INTRODUCTION TO XIDML 3.0 AN OPEN XML STANDARD FOR FLIGHT TEST INSTRUMENATION DESCRIPTION

Alan Cooke, ACRA CONTROL
Christian Herbepin, Eurocopter

Abstract: A few years ago XidML was introduced as an open XML standard for capturing the meta-data associated with flight test instrumentation (FTI). This meta-data schema was broken down into elements for Parameter (name, range, units, offset-binary), Instrument (name, serial number, misses-to loss), Package (bits per word, words per minor-frame, rate) and Link (name, type) and so on. XidML remains one of the only published schema for FTI meta-data and with XidML 3.0 many simplifications have been introduced along with support for nested tree structures and a single instrument schema allowing anyone to define the validation for instruments from any vendor. This paper introduces the XidML schema and describers the benefits of XidML 3.0 in particular. It begins by giving a brief description of what XidML is and describes its history and motivation. The paper then outlines the main differences between XidML-3.0 and earlier versions, and how the XidML schema has been further refined to meet the challenges faced by the FTI community. As an example of usage the FTIManager software developed at Eurocopter will be briefly presented in order to illustrate the XidML ability to describe a multi-vendor FTI configuration

Keywords: XML, XidML, Meta-data, Meta-data validation, Fight Test Instrumentation,

1 WHAT IS XIDML?

XidML the Extensible Instrumentation Meta-data Mark-up Language is an XML schema [6] first released in 2005. It is an open, vendor-neutral meta-data standard that is designed to meet the needs of the FTI community [1] [2] [3] [4]. The XidML schema encompasses everything in the FTI domain from sensors on aircraft, to data displays on a computer screen, and everything in between.

The XidML schema has been designed to be future proof with extensibility inherent in the schemas structure. For example, in addition to supporting traditional data acquisition protocols such IRIG-106 Chapter 4 PCM frames, MIL-STD1553, and so on, it also supports newer networked-based protocols. Any new protocols that emerge in the future can also be easily accommodated in the schema without affecting the basic schema structure.

The data model behind the XidML schema is built around the five key concepts of instruments, parameters, packages, links and algorithms. In the XidML schema, these concepts are mapped directly to the five key elements of instruments, parameters, packages, links and algorithms.

2 THE FIVE MAIN COMPONENTS IN XIDML

2.1 INSTRUMENTS

Instruments are the physical hardware used in data acquisition. They describe how FTI devices such as data acquisition modules and sensors are configured. Examples of the meta-data associated with an instrument include, misses-to-loss, filter cut-off, excitation voltage and so on. All instruments have a globally unique name.

2.2 PARAMETERS

Parameters describe all there is to know about sampled data. This information can include, the number of bits used to encode the data, value ranges, whether the data is encoded as Offset Binary or Binary Coded Decimal etc., measurement units and so on. All parameters have a globally unique name.

2.3 PACKAGES

Packages describe how data is either packaged for transmission, or for storage on a physical medium. Examples of transmission packages would be IRIG-106 Chapter 4 PCM frame definitions, MIL-STD-1553 message definitions and IRIG Chapter 10 storage descriptions. All packages have a globally unique name.

2.4 LINKS

Links describe the physical connections between two instruments. Examples would be an Ethernet connection between two networked devices or an RF link between an aircraft and a ground-station card in a PC. All links have a globally unique name.

2.5 ALGORITHMS

Algorithms describe how data is processed. Examples include a polynomial algorithm used to linearize data or an algorithm used to extract the twelve most significant bits of a sixteen bit parameter before being transmitted in an IRIG-106 Chapter 4 PCM frame. All algorithms have a globally unique name.

3 MAIN CHANGES TO XIDML 3.0

XidML 3.0 is the latest version of the XidML schema. It incorporates the lessons learned from five years of continuous use on scores of flight test projects. In particular, the number of schemas has been reduced and the flexibility of those remaining has been increased. The following outlines the key changes to the schema.

3.1 NESTING OF INSTRUMENTS

In earlier versions of the XidML schema, instruments were classified as either platform instruments or standard instruments. Platform instruments could contain standard instruments while standard instruments could not. For example, a DAU was typically defined using a platform instrument while function specific child modules were typically

defined using standard instruments. It was not possible to nest platform instruments in other platform instruments. In version 3.0 there is no longer a distinction made between platform and standard instruments and it is now possible to nest instruments to any arbitrary depth. This allows the XidML file to more naturally reflect the system under test.

3.2 A SINGLE INSTRUMENT SCHEMA

The biggest change to the XidML schema is the replacement of the 40 or so instrument schemas that existed in earlier versions with one single instrument schema. Specifically, instrument settings, those values that affect the behaviour of an FTI device, are now specified using name-value pairs. This means that individual FTI vendors no longer need to wait for changes to particular instrument schemas to incorporate a custom setting that they need to configure their hardware. Also, all inputs or outputs from a device are defined using channels. Example input channels would be the signal from a sensor or an RS-422 data stream. An example output channel would be an IRIG-106 Chapter 4 PCM frame.

```
<Instrument Name="MyPlane">
  ..
  <Instrument Name="MyCabin">
    ..
    <Instrument Name="MyDAU">
      ..
      <Instument Name="MyModule">
        ..
          <Instument Name="MyDaughterBoard"
```

Figure 1: An example of nested instruments

3.3 XDEFML

In earlier versions of the XidML schema a lot of the logic required for validation of user data was incorporated into the individual instrument schemas. However, in version 3.0, there is only one generic instrument schema and only generic validation is now provided. To address this issue, a new complementary optional schema called XdefML was created for vendors. This optional schema allows vendors to fully specify all validation criteria and data input constraints for their equipment. For example, it may specify the allowed ranges for specific hardware settings, or the number of channels on an instrument. Vendors can supply an XdefML instance file for each class of device they sell. Some FTI vendors, such as ACRA Control, and users of FTI equipment also use XdefML files in order to auto-generate user interfaces in software applications. As an example, see below for a description of the FTI Manager software produced by Eurocopter France.

3.4 INSTANTIATION AND REFERENCING OF COMPONENTS

It is now possible to either instantiate or reference a parameter, package, link or algorithm from anywhere within the XidML file. This allows users, for example, to define a parameter under a channel on an instrument or within the content section of an IRIG-106 PCM frame. The option of defining parameters and so on in a global list,

however, is still supported and is preferred by a lot of FTI users that share system definition across groups and thus incrementally define XidML files.

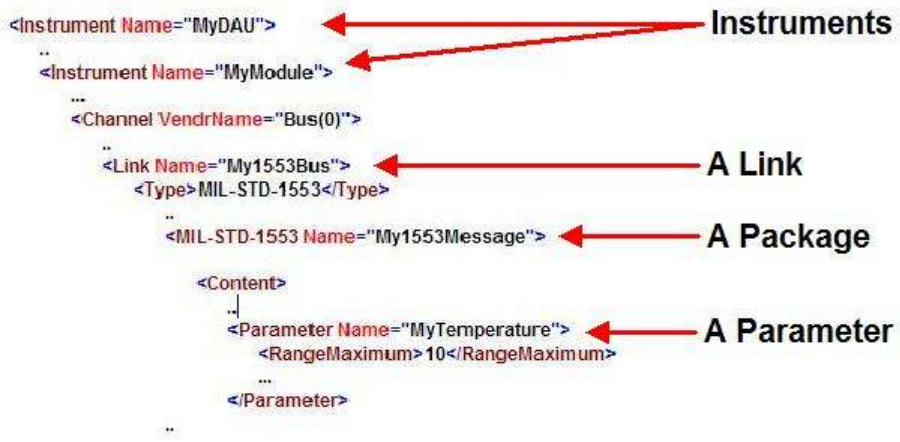

Figure 2: An example of parameters and packages defined in the XML tree

3.5 SIMPLIFIED PARAMETER SCHEMA

In earlier versions of XidML it was necessary to specify the source and destination(s) of a parameter in the parameter definition itself. These have been removed from the parameter schema in XidML 3.0. The source and destination(s) of a parameter can now be inferred from where it is defined in the XidML file. Balance and shunt settings have also been removed from the Parameter element. These are now defined as settings on the channel(s) of an instrument.

3.6 PARAMETERS WITHIN PARAMETERS

XidML 3.0 allows parameters to be defined within parameters. For example, a single IRIG-106 time parameter could have high, low and micro time parameters contained within it. Now, depending on how the instrumentation from a particular FTI vendor behaves, parameters that are contained within other parameters may also be treated as stand alone entities in a XidML file.

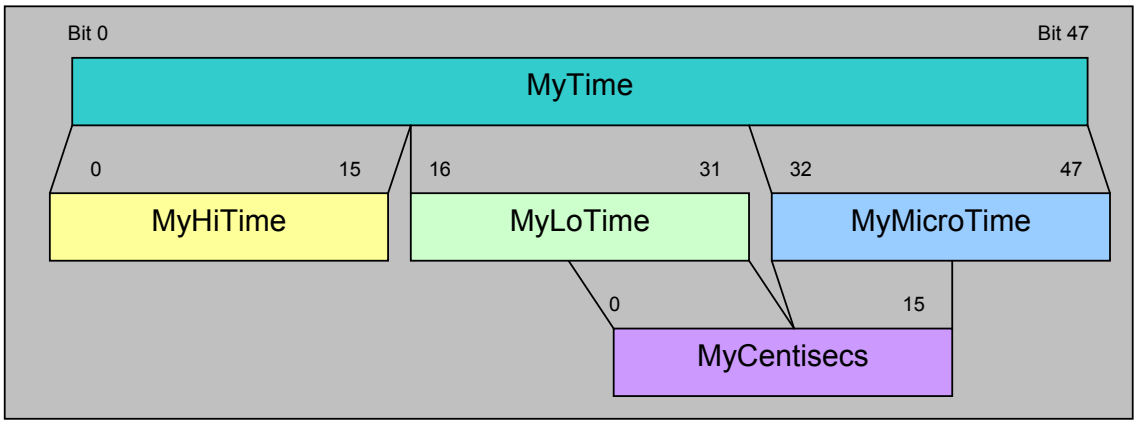

Figure 3: An example of parameters contained within other parameters.

3.7 SIMPLIFIED LINK SCHEMA

The link schema has been greatly simplified. Most settings that were in the link schema in earlier versions of XidML now appear as settings on instrument channels. Settings that are common to both sides of a link can be indicated using XdefML. In most cases it is only necessary to specify the link's type (e.g. Ethernet, MIL-STD-1553 and so on) but it is also possible to specify how data is transmitted over a link (e.g. describe the messages on a MIL-STD-1553 bus).

3.8 AUXILIARY FILE REUSE

It is now possible to reuse entire bus definitions using auxiliary files. XidML file authors achieve this by using standard XPath [5] syntax to either add a prefix or suffix to the names of XidML components or to replace specific values. A practical example of this would be two engines on an aircraft having the same ARINC-429 bus defintion.

4 A CASE STUDY IN USING XIDML & XDEFML: FTI MANAGER

4.1 WHAT IS FTI MANAGER?

FTIManager [7] is an integrated and modular tool, developed by Eurocopter in Java for defining the setup and configuration of FTI hardware. It is also used to track the history of individual items of hardware and calibration information, and acts as a hardware inventory for all equipment used in flight test projects.

The FTI Manager under development uses the latest XidML schema to store the FTI configuration and equipment setup and not only for the ACRA Control acquisition system but also for the other components such as sensors, recorders, remote control units, telemetry and video systems

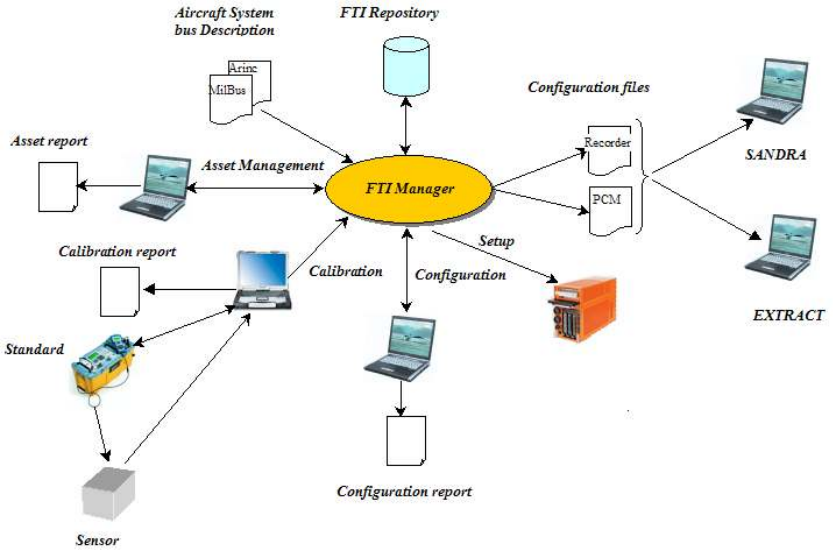

Figure 4: FTI Manager overview

4.2 AUTO-GENERATION OF USER INTERFACES

In order to minimize the effort required to adopt a new hardware, FTI Manager uses XdefML to auto-generate setup screens and the definition of storage formats. It does this by reading the XdefML associated with specific devices and generating a user interface based on the restrictions and constraints that apply to that device.

For example, an XdefML file may specify that a particular channel setting value must be within a particular range and be an integer value. In this case, the generated UI would prevent the user from entering in values outside the specified range and only allow integer values to be entered.

Simply by using an XdefML file it is to support new equipment from multiple vendors without recompiling a line of Java code

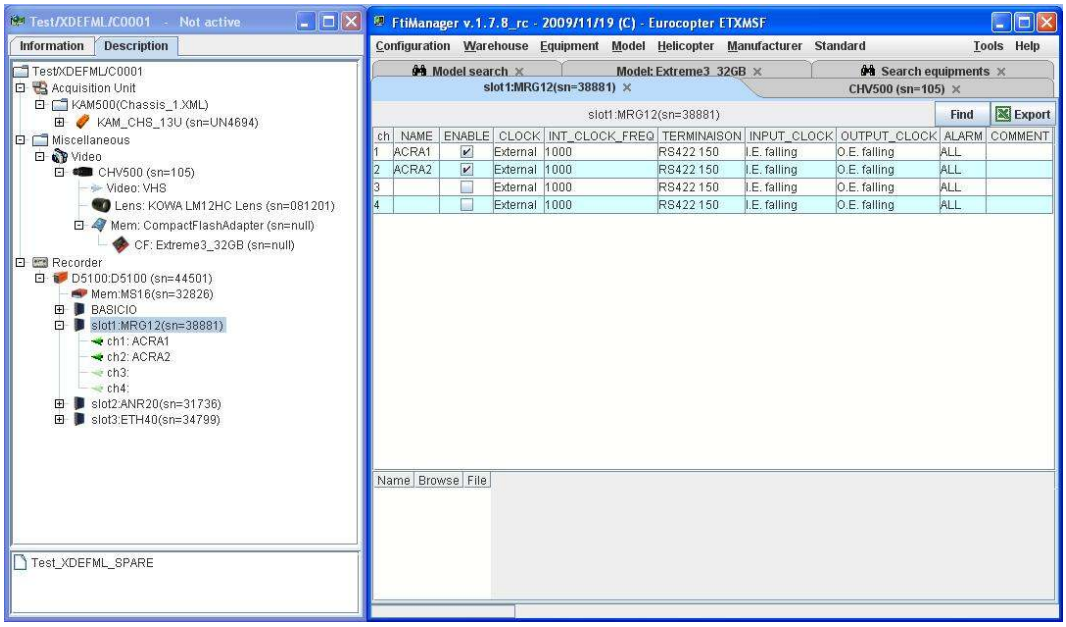

Figure 5: A screen shot some screens generated using XdefML

5 CONCLUSIONS

This paper outlined the benefits of XidML and XidML-3.0 in particular. It described the origins of the XidML schema and how it was developed to meet the needs of the FTI community. In particular, XidML has been designed to be future proof, extensible and vendor neutral. The paper then described the key changes to XidML 3.0. These changes incorporate the many years of experience of users across many FTI projects. In simple terms, the new XidML schema has been simplified greatly and many sub-schemas have been removed. XidML 3.0 also sees the introduction of a complementary optional schema called XdefML that provides a powerful mechanism for vendors to completely specify and validate meta-data for their equipment. Finally, a practical example in the benefits of XidML and XdefML was given with the description of the FTI Manager application from Eurocopter.