Technical University of Munich
TUM School of Engineering and Design
Prof. Dr. Martin Werner

Computational Foundations I (WiSe 2023/2024)
Task Sheet 2

Tasks marked with a star like **Optional Task*** are optional. Tasks marked like **Hard Task+** are given, but it is not expected that you solve them now. It is great if you learn to solve them during the lecture. Go back to them after a few weeks and see your own progress.

Learning Outcome: You will learn how to formulate requirements in a reasonable way. Concretely, given vague descriptions of software from business owners/stakeholders, you can transform them into precise statements without ambiguity and ideally with testability. This process would eventually help you to decompose a programming goal into several achievable, easy-to-implement functionalities.

Task 3: Requirement analysis

Suppose we are developing an online judge (OJ) system aiming to automatically evaluate the correctness and quality regarding the running time and the resource usage (complexity) of the submitted code. Now, the OJ website owner wants to ensure that the website can simultaneously deal with at least 10,000 submissions. Meanwhile, the server rental/purchase and maintenance budget should be less than 5,000 euros per month.

a. List functional requirements and non-functional requirements. For non-functional requirements, list quality attributes.

b. The OJ system is finally online. Unfortunately, a hacker begins to attack our system by making a large number of submissions in a very short time, resulting in a Distributed Denial of Service (DDoS) issue. Therefore, we design a new requirement that adding a user to the blocklist if the user makes ten submissions within 1 second. Use `[Trigger][Precondition] Actor Action [Object]` grammar to describe this new requirement and specify every part precisely (e.g., Actor: the OJ system).

c. Recall EARS (Easy Approach to Requirements Syntax). The requirement in (b) is (select the most suitable one)

   (i) a ubiquitous requirement

   (ii) an event-driven requirement

   (iii) an unwanted behavior requirement

d. Now, as a user of this OJ system, write a user story like `As a [role], I want to [capability], so that [benefit]` related to knowing the computation resource consumption of my submission.