

iNET BASED AUTOMATIC HARDWARE SELECTION

Benjamin Kupferschmidt
Technical Manager – TTCWare
Teletronics Technology Corporation

Abstract

One of the principle goals of the Integrated Network Enhanced Telemetry (iNET) program is to help flight-test engineers configure their data acquisition systems more rapidly. This will allow them to focus more of their energies on the collection of data rather than on the design and configuration of their data acquisition hardware. Currently most flight-test engineers spend the majority of their time configuring their data acquisition systems to acquire data for test flights. Typically, the flight test engineers must manually transform the requirements that are given to them into actual measurements from the data acquisition system. This process forces the flight test engineers to become experts in the implementation details of their data acquisition systems.

This paper will discuss a possible design for an automatic hardware selection system. This system would allow flight test engineers to step away from the implementation details of their data acquisition system and focus instead on the parameter data that the system is acquiring. The key design goal for this system is to create a mechanism that can automatically transform the requirements for a flight test program into a list of hardware that can accomplish the desired task.

Key Words

Automatic Hardware Selection, iNET, Data Acquisition Systems, XML

Introduction

A vital aspect of every flight test program is the selection of the data acquisition hardware that will collect data during test flights. This process must take into account all of the unique requirements of the flight-test program. It is crucial that the correct hardware be selected at the beginning of each project because the project will invest a significant amount of time and money in acquiring and learning to use the hardware. This investment makes it impractical to switch hardware systems later in the flight test process even if the selected system does not perform all of the required functions optimally.

In order to select appropriate hardware for a flight test program, flight test engineers must contact many different instrumentation system vendors to learn about the capabilities of

their equipment. This process typically takes a long time because the vendors each have their own terminology and unique technology. This adds a high degree of confusion to the process and forces flight test engineers to become experts in the design and operation of many different vendors' data acquisition systems.

This paper discusses the creation of a system that will allow flight test engineers to select data acquisition hardware more efficiently. This system will automatically select the appropriate hardware for a flight test project based on the requirements of the program. Several critical aspects of the design of the automatic hardware selection system will be discussed in depth. First, the paper will show how to refine the process of gathering information to automate the transformation of measurement requirements to data acquisition hardware. Second, the paper will cover how the physical configuration of a test article interacts with the requirements for each measurement. Next, the topic of automatically selecting hardware from a pool of available devices will be discussed. After covering these topics, the paper will describe a possible implementation of an automatic hardware selection system.

The creation of an automatic hardware selection system will help to accomplish the principal objective of the INET program. The goal of the INET program is to create a set of standards that will allow multiple vendors' hardware to interoperate within a single data acquisition system. An important objective of the INET program is to design standards that use XML as the primary data interchange method. The power and flexibility of XML makes it appropriate for use in the automatic hardware selection system as well. This system will make use of several generic XML-based languages to describe flight test requirements and the capabilities of the data acquisition hardware.

Eventually, the XML languages that this paper proposes could be integrated into the INET program's Measurement Description Language (MDL). The MDL is designed to describe the configuration of a data acquisition system via a generic XML language. The goal of this language is to give flight test engineers the ability to use a single file to program a network of data acquisition devices manufactured by different vendors. To do this, each vendor must provide a translator that can convert the MDL language into their native format in order to program their hardware devices. The translator must also be able to export data from each vendor's proprietary format into MDL. The XML described in this paper could be used as a preliminary stage in the process of creating an MDL file.

Selecting Data Acquisition Hardware

In order to design a system that will improve the hardware selection process, it is important to understand what the problems are with the current approach. The current process for selecting the appropriate piece of hardware for a particular data acquisition task involves a great deal of communication between the flight test engineers and the vendors. The flight test engineers must define their requirements and give them to the vendors. The vendors must then interpret the requirements and decide if they have an existing product that can satisfy the customer's objectives or if they need to propose designing a new product.

In addition to the functional requirements of a flight test program, many other factors need to be considered when selecting hardware. The most important of these factors is how the physical environment of the test article interacts with the hardware. The environment aboard the test article can significantly alter the requirements for the data acquisition hardware. For example, the size of each hardware device is much more important for a small fighter jet or UAV than on a large commercial jet. If the flight-test engineers need to monitor a 1553 bus on a large jet they might be more willing to use a rack mounted unit that had extra features like the ability to be reconfigured easily while in flight. A fighter jet program with the same data acquisition needs would have to insist on a small device that could fit into the extremely limited space aboard the vehicle.

Many other factors also need to be considered when selecting hardware. Some of these factors are basic environmental constraints like the weight of the data acquisition system and its operating / storage temperature range. Another consideration is the amount of shock and vibration that the device can handle. Many test articles have very limited available power so it is often very important for the data acquisition system to draw as little current as possible. The peak power requirements of the data acquisition system also need to be considered. Finally, if the data acquisition hardware is going to be used on production commercial aircraft then FAA certification becomes an important requirement as well.

Designing an Automatic Hardware Selection System

The first step in designing an automatic hardware selection tool is to refine the process by which hardware requirements are gathered and sent to vendors. The creation of industry standards has helped to address many of the factors that influence the selection of data acquisition hardware. Vendors can convey a great deal of information to the flight test engineers by stating that their hardware is compliant with a particular industry standard. For example, if a vendor says that a particular device is Chapter 10 compliant, they are communicating to the user that the device supports a particular set of features. The problem with using standards to determine device compatibility is that standards often leave room for misinterpretation. To solve this problem, flight test engineers and vendors must continue to work together to refine the process by which requirements are gathered for a flight test program.

Automatic hardware selection seeks to simplify the process further by completely standardizing the specification of flight test requirements. The key to refining the specification of requirements is to define a standard that is easy to use, flexible enough to describe the requirements for any data acquisition system and extensible so that it can accommodate new features. If the requirements for a flight test project are specified in a standardized manner then a software tool can be created to select the appropriate hardware for the project. This tool can consider all of the issues that are important for each flight test program. A key aspect of the standardization of hardware requirements is the ability to specify the relative importance of each factor that needs to be considered.

Depending on the nature of the flight test program, the importance placed on each factor can vary greatly.

The second step is to define a standard language that can be used to describe the capabilities, features and operational properties of any data acquisition hardware. Several existing standards could be used as the basis for this metadata language. At its most basic level the purpose of this language is very similar to the purpose of the Universal Description, Discovery and Integration (UDDI) standard. UDDI is an international standard that defines an XML language, which allows businesses to publish online directories of their web services. These directories include three items: contact information for the business, a list of products and technical information about the products and services that the business has for sale. Another standard that could be used for this task is the Instrumentation Hardware Abstract Language (IHAL). IHAL defines a standardized XML language for describing all aspects of the features of instrumentation hardware.

While these existing standards can be used to define the capabilities of data acquisition hardware, they may be too generalized to use as part of the automatic hardware selection process. This paper proposes the creation of a completely new language that is designed explicitly for this task. This would eliminate any issues involved in trying to use a language that was designed for a different purpose.

The third step in designing an automatic hardware selection system is to integrate the concept of selecting devices from a pool of shared hardware that is already owned by a flight test group. This concept allows flight test engineers to accomplish their flight tests more efficiently by reusing existing hardware. It can also help to define a common set of hardware devices that can be used interchangeably on multiple programs. This makes the training, configuration and maintenance of the data acquisition hardware much easier. For example, if a particular device is damaged and needs to be replaced, it is simpler to take an extra unit from the hardware pool than it is to send the damaged device back to the vendor for an urgent repair.

The final step is to create a software application that combines the three inputs and generates a list of the suggested data acquisition hardware. This tool will work by checking the list of data acquisition requirements against the features of each vendor's hardware. It will have to consider all of the restrictions that are specified in the requirements document when selecting the appropriate hardware to use. The optional list of available hardware devices will be considered as well during the hardware selection process. This will allow the end-users to see if it is possible to select the hardware for a test flight without having to purchase any new devices. When finished, the application will output a list of the suggested hardware to use for the flight test program.

To use the automatic hardware selection software, the flight test engineer will need to provide two input files. The first file specifies the requirements for the flight test program while the second file contains a list of all of the available hardware devices that the program owns. The third input file is provided by the vendors. Each vendor submits

a file that lists all of their hardware devices and includes detailed information on the features of each device. These three input files will be used by the software to generate the list of suggested hardware.

An Implementation of Automatic Hardware Selection

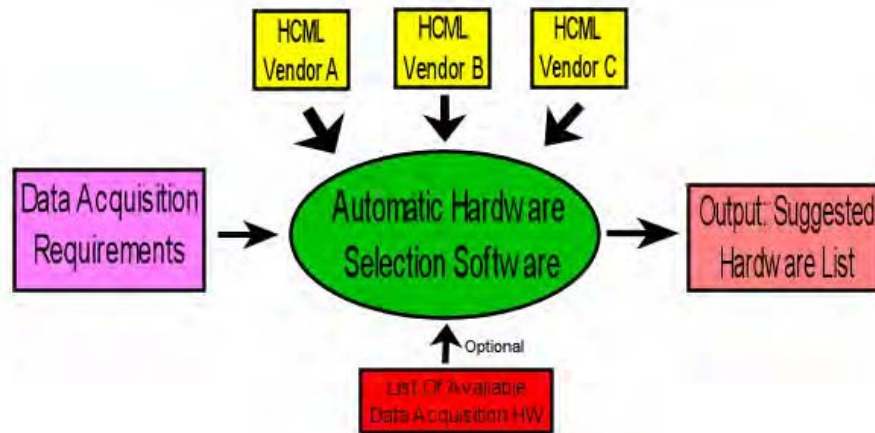


Figure 1: Automatic Hardware Selection System Flow Chart

Implementing the automatic hardware selection system involves creating four main components. The four components are the data acquisition system requirements language, the hardware description and compatibility language, the available hardware list language and the software application that processes the three input languages. The three input languages will be based on XML and they will be released as XML schemas. This will make it easy for flight-test engineers and vendors to create documents that comply with the standards. The software application will be designed to read these documents, validate their contents against the appropriate schema and generate the desired output.

System Requirements Markup Language

The most important of the three input XML languages is the Data Acquisition System Requirements Markup Language (SysReqML). The SysReqML language can be used to describe the data acquisition system that is needed for a particular flight test program. The flight test engineers will create a SysReqML document at a very early stage in each flight test program. The requirements will be specified in a generalized manner in the document to allow for more flexibility. A key example of this flexibility is the ability to specify ranges for each item that is described in the document. This makes it possible to easily change things later in the setup process.

The content of the SysReqML language will be similar to the content that is included in the INET program's MDL language. In fact, an MDL document could be used as the data acquisition system requirements document. However, it is unlikely that a flight test engineer would have created an MDL file at this early of a stage in the configuration of the data acquisition system. A more likely alternative is that the SysReqML document

would be used as the basis for creating the MDL document when it is time to configure and program the hardware.

The SysReqML language is structured around the concept of data acquisition nodes. Each node represents a physical location on the vehicle where one or more measurements need to be collected. When the requirements are transformed into actual hardware, these nodes will be converted into data acquisition units. Nodes that are physically close to each other on the test article can be grouped together in a single large data acquisition system or kept as separate units. If a node has too many measurements, it may not be possible to collect all of them with a single data acquisition unit. In this case, a node could be separated into multiple units. The decision to combine or separate nodes will be based on many factors including the size of the data acquisition units and the available space on the test article.

```

- <DAS_REQUIREMENTS>
  <PROGRAM_NAME>Sample Requirements File</PROGRAM_NAME>
- <TEST_ARTICLE>
  - <NETWORK Type="CAIS">
    <OUTPUT Type="PCM Stream">...</OUTPUT>
    - <OUTPUT Type="Recorder">
      <RECORDER_TYPE>Chapter 10</RECORDER_TYPE>
    </OUTPUT>
  </NETWORK>
  <PHYSICAL_LAYOUT>...</PHYSICAL_LAYOUT>
</TEST_ARTICLE>
- <NODES>
  - <NODE Name="RWNG">
    <DESCRIPTION>Right Wing Data Acquisition Node</DESCRIPTION>
    <PHYSICAL_LOCATION>...</PHYSICAL_LOCATION>
    <ENVIRONMENTAL_CONDITIONS>...</ENVIRONMENTAL_CONDITIONS>
    - <NETWORK_INTERCONNECTION>
      <NETWORK_TYPE>CAIS</NETWORK_TYPE>
      ...
    </NETWORK_INTERCONNECTION>
    - <DATA_ACQUISITION>
      - <BUS_DATA>
        <BUS Type="1553">...</BUS>
        ...
      </BUS_DATA>
      - <SENSOR_DATA>
        - <MEASUREMENTS Type="Strain">
          <MIN_COUNT>5</MIN_COUNT>
          <MAX_COUNT>12</MAX_COUNT>
        </MEASUREMENTS>
        - <MEASUREMENTS Type="Pressure">
          <MIN_COUNT>35</MIN_COUNT>
          <MAX_COUNT>55</MAX_COUNT>
        </MEASUREMENTS>
      </SENSOR_DATA>
    </DATA_ACQUISITION>
  </NODE>
  ...
</NODES>

```

Figure 2: An Example of a SysReqML Document

The description of each node on the test article will contain several sections. Each node will contain information on its location in the test article and the environmental conditions in the area surrounding the node. The most important section in each node describes the measurements that need to be collected by the node. The measurements are broken down by type and the flight test engineer provides a range for the minimum and maximum number of each type of measurement that needs to be collected by the node. For example, a node could specify that it needs to collect between 35 and 55 pressure measurements. The node also includes information on any additional constraints that limit the flexibility of the automatic hardware selection system. For example, available

space, temperature or vibration restrictions may limit the types of devices that can be used on a node.

In addition to the list of nodes, the SysReqML document specifies several other aspects of the flight test requirements. It includes a section that details the layout of the nodes on the test article. This section contains information on the physical distances between nodes. This information helps the system to know which nodes can be grouped together and which nodes need to be separate data acquisition units. The document also lists the desired network type for the aircraft; for example CAIS or TCP/IP. In addition it will specify all of the required outputs from the data acquisition system and which outputs are recorded, transmitted or decommutated in real-time on the test article.

The final major section of the system requirements document lists the priorities that the flight test engineers have assigned to each factor that is considered when selecting hardware. This helps to optimize the hardware selection process so that it reflects the intention of the flight test engineer. Some of the factors that can be weighted include the size of the units, the channel density of the devices, the type and speed of the network that is used to communicate between the nodes and the engineer's desire to reuse existing hardware.

Hardware Compatibility Markup Language

The second input language for the automatic hardware selection system is the Hardware Compatibility Markup Language (HCML). This language will be used to describe each hardware device that a vendor manufactures in a detailed but generalized manner. The HCML allows vendors to describe the features and limitation of all of their products. It is designed with extensibility in mind to allow new types of hardware to be described without radically changing the language.

In order to dynamically select the appropriate hardware device to satisfy a particular data acquisition requirement, it is vital that the vendors fully describe their products using HCML. In order for the software application to select the hardware to use for a flight test project, it must examine the set of HCML documents that contain all of the possible devices that can be used on the program. This will allow the software to make a fully informed decision when selecting a device to use.

The HCML language allows vendors to describe many different characteristics of their products. HCML includes information about the inputs and outputs of each device. It also includes a description of the transformation that is performed by the device on the input to generate the output. This conversion can be a simple analog to digital conversion for orange wire measurements or a selected data extraction for bus data. The HCML also includes information on the environmental and physical configurations of the device. This information includes the weight and size of the device and its operating temperature range. Any special features or limitations on the use of the device must be included in the HCML as well. Two examples of these limitations are the device's maximum sampling rates and the number of slots occupied in a unit by the device.

```

- <HCML>
- <VENDOR Name="TTC">
  <CONTACT_INFORMATION>...</CONTACT_INFORMATION>
- <DEVICES>
+ <DEVICE Name="PSO-103-1">
+ <DEVICE Name="BIM-553-8">
- <DEVICE Name="MSCD-104D-2">
  <SUPPORTED_BUS>R-Bus</SUPPORTED_BUS>
- <BOX_COMPATIBILITY>
  <BOX>MCDAU-2000</BOX>
- <SUPPORTED_SLOT>
  <MIN>0</MIN>
  <MAX>30</MAX>
</SUPPORTED_SLOT>
</BOX_COMPATIBILITY>
- <PHYSICAL_DIMENSIONS>
- <SIZE>
  <LENGTH>63.25 mm</LENGTH>
  <WIDTH>66.8 mm</WIDTH>
  <HEIGHT>8 mm</HEIGHT>
</SIZE>
  <WEIGHT>62 grams</WEIGHT>
  <POWER Source="BOX">28V</POWER>
</PHYSICAL_DIMENSIONS>
- <ENVIRONMENTAL>
- <TEMPERATURE>
  - <OPERATING>
    <MIN>-50 C</MIN>
    <MAX>+100 C</MAX>
  </OPERATING>
  - <STORAGE>
    <MIN>-55 C</MIN>
    <MAX>+100 C</MAX>
  </STORAGE>
</TEMPERATURE>
</ENVIRONMENTAL>
- <INPUTS Channels="4">
  - <INPUT Channel="1" Type="Sensor Data">
    <DATA_SOURCE>Analog Voltage</DATA_SOURCE>
  </INPUT>
  + <INPUT Channel="2" Type="Sensor Data">
  + <INPUT Channel="3" Type="Sensor Data">
  + <INPUT Channel="4" Type="Sensor Data">
</INPUTS>
- <OUTPUTS Channels="4">
  - <OUTPUT InChannel="1" OutChannel="1" Type="PCM Sample">
    - <CONVERSION Type="Analog To Digital">
      <RESOLUTION>16 Bits</RESOLUTION>
    - <GAIN>
      <MIN>1</MIN>
      <MAX>10000</MAX>
    </GAIN>
    <EXCITATION>...</EXCITATION>
    </CONVERSION>
  </OUTPUT>
  + <OUTPUT InChannel="2" OutChannel="2" Type="PCM Sample">
  + <OUTPUT InChannel="3" OutChannel="3" Type="PCM Sample">
  + <OUTPUT InChannel="4" OutChannel="4" Type="PCM Sample">
</OUTPUTS>
  <SPECIAL_FEATURES>...</SPECIAL_FEATURES>
  <LIMITATIONS>...</LIMITATIONS>
</DEVICE>
</DEVICES>
</VENDOR>
</HCML>

```

Figure 3: An Example of an HCML Document

The example in Figure 3 shows a possible HCML description of the MSCD-104D-2, which is one of TTC's signal conditioner modules. This module has four input channels and performs an analog to digital conversion on each input with 16-bit resolution. The description of the module includes the information that the output can be sampled in the PCM Format. It also lists the physical size of the MSCD-604D-2 module and its operating and storage temperature ranges.

Hardware List Markup Language

The final language that is used by the automatic hardware selection system is the Hardware List Markup Language (HardwareListML). This language describes a list of hardware devices. For each device, the number of units that the flight test program currently owns is listed. The number of units that are assigned to programs and the number of units that are required for the current flight test program can also be specified.

Documents written in the HardwareListML language are used for two purposes in the hardware selection software. The first use is to provide a list of the hardware that a flight test program owns as an input to the hardware selection system. The system can try to fit the existing hardware to the requirements for the program. This can minimize the number of new devices that need to be purchased. The second use of HardwareListML is in the output of the software. The software generates a list of required hardware from the project requirements. This list can be written in HardwareListML.

Automatic Hardware Selection Software

In order to generate a list of hardware to use for a flight test program, the flight test engineers will provide the three inputs to the automatic hardware selection software. The software will process the requirements from the SysReqML document. It will then select appropriate hardware from the HCML files that are provided by the various vendors. If multiple hardware devices exist that can satisfy the basic requirements then the software will use the priorities specified in the SysReqML file to determine which device should be used. If an existing hardware list file in HardwareListML is provided then the software will try to select hardware that the flight test program already owns. Finally, after processing all of the requirements the software will generate a list of all of the required hardware and how to interconnect all of the pieces.

Conclusion

The automatic hardware selection system described in this paper still has a long way to go before it can be completely implemented. The idea needs to be refined further and the definitions for the input XML languages need significant additional work to make sure that they are complete enough to describe any hardware device and any set of data acquisition requirements. The exact mechanism that would be used by the software application to select the hardware for a particular task also needs to be developed.

There are several additional enhancements that could be made to the system that are not mentioned in this paper. One crucial enhancement would be the creation of a graphical user interface to allow flight test engineers to more easily define their requirements. This interface would convert the user specified requirements into an automatically generated SysReqML document.

Automatic hardware selection has the potential to significantly improve the process that flight test engineers use to select hardware for their test articles. The automatic hardware selection system is an important component of the INET vision of simplified data acquisition system setup. This system will help to take us further down the road towards completely automatic system configuration.

References

1. William Malatesta, and Clay Fink, "Measurement-Centric Data Model For Instrumentation Configuration", Proceedings of the 43rd Annual International Telemetry Conference (ITC/USA 2007), USA, October 23-25, 2007.
2. John Roach, "The Architecture of Aircraft Instrumentation Networks", Proceedings of the 43rd Annual International Telemetry Conference (ITC/USA 2007), USA, October 23-25, 2007.
3. Charles H. Jones, PhD, "Towards Fully Automated Instrumentation Test Support", Proceedings of the 43rd Annual International Telemetry Conference (ITC/USA 2007), USA, October 23-25, 2007.

4. Hamilton, Fernandes, Koola, and Jones, “An Instrumentation Hardware Abstraction Language”, Proceedings of the 42nd Annual International Telemetry Conference (ITC/USA 2006), USA, October 23-26, 2006.
5. Knowledge Based Systems, Inc., “IHAL 2.0 Documentation”, <http://www.kbsi.com/ISAIH/Documentation.htm>, 2007.
6. UDDI Oasis Standard, “UDDI Version 3.0”, <http://uddi.xml.org/>, February 3, 2005.