Test Tools

classification according to test activities

Tools for management of testing and testware

* Test management tools and application lifecycle management tools (ALM)
* Requirements management tools (traceability to test objects)
* Defect management tools
* Configuration management tools
* Continuous integration tools

special considerations

Test management tools often need to interface with other tools or spreadsheets for various reasons, including:

- to produce useful information in a format that fits the needs of the organization
- to maintain consistent traceability to requirements in a requirements management tool
- to link with test object version information in the configuration management tool

This is particularly important to consider when using an integrated tool (ALM), which includes a test management module, as well as other modules (project schedule and budget information) that are used by different groupd within an organization.

Tools for static testing

* Static analysis tools

NB: Some tools offer support that is typically more appropriate for developers.

Tools for test design and implementation

* Model-Based testing tools
* Test data preparation tools

Tools for test execution and logging

* Test execution tools (to run regression tests)
* Coverage tools (requirements coverage, code coverage)
* Test harnesses – automated test frameworks; collection of software and test data configured to test a program unit by running it under varying conditions and monitoring its behavior and output

special considerations

Test execution tools execute test objects using automated test scripts. This type of tools often requires significant effort in order to achieve significant benefits.

* **Capturing test approach:** Capturing tests by recording the actions of a manual tester seems attractive, but this approach does not scale to large number of test scripts. A captured script is a linear representation with specific data and actions as part of each script. This type of script may be unstable when unexpected events occur, and require ongoing maintenance as the system's user interface evolves over time.
* **Data-driven test approach:** This test approach separates out the test inputs and expected results, usually into a spreadsheet, and uses a more generic test script that can read the input data and execute the same test script with different data.
* **Keyword-driven test approach:** This test approach, a generic script processes keywords describing the actions to be taken (also called action words), which then calls keyword scripts to process the associated test data.

The above approached required someone to have expertise in the scripting language (testers, developers or specialists in test automation).

Model-Based testing tools enable a functional specification to be captures in the form of a model, such as an activity diagram. This task is generally performed by a system designer. The MBT tool interprets the model in order to create test case specifications which can then be saved in a test management tool and/or executed by a test execution tool.

Tools for performance measurement and dynamic analysis

* Performance testing tools
* Dynamic analysis tools

Tools for specialized testing needs

* Other tools supporting more specific testing for non-functional characteristics

NB: Some tools can be intrusive: they may affect the actual outcome of the test (response times, amount of code coverage).
This is called the probe effect.

Test Tools

Benefits and Risks. Effective Use.

Benefits

- Reduction in repetitive manual work (running regression tests, environment set up/tear down tasks, re-entering the same test data, and checking against coding standards), this saving time.

- Greater consistency and repeatability (test data is created in coherent manner, tests are executed by a tool in the same order with the same frequency, and tests are consistently derived from requirements).

- More objective assessment (static measures, coverage).

- Easier access to information about testing (statistics and graphs about test progress, defect rates and performance).

Risks

- Expectations for the tool may be unrealistic (including functionality and ease of use).

- The time, cost and effort for the initial introduction of a tool may be under-estimated (including training and external expertise).

- The time and effort needed to achieve significant and continuing benefits from the tool may be under-estimated (including the need for changes in the test process and continuous improvement in the way he tool is used).

- The effort required to maintain the test work products generated by the tool may be under-estimated.

- The tool may be relied on too much (seen as a replacement for test design or execution, or the use of automated testing where manual testing would be better).

- Version control test work products may be neglected.

- Relationships and interoperability issues between critical tools may be neglected, such as requirements management tools, configuration management tooks, defect management tools and tools from multiple vendors.

- The tool vendor may go out of business, retire the tool, or sell the tool to a different vendor.

- The vendor may provide a poor response for support, upgrades, and defect fixes.

- An open source project may be suspended.

- A new platform or technology may not be supported by the tool.

- There may be no clear ownership of the tool (for mentoring, updates, etc.)

Main Principles for Tool Selection

***** Assessment of the maturity of the own organization, its strenghts and weaknesses

***** Identifications of oppotunities for an improved test process supported by tools

***** Understanding of the technologies used by the test object(s), in order to select a tool that is compatible with that technology

***** Understanding the build and continuous integration tools already in use within the organization, in order to ensure tool compatibility and integration

***** Evaluation of the tool against clear requirements and objective criteria

***** Consideration of whether or not the tool is available for a free trial period (and for how long)

***** Evaluation of the vendor (including training, support and commercial aspects) or support for non-commercial (open source) tools

***** Identification of internal requirements for coaching and mentoring in the use of the tool

***** Evaluation of training needs, considering the testing (and test automation) skills of those who will be working directly with the tool(s)

***** Consideration of pros and cons of various licensing models (commercial or open source)

***** Estimation of a cost-benefit ratio based on a concrete business case (if required)

Pilot Project for Introducing a Tool

After selecting a tool based on a proof-of-concept evaluation, introduce the selected tool via starting with a pilot projects, which has the following objectives:

- Gaining in-depth knowledge about the tool, understanding both its strenght and weaknesses

- Evaluating how the tool fits with existing processes and practices, and determining what would need to change

- Deciding on standard ways of using, managing, storing, and maintaining the tool and the test work products (deciding on naming conventions for files and tests, selecting coding standards, creating libraries and defining the modularity of test suites)

- Assessing whether the benefits will be achieved at reasonable cost

- Understanding the metrics that you wish to collect and report, and configuring the tool to ensure these metrics can be captured and reported

Success Factors for Tools

- Rolling out the tool to the rest of the organization incrementally

- Adapting and improving processes to fit with the use of the tool

- Providing training, coaching, and mentoring for tool users

- Defining guidelines for the use of the tool (internal standards for automation)

- Implementing a way to gather usage information from the actual use of the tool

- Monitoring tool use and benefits

- Providing support to the users of a given tool

- Gathering lessons learned from all users

- Ensuring that the tool is technically and organizationally integrated into the SDLC

https://github.com/fromlegaltotech