9 Testing

Despite this fact that testingphase is technically situated after the implementation phase, however, it practically starts during the earlier stages. Normally it starts during analysis, and remains as a continuous activity throughout the development process. Clearly, the emphasis would grow stronger when the development enters the testing phase and the nature of testing activities would change during the development course. Therefore, it is important to understand the testing process in its entirety and not to restrict it as an activity after building the system. Again, like other topics in this book, I assume that you have some backgrounds on the testing concepts and issues. Therefore, the discussion would be arranged around your final year project and the way that you can adapt the testing process in it.

9.1 Testing Process

Testing is a process; it means that it normally should receive some inputs, processes them, and provides some outputs. It includes several activities, which should be carried out in different stages of software development. However, as it was mentioned, the core activities would be carried out during the testing phase. Normally, several different specialties are needed for the testing process to be accomplished. But, again, we are not discussing the process outside of the context of the final year projects. Consequently, you are expected to play most of the roles that are required in this process.

In fact, you should design test cases, provide test scenarios, collect and arrange test data, perform the test, and document the result. These activities have been shown by using a UML activity diagram as you can find in Figure 9-1. If you compare this simplified testing process with similar testing process diagrams, regardless of the way that they are presented, you can realize that some detailed activities and loop-backs in the process have been removed. It does not mean that this has unreasonably diluted the testing process.

Again, I would like to emphasize that many parameters such as size, human resources, time, and the complexity level of the system affects the testing process. For example, “regression test” is an important concept in the testing process. Simply, regression test means that you have to repeat the tests that you have performed earlier if you change your system. This idea has been implemented in many testing tools, however, to my opinion, it is not necessary for the majority of final year projects.

As another example, we can talk about “automated testing”. Simply, it means that either you use automated tools to test your system, or you create necessary programs that performing the tests, automatically. Isautomating the testing process necessary in your final year project? Well, in most of the cases the answer would be “No”. However, you have to look at the context and the development environment. If the development tool that you have selected provides you with efficient testing tool, certainly, it would be a good idea to use it. However, you should be careful to not spend more than the scheduled time that your project allows you for testing just because the “automated testing” is a fashion that you should follow.

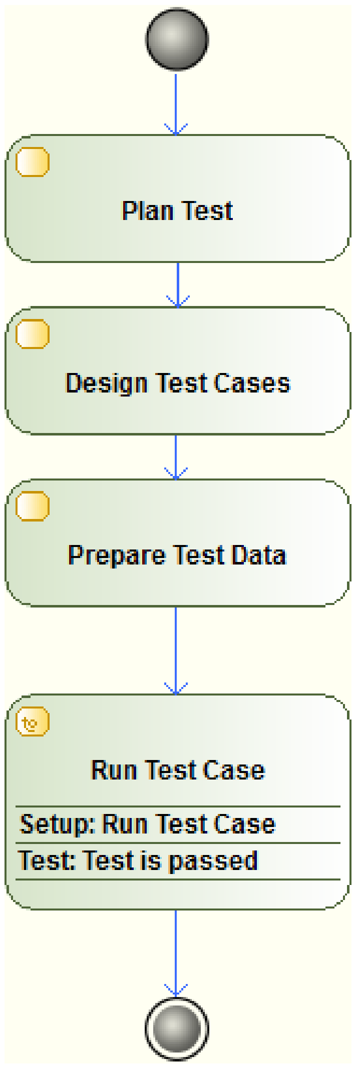

Figure 9-1 Testing Process Activity Diagram

Below we are going to discuss the test classification at the first step, and then we will discuss test cases and test data preparation. In addition, the test result documentation would be discussed.

9.2 Test Categories

Throughout the years, test has been classified into several categories. Below, thesecategories have been summarized in order to provide you with the general topics of testing.

Software Test Categories

Software testing can be categorized from different perspectives as below. However, these categories are neither complete in terms of its diversity nor rigid in terms of classification. They have been provided to let students have a general view on the testing categories.

- From system perspective:
  - System Test
  - User Test
    - Interface Test
    - Acceptance Test
- From software perspective:
  - Component Test
  - Unit Test
  - Integration Test
- From non-functional requirement perspective:
  - Performance Test
  - Security Test
  - Reliability Test
  - Scalability Test
- From test data perspective:
  - Conformance Test
  - Destructive Test
  - Non-destructive Test

Usually, you should not focus on all the above categories.In fact, unless specific requirements are asking you to perform other tests, considering some of them would properly suffice your testing process. These selective tests are Interface Test, Component Test, Unit Test, Integration Test, and Security Test. In addition, in some cases you may combine these tests together. For instance, you can embed the Interface Test within the Unit Test.

9.3 Test Case

A test case is a document that explains how a software product would be tested against a specific requirement in order to show that the requirement has been fulfilled by the system or if the system has failed to fulfill it. For large projects, you have to have formal documents for test cases. Different templates have been suggested by the experts in the field. In some contexts, you may find "Test Suite" as well. Test suite s is documents that usually include several related test cases. There is no need for you to worry about test suites in your final project. However, you can read more about test suites and test cases by consulting the related books in the bibliography section.

But, does it necessary for you to be so formal in your project? The blunt answer is "No". You have to be as formal as a project of the size of a final year project should be. Well, how much is this amount of formality? To illustrate, I suggest you to prepare two tables to document your test cases. The tables can be formed as you can see in Table 9-1 and Table 9-2. These tables can be prepared using a word processor or a spreadsheet. They help you to manage your test even if you ask a third person as a user to perform a user test or acceptance test, for example.

Test Cases					
ID	Test Case Name	Test Subject	Test Sequence	Test Data	Req/UC ID

Table 9-1 Test Case template

9.3.1 Test Scenario

Test scenario is a term that if you look it up in different texts, you would get different meanings, some of which is quite similar to what I said about the test case. To reduce the confusion, I would say the relation of a test case to a test scenario is similar to the relation of a use case to use case scenario. To simplify, test case explains the general information about how a test should be done, while, test scenario explains how you would apply those general explanations in specific context, and with specific data. To illustrate, a test casewhich aims to test a catalog information (a requirement with ID = 1 in the sample 1 of chapter 5), has been shown in Table 9-3. This test case can have several scenarios. For instance, the first scenario provides data to catalog a book, and the other one to catalog a journal, yet the third one tests recording a CD in the catalog. However, some experts may disagree with me on this issue.

9.3.2 Test Data

Test data is the data based on which you perform the test. Test should be prepared in way that covers different cases. You have to expect the behavior of the system based on different inputs. It means that you provide test data and you predict the system response if the mentioned data entered. This way you can test the system and see whether it is successful in performing its functions or fails to do it. Below you can find some hints on test data preparation. Clearly, this simplified guideline should be taken as a version that fit your final year project. In commercial projects, test data preparation is a highly professional activity and needs more effort to be accomplished. You can consult the bibliography section at the end of the book form more information on this issue.

Tips on Test Data

- Provide different test data for different test scenarios.
- If data is numeric, it must cover:
    - Lower bound (e.g. if this is a money attribute with 8 digits and no fractions it must take 0, 1, 2, for instance)
    - Middle (e.g. 55,870,20 and 11,659,450)
    - Upper bound (e.g. 88,999,999 and 99,999,999)
    - Negatives (e.g. -1)
- If data is alphanumeric, it must cover:
    - Maximum expected length
    - Mixed alphanumeric samples
    - Empty (blank)
- If data is an image, it must cover:
    - Images larger than the specified area
    - Images which exactly fit the are
    - Images which are too small to be used

9.3.3 Test Results

Table 9-1 documents the test cases. You should provide a unique ID by which you can identify the test case throughout your project and its related documents. You should give a name, which makes it easier to understand the test case. In addition, you should specify your test subject by specifying the subject type, i.e. screen, module, subsystem, system and the name/address of the test subject. In the Test Sequence field, you should explain the sequence through which the test should be performed. It is a good practice to have this sequence in a bulleted format to allow yourself or a tester to follow it in a stepwise manner. In the Test Data field, you can provide the data that you intend to use for the test activity. If the place was not spacious enough to hold what you want to document, then, you can put the address of related document(s) that you used for this purpose, instead. However, in most of the cases they are sufficient for their purposes. The Requirement/Use Case ID is the same ID that you have used in the Requirement Specification document in Table 5-1and Table 5-2. To make a cross-reference and trace test cases back to the requirements.

Test Results			
Test Case ID	Attempt	Scenario	Result
			Pass/Fail – Comments

Table 9-2 Test Result template

Table 9-2 documents the test results. The ID is the same ID that you have used in Table 9-1 to make a cross-reference between the results and the test cases. The Attempt refers to the repetition attempts in case a test case has failed and the activity has repeated. In this situation, you would have more than one row for a specific ID, clearly, with different results. The Result field holds the result as Pass or Fail. In the latter case, you can explain the result or attach a screen capture image by providing a link to the image file, if you prefer to do so.

Table 9-3 shows an example based on the samples that were used in the previous chapters

Test Cases					
ID	Test Case Name	Test Subject	Test Sequence	Test Data	Req/ UC ID
1	Test Catalog	Screen	Click on "New Catalog" button. Enter all required data. Click on "Save" button. Click on "Yes" button of the message box.	**First scenario**: Material Type: Book Author: Pressman, R.S. Title: Software Engineering: A Practitioner's Approach Year: 2010 Edition: 7th Publisher: McGraw-Hill ISBN-10: 0072496681 ISBN-13: 978-0073655789 Number of Copies: 20 Classification: Dewey ***Expected Result:*** Material Saved **Second scenario**: Same data except for the following part: ISBN-10: 000123 ***Expected Result:*** System gives a warning that the ISBN-10 format is not correct.	1

Table 9-3 Test Cases sample

If you look at the test data column of Table 9-3, you can see that the data has been organized under different scenarios. Two scenarios were given as an example. After conducting the test, the result has been documented as you can see in Table 9-4.

Test Results			
Test Case ID	Attempt	Scenario	Result
1	1	1	Pass
1	1	2	Fail – System crash – See screen shot at this address

Table 9-4 Test Results Sample

9.4 Object Oriented Test

Although the foundations of software testing process are independent of the software implementation, but the techniques used in different development paradigms are different. This difference becomes significant in some cases such as the difference between testing of traditionally implemented software versus object-oriented software. Detailed discussion on this issue is beyond the scope of this book and you can consult the related books in the bibliography section. However, I advise you to consider the followings as the minimum testing objectives if you are going to use object-oriented method.

9.4.1 Class Test

Prepare test data to test your implemented classes. Focus on business layer, but do not forget to test your data layer and presentation layer classes as well.

9.4.2 Package Test

If you have combined your classes into packages that participate in the system as a subsystem, prepare test cases that test the input/output of these packages and their communication with other parts of system.

9.4.3 Interface Test

By "interface", here, I mean Interface classes, which you have studied in your object-oriented programming modules/courses. You should be careful about these interfaces and the way that they perform their functions. You should prepare test cases that checks these interface and not only make you sure that they do what they are expected to do, but also they do not do anything else which leads to the breaching of object-oriented rules.

9.5 Validation and Verification

Validation and Verification (V&V) is a concept in software testing and quality control. By validation, you have to answer this question, "Are you developing the right system?" whereas by verification, you have to answer this question, "Are you developing the system right?" As you can realize, the first question is validates the system, which you are doing it through the testing process. But, the second question is about the process that you have followed to develop the system. Although you should answer this question and you should be able to show evidence of the positive answer to this question, however, this is your supervisor that can help you through her/his feedbacks by letting you know that you are on the right track. Furthermore, your supervisor or evaluation panel would judge on this issue when you deliver your final project. You can find more on this issue by consulting bibliography section of this book.

Finally, I did not discuss the debugging process in this book as I assume that you have received enough lessons about it in different programming modules/courses, and practiced it during laboratory exercises. However, although debugging is considered as a test activity in many contexts and it is specially categorized under the validation process, it should not be confused with the testing process. To my view, debugging is part of programming. It is so interconnected to the programming tasks that you cannot assume at as a separate task. On the other hand, as it was mentioned, test is a planned process that might be performed independent of programming.

9.6 Summary

Evaluation and testing is an unavoidable part of any software and information technology projects. You might have studied different topics on testing through modules/courses such as programming, software engineering, project management and such. Although, you might have received a great deal on the importance of testing, however, it is quite normal that you might have a tendency to underestimate it.

Despite my suggestion on the simplification of software development process in your final year project, I would like you to pay as much attention as you can to the testing process. This is what that gives your project a strong flavor of both engineering and scientific approach. Although general guidelines of testing, verification and validation process exist that you can follow, however, the testing could differ according to the methodology that you have chosen to follow. For example, if you have decided to use object-oriented programming you should either know or learn about this method's specific testing techniques.

Finally, you should appreciate the role of use cases, especially if they are well prepared, in designing test scenarios. This is one of the best practices that I have found during developing several projects. I have realized that how useful the use cases can be in the design of test scenarios and in the process of conformance test and user test. Therefore, my advice is to utilize the use cases, which you have documented in the analysis phase, in the testing stage. You can find more on this in the related subjects in the bibliography section at the end of this book.