case, the Node Object). The list is not exhaustive, as optional properties can be added to any functional block.



The Open Loop Sensor (Object 1) is an example of a “generic” or basic functional block, where the function is defined by the node, rather than by profiling an application-specific function.

This is an easy functional block to implement, but it does not immediately show its use.



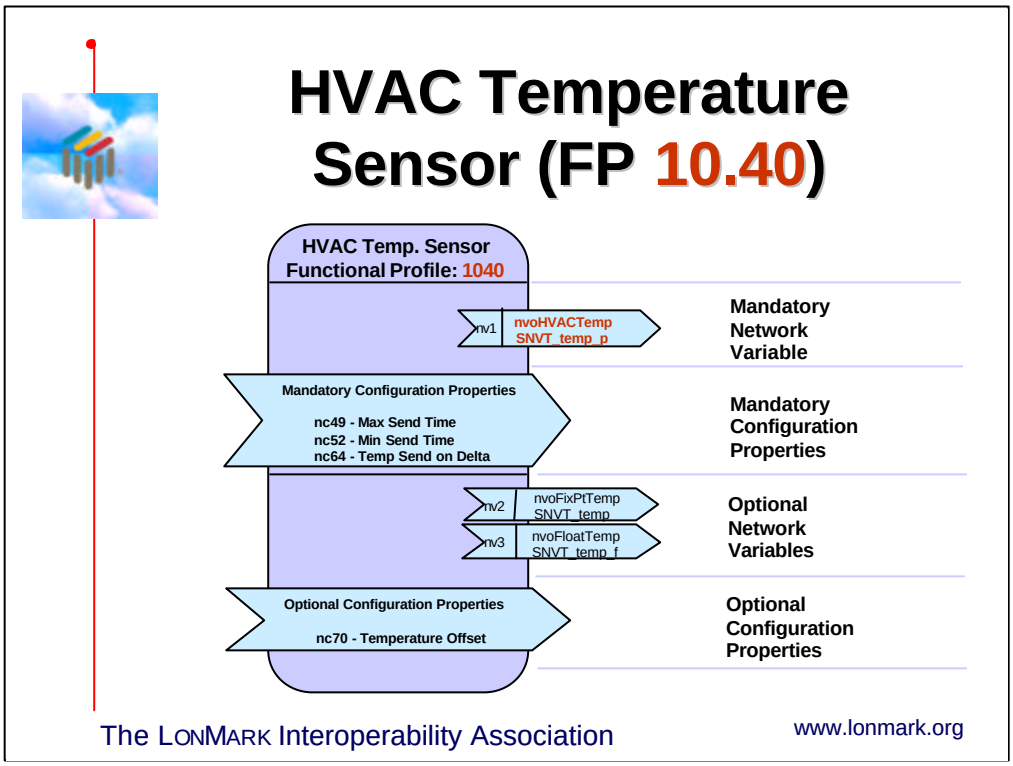
The same situation exists for the Closed Loop Actuator (Object 4).

These basic functional blocks contain “SNVT_ xxx”; a placeholder for the manufacturer’s choice of input/output. A manufacturer can choose from any of the valid Standard Network Variable Types (SNVTs) that exist in the *SNVT Master List* (SNVT.PDF).

All basic functional blocks (Objects 1 through 5) are described in the *LonMark Application Layer Interoperability Guidelines, version 3.2* , along with the Node Object description, or in separate documents for version 3.3 or later.

Because of the ambiguity of the basic functional blocks, LonMark Task Groups have been created to profile application-specific functions that could be implemented as a functional block (Object) in a node.

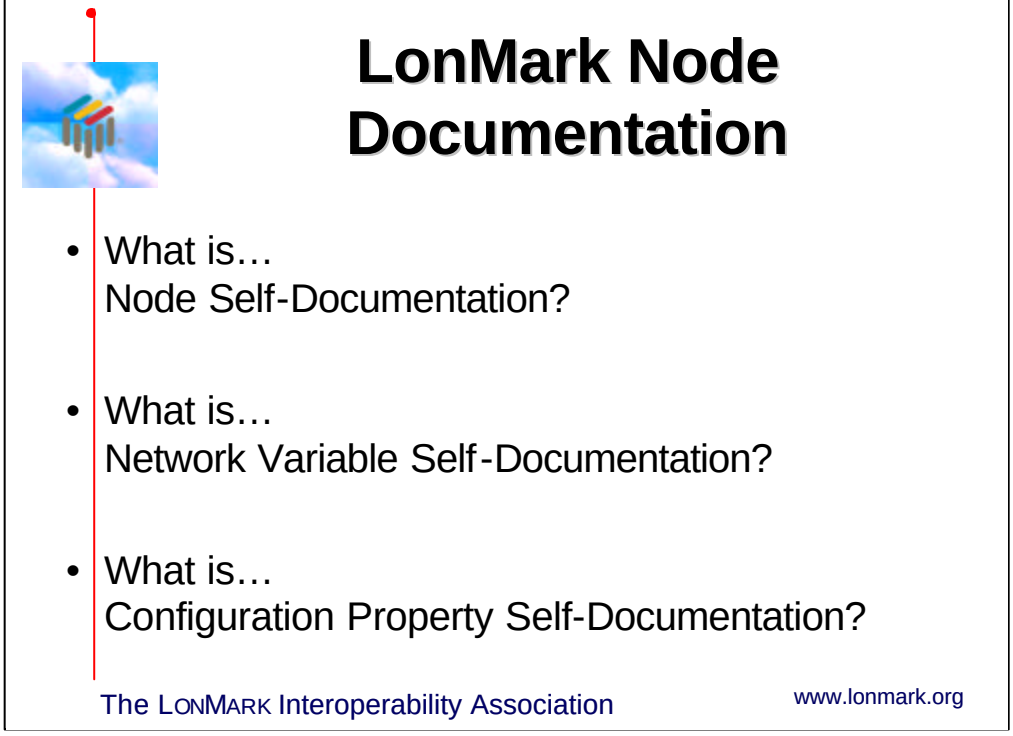
Each Task Group has an elected Leader, whose job it is to coordinate activities of the Task Group.



From out of the depths of the Task Group come simple, as well as complex, profiled application-specific functional blocks. These are known as Functional Profiles.

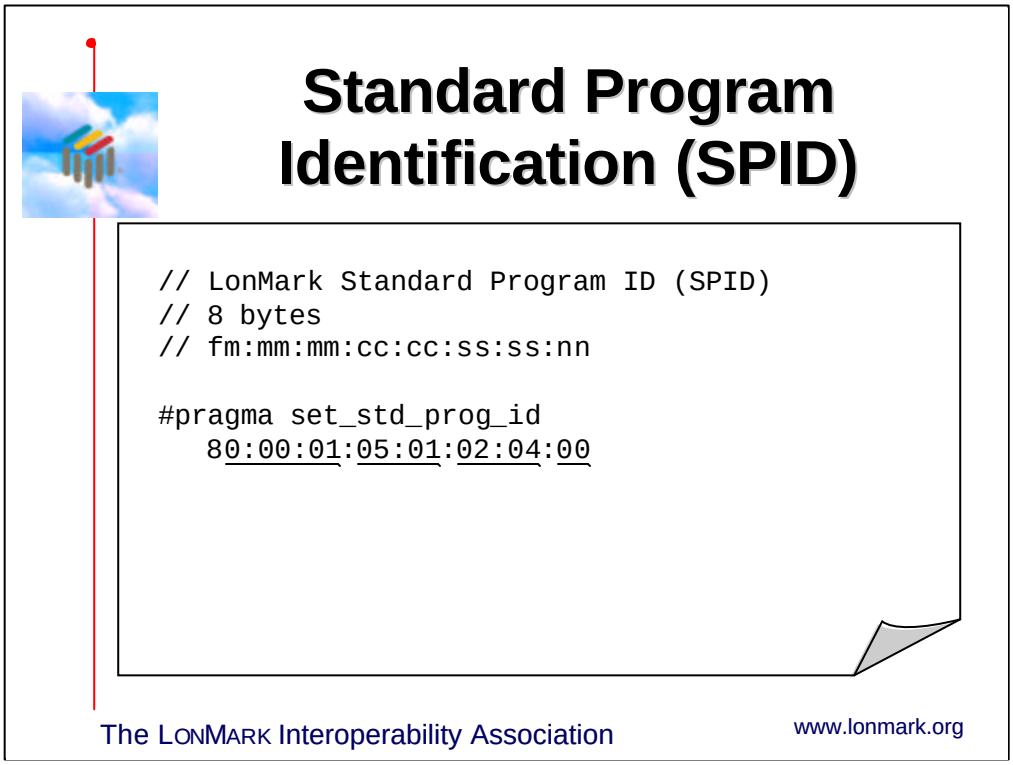
They too are given Object numbers so that they can be implemented in any language, and it is easy for a network specification to call-out an Object number that a purchased node must contain.

You will notice that the HVAC Temperature Sensor (Object 1040) looks remarkably like the Open Loop Sensor (Object 1). The primary difference (aside from mandatory configuration properties, and a difference in optional configuration properties) is that the network variable outputs have a defined set of SNVTs being used. Gone is the "SNVT_xxx" reference, replaced with temperature types that have determined size and resolution. This sensor is application-specific. It gives us temperature in Celsius at defined resolutions. Each node from every manufacturer who implements Object 1040 will contain at least a network variable output of SNVT_temp_p. This aids us in our quest for interoperability.

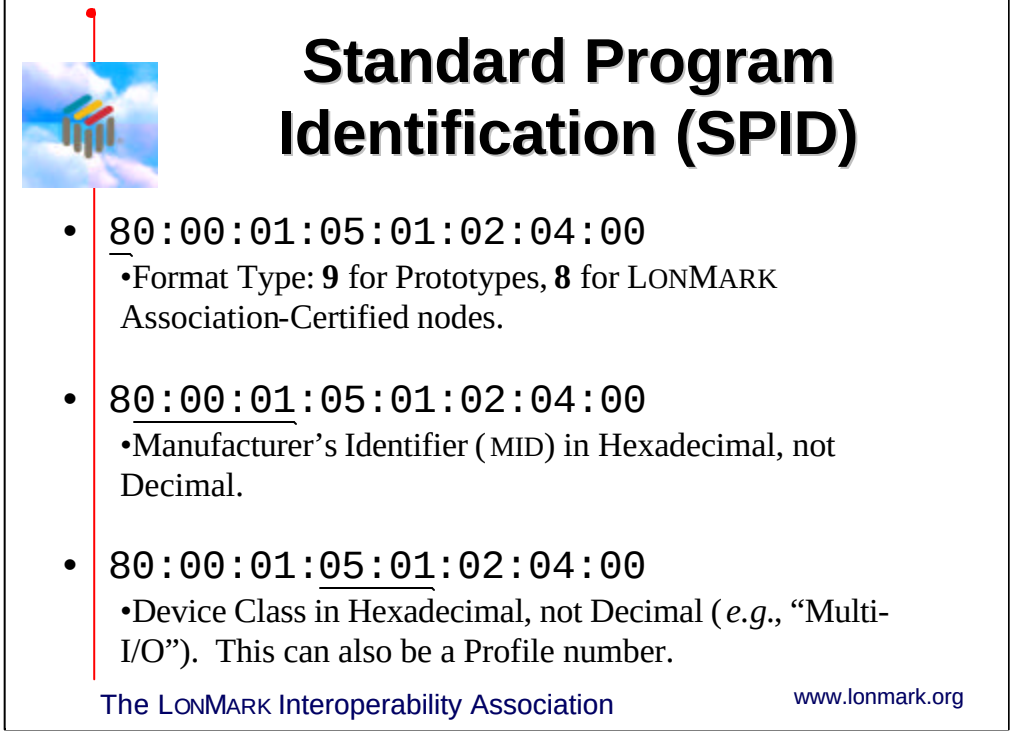


However, there is a cost to interoperability: A node must self-document what it contains. There needs to be a way to determine what is in a node, and how those parts operate—in relation to the external interface.

Remember, LonMark is not about defining products, nor about ensuring interchangeability. LonMark is about making a temperature sensor from Company A, communicate with a thermostat from Company B—without engineers from those companies having to force-fit the nodes together, and without having a translating device sitting between the two nodes.



The first method of differentiation of one node from another is by way of the Standard Program Identifier (SPID). This is a special way for representing a program ID such that network tools can determine a great number of things about a unique product interface.

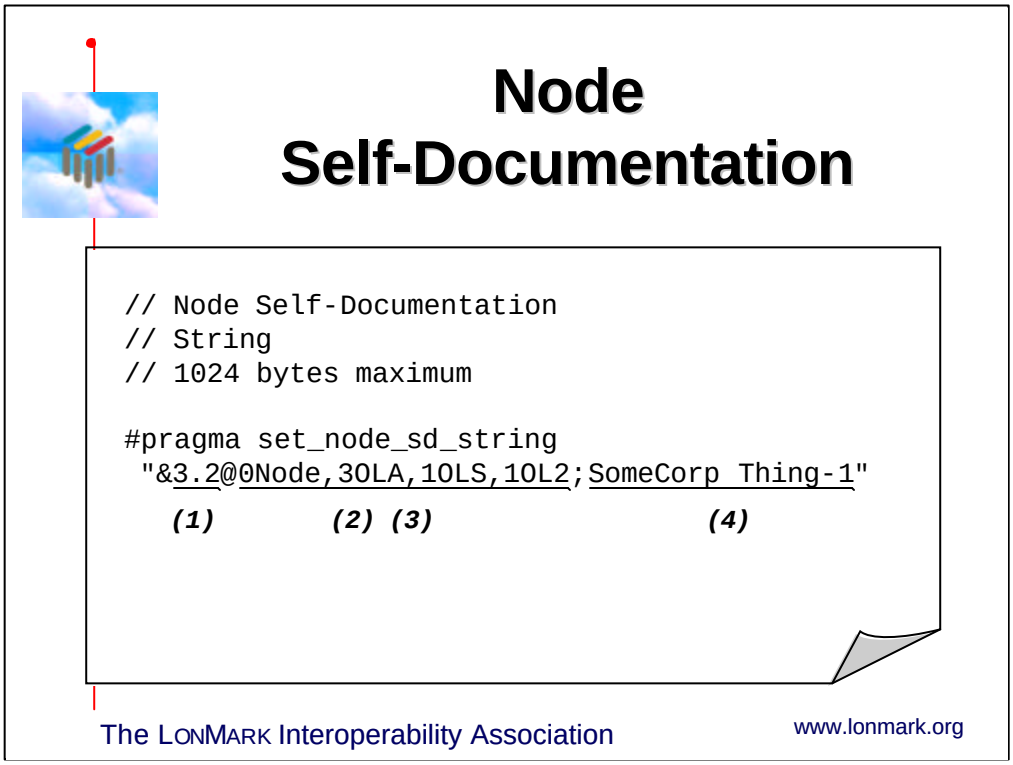


Here you can see how the interface is described by virtue of the parts in the SPID. The definition of these parts is available under the "Products" heading of the LonMark web site in the "Design Guidelines" section. It is available in a downloadable Extensible Markup Language file called SpidData.XML for use with any network management tool that can read such a file.

Standard Program Identification (SPID)

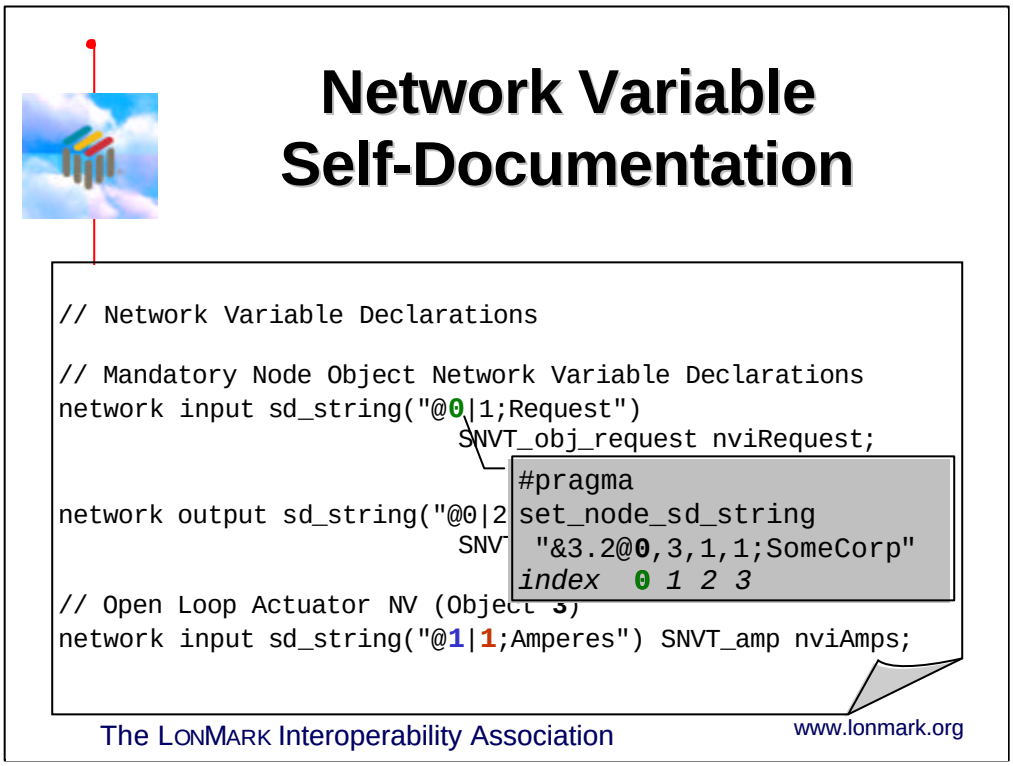
- 80:00:01:05:01:02:04:00
  - •Changeable Types: If the High bit is Set (OR 0x80), then the node has changeable types (*e.g.*, if SPID has 1A:04, then OR 0x80 = 9A:04 *{currently 0x0B is the highest}*).
- 80:00:01:05:01:02:04:00
  - •Device Subclass: Industry Class or Profile-Defined, and Channel Type (Industrial, TP/FT-10).
- 80:00:01:05:01:02:04:00
  - •Model or Revision number.

< continued >



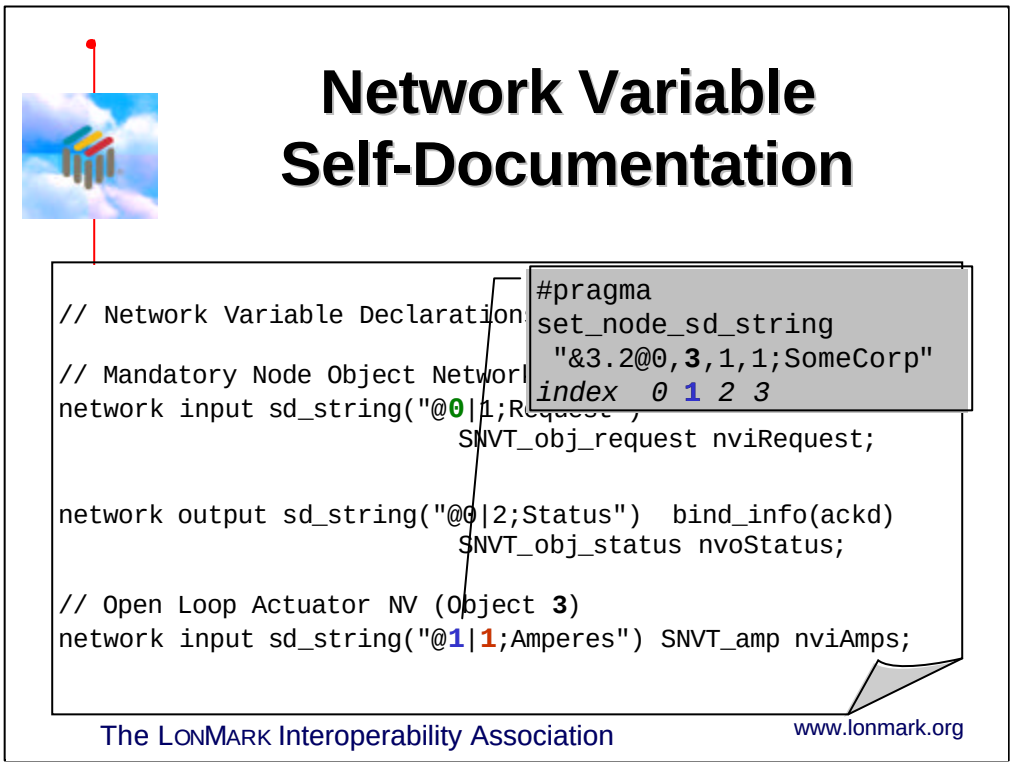
This declaration gives us:

1. The Guidelines version to which the node was written: 3.2
2. A list of functional blocks (Objects) contained in the node, and their declaration order: 0, 3, 1, 1
3. Optional text to give meaning to the Object numbers: Node, OLA, OLS, OL2
4. A string to describe the product in short detail using a product number and/or our company name



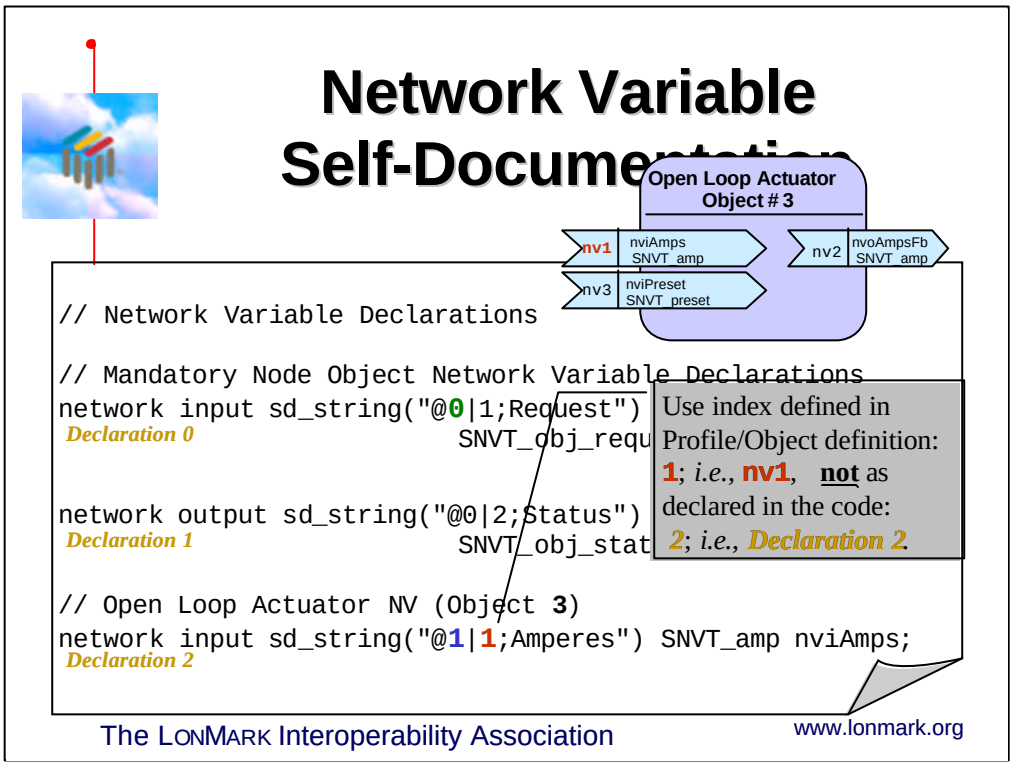
Even the network variables get documented. This is important so that a monitoring tool, or network management tool understands with network variable is being described.

Here the self-documentation string (sd_string) contains a reference to the first functional block on the node. It refers to index 0, which if a node contains a Node Object, will always be the Node Object (which happens to be Object 0). The next example should clarify some of this.



In this example, the index in the sd_string is for the second functional block on the node: index 1. In this case, the second (index 1) functional block is an Open Loop Actuator (Object number 3).

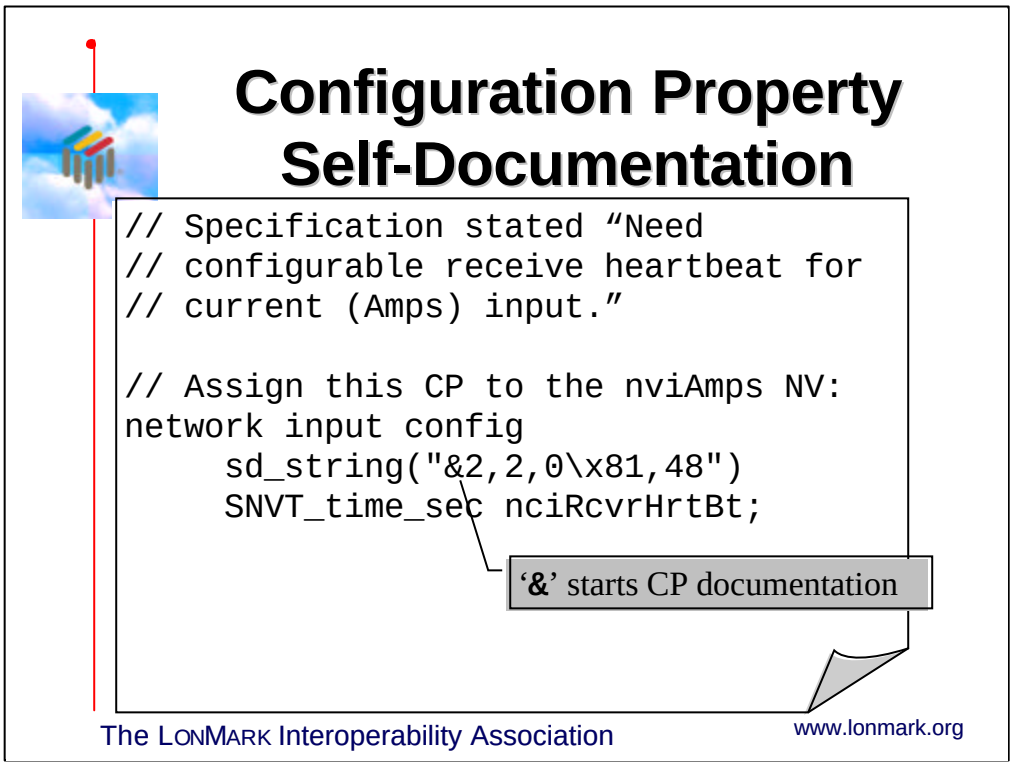
The value in the network variable sd_string will always be the position (index) of the referenced functional block—that reference being from the node's self-documentation string.



In this third example, we give the index of the network variable as it was defined in the Object definition (or Functional Profile). So in this case, since the Object being referenced is Object 3, we need to define which network variable from Object 3 is being declared.

In our case, we are defining network variable 1 (nv1), so that index is placed in the sd_string.

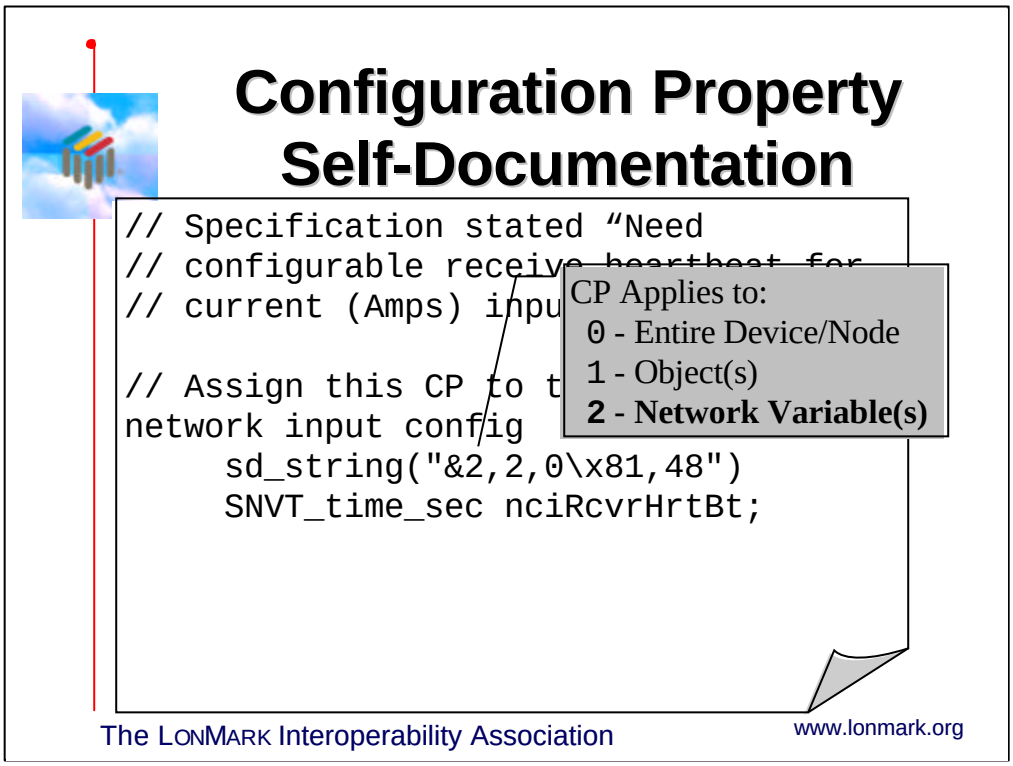
Note that I define the declarations with Declaration numbers (0, 1, and 2). This is important for configuration properties, but should not be used in network variable sd_strings. I point them out here only for clarification of the next examples.



Now we move-on to Configuration Properties’ Self-Documentation.

If our specification states that we “need configurable receive heartbeat for current (Amps) input,” then we need to create a configuration property to facilitate that receive timer value.

Again we use the sd_string method, but the content changes.



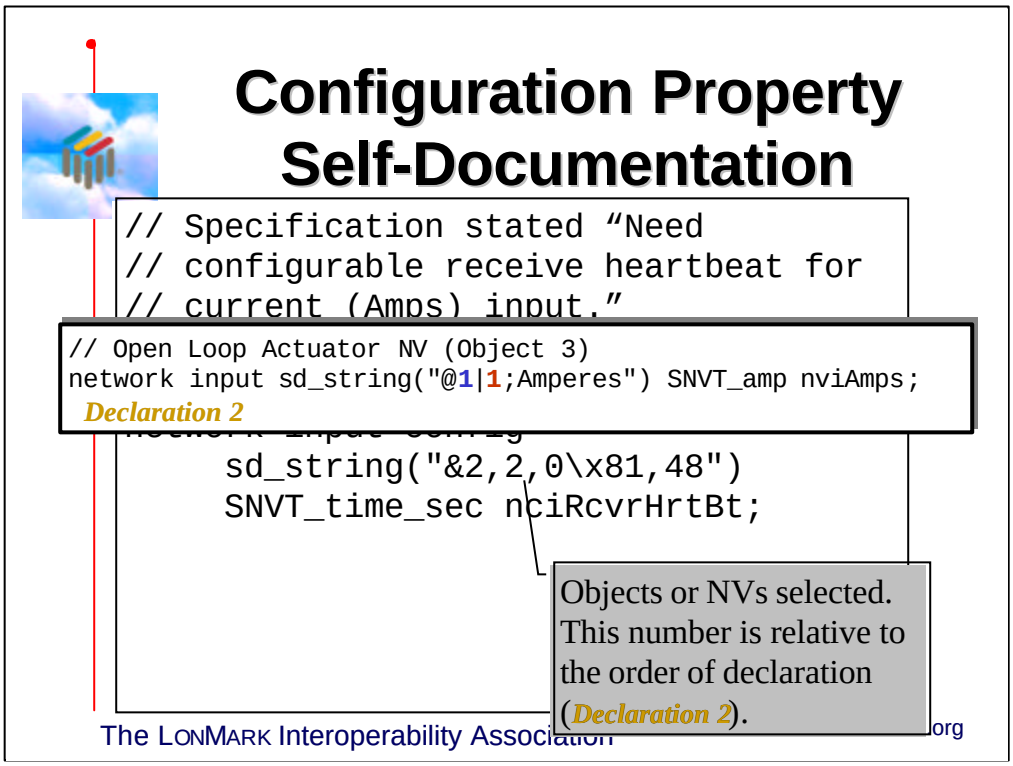
Configuration properties can be applied to

0 = the entire node,

1 = one of more functional blocks on the node, or

2 = one or more network variables on the node.

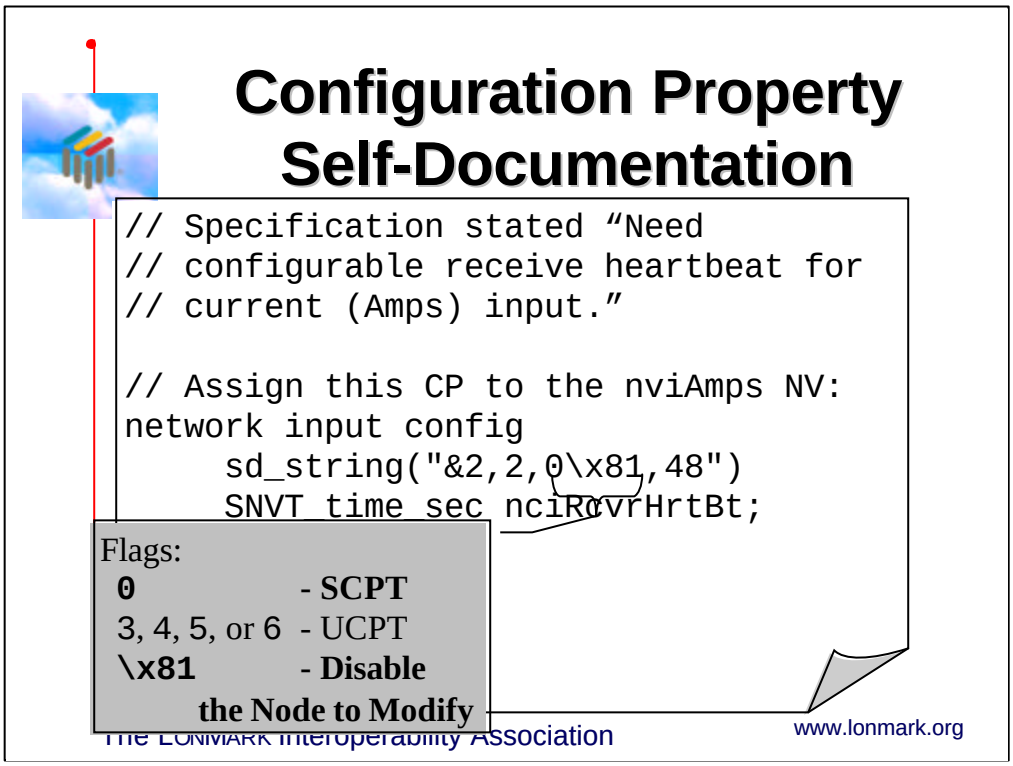
The specification said it should apply to the nviAmps network variable, so we use a "key" to attribute that application. A "2" means that the configuration property will apply to one-or-more network variables.



"Which network variable(s)?"

That is answered in the next field. If you recall, I gave a Declaration number to the network variable declarations on our node. nviAmps was the third (3rd) declared network variable on the node—Declaration 2, since we begin counting at zero ("0").

So, the declaration order is placed in the next field to show us to which network variable the configuration property should be applied. If there were more than one network variable that needed to be placed here, there are several ways to show this. Please consult the *LonMark Application Layer Guidelines* for details.

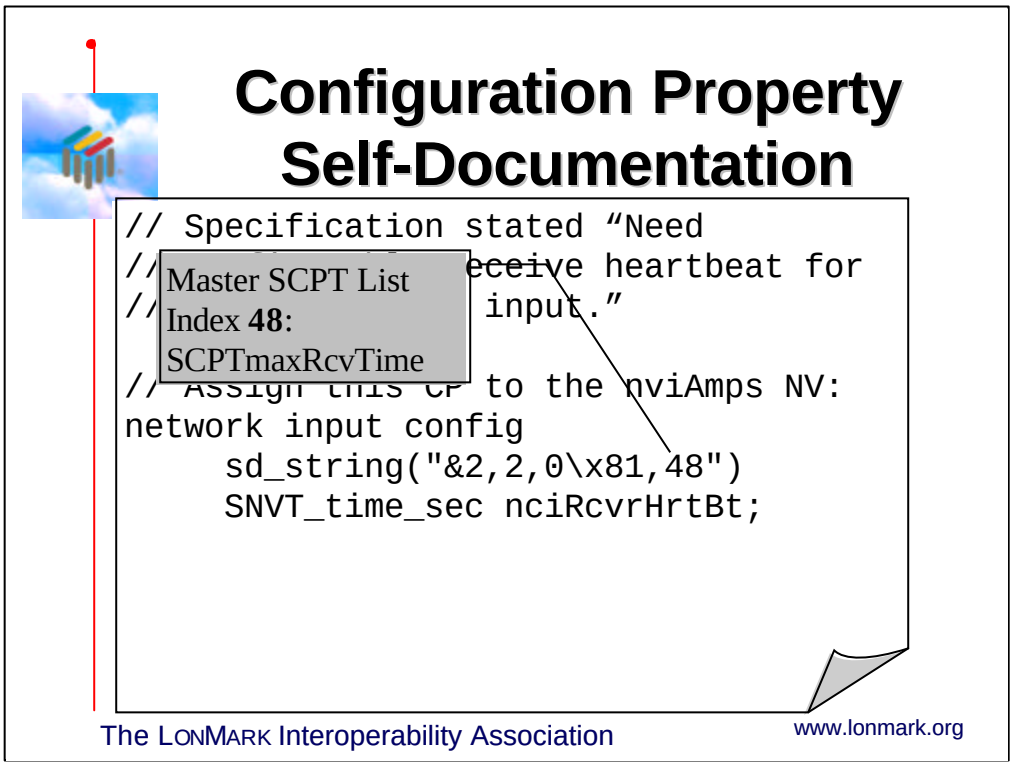


The next field is actually 2 fields: a scope and a flag.

The "scope selector" tells us from what set of files we should get the configuration property definition. If the configuration property is a Standard Configuration Property Type (SCPT), then we should get its definition from the STANDARD.TYP file. That file has a scope selector of zero ("0").

If you are unclear about the whole "scope selector" concept, you can find-out more detail later in this document under: "Common Certification Issues: Scope & SPID."

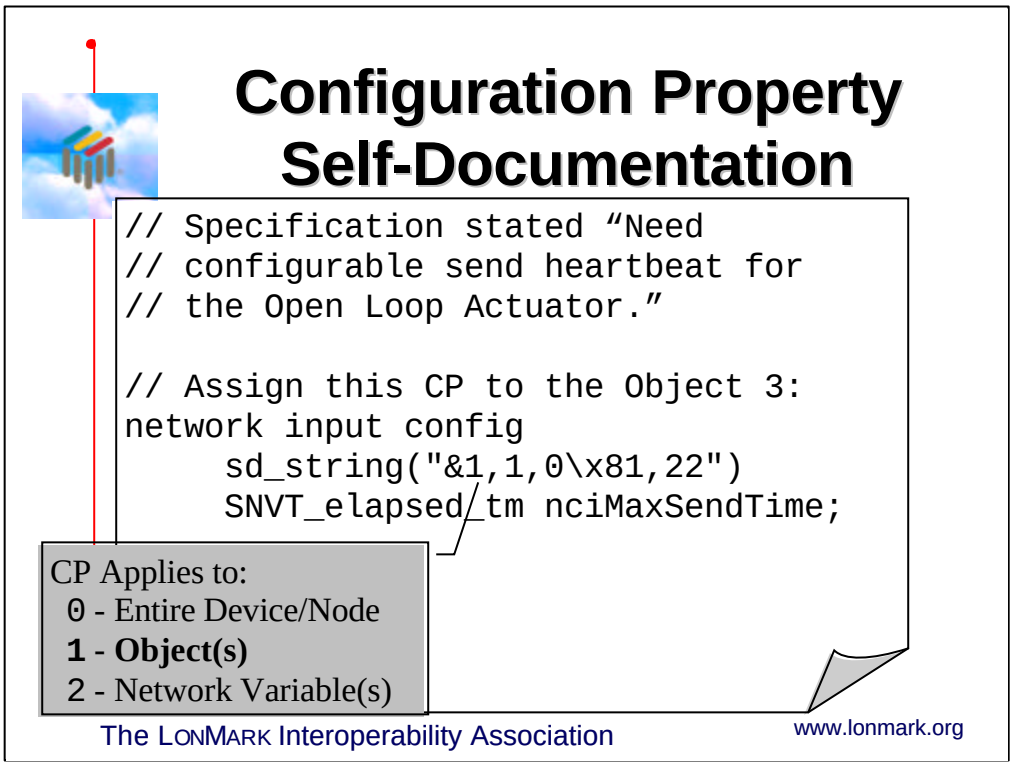
The "flag" is a way to tell the network management tool when to modify the configuration property, and what to do after it is modified. A list of current flags is available in the *LonMark Application Layer Guidelines*.



This last field (for “config” network variable implementations of configuration properties) contains the index into the *.TYP file.

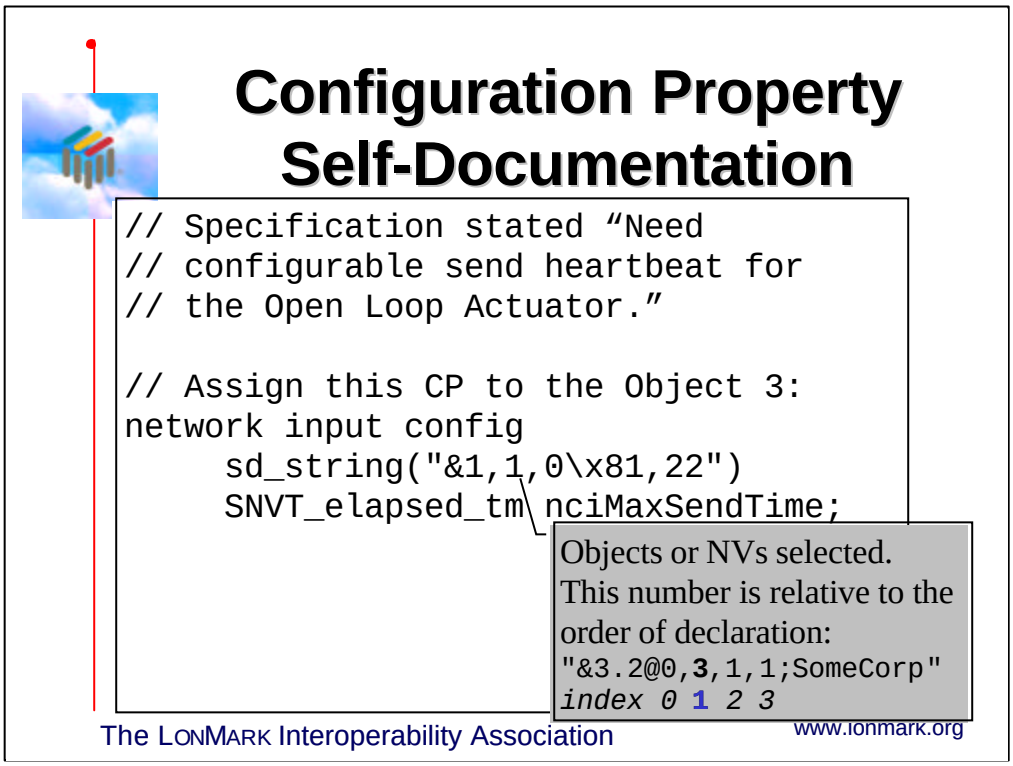
Since we used scope selector 0, we will index into the STANDARD.TYP file to find index 48 (SCPTmaxRcvTime).

We do not need to specify the size of the SCPT, since it is defined by SNVT_time_sec in our declaration. There are two other ways of placing configuration property information into a node, and they both require that we define the size (in bytes) of the SCPT. More on this later.



In this example, our specification says “Need configurable send heartbeat for the Open Loop Actuator.”

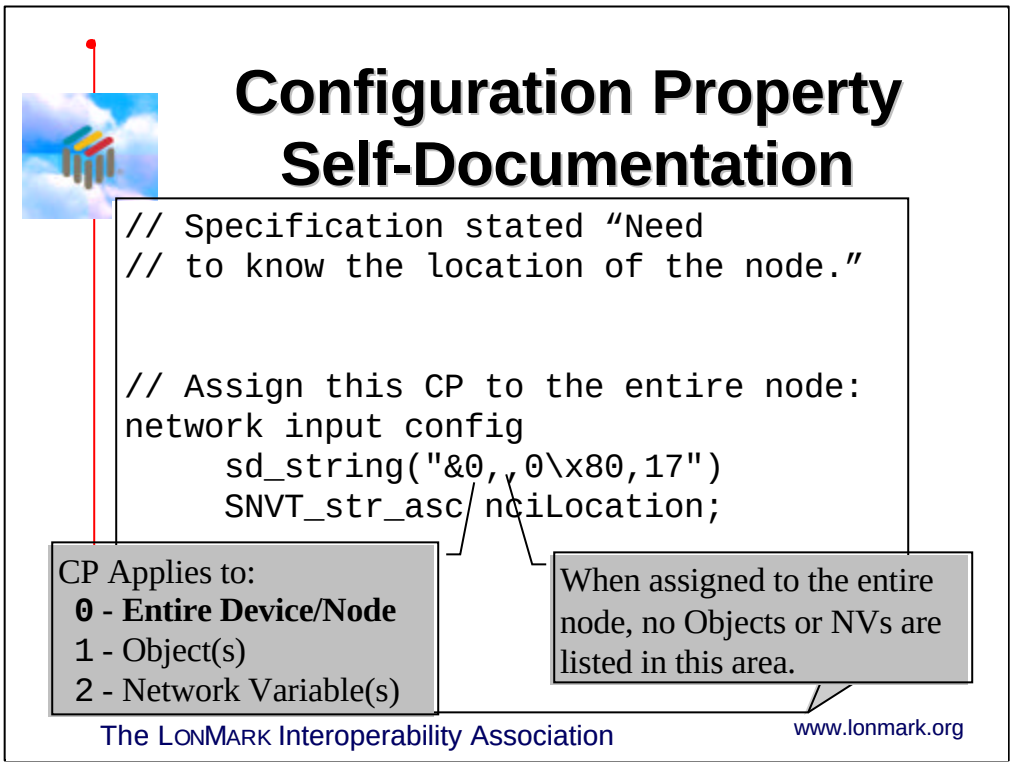
So, we need to apply the configuration property to the entire Open Loop Actuator functional block. To do this we first select “1” as our key to direct this configuration property to one-or-more Objects.



"Which Object(s)?"

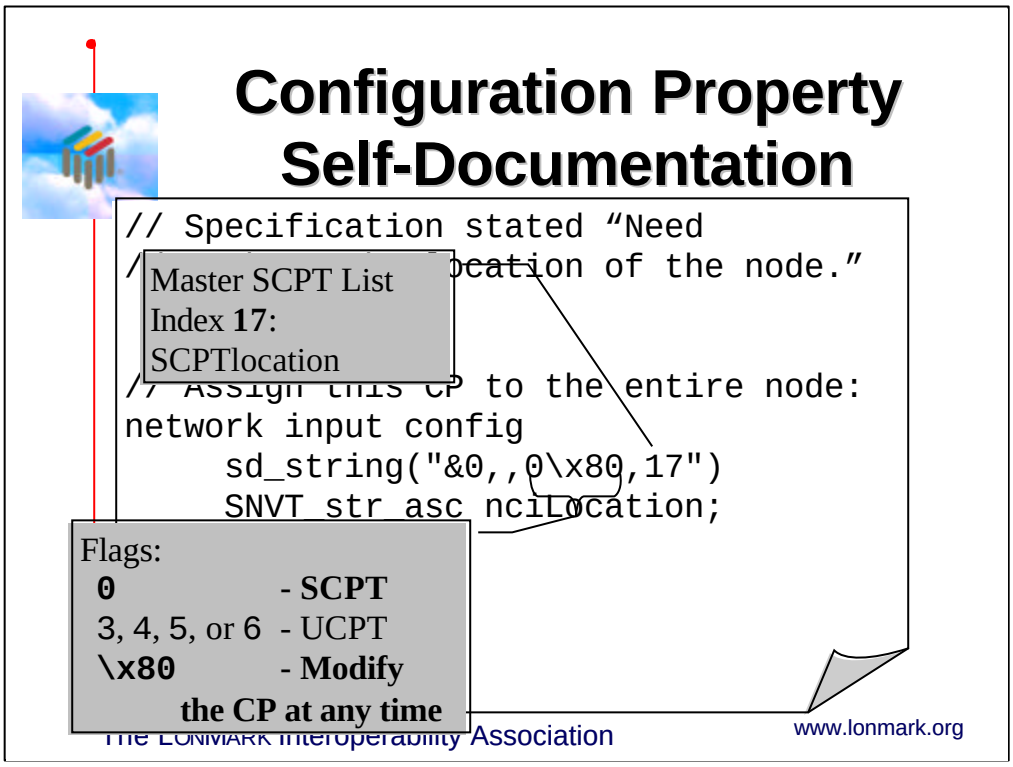
The next field tells us that it is the functional block at index 1. That is an index into the node's self-documentation string. Index 1, in this case, refers to the Open Loop Actuator (Object 3)—the one desired in our specification.

If there were more than one functional block that needed to be placed here, there are several ways to show this. Please consult the *LonMark Application Layer Guidelines* for details.



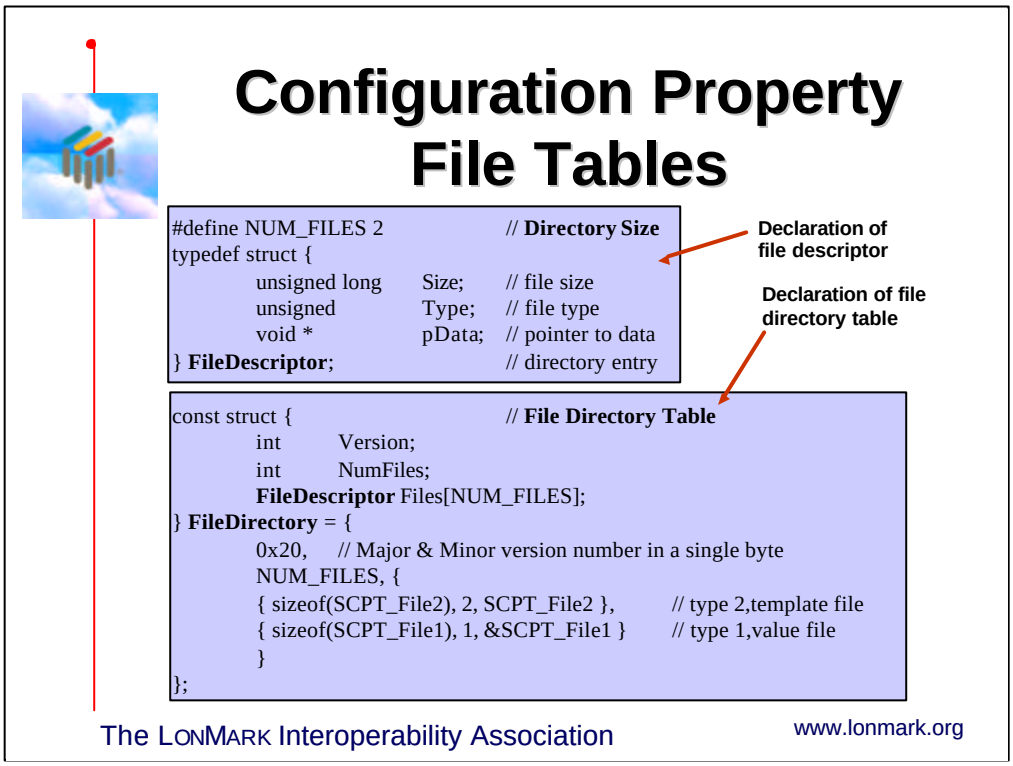
In this example our specification said “Need to know the location of the node.”

So that means our configuration property will apply to the entire node. For that we use “0” in the first field. Since there are no Objects or network variables for this configuration property, the second field is left empty (no spaces or zeroes here, just two commas next to each other).



The flag for this example is different. It is \x80, rather than \x81. The \x80 flag means that the configuration property can be modified at any time, and there is nothing that needs to be done to the node after the modification.

Here again I show you the index into the STANDARD.TYP file for the SCPT (SCPTlocation, index 17).

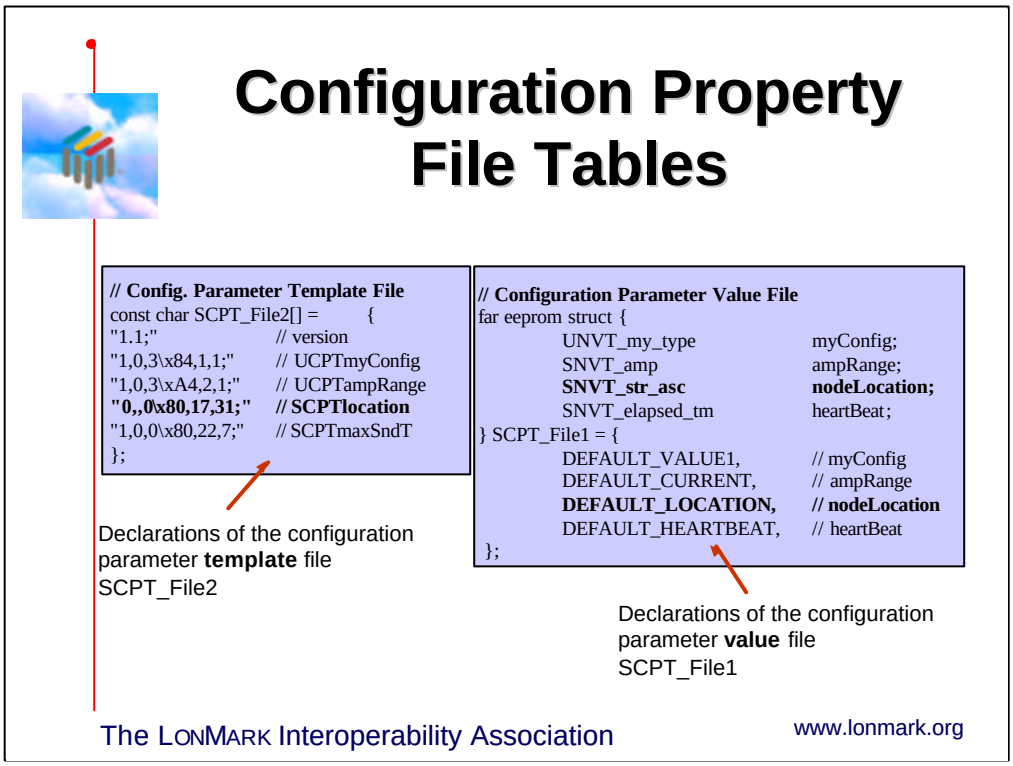


I mentioned two other ways to put configuration information into a node, besides the config network variable examples I've just shown you:

- LONWORKS Interoperable File Transfer method (LW-FTP), and
- Direct-Memory Reading and Writing (DM-R/W).

Each of the three methods have advantages and disadvantages over the others, but that is beyond the scope of this document.

The Neuron C code you see here shows how you would set-up a configuration property table within your node.



I do not go into detail about the coding methods, since it too is beyond the scope of this document.

What I want you to see here is that the configuration properties' self-documentation is very similar to the documentation for config network variables.

The main difference is the addition of the configuration property size (in bytes). In the emphasized (**bold**) text you will see the same declaration as our last example of config network variables. The addition being the "31" after the SCPTlocation index of 17. This allows network tools to know the exact number of bytes to read/write for that configuration property.

LONMARK Product Conformance Review

1. Use the LONMARK Interoperability Guidelines as you design your product.
2. When a Profile exists for your device, use it. Otherwise, you should propose a Profile, or use Objects 1– 4, or create one to meet your needs.
   - Direct questions on the Functional Profiles to the appropriate **Task Group Leader**.
   - Direct questions about the Guidelines and Certification process to **cert@lonmark.org**.

The LONMARK Interoperability Association www.lonmark.org

This next section is a “bonus section.” I have added it as a quick reference to understanding what is needed for Certification of a node.

LONMARK Product Conformance Review

3 When your design is complete, submit
- The Conformance Application (LM_CONFM.PDF) and
- Completed Checklist, and
- LonMark Logo License Agreement

via fax to LonMark at +1-408-328-3832,

and send by postal mail also.

4 Submit your product datasheet as a single **ZIP** attachment to cert@lonmark.org in **DOC** or **PDF** format.

< continued >

LONMARK Product Conformance Review

5 Submit your product XIF file, and if needed, your TYP, FMT, FPT, and language files—as a single **ZIP** attachment to cert@lonmark.org.

(currently, 2-byte characters, like Chinese, Japanese, and Korean are not an option, but we hope to correct this problem in the future.)

How will I know if the TYP, FMT, FPT, and language files are "needed" for my device?

< continued >

LONMARK Product Conformance Review

How will I know if the TYP, FMT, FPT, and language Device Resource Files (DRFs) are "needed" for my device?

If you have:

- a node that must have its UCPTs set to function,
- created a UNVT or UCPT that you want exposed,
- added UNVTs to a Standard Object (SFPT), or
- created your own Object (UFPT)

then you need Device Resource Files (DRFs), or a Passive Configuration Tool (PCT).

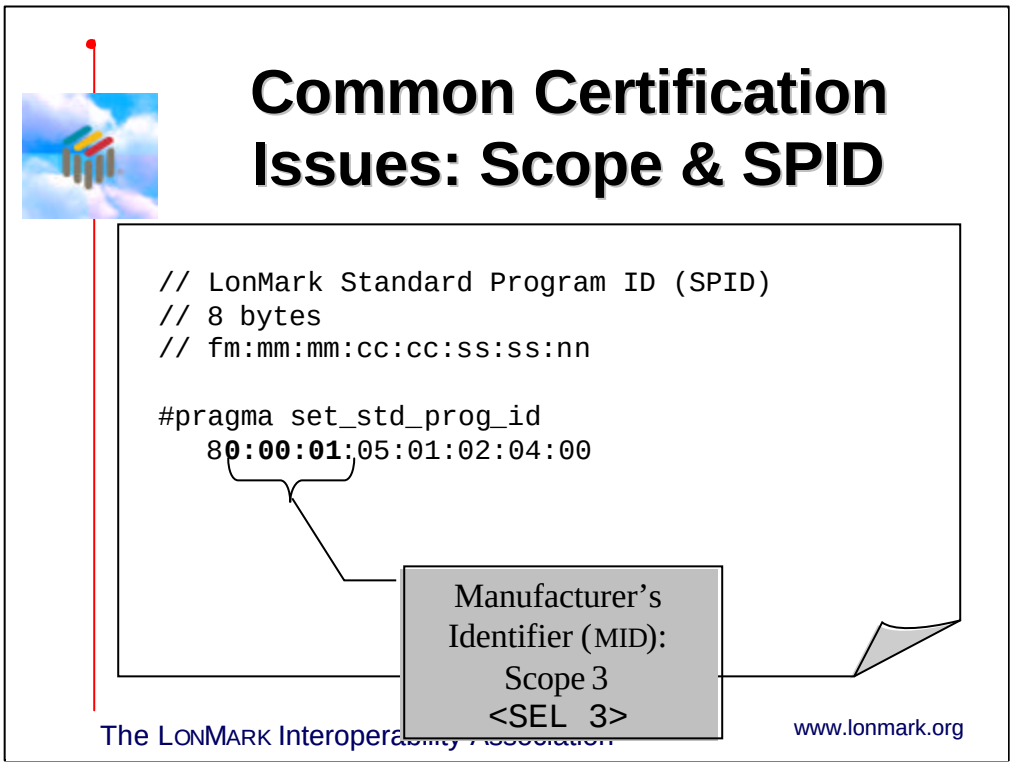
The LONMARK Interoperability Association www.lonmark.org

< continued >

Common Certification Issues

- User-defined NVs, CPs, and Objects should use Selector/Scope of 3, 4, 5, or 6 — not 0, 1, nor 2 *(these are reserved)*.
- User-defined Objects should be indexed from 200.00 to 250.00 (C8:00 – FA:00).
- Nodes created using Objects 1– 4 should use a Device Class that is not assigned to a Standard Functional Profile. Object 5 should never be used.

< continued >

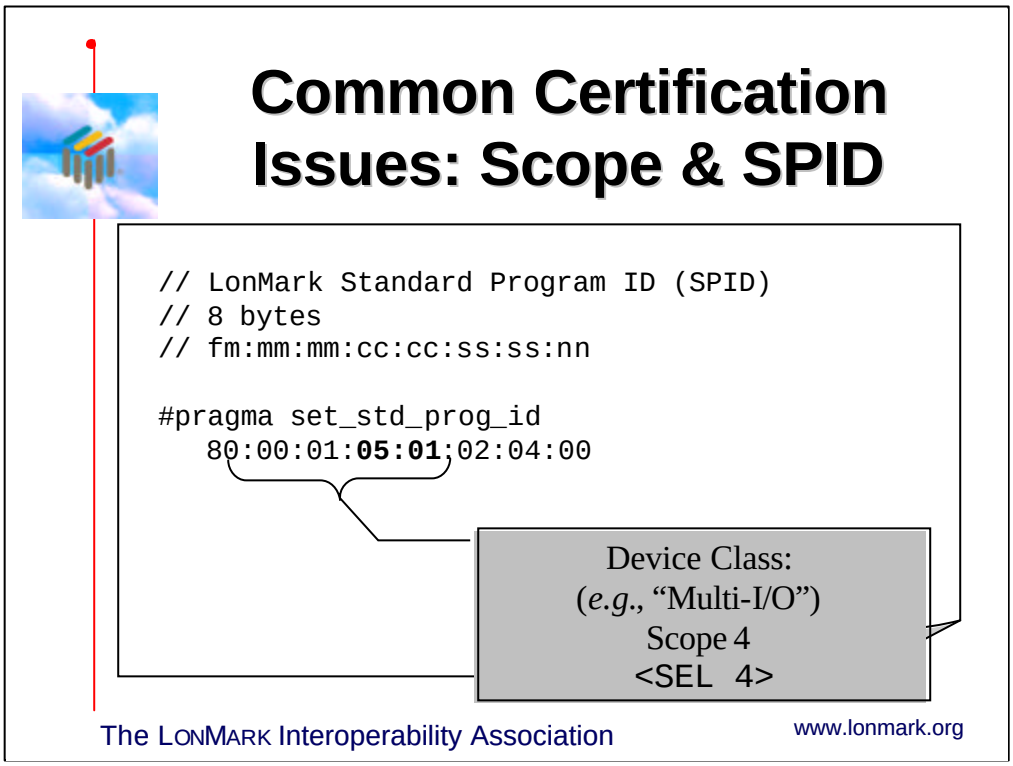


I mentioned earlier in the document that if "scope selector" concepts were new to you that they would be discussed here.

A scope selector of zero (0) is used for all references to STANDARD.* files. Scope selectors of 1 and 2 are reserved for future use by the LonMark Association, so those should not be used, nor referenced by manufacturers at this time.

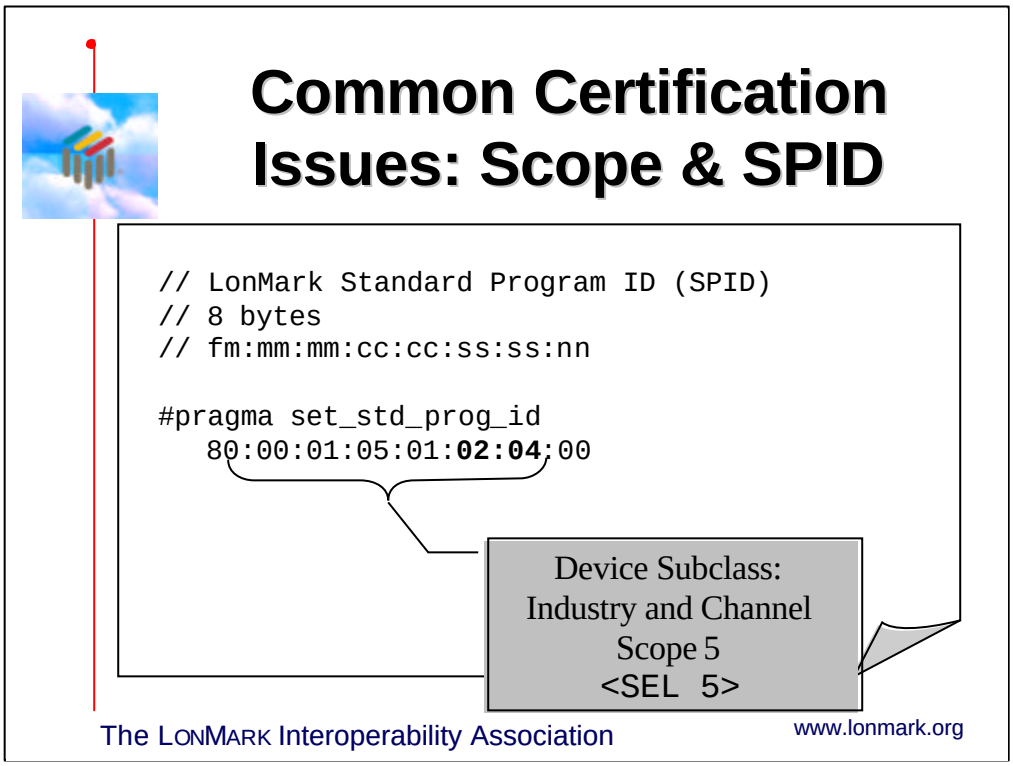
Scope selectors of 3 define files that represent every LonMark Association-compliant product a manufacturer has created, or will create in the future.

Scope selector 3 files can reference scope selector 3 and 0 files.



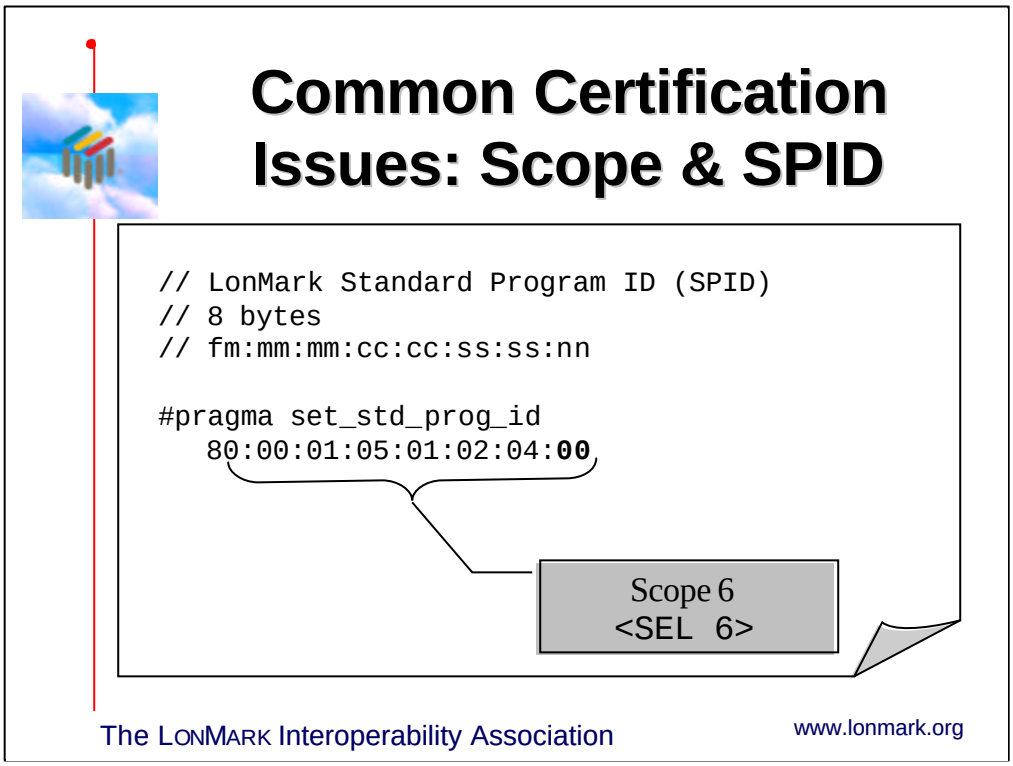
Scope selectors of 4 define files that represent every LonMark Association-compliant product a manufacturer has created, or will create in the future, **for a specific Device Class**.

Scope selector 4 files can reference scope selector 4, 3, and 0 files.



Scope selectors of 5 define files that represent every LonMark Association-compliant product a manufacturer has created, or will create in the future, for a specific Device Class, **within a Device Subclass listing (*e.g.*, No Changeable Types, Industrial Use, and TP/FT-10 Channel)**.

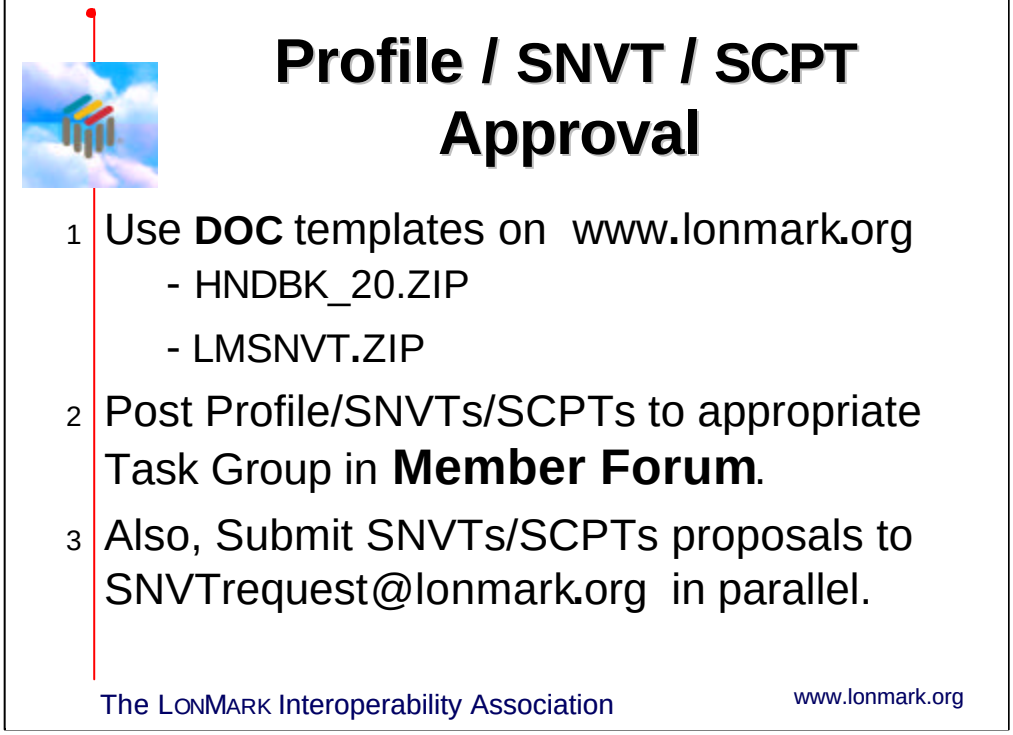
Scope selector 5 files can reference scope selector 5, 4, 3, and 0 files.



Scope selectors of 6 define files that represent only one LonMark Association-compliant product external interface that a manufacturer has created for a specific Device Class, within a Device Subclass listing (*e.g.*, No Changeable Types, Industrial Use, and TP/FT-10 Channel), **for a single model number**. The model number is arbitrary, but is unique for a specific external interface.

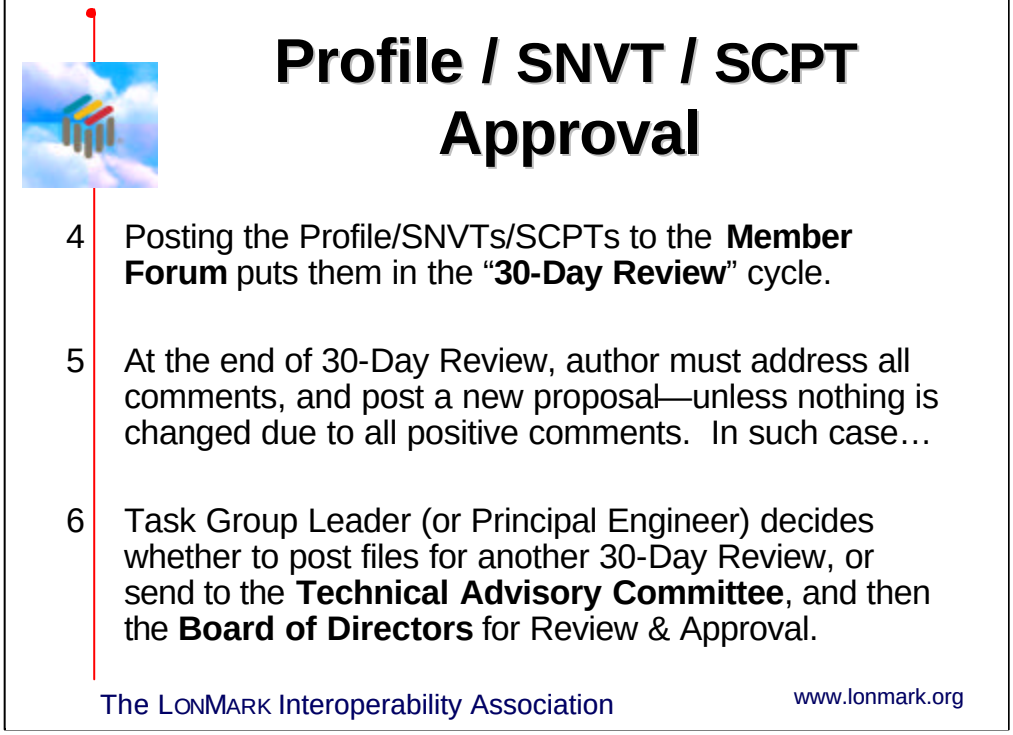
This means that a node can possible have several physical iterations, but they all contain the exact same external interface—right down to the communications channel, the industry use, the device class, manufacturer Identifier, and external interface model number.

Scope selector 6 files can reference scope selector 6, 5, 4, 3, and 0 files.



There are templates, on the LonMark web site under the “Members area” heading, that should be used for submitting new Functional Profiles, SNVTs, and SCPTs for approval and addition to the Master Lists.

There will soon be (if there is not already) a new set of templates with the new Member Handbook (HNDBK_20.ZIP).



The Technical Advisory Committee is a group of six (6) people appointed yearly by the Board of Directors to make approval/denial recommendations on technical issues that are to be voted-on by the Board of Directors.

LONMARK® Association

Headquarters
(Main Office)
550 Meridian Avenue
San Jose CA 95126-3422 USA

TEL +1-408-938-5266
FAX +1-408-790-3838

The LONMARK Interoperability Association

www.lonmark.org

Please feel free to contact us if you have questions about this document, or any other LonMark Association-related question.

We are always happy to assist if we can.

- Jeremy ROBERTS
  Main Technical Office