# Integrating Software Data Loaders into ATE Systems

Troy Troshynski
Avionics Interface Technologies
(A Division of Teradyne)
Omaha, NE U.S.A.
troyt@AVIFTECH.com

*Abstract*— **The Integrated Modular Avionics (IMA) concept has been adopted into several new military and commercial aircraft programs such as the F-22, F-35, Airbus 380, and Boeing 787. The goal of the IMA concept is to reduce the number and varieties of hardware computing modules and to increase the portability of avionics software. The IMA concept is currently being driven by industry initiatives such as the Future Airborne Capability Environment (FACE$^{TM}$) Consortium. It can also be seen in new industry standards such as ARINC 653 which defines an avionics application standard software interface. As a result of the adoption of the IMA concept, new avionics hardware modules are becoming increasingly generic, multipurpose, and reconfigurable based on the loaded software applications. Therefore, the automated test equipment (ATE) used to support maintenance and service of these new avionics systems must consider the use of standard approaches for handling loadable avionics software such as ARINC 615 and 615A software data loaders. This paper provides a brief technical overview of IMA systems and ARINC software data load protocols. It also explores strategies for integrating software data loaders into ATE systems that are required to support IMA based Units Under Test.**

***Keywords—IMA, ARINC 653, ATE, Data Loader, ARINC 615, ARINC 615A, FACE$^{TM}$***

## I. Integrated Modular Avionics

Integrated Module Avionics (IMA) is the primary avionics system design concept that has adopted for use in modern commercial and military avionics applications.

### A. *IMA in Commercial Avionics Systems*

The commercial airline industry has formally defined IMA in the ARINC Report 651 (Design Guidance for Integrated Modular Avionics). ARINC 651 defines an IMA system as consisting of a set of common core computing cabinet subsystems with each subsystem being a standardized chassis housing multiple modular computing modules. The modules communicate within the chassis over a standardized backplane bus. The computing subsystem communicates with other subsystems, external sensors, actuators, and controls via the aircraft system data busses or data network.

In early IMA avionics systems, such as the Boeing 777, ARINC 629 was used as the data bus for communications between cabinet subsystems and between computing resources in the IMA cabinet and external sensors and actuators. ARINC 629 is a 2 Mbps shared data bus that can support up to 128 terminals. The cabinet backplane adopted for early IMA systems is defined by ARINC 659. ARINC 659 defines a time division multiplexed bus system with each node on the bus programmed with a communications schedule table.

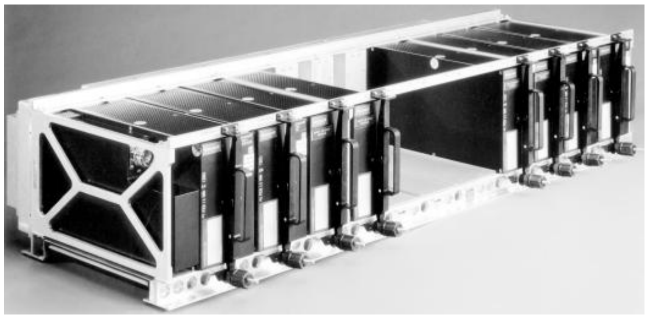
**Figure 1 IMA Cabinet**

The most recent IMA systems use deterministic Ethernet networks (defined by ARINC 664) for both internal cabinet communications and external communications between cabinets and remote sensors and actuators.

A primary goal of IMA systems design is to enhance the modularity of the avionics system in order to reduce the overall cost of ownership. The IMA approach is to have common computing resources in the cabinet subsystems and software defined avionics applications. A standardized form factor and cabinet system allows for economies of scale in the design and manufacture of the avionics components leading to reduced costs of the avionics systems. Common, standardized avionics hardware components also reduce maintenance costs for avionics systems by reducing the required inventory of spare system components since a single hardware component type may be used as a spare for multiple parts of the avionics system.

### B. *Common Avionics Software Applications Environments*

In addition to commonality and modularity in the avionics hardware, the IMA concepts is also based on a modular and portable approach to software design. The commercial airline industry has defined a common application environment in ARINC 653. This specification defines a standardized set of interfaces to guide the development of operating system (OS) platforms which provide a standardized set of application programming interfaces (APIs) and services to the avionics

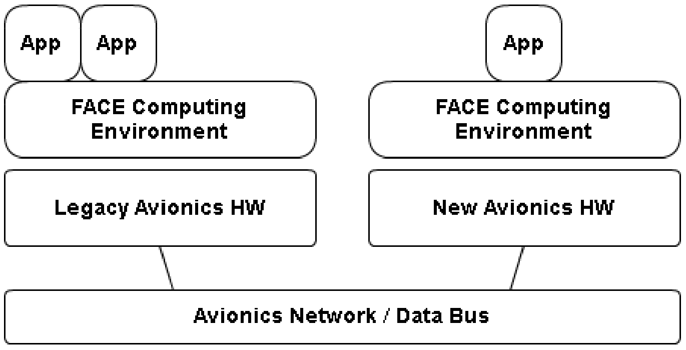
App
App
App
FACE Computing Environment
FACE Computing Environment
Legacy Avionics HW
New Avionics HW
Avionics Network / Data Bus

**Figure 2 FACE System Concept**

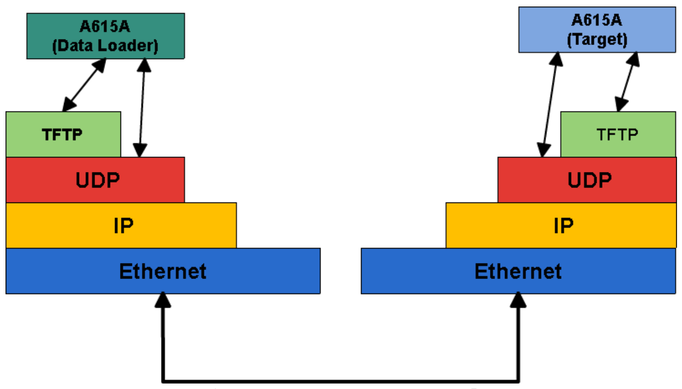

**Figure 3 A615A Protocols**

applications. The ARINC 653 based operating systems also provide a portioned platform which ensures that multiple avionics applications can operate on the same system without interference and with fault isolation.

While the ARINC standards are focused on representing and defining the goals of the commercial aerospace industry, the FACE$^{TM}$ consortium is a U.S. Government and industry partnership focused on defining an open, modular, and portable software environment supporting IMA in military avionics systems

The FACE$^{TM}$ technical standard defines a standardized software platform and environment in which avionics applications can operate. The platform is defined such that it can provide a common computing environment for avionics applications operating on different hardware implementations. The goal is to foster the portability and reuse of avionics software applications and to make the application, as much as possible, independent from the hardware platform on which it operates.

## II. Loadable Software Parts & Data Loaders

### A. *Software Data Load Overview*

As the commercial and military avionics industries continue to become more and more focused on software centric avionics systems, it has become increasingly important to also have standard methods of loading software and data into the avionics systems. The commercial industry has addressed this with the development of software packaging standards such as ARINC 665 and with software loading standards such as ARINC 615 and 615A. These standards rely heavily on common file transfer protocols, such as Trivial File Transfer Protocol (TFTP). Also, common network communications protocols, such as Ethernet, Internet Protocol (IP), and User Datagram Protocol (UDP) are also utilized by the ARINC data loader standards.

While the military avionics industry has not standardized software data loading to the same extent of the commercial industry, military systems typically utilize the same core technologies and protocols (UDP/IP/Ethernet) as employed by the ARINC standards.

### B. *ARINC 615A Software Data Loaders*

ARINC 615A defines the standard file and data transfer method for moving data and software files between the target avionics system and an onboard data loader (ODL) or a portable data loader (PDL). ARINC 615A utilizes a an Ethernet LAN interface (wired) for communications and employs TFTP over UDP/IP.

ARINC 615A specifies four operations (from the perspective of the data loader) that a user of a data loader can execute. The INFORMATION operation is used to read software configuration and version information from the avionics system. The UPLOAD operation is used to upload files from the data loader to the avionics system. This operation is used for software updates and to load software configuration data items. The MEDIA DEFINED DOWNLOAD operation allows the user to provide a list of a known set of files residing on the avionics system which are to be downloaded to the data loader. The OPERATOR DEFINED DOWNLOAD operation allows the operator of the data loader to query the avionics system for a list of files that can be downloaded. The operator and select from the available files which are to be downloaded.

In all of the operations, the data loader initially acts as a TFTP client and requests the initiation of an operation by executing a TFTP read file request to the avionics system. The name of the requested file signals the type of operation requested by the data loader. The target avionics system acknowledges the request with acceptance or denial signaled in the contents of the requested file. After acceptance the data loader becomes a TFTP server and control is taken over by the avionics system. For UPLOADS the avionics system reads files from the TFTP server of the data loader. For DOWNLAODS and INFORMATION operations the avionics system writes files to the TFTP server of the data loader. While the upload or download files are transferred, the avionics system provide periodic status files (TFTP writes) to communicate the progress and results (fail or succeed) of the operation.

### C. *Software Data Loader Use*

ARINC data loader are commonly used by airline maintenance crews and technicians on flight lines and in MRO shops to load software updates to in-service aircraft. ARINC

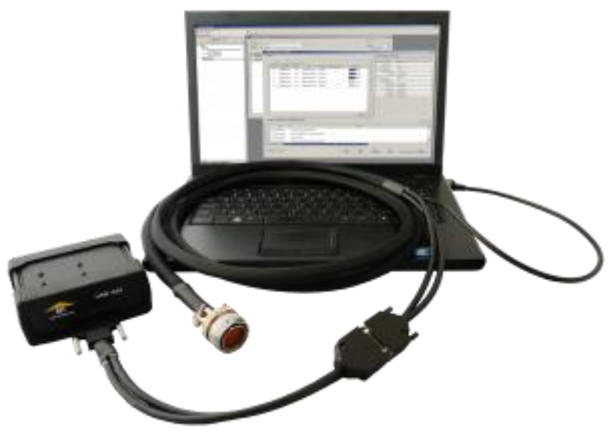

Figure 4 ARINC 615/615A PDL

data loads are also used to load aircraft navigation databases to the flight management systems. These navigation databases typically are required to be updates monthly.

A typical PDL consists of a portable PC system or notebook, with an ARINC 615 adapter cable. The user executes the software data loads by interfacing with a data loader graphical user interface (GUI) application.

The software data loaders are also heavily used in system integration labs, MRO bench top applications and in avionics component manufacturing and production areas to load individual avionics LRU's which are off aircraft. Because there is often automated test equipment (ATE) systems available in these off aircraft environments, there can be obvious benefits and cost savings by utilizing the avionics network and bus interfaces as well as the computing resources of the ATE to host ARINC software data load functions. By integrating the data load functions into the ATE, the requirements for PDL equipment can be reduced in these off aircraft situations.

## III. INTEGRATING DATA LOAD FUNCTIONS IN TO ATE

### A. ATE Data Loader Hardware & Interfaces

ATE systems for avionics usually include one or more modular electronic instrumentation subsystems and a host controller PC system.

Figure 5 ATE System

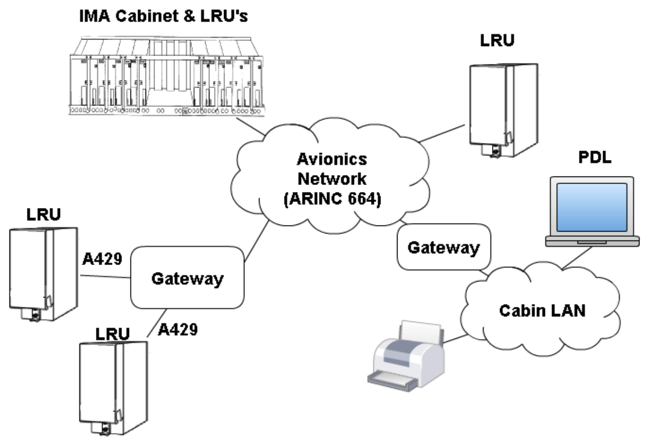

Figure 6 Avionics Network with PDL

The typical standardized backplane platforms employed in the instrumentation subsystems are VXI (VME eXtensions for Instrumentation), PXI Express (PCI Express eXtensions for Instrumentation), and LXI (LAN eXtensions for Instrumentation). These backplane chassis house avionics bus and network interface instruments which are used for communications from the ATE to the avionics UUT. MIL-STD-1553, ARINC 429, Ethernet, ARINC 664, and Fibre Channel are some of the common avionics communications interfaces typically supported in ATE systems. With the avionics busses and networks such as ARINC 429, Ethernet, and ARINC 664 commonly available within ATE subsystems, portable data loader functions can easily be integrated into the ATE eliminating the need for an external dedicated PDL in situations where ATE equipment is already available.

For on aircraft load scenarios, the PDL usually interfaces to the avionics system via a connection to a cabin LAN. The cabin LAN provides a portal to the avionics system via a gateway to the avionics system network which is typically a avionics Ethernet (ARINC 664) network.

Some software loadable LRU's reside on the far side of additional gateways between the ARINC 664/Ethernet network and legacy buses such as ARINC 429. In on aircraft loads the PDL may only require a simple Ethernet LAN interface to communicate to all of the systems loadable LRUs with bus type translations handled transparently by the systems gateways. However, for shop loads of individual LRUs which are off aircraft these gateways may not be present or available. Therefore the ATE software data loader function must be capable of direct interface bus type to the LRU. This means that the ARINC data loader protocol software must be capable of communications directly via a UDP/IP/Ethernet LAN, an ARINC 665 test instrument, and an ARINC 429 test instrument.

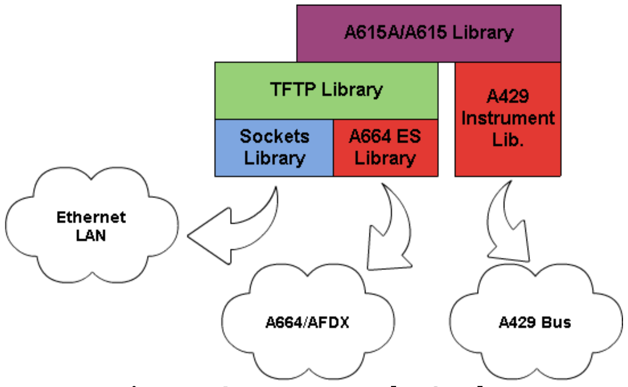

**Figure 7 ATE Data Loader Stack**

## *B. Data Loader Software Interfaces*

ATE systems are used to execute automatic tests designed for the specific type of UUT. The automated test program sets (TPS) are authored in high level programming languages such as C, C++, Visual Basic, and LabVIEW and utilize API libraries provided by the ATE platform. While on aircraft data loads usually require access to load functions via a GUI, in the ATE system the data load functions are more commonaly accessed via a software API that is accessible from the TPS.

To be sufficiently useful to the TPS, the data loader APIs must present a high level set of functions to the TPS and not burden the TPS with any knowledge of the details of the protocol operations such as the TFTP file transfers. With a software architecture built on a API library holding the core of the ARINC 615 data loader protocol implementation, a high level API can be provided for use by a TPS and also simultaneously allow access to the data loader function from a GUI and a scriptable command line application.

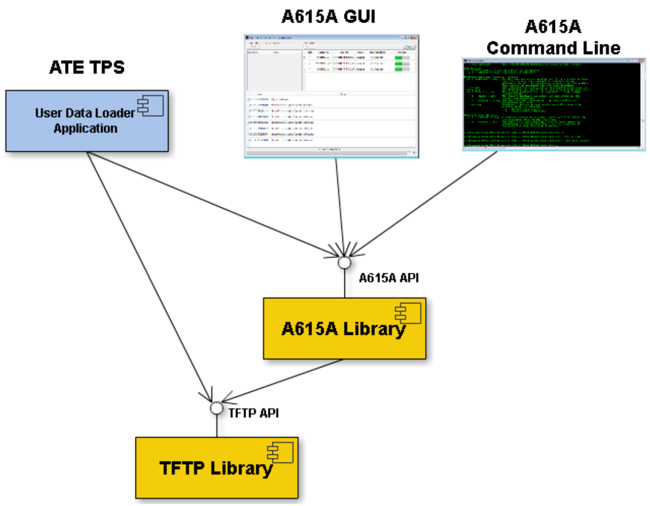

**Figure 8 Data Loader APIs**

The high level API to the data loader core should provide high level functions such as:

### *1) Upload Operations*

```
a615aTargetOpUpload(char* aMediaSetPath,
bool aValidateMediaSet);
```

For upload operations, a simple function is required that takes as input the filename and path to the media set of software to be loaded. Also an indicator can be provided as input to instruct the loader library to validate the loadable files if requested by the TPS. The loadable files can be validated because they are packaged into a media set (defined by ARINC 665) which includes checksums over each of the files.

### *2) Download Operations*

```
a615aTargetOpDownload(char* aMediaPath,
char* aRequestFilePath);
```

For download operations, a simple function is required that takes as input the destination folder path on the ATE system where the files will be stored and the pathname of a file that contains the list of filenames to be downloaded from the avionics system.

### *3) Information Operations*

```
a615aTargetOpInfo();
```

For Information operations a simple function is required with not input to kick off the operation. Once the operation is complete an additional function can be provided to allow the TPS to read the configuration and software version information retrieved from the avionics target.

For all operations, the functions used to initiate should not be blocking functions because data load operations can take several minutes or longer to execute and complete. Therefore a status reading function should also be provided in the data loader API to allow the TPS to poll the library until the operation is completed.

## IV. CONCLUSION

Commercial and Military avionics systems continue to become increasingly more software centric and characterized by standardized, generic hardware with software defined functions. This IMA design concept is well defined and supported by ongoing standardization efforts within the ARINC organization for commercial applications and within the FACE$^{TM}$ consortium for military applications.

As the avionics systems become more software centric, there is an increase in the requirements to be able to execute software updates and loads to the avionics systems both onboard the aircraft and off the aircraft where individual avionics systems components are maintained in bench top situations. Several commercial PDL products exist to support on aircraft software loads. However, in off aircraft situations, the need for these often expensive tools and be eliminated to reduce costs by integrating the PDL functions into existing ATE systems.

## References

[1] Design Guidance For Integrated Modular Avioncs, ARINC Report 651-1, November 7, 1997

[2] Avionics Application Software Standard Interface Part 0 Overview of ARINC 653, ARINC Specifiation 653P0, August 3, 2015

[3] Software Data Loader Using Ethernet Interface, ARINC Report 615A-3, June 30 2007

[4] Technical Standard - Future Airborne Capability Environment (FACETM), Edition 2.1, ISBN: 1-937218-52-2, May 2014