Some tips on Assignment 1
February 16, 2023

In the specification of Assignment 1 it is suggested (it is not mandatory) to the student groups to focus the survey work on a specific aspect (a subtopic) of the theme proposed for the assignment.

What is written in the specification of the assignment is the following:

> *"Since the theme "Survey on software quality and dependability concepts, terminology and software fault tolerance techniques" is very broad, it is recommended to decide what you are going to cover in more detail in your survey. It is wise to start with the general overview before focussing on a possible subtopic. This is essential to provide the reader with a clear view on where the subtopic focused on your survey is in the context of the whole theme."*

Here are some suggestions of possible subtopics (just some of them… this is not a complete list; the order is not relevant):

Subtopics about **means for dependable software** (please, also have a look at the slides of the course):

- **Runtime error detection** (consider only the software implemented techniques)
  Software techniques to detect errors during the execution of programs. These techniques are used during software development, but they can also be integrated in the software and used during the entire life cycle of the product. They are also known as concurrent software error detection.
- **Software fault tolerance**
  This is a general topic that includes many types of techniques. The general idea if to include software mechanisms in the systems to tolerate faults. The word "tolerate" means to deal with the effects of faults (the errors) to avoid failures (i.e., wrong behaviour and wrong outputs of. the software system). Examples of specific software tolerance techniques includes:
  - Software diversity and N-version programming
  - Self-checking software
  - Error recovery and recovery blocks
  - (there are more)

  The assignment can be focused on one of these types of fault tolerance techniques.
- **Intrusion tolerance**
  This is basically the equivalent of fault tolerance techniques focused on defending and protecting the software systems against malicious attacks. In other words, while software fault tolerance techniques deal with accidental faults (software and hardware faults) intrusion tolerance deals with security attacks. There is a (partial) parallel between the two
- **Software fault injection**
  There are many techniques of fault injection. All of these work a bit like vaccines: we "inoculate" to evaluate how they behave in the presence of faults and verify that the mechanism available in the system to deal with faults are really working as expected. For the Assignment 1 the relevant type of fault injection is "software fault injection". That is, the injection of software faults (i.e., bugs) in the software to observe how the system behaves in the presence of components with active bugs. Software fault injection is used for many purposes relate to the evaluation of software and system dependability.
- **Mutation testing**
  It is a specific type of fault injection used to evaluate the quality of test suits (i.e., set of tests). The faults injected in mutation testing do not try to emulate (simulate) real software bugs, a is the case of general software fault injection. In mutation the idea is simply change (mutate) very small portions of the code in a way that is syntactically correct (the program must compile) to evaluate if the test cases can detect all the mutants. The target is to evaluate the test cases (not the software). But, as we will see in the next T classes, having good test cases is essential to have high quality software.

- **Software verification and validation**
  It is the general process (i.e., it includes several methods and techniques) of verifying that the software meets the requirements and specifications and validates that the software really fulfils the intended purpose. The two words (verification and validation) have different technical meaning. There is a kind of anecdotical saying that is often used to clarify the difference: "Verification means: Are we building the (software) product right? Validation means: Are we building the right product?

- **Static program code analysis**
  The idea of static code analysis is to analyse the source code to identify errors or potential problems. In general, the IDEs (integrated development environments) provide some basic static code analysis features, but this is normally done by automatic tools (outside the IDEs). Although static code analysis can detect all types of problem, most of the tools are focused on software vulnerabilities. That is why static code analysis is often associated to software security.

- **Vulnerability and attack injection**
  Follows a similar idea of software fault injection. But in this case the goal is to inject software vulnerabilities in a given software component (e.g., a web application) and use such vulnerabilities to launch attacks into the system. The goal is to evaluate intrusion tolerance techniques implemented in the system such as intrusion detection systems.

- **Software maintainability and code complexity**
  Software maintainability is a large software quality area that includes the concepts of modularity, code understandability, code testability, reusability, etc. For the groups interested on this topic, the suggestion is to look at software maintainability from the point of view of software complexity metrics, as these metrics play an important role on software maintainability. Furthermore, this is a way to focus the work on subtopics of maintainability (i.e., complexity metrics and software maintainability) that are directly related to software quality.

- **And many more…**