The Open Group Guide

FACE™ Software Supplier Getting Started Guide, Edition 3.x

Future Airborne Capability Environment™

THE Open GROUP

Prepared by The Open Group FACE™ Consortium Integration Workshop Standing Committee

NAVAIR Public Release 2021-965

Distribution Statement A – “Approved for public release; distribution is unlimited”

Copyright © 2022, The Open Group LLC for the benefit of the FACE Consortium Members. All rights reserved.

The Open Group hereby authorizes you to use this document for any purpose, PROVIDED THAT any copy of this document, or any part thereof, which you make shall retain all copyright and other proprietary notices contained herein.

This document may contain other proprietary notices and copyright information.

Nothing contained herein shall be construed as conferring by implication, estoppel, or otherwise any license or right under any patent or trademark of The Open Group or any third party. Except as expressly provided above, nothing contained herein shall be construed as conferring any license or right under any copyright of The Open Group.

Note that any product, process, or technology in this document may be the subject of other intellectual property rights reserved by The Open Group, and may not be licensed hereunder.

This document is provided "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT. Some jurisdictions do not allow the exclusion of implied warranties, so the above exclusion may not apply to you.

Any publication of The Open Group may include technical inaccuracies or typographical errors. Changes may be periodically made to these publications; these changes will be incorporated in new editions of these publications. The Open Group may make improvements and/or changes in the products and/or the programs described in these publications at any time without notice.

Should any viewer of this document respond with information including feedback data, such as questions, comments, suggestions, or the like regarding the content of this document, such information shall be deemed to be non-confidential and The Open Group shall have no obligation of any kind with respect to such information and shall be free to reproduce, use, disclose, and distribute the information to others without limitation. Further, The Open Group shall be free to use any ideas, concepts, know-how, or techniques contained in such information for any purpose whatsoever including but not limited to developing, manufacturing, and marketing products incorporating such information.

If you did not obtain this copy through The Open Group, it may not be the latest version. For your convenience, the latest version of this publication may be downloaded at www.opengroup.org/library.

The Open Group Guide

FACE™ Software Supplier Getting Started Guide, Edition 3.x

ISBN: 1-937218-94-2

Document Number: G173

Published by The Open Group, March 2022.
This document supersedes Version 1.0, published in August 2017.

Comments relating to the material contained in this document may be submitted to:

The Open Group, 177 Huntington Avenue, Suite 1703, PMB 54820, Boston, MA 02115-3153, US

or by electronic mail to:

ogface-admin@opengroup.org

Contents

Table of Figures

Preface

The Open Group

The Open Group is a global consortium that enables the achievement of business objectives through technology standards. With more than 870 member organizations, we have a diverse membership that spans all sectors of the technology community – customers, systems and solutions suppliers, tool vendors, integrators and consultants, as well as academics and researchers.

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.

The Open Group publishes a wide range of technical documentation, most of which is focused on development of Standards and Guides, but which also includes white papers, technical studies, certification and testing documentation, and business titles. Full details and a catalog are available at www.opengroup.org/library.

This Document

This document is the FACE™ Software Supplier Getting Started Guide (GSG) used by Software Suppliers who are implementing the Future Airborne Capability Environment™ Technical Standard (the FACE Technical Standard). It is designed to be a navigational quick start guide providing access to example software designed according to the requirements of the FACE Technical Standard, developed FACE data models, and corresponding verification artifacts.

It was developed and is maintained by The Open Group FACE Consortium.

Trademarks

ArchiMate, DirecNet, Making Standards Work, Open O logo, Open O and Check Certification logo, Platform 3.0, The Open Group, TOGAF, UNIX, UNIXWARE, and the Open Brand X logo are registered trademarks and Boundaryless Information Flow, Build with Integrity Buy with Confidence, Commercial Aviation Reference Architecture, Dependability Through Assuredness, Digital Practitioner Body of Knowledge, DPBoK, EMMM, FACE, the FACE logo, FHIM Profile Builder, the FHIM logo, FPB, Future Airborne Capability Environment, IT4IT, the IT4IT logo, O-AA, O-DEF, O-HERA, O-PAS, Open Agile Architecture, Open FAIR, Open Footprint, Open Process Automation, Open Subsurface Data Universe, Open Trusted Technology Provider, OSDU, Sensor Integration Simplified, SOSA, and the SOSA logo are trademarks of The Open Group.

CentOS and Cygwin are trademarks of Red Hat, Inc.

FreeBSD is a registered trademark of the FreeBSD Foundation.

Java is a registered trademark of Oracle and/or its affiliates.

Linux is a registered trademark of Linus Torvalds in the US and other countries.

Microsoft and Windows are registered trademarks of Microsoft Corporation in the United States and/or other countries.

POSIX is a trademark of the IEEE.

Raspberry Pi is a trademark of the Raspberry Pi Foundation.

Ubuntu is a registered trademark of Canonical Ltd.

All other brands, company, and product names are used for identification purposes only and may be trademarks that are the sole property of their respective owners.

Acknowledgements

(Please note affiliations were current at the time of approval.)

The Open Group FACE Consortium gratefully acknowledges the contribution of the following people in the development of this document:

Principal Authors

- Thomas Brixey, TES-SAVi
- Chris Crook, US Army PEO Aviation
- John Gray, US Army PEO Aviation
- Jeff Mayes, OAR Corporation
- Andre Odermatt, Real-Time Innovations
- Joel Sherrill, US Army DEVCOM Aviation & Missile Center/OAR Corporation
- Alicia Taylor, US Army PEO Aviation

Additional Contributors

- Page Downer, NAVAIR
- Ken Erickson, TES-SAVi
- Jeffry Howington, Collins Aerospace
- Leanne May, Collins Aerospace
- Sean Mulholland, TES-SAVi
- Stephen Simi, TES-SAVi
- John Thomas, LDRA

Referenced Documents

The following publications are referenced in this document:

- ARINC 653: Avionics Application Software Standard Interface
- ARINC 661: Cockpit Display System Interfaces to User Systems
- FACE Library Requirements, Version 3.0, January 2017; refer to: https://publications.opengroup.org/downloadable/download/link/id/MC4wMDExOTEwMCAxNjQzMjA2NDkwMTI4MDIwODEzMDk0NjE5NjI%2C/
- FACE™ Overview, The Open Group Guide (G20B), published by The Open Group, July 2020; refer to: www.opengroup.org/library/g20b
- FACE™ Problem Report (PR) and Change Request (CR) Process, The Open Group Guide (G166), published by The Open Group, October 2020; refer to: www.opengroup.org/library/g166
- FACE™ Technical Standard (all Editions); refer to: www.opengroup.org/face/docsandtools
- Reference Implementation Guide for FACE™ Technical Standard, Edition 3.0, The Open Group Guide (G209), published by The Open Group, May 2020; refer to: www.opengroup.org/library/g209
- The Open Group FACE™ Conformance Verification Matrix User's Guide (MUG), Certification Program Documentation (X1704), published by The Open Group, September 2020; refer to: www.opengroup.org/library/x1704
- The Open Group FACE™ Contract Guide: Guidance in Writing Solicitations and Proposals with FACE Requirements, Version 2.0, The Open Group Guide (G18D), published by The Open Group, September 2018; refer to: www.opengroup.org/library/g18d

The following webinars are referenced in this document:

- FACE Conformance Program, Webinar (D177), published by The Open Group, October 2016; refer to: www.opengroup.org/library/d177
- Introduction to The Open Group FACE™ Contract Guide: Guidance in Writing Solicitations & Proposals with FACE Requirements, Webinar (D161), published by The Open Group, May 2016; refer to: www.opengroup.org/library/d161
- The Open Group FACE™ Consortium Business & Technical Overview, Webinar (D210), published by The Open Group, February 2018; refer to: www.opengroup.org/library/d210

The following web pages are referenced in this document:

- BALSA: https://www.opengroup.org/face/balsa

- FACE Conformance: www.opengroup.org/face#conformance
- FACE Conformance Test Suite(s): https://www.opengroup.org/face/conformance-testsuites
- FACE Consortium: www.opengroup.org/face
- FACE Consortium Conformance Publications & Tools: www.opengroup.org/face/conformance-publications-and-tools
- FACE Documents & Tools: www.opengroup.org/face/docsandtools
- FACE Problem Report (PR) & Change Request (CR) Ticketing System: ticketing.facesoftware.org
- New Users – Getting Started: https://www.opengroup.org/face/gettingstarted
- Third-Party Tools: www.opengroup.org/face/third-party-tools

1 Introduction

1.1 Background and Scope

This document is designed to be a navigational quick start guide for Software Suppliers to develop FACE conformant software. It provides the reader with access to sample software developed according to the requirements of the FACE Technical Standard, developed FACE data models, and corresponding verification artifacts. The sample FACE software contains Units of Conformance (UoC) that pass the FACE Conformance Test Suite (CTS). It does not cover all components of the FACE Technical Standard. The intent is to navigate readers through a demonstration example, with information on how to continue advanced FACE development efforts, including accessing and navigating the FACE website, FACE Library, FACE tools, and published FACE business and technical documents.

Note: This document is *not* designed to be an introduction to the overall FACE approach. Readers should have a basic understanding of the FACE approach and of embedded systems software development principles and practices. If you are new to the FACE approach it is recommended that you review the New Users – Getting Started section of the FACE Consortium website at https://www.opengroup.org/face/gettingstarted. A Software Supplier is anyone wishing to find out more information about or programming to the FACE approach and the FACE Technical Standard or anyone interested in or applying for certification in the FACE Conformance Program. This may include the original software developer, an integrator, or another entity wishing to certify software developed from another party. This document is specifically designed for those wishing to develop software that meets the requirements of the FACE Technical Standard.

As a working example, Basic Avionics Lightweight Source Archetype (BALSA) is the application being used as the software working example for this document. It is an implementation of the FACE Reference Architecture that features Portable Components Segment (PCS) and Platform-Specific Services Segment (PSSS) component applications that demonstrate a basic avionics process. More detailed information about BALSA can be found in Chapter 2.

Distributions of the BALSA source code are available on The Open Group FACE website in the New Users – Getting Started section (see above). The direct link to the BALSA website where public distributions can be downloaded is at https://www.opengroup.org/face/balsa. This document is to be used in conjunction with the FACE Reference Implementation Guide (RIG), the FACE Technical Standard, and other primary FACE documents. It does not add or modify requirements stated in these documents. In the event of conflict between the FACE Technical Standard and this document, the FACE Technical Standard takes precedence.

This document is designed to:

- Provide start-up guidance by directing the reader to basic and primary documentation, demonstration examples, and tools available for the implementation and testing of FACE products written to the FACE Technical Standard

- Introduce the user to an example FACE application that illustrates use and/or implementation of various FACE segments to provide a common avionics capability
- Provide the user with access to software that can be used to increase knowledge of the FACE approach through a "learning by doing" approach; the example software and artifacts can be downloaded, explored, analyzed, updated, integrated, tested, etc.
- Assist the user in gaining an introduction to and a basic understanding of the verification process required in the FACE Conformance Program

References and descriptions of other published standards (e.g., ARINC 653, ARINC 661, and POSIX™) are presented at an overview level relative to the FACE Technical Standard and are not designed to conflict with the referenced standards.

1.2 Primary FACE Documents

FACE Consortium information, documents and tools, procurement activity, conformance, and many other resources are available at The Open Group FACE Consortium public website at www.opengroup.org/face.

Software Suppliers are encouraged to review the document links listed here as this document is designed to be used in conjunction with these and other FACE documents. The intent is to direct readers to primary FACE documents for further study. While embedded links in this document point to the most current editions of the primary FACE documents, please note that The Open Group FACE Consortium public website lists all editions of these documents under the Documents & Tools section found at www.opengroup.org/face/docsandtools.

Note: The most current editions of the FACE Technical Standard, FACE CTS, and supporting documentation should be used for any new development and verification of FACE UoCs unless otherwise contractually required.

In order to download these documents, you do not have to be a FACE Consortium member; however, the user must create an account for access to The Open Group Library. This account is free-of-charge, and requires only basic information such as name and a valid email address.

1.2.1 FACE Overview

The FACE Overview (www.opengroup.org/library/g20b) provides the reader with an introduction to the structure, processes, documents, and tools produced by the FACE Consortium. This document is designed for those who want to learn more about the business and technical aspects of the FACE Consortium, to use and provide feedback on the FACE Consortium products, and to engage with and contribute to the FACE Consortium.

1.2.2 FACE Consortium Business & Technical Overview

To gain a better initial understanding of the business and technical aspects of the FACE Consortium, it is recommended that The Open Group FACE Consortium Business & Technical Overview webinar be viewed (available at www.opengroup.org/library/d210).

1.2.3 FACE Technical Standard

The FACE Technical Standard (available at www.opengroup.org/face/docsandtools) is the "keystone" document of the FACE Consortium. It embodies a set of requirements and descriptions that define the FACE Reference Architecture.

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, and uses industry standards for distributed communications, programming languages, graphics, operating systems, and other areas as appropriate.

The FACE Technical Standard begins with an architectural overview introducing the Reference Architecture, followed by a detailed description of each architectural segment and interface. It defines the requirements for the creation of UoCs in each architectural segment of the FACE Reference Architecture of each software component of the Reference Architecture. Finally, the appendices specify the specific Application Programming Interfaces (APIs) required by the FACE Technical Standard, the FACE Data Architecture required by the FACE Technical Standard, as well as other applicable standards.

Each Edition of the Technical Standard has a corresponding RIG (starting with Edition 2.0), Shared Data Model (SDM) (starting with Edition 2.0), Data Model Governance Plan, Conformance Verification Matrix (CVM), and CTS.

1.2.4 FACE Reference Implementation Guide

The Reference Implementation Guide (RIG) is designed to support the FACE Technical Standard by providing more detailed information including examples about how to develop UoCs and data models and how to guide the developer through integration and implementation scenarios.

The RIG for FACE Technical Standard, Edition 3.0 (available at www.opengroup.org/library/g209) is divided into three volumes, each with a different focus. It is designed to support all 3.x editions.

- Volume 1 provides General Guidance and is focused on PCS and PSSS UoCs, Graphics Services, and Safety and Security considerations
- Volume 2 focuses more on the Computing Environment and developing the Transport Services Segment (TSS), I/O Services Segment (IOSS), and Operating System Segment (OSS) UoCs; in addition it provides:
  — More detailed information on FACE OSS Profiles and how Health Monitoring and Fault Management (HMFM) and Configuration Services may be implemented within a FACE OSS

  — Guidance on implementing and using the Injectable Interface for FACE Interfaces and Programming Language Mapping Rules
- Volume 3 (Data Architecture) provides content to aid in understanding the overall approach the FACE Consortium has taken with respect to data management within the FACE Technical Standard; central to the development and integration of FACE UoCs is the representation and documentation of data, according to the FACE Data Architecture

1.2.5 FACE Conformance

The FACE Consortium website has a section called FACE Conformance (www.opengroup.org/face#conformance), which provides links to information such as Conformance FAQs, Verification and Certification Authorities, Product Certification, FACE Software Registry, Approved Corrections, and Conformance Publications & Tools.

The FACE Consortium Conformance Publications & Tools section (available at www.opengroup.org/face/conformance-publications-and-tools) contains the documents and tools that define the process and policies that govern the FACE Conformance Program. Of particular importance to the Software Supplier is the FACE Conformance Certification Guide, the FACE CVM, and the FACE Conformance Verification Matrix User's Guide (MUG), which assists in interpreting and using the FACE CVM.

The FACE CVM clarifies the Conformance Requirements from the FACE Technical Standard that a product must meet in order to be first verified and then certified as FACE conformant, along with the specific techniques to be used to verify each of these Conformance Requirements. Each edition and corrigendum of the Technical Standard has a corresponding CVM. A corrigendum is a correction to a document that fixes errors discovered after the document has been published. Any corrigendum published against a FACE document will result in versioning of the associated document to the next (.x) edition; only the corresponding (.x) edition will be used for conformance.

The FACE Conformance Program webinar, designed to facilitate understanding, is available at www.opengroup.org/library/d177.

1.2.6 FACE Conformance Test Suite(s)

The FACE Conformance Test Suite (CTS) (available at www.opengroup.org/face/conformance-testsuites) is the test tool used to measure FACE Conformance for interfaces and applications built according to the FACE Technical Standard. It is designed to be development tool-agnostic. The test suite tests for conformance to the FACE Technical Standard specified APIs for a specific segment. The test suite provides a means to test all requirements identified in the CVM that have the Verification Method marked "Test" and is a mandatory step of FACE Verification testing. The test suite ensures only required APIs are defined, implemented, and/or used based on the specific segment where the tested software resides. This test suite only verifies conformance to the FACE Technical Standard; functional verification is the responsibility of the vendor or test agency.

Note: Users should ensure that the selected CTS version aligns with the same FACE Technical Standard edition.

The FACE CTS is available to users at no cost and can be run during all phases of the software development process to provide feedback to the Software Supplier in support of FACE Conformance. It is recommended that the running of the CTS be included early in the development cycle both for model and software development and be included in any continuous integration processes. It will produce a pass/fail with respect to all Conformance Requirements covered by the test suite plus a detailed report of the test results.

1.2.7 Third-Party Tools for FACE Applications

The Third-Party Tools section of the website (available at www.opengroup.org/face/third-party-tools) is designed to provide information and access to tools intended to support the FACE Ecosystem including creating, editing, debugging, maintaining, updating, and integrating FACE UoCs.

The following text is an extract:

DISCLAIMER: The support products themselves in this listing are not to be perceived as FACE conformant, and have not been vetted or approved by The Open Group FACE Consortium. Listings are the sole responsibility of the posting company and are provided as a resource "as is" to anyone using the directory.

1.2.8 FACE Problem Report and Change Request Process

The FACE Problem Report (PR) and Change Request (CR) process document (available at www.opengroup.org/library/g166) describes the PR/CR process used by the FACE Consortium to track and resolve issues with products created and maintained by the FACE Consortium. This document is intended to be internally focused with the audience being FACE Consortium members. A web-based PR/CR tool has been designed to support the FACE PR/CR process by providing a venue for formal submission of concerns related to the FACE Consortium products, processes, and/or policies, and a means for the Consortium to track issues with published products. Anyone can access the PR/CR tool and create a ticket by setting up an account at the FACE PR/CR Portal link: https://ticketing.facesoftware.org/.

A PR is created by a user to identify an issue or error with published FACE Consortium products that prevents a UoC from obtaining a Conformance Certificate. A CR is created by a user who has identified either an issue with, or a desired improvement to, one or more FACE Consortium products.

1.2.9 FACE Contract Guide

The Open Group FACE Contract Guide: Guidance in Writing Solicitations and Proposals with FACE Requirements (available at www.opengroup.org/library/g18d) serves as a reference for including FACE requirements into a solicitation or proposal. The goal is two-fold:

- To enable a Department of Defense (DoD) Contracting Officer or Program Manager (PM) to include FACE requirements in a solicitation
- To assist Offerors, Government Contractors, and Subcontractors in understanding the FACE requirements in a solicitation and appropriately addressing them in a proposal

A webinar called Introduction to The Open Group FACE Contract Guide: Guidance in Writing Solicitations & Proposals with FACE Requirements is available at www.opengroup.org/library/d161 to facilitate understanding.

2 FACE Reference Architecture Implementation

2.1 Overview

One of the underlying principles of this document is that "learning by doing" enhances understanding. To enrich the learning experience, Basic Avionics Lightweight Source Archetype (BALSA) is used throughout. It serves as a working example for FACE Software Suppliers of how to develop FACE conformant software using the architecture defined in the FACE Technical Standard.

The following sections contain basic information and guidance on how to establish a FACE development environment, including information about the BALSA environment, and how to execute example FACE software environments. Sample artifacts used for FACE Verification in the FACE Conformance process are also provided, along with an introduction to data modeling, the FACE Library, and the FACE PR/CR process. Supporting appendices demonstrate how to obtain, review, and test BALSA, verify a data model using the CTS, provide a data model example, and walk the user through the FACE PR/CR process.

The FACE Reference Architecture defined by the FACE Technical Standard consists of five segments as follows:

- Operating System Segment (OSS)
- I/O Services Segment (IOSS)
- Platform-Specific Services Segment (PSSS)
- Transport Services Segment (TSS)
- Portable Components Segment (PCS)

More specific information can be found in Section 2.3, Figure 1.

A software component or domain-specific data model designed to meet the applicable requirements defined in the FACE Technical Standard is defined as a UoC. It is referenced as a UoC at any point in its development, and becomes a FACE Certified UoC upon completion of the FACE Conformance process. A Unit of Portability (UoP) is a UoC that resides within the PCS or PSSS. These UoPs are inherently more portable due to the requirements placed on these sections by the FACE Technical Standard.

2.2 Setting up an Environment for FACE Development

The FACE Technical Standard is an open standard and thus it is intended not to restrict the development environment or interfere with a Software Supplier's existing practices and tool chains. The FACE Consortium does not endorse or promote specific approaches to development environments. Thus, the choice of development environment is at the discretion of the Software Supplier.

The development environment in the context of this document is a logical construct that may be composed of one or more computers, operating systems, and tool chains. FACE software development is typically embedded "cross-development" where the target is not self-hosting. In other words, there is a difference between the host development environment and the target deployment environment.

The following sections describe aspects that Software Suppliers should consider when engaging in the development of a UoC aligned to the FACE Technical Standard. Software Suppliers who develop software may need to make accommodations to support FACE development and testing.

2.2.1 Target Operating System to Support Deployable Units of Conformance

An operating system meeting the requirements of the FACE Technical Standard is generally designed for applications with real-time constraints that are deployed on custom hardware configurations. In this discussion, the operating system is the target system for deployment.

The FACE Safety and Security Profiles require the operating system to support a subset of POSIX APIs as well as ARINC 653 with time and space partitioning. Although most of the POSIX APIs may be found in "normal" POSIX based operating systems such as GNU/Linux® or FreeBSD®, ARINC 653 support is not. Each edition of the FACE Technical Standard has unique requirements for time and space partitioning, as they pertain to the FACE Profiles.

These "normal" POSIX based operating systems currently do not make any claims of FACE Conformance. To date, testing has demonstrated that no "normal" POSIX based operating system provides the complete set of required APIs for any of the FACE Profiles. However, it is possible for FACE conformant software using POSIX APIs to be developed and debugged using "normal" POSIX based operating systems. This can be done if, and only if, the software restricts itself to the set of FACE APIs that are actually available on the target operating system. The Software Supplier should be fully aware that this is just a debug aid and that the software capability will most likely be deployed on a Real-Time Operating System (RTOS) that meets the requirements of the FACE Technical Standard.

2.2.2 Host Operating System to Support Desired Tool Chain

Many Software Suppliers have existing tool chains to support software development. There are no operating system restrictions in the FACE Technical Standard for hosting analysis, design, and development tools. The FACE Technical Standard was intentionally designed to be development tool-agnostic. As a minimum, the FACE development environment should support development of programs, which execute on the targeted operating system that meets the requirements of the FACE Technical Standard. To aid in ensuring FACE Conformance, the FACE development environment should include the FACE CTS and could include a FACE data modeling tool.

The primary capability required of a development environment for FACE UoCs is support for compiling and linking applications, which execute on the target operating system. These programs will then be loaded onto the target hardware for execution.

The Software Supplier may choose a development environment where all tools operate in a single host operating system, or they may choose (or be required) to use multiple operating systems. It is not uncommon for some tools to execute on Microsoft® Windows® and some to operate under GNU/Linux. For example, the cross-compilation environment may only support GNU/Linux, while the data modeling tools only support Microsoft Windows. In this situation, a common solution is to use virtualization software to host a GNU/Linux environment virtually on the Microsoft Windows host.

A development environment will most likely include support for compilation of programs for the host operating system in a virtualized configuration. The Software Supplier will need this feature if custom utilities are created as part of the development, debug, or test process.

Although there will be common tools across Software Suppliers, each Software Supplier's development environment will likely include a unique mix of tools to satisfy their development requirements. When integrating these different tools, the Software Supplier should be careful to check for prerequisites, compatibility, and conflicts. The Software Supplier's information technology requirements for security and host operating system version will be a factor in the integration of their development environment for UoCs.

For Software Suppliers new to FACE development, it is desirable to explore BALSA or other examples found in the RIG, as well as available tools for FACE software development and integration in an isolated environment such as a virtual machine, cloud, or an independent computer. This allows the user to assess compatibility with existing Software Supplier development tools and practices and to conduct experiments, which can be reconfigured without violating production environment restrictions.

2.2.3 Example Development Environment for BALSA

This section provides a brief description of the development environment used for BALSA.

BALSA is written primarily in C++ with some portions in C. It uses "make" as the software build system and was developed using the GNC Compiler Collection (GCC). The current primary development host system is the GNU/Linux CentOS™ operating system and executed in a virtual machine hosted on Microsoft Windows. The use of a virtual machine allows the exploration of source code and tools without impact to an existing computing environment and allows for use of different tools on different operating systems.

The CentOS operating system was also used as the target environment in which BALSA was originally developed and executed. GNU/Linux is not FACE conformant but is sufficiently aligned to support development of FACE conformant UoCs that require only the POSIX APIs. BALSA has also been successfully compiled and executed on Ubuntu® (x86_64) and Raspbian (ARM) distribution for the Raspberry Pi™.

Although details vary based upon the host operating system to be used, in order to compile BALSA, you must install the native development tool suite. In order to edit the BALSA source code, you must have at least a text editor such as Vim or Emacs.

The BALSA FACE UoP Supplied Model (USM) is in FACE XML Metadata Interchange (XMI) format and can be explored with any environment or Integrated Development Environment (IDE) of choice.

The FACE CTS is an important component in a FACE development environment. The FACE CTS is useful to both the development process as well as for formal verification. Integrating the CTS into the development process allows points of non-conformance to be detected and resolved before formal verification.

Although it is desirable as a learning exercise to build and execute BALSA, you certainly do not have to build it to explore the source code and learn more about the construction of FACE UoCs.

Distributions of the BALSA source code are available on The Open Group FACE website in the New Users – Getting Started section. The direct link to the BALSA website where public distributions can be downloaded is https://www.opengroup.org/face/balsa.

2.3 Development of FACE Units of Conformance

FACE Units of Conformance (UoCs) are developed as modular software components that can be implemented in a variety of ways. They can be designed as a service, or as an object made up of one or more classes (in C++ and Java®) using well-known software engineering patterns.

At a minimum, a FACE UoC should be considered a consumer and/or a provider of a FACE Interface regardless of its design. Its design must meet the UoC requirements specified in the FACE Technical Standard, as well as the requirements for the FACE segment in which it will reside. There are no design restrictions on the development of a FACE UoC outside of those requirements and the allowed APIs that are provided by the FACE OSS for the designated FACE Profile. However, for certain FACE segments, like the TSS, there are different UoC types that can be implemented, each of which will have its own unique requirements.

2.3.1 BALSA – An Example FACE Implementation

2.3.1.1 BALSA: Overview

BALSA serves as a working example for software developers on how to implement the FACE Reference Architecture. It provides a FACE Computing Environment; composed of FACE TSS and FACE IOSS UoCs, as well as the FACE Configuration Services for versions that use FACE Technical Standard, Edition 3.x. BALSA also features a set of PCS and PSSS UoCs that implement a very basic avionics process in order to demonstrate the flow of data throughout a FACE infrastructure.

The PCS and PSSS UoCs co-operate to combine position and altitude information with an aircraft ID and send it out "to the world" as an Automatic Dependent Surveillance – Broadcast (ADS-B) message. The communication paths that connect the UoCs in this example are all internal TSS connections, which can be configured in a variety of ways. IOSS connections are used to receive position data from a "notional" Embedded GPS/Inertial Navigation System (EGI) source, as well as to write the ADS-B message over User Data Protocol (UDP) multicast for potential listeners. To fully demonstrate the flow of data, an additional PSSS is provided to read the ADS-B data using an IOSS UoC that reads in data from a UDP connection.

2.3.1.2 BALSA: Basic Information

The BALSA UoCs were written targeting the Safety Base Profile. The USM for all BALSA UoCs is provided in each BALSA source distribution. See Appendix A for more detailed information on how to obtain, review, test, and verify a data model using the CTS.

Comments in the BALSA source code were written using the Doxygen (www.doxygen.org) markup. This allows for generation of documentation in various formats. BALSA was developed initially on the CentOS GNU/Red Hat Enterprise Linux distribution and has been tested on multiple other distributions, as well as Cygwin™.

BALSA uses a Makefile structure in order to compile a set of executables to demonstrate the operation of the various UoCs. The Makefiles provided can also be invoked in order to compile the BALSA UoCs using the Gold Standard Libraries provided by the FACE CTS. This allows for the user to gain experience in testing a UoC using the FACE CTS via a robust example. For Software Suppliers who are just starting to explore the FACE software environment, it is helpful to realize that a "Hello World" example is too simple to be useful to demonstrate a FACE Reference Architecture. BALSA is a simplified implementation of a FACE Reference Architecture that provides a means to understand how to build a simple UoC and how it will interact with the rest of the system. Additionally, it is intended to provide a scalable baseline for vendors to use as a sample FACE Computing Environment.

The FACE Technical Standard outlines five segments, four of which have UoCs implemented in BALSA (PCS, TSS, PSSS, and IOSS). BALSA does not provide a full OSS. However, for versions that leverage FACE Technical Standard, Edition 3.x, a FACE Configuration Services UoC as well as OSS-provided definitions for POSIX *devctl* are also included. The BALSA UoCs were written using the POSIX functions allowed per the Safety Base Profile. Therefore, any operating system providing the POSIX APIs and a C++ Programming Language runtime is sufficient for integrating a BALSA distribution.

In Figure 1, the components outlined in solid lines are part of the current BALSA architecture.

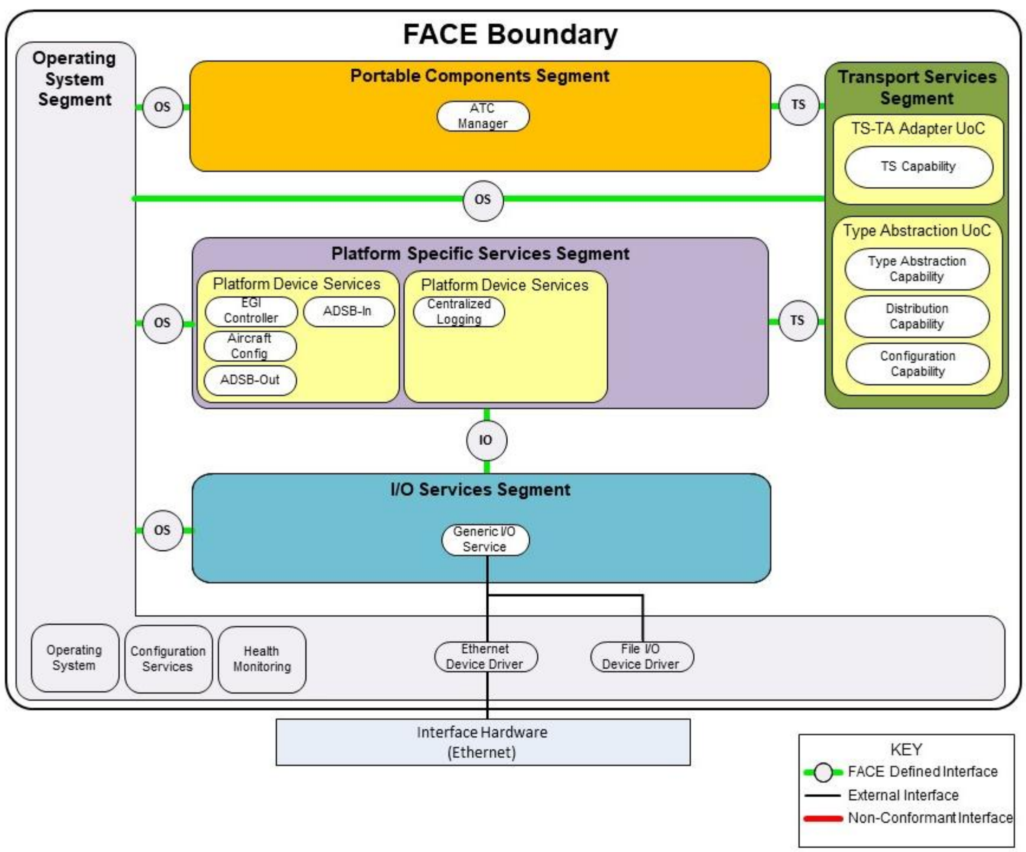

Figure 1: FACE Boundary for BALSA

2.3.1.3 *BALSA: How the Source Code is Organized*

The BALSA source code is organized as a suite of separate processes contained in a single common BALSA application project tree. It is built with an upper-level "Makefile" at the top of the BALSA project tree, and the FACE processes are started by a start-up script called "*launch.sh*" Each version of BALSA contains a README which explains how to compile, run (along with arguments), and test the UoCs provided by the source distribution.

The BALSA source distribution is versioned and released as a source code project. No objects, libraries, or executables are provided to the user. The user is expected to perform all compilation according to the README instructions.

The BALSA source distribution contains five major subdirectories: BALSA, FACE, FIA, libADSB, and PingPong. All of the header files and support classes defined by the FACE Technical Standard are provided within the FACE/ subdirectory. These are all the files with classes and APIs defined within the FACE namespace. In addition, implementations of class definitions provided by the FACE Technical Standard are provided in this subdirectory as well.

The FIA/ subdirectory, whose acronym stands for FACE Implementation Architecture, is where all of the type-agnostic, reusable infrastructure resides. This includes the HMFM implementation for C, the TSS Type Abstraction UoC, the IOSS UoCs, as well as any OSS UoCs provided. The FIA acronym corresponds to the chosen namespace where implementations of FACE Interfaces reside. There is no formality in the chosen acronym. It is merely to identify, in a simplistic way, the purpose of the subdirectory and corresponding namespace.

The BALSA/ subdirectory houses the source code for all PCS and PSSS UoCs, as well as the Transport Services (TS) Interface implementation that corresponds to the BALSA USM. The data structures for the Views/Templates defined in the BALSA USM are contained within this subdirectory tree as well.

The libADSB/ subdirectory provides the source code for encoding and decoding certain ADS-B messages, particularly *position* and *callsign*. The source files contained therein should not be considered a reusable source for an ADS-B library that is poised for any sort of safety certification. However, the source files are reusable as an example or test library for UoCs that require the ADS-B capabilities provided.

The PingPong/ subdirectory provides a benchmarking set of applications, Ping and Pong, to test delivery time of messages over the BALSA TSS. It is a helpful tool for testing TSS modifications, as well as for measuring network latency.

2.3.1.4 BALSA: Obtaining BALSA

Distributions of the BALSA source code are available on The Open Group FACE website in the New Users – Getting Started section. The direct link to the BALSA website where public distributions can be downloaded is https://www.opengroup.org/face/balsa.

It is important that all versions of the BALSA source distribution not retrieved from The Open Group public site are checked for appropriate Distribution A markings, featuring a control number. If a control number is not present, the distribution is not valid for public use.

2.3.2 FACE Conformance Test Suite

The FACE Conformance Test Suite (CTS) is the test tool used to verify conformance to the applicable FACE Technical Standard. The test suite ensures only required APIs are defined, implemented, and/or used based on the specific segment where the tested software resides.

2.3.2.1 Obtaining the FACE Conformance Test Suite

The FACE CTS may be downloaded from the FACE Consortium website at https://www.opengroup.org/face/conformance-testsuites. A list of prerequisites is provided on the website.

There is a CTS for each edition (and corrigendum) of the FACE Technical Standard. Download the *tar* or *zip* file for the version of the CTS aligned to the edition of the FACE Technical Standard you used to develop your products. Once that is completed, follow the instructions listed in the CTS manual, located in the "docs" file.

2.4 Data Modeling Using the FACE Technical Standard

To better appreciate FACE data modeling, it is important to understand its purpose within the overall FACE Data Architecture. The following sections are an overview of the FACE Data Architecture and how data modeling fits into it.

The Reference Implementation Guide (RIG) for the FACE Technical Standard, Edition 3.0, Volume 3: Data Architecture (see Referenced Documents) provides further information on the purposes behind data modeling and an in-depth description of modeling concepts and methods through the use of examples.

2.4.1 FACE Data Architecture

The FACE Data Architecture is designed specifically to achieve its key objectives by developing an architecture supporting engineering design specifications for data selection and data organization such that there are no ambiguities in these designs for the purpose of component integration. By providing for unambiguous descriptions of data elements, developers can specify information with sufficient specificity to increase interoperability of FACE UoPs.

The FACE Data Architecture is made of a Data Model Language, a common SDM that establishes foundational elements for modeling, and a Conformance Policy with accompanying data model CTS.

2.4.1.1 FACE Data Model Language

The FACE Data Model Language utilizes the Meta Object Facility (MOF) metamodel language as its foundation, two grammar-based languages, and Object Constraint Language (OCL)-defined constraints to define the structure and rules for construction, interaction of model elements, and software code generation. Figure 2 shows the various constituent models that make up the FACE Data Architecture and the relationships between the models. The progression from left to right is from abstract to concrete. With each level of refinement, the degree of specificity increases, moving the model closer to a complete definition of the software application solution required for generation of code. The FACE Data Model Language is utilized across the data modeling element groupings and the UoP Model element groupings.

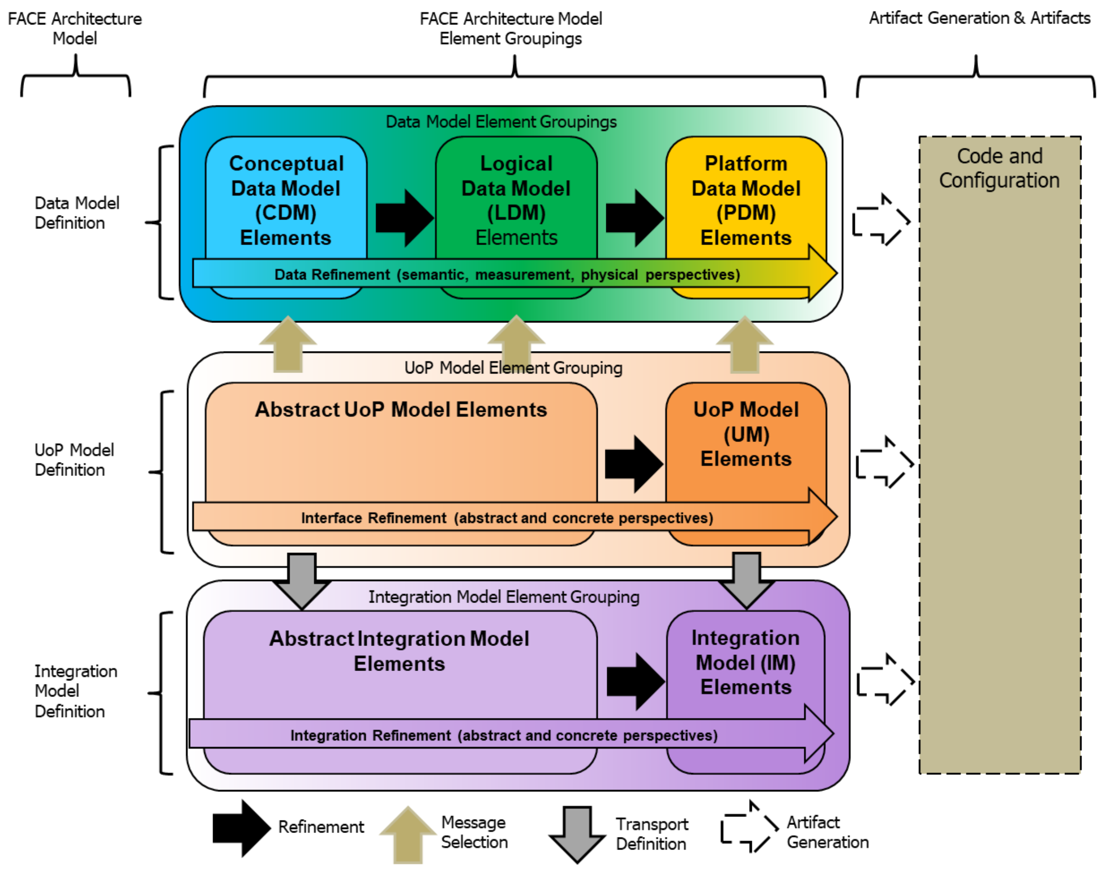

Figure 2: FACE Architecture Models without Traceability Model

The first of the architectural model elements, the Data Model, is used to define data elements with unambiguous specificity for use in the FACE Technical Standard. The Data Model is divided into three different data elements for use by FACE UoP or Domain-Specific Data Models (DSDM) as defined by the FACE Technical Standard. DSDM is a data model designed to the FACE Data Architecture requirements. It captures domain-specific semantics and generally does not contain UoP Models. The Data Model consists of the following:

- The Conceptual Data Model provides the semantic definition of the entities and their relationships characterized by observables
- The Logical Data Model adds measurement information to each characterization by defining value type, units, measurement, and frame of reference (through the measurement system)
- The Platform Data Model adds physical data type information to the logical measurement characterization
- The UoP Model provides the definition of a software component and its defined interfaces utilizing the Platform Data Model elements
- The Integration Model provides a mechanism to describe the TSS integration details between two or more UoPs and is not part of the Data Model
- The Traceability Model, not shown in Figure 2, provides a mechanism to trace model elements to an external model and also is not part of the Data Model

2.4.1.2 *Shared Data Model*

The Shared Data Model (SDM) establishes a foundation of core data elements used as building blocks to create all other data models. Because the SDM is Configuration Control Board (CCB)-managed, it is reliable and provides confidence in the core modeling elements and utility of reuse and potential of reduced data conversion needs.

2.4.1.3 *Conformance Policy*

The Conformance Policy, and supporting Data Model CTS, maintains the integrity of FACE data models for the purpose of data model reuse and integration. Since the USM and DSDM must be validated as conformant in order to be placed in the FACE Library, developers can expect a certain level of data integrity in these models.

Note: The Conformance Policy and tests do not perform a qualitative analysis of data model elements. Conformance only means that the elements adhere to the structure and constraints of the FACE Data Architecture.

2.4.2 Building a Data Model Using the FACE Technical Standard

In order to provide a useful data model, a variety of data modeling tools have been developed and are available to assist with the creation of a data model. If you are new to data modeling, training can be obtained to help you better understand how to use some of these tools. The FACE Consortium has provided a location under the "Documents & Tools" tab on The Open Group FACE Consortium website called "Third-Party Tools" (www.opengroup.org/face/third-party-tools). Here you can find links to some of the data modeling tools.

2.4.3 Data Model Testing in the FACE Conformance Test Suite

The FACE USM's importance for the CTS goes beyond the actual data model portion of the tests. The USM is used by the test suite to generate header files, which are utilized in some of the segment-specific tests. This means that if there are failures in the data model tests in the CTS, as well as in the non-data model-related tests, the failures in the data model tests might have caused the failures in the other portions of the test suite; and, therefore, the failures in the data model tests most likely need to be addressed first.

Specific information about how to obtain, view, test, and verify a FACE UoP data model can be found in Appendix A.

2.4.4 USM Data Model Example – BALSA Aircraft Configuration PSSS

The Aircraft Configuration component within BALSA is a PSSS component. In the example outlined in Appendix B, the reader will be shown various packages within the BALSA UoP data model that specifically pertain to the Aircraft Configuration component.

2.5 Verification Evidence and Conformance Artifacts

For a Software Supplier to achieve FACE Conformance for their FACE UoC, the supplier must first undergo the FACE Verification process. One of the requirements for verification is for the UoC to be accompanied by Verification Evidence. Verification Evidence is defined as the documentation a Verification Authority (VA) requires from a Software Supplier that details how the FACE Technical Standard requirements have been satisfied. It is part of the overall Software Verification Package.

The starting point for generating Verification Evidence is the FACE CVM, found by visiting the FACE Consortium Conformance Publications & Tools web page at www.opengroup.org/face/conformance-publications-and-tools.

2.5.1 Conformance Verification Matrix

The Conformance Verification Matrix (CVM) contains requirements that a FACE UoC must satisfy based on its respective segment in the FACE Reference Architecture. Accompanying these requirements are descriptions of conformance artifacts intended to be used as evidence of conformance.

Each approved version of the CVM corresponds to a unique version of the FACE Technical Standard. It is published in a spreadsheet format and is intended to be filtered depending on the FACE segment where the candidate UoC resides. There is also a column that allows the user to filter based on whether the referenced text requires Verification Evidence or not. To elaborate, some of the CVM rows contain section headers and supporting text for requirements.

For detailed information on how to use the CVM, consult the FACE Conformance Verification Matrix User's Guide (see Referenced Documents).

Once a CVM has been filtered by the FACE segment, the user must then consider the Verification Method and Conformance Artifacts columns that will dictate what must be supplied to satisfy the requirement. A Conditional Requirements column also provides details on what preconditions apply to the requirement (e.g., C++ Programming Language). If a filled out CVM is to be used as part of the Verification Evidence (*in lieu* of using a separate document to list all requirements), then the Software Supplier must input the respective artifact reference and accompanying notes in their respective columns. It should be noted that all requirements that have a Verification Method of "Test" are ones that will be verified via the VA using the FACE CTS. These requirements will also have "Test Suite" listed in the Conformance Artifacts column.

In Figure 3, the candidate UoC resides in the PSSS. In the first row, the requirement states that when implementing a component framework, a PSSS must do what is referenced. However, the UoC used in this example does not provide one. The Verification Method is "Inspection", which means that the VA will verify this requirement via inspection of the referenced artifacts. Since the PSSS does not provide a component framework, the user states in the Cross-Reference and Notes columns that the UoC does not provide the referenced feature.

In the second row of the example CVM excerpt in Figure 3, the sample PSSS UoC does indeed retrieve configuration information. The Verification Method is once again "Inspection"; however, the artifact type required is UoC Design. The Software Supplier Artifacts referenced to show adherence to this requirement are the Software Architecture Description (SAD) and Software Design Description (SDD) for the sample PSSS component. In the Verification Notes, the user has listed a particular requirement in the SDD for the VA to reference.

FACE Segment	Technical Standard for FACE™ Reference Architecture Edition 3.0	Verification Method	Conformance Artifacts	Software Supplier Artifact Cross-Reference	Verification Notes
PSSS	10. When implementing a Component Framework, a PSSS UoC shall do so in accordance to the requirements in Section 3.6.1.2.	Inspection	Per Referenced Section	UoC does not use a Component Framework; Does not apply	UoC does not use a Component Framework; Does not apply
PSSS	11. When a PSSS UoC retrieves Configuration Information, the UoC shall use the Configuration API as defined in Section 3.2.5.	Inspection	UoC Designs	SAD SDD	See requirement SDD_016 in SDD
PSSS	12. When using Centralized Configuration, a PSSS UoC shall use the TSS API.	Inspection	UoC Designs	UoC does not use a Centralized Configuration; Does not apply	UoC does not use a Centralized Configuration; Does not apply

Figure 3: Sample FACE Conformance Verification Matrix Excerpt

The CVM User's Guide contains clarification on verification methods, as well as further details on the different verification categories that can be found in the Conformance Artifacts column of the CVM. Each category lists details on what the verification review for the given item shall entail, as well as examples of the type of artifacts that may be used to satisfy the requirement.

2.5.2 Conformance Artifacts

Conformance artifacts are comprised of the documents that a Software Supplier submits as part of the Verification Evidence to show that a FACE UoC conforms to the requirements in the FACE Technical Standard. The artifacts are not required to be in any particular form, so long as they sufficiently satisfy what is expected.

The following list contains commonly used artifacts that a Software Supplier may submit. However, the names associated with the content of the artifacts are not required:

- Software Architecture Description (SAD)
- Software Design Description (SDD)
- Interface Design Description (IDD)
- Interface Control Document (ICD)
- Interface Requirements Specification (IRS)
- Software Requirements Specification (SRS)
- Software Test Plan (STP)
- Software Test Report (STR)
- Software Product Specification (SPS)
- Software User Manual (SUM)

2.5.2.1 Example Artifact Cross-Reference and Verification Note

In the example shown in Section 2.5.1, Figure 3, the PSSS intended for FACE Verification is the BALSA ADS-B Out. The second row of Figure 3 shows the following requirement:

- When a PSSS UoC retrieves Configuration Information, the UoC shall use the Configuration API as defined in Section 3.2.5

The Verification Method for this requirement is Inspection, which means that the VA will require examination of the UoC and its associated documentation to determine conformance. In the Conformance Artifacts column, it specifies UoC Designs. This indicates that a review of component design or architecture is needed. The artifacts presented as Cross-References are the SAD and SDD.

The SAD for the BALSA ADS-B Out features a diagram of all interfaces used by the PSSS component. Among them is the FACE Configuration Services Interface. However, as specified in the Verification Notes of the CVM, requirement SDD_016 of the SDD is the reference to which the user is directing the VA. As specified in the CVM User's Guide, a direct reference/pointer to a location in the artifact which confirms the requirement is required.

Figure 4 shows SDD_016 as it reads in the BALSA ADS-B Out SDD document. In this requirement, we have the behavior Configuration, which shall retrieve TSS Connection Name (a variable) using the Configuration Services Interface. There is a function within the source code where this behavior will take place, as well as the CVM RowID that is satisfied by this behavior. Also, three SRS requirements trace to this item as well. They are as follows:

- SRS_001 – the BALSA ADS-B Out component shall only use the interfaces defined in FACE Technical Standard, Edition X.x
- SRS_017 – the BALSA ADS-B Out component shall perform configuration to acquire its resources required for execution
- SRS_010 – the BALSA ADS-B Out component shall use the OSS Configuration Services Interface for retrieving all information required to configure the UoP

Req #	Requirement Body	Class	Function(s)	SRS ID	CVM Requirement
SDD_016	Configuration shall retrieve TSS Connection Name using the Configuration Services interface	BALSA::ADSB_Out	Configure()-> Create_TSS_Connections()	SRS_001 SRS_017 SRS_010	580

Figure 4: Artifact Example

In the example in Figure 4, a CVM requirement maps to a specific behavior in the component, described in the SDD document. The SDD element (behavior) is a realization of SRS requirements and maps to places in the ADS-B Out source code.

The SAD diagrams in the BALSA ADS-B Out artifacts are in the context of the FACE Reference Architecture, and show that the component is a user of the FACE Configuration Services Interface. It also shows TSS connections, based on their respective name, modeled in the USM supplied for ADS-B Out.

For a full example of conformance artifacts for a UoC, each distribution of the BALSA UoCs includes at least one set of artifacts for a UoC, complete with SRS, SDD, SAD, STP, and STR documents.

2.5.2.2 *FACE Software Supplier's Statement of Conformance*

The FACE Software Supplier's Statement of Conformance is the form that must be submitted to the VA when the verification process is initiated. Specific VAs may require other forms to accompany it; however, this document is a FACE requirement.

It contains the following sections that a Software Supplier must fill out with regards to their candidate UoC:

- Unit of Conformance Supplier Information
- UoC/UoP Package Information
- Scope of Information – this section is where the supplier will indicate the edition of the FACE Technical Standard that the UoC implements, its respective FACE segment, and the FACE Profile it targets
- Conformance Test Environment – this section is where the supplier indicates the version of the FACE CTS that was used for testing, as well as the CVM version for the artifacts
- Implementation Details – if a supplier chooses to, it may use this section to populate the applicable CVM requirements for the UoC, the requirement text, and the implementation that satisfies the requirement

It should be noted that if the implementation details section is not populated in the Conformance Statement, it must be supplied elsewhere in the conformance artifacts.

2.6 Introduction to the FACE Library

2.6.1 Purpose

The following scope-limited content is taken from the FACE Library Requirements, Version 3.0 (see Referenced Documents).

The purpose of the FACE Library is to enable the exchange of information between Software Suppliers, VAs, and the FACE Certification Authority (CA), and to securely and efficiently collect information that supports the FACE Conformance process.

A Software Supplier would use the FACE Library to:

- Discover FACE Consortium approved or accredited VAs
- Search the FACE Registry for FACE Certified Products
- Submit and track progress of software products during the FACE Conformance process
- Submit FACE Certified Products for listing in the FACE Registry
- Manage metadata information about FACE Certified Products listed in the FACE Registry

The web-based FACE Conformance Workflow Tool enables information about a FACE UoC to flow from Software Suppliers into the FACE Registry. The Tool is designed to minimize the impact of collecting required information about UoCs on Library users; i.e., the Software Suppliers, FACE VAs, and FACE CAs. The Portal is also designed to maintain security protocols and protect intellectual property and data rights. The FACE Library Administrator manages library operations.

2.6.1.1 FACE Conformance Process

The FACE Consortium has defined a process for developing, verifying, and certifying UoCs. Once UoCs are certified, they will be discoverable by FACE Consortium members.

The FACE Library Portal has been developed to support each of the steps in the FACE Conformance process that has been defined by the FACE Consortium, as illustrated in Figure 5.

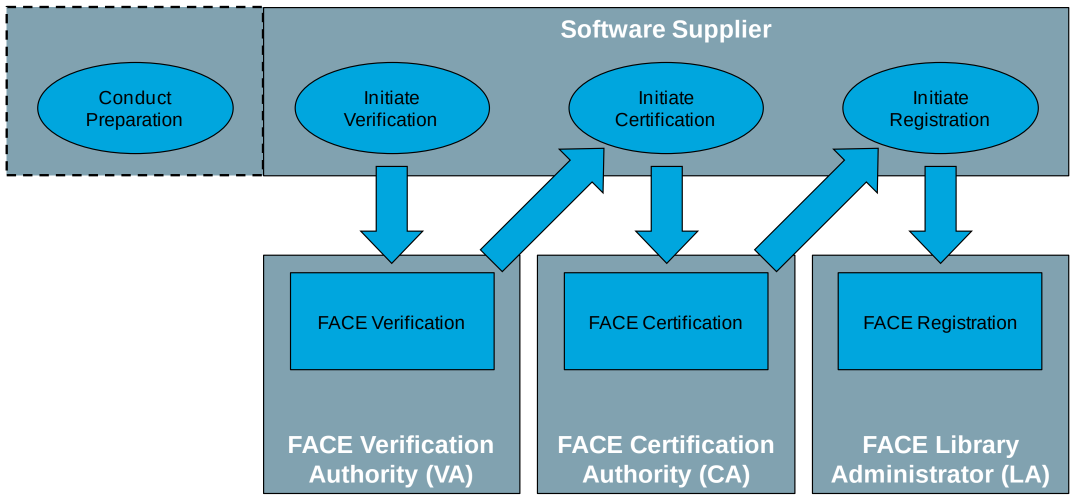

Figure 5: FACE Conformance Process

2.6.1.2 FACE Library Users

There are three categories of user:

- Software Supplier users
- Verification Authority (VA) users
- Certification Authority (CA) users

Users access the FACE Library and Registry with a user account. User accounts are, by default, created with Software Supplier user privileges. Software Supplier users are able to search the FACE Registry and can initiate UoCs into the FACE Conformance workflow. The scope of this document is limited to Software Supplier users.

VA users are able to provide information in the Conformance Workflow Tool related to the verification of a FACE UoC.

CA users can approve a UoC to be published into the FACE Registry.

Each of these users will interact with the Library Conformance Workflow Tool from the User Dashboard.

2.6.2 FACE Library User Account Creation – for Software Suppliers

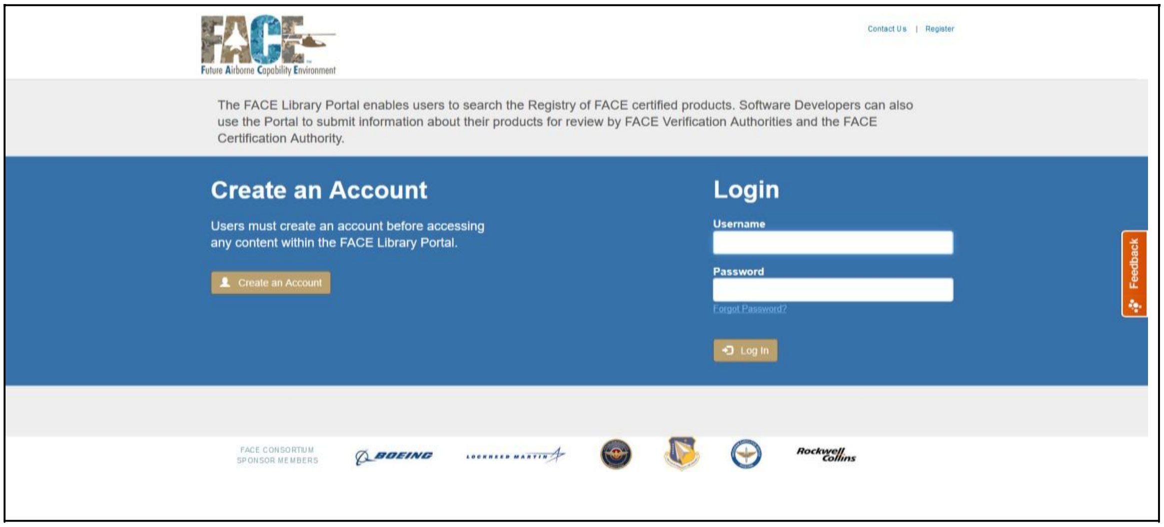

Figure 6: Register for a Library Account

To create a user account:

- Log on to facesoftware.org
- Click on "Create an Account", then enter the required information
- When all required fields are complete, click the "Create Account" button
- The Software Supplier will receive an email to verify creation of the account – follow the directions in the email to complete account creation

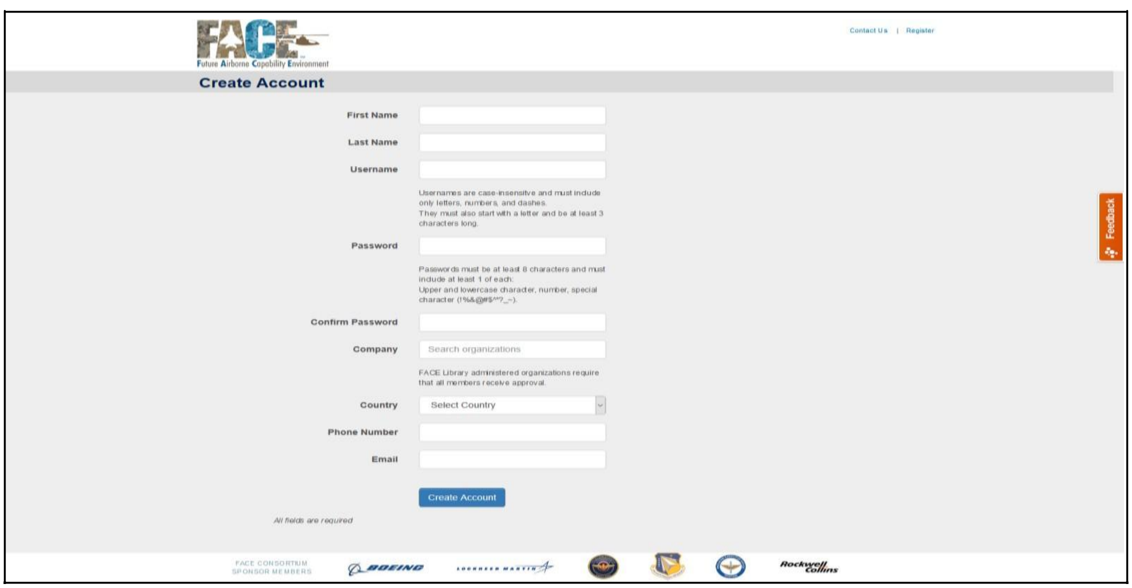

Figure 7: Create an Account

2.6.3 FACE Problem Report and Change Report Ticketing System

The FACE Problem Report (PR) and Change Request (CR) Ticketing System provides FACE Software Suppliers with an avenue to seek clarifications and alleviate issues regarding products and tools created and maintained by the FACE Consortium. For a complete discussion of the FACE PR/CR process, refer to the FACE PR/CR process document (see Referenced Documents).

The FACE PR/CR Portal link is ticketing.facesoftware.org. The login screen is illustrated in Figure 8.

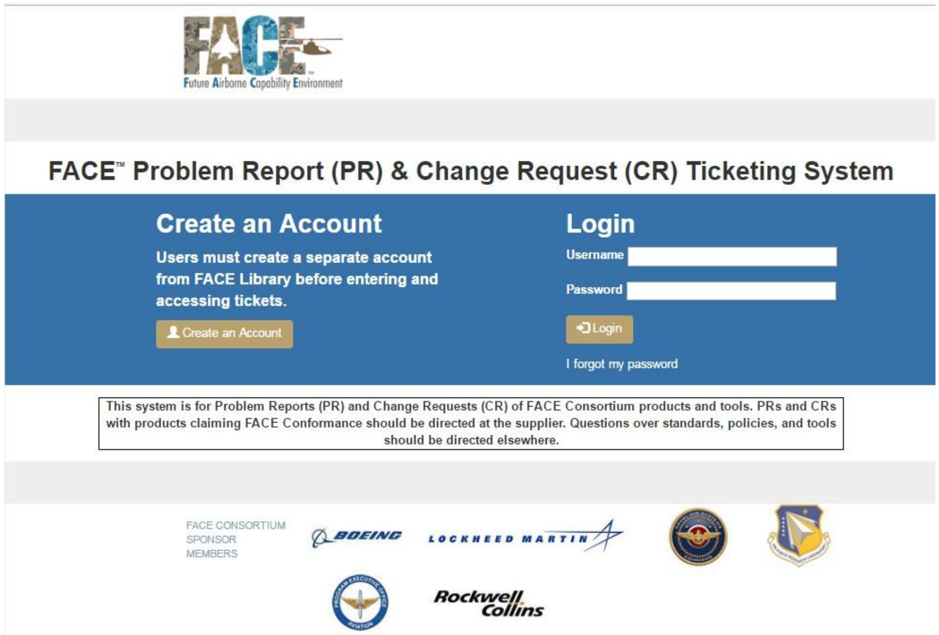

Figure 8: Create a PR/CR Account

See Appendix C for examples of the FACE PR/CR process in use.

A FACE 3.0 Shared Data Model and Conformance Test Suite Example

A.1 Overview

An example FACE UoP Supplied Model (USM) data model has been developed by FACE Consortium members. These, as well as other products, are available to assist FACE software product development efforts. This appendix points to these products and instructs how to obtain, view, test, and verify a FACE data model using the FACE Conformance Test Suite (CTS) tool.

A.2 FACE Data Models and Data Model Governance

A.2.1 FACE Shared Data Model

The FACE Consortium Documents & Tools web page contains links to many of the published documents and software tools, including Shared Data Models (SDM) and Data Model Governance Plans – see www.opengroup.org/face/docsandtools. There are SDMs and Governance Plans for each edition of the FACE Technical Standard, starting with Edition 2.0. These products have been vetted and approved by The Open Group FACE Consortium.

DOCUMENTS & TOOLS

Published Documents

(subset of the entire listing)

- Data Model Guidance

  — FACE Shared Data Model Governance Plan, Edition 3.1

  — FACE Shared Data Model, Edition 3.0.x

Figure 9: FACE Data Models, Edition 3.0

Note: The following guidance (within this appendix) assumes use of FACE CTS Version 3.0.0, and use of the Basic Avionics Lightweight Source Archetype (BALSA) USM based upon the FACE SDM Version 3.0.4. No third-party tools are needed to follow through the remainder of this appendix.

A.2.2 BALSA UoP Supplied Model

The BALSA USM, source code, and artifacts are available on The Open Group FACE website in the New Users – Getting Started section. The direct link to the BALSA website where public distributions can be downloaded is https://www.opengroup.org/face/balsa.

A.2.3 Obtaining the Conformance Test Suite

There is a link on the FACE Consortium Documents & Tools web page to the FACE CTS page. It includes all current versions of the FACE CTS including the CTS Guide and installation packages for both Microsoft Windows and Linux. These can be found at https://www.opengroup.org/face/conformance-testsuites.

FACE Conformance Test Suite(s)

(subset of the entire listing)

FACE Edition 3.0 – Conformance Test Suite, Version 3.0.0

- 3.0 CTS for Windows
- 3.0 CTS for Linux
- 3.0 CTS Guide

Figure 10: FACE Conformance Test Suite for the FACE Technical Standard, Edition 3.0

Note: The following guidance (within this appendix) assumes use of FACE CTS Version 3.0 and use of the ADS-B USM based upon the FACE 3.0 SDM.

A.2.4 Launch and Configure the Conformance Test Suite

Note: The CTS Guide should be carefully followed with respect to installation.

To launch the FACE CTS double-click (or *run.sh*) the *runConformanceTest.sh* file.

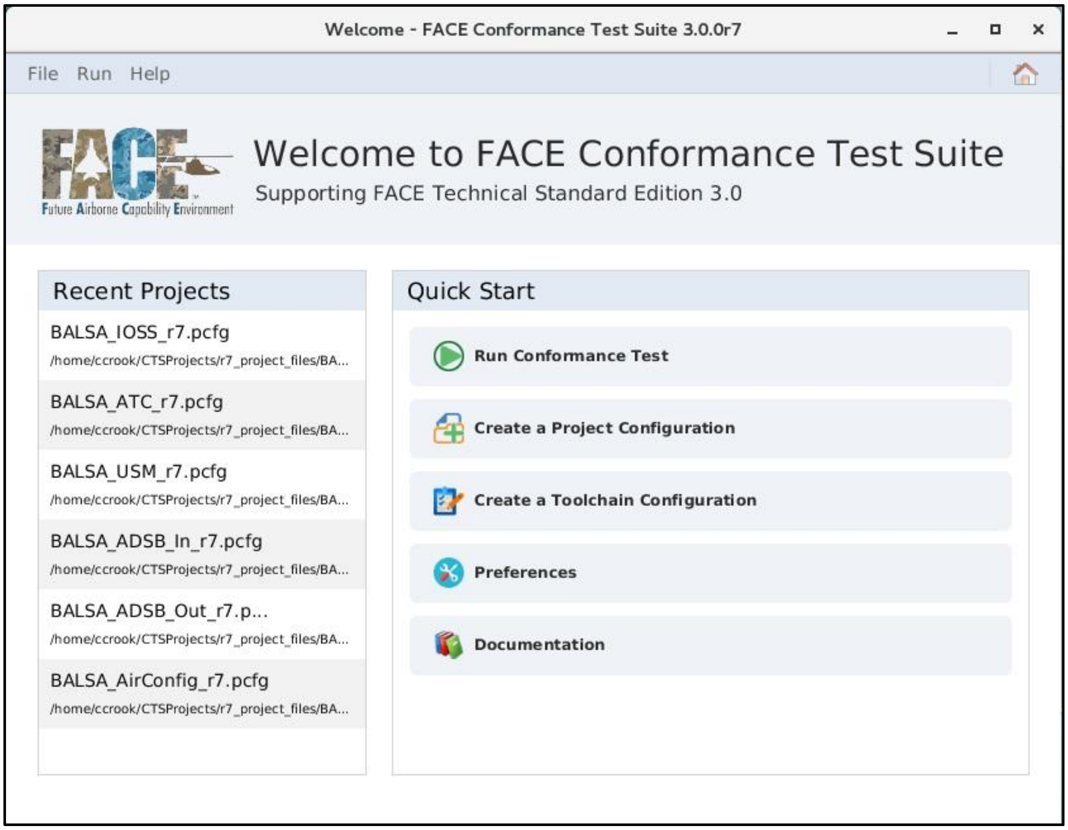

Figure 11: FACE Conformance Test Suite Dialog, Version 3.0

A.2.5 Configure the Shared Data Model

Within the FACE Conformance Test Suite dialog, select the Data Model tab to launch the Data Model configuration panel. Then, click the ellipsis […] button to the right of the Shared Data Model field and navigate to the location of the FACE SDM file. Select [Open].

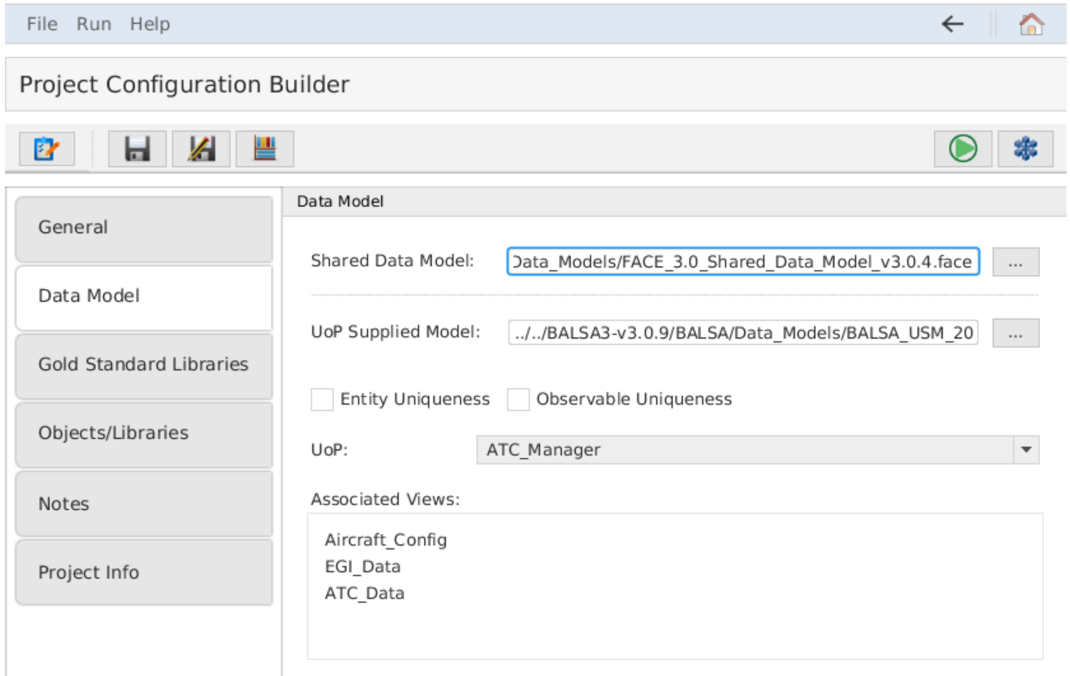

Figure 12: FACE Conformance Test Suite Shared Data Model Dialog

A.2.6 Configure the ADS-B Data Model

Click the ellipsis […] button to the right of the UoP Supplied Model within the Data Model configuration panel and navigate to the location of the USM file. Select [Open].

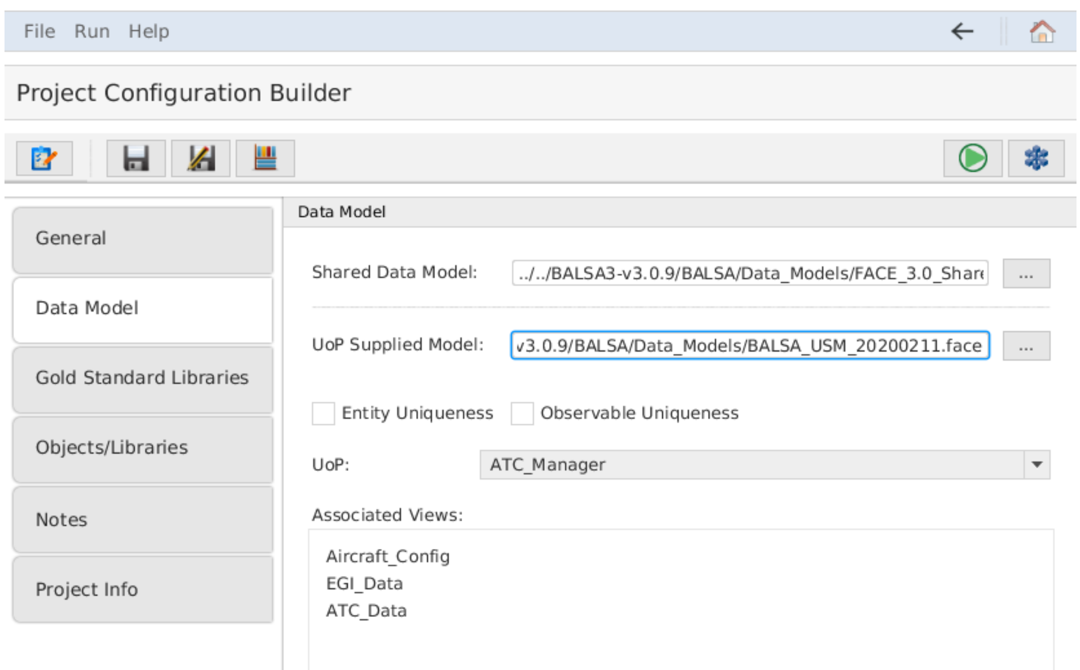

Figure 13: FACE Conformance Test Suite UoP Supplied Model Dialog

Once the SDM and USM have been selected and imported into the CTS, then the Units of Portability (UoP) can be configured as shown in Figure 14.

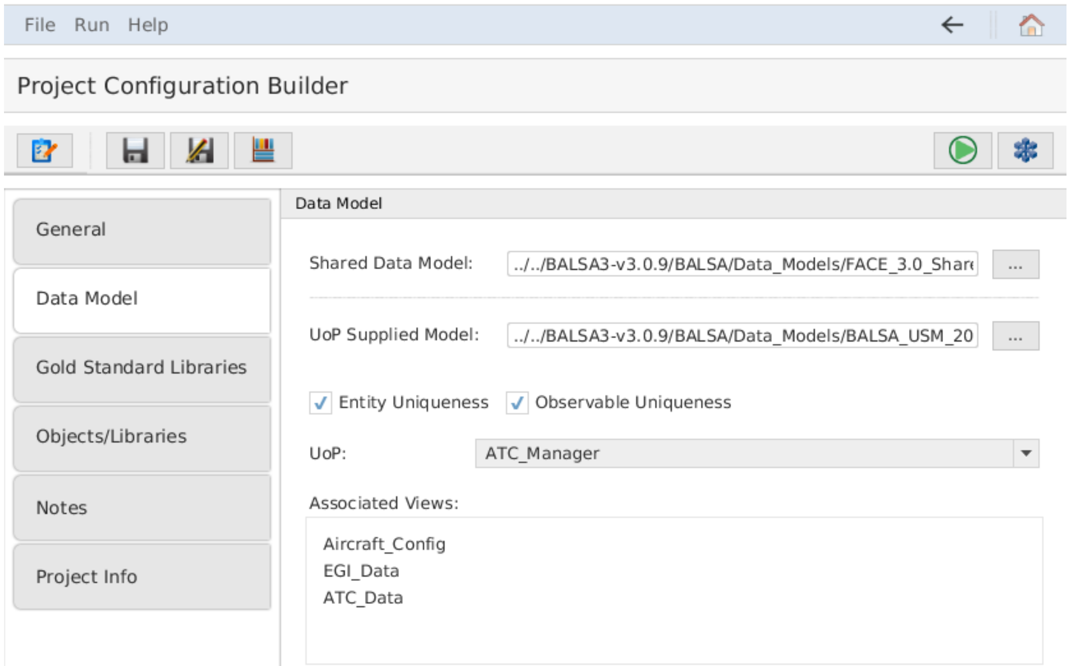

Figure 14: Configuring Units of Portability

Select [Save] or [Save As]  to save the current test configuration.

A.2.7 Run the FACE Data Model Conformance Test

After selecting the SDM and USM files, the CTS can check the data model for FACE Conformance.

Within the CTS main dialog screen, select the Test Data Model Only icon  to begin testing.

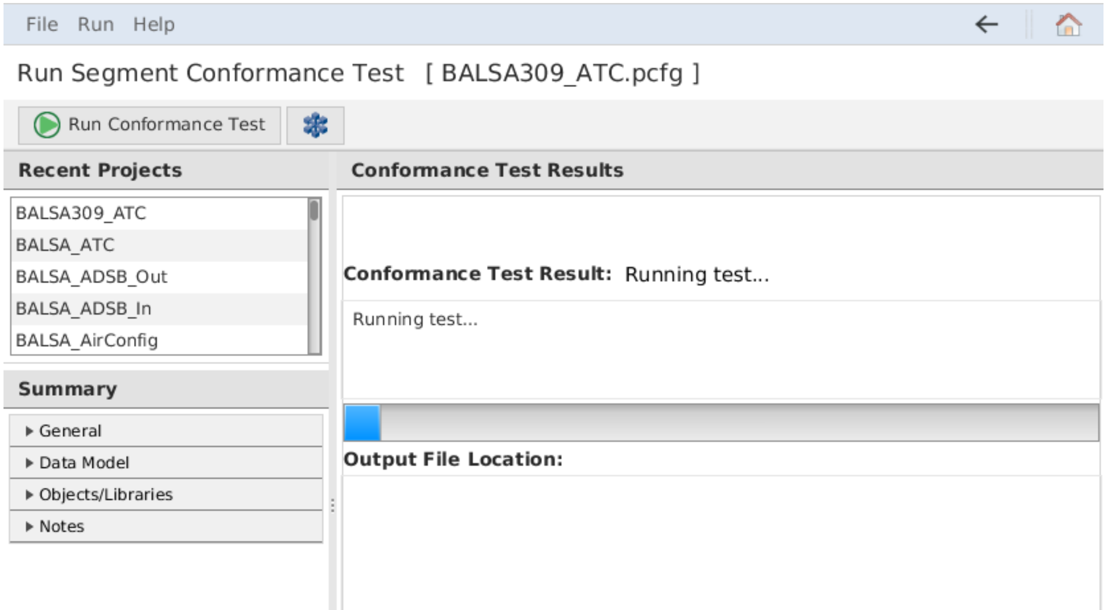

Figure 15: Running the Data Model Test

A.2.8 Interpret Results

The results of the Data Model conformance test are automatically displayed to the screen at the end of the test along with the location of the detailed output report in PDF format.

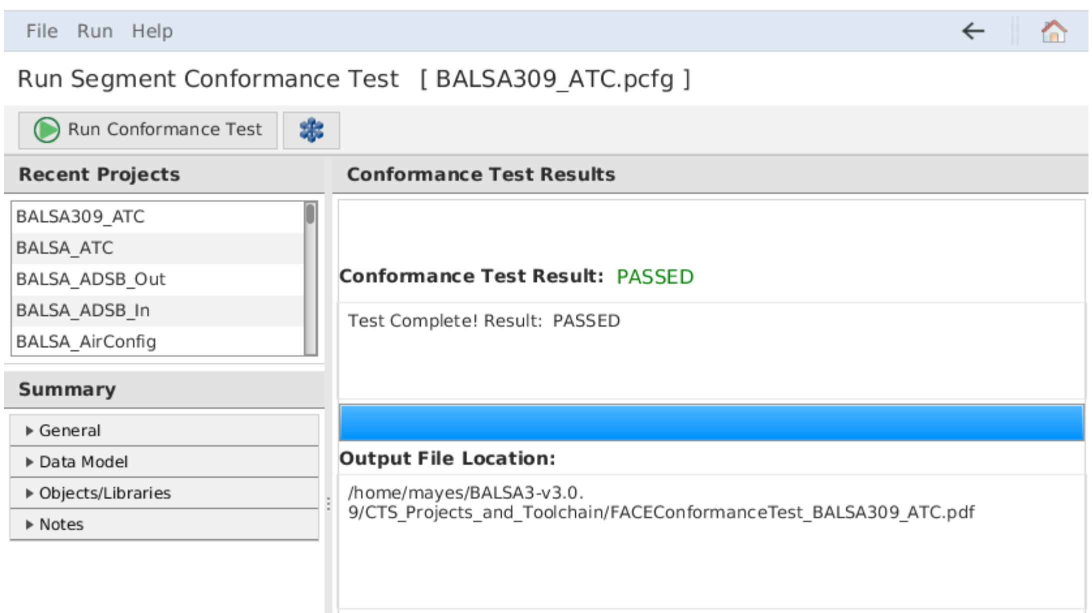

Figure 16: Data Model Conformance Test Results

A.2.9 Projection Validator

After the Data Models have been confirmed to be conformant by the CTS, the next step is to run the Project Validator by clicking the  highlighted icon in Figure 17.

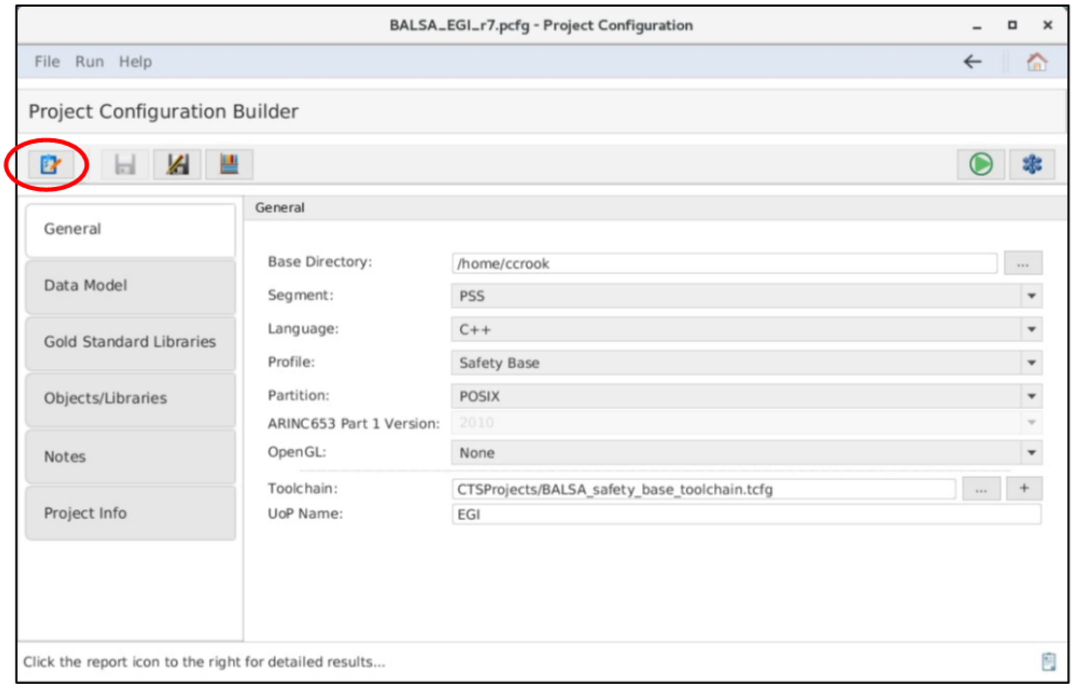

Figure 17: Project Validator

This process checks the current project configuration file and the identified tool chain configuration file to make sure they are valid for this version of the CTS. If successful, the notification given in Figure 18 will appear.

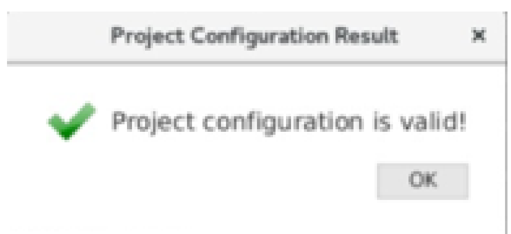

Figure 18: Check Configuration Passed

If the configuration check fails, the notifications given in Figure 19 will appear.

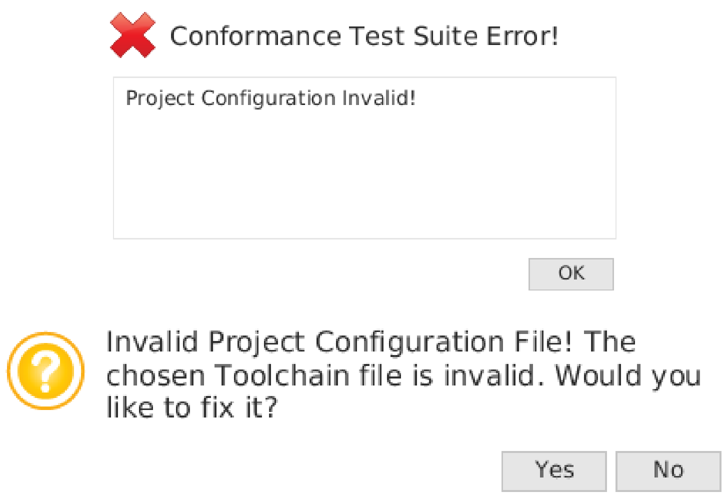

Figure 19: Check Configuration Failed

If this occurs, it means there is something wrong with the CTS configuration file or the tool chain file. It is up to the user to correct this issue, which is beyond the scope of this document.

A.2.10 Run the Conformance Test

Finally, the CTS can run to check the USM and any included FACE segments (or UoPs) for FACE Conformance.

The process is started by clicking the  icon. Be patient, as this process can be very time-consuming, especially on older hardware platforms.

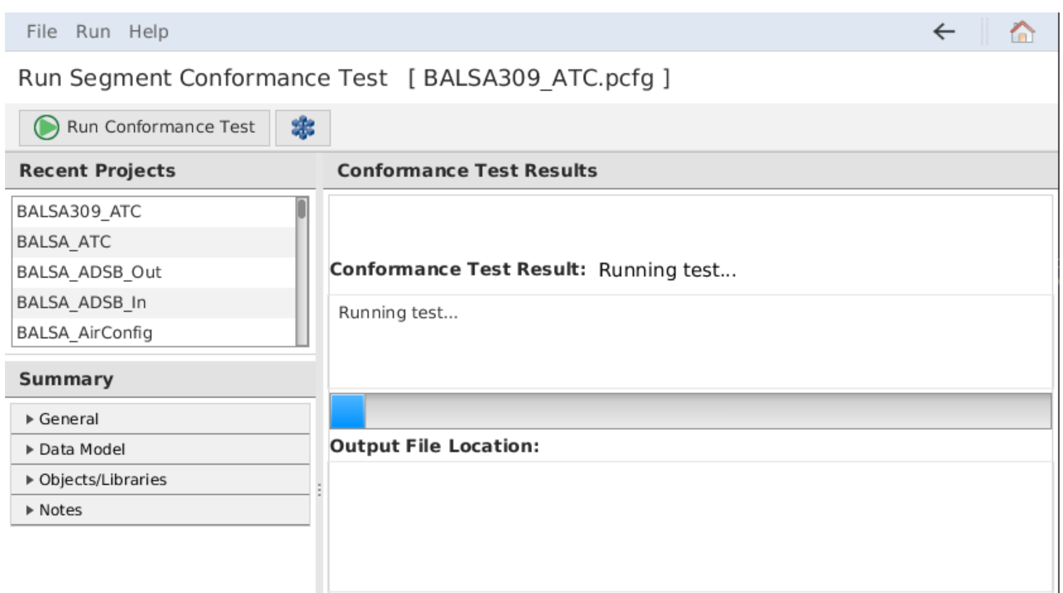

Figure 20: Running the Conformance Test Suite

A.2.11 Test Results

Once the CTS execution is complete, the test results and location of the output file will be displayed.

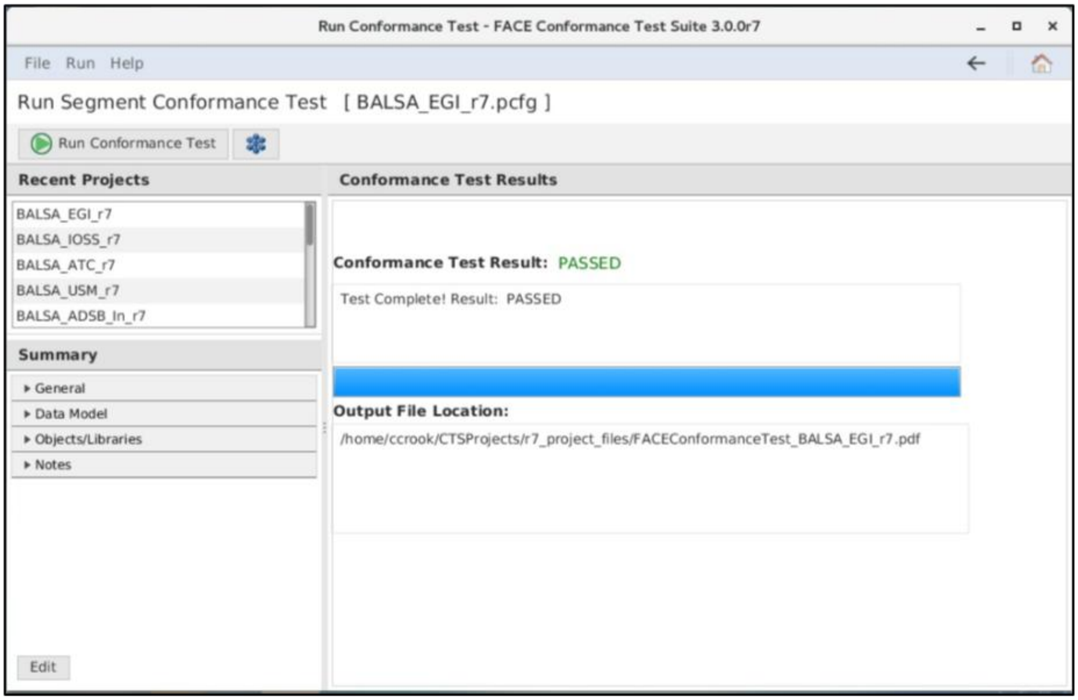

Figure 21: Test Results

B Data Model Example

In Section 2.1 it was noted that a software component or domain-specific data model designed to meet the applicable requirements defined in the FACE Technical Standard is defined as a Unit of Conformance (UoC). It is referenced as a UoC at any point in its development, and becomes a FACE Certified UoC upon completion of the FACE Conformance process. A Unit of Portability (UoP) is a UoC that resides within the Portable Components Segment (PCS) or Platform-Specific Services Segment (PSSS). These UoPs are inherently more portable due to the requirements placed on these sections by the FACE Technical Standard. The terms UoC and UoP are used in this appendix.

B.1 BALSA – Aircraft Configuration PSSS

The Aircraft Configuration UoC within BALSA is a PSSS UoC that receives Aircraft ID and Tail Number data from the I/O Services Segment (IOSS) and transmits it over the Transport Services Segment (TSS) to the ATC Manager PCS UoC. In the example outlined in this appendix, the reader will be shown various packages within the BALSA UoP Supplied Model (USM) that specifically pertain to the Aircraft Configuration UoC. For a more expanded example showing multiple UoCs that interact with one another, consult Chapter 6 (Example Data Models) of the Reference Implementation Guide (RIG) for the FACE Technical Standard, Edition 3.0, Volume 3: Data Architecture (see Referenced Documents).

B.2 Aircraft Configuration Data Model Organization

The organization of the Aircraft Configuration data model uses all four aspects of the FACE data model structure:

- The Conceptual Model – showing observables, entities, and views
- The Logical Model – showing entities, entity realizations of conceptual entities, views, view realizations of conceptual views, measurements and their realizations of observables, and measurement axes
- The Platform Model – showing physical data types and their realizations of measurements, entities, entity realizations of logical entities, views, and view realizations of logical views
- The UoP Model – showing the individual UoP, *MessageTypes*, and templates interfacing a UoP via *MessageTypes*

B.3 Aircraft Configuration Data Model Elements

The Aircraft Configuration UoC handles two types of data: Aircraft ID and Tail Number. The modeling of the PSSS UoC concerns only those data values.

B.3.1 Conceptual Model

The first task the modeler performs is to identify the conceptual elements to be represented in the model. This includes fundamental entities and associations about pertinent items and any observable or informational attributes those possess. Rather than create from scratch, the modeler should first explore the Shared Data Model (SDM) for existing model elements and incorporate them into the model being constructed using a data modeling tool. New elements should only be created when there does not exist a corresponding element in the SDM. Remember new elements to the SDM are CCB controlled.

The modeler identified an entity called Aircraft, which is composed of two composition data attributes, named in the entity as:

- *aircraftID*
- *tailNumber*

It was decided that both *aircraftID* and *tailNumber* should have the informational observable, Identifier, as their type. Unfortunately, an entity's composition cannot have the same observable listed twice for two different attributes. To resolve this situation the FACE Technical Standard, Edition 3.0 permits multiple conceptual composition attributes having the same observable type to be represented by a single composition data attribute and observable. Because of the FACE Technical Standard, Edition 3.0 solution, the modeler represented the *aircraftID* and *tailNumber* compositions by the single composition *elementID* with observable type, Identifier. The following shows the relationships.

Data	**Conceptual Observable**
elementID	Identifier::Identifier

Defined Conceptual Aircraft Entity Attributes	**Selected Conceptual Aircraft Entity Attribute**	**Conceptual Observable**
aircraftID tailNumber	elementID	Identifier::Identifier

The FACE Technical Standard, Edition 3.0 additionally has a requirement that all conceptual entities must have one composition attribute with the observable type, Identifier. Using the *elementID* attribute with observable Identifier in the conceptual entity resolves the two identified entity conditions:

- The required single observable, Identifier type
- No two attributes having the same observable type

Be sure to note that the required observable, Identifier only occurs in the Conceptual Model. The Logical or Platform Models do not have a required measurement or physical data type, respectively.

To conclude the Conceptual Model for the UoC, the modeler created a view that identifies what data from the entity will be included as message data being sent or received by the UoC. The

Conceptual Model view is optional and was created as an example. The Conceptual Model view acquires its message data values by querying the conceptual entity. This approach is different from FACE Technical Standard, Edition 2.1. The query is represented by an SQL-like query language defined for the FACE Technical Standard, Edition 3.0. The view is named *Aircraft_Config_CDM*, labeled appropriately for the level at which it resides. This was a modeler preference only and is not a formalized naming convention. Figure 22 shows the Identifier observable, the Aircraft entity, and the view, *Aircraft_Config_CDM*.

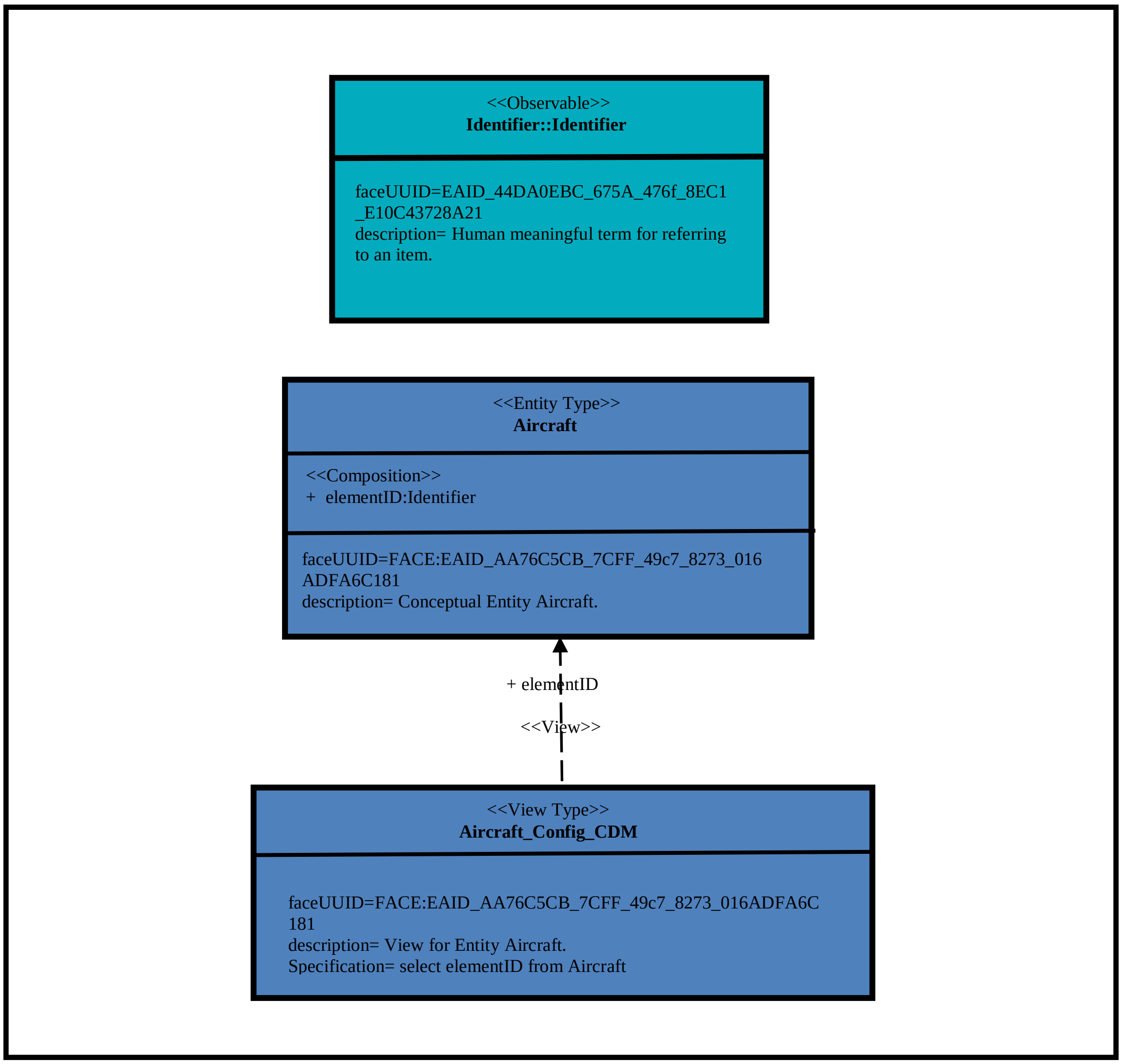

Figure 22: Configuration Conceptual Model

B.3.2 Logical Model

Now that the elements in the Conceptual Model have been established, attention can be turned to the Logical Model. From here, the modeler determines measurements that correspond to the observables in the Conceptual Model. The modeler also creates a logical entity and logical view based on these measurement realizations. Note that like the Conceptual Model view the Logical Model view is optional and is created only as an example. The Logical Model view acquires its message data values using the same SQL-like query language discussed in Section B.3.1 by querying the logical-level entity. The logical-level entity and view are realizations of the entity and view at the conceptual level. Figure 23 shows the relationships between entity and view at the Logical and Conceptual Model and entity-to-entity and view-to-view between the Logical and Conceptual Models.

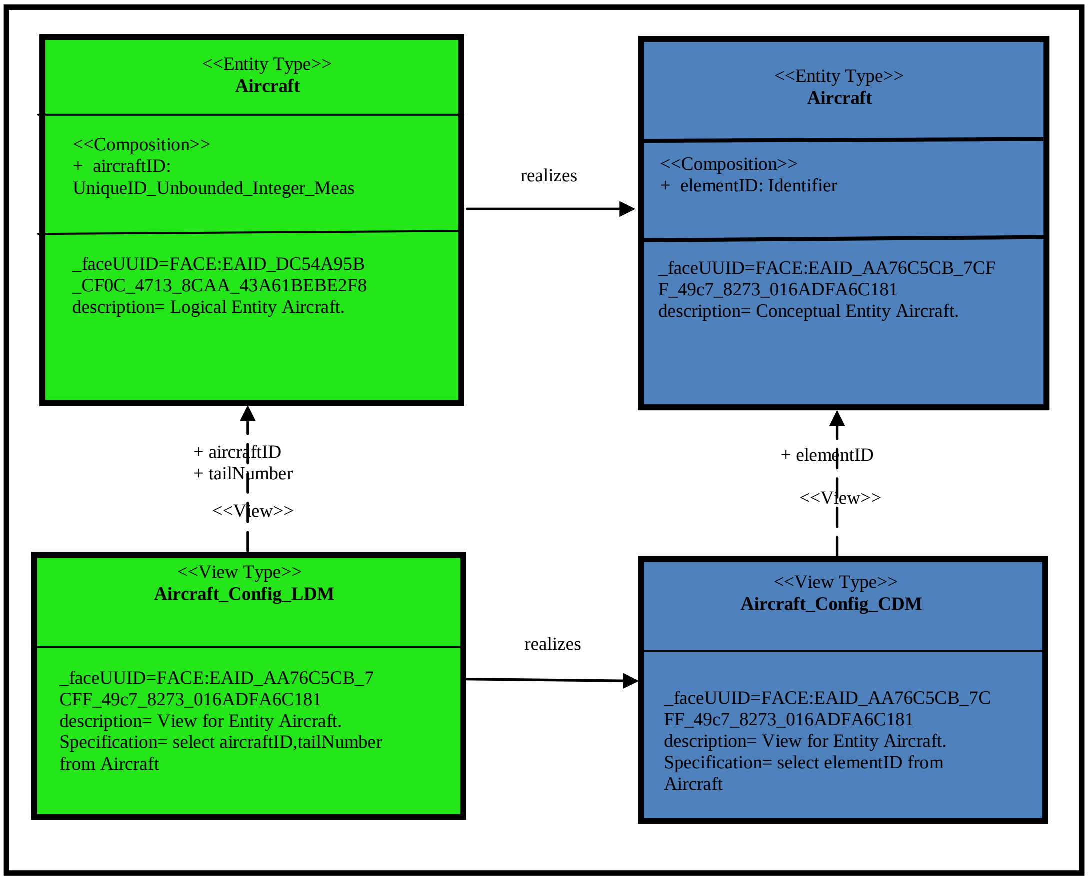

Figure 23: Aircraft Configuration Logical Model (Entities and Views)

In further defining the data variables, the data modeler chose the following measurements in the SDM that realize the observables:

Data	Logical Measurement	Conceptual Observable
aircraftID	UniqueID_Unbounded_Integer::UniqueID_Unbounded_Integer_Meas	Identifier::Identifier
tailNumber	UniqueID_Unbounded_Text::UniqueID_Unbounded_Text_Meas	Identifier::Identifier

Figure 24 demonstrates that the conceptual-level observables are realized by the logical-level measurements selected from the SDM. Remember that in the Conceptual Model the composition was on *elementID* and the observable was type Identifier. The Logical Model is not under the same requirements as the Conceptual Model. In the Logical Model the Aircraft entity can have its composition defined in more detail. That is why in the Aircraft entity composition attributes are now *aircraftID* and *tailNumber*. Notice that the composition attributes have different measurements. In the FACE Technical Standard, Edition 3.0 two different logical-level measurements can "realize" the same observable. The following shows the relationship from Logical Model to Conceptual Model.

Logical Entity	Relationship	Conceptual Entity
Aircraft	realizes	Aircraft

Logical Entity Data	Relationship	Conceptual Entity Data
aircraftID	realizes	elementID
tailNumber	realizes	elementID

Figure 24 demonstrates that a single conceptual-level observable can be realized by multiple logical-level measurements. The measurements must be of a different type. It also shows that each logical-level measurement has a corresponding measurement axis. A measurement axis does not necessarily mean dimension. Refer to the Reference Implementation Guide (RIG) for the FACE Technical Standard, Edition 3.0, Volume 3: Data Architecture (see Referenced Documents) for a more detailed explanation.

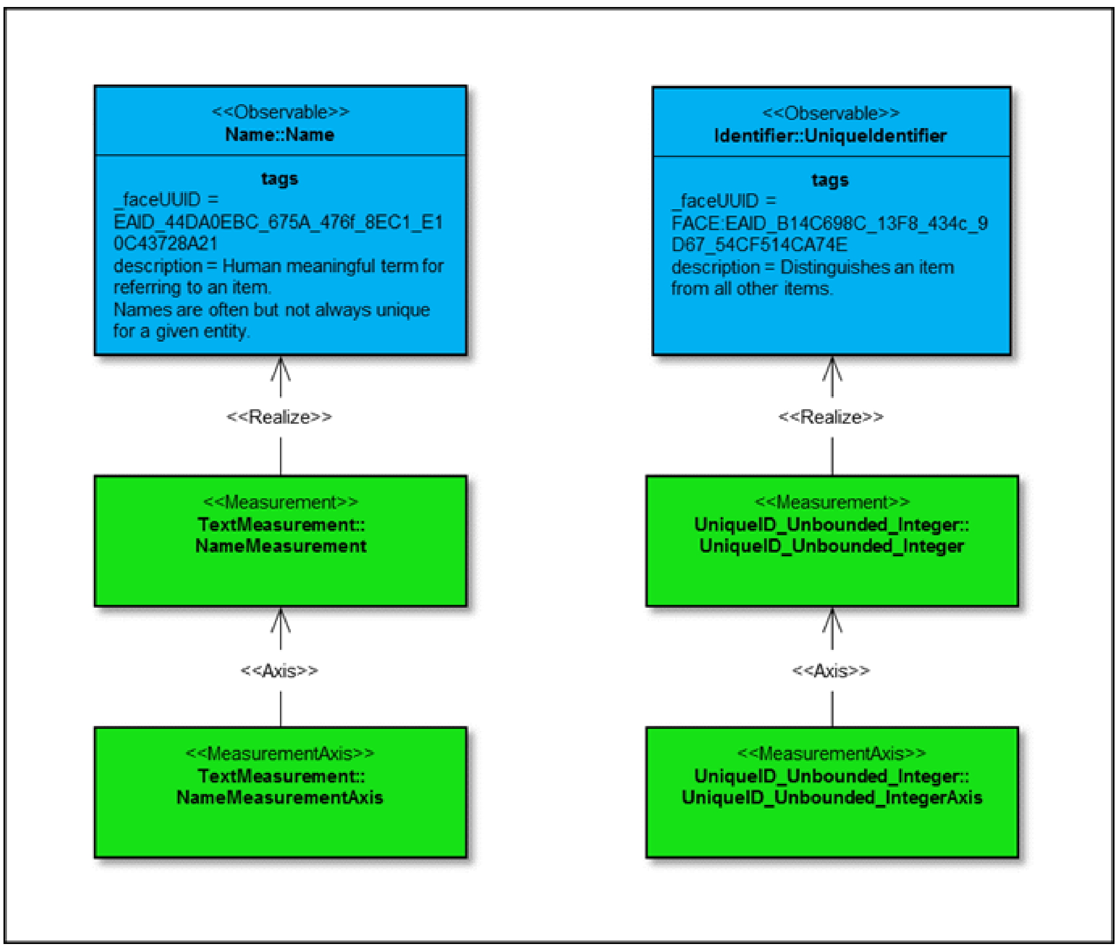

Figure 24: Aircraft Configuration Logical Model (Measurements)

B.3.3 Platform Model

In implementing the Platform Model, the modeler has created new instances of Interface Definition Language (IDL) constructs that will be used to represent the measurements identified in the Logical Model. The IDL constructs provide a type to the Aircraft entity's composition data types. In the platform Aircraft entity the composition data value, *aircraftID*, is of IDL construct type *AircraftID* which is defined as an ULong (unsigned long) and the composition data value, *tailNumber*, is IDL construct type *TailNumber* which is defined as a String. Figure 25 shows the platform Aircraft entity and its composition along with a platform view and a platform template bound to the platform query. It also shows the platform entity realizing the corresponding logical Aircraft entity. This demonstrates the relationship of data modeling elements from platform entities and views to logical entities and views. Take note that the platform view designation changes from a query as it was in the Conceptual and Logical Models to a template bound to a query in the Platform Model. A template bound to a query means that the template gets its data for the message structure from the query to which it is bound. The platform view is a query/template pair.

Figure 25 shows that the platform-level entity and view realizes the logical-level entity and view in a similar manner as the logical-level entity and view realized the conceptual-level entity and view in Figure 23; thus, completing the modeling relationship from the Conceptual Model to the Platform Model. In other words, it completes the evolution from observable to measurement to data type needed to generate code. It also shows that the Platform Model view acquires its message data values by querying the Platform Model entity and formulates the message structure using the platform template.

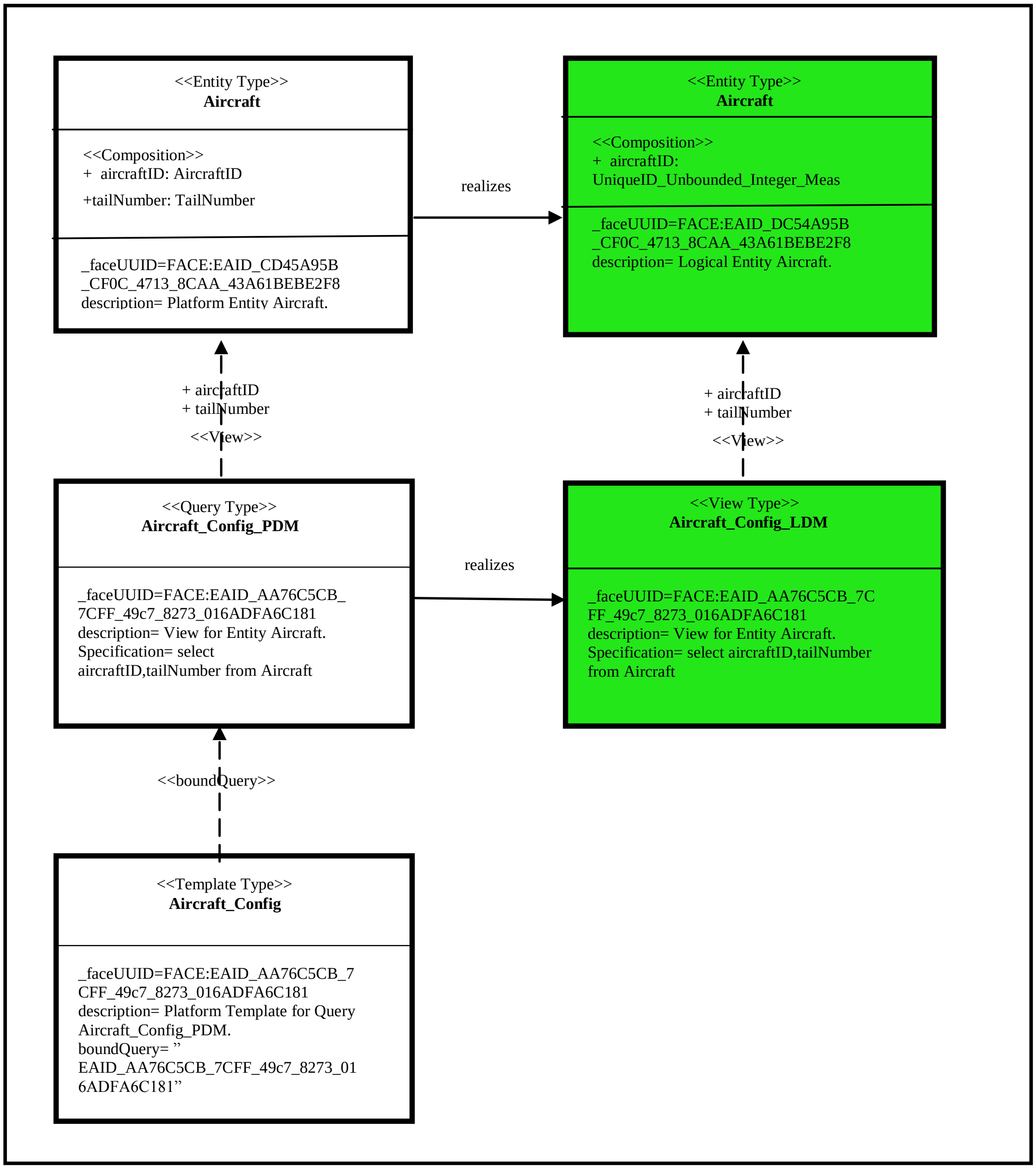

Figure 25: Aircraft Configuration Platform Model (Entities, Views, and Template)

The IDL types defined for the data values are as follows:

Data	**Platform IDL Type**
Aircraft ID	Unsigned Long (ULong)
Tail Number	String

Figure 26 demonstrates how the Platform Model IDL primitives realize the Logical Model measurements in a similar way that the Logical Model measurements realize the Conceptual Model observables in Figure 24.

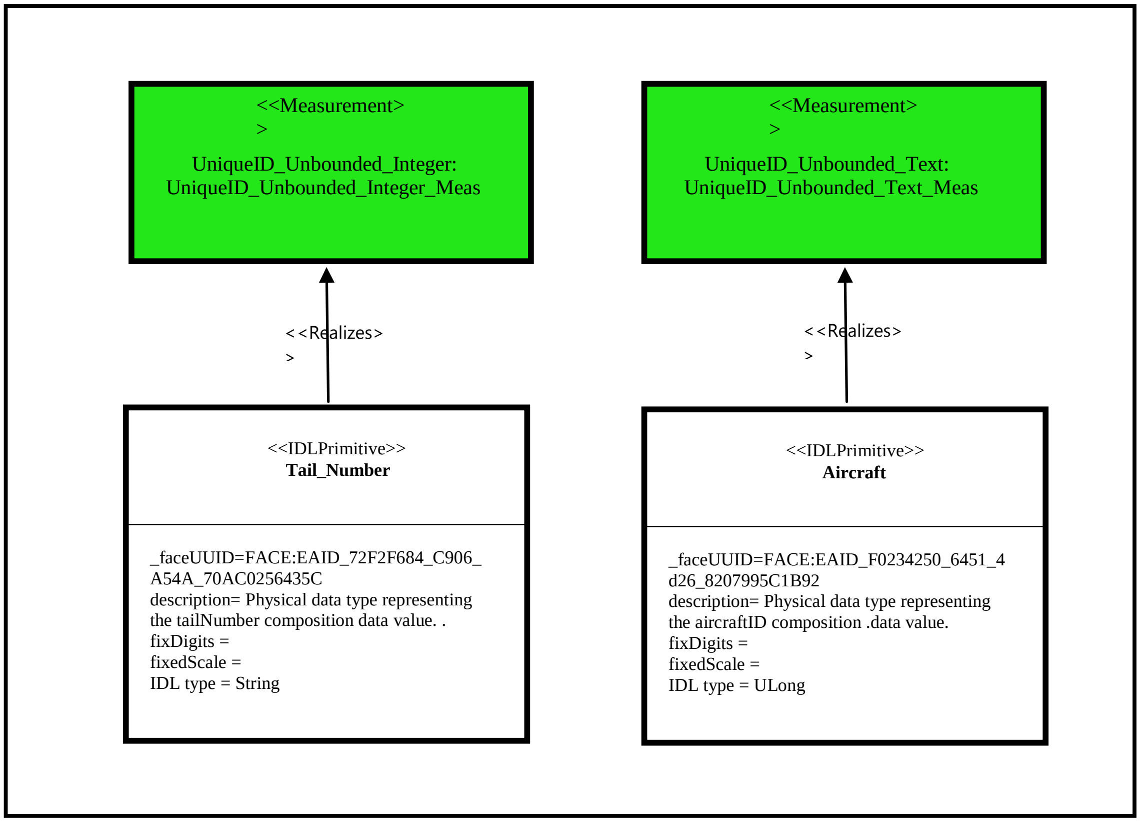

Figure 26: Aircraft Configuration Platform Model (Data Types)

B.3.4 Unit of Portability Model

The Unit of Portability (UoP) Model is where the UoP itself is modeled along with its characteristics (programming language, profile, etc.). This is also where incoming and outgoing messages are modeled using templates to model the message types for each port. In this part of the example, the modeler identified the UoP as *AirConfig*, with an outgoing *MessageType* that has a name of *SingleInstanceMessageConnection* to the *Aircraft_Config* template from the Platform Model. See Figure 27 for the implementation of the UoP Model for the Aircraft Configuration in BALSA.

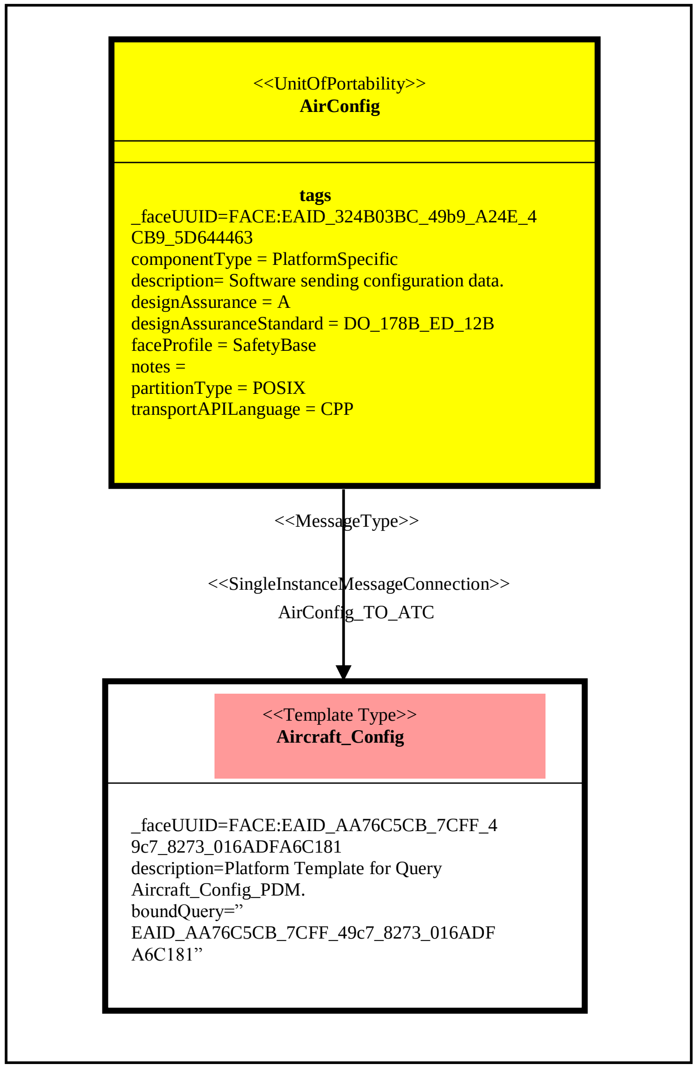

Figure 27: Aircraft Configuration UoP Model

C FACE PR/CR Process Walk-Through

C.1 FACE PR/CR Process Overview

The following scope-limited content is a combination of content from the FACE PR/CR process document and content found on the FACE PR/CR Portal. For a complete discussion of the FACE PR/CR process, refer to the FACE PR/CR process document (see Referenced Documents).

C.2 FACE PR/CR Portal Access

The FACE PR/CR Portal is located at ticketing.facesoftware.org. The login screen is illustrated in Figure 28.

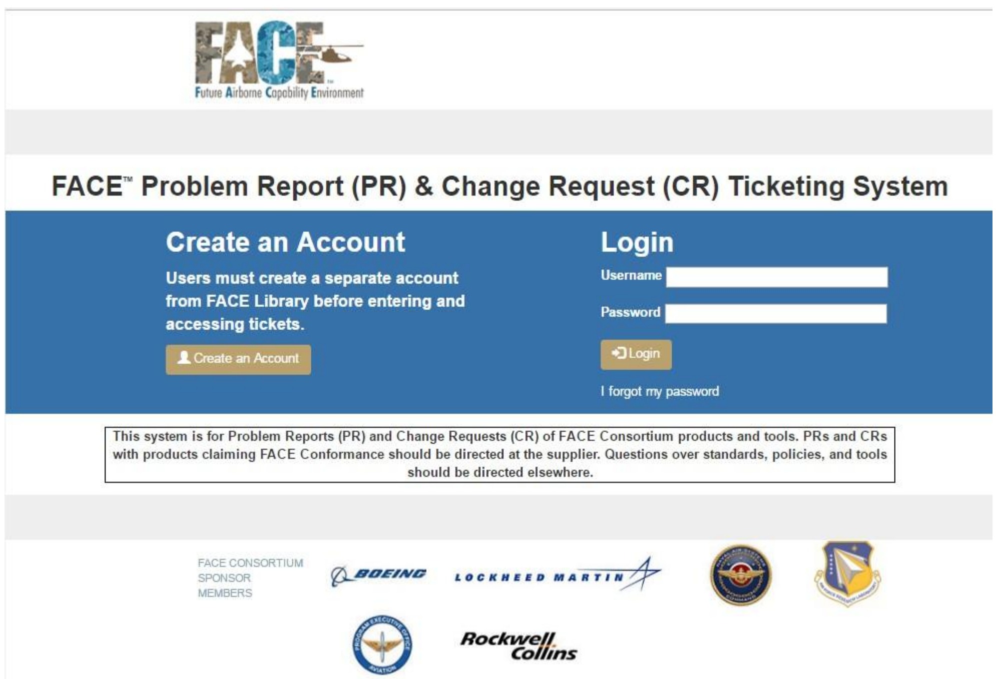

Figure 28: PR/CR Portal Login Screen

New users need to register an account by clicking the Create an Account link.

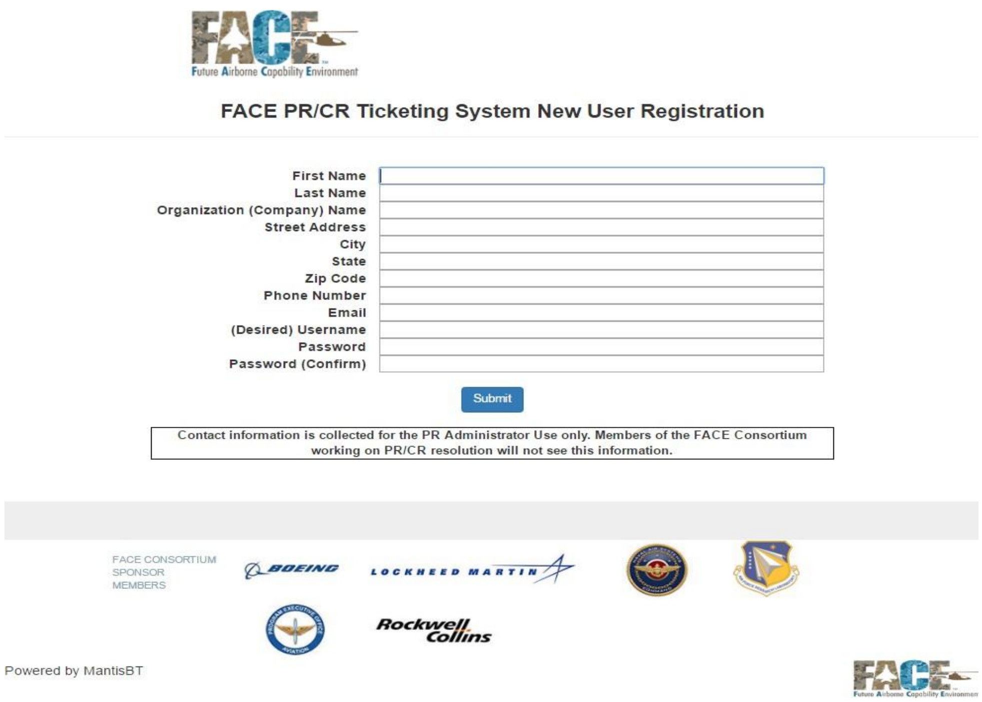

Figure 29: PR/CR Portal Account Creation

After entering organizational information, the user clicks Submit to complete the account creation process.

C.3 Working with Tickets – Home Panel

On successful login, the user lands on the Home Panel. The Home Panel provides the ability to view existing tickets, enter new tickets, and review submitted tickets, as shown in Figure 30.

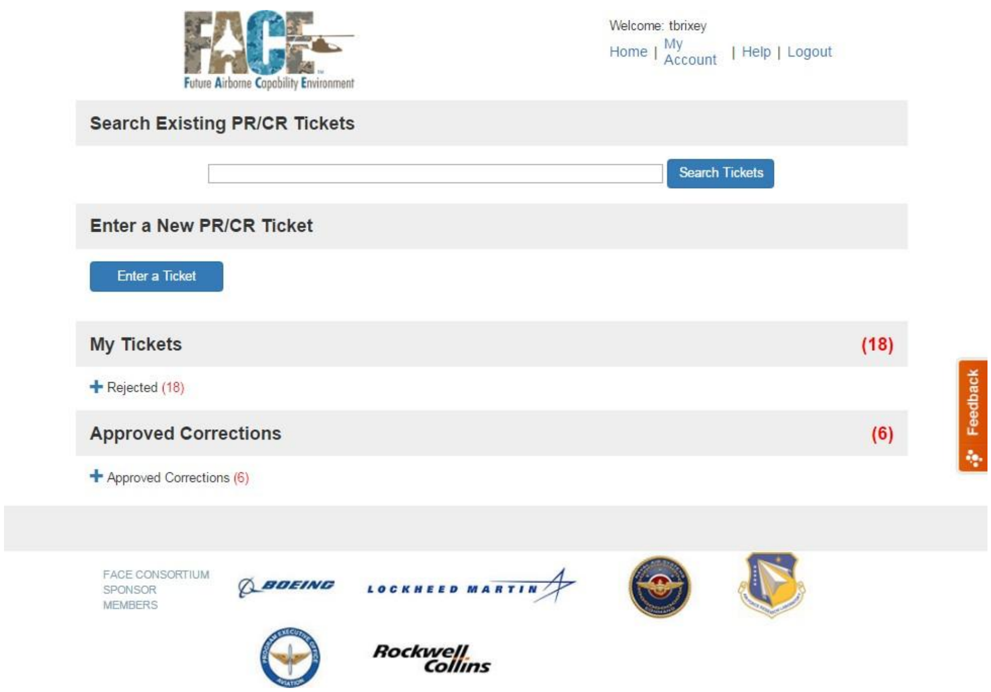

Figure 30: PR/CR Portal Search Dialog

C.3.1 Searching for Existing Tickets

Existing tickets can be found by searching using recognizable words that occur in the ticket fields.

(Hint: To view a list of all tickets click on Search Tickets with an empty keyword field.)

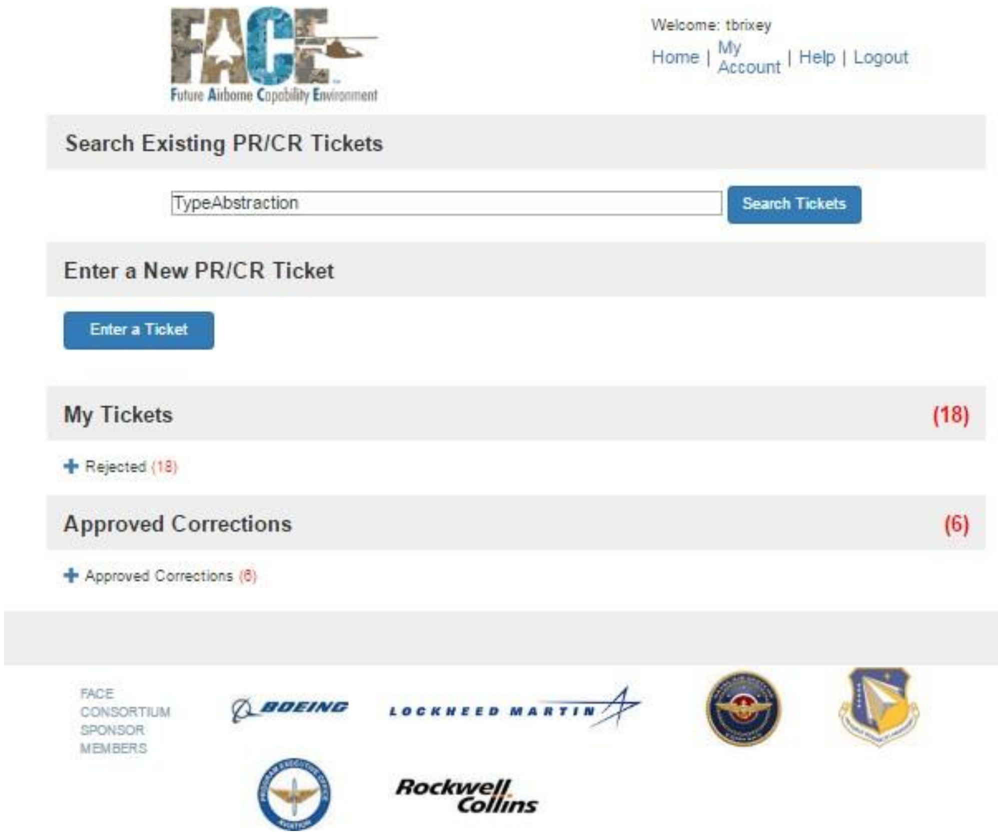

Figure 31: PR/CR Portal Search Dialog – Existing Tickets

The resulting set can be filtered by Ticket State, Criticality, Priority, and Product Status, as shown in Figure 32.

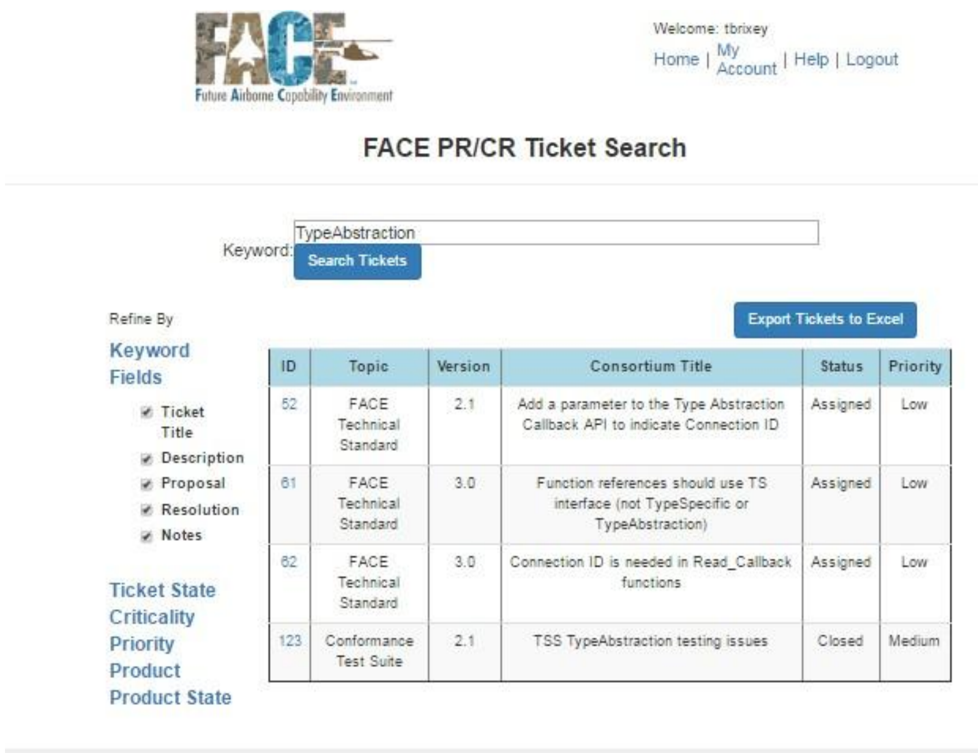

Figure 32: PR/CR Portal Search Dialog – Existing Tickets with Filters

C.3.2 Submitting a New Ticket

The user (submitter) enters a new ticket by clicking the Enter a Ticket button on the Home Panel, as shown in Figure 33.

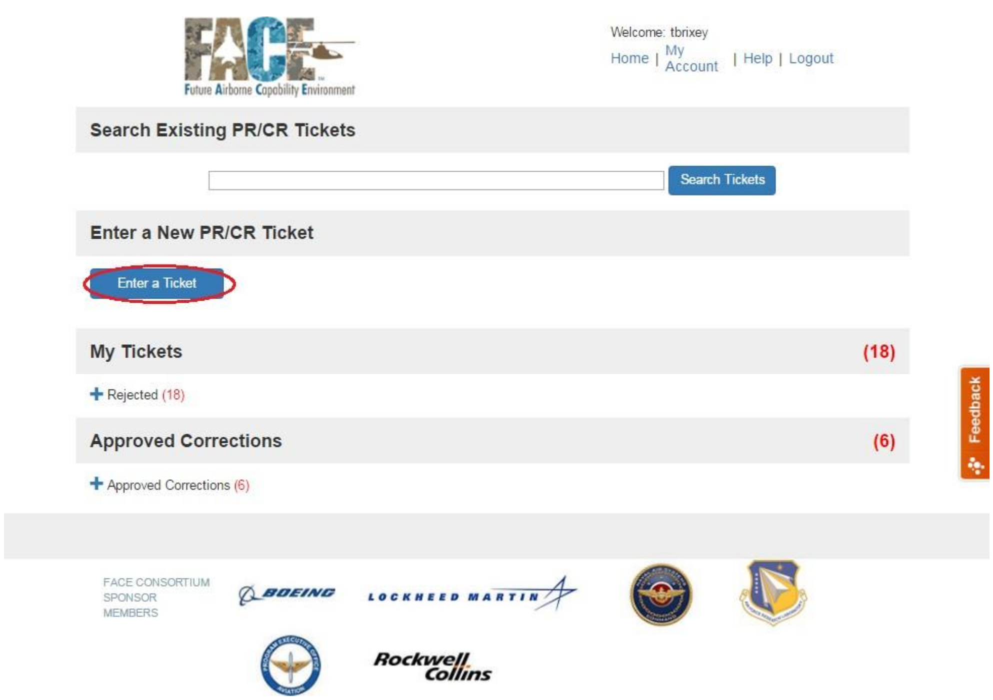

Figure 33: PR/CR Portal Search Dialog – New Tickets

Note: Ticket information entered into the FACE PR/CR system is visible to the general public. The FACE Consortium and the FACE PR/CR Portal Manager accept no liability over the release of information through this system.

Submitter input should include all applicable information that can aid the FACE Triage Group in determining courses of action regarding the ticket.

Note: Submitters of tickets remain anonymous throughout the process.

Data to be entered includes, but is not limited to, the following:

- FACE Consortium Product – the primary FACE Consortium controlled item that has the issue
- Product Version – the version/edition of the FACE Consortium product/document that exhibits the issue
- Comment Type – classifies the issue as either editorial (clarification, punctuation, grammar, etc.) or technical (incorrect, inconsistent, conflicting, etc.)
- Submitter Priority – indicates Submitter's Priority as Low (should be fixed, but I can live with it), Medium (must be fixed, but don't drop everything to fix it now), or High (I need this fix immediately)
- Location – where in the product the issue is found (section and paragraph, function call in tool)
- Submitter Title – a short name containing a recognizable/searchable aspect of the issue
- Submitter Proposed Solution – submitter's preferred method for correcting the issue
- Certification Need – answers the question: does this issue prevent the successful completion of the FACE Conformance process for software currently under development to a published edition of the FACE Technical Standard?
- Attachments – any files that clarify the issue

C.3.3 Ticket Status

Ticket Status is captured in the Ticket Details panel. Details of actions taken on the ticket are found on the Submitter Input, Consortium Response, Impacted Products, and Notes sub-tabs found on the panel. Details for the ticket found in the section Searching for Existing Tickets are provided as examples in Figure 34 through Figure 37.

Ticket #123
TSS TypeAbstraction testing issues

Current Status: Closed
This ticket is in the **CLOSED** state. No further action will be taken on this ticket.

Submitter Input | Consortium Response | Impacted Products | Notes

Submitter Input	
ID	123
Submitter Title	TSS TypeAbstraction testing issues
FACE Consortium Product	Conformance Test Suite
Product Version	2.1
Report Type	Technical
Submitter Priority	Low
Submitter Certification Need	No
Location	
Submitter Description	The TypeAbstraction tests cause a false failure when evaluating the register_read_callback interfaces.
Submitter Proposed Resolution	Correct the TypeAbstraction test to correctly evaluate the register_read_callback interfaces when required by the CTS configuration.
Submitter Attachments	

Figure 34: PR/CR Portal Ticket Status Dialog (Submitter Input)

Ticket #123
TSS TypeAbstraction testing issues

Current Status: Closed
This ticket is in the **CLOSED** state. No further action will be taken on this ticket.

Submitter Input | Consortium Response | Impacted Products | Notes

Consortium Response	
Consortium PR/CR Title	TSS TypeAbstraction testing issues
Consortium Description	The TypeAbstraction tests cause a false failure when evaluating the register_read_callback interfaces.
Workflow State	Closed
Consortium PR/CR Criticality	Essential
Consortium PR/CR Priority	Medium
Proposal Subcommittee	Conformance Tools CCB
Local Impact Only	Yes
Proposed Correction	A bug in the CTS triggered false failures when testing the TSS TypeAbstraction interface.
Proposed Correction Approved	Yes
Proposed Correction Approved Date	11:11am 13 Apr 2016

Figure 35: PR/CR Portal Ticket Status Dialog (Consortium Response)

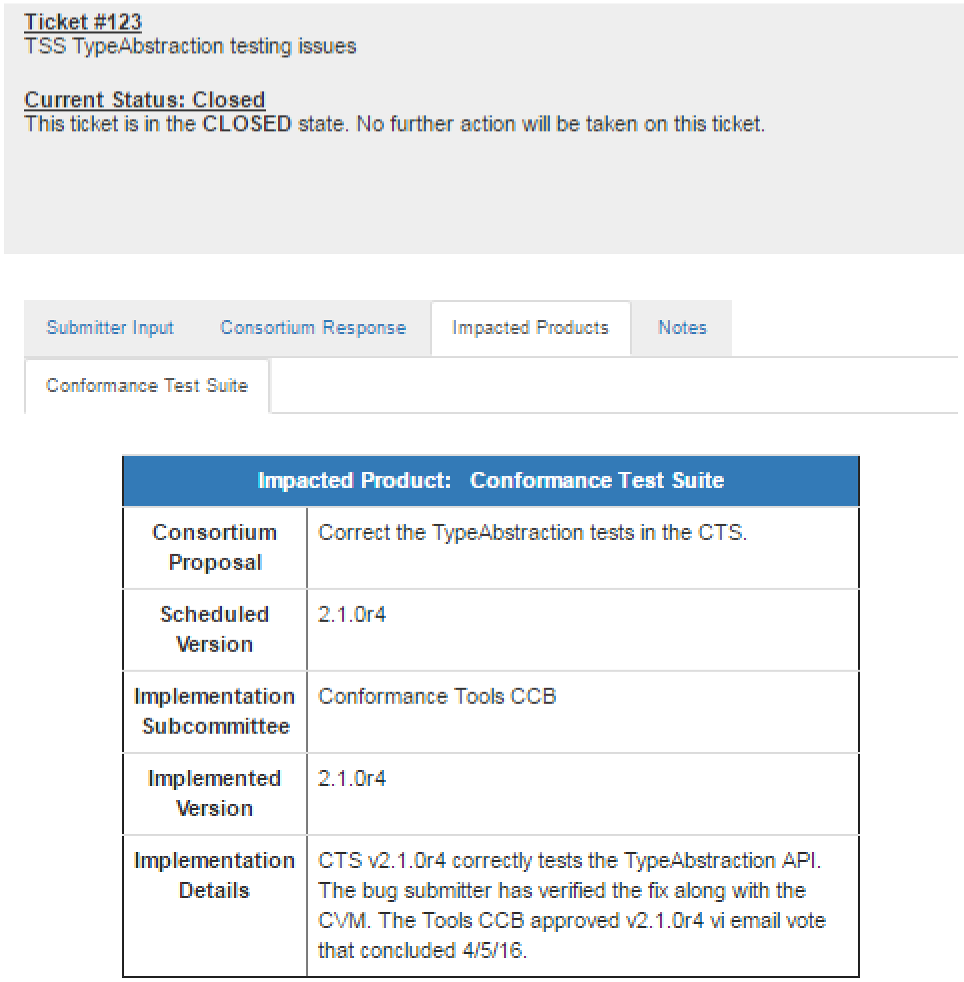

Figure 36: PR/CR Portal Ticket Status Dialog (Impacted Products)

Ticket #123
TSS TypeAbstraction testing issues

Current Status: Closed
This ticket is in the **CLOSED** state. No further action will be taken on this ticket.

Submitter Input | Consortium Response | Impacted Products | Notes

Notes		
01:10pm 23 Mar 2016	Investigate	23-Mar-2016 - Triaged to Tools CCB to investigate.
11:09am 13 Apr 2016	Proposed	A bug in the CTS triggered false failures when testing the TSS TypeAbstraction interface. CCB email vote held 3/23/16
11:11am 13 Apr 2016	Assigned	CCB approved by email vote CTS / TWG on 23 March 2016
02:02pm 21 Jun 2016	Implemented	CTS was tested and released
09:54am 04 Aug 2016	Verified	FACE CCB approved. 8/4/2016
07:38pm 06 Sep 2016	Closed	Implemented as proposed in 2.1.0r4

Figure 37: PR/CR Portal Ticket Status Dialog (Notes)

Acronyms & Abbreviations

ADS-B	Automatic Dependant Surveillance – Broadcast
API	Application Programming Interface
BALSA	Basic Avionics Lightweight Source Archetype
CA	Certification Authority
CCB	Configuration Control Board
CR	Change Request
CTS	Conformance Test Suite
CVM	Conformance Verification Matrix
DoD	Department of Defense
DSDM	Domain-Specific Data Model
EGI	Embedded GPS/Inertial Navigation System
FACE	Future Airborne Capability Environment
FIA	FACE Implementation Architecture
GCC	GNC Compiler Collection
GSG	Getting Started Guide
HMFM	Health Monitoring and Fault Management
ICD	Interface Control Document
IDD	Interface Design Description
IDE	Integrated Development Environment
IDL	Interface Definition Language
IOSS	I/O Services Segment
IRS	Interface Requirements Specification
MOF	Meta Object Facility
MUG	Matrix User's Guide
OCL	Object Constraint Language

OSS	Operating System Segment
PCS	Portable Components Segment
PM	Program Manager
PR	Problem Report
PSSS	Platform-Specific Services Segment
RIG	Reference Implementation Guide
RTOS	Real-Time Operating System
SAD	Software Architecture Description
SDD	Software Design Description
SDM	Shared Data Model
SPS	Software Product Specification
SQL	Structured Query Language
SRS	Software Requirements Specification
STP	Software Test Plan
STR	Software Test Report
SUM	Software User Manual
TS	Transport Services
TSS	Transport Services Segment
UDP	User Datagram Protocol
UoC	Unit of Conformance
UoP	Unit of Portability
USM	UoP Supplied Model
VA	Verification Authority
XMI	XML Metadata Interchange
XML	Extensible Markup Language

Index