

INF3121 : Software Testing

26. 02. 2015

Lecture 6

Tool support for testing



**UNIVERSITETET
I OSLO**

Lecturer: Raluca Florea

Overview

1. Types of test tools
2. Effective use of tools: benefits / risks
3. Introducing a tool into an organization



- 1.1 Test tool classification
- 1.2 Tool support for management of testing & tests
- 1.3 Tool support for static testing
- 1.4 Tool support for test specification
- 1.5 Tool support for test execution and logging
- 1.6 Tool support for performance and monitoring
- 1.7 Tool support for specific application areas
- 1.8 Tool support using other tools

1. Types of test tools



1.1 Test tool classification

- Tools are classified in this course according to the testing activities that they support.
 - one activity;
 - more than one activity, but classification falls under the main activity
- Testing tools can improve:

Efficiency of testing

Reliability of testing

by automating repetitive tasks.

by automating large data comparisons or simulating behavior.

Notes:

1. Some types of test tool can be intrusive - the tool itself can affect the outcome of the test. (i.e. timing measurements may be different depending on how you measure it with different performance tools).

The consequence of intrusive tools is called the probe effect.

2. Some tools offer support more appropriate for developers. Such tools are marked with “(D)” in this chapter.



1.2 Tools support for management of testing & tests

Characteristics :

Support for the management of tests and the testing activities.

Interfaces to:

- test execution tools
- defect tracking tools
- requirement management tools.

Support for traceability of tests, test results and incidents to source documents, such as requirements specifications.

Generation of progress reports.

Logging test results.

Offer info on metrics related to the tests.



1.2 Tools support for management of testing & tests

Tools:

Requirements management tools

- store requirements
- check for consistency and undefined (missing) requirements
- allow prioritization
- enable individual tests to be traceable to requirements

Incident management tools

- store and manage incident reports
- support management of incident reports

Configuration management tools

- are necessary to keep track of different versions and builds of the SW and tests
- are particularly useful when developing on more than one configuration of the HW/SW environment



1. Types of test tools

•Requirements management tools

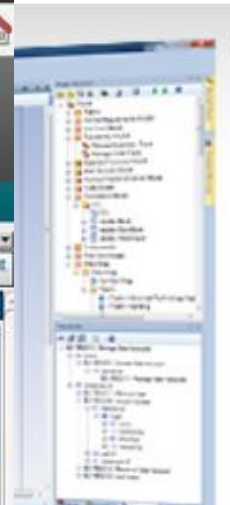
The screenshot displays the TestLink 1.8.5 web interface in a Mozilla Firefox browser. The browser's address bar shows the URL <https://newhope.labranet.jamk.fi/ProjectTESTLINK/>. The page header includes the 'nest project platform' logo and navigation tabs for Home, Knowledge Sharing, Communication, Work Collaboration, and Administration. Below this, there are sub-tabs for Tickets & Tasking, Project Management, Error Management, Change Management, Test Management, and SCM Statistics. The main content area is titled 'TestLink 1.8.5 : AdminUser [admin]' and shows a breadcrumb trail: Home | Specification | Execute | Results | Users | Events | Test Case ID: TC_-. The left sidebar contains a 'Navigator - Test Specification' tree with a 'Filter & Settings' section. The tree shows a hierarchy of test plans, including 'IFDK Project (112)' and 'Calory Counter(11)'. The selected test case is 'TC_-2:Reseting calory counter from UI'. The main content area for the test case shows a message: 'You can not edit this version because it has been executed'. Below this, the test case details are displayed, including the title 'TC_-2:Reseting calory counter from UI', version information, summary, and a table of steps and expected results.

Test Case Details:

- Title:** TC_-2:Reseting calory counter from UI
- Version 1:** Created on 10/11/2010 19:34:01 by AdminUser
- Summary:** Wristband + Vibrator/bare hands
- Steps and Expected Results:**

| Steps | Expected Results |
|---|---|
| <ul style="list-style-type: none">Open Calory Meter tabSelect "Reset Calory Meter"Check value on screen (should be 0 calh)Activate Play mode and use some energy or use vibrator for 20 secondscheck calory counter value | <p>PASS Criteria: Value are counting from zero after reset</p> <p>FAIL criteria: Values are not resetting correctly</p> |
- Execution type:** Manual
- Test importance:** Medium
- Keywords:** None
- Requirements:** [IFDK System Requirements] UserStoryId5003:UserStoryId5003
- Test Plan usage:**

| Version | Test Plan |
|---------|---------------------------|
| 1 | EXAMPLE TEST PLAN v0.1 |
| 1 | IFDK System Test Plan 0.1 |
| 1 | GroupCancerAxe |
| 1 | Team2 |
| 1 | FarEasternFirePig |



Purchase

•Incident management tools

1. Types of test tools

The screenshot displays the Microsoft Visual Studio Team System Web Access interface. The browser address bar shows the URL `http://tfsrtmsp1:8090/index.aspx?pid=7`. The interface includes a navigation menu with tabs for Home, Work Items, Reports, Documents, Source, and Build. A dropdown menu for 'New Work Item' is open, listing options: Bug, Task, Issue, Change Request, Risk, Requirement, and Review. The main content area is divided into several sections:

- Search:** A search bar with the text 'Enter search text' and a 'Search' button.
- Favorites:** A list of favorite items including 'Active Bugs' and 'Development Tasks'.
- New Work Item:** A list of new work item types including 'Bug', 'Task', 'Change Request', 'Requirement', and 'Risk'.
- My Queries:** A section indicating 'No recent items.'
- Team Queries:** A list of team queries including 'Active Bugs', 'Development Tasks', 'All Work Items', 'All My Team Project Work Items', and 'All Tasks'.
- WI's Assigned to Me:** A section showing work items assigned to the user, including '4 Bug' (1 Active, 3 Resolved), '2 Task' (2 Active), '1 Risk' (1 Active), '2 Requirement' (2 Proposed), and '1 Change Request' (1 Active).
- Recently Accessed Work Items:** A table listing recently accessed work items with columns for ID, type, status, and description.

| ID | Type | Status | Description |
|----|----------------|----------|---|
| 41 | Task | Active | Implement support for Integrated authentic... |
| 40 | Change Request | Active | Support Integrated authentication |
| 39 | Requirement | Proposed | Application should work with limited disk sp... |
| 38 | Requirement | Proposed | About page displays the version information |
| 37 | Risk | Active | Temporary files can fill up the disk |

The status bar at the bottom shows 'Done', 'Local intranet', and '100%' zoom. A summary bar at the very bottom indicates 'Total Records: 5 Type: TMSestCase Selected: 5'.



•Configuration management tools

1. Types of test tools



History

Configurat
managem
became it
series" (i.
into a sing
reduced th
(SDO). [8]
649-1998
(SE), inte
methodolo
managem
Some trea
discipline.

Ansible
built in
your o

[edit]

1950s as a technical
CM process
called the "480
was consolidated
DoD goal that
g Organizations
d on CM, ANSI-EIA-
systems engineering
management
ation lifecycle
cal management.
parate or stand alone



1.3 Tools support for static testing

Review tools

- store information about review processes,
- store and communicate review comments, report on defects and effort,
- They can provide aid for online reviews, which is useful if the team is geographically dispersed.

Static analysis tools (D)

- support developers, testers and quality assurers in finding defects before dynamic testing.
- Major purposes :
 - The enforcement of coding standards.
 - The analysis of structures and dependencies (e.g. linked web pages).
 - Aiding in understanding the code.
- Static analysis tools can calculate metrics from the code (e.g. complexity), which can give valuable information for planning or risk analysis.

Modeling tools (D)

- Validate models of the software.

The major benefit of static analysis tools and modeling tools is the cost effectiveness of finding more defects at an earlier time in the development process. As a result, the development process may accelerate and improve by having less rework

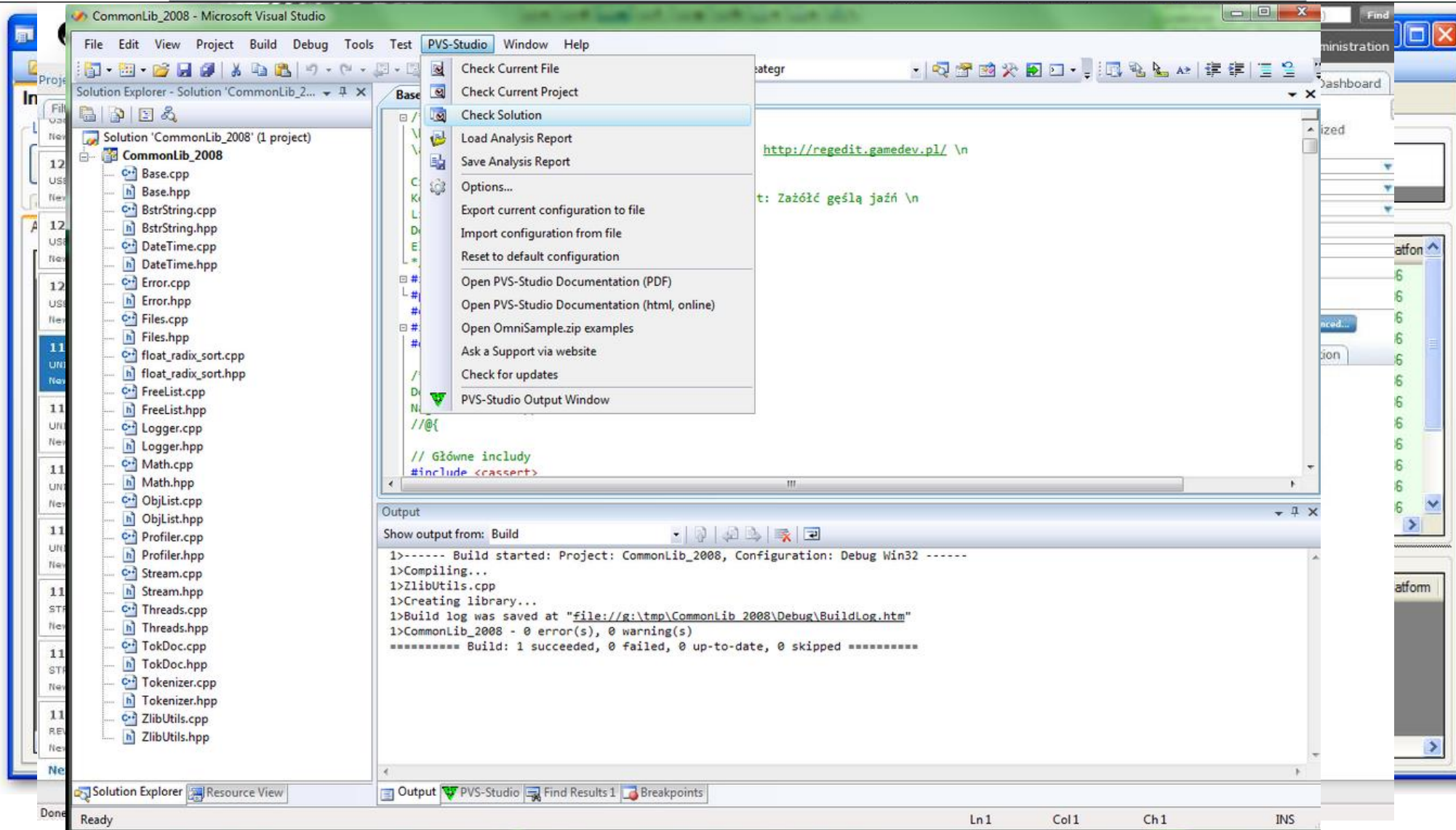
•Review tools

1. Types of test tools

The screenshot shows the Review Board 1.6.3 dashboard. The page title is "Review Board 1.6.3" and the URL is "reviews.reviewboard.org/dashboard/". The dashboard includes a navigation menu with "My Dashboard", "New Review Request", "All review requests", "Groups", and "Submitters". A sidebar on the left shows review counts: "Starred Reviews" (0), "Outgoing Reviews" (0), "Incoming Reviews" (0), "To Me" (0), "reviewboard" (62), and "All My Requests" (0). The main content area is titled "All Incoming Review Requests" and contains a table with the following data:

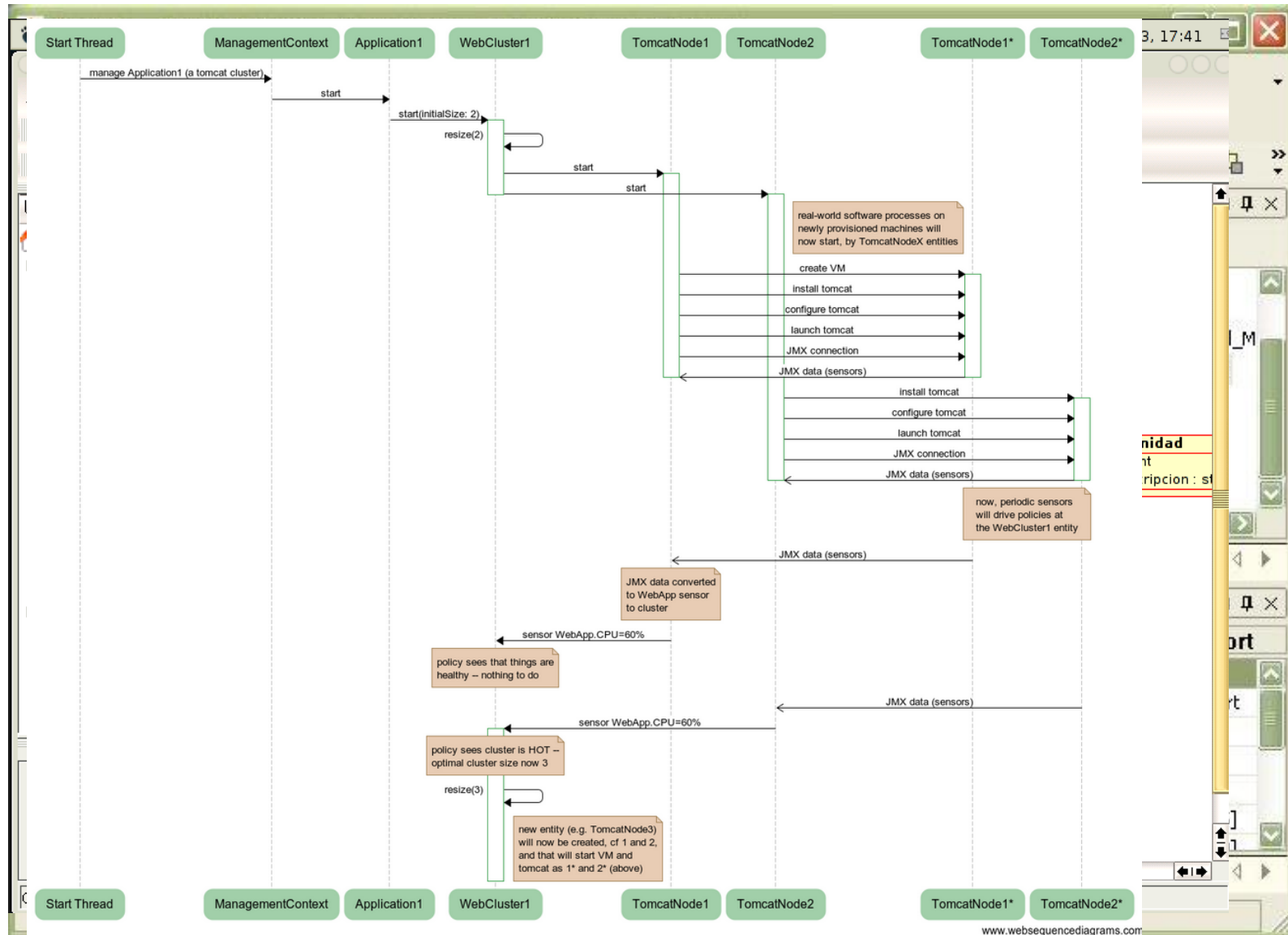
| Summary | Submitter | Posted | Last Updated |
|--|-------------|--------------------------------|----------------------|
| ☆ Response to feature 2282 | wilsonyeung | January 28th, 2012, 3:21 p.m. | 12 hours ago |
| ☆ Added issue tracking integration for drafts and review requests. | ammok | February 11th, 2012, 9:12 p.m. | 1 day ago |
| ☆ Added Show Links for Interdiffs | cim1 | January 21st, 2012, 2:22 p.m. | 1 day, 1 hour ago |
| ☆ Added * flag to required review request fields | medanat | February 13th, 2012, 5:43 p.m. | 1 day, 1 hour ago |
| ☆ WebHooks Extension [WIP] | smacleod | February 10th, 2012, 6:28 p.m. | 1 day, 12 hours ago |
| ☆ Navigation and action hooks missing extension parameter | bartek | February 1st, 2012, 3:49 a.m. | 2 days, 18 hours ago |
| ☆ Created a new script for generating scaffolding for new extensions. | ammok | January 28th, 2012, 5:05 p.m. | 3 days, 17 hours ago |
| ☆ Reviewboard-Social Extension | cim1 | February 4th, 2012, 8:36 a.m. | 6 days, 8 hours ago |
| ☆ Switch to using middleware for initialization. | chipx86 | January 30th, 2012, 3:09 a.m. | 1 week, 2 days ago |
| ☆ Remove all usage of ifuserorperm/ifnotuserandperm. | chipx86 | January 30th, 2012, 3:37 a.m. | 1 week, 2 days ago |
| ☆ Added support for authentication against Active Directory subdomains. | aineko | January 31st, 2012, 5:48 a.m. | 2 weeks ago |
| ☆ Use Django's smarter if tag. | chipx86 | January 30th, 2012, 3:11 a.m. | 2 weeks, 1 day ago |
| ☆ Add optional enable_highlighting parameter in get_file_chunks_in_range | kamlani | April 5th, 2011, 8:16 p.m. | 2 weeks, 3 days ago |
| ☆ Localized timezone support (Review Board) | ddruska | December 1st, 2011, 4:01 p.m. | 2 weeks, 6 days ago |
| ☆ Fix authentication issue with perforce repository | stid | December 24th, 2011, 5:34 a.m. | 1 month, 3 weeks ago |





CodeSonar chart summarizing warning locations and classes for a project.





1.4 Tools support for test specification

Test design tools

- Generate test inputs or executable tests
 - from requirements,
 - from a graphical user interface,
 - from design models (state, data or object)
 - or from code.
- This type of tool may generate expected outcomes as well (i.e. may use a test oracle)
- They can save valuable time and provide increased thoroughness of testing because of the completeness of the tests that the tool can generate.

Test data preparation tools

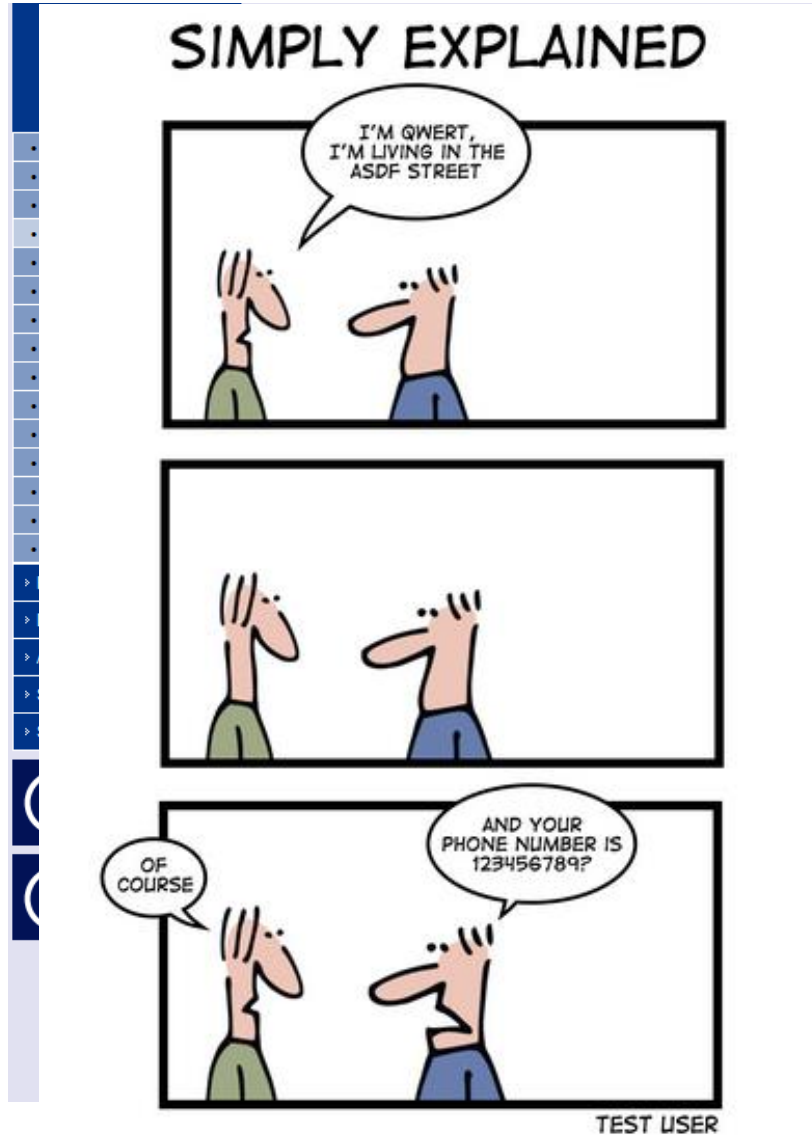
- manipulate databases or files to set up test data to be used during the execution of tests
- benefit: they ensure that live data in a test environment is made anonymous, for data protection.

•Test design tools

1. Types of test tools

| # | Operating System | Language | Browser | # | Operating System | Language | Browser |
|----|--------------------|----------------|----------------------|----|--------------------|----------------|----------------------|
| 1 | Windows 8 64bit | French | Chrome latest | 28 | Ubuntu 12.10 | German | Opera latest |
| 2 | Windows 7 Ultimate | Italian | Internet Explorer 8 | 29 | Ubuntu 12.10 | Spanish | Chrome latest |
| 3 | Windows XP home | French | Internet Explorer 8 | 30 | Ubuntu 12.10 | Chinese Simpl. | Opera latest |
| 4 | Ubuntu 12.10 | French | Chrome latest | 31 | Ubuntu 12.10 | Chinese Trad. | Chrome latest |
| 5 | Windows 8 64bit | German | Internet Explorer 10 | 32 | Ubuntu 12.10 | Japanese | Firefox latest |
| 6 | Windows 8 64bit | Spanish | Safari latest | 33 | Ubuntu 12.10 | Korean | Firefox latest |
| 7 | OS X 10.8 | Chinese Simpl. | Chrome latest | 34 | OS X 10.8 | French | Safari latest |
| 8 | Windows 8 64bit | Chinese Trad. | Firefox latest | 35 | OS X 10.8 | Italian | Firefox latest |
| 9 | Windows 8 64bit | Japanese | Opera latest | 36 | OS X 10.8 | German | Opera latest |
| 10 | Windows 8 64bit | Korean | Internet Explorer 10 | 37 | OS X 10.8 | Spanish | Opera latest |
| 11 | Windows 8 64bit | Italian | Internet Explorer 10 | 38 | OS X 10.8 | Chinese Trad. | Safari latest |
| 12 | Windows 8 64bit | Chinese Simpl. | Internet Explorer 10 | 39 | OS X 10.8 | Japanese | Safari latest |
| 13 | Windows 7 Ultimate | French | Internet Explorer 10 | 40 | OS X 10.8 | Korean | Safari latest |
| 14 | Windows 7 Ultimate | German | Chrome latest | 41 | Windows XP home | French | Firefox latest |
| 15 | Windows 7 Ultimate | Spanish | Safari latest | 42 | Ubuntu 12.10 | French | Opera latest |
| 16 | Windows 7 Ultimate | Chinese Simpl. | Firefox latest | 43 | Windows 7 Ultimate | Italian | Safari latest |
| 17 | Windows 7 Ultimate | Chinese Trad. | Opera latest | 44 | OS X 10.8 | Italian | Opera latest |
| 18 | Windows 7 Ultimate | Japanese | Internet Explorer 8 | 45 | Windows 7 Ultimate | German | Internet Explorer 8 |
| 19 | Windows 7 Ultimate | Korean | Internet Explorer 8 | 46 | Windows 8 64bit | German | Firefox latest |
| 20 | Windows XP home | Italian | Chrome latest | 47 | Windows 7 Ultimate | Spanish | Internet Explorer 8 |
| 21 | Windows XP home | German | Safari latest | 48 | Windows 8 64bit | Spanish | Internet Explorer 10 |
| 22 | Windows XP home | Spanish | Firefox latest | 49 | Windows XP home | Chinese Simpl. | Internet Explorer 8 |
| 23 | Windows XP home | Chinese Simpl. | Opera latest | 50 | OS X 10.8 | Chinese Simpl. | Safari latest |
| 24 | Windows XP home | Chinese Trad. | Internet Explorer 8 | 51 | Windows 7 Ultimate | Chinese Trad. | Internet Explorer 10 |
| 25 | Windows XP home | Japanese | Chrome latest | 52 | Windows 8 64bit | Japanese | Internet Explorer 10 |
| 26 | Windows XP home | Korean | Chrome latest | 53 | Windows 8 64bit | Korean | Opera latest |
| 27 | Ubuntu 12.10 | Italian | Firefox latest | | | | |





generated by the tool.

| Id | Country | Phone | Fax |
|----|------------|------------------|---------------|
| 1 | 'Brazil' | NULL | '(926) 084-51 |
| 2 | 'Thailand' | '(853) 132-8154' | '(892) 349-78 |
| 3 | 'Tunisia' | '(729) 151-9889' | '(164) 583-66 |
| 4 | 'Dominica' | '(943) 332-5127' | '(917) 302-86 |
| 5 | 'Haiti' | '(664) 150-7205' | '(304) 306-20 |
| 6 | 'Rwanda' | '(563) 506-3486' | '(516) 163-37 |
| 7 | 'Haiti' | '(115) 504-5632' | '(134) 381-36 |
| 8 | 'Liberia' | '(145) 726-2437' | '(660) 874-86 |
| 9 | 'Colombia' | '(301) 818-7892' | '(512) 900-21 |
| 10 | 'Kiribati' | '(402) 975-9323' | '(488) 833-41 |

Order

Attributes



1.5 Tools support for test execution & logging

Test execution tools

- Enable tests to be executed automatically using stored inputs & expected outcomes
- The scripting language allows to manipulate the tests with little effort (i.e. repeat the test with other data)
- Can also be used to record tests (capture & playback tools)

Test harness/unit test framework tools (D)

- Facilitate the test of components of a system - simulating the environment in which that test object will run.
- They may be called unit test tools when they have a particular focus on the component test level.

Test comparators

- Determine differences between files, databases or test results
- Test execution tools include dynamic comparators, but post-execution comparison may be done by a separate comparison tool.
- A test comparator may use a test oracle, especially if it is automated.

Coverage measurement tools (D)

- Can be intrusive or non intrusive (depends on the measurement technique used)
- Measure the percentage of specific types of code structure that have been exercised

Security tools

- Search for specific vulnerabilities of the system



•Test execution tools

1. Types of test tools

Test Results

Results: 17/17 passed; Item(s) checked: 0

| Result | Test Name | Project | Error Message | Duration |
|--------|--|--------------------|---------------|------------------|
| Passed | LogOff_LogsOutAndRedirects | Big.Fat.Dish.Tests | | 00:00:00.0906167 |
| Passed | Register_Post_ReturnsViewIfRegistrationFails | Big.Fat.Dish.Tests | | 00:00:00.0120230 |
| Passed | Register_Post_ReturnsViewIfModelStateInvalid | Big.Fat.Dish.Tests | | 00:00:00.0007535 |
| Passed | LogOn_Post_ReturnsViewIfModelStateInvalid | Big.Fat.Dish.Tests | | 00:00:00.0014254 |
| Passed | LogOn_Get_ReturnsView | Big.Fat.Dish.Tests | | 00:00:00.0005725 |
| Passed | ChangePassword_Post_ReturnsViewIfChangePasswordFails | Big.Fat.Dish.Tests | | 00:00:00.0105882 |
| Passed | Register_Post_ReturnsRedirectOnSuccess | Big.Fat.Dish.Tests | | 00:00:00.0006481 |
| Passed | Index | Big.Fat.Dish.Tests | | 00:00:00.0008837 |
| Passed | LogOn_Post_ReturnsRedirectOnSuccess_WithReturnUrl | Big.Fat.Dish.Tests | | 00:00:00.0017549 |
| Passed | LogOn_Post_ReturnsRedirectOnSuccess_WithoutReturnUrl | Big.Fat.Dish.Tests | | 00:00:00.0005188 |
| Passed | ChangePassword_Post_ReturnsViewIfModelStateInvalid | Big.Fat.Dish.Tests | | 00:00:00.0004918 |
| Passed | ChangePasswordSuccess_ReturnsView | Big.Fat.Dish.Tests | | 00:00:00.0004222 |
| Passed | Register_Get_ReturnsView | Big.Fat.Dish.Tests | | 00:00:00.0005678 |
| Passed | ChangePassword_Get_ReturnsView | Big.Fat.Dish.Tests | | 00:00:00.0005879 |
| Passed | LogOn_Post_ReturnsViewIfValidateUserFails | Big.Fat.Dish.Tests | | 00:00:00.0005958 |
| Passed | ChangePassword_Post_ReturnsRedirectOnSuccess | Big.Fat.Dish.Tests | | 00:00:00.0007185 |
| Passed | About | Big.Fat.Dish.Tests | | 00:00:00.0004423 |



The screenshot shows an IDE window with a workspace named 'TFSRTM08'. The main editor displays a C# unit test method named 'RhinoMocksDemo()' within a 'Fo' namespace. The code includes mock creation, expectation setting, and test execution. A 'Test References' pane on the left shows 'UnitTests.Subtext'. On the right, a 'dependencies' table is visible.

```

[Test]
So
public void RhinoMocksDemo()
Fo{
    MockRepository mocks = new MockRepository();

    // create some mocks
    IEmailBuilder emailBuilder = mocks.CreateMock<IEmailBuilder>();
    IEmailSender emailSender = mocks.CreateMock<IEmailSender>();

    Email email = new Email();

    // set expectations
    Expect.Call(emailBuilder.Create()).Return(email);
    emailSender.Send(email);

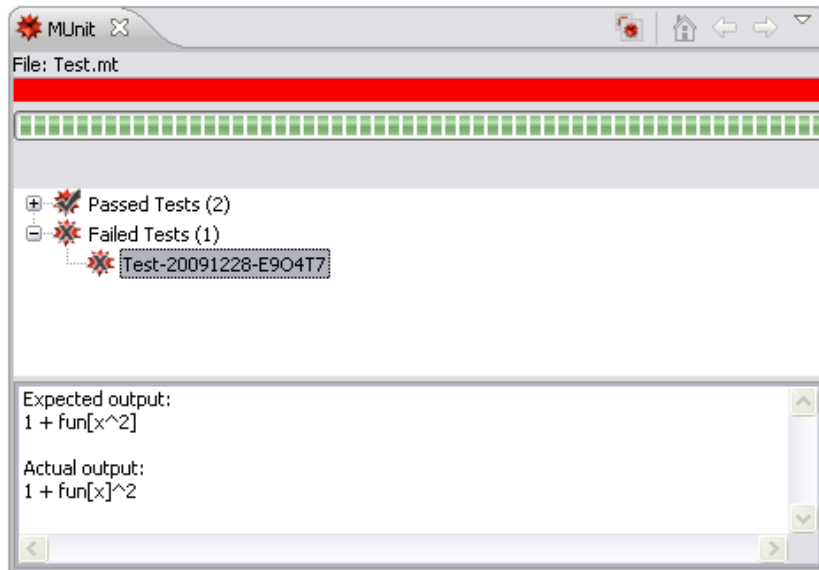
    // execute the test
    mocks.ReplayAll();
    Reporter reporter = new Reporter(emailSender, emailBuilder);
    reporter.SendSomeEmails();
    mocks.VerifyAll();
}
    
```

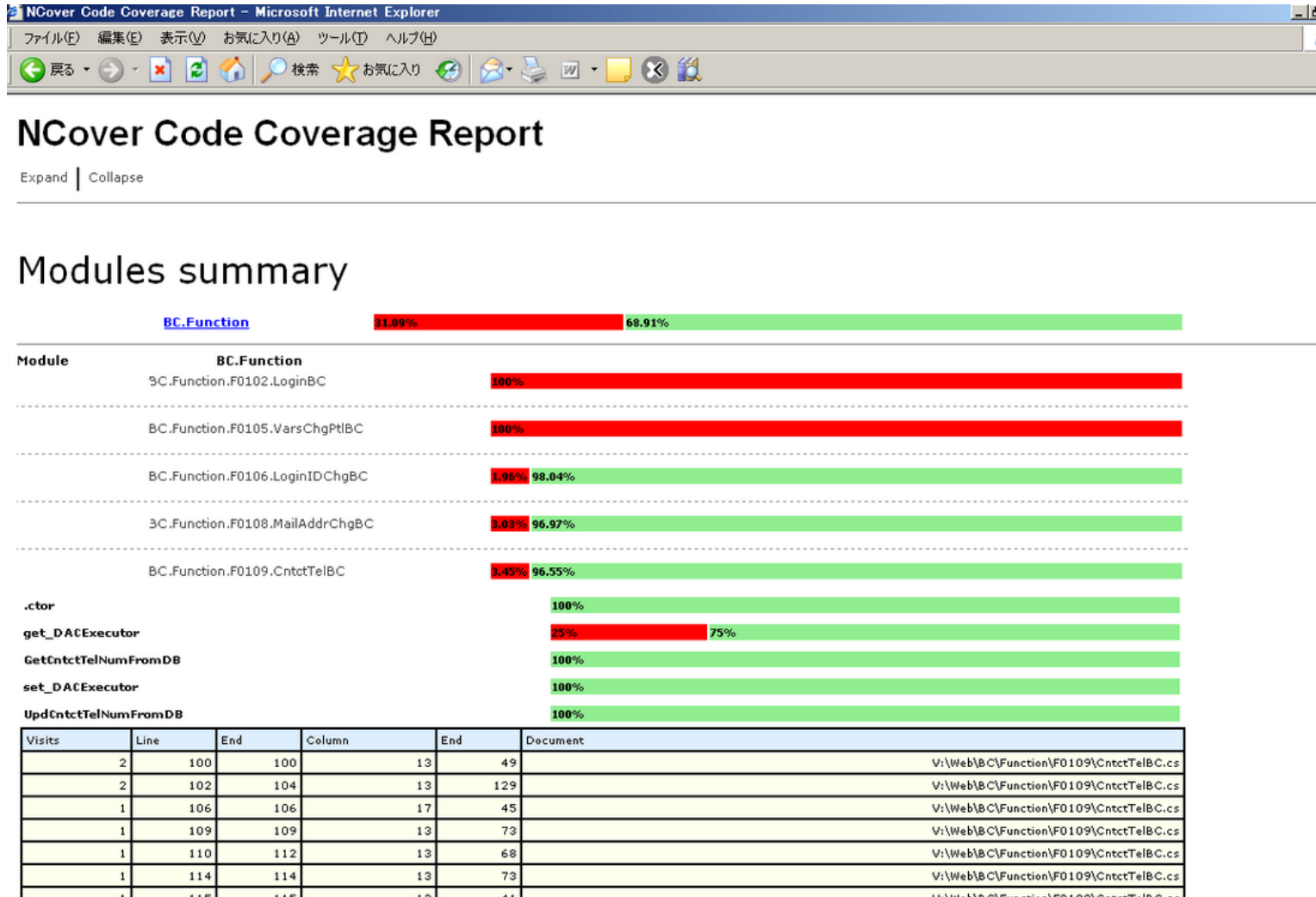
| dependencies | |
|--------------|--------|
| | Latest |
| | Yes |
| stor | Yes |
| stor | Yes |

The Test Comparator

The test comparator is a useful feature for seeing exactly what the error is in a test failure.

First, you should run a [test file](#) or [test suite](#). Then select one of the test failures: this should show the expected and actual results in the test report. A sample is shown in the following.





•Security tools

Modules

- Title Screen
- Firewall
- FilePermissions
- AccountSecurity
- BootSecurity
- Securelnetd
- DisableUserTools
- ConfigureMiscPAM
- Logging
- MiscellaneousDaemons
- DNS
- Apache
- Printing
- FTP
- TMPDIR
- End Screen

Question

Explanation

(Tk User Interface)

v1.2.0.prerelease

Please answer all the questions to build a more secure system.

The Next and Back buttons navigate forward and backward in the questions database. Changes made in the Answer field are *only* saved when you push the Next button! The "modules" in the questions database are listed to the left. You can jump to the start of any module simply by clicking on its name.

Some questions have two levels of explanatory text, which you can adjust with the Explain Less/More button.

Please address bug reports and suggestions to jay@bastille-linux.org
Bugs in the Tk user interface are the fault of allenp@nwlinc.com.

Answer

Back Next Explain Less

1.6 Tools support for performance & monitoring

Dynamic analysis tools (D)

- find defects that appear only when software is executing (i.e. memory leaks)
- They are typically used in component and component integration testing

Performance testing/load testing/stress testing tools

- Monitor and report on how a system behaves under a variety of simulated usage conditions.
- They simulate a load on:
 - an **application**, a **database**, or a **system environment**.
- They are often based on automated repetitive execution of tests, controlled by parameters.

Monitoring tools

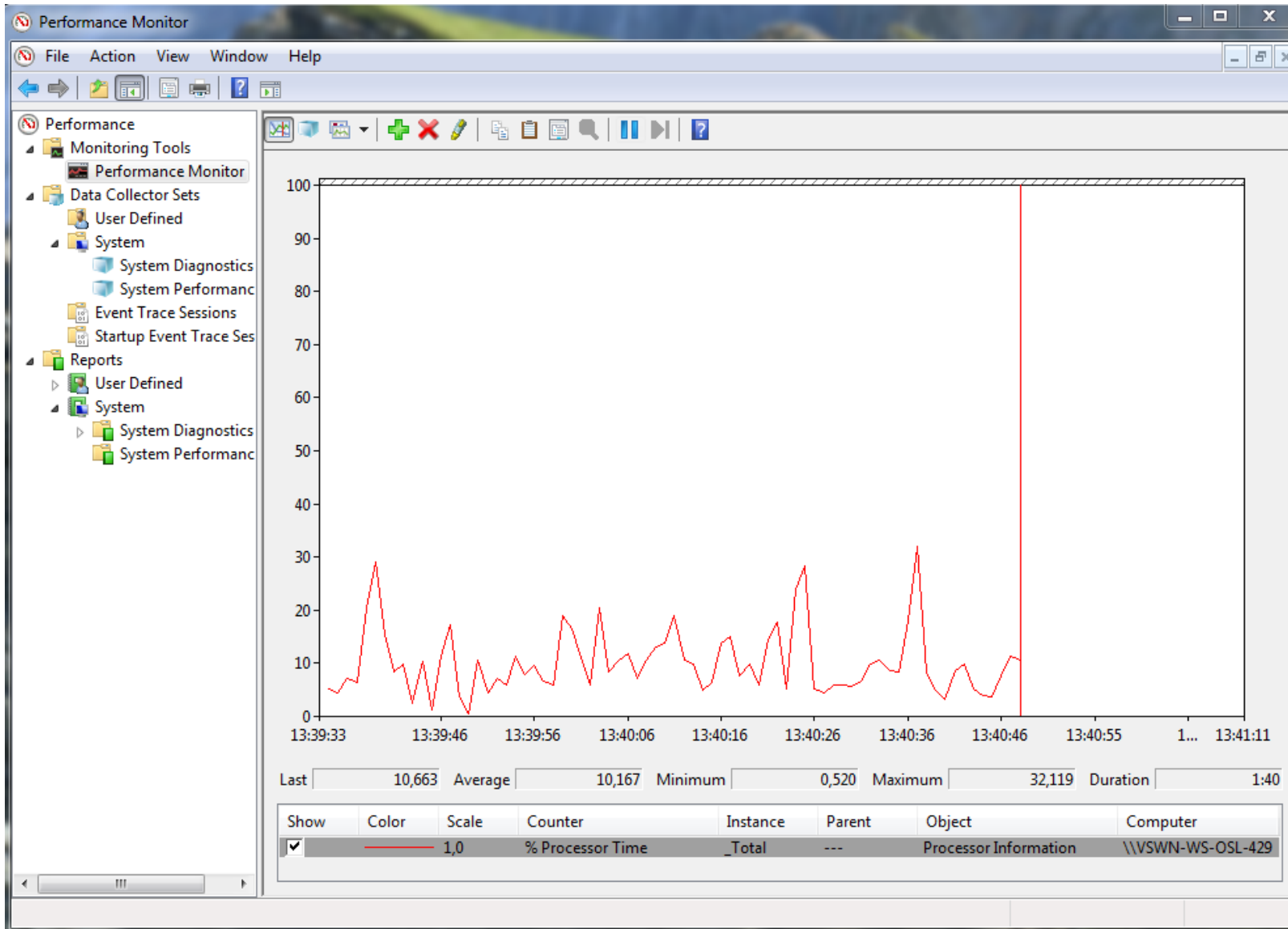
- Are not strictly testing tools but provide information that can be used for testing purposes.
- Analyze, verify and report on usage of specific system resources, and give warnings of possible service problems.

The screenshot shows the Windows Task Manager window with the 'Processes' tab selected. The window title is 'Windows Task Manager'. The menu bar includes 'File', 'Options', 'View', and 'Help'. Below the menu bar are tabs for 'Applications', 'Processes', 'Services', 'Performance', 'Networking', and 'Users'. The main area displays a table of running processes.

| Image Name | User Name | CPU | Memory (Private...) | Command Line | Description |
|-----------------------|-----------------|-----|---------------------|--------------------------------------|-----------------------------------|
| iexplore.exe *32 | raluca.madalina | 00 | 953 488 K | "C:\Program Files (x86)\Internet... | Internet Explorer |
| sqlservr.exe | NETWORK SERVICE | 00 | 811 276 K | "C:\Program Files\Microsoft SQL ... | SQL Server Windows NT - 64 Bit |
| firefox.exe *32 | raluca.madalina | 00 | 307 012 K | "C:\Program Files (x86)\Mozilla F... | Firefox |
| devenv.exe *32 | raluca.madalina | 00 | 236 440 K | "C:\Program Files (x86)\Microsof... | Microsoft Visual Studio 2008 |
| OUTLOOK.EXE *32 | raluca.madalina | 00 | 201 692 K | "C:\Program Files (x86)\Microsof... | Microsoft Outlook |
| svchost.exe | SYSTEM | 00 | 201 072 K | C:\Windows\System32\svchost.... | Host Process for Windows Services |
| spotify.exe *32 | raluca.madalina | 00 | 156 220 K | Spotify.exe /UPDATECOMPLETE | Spotify |
| chrome.exe *32 | raluca.madalina | 00 | 146 760 K | "C:\Users\raluca.madalina\AppData... | Google Chrome |
| Skype.exe *32 | raluca.madalina | 00 | 136 352 K | "C:\Program Files (x86)\Skype\P... | Skype |
| UltimaTestCase.ex... | raluca.madalina | 00 | 112 124 K | "C:\Users\raluca.madalina\AppData... | UtcDesigner |
| chrome.exe *32 | raluca.madalina | 00 | 91 056 K | "C:\Users\raluca.madalina\AppData... | Google Chrome |
| chrome.exe *32 | raluca.madalina | 00 | 83 360 K | "C:\Users\raluca.madalina\AppData... | Google Chrome |
| explorer.exe | raluca.madalina | 00 | 80 332 K | C:\Windows\Explorer.EXE | Windows Explorer |
| chrome.exe *32 | raluca.madalina | 00 | 76 664 K | "C:\Users\raluca.madalina\AppData... | Google Chrome |
| POWERPNT.EXE *32 | raluca.madalina | 00 | 71 852 K | "C:\Program Files (x86)\Microsof... | Microsoft PowerPoint |
| ReportingServicesS... | NETWORK SERVICE | 00 | 71 044 K | "C:\Program Files\Microsoft SQL ... | Reporting Services Service |
| chrome.exe *32 | raluca.madalina | 00 | 61 112 K | "C:\Users\raluca.madalina\AppData... | Google Chrome |

•Performance testing/load testing/stress testing tools





1.7 Tools support for specific application areas

- There are tools specialized for use in a particular type of application.

Example:

performance testing tools specifically for web-based applications

and dynamic analysis tools specifically for testing security aspects.

- Example of targeted areas: embedded systems.

1.8 Tools support using other tools

- This is not a complete list of tools of this category.
- Testers may use:
 - word processor
 - spreadsheetsas a testing tool, but they are often used to store:
 - test designs
 - test scripts
 - test data.
- Testers may also use SQL to set up and query databases containing test data.
- Tools used by developers when **debugging**, to help localize defects and check their fixes, are also testing tools.
- It is a good idea to look at any type of tool available to you for ways it could be used to help support any of the testing activities.



- 2.1 Potential risks & benefits
- 2.2 Special considerations

2. Effective use of tools



2.1 Potential benefits and risks

- Simply purchasing or leasing a tool does not guarantee success with that tool.
- Each type of tool may require additional effort to achieve real and lasting benefits.

Potential benefits of using tools :

Reduced repetitive work (running regression tests, re-entering the same test data. Etc)

Greater consistency and repeatability (tests executed by a tool, tests derived from requirements).

Objective assessment (static measures, coverage).

Ease of access to information about tests or testing (statistics / graphs about test progress, incident rates, performance)

Risks of using tools:

Unrealistic expectations for the tool (functionality & ease of use).

Underestimating time, cost and effort for the introduction of a tool (training, external expertise).

Underestimating the time and effort needed to achieve significant and continuing benefits from the tool

Underestimating the effort required to maintain the test assets generated by the tool.

Over-reliance on the tool (replacement where manual testing would be better).



2.2 Special considerations

1. Test execution tools

- This type of tool often requires significant effort in order to achieve significant benefits.
- Capturing tests by recording the actions of a manual tester seems attractive, but this approach does not scale to large numbers of automated tests. This type of script may be unstable when unexpected events occur.
 - Data-driven approach: separates out the test inputs (the data) and uses a more generic script that can read the test data and perform the same test with different data.
- In a keyword-driven approach: the spreadsheet contains keywords with the actions to be taken (also called action words), and test data. Testers can then define tests using the keywords.

2. Performance testing tools

- These tools need tester with expertise in performance testing to design the tests and interpret results.

3. Static analysis tools

- These tools applied to source code can enforce coding standards, but if applied to existing code may generate a lot of messages
- A gradual implementation with initial filters to exclude some messages would be an effective approach.

4. Test management tools

- They need to interface with other tools or spreadsheets in order to produce information in the best format for the current needs of the organization.
- The reports need to be designed and monitored so that they provide benefit.



3. Introducing a tool into a organization



3. Introducing a tool into an organization

- The main considerations in selecting a tool for an organization include:

Assess the organizational maturity, strengths and weaknesses

Evaluate against clear requirements and objective criteria.

A proof-of-concept to test the required functionality and determine whether the product meets its objectives.

Evaluation of the vendor (including training, support and commercial aspects).

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

- Introducing the selected tool into an organization starts with a pilot project, with the following objectives:

Learn more detail about the tool.

Evaluate how the tool fits with existing processes and practices, and determine what would need to change.

Decide on standard ways of using and maintaining the tool and the test

Assess whether the benefits will be achieved at reasonable cost.



3. Introducing a tool into an organization

- Success factors for the deployment of the tool within an organization include:

Roll out the tool to the rest of the organization incrementally.

Adapt and improve processes to fit with the use of the tool.

Provide training and coaching/mentoring for new users.

Define usage guidelines.

Implement a way to learn lessons from tool use.

Monitor the tool use and benefits.

