DOI 10.5162/ettc2018/9.2

# DASP towards a generic Data Acquisition Configuration System

*González-Martín Moisés, Gonzalez-Pastrana Jose Antonio, Behr Christian, Brueggemann Thomas, Rominger Claude, Keil Karsten, Montes-Sánchez Sergio, Paredes-Huerta Roberto*

Flight Test – Airbus Defence and Space

{moises.gonzalez, jose.o.gonzalez,christian.behr2, thomas.brueggemann, claude.rominger,Karsten.keil}@airbus.com, {Sergio.montes, Roberto.paredes}@altran.com

**Abstract**

Flight Test Instrumentation is a process where multiple measurements must be acquired by means of different equipment and acquisitions systems. With the Ethernet adoption, multiple vendor solutions can be connected to the Flight Test acquisition network. As a consequence, vendor-specific systems configuration is not enough for the control and generation of a global FTI configuration.

Having in mind the requirement of flexibility and compatibility, Flight Test in Airbus Defence and Space is working on the implementation of a generic architecture capable to set up complex Ethernet architectures using a business-model-agnostic paradigm. This document explains the main steps in the programming generation process: definition of the input interfaces, design of a core solution based on a plug-in philosophy (allowing an easy integration of new components/manufacturers/protocols), and output generation.

**Key words:** DASP, FTI, Plug-in, XML, Database, Data Acquisition Configuration

## Introduction

Until now, acquisition systems have been programmed by using manual processes, combined with ad-hoc tools which have to be updated every time a manufacturer changes its systems or releases new hardware components. This tailor-made solution implies a constant evolution and is definitely vendor-dependent.

DASP is an all-in-one tool to generate the programming and configuration files for the acquisition systems given a prototype.

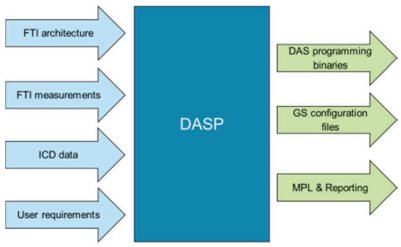


*Fig. 1.DASP schema*

The main benefit of DASP is to have a common normalized starting point for generating FTI configurations for heterogeneous acquisitions systems. There is a global need of having a tool for supporting multi-vendor FTI architectures and handling large flight test programs with a huge set of parameters and measurements.

The new approach for generating configurations allows facing all the challenges considered as big difficulties in the past:

- Ensure backward compatibility with the current supported vendors.
- Adopt and maintain new manufacturers with specific hardware.
- Mitigate the problems derived of manual processes, such as human-errors and delays in programming schedules.
- Support a huge amount of data. Often dozens of thousands parameters have to be acquired and doing this with manual processes is not affordable.

- Provide an appropriate preliminary feedback to final users about errors or warnings which can derive in future problems difficult to detect causing delays and the need of releasing new configurations.
- Give a fast response to constant changes requests during a test campaign

## DASP architecture design

Common data model design

The tool is based on a shared common data model for supporting all the data acquisition systems features but also parameters, hardwired measurements and user requirements have to be programmed.

The information can be provided by means of XML files or retrieved from a database (standalone mode vs database integrated mode). All this data is managed by DASP which is in charge of loading in the internal data model to be analyzed before generating the expected outputs.

Having a common data model reduces the compatibility problems between different vendors, allows the reusability of SW modules and promotes the simplicity of the solution.

The different models have been defined avoiding specific features or restrictions; the definition is generic and flexible. This concept guarantees the agnosticism of the application which is essential in DASP philosophy.

Plug-in philosophy (Low level shared API)

The plug-in architecture solves the complexity of having different acquisition system vendors with their own features and restrictions. By means of a generic interface definition DASP manages the different plug-ins on the same way. From the shared FTI architecture data model (flexible and generic) is possible to support all the manufacturer relevant information.

The different vendor plug-ins must implement the following methods:

- Set inputs: How DASP data models are received and checked by the plug-in.
- Build: Each plugin specific model must be loaded and acquisition system sources generated from it.
- Generate binaries: The vendor compiler has been also integrated in the solution, having a full-row tool which generates also the acquisition system binaries with the required parameter programming.

This design allows the tool extensibility, the application can be dynamically extended to include new vendors and capabilities, as features can be implemented as separate components they can be developed in parallel without having cross dependencies, and therefore an agile environment is guaranteed. The plugin framework ideally provides simplicity with a well-defined interface and documentation for plugin implementation, developers have a clear roadmap.

By means of using a generic plugin loader (direct access to build folder), DASP is able to know how many plug-ins are available to be used whenever the plugin name matches with the manufacturer name defined in the FTI architecture data model.

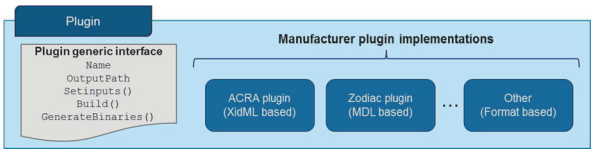


*Fig. 2.Plug-in interface*

The plug-in architecture is not only intended to be used for acquisition systems management, also configurable FTI equipment and Data Processing generation is considered as a plugin-able feature in order to support multiple FTI modules (recorders, switches, Data processing files…).

Configurable application

Using configurable models allow the user to change internal capabilities and execute different configurations without changing the source code. There is no need to know about low level implementation details or having programming skills, final user can easily change configuration parameters.

DASP has been implemented as a console application, in the future it will be included as a service in the framework FIDA, for managing the tool some configuration files (currently XML files but in the future it will be handled by forms in FIDA) have been implemented where the user can define configuration parameters, input paths, constant values…

A general configuration file (for DASP) and other specific ones (each plug-in has one) are provided. The idea is not to force the user to

provide all the configuration values so DASP has those parameters internally defined with default values according to the most used configurations.

**Required inputs building & integration**

Input data models

All the data is serialized into DASP model from the different input data sources. The model is divided in four layers:

- ICD data: Includes all the information about bus parameters (currently Arinc-429, Serial, CAN, Mil-Std-1553, Ethernet and Stanag-3910 buses are supported)
- Hardwired measurements: Defines the analog and discrete measurements installed on the A/C such as sensors, on/off signals and so on.
- User requirements: Defines the parameters to be programmed and where to send them. This input is based on the defined hardwired measurements and also on the ICD data so this model is directly correlated with the previous ones. After considering the requirements we could have a subset of measurements and bus parameters.
- FTI architecture: Includes all the information about the HW acquisition architecture installed on the A/C; the location of the acquisition cards, internal card settings, channel parameters configuration, bus and measurement connections as well as how the input data is linked to the outputs.

The main reason of having these layers is because there are different actors involved in the FTI configuration definition process: design offices, HW specialists and FTI departments. Each layer has its own restrictions and can be validated apart from the others making easier the process of reporting preliminary errors or warnings.

Database approach & XML file generation

DASP supports two different ways for providing the required information: Directly from a database or by a set of defined XML files, in this case the information can be stored in a database as well, but it is translated into a structured file by means of an auxiliary tool.

Although there is no difference between both approaches (from the final user point of view), there are some considerations to be taken account:

- Using XML files allows DASP to be compatible with different databases, making it possible sharing information between different sites. Most of FTI departments have its own database models and it is a hard challenge to define a common model for every organization or department, ensuring the lack of backward incompatibilities and the backup of the legacy data into this new model.

  The use of these auxiliary files solves this problem with the disadvantage of having the need of defining some tools or methods for generating the files but with the advantage of avoiding the need of a database connection in DASP (standalone execution) and the need of retrieving the information from a database which is a less efficient process.
- DASP can be integrated with a Database and retrieve the data directly from it. This way of working implies the implementation of specific modules (based on database model) for querying the information. Despite the fact that a database promotes centralization (different applications connected to the same data source), data security and minimizes data inconsistency it requires a dedicated connection and more computational resources.

**Manufacturer support & implementation**

Current challenge

The lack of a common language for different manufacturers and acquisition equipment configuration, and the wide casuistic due to the multiple vendor restrictions, derives into a complex scenario difficult to be managed and maintained.

Using a manufacturer common model

Each manufacturer has its own standard and format but the required information is the same for all of them. Starting from a common input model (FTI architecture data model) is possible to support different data formats.

Each plugin has its own data model and is in charge of loading it from the provided common model. The conversion restrictions and

particularities have to be implemented and known at the plug-in level, isolating vendor dependencies between them, but also with DASP that does not need to know specific considerations. One of the benefits of having the plug-in based architecture is the fact that we can manage this complex scenario without compromising the design of the whole application. Specific tasks can be performed at the plug-in level, by doing this, only the plug-in will depend on third-party libraries, for example, a development toolkit provided by a manufacturer to program acquisition cards, or generate binaries for specific hardware architectures.

Plugins could also implement its own data models in order to generate the required source files to be compiled for generating specific binary files.

Acquisition system Datasheet automatization (by using a common HW description model)

Some automated processes have been implemented promoting the application maintainability, regarding the different acquisition system vendors, a procedure has been defined for avoiding code modifications every time a new card or HW is released.

In the case of CW-Controls (formerly ACRA), the use of XdefML (similar to .xsd schemas) allows an easier way for supporting the different cards provided by this manufacturer. This file includes all the information for configuring a specific card (allowed ranges, data types, properties to be configured…) with the same structure. This schema is the translation of the system data sheet into an open-standard format file. The implementation of generic parsing method (in the plugin level) allows the support of new cards without modifying the code, just by managing this datasheet files (in most cases provided and supported by the vendors).This philosophy allows to know in a simple way how to configure the acquisition cards, set default values or validate the given data.

In other cases, vendors do not provide any file or format to be automated, we've seen that investing time and effort in generating and maintaining this information really worth it instead of implementing vendor-specific solutions sensitive to changes or updates of existing components.

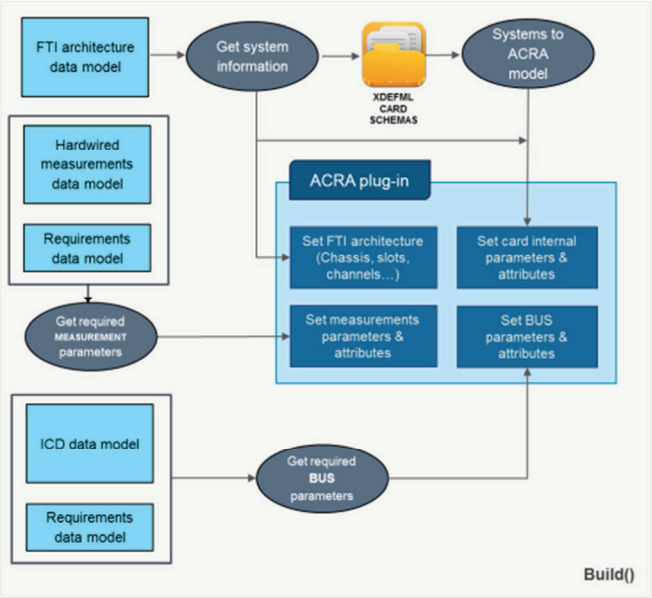


*Fig.3. ACRA Plug-in building process diagram*

**DASP output generation**

Output description

DASP generates multiple outputs, in some cases these outputs are not directly generated by DASP but by the plug-ins:

- DAS binaries: As a process of each plug-in, the required binary files are generated in order to configure the different acquisition systems (this process depends directly on the compiler provided by the manufacturer).
- Data Processing configuration files: In order to know how to decode the information programmed in the

acquisition systems which is send to the recorder and telemetry facilities, some configuration files are generated based on Data Processing Software Implementation (this procedure is implemented as a plug-in).

- General architecture configuration files: DASP has a global picture of the whole FTI design so it generates a configuration file which includes not only information for the programmable systems, but also includes non-programmable sources (i.e. GPS signals) which also belongs to the architecture too.
- User report files: An ad-hoc information module has been implemented in order to provide information about the execution process (detected errors and warnings), the programmed parameter list (for monitoring purposes) and some other information considered as relevant for debugging purposes. Modular design and multi-format support (plain text, .csv, .xlsx etc.) allows an easy way for generating reports and logs.

  Preliminary data validation & error handling is considered as a key feature for reducing time and costs while generating parameters programming,

  This is a module in permanent progress.

## Future capabilities

Integration with diagnosis & analysis tools

An important phase is the verification phase when all the raw date has to be processed and analyzed. To perform this task, some tools are used, such as the network sniffer Wireshark, so all the traffic must be dissected and processed in order to know the internal characteristics of the received parameters.

The most important problem is that these applications cannot analyze deep enough the UDP packets because they do not have information about the structure of each message. This structure depends on the nature of parameters which have been programmed in the acquisition systems, so DASP can take the advantage of having all this information and provide an auto-generated script allowing the filtering and a deep analysis of the data.

In the case of Wireshark, the tool is ready to be integrated with LUA scripts, which can be a new output in DASP, but also it is possible to build dynamic libraries or any other ad-hoc module to be integrated with any monitoring tool.

A/C DAS binary deployment

The process of installing the configuration files in the acquisition systems is a manual process done by the specialist in the A/C. In order to automate this procedure, DASP will provide an installation script for uploading the binary files directly in the A/C facilitating the installation process. The script needs to be executed in the FTI A/C network.

Sanitization of Data Processing configuration files

DASP will support the process of removing sensitive information from the generated outputs. As an optional input defines a list of specific parameters which have to be programmed but filtered from GS configuration files and parameter reporting, DASP will be able to deal with classified information, reducing the generated outputs classification level.

Automatic unitary test generation

An important stage while programming all the acquisition systems is to verify the configuration in the laboratory before installing the binaries in the A/C. Until now, this process is manually performed, DASP is able to generate unit test or applications for validating the expected behavior of every acquisition component reducing the scheduling time

## Conclusions

DASP will be the future tool in FTI configuration programming tasks in Airbus Defence and Space, its greatest success will depend on the participation of the different actors involved in the process. Since it has been developed as a scalable modular tool it allows new features implementation in an easy-way. Having a multidisciplinary normalized and compatible programming tool will reduce costs and timing.

## Acronyms

A/C: AirCraft

DAS: Data Acquisition System

DASP: Data Acquisition System Programmer

FIDA: Flight Instrumentation DAtabase

FTI: Flight Test Instrumentation

HW: Hardware

ICD: Interface Change Document

XML: eXtensible Markup Languaje