XidML – Two Years On

Diarmuid Corry

Acra Control Ltd.

Abstract

In 2004 ACRA CONTROL introduced XidML as a published standard for instrumentation definition via XML. After two years in the field, much feedback and two revisions, this paper outlines where XidML is now, some of the lessons learned and discusses some ideas for where next.

XidML allows any package, message or frame to be defined including PCM, MIL-STD-1553, Ethernet and storage formats, it is also used to define the settings for instrumentation as diverse as Analog to Digital modules, MIL-STD-1553 monitors, PCM encoders, recorders, bit-syncs, and decoms. Importantly it facilitates the EU range and data format to be defined for large parameter lists.

The key elements in XidML are discussed along with some lessons learned.

1 Introduction

Two years ago ACRA CONTROL introduced XidML as a published XML schema for storing and managing the meta-data associated with flight test instrumentation [1].

As of March 1st 2006 engineers from a dozen different companies are working with XidML - developing tools to generate or read XidML files. Many more companies and engineers are using these tools without needing to know that they are operating on XidML files "under the hood".

This paper presents an overview of XidML focussing on the three key elements. It describes some of the ways that XidML has evolved since its introduction through usage and feedback from users.

2 XidML – An Overview

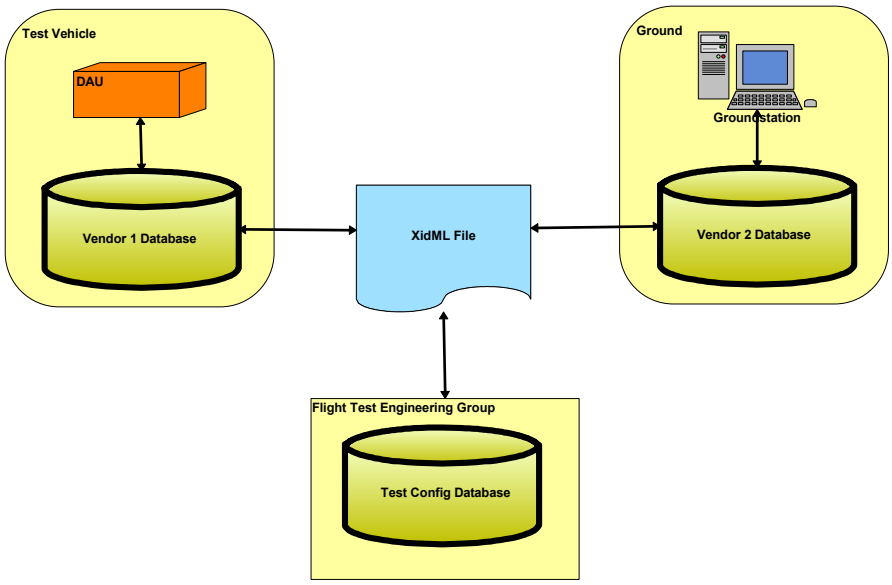

Figure 1: XML acts as a database interchange medium

XML provides a standard means for creating structured documents for automated data interchange. XidML is an XML schema that provides a modular XML framework for expressing all the configuration required in a flight test instrumentation system (Figure 1). Modularity and expandability are key to the design so that people can include sub-schemas for particular instruments or data protocols (called packages in XidML). For example, some programs require the contents of CCDL packet for their Cross Channel Data Link. XidML allows the optional inclusion of this "package" and the instrumentation definition for instrumentation used to monitor that bus.

Key to this modular frame-work is an emphasis on vendor-neutral parameter definition (e.g. sample rates and EU conversion) and vendor-neutral packet definition (e.g. IRIG-106 Ch. 4 PCM stream definition). The intention of the

design is to free the user from worrying about vendor specific details of the hardware or software used in a test, and instead allows them to focus on the parameters and attributes that they care about.

3 Key Elements of XidML

The XidML standard defines seven elements for describing a system. Three of these could be considered “primary” elements because they contain the majority of the information required for data acquisition. The four others are secondary elements that provide extra information for configuring the system or processing the acquired data. The three primary elements are:

Element	Description
Parameters	The parameter set. This element contains a list of all parameters in the system and attributes associated with those parameters. This element also allows parameters to be grouped, and defines parameter classes to store default parameters setup.
Packages	This element contains all the protocols and message descriptions in the system. It stores information about how data is transported through the system, and into and out of the system.
Instrumentation	This element contains information about all the hardware in the system – the instruments that create, transport and manage the data in the system.

In addition to these primary elements there are four secondary elements:

Element	Description
Algorithms	This element stores all the rules for transformation of data – whether that is by application of a mathematical formula, combining of one or more inputs to generate one or more outputs, integrating customer software, look-up tables etc. etc.
Documentation	This element contains documentation information. This element can appear in many locations and is often the child of other elements.
Datalinks	This element contains information about the wires and physical interconnects of the system.
Addendum	This is a “catch-all” element that allows other XML schema to be integrated into a XidML file. Children of the Addendum element do not need to be XidML compliant. This is a useful way of adding customized information.

The original version of XidML discussed two years ago had two extra elements which have been removed after discussion with users. The *AcquisitionCycle* element became redundant with the addition of extra children in the Packages element. The *AuxiliaryFiles* element will eventually become redundant as more parsers support the latest revisions of the XML

standard that introduced an *X-Include* element which essentially does exactly the same thing. [2]

Changes like this are to be expected as XidML matures with use and application by many different engineering groups.

3.1 The Parameter Definition Element

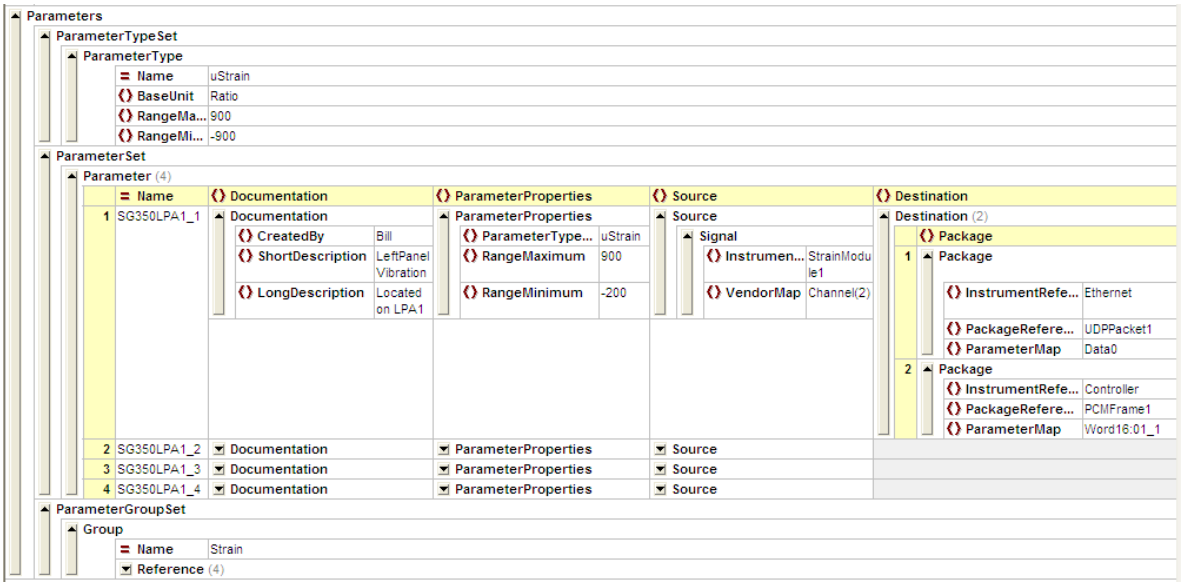

Figure 2: A parameter element as viewed in XMLSpy®

The *Parameters* element allows the meta-data associated with a parameter to be defined. Each parameter has a source (one only) and at least one (although possibly many) destinations. In addition it has certain properties, for example, the format (e.g. offset binary) and size (e.g. 16-bits) and units (e.g. volts) along with the desired minimum sample rate. The actual sample rate can also be stored (which may differ from the minimum because of hardware or package restrictions).

It is important to note that a source or destination for a parameter can be either a physical register on an instrument (such as an ADC on an analog channel), a location in a package (such as a word in a MIL-STD-1553 message) or the input or output of a calculation (such as the result of an EU conversion). XidML provides a common set of properties for a parameter in it's most abstract form. Additional properties specific to a particular application or organization, and not defined by XidML, can also be included through the *Addendum* element. This is consistent with XidML's design goal to be comprehensive, but not prescriptive – it covers a wide range of setup information common to all configurations but does not try to constrain or force the user into a particular way of operation.

The parameter definition exists independently of any instrumentation or data protocols that may support that parameter. In this sense, XidML is "hardware" agnostic. If you want to change from IRIG-106 PCM to TCP/IP for data

transfer during the life of a program, then the changes to the XidML expression of your setup are minimal.

3.2 The Package Definition Element

Although parameters exist independently, they usually need to be transferred from place to place in the system. Common protocols today for moving data around include IRIG-106 PCM, MIL-STD-1553 and IRIG Chapter 10 log files. In the future these will be more network oriented (e.g. iNET). XidML refers to data protocols as packages and defines a set of sub-schema to describe particular package structures.

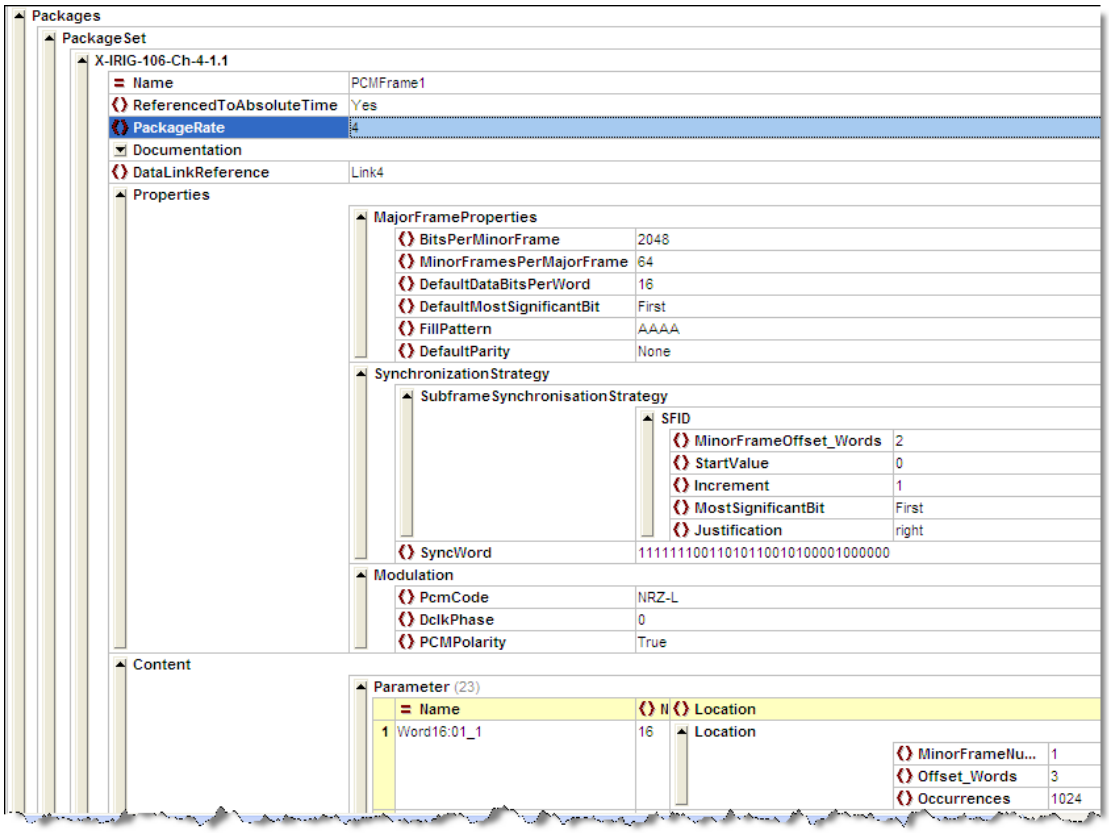

Figure 3: PCM Frame described in XidML as viewed in XMLSpy

Figure 3 shows a partial description of an IRIG-106 PCM frame as it would be described in XidML elements. XidML supports the following common protocols:

Name	Description
X-CAIS	Defines data on a CAIS bus
X-CCDL	Describes data in a CCDL interface
X-Firewire	Describes data on an IEEE1394 bus
X-IENA-Ethernet-UDP-Basic	Describes Ethernet data when formatted in IENA compatible UDP packets
X-IRIG-106-Ch-4	Describes any IRIG-106 PCM frame
X-IRIG-Time	Describes any IRIG Time code message
X-MIL-STD-1553-Message	Describes any MIL-STD-1553 message

Name	Description
X-ARINC-429-Message	Describes any ARINC 429 message
X-Memory-Storage	Describes how data is stored in a solid state memory device
X-Snarfer	Describes any FIFO of data used for all-bus monitoring
X-RS-232-Basic	Describes any serial stream
Package	Vendor/User specific packages

Users are encouraged to add additional protocols for their application if the existing schema is not enough for their needs. XidML is designed to embrace expansion as new protocols become used in the industry.

An important feature of packets is that they can be re-used, even within the same program. For example, a particular PCM frame can be defined just once, but instantiated several times within a configuration. The *PackageInstance* element in the source or destination elements of a parameter allow a particular instance of a frame to be referenced independently. Re-use is an important design consideration for XidML. The goal is to eliminate redundancy in the configuration description and to support the use of libraries of common elements such as PCM frames.

3.3 The Instrument Definition Element

All data is acquired and processed by hardware and software. The Instrumentation element is intended to describe these acquisition and processing nodes.

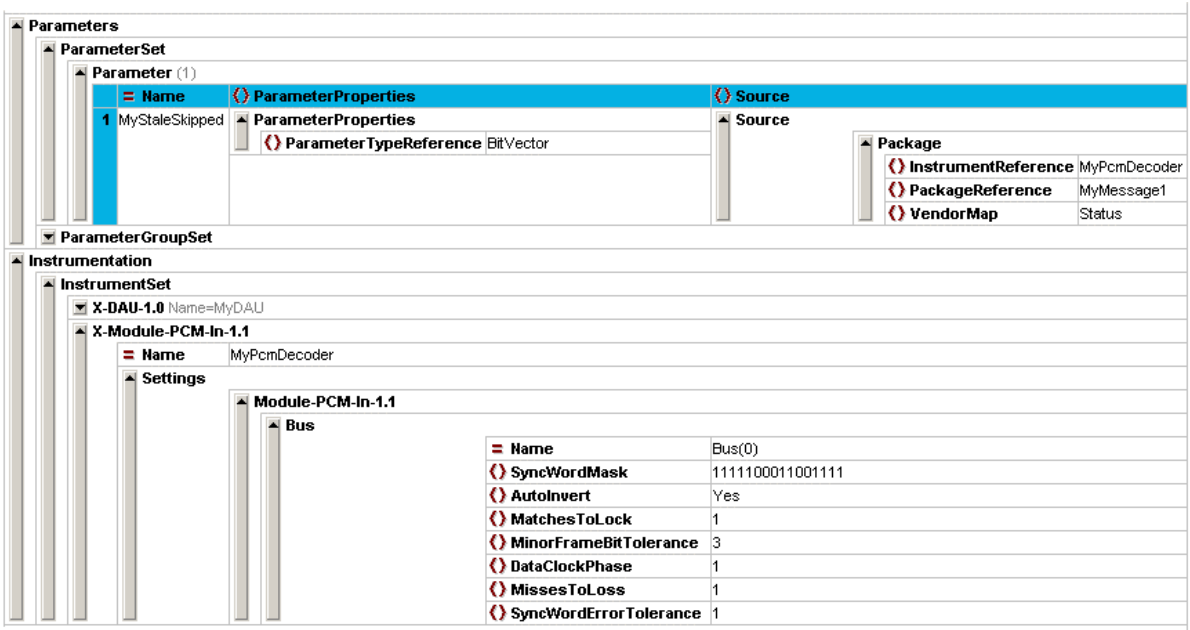

Figure 4: PCM decoder in XidML as viewed in XMLSpy

Figure 4 shows a definition for a PCM decoder. In XidML, a lot of effort has gone in to making the definition vendor neutral. Most PCM decoders should be able to work from the elements defined in the schema. In the event of extra

information being the *Addendum* element can be used, or a new sub-schema defined for the instrument.

Like packages, XidML provides a set of sub-schema to define particular instruments.

Name	Description
X-Module-Analog-In	Any instrument that accepts analog inputs
X-Module-Analog-Out	Any instrument that outputs analog
X-Module-Audio-In	Any instrument that accepts audio input
X-Module-Discrete-In	Any digital input instrument
X-Module-ExternalMultiplexer	Any instrument that interfaces to a multiplexer (e.g. a pressure scanner)
X-Module-Video-In	Any instrument that accepts video input
X-Module-Arinc-573-Monitor	Any instrument that can monitor ARINC-573 data
X-Module-Arinc-573-Transmitter	Any instrument that can act as an ARINC-573 transmitter
X-Module-Arinc-429-Monitor	Any instrument that can monitor ARINC-429 data
X-Module-Arinc-429-Transmitter	Any instrument that can act as an ARINC-429 transmitter
X-Module-1553-Monitor	Any instrument that can monitor 1553 data
X-Module-1553-RT	Any instrument that can act as a 1553 remote terminal
X-Module-CCDL-Monitor	Any instrument that monitors CCDL data
X-Module-Ethernet	Any instrument that can output Ethernet
X-Module-Firewire-Monitor	Any instrument that can monitor firewire (IEEE-1394)
X-Module-PCM-In	Any instrument that accepts PCM input
X-Module-PCM-Out	Any instrument that generates PCM out
X-Module-RS-232	Any instrument that can accept RS-232 type input
X-Module-IRIGTime-InOut	Any instrument that can recognize or generate IRIG time

The schema for common instruments are designed to be generic. For example, *X-Module-Analog-In* describes an analog data acquisition module. However, the schema does not specify vendor dependent restrictions such as the number of channels in the instrument or whether these channels have excitation or not. The schema supports an infinite number of channels on the instrument, with optional excitation. The tools provided by the vendor to configure their hardware must check that the settings are applicable to their hardware.

New schema can be added to support hardware that is sufficiently different that it cannot be supported by the existing schema.

4 Additional Elements

4.1 Algorithms

The algorithms element allows XidML file to store information relating to processing applied to the data. XidML directly supports schema for common processing elements such as alarms, triggers, concatenation, polynomials, software DLLs etc. It also supports any mathematical expression through the use of MathML [3]. Through the algorithms element, any processing that the configuration applies can be stored and accessed by the instruments that do the processing.

4.2 Documentation

The documentation element is used throughout XidML to append information to an element that explains the who, what and why of that element.

4.3 Data-links

The data links element describes the physical layer elements of links within the system (e.g. is that PCM link Rs-422 or TTL?). XidML supports several common physical layer link types.

Name	Description
X-1553-Datalink	MIL-STD-1553 bus (max response time)
X-ARINC-429-Datalink	ARINC-429 bus (bit rate)
X-ARINC-573-Datalink	ARINC-573 bus (bit rate)
X-CAIS-DataLink	CAIS (bits per word)
X-Ethernet-UDP-DataLink	Ethernet type
X-Memory-Storage-DataLink	Memory storage capacity and type
X-PCM-DataLink	PCM (electrical levels)
X-RS-232-DataLink	Serial bus – protocol details

Data links may be expanded in the future to provide information on wiring – allowing XidML to represent the wiring diagram for the configuration in addition to everything else.

4.4 Addendum

The *Addendum* element is a ‘catch-all” element which acknowledges the reality that XidML cannot cover everything. Addendum allows the inclusion of any valid XML data but XidML does not dictate what schema that data should follow. This hook allows vendor specific information to be included (for example) or allows a sensor database to be included via SensorML [4].

4.5 Auxiliary files

XidML originally specified an element called *AuxiliaryFiles* which allowed the inclusion of other XidML files. This was intended to get around a shortcoming in the XML standard which did not allow for embedding references to other

files in an XML file. The *AuxiliaryFiles* element was intended to permit the use of libraries of XidML files.

In December 2004 the X-include directive became a full recommendation. This negates the need for *AuxiliaryFiles* and this element will become redundant when X-Include is widely supported in parsers. .

5 Using XidML

XidML was designed from the bottom up to be flexible. The concept is that the user can choose to define as much or as little of their project as is appropriate for their needs.

For example, consider a bench test system where the user just wants to view raw data in quicklook and verify that their acquisition system is working. In this case the XidML file needs only three elements – *Parameters* to describe the parameters, *Instrumentation* to describe the data acquisition unit and *Packages* to describe the PCM frame (Figure 5).

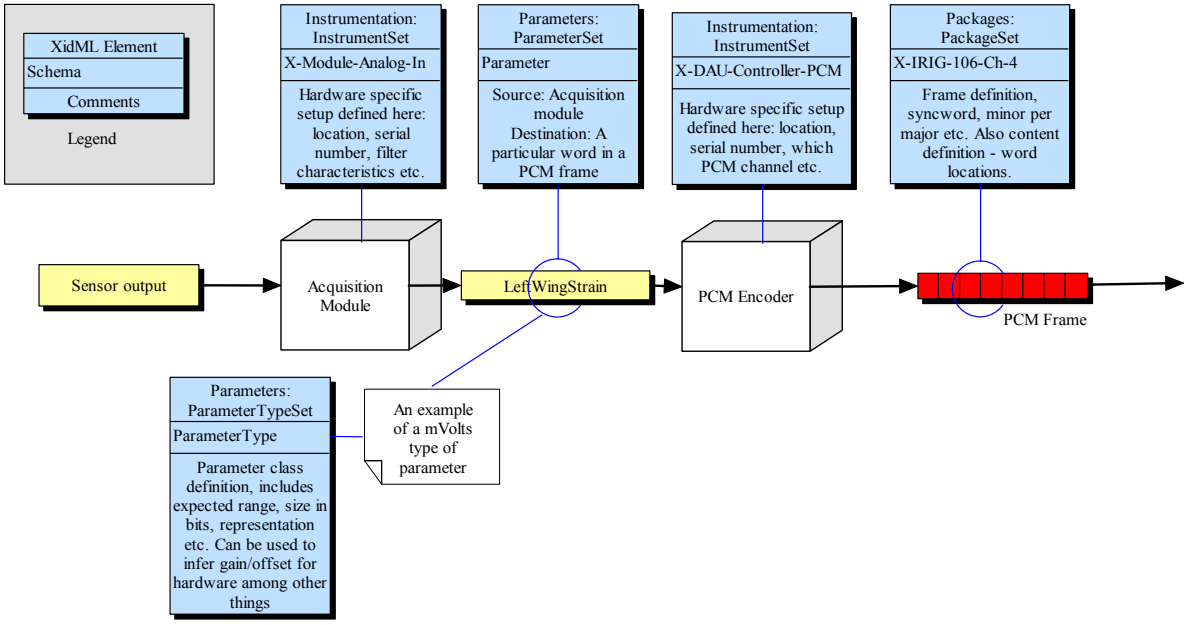

Figure 5: Minimal configuration in XidML

There is no need to redefine EU conversion or ground-station hardware if used with a ground station software packages that import XidML files directly.

At the other end of the scale, consider the complete data path for a parameter.

Every element in this path can be described using XidML (Figure 6). The choice of where to apply XidML is up to the user and engineers working on the application. It is not prescriptive, and is not intended to force a particular modus operandus into an organization.

XidML has been used in different ways:

1. One company uses XidML simply as a transfer file to import the PCM frame setup of the hardware into their groundstation software. Because XidML is XML, and XML comes with many tools, the integration work for this task took two engineers less than two days.
2. Another company is already well into a program to use XML as a common data interchange medium for all aspects of their flight program management, from inventory control through to data post-processing and archiving. They are adapting XidML into the flight test instrumentation as it fits neatly with their existing and proposed systems. Because XidML is already defined there is a negligible effort involved in integrating it, although of course they will spend several years on the overall XML conversion program.

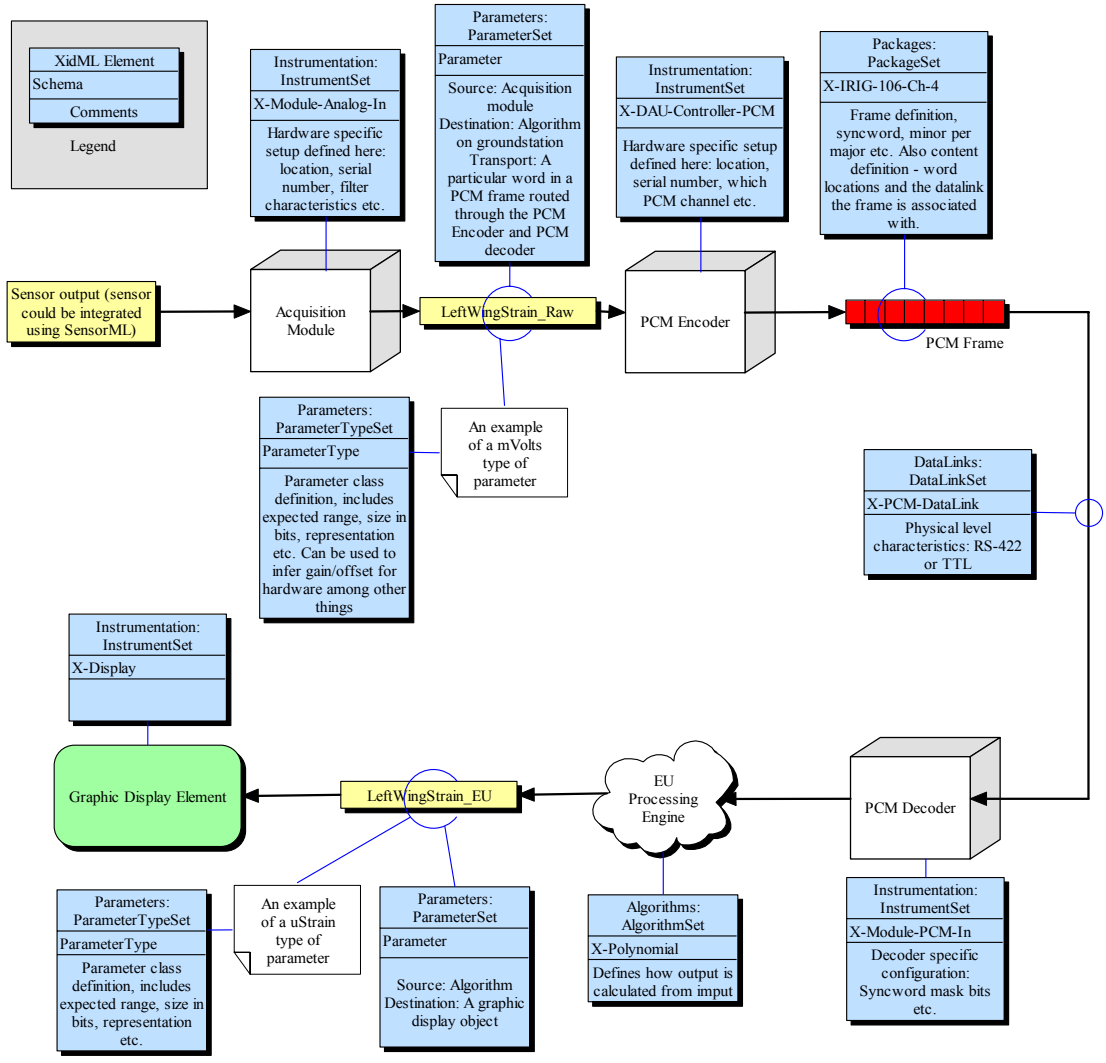

Figure 6: Complete data path in XidML

6 Conclusion

XidML is a mature, published XML schema for flight test instrumentation. It has been in use for over two years and through experience with users has evolved into a comprehensive language for describing test configurations. It is designed to be flexible enough to support small minimal configuration as well as full scale installations with multiple programs and systems. It si designed for both PCM and network based architectures. Experience has shown that XidML can be easily incorporated into existing processes and provides a viable solution for long term characterization of test setup configuration.

Further information on XidML is available at the XidML website: **www.xidml.org**. A discussion on XML is available at **www.xidml.org\whyxml.ppt**. An introduction to XidML is available at **www.xidml.org\xidmlanoverview.ppt**.

References

[1] "XML: A Global Standard for the Flight Test Community", Alan Cooke and Diarmuid Corry, *ETTC Proceedings 2004*

[2] The *X-Include* directive recommendation is at http://www.w3.org/TR/2004/REC-xinclude-20041220/

[3] An introduction to MathML is available at http://www.dessci.com/en/support/tutorials/mathml/default.htm

[4] An introduction to SensorML is available at http://vast.nsstc.uah.edu/SensorML/