2019 NDIA GROUND VEHICLE SYSTEMS ENGINEERING AND TECHNOLOGY SYMPOSIUM
VEHICLE ELECTRONICS AND ARCHITECTURE (VEA) AND GROUND SYSTEMS CYBER ENGINEERING (GSCE) TECHNICAL SESSION
AUGUST 13-15, 2019 - NOVI, MICHIGAN

Potential for VICTORY and FACE™ Alignment – Initial Exploration of Data Interoperability and Standards Compliance / Conformance

Leonard Elliott
U.S. Army Combat Capabilities Development Command (CCDC)
Ground Vehicle Systems Center (GVSC)
Warren, MI

Syltinsy P. Jenkins
U.S. Navy Naval Aviation Systems (NAVAIR)
Air Combat Electronics (PMA 209)
The MITRE Corporation
Patuxent River, MD

Michael S. Moore, PhD
Southwest Research Institute
San Antonio, TX

Howell S. Yee, PhD
The MITRE Corporation
Bedford, MA

ABSTRACT

The Vehicular Integration for C4ISR/EW Interoperability (VICTORY) and Future Airborne Capability Environment (FACE) Standards are two open standards that support Modular Open System Approaches (MOSA) to U.S. Department of Defense (DoD) weapon system development and acquisition. Both standards share similar high-level goals (e.g. interoperability, lower integration costs, and open competition). Due to differences in the business goals and application environments, the technical objectives were significantly different. The airborne avionics business and application space led the FACE™[1] approach to define standards to make software portable and independent of the existing safety-critical and real-time system architectures in various airborne platforms. The FACE Technical Standard defines software application program interfaces and architectures for a flexible, operating environment to host platform independent software components. The ground vehicle environment led VICTORY to define standards to create interoperability where there was none previously. VICTORY defines standards for an in-vehicle network and on-the-wire network interfaces to integrate C4ISR/EW equipment and interface to vehicle systems. VICTORY and FACE define different kinds of standards because they had different objectives. This paper explores methods that could be used to integrate systems using the two standards, and suggests areas in which alignment is possible and may be mutually beneficial.

Citation: Leonard Elliott, Syltinsy P. Jenkins, Howell S. Yee, PhD, Michael S. Moore, PhD, "Potential for VICTORY and FACE Alignment – Initial Exploration of Data Interoperability and Standards Compliance / Conformance", In *Proceedings of the Ground Vehicle Systems Engineering and Technology Symposium* (GVSETS), NDIA, Novi, MI, Aug. 13-15, 2019.

[1] FACE™ is a trademark of The Open Group.

1. INTRODUCTION

In the realm of Open System Architecture and the Modular Open System Approach (MOSA), the U.S. Army and U.S. Navy have both developed open standards to meet their specific technical needs and acquisition goals. The Vehicular Integration for C4ISR/EW[2] Interoperability (VICTORY) standard was developed by the U.S. Army for military ground vehicles, while the U.S. Navy conceived the Future Airborne Capability Environment (FACE) Technical Standard for hosting portable software in airborne platforms.

The FACE Technical Standard is developed and maintained by the The Open Group FACE Consortium. Government members including the U.S. Navy Naval Air Systems Command (NAVAIR) Air Combat Electronics program office (PMA-209); Army Program Executive Office (PEO) Aviation (AVN); the Army's Aviation and Missile Research, Development, and Engineering Center (AMRDEC); and the Air Force Life Cycle Management Center (AFLCMC), in conjunction with industry contributors provide sponsorship and leadership for the FACE Consortium to develop the FACE Technical Standard and its supporing artifacts.

VICTORY is developed and maintained under the sponsorship of the Army PEO Ground Combat Systems; PEO Combat Support and Combat Service Support (CS&CSS); Army Combat Capabilities Development Command (CCDC) Ground Vehicle Systems Center (GVSC), Command, Control, Communications, Computers, Combat Systems, Intelligence, Surveillance, and Reconnaissance (C5ISR) Center; and other Army PEO organizations, supported by a broad community of industry partners in the VICTORY standards body, and managed by the VICTORY Standards Support Office (VSSO).

The VICTORY and FACE technical standards have both applied open system architecture concepts to challenges that exist in the U.S. DoD Acquisition processes and have been developed and maintained in alignment with MOSA. As will be discussed in this paper, even with the different technical approaches, there are some commonalitities and opportunities for alignment between VICTORY and the FACE Technical Standard.

Note that there are other MOSA initiatives, such as Open Mission Systems (OMS), Sensor Open System Architecture (SOSA), Unmanned Systems (UxS) Control Segment (UCS) Architecture, and many others, all of which take various technical approaches based on the relevant business and technical drivers. There are likely commonalities with and between these efforts that offer opportunities for reuse and alignment. However, this paper focuses on VICTORY and the FACE Technical Standard.

As will be shown, both the VICTORY and FACE architectures apply suitable technical approaches derived from the overall goals and technical context.

VICTORY specifies standards for a vehicle network and network-based interfaces to create interoperability between systems and system components. The FACE Technical Standard defines a Reference Architecture intended for the development of portable software components targeted for general purpose, safety, and/or security purposes [1]. The FACE Technical Standard specifies a layered software operating environment, data architecture, and reference architecture framework that supports portable software components on a wide range of implementations.

Technically the differences are clear, as are the business needs that drove the choices. But the billion-dollar questions asked by leadership include:

- "Why should the government invest in and maintain two architectures, and are they worth the investment?"
- "Is it possible for capabilities implemented in one architecture to be integrated with those from another, and what would be the level of effort?"
- "Are there efficiencies or technical benefits to be achieved by somehow aligning these architectures, and what are the costs and benefits?"

This paper first provides a high-level comparison of the VICTORY and FACE iniatives, concentrating on the business drivers, technical objectives, and artifacts, and why they are different. It will make the case that each architecture is a reasonable solution to its business need, but a single technical architecture could not achieve both sets of goals. This will provide an answer to the first question.

It then discusses technical approaches that could be used to integrate FACE conformant and VICTORY compliant implementations. This will suggest answers to the second question. Note that this analysis will describe the technical approaches and costs (how it could be done), but will not attempt to answer whether it should it be done.

The paper will investigate areas in which VICTORY and the FACE Technical Standard conceptually or technologically overlap, present areas in which there is a potential for commonality between the architectures and discuss the avenues for capitalization on the commonalities. This will provide a partial answer to the third question, but will stop short of attempting to completely characterize the costs and benefits of such an effort. Technical areas that will be explored include interface and data definitions (data models), security requirements, and verification of compliance/conformance with the standards.

[2] Command, Control, Communications, Computers, Intelligence, Surveillance, Reconnaissance and Electronic Warfare (C4ISR/EW)

Finally, the paper will suggest a potential path forward for aligning VICTORY and the FACE Technical Standard, including actions that the authors feel would provide a suitable return on investment.

2. HIGH LEVEL COMPARISON

This section will cover some of the general information about the respective standards, the business drivers, technical objective and available artifacts.

2.1. General

VICTORY defines standards for an Ethernet-based in-vehicle network framework and network messaging interfaces. VICTORY completely defines the syntax and semantics of network messages as they will exist on-the-wire so that physical components developed by different vendors and for different systems will interoperate when plugged into the same network. This approach was practical because VICTORY was able to define the vehicle network itself, which did not exist in most ground vehicles. VICTORY was able to specify that the vehicle network adopt ubiquitous networking technologies such as Ethernet, Internet Protocol (IP), Transmission Control Protocol (TCP), User Datagram Protocol (UDP), and Internet Group Management Protocol (IGMP). Having the freedom to choose the underlying network technologies greatly simplifies the problem of network interface standardization. VICTORY was also able to completely define the messaging interfaces; the message exchange protocol, contents, parameter semantic, syntax, encoding, and encapsulation. Fully specifying on-the-wire network interfaces reduced the number of choices that must be made when integrating a system (the design space). Since the systems had not been been designed to interoperate within and with the vehicles, VICTORY had more control to specify and thus make assumptions about the network interfaces. VICTORY was also able to assume that safety-critical systems, such as fire-control, were outside of its scope.

Compared with that of VICTORY, the FACE approach addresses a far more challenging problem space for software portability due to the fact that airborne platforms already had networks that integrated the electronic systems, as well as computing platforms that hosted the avionics software. These network and processing architectures are different across the vehicle platforms. Additionally, the FACE Technical Standard was designed to accommodate safety and time-critical systems related to airworthiness.

The FACE approach solved the problem of standardizing a software common operating environment (COE) that would map to a wide range of underlying processing and network architectures, runtime environments, and messaging protocols to support software portability across embedded systems. This problem space led to the development of the FACE Technical Standard which specifies a segmented, layered software architecture with standard application program interfaces, in which the layers, and components within the segments could be replaced to work in the different vehicle implementation architectures.

Another aspect of the problem space of software portability and component interoperability was vendor-lock and the use of proprietary interfaces. Traditionally, sofware components were custom built to the platform using proprietary interfaces with limited reuse. To address this problem, the FACE approach defined key interfaces, a data modeling language, and data architecture which would be agnostic to the platform and the vendor.

Instead of defining domain specific data content (data structures, parameter semantic, encoding, etc.) for the interfaces at each segment, the FACE approach created a generalized data modeling language and framework with which programs are to define vehicle and application specific data types. The software that implements the binding of application data to the underlying interface technologies is generated from FACE Data Models which include Unit of Portability (UoP) Supplied Data Models (USMs), Domain Specific Data Models (DSDMs), and integration models.

2.2. Governance and Standards

The VICTORY Standards Working Group started in the 2010 timeframe. The stewardship of the VICTORY standard falls under purview of the VICTORY Standards Support Office [2].

VICTORY defines a network architecture and messaging standards to reduce size, weight, and power (SWaP) claims and to reduce schedule and cost of integrating new capabilities in the electronic systems inside ground vehicles. The top objectives are defining standards for *network infrastructure* and *messaging interfaces* to promote *interoperability* between components and subsystems. The scope is integrating non-time-critical systems within ground vehicles. VICTORY concentrates on interoperability between elements connected by networks, standardizes network messaging interfaces. Figure 1 depicts a high-level view of the VICTORY architecture [3].

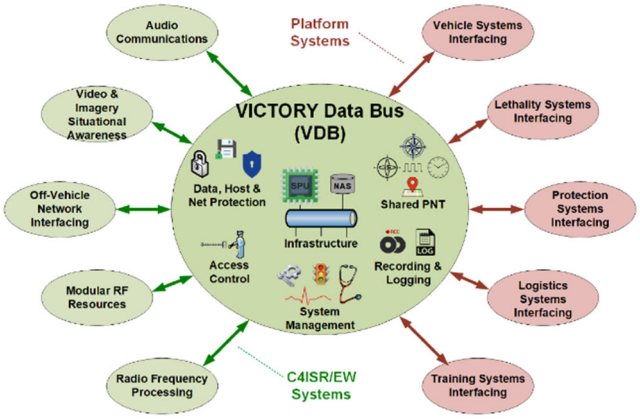

Figure 1. VICTORY Architecture

The FACE Consortium was also formed in the 2010 timeframe. The stewardship of the FACE Technical Standard falls under purview of The FACE Consortium. The FACE Consortium is a voluntary consensus standards body[3] operated under the auspices of The Open Group and is comprised of over 90 member organizations in an industry and government collaborative partnership [4].

The FACE Technical Standard defines a software COE designed to promote portability and create software product lines across the military aviation domain [1]. The FACE Technical Standard defines a layered software architecture and applies model-based techniques to promote *software portability*. The objectives of The FACE Technical Standard are to define the FACE Reference Architecture for developing and verifying software components, defining interfaces allowing communication between software components, and to enable affordability, interoperability, and time-to-field across military systems based upon fundamental software engineering principles and practical experience [1]. The top FACE Reference Architecture defines a *vertically layered software architecture* with standardized *application program interfaces (APIs)*, to abstract interfaces between software applications and the underlying runtime and communications resources to achieve *portable software*, and to provide model-based languages and tools for data modeling and code generation to reduce *software development cost*. The scope is the ability to implement portable *safety-critical*, *time-critical,* and *general purpose* software in *airborne vehicles*. The focus of the FACE approach is on portability of software components in an *operating environment*, so standardizes software APIs and modeling languages. Figure 2 depicts the layered FACE Architectural Segments.

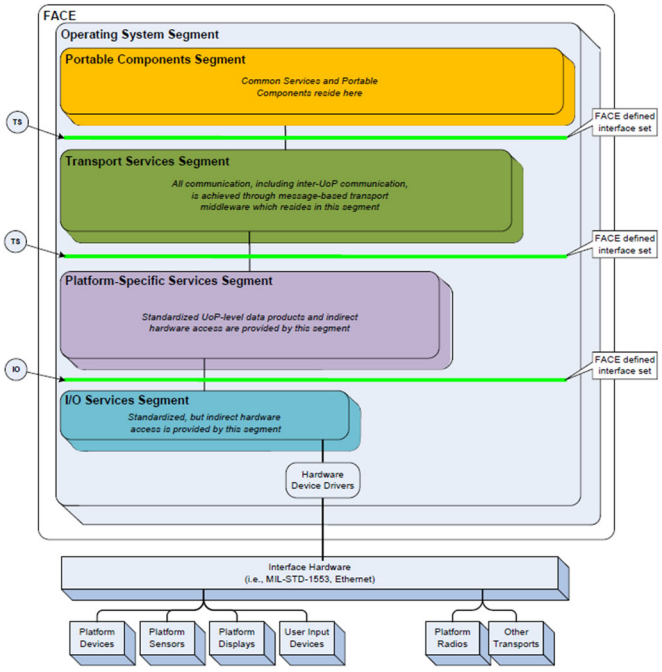

Figure 2. FACE Architectural Segments [1]

2.3. Business Drivers

Figure 3 depicts the business drivers for the FACE Technical Standard. These high-level business drivers apply to the VICTORY standard as well because the U.S. Army and U.S. Navy are both driven by the same policies and desire to continually field technically advanced and affordable capabilities to the warfighters.

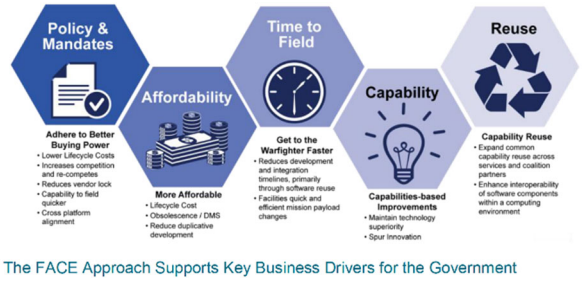

Figure 3. FACE Business Drivers [4]

Further analysis of these business drivers is required to gain clarity. For instance, in order to affect affordability and time to field, it is necessary to understand where the "pain points" (cost and schedule drivers) are in the business process. The

[3] Voluntary consensus standards body as defined by Office of Management & Budget (OMB) Circular A-119

business drivers must also be evaluated in order to identify *what* should be standardized and mandated through policies.

The next level of detail in the FACE business drivers identified software development and integration as the major cost and schedule drivers of programs. Because of the different implementation environments on airborne platforms, in the past the government often had to pay for the same capability to be developed in software separately for multiple airborne platform programs. Further analysis led to the objective of more portable software that can be reused across airborne platforms, and the need for a layered and segmented common operating environment that can be integrated using different computing and network architectures. The segments decouple and separate portable components from the data transfer technologies and network architectures.

The VICTORY business goals, as paraphrased from the VICTORY architecture [3], are:

- Eliminate the practice of "Bolt On" systems.
- Reduce the SWaP and system cost impact of adding electronic systems to vehicles.
- Simplify integration, enhance interoperability, increase capabilities for the warfighter, and reduce life cycle costs.
- Maximize C4ISR/EW portability by defining open interface standards, data formats, and protocols that can be used by vehicle communities.
- Support current and future Information Assurance (IA) requirements.
- Provide an evolutionary approach towards network-centric C4ISR/EW, starting with interoperability with current systems, and providing a pathway for insertion of new capabilities and technologies.

The VICTORY goals are somewhat more specific than those of FACE approach. However, these goals are truly derived from similar high-level drivers, such as the need for affordability, shorter time to the field, and the ability to insert new capabilities.

Further analysis of the VICTORY business drivers identified integration of government-furnished equipment, theater specific capabilities, and new sensors with vehicles as a major cost and schedule driver. The concern with reducing the SWaP impact of the electronic systems derives from the vehicle integration business model. Since vehicles were not required to provide common computing or network resources, each system was designed to include its own computing and user interface devices and interface wiring. The effect is that integrating systems onto vehicles quickly overwhelms the SWaP available on the platform.

2.4. Technical Objectives

The first technical objective of the FACE approach was to make software more portable. The approach to portability was to identify and define the maximally portable components, to resolve dependency upon infrastructure and external components, and to provide a COE to host the portable components and the infrastructure. The result is a segmented and layered software operating environment to allow software within each segment to be independent of the implementation technologies of the layers and segments below. The FACE architectural diagrams reflect clearly the concepts found within a software architecture stack.

Another technical objective of FACE was to reduce the cost of integrating the components within the segments with the underlying processing, network, and middleware implementations through the use of modeling and code generation. This is one of the keys to making it cost-effective to enable interoperability between software components without predefined infrastructure implemented at the lower layers.

The first technical objective of the VICTORY architecture was to define a network-based data bus architecture to provide shared processing and network resources and common data services, providing the mechanisms through which C4ISR/EW systems could share information and computational resources. This promotes reuse of hardware resources, and tends to reduce the overall vehicle SWaP impact of the overall integrated system.

A second technical objective of VICTORY was to define network-messaging interfaces at the on-the-wire level, specifying the protocols, data semantics, encoding, encapsulation, and transport levels for C4ISR and EW data interchanged between the systems. This on-the-wire approach enabled simplicity in integration but was only possible because VICTORY had the ability to standardize the network itself.

Another technical objective was to define a flexible security framework and services that are *baked in* to the network architecture, as opposed to being added on after systems were designed and implemented. This allows varying levels of security requirements to be supported as needed without modifying the core network architecture.

2.5. Artifacts

VICTORY develops a set of products that include the Architecture, Standard Specifications, Compliance Tests Suite, Reference Designs, Validation Artifacts, and reusable software packages. More details about these artifacts are

provided in this section. For details beyond the scope of this paper the reader can refer to the VICTORY web portal[4].

Architecture: Defines common terminology and the framework (scope, boundaries, structures), and identifies the modular entities and their interfaces. The modular entities in VICTORY are called *component types*, which are the logical encapsulation of a set of functions and the interfaces related to those functions. VICTORY Architecture version A Distribution A was published in April 2019.

Standard Specifications: Defines the detailed technical specifications for interfaces that are identified by the Architecture. The specifications are detailed enough for a developer to develop against. VICTORY Standards Specifications version V1.7 Distribution A was published in April 2019, V1.8 Distribution C [3] was published in February 2019, and V1.8 Distribution A is to be published the summer of 2019.

The VICTORY standard includes the web-service description language (WSDL) and XMLs schema definition (WSDL/XSD) files necessary to implement the component type interfaces. The WSDL/XSDs contain machine-readable interface definitions which can be used to create bindings for a number of programming languages. The paper's exemplar will be drawn from the XSD and WSDL files.

Compliance Test Suite: Provides "golden standard" test plan, report template and test tool for testing compliance with Component Types. CTS V3.2 was published in March 2018, and a Distribution A version supporting standard versions 1.7 and 1.8 are pending.

Validation Artifacts: Source code and documentation.

Reusable Software: Software applications and libraries developed and shared in order to jumpstart implementation.

The VSSO Reference Software Library is a software package that implements VICTORY component types and test clients. The reference software originates from initial implementation during the validation process used to mature the specifications before they are considered ready for use in programs. The VSSO Reference Software Library, as of May 2018, can be downloaded from the VICTORY portal.

The CCDC-GVSC Vehicle Electronics and Architecture group has developed libVICTORY, a software library which is used for implementing VICTORY clients and services. libVICTORY can be obtained through request on the VICTORY portal.

The component type implementations provided as reusable software have been verified with the use of Compliance Test Tool (CTT) which is part of the CTS.

VICTORY standards are still under active development. The set of standards necessary to implement a vehicle network and core services has been stable and transitioned into maintenance mode in the 2015 timeframe. New interfaces specifications are being been developed over time as needed to support additional capabilities.

The VICTORY standard is released under varying distribution arrangement ranging from *unrestricted* release to ITAR control. From the VICTORY standard's web portal, Figure 4 depicts an exemplar of the materials that can be found.

Figure 4. Exemplar Available VICTORY Material

The FACE Technical Standard is also still under active development. The latest FACE Technical Standard is version 3.0 [1]. The FACE Technical Standard and supporting technical and business guidance documents are published by the Open Group Standards Body, and are freely and globally available from the Open Group Bookstore at www.opengroup.org/publications [5]. Figure 5 depicts some of the material available for the FACE Standard.

[4] VICTORY Portal: https://victory-standards.org

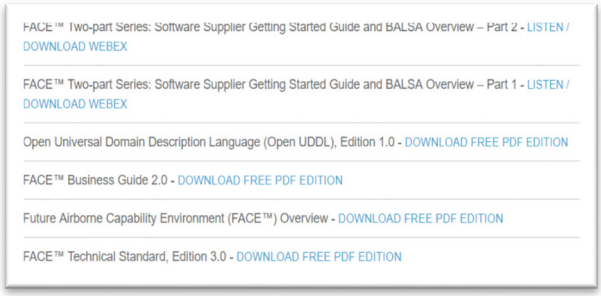

Figure 5. Exemplar Available FACE Material

The FACE Technical Standard is document-centric. As such the readily available material for the FACE Standard does not include machine-readable code packages. In this context, machine-readable code essentially means artifacts that can readily be used within software engineering tools like the Eclipse Integrated Development Environment (IDE) [6], Qt C/C++ IDE [7], Altova's XMLSpy [8] and other similar tooling. However, embedded within the FACE standard documentation would be *code* that the reader would need to extract.

As a historical note, Vanderbilt University created machine-readable artifacts that supported earlier versions of the FACE Technical Standard [9].

As it pertains to this paper, material for data interoperability and FACE conformance will be drawn from the FACE technical standard itself and artifacts available to FACE consortium members.

3. DATA INTEROPERABILITY

VICTORY and the FACE approach use significantly different mechanisms for defining the data content of types and structures to be interchanged between components. VICTORY messages are defined using a combination of technical documentation and XML schemas and WSDLs. FACE messages are defined using the FACE Data Architecture, the Open Universal Domain Description Language (Open UDDL) Edition 1.0, and the FACE Shared Data Model.

Descriptions and Exemplars demonstrating the approaches are provided in the following sections.

3.1. VICTORY Approach

VICTORY defines a set of specific data types that are exchanged between the C4, ISR, EW systems integrated with vehicles, and with the automotive, power, and weapon systems that are part of the vehicles.

VICTORY standard defines several types of network-based messaging interfaces that are used for different functionality, including, but not limited to:

- Data interfaces: A publish-subscribe pattern implemented with XML encoded messages encapsulated in UDP datagrams with a binary header and multicast on the network. Data interfaces are used to share data on the network, and the format is called VICTORY Data Message (VDM).

- Management interfaces: A request-response pattern implemented in some cases with a service orientation architecture (SOA) technology called Simple Object Access Protocol (SOAP), and in other cases with a technology called simple network management protocol (SNMP). They apply operations such as get() and set() to management parameters.

Both data and management interfaces are composed of parameters, be they data or management parameters. VICTORY defines messages structures, parameter naming, semantic, coordinate reference systems (frame of reference and datum), units, valid values, and other constraints in the text of the standard specification document using structured English language. The message and parameter types are also encoded into the appropriate XML, WSDL, and SNMP files, but no additional structure, meaning, or constraints are defined in these files that are not defined in the text of the standard.

This prose-centric approach, versus a model-based approach for defining interfaces and data types, was chosen for VICTORY for a variety of reasons, including the state of data modeling technology at the time. VICTORY does not define a machine-readable data model, but the specification defines all of the aspects required for definition, implementation, and common interpretation of the data types. The details provided is more than sufficient to populate formal data models and could be encoded into a machine-readable form given a sufficient data modeling language.

VICTORY thus defines all necessary aspects of a domain specific message set, but does not encode this information into a machine-readable formal data mode.

Example Data Definitions

Consider as an example the VICTORY data definitions for geodetic position, which is encoded into VDMs and reported by the Position Service to represent the position of the vehicle in global coordinates. The Position Service also implements request-response calls via which a client can request the current position of the vehicle or set the Position values in the service (used for initialization).

The VICTORY working group came to agreement on what parameters constitute "position", the structure of the VDM, the coordinate reference frame, units, and valid values for the parameters, and the ways in which Position would be encoded and transported on the network.

The VICTORY standard defines the contents of the Position message to as follows:

Position Message Content (partial definition for brevity)

Latitude:

- *The latitude of the platform, where latitude is defined as the angle from the equatorial plane to the perpendicular of the ellipsoid through a given point with northwards treated as positive.*
- *The latitude shall be in units of arc degrees.*
- *The latitude shall be a real value greater than or equal to –90°and less than or equal to 90°, where north latitudes are positive and south latitudes are negative.*

Longitude:

- *The longitude of the platform, where longitude is defined as the angle from the prime meridian plane to the meridian plane of a given point with eastward treated as positive.*
- *The longitude shall be in units of arc degrees.*
- *The longitude shall be a real value greater than –180°and less than or equal to 180°, where east longitudes are positive and west longitudes are negative.*

Altitude:

- *The altitude of the platform, where altitude is defined as a height, and the chosen reference is mean sea level.*
- *The altitude shall be in units of meters.*
- *The altitude shall be a positive or negative real value.*

Latitude Valid:

- *An indication that the latitude data <…> is valid.*
- *A Boolean value of True indicates the data is valid <…>.*
- *<…>.*

Latitude Uncertainty:

- *The uncertainty of the latitude measurement, where uncertainty is defined as a measure of the inherent variability of repeated measurements of a quantity.*
- *The measure of uncertainty shall be in arc degrees.*
- *The uncertainty shall be a real value at 1 σ.*
- *If the latitude valid flag is True, the latitude uncertainty shall be provided as part of the position message.*
- *<…>.*

Latitude Estimated:

- *An indication that the latitude data <…> is estimated.*
- *A Boolean value of True indicates the data is estimated.*
- *If the latitude valid flag is True, the latitude estimated flag shall be provided as part of the position message.*
- *If the latitude valid flag is False, the latitude estimated flag shall not be provided as part of the position message.*

<…>

Timestamp:

- *A timestamp corresponding to the time at which the position data was sampled.*
- *The timestamp shall be in the format as described <…>*
- *<…>*

Note: the definition above was truncated for brevity. Locations where text was left out are marked by "<…>".

The message and parameter definitions in this example are representative of the level of detail included in each VICTORY interface definition. The meaning, coordinate reference frame, units, and limits of each parameter is defined in prose form.

The following will show how these data definitions are used when defining interfaces. Specifically, the example shows how the position parameters are instantiated by a request-response interface. As mentioned above, the Position Service publishes VDMs with the current position, and also allows for the current position to be queried or set via request-response mechanisms.

Figure 6, Figure 7, and Figure 8 illustrate the WSDL definition of the *setPosition()* operation, which is represented in prose in the VICTORY Standard document. WSDL is an XML-based language, so can be viewed using an XML editing tool. Note the request and response message definitions, which refer to the latitude, longitude, and altitude parameters defined in the text above.

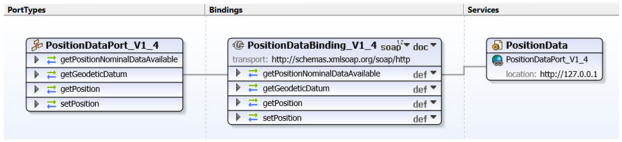

Figure 6. Exemplar VICTORY Web Service Definition Language (file: PositionData.wsdl)

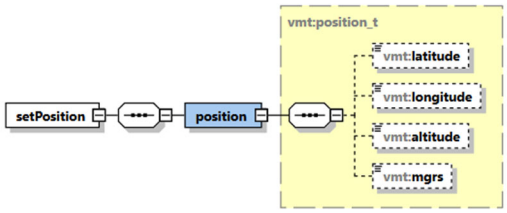

Figure 7. Exemplar VICTORY XML Schema Definition File (VICTORYMessages.xsd; element setPosition)

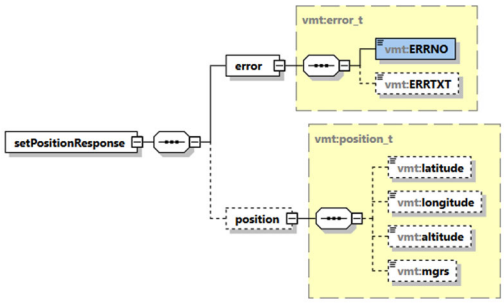

Figure 8. Exemplar VICTORY XML Schema Definition (file: VICTORYMessages.xsd; element: setPositionResponse)

This example represents one request-response operation that references the position parameters. There is also a *getPosition()* operation, and a VDM that is published, both containing these parameters. The position parameters represent a small subset of those defined in the VICTORY data definitions. Defining parameters and interfaces in this document-centric fashion has clearly been a large-scale endeavor.

3.2. FACE Approach

The FACE approach to data interoperability is based on the FACE Data Architecture, a model-based language for describing data and code generation tools to synthesize the software to implement the interface software. The FACE Data Architecture does not prescribe any a priori domain specific data or message representation. However, this is not to say that the FACE Data Architecture does not have a data interoperability strategy. The FACE Shared Data Model (SDM) forms the basis / basic building blocks from which *domain specific data models* (DSDM) are built. The FACE SDM has an underlying meta-model that is a domain specific language (built using EMOF, XMI and OCL). The goal is to semantically define the meaning of data in strict manner that eliminates ambiguity. The created FACE data model is not formed for the sake of documentation but support machine-readability and the use of the information in the engineering process. Semantics are defined through the expression of elements and relationships between elements in data models rather than through the use of text descriptions in schemas.

Figure 9 depicts the FACE data modeling approach – the data modeling starts with the Conceptual Data Model (CDM) follow by the Logical Data Model (LDM) and then the Platform Data Model (PDM). Progressing from CDM to LDM to PDM, the model is increasingly refined. With each level of refinement, the model is further constrained to a specific representation. Figure 10 is exemplar of what is part semantics available within the FACE data model. The informative portions and supporting documents of the FACE Technical Standard contain more information on how to use the FACE data modeling elements.

The PDM is probably the most familiar to implementers and software engineers and it is within the PDM that the assignment of data types and reference systems are made. However, the level of abstraction afforded by having the FACE CDM and FACE LDM provides semantic relationships to define the meaning and context of data fields as well as enable the means to reuse data elements across domains.

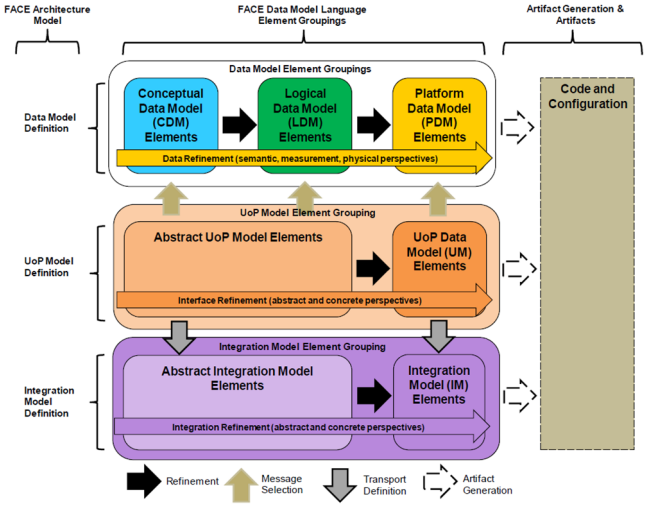

Figure 9. FACE Data Model Language

Figure 11 depicts exemplars elements from the FACE Shared Data Model. The screenshots are taken from the Enterprise Architect tool [10].

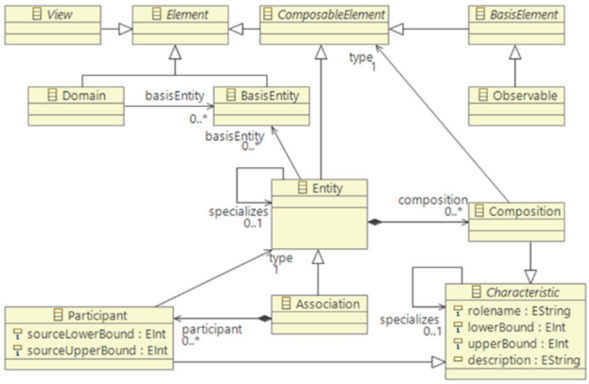

Figure 10. Conceptual FACE Data Model Metamodel Package

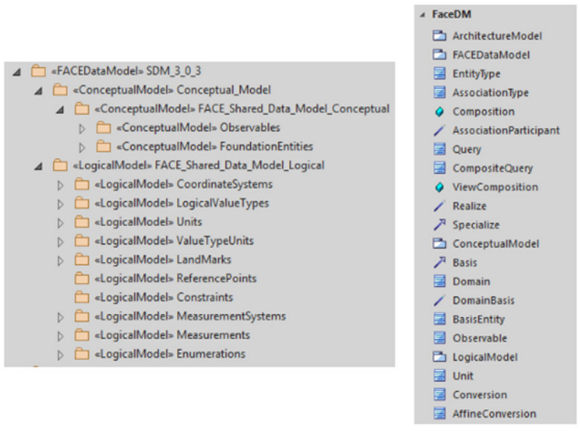

Figure 11. FACE Shared Data Model

Figure 12 and Figure 13 depict an example from one of the FACE tutorials and demonstrations, the Basic Avionics Lightweight Source Archetype (BALSA). The BALSA project serves as a working example application comprised of FACE conformant UoCs. The BALSA [11] data model (CDM, LDM and PDM) is a simplified representation of an aircraft's position with respect to a reference system. In progressing from the BALSA CDM to the LDM to the PDM, the level of detail and refinement is evident.

It was mentioned previously about reuse as result of the abstraction afforded by the CDM and LDM. In Figure 12(b) and Figure 13, the BALSA LDM defines only "position" but not its specific representation. In the PDM, *position* is further defined as *latitude, longitude* and *altitude*. Likewise, the BALSA PDM defines the geodetic reference frame to be *WGS84*. As an example of reuse, if the BALSA LDM was using in another domain it would be very possible to use a different representation for the geodetic reference frame.

As a thought exercise, if the BALSA CDM <<EntityType>>*Aircraft* was defined as <<EntityType>>*Vehicle*, then it becomes possible that the plausible LDM could represent an *Aircraft* or *Satellite* or *Boat*. Now the *position* of those entities could be represented within those domains.

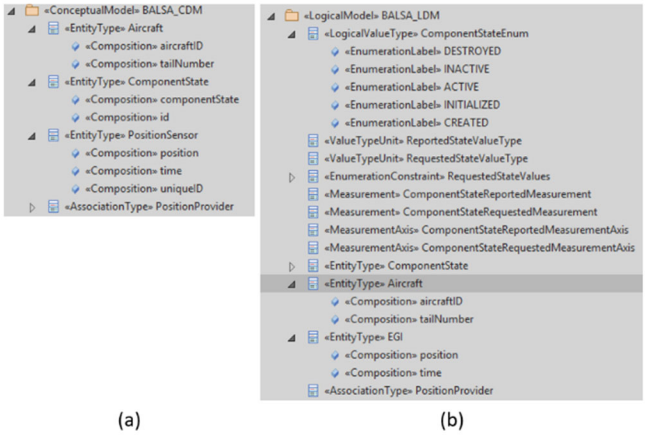

Figure 12. Exemplar (a) FACE CDM and (b) FACE LDM

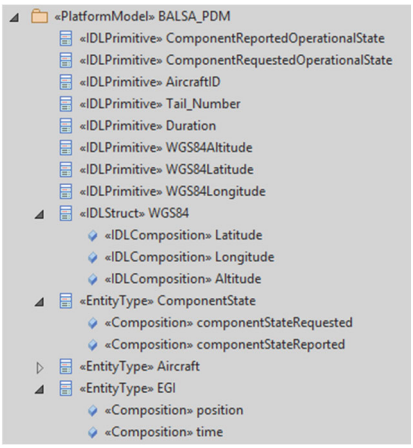

Figure 13. Exemplar FACE PDM

Figure 14 depicts an exemplar of the FACE Data Model query feature which is core to the FACE 3.0 Data Modeling approach.

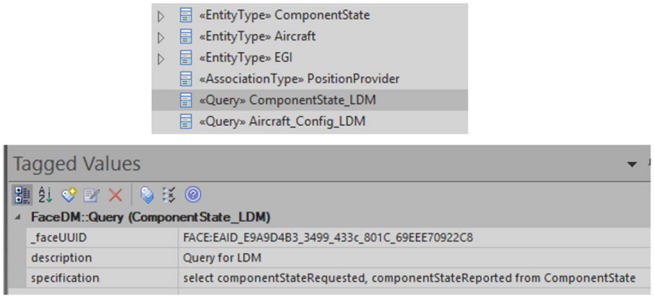

Figure 14. Exemplar FACE Data Model Query Semantics

As a first approximation, the BALSA position example provides a useful point of comparison with the VICTORY definition for position presented above. The FACE Data Architecture approach also employs *Query, View* and

Template. These serve as means to exercise "separation of concerns" as part of the FACE data modeling approach. The use of *Query, View* and *Template* provides the means to "pick out" the data of interest and to represent the data syntactically.

4. SECURITY

Security aspects of interoperability are addressed by both VICTORY and the FACE Technical Standard, although quite differently.

Safety and Security are addressed by the FACE Technical Standard defining key interfaces, by the FACE Operating System Segment (OSS) Profiles, and through the Transport Services Segment (TSS). The FACE Technical Standard itself does not prescribe specific safety and security requirements, but the architecture, OSS profiles, and TSS guidance make it easier for platforms, systems, and software to comply with regulations. Specification of key interfaces. OSS API, Transport Service APIs, and Input / Output APIs enable the control and regulation of those interfaces for security.

Transport Services Segment, section E.4.2.2 of the FACE Technical Standard describes the use of security within TSS [1]. Within TSS, it is implementation-defined security meta-data such as certificates, public keys, channel encryption, etc. While the architectural approach is addressed, specific implementations are not defined.

OSS Profiles is one of the FACE segments where safety and security are addressed. The FACE Technical Standard specifies four OSS Profiles, General Purpose Profile, Safety-Base, Safety-Extended, and Security Profiles. These profiles apply to usage of operating system APIs, programming language features, and support for time and space partitioning. While the FACE Technical Standard is not prescriptive of implementation, the profiles have been defined to ease alignment with airworthiness and safety of flight requirements.

The OSS Profiles were developed by the Operating Systems Subcommittee of the FACE Consortium Technical Working Group, using the standards development consensus process of the Open Group. In the consensus environment, it was determined that for the purposes of maximizing portability and history of safety-of-flight certifications, that OSS APIs for the Security Profile were the most stringent with known-implementations that have been safety-certified on systems to DO-178 DAL A. Similarly, the Safety-Base OS Profile has known-implementations that have been safety-certified to DO-178B DAL B or higher. The OSS Profiles maximize alignment to support application portability across different hardware and software systems. Using a less stringent OSS Profile does not preclude an application from meeting a more stringent safety and/or security level assessment. It is possible for software developed to the FACE OSS General Purpose Profile to satisfy high-assurance DO-178 DAL A certification. Portability of the application across different hardware and software implementations at the same DAL may be more challenging.

The FACE Technical Standard does not require adherence to safety and/or security standards for conformance. The aspects of safety and security requirements are outside the scope of the FACE Technical Standard which addresses portability and interoperability at the interfaces. However, software suppliers are encouraged to include information regarding additional certifications and standards applicable to the FACE Certified product or Unit of Conformance (UoC) in the FACE Library portal listing FACE Certified software.[5]

The VICTORY Standard addresses security in several areas, including protection (confidentiality via encryption) of data in transit and at rest, protection from unauthorized access (authentication and authorization via attribute-based access control), security event detection and logging, among others.

Most would consider the security focus of VICTORY to be on the higher level in the software stack. In general VICTORY adopts layered IT mechanisms such as Internet Protocol security (IPSec), Transport Layer Security (TLS) and Web-Service Security (WS-Security). VICTORY does define custom security-related interfaces where there are not existing standards and these customizations generally reside at the application layer.

The VICTORY Architecture and Standard Specifications cover some security areas that the FACE approach does not, such as: (i) data at rest encryption; (ii) authentication; (iii) authorization; (iv) data-signing; (v) host-based firewall management; and others.

The FACE Technical Standard generally focuses on accommodating safety, which can enhance security in a number of ways, but there are definitely areas in which security is in is the original scope – enterprise-centric vs. platform-centric. VICTORY does not attempt to directly address safety concerns.

5. ASSESSING STANDARDS ADHERENCE

Irrespective of whether technical standards use the term compliance or conformance, assessing adherence can be challenging. As there is no normative definition for either term, technical standards may choose to define the term to meet their respective needs.

Assessing adherence to any standard can occur via many different ways – ranging from self-inspection / determination

[5] FACE Library Portal: https://www.facesoftware.org/

to independent 3rd party assessment, ranging from using a subjective checklist to an objective machine-verifiable approach.

In the case of both VICTORY and the FACE approach, assessing adherence to their respective technical standard is achieved via machine-verifiable means and can be executed via an independent 3rd party. Software and/or components that can be subject to machine-based verification and independent 3rd party assessment provide the most rigorous and reliable form of assessment for standards adherence. If the assessment can be driven by machine-based verification, the results of whether compliance or conformance can be as simple as shown in Figure 15.

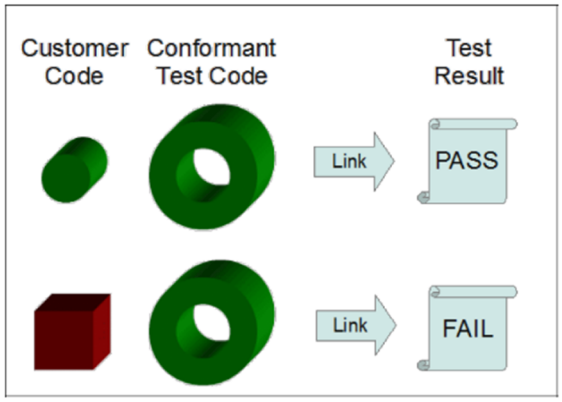

Figure 15. Standards Adherence [12]

5.1. VICTORY Compliance

The VICTORY standard [3] describes the nature and the scope of what constitutes VICTORY compliance. In summary, VICTORY compliance is a machine-verifiable approach to assessing VICTORY software, components etc.

Norman et. al. [13] describes the types of tests applied to evaluate VICTORY compliance, which are defined and supported by the VICTORY Compliance Test Suite (CTS). The VICTORY CTS is comprised of (i) Compliance Test Plans (CTP); (ii) Compliance Test Reports (CTR); (iii) Compliance Test Tool (CTT). Norman el al. also described how the VICTORY CTS could be used as a development tool as well.

5.2. FACE Conformance

The FACE Technical Standard ascribes to *conformance* rather than *compliance*. In the FACE ecosystem, FACE Conformance is applied at a modular level and is pass fail as opposed to a system level test, where a system may be comprised of heterogeneous components, a mixture or FACE UoCs and other software.

The FACE Conformance process is similarly rigorous but more formalized than the VICTORY Compliance process. FACE Conformance leverages a machine-verifiable approach along with a Conformance Verification Matrix (CVM) used by independent third parties for verification. Machine verification is performed using the FACE Conformance Test Suite (CTS), a publicly available software test suite. Figure 16 and Figure 17 depict screen shots from the FACE Conformance Tool Suite (CTS).

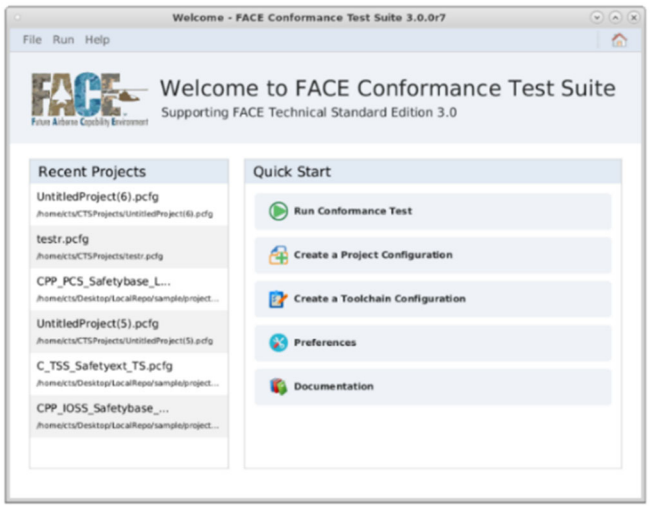

Figure 16. FACE Conformance Test Suite Main Screen

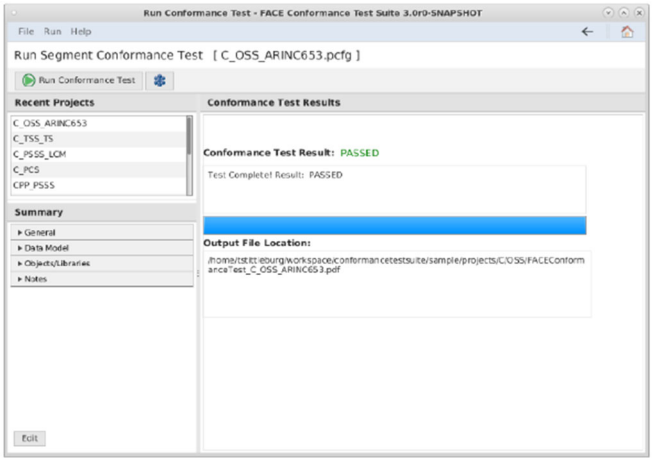

Figure 17. Exemplar FACE CTS Results

There are special cases where some requirements may not be verifiable by machine or supported by the CTS. These instances rely on inspection to verify adherence to the standard to complete the CVM. To achieve FACE Certification, UoCs must be tested using the CTS, evaluated against the CVM, approved by one of the official FACE Verification Authorities and then by the FACE Certification Authority.

Upon passing the FACE Verification, FACE Certification is finalized by the FACE Certification Authority. FACE

certified products can further be registered in the FACE Registry and the listing published on the FACE Library Portal.

Note that FACE Certification adherence to a particular OSS Profile does not make the component DO-178B/C certified. Airworthiness and safety of flight testing are conducted at the system level while FACE conformance is assessed at the modular component level. The FACE business approach recommends providing certification artifacts to ease level of effort and reduce cost for follow-on safety and security testing and accreditation.

6. VICTORY & FACE INTEGRATION

U.S. Army ground vehicle programs are developing systems which attempt to leverage the benefits of both the VICTORY and FACE standards. There are several emerging patterns of deploying these two standards that appear to be prevalent, in some cases the joining of the two technologies has been deliberate and in other cases coincidental.

The U.S. Army's CCDC C5ISR Center has created the C4ISR/EW Modular Open Suite of Standards (CMOSS) which has been under development for several years. CMOSS aims to provide an infrastructure and computing environment that can be used to share amplifiers, filters, sensors, and processors between multiple applications and systems. The approach that is advocated within CMOSS involves the layered integration of several open standards, including the FACE Technical Standard and VICTORY, as shown in Figure 18. This approach attempts to combine the software portability benefits of the FACE Standard with the interoperability and networking benefits of VICTORY.

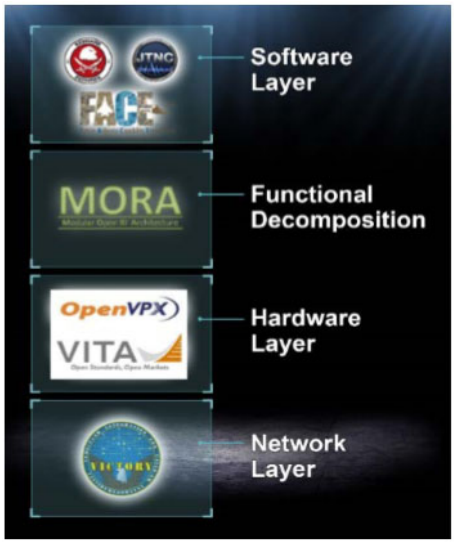

Figure 18. CMOSS Open Standards Stack [14]

The use of approaches such as CMOSS is intended to enable faster insertion of capabilities and cost savings during the procurement and sustainment phases of vehicle and radio frequency system lifecycle.

The VICTORY Standard Specifications and FACE Technical Standard have also been used together in other weapon-systems in a decoupled integration pattern. System developers seek to leverage FACE components within the safety critical portions of their system to achieve a level of software portability and modularity and because FACE products are often geared towards avionics applications and should have the pedigree and flight-safety certification artifacts needed to simplify safety certification. It is also recognized that it is impractical to safety-certify the entire vehicle, yet it is desirable to connect these weapons with other vehicle systems and developers have leveraged VICTORY to accomplish this. Figure 19 shows one example of how a weapon system can leverage both VICTORY and FACE standards in a mixed criticality system. This design shows a paired-firewall configuration that provides a strong security posture and has similarities with current approaches recommended for integrating remote access and local process control for industrial control systems [15].

Figure 19. FACE/VICTORY Weapon-System Implementation Example

These examples showing layered and decoupled implementations using FACE and VICTORY are a testament to the benefits of MOSA and provide a level of validation for these two initiatives.

7. OPPORTUNITIES FOR ALIGNMENT

The previous section demonstrated that the MOSA aligned VICTORY and FACE standards defined key interfaces and protocol support lend themselves well to cross-standard integration. At a high level, there are many opportunities for alignment for example:

1. The VICTORY Data Bus (VDB) can serve as part of the FACE TSS and/or the FACE TSS can be hosted on the VDB.
2. The FACE OSS can be used where operating systems are required (e.g. to support a VICTORY Shared-Processing Unit Component).
3. The FACE approach can be used to create APIs for VICTORY services.

Figure 20 depicts Notional VICTORY Architecture with FACE UoCs.

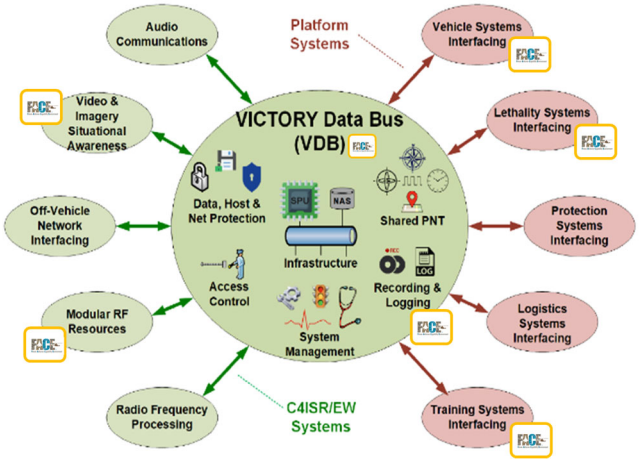

Figure 20. Notional VICTORY Architecture with FACE UoCs

Known areas of opportunities to leverage FACE UoCs are indicated by the FACE icons overlaying the VICTORY Architecture diagram.

Beginning with the VDB, in a combined system architecture, VICTORY's network-based data interfaces and management interfaces fit within the FACE TSS. The FACE TSS supports a variety of communication and data exchange paradigms to facilitate interoperability across different systems and technical implementations. An existing VDB can be integrated with a FACE TSS, or FACE TSS UoCs can be hosted within a VDB. For example, a FACE Conformant transport service (such as a FACE TSS implementation of DDS) can provide a means to implement publish subscribe message exchanges, or a different transport service technology can be leveraged. Using FACE allows the component interfaces to remain unchanged while the technology implementing the message pattern can be replaced.

The FACE OSS can be used as part of the computing environment within any subsystem to host FACE software components. Where operating systems are required for real-time and or safety-critical applications, FACE OSS should be considered as an option. FACE OSS Profiles are aligned with Software Communications Architecture (SCA) Software Application Profiles (AEP) for hosting software defined radio and related software applications, so as shown in Figure above, FACE software can be used to host and/or implement RF resources.

Section 6 provides an example of the application of the FACE approach in safety-critical systems depicted in Figure 19 as Lethality Systems Interfacing. Likewise, there may be opportunity for reuse of autonomous navigation and routing FACE software components that may be usable with Vehicle Interfacing Systems. Training Systems Interfacing services and capabilities can benefit from leveraging the FACE approach such that software source code used to implement capabilities within subsystems can be reused and re-hosted directly within training systems.

Finally, the FACE approach can be used to create VICTORY Services. To ensure VICTORY Compliance while leveraging FACE Conformance data interoperability is the key. Data interoperability not only has practical impact, as it may pertain to a VICTORY system sharing information with a system comprised of FACE UoCs, but also ramification to a shared core tenet of the respective standard which is VICTORY compliance and FACE conformance.

8. CONCLUSION

VICTORY and the FACE Technical Standard are open architecture standards that address different problems and have chosen distinctly different technical approaches to achieving their objectives. VICTORY Standard Specifications defines standards for a network framework and on-the-wire messaging interfaces to enhance subsystem interoperability while the FACE Technical Standard defines reference architectures to support real-time and embedded system software portability.

In this paper we have explored key aspects of the VICTORY Standard and the FACE Technical Standard and have presented evidence and rationale to answer the questions posed in Section 1:

"Why should the government invest in and maintain two architectures, and are they worth the investment?"

We have shown that both the VICTORY Standard and FACE Technical Standard are open system architectures that support MOSA, addressing different high-priority problems in their respective problem spaces. Each effort has shown progress towards accomplishing their technical objectives and represent two different sets of tools that system integrators can leverage when building military systems. They deserve to be supported and maintained because they do not provide redundant functionality and cannot easily be consolidated.

"Is it possible for capabilities implemented in one architecture to be integrated with those from another, and what would be the level of effort?"

Through principles of MOSA, the standards and architectures have defined key interfaces applied at different levels of scope. We have shown in sections 6 and 7 that there are a variety of ways to integrate the capabilities provided by each architecture, and that in some cases system developers appear to be doing this naturally, which implies a relatively low-level of effort.

"Are there efficiencies or technical benefits to be achieved by somehow aligning these architectures, and what are the costs and benefits?"

There are efficiencies and technical benefits to be achieved by working toward commonality between these two efforts. At a minimum, alignment of the FACE Technical Standard with VICTORY can help program managers and system developers understand the costs and benefits of each approach. Alignment can facilitate their choice of the architectural patterns that best suits the particular system they are designing. Joint FACE VICTORY capabilities could enable rapid integration of these capabilities onto VICTORY systems while supporting portability of software onto different hardware and computing platforms to optimize computing performance while addressing SWaP concerns. High-performance, and real-time, safety-critical components for new capabilities could be deployed across a broader range of platforms.

A quantitative assessment of costs and benefits was out of the scope of this paper, but it appears as if an alignment that consists of defining common VICTORY and FACE integration patterns would be low-cost while a more prescriptive mapping of data structures and semantics would require significantly more effort.

Finally, this paper demonstrates, through the flexible manner in which the two standards can be adapted with each other, that open system architecture and MOSA truly do enhance interoperability, resulting in capabilities that can potentially be utilized across air, land, sea, and space systems.

9. REFERENCES

[1] The Open Group FACE Consortium, The Future Airborne Capability Environment (FACE) Techncial Standard, Edition 3.0, Burlington, MA: The Open Group, 2017.

[2] VICTORY Standards Support Office, *Vehicular Integration for C4ISR/EW Interoperability (VICTORY) Standard Specifications, Version 1.7,* VICTORY Standards Support Office, 2018.

[3] VICTORY Standards Support Office, *Vehicular Integration for C4ISR/EW Interoperability (VICTORY) Architecture Version A,* 2019.

[4] The Open Group FACE Consortum, *Future Airborne Capability Environment Business Guide, Edition 2.0,* Atlanta, GA: The Open Group, 2019.

[5] The Open Group, "Future Airborne Capability Environment (FACE™) | The Open Group," TheOpenGroup.org, 2019. [Online]. Available: https://www.opengroup.org/face. [Accessed 22 June 2019].

[6] Eclipse Foundation, "The Platform for Open Innovation and Collaboration | The Eclipse Foundation," Eclipse.org, 2019. [Online]. Available: https://www.eclipse.org/. [Accessed 22 June 2019].

[7] The Qt Company, "Qt | Cross-platform software development for embedded & desktop," The Qt Company, 2019. [Online]. Available: https://www.qt.io/. [Accessed 17 April 2019].

[8] Altova Incorporated, "XML Editor: XMLSpy," Altova Incorporated, 2019. [Online]. Available: https://www.altova.com/xmlspy-xml-editor. [Accessed 17 April 2019].

[9] Vanderbilt University Institute for Software Integrated Systems, "Institute for Software Integrated Systems | FACE Downloads," Vanderbilt University, 2019. [Online]. Available: http://www.isis.vanderbilt.edu/FACE. [Accessed 28 February 2019].

[10] Sparx Systems, "UML modeling tools for Business, Software, Systems and Architecture," Sparx Systems Proprietary Limited, 2019. [Online]. Available: https://sparxsystems.com/. [Accessed 18 February 2019].

[11] The Open Group FACE Consortium, FACE Software Suppliers Getting Started Guide, Version 1.0, Burlington, MA: The Open Group, 2017.

[12] Georgia Tech Applied Research Corporation, *Conformance Test Suite Manual for Testing Interface and Application Code Against the FACE Standard 3.0,* Atlanta: The Open Group, 2019.

[13] D. Norman, J. Klein, A. Strickland, B. Meiners, K. J. Saylor and J. Broczkowski, "Ensuring VICTORY Compliance," in *Proceedings of the 2018 Ground Vehicle Systems Engineering and Technology Symposium (GVSETS)*, Novi, 2018.

[14] B. Peddicord, *Tri-service Convergence: C4ISR/EW Modular Open Suite of Standards (CMOSS),* Embedded Tech Trends, 2018.

[15] K. Stouffer, V. Pillitteri, S. Lightman, M. Abrams and A. Hahn, *Guide to Industrial Control Systems (ICS) Security,* Gaithersburg, MD, 2015.