Software Testing Tutorial LO 1 2025

There are *five* tutorial topics scheduled for the Software Testing course. Each of these will be covered over two weeks. In the first week you will work on a preparatory task individually and in the second the tutorial group will work with the tutor. The goal is to use the software you are working with to exemplify what each section of the portfolio should look like.

Recall LO 1 is "*Analyze requirements to determine appropriate testing strategies*" so this section of your portfolio should provide a range of requirements and some analysis of what might be necessary adequately to test those requirements. You may develop a longer list of requirements, but the portfolio should mention a small number that are distinctively different from one another.

Preparation, week 2 (22 Sept-26 Sept)

You will find reading Chapters 2 and 3 of Y&P useful in preparing for this activity. You should also refer ISO/IEC 29119-1 – have a look at Figure 8 on page 32 of 29119-1 and the following couple of paragraphs that provide an idea of how the main parts of 29119 relate to each other. There is an example of a test model on page 44 of 29119-2. In this section you are thinking of developing a test basis and required behaviour that are key elements in defining a test model. You do not need to understand fully the ideas on these pages but you should get a grasp of the sorts of requirements (test basis and required behaviour) you need to give you a diverse collection of tests to ensure you have good diversity for your LO1 portfolio.

You should work through the following activity:

- This example is very simplified compared to a real-world system of this type, but it does illustrate some of the ideas. Adaptive Cruise Control (ACC) is a very common feature in automobiles (e.g. see https://www.bosch-mobility.com/en/solutions/assistance-systems/adaptive-cruise-control/)
- Suppose we have been asked to develop a specification for a very simplified ACC system. Suppose the system has:
    - **Inputs**: current speed; current distance from obstacles in front of the car (e.g. other cars); the set speed the car should travel at; and
    - **Outputs:** accelerate, brake or do nothing.
- Can you develop some requirements for the system that are reasonably diverse and will result in the need for diverse approaches to test to ensure you have a good diversity of needed tests. Don't spend too long on this say around an hour but do think about the considerations spelt out below.
- Consider:
    - Who are the stakeholders in this software (remember that as well as the primary users they may be other stakeholders who have different perspectives on the software, e.g. an app may have primary users but there are also other potential users e.g. people who deliver the service ordered on the app)

- From each of the stakeholders' perspectives, identify a small number (1-3) requirements that are important to those stakeholders. These can be quite informal, stated in natural language and imprecise in some ways. Try to have a diversity of different kinds of requirement.
- In thinking about diversity of requirements, recall:
  - Some requirements are *functional* in that they specify some characteristic of the activity of the system e.g.
    - Safety properties say the system can't do some things.
    - Correctness properties say that the result is correct.
    - Liveness properties say that the system always returns to some recognised "ready" state.
    - Security properties
  - Some requirements make statements about a measurable quality attribute of the system e.g.
    - Statements about resource use (memory, compute time, …)
    - Statements about an "-ility" reliability, usability, availability, …
    - Statements about timing e.g. response times
- Try to ensure you have a good range.

- Once you have done this for the example system, think about the software you have chosen to test as part of the Software Testing Course and try to develop a diverse collection of requirements for it together with approaches to testing the requirements.

Tutored session, week 3 (29 Sept-3 Oct)

In this session you will consider some of the requirements and proposed tests you have developed in the preparatory week with the tutor and consider their strengths and weaknesses. The goal is to consider what would be a good section on LO 1 for your portfolio. This will involve assessing potential sections against the grading scheme. To do this we will consider the marking scheme for the portfolio because that indicates what emphasis is expected in the portfolio. Here are the four criteria LO 1 is graded on (recall each is graded in the range 0-5) with the interpretation provided in the Marking Scheme:

1.1. *Range of requirements, functional requirements, measurable quality attributes, qualitative requirements, …* At this point the focus is on the identification of system level requirements. The emphasis is on the range of different types of requirement so in the tutorial we should consider what would make a good combination of requirements to point to in the portfolio. The evidence to support this criterion is a list structured by types of requirement that good coverage of different types of requirement. This should be a short document stored in your GitLab project that this section of the portfolio can refer to.

1.2. *Level of requirements, system, integration, unit.* At this stage it may only be possible to refer to system level requirements because you have yet to consider an implementation where it will be necessary to develop unit and integration tests. It is possible to indicate the sorts of unit and integration tests that will be likely to be necessary and these can be refined once you have started developing some code. For some systems there are clear sub-systems that must appear in any implementation and for such systems you will be able to discuss requirements for sub systems and how they integrate.

1.3. *Identifying test approach for chosen attributes.* As the course progresses you will be able to elaborate this section somewhat but even at this early stage you can develop some broad statements about what approaches to test would be appropriate and which are not appropriate.

1.4. *Assess the appropriateness of your chosen testing approach.* At this early stage you should be able to point out some of the limitations of your approach to testing some particular requirement. For example, it may be that lack of data makes it difficult to do performance testing under realistic loads.