# *Secure Coding for Real-Time Embedded Systems:*

## *Cert Run-Time Profile for Ada*

Mable Benjamin
Georgia Tech Research Institute, Atlanta, GA
Mable.Benjamin@gtri.gatech.edu

***Abstract*— Safety and security in software coding is becoming increasingly important in mission-critical systems due to various emerging threats and to avoid potential disastrous consequences. These issues can be significantly reduced by software designs and implementations that adequately protect systems. This abstract describes coding methodologies in the Ada 95/2005 programming language for complex embedded software running on Real-Time Operating Systems (RTOS) for military applications.**

## I. INTRODUCTION

Adopting a pre-defined coding standard is one of the top 10 recommended secure coding practices [1]. Pre-defined coding standards are built into certain compilers to support mission-critical software meeting safety certifications such as DO-178B [2]. A run-time profile (RTP) is a compiler-enforced Ada language [3] subset with a corresponding (minimal or none) run-time library [4]. A run-time library is a set of low-level routines used by a compiler to invoke behaviors of a runtime environment, by inserting calls into the compiled executable binary [5].

## II. RUN-TIME PROFILES

Depending on the target, the GNAT Pro (AdaCore) compiler [6] provides multiple RTPs that streamlines implementation of security and safety practices, namely, 1) the Zero Footprint Profile (ZFP), an Ada subset requires no run-time support, 2) the Cert Profile, supports features in the ZFP with additional thread-safe restrictions, 3) the Ravenscar Profile, supports the above profiles with additional tasking restrictions, and 4) the Full RTP, supports the complete Ada language [4]. An RTP is a language subset enforced by the compiler with a corresponding set of approved libraries.

The Cert profile is based on an implicitly threaded run-time library. Some of the key areas that this profile addresses are exception handling, dynamic memory allocation, thread management, object deallocation, traceability from source code to object code, and optimization issues [4].

## III. CASE STUDY

The Cert RTP was implemented on an aircraft control system for the Department of Defense (DoD) to support safety and security requirements based on the nature of the software. The RTOS used for this system is VxWorks [7] ARINC 653 [8] 2.4 provided by Wind River [2]. ARINC 653 is a software specification used for space and time partitioning of multiple applications running on different partitions on the same hardware in safety-critical avionics RTOS [8]. In order to decouple the RTOS platform from the application software, ARINC 653 defines an API called APplication EXecutive (APEX) .

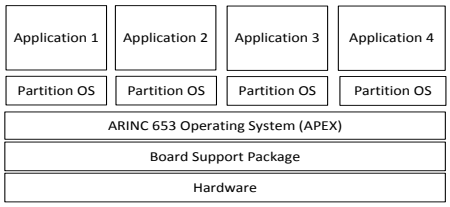


Fig. 1. ARINC 653 RTOS Architecure

To implement Cert for the application software, the RTP was changed from "full" to "cert" and the build mode was changed to Non-Debug. This exposed a set of compiler errors generated due to the Cert profile's unsupported features and libraries. The VxWorks [7] semaphore interfaces used in the code base was not supported by Cert. These interfaces were replaced with the APEX interface [8] that provides Cert compliant OS related utilities. Other key changes included removing all debug related information, developing user-defined Ada packages that are not supported in the Cert RTP (e.g. random number generators), and implementing mechanisms to prevent dead-locks.

### *A. Challenges*

There were several challenges in the process of converting existing code that compiled in the Full RTP to the Cert RTP. First, multiple vendors were responsible for different software applications on the ARINC 653 architecture, thus introducing some challenges to subsystem-level integration. However, the ARINC 653 architecture allowed for different partitions to use different libraries and RTPs which facilitated mitigating the issue. Second, switching from VxWorks OS [7] level calls to APEX interfaces [8] revealed some differences in the behavior of semaphores and threads that were identified during testing and were addressed accordingly. Finally, implementing user-defined libraries to support features that were not part of the Cert RTP added additional time to verify these new libraries against Cert restrictions.

### *B. Summary*

Using a pre-defined certified profile such as Cert was highly beneficial in addressing several of the safety-critical restrictions with minimal costs and risks. The ARINC 653 Apex [8] interface made transitioning to the Cert RTP straightforward. All changes made the code more reliable, secure, and robust.

## REFERENCES


[1] R. Seacord, "Top 10 Secure Coding Practices" SEI CERT, March 2011, (https://www.securecoding.cert.org/confluence/display/seccode/Top+10+Secure+Coding+Practices).

[2] "Wind River Platform for Safety Critical DO-178B", Wind River, (http://www.windriver.com/products/product-overviews/Platform-DO178B-product-overview.pdf).

[3] "Ada Reference Manual": ISO/IEC-8652:1995(E) with COR. 1:2000.

[4] "GNAT User's Guide Supplement for GNAT Pro Safety-Critical and GNAT Pro High-Security", AdaCore (https://docs.adacore.com/gnathie_ug-docs/html/gnathie_ug/gnathie_ug/about_this_guide.html).

[5] "Runtime library", Wikipedia, https://en.wikipedia.org/wiki/Runtime_library

[6] "GNAT Pro, Designed for the Ada Professional", AdaCore, http://www.adacore.com/gnatpro

[7] "Wind River VxWorks 653 Platform", Wind River, http://www.windriver.com/products/product-overviews/PO_VxWorks653_Platform_0210.pdf

[8] "Avionics Application Software Standard Interface Part 1- Required Services", AEEC, Aeronautical Radio, Inc, 15 November 2010.