Benefits realized in using Rational Functional Tester and Performance Tester

Presenters’ Name
Sumika Mukerji & Sharath TS
sumika.mukerji@accenture.com
sharath.t.s@accenture.com

Select View/Master/Slide Master to add Session Number Here
Project Overview

To develop a solution that will integrate two diverse PPM tools.

- **Tool A** is the standard project management tool for primarily used by maintenance projects for managing development, problem and maintenance requests.

- **Tool B** is the standard project management tool aligned to the requirements of CMMI level 5 organization.

Maintenance projects were required to use both the tools with overlapping functionalities resulting in redundant work and loss of productivity.

The goal of this project was to reduce manual effort of both the tool users to maintain synchronized data across the systems and obviate reconciliation effort.

Scope of the Project

- Project is responsible for complete SDLC for developing the integration solution.
- Scope included Requirements gathering, Analysis, Design, Build, Testing (Unit Testing, System Testing and Performance Testing) and deployment of solution.

Work Locations

- Bangalore, India

Program Delivery Statistics

- Team started with a size of 3.
- The team had a peak team size of 20 people.
- Tools: Rational Performance Tester, Rational Functional Tester, Rational XDE for .NET.
- Duration: 12 months.
Rational Tools Fitment

Few of the challenges faced by the team, before Rational tools were used

• Regression Testing of the solution consumed significant amount of time of the team; and was prone to human errors

• A simulation of real life scenario was essential to test the performance of these two PPM tools

Project used the following Rational Tools:

Stress Testing → Rational Performance Tester
Regression Testing → Rational Functional Tester
Functional Testing – Scope

Scope

- **Tool A Functionalities**
  - Development Request
  - Problem Tickets
  - Maintenance Request
  - Work Plan Creation

- **Tool B Functionalities**
  - Project Creation
  - Adding Resources to project
  - Create Modules and Tasks
  - Create Timesheets
  - Generate Reports

Additionally, Tool A had major releases very often. The changes in the functionality of Tool A was in a few places each time, and repetitive manual testing was consuming lots of time.

We had two web based PPM tools and used RFT exhaustively to test the integration solution developed.
Functional Testing Life Cycle – Change Management

Steps followed for any major release by Build team:

- Identify the changes in the functionality
- Freeze the changes in the application
- Create Use Case scenario for the changes

Steps followed for any major release by Test team:

- Record the script for the changes in the application
- Execute the test script which was recorded and verify the data in the backend
- Once this is done, we would get sign off for release
Two different machines were used to record the functionality of the two tools parallely.

Multiple Rational Functional test machines were used to test (playback) certain set of functionalities of one tool.

For any changes in the functionality, only the changes were being recorded and executed.
Technical Details of Using RFT

- **Recording a test script**
  - There was one RFT main machine where all the recording, scripting, code changes and other activities related to the script was carried out.

- **Verification points**
  - Inserted many data verification points
    - Data verification point – checking the existence of correct data in the application
    - Object verification point - checking the existence of object (E.g. command button) in the application
  - During the playback we were able to find out whether the testing is going as desired.

- **Running a test**
  - Using the Agent Controller, the scripts were played back on different machines – totally 3 machines.

- **Using Agent controller**
  - Agent controller was installed in all the playback machines
  - And during the test execution, it was used to playback the scripts
  - This saved a lot of time and effort as it eliminated the need to do repeated recording and playback for each machine.
Qualitative Benefits - using RFT

- Ease of testing for frequently changing application user interfaces
- Minimal code modification was required
- Eliminated repetitive manual testing work and hence saved significant amount of man hours
- The problem of insufficient testing and inaccurate testing was eliminated
- Use of verification points helped verify that testing was happening as desired
Quantitative Benefits – using RFT

Quantitative Benefits

Significant reduction of manual testing effort

- Tool A -> Integration Solution -> Tool B**
  - Cycle time of testing the functionality was brought down from 3 hrs to 30 minutes per tester for a team size of 10 testers
  - Benefits accrued over 3 cycles of testing for 8 releases of Tool B over a period of 3 months (nearly 600 Man Hrs of effort was saved)

- Tool B -> Integration Solution -> Tool A
  - Cycle time of testing the functionality was brought down from 5 hrs to 45 minutes per tester for a team size of 10 testers
  - Benefits accrued over 3 cycles of testing for 8 releases of Tool B over a period of 3 months (nearly 1020 Man Hrs of effort was saved)

**The purpose of an integration solution is to replicate data entered from one tool to the other tool.
Rational Performance Tester

The scope of the Performance Testing is to generate and execute Performance Test scripts to verify that the system performance expectations are met.

- Assumptions:
  - The testing tool hosted represents worst-case scenario from a network latency or expected response standpoint.
  - The functional accuracy of the application cannot and will not be tested when doing a performance testing. It is assumed that all the modules are functionally correct.
  - Access will be given to the performance testing team for monitoring the servers. If access is not provided, the teams will have to coordinate with the technical and testing teams and help in monitoring the servers.
  - The connectivity between the RPT tool and the tool being tested is connected via the internet and will be operational.
Architecture of Rational Performance Testing

Infrastructure used:
1 server for Tool A
1 server for Tool B
1 BizTalk server
1 Agent controller machine
4 load machines

Agent controller - helps in distributing the load to multiple machines and is used like the main machine for scripting, editing and result analysis.
Technical Details of Using RPT

- **Correlations**
  - Correlation is the technique to replace the values at run time - we can give static set of values from datapools or dynamic values from the previous responses
  - There were some values which were unique for every run e.g. Project Name
  - Data pool correlation was used to give the input data which was known before the run
  - Dynamic data correlation was used to replace with the value received from the previous response
  - Correlation was done for the scripts recorded, URL and data

- **Schedules**
  - There were different scenarios in the projects
  - The load was not the same for all the modules
  - We used schedules for loading different loads on different modules
  - Created the groups and divided the load for each module as required
Reports

- The response time of all the pages can be observed
- Response time graph can be used to analyse the page which is worst response
- This is useful while comparing the behavior of the pages
# Reports

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Minimum for Run</td>
<td>Average for Run</td>
<td>Standard Deviation for Run</td>
<td>Maximum for Run</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PPM Logon</td>
<td>1,625</td>
<td>6,195.7</td>
<td>11,702.6</td>
<td>94,706</td>
<td>0.39</td>
<td>286</td>
</tr>
<tr>
<td>Dashboard</td>
<td>406</td>
<td>6,543.8</td>
<td>18,635.9</td>
<td>119,925</td>
<td>0.36</td>
<td>266</td>
</tr>
<tr>
<td>Create New Request [HP]</td>
<td>640</td>
<td>1,505.8</td>
<td>938.5</td>
<td>6,994</td>
<td>0.16</td>
<td>120</td>
</tr>
<tr>
<td>Search Requests [HP]</td>
<td>797</td>
<td>2,267.7</td>
<td>1,096.3</td>
<td>7,751</td>
<td>0.15</td>
<td>112</td>
</tr>
<tr>
<td>global_autoPopulate</td>
<td>93</td>
<td>235.1</td>
<td>500.2</td>
<td>4,641</td>
<td>0.16</td>
<td>120</td>
</tr>
<tr>
<td>Search Projects [HP] {1} DMZ1</td>
<td>2,265</td>
<td>4,119.1</td>
<td>1,757.3</td>
<td>8,000</td>
<td>0.01</td>
<td>10</td>
</tr>
<tr>
<td>Submit New Report [HP] DMZ1</td>
<td>719</td>
<td>1,871.2</td>
<td>1,033.1</td>
<td>3,140</td>
<td>0.01</td>
<td>4</td>
</tr>
<tr>
<td>Create New AM Problem Resolution DMZ1: Gold [HP]</td>
<td>5,375</td>
<td>14,085.4</td>
<td>14,677.1</td>
<td>67,018</td>
<td>0.03</td>
<td>20</td>
</tr>
<tr>
<td>Search Projects [HP] DMZ1</td>
<td>13,688</td>
<td>27,186.4</td>
<td>9,104.6</td>
<td>39,705</td>
<td>0.01</td>
<td>10</td>
</tr>
<tr>
<td>Req #31503: Details [HP] DMZ1</td>
<td>14,250</td>
<td>40,649</td>
<td>35,657.5</td>
<td>99,175</td>
<td>0.02</td>
<td>16</td>
</tr>
<tr>
<td>Select Time Period</td>
<td>8,485</td>
<td>23,594.5</td>
<td>28,848.5</td>
<td>66,846</td>
<td>0.01</td>
<td>4</td>
</tr>
<tr>
<td>global_AutoCompPreVerify</td>
<td>109</td>
<td>562.1</td>
<td>639.8</td>
<td>3,594</td>
<td>0.14</td>
<td>100</td>
</tr>
</tbody>
</table>

## Response Time Report

- This report gives exact values of the response time of all the pages
- Other details which were useful - Response Time(Max., Min., Avg. and Std. deviation)
Qualitative Benefits - using RPT

- Easy simulation of a stressed environment
- Use of data pools helped in giving a large number of unique input values
- Code editing was not required - hence easy to learn and use
- Were able to put load on multiple remote machines - hence with limited hardware we were able to perform extensive testing
- Ease of data correlation (auto correlation, data-pool correlation and dynamic correlation) - hence quick editing and execution of code
- The break point of the applications could be determined easily
- Response time for each page was available
- Could arrive at meaningful conclusions and diagnosis based on the reports obtained
Quantitative Benefits – Using RPT

Scope

- **Tool A**
  - Connected user load of **50 concurrent virtual users** for all the standard functionalities (defined scenarios)

- **Tool B**
  - Connected user load of **90 concurrent virtual users** for all the standard functionality (defined scenarios)

- **A-B Integration Solution**
  - Connected load of data exchange for **2000 data elements** (Eg: 2000 tasks assigned to DCN resources / timesheets logged against these tasks, under multiple project plans)

Benefits

- Simulation of real-time scenario (connected load) of data flow across multiple instances of Tool A and Tool B

- Availability of standard performance reports (response time and page throughput) for analysis

- Automation of data preparation (simulating virtual users) using RPT for data flow across systems
  - Manual effort of 3 hrs per tester for data preparation versus 15 minutes using the tool for a team size of 10 people

- Benefits accrued over 3 cycles of testing for 4 releases of Tool B over a period of 3 months (nearly **330 Man Hrs of effort was saved**)

- Gave insights to code optimization for improving performance (number of round trips to database was brought down significantly and sizing the batch of data exchange was made configurable in the solution)
Benefits Accrued due to Usage of Rational Testing Tools

**Functional Testing**

- Tool A→Tool B Testing
- Tool B→Tool A Testing

**Performance Testing**

- Tool A - Tool B Integration Testing

Total Effort saving: 1620 ManHours

Total Effort saving: 330 ManHours
QUESTIONS
THANK YOU

Learn more at:
- IBM Rational software
- IBM Rational Software Delivery Platform
- Process and portfolio management
- Change and release management
- Quality management
- Architecture management
- Rational trial downloads
- Leading Innovation Web site
- developerWorks Rational
- IBM Rational TV
- IBM Rational Business Partners