RELIABLE SYSTEMS DESIGN AND PRODUCTION

Software has bugs, this is well known. Unfortunately, bugs in some critical application contexts (e.g., avionics) can be quite dangerous. Critical software development follows processes that have been developed in order to guarantee (as much as possible) the absence of errors. Historically, such processes have been applied only to niche areas, recognized as critical. However, nowadays, with the coming of the IoT, software correctness is becoming more and more important, since a small bug in a apparently harmless appliance can be exploited (actually, they have been exploited) 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 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

Tullio Vardanega
University of Padua, Italy
DAY 2: The Ravenscar profile in software development

Riccardo Bernardini
University of Udine, Italy
DAY 3: Introduction to cryptography

REQUIREMENTS

DAY 1 No special requirements

DAY 2 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.

DAY 3 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

A PC with all the required software pre-installed can be rented at an additional handling fee of 75 Euro.

ADMISSION AND ACCOMMODATION

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

The registration fee is **250,00 Euro** + VAT taxes*, where applicable (bank charges are not included).

The registration fee includes a complimentary bag, three fixed menu buffet lunches, coffee breaks, downloadable lecture notes and wi-fi internet access.

Applications should be made on-line through our web site:
https://www.cism.it/en/activities/courses/E2004/
A message of confirmation will be sent to accepted participants. Information about travel and accommodation is available on our web site, or can be mailed upon request.

A limited number of rooms is available at our Guest House at the rate of Euro 30,00 per person/night.
Applicants may cancel their course registration and receive a full refund by notifying CISM Secretariat in writing (by email) no later than two weeks prior to the start of the course.
If cancellation occurs less than two weeks prior to the start of the course, a Euro 50,00 handling fee will be charged.
Incorrect payments are subject to Euro 50,00 handling fee.

* 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

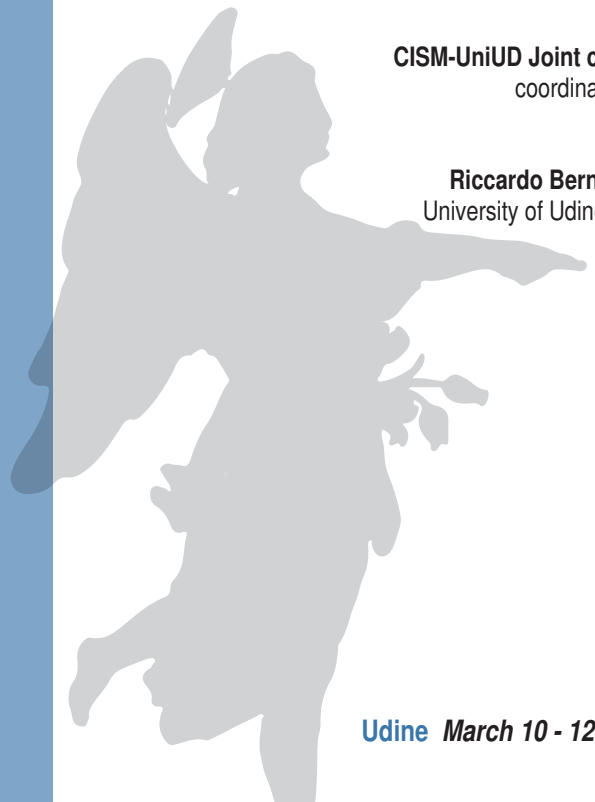
ACADEMIC YEAR 2020

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 SYSTEMS DESIGN AND PRODUCTION



CISM-UniUD Joint course
coordinated by

Riccardo Bernardini
University of Udine, Italy

Udine ***March 10 - 12*** **2020**

March 10 - DAY 1
Software certification

Software certification has a long story 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 (the recent accidents of Boeing 787 MAX was due to a system design problem, not to a software bug). Nowadays, everything is interconnected and a bug in a seemly innocent device (a smart lamp, a connected toothbrush or stereo system) can result in a major failure of a larger system (remember, for example, the 2015 case of the Jeep Cheerokee remotely controlled via a bug in the entertainment system).

The emergence of IoT and consequent damages done by hackers exploiting buggy IoT devices, it is forcing the public 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: IoT, business software, AI, ...

The objective of this one-day seminar is to introduce the students to the concept of software certification, starting from the ideas behind certifications, moving to the DO-178C (the American certification standard for aviation) and completing with the study of some practical cases from real world.

OVERVIEW

- Software correctness: definition, theory and applications
- Safety-critical software and certification standards
- Avionics standard DO-178C and other derived certifications
- Principles of DO-178C: Plans, Standards, Process, Checklists, Deliverables
- Evolution of Software Certification and Correctness: Modeling, Formal Methods
- 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
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

March 11 - DAY 2
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. Having a comparatively long history, and having chosen to be an international standard, Ada has undergone multiple periodic revisions based on use feedback, requirements capture, and an official approval process. This trait has slowed down undoubtedly the launch of new features, but has given them the benefit of deep thinking.

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 very 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 of the excluded features. (2) Causing the compiler to ascertain statically the conformance of the source program to the profile restrictions. (3) Enabling the application of advanced schedulability analysis to the system, with full correspondence between the actual application behavior and the model of analysis. The resulting system would be a small-footprint image, perfectly equipped to run on resource-constrained bare-board hardware, without the need for operating systems or some such.

OVERVIEW

This seminar will center on the theory and practice of the Ravenscar Profile. The seminar will be comprised of two parts.

- In part 1 (morning session), the instructor – who was one of the key figures behind the specification of the RP – will walk the participants through the on the origin, intent, specification and use of the Ravenscar Profile.
- In part 2 (afternoon session), the instructor will guide the participants through live programming exercises, in the intent to match them to the promises and claims made in the morning.

TIME TABLE

09:15 - 10:00 The Ravenscar profile: origin and motivation
10:00 - 10:30 Coffee break
10:30 - 11:45 The Ravenscar profile: match with response time analysis
11:45 - 12:30 First Q&A session
12:30 - 14:00 Lunch
14:00 - 15:30 The Ravenscar profile: practical use cases (design and coding)
15:30 - 16:00 Coffee break
16:00 - 17:00 The Ravenscar profile: practical use cases (execution and analysis)
17:00 - 17:30 Second Q&A session

March 12 - DAY 3
Introduction to cryptography

This course aims to give an "engineering competence" about cryptography. At the end of this course the students

- will know the cryptography jargon
- will know the main cryptography building blocks (cyphers, hash functions, signatures, ...) and what they are used for
- will know the most common attack techniques and how to design a system in order to thwart them.

OVERVIEW

- Introduction and jargon
- Algebraic tools
- Cryptography building blocks
- Encryption (symmetric/asymmetric/identity-based)
- Cryptographic building blocks: Hash, Signatures, Key-exchange, Random number generation,...
- Most important weakness and attack techniques

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
14:00 - 15:30 Cryptographic building blocks: hash, signatures...
15:30 - 16:00 Coffee break
16:00 - 17:30 Most important weakness and attacks