RELIABLE SOFTWARE DESIGN AND PRODUCTION

Software has bugs, this is well known. Unfortunately, bugs in some critical application contexts (e.g., avionics) can be quite dangerous. Reliable software development follows processes that have been developed in order to guarantee (as much as possible) no software errors. Historically, such processes have been applied only to niche areas, recognized as critical. However, nowadays, with the development of the IoT, software correctness is becoming more and more important, since a small bug in an apparently harmless appliance can be exploited (actually, they have been exploited in quite a few occasions) in order to carry out attacks against critical infrastructures. The objective of this course is to teach techniques for the development of critical software. The course is organized in three parts. The first part of the course is an introduction to software correctness and certification; the second part concentrates on one specific technique, namely the use of the Ranvenscar profile in the context of safety-critical concurrent software; finally, the last part gives an introduction to "engineering-level" cryptography and its correct use in critical applications.

LECTURERS

Massimo Bombino
Software Sicuro srl, Italy
DAY 1: Software certification

Riccardo Bernardini
University of Udine, Italy
DAY 2: Introduction to cryptography

Tullio Vardanega
University of Padua, Italy
DAY 3: The Ravenscar profile in software development

REQUIREMENTS

DAY 1 No special requirements

DAY 2 A PC with Octave is required. Octave is available for Linux, MacOS, Windows and BSD and it is freely downloadable from www.gnu.org/software/octave/download.html

DAY 3 A PC with the AdaCore GPS environment is required for the second part. The GPS environment is available both for Windows and Linux and it is freely downloadable from www.adacore.com/download.

ADMISSION

The webinar course is addressed to doctoral students, young researchers and professionals with interest in software safety, on a first come first served basis.

The registration fee is 100,00 Euro + VAT taxes*, where applicable (bank charges are not included).

Undergraduate and postgraduate students (PhD) as well as young researchers are exempted from the registration fee.

Online registration is available at https://www.cism.it/en/activities/courses/E2004

A message of confirmation will be sent to accepted participants.

The application deadline is December 7, 2020.

For further information, please visit CISM website.

* Italian VAT is 22%

For further information please contact:
CISM
Palazzo del Torso
Piazza Garibaldi 18
33100 Udine (Italy)
tel. +39 0432 248511 (6 lines)
fax +39 0432 248550
e-mail: cism@cism.it | www.cism.it

ACADEMIC YEAR 2020

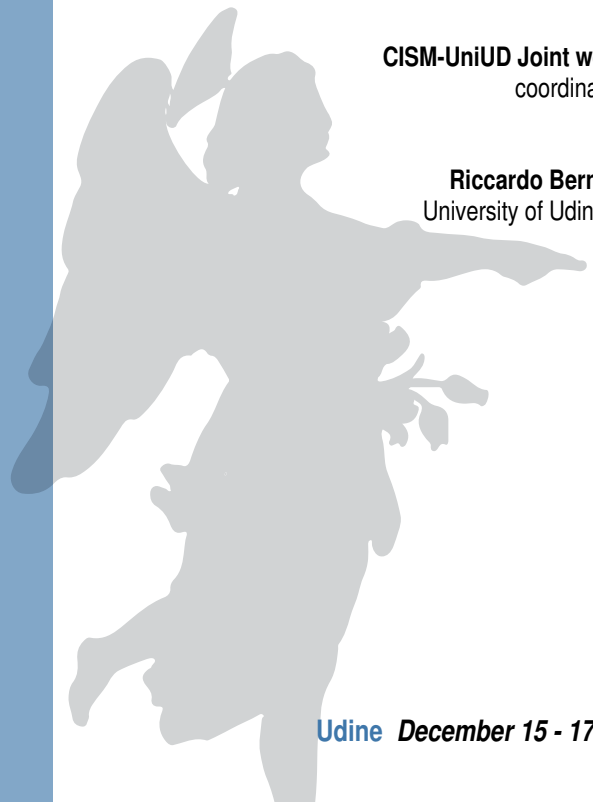
University of Udine
International Centre for Mechanical Sciences

UNIVERSITÀ DEGLI STUDI DI UDINE
hic sunt futura

Centro Internazionale di Scienze Meccaniche
International Centre for Mechanical Sciences

RELIABLE SOFTWARE DESIGN AND PRODUCTION

Webinar



CISM-UniUD Joint webinar
coordinated by

Riccardo Bernardini
University of Udine, Italy

Udine ***December 15 - 17*** 2020

December 15 - DAY 1
Software certification

Software certification has a long history of applications in contexts where a bug can result in important damages, including loss of human lives. The classical application context of software certification is aerospace where certification proved its usefulness: up today not a single death in flight accident can be ascribed to software bugs. Nowadays, everything is interconnected and a bug in a seemly innocent device (a smart lamp, a connected toothbrush or a stereo system) can result in a major failure of a larger system.

The rise of IoT and consequent damages done by hackers exploiting buggy IoT devices, it is forcing engineers to pay more attention to software correctness and it is reasonable to assume that in few years the need for software certification will leave aerospace to colonize all the software application areas such as IoT, business software, AI, and others.

OBJECTIVES

This part of the coursewill introduce the audience to the concept of software certification, starting from the ideas behind certifications, moving to the DO-178C (the American certification standard for aviation) and then closing with the study of some practical cases from real world.

TIME TABLE

09:15 - 10:00 Safety-critical software and certification standards

10:00 - 10:30 *Coffee break*

10:30 - 12:00 DO-178C and other derived certifications:

12:00 - 14:00 *Lunch break*

14:00 - 15:30 Evolution of Software Certification and Correctness: Model-ling, Formal Methods

15:30 - 16:00 *Coffee break*

16:00 - 17:30 Some practical cases from real world

December 16 - DAY 2
Introduction to cryptography

Software correctness is clearly a condition necessary for security, but it is not sufficient. A piece of software can be correct (in the sense that it behaves as desired), however if informations are not protected during transmission a malicious opponent could disrupt the system.

Cryptography provides a set of tools that can be used to protect the system. It can hide the transmitted information so that unauthorized recipient cannot access it; it can guarantee for the correctness of the received information so that users know that it was not tampered with; it can be used to authenticate the user, guaranteeing that only authorized users can actually use the system.

Cryptography can be taught at three different levels. At the most abstract level we find theoretical cryptography, a branch of mathematics, aiming to develop new cryptographic tools and primitives. The least abstract level is of interest of system managers and practitioners who need to know, for example, how to set up a certificate in a server. This part of the course will refer to the intermediate "engineering" level, that is how to use the tools developed at the most abstract level to design and build a system that satisfies some security requirements.

OBJECTIVES

At the end of this part the participants will know the cryptography jargon, the main cryptography building blocks (cyphers, hash functions, signatures, ...) and the most common attack techniques together with the corresponding countermeasures.

TIME TABLE

09:15 - 10:00 Introductory material: jargon and math tools

10:00 - 10:30 *Coffee break*

10:30 - 12:00 Cryptographic building blocks: encryption

12:00 - 14:00 *Lunch break*

14:00 - 15:30 Cryptographic building blocks: hash, signatures...

15:30 - 16:00 *Coffee break*

16:00 - 17:30 Most important weakness and attacks

December 17 - DAY 3
The Ravenscar profile in software development

The Ravenscar Profile (RP) is a compiler-enforced subset of the concurrency-and-synchronization model supported by the Ada programming language. The RP is interesting in a number of ways. Ada has been one of the first programming languages to embrace concurrent programming and to provide structured support for it, designed around algebraic principles that would give it solid grounds.

The RP originated in 1997, after the 1995 periodic revision of the language, which made Ada's concurrency slicker, making room for data-oriented synchronization in contrast with traditional control-synchronization. This feature was fit for use in real-time systems that would employ (restricted forms of) concurrency instead of static schedule tables, seeking better responsiveness, flexibility, and time-efficiency.

Technically, the RP is a set of restrictions, designed so that their use in an application would provide three key benefits: (1) Stripping the run-time system of all excluded features; (2) allowing the compiler to ascertain statically the conformance of the source program to the profile restrictions and (3) enabling the application of advanced scheduling analysis to the system. The resulting system would be a small-footprint image, perfectly equipped to run on resource-constrained bare-board hardware, with no operating system.

OBJECTIVES

This part of the course will introduce the audience to the Ravenscar Profile, both from a theoretical (in the morning) and practical (in the afternoon) viewpoint. At the end of this part the participants will know what the RP is and how to use it in practical cases.

TIME TABLE

09.15 - 10.00 Introduction: Basic model

10.00 - 10.30 *Coffee break*

10.30 - 11.15 Fixed-priority scheduling: Allowing task interaction; Exercises on task interaction

11.15 - 11.45 The Ravenscar profile

11.45 - 12.00 Q&A session

12.00 - 13.30 *Lunch break*

13.30 - 15.00 Hands-on lab

15.00 - 15.15 *Coffee break*

15.15 - 16.15 Lessons learned (interactive)