Model-Based Testing

Wikipedia [41], the free encyclopedia on the World Wide Web (www), refers to model-based testing as “software testing where test cases are derived in whole or in part from a model that describes some (if not all) aspects of the system under test (SUT)” [39]. The SUT may be something as simple as a method or class, or as complex as a complete system or a solution consisting of multiple systems. For testing, a model provides a behavioral description of the SUT.\(^1\) This description can be processed to yield a set of test cases that can be used to determine whether the SUT conforms to a desirable property that is represented in the model. In this chapter, we identify the phases in the software development process where models are designed and describe the principles of test development based on models.

1.1 The Software Development Process

Literature distinguishes between different software development processes. Examples of such processes are the software life cycle, the waterfall model, the spiral model, the unified process, the V-model, and the W-model [33, 34, 40]. In all these different processes, software is developed in phases. Most of the processes have similar phases and mainly differ in the conditions and possibilities for progressing into the next phase or revisiting a previous phase. A specialty of the V- and W-models is an integrated view of construction and corresponding testing phases. In this book, we use V- and W-models to explain model-based testing and the role of UML Testing Profile (UTP) in the software development process.

\(^1\) A model is an abstraction of a complex problem or system which should be solved or implemented in software or hardware. Behavioral descriptions are only one aspect of a model; further aspects may be related to structural and non-functional requirements.
The principle structure of the V-model is shown in Figure 1.1. The V-model distinguishes between construction phases (shown on the left-hand side in Figure 1.1) and testing phases (right-hand side in Figure 1.1).

System development starts with the Requirements Definition phase. Requirements are captured from the customer and future users. They are used in the following Functional System Design phase to develop a functional model of the system. The functional model should be independent from the future implementation of the system to avoid early design decisions. It may include artifacts from the environment of the system and even business processes of the customer. The software architecture is modeled in the Technical System Design phase. This phase structures the software system into components and defines the interfaces among the constituents. The detailed behavior of the components is defined in the Component Specification phase. The construction phases of the V-model end with the Implementation of the components.

The implementation of the components is the basis for the following testing phases. In the Unit Level Testing phase, component implementations are tested against their specifications. In the next step, Integration Level Testing is used to test the smooth interworking of the finalized components. The integration testing phase ends when all components are integrated and the complete system is ready for System Level Testing. System level testing is the first test where the complete system is available and the complete functionality is tested. The basis for system level testing is the functional system design but may also include tests from the perspective of the developers and system