



IBM Software Group

软件质量保证

IBM 软件部 傅纯一

A horizontal decorative bar containing a series of small, square icons. From left to right, the icons include: a green square, a yellow square, a red square, a purple square, a cyan square, a grayscale image of a person's face, a circular arrow icon, a grayscale image of a person's hands, and a grayscale image of a person's profile.

@.business on demand.

© 2004 IBM Corporation

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

指出数组越界读错误。

指出错误发生的位置。

指出错误引入的位置。

内存访问错误内存位置。

指出内存分配的代码。

是否支持Tuxedo/CICS中间件服务？

是否支持Oracle Pro*C程序？

充分利用IBM Rational技术支持网站：
<http://www.ibm.com/software/rational/support>

```

Purify: a.out
File View Actions Options Help
Finished a.out ( 1 error, 12 leaked bytes)
Purify instrumented a.out (pid 8701 at Fri Aug 8 16:22:57 2003)
ABR: Array bounds read
This is occurring while in:
_doprint [libc.so.1.9]
printf [libc.so.1.9]
main [hello_world.c:22]
#include <st
#include <ma
static char *helloWorld = "Hello, World";
main()
{
char *mystr = malloc(strlen(helloWorld));
strcpy(mystr, helloWorld, 12);
printf("%s\n", mystr);
}
Address 0x4423c is 1 byte past end of a malloc'd block at 0x44230 of 12 bytes
This block was allocated from:
malloc [rtlib.o]
main [hello_world.c:19]
start [crt0.o]
Current file descriptors in use: 5
Memory leaked: 12 bytes (100%); potentially leaked: 0 bytes (0%)
Program exited with status code 1.

```


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

代码运行次数

源代码

Line	D	I	T	Hits	Annotated Source
16					void display_message();
17					
18					main(argc, argv)
19					int argc;
20					char** argv;
21					{
22				1	if (argc == 1)
23				1	display_hello_world();
24					else
25				0	display_message(argv[1]);
26				1	exit(0);
27				1	}
28					
29					void
30					display_hello_world()
31					{
32				1	printf("Hello, World\n");
33				1	}
34					
35					void
36					display_message(s)
37					char *s;
38					{
39				0	printf("%s, World\n", s);
40				0	}

代码行

未被测试代码

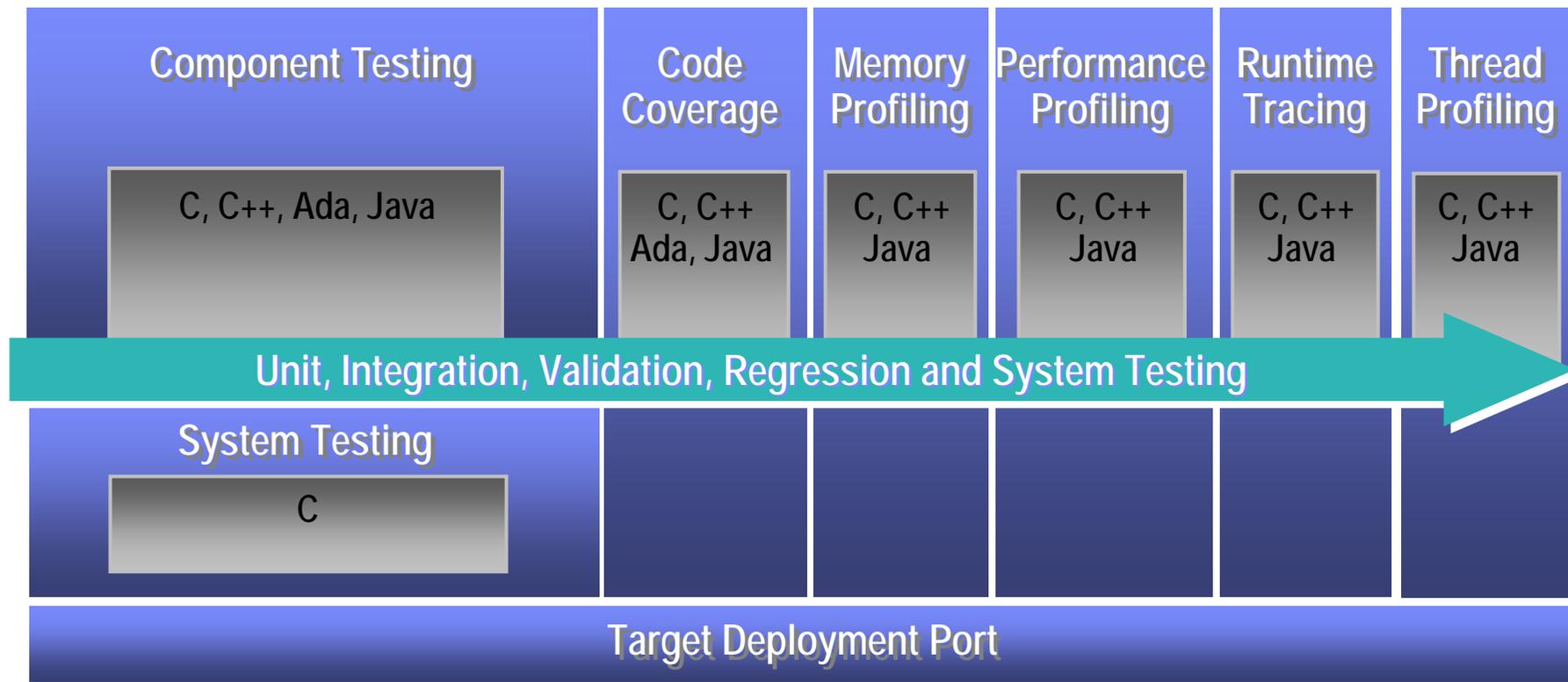
未被测试代码

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

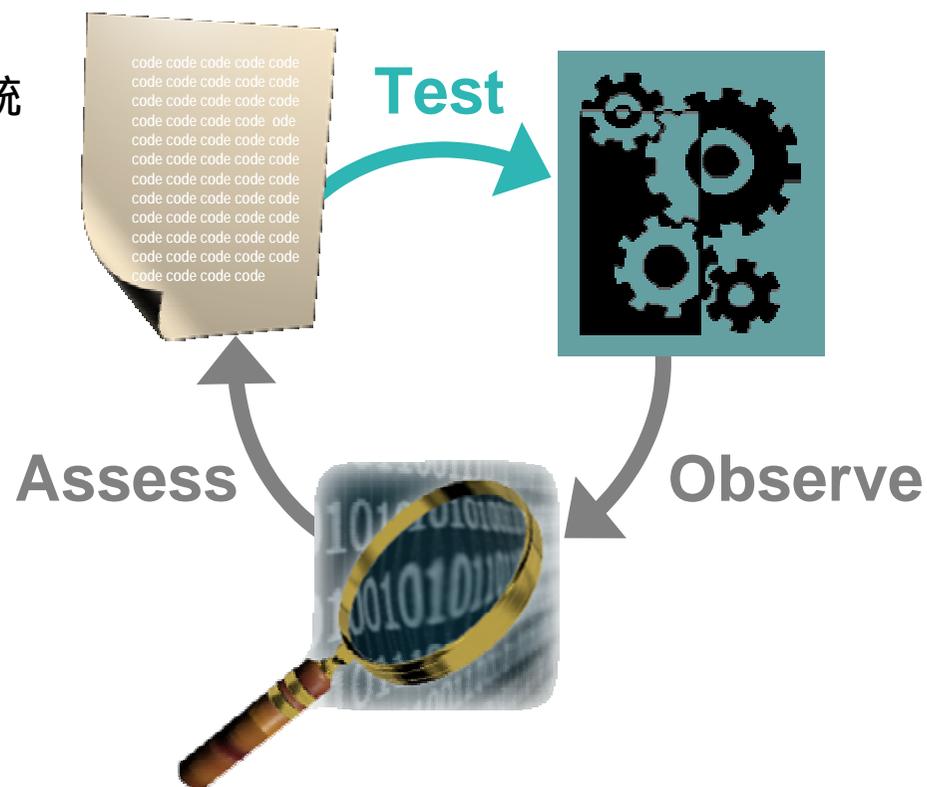
4-Bit to 64-Bit Cross-Development Environments			Languages
▪ <i>WindRiver</i>	▪ <i>Montavista</i>	▪ <i>Tasking</i>	▪ C
▪ <i>GreenHills</i>	▪ <i>TI</i>	▪ <i>CAD-UL</i>	▪ C++
▪ <i>ARM</i>	▪ <i>NEC</i>	▪ <i>Cosmic</i>	▪ Ada
▪ <i>Enea</i>	▪ <i>Hitachi</i>	▪ <i>Hiware</i>	▪ J2ME/J2SE
▪ <i>Windows CE</i>	▪ <i>Apex</i>	▪ <i>Hitex</i>	
▪ <i>LynuxWorks</i>	▪ <i>Sun</i>	▪ <i>Symbian</i>	
▪ <i>Lauterbach</i>	▪ <i>Microtec</i>	▪ <i>.....</i>	
			Platforms
			▪ <i>Windows</i>
			▪ <i>Solaris</i>
			▪ <i>RedHat</i>
			▪ <i>HP-UX</i>
			▪ <i>AIX</i>



IBM Rational Test RealTime

自动化软件测试过程

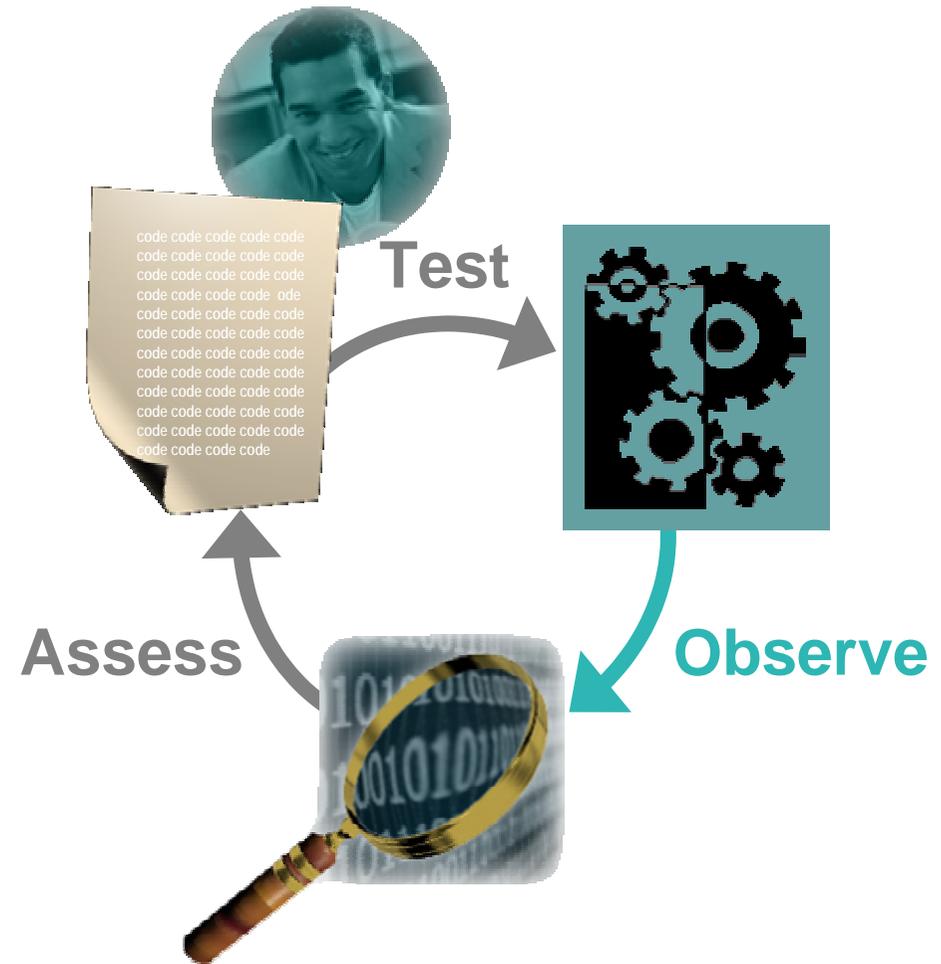
- 全面的自动化测试
 - 自动生成测试脚本模版和测试数据
 - 黑盒测试和白盒测试相集成
 - 多层次测试：从函数到分布式系统
 - 静态分析功能：
 - 明确测试优先级
 - 代码复杂度计算
 - 全面的回归测试



IBM Rational Test RealTime

自动化软件测试过程

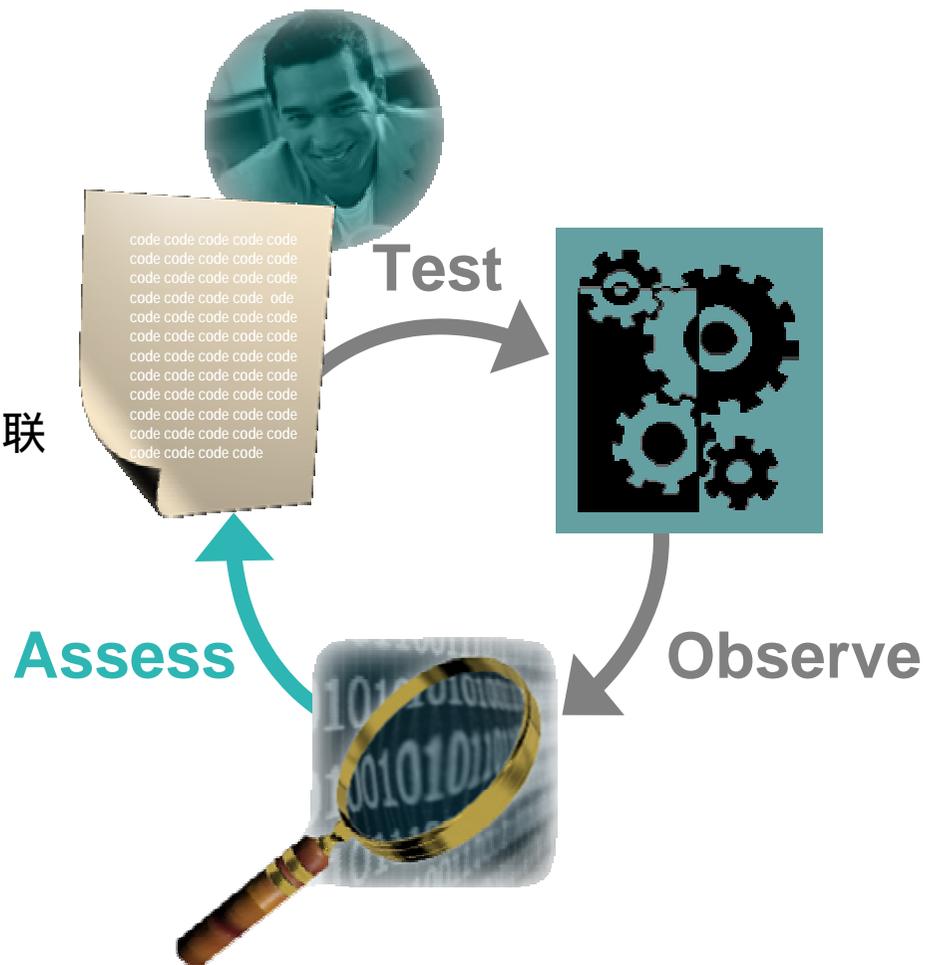
- 全面的自动化测试
- 全面的运行时分析
 - ▶ 代码覆盖率分析
 - ▶ 内存分析
 - ▶ 性能分析
 - ▶ 运行时分析
 - ▶ 线程分析



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

The screenshot displays the IBM Rational Test RealTime interface. The main window shows a test script for 'simple.ptu' with the following content:

```
HEADER Test_runtime_simple, ,
-----
-- variable declaration for simple.ptu
-----
#int coverage ;
#int simple_aa,simple_ab,simple_ac,simple_ad,simple_af,simple_ag;
#int s_link1,s_link2 ;

BEGIN

SERVICE simple_s1

TEST t1
FAMILY nominal

ELEMENT

VAR simple_aa, INIT = -5, EV = -5
VAR simple_ab, INIT = 5, EV = 6, delta=40*
VAR simple_ac, INIT = 5, MIN= 5 , MAX=5
VAR simple_ad, INIT = 5, EV==

VAR s_link1, INIT IN {1,2,3},EV==
VAR simple_af, INIT = 5, EV(s_link1) IN {5,5,5}
VAR simple_ag, INIT = 5, EV(s_link1) IN {5,5,5}

END ELEMENT
END TEST
END SERVICE

-----
-- init IN {...}
SERVICE simple_s2

TEST t1
FAMILY nominal
ELEMENT
VAR simple_aa, INIT IN {5,5,5}, EV = 5
END ELEMENT
END TEST

TEST t2
```

The right-hand pane shows a 'Settings...' dialog with a tree view of the project structure:

- Testing_C
 - double
 - Results
 - double.ptu
 - source.c
 - float
 - Results
 - float.ptu
 - source.c
 - general
 - Results
 - general.ptu
 - source.c
 - simple
 - Results
 - simple.ptu
 - source.c
 - str_nat
 - Results
 - str_nat.ptu
 - source.c
 - string
 - Results
 - string.ptu
 - source.c
 - struct
 - Results
 - struct.ptu
 - source.c
 - table

The status bar at the bottom indicates 'Ready' and 'Line: 31 Col: 1'.

IBM Rational Test RealTime: 静态分析 - 1

Component Testing Wizard

Components Under Test - Step 2 / 5

Select the files or units that you want to test.
Use the combo box to choose the selection mode.
Click on the next button to specify details of test case generation for the selected items.
You can also make your selection using the complexity metrics diagram accessible via the button.

Unit Selection

Functions	File Names	V(g)
<input type="checkbox"/> ◆ splitLine(char ** Values, char *line10)	rtrtDataInLine.c	7
<input type="checkbox"/> ◆ readVar(char * varName)	rtrtDataInLine.c	7
<input type="checkbox"/> ◆ dataInLineSetUp(char *filename, char sep, int flag)	rtrtDataInLine.c	6
<input type="checkbox"/> ◆ sinsin(float a, float b)	filter3D.c	5
<input type="checkbox"/> ◆ coma2point(char *s)	rtrtDataInLine.c	3
<input type="checkbox"/> ◆ add(int x, int y)	filter3D.c	1

Cancel < Back

计算出被测试函数/方法的代码负载度

Testability Metrics Graph [Unit Selection]

Select the units that you want to test.
You can configure the graph by clicking on the "Options..." button.

Graphical representation of units depending on their testability

Tested Units	Filename
<input checked="" type="checkbox"/> ◆ coma2point(char *s)	rtrtDataInLine.c
<input checked="" type="checkbox"/> ◆ splitLine(char ** Valu...	rtrtDataInLine.c
<input checked="" type="checkbox"/> ◆ sinsin(float a, float b)	filter3D.c
<input checked="" type="checkbox"/> ◆ readVar(char * varN...	rtrtDataInLine.c
<input checked="" type="checkbox"/> ◆ dataInLineSetUp(ch...	rtrtDataInLine.c
<input checked="" type="checkbox"/> ◆ add(int x, int y)	filter3D.c

OK Cancel Options... Help

利用代码负载度来确定测试优先级



IBM Rational Test RealTime: 静态分析 - 2

The screenshot displays the 'Metrics Viewer' window for a project named 'BaseStation_C'. It features a 'File View' on the left showing a project tree with files like 'baseStation.cpp', 'UMTSMMSG.C', and 'UmtsConnection.cpp'. The main area shows a 'Halstead Metric - Vocabulary' bar chart and a corresponding table. The table lists various code elements and their metrics.

Name	V(g)	Statements	Nested Level	Ext Comp Call	Ext Var Use
UmtsServer::~UmtsServer	1	1	1	1	0
UmtsServer::checkHardware	1	2	1	0	0
List::isLast	1	1	1	0	0
LostConnection::LostConnection	1	1	1	0	0
PhoneNumber::operator=	1	4	1	2	0
NetworkNode::NetworkNode	1	2	1	0	0
reset	1	1	1	N/A	N/A
PhoneNumber::PhoneNumber	1	0	0	0	0
List::isFirst	1	1	1	0	0
tcpsck_get_last_error	1	1	1	N/A	N/A
tcpsck_end	1	2	1	N/A	N/A
tcpsck_recv	1	2	1	N/A	N/A
UmtsServer::	1	1	1	1	0

Below the table, a status bar shows: 'at 2s 14ms 350us: void UmtsConnection::processMessages ()'.

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

The screenshot displays the IBM Rational Test RealTime interface for a project named 'BaseStation_C'. The main window shows the 'Code Coverage [BaseStation]' view for the function 'tcpsock_set_addr'. The code is color-coded to indicate coverage: green for covered lines and red for uncovered lines. The function signature is 'tcpsock_return_t tcpsock_set_addr (tcpsock_sock_addr_t *addr, char *hostname, int portnum)'. The code includes a struct definition for 'hostent', an initialization check, and logic for setting up a socket and host information. Two test cases, 'Test #1' and 'Test #2', are shown to have executed the code.

```
tcpsock_set_addr
tcpsock_return_t tcpsock_set_addr ( tcpsock_sock_addr_t *addr,
char *hostname,
int portnum )
{
    struct hostent *phe;
    struct sockaddr_in *psin;

    if (!tcpsock_is_init)
    {
        return (TCPSTACK_ERROR);
    }

    psin = &(sin_tab[sin_idx]);
    sin_idx++;
    *addr = (tcpsock_sock_addr_t)psin;

    memset ((void *) psin, 0, sizeof (struct sockaddr_in));
    psin->sin_family = AF_INET;
    if (hostname == (char *)0)
    {
        psin->sin_addr.s_addr = INADDR_ANY;
    }
    else
    {
        phe = gethostbyname (hostname);
        if (phe == (struct hostent *)0)
    }
}
```

The interface also shows a 'Settings...' window on the right with a project tree for 'BaseStation_C' and a console window at the bottom with the following output:

```
Executing \cvisual6\BBaseStation.exe ...
\cvisual6\BBaseStation.exe
Split unneeded
BaseStation:A connection was forcibly closed by a peer
```

The status bar at the bottom indicates 'Ready' and 'Line: 130'.

IBM Rational Test RealTime: 内存分析

The screenshot displays the 'Memory Profile [BaseStation]' window. On the left, a tree view shows test results for 'BaseStation' with warnings for '164 Leaked bytes in 5 blocks' and 'MLK 60 bytes (x2)'. The main area features two bar charts: 'Blocks Summary' and 'Bytes Summary'. The 'Blocks Summary' chart shows 12 allocated blocks, 10 unfreed blocks, and a maximum of 12 blocks. The 'Bytes Summary' chart shows 279 allocated bytes, 211 unfreed bytes, and a maximum of 279 bytes. Below the charts, a list of statistics includes: 'A Total of 12 blocks were allocated', '10 blocks were not freed', 'A maximum of 12 blocks were allocated at the same time', 'A Total of 279 bytes were allocated', '211 bytes were not freed', and 'A maximum of 279 bytes were allocated at the same time'. A specific error is highlighted: 'ABWL (Late Detect Array Bounds Write) A write operation 1 byte(s) past the end of the memory block at 0x431660'. On the right, a 'Settings...' panel shows a project tree for 'BaseStation_C' with various source files like 'UmtsServer.cpp' and 'baseStation.cpp'. At the bottom, a table shows project properties:

Name	Value
Name	BaseStation
Exclude from Build	No
Execute in background	Yes

IBM Rational Test RealTime: 性能分析

Top 3 Functions

45.45 %
27.27 %
27.21 %
0.08 %
Others (< 5%)

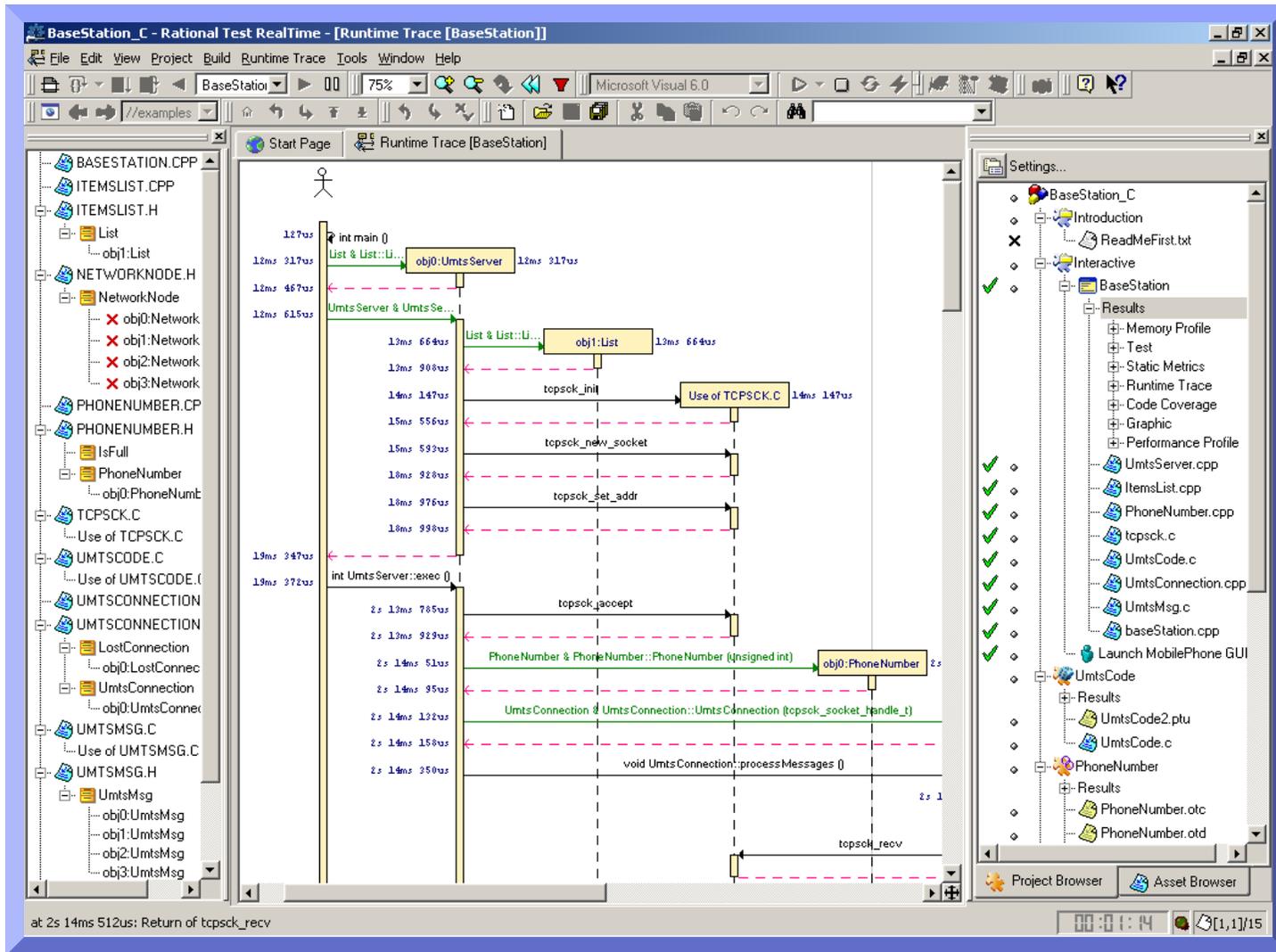
1.1 -Summary
All Times are expressed in us

Name	Calls	Function time	F+D time	F time (% of .root.)	F+D time (% of .root.)	Avg F time
tcpsck_data_ready	10	10014261	10014261	45.45	45.45	1001426
void UmtsServer::checkUmtsNetworkConnection ()	6	6007983	6007983	27.27	27.27	1001330
void UmtsServer::checkPowerSupply ()	6	5994633	5994633	27.21	27.21	999105
tcpsck_new_socket	1	6949	6949	0.03	0.03	6949
tcpsck_init	1	1394	1394	0.01	0.01	1394
tcpsck_send	2	1378	1378	0.01	0.01	689
lint main ()	1	92222034733	<0.01	100.00	922	

Settings...

- BaseStation_C
 - Introduction
 - ReadMeFirst.txt
 - Interactive
 - BaseStation
 - Results
 - UmtsServer.cpp
 - ItemsList.cpp
 - PhoneNumber.cpp
 - tcpsck.c
 - UmtsCode.c
 - UmtsConnection.cpp
 - UmtsMsg.c
 - baseStation.cpp
 - Launch MobilePhone GUI
 - UmtsCode
 - Results
 - UmtsCode2.ptu
 - UmtsCode.c
 - PhoneNumber
 - Results
 - PhoneNumber.otc

IBM Rational Test RealTime: 运行追踪图



IBM Rational Test RealTime: 测试报告

UmtsCode.wrd

- UmtsCode2
 - code_int
 - Test 1
 - Element 1
 - Test Coverage
 - Test 2
 - Element 1
 - Test Coverage
 - Test 3
 - Element 1
 - Test Coverage
 - Service Coverage
 - decode_int
 - Test 1
 - Element 1
 - Test Coverage
 - Test 2_1 (1/3)
 - Element 1
 - Test Coverage
 - Test 2_2 (2/3)
 - Element 1
 - Test Coverage
 - Test 2_3 (3/3)
 - Element 1
 - Test Coverage

1.2.3.2 - Element 1

1.2.3.2.1 - Variables

| Variable | Status | Init Value | Expected Value | Obtained Value |
|----------|--------|------------|----------------|----------------|
| x | Passed | 34 | 34 | 34 |
| buffer | Passed | "" | "1243" | "1243" |

1.2.3.3 - Test Coverage

File UMTSCODE.C

| code_int | Functions and exits | Statement blocks | Implicit blocks | Decisions | Loops |
|----------|-------------------------|------------------------|-----------------|------------------------|-------------------------|
| | 100.0% (2/2), +0.0 (+0) | 66.7% (2/3), +0.0 (+0) | none | 66.7% (2/3), +0.0 (+0) | 33.3% (2/6), +16.7 (+1) |

1.2.4 - Test 3

1.2.4.1 - Information

| | | | |
|------------------|--------|----------------|---------------|
| Test Name | 3 | Test Family | nominal |
| Status | Failed | Execution Time | 29 micro sec. |
| Failed Variables | 1 | | |

1.2.4.2 - Element 1

1.2.4.2.1 - Variables

| Variable | Status | Init Value | Expected Value | Obtained Value |
|----------|--------|------------|----------------|----------------|
| x | Passed | 0 | 0 | 0 |
| buffer | Failed | "" | "110" | "10" |

1.2.4.3 - Test Coverage

File UMTSCODE.C

| code_int | Functions and exits |
|----------|-------------------------|
| | 100.0% (2/2), +0.0 (+0) |

Console Output:

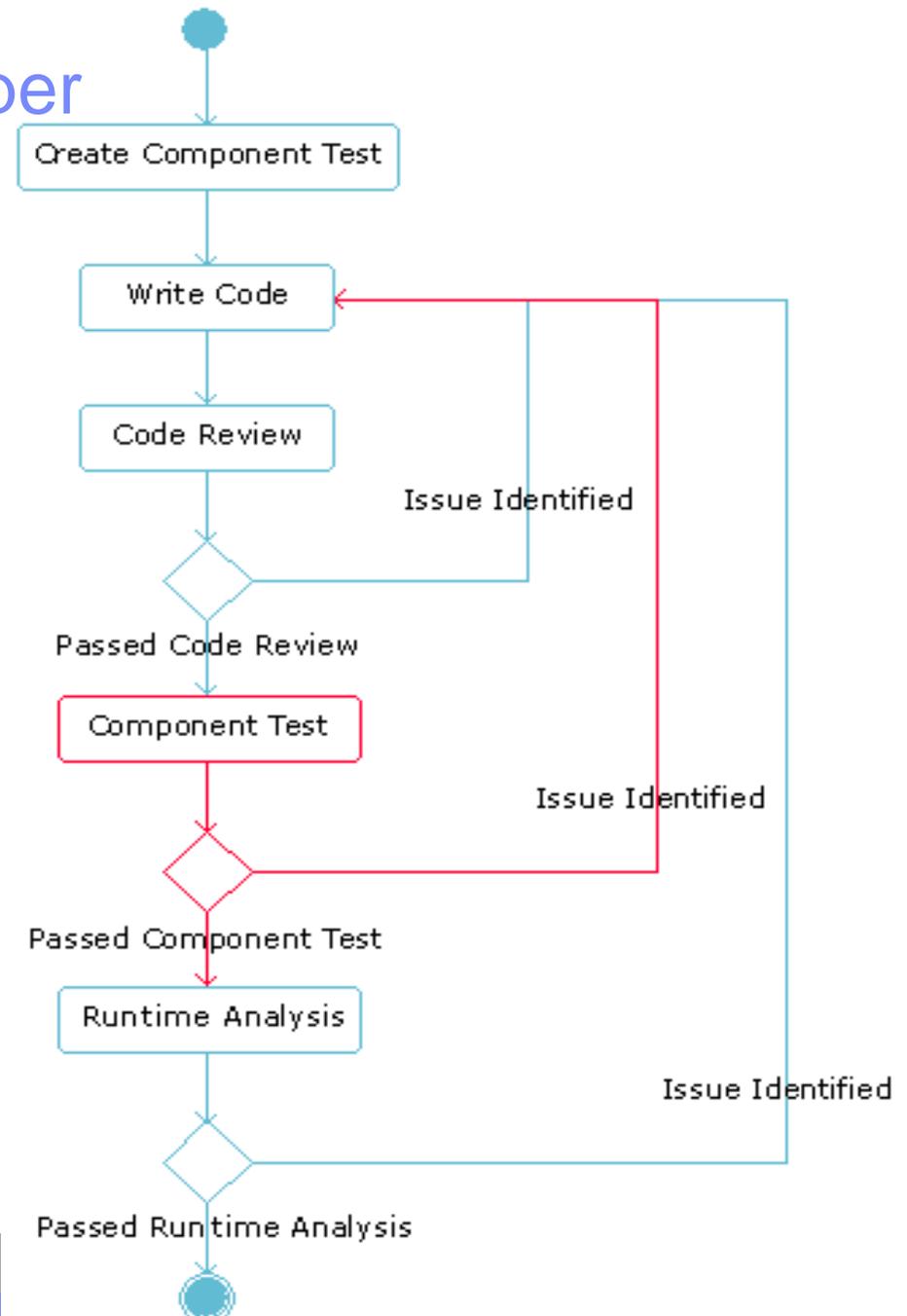
```
-cio="cvisual6\atu.cio" -VA=EVAL
TestRT+-STARTEXEC, Rational(R) Test RealTime C and Ada Test Report Generator 2003.06.00.000.004
TestRT+-COPYRIGHT, Copyright(C) 1992-2002 Rational Software Corporation. All rights reserved.
C:\Rational\TESTRE~3\bin\intel\win32\rod2xd -g "I:\Rational\TestRealTime\examples\BaseStation_C\intermediates_files79024449.log" "-ocvisual6\UmtsCode.wrd" "cvisual6\TUmtsCode.rod"
TestRT+-TEST_ERRD, Unit Test Report Generator execution completed with incorrect tests
TestRT+-ENDNOEWAR, End of execution with 1 warning(s)
(rod2xd) Generation of graphic results "cvisual6\TUmtsCode_1.rtx" for decode_int/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

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

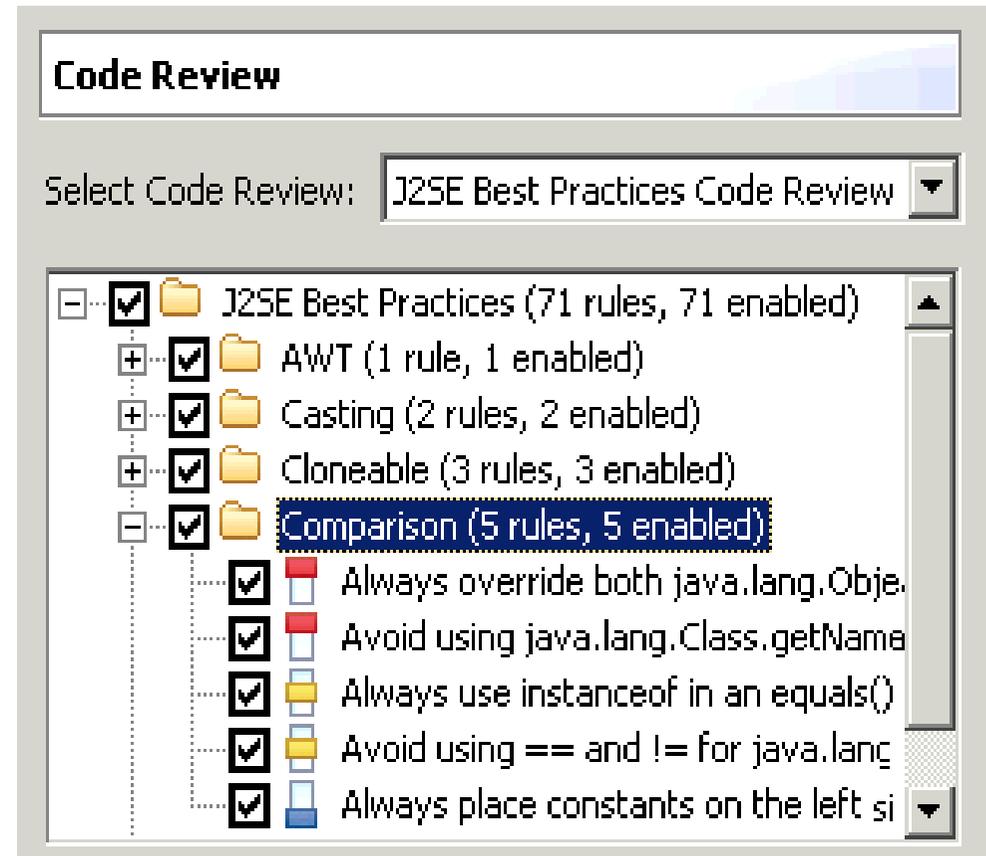
- 利用UML图对代码中的类和依赖以及方法的内部行为进行可视化

The screenshot displays the IBM Rational Software Development Platform interface. The top window shows the source code for the `cashCheck` method in `PiggyBankControllerBean.java`. The code includes validation logic for a check reference and amount, and a credit account operation. The bottom window shows a Static Method Sequence Diagram (SM-SD) for the `cashCheck` method. The diagram illustrates the internal behavior of the method, including the creation of a `CityBankDataAccessObject` instance, a call to its `checkCityBankAccount` method, and the subsequent credit account operation. The diagram is annotated with a `distruptable` block and a `[try]` block. The IDE interface also shows a package explorer on the left and a status bar at the bottom.

Static Method Sequence Diagram : 理解特定方法的内部行为

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

Create Java Component Test

Select the components under test
Use the calculated metrics to help you choose the components to test. Numbers that are above average for the column are highlighted.

Components:

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

Test name and location

Use defaults

Name: CityBankServiceLocatorTest

Package: test

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

The screenshot displays the Rational Application Developer IDE. The top window shows the source code for `ZipcodeTest.java` in the `test` package, which extends `TestCase` and contains two test methods: `test_isZipcode()` and `test_getCity()`. The bottom window, titled "Test Data Comparator", shows the execution results for the `test_isZipcode()` method. The table below summarizes the data shown in the comparator.

| Action | Type | New Test Data 2 | | |
|--|-------------------|-----------------|----------------|----------------|
| | | In | Expected | Actual |
| objZipcode = new Zipcode() | XY | | | |
| objZipcode | com.acme.commo... | | | |
| <expected exception> | Throwable | | <no exception> | <no exception> |
| retValue = objZipcode.isZipcode(zipcode) | XY | | | |
| 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：性能分析

执行统计信息 - ibm-myang 上的 Sort [PID : 8704] (过滤器：没有过滤器)

| >包 | 额定时间 (秒) | 平均额定时间 (秒) | 累积时间 (秒) | 调用次数 |
|-------------------------------|----------|------------|----------|------|
| Bsort(int[]) void | 0.000063 | 0.000063 | 1.267116 | 1 |
| bsort(int[], int, int) void | 1.267053 | 1.267053 | | 1 |
| computemed(int[]) void | 0.027202 | 0.013601 | | 2 |
| main(java.lang.String[]) void | 0.011638 | 0.011638 | | 1 |
| Qsort(int[]) void | 0.000125 | 0.000125 | 5.886789 | 1 |
| quick(int[], int, int) int[] | 5.886664 | 0.001174 | 5.886664 | 5015 |
| java.lang | 0.000000 | 0.000000 | 0.000000 | 0 |

方法的执行时间、测试

性能调用图 - Java Profiling Agent - 进程：8704

缩放： [Slider] 突出显示：到根的最大路径

调用图，显示性能瓶颈

线程：main
线程：Reference Handler
线程：Finalizer
线程：Thread-0

Sort.Qsort
Sort.computemed
Sort
RandNum.RandNum

可视：13/13 突出显示：5/5 Sort.quick

方法详细信息 - Java Profiling Agent - 进程：8704

突出显示：到根的最大路径

Sort.quick

分析特定方法的详细性能数据

| 调用者 | 调用者 | 主机 | 进程 | >百分比 (秒) | 调用 | 传播时间 |
|------------|------------|-----------|------|----------|-------|------|
| Sort.Qsort | Sort.Qsort | ibm-myang | 8704 | 100.00 | 1 | 5.89 |
| Sort.quick | Sort.quick | ibm-myang | 8704 | 99.02 | 5,014 | 5.83 |

子节点

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

The screenshot displays the Rational Application Developer interface with the following components:

- Editor (Sort.java):** Shows Java source code with coverage markers. Green checkmarks (✓) indicate covered lines, and red dashes (—) indicate uncovered lines. The code includes a swap function, a recursive call, and a quick sort demonstration algorithm.
- Package Explorer (右侧):** Shows the project structure for 'ibm-myang 上的 Sort [PID : 7768]'. It lists classes and methods with coverage icons (green checkmarks or red dashes) next to them.

```
Sort.java
/* Swap elements a[lo] and a[hi] */
✓ if (lo < hi) {
✓     int T = numbers[lo];
✓     numbers[lo] = numbers[hi];
✓     numbers[hi] = T;
}

/* Put the median in the "center" of the list */
✓ numbers[hi0] = numbers[hi];
✓ numbers[hi] = pivot;

/* Recursive calls */
✓ int outNum[] = quick(numbers, lo0, lo - 1);
✓ System.arraycopy(outNum, lo0, numbers, lo0, lo-lo0);
✓ outNum = quick(numbers, hi + 1, hi0);
✓ System.arraycopy(outNum, hi+1, numbers, hi+1, hi0-hi);
✓ return numbers;
}

/* Quick Sort demonstration algorithm */
static void quickFast(int numbers[], int lo0, int hi0) {
-     int lo = lo0;
-     int hi = hi0;

-     if (lo >= hi) {
-         return;
-     } else if (lo == hi - 1) {
-         if (numbers[lo] > numbers[hi]) {
-             int T = numbers[lo];
-             numbers[lo] = numbers[hi];
-             numbers[hi] = T;
-         }
-         return;
-     }
}
```

Package Explorer Structure:

- ibm-myang 上的 Sort [PID : 7768]
 - 缺省程序包
 - Sort
 - main(java.lang.String[]) ✓
 - Bsort(int[]) void ✓
 - bSort(int[], int, int) void ✓
 - Sort() ✓
 - Qsort(int[]) void ✓
 - quick(int[], int, int) int[] ✓
 - quickFast(int[], int, int) v ✓
 - computedmed(int[]) void ✓
 - RandNum
 - RandNum(int) ✓
 - getRandNum() int[] ✓

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

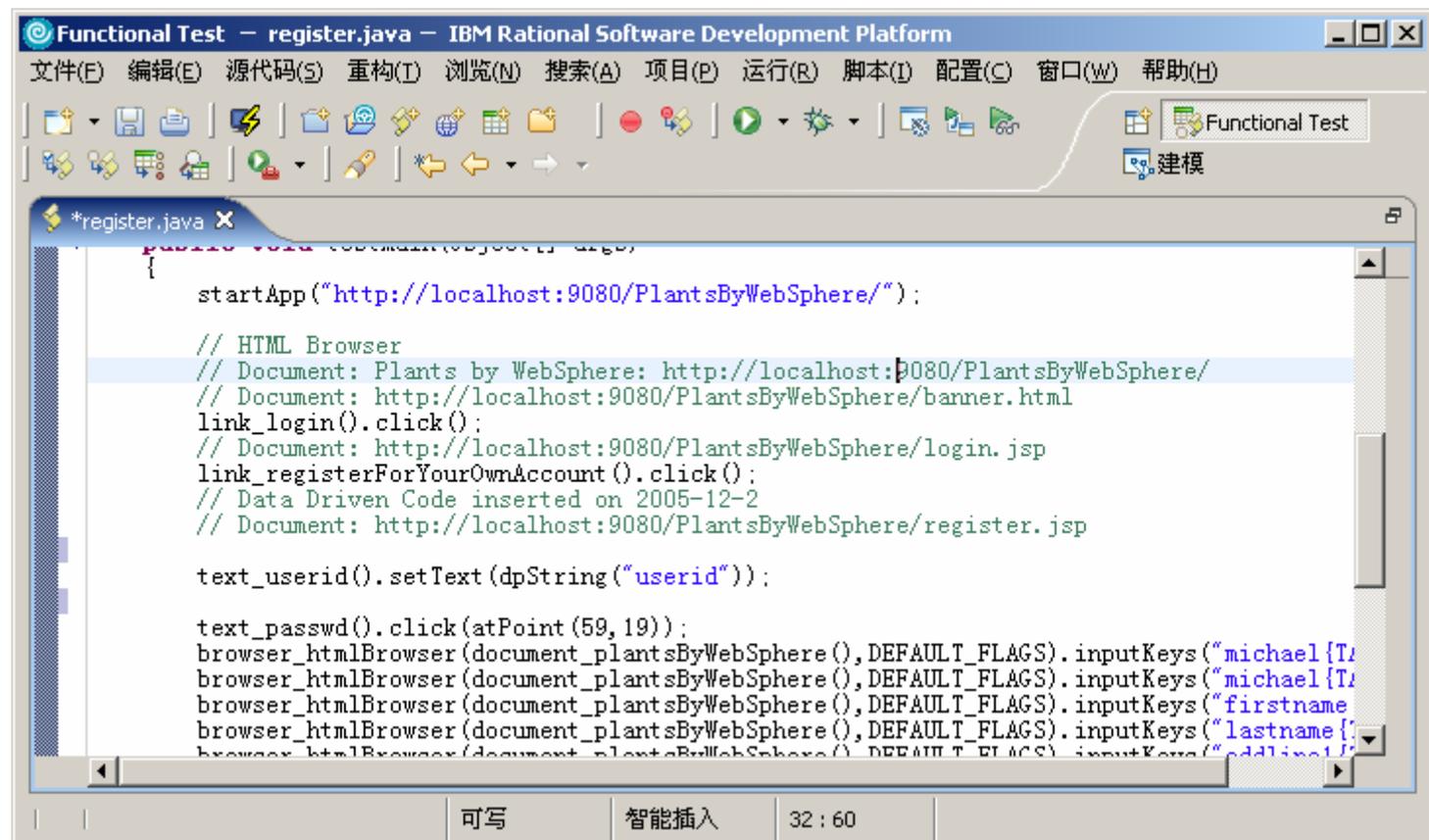
GUI功能测试自动化工具：Rational Functional Tester

- 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



The screenshot shows the IBM Rational Software Development Platform interface. The main window displays a Java test script named `register.java`. The script is written in a functional testing style, starting with `startApp("http://localhost:9080/PlantsByWebSphere/")`. It then performs several actions: navigating to `link_login().click()`, `link_registerForYourOwnAccount().click()`, and setting the `text_userid()` field to `userid`. The script also includes several `browser_htmlBrowser` calls with `inputKeys` to enter test data like "michael" and "firstname". The interface includes a menu bar with options like "文件(E)", "编辑(E)", "源代码(S)", "重构(T)", "浏览(N)", "搜索(A)", "项目(P)", "运行(R)", "脚本(I)", "配置(C)", "窗口(W)", and "帮助(H)". A toolbar with various icons is visible below the menu. The status bar at the bottom indicates "可写", "智能插入", and "32 : 60".

```
startApp("http://localhost:9080/PlantsByWebSphere/");

// HTML Browser
// Document: Plants by WebSphere: http://localhost:9080/PlantsByWebSphere/
// Document: http://localhost:9080/PlantsByWebSphere/banner.html
link_login().click();
// Document: http://localhost:9080/PlantsByWebSphere/login.jsp
link_registerForYourOwnAccount().click();
// Data Driven Code inserted on 2005-12-2
// Document: http://localhost:9080/PlantsByWebSphere/register.jsp

text_userid().setText(dpString("userid"));

text_passwd().click(atPoint(59, 19));
browser_htmlBrowser(document_plantsByWebSphere(), DEFAULT_FLAGS).inputKeys("michael{T
browser_htmlBrowser(document_plantsByWebSphere(), DEFAULT_FLAGS).inputKeys("michael{T
browser_htmlBrowser(document_plantsByWebSphere(), DEFAULT_FLAGS).inputKeys("firstname
browser_htmlBrowser(document_plantsByWebSphere(), DEFAULT_FLAGS).inputKeys("lastname{T
browser_htmlBrowser(document_plantsByWebSphere(), DEFAULT_FLAGS).inputKeys("address{T
```

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

The screenshot displays the IBM Rational Software Development Platform interface. The main window is titled "Functional Test - data_driven.java - IBM Rational Software Development Platform". The menu bar includes options like "文件(E)", "编辑(E)", "源代码(S)", "重构(I)", "浏览(N)", "搜索(A)", "项目(P)", "运行(R)", "脚本(I)", "配置(C)", "窗口(W)", and "帮助(H)". The toolbar contains various icons for file operations and testing. The code editor shows the following Java code:

```
// Data Driven Code inserted on 2005-12-2  
  
// HTML Browser  
// Document: Plants by WebSphere: http://localhost:9080/PlantsByWebSphere/  
// Document: http://localhost:9080/PlantsByWebSphere/register.jsp  
text_userid().setText(dpString("userid"));  
}  
}
```

Below the code editor, there is a "测试数据池" (Test Data Pool) section with tabs for "任务" (Tasks) and "问题" (Issues). The "测试数据池" tab is active, showing a table of test data:

| 专用测试数据池 | |
|---------|--------------------------|
| | userid::java.lang.String |
| 0 | rft001@cn.ibm.com |
| 1 | rft002@cn.ibm.com |
| 2 | rft003@cn.ibm.com |

The status bar at the bottom indicates "可写" (Writable), "智能插入" (Smart Insert), and "34 : 38".

利用Object Map降低脚本维护成本

脚本"ShoppingCart.register"的专用测试对象图

文件(F) 编辑(E) 查找(I) 测试对象(T) 首选项(P) 应用程序(A) 显示(D) 帮助(H)

Html: Frame: banner: Html.FRAME
 Html: Document: HtmlDocume...
 Html: Table: HtmlTable_1: Html.TABLE
 Html: Link: LOGIN: Html.A

| 属性 | 值 | 权重 |
|-------------|---|-----|
| .class | Html.A | 100 |
| .classIndex | 6 | 50 |
| .href | HRef(http://localhost:9080/PlantsByWebSphere/login.jsp) | 40 |
| .id | | 90 |
| .name | | 90 |
| .title | LOGIN | 50 |

采用ScriptAssure技术，实现脚本执行时候对象的模糊匹配

39

采用模式匹配快速实现结果验证

ClassicsCD.com

CD Order Placed

Your order has been placed. For future reference, your order ID is 230. Thank you for shopping at ClassicsCD.com!

Catalog Go

Shopping Cart Go

Cashier Go

Order Status Go

Order ID 是动态数据

###

###

###

###

###

###

###

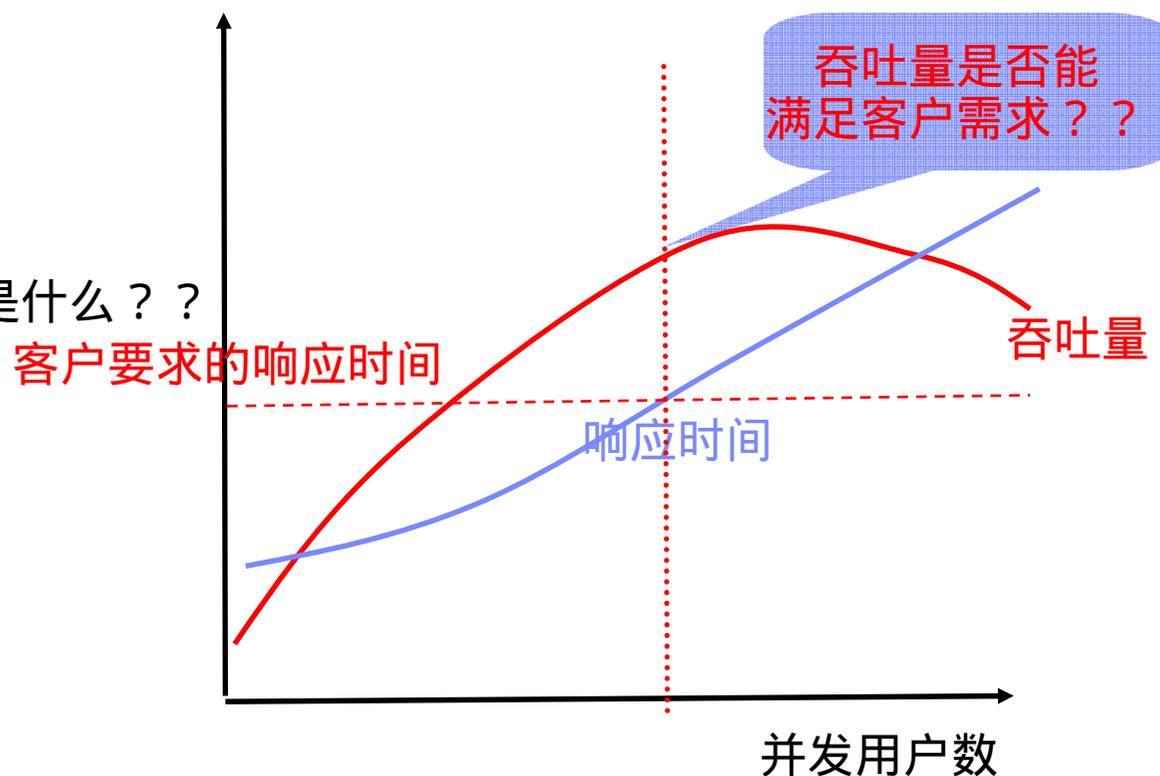
- 动态数据模式匹配
 - ▶ 如对“Order ID 230”，验证点中的模式串为“Order ID ###”或“Order ID 2##”

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

Test Element Details

localhost:9080/PlantsByWebSphere/servlet/AccountServlet?action=register

Request Attributes

Version: 1.1 Method: POST

Host: localhost Port: 9080

URL: /PlantsByWebSphere/servlet/AccountServlet?action=register

Data:

```
userid=pt0001@cn.ibm.com&passwd=password&vpasswd=password&fname=michael&lname=yang&addr1=line1&addr2=line2&city=beijing&state=beijing&zip=100027&phone=5391108&x=42&y=24
```

Request Headers

| Header Name | Value |
|-------------|---|
| Accept | image/gif, image/x-bitmap, image/jpeg, image... |
| Referer | http://localhost:9080/PlantsByWebSphere/regi... |

- Primary request for page Delay: 0 milliseconds

Character set: GB2312

Connection localhost:9080 Response 200 - OK

Enable application monitoring

HTTP请求和响应列表

Common properties

Enable application monitoring

Tasks Recorder Control Protocol Data

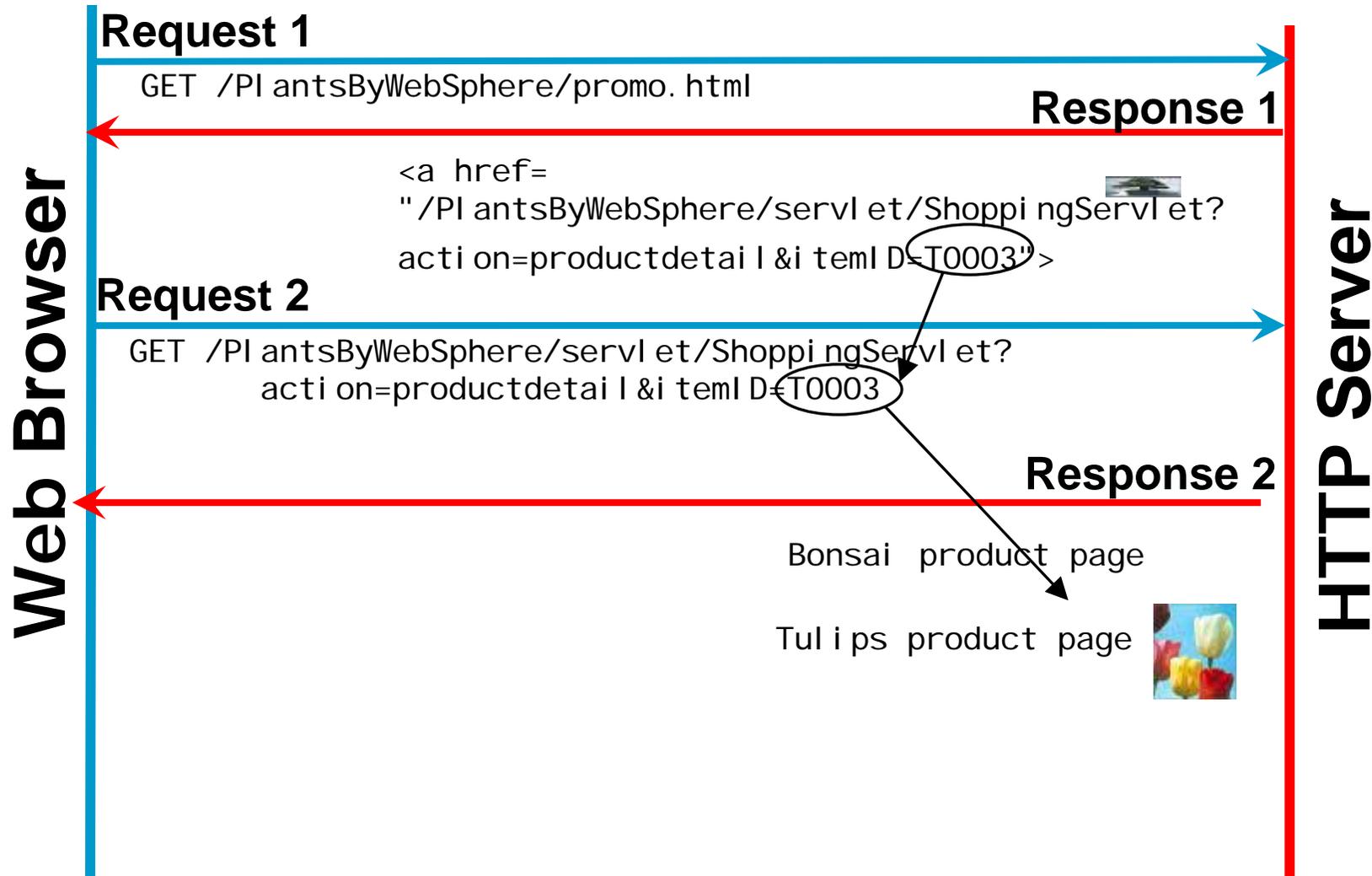
Request Response Headers Response Content Browser

× Gardens of Summer: They all start with the right colors, and we've got them all.

× Image of a colorful garden

HTTP响应数据

修改测试：数据关联



RPT: 自动建立数据关联

Test Data

| Name | Value | Substituted with |
|-------------|-------------|----------------------|
| action | getimage | "getimage" - Content |
| action | productdata | Content |
| inventoryID | T0003 | |
| itemID | T0003 | of Pl... |

- Go To
- Remove Substitution
- URL Encode
- Show Locations
- Show References

- localhost:9080/PlantsByWebSphere/images/1
- localhost:9080/PlantsByWebSphere/images/it
- localhost:9080/PlantsByWebSphere/images/b
- localhost:9080/PlantsByWebSphere/images/p
- localhost:9080/PlantsByWebSphere/images/t.
- Shopping Cart - Checkout
- Customer Login

Up
Down

URL: /PlantsByWebSphere/servlet/ImageServlet?action=getimage&inventoryID=T0003

Data:

Request Headers

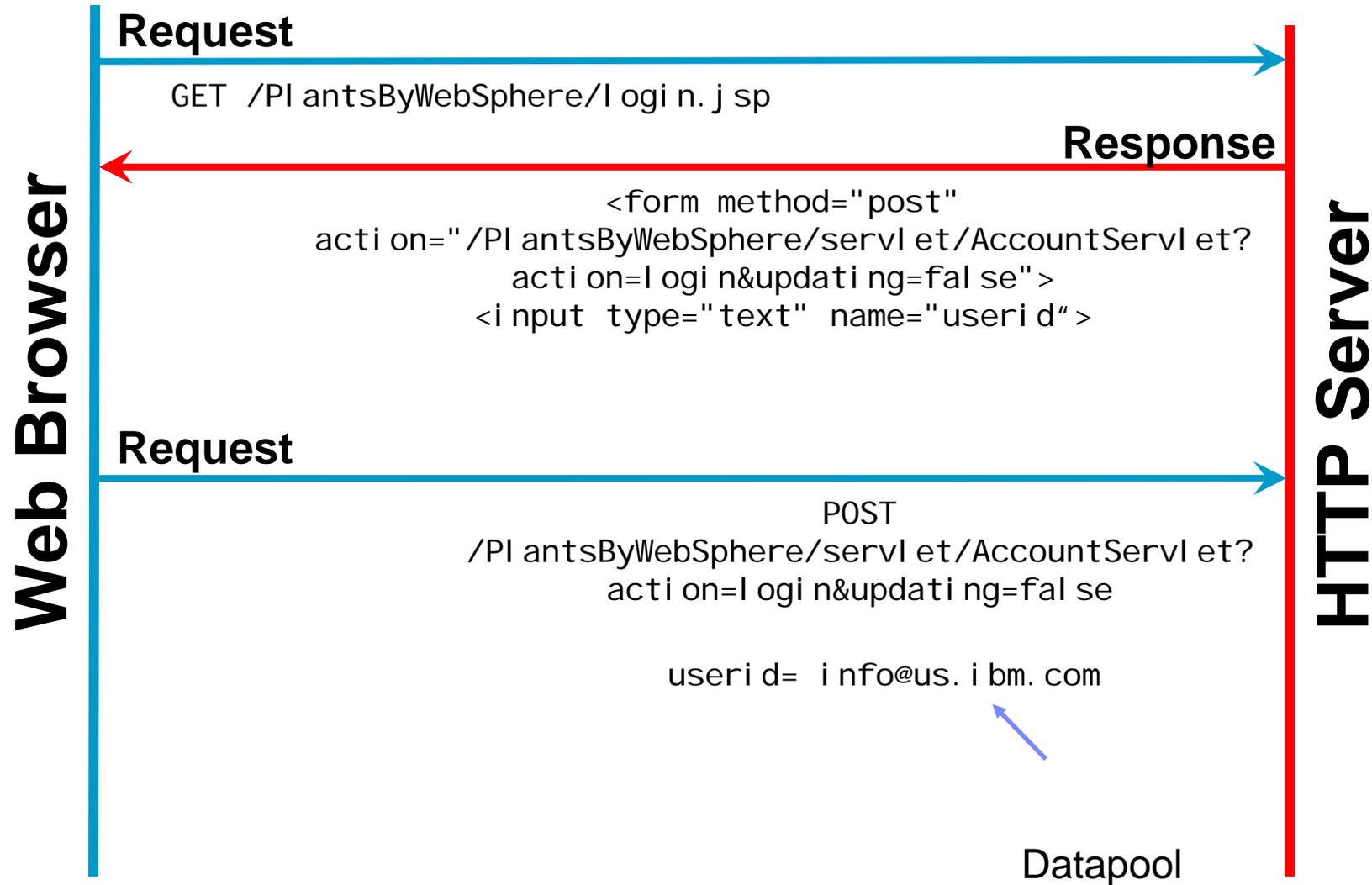
| Header Name | Value |
|-------------|-------|
| Accept | |

- Cut
- Copy
- Paste
- Delete
- Select All
- Create Field Reference
- Substitute From
- Remove Substitution
- URL Encode
- Go To

Content:

```
<td rowspan="9" valign="top">
  <br>
```

数据参数化



快速设置验证点 (Verification Point)

Performance Test - Register

Test Contents

This section shows the test contents

Test Element Details

localhost:9080/PlantsByWebSphere/servlet/AccountServlet?action=register

Request Attributes

Version: 1.1 Method: POST

Host: localhost Port: 9080

URL: localhost:9080/PlantsByWebSphere/servlet/AccountServlet?action=register

引用了变量

使用了Datapool

Data:

```
userid=rpt0001@cn.ibm.com&passwd=password&vpasswd=password&fname=
name=yang&addr1=line1&addr2=line2&city=beijing&state=beij
00027&phone=65391188&x=42&y=24
```

Datapool备选

- Centent 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

| Value |
|---|
| image/gif, image/x-bitmap, image/jpeg, image... |
| http://localhost:9080/PlantsByWebSphere/regi... |

Request for page Delay: 0 milliseconds

Charset: GB2312

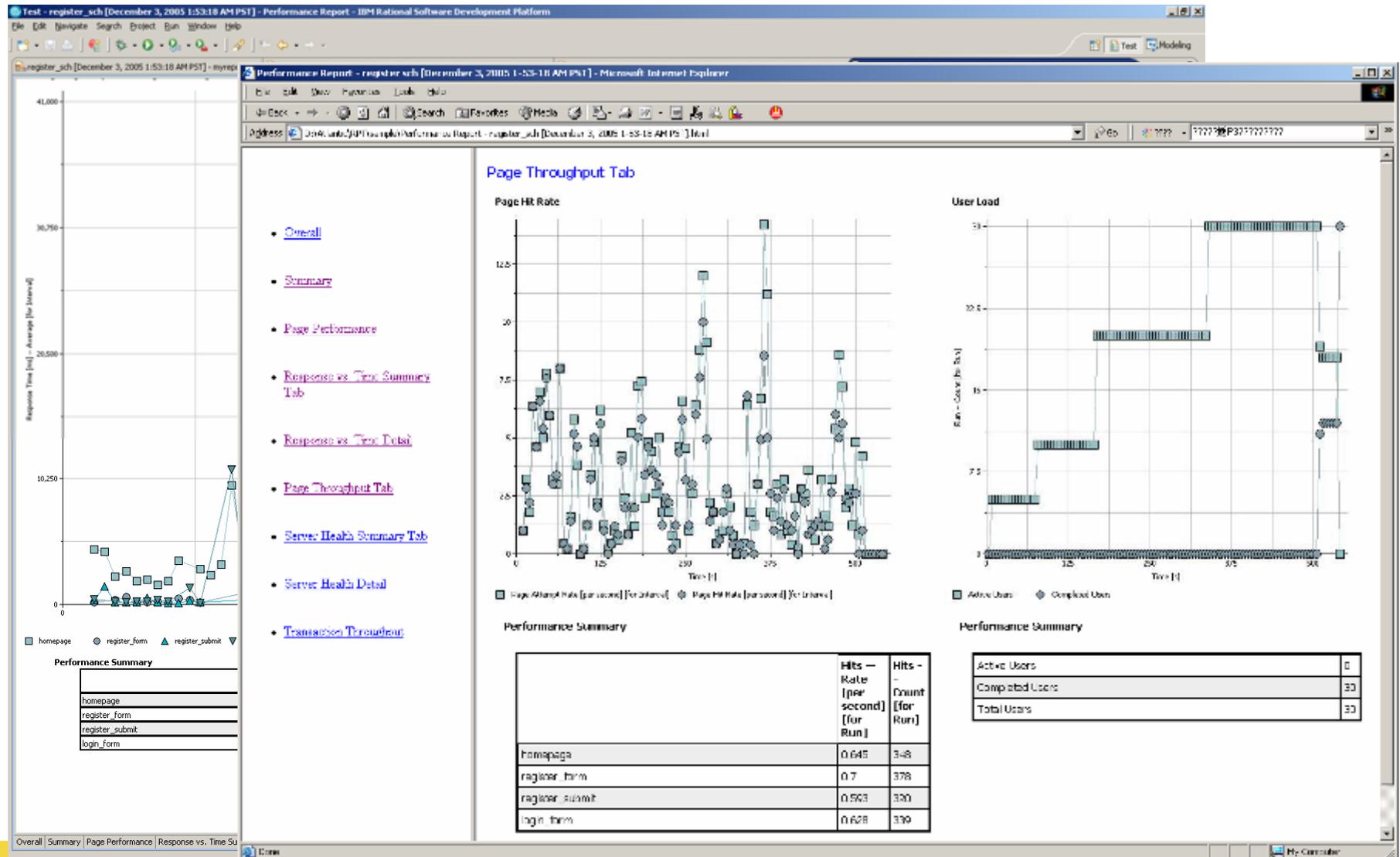
Host: localhost:9080 Response: 200 - OK

定义负载模型:Schedule

The screenshot shows the 'Performance Schedule - register_sch' configuration window. The left pane, 'Schedule Contents', displays a tree view with 'register_sch' containing 'User Group 1 (100%)', 'Loop (50 iterations)', and 'Register'. The right pane, 'Schedule Element Details', is for 'register_sch' and includes the following settings:

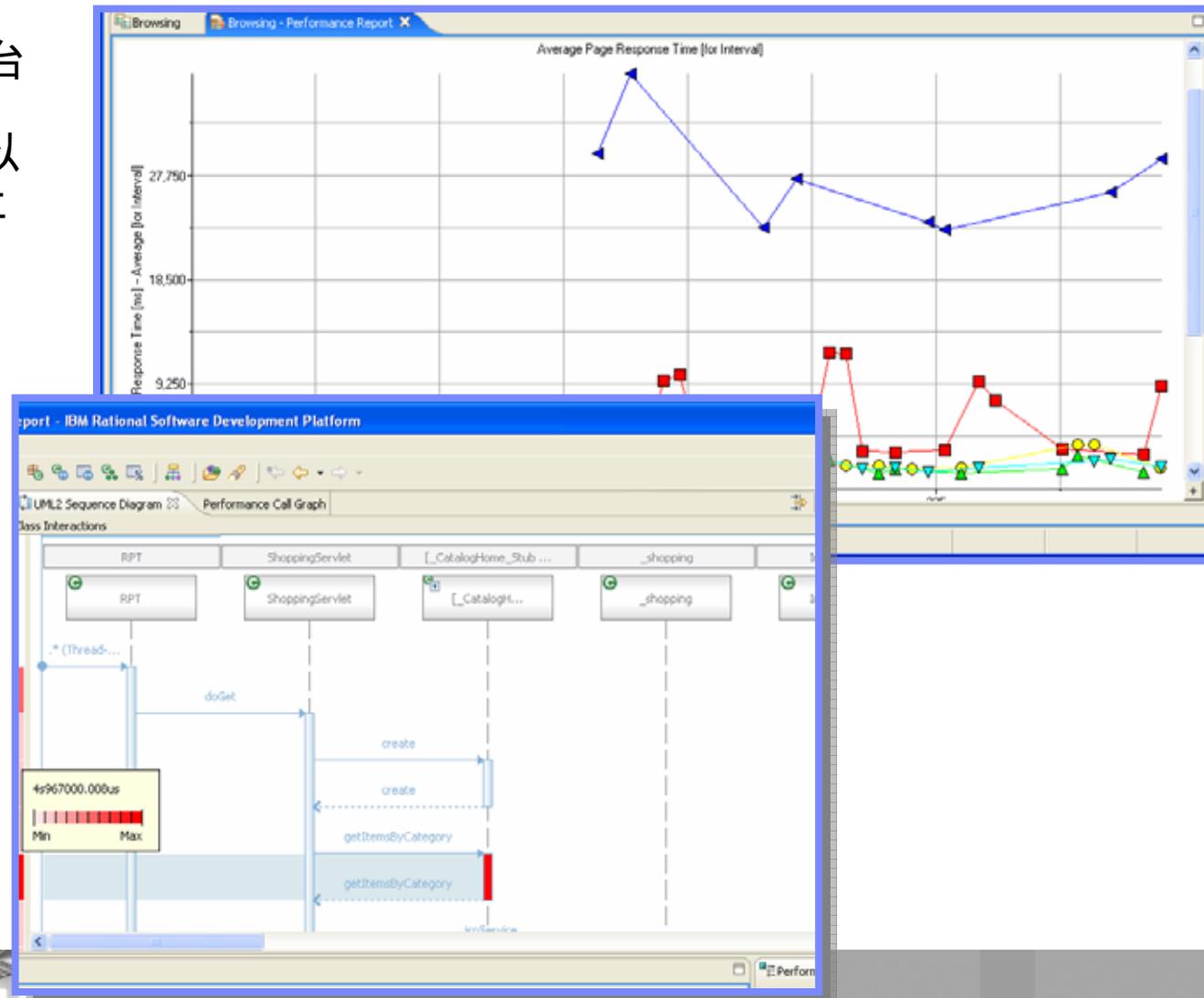
- Number of users:** 5 (highlighted with a blue callout: 并发用户数, 执行时可增加用户数)
- Add a delay between starting each user
Delay: 0 milliseconds
- Stop running the schedule after an elapsed time
Stop after: 10 minutes
- Think Time**
 - Modify the duration of think time delays: Use the recorded think time
 - Limit think times to a maximum value
Maximum think time: 0 seconds
- Execution History**
 - Execution history log level: Full (highlighted with a blue callout: 日志控制)
 - Only sample information from a subset of users

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