

In this chapter you will learn to:

- differentiate between systems and program test data
- test your solution with the test data created at the design stage, comparing actual output with that expected
- demonstrate the features of a new system to users, facilitating open discussion and evaluation

Which will make you more able to:

- identify and evaluate legal, social and ethical issues in a number of contexts
- construct software solutions that address legal, social and ethical issues
- apply appropriate development methods to solve software problems
- apply a modular approach to implement well-structured software solutions and evaluate their effectiveness
- apply project management techniques to maximise the productivity of the software development
- create and justify the need for the various types of documentation required for a software solution
- select and apply appropriate software to facilitate the design and development of software solutions
- assess the relationship between the roles of people involved in the software development cycle
- communicate the processes involved in a software solution to an inexperienced user
- use a collaborative approach during the software development cycle
- develop effective user interfaces, in consultation with appropriate people.

In this chapter you will learn about:**Testing the software solution**

- comparison of the solution with the original design specifications
- generating relevant test data for complex solutions
- levels of testing
 - unit or module
 - program
 - system
- the use of live test data to test the complete solution
 - larger file sizes
 - mix of transaction types
 - response times
 - volume data
 - interfaces between modules
 - comparison with program test data
- benchmarking
- quality assurance

Reporting on the testing process

- documentation of the test data and output produced
 - use of CASE tools
- communication with those for whom the solution has been developed, including:
 - test results
 - comparison with the original design specifications

TESTING AND EVALUATION OF SOFTWARE SOLUTIONS

Testing and evaluation is integral to all stages of the software development cycle. In this chapter, we focus on higher levels of testing that take place after the project has been implemented in a programming language. Personnel within the software development process perform alpha testing with real data. Beta testing occurs when the product is distributed for use to a limited number of outside users. These users are engaged to report any faults or recommendations back to the software development company. In this chapter, we are concerned with both alpha and beta testing.



Alpha testing

Testing of the final solution by personnel within the software development company prior to the product's release.

The testing and evaluation process is central to a software development company's quality assurance. Essentially, we are ensuring the product meets the original objectives and design specifications that were created during the 'Defining and Understanding the Problem' stage. Another purpose of testing is to evaluate the product's performance against recognised industry standards or benchmarks.



Beta testing

Testing of the final solution by a limited number of users outside the software development company using real world data and conditions.

We discuss levels of testing, from unit or module level testing, through program testing and finally system testing. The generation of test data for specific purposes is discussed. Live test data is produced to test large file sizes, different transaction types, response times, volume data and also, of course, to ensure processing is correct. We examine CASE tools used to automate and report on the testing process. The results of the testing process are used to provide feedback so problems encountered can be corrected before the product is finally released.

We have already discussed many methods of testing in chapters 3, 4 and 5. Feasibility studies are used to test and ensure that the objectives are achievable. Desk checks ensure algorithms operate as intended. Many of the debugging tools and techniques discussed in chapter 5 are used to test the code for errors.

COMPARISON OF THE SOLUTION WITH THE ORIGINAL OBJECTIVES AND DESIGN SPECIFICATIONS

The original objectives, which may have been modified during the development of the product, must be tested for compliance. The aim of the project is to implement and solve each objective. Because the objectives are written in such a way that they can be readily tested, a pass or fail can be allocated to each objective. A checklist is usually created listing all the objectives. Each objective is tested in turn. If faults are encountered then that aspect of the product is altered until the objective is realised.

Design specifications were developed as a framework for the development process. Each of these specifications needs to be tested against the final product. Many of the design specifications will involve internal issues; the results of the testing process will uncover inconsistencies in approach that have occurred during development. These inconsistencies must be identified and rectified both in the existing product and for future products.

HOW DO WE TEST THE OBJECTIVES AND DESIGN SPECIFICATIONS?

The remainder of this chapter is designed to address this central question. During the testing and evaluation stage we aim to identify any errors, omissions and other problems within the product. Each problem identified will result in further development to correct the problem.

There are essentially two techniques used to test software solutions: white box testing and black box testing. Both techniques are required to identify and correct the majority of errors. Black box testing does not attempt to identify the source of the problem, but rather to identify that a problem exists. Once the problem has been identified, it may be necessary to move to white box testing to find and correct the problem. At this testing and evaluation stage, we are primarily concerned with black box testing techniques. The testing techniques used in previous chapters were white box testing techniques that aimed to identify and to correct problems emerging during the product's development.



Black box testing

Also known as functional testing. The inputs and expected outputs are known; the processes occurring are unknown.



White box testing

Also known as structural, or open box testing. A software testing technique whereby explicit knowledge of the internal workings of the item being tested is used.



GROUP TASK Discussion

Compile a list of testing techniques from both the 'Planning and Design' and 'Implementation' stages of the Software Development Cycle. Explain why these techniques can be viewed as white box testing techniques.

LEVELS OF TESTING

Software products can be tested at various levels. Each individual module within a program is tested as it is created. Each program will be tested to ensure modules work together correctly. We have examined both these levels of testing as part of the Implementation stage in chapter 5 (see section on Program Development Techniques). During the implementation stage, we were interested in the operation and processing of the actual programming code for the particular product. We now need to consider broader, system-level testing.

WHAT IS SYSTEM-LEVEL TESTING?

System-level testing aims to ensure that the hardware, software, data, personnel and procedures that form the components of the final system are able to work together efficiently, correctly and in the manner intended with the new software product. Not only does the software need to operate correctly, it must operate correctly within the various system environments in which it will be implemented.

To achieve this aim requires the use of real test environments utilising live test data. Let us examine each of the system components in terms of creating suitable test environments.

Hardware

The software product should be installed and tested on different combinations of hardware ranging from those specified as the minimum requirements to those that include any additional hardware devices.

Large systems are often designed for particular hardware configurations. This allows the product to be optimised for these specific machines. In this case, it is possible to test all possible hardware configurations. When testing products designed for a broad hardware base, it is common to encounter problems that are beyond the control of the software developer. For example, there are an almost countless number of different model printers available. It is not possible for software developers to guarantee their product will work correctly with all possible printers. Even if testing all printers were possible, any errors encountered are beyond the boundary of software products, and hence the software developer is often unable to solve the problem.

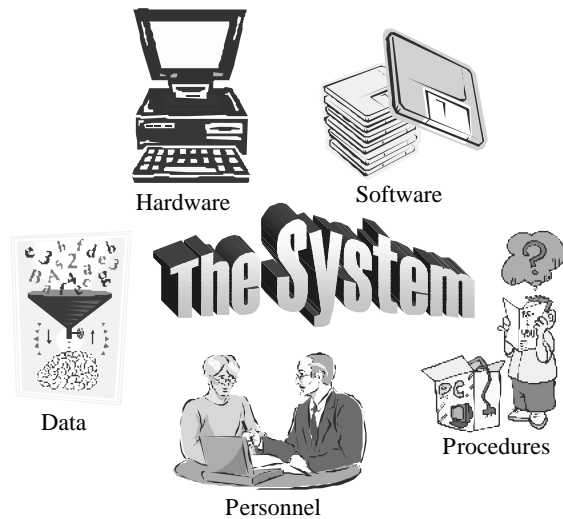


Fig 6.1

Testing aims to ensure all components of the system work with the new software product.



GROUP TASK Discussion

The office area of your school contains hardware, software, data, personnel and procedures. Describe the components of your school's office in terms of these five components.

Software

Other software installed on the system will have an effect on the operation of new software products. This is particularly the case when COTS products have been used during development. Different versions of the base product can cause incompatibilities with the new product. Similar problems can occur with different versions of the intended operating system. It is important to test the new product with various combinations of software installed.

New products that are designed to interface with existing products will require special attention. Often data is passed between applications and across networks. The only way to confidently test the interfaces between these applications is using live test data. Internet applications also require special attention, as they often need to operate across a broad range of operating systems using various software browsers.

Data

The data input into the system is the most likely source of errors. The product needs to be tested with a large combination and quantity of live test data to ensure its reliability under live conditions. The section that follows deals with types of live test data necessary to ensure the robustness of the product when confronted with various live data scenarios.

Personnel and Procedures

The users of the system will have various skill and knowledge levels. Does the product cater effectively to the needs of users of the product? Particular attention needs to be given to the evaluation of the user interface. Beta testing is one of the most effective methods of testing system objectives related to the personnel who will use the system. It is vital to choose a variety of different personnel to participate in the beta testing program.

The introduction of most new software products will involve changes in the way tasks are completed. Beta testing can be used to analyse the ability of personnel to efficiently implement any new procedures and to modify old procedures. Normally users are asked to comment on the effectiveness of any new procedures resulting from the introduction of the new software product.



Consider the following:

A new website has been developed by an office supply company. The company traditionally sells most of its goods by mail order. The new website allows purchases to be made using the website. The product is now at the beta testing stage.



GROUP TASK Discussion

Describe the type of businesses the office supply company may choose to be part of their beta testing program. Compile a questionnaire that could be used to evaluate the success of the new website in terms of issues related to personnel and procedures.

USE OF LIVE TEST DATA FOR SYSTEM TESTING

When a system is finally installed and implemented within the total system it is said to be 'live'. Once a product goes live it needs to undergo a series of tests to ensure its robustness under real conditions. This testing occurs using live test data.

Live test data is produced to simulate extreme conditions that could conceivably occur. The use of live test data aims to ensure the software product will achieve its objectives under these extreme conditions. Different sets of live test data are used to test particular scenarios. CASE tools are available to assist in the task of producing live test data sets and also to automate the testing process. Later in this chapter, we examine CASE tools used for this purpose.

For most products live test data should be created to test each of the following conditions:

- Large file sizes
- Mix of transaction types
- Response times
- Volume data
- Interfaces between modules and programs
- Comparison with program test data

LARGE FILE SIZES

Many commercial applications obtain input from large databases. During the development of software products, relatively small data sets are used. At the alpha and beta testing stages large files should be used to test the system's performance.

The use of large files will highlight problems associated with data access. Often systems that perform at acceptable speeds with small data files become unacceptably slow when large files are accessed. This is particularly the case when data is accessed via networks.

Large data files highlight aspects of the code that are inefficient or require extensive processing. Many large systems will postpone intensive processing activities until times when system resources are available. For example, updating of bank account transactions takes place during the night when processing resources are available.



Consider the following:

A rental car company is implementing a new computer system across its national network of 1200 branches. The branches are linked via a network of computers. Because of the remoteness of many branches, communication links are unreliable hence data must be held locally at each branch. Because cars can be returned to any branch, each branch requires a record of all rentals from all branches.

A system known as data replication is used. Essentially, every branch has a copy of the entire database. Changes to a particular record at one branch are duplicated across all branches every 10 minutes. The rental car company has a fleet of some 7000 vehicles with approximately 4000 new rental transactions occurring each day.

The software company developing this product wishes to create a large data file to use during alpha testing. Initially they create a file containing 400,000 transactions. The data replication is found to take on average 5 seconds when using this file.



GROUP TASK Discussion

Each rental transaction is held for five years. How many transactions should be in the test file to ensure performance remains satisfactory?



GROUP TASK Discussion

What types of problems are likely to be highlighted as a result of testing the product with such a large file? Explain.

MIX OF TRANSACTION TYPES

Testing needs to include random mixes of transaction types. Module and program testing usually involve testing specific transactions or processes one at a time. During system testing we need to test that transactions occurring in random order do not create problems.

Often the results of one process will influence a number of other processes. If a transaction is currently being completed on specific data and that data is altered by another transaction then problems can occur. When a number of applications are used to access the same database then conflicts are inevitable. Software must include mechanisms to deal with such eventualities.



Consider the following:

Banking systems involve a multitude of applications that access individual account details. Some accounts may experience many thousands of transactions per hour. This is particularly the case with public bodies such as the tax office and social security. Deposits and withdrawals are taking place at an enormous pace. These transactions originate from a large variety of sources: ATMs, bank branches, post offices, the Internet, etc.

New applications that interface with existing banking systems need to be able to deal with this situation. Transactions must be absolutely secure, given the nature of the application.



GROUP TASK Discussion

Describe possible scenarios that could result in problems when multiple transactions attempt to access a single account's details.



GROUP TASK Discussion

What type of live test data could be used to test the correctness of these banking transactions?

RESPONSE TIMES

The response time is the time taken for a process to complete. The process could be user activated or it could be activated internally by the application. Response times are dependent on all the system components, together with their interactions with each other and other processes that may be occurring concurrently.

Any processes that are likely to take more than one second should provide feedback to the user. Progress bars are a common way of providing this feedback. Data entry forms that need to validate data before continuing should be able to do so in less than one second; 0.1 second is preferable. Studies have shown that users will tolerate delays of up to one second, however their concentration will deteriorate if the delays are frequent. Response times of around 0.1 second give the user the impression of instantaneous response; this is of course the ideal situation.

Response times should be tested on minimum hardware using typical data of different types. Any applications that affect and/or interface with the new product should be operating under normal or heavy loads when the testing takes place.



Consider the following:

On the World Wide Web, response time rules! Sites that include animations and complex graphic advertisements often reduce the effectiveness of the site by increasing response times. The time taken to download these graphics can often cause prospective customers to give up and go to a faster competitor's site. There is money to be made by closely examining the response times for each transaction on a new website.

Companies are emerging that will analyse the response time for different aspects of websites. Their findings can result in significant savings for their clients. One company was founded by an individual who advertised his business with the slogan 'I'll tell you why your website is rubbish'. His business was based on response time analysis.



GROUP TASK Discussion

Explain why response times on websites are so crucial to the success of these sites.



GROUP TASK Discussion

Describe the nature of tests that could be implemented to test the response times for websites. Justify your answers.

VOLUME DATA

Large amounts of data should be entered into the new system to test the application under extreme conditions. Multi-user products should be tested with large numbers of users entering and processing data simultaneously. Large systems that require extensive data input require special consideration.

Alpha testing by the software developer must try to simulate the number of users who will simultaneously use the product. This can be done using a laboratory of computers and users in conjunction with CASE tools. In many cases, it is difficult to simulate real volumes of data in a beta testing environment without actually implementing the system.

CASE tools are available that enable automatic completion of data entry forms. Many of these tools allow the creation of virtual users. This allows one machine to simulate the activities of hundreds or even thousands of simultaneous users entering data. We examine CASE tools in detail in the next section.



Consider the following:

The post office is implementing a new application in its existing computer system. This application allows businesses to purchase and print stamps via the Internet. It is envisaged that some 20,000 businesses will take advantage of this initiative within one year of its introduction. Within five years, some 150,000 businesses will be using the service.

The software has been written and is about to undergo volume testing. Obviously it is not possible to beta test the product with 150,000 users, so a simulation using CASE tools is used.



GROUP TASK Discussion

The volume of data to be processed is crucial to the success of this product. How can volume testing be undertaken when the customer base does not at present exist?



GROUP TASK Discussion

The security of these transactions is crucial. Discuss methods that could be used to test the security of the system, both in terms of payment for stamps and also in terms of ensuring printed stamps are not counterfeit.

INTERFACES BETWEEN MODULES AND PROGRAMS

An interface provides a communication link between system components. In terms of modules within a program, the interface is usually provided through the use of parameters that are used to pass data to and from modules. The screens of an application provide an interface between the users and the program. Hardware drivers are programs that provide an interface between applications and hardware devices. Programs that interact with other programs require an interface to complete their tasks. All these interfaces require testing.

In this section, we are interested in testing the performance of interfaces that connect modules and programs. Tests and test data need to be created to analyse the interaction between different software components of the system. In many large systems the interface between programs will include a communication link, this communication link being an integral part of the interface. Tests should examine the accuracy of data being passed as well as response times and the ability of the interface to cope with large files, different transaction types and large volumes of data.



Consider the following:

An upgrade of the software that operates a particular type of Automatic Teller Machine (ATM) is being alpha tested. The system flowchart in *Fig 6.2* describes the programs with which this application interfaces.

The ATM application must wait for authorisation from the main bank application. The bank application sends an enquiry to either its own database or to another financial institution. It must then await a response. The other financial institution must then perform an enquiry and wait for a response before sending the result to the originating bank. The originating bank then sends a response to the ATM application.

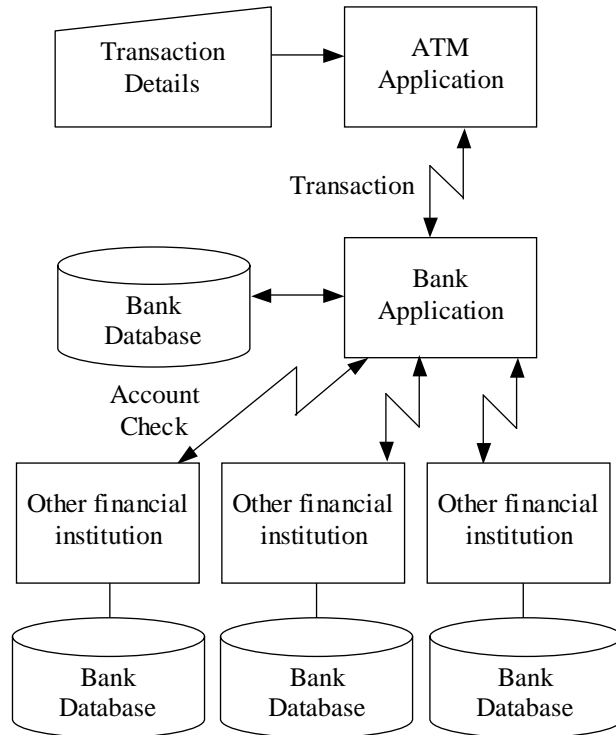


Fig 6.2
System flowchart for ATM interfaces.

Tests designed for testing this system must examine any eventuality. Perhaps a data link is down, or an application is slow to respond. The test data must ensure that the ATM application responds appropriately to all occurrences.



GROUP TASK Discussion

Develop a list of possible scenarios that must be tested to ensure the correct operation of the ATM application in terms of interfaces to other applications. For each scenario describe the type of test and test data that could be used to ensure correct operation of the ATM.

COMPARISON WITH PROGRAM TEST DATA

During previous stages of the software development cycle, test data together with expected outputs were created. This test data should be entered into the finished product to ensure correct outputs are obtained. Because the outputs are already known, this testing process ensures the final product is able to perform its processing correctly.



GROUP TASK Discussion

Program test data has already been entered and the results analysed during earlier stages of the software's development. Why is it necessary to input and check the results again? Discuss.

BENCHMARKING

Benchmarking is the process of evaluating a product, in this case a software product, using a series of tests. These tests provide numerical results that are compared with recognised standards. The process of benchmarking aims to compare the performance of a product with these standards. Results of benchmark tests are used to compare products. These results allow users of software products to make informed purchasing decisions. They also provide software developers with objective data, which can be used to compare their products with those of their competitors.



Benchmark

A point of reference from which quality or excellence is measured.

There are many variables that affect the performance of a software product. A particular product may outperform its competitor on a particular hardware configuration, however the results may be reversed on a different configuration. Benchmarking aims to reduce the number of variables and to report results objectively.

Successfully completing benchmark tests requires copies of competitors' products. Both products are subjected to a series of tests using identical hardware configurations and test data sets. The results are used to evaluate the performance of the product compared to its competitors. Benchmarking a product is vital to maintaining and improving a product's market penetration.

Although benchmarking is often undertaken internally by software development companies without the cooperation of their competitors, this is not always the case. It is often in the interest of both companies to work together on benchmarking activities. Cooperation between software companies can result in mutual gains and an increase in quality and performance across the industry.

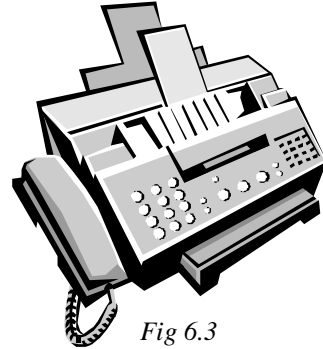
Companies specialising in benchmarking are emerging. These companies provide benchmarking software that can be utilised to test specific aspects of computer systems. Industry associations, as a result of their unbiased nature, often create and distribute benchmarking products. Most of these products test the performance of specific hardware functions when subjected to particular software commands: for example, graphics rendering, file input and output, or floating-point calculations. Results from these benchmarking tests can be used to optimise software products for particular system configurations.



Consider the following:

A software development company has developed a new product to allow businesses to fax directly from their computer to multiple recipients. The details of fax recipients can be extracted from a number of common database formats.

There are two other products on the market that perform similar functions. The software development company wishes to evaluate their new product by comparing it with their competitors' products.



*Fig 6.3
Software products are available to allow faxes to be sent directly from a computer.*



GROUP TASK Discussion

Make up a list of tests that could be employed to assist in the benchmarking process. Explain why each test has been included.



GROUP TASK Discussion

The system on which the tests take place is vital to obtaining reliable results. What criteria could be used to assist in the selection of appropriate system components for these tests? Discuss.

QUALITY ASSURANCE

The quality of a software product is measured against how well the product meets or exceeds users' expectations. Quality assurance is about ensuring that this occurs. In chapter 1, we examined aspects of quality software. Each of these aspects requires testing and evaluation. The following factors and related questions are reproduced from chapter 1:

- Correctness: Does it do what it is supposed to do?
- Reliability: Does it do it all of the time?
- Efficiency: Does it do it the best way possible?
- Integrity: Is it secure?
- Useability: Is it designed for the end user?
- Maintainability: Can it be understood?
- Flexibility: Can it be modified?
- Testability: Can it be tested?
- Portability: Will it work with other hardware?
- Re-useability: Can parts of it be used in another project?
- Interoperability: Will it work with other software?

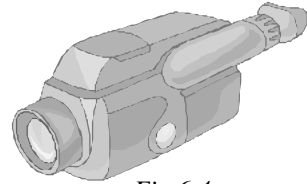
Quality assurance is not just about testing a product once it has been completed. Quality assurance techniques should be implemented throughout the software development process as well as throughout the processes of the software development company. National and international standards exist for quality assurance. The International Standards Organisation (ISO) develops standards for quality assurance. Companies and products can adopt these standards. Organisations throughout the world provide certification services to businesses wishing to participate. In Australia, Quality Assurance Services (QAS) is one of the largest suppliers of such services. QAS provides initial certification services, product testing, and certification, together with auditing and education services. Products and companies are able to use standards logos on their advertising and product packaging.



Consider the following:

A software quality assurance company is employed to test a software application that has been written to control the operation of a digital video camera. The software company is particularly interested in results pertaining to correctness, reliability, efficiency and useability.

The camera performs all the usual video camera functions together with some new and innovative functions. For example, the camera can be set to commence filming once it detects movement. The camera will then follow the source of the movement, maintaining the subject within each frame. This function is particularly useful for wildlife photographers who traditionally spent hours waiting to capture particular wildlife footage.



*Fig 6.4
Digital video cameras are
controlled by software
applications.*



GROUP TASK Discussion

Describe the types of tests that could be conducted by the quality assurance company to determine the correctness, reliability, efficiency and useability of this product.



GROUP TASK Discussion

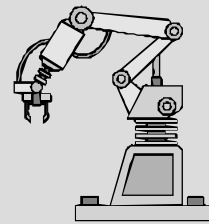
Results of quality assurance are often used to assist in the marketing of products. How could the results, assuming they are positive, be used to market this new video camera?

SET 6A

1. Internal system testing using live test data can be best described as:
 - (A) white box testing.
 - (B) black box testing.
 - (C) alpha testing.
 - (D) beta testing.
2. The main purpose of the testing and evaluation stage of the software development cycle is to:
 - (A) discover bugs in the source code, at both the module and program level.
 - (B) ensure the program operates correctly with live test data.
 - (C) train users in the operation of the new product.
 - (D) ensure the product achieves its objectives and design specifications.
3. Explicit knowledge of the internal workings of an item is utilised when performing:
 - (A) white box testing.
 - (B) black box testing.
 - (C) alpha testing.
 - (D) beta testing.
4. System-level testing should include all components of the system. This includes:
 - (A) modules, programs and systems.
 - (B) hardware, software, data, personnel and procedures.
 - (C) large file sizes, mix of transaction types, response times, and volume data.
 - (D) benchmarking and quality assurance.
5. Benchmarking is:
 - (A) used to ensure the product is of a high quality.
 - (B) a technique used to isolate and report weaknesses in the product.
 - (C) used to compare a product to other, similar products using a standard point of reference.
 - (D) intended to gauge the extent to which a product has realised its goals and objectives.
6. Users testing a software product before final release are participating in:
 - (A) white box testing.
 - (B) black box testing.
 - (C) alpha testing.
 - (D) beta testing.
7. The inputs and expected outputs are known, however the details of the processes that transform inputs into outputs are unknown during:
 - (A) white box testing.
 - (B) black box testing.
 - (C) alpha testing.
 - (D) beta testing.
8. Large files are likely to highlight problems in regard to:
 - (A) errors in the logic of the source code.
 - (B) inefficient or processor intensive operations.
 - (C) the operation of the user interface.
 - (D) All of the above.
9. Tests in regard to response times are formulated to primarily test the:
 - (A) operation of the user interface.
 - (B) hardware components of the system.
 - (C) procedures required when operating the new product.
 - (D) product compared to competitors' products.
10. Which of the following is NOT true in regard to Quality Assurance (QA)?
 - (A) QA practices should be implemented throughout the development process.
 - (B) QA is primarily about ensuring customers' expectations are realised in the final product.
 - (C) QA ensures testing of software products results in error-free applications.
 - (D) QA standards are available for implementation by businesses. Certification allows use of standards logos.

11. Black box testing is the major technique used during the testing and evaluation stage of the software development cycle. Why is this the case, and why is black box testing an appropriate technique? Justify your answer.
12. Live test data is used to test a number of aspects of the total system. Describe aspects of the system that are best tested using live test data. Why can't program test data be used to achieve the same result? Discuss with the use of examples.
13. Benchmarking a product against its competitors is a common technique. Software companies, media groups and industry organisations all complete benchmark testing. Describe the process of benchmarking. How can benchmarking assist in raising the standard and quality of software products?

A software product has been developed to control a robotic production line. The production line manufactures plastic automotive components. These components must be produced within particular size tolerances. The robots perform both the manufacturing and the measuring of each component prior to its acceptance.



Details of the plans for each component are obtained directly from a CAD (Computer Aided Design) file. Size tolerances are input into the new system via a graphical user interface.

14. The software development company has obtained a robot that is able to produce prototypes of individual components. This robot is not able to perform the size tolerance checks. Explain how this robot could be used during alpha testing. What techniques could be employed to check the size tolerance modules of the software product without the use of the correct robotic hardware?
15. Quality Assurance is vital to the motor vehicle manufacturers for whom the components are manufactured. What types of procedures and testing methods should be used to ensure the new software product meets required quality standards?

CASE TOOLS FOR TESTING SOFTWARE SOLUTIONS

A large variety of Computer Aided Software Engineering tools are available for use during the testing and evaluation stage. These tools automate the more tedious tasks associated with the testing process. Many tools specialise in particular testing functions such as test data generation, user interface testing, or volume testing.

Let us examine a number of CASE tools available at the time of writing:

WinRunner

WinRunner is a functional testing CASE tool. It aims to test that user interface-based applications behave as expected. This product automates the tasks associated with user data entry and navigation.

WinRunner is able to test applications written in most popular languages. Applications written for mainframes can be tested using terminal emulation software.

The tester records the keystrokes necessary to complete a particular process. A script is created based on these recorded keystrokes. The script can then be edited. Data values are replaced with parameters: this allows the test to be executed with multiple test data items. Similarly, parameters can be included to cause outputs to be stored for future analysis. Checkpoints can be inserted into the script that cause WinRunner to compare expected results with actual results.



Fig 6.5

Sample screens from Mercury Interactive's WinRunner product.

LoadRunner

LoadRunner is a volume or load testing CASE tool. This tool is able to simulate hundreds or even thousands of users simultaneously using an application.

Scripts are recorded and then parameterised to allow the use of different live test data. Virtual users are created. Each virtual user can have different characteristics. For example, different time intervals can be set between an individual user's inputs. Tests can be designed that cause multiple virtual users to enter data into the same field simultaneously.

This product contains a number of real-time monitors allowing you to view the application's performance whilst the test is executing. For example, the transaction monitor provides information on average transaction response times, and the server monitor identifies problems associated with application and database servers.



Fig 6.6

Sample screens from Mercury Interactive's LoadRunner product.

DataTech

DataTech is a universal test data generation CASE tool. This product is able to create millions of test data records. Rules for data creation can be produced within the product or determined from existing databases. The test data sets can be written to a large variety of database formats or to text files.

Data can be randomly generated or can be extracted from other data sources. For example, address details such as cities and postcodes can be extracted from a database containing this information.

Data generated by DataTech can be used in conjunction with other testing CASE tools.

UsableNet

This is a website dedicated to testing the useability of other websites. The application analyses the HTML code behind a website and then generates a series of useability reports as well as making recommendations.

Reports are generated on each page of the tested website. Details of each potential problem are made available by clicking on each problem area.

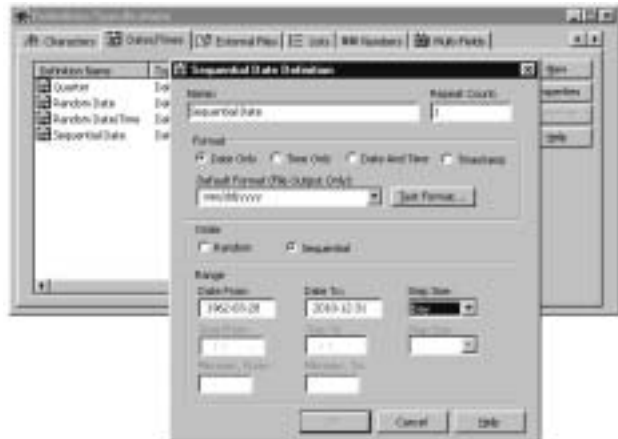


Fig 6.7

Test data generation tool DataTech by Banner Software Inc.

problems	severity	impact	usability problems
Unreadable link or image area	3	high	4
Link unavailable	3	high	1
Unreadable link	2	high	2
Failed downloading of images	2	images	3
Unreadable link area	2	high	2
Link missing in link area	2	images	1
Overlinks to destination	0	searchability	1
Additional keyword	0	searchability	1
Link list	0	high	0
List of links	0	high	1
Page content obscured	0	visibility	3

Fig 6.8

UsableNet provides a test service for websites.



GROUP TASK Research

Use the Internet to search for software testing tools. Collect data on the name of the product, its main function and a brief description of how it works. Create a table to summarise your findings.



GROUP TASK Activity

Download a demonstration version of a functional testing tool from the Internet. Use this tool to create and execute a test on a data entry screen for an application with which you are familiar.

REPORTING ON THE TESTING PROCESS

Documentation and communication of the testing process and its results are vital to any testing program. Results of testing need to be evaluated and acted upon if the problems highlighted are to be corrected. For this to successfully occur requires documentation of the testing process and communication of the results to the development team and to users.

DOCUMENTATION OF THE TESTING PROCESS

Documentation should be developed and retained as the testing process is undertaken. Documents produced will include the following:

- Test requirements What needs to be tested?
- Test plan How do we implement these tests?
- Test data and expected results What are the necessary test inputs and the expected outputs?
- Test results Do the actual results match the expected results?
- Recommendations What needs to be done now?

Test requirements

The test requirements will reflect the objectives and design specifications for the project. Essentially, test requirements detail the scope of the testing process: What tests need to be undertaken, and why? Each component of the program and the system should be considered as part of the test requirements. Hardware, software, data, personnel and processes should all be tested.

Test plan

The test plan will include a schedule and time line of events that will occur during the testing process. It also includes the tools to be used, including software CASE tools and hardware.

Test data and expected results

The actual test data and expected results used throughout the testing process should be retained as documentation. The test data may be required again after modifications have been made to correct problems.

Test results

The test results should be kept to justify any recommendations for changes to the product. These results will be invaluable later to confirm that changes have in fact been implemented.

Recommendations

As a result of the testing process many recommendations will be made in regard to bugs and other problems encountered during the testing process. The recommendations should succinctly describe each problem and explain its severity. Management, in conjunction with the client, will decide which recommendations to act upon and which are acceptable problems. The development team is then handed the task of applying these accepted recommendations.

COMMUNICATION OF TEST RESULTS

The nature of the project will determine the audience for the test results. Projects written for a specific client will require communication of results directly to the client. Large software development companies usually have separate development and testing personnel. In this case, the testers must communicate their findings back to the development team. Often software developers outsource their system testing to specialists. If this is the case, the testing specialists must communicate their findings back to the software development company.

The test results will include problems from all aspects of the system tests. Bugs in the code, inconsistencies in user interface design, unacceptable response times from other applications, hardware conflicts and ambiguous error messages are just some of the possible problems.

Communication with the client for whom the solution has been developed

The language used to report back to users needs to be non-technical. Reporting should address any objectives and design specifications that have not been adequately fulfilled. The reporting process should be in terms of the client's original requirements.

It is normal to rank problems in terms of their severity: a crucial objective that has not been fulfilled would rank highly; minor problems with the response time of a particular function would rank lower. Some problems encountered may be outside the scope of the project; these problems should be made known. It is always better to be upfront and honest about the capabilities of the product. After all, the client will be using the product regularly so eventually they will become aware of problems.

To assist in the communication of test results it is common to give a demonstration of the system to the users. During this demonstration both positive and negative features can be shown. Users are then able to discuss and evaluate the product more realistically.



Fig 6.9

Onsite practical demonstrations are an excellent method of communicating test results to clients.

Communication with developers

Testing departments or companies outsourced to complete the testing process need to communicate their results back to the development team. The communication should highlight problems in order of severity. The recommendations must be backed up by technical justifications. The actual test data together with the erroneous results will provide valuable information for the developers who will be correcting or modifying the product.

Personnel problems are common at this time. Developers often assume the product is finished; to be presented with a list of faults is often viewed personally, as an assault on their professionalism. Testers and developers need to encourage a climate of cooperation and teamwork. The quality of the final product should be the overriding concern.



Consider the following:

A pest control company called ‘Erad-A-Pest’ has some 50 trucks completing domestic pest control services. The company has employed a software development company to develop and implement an automated booking, scheduling and invoicing system. Each pest controller is notified of jobs via text messages sent over the Internet to the mobile phone digital network.



The software has just undergone alpha and beta testing. The following table lists a number of the problems discovered during the testing process.

Problem	Source	Description	Severity
Duplicate orders processed.	Orders module	If a client sends an order twice, this is not detected by the system.	High
Address validation slow.	Scheduling module	Client addresses are validated against their phone number. This takes between 2 and 3 seconds.	Low
Discounts calculated incorrectly.	Invoicing module	Discounts are deducted after GST has been calculated.	Extreme
Inconsistent acknowledgement of receipt of text messages.	Scheduling module	Acknowledgement is not received for approximately 5% of text messages sent to pest controllers.	High
System does not reconnect to ISP.	All modules	Loss of connection to ISP requires restarting the software.	Low
Completion of job notifications inconsistent.	Procedural	Pest controllers do not submit 7% of job completion notifications.	Medium



GROUP TASK Discussion

Examine each of the problems identified in the above table. Suggest strategies for correcting each of these problems.



GROUP TASK Activity

Create two written reports outlining your recommendations as a consequence of these test results. One report should be targeted at the users of the system and the other at the development team.



GROUP TASK Discussion

Which method of conversion is appropriate for the introduction of this product: pilot, parallel, direct cut over or phased? Justify your answer.

CHAPTER 6 REVIEW

1. Testing is carried out during which stage of the software development cycle?
 - (A) Planning and design.
 - (B) Implementation.
 - (C) Testing and evaluation.
 - (D) All of the above.
2. Problems encountered during system testing are often the result of hardware problems. Which of the following is true?
 - (A) Correcting problems due to hardware is beyond the scope of the software developer.
 - (B) The problem is usually solved by modifying the hardware requirements of the software.
 - (C) Correcting faults in hardware is generally not possible; therefore software developers need to find software solutions to allow their products to operate correctly.
 - (D) Hardware faults are inevitable, as are software faults. Users generally accept this as a fact of life.
3. CASE tools used during the testing evaluation stage:
 - (A) are used to automate tedious and repetitious tasks.
 - (B) provide a means of producing system models.
 - (C) assist developers in the construction of code.
 - (D) ensure design specifications remain consistent across the application.
4. Customised off-the-shelf (COTS) packages can greatly simplify and shorten the software development process. Testing of software based on COTS packages can be difficult because:
 - (A) differing versions of the COTS package may cause changes and/or problems to the product.
 - (B) errors in the COTS package cannot generally be rectified by the software developer.
 - (C) other products that use the services of the COTS package can affect functionality in unexpected ways.
 - (D) All of the above.
5. The difference between live test data designed to test large file sizes and that designed to test volume data is:
 - (A) large file sizes test the ability of the software to retrieve data; volume data tests the ability of the software to process large amounts of input simultaneously.
 - (B) large file sizes test the ability of the software to process data correctly; volume data tests the ability of the software to process large amounts of data.
 - (C) large files ensure the product is robust in design; volume data ensures logic errors are identified.
 - (D) large file sizes can cause arithmetic overflow errors which must be identified; volume data is used to test the user interface.
6. Why should program test data be used again during system testing?
 - (A) It is easier than producing new test data sets.
 - (B) Problems identified during system testing are the same as those identified during program testing.
 - (C) The outputs from both tests can be compared. Differences will highlight system issues compared to program issues.
 - (D) The expected outputs are known, hence correcting any errors will be straightforward.
7. Live test data:
 - (A) ensures the logic of source code is error free.
 - (B) is used to test the operation of the system under extreme conditions.
 - (C) is only used during beta testing.
 - (D) can be used most effectively once the program has been distributed.
8. CASE tools that can simulate the activities of a large number of users are used for:
 - (A) functional testing.
 - (B) load testing.
 - (C) test data generation.
 - (D) communicating test results.

9. Documentation should be created throughout the testing process. The major purpose of this documentation is to:
- justify the costs of the testing process.
 - provide evidence to support the recommendations.
 - involve testers in the development of the product.
 - ensure that the integrity of the testing process is maintained.
10. Communicating test results effectively is crucial to the testing operation. Why?
- This is the means by which recommendations will be put into practice.
 - The significance of problems must be made clear.
 - Problems not corrected now will prove more costly to correct once the product has been distributed.
 - All of the above.
11. The user interface is a vital component of most data-oriented software products. The layout and nature of screen elements together with response times and data validation processes all influence the experience for the user. What testing techniques can be used to test the user interface? Describe how live test data can be used to assist this process.
12. Live test data is commonly used to perform system testing. Different test data sets are designed to test particular aspects of the final product. Describe at least three different scenarios that require different test data sets. How can each of these test data sets be obtained or created?

Use the following scenario to answer Questions 13 and 14.

A watch which is controlled by a microprocessor is under development. This watch combines the functions of a traditional watch with a number of innovative features designed specifically for use by athletes. Heart rate, blood pressure and body temperature are constantly monitored. A low and high value for each of these three values can be input using buttons on the side of the watch to scroll through possible values. Readings that are outside the input range cause an audible alarm to be emitted that alerts the wearer. The software that controls the watch is now ready for alpha testing, together with a prototype of the final watchcase and associated hardware.



13. What is alpha testing? Develop a list of tests that should be performed on the watch during the alpha testing process. For each test describe the nature of any test data required.
14. A number of faults are identified during the alpha testing. The following list describes these faults:
- Watch strap needs to be tight for temperature sensor to operate reliably.
 - Once a range is set for the heart rate function, removing the watch causes the alarm to sound.
 - Body temperatures reported by the watch are wrong. Many temperatures reported are outside those possible for a living person.
 - The date does not take account of leap years correctly.

It seems strange to the testers that many of these faults were not encountered during the coding of the software. Can you explain this situation? Discuss in relation to each fault.

15. Imagine you have been enlisted to beta test a new software product. A large multi-national software company has developed this product. The product is called 'Smart Type'. Essentially, it attempts to guess the word you are typing, the aim being to reduce the time spent entering data into word processors. Guesses are based on past words you have entered, the theory being that it is likely you will want to use these words often. The product can, apparently, be used with any existing word processor.

Develop a series of tests you would use to beta test this product. Describe the purpose of each test you devise. Your response should include tests of the hardware, software, data, personnel and procedures involved in the use of this software product.

