Ada Language Run-Times and the FACE™ Technical Standard: Achieving Application Portability and Reliability

Army FACEτ TIM Paper by:

Benjamin M. Brosgol, Patrick Rogers and Dudrey Smith
AdaCore

August, 2018

Executive Summary

A programming language run-time library for a Future Airborne Capability Environment (FACE™) Operating System Segment (OSS) profile needs to supply the functionality specified in the corresponding Capability Set and do so in a manner that is consistent with the Design Assurance Level (DAL) targeted by that profile. AdaCore's *Cert* and *Ravenscar-Cert* libraries have been designed to meet these requirements, for the Ada Safety Base/Security and Ada Safety Extended Capability Sets. They implement the required functionality while being straightforward enough for inclusion in airborne systems requiring certification against the highest DALs of DO-178B/C [1, 2]. This paper summarizes these run-time libraries and shows how they meet the portability requirements of the FACE Capability Sets together with the assurance requirements of high-DAL systems. The paper is based on Edition 3.0 of the FACE Technical Standard [3] and is oriented towards software developers or project managers; no previous knowledge of Ada is required.

Introduction

A programming language's run-time library comprises the routines that implement the language's dynamic semantics, most notably memory management (allocation and deallocation), exception handling, and concurrency control. In the FACE Technical Standard a run-time library is considered as part of the Operating System Segment (OSS), but it differs from other OSS elements in how its routines may be invoked. Instead of only being able to obtain the needed functionality through explicit calls on an Application Program Interface (API), a software developer can use standard programming language syntax (a subset of language features known as a *Capability Set*). Using programming language syntax has the advantage of source code portability, since different compilers may implement the standard semantics through different (and vendor-specific) APIs that in turn interface with the underlying RTOS and hardware platform.

The content of a Capability Set depends on the OSS profile (General Purpose, Safety Extended/Base, Security) and associated DAL. In particular, the extent of a Capability Set will vary with the degree of assurance required. In addition, the more restrictive profiles impose more stringent implementation requirements that are outside the scope of the FACE Technical Standard. For example, requirements for reliability, safety and/or security may entail certification under domain-specific standards such as DO-178B/C for airborne software. Successful certification under such standards requires a degree of implementation simplicity that directly affects the content of Capability Sets. A programming language implementation thus has the challenge of providing run-time libraries that implement the specified functionality across all the Capability Sets while satisfying any such auxiliary requirements. This paper shows how several Ada run-time libraries provided with AdaCore's GNAT Pro toolset meet this challenge.

Ada and the OSS Profiles

The FACE Technical Standard 3.0 [3] defines several OSS Profiles:

- Security [safety/security assurance and determinism; time/space partitioning required]
- Safety [safety assurance and determinism; time/space partitioning required]
  - Base
  - Extended
- General-Purpose [assurance as required, determinism not guaranteed; time partitioning optional, space partitioning required]

The Standard defines programming language Capability Sets for C, C++, Ada and Java with respect to these profiles, although the correspondence between the profiles and the Capability Sets is not exact (in some cases a single Capability Set relates to several profiles). The Capability Sets are intended to promote component portability and include restrictions on both run-time and non-run-time functionality.

Several Capability Sets are defined for Ada:

- Ada 95 Safety Base and Security
- Ada 95 Safety Extended
- Ada 95 General-Purpose
- Ada 2012 General-Purpose

The FACE Technical Standard's provision of separate Capability Sets for Ada 95 [4] and Ada 2012 [5] is based on balancing the tradeoffs between using an older version of the language that is still in common practice, and using a newer version of the language whose features help reduce development and verification costs. As Ada 2012 usage spreads in avionics projects and elsewhere, the inclusion of Ada 2012 features in Capability Sets in future versions of the FACE Technical Standard is expected to increase. Ada 2012's contract-based programming features (subprogram pre- and post-conditions) are particularly useful, since they allow the programmer to embed low-level requirements in the source code, where they can be checked either statically (with appropriate analysis tools) or at run time. The high-degree of upward compatibility across Ada revisions facilitates upgrading to Ada 2012 from earlier versions of the language standard.

The various FACE Capability Sets principally constrain Ada's run-time functionality -- memory management (dynamic allocation and deallocation), exception handling, concurrency/threading -- and also impose a variety of restrictions on other features as well as the language's predefined environment (Input-Output, etc.).

An Ada run-time library for an OSS at a given profile must provide the functionality required by the associated Capability Set, while being amenable to the analysis for certification against external safety or security standards required for that profile. The run-time libraries supplied by GNAT Pro meet these objectives, as they are closely matched to the definition and intended usage of the various Capability Sets. The libraries implementing the Safety Capability Sets (Safety Base and Safety Extended) have met the objectives at the highest DAL (Level A) in DO-178B/C in systems fielded by major aerospace companies. Although the profiles and their associated Capability Sets do not guarantee meeting such certification requirements, they facilitate library implementations that can meet these requirements and thus can encourage what is accepted as common practice.

The table in the Summary section later in this paper shows the relationships among the OSS profiles, the Ada Capability Sets, and the AdaCore run-time libraries that support the Capability Sets.

Ada 95 Safety Base and Security Capability Sets

These capability sets define a variety of restrictions on Ada's run-time features:

- Dynamic allocation (and thus deallocation) is prohibited.
- Exception handling is limited to handling predefined exceptions using a single default handler.
- Concurrent programming is limited to the features allowed by the Ravenscar tasking subset [6], but a FACE Unit of Conformance can also use the tasking/threading API supplied by the OS environment (i.e., POSIX [7] or ARINC 653 [8] standards).

The Safety Base and Security Capability Sets exclude much of the predefined environment (for example omitting wide characters/strings, input/output, and string handling) and most of the Ada standard's specialized needs annexes (features that support systems programming, real-time systems, and other

application areas). However, these Capability Sets include some support for interfacing with C, interrupt handling, and real-time task scheduling.

The Cert Ada library

AdaCore's Cert library ([9], Section 4.3) implements the features specified in the Safety Base and Security Capability Sets:

- Memory management

The Cert library includes an implementation of a secondary stack, for functions returning values whose size is not known until the point of return. For example:

```
function Pad (S : String; C : Character; N : Natural) return String is
begin
   return S & (1..N => C);
end Pad;
-- Pad("Hello", '*', 3) returns "Hello***"
```

Since the Safety Base and Security Capability Sets do not allow dynamic allocation, this style of returning "unconstrained" values, both for arrays and discriminated records, is likely to be useful in practice. The implementation of a secondary stack thus provides an important functionality.

- Exception handling

The Cert library supports the simple semantics required by the Safety Base and Security Capability Sets. The user can supply a "last chance handler" to respond to unhandled exceptions:

```
procedure Last_Chance_Handler (Except : Ada.Exceptions.Exception_Occurrence);
```

This procedure can interrogate the `Except` parameter to produce the relevant diagnostic output.

- Concurrent programming

The Cert library on Wind River VxWorks 653 is based on an implicitly thread-safe run-time library. Although there are no Ada tasks, the programmer can use ARINC 653 APEX processes to implement the necessary concurrency and real-time logic.

The Cert library has been implemented for Wind River's FACE conformant VxWorks 653 2.5 RTOS (conforming to the FACE Technical Standard Edition 2.0, Safety Base profile) and also for VxWorks 653 3.x as well as Lynx Software Technologies LynxOS-178 2.2.4.

Ada 95 Safety Extended Capability Set

This Capability Set is largely the same as the Safety Base and Security Capability Set, with additional features allowed in several areas including the following:

- Dynamic allocation is permitted but only at system startup (during the elaboration phase of program execution). Deallocation is prohibited.
- Exception handling is allowed, but usage of the exception querying functions from the package

Ada.Exceptions is limited to those whose effect is not implementation defined.

The Ravenscar-Cert Ada library

AdaCore's Ravenscar-Cert library ([9], Section 4.4) augments the Cert library with support for the Ravenscar tasking subset. It implements the functionality required by the Ada Safety Extended Capability Set:

- Memory management
  The standard Ada allocator (the **`new`** construct) maps to a simple function in the implementation package `System.Memory` (basically a wrapper for the underlying `malloc` routine).
- Exception handling
  The Ravenscar-Cert library supports exception propagation and handling, along with a "last chance" handler as described above.
- Concurrency
  As implied by its name, the Ravenscar-Cert library supports the Ada tasking model as constrained by the Ravenscar subset rules. For example:
    - Task object/type declarations must be at library level, and task entries are prohibited.
    - Protected object/type declarations must be at library level, with at most one entry each.
    - Absolute delay statements are permitted, but not relative delay statements.
    - No select, requeue, or abort statements are permitted.
    - At most one task at a time is allowed to be queued on a given protected entry.

  These restrictions allow a deterministic and extremely efficient implementation in both time and space.

The Ravenscar tasking subset (known in the literature as the *Ravenscar Profile*) and thus the Ravenscar-Cert library were designed to strike a balance between simplicity (and thus appropriateness, for example, when safety certification is required) and expressiveness (support for the Safety Extended Capability Set). The library has been included in systems certified at DO-178B Level A.

The Ravenscar-Cert library has been implemented for Wind River's FACE conformant VxWorks 653 2.5 RTOS (conforming to the FACE Technical Standard Edition 2.0 Safety Base profile) and also for VxWorks 653 3.x as well as Lynx Software Technologies LynxOS-178 2.2.4.

Ada General Purpose Capability Sets

For general purpose applications, i.e., where assurance is needed but where a failure will not reduce safety or security unacceptably, the main consideration from a FACE Technical Standard perspective is code portability, and thus the general purpose Capability Sets exclude features that are implementation specific.

AdaCore supplies a full run-time implementation for the Ada 95 and Ada 2012 General Purpose Capability Sets, along with specialized run-times that likewise support these sets.

Summary

The following table lists major features either allowed or prohibited in the various Ada Capability Sets of the FACE 3.0 Technical Standard, and their relationship to the OSS profiles and AdaCore run-time libraries. A complete description of the Capability Sets appears in [3], Section 3.2.3.4. A complete description of the run-time libraries appears in [9], Section 4.

OSS Profile	Ada 95 Capability Sets	Ada 2012 Capability Sets	Run-Time Library
Safety Base / Security	• Interrupt support • Predefined exceptions with single default handler • Ravenscar subset • No dynamic allocation • Minimal support for predefined environment		**Cert library** • Safety Base / Security Capability Sets (sequential features) • Secondary stack • ARINC 653 APEX processes on VxWorks 653
Safety Extended	• Safety Base / Security features • Dynamic allocation at startup • Implementation-independent exception handling functions • No deallocation		**Ravenscar-Cert library** • Cert library features • Dynamic allocation function • Exception propagation, “last chance” handler • Ravenscar tasking subset
General Purpose	• Full Ada 95 with some restrictions • No Asynchronous Transfer of Control • No implementation-dependent features	• Ada 95 General Purpose Capability Set plus some Ada 2012 features • No implementation-dependent features • No pre- or postconditions	**Full Ada run-time library**

In practice, both the Cert and Ravenscar-Cert libraries can be used by an application that adheres to either the Safety Base / Security or Safety Extended Capability Sets, since both libraries supply the functionality required by these Capability Sets.

Enforcing the Capability Set Restrictions

A Capability Set for an OSS profile specifies features that are allowed, and it also specifies features that are prohibited. A run-time library that implements a Capability Set has to support at least the allowed features. It may provide more, but the code in a FACE Unit of Conformance needs to adhere to the restrictions that the Capability Set and its associated OSS profile impose. The Ada language and AdaCore tools offer a number of techniques that can help enforce these restrictions.

- **pragma** Restrictions

This standard Ada pragma identifies features that are not to be used; a violation causes the program to be rejected. For example, `pragma Restrictions(No_Allocators)` could be used in an application targeted to the Safety Base and Security Capability Sets.

- **pragma** Profile (Ravenscar)

This standard Ada pragma (as of Ada 2005) corresponds to a collection of Restrictions pragmas that define the Ravenscar rules. (This pragma is not required in the AdaCore toolset, since the program builder will reject a build if the any Ravenscar restrictions are violated.)

- GNATcheck tool

AdaCore's GNATcheck is an extensible rule-based static analysis tool that can enforce a coding standard. It may be useful when pragma Restrictions cannot capture a particular coding limitation.

Conclusions

The variety of Capability Sets for Ada presents a challenge to the run-time library provider: implement the needed functionality, but, in particular for the Safety Capability Sets, ensure that the run-times are simple enough to be included in safety- or security-certified applications. AdaCore's Cert and Ravenscar-Cert Profiles meet this challenge, helping to achieve the portability goal underlying the FACE initiative while providing the analyzability, predictability and efficiency required in high-assurance real-time avionics systems.

References

[1] RTCA DO-178B/EUROCAE ED-12B, *Software Considerations in Airborne Systems and Equipment Certification,* December 1992

[2] RTCA DO-178C/EUROCAE ED-12C, *Software Considerations in Airborne Systems and Equipment Certification,* December 2011

[3] *FACE™ Technical Standard, Edition 3.0*.
https://publications.opengroup.org/c17c/

[4] *Ada Reference Manual ISO/IEC 8652:1995(E) with Technical Corrigendum 1, Language and Standard Libraries.*
http://www.adaic.org/resources/add_content/standards/95lrm/ARM_HTML/RM-TTL.html

[5] *Ada Reference Manual ISO/IEC 8652:2012(E) with Technical Corrigendum 1, Language and Standard Libraries*
http://www.ada-auth.org/standards/rm12_w_tc1/html/RM-TTL.html

[6] B. Dobbing and A. Burns, *The Ravenscar Tasking Profile for High Integrity Real-Time Programs*.
http://www.sigada.org/conf/sigada2001/private/SIGAda2001-CDROM/SIGAda1998-Proceedings/dobbing.pdf

[7] *POSIX.1-2017*.
http://pubs.opengroup.org/onlinepubs/9699919799/
https://standards.ieee.org/findstds/standard/1003.1-2017.html

[8] *ARINC 653; Avionics Application Software Standard Interface, Parts 0, 1, 2, 3A, 4 and 5*.
https://www.aviation-ia.com/product-categories/600-series

[9] *GNAT User's Guide Supplement*
http://docs.adacore.com/live/wave/gnathie_ug/html/gnat_hie/gnathie_ug.html

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

About the Author(s)

Dr. Benjamin Brosgol is a senior member of the technical staff at AdaCore. He has been involved with programming language design and implementation throughout his career, concentrating on languages and technologies for high-integrity systems with a focus on Ada and safety certification (DO-178B/C). Dr. Brosgol is an active member of the FACE Technical Work Group, and in particular he has been involved with the development of the IDL-to-Ada mapping.

Dr. Patrick Rogers has been a computing professional since 1975, primarily working on microprocessor-based real-time applications. He began working with Ada in 1980 and was director of the Ada9X Laboratory for the U.S. Air Force's Joint Advanced Strike Technology Program, and Principle Investigator in distributed systems and fault tolerance research projects for the U.S. Air Force and Army. As a member of the technical staff at AdaCore, he is a developer of the bare-board products for Ada and provides support and training focused on embedded real-time applications.

Dr. Dudrey Smith is a senior embedded system development consultant at AdaCore. He has been involved with military and commercial embedded system/software development and certification for more than 40 years, with major leadership roles at companies including Lear Siegler, Smiths Aerospace, and General Electric Aviation Systems. Dr. Smith is an active member of the FACE Operating Systems and Conformance Subcommittees.

About The Open Group FACE™ Consortium

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

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

About The Open Group

The Open Group is a global consortium that enables the achievement of business objectives through technology standards. Our diverse membership of more than 600 organizations includes customers, systems and solutions suppliers, tools vendors, integrators, academics, and consultants across multiple industries.

The Open Group aims to:

- Capture, understand, and address current and emerging requirements, and establish policies and share best practices
- Facilitate interoperability, develop consensus, and evolve and integrate specifications and open source technologies
- Offer a comprehensive set of services to enhance the operational efficiency of consortia
- Operate the industry's premier certification service

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