STANDARDIZATION OF THE INSTRUMENTATION HARDWARE ABSTRACTION LANGUAGE IN IRIG 106

John Hamilton, Ronald Fernandes, Timothy Darr
Knowledge Based Systems, Inc
1408 University Drive East
College Station, TX 77840

Charles H. Jones
812TSS/ENTI
Edwards AFB, CA

Ray Faulstich
CSC Range and Engineering Services
21841-B Three Notch Road
Lexington Park, MD 2065

ABSTRACT

Previously, we have presented an approach to achieving standards-based multi-vendor hardware configuration using the Instrumentation Hardware Abstraction Language (IHAL) and an associated Application Programming Interface (API) specification. In this paper we describe the current status of the IHAL standard. Since the first introduction of IHAL at ITC 2006, the language has undergone a number of additions and improvements. Currently, IHAL is nearing the end of a 2-year standardization task with the Range Commanders Council Telemetry Group (RCC TG). This paper describes the standardization process in addition to providing an overview of the current state of IHAL. The standard consists of two key components: (1) the IHAL language, and (2), the IHAL API specification.

KEYWORDS

IHAL, TMATS XML, PCM, Instrumentation configuration, metadata, web services, XML, API

INTRODUCTION

Current Instrumentation Support Systems (ISSs) are required to use vendor-specific languages in order to support configuring instrumentation systems prior to testing. Since there are a number of vendors selling various data acquisition, control, transmission and storage components, an ISS not only has to interface with these different systems, but also different vendor software that supports various instrumentation systems. Moreover, hardware vendors typically supply their own configuration software that works only with their own products. Thus, an instrumentation engineer who is trained on one vendor's software must learn a new software system in order to

use a different vendor's hardware, raising the cost of switching vendors, reducing competition, and making interoperability almost impossible.

In 2006 [1] and 2008 [2], we introduced the Instrumentation Hardware Abstraction Language (IHAL) and subsequent extensions. The IHAL language standard specifies an XML-based language for describing instrumentation hardware in a vendor-neutral way. The central concept in IHAL is the "configurable attributes" (i.e. settings) that each device exposes to the user. However, IHAL is also capable of describing environmental and physical attributes of each device, such as its size and shape, operating temperatures, and power consumption.

In 2010 [3], we introduced an application programming interface (API) and methodology for interacting with vendor hardware configuration engines with IHAL. The IHAL vendor API enables IHAL to be used not only as a description language for describing instrumentation hardware, but also as a command and query language for configuring instrumentation hardware.

The IHAL API standard defines a set of functions that an instrumentation hardware vendor can implement to provide access to their configuration engine to external applications. All inputs and outputs to the functions are properly-formatted IHAL XML documents. This API allows vendors to expose the functionality of their configuration engines in a vendor-neutral way, without disclosing the inner workings of their proprietary configuration logic. In this way, vendor-neutral 3rd-party applications can be developed to configure the hardware of any vendor who implements the standard API.

Currently, IHAL is nearing the end of a 2-year standardization task with the Range Commanders Council Telemetry Group (RCC TG). This paper describes the standardization process in addition to providing an overview of the current state of IHAL with examples.

IHAL STANDARDIZATION PROCESS

In October 2010, the Vehicular Instrumentation/Transducer (VIT) committee of the RCC TG initiated a task to develop an XML format for describing instrumentation hardware. The initial requirement was for the task to satisfy the original intention of the Telemetry Attributes Transfer Standard (TMATS) 'H' group while making use of the previous work done in developing IHAL. TMATS 'H' group has been added as a placeholder for hardware descriptions, but has never been implemented.

The VIT committee formed a working group consisting of committee members, original IHAL developers, instrumentation vendors, and other government and commercial members of the T&E community. At the time the task was initiated, IHAL had been shown to support configuration of analog signal conditioning hardware and PCM data encoders. The working group first reviewed the existing IHAL design in detail, making some minor changes to the fundamental structure of the language and to the way analog signal conditioners are described. The group then expanded the language with support for additional hardware types, including bus monitor cards, data acquisition units, network data encoders, and recorders.

Task work was carried out during the RCC TG's twice-a-year meetings and also during monthly teleconferences with the working group. A web-based portal was maintained for sharing the

latest schema, documentation, and meeting minutes. The portal also included discussion forums enabling working group members to continue discussing the emerging standard between meetings.

Following the first year of standardization work, the IHAL language and API were validated through a multi-vendor demonstration held at the 2011 International Telemetering Conference. To support this demo, two different instrumentation hardware vendors implemented the IHAL API for a subset of their hardware. Meanwhile, Knowledge Based Systems, Inc (KBSI) updated their "InstrumentMap™" tool to comply with the latest IHAL standard. During the demonstration, InstrumentMap™ was able to successfully communicate with both L-3 Telemetry-East's VistaTEC application and the XidML-based DAS Studio application.

In the demonstration scenario, each vendor configuration tool was pre-loaded with a configuration consisting of one data acquisition unit connected to three different analog signal conditioning cards, for a total of seven unique devices from two different vendors. InstrumentMap™ then queried each configuration tool for the current configuration, and displayed both configurations in a single view. The settings on each device were then changed individually within InstrumentMap™. Each setting change was immediately communicated to the appropriate vendor configuration tool through the IHAL API. The vendor configuration tools in turn validated the change and provided immediate feedback about the impact to other settings. A screenshot of the InstrumentMap™ application is shown in Figure 1.

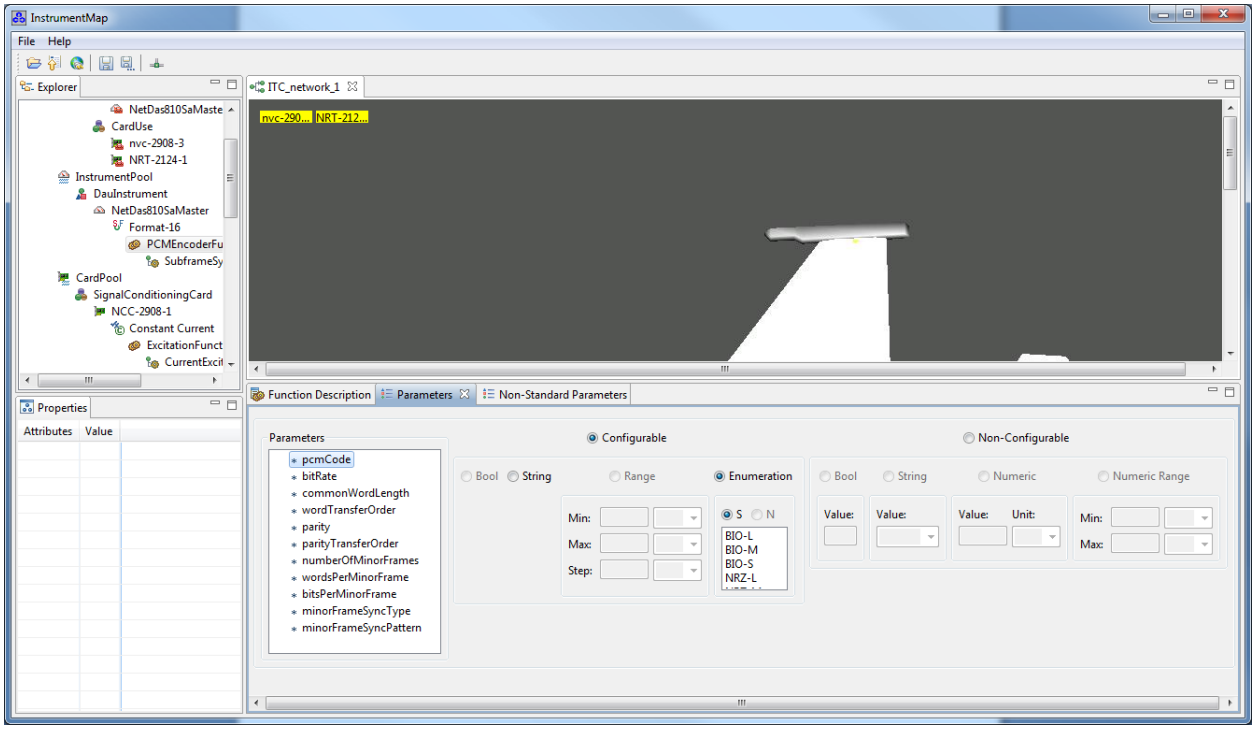

Figure 1: InstrumentMap™ application showing controls for changing device settings

As of the writing of this paper, work on the IHAL schema and API specification has concluded, and the standard will be distributed throughout the RCC for feedback. The IHAL standard is expected to be published as part of the IRIG 106, Chapter 9 standard in 2013.

KEY CONCEPTS IN IHAL

The IHAL language is focused on the settings available on devices, referred to as "configurable attributes." The primary purpose of IHAL is to describe these attributes both in terms of how they are currently configured and how they can be configured.

One important concept in IHAL is that of "pool"- versus "use"-level device descriptions. The IHAL instrument pool is used to describe instrumentation hardware that is available for use in a configuration. At the pool level, each device is described in detail with all of the information you would typically find in a vendor's spec sheet, such as the settings, size, weight, environmental characteristics, connectors, etc. Pool-level descriptions are independent of any specific configuration, and describe what the device does (i.e. what functions it performs) and what settings are available, as well as what the valid values are for each setting.

On the other hand, use-level descriptions define a specific instance of a device along with the current values for each setting in a specific configuration. One way to understand the difference between pool and use-level descriptions is to think of pool-level devices as being uniquely identified by a model number and use-level devices by a serial number.

Another key concept is that of a hardware "function." Functions in IHAL represent a specific capability provided by a device. Each function can contain attributes (configurable and non-configurable) as well as sub-functions. For example, the analog signal conditioning function can have configurable attributes such as minimum and maximum input voltages and an offset. The analog signal conditioning function can also contain amplification, filtering, and analog-to-digital conversion sub-functions.

A device in IHAL may also be composed of one or more hardware "channels." A channel in IHAL defines a collection of functionality which may be repeated multiple times within the device. IHAL channels are directly analogous to the channels on a card. Channels in IHAL are always composed of functions, but not all functions are part of a channel. Some functions may be associated directly with the device itself.

Another way in which functionality can be grouped in IHAL is through a "format." Formats are used only in the description of data encoders and are used to describe the configurability of a data format. For example, a PCM data encoder can contain one or more PCM formats, which in turn are described in terms of the PCM encoder "function" and associated attributes such as number of minor frames, word length, and parity.

The complete logical model of instrumentation devices in IHAL is illustrated in Figure 2. This shows how a device in IHAL can be decomposed into functions, channels, and formats.

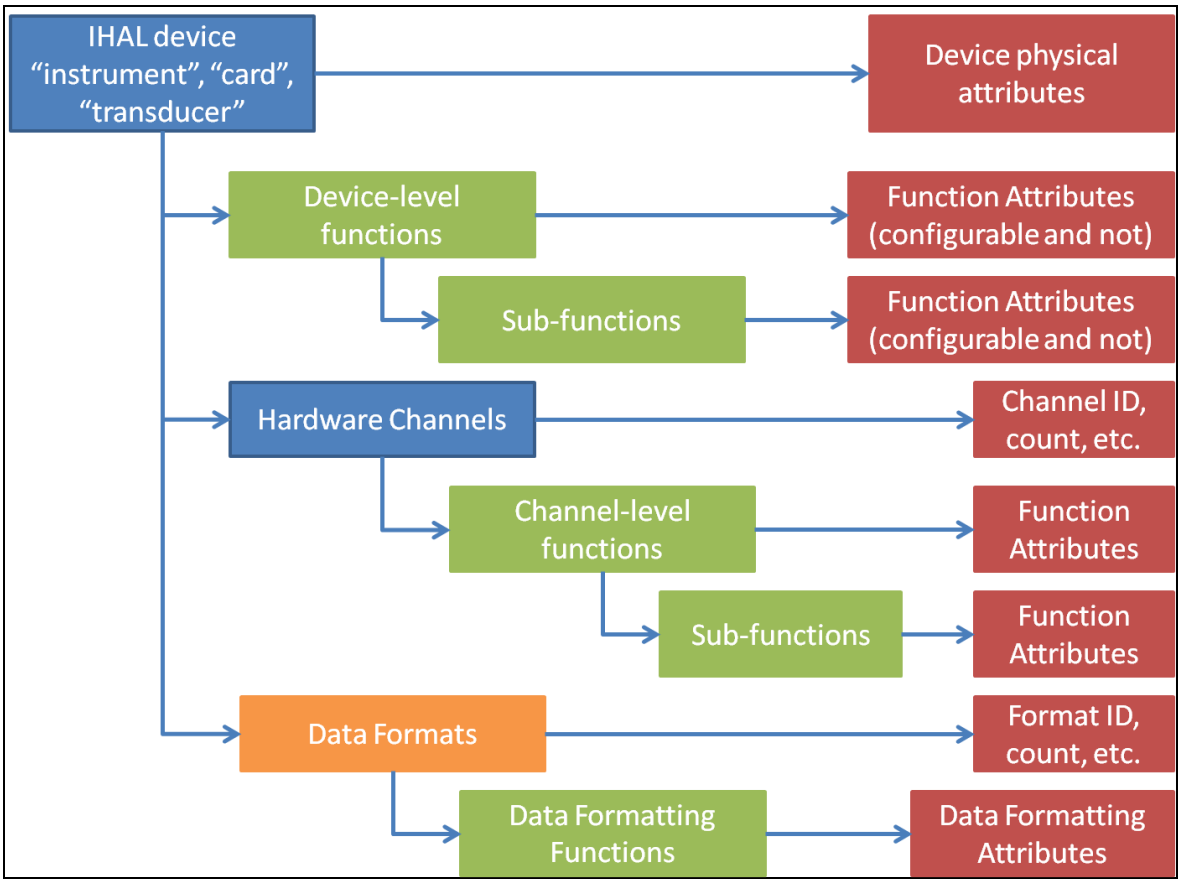

Figure 2: Logical model of an IHAL device.

GENERIC CONSTRUCTS IN IHAL

Much care has gone into the design of the IHAL schema to ensure that there is a generic structure defined for each key concept, and that more specific concepts can “inherit” from these concepts. In addition to simplifying the schema design process, doing this also allows for these generic constructs to be used to describe devices, functions, channels, attributes, or concepts that don’t fit into the more specific descriptions. These constructs are available for defining aspects of instrumentation that are vendor-specific or uncommon. IHAL has constructs for describing generic instruments, cards, functions, and attributes.

IHAL EXAMPLE: DEFINING A SIGNAL CONDITIONING CARD IN THE POOL

In this section we give a simple example pool specification for an analog signal conditioning card. Each device in the IHAL pool has an “id” attribute that can be used to uniquely identify it. This id is how other parts of an IHAL document can reference the device. For signal conditioning cards, all of the functionality (and settings) of the card is defined in the description of the card’s channels.

Like other concepts in IHAL, there is a generic “hardwareChannel” as well as more specific channel types. The analog signal conditioning channel in IHAL extends the generic description

by adding the analog signal conditioning function. Each channel definition must include a value for multiplicity. This value defines how many channels the card has that match the current description. So, if a signal conditioning card has 8 identical channels, the channel can be defined once in IHAL with the multiplicity equal to 8. Each channel description can include a number of elements describing its electrical characteristics and connectors, as well as the functions that channel can perform. For this example, we will look more closely at the signal conditioning function.

Figure 3 shows an example IHAL instance of an <analogSignalConditiongCard> element. In this example, the only card information that has been defined is the general description of the device, which includes its manufacturer and part number (e.g. Model Number). Additionally, the channel definition defines that this card consists of 8 identical channels, where each channel consists only of an analog signal conditioning function. For now, the description of the function has been left blank.

```
<ihalanalogsc:analogSignalConditioningCard ihalcommon:ID="ID_1">
  <ihaldevice:deviceDescription>
    <ihalcommon:name>ACME analog conditioning card</ihalcommon:name>
    <ihaldevice:manufacturerName>ACME</ihaldevice:manufacturerName>
    <ihaldevice:partNumber>ACME-ASC-1234</ihaldevice:partNumber>
  </ihaldevice:deviceDescription>
  <ihalanalogsc:analogSignalConditioningChannel id="ID_13">
    <ihalfunction:description>
      <ihalcommon:name>Analog Channel</ihalcommon:name>
    </ihalfunction:description>
    <ihalfunction:multiplicity>8</ihalfunction:multiplicity>
    <ihalanalogsc:analogSignalConditioningFunction />
  </ihalanalogsc:analogSignalConditioningChannel>
</ihalanalogsc:analogSignalConditioningCard>
```

Figure 3: Signal conditioning card example (pool-level)

The analog signal conditioning function may be composed of a number of attributes (minimum/maximum input signal voltage, offset) and sub-functions (amplification, filter, analog-to-digital conversion, voltage/current excitation). Each attribute may be defined as either fixed or configurable.

An example IHAL description of an analog signal conditioning function is shown in Figure 4. For the sake of brevity, this example has been limited to a single attribute, “offset”, and a single sub-function, “amplification.” The offset attribute is defined as configurable within the range of 0 to 10. Details of the units and the amplification sub-function have been omitted from this example.

The example in the next section will refer to the pool-level concepts defined in this section.

```
<ihalanalogsc:analogSignalConditioningFunction>
  <ihalfunction:functionDescription>
    <ihalcommon:name>analog signal conditioning function example</ihalcommon:name>
  </ihalfunction:functionDescription>
  <ihalanalogsc:analogSignalConditioningFunctionCharacteristics>
    <ihalanalogsc:analogSignalConditioningFunctionAttributes>
      <ihalattribute:offset ihalcommon:ID="offset1">
        <ihalattribute:configurableNumericAttribute>
          <ihalattribute:minimumValue>
            <ihalattribute:value>0</ihalattribute:value>
          </ihalattribute:minimumValue>
          <ihalattribute:maximumValue>
            <ihalattribute:value>10</ihalattribute:value>
          </ihalattribute:maximumValue>
        </ihalattribute:configurableNumericAttribute>
      </ihalattribute:offset>
    </ihalanalogsc:analogSignalConditioningFunctionAttributes>
    <ihalanalogsc:analogSignalConditioningFunctionSubFunctions>
      <ihalamplification:voltageAmplificationFunction />
    </ihalanalogsc:analogSignalConditioningFunctionSubFunctions>
  </ihalanalogsc:analogSignalConditioningFunctionCharacteristics>
</ihalanalogsc:analogSignalConditioningFunction>
```

Figure 4: Example IHAL description of an analog signal conditioning function.

IHAL EXAMPLE: CONFIGURING A CARD AT THE USE-LEVEL

Compared to the pool-level descriptions, IHAL use-level descriptions are much less complicated. Descriptions at the use level simply make references to items in the pool, and set the configurable attributes to specific values. There are also additional use-level concepts for further restricting the valid values for each setting (based on the current configuration), and for defining encoded data formats.

The top-level container for a set of use-level descriptions in IHAL is the <configuration> element. Each <configuration> can then be composed of one or more <instrumentationGraph> elements. An instrumentation graph in IHAL is simply a collection of interconnected hardware.

Each <instrumentationGraph> element is then composed of one or more instrumentUse elements and optional connection elements. Each of these instrument “use” element contains a reference to the pool-level specification of an instrument, some use-specific details such as serial number and location, and details about the current configuration of the device. In this example, we will add a use-level specification for the card we defined in the previous example.

A minimal example of an IHAL configuration specification is shown in Figure 5. This example assigns a unique id to the configuration, and the name “Configuration Example”. It consists of a single instrumentation graph with one <instrumentUse> specification that uses the “Ref”

attribute to refer to the card defined in the pool example. The instrument use specification in this example sets the value of the "offset" attribute on channel 2 of the card to 5.

```
<ihalconfig:configuration ihalcommon:ihalVersion="3.115" ihalcommon:ID="config01">
  <ihalconfig:description>
    <ihalcommon:name>Configuration Example</ihalcommon:name>
  </ihalconfig:description>
  <ihalconfig:dateCreated>2011-05-03</ihalconfig:dateCreated>
  <ihalconfig:dateImplemented>2011-05-04</ihalconfig:dateImplemented>
  <ihalconfig:location>KBSI Corporate Office</ihalconfig:location>
  <ihalconfig:attachedFile>\\MyServer\MyFolder\MyMetadataFile.mdl</ihalconfig:attachedFile>
  <ihalinstgraph:instrumentationGraph ihalcommon:ID="graph1">
    <ihalinstgraph:description>
      <ihalcommon:name>Example Instrumentation Graph 1</ihalcommon:name>
    </ihalinstgraph:description>
    <ihalinstuse:instrumentUse ihalcommon:ID="iUse01" ihalcommon:Ref="ID_1">
      <ihalinstuse:serialNumber>AA000001</ihalinstuse:serialNumber>
      <ihalinstuse:channelUse ihalcommon:ID="channelUse01" ihalcommon:Ref="ID_13">
        <ihalinstuse:channelNumber>2</ihalinstuse:channelNumber>
        <ihalinstuse:description>
          <ihalcommon:name>ACME Channel 2</ihalcommon:name>
        </ihalinstuse:description>
        <ihalinstuse:attributeSettings>
          <ihalinstuse:setAttribute ihalcommon:ID="setAttribute01" ihalcommon:Ref="offset1">
            <ihalinstuse:setConfigurableNumericAttribute>
              <ihalattribute:value>5</ihalattribute:value>
            </ihalinstuse:setConfigurableNumericAttribute>
          </ihalinstuse:setAttribute>
        </ihalinstuse:attributeSettings>
      </ihalinstuse:channelUse>
    </ihalinstuse:instrumentUse>
  </ihalinstgraph:instrumentationGraph>
</ihalconfig:configuration>
```

Figure 5: Simple IHAL <configuration> example

UTILIZATION OF THIRD-PARTY STANDARDS

In order to avoid "reinventing the wheel", the working group decided to make use of existing and emerging third-party standards for certain concepts. Portions of the iNet program's "Metadata Description Language" (MDL) are imported into IHAL for defining units, buses, and measurements. Portions of the Telemetry Attributes Transfer Standard (TMATS) XML schema are imported for defining PCM data streams. Finally, the eXtensible Instrumentation Definition Markup Language (XidML) is used for defining how non-iNet network data is encoded.

THE IHAL API

The IHAL API Specification defines an IHAL-based interface for instrumentation vendors to implement in their configuration engines. The purpose of the IHAL API is to allow vendor-neutral software to interact with multiple vendor-specific configuration engines in the same way as the vendors' own configuration software. Following this approach, such a third-party tool can

submit individual setting changes to the configuration engine and receive immediate feedback as to the impact of each change. This "impact" may consist of the requested setting change, changes to other settings that are affected by this setting change, and changes to the range of valid values for other settings.

This IHAL API specification requires that the API be implemented as a Representation State Transfer (REST) web service. The API specification defines eleven methods that must be implemented by vendors who support IHAL. Each method specification consists of a URL for accessing the method, the HTTP verb to be used when calling the method, and required inputs and outputs. The API specification also provides a mechanism for returning error messages from method calls. The methods defined in the API are listed below:

1. Retrieve the vendor's pool-level descriptions
2. Retrieve a list of configurations currently available in the vendor's system
3. Retrieve the complete IHAL specification of a specific configuration
4. Create a new configuration
5. Modify a configuration by changing a setting
6. Add a device to a configuration
7. Remove a device from a configuration
8. Program all hardware with a specific configuration
9. Add a new format to a data encoder
10. Add a measurement to a data encoder's format
11. Remove a measurement from a data encoder's format

CONCLUSIONS

The existing IHAL language and API provide a vendor-neutral way to configure all hardware settings in an instrumentation system regardless of the vendor or the vendor's product line. Following more than six years of development, IHAL has been validated, reviewed, and prepared for official standardization as part of IRIG 106. In this paper we have described the standardization process, presented example IHAL descriptions, and provided an update to the current API functionality.

REFERENCES

[1] Hamilton, Fernandes, Koola, and Jones, *An Instrumentation Hardware Abstraction Language*, Proc. International Telemetering Conf., Vol. XLII, (2006) Paper 06-10-02, San Diego, CA.

[2] Hamilton, Fernandes, Graul, Darr, and Jones, *Extensions to the Instrumentation Hardware Abstraction Language (IHAL)*, Proc. International Telemetering Conf., Vol. XLIV, (2008) Paper 08-19-04, San Diego, CA.

[3] Hamilton, Darr, Fernandes, Jones, and Sulewski, *IHAL and Web Service Interfaces to Vendor Configuration Engines*, Proc. International Telemetering Conf., Vol. XLVI (2010), Paper 10-08-01, San Diego, CA.