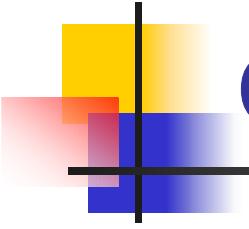


# *RESORT for Java (JSP)*

## 製品紹介

**Soft4Soft**

<http://www.soft4soft.com>



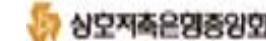
# Contents

---

- 会社概要
- RESORT for Javaの製品群
- RESORT for Javaの特徴
- RESORT for Java - Reverseの重要機能
- RESORT for Java - Qualityの特徴
- RESORT for Java - Qualityの重要機能
- RESORT for Java - Testingの特徴
- RESORT for Java - Testingの重要機能
- 品質の測定及び評価の事例
- 結論

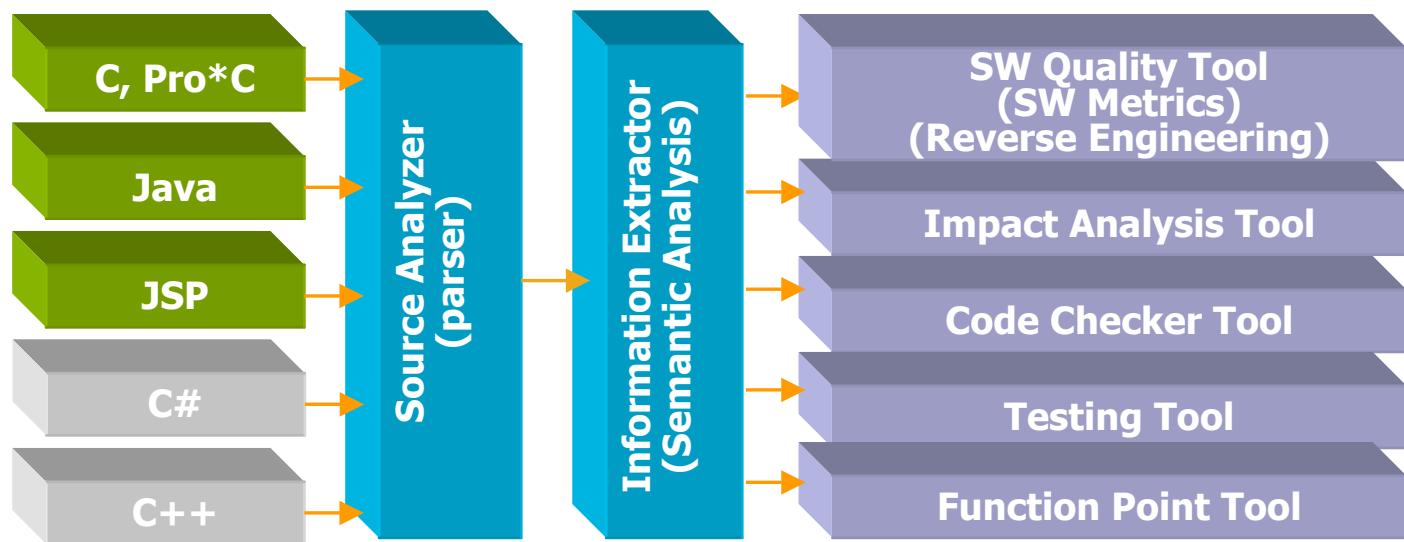
# 会社概要

- 2001年 法人設立(ETRI創業企業)
- 主要事業分野
  - ソフトウェア品質ソルーション専門道具開発
  - ソフトウェア品質コンサルティング及び教育
- 技術認証
  - ITマーク認証-情報通信部
  - GS品質認証-情報通信部
  - KTマーク認証-科技部
- 主要顧客



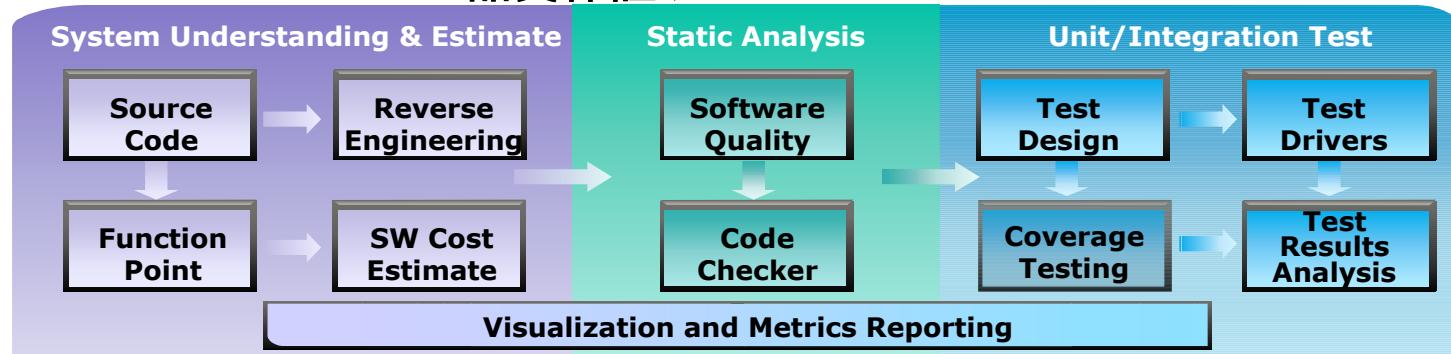
# RESORT for Javaの製品群

- RESORT for Java ?
  - Java言語用S/Wの分析, 品質の管理, テストのための品質統合解決の道具

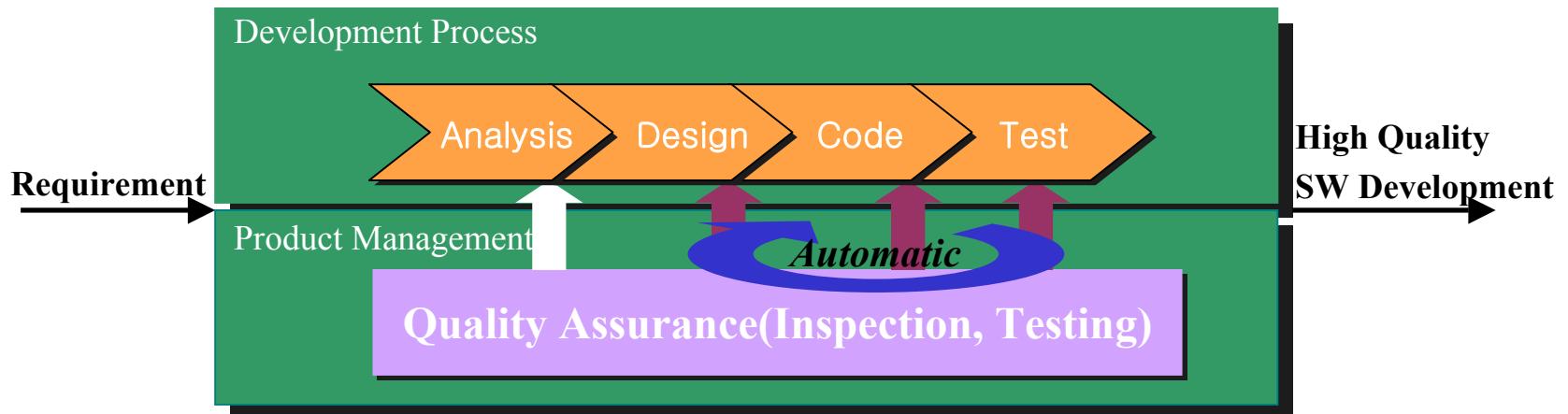


# RESORTの品質プロセス

- Soft4Soft – SW 品質保証ソリューション



- SWプロセスモデルとRESORT製品連繋も



# RESORT for Java 特徴

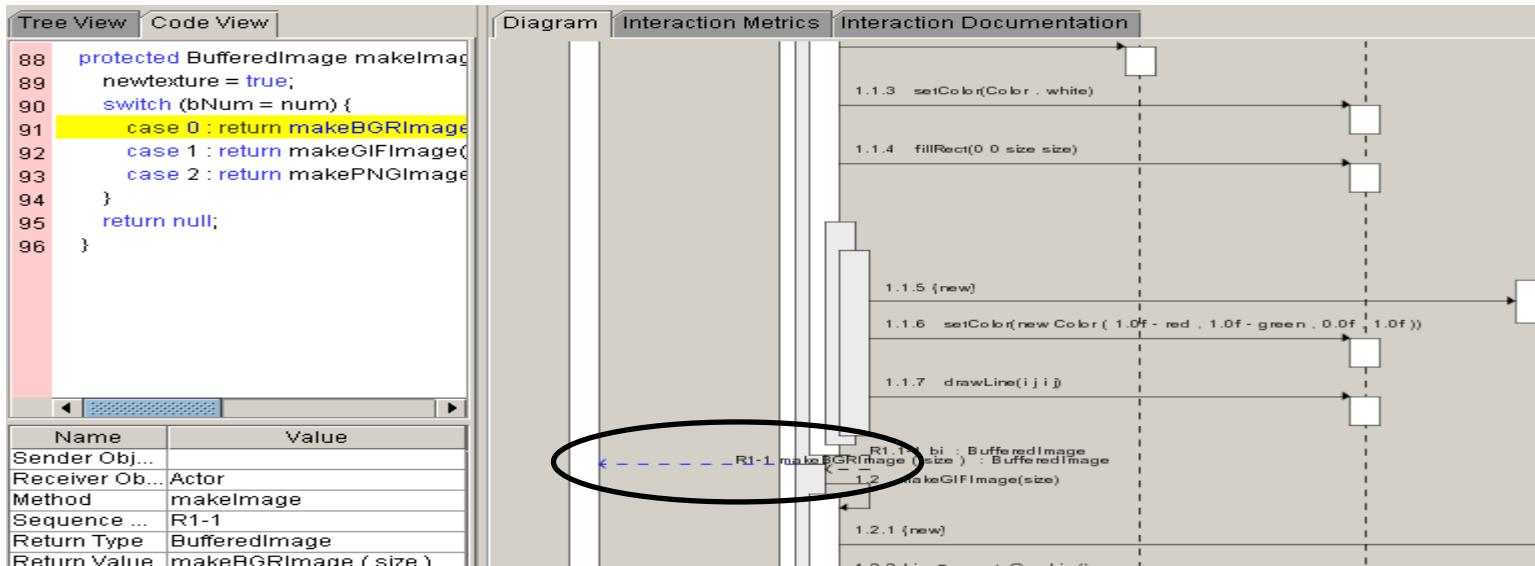
観点	特徴
支援対象	✓ Java 応用プログラミング(BC4J)及びwebプログラム(jsp, xml 除外)
支援範囲	✓ 具現初期のCode品質から統合テストまで品質過程
使用者	✓ Project Manager, Quality Manager, Developer, Tester
逆行分析工学の特徴	✓ 完璧なUML 1.4 Diagram 及び多様なGraph自動生成 ✓ 多様な品質及び文書の報告書
品質管理の特徴	✓ コディングの標準検査 – 構文品質の点検○○ 品質検証 – サイズ, 構造, 意味的な品質の測定
テスティングの特徴	✓ Test Plan及びTest Case Designの支援 ✓ Static/Unit/Integration Testingの支援 ✓ テスティングの結果, カバレッジ, 性能の分析 ✓ Graph基盤予想経路及びTesting実行経路をモニタリング
道具の効果	✓ 低費用、高品質、高性能のソフトウェア開発プロセスの構築

# RESORT for Java – Reverseの重要機能

## Reverse Engineering

- UML 1.4 Diagramの自動生成
- Procedural Graphの自動生成
- Diagram間の移動
- DiagramとCode間のマッピング

- 構造/行為/機能などシステムの分析
- ソフトウェア理解の向上
- ソフトウェアの構造及び設計の検証



# RESORT for Java – Reverseの重要機能

## Reverse Engineering(つづく)

- OO Metrics: 100+
- サイズ、構造、OO Metrics品質の測定
- Multi-levelの統計品質の報告書  
: Total, Avg, Max, Min

- ソフトウェア構造の理解
- 高危険品質のモニタリング
- 潜在的エラーの予防
- チーム/メンバー間のコミュニケーションの向上

Diagram Package Metrics Package Documentation Class Metrics Class Documentation

Project Name : Java2d

Name	Total	Average	Maxim...	Minim...
Number of Classes	36			
Number of Interfaces	0			
Number of Inner Classes	13			
Number of Inner Interfaces	0			
Class Complexity	573	15.91	58	3
Weighted Methods per Class	395	10.97	35	1
Lack of Cohesion of Methods	106	2.94	5	1
Response for a Class	201	5.58	13	1
Coupling between Objects	45	1.25	3	0
Depth of Inheritance Tree		1.00	1	1
Number of Methods Added	0	0.00	0	0
Number of Methods Inherited	0	0.00	0	0

Lack of Cohesion of Methods

Metrics Value	Classes	ClassPath
Tree	demos.Fonts	
Joins.DemoControls.Join1	demos.Lines	
5 DukeAnim	demos.Images	

Class Structure Metrics Class Size Metrics All Class Metrics Class Metrics per Package

# RESORT for Java – Qualityの特徴

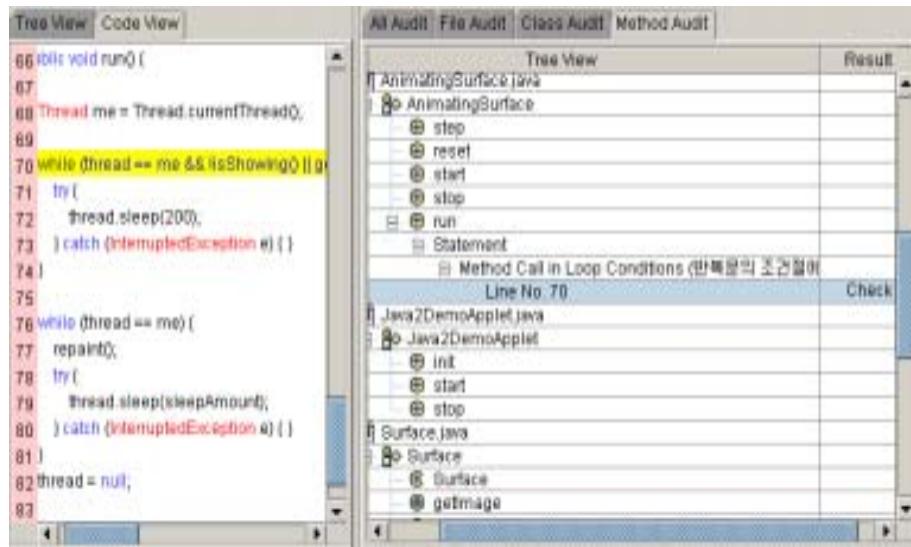
観点	特徴
支援対象	✓ Javaの応用プログラム
支援範囲	✓ Code品質からS/W品質までの全過程
品質の特徴	<ul style="list-style-type: none"><li>✓ ProjectからMethodまでの品質の測定, 評価, モニタリング<ul style="list-style-type: none"><li>• Codeの品質向上のためのCode欠陥検査</li><li>• 非最適化されたcodeの点検</li><li>• 品質改善のため、High-risk Classなどを識別</li><li>• ISO/IEC 9126-3の維持補修性の品質評価</li><li>• 設計明細の検証(verification)をPackage/Projectで品質評価</li></ul></li><li>✓ 品質の標準カスタマイズ</li><li>✓ Multi-levelの統計品質の報告書: Total, Avg, Max, Min</li></ul>
期待効果	<ul style="list-style-type: none"><li>✓ Code可読性及び維持補修性が容易</li><li>✓ S/W エラーを前もって予防</li><li>✓ オブジェット指向ソフトウェア品質の向上</li><li>✓ テスト及び維持補修時間の短縮</li></ul>

# RESORT for Java – Qualityの重要機能

## Code Checker

- BC4Jを含むJavaコードの標準点検  
: 140+(Style, Naming, Codingなど)
- Multi-levelのコード自動点検
- 説明及び練習問題の提供(Good/Bad Code)
- コード欠陥とCode間のマッピング

- 性能の低下、データなどのコードエラーの識別
- 性能の向上及びテスト費用の節約
- コードの可読性及び維持補修の向上
- 開発者の成熟度及び生産性の向上



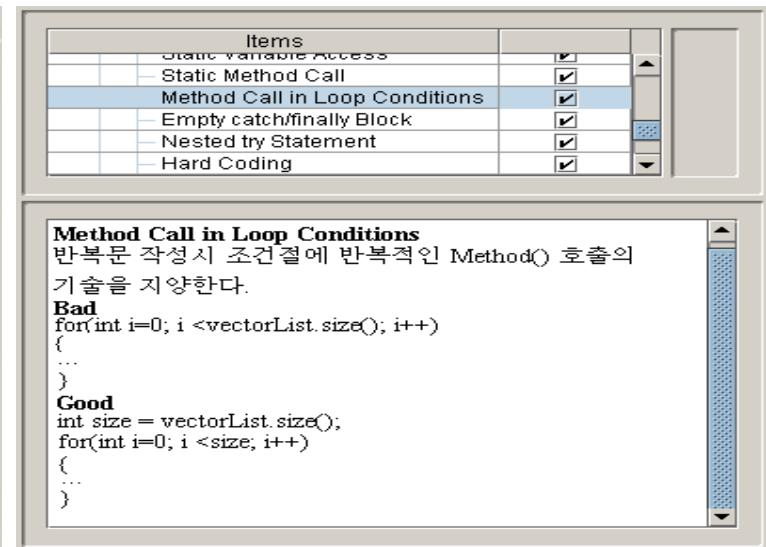
Code View

```
66 public void run() {  
67       
68     Thread me = Thread.currentThread();  
69       
70     while (thread == me && !isShowing()) {  
71         try {  
72             thread.sleep(200);  
73         } catch (InterruptedException e) {}  
74     }  
75       
76     while (thread == me) {  
77         repaint();  
78         try {  
79             thread.sleep(sleepAmount);  
80         } catch (InterruptedException e) {}  
81     }  
82     thread = null;  
83 }
```

Tree View

All Audit: File Audit: Class Audit: Method Audit

- AnimatingSurface.java
  - AnimatingSurface
  - step
  - reset
  - start
  - stop
  - run
  - Statement
  - Method Call in Loop Conditions (반복문의 조건절에 Method() 호출)
- Java2DDemoApplet.java
  - Java2DDemoApplet
  - int
  - start
  - stop
  - Surface.java
    - Surface
    - Surface
    - getImage



Items

Static Variable Access	<input checked="" type="checkbox"/>
Static Method Call	<input checked="" type="checkbox"/>
Method Call in Loop Conditions	<input checked="" type="checkbox"/>
Empty catch/finally Block	<input checked="" type="checkbox"/>
Nested try Statement	<input checked="" type="checkbox"/>
Hard Coding	<input checked="" type="checkbox"/>

Method Call in Loop Conditions

반복문 작성 시 조건절에 반복적인 Method() 호출의  
기술을 지양한다.

**Bad**

```
for(int i=0; i < vectorList.size(); i++)  
{  
}  
}
```

**Good**

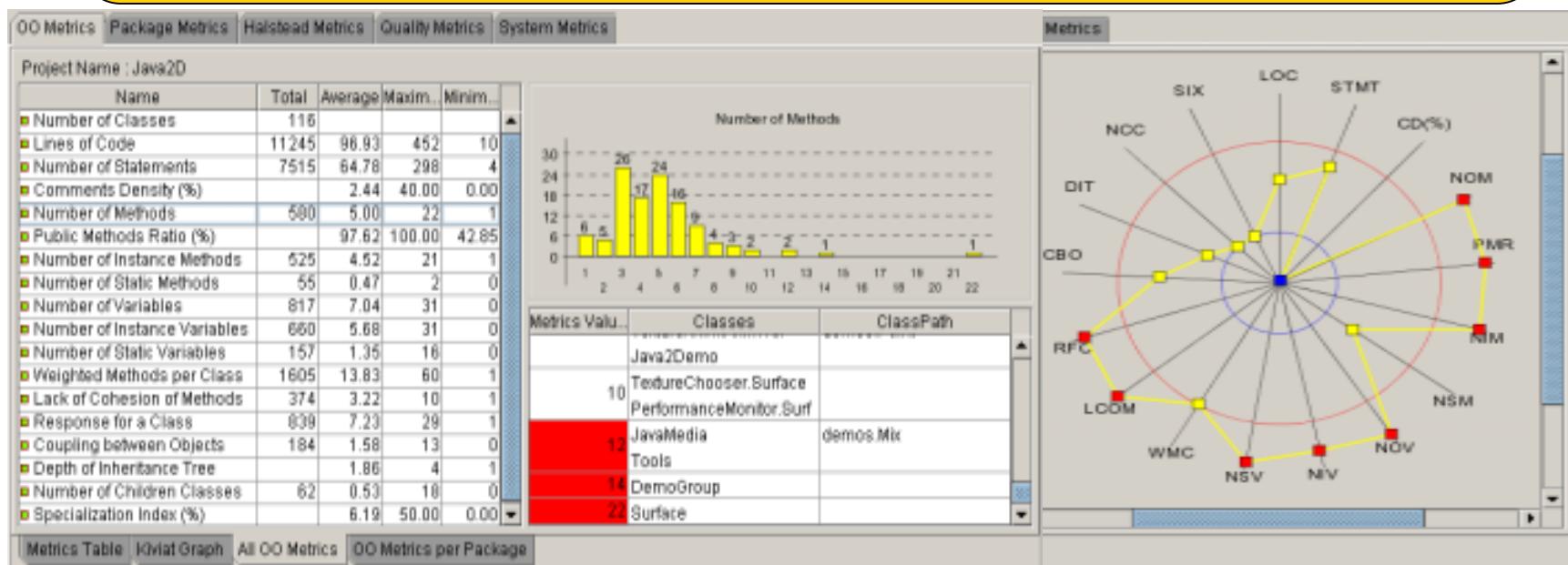
```
int size = vectorList.size();  
for(int i=0; i < size; i++)  
{  
}  
}
```

# RESORT for Java – Qualityの重要機能

## Software Metrics

- 最適化のコード検証
- OO Metrics品質の測定及び評価  
: +80(Kiviat, Bar, Circle Graphの生成)
- Metricsの基準値の設定
- Multi-levelの統計品質の報告書

- 非最適化されたコードの検出: プログラム  
サイズの縮小及び実行時間の短縮
- コードの可読性及び維持補修の向上
- 高危険品質のモニタリング
- Testing及び維持補修時間の短縮



# RESORT for Java – Testingの特徴

観点	特徴
支援対象	✓ Java応用プログラム及びWebプログラム(jsp,xml 除外)
自動化の範囲	✓ テスティング計画から結果分析まで至る全ての過程
テスティングの特徴	✓ Test Case作成のための最適化情報の支援 ✓ UnitとIntegration Testing レベルの同時支援 ✓ 順次/OO カバレッジ及び性能のMulti-level統計報告書 ✓ 実行結果をSequence Diagramなどの基盤で精密分析 ✓ 便利な使用者インターフェースの支援 - Diagram間の移動 - DiagramとCode間のマッピング
期待効果	✓ S/W 信頼性の向上、プログラムError検出及びS/W品質の向上 :正確性、性能性、信頼性、安定性 ✓ テスティング計画の期間及び費用の最小化 ✓ テスティング実行費用の最小化

# RESORT for Java – Testingの重要機能

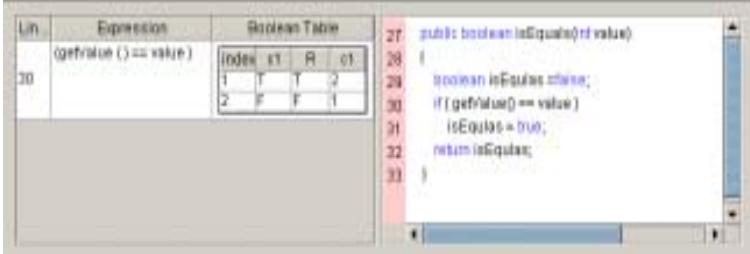
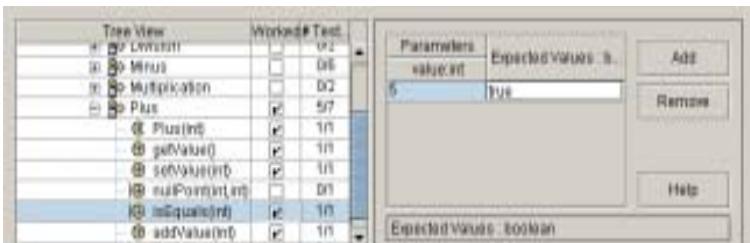
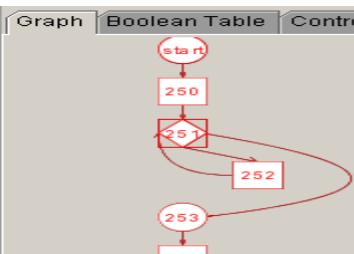
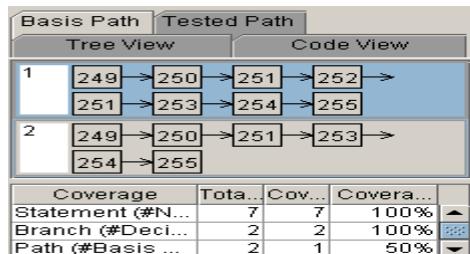
## Unit/Integration Testing 自動化

### Test Plan(Static Analysis)

- 最小経路(Basis Path)
- メッセージ流れの経路(ME-Path)
- テスティング優先順位の提供(V(g))
- 変数Dead Codeの検査

### Test Case Design

- Test Case/Scenario 設計UIの提供
  - : 重複テスト削除(Fan-In)
  - : Class/Package/Project 単位テスティング
- JUnit Framework 自動変換機能の提供



# RESORT for Java – Testingの重要機能

## Unit/Integration Testing 結果分析

Pass/Fail/Error分析の報告書: Test Dataの入力値と期待値の評価

-各Test Caseの実行経路を各グラフと連携して識別

-Error分析: Error Trace, Error Message(ソースコード及びシナリオエラーラインなど)

**Control Flow Testing Results**

Test Case	# Tes...	ClassPath	Scenario	Time...	Parameters	Return ...	Expect...	Result
Plus(int)	1	demos.calcul...	demos...	0.061	initValue:int =			Pass
getValue0	9	demos.calcul...	demos...	0.010		5	-1	Fail
getValue0	8	demos.calcul...	demos...	0.010		5	0	Fail
getValue0	13	demos.calcul...	demos...	0.020		5	5	Pass
getValue0	5	demos.calcul...	demos...	0.009		5	12	Fail
getValue0	11	demos.calcul...	demos...	0.010		5	199	Fail
tenLoop(int)	1	demos.calcul...	demos...		breakPoint:int		true	Error

**Error Trace**

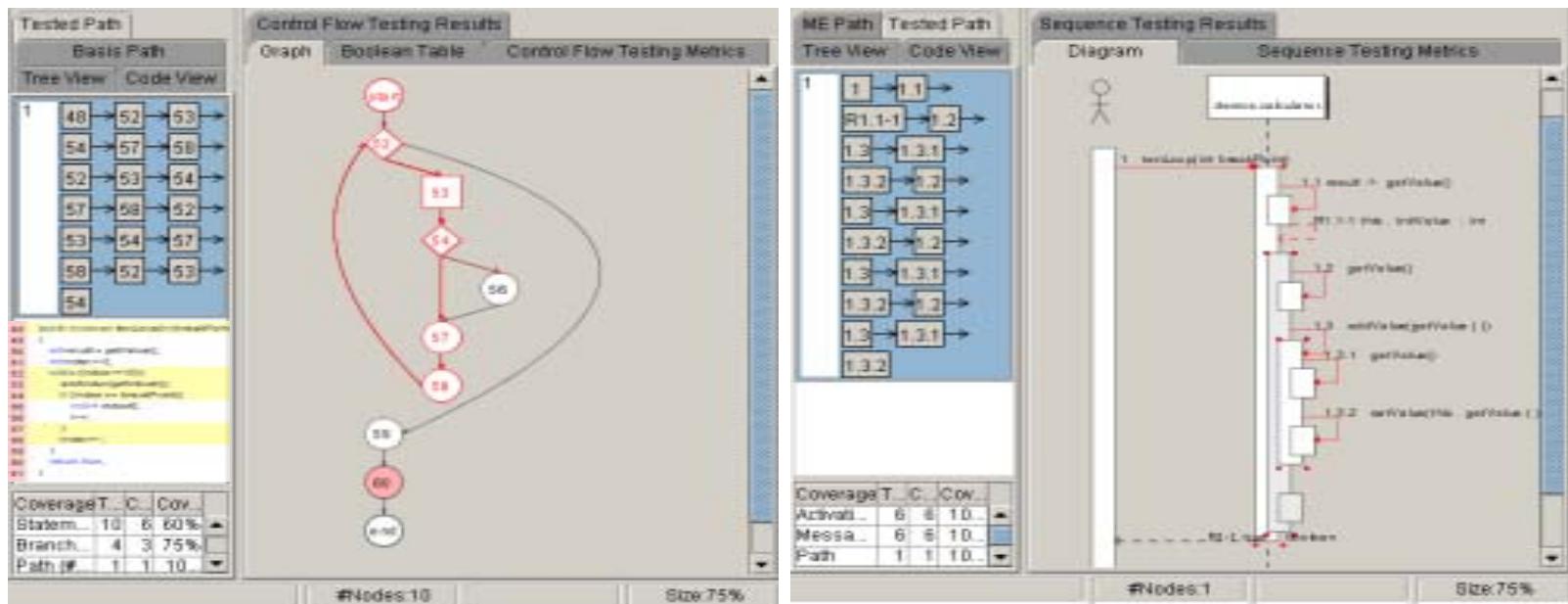
Test Case	# Tes...	Class...	Scen...	Error Trace	Error Messages
tenLoop(...)	1	demos...	de...	demos.calculator.Plus.getValue0 demos.calculator.Plus.setValue(int) demos.calculator.Plus.getValue0 demos.calculator.Plus.addValue(int) demos.calculator.Plus.getValue0 demos.calculator.Plus.setValue(int) demos.calculator.Plus.getValue0 demos.calculator.Plus.addValue(int)	demos.calculator.Plus.tenLoop(Plus.java:55) at demos.calculator.Plus Scenario0RTTest.<init> (PlusScenario0RTTest.java:26) at

# RESORT for Java – Testingの重要機能

## Unit/Integration Testing 結果のモニタリング

## Pass/Fail/Errorに対する実行経路を精密分析

- Integration Testingの実行経路: Sequence Testing
  - Unit Testingの実行経路: Control Flow Testing, Data Flow Testing
  - 実行された文章に対する実行経路のハイライティング
  - Errorの精密分析: Logic, Coding, Computational, Interface, Data Definition Errorなど



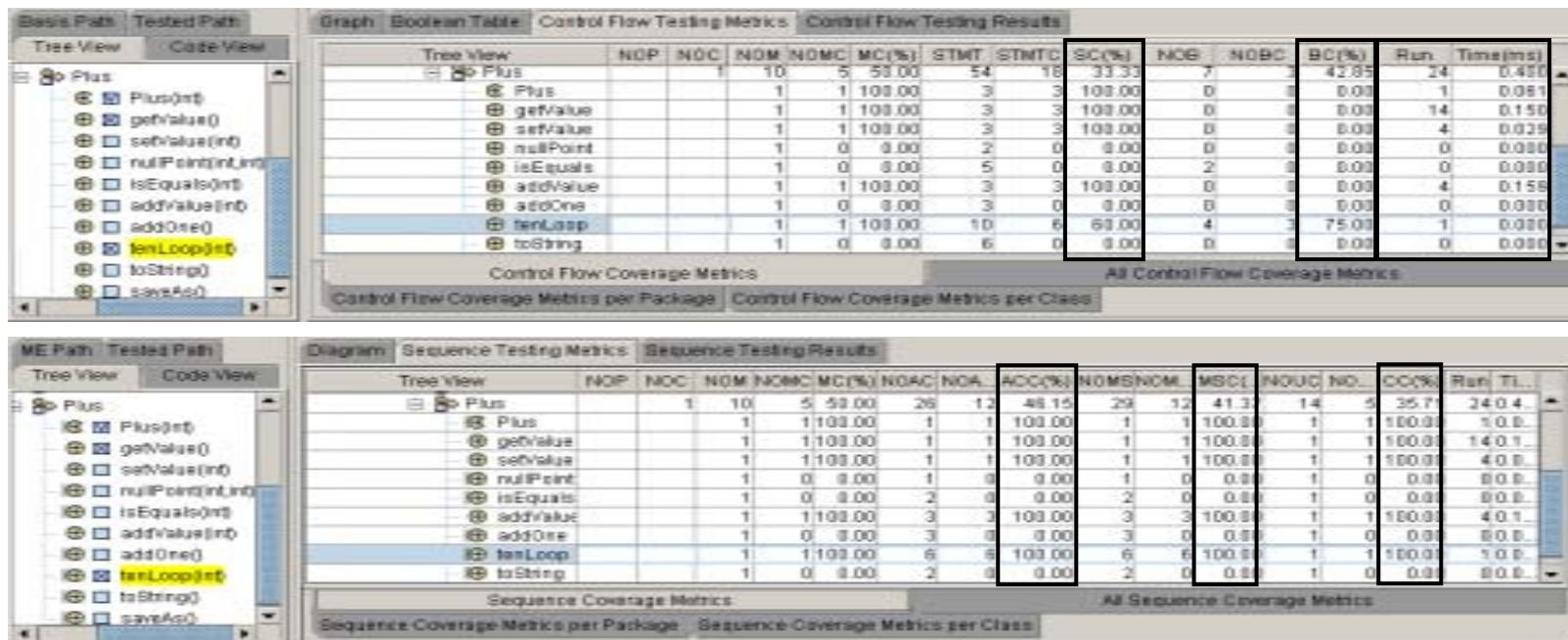
# RESORT for Java – Testingの重要機能

## Unit/Integration Testing 結果分析

Coverage分析の報告書: Multi-level統計のカバレッジ報告書 (30+ coverage )

- Control Flow Coverage: Statement, Branch
- Data Flow Coverage: All-DU Path, All-C-Uses Path, All-P-Uses Path
- OO Coverage: Class, Method, Message-send

性能(Time/Run) 分析の報告書: 性能ボトルネック現状の分析



The screenshot displays two windows of the RESORT for Java interface, both showing testing metrics for a 'Plus' class.

**Control Flow Testing Metrics Window:**

	NOP	NOC	NOM	NOMC	MC(%)	STMT	STMTC	SC(%)	NOB	NOBC	BC(%)	Run	Time(ms)
Plus	1	10	5	53.00	54	18	33.33	7	3	42.85	24	0.450	
@ Plus	1	1	103.00		3	103.00		0	0	0.00	1	0.081	
@ getValue	1	1	103.00		3	103.00		0	0	0.00	14	0.150	
@ setValue	1	1	103.00		3	103.00		0	0	0.00	4	0.029	
@ nullPoint	1	0	0.00		2	0.00		0	0	0.00	0	0.000	
@ isEquals	1	0	0.00		5	0.00		2	0	0.00	0	0.000	
@ addValue	1	1	103.00		3	103.00		0	0	0.00	4	0.159	
@ addOne	1	0	0.00		3	0.00		0	0	0.00	0	0.000	
@ tenLoop	1	1	103.00		10	6	63.00	4	3	75.00	1	0.000	
@ toString	1	0	0.00		6	0.00		0	0	0.00	0	0.000	

**Sequence Testing Metrics Window:**

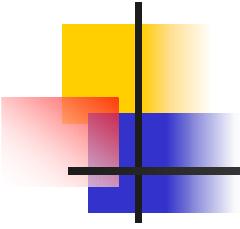
	NOP	NOC	NOM	NOMC	MC(%)	NOAC	NOA	ACO(%)	NOMS	NOML	MSCL	NOUC	NOA	OC(%)	Run	Ti...
Plus	1	10	5	53.00	26	12	48.15	29	12	41.3	14	5	35.7	24	0.4	
@ Plus	1	1103.00	1	1	100.00	1	100.00	1	1	100.00	1	1	100.00	1	0.0	
@ getValue	1	1103.00	1	1	100.00	1	100.00	1	1	100.00	1	1	100.00	14	0.1	
@ setValue	1	1103.00	1	1	100.00	1	100.00	1	1	100.00	1	1	100.00	4	0.0	
@ nullPoint	1	0	0.00	1	0	0.00	1	0	0	0.00	1	0	0.00	0	0.0	
@ isEquals	1	0	0.00	2	0	0.00	2	0	0	0.00	1	0	0.00	0	0.0	
@ addValue	1	1103.00	3	3	100.00	3	100.00	1	1	100.00	1	1	100.00	4	0.1	
@ addOne	1	0	0.00	3	0	0.00	3	0	0	0.00	1	0	0.00	0	0.0	
@ tenLoop	1	1103.00	6	6	100.00	6	100.00	6	6	100.00	1	1	100.00	1	0.0	
@ toString	1	0	0.00	2	0	0.00	2	0	0	0.00	1	0	0.00	0	0.0	

# 品質の測定及び評価の事例(総合)

## ■ 初期品質確保の重要性

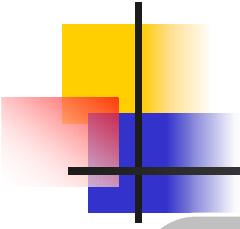
- 開発の初期からソフトウェアエラーを前もって予防できる環境の必要
- Testingの危険性能(Run/Time)問題は大部分高危険Code Qualityと関連され、Coding段階で高品質の確保が必要

Class Name	Code Quality		Test		Method (covered)
	Understandability	Stability	Run	Time	
ButtonTagHelper (Sub-Class)	Good	RC	1	0.029	1 (1)
CheckBoxTagHelper (Sub-Class)	Good	RC	4	0.150	1 (1)
FieldPropertyGenBean	FD, IC	RC, BD, RCC, RMC	1195	22.466	49 (30)
ImageInputTagHelper (Sub-Class)	IC	RC, RMC	0	0	1 (0)
RadioTagHelper (Sub-Class)	Good	RC, BD	6	0.180	2 (2)
TagHelper (Top Level)	BD, IC	RC, BD, RMC	1084	5.737	36 (21)
TagUtil	Good	RC, BD, RMC	2302	3.825	21 (11)
TextAreaTagHelper (Sub-Class)	Good	RC	0	0	1 (0)
TextInputTagHelper (Sub-Class)	IC	RC	28	1.573	1 (1)
Bad Quality Ratio(Bad Quality/Total)	8/9	13/13	3/13	4/13	69/116



# 結論

- RESORT品質solutionの導入効果
  - ソフトウェアの品質
  - ソフトウェアのプロセス成熟度の向上
  - ソフトウェアの開発及び維持補修費用の節約
  - ソフトウェアの開発及び維持補修期間の短縮
  - ソフトウェアの品質プロセス及び外部注文管理サービスの支援
- 主仕事分野
  - ソフトウェアの品質に対するsolution道具の開発(Java, C, C++, C#)
  - ソフトウェア品質のカンサルティング
- サービス支援の対策
  - 迅速な技術の支援及び教育
  - 使用者の要求に答える製品の供給
- 技術の資料及び品質の診断ガイド
  - <http://www.soft4soft.com>



**If you cannot MEASURE it, you cannot IMPROVE it**

# Soft4Soft

**T205, ICU VBI Center, 103-6, Munji-Dong,  
Yousung-Gu, Daejon, 305-714**

**Tel : +82-42-866-6632~3  
Fax: +82-42-866-6626**

**Sale Supports : [sales@soft4soft.com](mailto:sales@soft4soft.com)  
Technical Supports : [info@soft4soft.com](mailto:info@soft4soft.com)**

**[www.soft4soft.com](http://www.soft4soft.com)**