Chapter 6

Software Testing

ISTQB / ISEB Foundation Exam Practice

1 Principles	2 Lifecycle	3 Static testing
4 Dynamic test techniques	5 Management	6 Tools

Tool support for testing

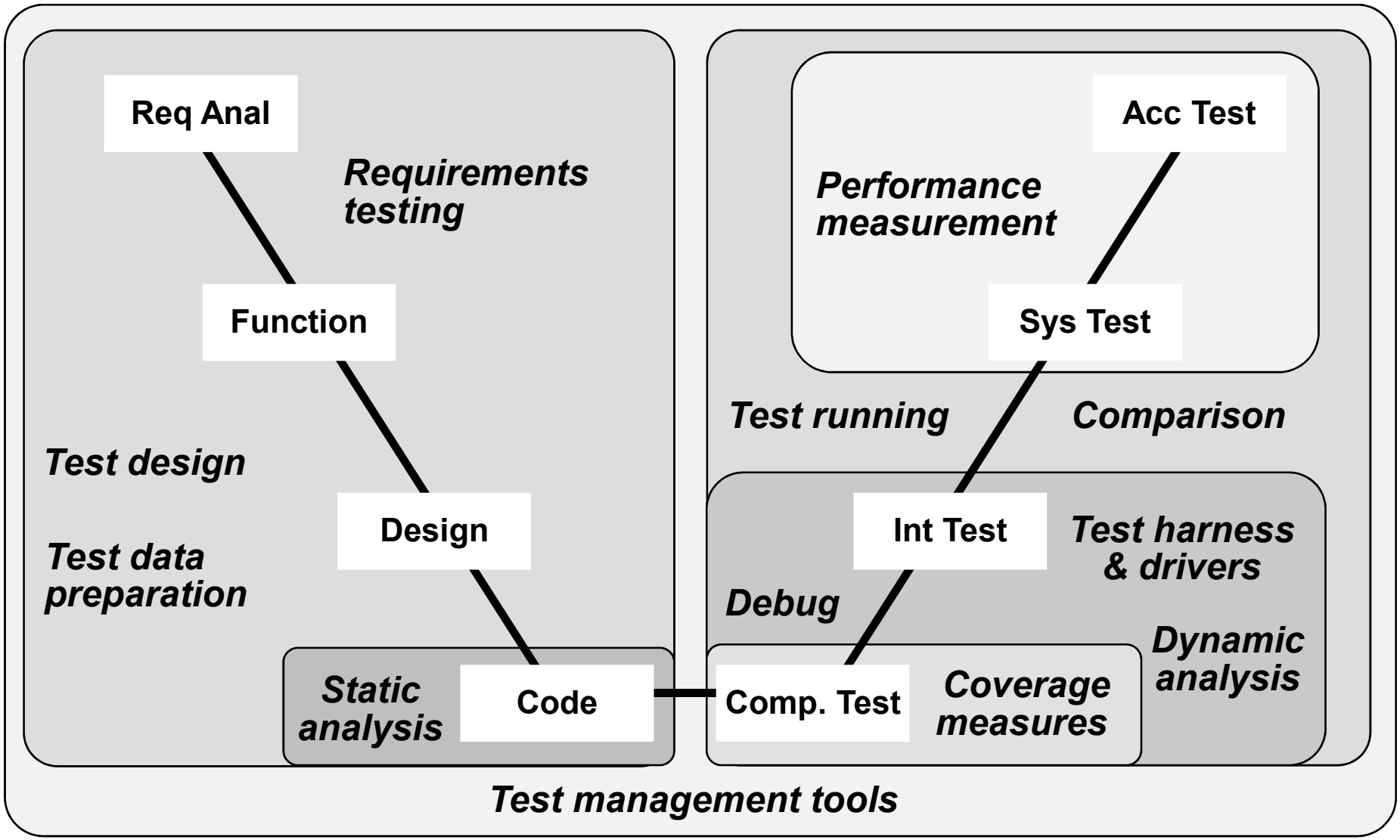
1	2	3
4	5	6

Contents

Testing tool classification

- Requirements testing tools
- Static analysis tools
- Test design tools
- Test data preparation tools
- Test running tools - character-based, GUI
- Comparison tools
- Test harnesses and drivers
- Performance test tools
- Dynamic analysis tools
- Debugging tools
- Test management tools
- Coverage measurement

Where tools fit



Requirements testing tools

- **Automated support for verification and validation of requirements models**
  - **consistency checking**
  - **animation**

Tool information available from:

Ovum Evaluates Software Testing Tools (subscription service)
CAST Report, 1999
World Wide Web

Static analysis tools

- Provide information about the quality of software
- Code is examined, not executed
- Objective measures
  - cyclomatic complexity
  - others: nesting levels, size

Test design tools

- **Generate test inputs**
  - **from a formal specification or CASE repository**
  - **from code (e.g. code not covered yet)**

Test data preparation tools

- **Data manipulation**
  - **selected from existing databases or files**
  - **created according to some rules**
  - **edited from other sources**

Test running tools 1

- Interface to the software being tested
- Run tests as though run by a human tester
- Test scripts in a programmable language
- Data, test inputs and expected results held in test repositories
- Most often used to automate regression testing

Test running tools 2

- **Character-based**
  - simulates user interaction from dumb terminals
  - capture keystrokes and screen responses
- **GUI (Graphical User Interface)**
  - simulates user interaction for WIMP applications (Windows, Icons, Mouse, Pointer)
  - capture mouse movement, button clicks, and keyboard inputs
  - capture screens, bitmaps, characters, object states

Comparison tools

- **Detect differences between actual test results and expected results**
  - **screens, characters, bitmaps**
  - **masking and filtering**
- **Test running tools normally include comparison capability**
- **Stand-alone comparison tools for files or databases**

Test harnesses and drivers

- Used to exercise software which does not have a user interface (yet)
- Used to run groups of automated tests or comparisons
- Often custom-build
- Simulators (where testing in real environment would be too costly or dangerous)

Performance testing tools

- Load generation
  - drive application via user interface or test harness
  - simulates realistic load on the system & logs the number of transactions
- Transaction measurement
  - response times for selected transactions via user interface
- Reports based on logs, graphs of load versus response times

Dynamic analysis tools

- Provide run-time information on software (while tests are run)
  - allocation, use and de-allocation of resources, e.g. memory leaks
  - flag unassigned pointers or pointer arithmetic faults

Debugging tools

- Used by programmers when investigating, fixing and testing faults
- Used to reproduce faults and examine program execution in detail
  - single-stepping
  - breakpoints or watchpoints at any statement
  - examine contents of variables and other data

Test management tools

- Management of testware: test plans, specifications, results
- Project management of the test process, e.g. estimation, schedule tests, log results
- Incident management tools (may include workflow facilities to track allocation, correction and retesting)
- Traceability (of tests to requirements, designs)

Coverage measurement tools

- Objective measure of what parts of the software structure was executed by tests
- Code is instrumented in a static analysis pass
- Tests are run through the instrumented code
- Tool reports what has and has not been covered by those tests, line by line and summary statistics
- Different types of coverage: statement, branch, condition, LCSAJ, et al

1	2	3
4	5	6

Content

Types of test tool

Effective Use of Test Tools

Introducing a Tool into an Organization

Effective Use of Tools: Potential Benefits and Risks

- Potential benefits of using tools
- Risks of using tools
- Special considerations for some types of tools

Potential benefits of using tools

- Reduction of repetitive work
- Greater consistency and repeatability
- Objective assessment
- Ease of access to information about tests or testing.

Risks of using tools

There are many risks that are present when tool support for testing is introduced and used, whatever the specific type of tool.

- unrealistic expectations for the tool;
- underestimating the time, cost and effort for the initial introduction of a tool;
- over-reliance on the tool
- failing to consider and manage issues related to relationships and interoperability between critical tools

Risks of using tools

- Possibility of the tool vendor going out of business, retiring the tool, or selling the tool to a different vendor, or in the open-source world
- Poor or completely non-existent vendor response, whether for support, upgrades, or defect fixes
- Various uncertainties and unforeseen problems, such as the inability to support a new platform

Special considerations for some types of tools

- Test execution tools
- Performance testing tools
- Static analysis tools
- Test management tools

1	2	3
4	5	6

Content

Types of test tool

Effective Use of Test Tools

Introducing a Tool into an Organization

Introducing a Tool Into an Organization

- State the main principles of introducing a tool into an organization.
- State the goals of a proof-of-concept for tool evaluation and a piloting phase for tool implementation.
- Recognize that factors other than simply acquiring a tool are required for good tool support.

Main principles

The following factors are important in selecting a tool:

- assessment of the organization's maturity (e.g. readiness for change);
- identification of the areas within the organization where tool support will help to improve testing processes;
- evaluation of tools against clear requirements and objective criteria;
- proof-of-concept to see whether the product works as desired and meets the requirements and objectives defined for it;

Main principles

The following factors are important in selecting a tool:

- evaluation of the vendor (training, support and other commercial aspects) or open-source network of support;
- identifying and planning internal implementation (including training, coaching and mentoring for those new to the use of the tool);
- estimation of the return on investment (cost-benefit ratio) based on a concrete and realistic business case.

Pilot project

The objectives for a pilot project for a new tool are:

- To learn more about the tool (more detail, more depth);
- To see how the tool would fit with existing processes or documentation, how those would need to change to work well with the tool and how to use the tool to streamline existing processes;

Pilot project

- To decide on standard ways of using the tool that will work for all potential users (e.g. naming conventions, creation of libraries, defining modularity, where different elements will be stored, how they and the tool itself will be maintained);
- To evaluate the pilot project against its objectives (have the benefits been achieved at reasonable cost?).

Success factors

Here are some of the factors that have contributed to success:

- incremental roll-out (after the pilot) to the rest of the organization;
- adapting and improving processes, testware and tool artefacts to get the best fit and balance between them and the use of the tool;
- providing adequate support, training, coaching and mentoring of new users;
- defining and communicating guidelines for the use of the tool, based on what was learned in the pilot;

Success factors

- implementing a continuous improvement mechanism as tool use spreads through more of the organization;
- monitoring the use of the tool and the benefits achieved and adapting the use of the tool to take account of what is learned;
- provide continuing support for anyone using test tools, such as the test team; for example, technical expertise is needed to help non-programmer testers who use keyword-driven test automation;
- continuous improvement of tool use should be based on information gathered from all teams who are using test tools.

Summary: Key Points

There are many different types of tool support for testing, covering all areas of the life cycle.

Effective Use of Tools: Potential Benefits and Risks

Introducing a Tool Into an Organization