White Paper

A Model-Based Solution for Avionics Systems Engineering: The New Industry Imperative

Thanks to significant advances in electrical and software engineering, today's aircraft have an incredible level of communication, sensing and control functionality. The complexity of a modern display panel makes it easy to see all the information that is being collected and used to inform safe flight.

Navigational systems, radar, braking systems, autopilot, collision avoidance and other technologies must work seamlessly together to ensure consistent, reliable performance and passenger safety. However, because these technologies are provided by diverse suppliers, the task of integrating and managing them has become extremely challenging. The sophistication required to manage system-level engineering today means that manual processes and simple tools, such as Excel spreadsheets, are no longer sufficient. Fortunately, a new model-based approach enables system engineers to quickly build a reliable architecture and generate a fully realized interface control document (ICD) that satisfies all regulatory and technical requirements.

Systems-level Avionics

The cockpit displays of a modern commercial jet would be unrecognizable to the Wright brothers and other aviation pioneers. Every second, thousands of signals are transmitted from communications, navigational, flight and equipment monitoring, and control systems across the aircraft. Critical functions such as braking, ascent and descent, and autopilot are each managed by their own software and controls. With so many systems and signals, how can they all be managed with 100-percent reliability and confidence?

Ensuring robust system management and control is one of the most complex and critical jobs of today's avionics engineers. Creating a well-integrated systems architecture, and expressing that architecture in an ICD, is an incredibly challenging exercise. Many standard protocols must be understood and met, including ARINC 653, ARINC 429, CAN and ARINC 664. With multiple protocols, many sophisticated and safety-critical functions, diverse hardware and software redundancies, and a high level of network complexity — including multiple configuration switches — to manage, generating an ICD is a time-consuming activity.

However, in today's fast-paced aerospace landscape, time is of the essence. And, because systems integration and control represent one of the final steps in assembling a new aircraft, systems-level avionics engineers are under special pressure to complete their tasks rapidly.

Advanced Tools: A Demonstrated Need

While individual avionics technologies have become incredibly advanced, the tools and processes used to architect and integrate these systems have remained labor-intensive and manually based.

An ICD is typically a large Excel® spreadsheet that gathers data and inputs from multiple avionics system suppliers. To produce this document — which details all system hierarchies, interactions, timing and controls — systems-level engineers must ensure that all inputs are not only accurate, but consistent with one another. This requires tedious manual checking and rechecking.

If a modification is made in one component, then that change must be verified against hundreds of other inputs to ensure that overall system integrity is being maintained.

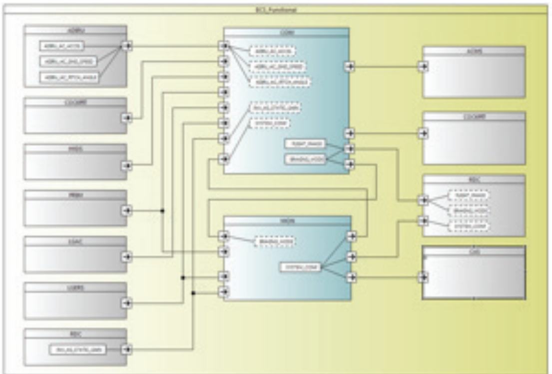
Meanwhile, there is pressure to finish this task quickly, so that the aircraft can be launched confidently. To keep up with the demands of the modern aviation industry, clearly a new, more automated and more intelligent solution is required.

The Benefits of a Model-Based Engineering Solution

What's needed to simplify and accelerate the job of avionics system architects is a flexible, model-based tool that walks them through the various steps of systems integration — and automates any component modifications at the overall system level. By leveraging this technology, the time-to-launch for a new aircraft could be significantly reduced, without sacrificing system quality or reliability.

To make the greatest overall impact, this new model-based solution should have the following characteristics to add the greatest possible value for system architects:

- **Customization for different engineering environments and processes.** Every avionics system supplier works differently. While a standardized, step-by-step approach is needed, a model-based tool would still need to be flexible and customizable, to accept and adapt to diverse inputs from multiple suppliers.
- **Support for different operating platforms and domain-specific languages.** Similarly, the model-based solution must accommodate the broad range of IT platforms and domain-specific programming languages used by systems suppliers from around the world.
- **Easy-to-understand delineation of the multiple layers in an avionics system.** The functional, hardware and software layers of the overall avionics system should be cleanly and visually separated, making all interrelationships clear at a glance.

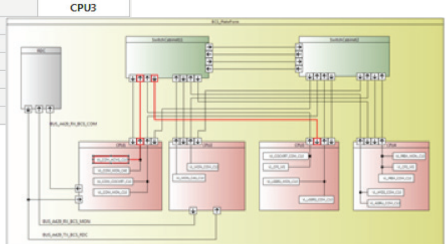


A graphic model of the functional architecture of a braking control system

		A	B
		Block Source	Block Target
22	A429_BCS_LGRDC1_L6	RDC	BCS_COM
23	A429_BCS_LGRDC1_L7	RDC	BCS_COM
24	A429_BCS_LGRDC1_L10	RDC	BCS_COM, BCS_MON
25	A429_BCS_LGRDC1_L11	RDC	BCS_COM, BCS_MON
26	A429_BCS_LGRDC1_L12	RDC	BCS_COM, BCS_MON
27	A429_BCS_LGRDC1_L13	RDC	BCS_COM, BCS_MON
28	A429_BCS_COM_L14	BCS_COM	RDC
29	A429_BCS_COM_L15	BCS_COM	RDC
30	A429_BCS_COM_L16	BCS_COM	RDC
31	A429_BCS_COM_L17	BCS_COM	RDC
32	A429_BCS_LGRDC1_L14	RDC	BCS_COM
33	A429_BCS_LGRDC1_L15	RDC	BCS_COM
34	A429_BCS_LGRDC1_L16	RDC	BCS_COM, BCS_MON
35	A429_BCS_LGRDC1_L17	RDC	BCS_COM, BCS_MON
36	A429_BCS_LGRDC1_L20	RDC	BCS_MON
37	A429_BCS_LGRDC1_L21	RDC	BCS_MON
38	A429_BCS_LGRDC1_L22	RDC	BCS_MON
39	A429_BCS_LGRDC1_L23	RDC	BCS_MON
40	A429_BCS_LGRDC1_L24	RDC	BCS_MON
41	A429_BCS_LGRDC1_L25	RDC	BCS_MON
42	A429_BCS_LGRDC1_L26	RDC	BCS_MON
43	A429_BCS_LGRDC1_L27	RDC	BCS_COM, BCS_MON
44	MSG_ADIRU_COM_C10	ADIRU	BCS_COM
45	MSG_COM_ACMS_C10	BCS_COM	ACMS
46	MSG_COM_COCKPIT_C10	BCS_COM	COCKPIT
47	MSG_COM_MON_C40	BCS_COM	BCS_MON
48	MSG_COM_MON_C10	BCS_COM	BCS_MON
49	MSG_MON_COM_C10	BCS_MON	BCS_COM
50	MSG_COCKPIT_COM_C10	COCKPIT	BCS_COM
51	MSG_HSMU_COM_C10	HYDS	BCS_COM
52	CP1_FW_LG	LG_CP1	BCS_COM
53	MSG_PRIM_COM_C10	PRIM	BCS_COM
54	MSG_PRIM_MON_C10	PRIM	BCS_MON
55	MSG_LGERS_COM_C10	LGERS	BCS_COM
56	MSG_LGERS_MON_C10	LGERS	BCS_MON
57	MSG_MON_CAS_C10	BCS_MON	CAS2_CP_CH1, CAS2_CP_CH2, CAS2_CP_CH3
58	CP2_FW_LG	LG_CP2	BCS_COM
59	A429_BCS_COM_L141	BCS_COM	RDC
60	A429_BCS_COM_L151	BCS_COM	RDC
61	A429_BCS_COM_L161	BCS_COM	RDC

The model-based tool maps the software architecture, via a table showing the software messages exchanged between sources and target software components

		A
		AllocatedTo
1	ACMS	CPU3
2	ADIRU	CPU4
3	BCS_COM	CPU1
4	BCS_MON	CPU2
5	CAS2_CP_CH1	CPU3
6	CAS2_CP_CH2	CPU4
7	CAS2_CP_CH3	CPU3
8	COCKPIT	CPU3
9	HYDS	
10	LG_CP1	
11	LG_CP2	
12	LGERS	
13	PRIM	
14	RDC	



A visual representation of the platform architecture based on software interactions to specific CPUs

- **Protocol-driven templates.** Will the end product, the ICD, address all the required protocols such as ARINC 429? The only way to ensure this is to incorporate pre-defined templates that guarantee all standards are being met as the architecture is designed.
- **Automatic ICD generation.** To optimize speed and efficiency, the solution should be able to automatically gather information from across the model, place it into a hierarchy and quickly produce a robust ICD that meets all industry standards.
- **Development by a proven expert in the global avionics industry.** Because avionics engineering is a highly specialized field, any model-based solution should be developed by a company that understands the intricacies of individual avionics system design, as well as the challenges of systems architecture and integration.

If avionics systems architects had access to a solution with all these features, they would be able to quickly and seamlessly create an integration scheme and an accompanying ICD, at a fraction of the time and cost associated with traditional methods.

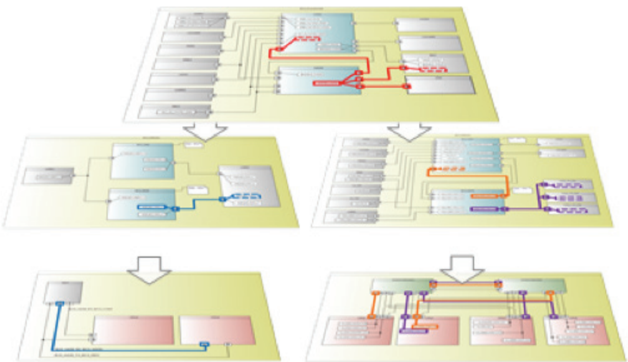
Case in Point: Modeling a Braking System

How exactly would a model-based solution work when applied to a real-world system-level engineering task? As an example, consider an aircraft braking system with a classic command-monitor (COM-MON) architecture. Without delving into the principles behind the COM-MON design, here is a step-by-step look at how a model-based solution would create a system architecture and generate an ICD:

First, the solution would graphically model the **functional architecture**, demonstrating how the braking control system interacts with other functions and exchanges hundreds of functional data inputs.

Next, the model-based tool would map the **software architecture**, via a table showing the software messages exchanged between sources and target software components. In the example to the left, there are 14 individual software components. By revealing how these messages relate to industry protocols, the solution is ensuring compliance — and enabling that compliance to be communicated in the ICD document.

The model-based tool maps these software interactions to specific CPUs to create a visual representation of the **platform architecture**. In the example to the left, four CPUs, two AFDX switches and three ARINC 429 buses form the communications and processing backbone of the braking control system.


The model-based solution allows system architects to verify the flow of all system interactions and check data integrity over the complete application.

		A	B	C	D
		Name	Address	Length	Rate
1	MSG_ADIRU_COM_C10	MSG_ADIRU_COM_C10			
3	MSG_ADIRU_COM_C10	MSG_ADIRU_COM_C10			
5	Res	Res	0	4	
6	FS1	FS1	4	1	
7	FS2	FS2	5	1	
8	FS3	FS3	6	1	
10	DS_ADIRU_AC_GND_SPEED	DS_ADIRU_AC_GND_SPEED	8	4	
11	DS_ADIRU_AC_ACCEL	DS_ADIRU_AC_ACCEL	12	4	
12	DS_ADIRU_AC_PITCH_ANGLE	DS_ADIRU_AC_PITCH_ANGLE	16	4	
14	To_BCS	To_BCS			40

The ICD for compliance with ARINC 429, showing relevant messaging information at the system level.

By visually describing the comprehensive architecture of the entire braking control system, at all levels, the model-based solution allows system architects to verify the flow of all system interactions and check data integrity over the complete application. The groundwork is created for switch configuration, with the assurance that the network has the bandwidth to manage total message volume. System architecture can be synchronized with software design.

Finally, the model-based solution supports automated generation of ICDs that comply with specific avionics protocols. Shown to the left is the ICD for compliance with ARINC 429, showing relevant messaging information at the system level.

Increased Efficiency and Productivity Take Flight

A model-based approach to avionics system management promises enormous improvements in the speed at which ICDs can be generated — and a corresponding increase in staff productivity.

One of the most critical benefits of this model-based approach is its ability to update automatically as system parameters are changed. If a modification is made to a single software component or data point, that change is reflected throughout the model, all the way through ICD generation.

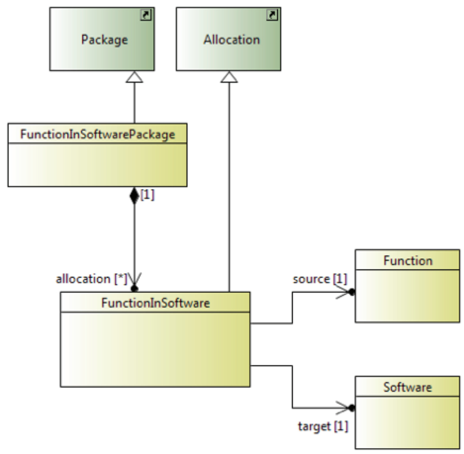
This process of design modification, which currently takes weeks, could be accomplished in mere days via the proposed model-based tool. The efficiency improvement is estimated to be as great as 300 percent. Clearly, today's complex avionics environment requires a new approach that replaces manual labor with rapid system mapping and process automation. The time has come for a new model-based solution that recognizes the evolving needs of modern avionics engineers.

Summary

Today's complex, diverse avionics technologies require new approaches to system management and integration — particularly a more intelligent, more automated solution for mapping the system and generating an ICD. System integration is the final step before aircraft can be launched, and systems-level avionics engineers need to demonstrate their value by streamlining and accelerating their work to the greatest extent possible.

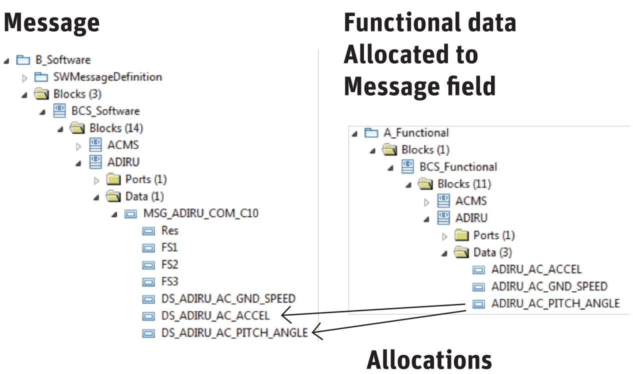
Just as modern aircraft cannot rely on the outdated navigation and communication systems of the past, avionics architects cannot afford to be constrained by outdated, general-purpose tools such as Excel spreadsheets and tedious manual processes.

A new generation of model-based solutions, created specifically for avionics systems-level engineers, will create a significant competitive advantage, deliver added customer value and ensure compliance with all relevant industry protocols — all at a much lower investment of time and resources.



DSL Versus SysML? The Value of a Hybrid Approach

Traditionally, there have been two approaches to systems-level avionics modeling. Many engineering organizations advocate the use of "pure" domain-specific languages (DSLs) that provide greater freedom and programming customization. Other engineering teams have adopted generic, universal languages such as systems modeling language (SysML), a general-purpose modeling language for systems engineering applications, to maximize standardization and interoperability. SCADE System and SCADE Avionics Package take a hybrid approach that offers customers the best of both worlds. While the foundation of SCADE solutions is based on standard SysML, a pure DSL "virtual layer" is implemented transparently on top of SysML as part of SCADE Avionics Package. This hybrid approach offers the benefits of a standard, universal programming language with customized operability suited to the needs of avionics applications.



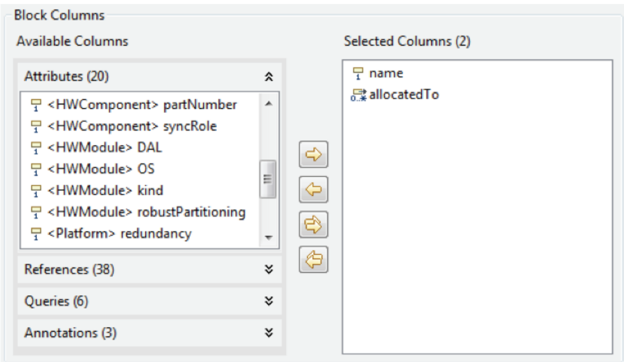
Managing Complex Interfaces: An Engineering Challenge

When designing a "layered" system such as an avionics architecture, the integrity of each layer must be ensured. However, this task is complicated by the fact that these layers are not completely hierarchical. They have overlaps and interdependencies that are difficult to model.

SCADE System and SCADE Avionics Package address this challenge by leveraging a capability in systems modeling language (SysML) called allocations. Allocations allow models to be built in a straightforward manner that recognizes three key facts:

- Every function is delivered by a software component
- Every software component is run by a hardware component
- Every functional data input shared between functions is transported by a software message propagated between software components

By mapping these relationships in SysML — and allocating hardware for each individual software component — SCADE solutions can then enable engineers to build in system dependencies, redundancies and interrelationships. They can determine which software components need to share messages, by creating these linkages to ensure interoperability and compliance with communications protocols such as ARINC 429.



Flexible, Automated Production of Interface Control Documents

Interface Control Documents (ICDs) are required by many different regulatory organizations to ensure that avionics systems are safely and reliably integrated with one another. Currently, system architects are creating customized ICDs for each regulator in a tedious, time-intensive manual process. Now SCADE Avionics Package enables systems integration data to be automatically generated in a variety of formats, very quickly, to satisfy diverse standards and protocols. Once the system model is constructed in SCADE Avionics Package, a variety of reports can be generated, including:

- Partition tables
- Message definitions, with parameters
- Message sources and targets, including ports, transmission rates, message length, etc.
- Virtual link definitions

With a few mouse clicks, SCADE Avionics Package can gather the relevant information for a variety of reporting needs — replacing days of manual data gathering and customized report generation.

ANSYS, Inc.
Southpointe
2600 ANSYS Drive
Canonsburg, PA 15317
U.S.A.
724.746.3304
ansysinfo@ansys.com