ST笔记

Software Testing Technique

1. Overview

Software testing is an empirical, technical investigation conducted to provide stakeholders with information about the quality of the product or service under test, with different approaches. It is one aspect of software quality. As a separate phase in software development, it is typically performed by quality assurance staff or a developer other than the one who wrote the code.

1.1 Purpose

1. Detect software failures so that defects may be discovered and corrected
2. Evaluate whether software functionality and quality achieve expected results

1.2 Faults and Failures

Software faults occur through the following process: A programmer makes an ***error (mistake)***, which results in a ***fault (defect, bug)*** in the software source code. If this fault is executed, in certain situations the system will produce wrong results, causing a ***failure***.

2. Testing Approach

2.1 Black-box testing

Black-box testing is a method of software testing that examines the functionality of an application without peering into its internal structures or workings.

2.1.1 Equivalence partitioning

Equivalence partitioning or equivalence class partitioning (ECP) is a software testing technique that divides the input data of a software unit into partitions of equivalent data. In principle, test cases are designed to cover each partition at least once.

For example:

function getMonth(int m){...}

In this function, valid values for m are integers 1-12, we can partition the equivalence class like this:

```
       ... -2 -1  0 1 ..... 6 7 ..... 12 13  14  15 .....
    ---------------|---------|----------|--------------------
invalid partition 1      P1         P2     invalid partition 2
                       valid partitions
```

From this, we can get four test cases, and the m values are:

No.	Test case	Expected results
1	m=-2	“wrong input”
2	m=4	“April”
3	m=8	“August”
4	m=15	“wrong input”

2.1.2 Decision table testing

Decision tables are a concise visual representation for specifying which actions to perform depending on given conditions.

For example:

Printer troubleshooter

		Rules							
Conditions	Printer prints	No	No	No	No	Yes	Yes	Yes	Yes
	A red light is flashing	Yes	Yes	No	No	Yes	Yes	No	No
	Printer is recognized by computer	No	Yes	No	Yes	No	Yes	No	Yes
Actions	Check the power cable			✓					–
	Check the printer-computer cable	✓		✓					–
	Ensure printer software is installed	✓		✓		✓		✓	–
	Check/replace ink	✓	✓				✓		–
	Check for paper jam		✓		✓				–

2.1.3 Use case testing

The main content of use case testing is to describe an event flow in a certain scenario.

For example:

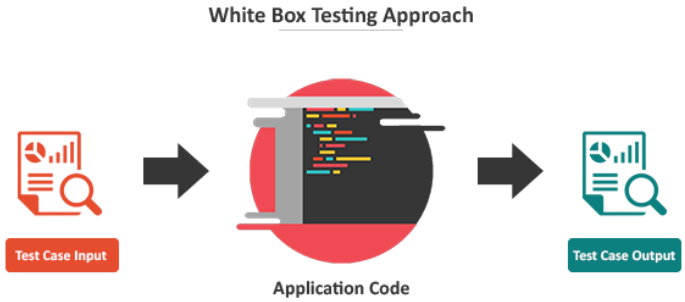
No.	Scenario	Action	Condition	Expected results
1	Bank card inserted into ATM	Withdraw 100 yuan	Balance greater than 100	Successful
2			Balance is less than 100	Prompt ’Insufficient balance’
3	Bank card inserted into ATM	Deposit 100 yuan	–	Successful

2.1.4 Test Case

Module	Test Case ID	Priority	Test case Summary	Pre-requisites	Test steps	Test data/Condition	Expected results	Actual results	Pass/Fail	Author&Remarks
Login	1	middle	Username check	User enters login page	input user name	"#testN123"	Prompt username is invalid.	No results	Fail	Author: zy
	2					"testN123"	No results (username is valid).	No results	Pass	
	...							...	...	...
	4	high	Login function check	A valid username and password has been entered	click the login button	Username and password are correct	Switch the page to the system home page	Prompt to log in successfully but the page does not automatically switch	Fail	Author: zy Remarks: The bug here may be that username and password checks are not done
	5	high	Login function check	A valid username and password has been entered	click the login button	Username is correct, password is incorrect	Prompt login failed, wrong username or password	Prompt to log in successfully but the page does not automatically switch	Fail	
Upload	6	middle	upload excel file	Entered the upload page	Click the upload button and select the excel file	The size of file is less than 50M	Prompt that the upload was successful and the current file is displayed	Prompt that the upload was successful and the current file is displayed	Pass	Author: zy
	7					The size of image is larger than 50M	Prompt upload failed, the size of file cannot be larger than 50M			

2.2 White-box testing

White-box testing verifies the internal structures or workings of a program, as opposed to the functionality exposed to the end-user. While white-box testing usually be applied at the unit and integration levels of the software testing process.



2.2.1 Code coverage

Creating tests to satisfy some criteria of code coverage (e.g., the test designer can create tests to cause all statements in the program to be executed at least once).
There are a number of coverage criteria, but the main ones are:

- Function coverage — has each function in the program been called?
- Statement coverage — has each statement in the program been executed?
- Edge coverage — has every edge in the control-flow graph been executed?
- Branch coverage — has each branch of each control structure been executed?
- Condition coverage — has each Boolean sub-expression evaluated both to true and false?

2.2.2 API testing

API testing is a type of software testing that involves testing application programming interfaces (APIs) directly and as part of integration testing to determine if they meet expectations for functionality, reliability, performance, and security.
API testing commonly includes testing REST APIs with JSON or XML message payloads being sent over HTTP and HTTPS.

3. Testing Levels

3.1 Unit testing

Unit testing refers to tests that verify the functionality of a specific section of code, usually at the function level. In an object-oriented environment, this is usually at the class level, and the minimal unit tests include the constructors and destructors.
These types of tests are usually written by developers as they work on code (white-box style), to ensure that the specific function is working as expected.

3.2 Integration testing

Integration testing is any type of software testing that seeks to verify the interfaces between components against a software design. It works to expose defects in the interfaces and interaction between integrated components (modules).

3.3 System testing

System testing tests a completely integrated system to verify that the system meets its requirements. For example, a system test might involve testing a login interface, then creating and editing an entry, plus sending or printing results, followed by summary processing or deletion (or archiving) of entries, then logoff.

3.4 Acceptance testing

Acceptance testing commonly includes the following types:

- User acceptance testing (UAT)
- Operational acceptance testing (OAT)
- Contractual and regulatory acceptance testing

References:

思维导图
Software Testing – Wikipedia