Module Description, available in: EN

Software Assurance

General Information

Number of ECTS Credits

3

Module code

TSM_SoftwAs

Valid for academic year

2025-26

Last modification

2023-07-03

Coordinator of the module

Stephan Neuhaus (ZHAW, stephan.neuhaus@zhaw.ch)

Explanations regarding the language definitions for each location:

- Instruction is given in the language defined below for each location/each time the module is held.
- Documentation is available in the languages defined below. Where documents are in several languages, the percentage distribution is shown (100% = all the documentation).
- The examination is available 100% in the languages shown for each location/each time it is held.

	Lausanne			Lugano	Zurich		
Instruction					**X** E 100%		
Documentation					**X** E 100%		
Examination					**X** E 100%		

Module Category

TSM Technical scientific module

Lessons

2 lecture periods and 1 tutorial period per week

Entry level competences

Prerequisites, previous knowledge

Students will need knowledge in software engineering, specifically testing.

Students will need to be reasonably fluent in a variety of languages including but not limited to C and Python. Knowledge of some assembly (e.g., x86, x86-64, or ARM) will be advantageous.

Students will need to be familiar with the idea that there are standards for software development and testing.

Brief course description of module objectives and content

Students shall gain an overview over current methods for software assurance. This includes

- automatic test case minimisation;
- negative test case generation ("fuzzing");
- side channels and their avoidance ("constant-time computing");
- security implications when designing safety systems

- exposure to standards-compliant software development;
- software verification and validation;
- safe testing according to the standards; and
- fault tolerance.

Aims, content, methods

Learning objectives and acquired competencies

- Students can apply test case minimisation techniques to their own test cases.
- Students know how fuzzing works, to what class of faults it applies, how to interpret its output, and how to use it in their own projects.
- Students know that side channels exist and how they are exploited, that they are a serious danger to software assurance and security, and how to avoid certain types of side channel, especially those that have to do with variable-time computation based on secret inputs.
- Students know about the safety life cycle according to IEC 61508 and its adaptation to automotive security in ISO 26262, and can apply it in their own projects.
- Students can apply probabilistic methods used to estimate the impact of device failures on overall safety.
- Students know what options there are to certify, validate, and verify software components, and what that means.

Contents of module with emphasis on teaching content

- Safety life cycle according to IEC 61508 (2 lectures)
- Application of ISC 61508 to automotive software (ISO 26262) (1 lecture)
- Probabilistic methods to estimate impact of failure (2 lectures)
- Certification, validation, and verification of software (2 lectures)
- Test cases and their minimisation (2 lectures)
- Negative test case generation ("fuzzing") (2 lectures)
- Side channels (3 lectures)

Teaching and learning methods

Lectures will be part ex-cathedra, part in-class exercises. These exercises are designed to be done either individually or in groups and can therefore be done remotely.

Literature

Andreas Zeller, *Why Programs Fail*. Morgan Kaufman. Second Edition, 1770. (Yes, that's the date that Amazon has for the book. In reality, the second edition is from 2008.)

Ari Takanen, *Fuzzing for Software Security Testing and Quality Assurance*. Artech House Publishers. Second Edition, 2018.

Seokhie Hong (Ed.), *Side Channel Attacks*. MDPI. 2019.

David J. Smith and Kenneth G. L. Simpson, *The Safety Critical Systems Handbook: A Straightforward Guide to Functional Safety*: IEC 61508 (2010 Edition), IEC 61511 (2015 Edition) and Related Guidance. Butterworth-Heisman. Fifth edition, 2020.

Assessment

Additional performance assessment during the semester

The module does not contain an additional performance assessment during the semester

Basic principle for exams

As a rule, all the standard final exams for modules and also all resit exams are to be in written form

Standard final exam for a module and written resit exam

Kind of exam

written

Duration of exam

120 minutes

Permissible aids

Aids permitted as specified below:

Permissible electronic aids

- Open book
- Open Internet

Other permissible aids

None

Exception: In case of an electronic Moodle exam, adjustments to the permissible aids may occur. Lecturers will announce the final permissible aids prior to the exam session.

Special case: Resit exam as oral exam

Kind of exam

oral

Duration of exam

30 minutes

Permissible aids

Aids permitted as specified below:

Permissible electronic aids

- Open Book
- Open Internet

Other permissible aids

None