IBM Rational Performance Tester - What's New in Version 8.1

Rational. software

David Chadwick, Rational Testing Evangelist

dchadwick@us.ibm.com

Performance Tester Version 8.1 in Action

- Performance Testing in an Agile Environment – Test early and often!
- Rational Performance Tester for the Performance Test Specialist
- Rational Performance Tester for the Performance Analyst

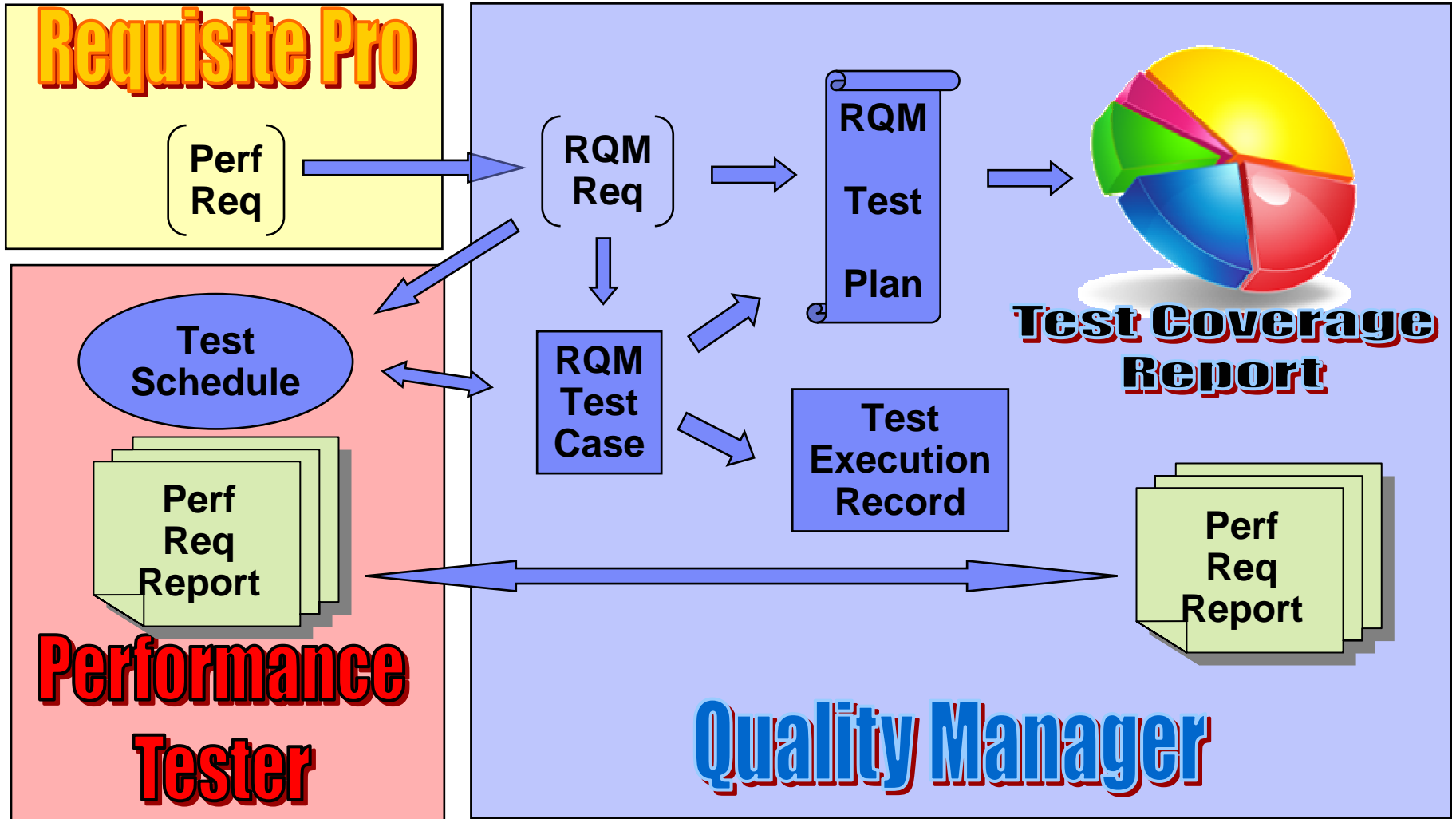# Performance Testing in an Agile Environment

Test early – Test often!!

Rational. software

Performance Requirements Workflow

Agile Performance Testing with RQM/RPT

1. Define a Performance Requirement

2. Write the Test Case

3. Capture the Test

4. Set the Performance Metric

5. Run the Test

6. Analyze the Results

Define a Performance Requirement

- The user story says: "The website user shall be able to access and update the user's profile information."

- The performance requirement –
  - ▸ For user profile update operations,
  - ▸ 90 percent of the responses must return
  - ▸ in less than 2 seconds

- The expected peak workload –
  - ▸ During the busy hour along with other operations,
  - ▸ User profile updates may occur at 2,000 / min

Rational Quality Manager

- Enter the performance requirement in RQM

Performance Test Case in RQM

Home	View Test Cases	Performance - Update...

Manage Sections

Table Of Contents

Summary
Test Case Design
Formal Review
Requirements
Pre-Condition
Post-Condition
Test Scripts
Test Execution Records
Attachments

☐ Show All Sections

Performance - Update User Profile ?
Test Case Overview | View Snapshots

Discard Changes   Save

☑ *Saved successfully at: 21:08:39*

Originator: ADMIN   Action: Select Action ▾   ⇨ State: Draft

Test Case Design ?

Work Item: Create

Define the overall design for the current test case. This can include any background setup information or topologies.

Edit

- The user profile information is saved on the mainframe database
- The SAVE button for the profile update web page causes a POST of the profile information
- The user scenario is as follows:
  - Login as a random registered user
  - Click on the update user profile link
  - Enter some new information in the form
  - Click the SAVE button

8

Capture the Test and Set the Metric

- Capture the user scenario
- Add the 90th percentile < 2 seconds requirement for update response time

Performance Testing in an Agile Environment

- **Write** <u>performance requirements</u> in RQM

- **Attach** a test case defining the workload and performance test

- **Capture** an automated performance test for the feature to be tested

- **Add** the <u>performance requirements</u> to the test

- **Run** the test case which links to the automated performance test

- **Results** of the performance test are automatically recorded in RQM

- **Repeat** the performance test **<u>in every iteration's regression</u>** suite

Performance Requirements Reporting
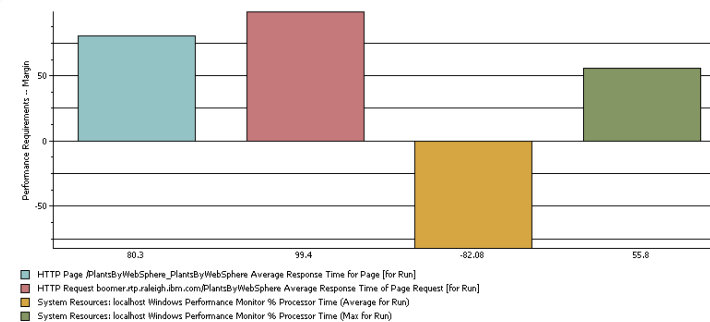
Summary of Pass/Fail and Performance Margin

Status Summary

Performance Requirement Status for Run	Failed
Performance Requirements Percent Passed	75

Summary

|			Performance Requirements -- Specification	Performance Requirements -- Status
HTTP Page	/PlantsByWebSphere_PlantsByWebSphere	Average Response Time for Page [for Run]	<= 3000	Passed
HTTP Request	boomer.rtp.raleigh.ibm.com/PlantsByWebSphere	Average Response Time of Page Request [for Run]	< 1000	Passed
System Resources: localhost	Windows Performance Monitor	% Processor Time (Average for Run)	< 10	Failed
System Resources: localhost	Windows Performance Monitor	% Processor Time (Max for Run)	< 70	Passed

Performance Requirement Margins

80.3	99.4	-82.08	55.8

- HTTP Page /PlantsByWebSphere_PlantsByWebSphere Average Response Time for Page [for Run]
- HTTP Request boomer.rtp.raleigh.ibm.com/PlantsByWebSphere Average Response Time of Page Request [for Run]
- System Resources: localhost Windows Performance Monitor % Processor Time (Average for Run)
- System Resources: localhost Windows Performance Monitor % Processor Time (Max for Run)

Performance Requirements Summary

Overall Summary

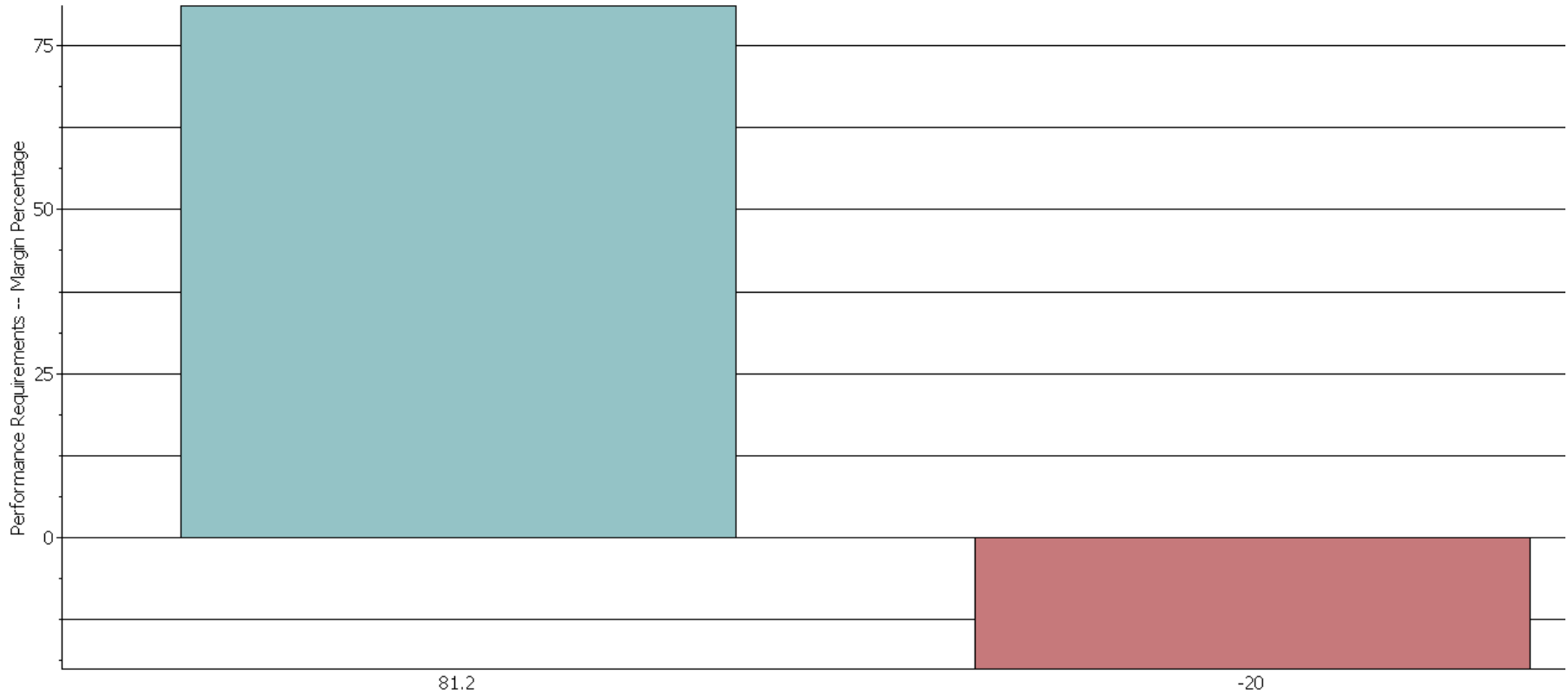| Performance Requirement Status for Run	Failed
Performance Requirements Percent Passed	50
Total Performance Requirements Percent Passed	50

Performance Requirements
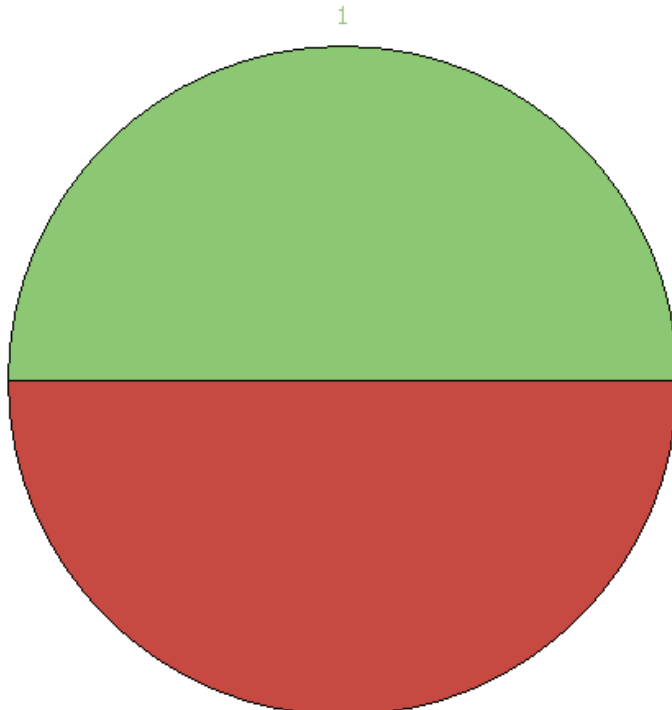
Performance Requirement Margins

Details

Performance Requirement Margins

- ◻ HTTP Protocol Performance Schedule - "perf_requirements" 90th Percentile Response Time for all Page Elements [ms] [for Run]
- ◻ Core Feature Performance Schedule - "perf_requirements" 75th Percentile Execution Time for all Transactions [ms] [for Run]

Performance Requirement Details

Performance Requirement Details

			Performance Requirements -- Margin Percentage	Performance Requirements -- Observed Results	Performance Requirements -- Specification	Performance Requirements -- Status
Core Feature	Performance Schedule - " perf_requirements"	75th Percentile Execution Time for all Transactions [ms] [for Run]	-20	6,000	<= 5000	Failed
HTTP Protocol	Performance Schedule - " perf_requirements"	90th Percentile Response Time for all Page Elements [ms] [for Run]	81.2	94	<= 500	Passed

Performance Requirements in RPT

- Define specific request, page, or transaction requirements in test

- Define aggregate request, page, or transaction requirements in schedule

- Define resource measurement requirements in schedule

- Requirements can be on minimum, maximum, average, percentile or std deviation

- Quality of the performance test results can also be checked
  - Verify quantity of good transactions or page hits is large enough
  - Verify tight enough set of samples – test against standard deviation
  - Verify small number of errors – test percentage of VPs passed

RPT Schedule – Performance Requirement List

HTTP Protocol		
⊟ HTTP Protocol		
Average Response Time For All Page Elements [ms] [for Run]		
Response Time Standard Deviation For All Pages [for Run]		
Percent Page VPs Passed [for Run]		
Page Hit Rate [per second] [for Run]		
Minimum Response Time For All Pages [ms] [for Run]		
25th Percentile Response Time for all Page Elements [ms] [for Run]		
50th Percentile Response Time for all Page Elements [ms] [for Run]		
75th Percentile Response Time for all Page Elements [ms] [for Run]		
85th Percentile Response Time for all Page Elements [ms] [for Run]		
90th Percentile Response Time for all Page Elements [ms] [for Run	**<=**	**500**
95th Percentile Response Time for all Page Elements [ms] [for Run]		
Average Response Time For All Pages [ms] [for Run]		
Maximum Response Time For All Pages [ms] [for Run]		
Response Time Standard Deviation For All Page Elements [for Run]		
Percent Page Element VPs Passed [for Run]		
25th Percentile Response Time for all Pages [ms] [for Run]		
50th Percentile Response Time for all Pages [ms] [for Run]		
75th Percentile Response Time for all Pages [ms] [for Run]		
85th Percentile Response Time for all Pages [ms] [for Run]		
90th Percentile Response Time for all Pages [ms] [for Run]		
95th Percentile Response Time for all Pages [ms] [for Run]		

RPT for the Performance Test Specialist

New Content in RPT 8.1 for test development

Rational. software

HTTP record-time annotations from the browser
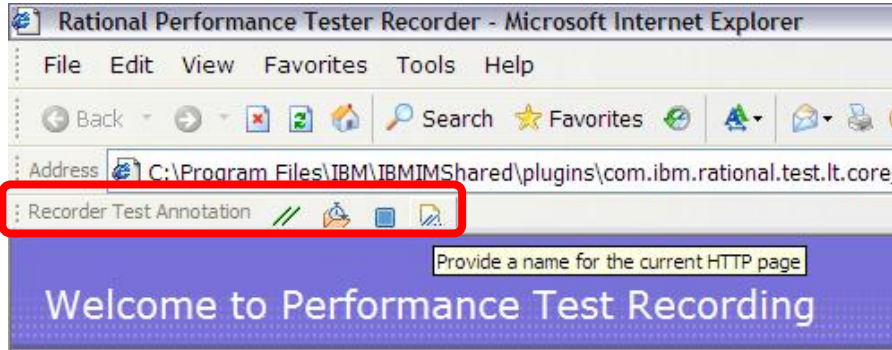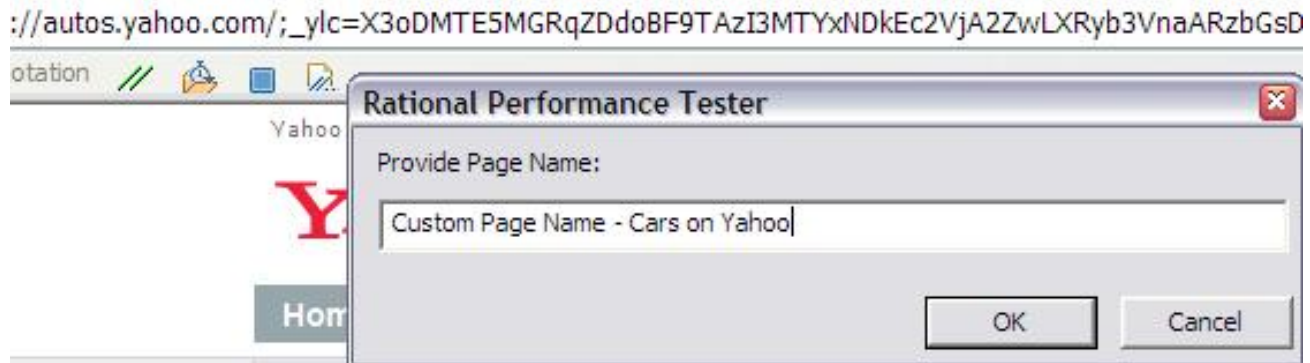
HTTP test split action in editor

- Make multiple tests from one
- Separate login from work to repeat

HTTP test split dialog

Focus

HTTP test after split

Test Variables

- Different from previous version
  - Different initialization methods
  - "Not initialized error" handling at runtime
    - Do Nothing
    - Issue a Warning
    - Issue an Error
    - Exit Test
  - Visibility
    - All tests for this user
    - This test only
- See variable usage through the test;
- See all assignments to the variable;

Test Element Details

Variable: _ylp = "AOWTTaq4nFRKWLcAEBv1cSkA"

Variable Data

Name: _ylp

Visible in: All tests for this user ▾

☑ If not initialized, set to

⦿ Text: AOWTTaq4nFRKWLcAEBv1cSkA

◯ Datapool value: . [Data Source...]

Run-time error if variable not initialized: Issue test log error ▾

Assignments:

⊟ test1_a
  Set _ylp = ["_ylp" ("AOWTTaq4nFRKWLcAEBv1cSkA" - Content)]

Variable Usage:

Test Variables (cont.)

- Set/assignment operator
  - Set existing variable to
    - Literal, text value
    - A value from an Data Source
    - Nothing – un-initialize

Test Element Details

Variable Assignment: Set _ylp = ["_ylp" ("A0WTTaq4nFRKWLcAEBv1cSkA" - Content)]

Variable Data

Name: &= _ylp   [Change...]

Visible in: All tests for this user

┌─ Set to ───
│ ○ Text:
│
│ ◉ Datasource value: &= "_ylp" ("A0WTTaq4...1cSkA" - Content)   [Data Source...]
│
│ ○ Not initialized
└──

Run-time error if variable not initialized: [Issue test log error   ▼]

Variable Usage:

HTTP selecting a random reference from list
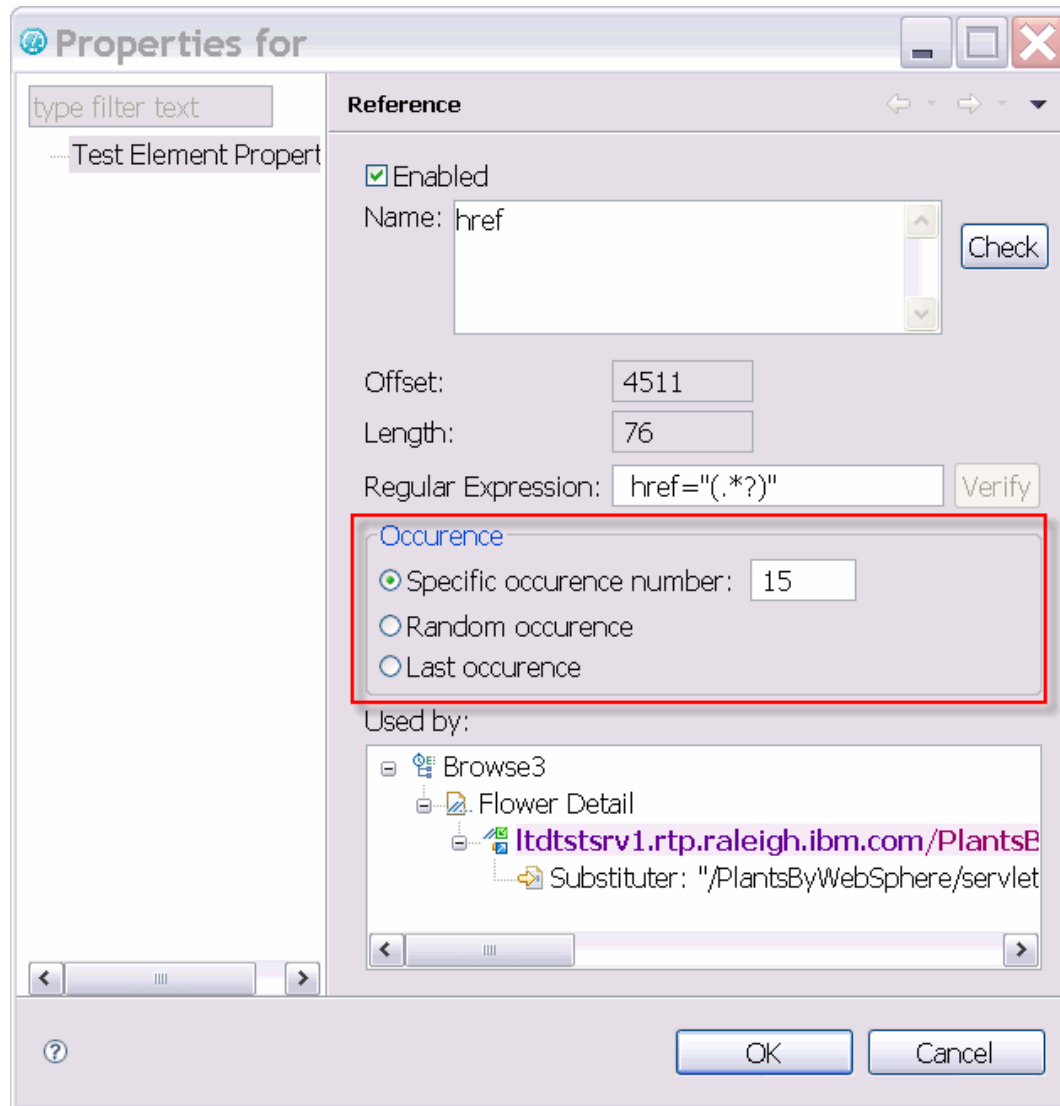
Test editor support for binary data

# Test editor support for data transforms
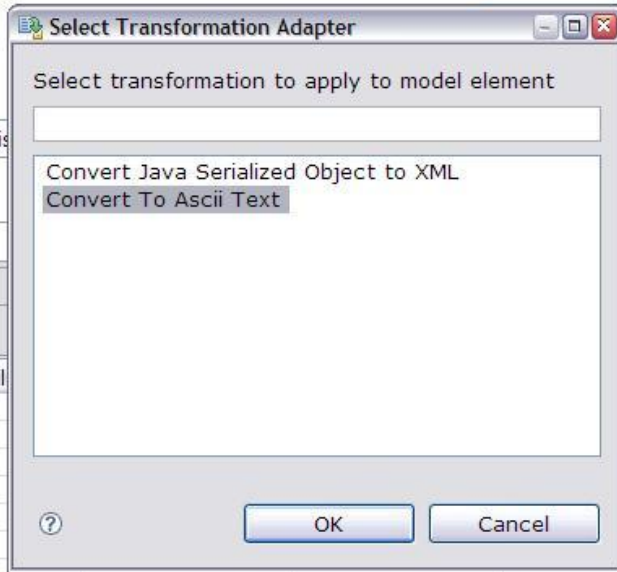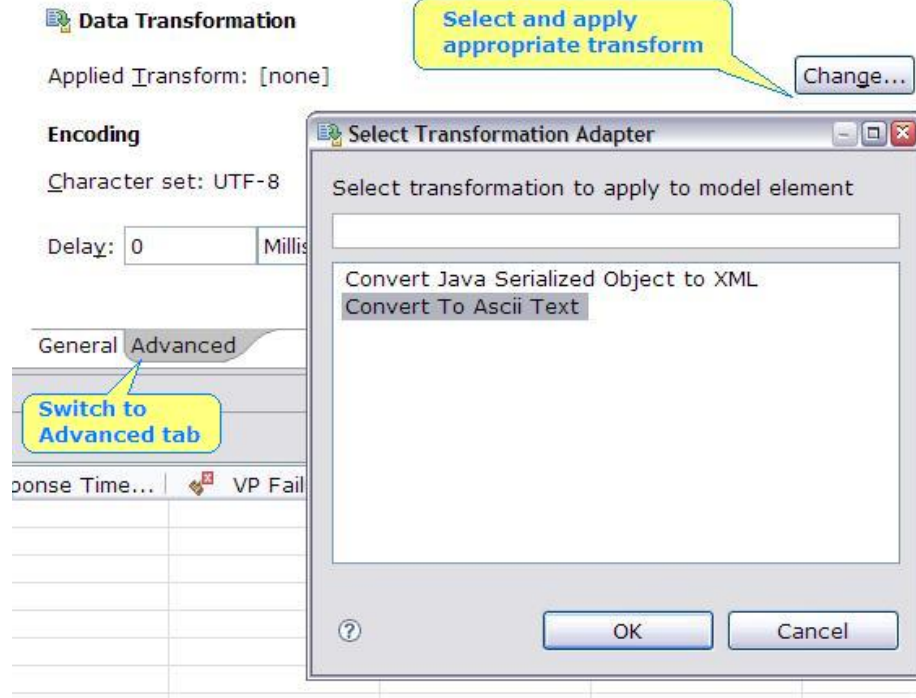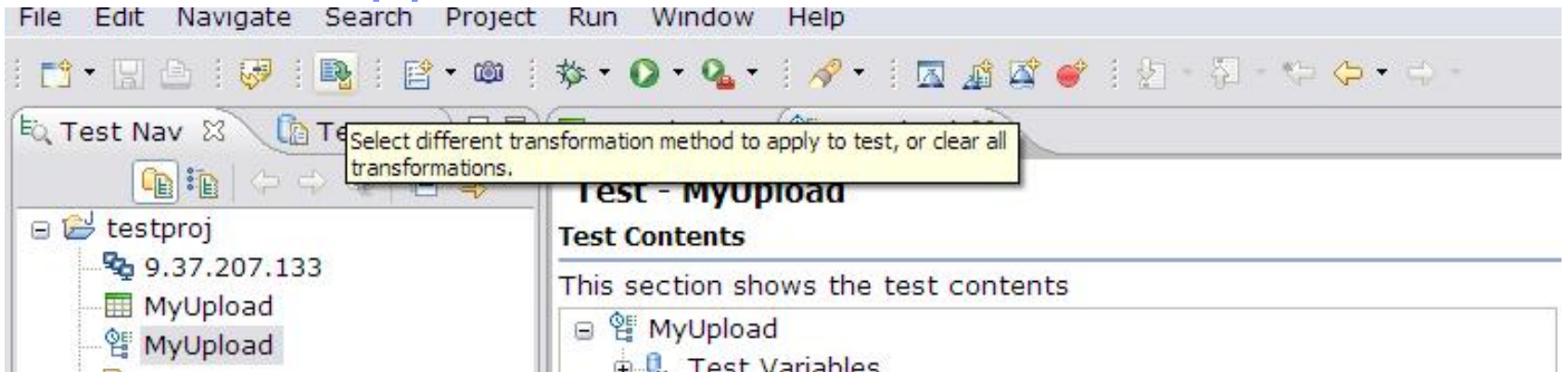
Stop test & stop user from custom code

```java
// Stop the user
throw new RPTStopUserEvent("User Stopped");


// Stop the test
RPTStopTestEvent stopTest =
    new RPTStopTestEvent("Test Stopped");
// Default is ERROR
stopTest.setSeverity(MessageEvent.WARNING);
// No stack trace in the test log
stopTest.setShowStackTrace(false);
throw stopTest;
```
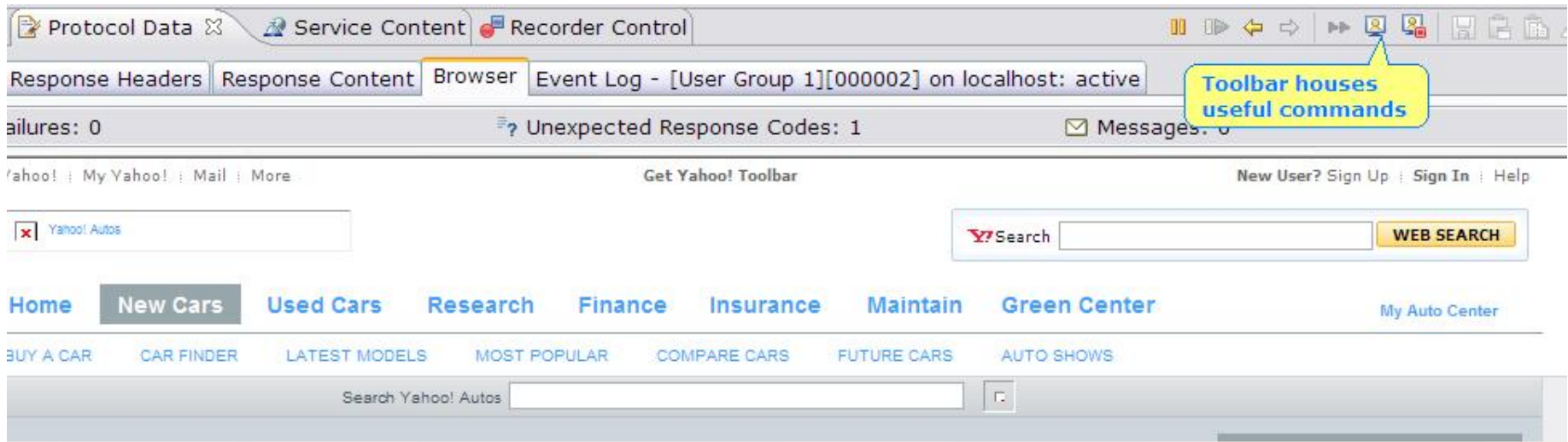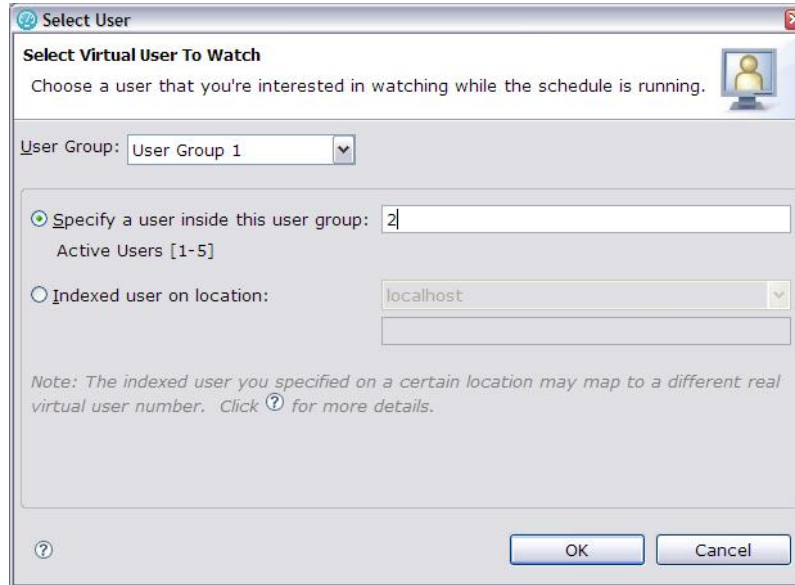
Fast test launch

- In some cases up to 5x improvement
- Sub five second launch on T60
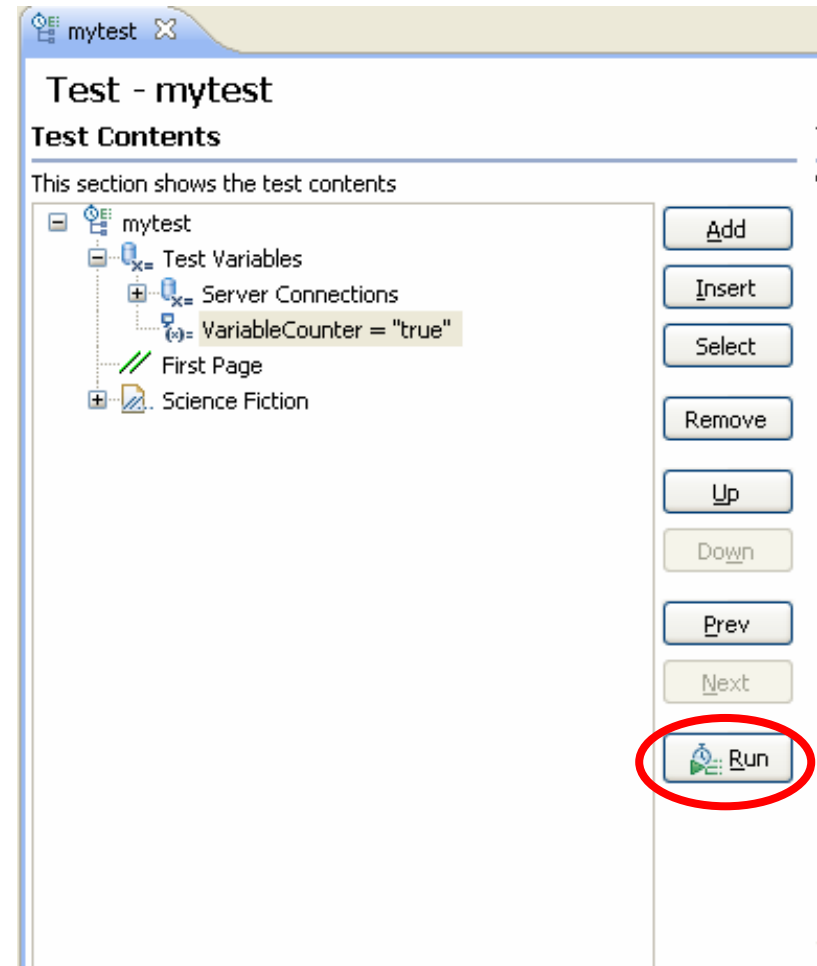
HTTP run-time protocol views for any virtual user

RPT for the Performance Analyst

New Content in RPT 8.1 for analysis

Rational. software

SmartLoad

- **Stage**
  - ‣ Time period with specified number of users

- **Users**
  - ‣ How many users should run at the same time during the stage
  - ‣ It is not how many users to add or subtract

- **Stage Duration**
  - ‣ How long the stage should last
  - ‣ It is not the time at which the stage should begin
  - ‣ It begins AFTER all users have been started for the stage and AFTER any Settle Time

- **Change Rate**
  - ‣ How quickly users should start
  - ‣ Default is to start all users as fast as possible
  - ‣ All / 1 Minute means users for the stage are added or removed in a random uniform fashion over 1 minute

- **Settle Time**
  - ‣ How long to delay before beginning Stage Duration
  - ‣ Affects when the time range for the stage is created
  - ‣ Allows the System Under Test to "settle" after the disruption of changing the number of users running

Schedule Element Details

Schedule_Monday

| User Load | Think Time | Resource Monitoring | Statistics | Test Log | »2

	Users	Stage Duration	Change Rate	Settle Time
	5	10 Minutes	All/1 Minute	30 Seconds
	50	10 Minutes	All/1 Minute	30 Seconds
	500	10 Minutes	All/1 Minute	30 Seconds

Add...
Edit...
Remove
Up
Down

☑ Show Advanced

Time limit for a user to respond to a stop request: 30 Seconds

User Load Preview

Resource Monitoring

Run-time Percentile Reports

Backup Material – Background Sales Slides

Rational. software

Performance Testing in Quality Management

Why do continuous performance testing?

Rational. software

Additional Quality Management considerations drive up costs of remediation

80% of development costs are spent **identifying and correcting defects**!

During the QA/Testing phase

$960/defect

Once released as a product

$7600/defect

During the build phase

$240/defect

During the coding phase
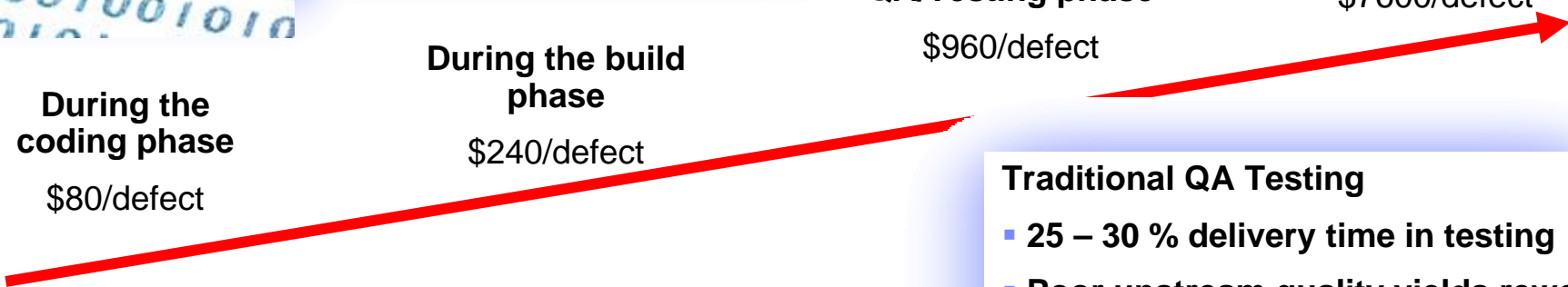
$80/defect

Traditional QA Testing

- **25 – 30 % delivery time in testing**
- **Poor upstream quality yields rework**
- **Compressed schedules make it worse**

Source: 2008 GBS Industry standard study
Defect cost derived in assuming it takes 8 hrs to find, fix and repair a defect when found in code and unit test.
Defect FFR cost for other phases calculated by using the multiplier on a blended rate of $80/hr.

Quality Management Challenges & Importance to the Business

- Business Transformation through Innovation **creates tremendous pressure on IT**
  - ▸ Drives to SOA and agile practices for speed to market
- Must "Break the Iron Triangle" **– traditional methods will not deliver**
  - ▸ On time, with fixed budget and improved quality/performance
- Quality & performance require **continuous, iterative and integrated lifecycle processes**

Scope
(features, functionality)

Quality & Performance

Resources
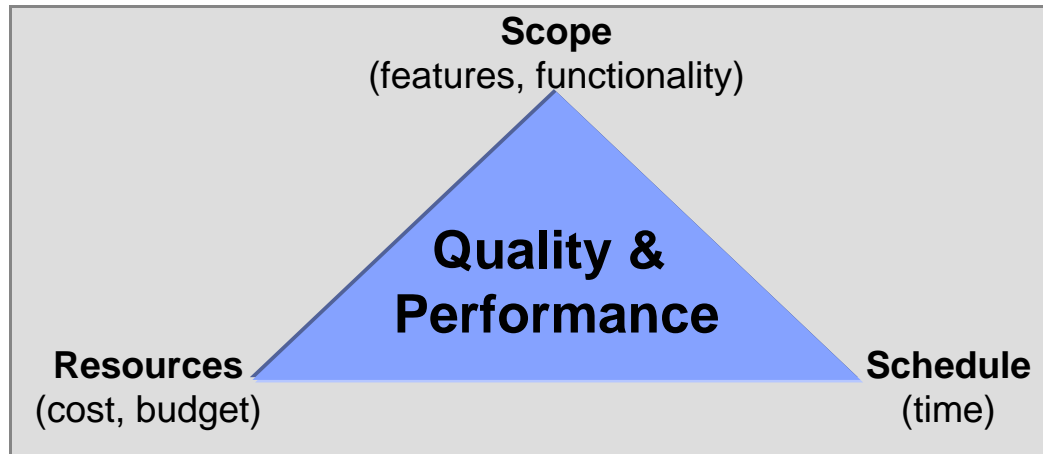(cost, budget)

Schedule
(time)

Quality-enabling activities must start here and involve the entire team

IT Operations must be tied into the lifecycle

Analyst · · · · Architect · · · · Developer · · · · Tester · · · · Deployment Manager · · · · Operations

Quality paradigm is changing

Test – functional silo within software development

- Testing addressed by QA before deployment

- Test as gatekeeper on final release decisions

- Distributed, single project orientation

- Scope – fit for use, reduction of application level defects

Quality – A strategic IT competency

- Quality considerations addressed at all phases

- Agile iterations involve multiple business & technical stakeholders

- Consolidation with internal center of excellence or outsourcing

- Scope – Customer satisfaction and improved business performance

IBM Rational raises the bar for Quality Management

Collaborative process, powered by automation to govern software delivery

Mitigate Business Risk	**Improve Operational Efficiency**	**Make Confident Decisions**
Collaborate	**Automate**	**Report**
Collaborative, continuous, and comprehensive information sharing reduces defects, improves handoff and increases customer satisfaction	*Automating labor-intensive activities reduces time-to-market and increases predictability and consistency to improve return on investment*	*Moment by moment understanding of software quality metrics for immediate corrective action and release decisions addressing both business and IT needs*

Centralized test management hub allowing full lifecycle support across all types of testing and platforms