01Lab
Requirements, Acceptance Testing, and BDD/ATDD

Mirko Viroli
mirko.viroli@unibo.it

C.D.L. Magistrale in Ingegneria e Scienze Informatiche
Alma Mater Studiorum—Università di Bologna, Cesena

a.a. 2023/2024

Again on labs at ASMD, and exam

Lab

- each module has a lab activity
- in each lab we propose various tasks, exercising various skills
  - some are operative (and quick), some requires some additional work at home
  - some are actual "R& D tasks", like exam projects
- students pick those they like most, and start working on them
- freely, students complete some tasks at home
- they work alone, or in small groups (typically, in pairs)
- the teacher provides assistance at need
  - during the lab, or later (offline)
  - share with the teacher interesting results or completed tasks
  - send email with subject "`[ASMD23-LABXX-TASKYY]`" with authors/repo in the body

Exam

- assume $\sim$ 90 hours of work (additional to labs)
- discussion of tasks completed by students, e.g.:
  - one big task, producing a software artifact or a scientific paper
  - various smaller tasks
- discussion of links to other parts of the course

One slide sum-up on Acceptance Testing

Requirements and specification

- a requirement is a stakeholder's expression of a need/wish with regard to the system
- a specification is a description of system behaviour required to fulfil a requirement

ATDD/BDD

- acceptance testing: testing on overall system to determine approval by stakeholders
- ATDD: determine/pick an acceptance test, TDD to make it pass
- BDD: essentially ATDD, using behaviour-oriented tests vs acceptance tests
- in ATDD/BDD, use tests expressing "features"

Gherkin/cucumber

- Gherkin: language to express features (mixing prose/keywords)
- Cucumber: binding Gherkin to code execution (of "steps")

Starting point and goals

References

- Cucumber: `https://cucumber.io`
- Gherkin: `https://cucumber.io/docs/gherkin/reference/`
- 01-repo-atdd: `https://github.com/mviroli/asmd23-public-01-atdd`
- ChatGPT: `https://chat.openai.com/`
- Slide 01b provide rather complete examples of Gherkin syntax

General goals

- be operative with Cucumber/JUnit/Java/IntelliJ
- exercise writing requirements with Gherkin
- exercise ATDD
- pre-check ability of ChatGPT to deal with Gherkin

Tasks

CALCULATOR

The repo has a calculator example (with Gherkin specification, step implementation, and production code). Play with it. Add examples and operations, use all Gherkin features. Strictly use ATDD. Does everything work as expected?

TOOLING

Experiment with installing/using Cucumber with Scala and/or in VSCode. Is VSCode better at all here? Does Cucumber play well with Scala 3?

REENGINEER

Take an existing implemented small app with GUI, e.g. an OOP exam. Write Gherkin specifications explaining what the system is expected to do, and make acceptance tests pass. Does the system need a refactor of implementation? What does it tell us about how an application has to be designed to be easily acceptance tested?
Search here: `https://bitbucket.org/mviroli/oop2023-esami` (2023, 2022,...)

REQUIRE

Write Gherkin specifications that completely capture specification of requirements (functional, non-functional) of a real application. Might use a previous project of yours, with requirements already written down, or use any existing/hypothetical (small) application. What are good/bad aspects of Gherkin?

ATDD-LLM

LLMs/ChatGPT can arguably help in write/improve/complete/implement/reverse-engineer a Gherkin specification. Experiment with this, based on the above tasks or in other cases. Is ChatGPT useful for all that?