IBM Software Group

软件质量保证

IBM 软件部　　傅纯一

e business on demand.

议程

- 基本概念
- 开发人员测试
  - 如何保证C/C++传统软件代码质量：Rational PurifyPlus
  - 如何保证C/C++嵌入式软件代码质量：Rational Test RealTime
  - 如何保证Java代码质量：Rational Application Developer
- 系统测试
  - 功能测试自动化：Rational Functional Tester
  - 性能测试：Rational Performance Tester

议程

- 基本概念
- 开发人员测试
  - 如何保证C/C++传统软件代码质量：Rational PurifyPlus
  - 如何保证C/C++嵌入式软件代码质量：Rational Test RealTime
  - 如何保证Java代码质量：Rational Application Developer
- 系统测试
  - 功能测试自动化：Rational Functional Tester
  - 性能测试：Rational Performance Tester

基本概念

- 开发人员测试：由开发人员自己完成的测试
  - 测试驱动的开发
  - 静态分析：在不执行源代码的情况下，对代码中组件关系和方法调用，代码复杂度量以及代码的编写方法进行分析
  - 组件测试：对独立的组件编写测试驱动程序，并执行被测试组件，从而验证其功能
  - 运行时分析：通过分析在软件执行时所收集的数据来理解软件的内部运行状况，从而发现如下问题：
    - 执行路径
    - 代码覆盖
    - C/C++应用的内存访问错误和内存泄漏
    - .NET应用和Java应用的内存泄漏
    - 性能分析
    - 线程分析
- 系统测试：由系统测试人员完成的测试
  - 功能测试自动化：通过执行自动化测试脚本来执行测试，从而验证系统的功能
  - 性能测试：通过模拟大量客户端来访问被测试系统，验证系统的性能和可靠性

议程

- 基本概念
- 开发人员测试
  - 如何保证C/C++传统软件代码质量：Rational PurifyPlus
  - 如何保证C/C++嵌入式软件代码质量：Rational Test RealTime
  - 如何保证Java代码质量：Rational Application Developer
- 系统测试
  - 功能测试自动化：Rational Functional Tester
  - 性能测试：Rational Performance Tester

开发人员测试现状分析

- 强化开发人员测试的意义：
  - 边开发边测试，可提高软件代码质量，减少缺陷修复成本
- 特点：
  - 和开发工具相关
  - 和编程语言相关
- 为什么不进行开发人员测试？
  - 编写测试驱动工作量大
  - 通过手工插帧（在代码中加入调试指令）来获得被测系统内部运行行为，效率低
  - 测试手段和开发环境集成性不好，使用不方便
  - 执行和报告生成依赖于手工

```
If( a > b){
...
printf("DEBUG: IF BRANCH\b");
}else{
...
printf("DEBUG: IF BRANCH\b");
}
```

基于手工方式的开发人员测试可能需要耗费开发人员50%左右的时间

议程

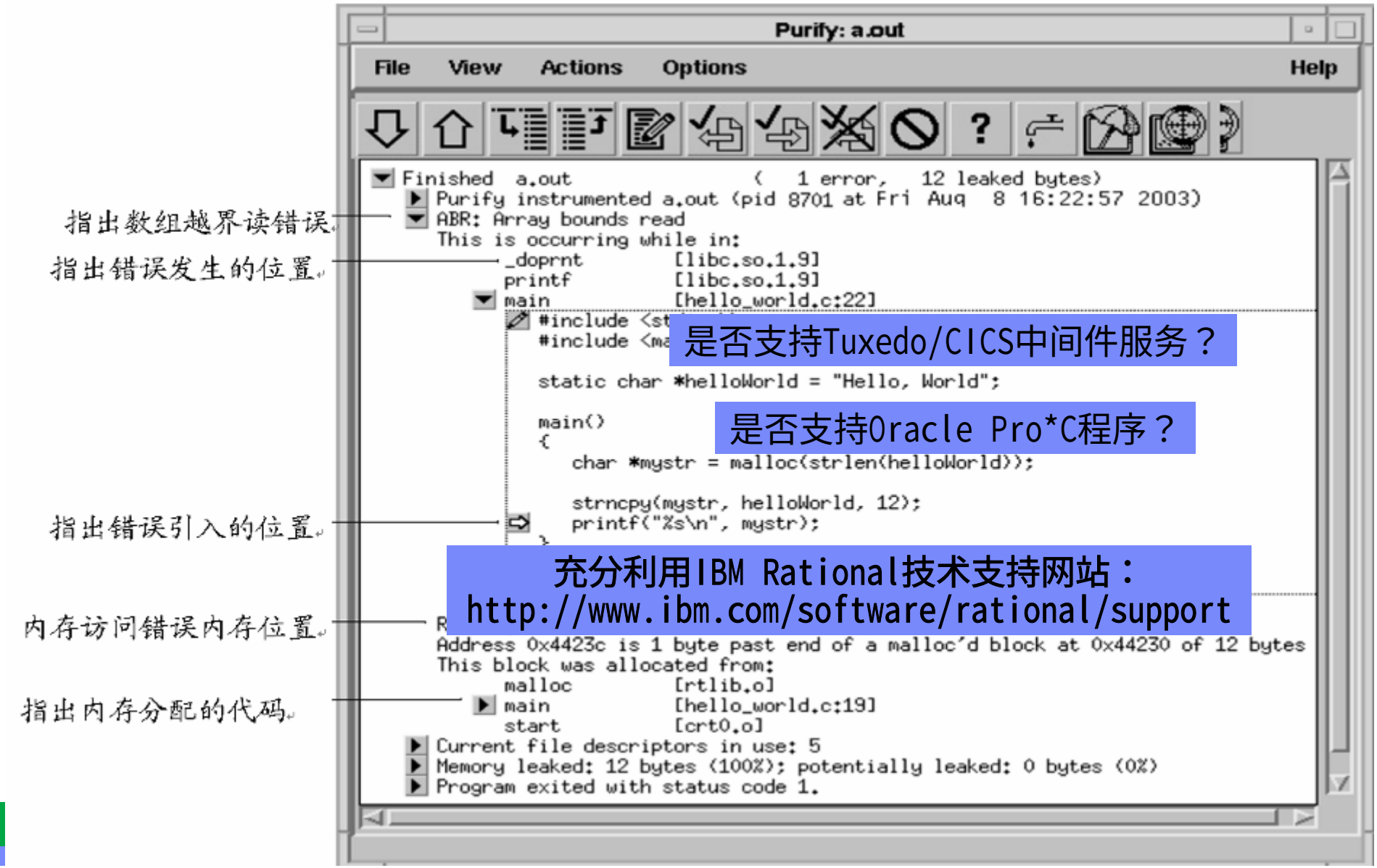
- 基本概念
- 开发人员测试
  - 如何保证C/C++传统软件代码质量：Rational PurifyPlus
  - 如何保证C/C++嵌入式软件代码质量：Rational Test RealTime
  - 如何保证Java代码质量：Rational Application Developer
- 系统测试
  - 功能测试自动化：Rational Functional Tester
  - 性能测试：Rational Performance Tester

Rational PurifyPlus

- 传统C/C++应用的运行时分析工具
  - 内存分析
  - 性能分析
  - 代码覆盖
- 二进制代码插帧
  - 对编译后形成的目标代码进行自动插帧，然后再连接形成可执行文件。
- 其支持能力与编译器相关
  - Visual Studio 6.0，.NET 2003/2005
  - UNIX/LINUX平台下的gcc
  - Xlc(AIX)、C workshop(SUN Solairs)、acc(HP-UX)

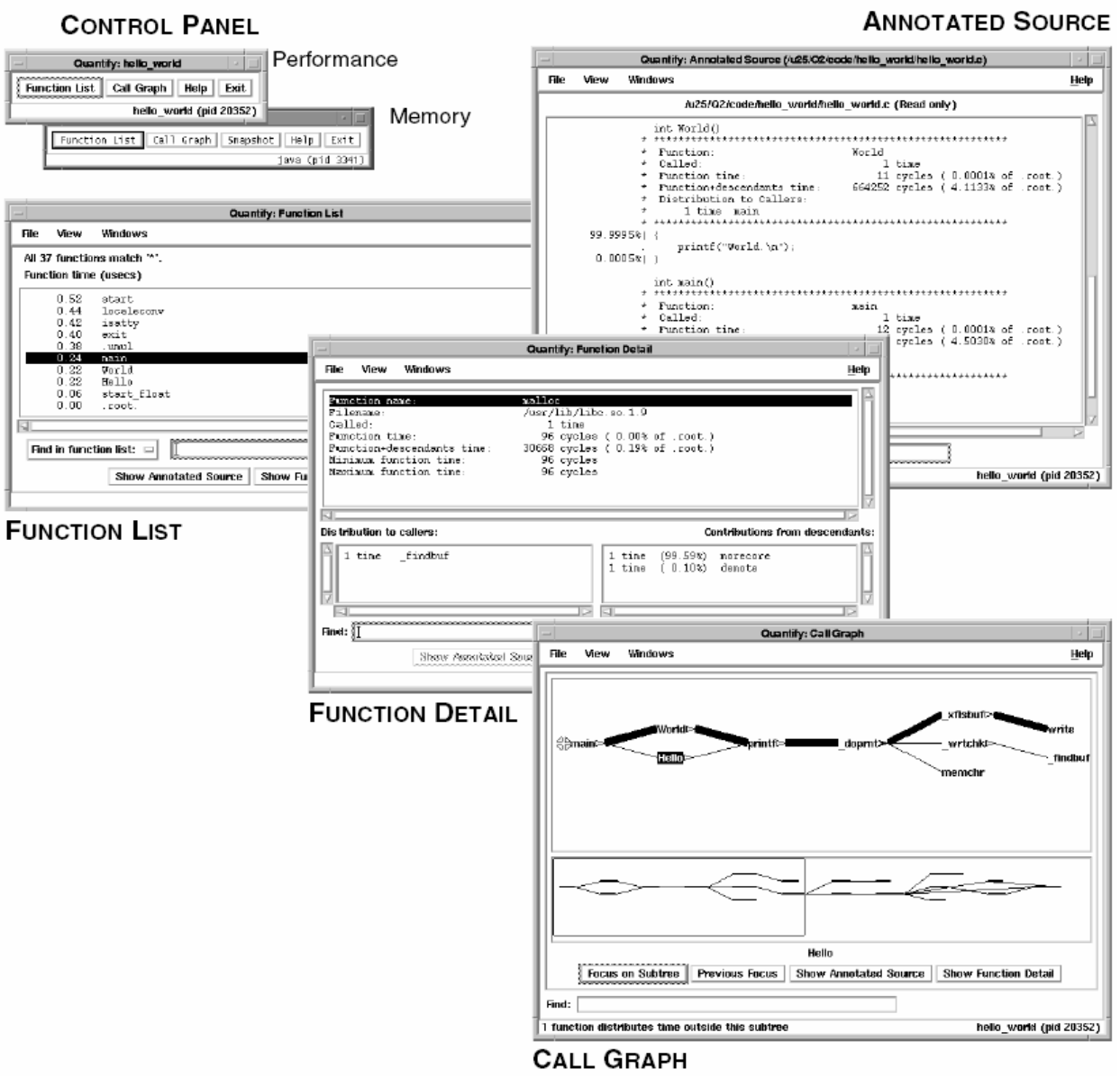
```
# 连接成可执行文件
CC=cc
hello.pure: hello.o
          purify $(CC) -o hello.pure hello.o
# comiple成目标文件
hello.obj: hello.c
          $(CC) -g -c hello.c
```

Rational PurifyPlus/内存分析



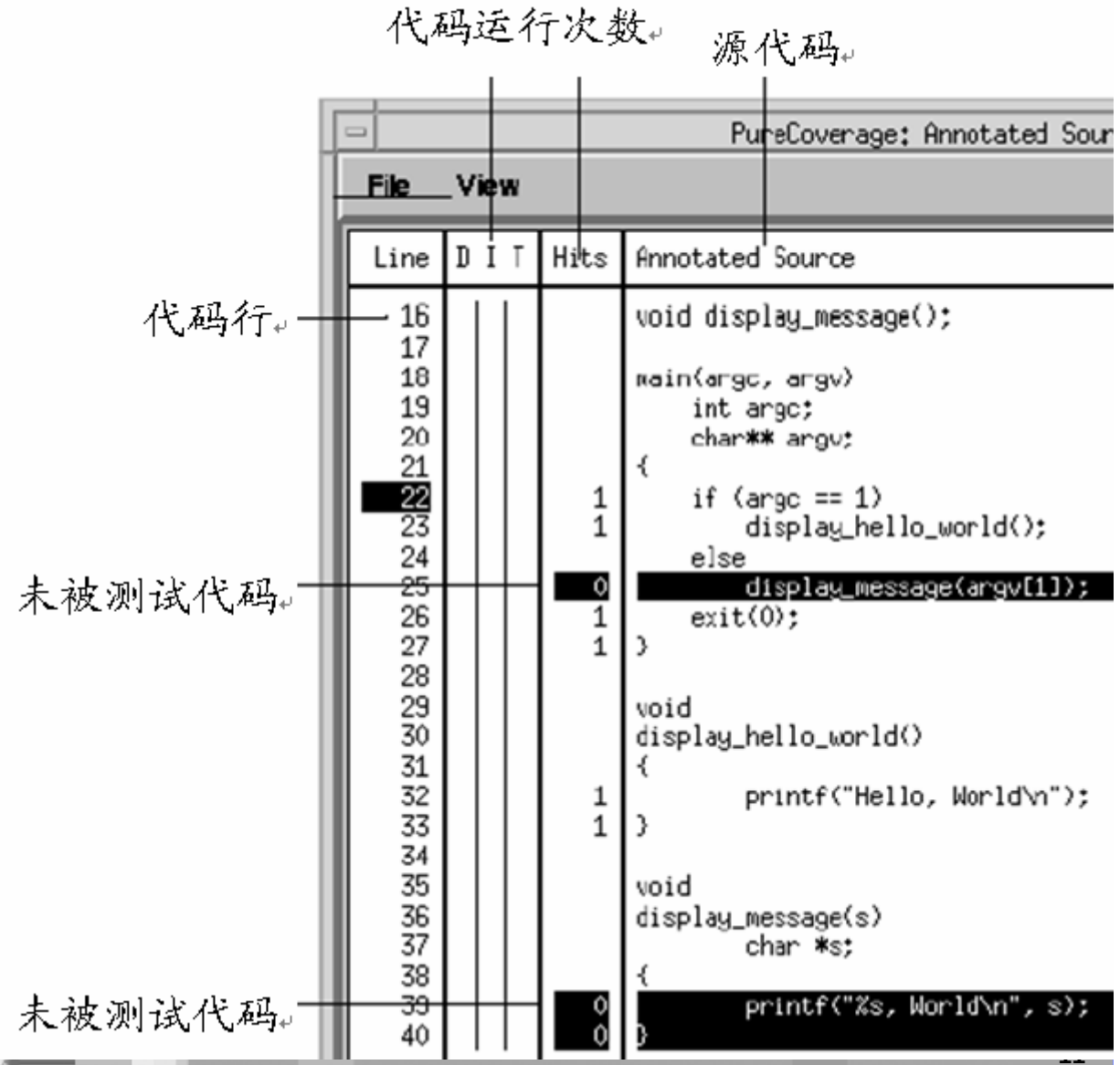
Rational PurifyPlus/性能分析

- 得到准确、可靠的性能数据，能统计出每个函数、每行代码的执行时间、执行次数以及耗时比重。
- 能比较代码调整前后的性能差异。
- 突出显示潜在的性能瓶颈。



Rational PurifyPlus/代码覆盖分析

- 目的：帮助开发人员不断丰富测试用例，提高测试完备性
- 指出未被测试的代码
- 实现多次执行的代码覆盖结果合并
- 自动生成各种代码覆盖统计报告

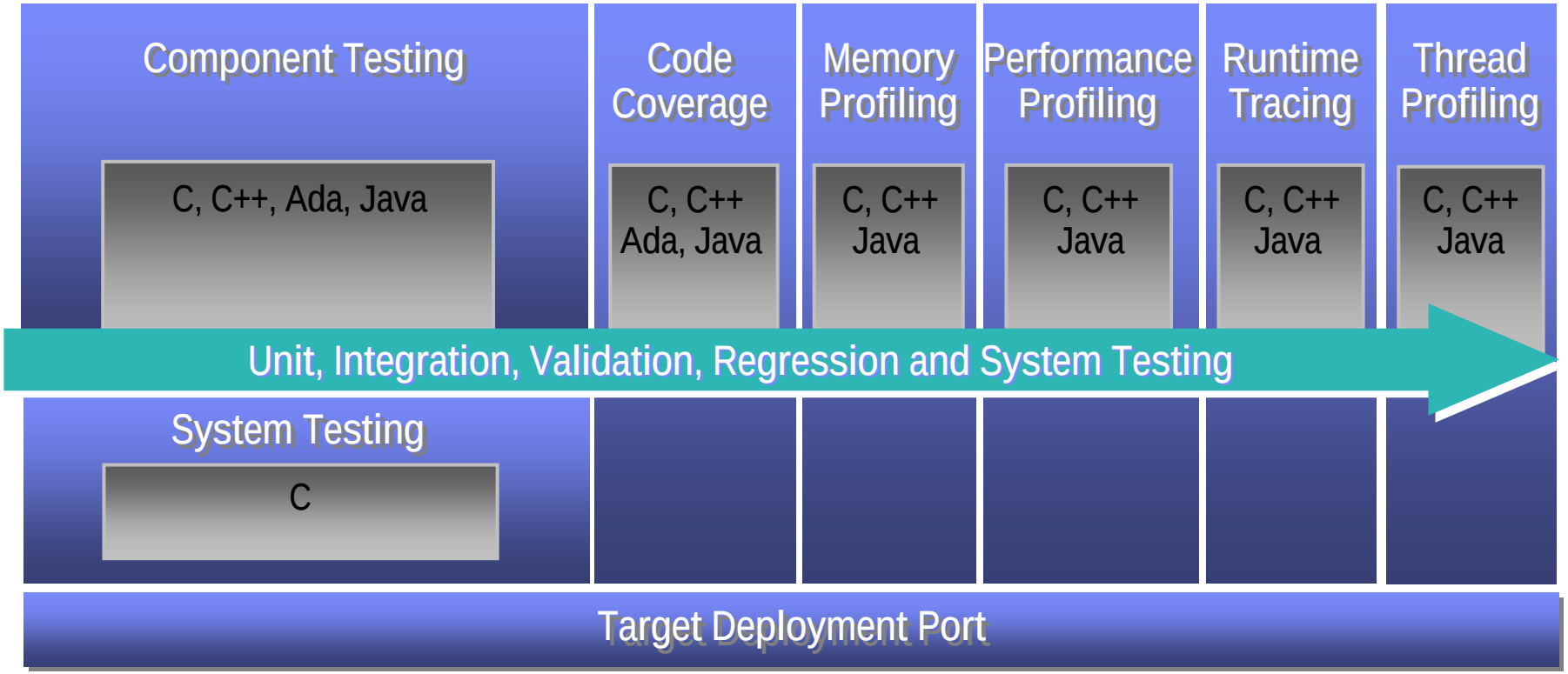


议程

- 基本概念
- 开发人员测试
  - 如何保证C/C++传统软件代码质量：Rational PurifyPlus
  - 如何保证C/C++嵌入式软件代码质量：Rational Test RealTime
  - 如何保证Java代码质量：Rational Application Developer
- 系统测试
  - 功能测试自动化：Rational Functional Tester
  - 性能测试：Rational Performance Tester

IBM Rational Test RealTime

同时支持开发机和目标的全面测试方案



- 基于源代码插帧技术
- 利用可定制的TDP来屏蔽不同嵌入式软件开发时C/C++语言特殊性和开发工具的差异性

IBM Rational Test RealTime

广泛的平台支持

4-Bit to 64-Bit Cross-Development Environments		
▪ *WindRiver*	▪ *Montavista*	▪ *Tasking*
▪ *GreenHills*	▪ *TI*	▪ *CAD-UL*
▪ *ARM*	▪ *NEC*	▪ *Cosmic*
▪ *Enea*	▪ *Hitachi*	▪ *Hiware*
▪ *Windows CE*	▪ *Apex*	▪ *Hitex*
▪ *LynuxWorks*	▪ *Sun*	▪ *Symbian*
▪ *Lauterbach*	▪ *Microtec*	▪ *……*

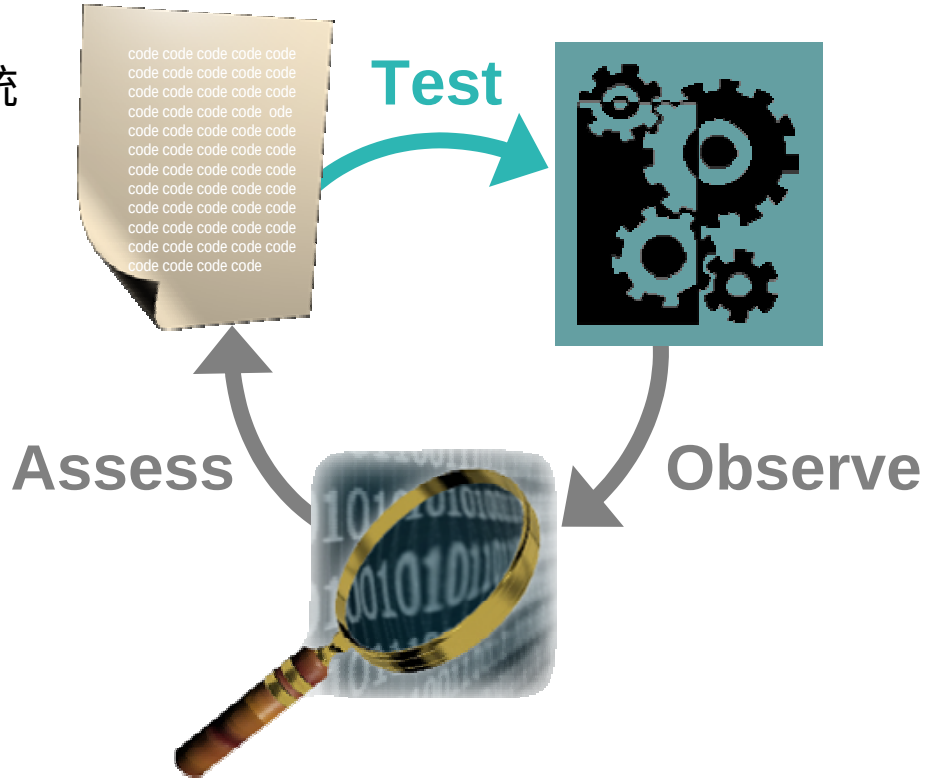
Languages
▪ C ▪ C++ ▪ Ada ▪ J2ME/J2SE

Platforms
▪ *Windows* ▪ *Solaris* ▪ *RedHat* ▪ *HP-UX* ▪ *AIX*

IBM Rational Test RealTime

自动化软件测试过程

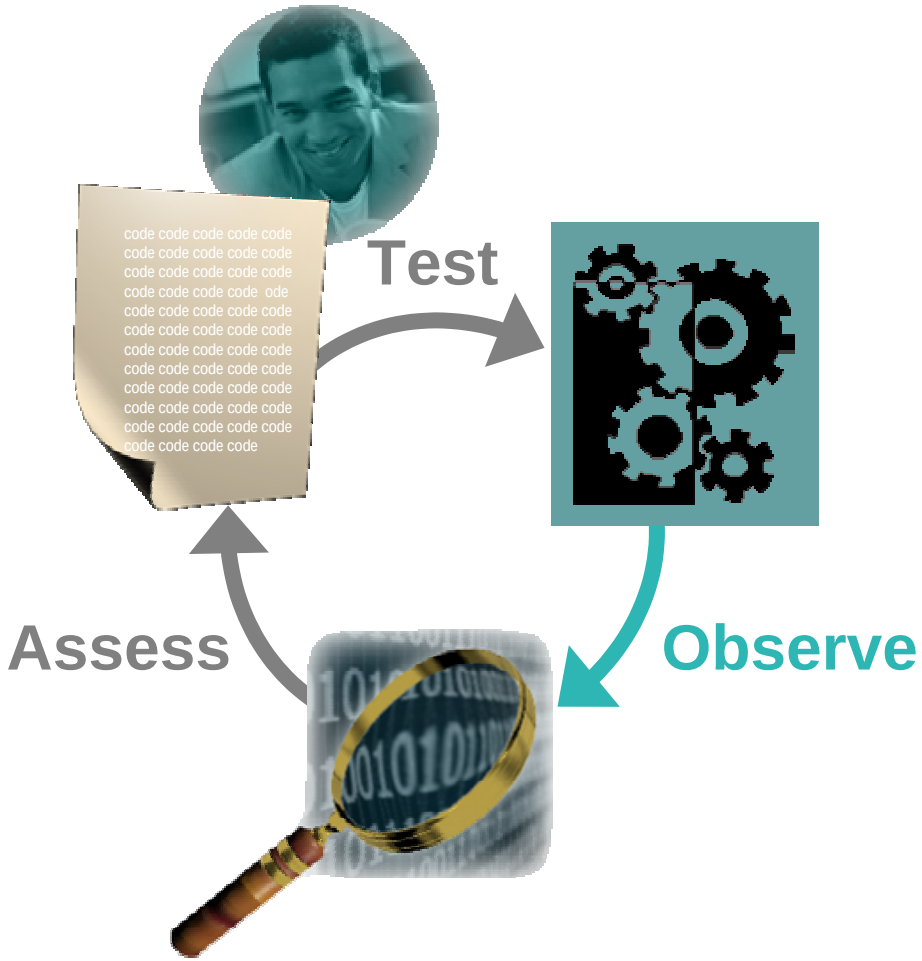
- 全面的自动化测试
  - 自动生成测试脚本模版和测试数据
  - 黑盒测试和白盒测试相集成
  - 多层次测试：从函数到分布式系统
  - 静态分析功能：
    - 明确测试优先级
    - 代码复杂度计算
  - 全面的回归测试



IBM Rational Test RealTime

自动化软件测试过程

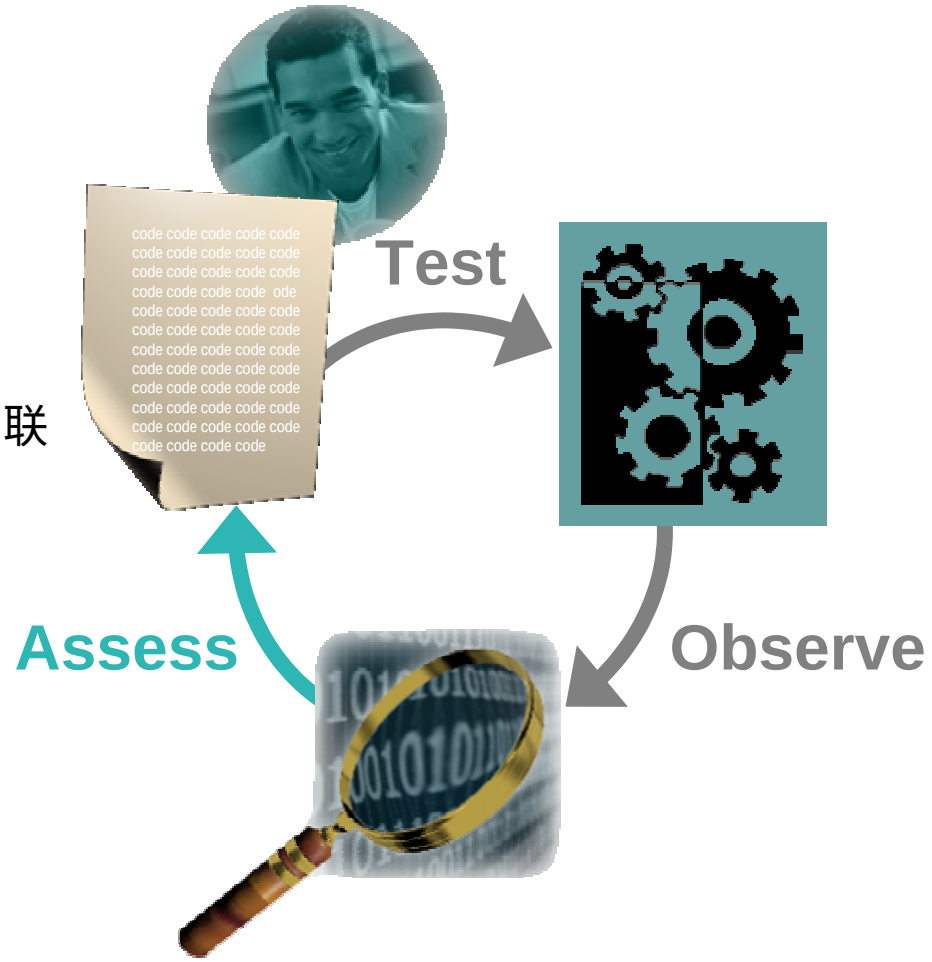
- 全面的自动化测试
- 全面的运行时分析
  - 代码覆盖率分析
  - 内存分析
  - 性能分析
  - 运行时分析
  - 线程分析



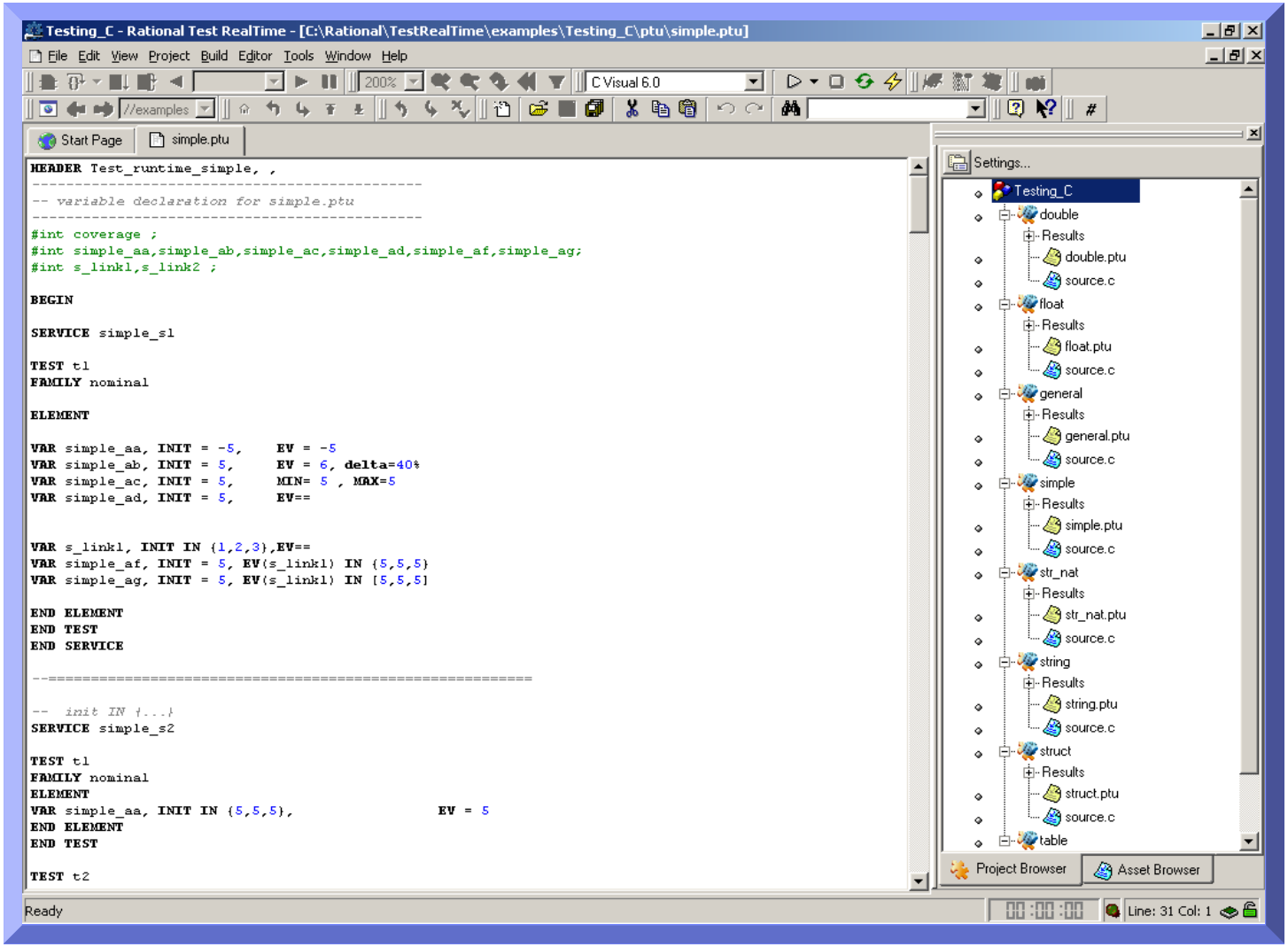
IBM Rational Test RealTime

自动化软件测试过程

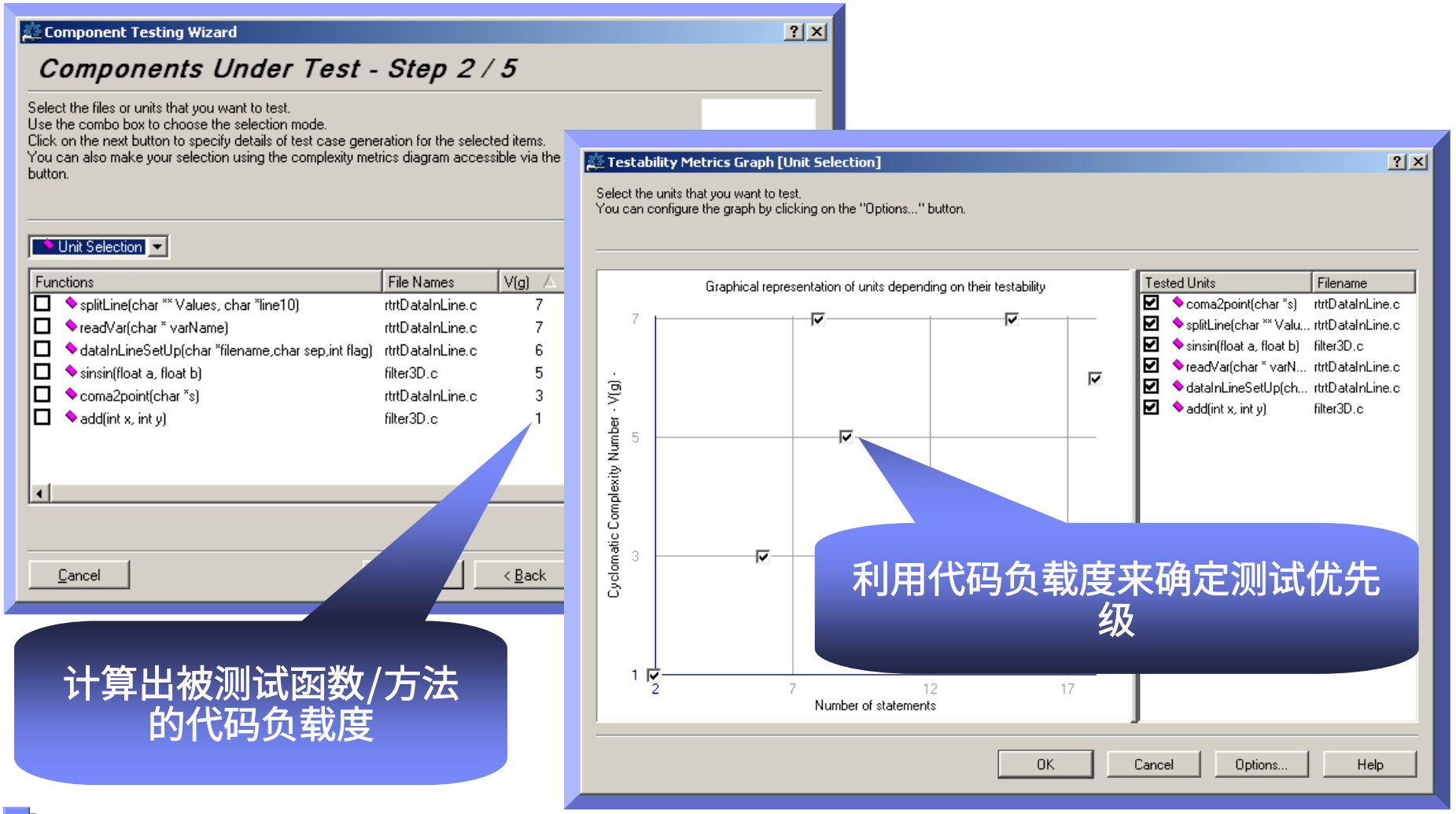
- 全面的自动化测试
- 全面的运行时分析
- 直观的测试结果
  - 测试执行和调试器集成
  - 统一详细的测试报告
  - 运行时分析数据和测试数据相关联



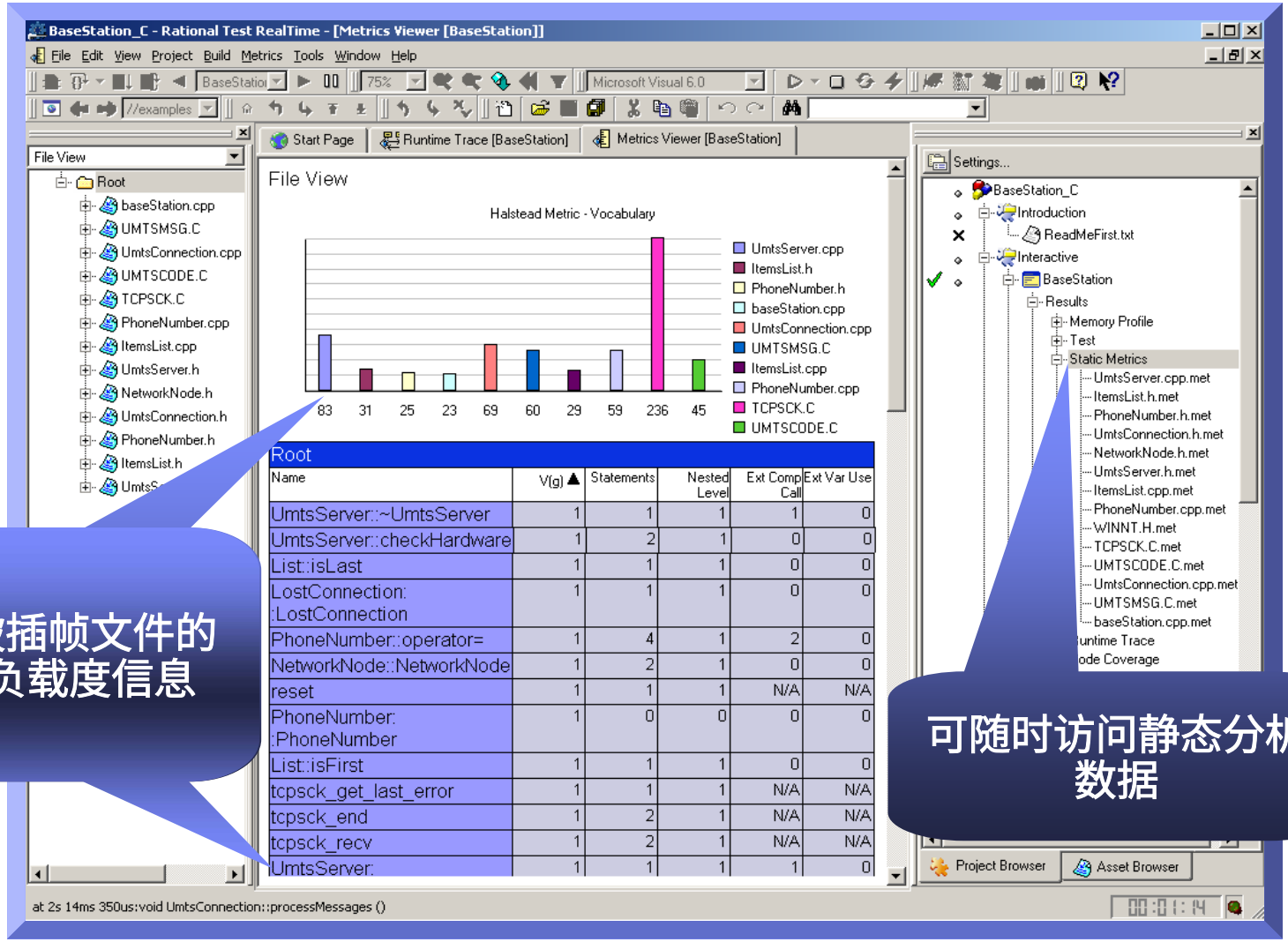
IBM Rational Test RealTime: *测试脚本*



IBM Rational Test RealTime: *静态分析 - 1*



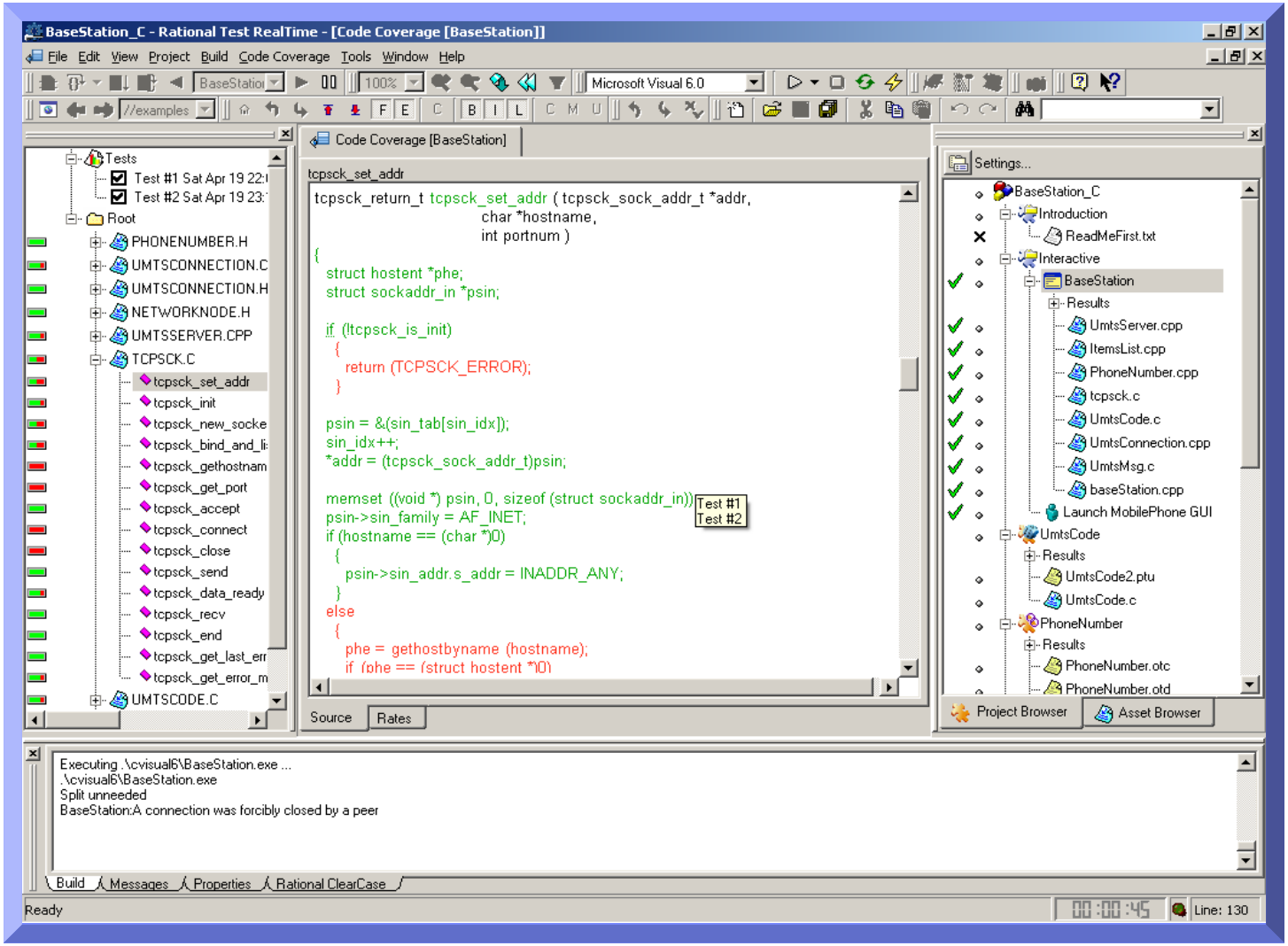
IBM Rational Test RealTime: *静态分析 - 2*



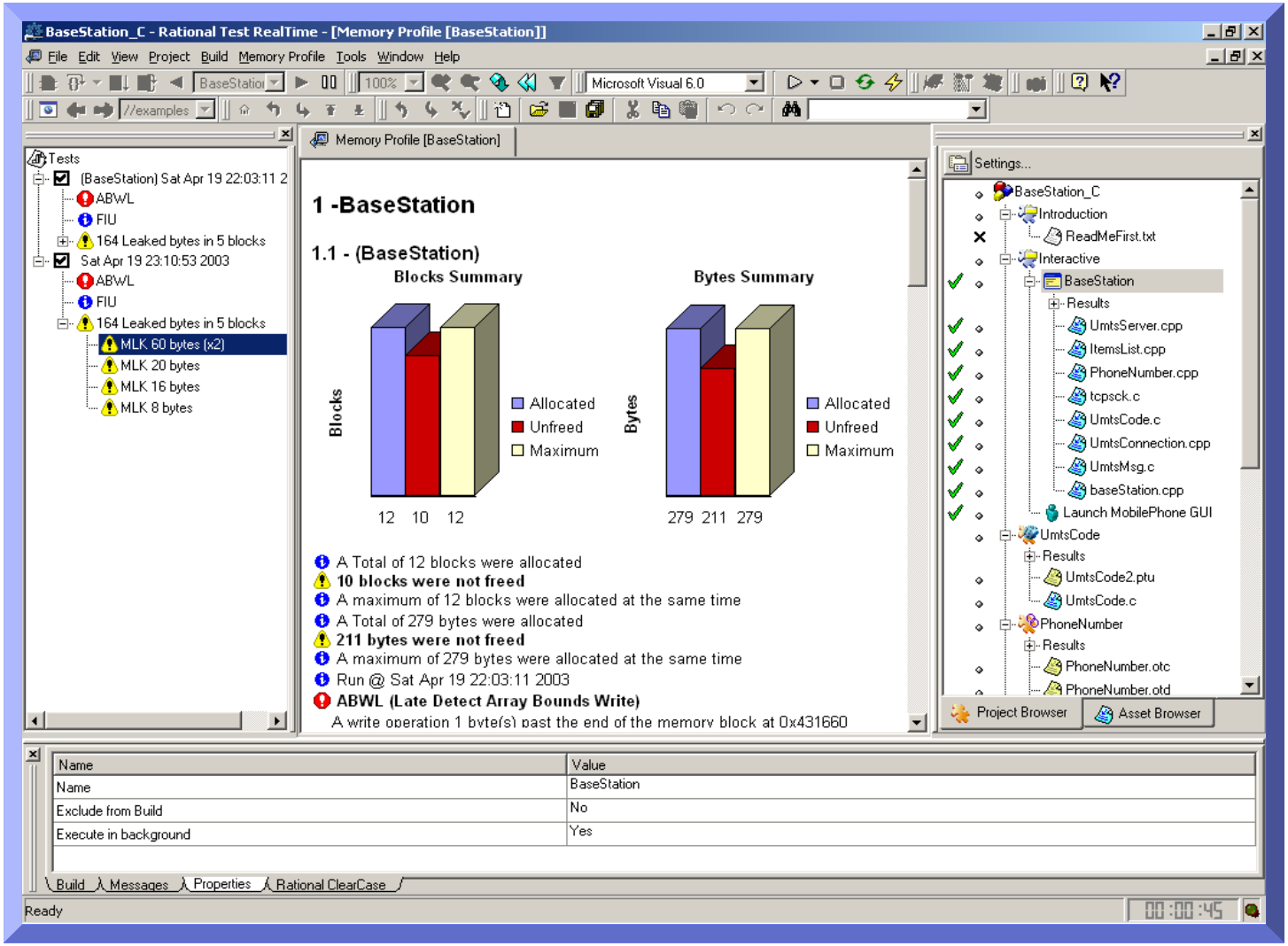
获得被插帧文件的代码负载度信息

可随时访问静态分析数据

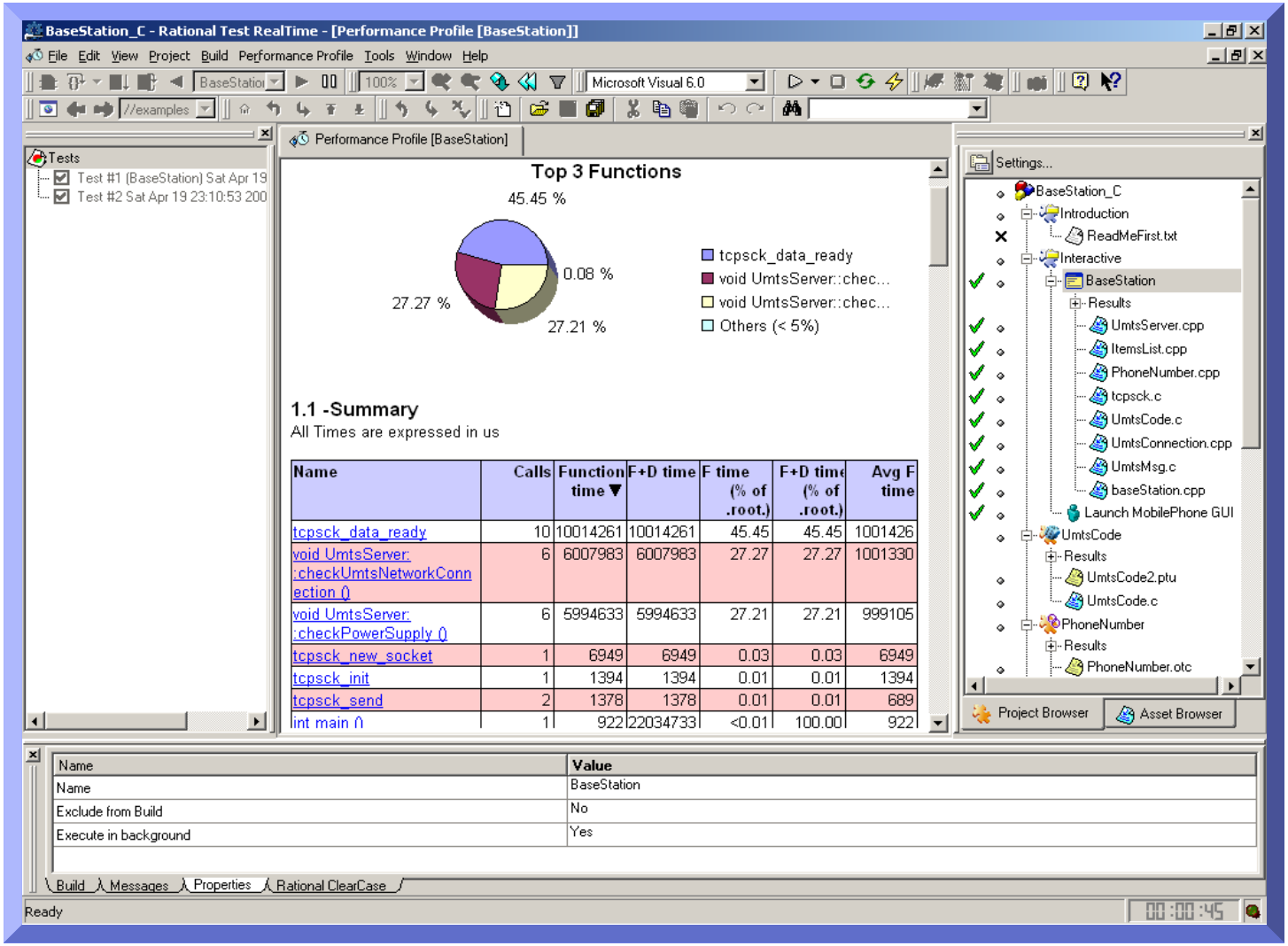
IBM Rational Test RealTime: *代码覆盖*



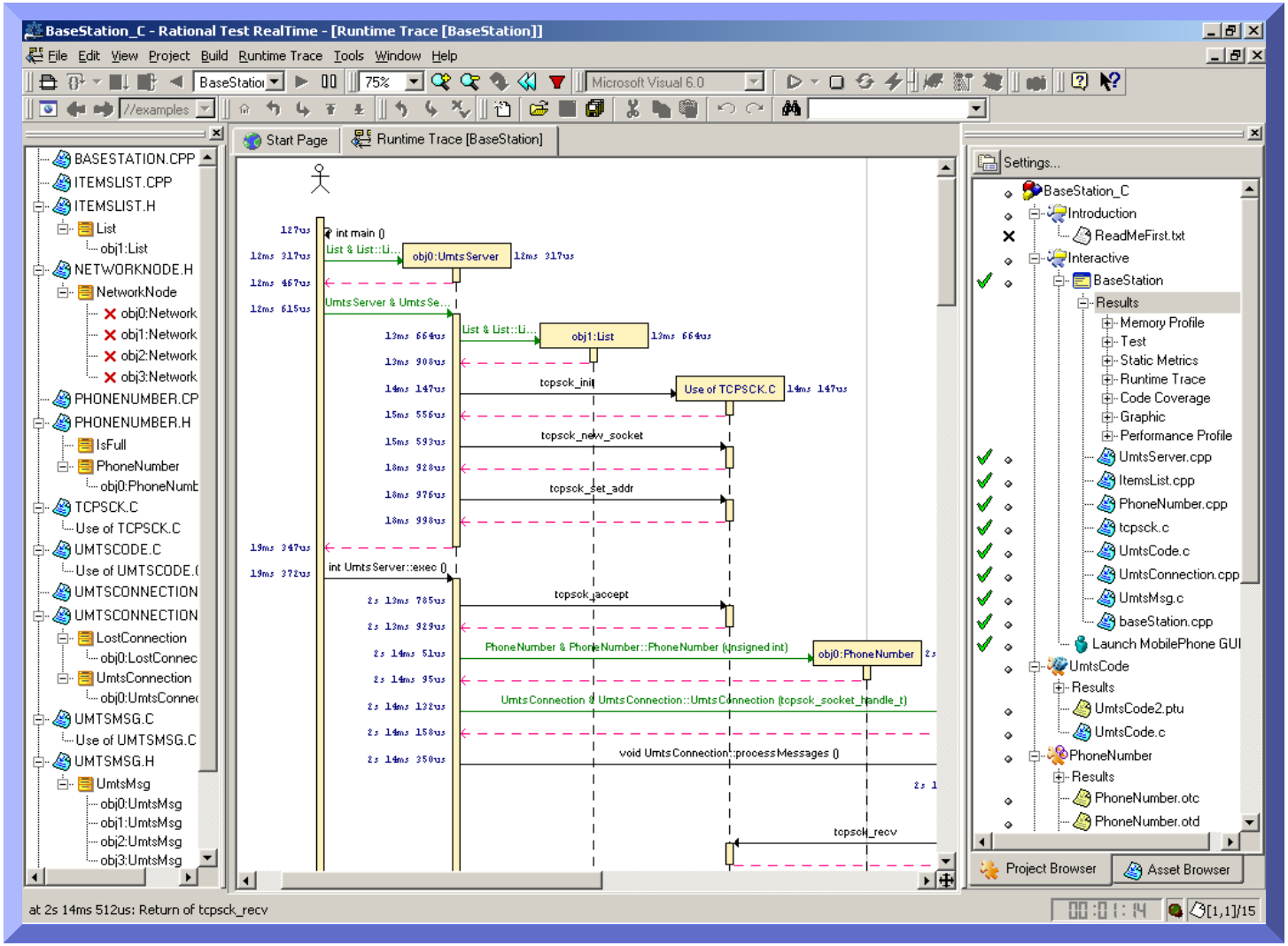
IBM Rational Test RealTime: *内存分析*



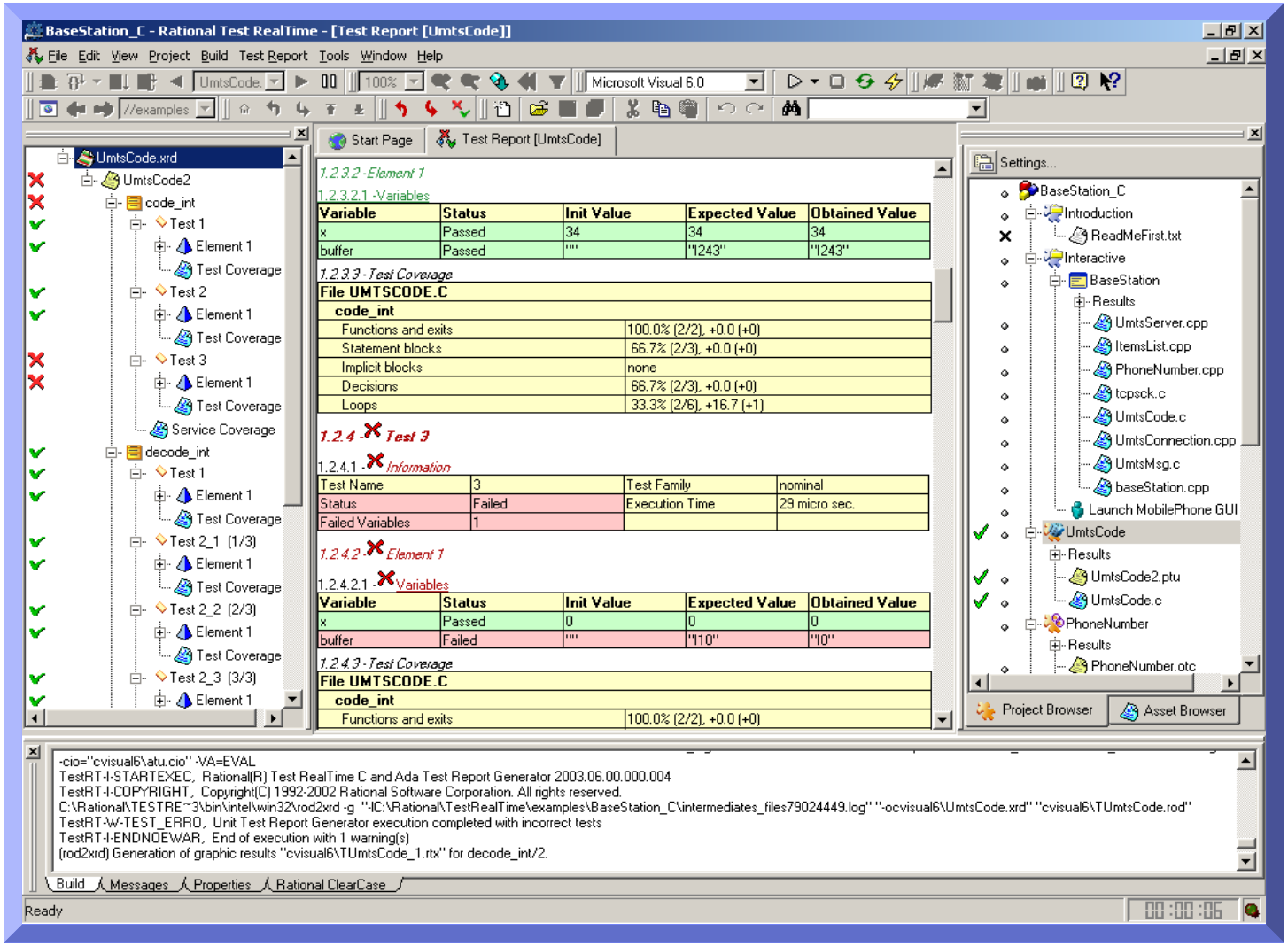
IBM Rational Test RealTime: *性能分析*



IBM Rational Test RealTime: *运行追踪图*



IBM Rational Test RealTime: *测试报告*

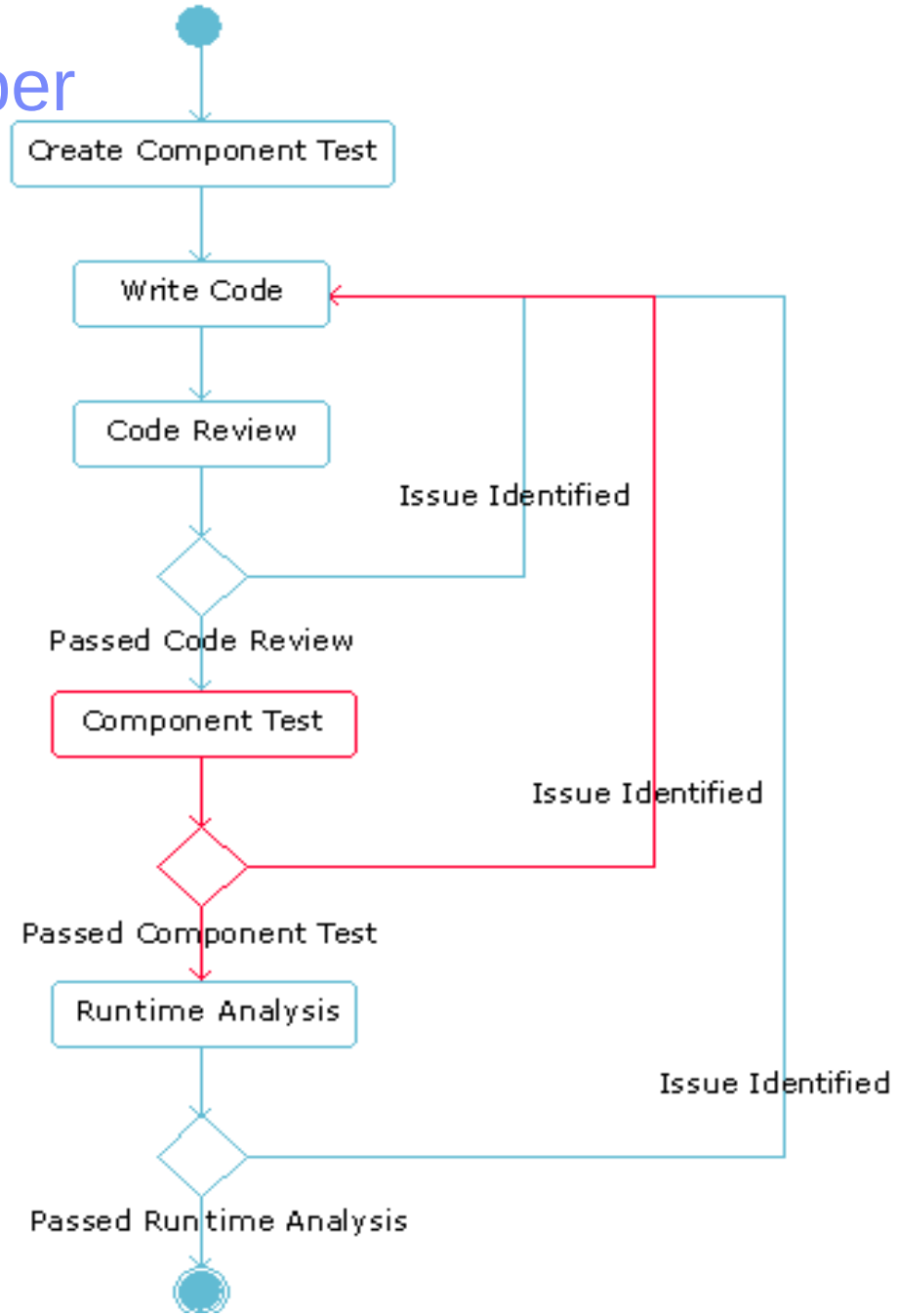


议程

- 基本概念
- 开发人员测试
  - 如何保证C/C++传统软件代码质量：Rational PurifyPlus
  - 如何保证C/C++嵌入式软件代码质量：Rational Test RealTime
  - 如何保证Java代码质量：Rational Application Developer
- 系统测试
  - 功能测试自动化：Rational Functional Tester
  - 性能测试：Rational Performance Tester

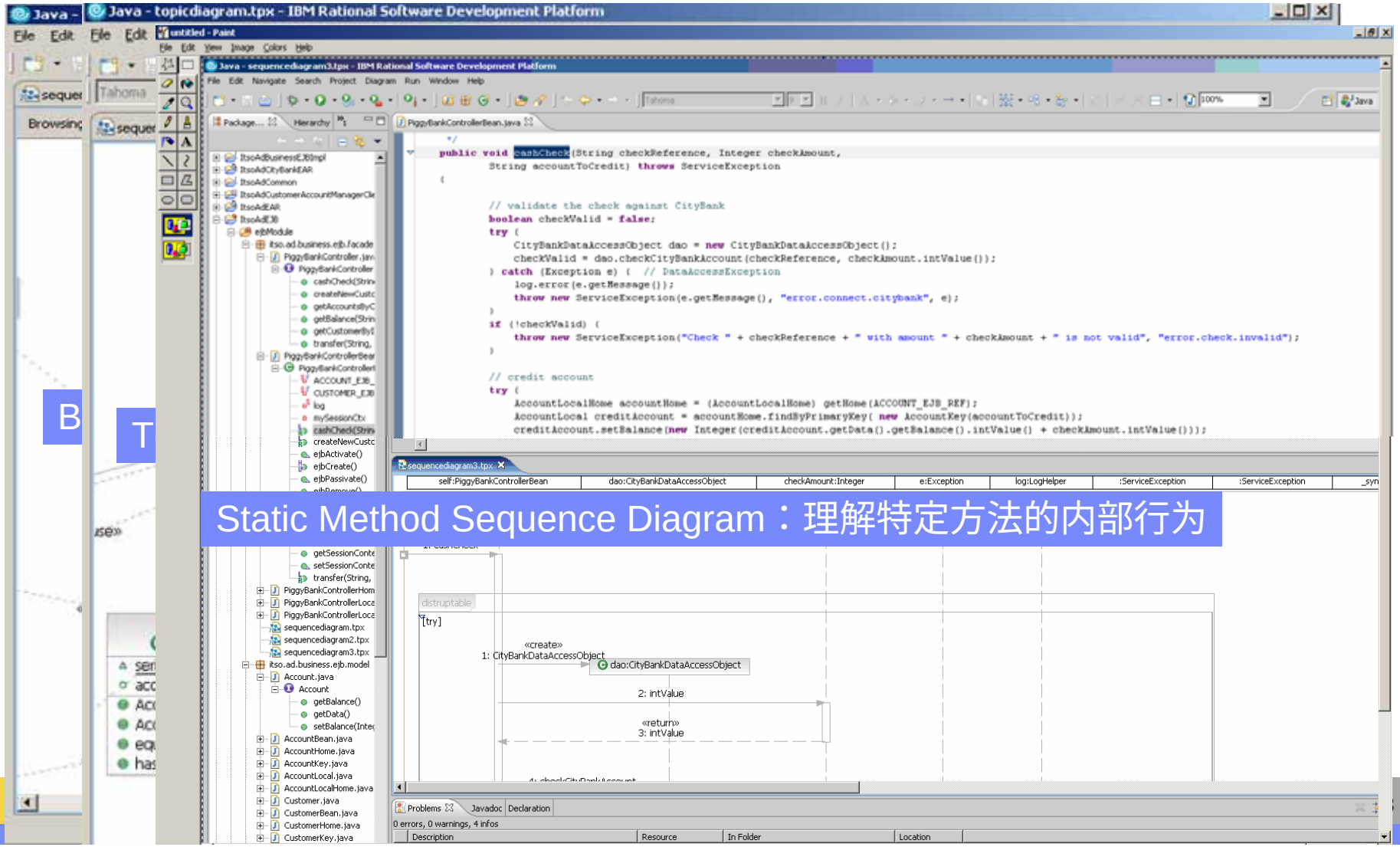
Rational Application Developer

- Java扩展集成开发环境
  - UML建模：类图和序列图
  - 集成开发环境
  - 开发人员测试手段
    - 静态分析
    - 组件测试
    - 运行分析功能



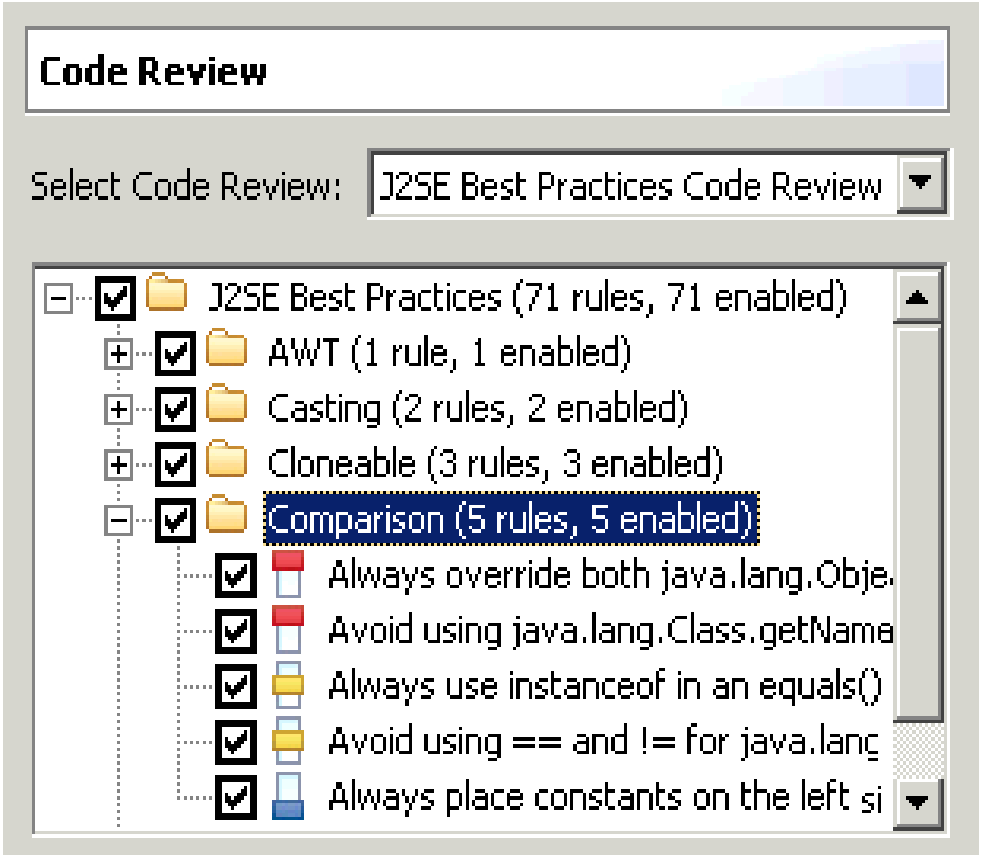
Rational Application Developer/静态分析

- 利用UML图对代码中的类和依赖以及方法的内部行为进行可视化



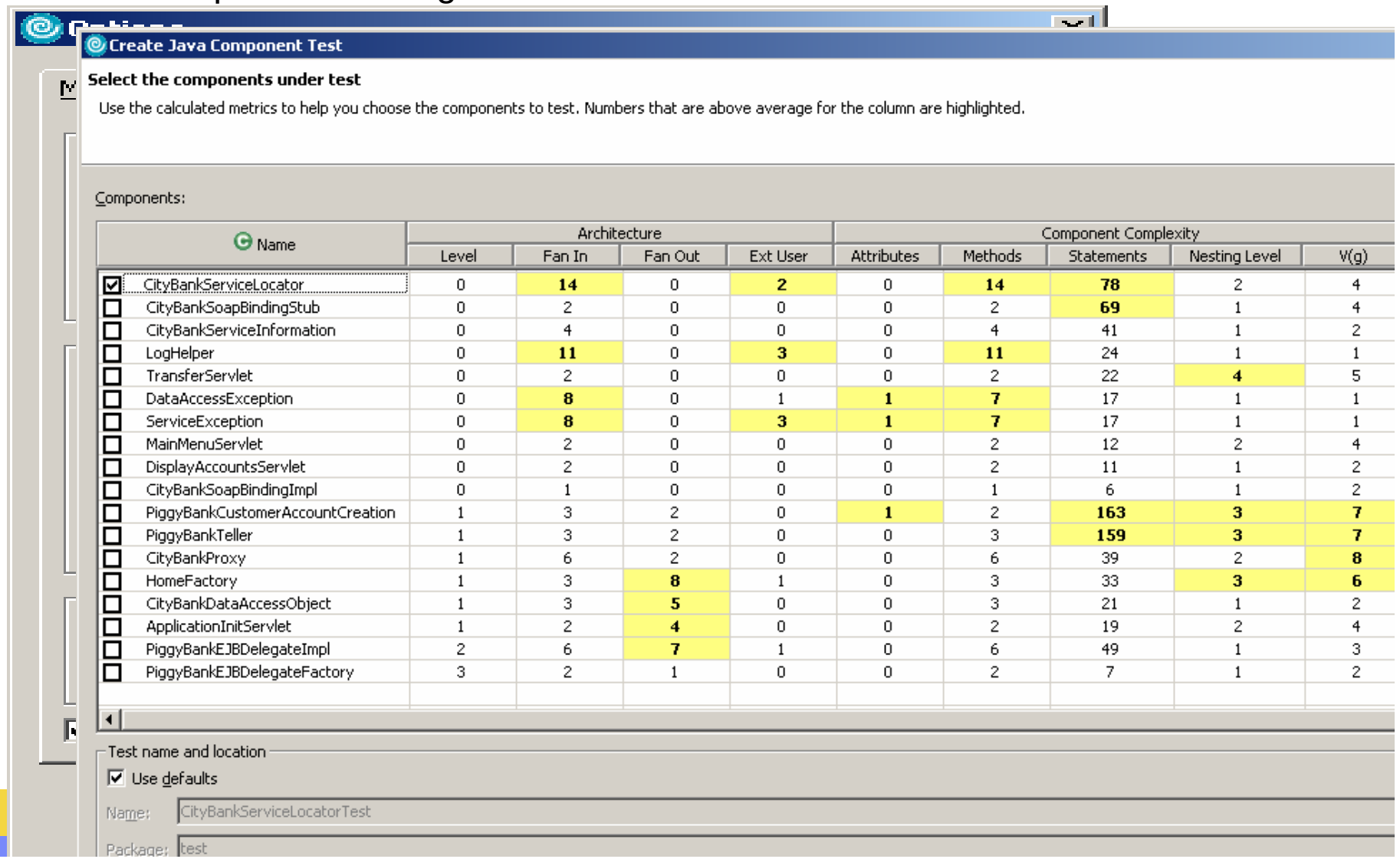
Rational Application Developer/静态分析

- 代码评审：依据一系列规则来自动发现代码中的问题
  - 提高代码评审的效率，而且保证所有代码的一致性
  - 发现代码中的缺陷
  - 检查代码是否遵循编码规范和经验
  - 提供每个缺陷的详细说明以及解决方案
  - 对某些问题进行自动修复。
  - 可自定义规则来确保代码遵循特定的编码标准



Rational Application Developer/静态分析

- 利用Component Testing来获得代码的度量



Name	Architecture				Component Complexity				
	Level	Fan In	Fan Out	Ext User	Attributes	Methods	Statements	Nesting Level	V(g)
CityBankServiceLocator	0	**14**	0	**2**	0	**14**	**78**	2	4
CityBankSoapBindingStub	0	2	0	0	0	2	**69**	1	4
CityBankServiceInformation	0	4	0	0	0	4	41	1	2
LogHelper	0	**11**	0	**3**	0	**11**	24	1	1
TransferServlet	0	2	0	0	0	2	22	**4**	5
DataAccessException	0	**8**	0	1	**1**	**7**	17	1	1
ServiceException	0	**8**	0	**3**	**1**	**7**	17	1	1
MainMenuServlet	0	2	0	0	0	2	12	2	4
DisplayAccountsServlet	0	2	0	0	0	2	11	1	2
CityBankSoapBindingImpl	0	1	0	0	0	1	6	1	2
PiggyBankCustomerAccountCreation	1	3	2	0	**1**	2	**163**	**3**	**7**
PiggyBankTeller	1	3	2	0	0	3	**159**	**3**	**7**
CityBankProxy	1	6	2	0	0	6	39	2	**8**
HomeFactory	1	3	**8**	1	0	3	33	**3**	**6**
CityBankDataAccessObject	1	3	**5**	0	0	3	21	1	2
ApplicationInitServlet	1	2	**4**	0	0	2	19	2	4
PiggyBankEJBDelegateImpl	2	6	**7**	1	0	6	49	1	3
PiggyBankEJBDelegateFactory	3	2	1	0	0	2	7	1	2

Rational Application Developer/组件测试

- 基于JUnit测试框架
- 自动生成测试驱动代码
- 测试代码和测试数据分离
- 直观的测试报告

ZipcodeTest | ZipcodeTest.java

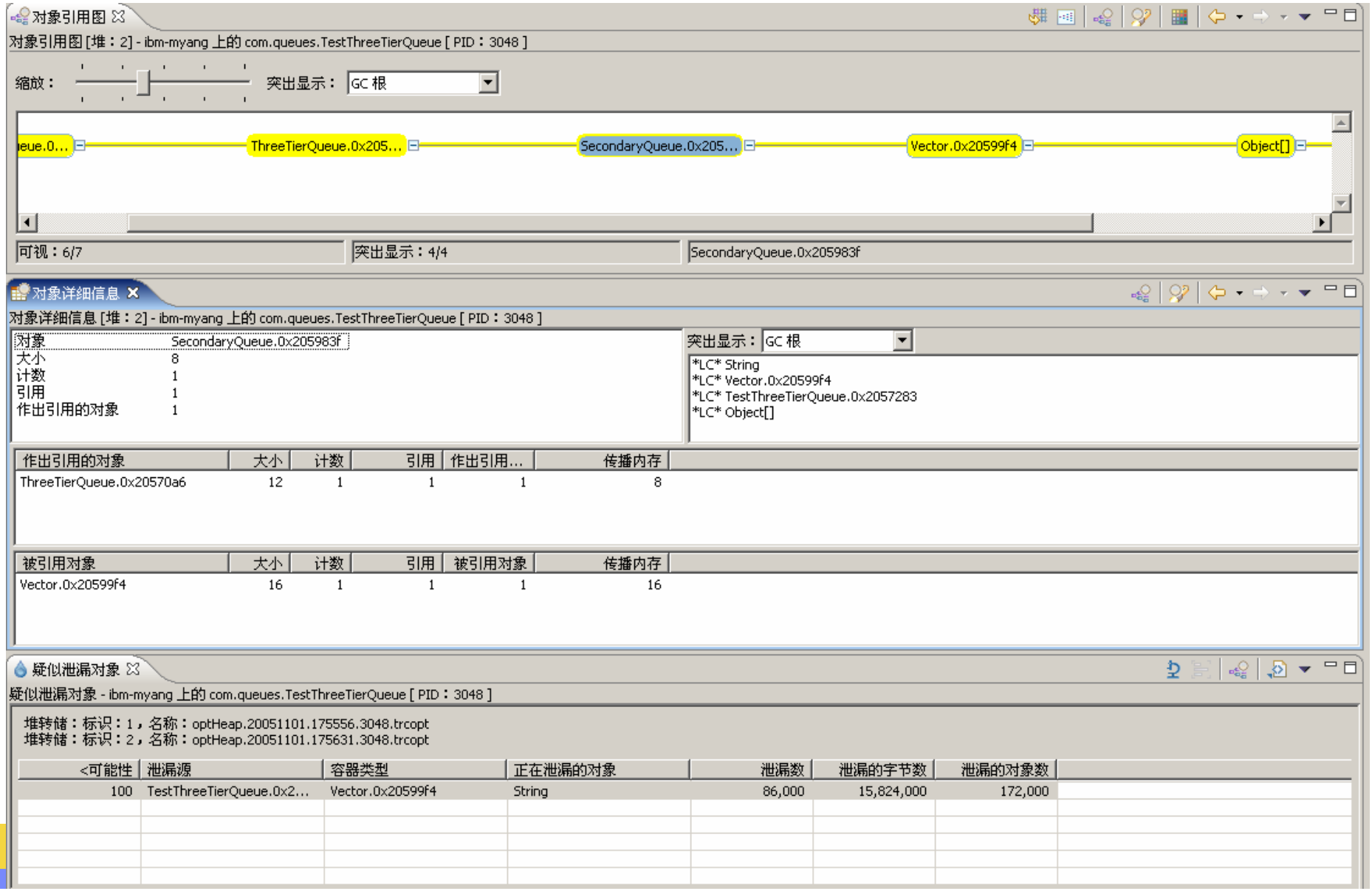
```
package test;

import junit.framework.*;

public class ZipcodeTest extends TestCase {

    public void test_isZipcode() {

        Zipcode objZipcode = new Zipcode();
        String zipcode = "";
        boolean retValue = false;
        retValue = objZipcode.isZipcode(zipcode);
    }

    public void test_getCity() {

        Zipcode objZipcode = new Zipcode();
        String zipcode = "";
```

Tasks | Test Data Comparator

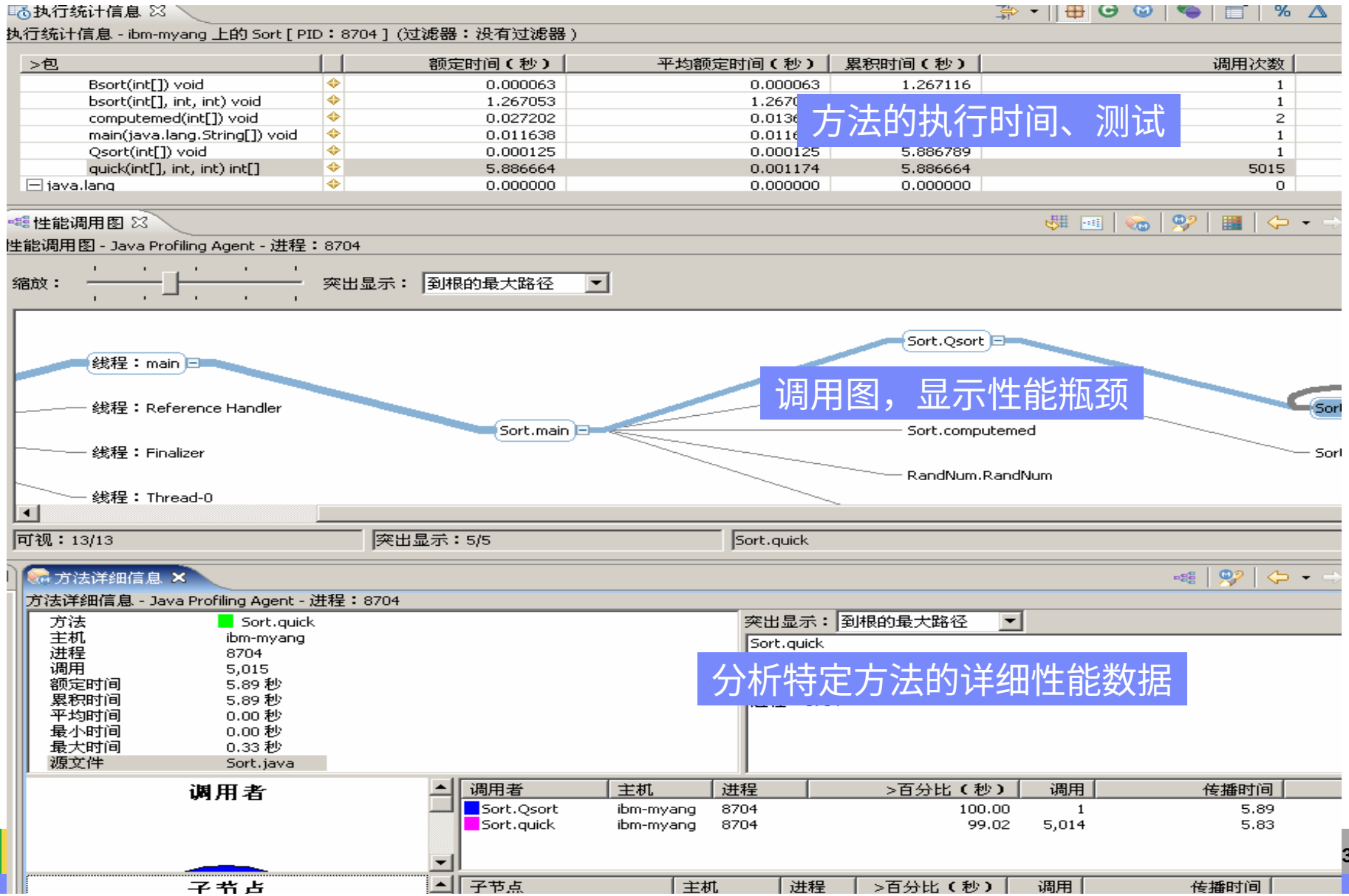
Action	Type	New Test Data 2		
		In	Expected	Actual
objZipcode = new Zipcode()				
objZipcode	com.acme.commo...			
<expected exception>	Throwable		<no exception>	<no exception>
retValue = objZipcode.isZipcode(zipcode)				
zipcode	java.lang.String	123456-789		
retValue	boolean		true	false
<expected exception>	Throwable		<no exception>	<no exception>

Test Data Comparator loaded.

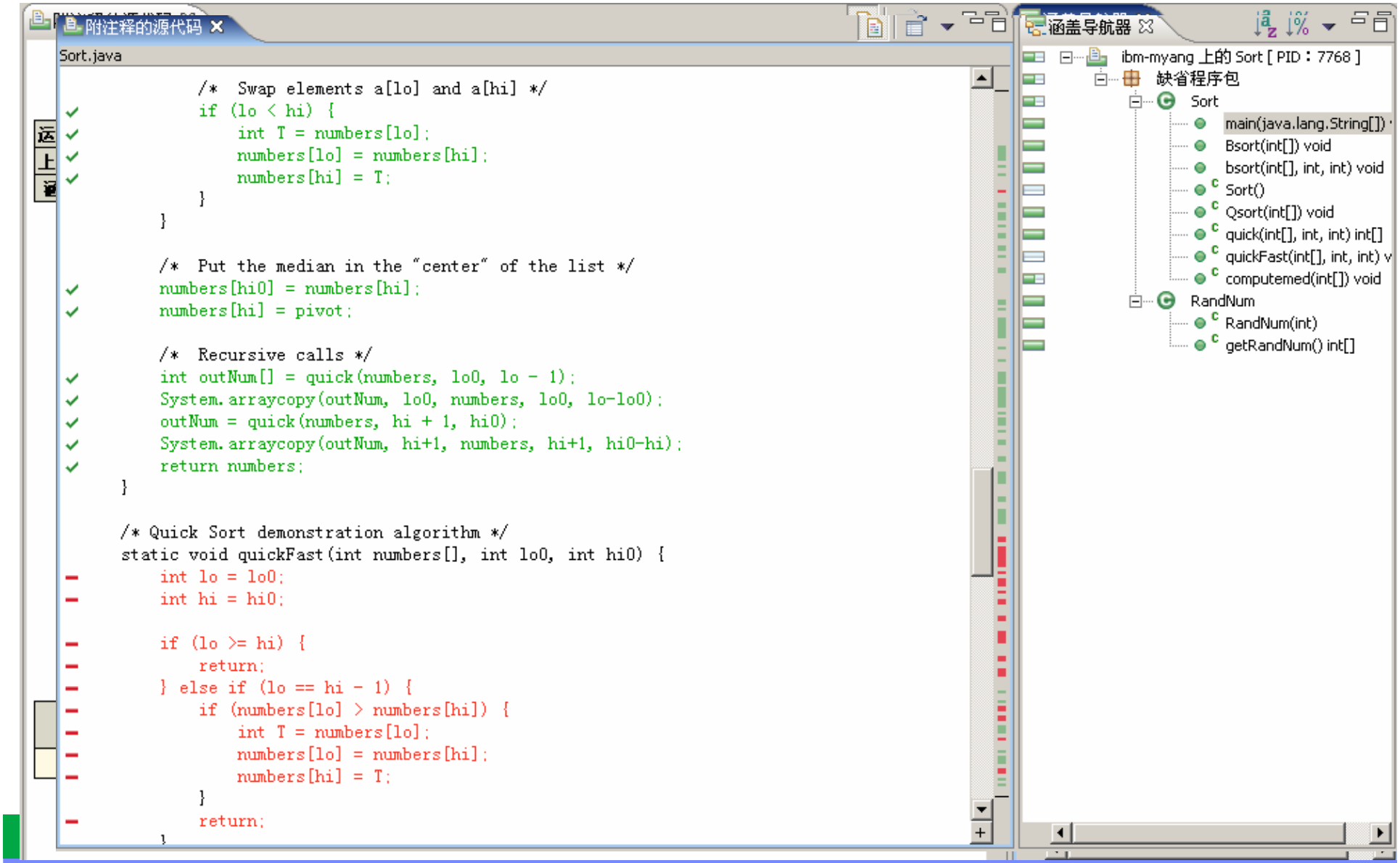
Rational Application Developer/运行时分析：内存泄漏



Rational Application Developer/运行时分析：性能分析



Rational Application Developer/运行时分析：代码覆盖



议程

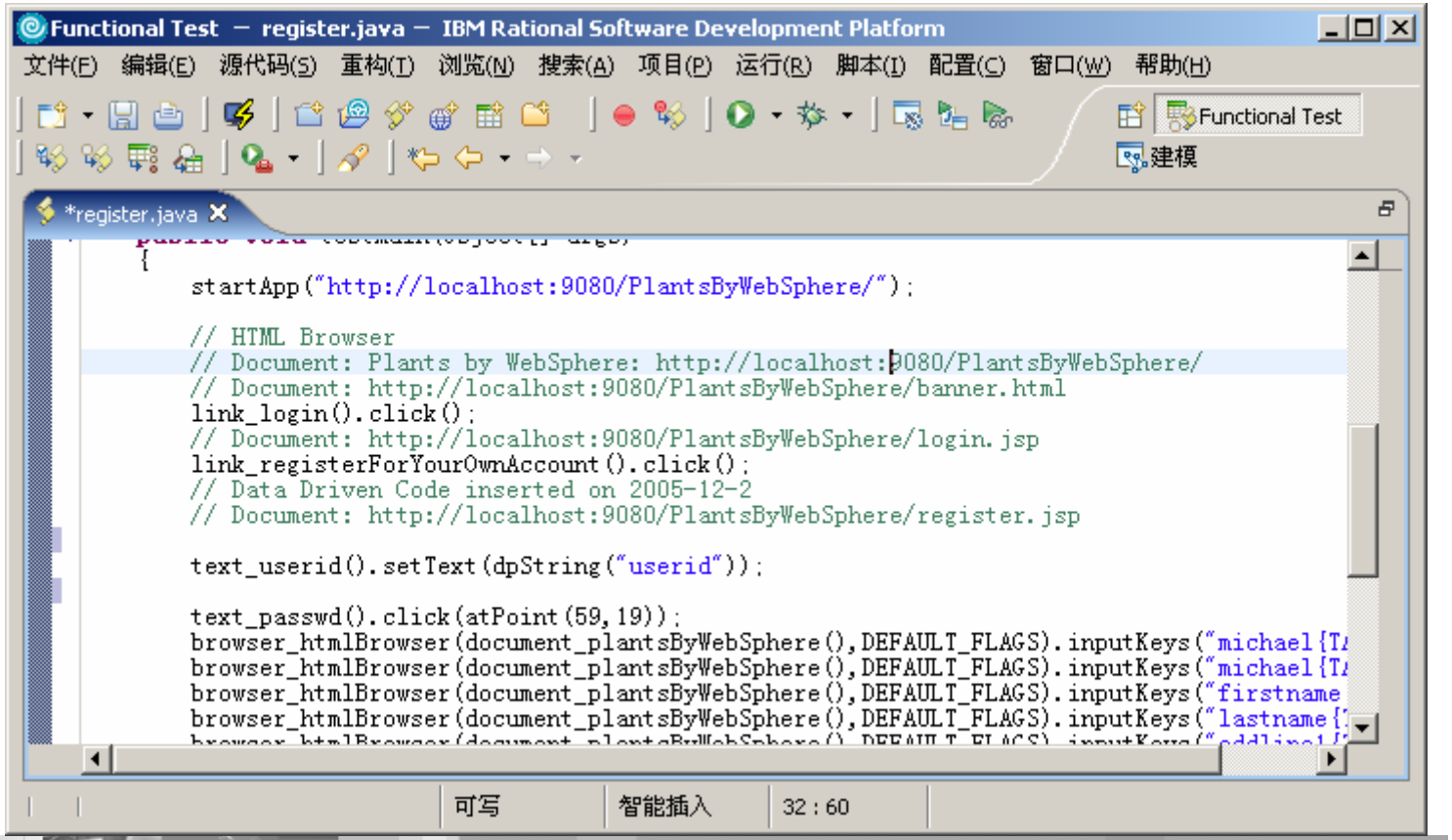
- 基本概念
- 开发人员测试
  - 如何保证C/C++传统软件代码质量：Rational PurifyPlus
  - 如何保证C/C++嵌入式软件代码质量：Rational Test RealTime
  - 如何保证Java代码质量：Rational Application Developer
- 系统测试
  - 功能测试自动化：Rational Functional Tester
  - 性能测试：Rational Performance Tester

GUI功能测试自动化工具：Rational Functional Tester

- 支持的应用类型：
  - Visual Studio .NET 2003
  - Java Application
  - Browser
  - Terminal-Based(3270, 5250, VT100)
- 测试脚本语言
  - Java
  - VB .NET
- 脚本环境
  - Eclipse 3.0
  - Visual Studio .NET

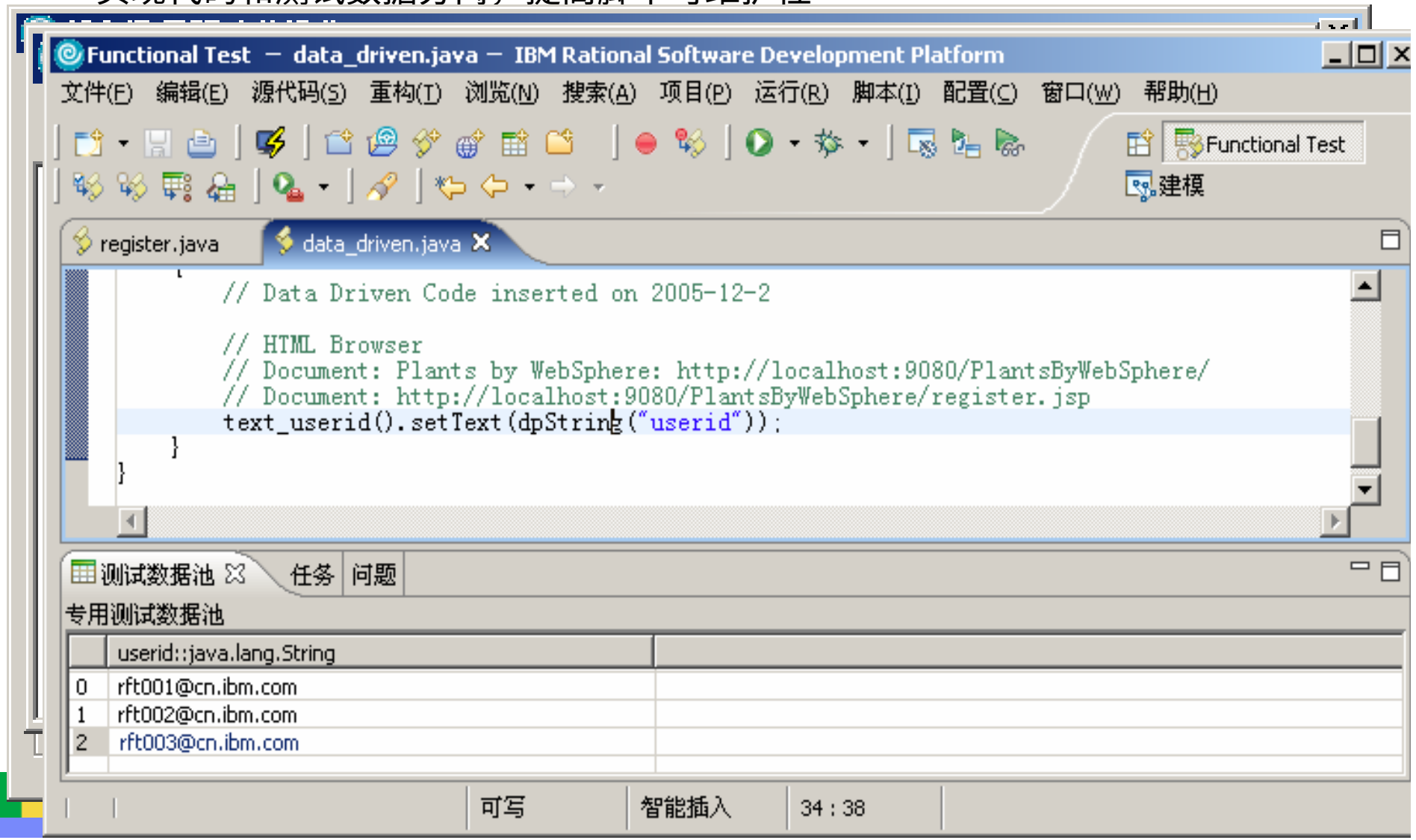
基于Java或VB .NET的测试脚本

- 无须学习特有的测试脚本语法和API
- 充分利用开发环境的强大的编辑和调试功能
- 开发人员也可以用RFT进行自动化测试

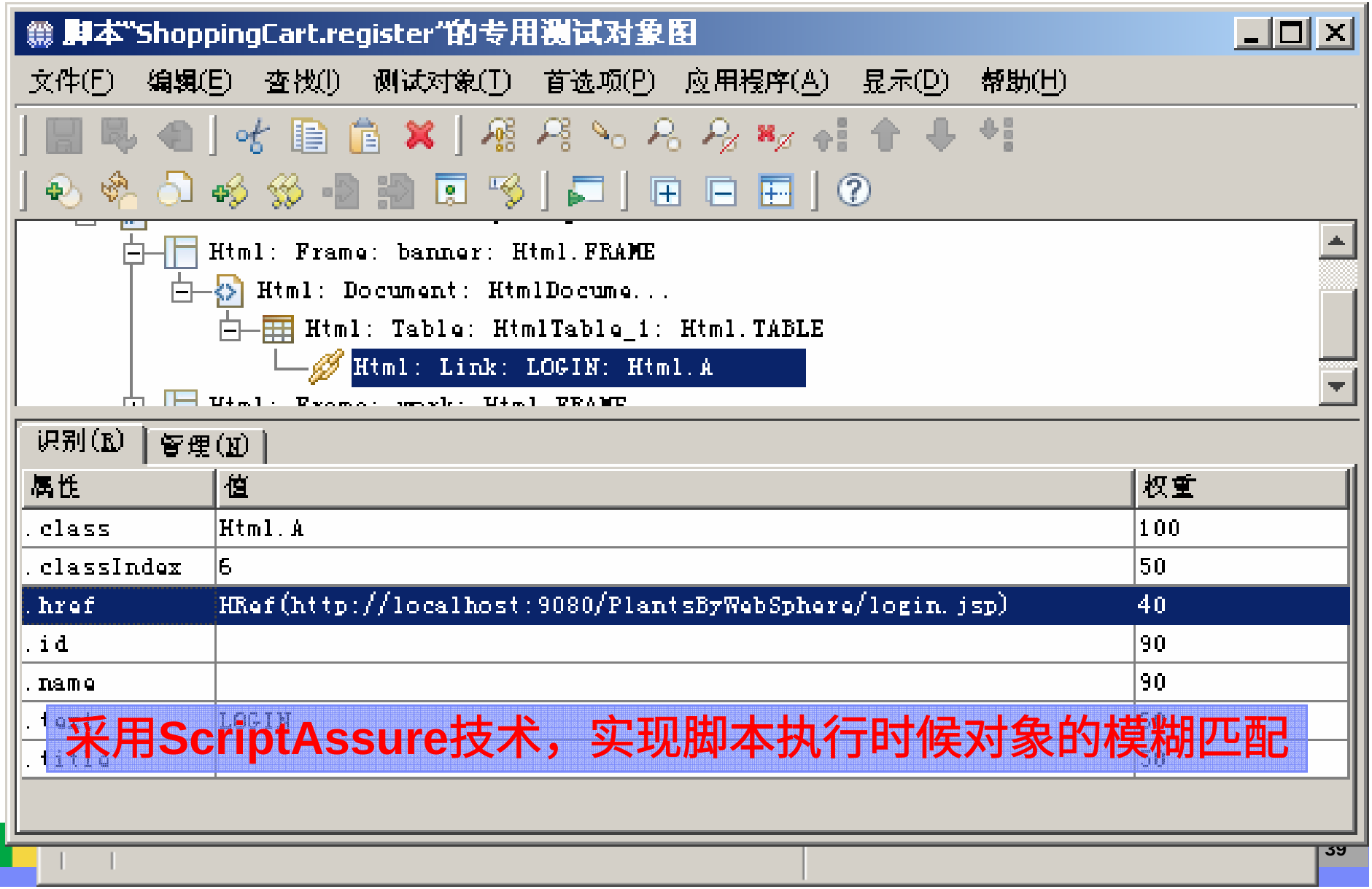


无需编程，快速实现数据驱动测试

- 实现代码和测试数据分离，提高脚本可维护性



利用Object Map降低脚本维护成本



采用模式匹配快速实现结果验证

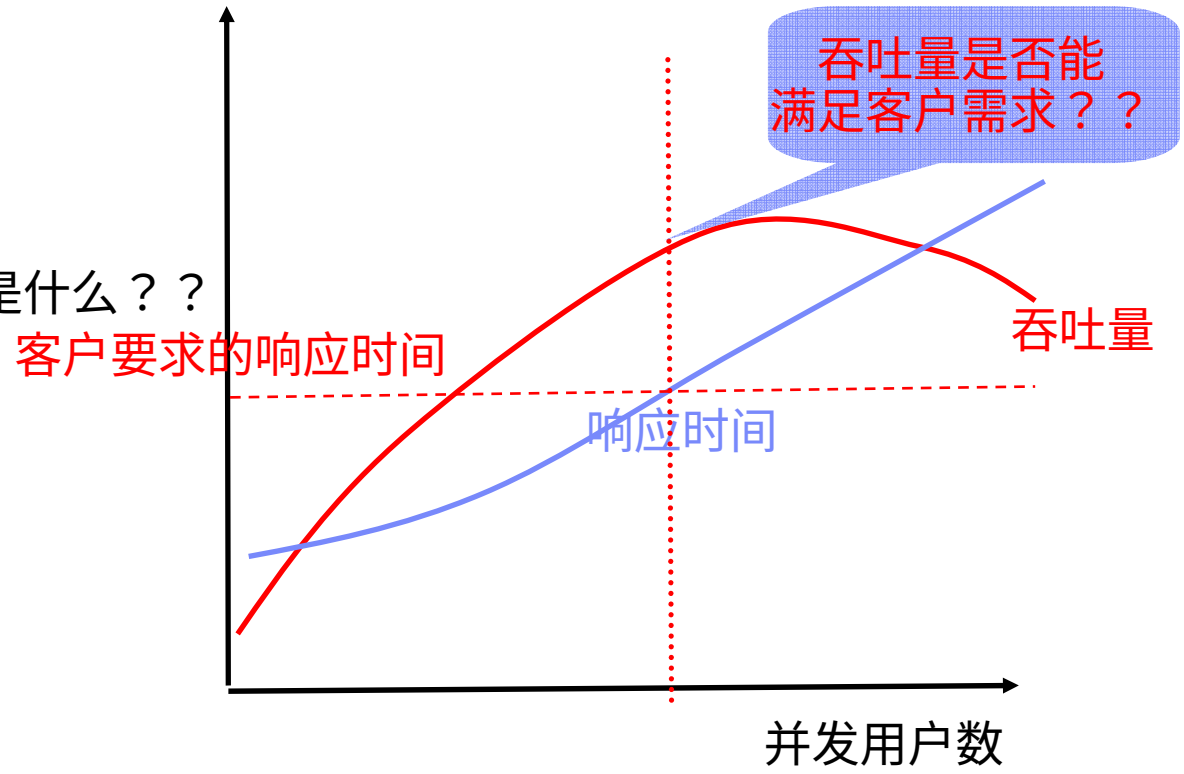
- 动态数据模式匹配
  - 如对“Order ID 230”, 验证点中的模式串为“Order ID ###” 或 “Order ID 2##”

议程

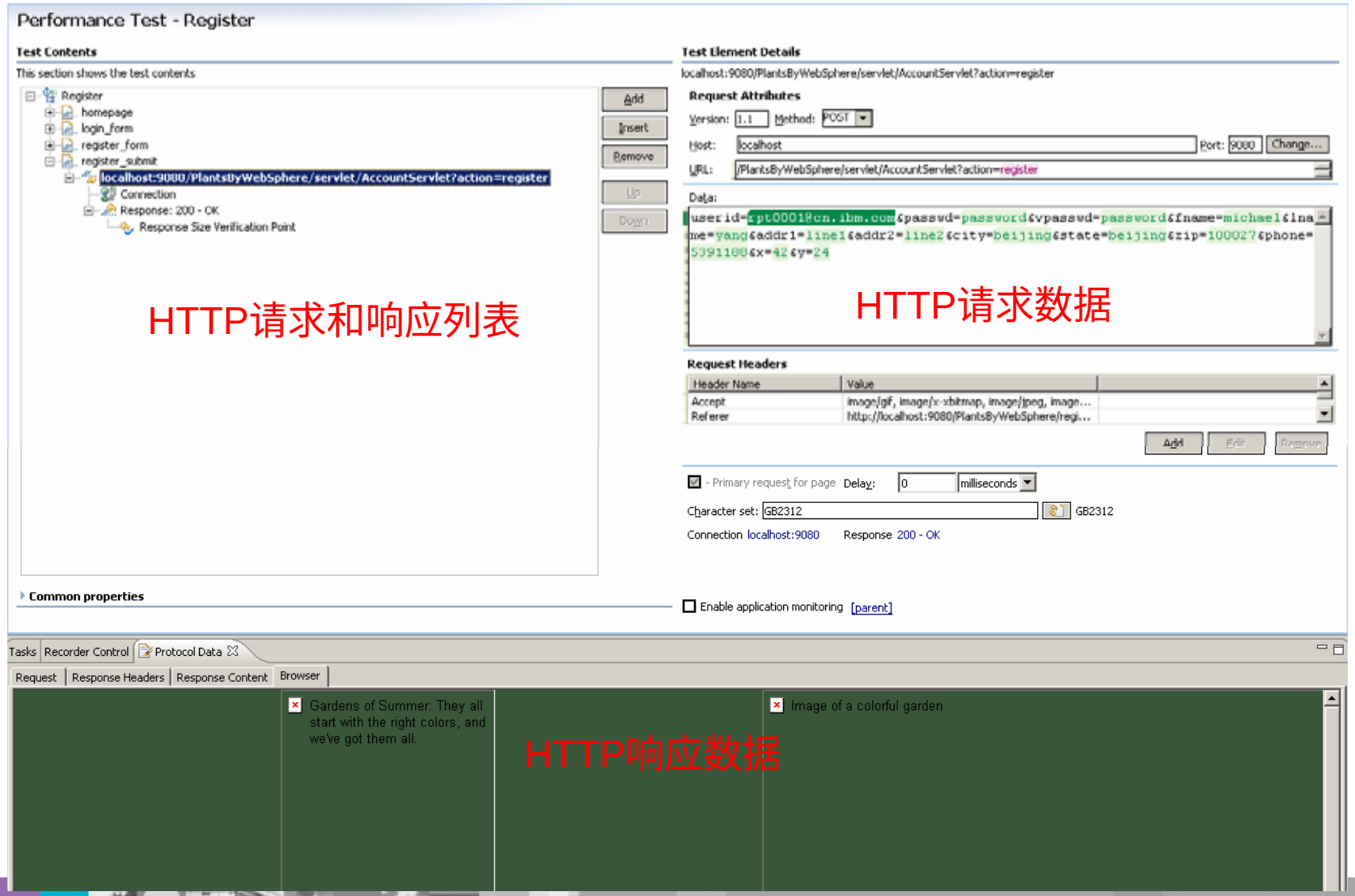
- 基本概念
- 开发人员测试
  - 如何保证C/C++传统软件代码质量：Rational PurifyPlus
  - 如何保证C/C++嵌入式软件代码质量：Rational Test RealTime
  - 如何保证Java代码质量：Rational Application Developer
- 系统测试
  - 功能测试自动化：Rational Functional Tester
  - 性能测试：Rational Performance Tester

基于Rational Performance Tester的性能测试

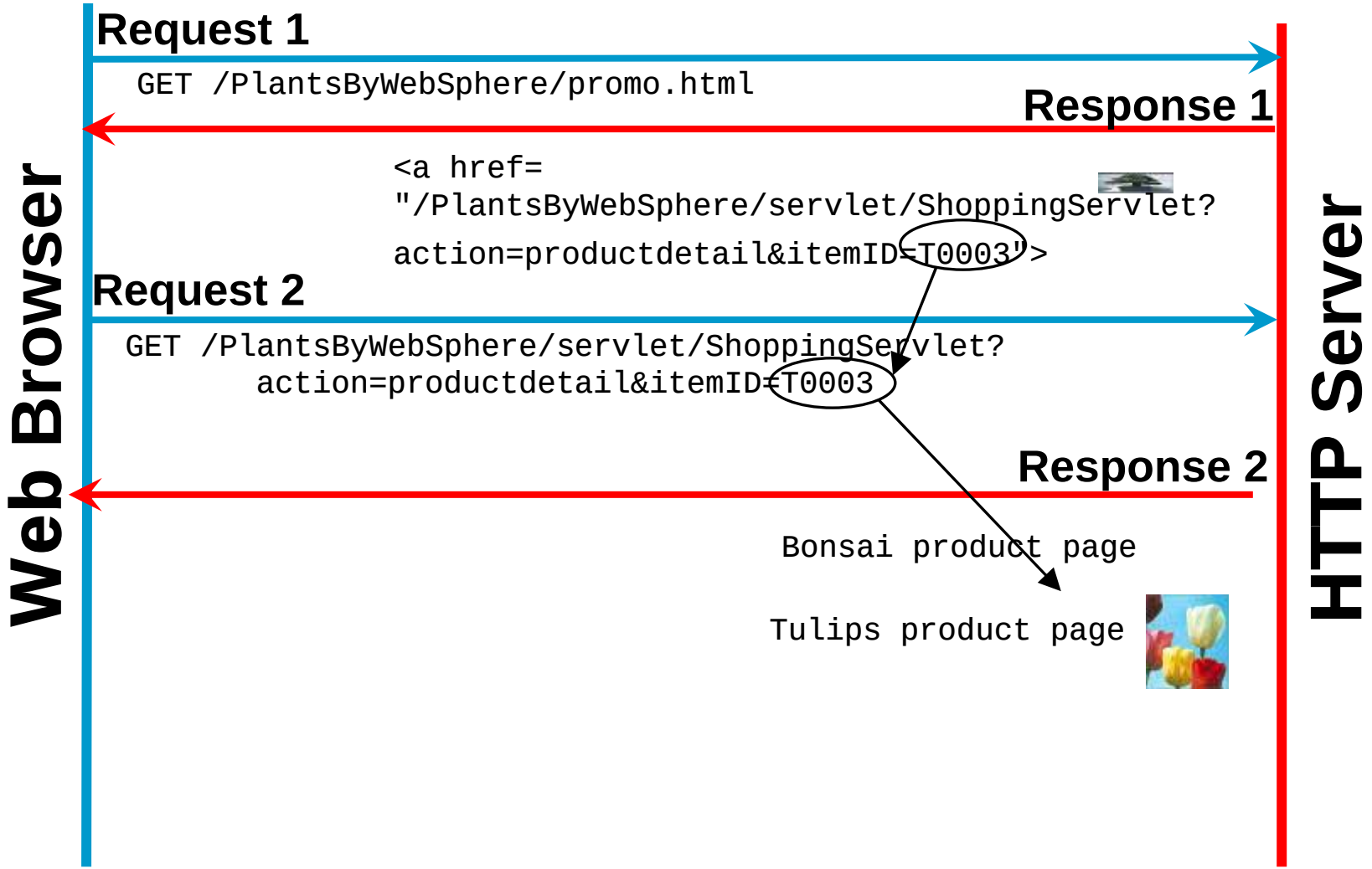
- 基于Eclipse的性能测试工具
- 支持协议：
  - HTTP
  - Siebel
  - SAP
- Web应用的性能测试策略是什么？？
- 基于RPT的性能测试过程
  - 录制测试
  - 修改测试
  - 创建负载模型
  - 执行测试
  - 分析结果



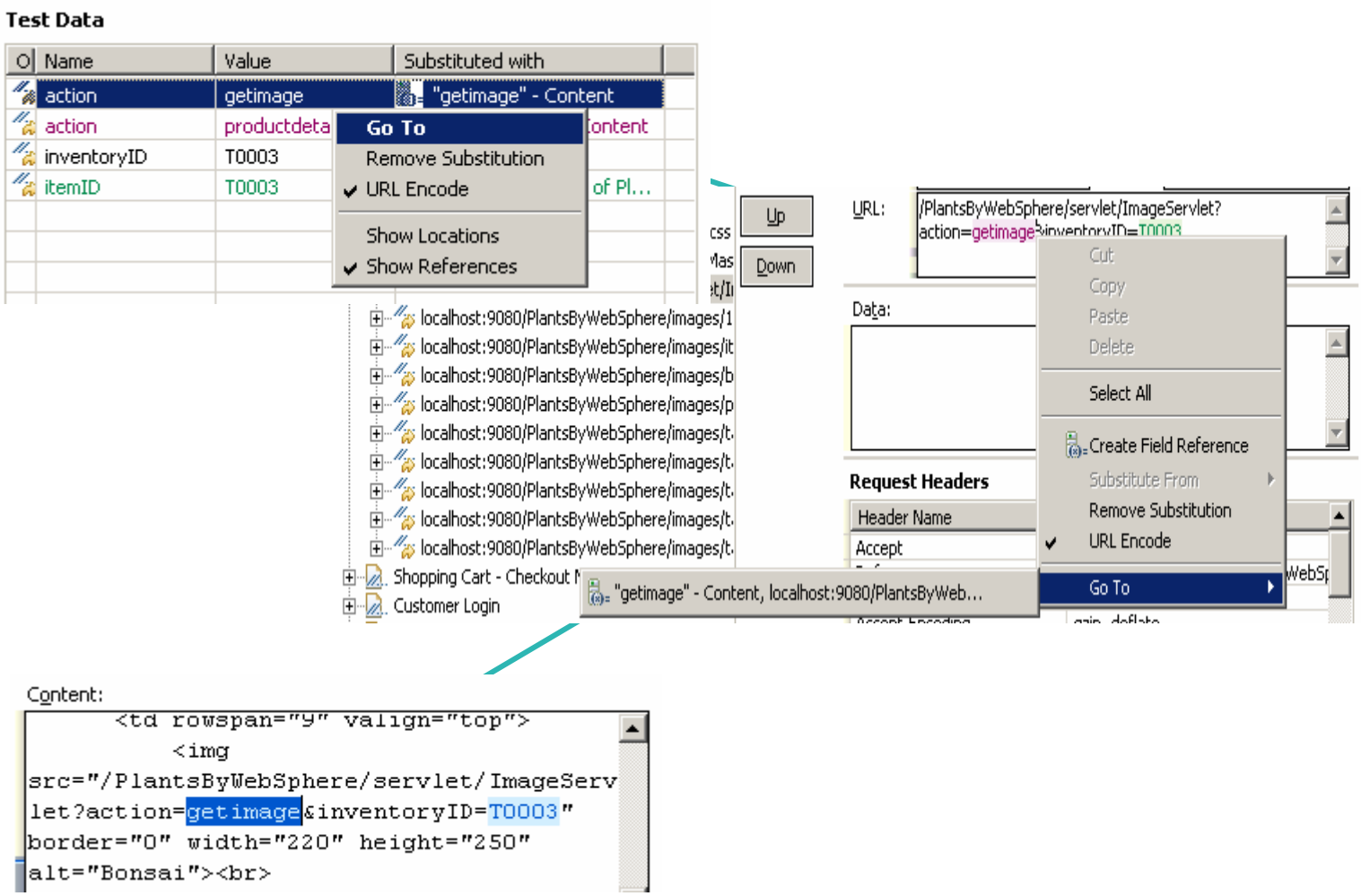
录制测试


Performance Test - Register
Test Contents
This section shows the test contents
Register
homepage
login_form
register_form
register_submit
localhost:9080/PlantsByWebSphere/servlet/AccountServlet?action=register
Connection
Response: 200 - OK
Response Size Verification Point
Add
Insert
Remove
Up
Down
HTTP请求和响应列表
Test Element Details
localhost:9080/PlantsByWebSphere/servlet/AccountServlet?action=register
Request Attributes
Version: 1.1 Method: POST
Host: localhost Port: 9080 Change...
URL: /PlantsByWebSphere/servlet/AccountServlet?action=register
Data:
userid=cpt0001@cn.ibm.com&passwd=password&vpasswd=password&fname=michael&lname=yang&addr1=line1&addr2=line2&city=beijing&state=beijing&zip=100027&phone=5391100&x=42&y=24
HTTP请求数据
Request Headers
Header Name
Value
Accept
image/gif, image/x-xbitmap, image/jpeg, image...
Referer
http://localhost:9080/PlantsByWebSphere/regi...
Add
Edit
Remove
- Primary request for page Delay: 0 milliseconds
Character set: GB2312
GB2312
Connection localhost:9080 Response 200 - OK
Common properties
Enable application monitoring [parent]
Tasks
Recorder Control
Protocol Data
Request
Response Headers
Response Content
Browser
Gardens of Summer: They all start with the right colors, and we've got them all.
HTTP响应数据
Image of a colorful garden

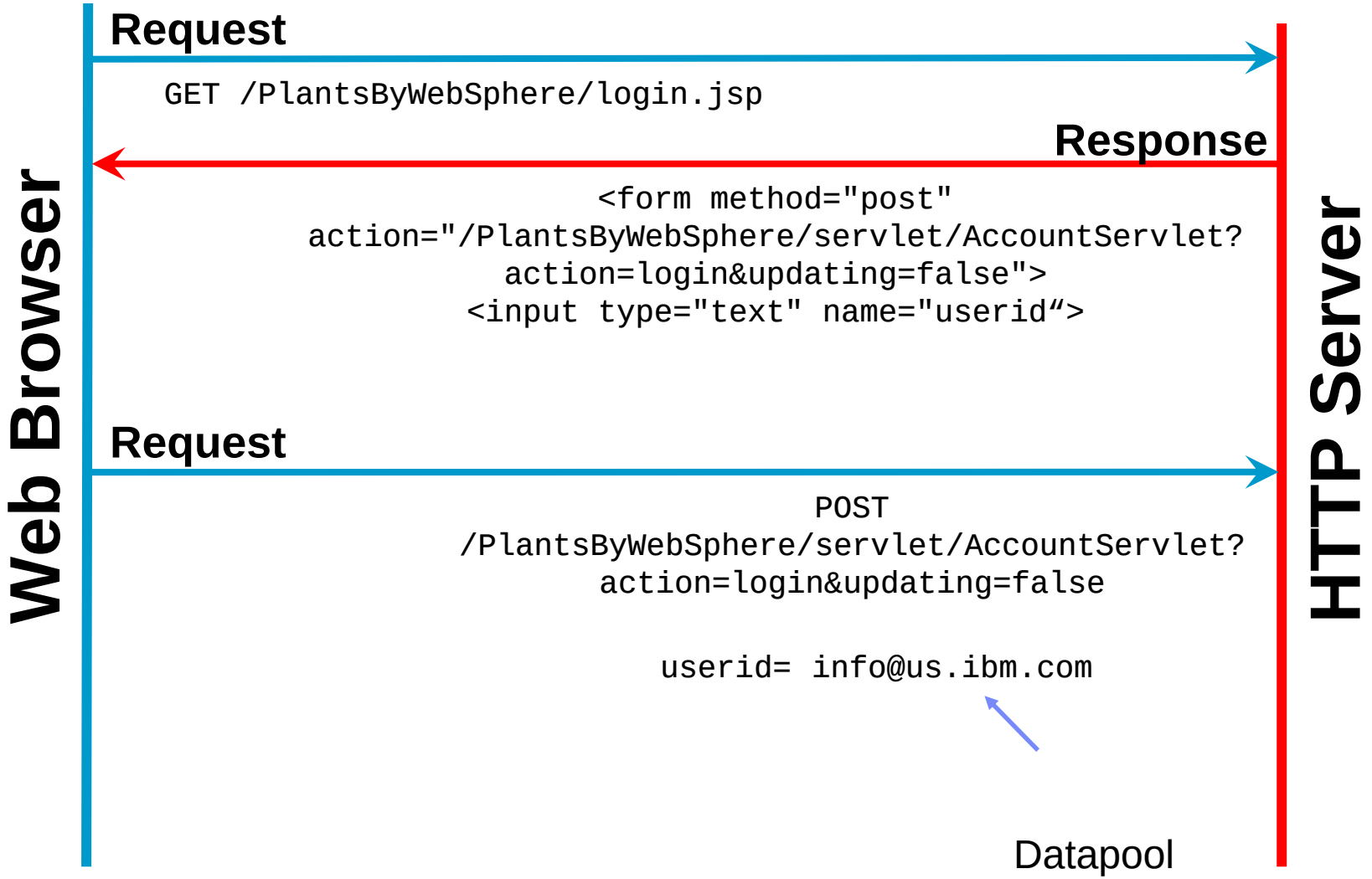
修改测试：数据关联


Request 1
GET /PlantsByWebSphere/promo.html
Response 1
<a href=
"/PlantsByWebSphere/servlet/ShoppingServlet?
action=productdetail&itemID=T0003">
Request 2
GET /PlantsByWebSphere/servlet/ShoppingServlet?
action=productdetail&itemID=T0003
Response 2
Bonsai product page
Tulips product page
Web Browser
HTTP Server

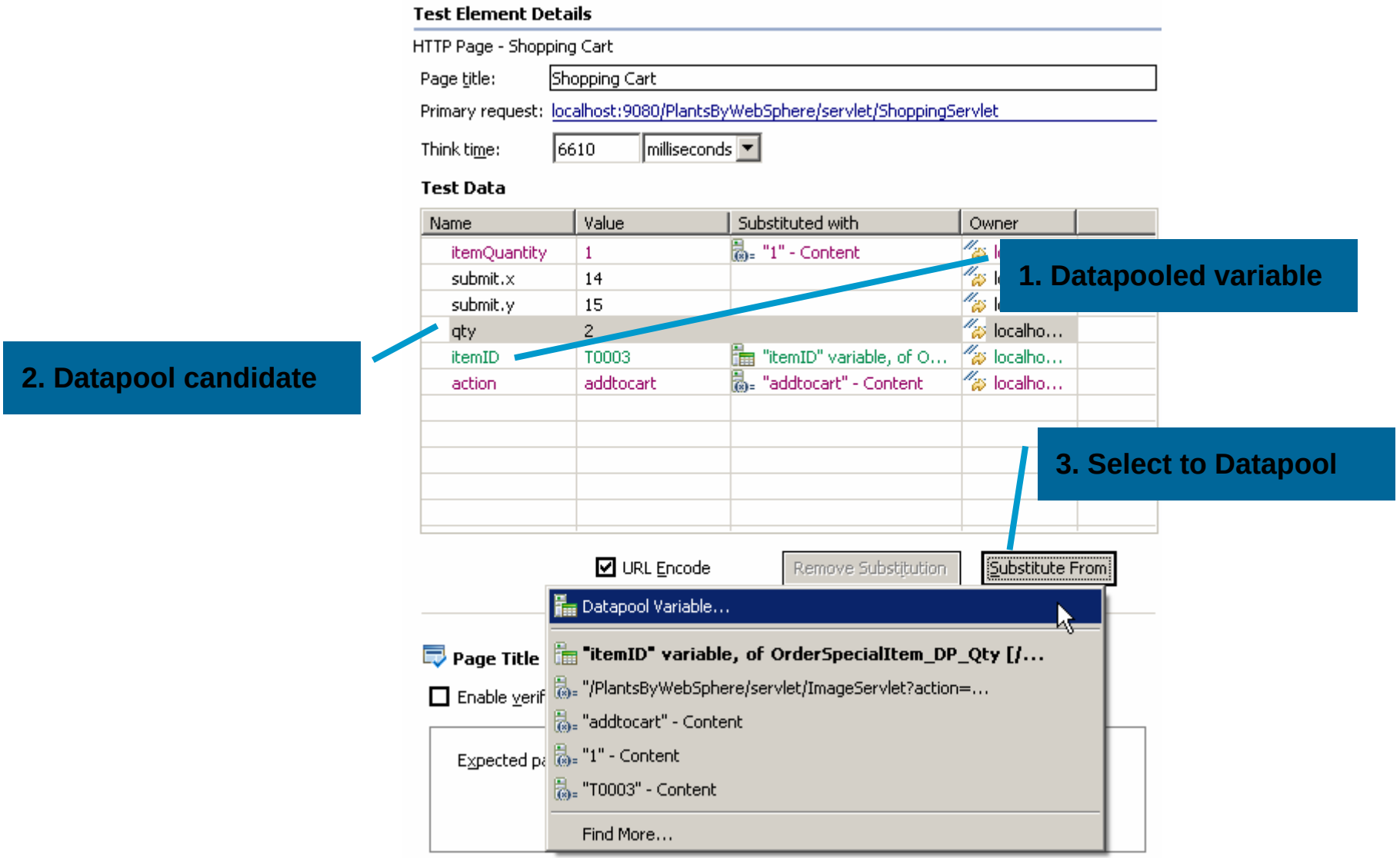
RPT: 自动建立数据关联



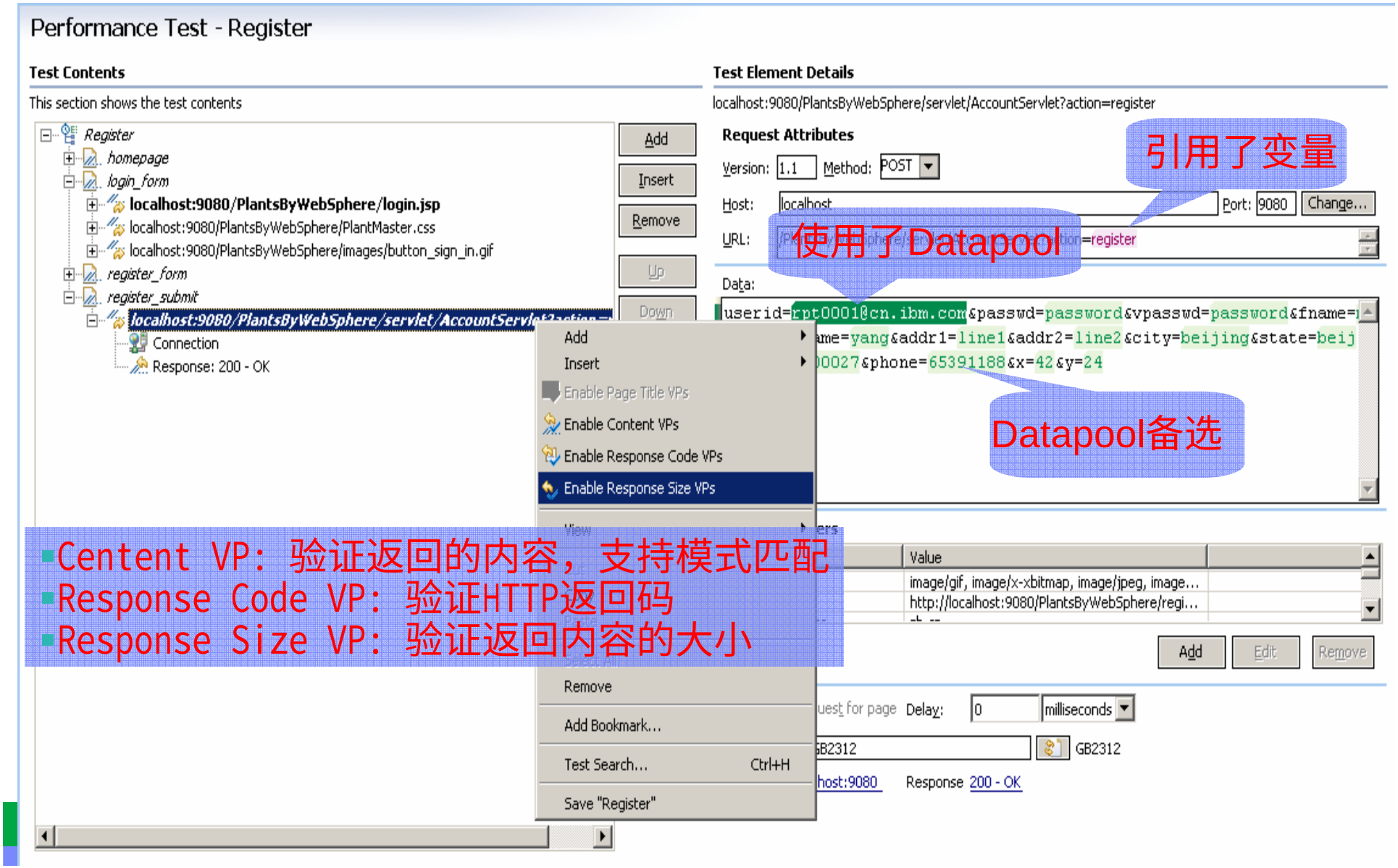
数据参数化


Request
GET /PlantsByWebSphere/login.jsp
Response
<form method="post"
action="/PlantsByWebSphere/servlet/AccountServlet?
action=login&updating=false">
<input type="text" name="userid“>
Request
POST
/PlantsByWebSphere/servlet/AccountServlet?
action=login&updating=false
userid= info@us.ibm.com
Datapool
Web Browser
HTTP Server

利用DataPool进行参数化

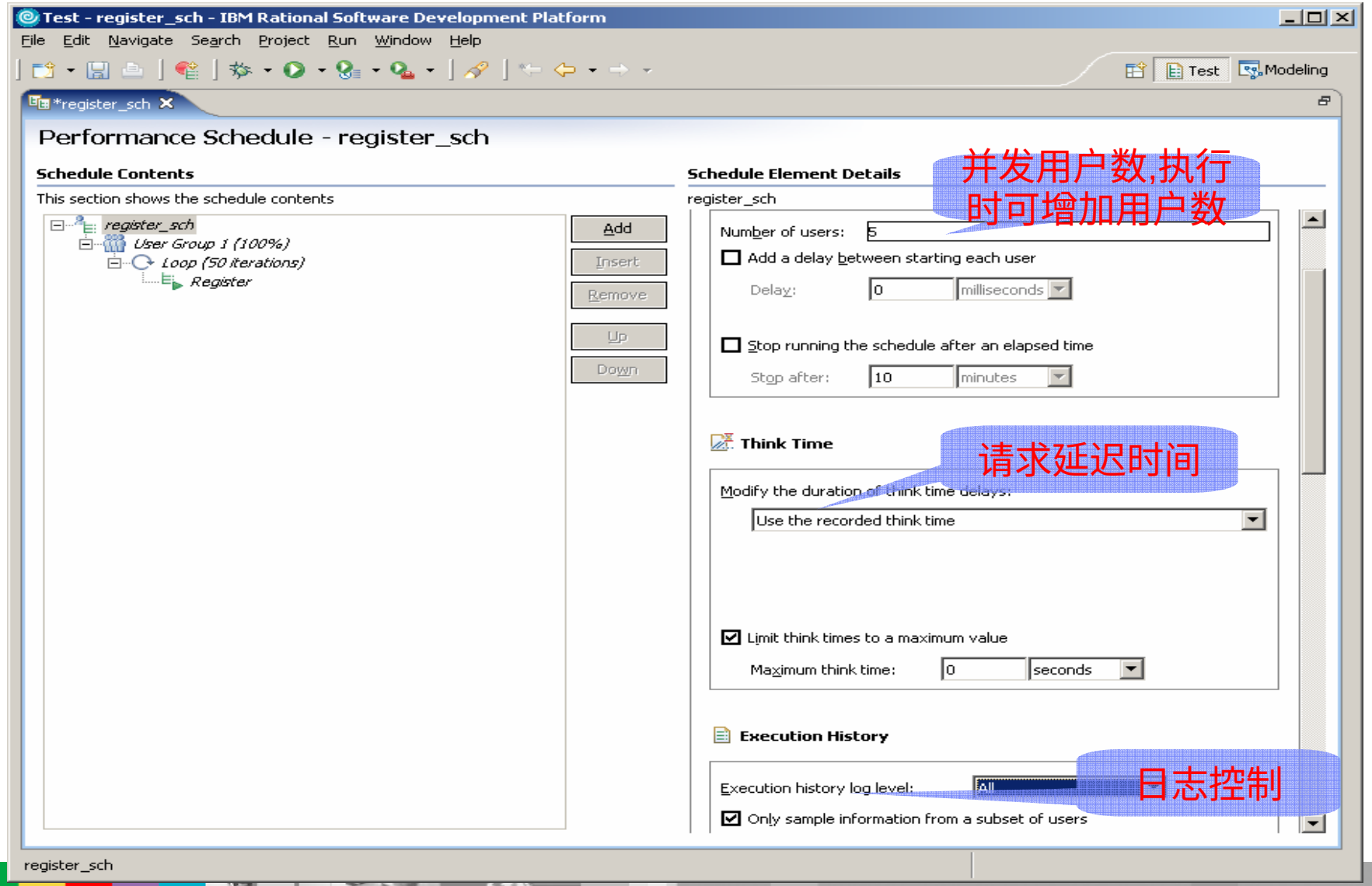


快速设置验证点（Verification Point)

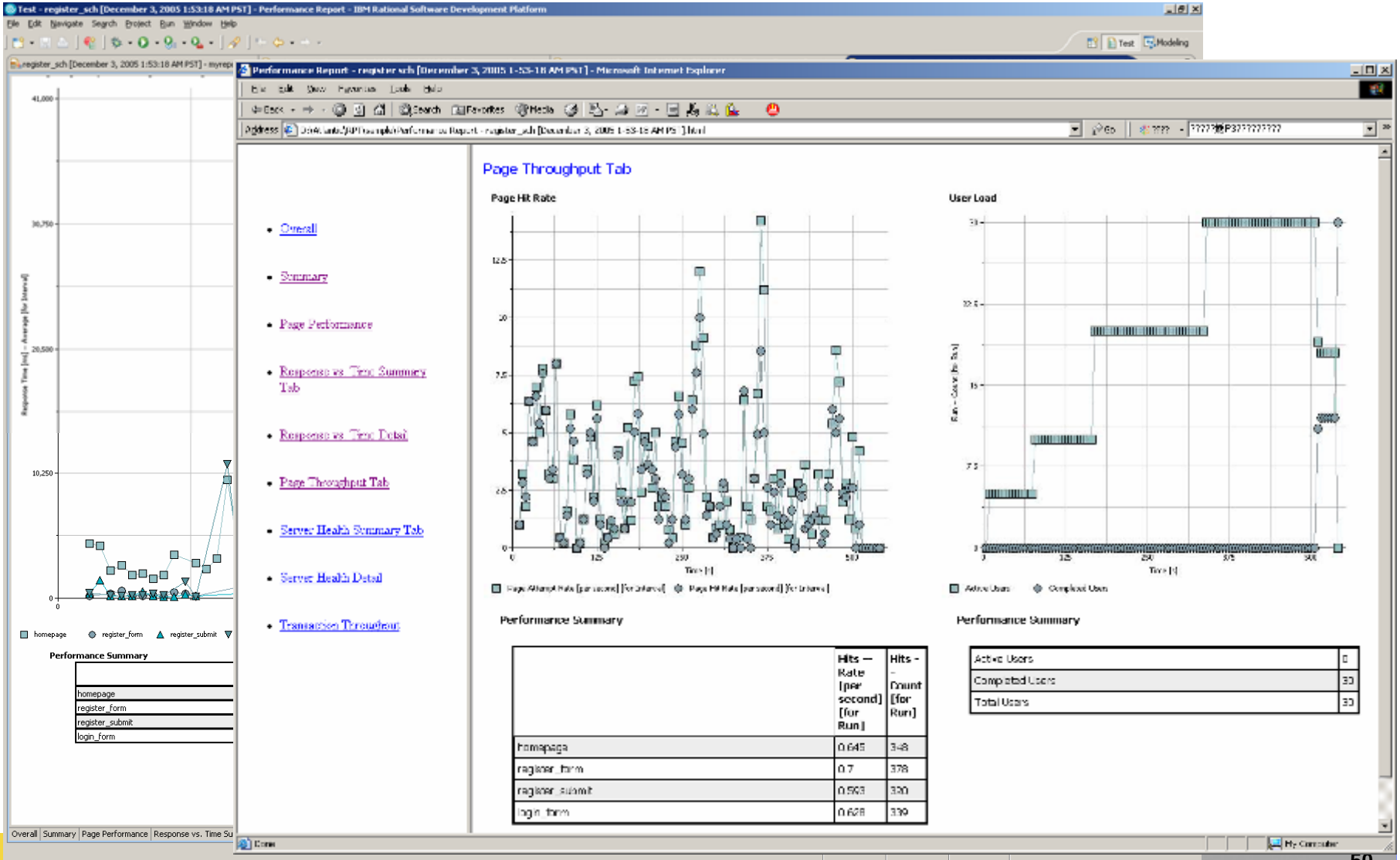


- Content VP：验证返回的内容，支持模式匹配
- Response Code VP：验证HTTP返回码
- Response Size VP：验证返回内容的大小

定义负载模型:Schedule

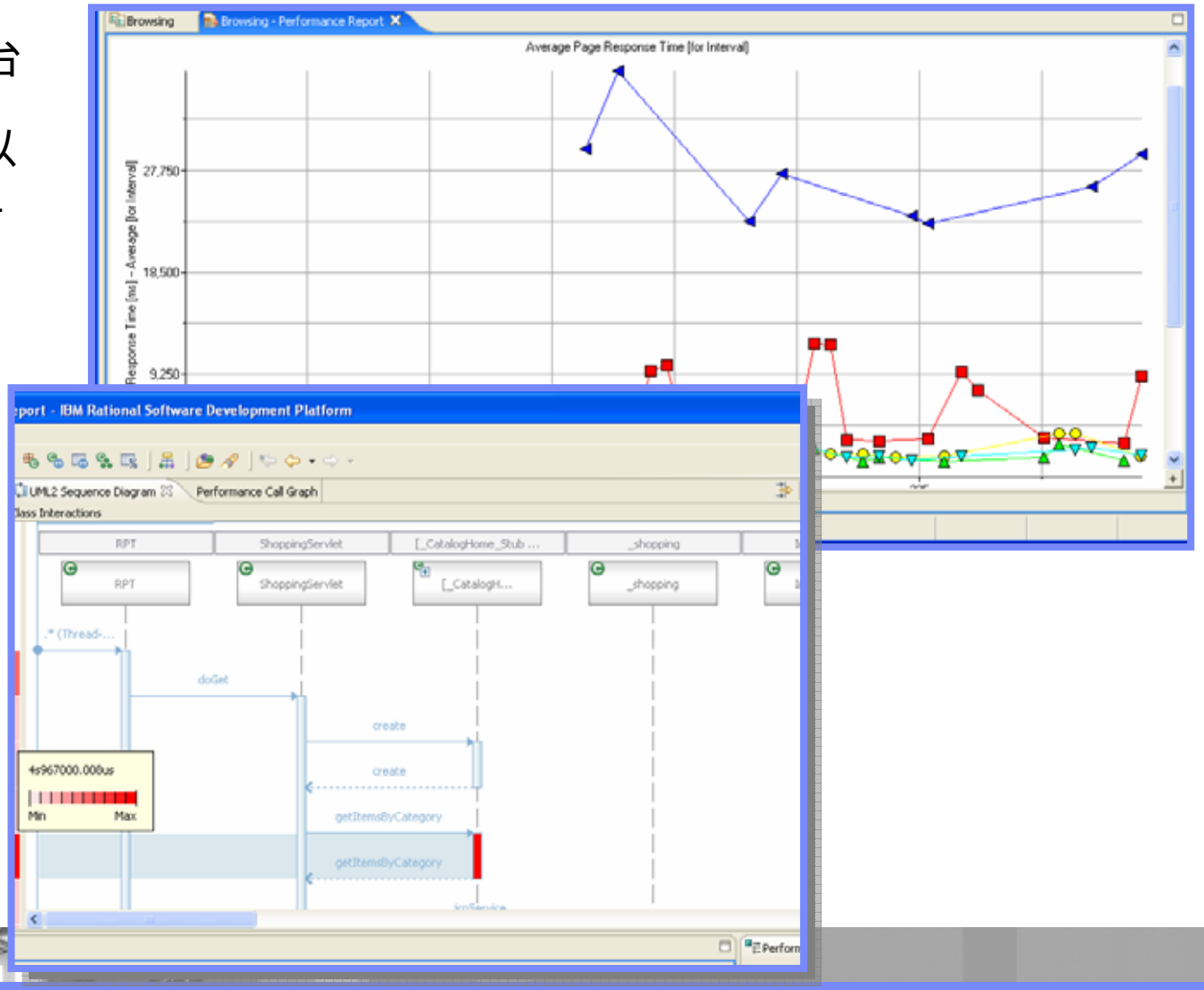


执行中和执行后的性能测报告



IBM Performance Optimization Toolkit：帮助发现应用的性能瓶颈

- 支持多种操作系统平台
- 支持WebSphere 5.x以上和WebLogic 7.x以上



IBM Rational软件测试工具

功能

- 运行时分析（内存、性能、代码覆盖）
- 代码自动评审、组件测试、系统功能测试和性能测试
- 覆盖系统测试周期的测试管理和分析

收益

- 确保软件的功能、性能和可靠性
- 加快测试周期
- 适合不同技术水平的测试人员
- 全员质量观

产品	业务人员	测试人员	开发人员
IBM Rational Manual Tester	✓	✓	✓
IBM Rational Functional Tester		✓	✓
IBM Rational Performance Tester		✓	✓
IBM Rational Robot		✓	
IBM Rational PurifyPlus		✓	✓
IBM Rational Test RealTime			✓
IBM Rational Application Developer		✓	✓


Questions


Thank
You