This chapter gives a detailed overview of the test construction process with MTCC, the roles involved in the process and the models and artifacts used.

The MTCC testing process consists of five steps: (1) the analysis and modeling of the system family under test, (2) the modeling of testees from the system family, (3) the composition of a system-specific test model, representing the available tests for a system, and finally (4) the construction of tests represented as sequences of **Test Step** configurations and (5) the transformation of these abstract representations into executable **Test Steps**.

The MTCC process distinguishes two different roles for participants in the test construction process, domain experts and domain engineers. Domain experts provide knowledge about the functional requirements of a system, domain engineers contribute implementation knowledge and modeling skills. Section 5.1 describes the phases of the MTCC test construction process and the responsibilities of the different roles.

MTCC uses a number of different models. Models address (1) different levels of specificity — for instance at the system family level and system level — and (2) different entities such as the **Services** of a system and the **Test Steps** that exercise these **Services** and (3) different aspects of a system, specifically its features and its dynamic behavior. Section 5.2 introduces the different types of models used in MTCC.

### 5.1 The MTCC Process

The testing process based on MTCC consists of three phases followed by an external test execution phase. Figure 6 illustrates these phases and their respective activities, Figure 5.1 illustrates the relation of the MTCC Phases to the phases of the system family engineering approach as presented by Czarnecki [Cza05].

In the **Domain Engineering** phase, a system family is analyzed with respect to its test relevant **Services**, the features of these **Services** and the **Test Steps** needed to exercise these. Domain Engineering in MTCC is comparable to Domain Analysis [CE00, Ara89, Ara94] applied to system families, it is also analogous to Software Product Line Engineering in that core assets for the testing of the system family under test are identified and implemented.
In Application Engineering, the results of domain engineering are refined. Models describing the system family or domain are specialized for the representation of specific systems. The focus of Test Engineering is the utilization of the test editor that supports domain experts in modeling tests based on the models instantiated in the previous phases of MTCC. Test modeling with the editor is done by the selection and configuration of Test Steps, we call this Test Step instantiation. A Test Step instance represents a concrete interaction with a Service instance of the testee. The feature model of a Test Step instance is a specialization of the Test Step feature model that represents the configurations for the Test Step instance, taking into account the specific features of the Service addressed by the Test Step. The parameterization of a Test Step is expressed as the configuration of the feature model representing the Test Step instance. The use of feature models to model the inputs and expected results of a test case leads to a more fine grained use of feature models than in product configuration, where features often represent the general capabilities of a system [WS06]. In the test code generation activity, Test Step configuration serves as the base for the configuration of executable test cases.

The last phase of Testing with MTCC is not part of the core MTCC process. How the activities of this last phase, test execution and test reporting, are conducted is determined by the test runner for which tests are generated and by the general testing regime in place.

While each of the phases of the MTCC process builds on the results of the previous phase, MTCC does not assume a sequential execution of the different phases. We assume that domain engineering, Application Engineering and test engineering are executed in iterations.