UNIVERSITÀ DEGLI STUDI DI NAPOLI FEDERICO II

itee PhD information technology electrical engineering

DIETI UNINA

PhD student Riccardo Carbone

Agile Development for Safety-Critical Software

Tutor: Valentina Casola

Cycle: XXXVII **Year:** 2

itee PhD information technology electrical engineering

My background

- **MSc degree** → Computer Engineering
- **Research group** → Seclab
- **PhD start date** → 01/11/2021
- **Scholarship** → No scholarship
- **Current position** → Software Embedded Engineer at Rete Ferroviaria Italiana S.p.A. Research and Development Department (no company funded scholarship)

Research field of interest

- Critical software engineering:

  1. The adoption of agile software development methods for the research and development of critical software:

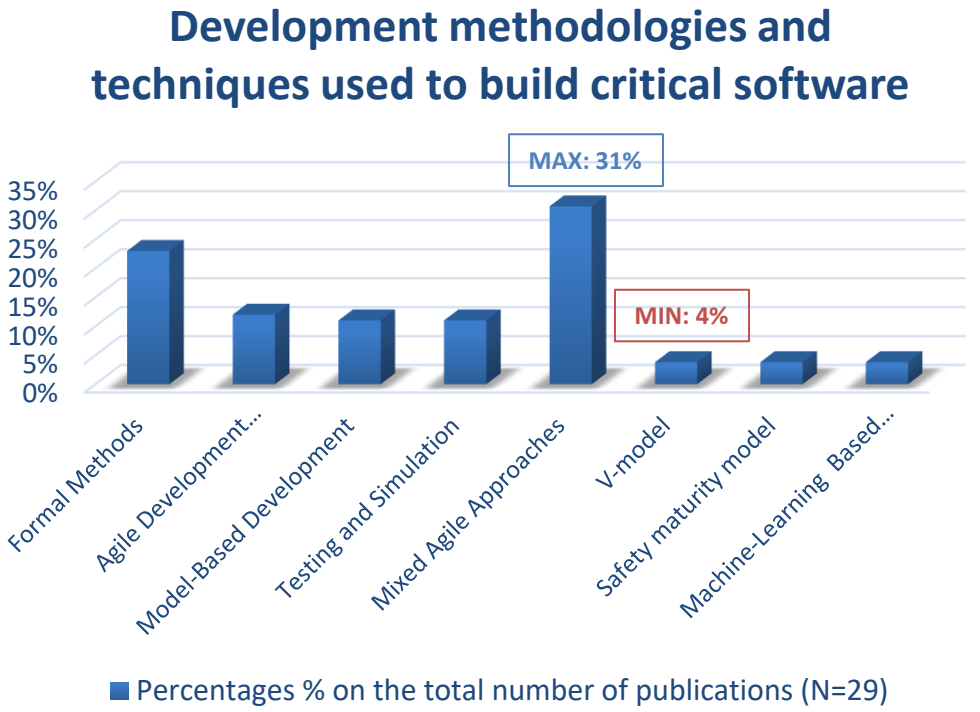
     - Who: R&D organization departments;
     - Why: the standard V-model makes it difficult to manage software requirements uncertainty while optimizing project costs;

  2. The documentation, traceability, and testing of software non-functional requirements when using agile development models:

     - Who: Agile development teams;
     - Why: agile software development practices are much more oriented to functional requirements despite the criticality of non-functional requirements.

Summary of the state of the art (1/2)

- Critical software engineering is shifting from a standard waterfall development approach to much more flexible **hybrid agile methodologies**:



Reference databases: IEEE Xplore, ACM DL, and Science Direct
Period: 2018-2023

Summary of the state of the art (2/2)

- The main documented reasons for that are:
  - Organization towards **requirements change**;
  - The **reduced project costs and delivery time**;
  - The possibility of performing **verification and validation activities** from the beginning of the project.

- However, the lack of explicit methods in current agile methodologies for the documentation, traceability, and testing of **non-functional software requirements** represents an essential obstacle for developers.

- The solution to these challenges is of great importance since they can significantly impact the agility of development teams.

The Scrum 4 Safety development model

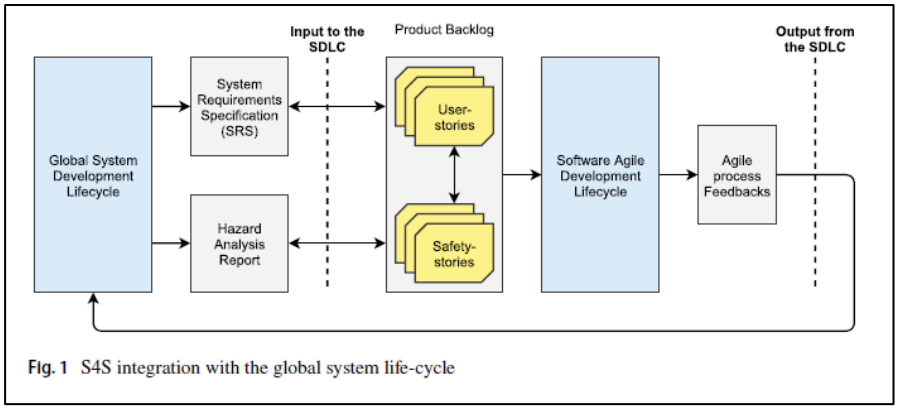


Fig. 1 S4S integration with the global system life-cycle

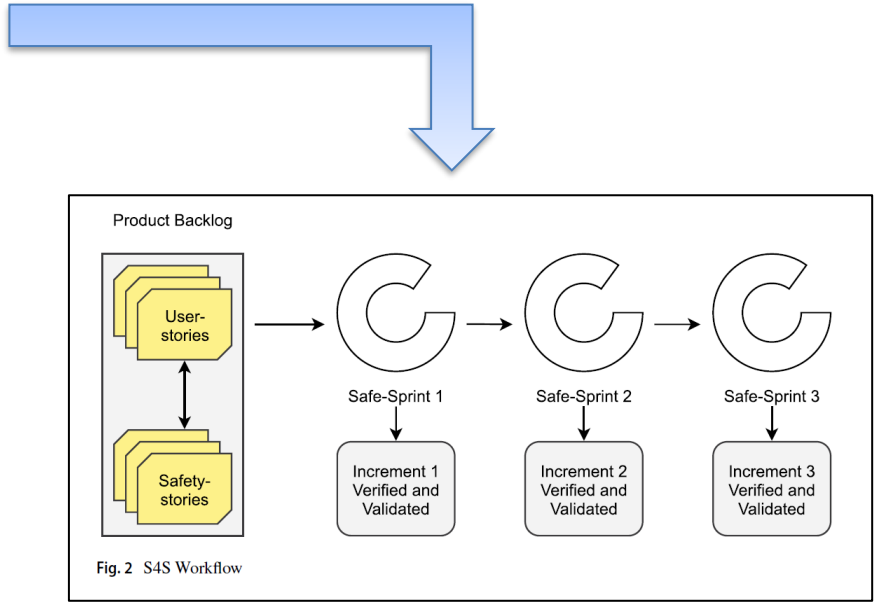


Fig. 2 S4S Workflow

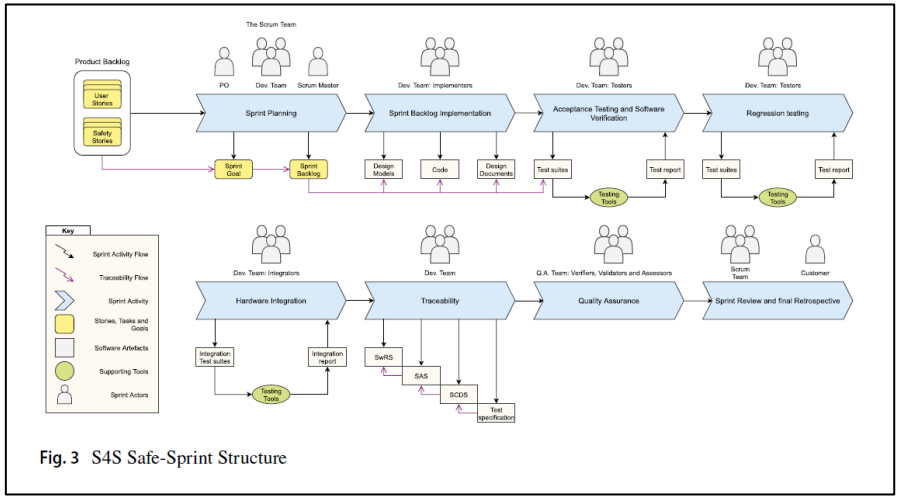


Fig. 3 S4S Safe-Sprint Structure

Figures from: https://link.springer.com/article/10.1007/s11219-022-09593-2

Research activity: Testing of safety-critical software

Problem

Performing safety-critical software testing with an agile development methodology is difficult since it is considered one of the most expensive and time-consuming activities.

State of the art

The test of safety-critical software is made possible by expensive and certified instruments, which allows developers to execute the product in its native environment with minimum interference. Nevertheless, it remains challenging for developers to learn and use these tools for their testing objectives.

Objective

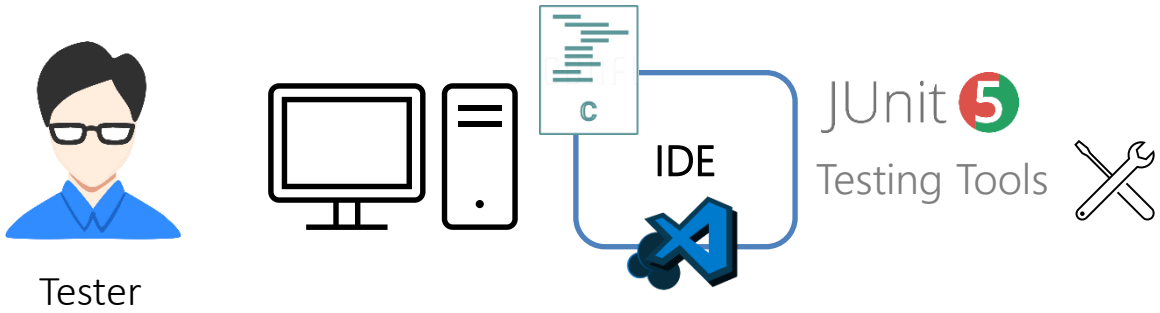
In order to make developers much more proficient in the production of software test cases, we propose to overcome the state-of-the-art limits by extending the available tools with the **core concepts of xUnit frameworks**, which represent the de-facto standard in the field of agile software testing.

Research Methodology

- **Step 1**: identification of the core concepts of xUnit that could be reused for safety-critical software testing;
- **Step 2**: construction of a set of guidelines, methods, and instruments to support developers during test case production;
- **Step 3**: validation of the solution with multiple case studies from Rete Ferroviaria Italiana S.p.A.

Reusing xUnit frameworks concepts in safety-critical environments (1/2)

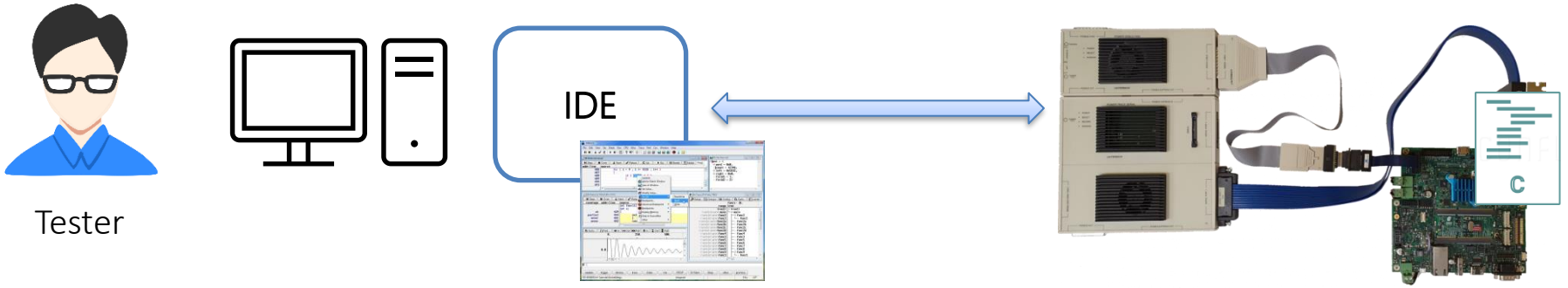
Safety unrelated world



xUnit frameworks:

- Assertion model;
- Conditional tests;
- Repeated tests;
- Parametrized tests;
- Dynamic tests;
- Test templates;
- …

Safety related world



In this case, we have a large gap between the Tester and its instrumentation.

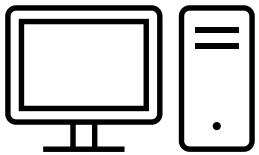
Lauterbach Trace32:

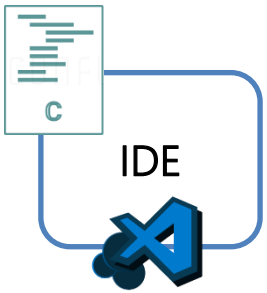
- CPU (ARM/Intel) non-intrusive tracing and debugging;
- Structural code coverage;
- Proprietary scripting language (PRACTICE) to write down automated test cases.

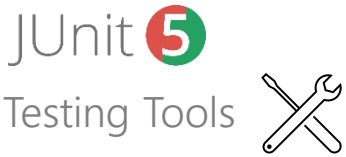
Reusing xUnit frameworks concepts in safety-critical environments (2/2)

Safety related world

Tester



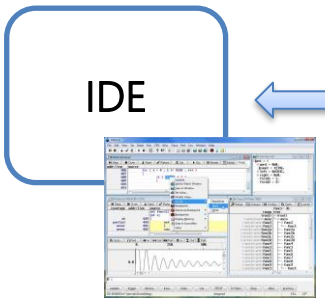


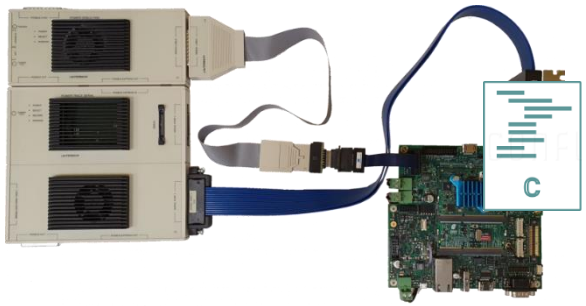


xUnit frameworks:

- Assertion model;
- Conditional tests;
- Repeated tests;
- Parametrized tests;
- Dynamic tests;
- Test templates;
- …

Translation of Junit test cases





Lauterbach Trace32:

- CPU (ARM/Intel) non-intrusive tracing and debugging;
- Structural code coverage;
- Proprietary scripting language (PRACTICE) to write down automated test cases.

Research activity: Documentation and traceability of quality requirements

Problem

Agile development methodologies mainly rely on direct communication and evolutive requirements that are often represented by developers with small pieces of information such as user stories and use cases.

State of the art

Researchers have extensively explored the importance of documenting quality requirements (QRs) and their related challenges. However, very few proposals were made for guidelines and practices to guide developers in achieving this task.

Objective

The definition of a complete framework that developers can follow to specify, trace, and test their QRs.

Research Methodology

- **Step 1:** identification of the current guidelines and practices for specifying QRs in agile environments;
- **Step 2:** construction of a consistent set of guidelines, practices, and templates to support developers during the QRs management process;
- **Step 3:** validation of the solution with multiple case studies from Rete Ferroviaria Italiana S.p.A.

Documentation and traceability of quality requirements