Digital Academy Thailand

Software Testing Foundation Level

Lecture 6 – Tool Support for Testing

Uwe Gühl

KASETSART UNIVERSITY

มหาวิทยาลัยเกษตรศาสตร์ วิทยาเขตศรีราชา

Kasetsart University Sriracha Campus

Contents

Contents

Test Tool Considerations

- Test tools can be used to support one or more testing activities.
- ***Test automation***:
  The use of software to perform or support test activities.

Test Tool Considerations

- Tools overview
  - Tools that are directly used in testing, like
    - test execution tools,
    - test data preparation tools.
  - Tools that help to manage
    - requirements,
    - test cases,
    - test procedures,
    - automated test scripts,
    - test results,
    - test data,
    - defects.

Test Tool Considerations

- Tools overview:
  - Tools for reporting and monitoring test execution
  - Tools that are used for
    - ➢ analysis,
    - ➢ evaluation.
  - Any tool that assists in testing
    Even a spreadsheet is a test tool in this meaning

Test Tool Classification

- Purposes of test tools
  - Improve the efficiency of test activities – especially test execution and regression testing – by automating
    - repetitive tasks,
    - tasks that require significant resources when done manually like static testing.
  - Improve the efficiency of test activities by supporting manual test activities throughout the test process

1.4

Test Tool Classification

- Purposes of test tools:
  - Improve the quality of test activities by
    - ➢ more consistent testing,
    - ➢ higher level of defect reproducibility.
  - Automate activities that cannot be executed manually
    Example: large scale performance testing
  - Increase reliability of testing; for example by
    - ➢ automating large data comparisons,
    - ➢ simulating behavior.

Test Tool Classification

- Classification based on several criteria possible
  - purpose,
  - pricing,
  - licensing model
    - commercial
    - open source
      see i. e.: http://www.opensourcetesting.org/
  - technology used.
- Tools are classified in this syllabus according to the test activities that they support.

Test Tool Classification

- Tools from a single provider, especially those that have been designed to work together, may be provided as an integrated suite, for example test management tools
  - often include a
    - ➢ requirements module,
    - ➢ defect management tool;
  - offer interfaces to a test automation or load and performance testing tool.

Test Tool Classification

- Some types of test tools can be intrusive, causing the ***probe effect***: An unintended change in behavior of a component or system caused by measuring it.
  - The actual outcome of the test might be affected, e.g.,
    - Performance test
      Actual response times for an application may be different due to the extra instructions that are executed by the tool
    - Code coverage
      The amount of code coverage achieved may be distorted due to the use of a coverage tool.
- Tools more appropriate for developers, e.g., tools that are used during component and component integration testing, are marked with **Dev**

Tool support for management of testing and testware

- Management tools
  may apply to any test activities over the entire software development lifecycle; examples
  - Test management tools and application lifecycle management tools (ALM)
  - Requirements management tools
    (e.g., traceability to test objects)
  - Defect management tools
  - Configuration management tools
  - Continuous integration tools **Dev**

Tool support for management of testing and testware

- ***Test management tools***:
  A tool that supports test management.
  - often offer interfaces to
    - ➢ produce useful information
      in a format that fits the needs of the organization
    - ➢ maintain consistent traceability to requirements
      in a requirements management tool
    - ➢ link with test object version information
      in the configuration management tool
- Integrated tools like Application Lifecycle Management typically integrate better distinct modules than modules from third-party supplier

Tool support for static testing

- Examples for a static testing tool:
  - Review Tools
    - ➢ Used to
      - ❖ store and communicate review comments,
      - ❖ report on defects,
      - ❖ report on effort.
    - ➢ Support with
      - ❖ review processes,
      - ❖ check lists,
      - ❖ review guidelines,
      - ❖ online reviews for large or geographically dispersed teams.

3

Tool support for static testing

- Examples for a static testing tool:
  - Static analysis tools **Dev**
    - ➢ help to find defects by
      - ❖ providing support for enforcing coding standards – including secure coding,
      - ❖ analysis of structures and dependencies.
    - ➢ can help in planning or risk analysis by providing metrics for the code like complexity
  - Modelling tools **Dev**
    - ➢ used to validate software models by
      - ❖ enumerating inconsistencies,
      - ❖ finding defects.
    - ➢ Example: Validation of a physical data model for a relational database

Tool support for test design and implementation

- Test design tools aid in the creation of maintainable work products in test design and implementation, including
  - test cases,
  - test procedures,
  - test data, considering data anonymity.
- Tools that support test design and implementation sometimes
  - support test execution and logging,
  - provide their outputs directly to other tools that support test execution and logging.
- Examples:
  - Test data preparation tools
  - Model-Based testing tools *(next slide)*

Tool support for test design and implementation

- Model-Based testing (MBT) tools
  - enable a functional specification to be captured in the form of a *model*, such as an activity diagram.

generally performed by a system designer.

  - interpret the *model* in order to create test case specifications to be
    - ➢ saved in a test management tool and/or
    - ➢ executed by a test execution tool.

Tool support for test execution and logging

- ***Test execution tools***:
  A test tool that executes tests against a designated test item and evaluates the outcomes against expected results and postconditions.
  - Areas covered:
    - ➢ smoke test,
    - ➢ setup tests,
    - ➢ configuration tests,
    - ➢ non-GUI tests (interfaces),
    - ➢ regression tests.

Tool support for test execution and logging

- Test execution tools
  - execute test objects using automated test scripts.
  - often require significant effort in order to achieve significant benefits.
  - Capturing test approach (*Capture & replay*)
    - ➢ Capturing tests by recording the actions of a manual tester
    - ➢ Considerations
      - ❖ **Scaling:** does not scale to large numbers of test scripts.
      - ❖ **Unstable:** a captured script is a linear representation with specific data and actions as part of each script:
        may be unstable when unexpected events occur
      - ❖ **Maintenance:** requires ongoing maintenance as the system's user interface evolves over time.

Tool support for test execution and logging

- Test execution tools
  - Data-driven test approach
    - ➢ separates out the test inputs and expected results, usually into a spreadsheet,
    - ➢ uses a more generic test script that can read the input data and execute the same test script with different data.
  - ***Data-driven testing***:
    A scripting technique that uses data files to contain the test data and expected results needed to execute the test scripts.

Tool support for test execution and logging

- Test execution tools
  - Keyword-driven test approach: a generic script
    - ➢ processes keywords (also called action words), describing the actions to be taken,
    - ➢ calls related keyword scripts to process the associated test data.
  - ***Keyword-driven testing***
    (***Synonym****: action word-driven testing*):
    A scripting technique in which test scripts contain high-level keywords and supporting files that contain low-level scripts that implement those keywords.

Tool support for test execution and logging

- Test execution tools
  - Both, data-driven test approach and keyword-driven test approach,
    - ➢ require expertise in the scripting,
    - ➢ could be supported by testers who are not familiar with the scripting language by creating
      - ❖ test data and/or
      - ❖ keywords.
    - ➢ need a comparison of the expected results to the actual results for each test,
      - ❖ dynamically – while the test is running or
      - ❖ stored for later - post-execution - comparison.

Tool support for test execution and logging

- Coverage tools **Dev**
  - requirements coverage,
  - code coverage
- Test harnesses **Dev**

Tool support for performance measurement and dynamic analysis

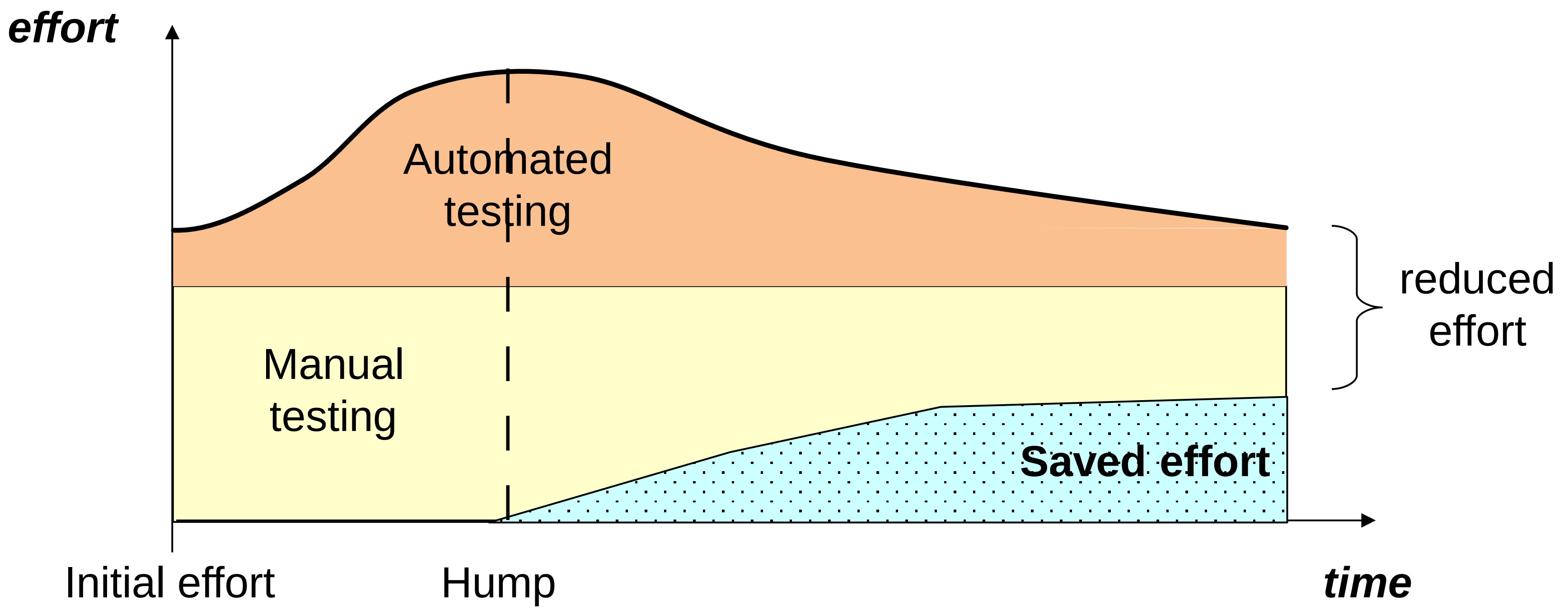
- Performance measurement and dynamic analysis tools
  - essential in supporting performance and load testing activities,
  - activities cannot effectively be done manually.
- Examples:
  - Performance testing tools
  - Dynamic analysis tools Dev

Tool support for specialized testing needs

- Other tools
  - tools that support more specific testing for non-functional characteristics, like for example
    - ➢ tools for security tests
    - ➢ data quality assessment related on
      - ❖ data conversion
      - ❖ data migration
    - ➢ usability testing tools
    - ➢ monitoring tools that give warnings of possible service problems

Benefits and Risks of Test Automation

- Idea, especially for test execution tools: The cost of automation is offset by the savings received from automation



Benefits and Risks of Test Automation

- Potential benefits
  - Reduction in repetitive manual work like
    - ➢ running regression tests,
    - ➢ environment set up/tear down tasks,
    - ➢ re-entering the same test data,
    - ➢ checking against coding standards).

    $\Bigg\}$ *Saving time*
  - More objective assessment, e.g.,
    - ➢ static measures,
    - ➢ coverage.

Benefits and Risks of Test Automation

- Potential benefits
  - Greater consistency and repeatability, e.g.,
    - ➢ test data created in a coherent manner,
    - ➢ tests are executed by a tool in the same order with the same frequency,
    - ➢ tests are consistently derived from requirements,
    - ➢ could be executed over night.
  - Easier access to information about testing, e.g.,
    - ➢ statistics and graphs about test progress,
    - ➢ defect rates and performance.

Benefits and Risks of Test Automation

- Potential risks
  - Expectations for the tool may be unrealistic including functionality and ease of use
  - Under-estimated factors
    - for the initial introduction of a tool
      - time,
      - cost,
      - effort,
      - training (for example for a proprietary scripting language)
      - required external expertise.
    - to achieve significant and continuing benefits
      - time,
      - effort,
      - need for changes in the test process and continuous improvement.

Benefits and Risks of Test Automation

- Potential risks
  - The effort required to maintain the test work products generated by the tool may be underestimated
  - Potential misunderstandings
    - ➢ Tool is seen as a replacement for test design or execution
    - ➢ Use of automated testing where manual testing would be better
  - Version control of test work products may be neglected
  - Introduction of new possible defect sources

Benefits and Risks of Test Automation

- Potential risks
  - Relationships and interoperability issues between critical tools may be neglected, such as
    - ➢ requirements management tools,
    - ➢ configuration management tools,
    - ➢ defect management tools,
    - ➢ tools from multiple vendors.
  - The tool vendor may
    - ➢ go out of business,
    - ➢ retire the tool,
    - ➢ sell the tool to a different vendor.

Benefits and Risks of Test Automation

- Potential risks
  - The vendor may provide a poor response for
    - ➢ support,
    - ➢ upgrades,
    - ➢ 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, e.g., for
    - ➢ mentoring,
    - ➢ updates.

Summary

- Main purpose of test tools is to automate
  - repetitive tasks,
  - tasks that could not be done manually.
- Tool support for different test activities possible
  - Tool support for management of testing and testware
  - Tool support for static testing
  - Tool support for test design and implementation
  - Tool support for test execution and logging
  - Tool support for performance measurement and dynamic analysis
  - Tool support for specialized testing needs

Summary

- **Data-driven test approach**:
  data used in a test script are separated, usually into a spreadsheet
- **Keyword-driven test approach**:
  keywords describe the actions to be taken
- Test automation tools
  - offer lots of benefits, a lot of cost savings is possible
  - risks to be considered like maintenance effort, too high expectations that could not be fulfilled

Contents

Main Principles for Tool Selection

- Several facts should be considered in selecting a tool for an organization – main considerations are listed below
- Considerations
  - Assessment of the maturity of the own organization
    - ➢ strengths
    - ➢ weaknesses
  - Current test process
    Identification of opportunities for an improved test process supported by tools

Main Principles for Tool Selection

- Considerations
  - Technologies used by the test object(s)
    - ➢ to select a tool that is compatible with that technology
  - Current build and continuous integration
    - ➢ to ensure tool compatibility and integration
  - Evaluation of the tool against
    - ➢ clear requirements,
    - ➢ objective criteria.
  - Tool available for a free trial period?
    For how long?

Main Principles for Tool Selection

- Considerations
  - Pros and cons of various licensing models like
    - Commercial tools: evaluation of the vendor including
      - training,
      - support,
      - commercial aspects.
    - Open source tools: evaluation of support
  - Coaching and mentoring
    Identification of internal requirements for the use of the tool

Main Principles for Tool Selection

- Considerations
  - Evaluation of training for people using the tool
    - testing skills
    - test automation skills
  - Estimation of a cost-benefit ratio based on a concrete business case (if required)

Pilot Projects for Introducing a Tool into an Organization

- Recommended proceeding:
  - Complete the tool selection: recommendation and decision
  - Conduct proof-of-concept, confirm its success
  - Introduce the selected tool with a pilot project

Pilot Projects for Introducing a Tool into an Organization

- Objectives of a pilot project:
  - Gaining in-depth knowledge about the tool
    - ➢ Strengths
    - ➢ Weaknesses
  - Test process
    - ➢ Evaluating how the tool fits with existing processes and practices
    - ➢ Determining what would need to change

Pilot Projects for Introducing a Tool into an Organization

- Objectives of a pilot project
  - Work procedure
    Deciding on standard ways of using, managing, storing, and maintaining the tool and the test work products like
    - ➢ deciding on naming conventions for files and tests,
    - ➢ selecting coding standards,
    - ➢ creating libraries,
    - ➢ defining the modularity of test suites.

Pilot Projects for Introducing a Tool into an Organization

- Objectives of a pilot project
  - Assessing whether the benefits will be achieved at reasonable cost
  - Reporting
    - ➢ How to configure the tool to get metrics 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 for tool users
  - training,
  - coaching,
  - mentoring.
- Guidelines
  for the use of the tool like internal standards for automation

Success Factors for Tools

- Implementing a way to gather usage information from the actual use of the tool
- Monitoring tool use and benefits
- Providing support to the users
- Gathering lessons learned from all users
- Ensuring that the tool is technically and organizationally integrated into the software development lifecycle,
- Clarifying responsibility for operations
  - separate organizations and/or
  - third party suppliers

Summary

- Test process maturity is more important than introducing a tool
- Considerations for a tool selection
  - Commercial/open source
  - Evaluation with clear requirements and objective criteria
- A pilot project recommended to introduce a tool
  - Test process to be adapted
  - Work procedure to describe the tool usage
- Success factors for tools include
  - incremental rolling out,
  - extensive support for the users,
  - clear responsibilities for support.