EXTENSIONS TO THE INSTRUMENT HARDWARE ABSTRACTION LANGUAGE (IHAL)

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

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

ABSTRACT

In this paper we describe extensions to the Instrument Hardware Abstraction Language (IHAL). Since IHAL was first presented to ITC in 2006 [1], a number of improvements were made to the design of IHAL. Major changes to the schema include splitting it into multiple XML Schema (XSD) files, separation of the description of instrumentation functions from the description of the hardware, and addition of a function pool.

KEYWORDS

Instrumentation, Transducers, Data Acquisition System, Hardware Abstraction Language, IHAL, XML

INTRODUCTION

At the 2006 International Telemetering Conference, we presented the Instrumentation Hardware Abstraction Language (IHAL), a neutral, XML-based language for specifying instrumentation hardware and instrumentation hardware networks [1]. The IHAL provides a means to describe not only the specific hardware components, but also the usage of those components in a specific network configuration. The language has been designed to serve three roles: (1) as a specification language for describing the hardware and hardware networks, (2) as a command language for configuring each hardware component, and (3) as a query language for polling the hardware components for their current configuration. Key concepts in IHAL include the following:

- An instrument pool for describing all available instrumentation hardware according to its functionality, capabilities, and configurability.
- An instrument use, contained within an instrument network, for describing the way a specific instance of an instrument from the pool is configured and connected in a given network.

- A “first-class” instrument or instrument function which is a commonly-used instrument function that is explicitly named in the schema.
- A generic instrument or instrument function which is used to describe any instrument function that is not explicitly named in the schema.
- Generic parameters, or “params”, which are simple name-value pairs for specifying attributes that are not explicitly contained in the schema.

In this paper, we describe the enhancements that have been made to the IHAL since the original paper in 2006. Since that time, the schema has been split into multiple XSD files for better maintenance and flexibility. Additionally, the description of instrument functions has been separated from the description of the physical hardware, which has enabled a number of other enhancements and simplifications to the schema. Finally, an instrument function pool has been added. Each of these enhancements is described in detail in the remaining sections.

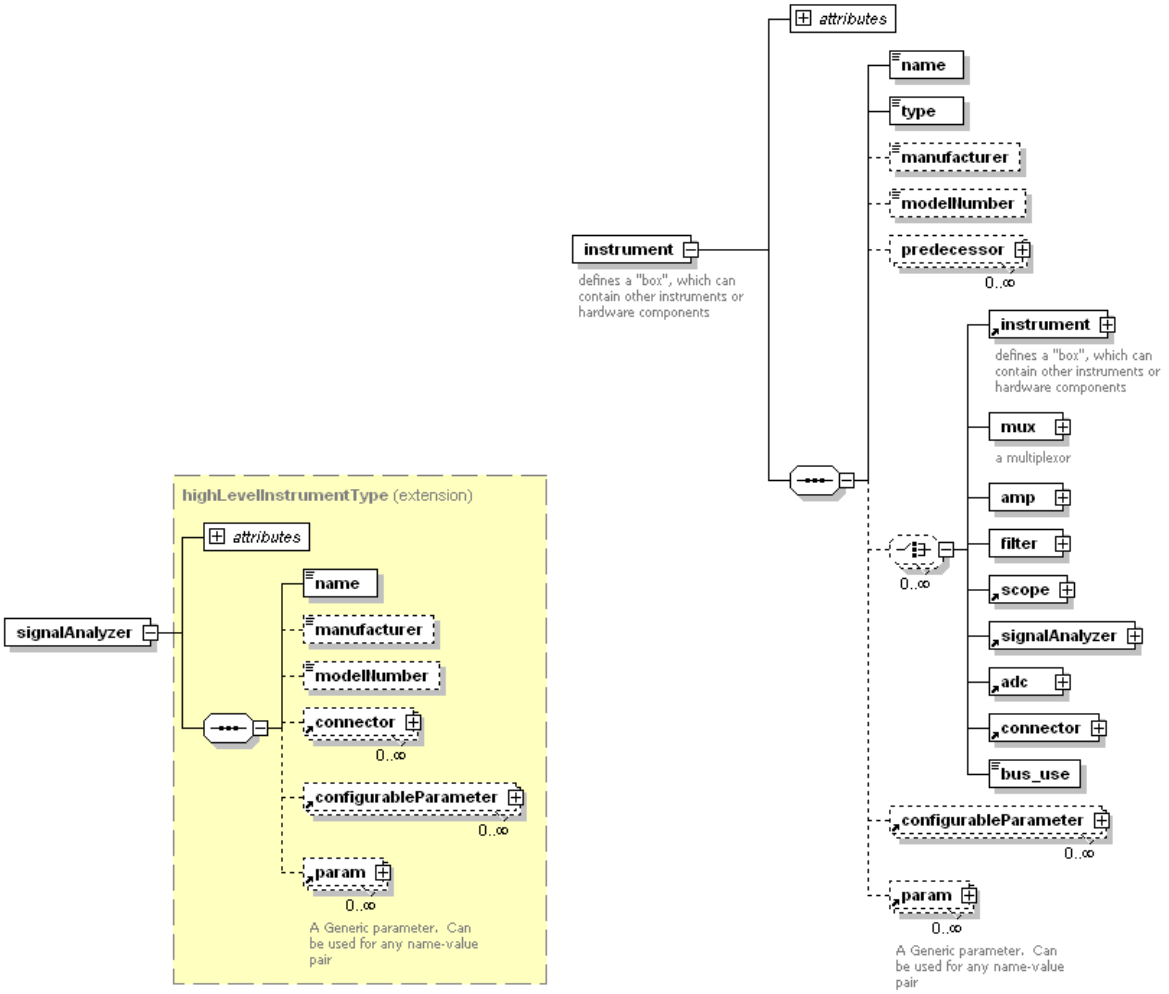
MULTIPLE XSD FILES

Previously, the entire IHAL schema was maintained in a single XML Schema (XSD) file. This approach made the schema difficult to maintain, and required that an IHAL file must contain the full IHAL tree in order to be valid. By splitting the schema into multiple sub-schemas stored in separate XSD files, we have made the language easier to maintain by enabling the designer to only work with the schema file that contains the portion being modified. More importantly, certain sub-structures of the IHAL can be stored alone in an XML file and still be considered a valid IHAL. For example, a description of a single pool-level instrument can be stored in a valid XML file without the extra overhead of storing the <ihal> and <instrument_pool> structures.

SEPARATION OF INSTRUMENT HARDWARE FROM INSTRUMENT FUNCTION

In the previous versions of the IHAL, complex instrumentation functionality was composed in one of two ways: (1) if the instrument performed only the basic functionality provided by a common “first-class” instrument type, it was represented as that instrument type; or (2) if the instrument was uncommon or performed several complex functions, it was represented as a generic “instrument” object and recursively composed of sub-instruments down to the level of simple amps, filters, and multiplexers.

The original IHAL schema of a first-class instrument, in this case the signal analyzer, is shown in Figure 1(a). The original IHAL schema of the generic instrument element is shown in Figure 1(b).



(a) First-Class Instrument **(b) Generic Instrument**

Figure 1: Previous IHAL Schema for a First-class Instrument

It was decided that this approach was lacking in several aspects. For one, the representation of first-class instruments provided no insight into the functions actually performed by that instrument. Also, the language was too restrictive in what could be considered a first-class instrument. For example, if a particular instrument performed some extra functions in addition to the functions commonly performed by a signal conditioner, the instrument could no longer be called a "signalConditioner" in IHAL and would have to be represented as a generic instrument. Finally, representing instruments, as being composed of sub-instruments, does not correctly capture the semantics of actual instrumentation hardware. In reality, there is a single "instrument" (a single piece of hardware) that may be composed of multiple instrumentation functions and sub-functions.

These problems were addressed by separating the concept of an instrument function from the concept of a physical instrument. Under the new design, each physically separate instrument (generic or first-class) corresponds to a single <instrument> element in IHAL. This single

element contains the specification of the system-level properties of that instrument, such as manufacturer, model number, accuracy, and input impedance.

Each instrument is then associated with one or more complex hardware functions, which can be composed of multiple sub-functions and sub-sub-functions, etc. Each first-class hardware function is composed of "primary" hardware functions, which are explicitly enumerated in the schema, and "additional" hardware functions, which are freely specified by the user. Thus, the IHAL schema for the first-class function serves as a model for the types of hardware functions commonly associated with that hardware type. For example, the primary hardware functions associated with the signal conditioner hardware function are filter, amp, and excitation. These three sub-functions are each optional, and other hardware functions can be added as "additional" hardware functions for a given instrument specification.

The basic hardware functions, filter, amp, and voltage excitation, represent the lowest-level hardware functionality. This set of lowest-level hardware functions will grow as the language is matured and more first-class instrument functions are added to the schema. A current schema diagram of the IHAL instrument pool is shown in Figure 2.

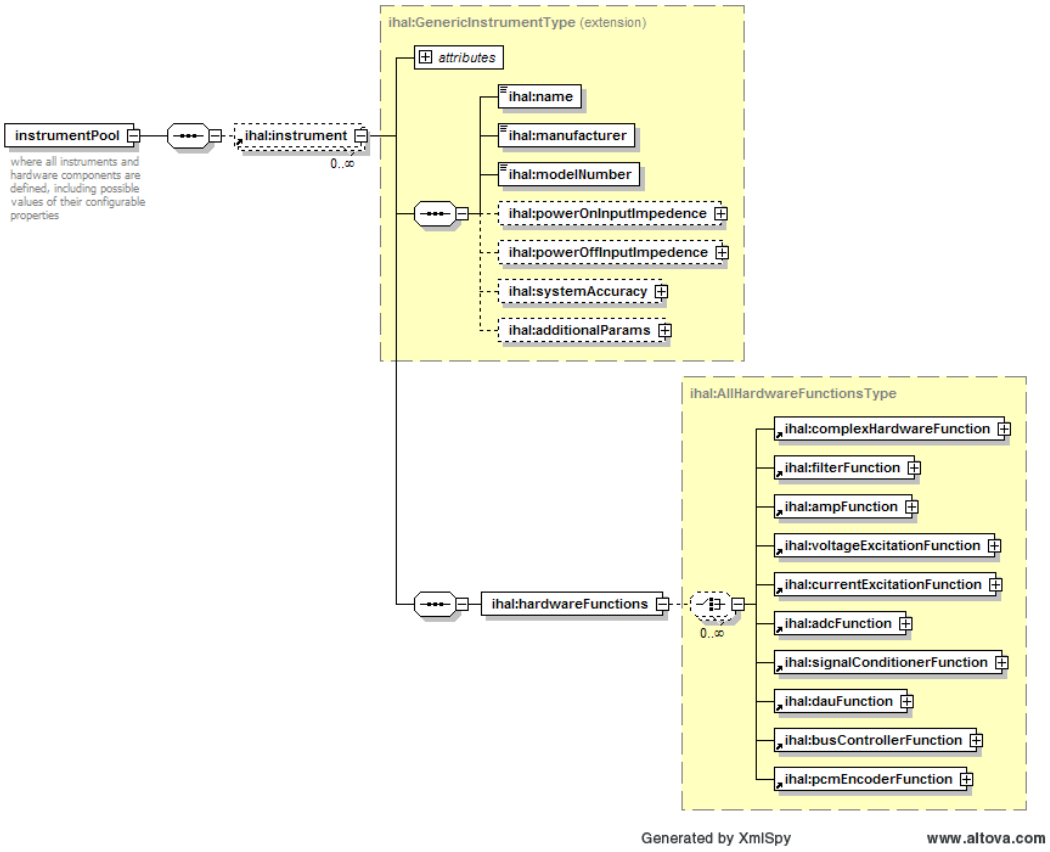


Figure 2: IHAL Instrument Pool Schema

REDUCING SPECIFICATION FOR MULTI-CHANNEL INSTRUMENTS

The previous version of IHAL did not contain explicit support for multichannel instruments (or hardware functions). Instead, a multichannel instrument had to be composed of multiple single-channel instruments, one for each channel. This was a very inefficient representation since it produced many large, identical blocks of IHAL code to represent even simple multi-channel instruments.

This problem was alleviated by allowing a single hardware function to be specified as multi-channel. The new "numberOfChannels" element can be set to one for single-channel hardware functions, or any integer greater than one for multichannel hardware functions. This element can be seen in Figure 3. Specifying a single hardware function, as containing multiple channels, implies that each channel performs identical functions.

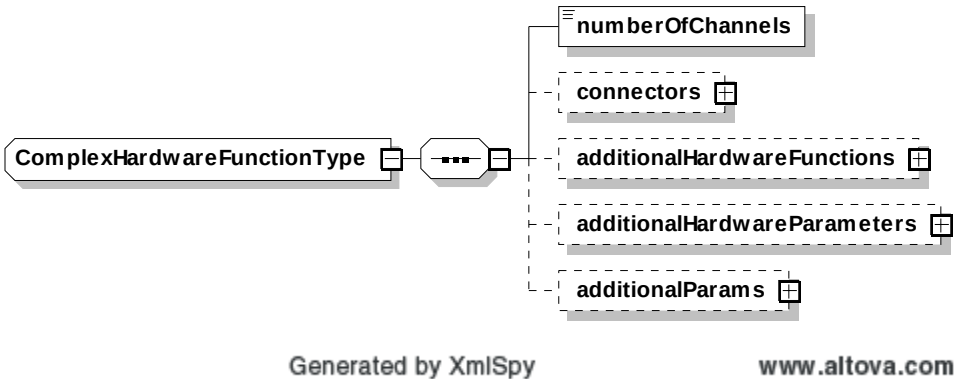


Figure 3: IHAL Complex Hardware Function Type

In order to allow separation of items associated with the multichannel hardware function as a whole and items associated with each channel connectors, sub-functions, and parameters are separated into those which are "systemLevel" and those which are "channelLevel." An example of this can be seen in the details of the "connectors" element, shown in Figure 4. Specifying a connector as system-level means that there is a single connector for the entire multichannel hardware function. Specifying a connector as channel-level means there is one connector for each channel.

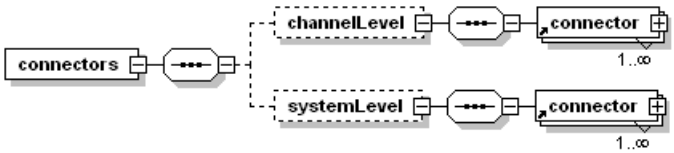


Figure 4: Connectors in IHAL can be either Channel-level or System-level

Since the need to specify each channel individually has been eliminated, it is necessary to add a mechanism for indicating whether a configurable parameter can be configured independently for each channel, or whether there is a single setting that affects all channels. Hence, a Boolean

<configurablePerChannel> element was added to the <configurable> element. The specification of configurable parameters is discussed in more detail below.

COMBINING CONFIGURABLE WITH NON-CONFIGURABLE PARAMETERS

In the previous version of IHAL, each instrument or sub-instrument could have a number of configurable parameters, which define the ways in which the hardware can be configured by the end user. The type of the configurable parameter (gain, cut-off frequency, etc.) was specified as a string value for the "type" element. The configurableParameter element in the previous version of IHAL is shown in Figure 5.

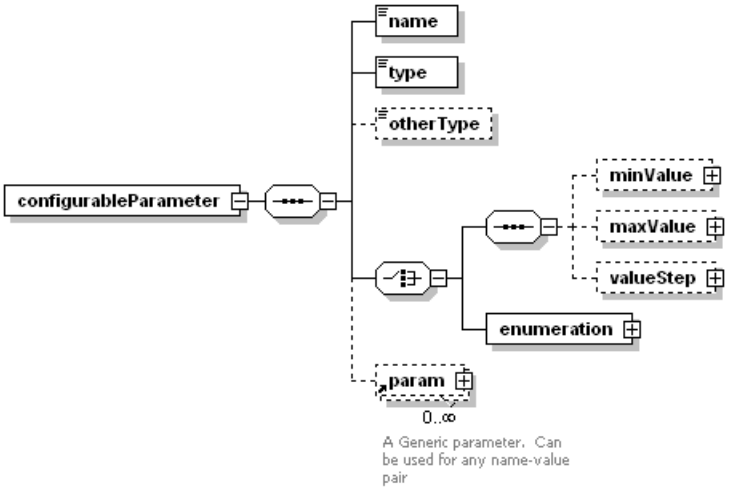


Figure 5: Previous IHAL's <configurableParameter> Element

One problem with this approach is that it provides no indication of how to deal with the situation where a particular parameter is configurable in one instrument, but not configurable in another nearly identical instrument. If it is specified as a configurable parameter in the first instrument but as a generic "param" element in the second instrument, it will be difficult for software to perform a comparison of these two otherwise-identical instruments. Further, the simple "type" element does not provide any indication of what types of parameters are typically available for each of the hardware functions. For example, it is expected that an amp would have a "gain" parameter (which may or may not be configurable) but this is not explicitly captured in the previous IHAL schema.

These problems were dealt with by making the "type" values first-class XML elements, and allowing them to be specified as either "configurable" or "set." Additionally, a "genericHardwareParameter" element was added for those parameters that are not explicitly defined in the IHAL schema. The schema diagram for the explicitly defined "gain" parameter and its parent element "amp" is shown in Figure 6. The schema diagram for the genericHardwareParameter is shown in Figure 7. Finally, the schema diagrams for the "configurable" and "set" parameters are shown in Figure 8(a) and Figure 8(b), respectively.

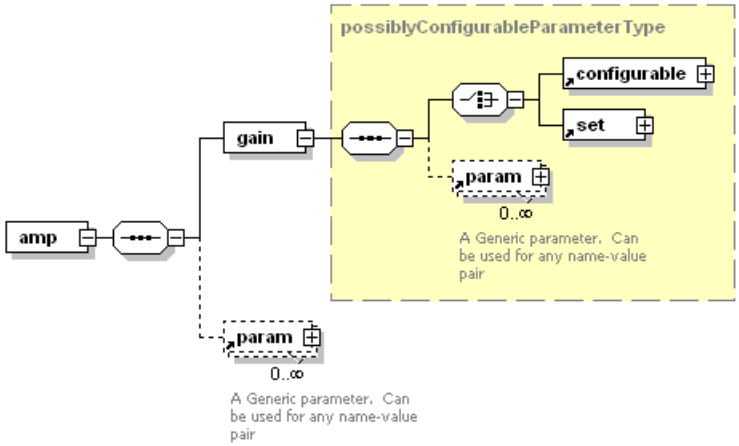


Figure 6: Schema Diagram of the <amp> Element

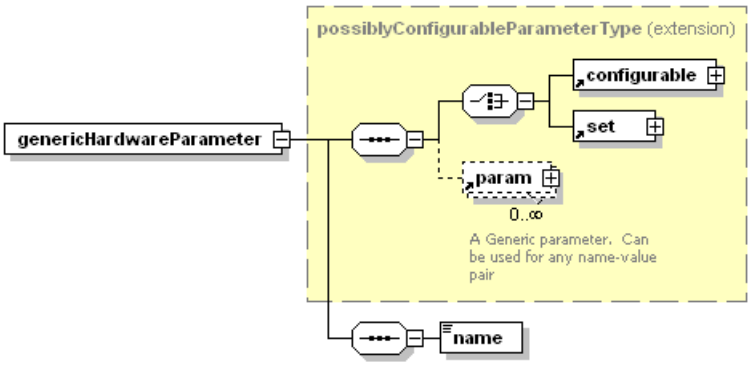
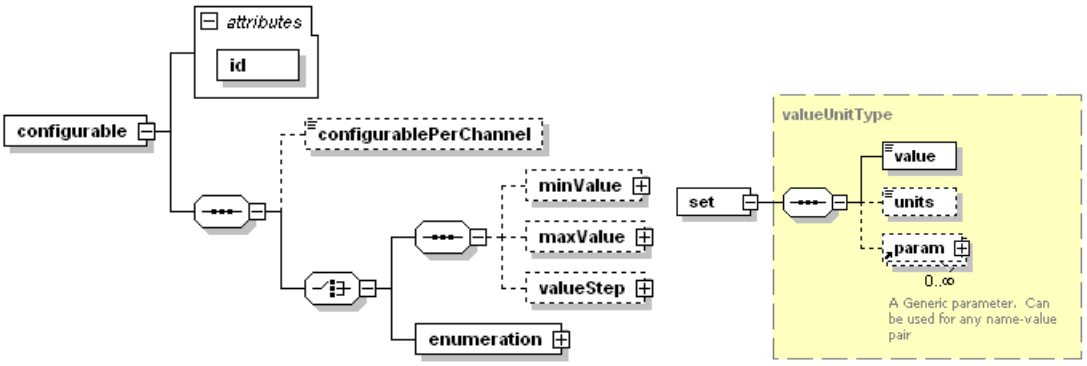


Figure 7: Schema Diagram of the <genericHardwareParameter> Element



(a) IHAL <configurable> Element **(b) IHAL <set> Element**

Figure 8: The IHAL Configurability

SIMPLIFICATION OF USE-LEVEL INSTRUMENTATION SPECIFICATION

In IHAL, an instrumentation network is specified by creating uses of instruments from the pool and defining their current configuration and connections. Previously, the use-level specification of an instrument was identical to the pool-level specification with the exception that the <configurableParameter> elements were replaced with <configuredParameter> elements. The previous <instrument_use> element is shown in Figure 9.

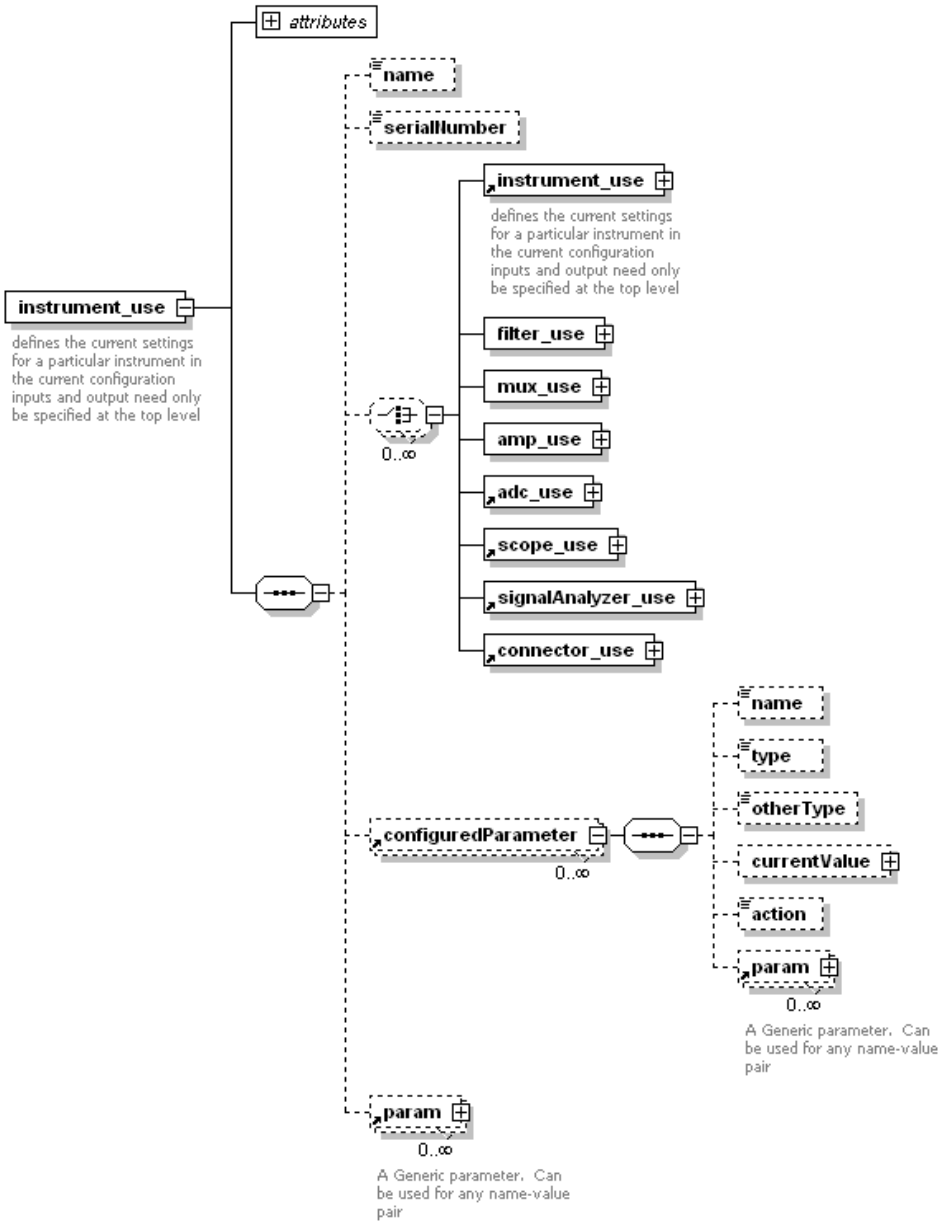


Figure 9: The <instrument_use> Element in the Previous Version of IHAL

This approach caused a lot of information from the pool specification to be duplicated in the use specification. This duplication was eliminated by redesigning the way instruments are specified at the use level. Under the new design, only information that is specific to a given usage of an instrument is put into the use-level instrument specification. This information consists of a name, serial number, location, and the instrument's current settings. The settings consist of a listing of the current values for all configurable parameters. Since the original hierarchy of hardware functions is not duplicated in the use-level specification, we had to require a unique ID be associated with each configurable parameter in the pool. The use specification must then reference these unique IDs in the specification of the current settings. For multichannel hardware functions, the settings may need to specify a channel number as well. The new <instrument_use> element is shown in Figure 10.

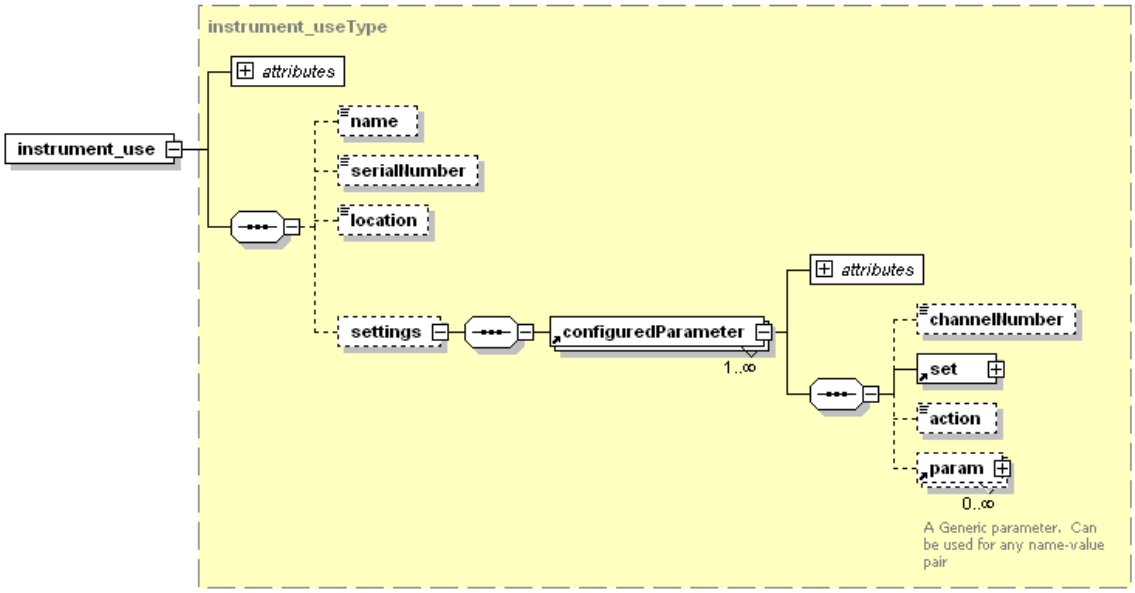


Figure 10: The <instrument_use> Element in the Latest Version of IHAL

ADDITION OF INSTRUMENT FUNCTION POOL

An instrument function pool was added to the IHAL specification to allow frequently used function combinations to be specified once and reused multiple times in the instrument pool. For example, if the combination of signal conditioning and PCM encoding are frequently used together in the instrument pool, these two functions could be combined into a single function in the function pool, and then referenced multiple times in the instrument pool. This results in both the reduction of code and in enabling vendors to define their own hardware functions without losing the underlying functions that compose them. The schema for the instrument function pool is shown in Figure 11.

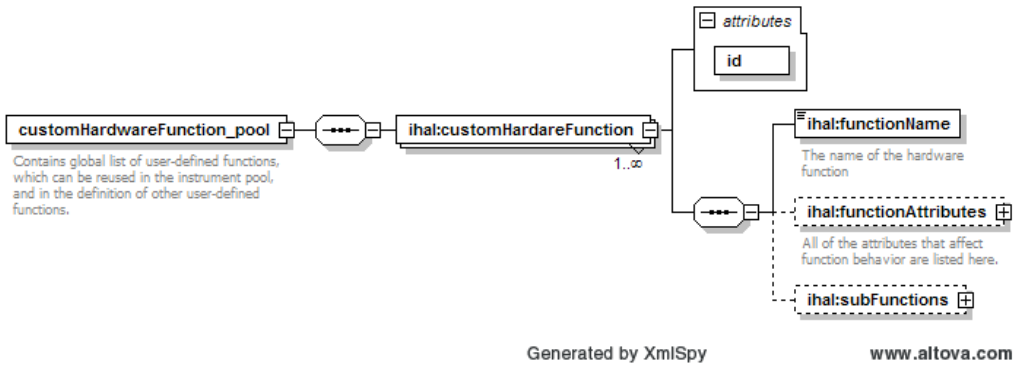


Figure 11: Instrument Function Pool Function

ACKNOWLEDGEMENTS

The IHAL has been developed by Knowledge Base Systems, Inc. as part of a Phase I and Phase II Small Business Innovative Research (SBIR) project funded by Edwards Air Force Base.

REFERENCES

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