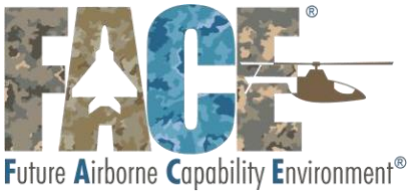

Connecting Open Standards using Modular Open Systems Approach

A perspective on connecting software systems

The Open Group FACE® and SOSA® Consortia 2025 Technical Interchange Meeting (TIM), by:

Andre Odermatt
Real-Time Innovations (RTI)

October 2025

Table of Contents

.

Executive Summary

A Modular Open Systems Approach (MOSA) is a technical and business strategy mandated by the U.S. Department of Defense for designing affordable and adaptable systems through modular design and open standards. These open standards are widely supported, consensus-based specifications that promote interoperability, portability, and scalability across different products and services to facilitate competition, innovation, and cost-effectiveness throughout the system's lifecycle. By using open standards, MOSA aims to create systems with loosely coupled, highly cohesive modules that can be independently upgraded or replaced, avoiding vendor lock-in and enabling rapid technology insertion.

In critical areas like defense, seamless communication between systems is paramount for achieving comprehensive situational awareness and enabling coordinated action. The ability to share real-time information across various sensors, platforms, and command centers provides a unified operational picture, facilitating timely and informed decision-making. This interconnectedness can be the difference between effective response and catastrophic failure.

Connecting open software communication standards can be challenging and often involves defining common data models and standardized APIs to facilitate translation between systems. Protocol bridges or message brokers can manage protocol bridging and data transformation. Employing architectural patterns such as adapters and designing loosely coupled systems further aids interoperability. Ultimately, successful integration relies on clear interface definitions, rigorous testing, and ideally the evolution or profiling of standards to promote convergence and seamless data exchange across disparate systems. This paper provides an example of how FACE® Units of Conformance (UoCs) can communicate with existing applications which are not designed to the FACE Reference Architecture by connecting the Basic Avionics Lightweight Source Archetype (BALSA) applications to the Android Team Awareness Kit (ATAK) and WinTAK (Windows Team Awareness Kit). This approach can be used for a wide variety of use cases where legacy applications, or applications built on different standards, must be integrated.

Implementing a MOSA

The Department of Defense (DOD) is strongly committed to a MOSA for several critical reasons, fundamentally aiming to overcome the limitations of traditional, proprietary defense acquisition. MOSA is seen as vital for accelerating the pace of technological innovation and deployment to the warfighter. In today's rapidly evolving threat landscape, where adversaries are developing new capabilities at an increasing speed, the DOD cannot afford to be locked into multi-decade development cycles. MOSA, with its emphasis on modularity and open standards, allows for quicker integration of new technologies, components, and software, enabling faster upgrades and the rapid fielding of cutting-edge capabilities to maintain a decisive technological edge. This is all part of the current DOD priority to accelerate fielding of new capabilities.

MOSA is a strategic imperative for reducing lifecycle costs and eliminating vendor lock-in. Historically, defense systems have often been designed with proprietary interfaces and components, creating a reliance on a single vendor for upgrades, maintenance, and spare parts. This vendor lock-in often leads to exorbitant sustainment costs and limited flexibility. By mandating open standards and modular designs, MOSA fosters competition among a broader range of suppliers, drives down costs, and allows the DOD to choose the best-of-breed solutions from various sources. This increased competition and ability to incrementally upgrade specific modules rather than replacing entire systems significantly reduces the total ownership cost over the life of a weapon system. In essence, MOSA empowers the DOD to acquire and sustain its systems more affordably and efficiently, ensuring taxpayer dollars are maximized while providing superior capabilities to the nation's defense.

MOSA is guided by five core principles that collectively drive its benefits for the Department of Defense.

- **Establishing an Enabling Environment** emphasizes the need for supportive requirements, business practices, and acquisition strategies that champion open systems development throughout the lifecycle. This includes fostering a culture of collaboration, ensuring proper training, and actively identifying and overcoming potential barriers to MOSA implementation.
- **Employ Modular Design** dictates that systems should be architected with cohesive, encapsulated, self-contained modules, allowing for isolated functionality. This design philosophy simplifies development, maintenance, and upgrades by enabling rapid changes to specific functions with minimal impact on the rest of the system.
- **Designate Key Interfaces** focuses on carefully identifying and managing critical interfaces between modules, prioritizing those that are technologically stable, reliable, and crucial for interoperability, while avoiding the costly and unnecessary management of every single interface.
- **Use Open Standards** is paramount, requiring that these key interfaces adhere to widely supported, consensus-based, and readily available standards, thereby promoting interchangeability, compatibility, and competition among vendors.
- **Certify Conformance** ensures that the compliance with standards is rigorously verified and validated through robust assessment mechanisms, stringent interface control, and proactive conformance testing, guaranteeing that the system and its components truly conform to the open standards for true plug-and-play (hardware and software) capability.

The Real World

In an ideal world a new system is created from scratch which is ideal for a MOSA as it allows you to have a fresh start to create a modular design and define key interfaces. A data-centric architecture is uniquely able to accomplish this goal because it not only allows systems to be decoupled and independent from each other as the application act on a common set of data. A data-centric architecture makes data the common element for systems to act on and consume, while also producing data that other systems can use. A data-centric architecture is an excellent use case for a Publish-Subscribe architecture, based on the open Data Distribution Service (DDS) standard [DDS]. Individual systems declare the intent to subscribe to data or advertise the data they have to offer. The Publish-Subscribe protocol then matches publishers and subscribers with each other and makes sure the data gets to the right place at the right time. New participants can join at runtime without the need for code modifications, enabling rapid updates and technology refreshes using the existing data architecture A data-centric architecture allows to quickly integrate new technologies without overhauling the whole systems. The ability to share real-time information across various sensors, platforms, and command centers provides a unified operational picture, facilitating timely and informed decision-making. System refreshes often end up having to interface with communication standards used in legacy systems or even proprietary communication protocols.

Ideally when starting a software architecture from scratch you would follow the FACE® Reference Architecture [C232] to create a system which is modular and follows MOSA. Unfortunately, not many projects have the luxury of starting from scratch. Most efforts are upgrades and technology refresh. For example, the F-15 was introduced in 1972. Not a lot of the software systems on board of an F-15 are the same as in 1972. There have been periodic technology refresh cycles. The same will be true with the latest aircraft. The airframes will be around for much longer than the software on it - one reason why MOSA is in place to help upgrade and adopt the software quickly to adapt to the changing requirements. The application of MOSA to existing systems is inherently a modernization strategy.

How to make Software Systems talk to each other

The Data Distribution Service (DDS) standard, managed by the Object Management Group (OMG) [DDS], is designed with interoperability and adaptability in mind, allowing it to seamlessly integrate with and support a wide array of other standards and technologies. This ability to "play well with others" is a key reason for its widespread adoption in complex, real-time, and mission-critical systems. At its foundation, DDS defines both a data-centric publish-subscribe API and a Real-Time Publish-Subscribe (RTPS) wire protocol [DDSI-RTPS]. The RTPS protocol is crucial because it ensures that different vendors' implementations of DDS can directly communicate and exchange data, providing fundamental cross-vendor interoperability at the network level.

Beyond its core specifications, DDS has been extended and adopted by numerous other industries and domain-specific standards to fulfill specific communication needs. For instance, in the realm of defense and aerospace, DDS is a foundational component of initiatives like the Sensor Open Systems Architecture™ (SOSA®) Technical Standard [C239] and FACE® (Future Airborne Capability Environment®) Technical Standard [C232]. These open architecture standards leverage DDS for their underlying data transport layer, ensuring that sensor data, mission computing information, and other critical avionics data can flow seamlessly between modules from different suppliers. Similarly, in the automotive industry, DDS is increasingly used within the AUTOSAR Adaptive.

Furthermore, DDS extends its reach to resource-constrained environments and high-level service interactions. The DDS-XRCE (DDS for eXtremely Resource Constrained Environments) specification [DDS-XRCE] allows lightweight devices like microcontrollers to interact with a full DDS network, expanding its applicability to the very edge of the Internet of Things. For higher-level service calls, DDS-RPC (RPC over DDS) [DDS_RPC] defines a Remote Procedure Call framework that leverages DDS's publish-subscribe capabilities for request-reply patterns. DDS also offers integrations with other significant standards like OPC UA (Open Platform Communications Unified Architecture) [DDS-OPC] for industrial automation, Automotive Open System Architecture (Autosar) [AUTOSAR], Robot Operating System 2 (ROS 2) [ROS2], Tactical Microgrid Standards Consortium (TMSC) [TSMC], Unmanned Maritime Autonomy Architecture (UMAA) [UMAA]. This extensive web of interconnected standards demonstrates DDS's versatility and its role as a robust, open-standard backbone for heterogeneous, distributed systems. Figure 1 shows some of the standards where DDS is used.

Figure 1: Examples of standards which use DDS

Connecting Open Standards using Modular Open Systems Approach

Integrating different communication protocols is a common challenge in modern distributed systems, as it's rare for all components to speak the exact same language. The primary goal of such integration is to enable seamless data exchange and interaction between disparate systems. One common approach is to use protocol converters or gateways. These are specialized software or hardware components that sit between systems using different protocols. They act as translators: receiving data in one protocol, converting it to a neutral or target format, and then retransmitting it using the other protocol. For example, a gateway might convert Modbus data from an industrial sensor into Message Queuing Telemetry Transport (MQTT) messages for a cloud IoT platform or translate a proprietary legacy protocol into a standardized Application Programming Interface (API) call. This method effectively isolates each system from the complexities of the other's communication method, allowing them to remain largely unchanged.

Another powerful strategy involves using a middleware layer or an Enterprise Service Bus (ESB). This approach introduces an intermediate software layer that understands multiple protocols. Instead of direct point-to-point conversions, systems publish messages or data to the middleware using their native protocol, and the middleware then translates and routes these messages to the appropriate consuming systems, which retrieve them using their own preferred protocols. This creates a flexible, decoupled architecture where systems don't need to know about each other's communication specifics; they only need to understand how to interact with the middleware. Examples include Apache Kafka for high-throughput messaging, or dedicated ESB platforms for more complex message routing, transformation, and orchestration across various enterprise applications.

Finally, for tightly coupled integrations or situations where real-time performance is paramount, APIs combined with adapter patterns are frequently employed. While APIs define how systems interact, an adapter pattern specifically helps integrate components with incompatible interfaces. An adapter for a legacy system might wrap its unique protocol and expose its functionalities through a modern REpresentational State Transfer (RESTful) API. Similarly, in a DDS environment, a "routing service" or "gateway" can be configured to translate between DDS data models and other protocols like TCP/IP or even other DDS domains. This allows specific modules to "speak" their native protocol internally while communicating with the broader system through standardized, higher-level interfaces defined by the API or the adapter layer. The choice of method depends on factors like latency requirements, data volume, system complexity, and the extent of decoupling desired.

Connecting Systems

As mentioned in most cases different systems need to be connected with each other. Often this means that the different systems have different data representation or data models. The required information is available but not always in the same format. Like people speaking different languages you need a translator to relay information back and forth.

RTI Connext Routing Service is an out-of-the-box software solution from Real-Time Innovations (RTI) designed to tackle the complex challenges of real-time data exchange in highly distributed systems. At its core, it acts as an intelligent data bridge, enabling DDS applications to seamlessly communicate even when they are separated by different DDS domains, network boundaries including Local Area Networks (LAN) and Wide Area Networks (WAN) with firewall and Network Address Translation (NAT) traversal, or even disparate data models. This capability is paramount for constructing complex "systems-of-systems" where numerous independent applications need to efficiently share and react to information, common in industrial IoT, robotics, defense, and autonomous vehicles.

Beyond simply forwarding data, the Routing Service offers sophisticated data transformation and filtering capabilities. It can modify data values, change data types, and apply content-based filtering on the fly. This means that if a publishing application uses one specific data structure or topic name and a subscribing application expects another, the Routing Service can perform the necessary conversions, ensuring interoperability without requiring changes to the applications themselves. This flexibility is crucial for integrating diverse components, including new DDS-based applications with existing legacy systems, or for managing the evolution of data models within a large system without breaking compatibility.

The deployment of RTI Connext Routing Service is one of the leading products in today’s market that brings significant benefits, particularly in accelerating system integration and enhancing scalability. By abstracting away the complexities of network topology and data format differences, it drastically reduces the effort required to connect independently developed subsystems. This directly supports the principles of MOSA by enabling truly plug-and-play integration of modules. Its ability to bridge data securely and efficiently across various network conditions, from local multicast to secure WAN connections, ensures that even the largest and most geographically dispersed systems can maintain real-time performance and reliability.

A Simple Example

Overview

The following is a simple example demonstrating the integration of different components using different communication or data models. This is a simple example using the FACE consortium example.

While real-world implementations exist, many are not publicly documented and therefore we have a simple example here using the FACE Consortium example BALSA, the windows version of the Team Awareness Kit (TAK) [TAK], and an open-source flight simulator (FlightGear) [FGFS]. This example shows how legacy systems with proprietary protocols (FlightGear) can connect to FACE UoCs (BALSA) and applications like TAK which are not aligned to the FACE Technical Standard.

All three applications are well known in their domain but are not designed to work together. Making use of the latest technology and building systems at a lower cost is faster than building a new system from scratch. It also allows for replacement of individual components. In this example FlightGear Flight Simulator could easily be replaced by a different flight simulator (e.g. Microsoft Flight Simulator, Lockheed Martin Prepare3D) without any changes to the rest of the system. In essence supporting the MOSA objective of fielding new technologies and features quicker.

As discussed earlier, to integrate different systems that use different messaging protocols a common bus with interfaces to the different protocols is needed. A common wire protocol makes it easy to add additional applications. DDS is a prime candidate for such a common databus as it already is the protocol used in many standards. There are many tools available to bridge different protocols or for data translation within DDS.

BALSA

The Basic Avionics Lightweight Source Archetype (BALSA) is a working example of integrated applications aligned to the FACE Technical Standard executing in a FACE Reference Architecture. It is an implementation of the FACE Reference Architecture that features Portable Component Segment (PCS) and Platform Specific Service Segment (PSSS) component applications that demonstrate a basic avionics process. The example consists of multiple UoCs, the Embedded GPS/Inertial Navigation System (EGI) Controller and Aircraft Config are PSSS sources that output into the Air Traffic Controller (ATC) Manager PCS which combines the two messages and outputs the result to the Automatic Dependent Surveillance – Broadcast (ADS-B) ADSB-Out PSSS into UDP packets through the I/O Services Segment (IOSS) layer. The red arrows in Figure 2 show the BALSA message flow.

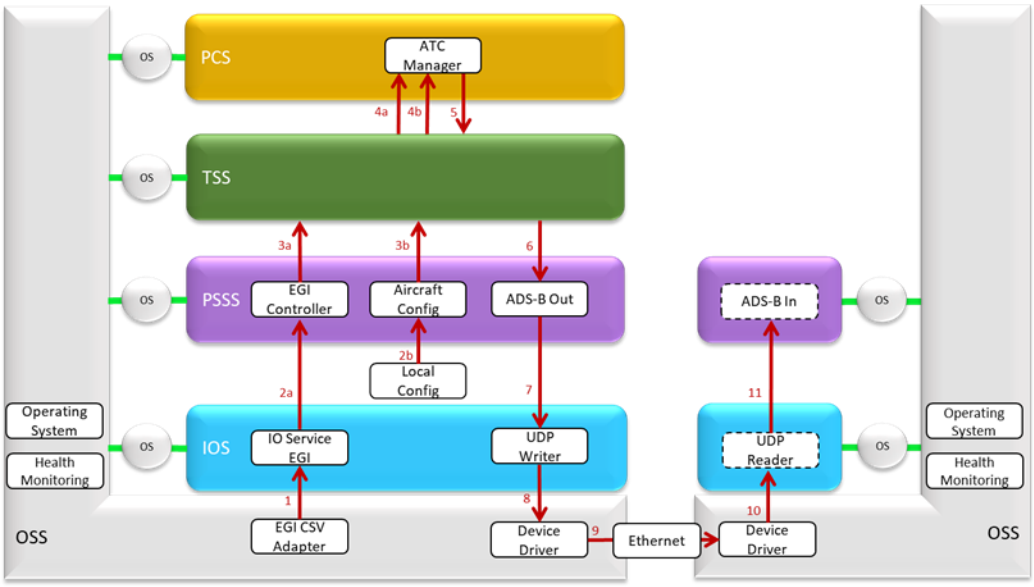

Figure 2: BALSA Message Flow

The EGI Controller UoC periodically reads position information from a local csv file and publishes the information through the TSS. ADS-B Out receives the combined message from ATC and sends it over UDP to the ADS-B In UoC.

Team Awareness Kit (TAK)

For this example, we used the civil version of the Tactical Assault Kit called Team Awareness Kit (TAK) which is a family of geospatial and situational awareness software applications designed for collaborative operations, primarily for military, government, and public safety use. At its core, TAK provides users with a common operating picture, enabling real-time sharing of location, data, and communication. ATAK (Android Team Awareness Kit, or Android Tactical Assault Kit for military) is the most widely known iteration, running on Android smartphones and tablets, allowing for features like precision targeting, navigation, and data exchange in the field. WinTAK is the Windows counterpart, offering similar functionalities on a desktop or laptop environment, often used in command centers to exchange data with field users. Both ATAK and WinTAK are extensible through a plugin architecture, allowing developers to add specialized functionalities for various missions and needs.

FlightGear

FlightGear is a free and open-source flight simulator project that has been under continuous development by a global community of volunteers since 1997. Unlike commercial flight simulators, FlightGear prioritizes realism, flexibility, and extensibility, offering a highly customizable experience for flight enthusiasts,

researchers, and educators. It boasts a massive database of over 20,000 real-world airports, a wide array of aircraft ranging from light planes to commercial airliners and helicopters, and advanced features like realistic flight dynamics, detailed weather modeling, and global scenery based on real-world topographic data. Its open-source nature means that users can not only download and use it for free, but also contribute to its development, create new aircraft, and expand its capabilities through a robust plugin system.

Putting it all together

In this example we use RTI Connext TSS, a DDS based FACE Certified Cconformant TSS implementation. Since FlightGear uses a proprietary protocol which is calls the "generic communication protocol" by FlightGear and communicates over UDP a converter to DDS is needed. Integrating a proprietary protocol with FACE Conformant UoCs is a common use case for integrators building MOSA aligned systems. Converting protocols is simple with the Connext Routing Service which converts the FligthGear UDP messages to a DDS topic. The mapping between FlightGear properties and DDS data elements is configured through Extensible Markup Language (XML). The XML configuration allows to map any of the properties internal to FlightGear to elements in one or more DDS topics.

The BALSA example provides its own implementation of an example TSS. Thanks to the FACE Technical Standard which defines the APIs to the TSS layer. The TSS can easily be replaced by linking to a different TSS library. For this exercise BALSA is built using RTI Connext TSS. RTI Connext TSS is built on top of the RTI DDS implementation.

Instead of using the BALSA EGI PSSS to feed data from a csv file, data is directly feed from the FlightGear flight simulator. The Connext Routing Service together with the FlightGear plugin maps the position properties from the flight simulator into the EGI Data message. The EGI Data message is sent through the common data bus to the BALSA ATC manager PCS. For the ATC manager the incoming data is formatted as if the data comes from the EGI PSSS. There is no change needed to the ATC PCS.

The TAK Connext plugin subscribes to data type which matches the TAK CoT format. The TAK CoT format is different from the ATC data type. Therefore, routing services are needed to transform from ATC Data to TAK CoT Data. ATC Data is very simple and contains the aircraft id, tail number, latitude, longitude, and altitude. The TAK CoT event [TAK-CoT] is a bit more complex but among other information contains a World Geodetic System 1984 (WGS84) [WGS84] position information. Figure 3 shows an overview of how the different components are integrated together.

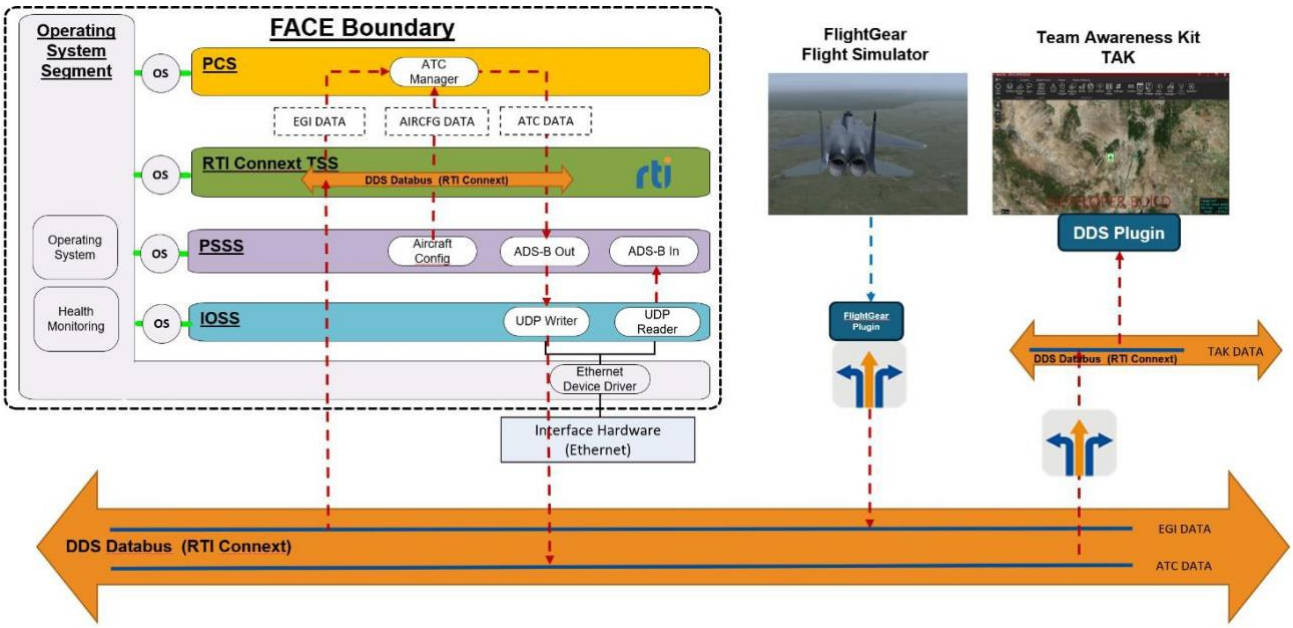

Figure 3: Architecture diagram

This is just a simple example of how different systems can be integrated together.

Conclusion

Connecting open software communication standards can be challenging and often involves defining common data models and standardized APIs to facilitate translation between systems. Protocol bridges or message brokers can manage protocol bridging and data transformation. Employing architectural patterns such as adapters and designing loosely coupled systems further aids interoperability. Ultimately, successful integration relies on clear interface definitions, rigorous testing, and potentially the evolution or profiling of standards to promote convergence and seamless data exchange across disparate systems. This paper shows an example of how FACE Units of Conformance (UoCs) can communicate with existing applications which are not designed to the FACE Reference Architecture by connecting the BALSA applications to ATAK and WinTAK. This is just one example of how legacy applications or applications built on different standards can be integrated. This approach can be used for a wide range of use cases.

For today's systems it a seamless communication between is critical in for achieving comprehensive situational awareness and enabling coordinated action. The ability to share real-time information across various sensors, platforms, and command centers provides a unified operational picture, facilitating timely and informed decision-making. This interconnectedness can be the difference between effective response and catastrophic failure. As shown, having a common protocol is crucial to achieving integration and interoperability of systems.

Referenced Documents

(Please note that the links below are good at the time of writing but cannot be guaranteed for the future.)

C232	FACE® Technical Standard, Edition 3.2 (C232), published by The Open Group, August 2023; refer to: www.opengroup.org/library/c232
C239	Technical Standard for SOSA® Reference Architecture, Edition 1.1 (C239), published by The Open Group, June 2023; refer to: www.opengroup.org/library/c239
DDS	Data Distribution Service (DDS), Version 1.4 published by The Object Management Group, March 2015; refer to: https://www.omg.org/spec/DDS
DDSI-RTPS	DDS Interoperability Wire Protocol (DDSI-RTPS), Version 2.3, published by The Object Management Group, May 2019; refer to: https://www.omg.org/spec/DDSI-RTPS
DDS-XRCE	DDS For Extremely Resource Constrained Environments, Version 1.0, published by The Object Management Group, February 2020; refer to: https://www.omg.org/spec/DDS-XRCE
DDS-RPC	RPC Over DDS, Version 1.0, published by The Object Management Group, April 2017; refer to: https://www.omg.org/spec/DDS-RPC
DDS-OPCUA	OPC-UA/DDS Gateway, Version 1.0, published by The Object Management Group, January 2020; refer to: https://www.omg.org/spec/DDS-OPCUA
AUTOSAR1027	Integration of DDS Security (1027), AUTOSAR Adaptive, R22-1, published by AUTOSAR, November 2022; refer to: https://www.autosar.org/fileadmin/standards/R22-11/AP/AUTOSAR_TR_DDSSecurityIntegration.pdf
ROS2	Robot Operating System 2: Design, architecture, and uses in the wild, Science Robotics vol. 7, May 2022; refer to https://www.science.org/doi/10.1126/scirobotics.abm6074
UMAA	Unmanned Maritime Autonomy Architecture (UMAA) Engineering Operations (EO) Interface Control Document (ICD), Version 4.2.1, February 2021; refer to: https://www.auvsi.org/sites/default/files/PDFs/UMAA/UMAA-SPEC-EOICD.pdf
TAK	TAK Overview, https://tak.gov/
TAK-CoT	De-mystifying the tak protocol, blog, https://www.ballantyne.online/de-mystifying-the-tak-protocol/
WGS84	World Geodetic System 1984 (WGS 84) by the Office of Geomatics, https://earth-info.nga.mil/index.php?dir=wgs84&action=wgs84
TSMC	Overview of MIL-STD-3071 – Tactical Microgrid Standard, published by Frank W Bohn, U.S. Army Combat Capabilities Development Command, C5ISR Center, APG, MD, https://powersourcesconference.com/PowerSources23/docs/P-10.pdf
FGFS	FlightGear open-source flight simulator, release 2024.1.1, https://www.flightgear.org/

About the Author

Andre Odermatt is a principal application engineer for Aerospace and Defense at Real-Time Innovations (RTI). Before joining the Marketing group at RTI, Andre was a field application engineer at RTI for five years, working closely with customers on distributed systems. He has over 30 years of experience with embedded systems, distributed applications, and communications protocols. ndre is an active participant in the FACE Consortium TWG Transport Services Segment (TSS) technical working group, and currently Chair of the FACE Integration Workshop (IWS) standing committee.

About The Open Group FACE® Consortium

The Open Group Future Airborne Capability Environment® Consortium (the FACE® Consortium), was formed as a government and industry partnership to define an open avionics environment for all military airborne platform types. Today, it is an aviation-focused professional group made up of industry suppliers, customers, academia, and users. The FACE Consortium provides a vendor-neutral forum for industry and government to work together to develop and consolidate the open standards, best practices, guidance documents, and business strategy necessary for acquisition of affordable software systems that promote innovation and rapid integration of portable capabilities across global defense programs.

Further information on the FACE Consortium is available at www.opengroup.org/face.

About The Open Group SOSA® Consortium

The Open Group SOSA® Consortium enables government and industry to collaboratively develop open standards and best practices to enable, enhance, and accelerate the deployment of affordable, capable, interoperable sensor systems. The SOSA Consortium is creating open system reference architectures applicable to military and commercial sensor systems and a business model that balances stakeholder interests. The architectures employ modular design and use widely supported, consensus-based, nonproprietary standards for key interfaces.

Further information on the SOSA Consortium is available at www.opengroup.org/sosa.

About The Open Group

The Open Group is a global consortium that enables the achievement of business objectives through technology standards and open source initiatives by fostering a culture of collaboration, inclusivity, and mutual respect among our diverse group of 900+ memberships. Our membership includes customers, systems and solutions suppliers, tool vendors, integrators, academics, and consultants across multiple industries.

The mission of The Open Group is to drive the creation of Boundaryless Information Flow™ achieved by:

- Working with customers to capture, understand, and address current and emerging requirements, establish policies, and share best practices
- Working with suppliers, consortia, and standards bodies to develop consensus and facilitate interoperability, to evolve and integrate specifications and open source technologies
- Offering a comprehensive set of services to enhance the operational efficiency of consortia
- Developing and operating the industry's premier certification service and encouraging procurement of certified products

Further information on The Open Group is available at www.opengroup.org.