Higher-Order Testing

A software error occurs when the program does not do what its end user reasonably expects it to do.

References

[3] – Chapter 7, 8, 9

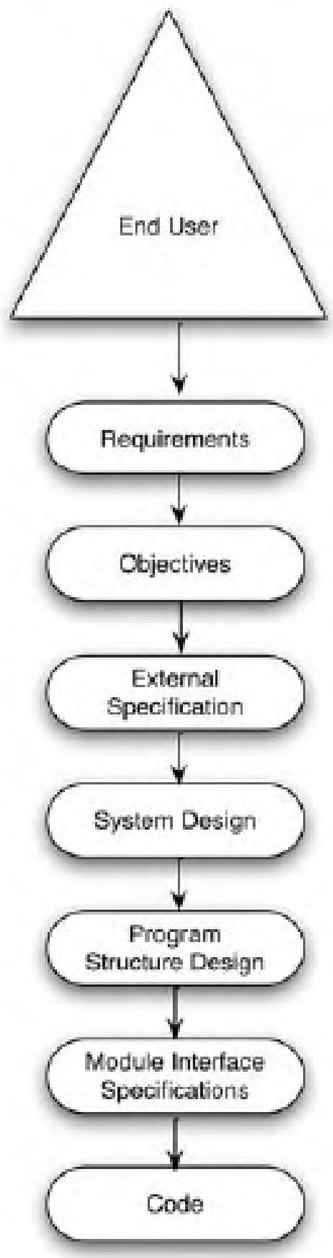
[4] – Chapter 5, 6

Content

- Introduction
- Function Testing
- System Testing
- Acceptance Testing
- Installation Testing
- Regression Testing

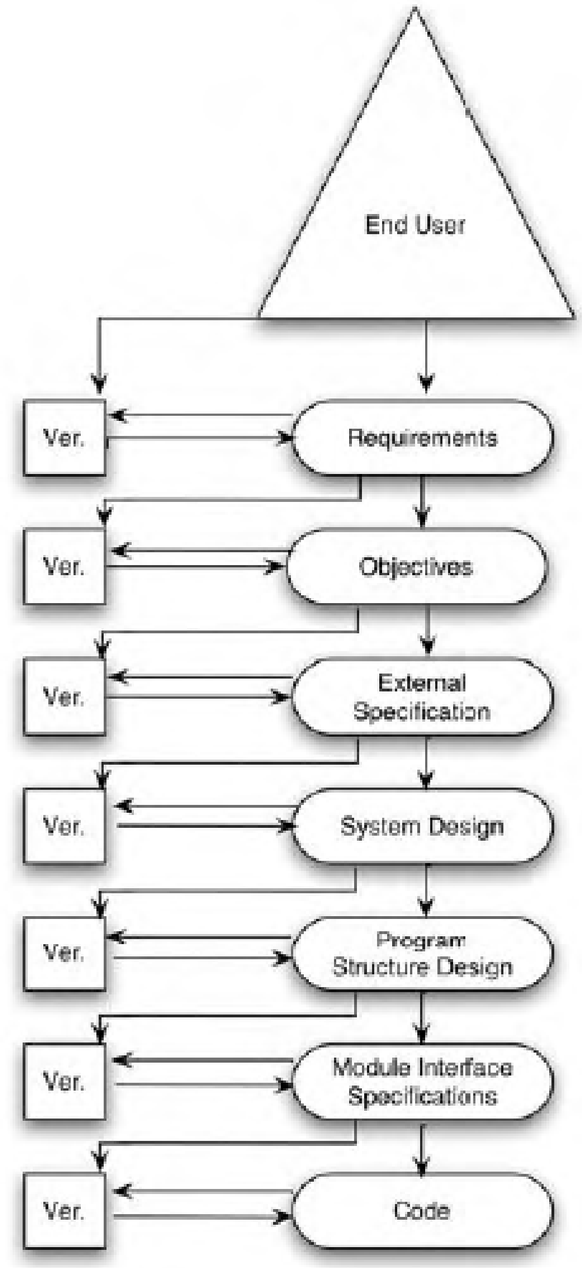
Introduction

- Even if you could perform an absolutely perfect module test, you still couldn't guarantee that you have found all software errors => *higher-order testing*
- Software development
  - Requirements specify why the program is needed.
  - Objectives specify what the program should do and how well the program should do it.
  - External specifications define the exact representation of the program to users.
  - Documentation associated with the subsequent processes specifies, in increasing levels of detail, how the program is constructed.



Introduction

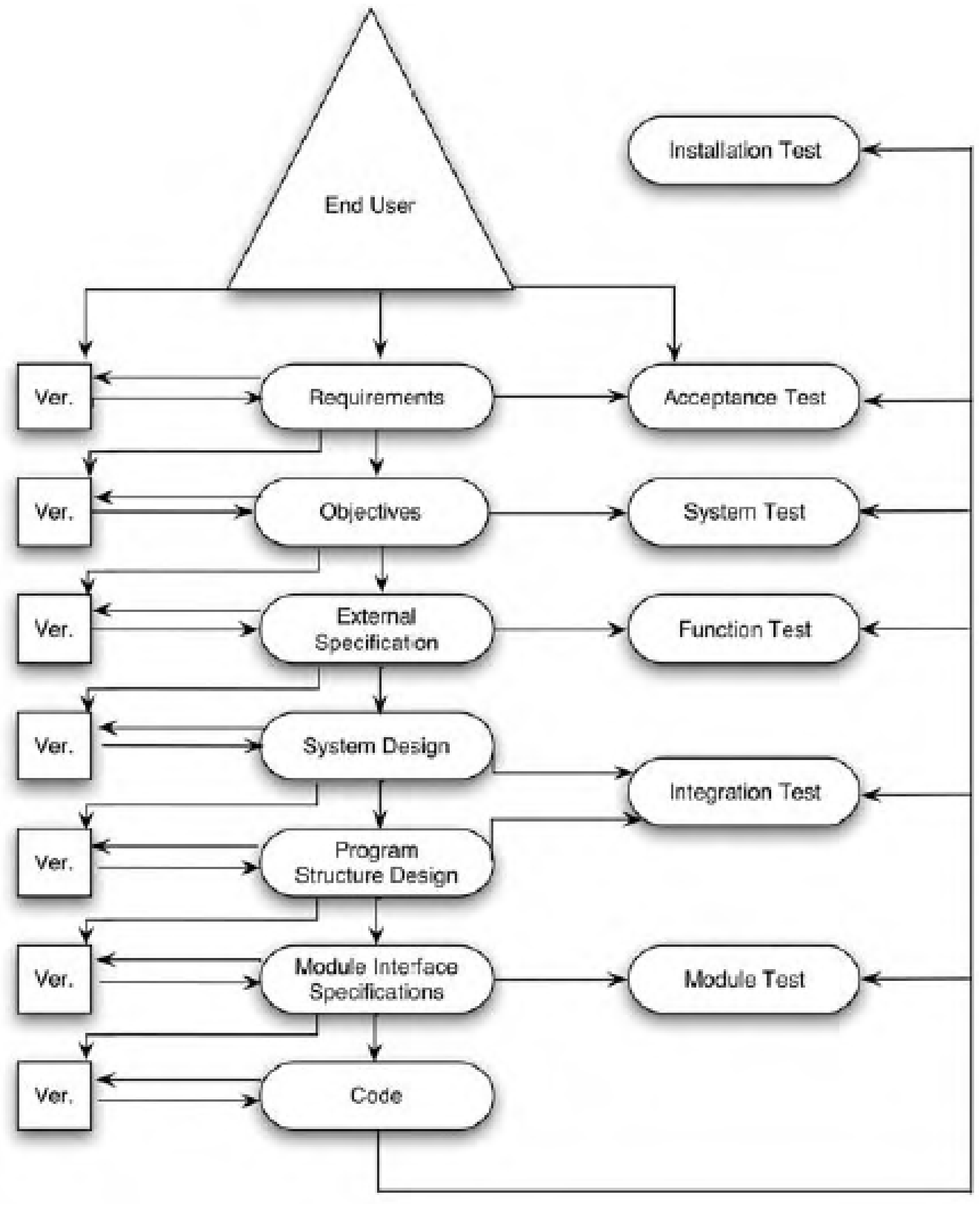
- Complementary approaches to prevent and/or detect these errors
  - Introduce more precision into the development process to prevent many of the errors
  - Introduce, at the end of each process, a separate verification step to locate as many errors as possible before proceeding to the next process (code inspection and walkthrough methods)



Introduction

- **Orient distinct testing processes toward distinct development processes.**

  Focus each testing process on a particular translation step, thus focusing it on a particular class of errors



Introduction

- The purpose of a module test is to find discrepancies between the program's modules and their interface specifications
- The purpose of a function test is to show that a program does not match its external specifications.
- The purpose of a system test is to show that the product is inconsistent with its original objectives.
- Advantages:
  - avoids unproductive redundant testing
  - prevents you from overlooking large classes of errors
- Note: the sequence of testing processes in the previous figure does not necessarily imply a time sequence

Function Testing

- A process of attempting to find discrepancies between the program and the external specification.

  An external specification is a precise description of the program's behavior from the point of view of the end user.
- Function testing is normally a black-box activity
  - Equivalence-partitioning
  - Boundary-value analysis
  - Cause-effect graphing
  - Use case
  - State Diagram
  - Etc…

Function Testing

- Approaches for function testing
  - User Navigation Testing
  - Transaction Screen Testing
  - Transaction Flow Testing
  - Report Screen Testing
  - Report Flow Testing
  - Database Create/Retrieve/Update/Delete Testing

Function Testing

- User Navigation Testing
  - Navigation screens are those
    - log on and log off screens that control access to the software
    - all menus that provide alternate activity paths through the software,
    - all screen-to-screen linkage that represents a continuum of some business activity.
  - User navigation testing focuses on
    - the user's ability to log on to the software with appropriate authority,
    - traverse the application to the desired transaction screens,
    - traverse the transaction screens correctly, and log off the software
  - **Note:** "stateless" user screens, user screens that you can land on from a variety of other screens without a prerequisite screen sequence. Indeed, it is common for several "stateless" screens to be active at the same time

Function Testing

- Transaction Navigation Testing
  - The transaction screen normally has
    - Input data fields, lists of choices, options, and action buttons (Add, Change, Delete, Submit, Cancel, OK, and so forth).
    - Some kind of results may be displayed on the transaction screen after appropriate action buttons are pressed.
  - The tester's job
    - Design tests that validate the operation of every field, list, option, and action button on each transaction screen against the business requirements, the user guide, and the administrator guide
    - If results are also displayed on the transaction screen, then the black box inputs versus the expected result technique is used to validate the displayed results.

Function Testing

- Transaction Flow Testing
  - Takes the transaction screens that have been validated by testing and determines if their combined results of correct navigation completes the intended business activity in some specified way.
  - Example: validate customer profile updates as
    - transaction screen 1 for customer name, address, and contact person
    - transaction screen 2 for customer line of credit approval
    - transaction screen 3 for customer payment terms and discounts
    - transaction screen 4 for profile summary and update action
    - transaction screen 5 for viewing updated customer profile
    - The result of the sequence of file screens being completed is expected to be a master file or database file update with all the information collected on these transaction screens.

Function Testing

- Transaction Flow Testing
  - The tester's job
    - Validate that correctly completing the sequence of transaction screens does provide a correct result
    - Validates that if any of the system's business rules are violated, the system does not provide any result

Function Testing

- Report Screen Testing
  - Similar to transaction screen testing.
  - You are attempting to retrieve and display data from the system using the report screens instead of entering data using the transaction screens.
  - The difficulty in report screen testing usually lies in the variety of ways an end user can specify which data are retrieved (search criteria) and how these data are displayed (sorting and formatting options).
  - The tester's job is to pay particular attention to the data retrieved and displayed because the wrong data may have been selected or, worse yet, not all data requested were displayed.

Function Testing

- Report Flow Testing
  - Different from report screen testing when the report results are provided in other modalities besides on-screen display (for example: hardcopy output, print to file)
  - Testers:
    - Does the software send exactly the same results to the printer that it displays on the report screen?

      Having a printer option implies the possibility of the report screen offering a selection of print fonts, another fertile area for testing
    - Validate the report results on all the alternative report modalities supported by the software.

Function Testing

- Database Create/Retrieve/Update/Delete Testing
  - Normally done in two steps:
    - Test the database design viability by successfully performing the application data manipulations outside of the application
    - Test the application software's use of the validated database design.
  - Requires cooperation and collaboration between the tester and the database administrator

System Testing

- Introduction
- Facility Testing
- Volume Testing
- Stress Testing
- Usability Testing
- Security Testing
- **Performance Testing**
- Storage Testing
- Configuration Testing
- Compatibility / Configuration / Conversion Testing
- Installability Testing
- Reliability Testing
- Recovery Testing
- Serviceability Testing
- Documentation Testing
- Procedure Testing

System Testing - Introduction

- Not a process of testing the functions of the complete system or program
- A particular purpose: to compare the system or program to its original objectives.
  - System testing is not limited to systems. If the product is a program, system testing is the process of attempting to demonstrate how the program, as a whole, does not meet its objectives.
  - System testing, by definition, is impossible if there is no set of written, measurable objectives for the product.

System Testing - Introduction

- Objective Examples
  - A command will be provided to view, from a terminal, the contents of main-storage locations.
    - Its syntax should be consistent with the syntax of all other system commands.
    - The user should be able to specify a range of locations, via an address range or an address and a count.
    - Sensible defaults should be provided for command operands.

System Testing - Introduction

- Output should be displayed as multiple lines of multiple words (in hexadecimal), with spacing between the words.
- Each line should contain the address of the first word of that line.
- The command is a "trivial" command, meaning that under reasonable system loads, it should begin displaying output within two seconds, and there should be no observable delay time between output lines.
- A programming error in the command processor should, at the worst, cause the command to fail; the system and the user's session must not be affected.
- The command processor should have no more than one user-detected error after the system is put into production.

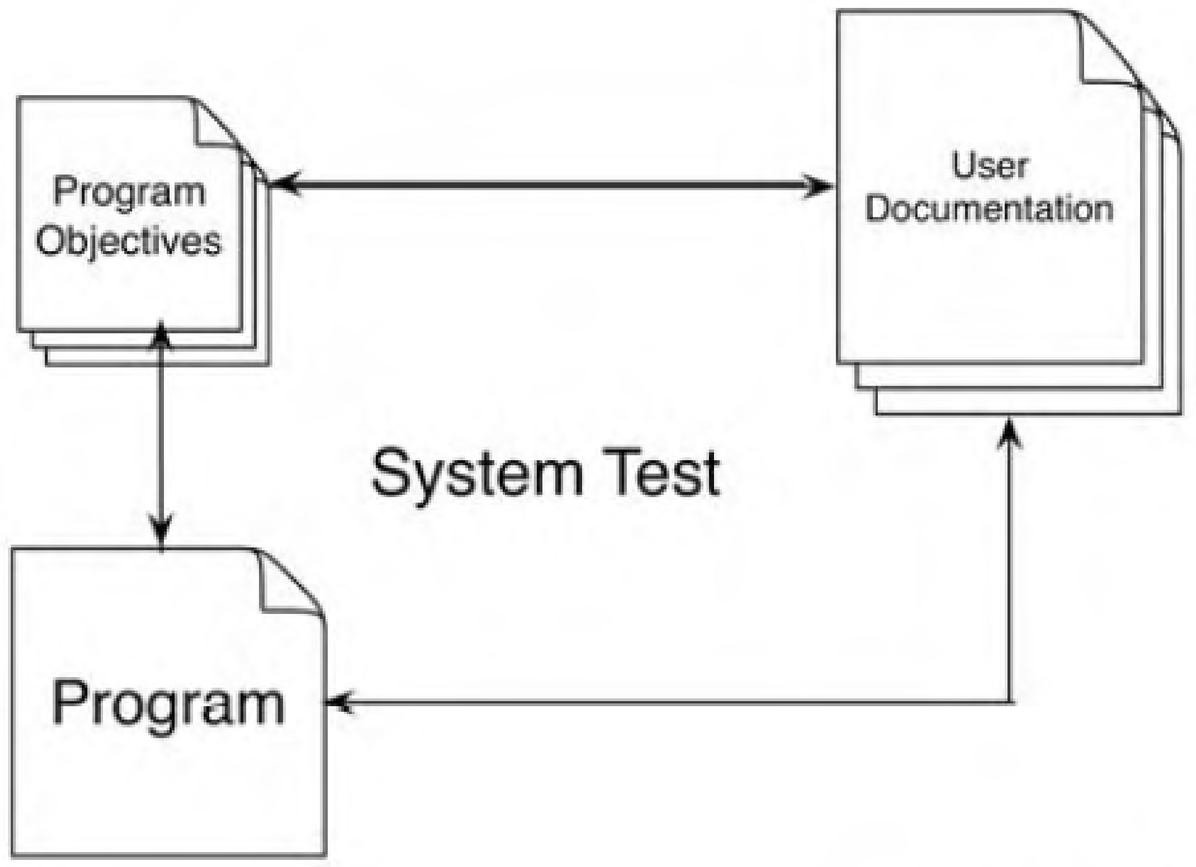
System Testing - Introduction

- The system test a vital test process
- Materials to create test cases
  - The external specification cannot be used as the basis for deriving the system test cases
  - The objectives document cannot be used, by itself, to formulate test cases, since it does not, by definition, contain precise descriptions of the program's external interfaces

=>using the program's user documentation or publications

- Design the system test by analyzing the objectives;
- Formulate test cases by analyzing the user documentation.

System Testing - Introduction



External Specifications

System Testing - Introduction

- System testing is the most difficult testing process
  - Comparing the program to its objectives: no known test-case-design methodologies.
  - A different approach to test-case design is taken
    - Rather than describing a methodology, distinct categories of system test cases are discussed.
    - Because of the absence of a methodology, system testing requires a substantial amount of creativity
    - 15 categories of test cases

Facility Testing

- Determine whether each facility mentioned in the objectives was actually implemented.
- Procedures:
  - Scan the objectives sentence by sentence
  - When a sentence specifies a what (for example, "syntax should be consistent . . . ," "user should be able to specify a range of locations . . ."), determine that the program satisfies the "what."
- Often can be performed without a computer; a mental comparison of the objectives with the user documentation is sometimes sufficient

Volume Testing

- Subjects the program to heavy volumes of data.
- The purpose of volume testing is to show that the program cannot handle the volume of data specified in its objectives.
- Examples:
  - A compiler would be fed an absurdly large source program to compile.
  - A linkage editor might be fed a program containing thousands of modules
- Require significant resources, in terms of machine and people time, you can't go overboard.

Stress Testing

- Subjects the program to heavy loads or stresses.
- A heavy stress is a peak volume of data, or activity, encountered over a short span of time
- Analogy:
  - A volume test would determine whether the typist could cope with a draft of a large report;
  - A stress test would determine whether the typist could type at a rate of 50 words per minute.
- Involves an element of time, it is not applicable to many programs

Stress Testing

- Examples:
  - If an operating system is supposed to support a maximum of 15 Multi-programmed jobs, the system could be stressed by attempting to run 15 jobs simultaneously.
  - Web-based applications: ensure that your application, and hardware, can handle some volume of concurrent users

Usability Testing

- An attempt to find human-factor, or usability, problems.
- Considerations
  - Has each user interface been tailored to the intelligence, educational background, and environmental pressures of the end user?
  - Are the outputs of the program meaningful, non-abusive, and devoid of computer gibberish?
  - Are the error diagnostics, such as error messages, straightforward, or does the user need a PhD in computer science to comprehend them?
  - Does the total set of user interfaces exhibit considerable conceptual integrity, an underlying consistency, and uniformity of syntax, conventions, semantics, format, style, and abbreviations?

Usability Testing

- Considerations
  - Where accuracy is vital, such as in an online banking system, is sufficient redundancy present in the input?
  - Does the system contain an excessive number of options, or options that are unlikely to be used?
  - Does the system return some type of immediate acknowledgment to all inputs?
  - Is the program easy to use?

Serviceability Testing

- The program also may have objectives for its serviceability or maintainability characteristics.
- All objectives of this sort must be tested. Such objectives might define the service aids to be provided with the system, including:
  - Storage dump programs or diagnostics
  - The mean time to debug an apparent problem
  - The maintenance procedures
  - The quality of internal logic documentation.

Security Testing

- The process of attempting to devise test cases that subvert the program's security checks.
- One way to devise such test cases is to study known security problems in similar systems and generate test cases that attempt to demonstrate similar problems in the system you are testing
- Web-based applications often need a higher level of security testing than do most applications. This is especially true of e-commerce sites.

Performance Testing

- Response times and throughput rates under certain workload and configuration conditions.
- The purpose of a system test is to demonstrate that the program does not meet its objectives, test cases must be designed to show that the program does not satisfy its performance objectives.

Storage Testing

- Programs occasionally have storage objectives that state, for example, the amount of main and secondary memory the program uses and the size of temporary or spill files.
- You should design test cases to show that these storage objectives have not been met.

Configuration Testing

- Programs such as operating systems, database management systems, and message-switching programs support a variety of hardware configurations
- Often the number of possible configurations is too large to test each one, but at the least, you should test the program with each type of hardware device and with the minimum and maximum configuration.

Compatibility/Configuration/Conversion Testing

- Most programs that are developed are not completely new; they often are replacements for some deficient system
- Programs often have specific objectives concerning their compatibility with, and conversion procedures from, the existing system
- Examples:
  - Upgrading a database management system.

Installability Testing

- Some types of software systems have complicated installation procedures.
- A malfunctioning installation program could prevent the user from ever having a successful experience with the main system you are charged with testing

Reliability Testing

- The goal of all types of testing is the improvement of the program reliability
- If the program's objectives contain specific statements about reliability, specific reliability tests might be devised
- Testing reliability objectives can be difficult.
  - For example, a modern online system such as a corporate wide area network (WAN) or an Internet service provider (ISP) generally has a targeted uptime of 99.97 percent over the life of the system.
  - There is no known way that you could test this objective with a test period of months or even years.

Recovery Testing

- Programs such as operating systems, database management systems, and teleprocessing programs often have recovery objectives that state how the system is to recover from programming errors, hardware failures, and data errors
- Objective:
  - Show that these recovery functions do not work correctly
  - show that the system fails to meet the service-level agreement for the mean time to recovery (MTTR).

Documentation Testing

- The system test also is concerned with the accuracy of the user documentation
- The principle way of accomplishing this is to use the documentation to determine the representation of the prior system test cases.
- The user documentation should be the subject of an inspection (similar to the concept of the code inspection), checking it for accuracy and clarity

Procedure Testing

- Many programs are parts of larger, not completely automated systems involving procedures people perform
- Any prescribed human procedures, such as procedures for the system operator, database administrator, or end user, should be tested during the system test

Acceptance Testing

- The process of comparing the program to its initial requirements and the current needs of its end users
- Usually performed by the program's customer or end user and normally not considered the responsibility of the development organization.
- In the case of a contracted program, the contracting (user) organization performs the acceptance test by comparing the program's operation to the original contract
- In the case of a program product, such as a computer manufacturer's operating system or compiler, or a software company's database management system, the sensible customer first performs an acceptance test to determine whether the product satisfies its needs.

Installation Testing

- Its purpose is not to find software errors but to find errors that occur during the installation process.
- Many events occur when installing software systems
  - User must select a variety of options.
  - Files and libraries must be allocated and loaded.
  - Valid hardware configurations must be present.
  - Programs may need network connectivity to connect to other programs.
- Test cases might check to ensure that
  - A compatible set of options has been selected
  - All parts of the system exist
  - All files have been created and have the necessary contents
  - The hardware configuration is appropriate
- Should be developed by the organization that produced the system, delivered as part of the system, and run after the system is installed

Regression Testing

- Regression testing means rerunning test cases from existing test suites to build confidence that software changes have no unintended side-effects.
  - Normally started while new code is being written and existing code is being corrected.
  - Also done from version to version.
- The "ideal" process would be to create an extensive test suite and run it after each and every change.
- Regression test selection (RTS) techniques
  - Unpredictable performance
  - Incompatible process assumptions
  - Inappropriate evaluation models
- Can be automated

Q&A