- As necessary, repeat test activities when actions are taken to resolve discrepancies. For example, we might need to re-run a test that previously failed in order to confirm a fix (confirmation testing). We might need to run an updated test. We might also need to run additional, previously executed tests to see whether defects have been introduced in unchanged areas of the software or to see whether a fixed defect now makes another defect apparent (regression testing).
- Verify and update the bi-directional traceability between the test basis, test conditions, test cases, test procedures and test results.

As before, which of these specific tasks applies to a particular project depends on various contextual issues relevant to the project; these are discussed further in Chapter 5.

Test completion

Test completion activities collect data from completed test activities to consolidate experience, testware and any other relevant information. Test completion activities should occur at major project milestones. These can include when a software system is released, when a test project is completed (or cancelled), when an Agile project iteration is finished (e.g. as part of a retrospective meeting), when a test level has been completed or when a maintenance release has been completed. The specific milestones that involve test completion activities should be specified in the test plan.

Test completion The activity that makes test assets available for later use, leaves test environments in a satisfactory condition and communicates the results of testing to relevant stakeholders.

Test completion includes the following major activities:

- Check whether all defect reports are closed, entering change requests or product backlog items for any defects that remain unresolved at the end of test execution.
- Create a test summary report to be communicated to stakeholders.
- Finalize and archive the test environment, the test data, the test infrastructure and other testware for later reuse.
- Hand over the testware to the maintenance teams, other project teams, and/or other stakeholders who could benefit from its use.
- Analyze lessons learned from completed test activities to determine changes needed for future iterations, releases and projects (i.e. perform a retrospective).
- Use the information gathered to improve test process maturity, especially as an input to test planning for future projects.

The degree and extent to which test completion activities occur, and which specific test completion activities do occur, depends on various contextual issues relevant to the project, which are discussed further in Chapter 5.

1.4.3 Test work products

'Work products' is the generic name given to any form of documentation, informal communication or artefact that is used in testing (or indeed in development). When testing is more formal, the majority of work products may be written documentation; when testing is very informal, the corresponding work product may just be a scrap of paper, or a note in someone's mobile phone. 'Work product' is a general term covering any type of information needed to do the work (testing or development).

Test work products are created as part of the test process, and there is significant variation in the types of work products created, in the ways they are organized and managed, and in the names used for them. The work products described in this section are in the ISTQB Glossary of terms. More information can be found in ISO/IEC/IEEE 29119-3 [2013].

Test work products can be captured, stored and managed in configuration management tools, or possibly in test management tools or defect management tools.

Test planning work products

You will not be surprised to find that test planning work products include test plans. There may be different test plans for different test levels. The test plan typically includes information about the test basis, to which all the other work products will be related via traceability information (see Section 1.4.4). Test plans also include entry and exit criteria (also known as definition of ready and definition of done) for the testing within their scope – the exit criteria are used during test monitoring and control.

Beware of what people call a 'test plan'; we have seen this name applied to any kind of test document, including test case specifications and test execution schedules. A test plan is a planning document – it contains information about what is intended to happen in the future, and is similar to a project plan. It does not contain detail of test conditions, test cases or other aspects of testing.

The test plan needs to be understandable to those who need to know the information contained in it. The two-page cryptic diagram that was called a 'test plan' at one organization would not be the right sort of work product for other organizations.

Test plans can cover a whole project, or be specific to a test level or type of testing. Test plans are covered in more detail in Chapter 5, Section 5.2.

Test monitoring and control work products

The work products associated with test monitoring and control typically include different types of test reports. Test progress reports are produced on an ongoing and/or a regular basis to keep stakeholders updated about progress, on a weekly or monthly basis, for example. Test summary reports are produced at test completion milestones, as a way of summarizing the testing for a particular unit of work or test project. Any test report needs to include details relevant to its intended audience, the date of the report and the time period covered by the report. Test reports may include test execution results once those are available, in summary form (e.g. number of tests run, failed, passed or blocked and number of defects raised of different severity), and the status compared to the exit criteria or definition of done.

Test monitoring and control work products should address project management concerns such as budget and schedule, task completion, resource allocation, usage and effort. If action needs to be taken on the basis of information reported in a test report, those actions should be summarized in the next report to ensure that the desired effect of the action has been achieved.

These work products are further explained in Chapter 5, Section 5.3.

Test analysis work products

Test analysis work products include mainly test conditions, as this is the output of the test analysis activity. Each test condition is ideally traceable to the test basis (and vice versa). In exploratory testing, a test charter may be a test analysis work product. Defect reports about defects found in the test basis as a result of test analysis can also be considered a work product from test analysis.

Here is an example: The specification for an ordering system describes a sales discount feature when customers put in large orders. The test conditions might be to test all the discount values for various order values. The discount values would be coverage items – we want to make sure we have tested each of them at least once. The work product would be the list of test conditions, that is, the discount values.

Test conditions are further discussed in Chapter 4.

Test design work products

The main work products resulting from test design are test cases and sets of test cases that exercise the test conditions identified in test analysis.

Sometimes the test cases at this stage are still rather vague and high-level, that is, without concrete values for inputs and expected results. For example, a test case for the sales discount might be to set up four existing customers, one who orders only a small amount so does not qualify for a discount, and the other three who order enough to qualify for a discount at each of the three discount levels respectively.

Having the test cases at a high level means that we can use the same test case across multiple test cycles with different specific or concrete data. For example, one application may have discounts of 2%, 5% and 10%, and another may have discounts of 10%, 20% and 25%. Our high-level test case adequately documents the scope of the test even though the details will be different in each application. The test case is traceable to and from the test condition that it is derived from.

We have seen that high-level test cases can have advantages, but there are also some aspects that you need to be aware of with high-level test cases. For example, it may be difficult to reproduce the test exactly; different testers may use different test data, so the test case is not exactly repeatable. A high-level test case is not directly automatable; the tool needs exact instructions and specific data in order to execute the test. The skill and domain knowledge of the tester is also critical; a junior new-hire with no domain knowledge may struggle to know what they are supposed to be doing, unless they are well supported by more experienced testers. These are not insurmountable problems, but they do need to be considered.

Test design work products may also include test data, the design of the test environment and the identification of infrastructure and tools. The extent and way in which these are documented may vary significantly from project to project or from one company to another.

When deriving test cases from the test conditions, we may also find defects or improvements that we could make to the test conditions, so the test conditions themselves may be further refined during test design. In our sales discount example, in test analysis we identified the three discounts as test conditions, but in test design, by looking at the test cases, we identified the 'no discount' test condition – a discount of 0%.

Test cases are further discussed in Chapter 4.

Test implementation work products

Work products for test implementation include:

- test procedures and the sequencing of those procedures.
- test suites.
- a test execution schedule.

At this point, because we are preparing for test execution, we need to further refine any high-level test cases into low-level test cases that use concrete and specific data, both for test inputs and test data. In our loyalty discount example, we would now need to specify the details of our existing customers, decide on exactly how much each order will come to and calculate the final amount they would pay, including the discount. So, for example, Mrs Smith puts in an order for $50.01. Because her order is over $50, she gets a 10% discount, so she pays $45.01. We calculate the expected result for the test using a **test oracle** – the source of what the correct answer should be (in this case simple arithmetic). We would also need to set up Mrs Smith and other customers in the database as part of the preconditions of running the test, and this would be included in the test procedure.

Test oracle (oracle) A source to determine expected results to compare with the actual result of the system under test.

In exploratory testing, we may be creating work products for test design and test implementation while doing test execution; traceability may be more difficult in this case.

Test implementation may also create work products that will be used by tools, for example, test scripts for test execution tools, and sometimes work products are created *by* tools, such as a test execution schedule. Service virtualization may also create test implementation work products.

As in test design, we may further refine test conditions (and high-level test cases) during test implementation. For example, by deciding on the concrete values for our sales discount example, we realize that a test condition we omitted was to consider two different ways of clients paying between $45.01 and $50.00 (that is, with or without a discount). This may not be important to include in our tests, but it is an additional test condition.

Test execution work products

Work products for test execution include:

- documentation of the status of individual test cases or test procedures (e.g. ready to run, passed, failed, blocked, deliberately skipped, etc.)
- defect reports (see Chapter 5, Section 5.6)
- documentation about which test item(s), test object(s), test tools and testware were involved in the testing.

In test execution, we want to know what happened when the tests were run. We want to know about any problems encountered that may have blocked some tests running, for example if the network was down for a time. We want to know whether or not we were able to execute all of the tests that we planned to execute (which is not necessarily all of the tests that we have designed or implemented).

When test execution is finished (or is stopped), we should be able to report test results based on traceability, so that if a set of tests that were all related to the same requirement or user story failed, we will know about that. We also want to know which requirements have passed all of their planned tests, and any that have not yet been tested – that is, the tests are still waiting to be run. This is particularly important when test execution is stopped rather than completed.

Stakeholders are more likely to appreciate the implications of failing tests if they can be related to user stories or requirements, rather than just being told that a certain number of tests passed or failed. This is why traceability is important.

Test completion work products

The work products for test completion include the following:

- test summary reports
- action items for improvement of subsequent projects or iterations (e.g. following an Agile project retrospective)
- change requests or product backlog items
- finalized testware.