使用 **EA**

Enterprise Architect 的测试管理

编辑: Dermot O'Bryan

http://www.sparxsystems.com

目录

介绍

在建模环境内管理测试有很多优点。这篇说明将向你阐述如何使用 Enterprise Architect 创建测试定义，及如何将这些定义和开发过程中的其它工件联系起来。

本文所涵盖的主题如下:

➢ 第一部分 测试工具 包含的核心功能有:

o 将内部资源库数据导入测试用例时，可以创建和浏览测试。

o 使用 Enterprise Architect 的建立，测试，运行和调试工具来定义和实现执行时的测试。

o 创建，存储和连接测试的计划和文档。

➢ 第二部分 建模环境下的测试管理 包含了在不同配置下使用测试工具的方法，它们包括:

o 为存储测试数据设置不同布局

o 在可实现测试用例间，连接元素来实现可跟踪性

o 测试用例信息报告

o 检索测试结果

o 使用 profiles 来添加元素的常规属性

测试工具

Enterprise Architect 支持三种与测试关联的核心功能。它们以多种形式结合使用。 这些功能是:

➢ 测试用例可以使用:
- 任意元素下的测试用例，请看: 元素的测试用例。
- 一个指定元素类型， 请看: 测试用例元素。

➢ 建立，运行，调试和单元测试。 这包含使用 MDA 转换建立 xUnit 测试， 从中生成测试结果，并当运行代码时，生成顺序图。

➢ 测试计划 – 附加测试文档 使用链接文档和文件工件元素列出文档和测试计划概要。

测试用例的定义:

元素的测试用例（Element Test cases）

多个用例可以指定给任意一个元素(从需求到类，节点和组件)。

元素的测试用例使用测试用例窗口来定义，可以从菜单处选取: View | Testing (Alt+3)，

如图 1 所示。

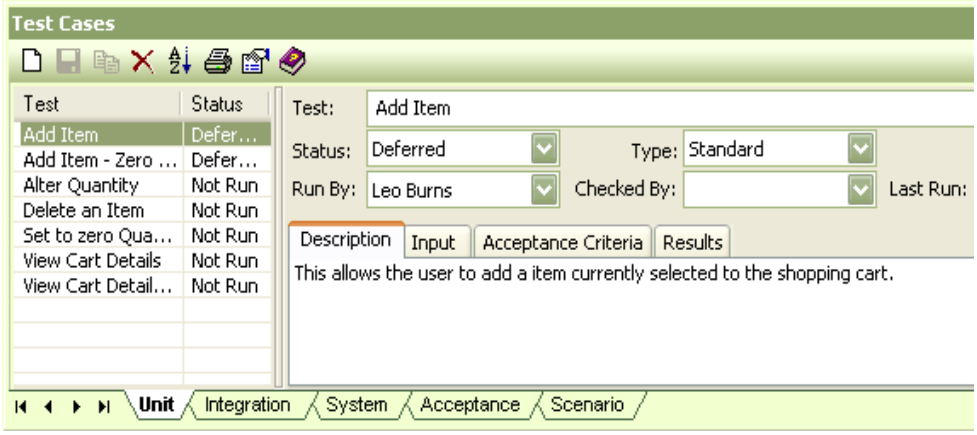


图 1 – 包含编辑面板的测试用例窗口。

该窗口有两种显示模式:

a) 显示属性模式:

可以在右侧面板对该域直接进行编辑。

b) 隐藏属性模式:

提供测试用例的概要浏览，在外部窗口执行编辑。

这两种模式可使用显示/隐藏属性按钮（Show/Hide Properties button）进行转换: 

测试用例区（Test Case Fields）

测试用例窗口（Test Cases ）允许在不同的测试中定义和编组任意数量的测试用例，如: **单元，集成，系统，验收和情况（Unit，Integration，System，Acceptance &Scenario）**。这些编组在测试用例窗口使用标签分类(请看上面 图 1)。

测试用例窗口有两个子窗口。左侧是测试用例的简短显示列表，右侧是输入和编辑数据区。

下面表格显示测试用例窗口的各编辑区的描述:

控制	说明	定义者
Test（测试）	测试名称	
Status（状态）	当前测试状态(通过，失败。。。)	
Type（类型）	测试类型	用户定义 *
Run By（运行者）	试运行测试人员	用户定义 **
Checked By（审核者）	试运行审核人员	用户定义 **
Last Run（最后运行）	最后运行的测试数据	
Description（说明）	测试说明	
Input（输入）	定义输入数据	
Acceptance Criteria（验收标准）	定义验收标准	
Results（结果）	最后测试结果清单	
Defined Tests（定义测试）	与该元素联系的被定义的测试清单	

* *Type 区域输入是用户可定义的，从: Settings | Maintenance | Testing 选择*

** *Run By 和 Checked By 也是用户定义的，请看:Settings | People | Resources*

设置可显示测试脚本的图

所有的图都可以选择显示元素内定义的测试用例。在图内显示测试定义的列表，可以使用以下的方法:

- 选择 F5 | Elements | Show Compartments
- 从主菜单选择: Diagram | Properties | Elements | Show Compartments
- 鼠标右键点击图，并且在上下文菜单选择: Properties | Elements | Show Compartments

通常，检验一组类需要定义测试。 不管如何，测试最好显示在同一个指定测试的图内。 这张图可以放置在远离主图(显示属性和操作)的包内。

图 2， 下面是一个带属性和操作的类图实例。

图 3 是一个测试图的实例，用来设置显示这些相同元素的测试用例。

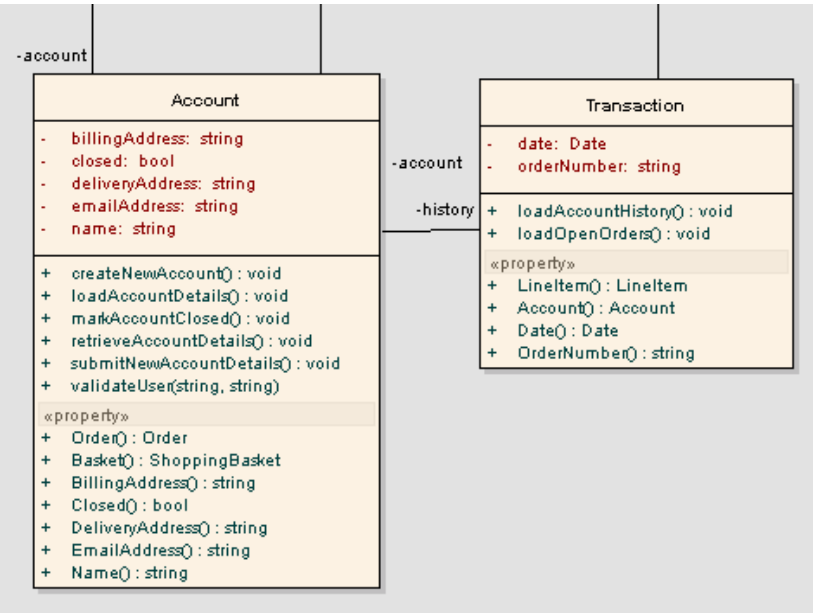


图 2 是一个带属性和操作的类图实例

图 2 – 原始类图（Class Diagram）

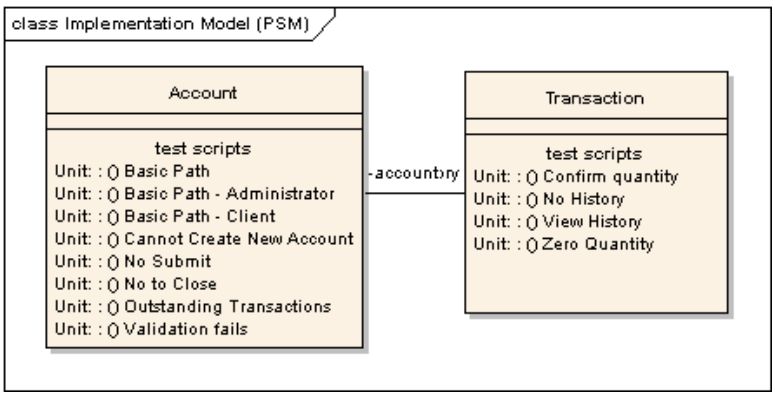


图 3: 包含与上图一致的相同元素实例，但是属性项被设置为:
- [✓]Tests
- [ ] Attributes and Operations

图 3 – 该图有被设置的测试视图，但是未设置属性和操作

测试的图内编辑

在图中测试用例是可见的，我们可选择任何一个测试用例对其编辑。所用方法如下:

1. 使用鼠标选中测试项目
2. 双击选中的该项目

我们可以直接访问测试用例的细节。如果测试用例窗口当前没打开，那么这将打开选定测试项目的细节。图 4， 元素中选中的测试用例被高亮，并显示在测试用例窗口中:

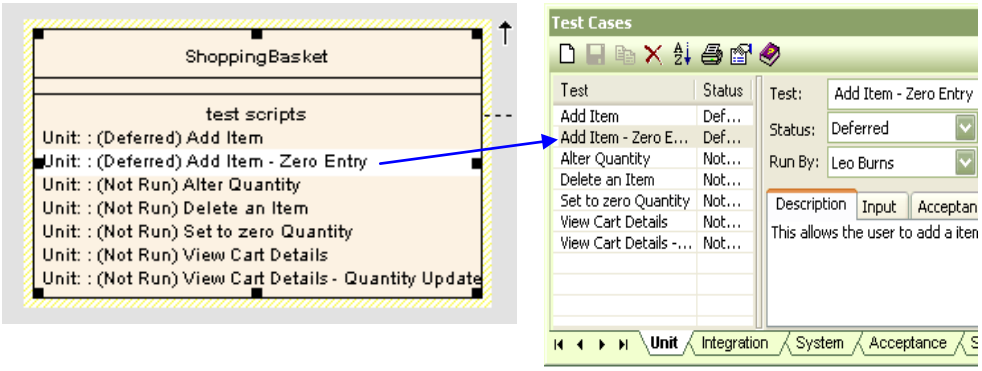
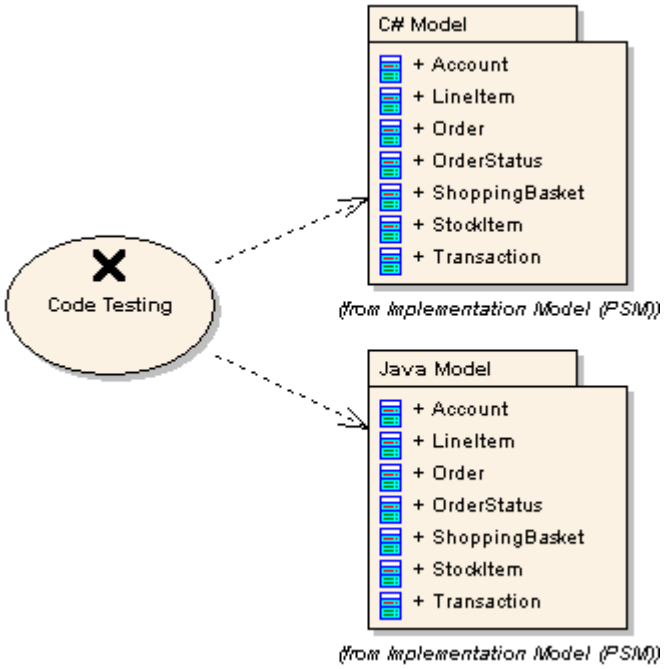


图 4 – 元素中测试用例的选择

从情况，条件，需求和其它测试的导入

测试用例窗口支持从其它元素导入数据。 这包括从情况，条件，内部需求和测试用例。更多详细信息，请看附录: Import from Scenario, Constraints, Requirements and Other Tests 测试用例元素（Test Case Element）

Enterprise Architect 支持一种定制的元素类型，称之为“测试用例”（Test Case）。它位于 Custom 下的工具箱（Toolbox）里 。下面是一个测试用例 的实例：



在这个例子中，对两个包而言，代码测试都是适合的。因此，相关测试用例被定义在一个测试用例下，能被重新使用。这种方法防止了复制，确保了测试用例定义改变时统一性。

图 5 – 测试用例元素

测试用例元素是用于:

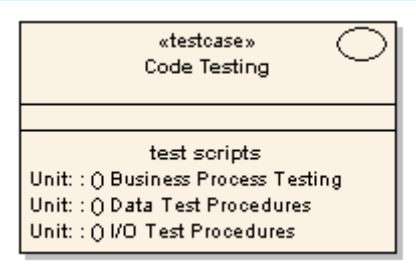
- 当一个测试用例需要配合其它元素在一个图中被查看
- 定义多个元素共同部分的测试
- 在附加文档包含测试方针和定义的过程

这些测试元素使用标准 UML 连接器（connectors）连接到其它 UML 元素。较好的例子是：一个包（package）含有为两个不同平台设计的类，但是它们有同一个用于测试的概要。

在上面的用例中，一般测试过程可被定义在一个文件内，并应用于这两个平台。因此在测试用例元素中，它可用链接文档或文件工件类形元素的形式，将其包含在这个模型里。 关于更多测试过程中包含链接文档的细节，请看下面的“Linked Documents and Document Artifacts”（链接文档和文件工件）。

注意: 测试用例元素（Test Case elements）可以利用测试窗口来包含可视的内部测试 (Alt-3)。

对于非矩形形状的元素，在同一张图上查看测试，如：测试用例元素。应首先设置矩形标号：右键单击元素，在弹出上下文菜单中选择: Advanced | Use Rectangle Notation。它的过程描述在: Setting a Diagram to Show Test Scripts.



建立，运行，调试和单元测试

Enterprise Architect 可以为主要的第三方开发平台提供建立，运行，调试工具的接口。这些功能包括以下:

- 建立测试: 使用 MDA 转换创建 NUnit 和 JUnit 类存根。
- 针对单元测试类记录单元测试结果。
- 从 EA 直接建立，测试和运行: 为 Builds 设置包构建脚本，单元测试，运 行和调试。
- 测试与可视化: 使用 EA 调试平台窗口工具。
- 性能分析和可视化: 从运行时调试生成顺序表。

Build 脚本功能维护一个包运行时的组件。你可以设定包怎样被建立 (编辑)， 指定调试器，配置测试，及包如何被部署的细节。

用来定义建立，测试和运行的选项在主菜单: Project | Build and Run

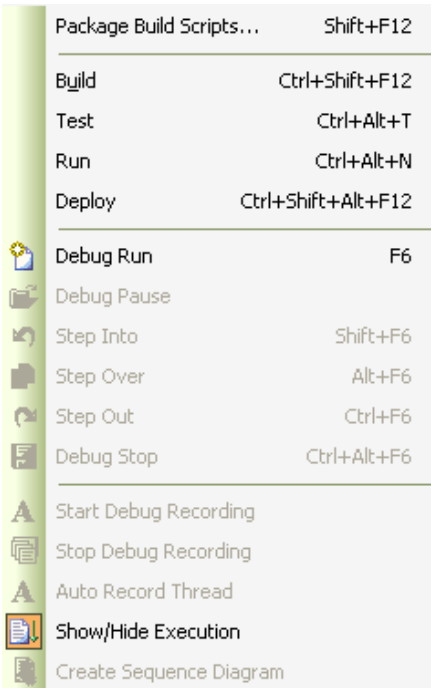


图 6 - Project - Build and Run 子菜单

Build Scripts, Builds, Test & Run 在此菜单中的输入可直接在 Debug Workbench（调试工作台）完成。 当与代码交互时，这样能在更新数据时进行高速存取。相关更多信息，请参看帮助文档并检索: “Debug and Profile”

下面是建立，测试和运行相关测试功能的简短描述 (及帮助文献)。

MDA 转换 NUnit 或 JUnit 测试类

在 NUnit 或 JUnit 被用来测试代码部分，单元测试类能被手工创建，也可以做为重复过程一部分，自动的使用 MDA 转换生成测试用例。

对现有类结构进行 MDA 转换可以生成一个简单类结构（class structures），使用 Project | Transformations | Transform Current Package:

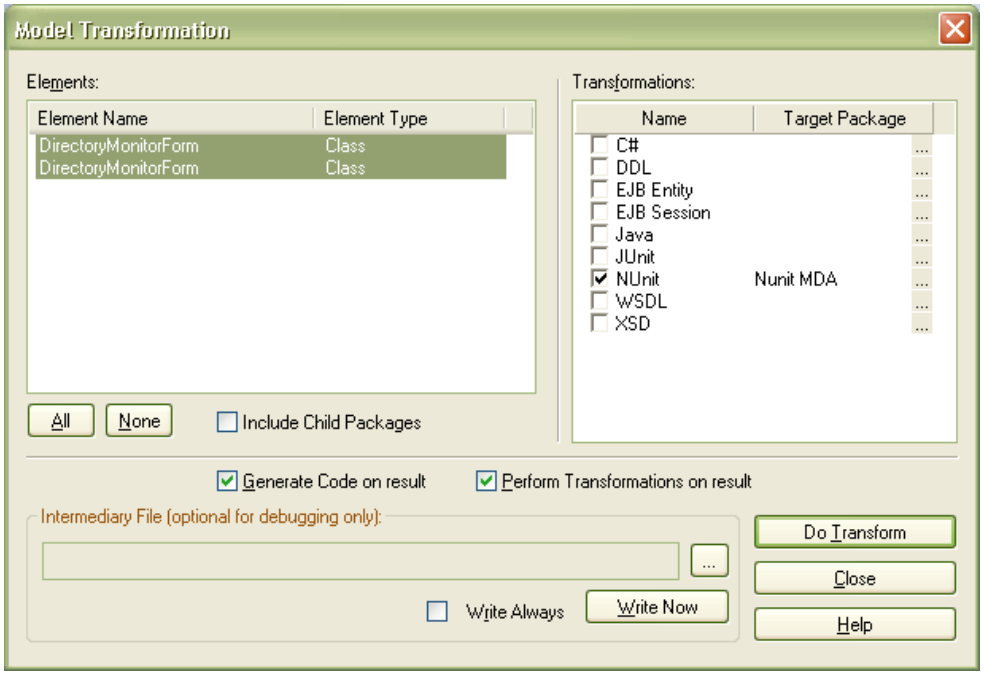


图 7 - NUnit 测试的 MDA 生成

关于使用 MDA 转换的信息，请到帮助索引下使用下列主题检索:“Built-in Transformations” “NUnit” “JUnit” 和 “Unit Testing”

对建模环境的编译错误跟踪

使用 Enterprise Architect 来初始化编译过程，它允许在结果视窗记录编译器的警告和错误(Ctr-Shift+F8)。 这些结果可以从输出视窗进行复制，以便于进一步处理。详细信息，请在帮助文档中查看: “Build Commands”。

测试用例记录 xUnit 测试结果

在 xUnit 操作中，输入检测代码，然后执行对 xUnit 类的汇编，为每个操作的测试用例都将被添加到类元素中。它们记录这些测试操作的时间和状态。其结果可以通过测试用例窗口查看，或者通过把图的属性设置为: [✓] Show Tests 查看。

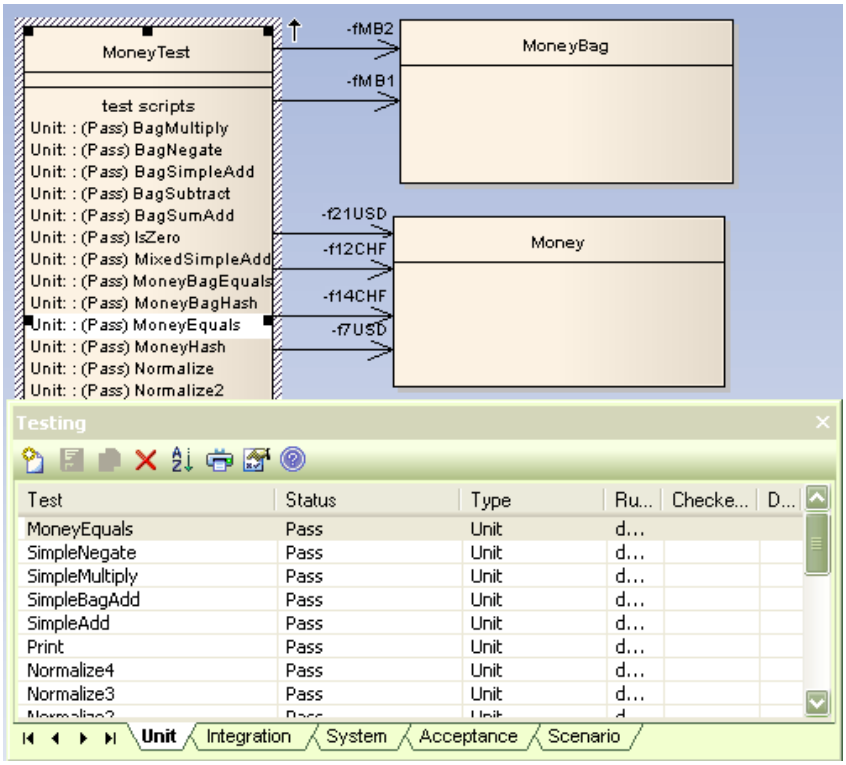


图 8: 在 EA 中执行 xUnit 的测试，其测试结果自动在单元类中生成测试用例。

此图显示了该图的属性已经设置为 show Test Cases。通过测试窗口，这些测试用例显示如下。

图 8 - xUnit 测试中测试用例的生成

更多信息，请看 Enterprise Architect 帮助索引: “Unit Testing”

测试与可视化 – 使用调试工作台

使用 Enterprise Architect 调试工作台（Debug Workbench），你可以轻松执行代码，并根据给定测试情况，检验系统性能。

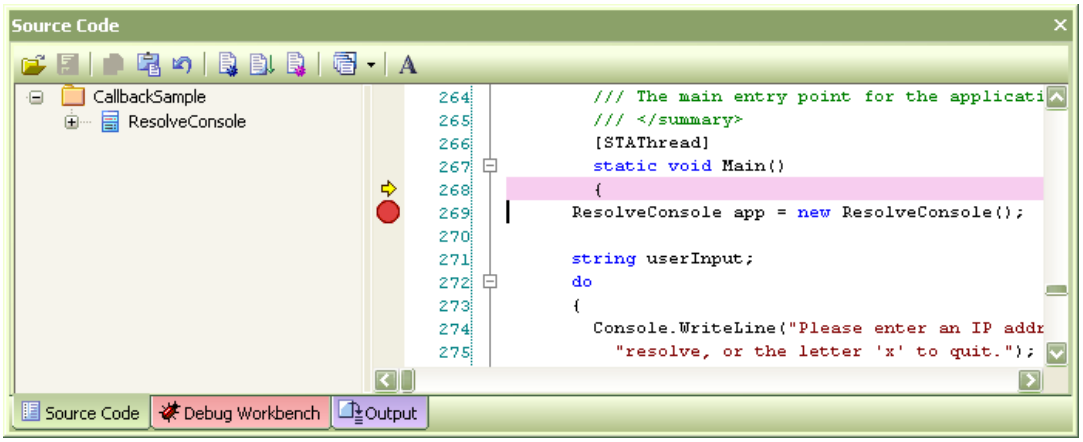


图 9 –浏览设置断点的当前执行源代码

Enterprise Architect 不仅允许在代码中设置断点，浏览局部变量和堆栈跟踪细节；也允许你通过记录顺序图的执行来可视化这些细节。通过调试工作台（Debug Workbench ）与 Enterprise Architect 源代码浏览（Source Code）配合来完成该任务。

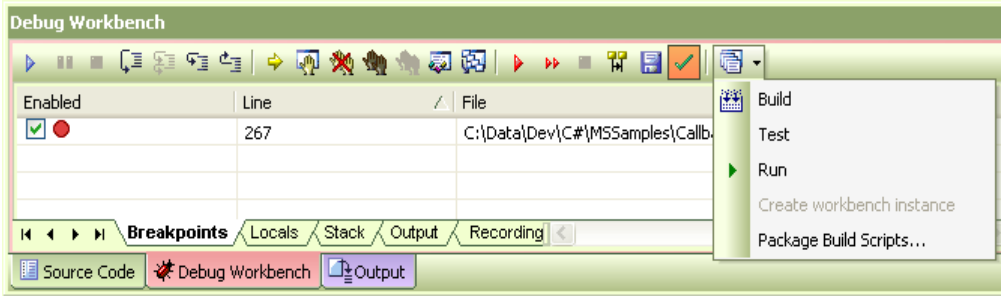


图 10 – 调试工作台

更多信息请参看 Enterprise Architect 帮助索引: "Profiling and Debugging"。

代码执行的顺序图（Sequence Diagrams of Code Execution）

调试器执行代码可以生成顺序图。这里有两种适用方式:在类中选择主运行程序，并在弹出上下文菜单中选择生成顺序图，或者在代码中设置断点，在两个断点中间生成顺序图。

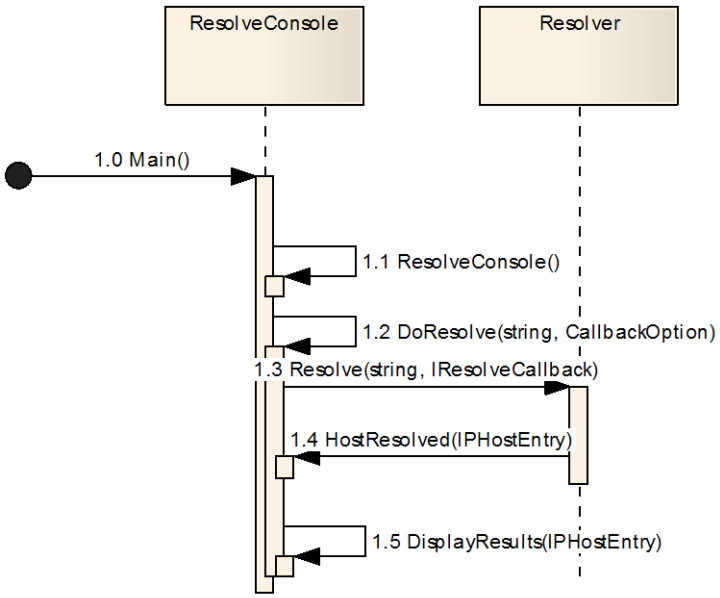


图 11 – 从执行代码生成执行图的实例。

生成顺序图的更多信息，请看帮助: "Generate Sequence Diagrams"。

测试计划 – 附加测试文档

这里有很多与测试过程关联的文档。Enterprise Architect 提供了几种向元素附加关联文件的方法，如下:

- 使用到外部文档的链接。
- 内部存储的 RTF 文档。 Enterprise Architect 有两种内部存储的 RTF 文档类型:

- 链接文档
- 文档工件元素（Document Artifacts Element）。

外部文档

Enterprise Architect 元素可以在 Element Properties 对话框下使用文件标签（file tab）来包含一组外部文件参考。 但是仅仅存储了此参考的文件路径，而不是内容，因此这意味着任何格式的文件都可以被用来参考 (“Launched”)。 类似的，生成报告也仅包含这些参考的元信息，如文件路径和说明，而不是文件内容本身。

链接文档

RTF 文档可以做为内部存储文件加入到任何元素中 (Linked Documents)。 完成这个工作，我们需要选择一个元素并使用下列:

- （右键点击）Right –click | Linked Document
- Ctrl-Alt-D

这会打开 RTF 编辑器 ，我们需要选择从一个模板或从空白文档开始。这里有一组默认的适用模板。你可以从资源区定义（或导入）你自己公司的模板: Resources View | Templates | Linked Document Templates。

文件工件元素（Document Artifacts Elements）

文件工件元素可以获得从:

- Toolbox | Deployment | Document Artifact。

这些元素一旦建立，元素属性窗口将默认为显示。关上此窗口后，双击元素将打开 RTF 编辑器。

RTF 编辑器

RTF 编辑器提供了完全可编辑的 RTF 文件。在编辑区内，上下文菜单所有选项都是可用的。下面是上下文菜单的视图:

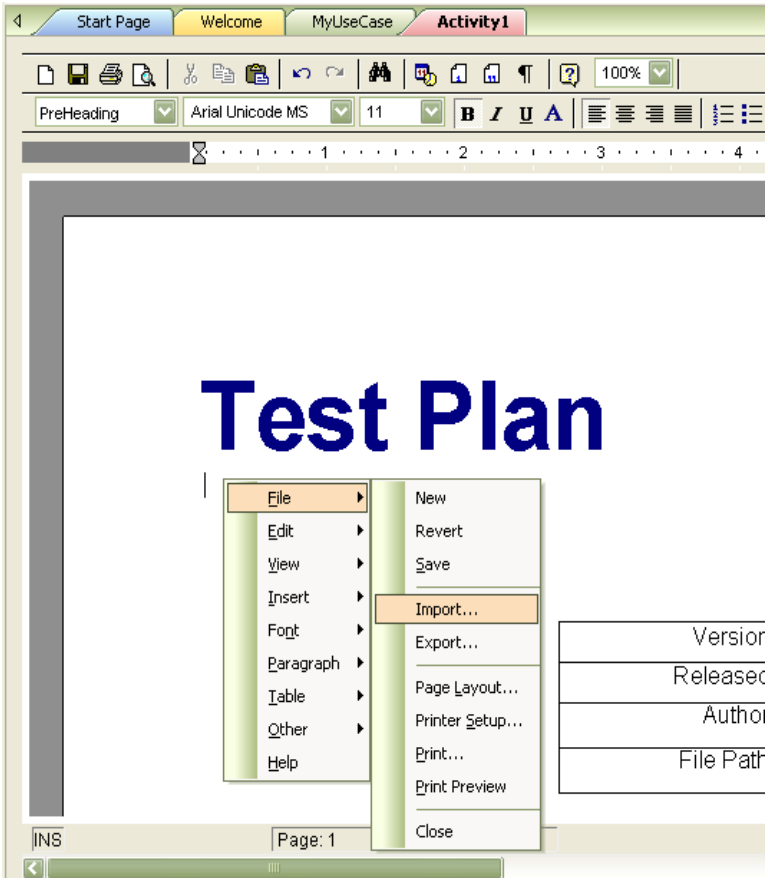


Figure 12: RTF 编辑器提供了一个全功能菜单的编辑器，我们可在文本区任何地方右键单击进入。

Figure 12 – 内部 RTF 编辑器菜单

使用上下文菜单向 RTF 编辑器导入文件: File | Import。例如，你能从外部文档导入 'RTF saved' （RTF 保存格式）的复制文件，如一个.doc 文件。

输出 RTF 文档

在 RTF 编辑器中，这些文件可以直接从上下文菜单输出: File | Print。

这些文档也可以使用 RTF 报告生成器输出，通过使用包含 RTF 报告模板的下列部分:

- Element::Model Document
- Package::Package Element::Model Document。

下面是 RTF 编辑器编辑区视图，它已经选中了:Element::Model Document

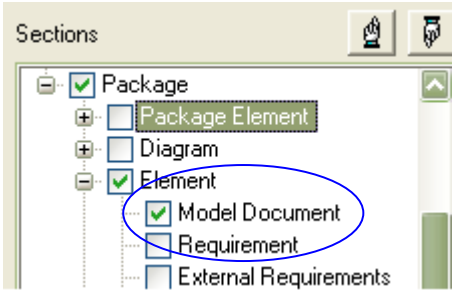


图 13: RTF 报告的输出模板部分:

- Element.Linked Documents （元素链接文档）
- Document Artifacts （文档工件）

图 13 - RTF 文档部分的报告生成器

更多信息， 请看帮助文档: “Creating Documents | RTF Documents” 和 “Modeling with Enterprise Architect | Working with Elements | Linked Documents”

模板

当使用“Copy Template”下拉菜单创建一个新文件，有一个默认适用的“测试计划”。用户定义模板（如一个公司标准测试计划模板）可以加入到 Enterprise Architect 资源区:

Resources | Templates | RTF Templates | Linked Document | Templates。

建模环境下管理测试用例

建模的测试用例根据开发方法学有不同的方法，在这部分，我们将讲述几种具体方法，它们容易适应不同的方法学。

我们第一个方法是最常用的“白匣子测试”，它是根据特定系统的主要具体方面来定义测试。第二种是适用于“黑匣子测试”条件下，测试针对系统功能需求，不考虑实现的细节。

在这两种情况下，我们主要考虑：测试用例对模型层次关系而言如何组织，与其牵连的报告，可跟踪能力，及其它分析活动。

白匣子测试: 实现元素定义测试

下面是 Enterprise Architect 模型实例中的一段摘要，它显示包含的测试元素，在不同图布局下的实例。 显示了标准类图格式，不显示测试信息。 使用测试浏览图形式。

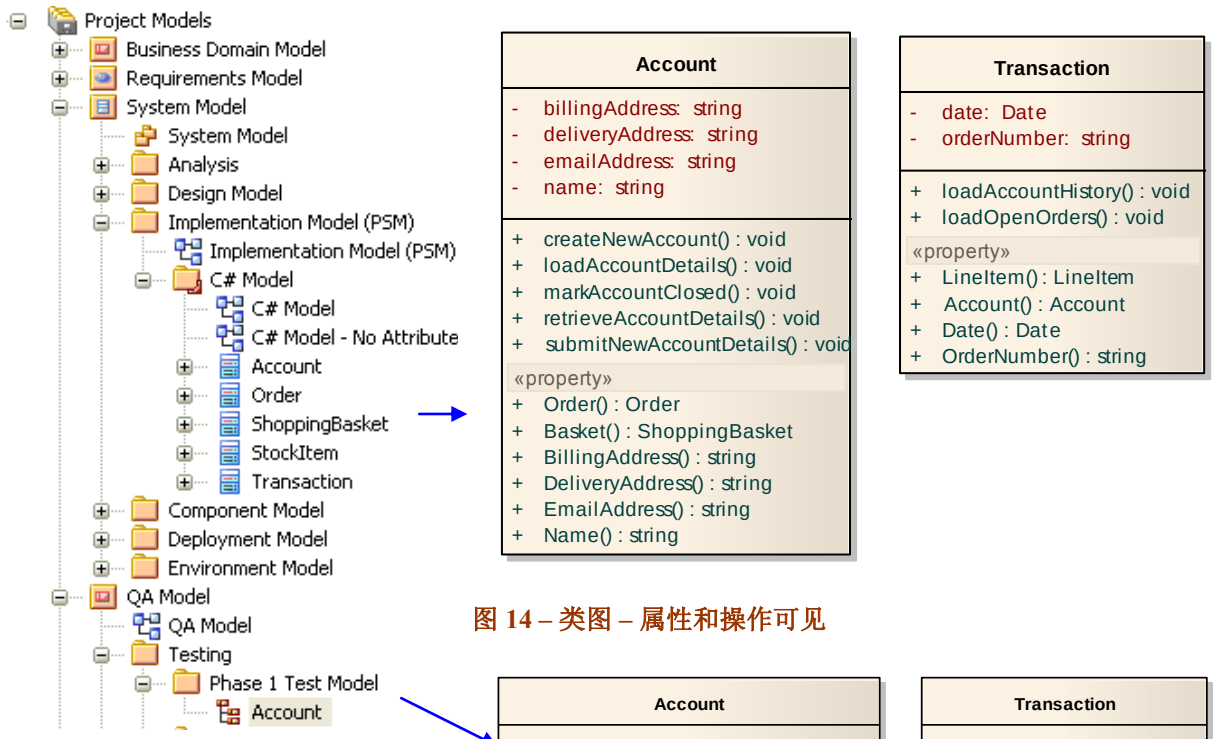


图 14 – 类图 – 属性和操作可见

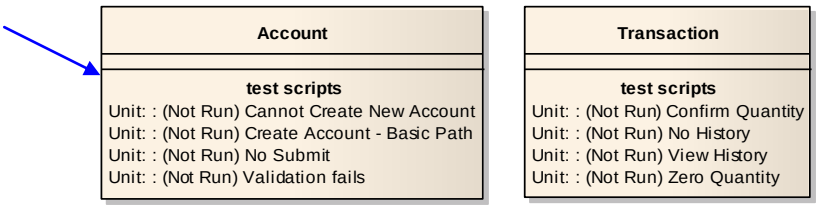


图 15 – 类图 – 测试用例可见

像上面直接针对执行元素定义的测试，适合于对新创建的类进行最初的测试或当作一个永久的“回归测试”。它也可以用于联系测试到指定系统行为的元素，如需求或用例元素。

> **注意:** 在高层次抽象阶段，以上方法也可以相应的采用“黑匣子测试”如：需求元素包含内部测试。

黑匣子测试: 使用维护元素来建立可跟踪性

在系统开发和维护阶段，问题报告针对指定组件，并随时间不断加强。Enterprise Architect 的 Issue and Change elements（问题和生成元素）包含测试定义，并以此种方式提供从特定系统维护事件到受影响系统组件的可跟踪性，及它的后继测试。下面是一个使用账户类的简单例子（早期描述）：

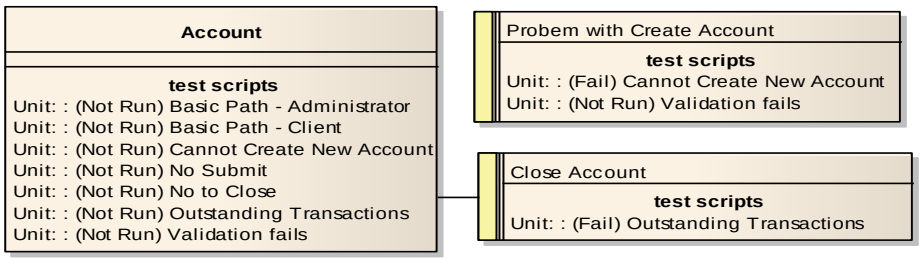


图 16 – 类和关联包含测试的问题

当使用单独元素记录测试，一个关系提供了从系统元素到它测试的直接跟踪，如一个依赖连接器（Dependency connector）。

这种关系可以被简略构成如上图所示的例子，或者如图 17 的关系矩阵。

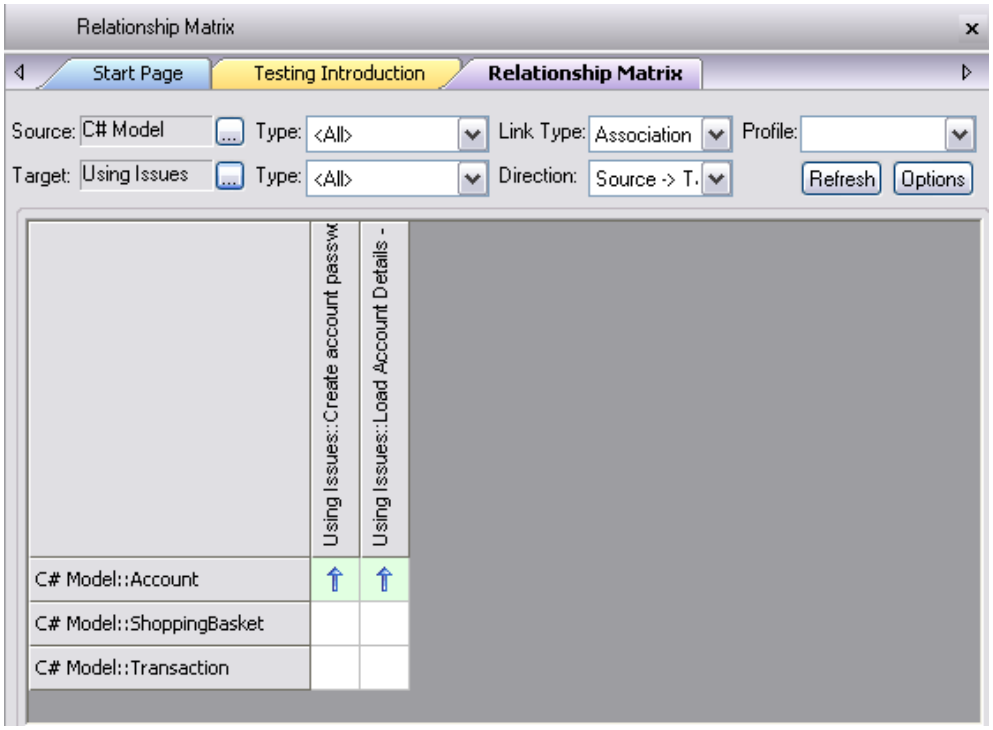


图 17 – 包含测试用例到类的元素关联。

层次结构视图（Hierarchy View， Ctrl+Shift+4)提供问题元素和类元素之间可跟踪能力的不同视图。

发布前组织测试

当开发多个系统版本时，应当分隔系统和测试模型元素，将测试分析保存到独立迭代分组的包，如 build 或版本发布。测试用例可以用包结构关联，或连接器。

下面是一个经典配置问题元素的实例，它在一组版本关联包中包含测试用例:

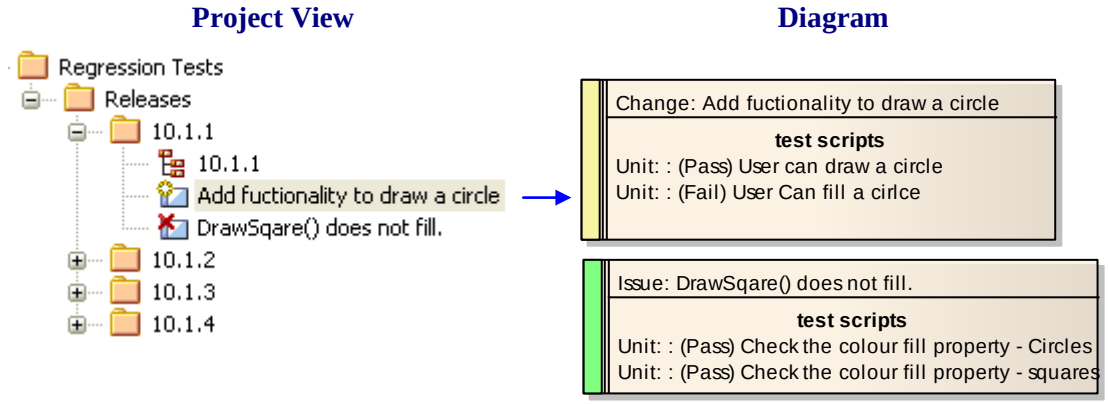


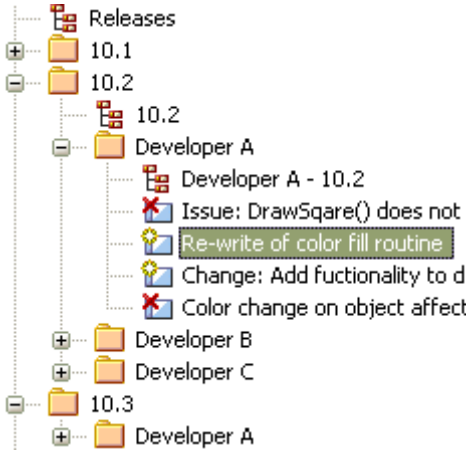
图 18 – 发布特定包含测试脚本的包

在需要连接器的地方，可以使用关系矩阵来设置(看 图 17)。

在一个版本发布中，有不同的方法可用来分组和安排修正任务。分组方式可根据开发人员，或不同的功能区，等等。这些分组是可变，可以用包，泳道和/或边界等等。

下面是版本发布/开发的实例。 此图为原始问题定义了一条泳道，为相关修正定义了另一条泳道。

Project View



Diagram

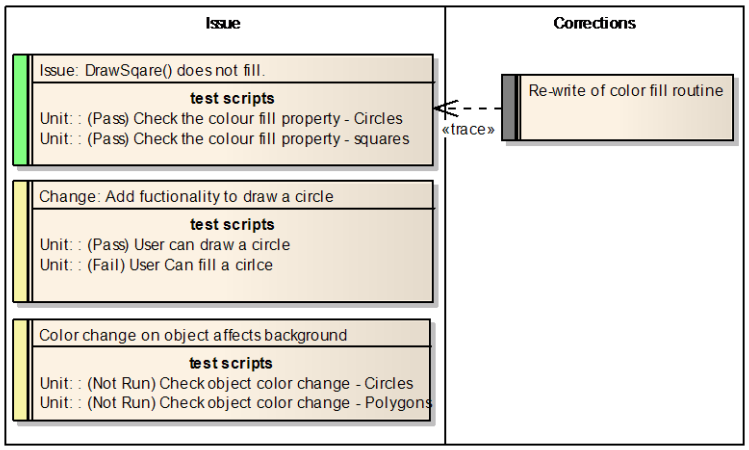
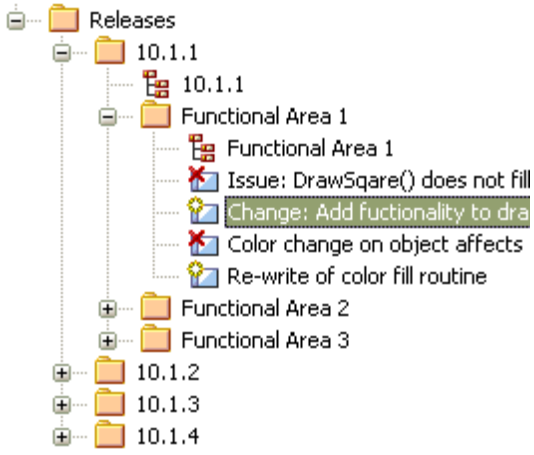


图 19 – 在“发布版本”中，被开发人员定义的测试问题

另外一种选择是将变化划分到各基于功能的区域 (图 20)，
其包含了人力资源的使用(如:开发和测试人员等等 – 看 图 21)。

Project View



Diagram

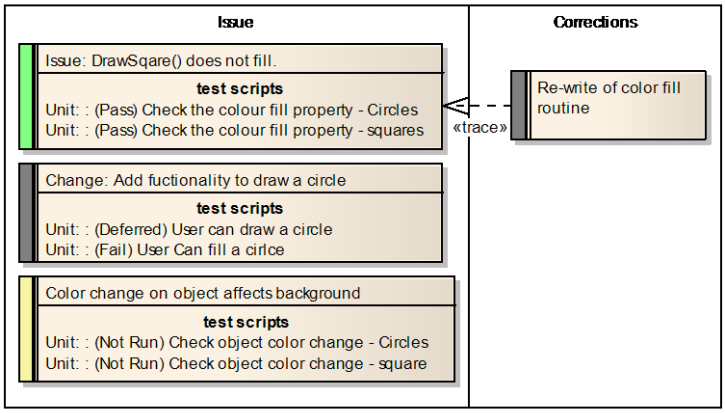


图 20 – 使用功能区对问题分组

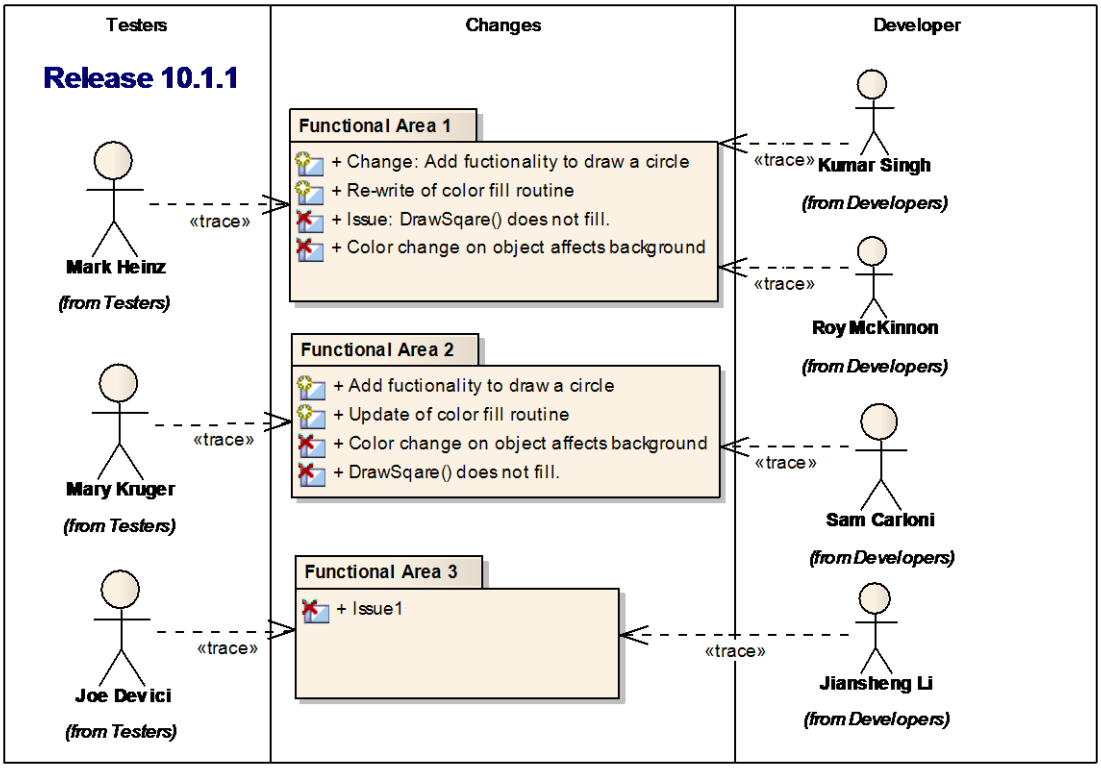


图 21 – 被分配不同包的开发人员和测试人员图

通常，测试人员和开发人员在发布版本主图被定义。

另一种方式是链接到代码修改时的分区，即这些类的实例或被测试的组件。这些可以象包含子类一样包含问题元素:

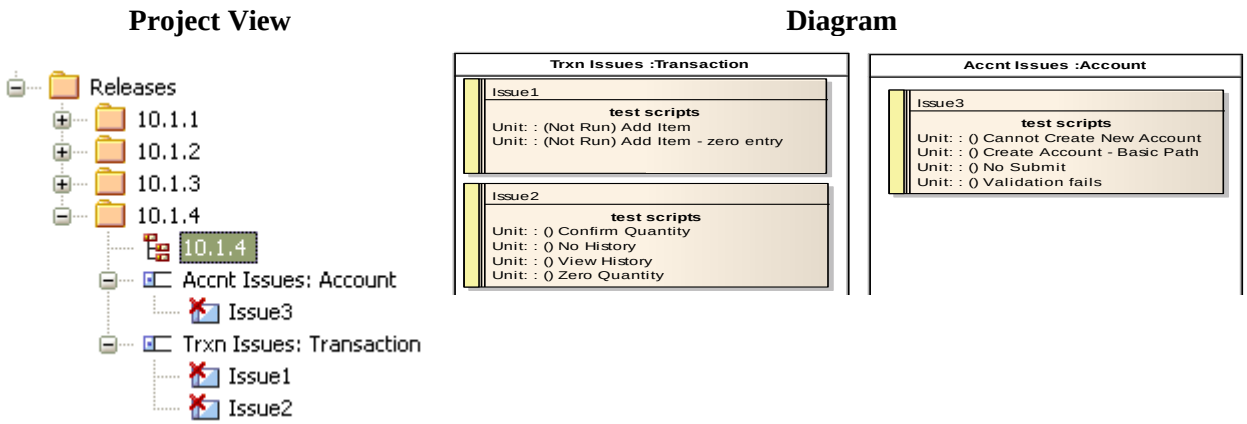


图 22 – 象类的“实例”那样使用分区来包含问题

> **提示: 分区可以设置为相关类的实例，使用**: Advanced | Instance Classifier, or Ctrl+L.

Enterprise Architect 提供了替代这种分组的许多方法。

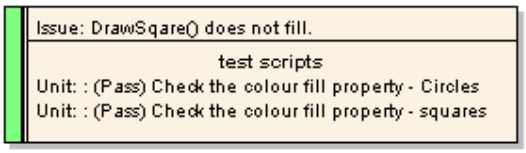
基于测试状态的颜色编码元素

我们可对常规元素（Custom Elements）如：问题，变化和需求，执行颜色编码来提供视觉线索，以指示它们的状态。对问题和变化进行颜色编码，需要:

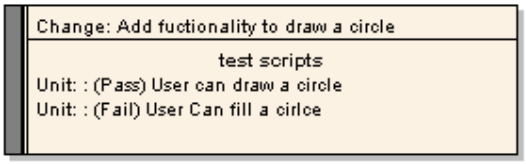
. 从菜单选择: Tools | Options | Objects

. 选中 [] Show Status colors on diagrams

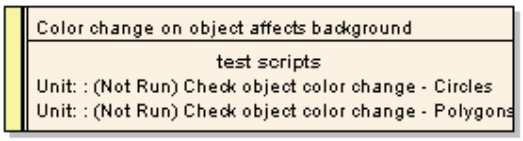
激活了颜色编码，你就可以选择一个元素来执行，在该元素属性设置里选择 Status。下面是一个问题的实例，从上往下依次设置为 'Validated'（确认的），'Implements'（执行的）和 'Proposed'（建议的）:



问题状态: Validated



问题状态: Implemented



问题状态: Proposed

上图清晰表达了一个问题进行成组测试之后的最后状态。 一旦完成测试，所有问题都将会显示为绿色 (确认) 。

提示: 状态类型和相关颜色可以在主菜单上使用用户定义:
Settings | General Types | Status Types

使用 Profiles 增加用户定义的测试区域

使用标签值，你能增加任意数量的属性，如汇报错误，用户的信箱，版本影响等等。

标签值可以一次性为任何元素定义，或者预定义为包含一个新元素。

任何元素的标签值可以成为一个单独的窗口，通过使用 Ctrl+Shift+6 访问 (或使用主菜单 View | Tagged Values)。

图 23 显示了一个一次性添加的标签值。

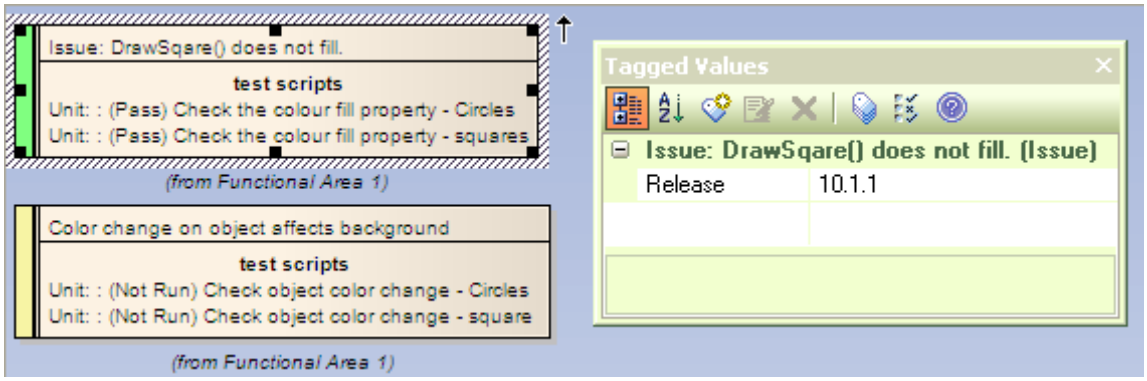


图 23 – 带标签值列表的问题允许属性分配

如何对需求加入一次性标签值

1. 右键单击需求
2. 选择 Add | Add Tagged Value…
3. 输入新属性名称。 (如 “Release”)
4. 为新属性输入值。 (如。 “10.1.4”)
5. 点击 OK 来设置属性。

为测试用例预定义标签值

Enterprise Architect 中的任何元素， 包括问题和变化元素，可为一个项目设置一组定义的扩展属性。元素属性可以使用 *UML Profile* 或 *Template 进行预定义*。

图 24 是为一个项目问题元素预设的一组标签值。

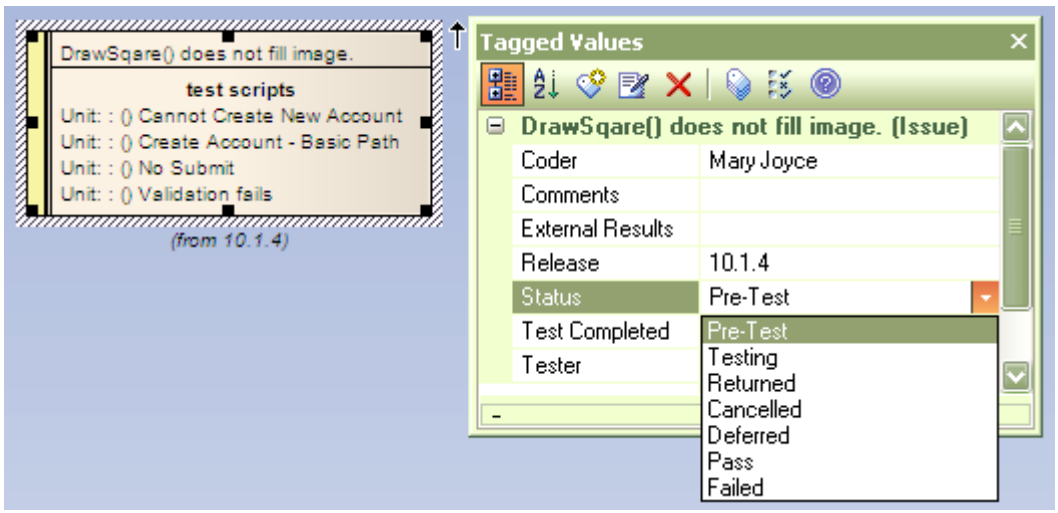


图 24 – 使用预设标签值

预设标签值类型能包含一定数量标准格式，如日期，时间，日历和下拉列表，等等。

扩展属性可以直接通过图中的元素进行查看。为图设置这种功能，需要右击图，从上下文菜单选择: Diagram Properties | Visible Compartments [✓] Tags。下图与图 24 中的元素相同， 查看这种模式:

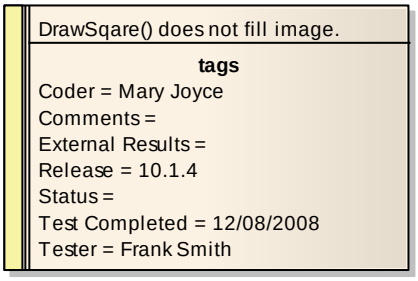


图 25 – 标签值可见的元素

关于更多使用标签值，扩展问题和变化元素属性信息 ，请看附录- Defining Attributes Using a Profile or a Template。

浏览和报告

除了打开测试窗口，我们还有几种工具来浏览测试脚本和报告测试脚本:

- 报告使用 :
  - 标准测试报告
  - RTF 报告生成器
  - 检索结果报告
  - HTML 报告

内置测试报告

Enterprise Architect 支持两种测试报告:

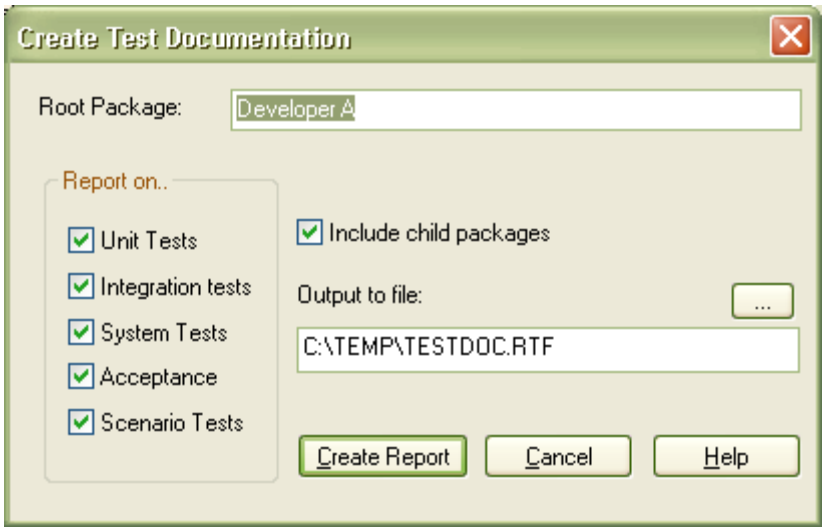
- 测试报告
- 测试细节

测试报告（Testing Report）

测试报告只允许过滤测试用例类型。使用此报告，请选择主菜单下:

Project | Documentation | Testing Report…

下图是创建测试报告的对话框;



其产生的文档格式如下:

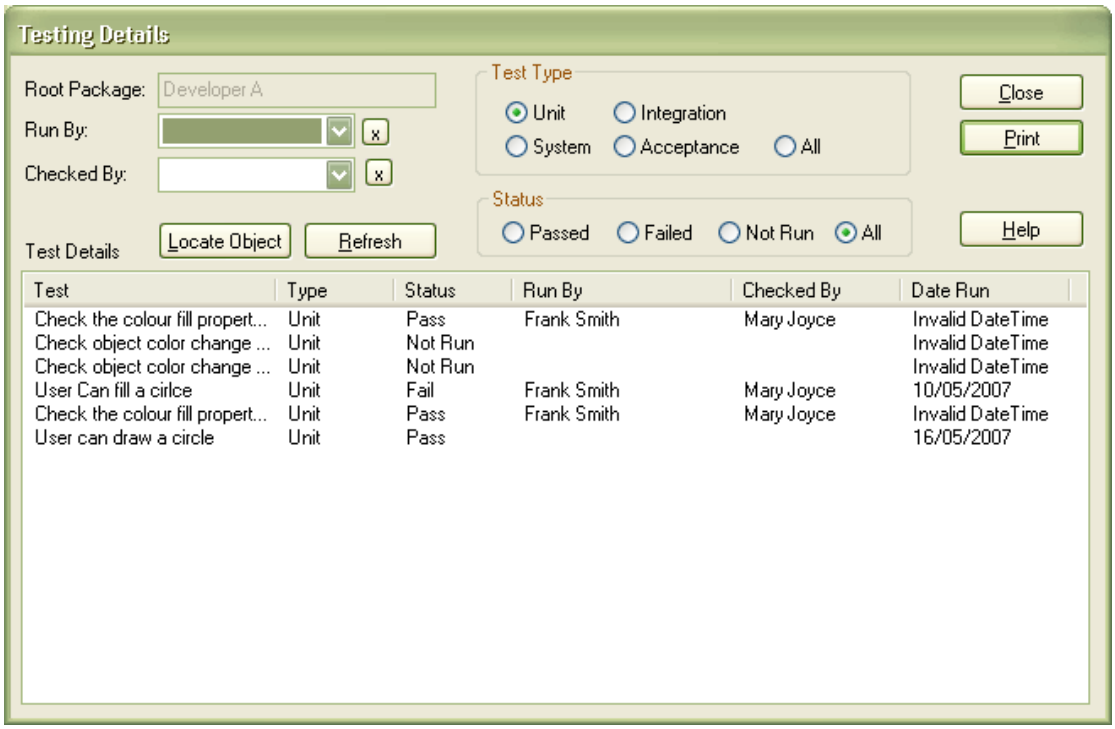
Unit Tests

Name	Object	Test Type	Current Status	Description	Input	Acceptance Criteria	Last Run	Result Details
Check the color fill property - squares	Issue: DrawSquare() does not fill.	Standard	Pass					
Check the color fill property - Circles	Issue: DrawSquare() does not fill.	Standard	Pass					
User Can fill a circle	Change: Add functionality to draw a circle	Standard	Fail		Select Point a, drag to point b.	A circle of diameter a-b is created.	10/05/2007	Check function that allows the user to fill a circle.
User can draw a circle	Change: Add functionality to draw a circle	Load	Pass				16/05/2007	
Check object color change - Polygons	Color change on object affects background		Not Run					
Check object color change - Circles	Color change on object affects background		Not Run					

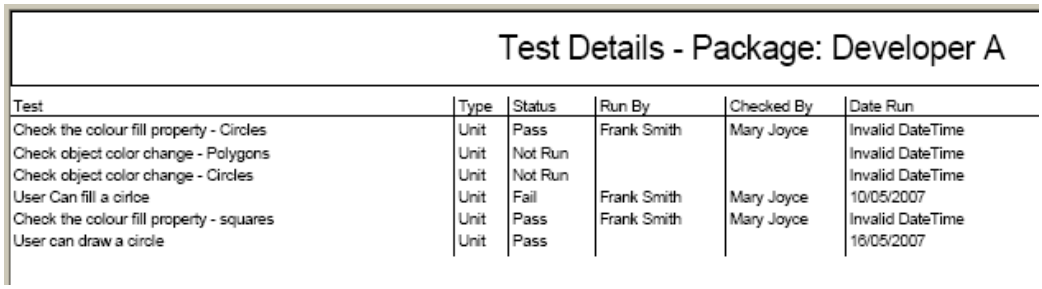
测试细节

测试细节报告允许设置过滤来产生严格的报告。获得此功能，可从主菜单选择: Project | Documentation | Testing Details…

下面对话框显示了使用过滤选项来创建测试报告;



这将生成如下格式的文档:



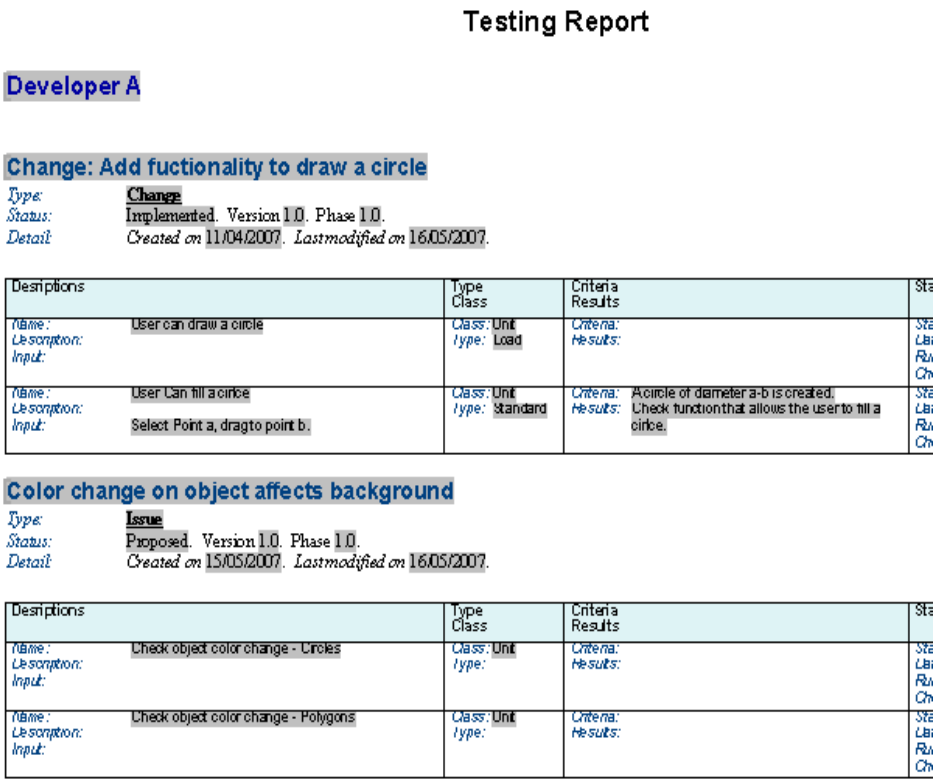
Test Details - Package: Developer A

Test	Type	Status	Run By	Checked By	Date Run
Check the colour fill property - Circles	Unit	Pass	Frank Smith	Mary Joyce	Invalid DateTime
Check object color change - Polygons	Unit	Not Run			Invalid DateTime
Check object color change - Circles	Unit	Not Run			Invalid DateTime
User Can fill a cirlce	Unit	Fail	Frank Smith	Mary Joyce	10/05/2007
Check the colour fill property - squares	Unit	Pass	Frank Smith	Mary Joyce	Invalid DateTime
User can draw a circle	Unit	Pass			16/05/2007

RTF 测试报告

RTF 模板报告生成器包含一个报告模板‘{测试模板}’，它可以做为一个初始模板，然后客户根据自己需要创建用户所定义的测试模板。

RTF 报告生成器可以从 Project | Documentation | Rich Text Format (RTF) Report 或 F8 获得。下面使用默认设置输出报告{测试模板};



Testing Report

Developer A

Change: Add fuctionality to draw a circle

Type: Change
Status: Implemented. Version 1.0. Phase 1.0.
Detail: Created on 11/04/2007. Last modified on 16/05/2007.

Descriptions	Type Class	Criteria Results	Sta
Name: User can draw a circle Description: Input:	Class: Unit Type: Load	Criteria: Results:	
Name: User Can fill a circle Description: Input: Select Point a, drag to point b.	Class: Unit Type: Standard	Criteria: A circle of diameter a-b is created. Results: Check function that allows the user to fill a circle.	

Color change on object affects background

Type: Issue
Status: Proposed. Version 1.0. Phase 1.0.
Detail: Created on 15/05/2007. Last modified on 16/05/2007.

Descriptions	Type Class	Criteria Results	Sta
Name: Check object color change - Circles Description: Input:	Class: Unit Type:	Criteria: Results:	
Name: Check object color change - Polygons Description: Input:	Class: Unit Type:	Criteria: Results:	

更多 RTF 格式模板信息，请看 Help | Creating Documents | RTF Documents 或 RTF 文档生成的白页:http://www.sparxsystems.com.au/resources/whitepapers/index.html

检索报告

Enterprise Architect 模型检索工具(Ctrl-F) 提供了两种方式定义检索过滤:Query Builder 和 SQL Search。

Query Builder 检索

标准 Query Builder 将根据元素(如一个元素有一次或多次检测失败)来返回结果。其输出结果可生成简单的报告。

预设检索有: 失败的内部测试（*Failed Internal Tests*）。*它检索那些包含有内部测试用例的元素，其检索条件是在任何常规测试用例字段，并且状态值显示为“失败”*（Fail）。 下图就是运行检索后显示的结果。

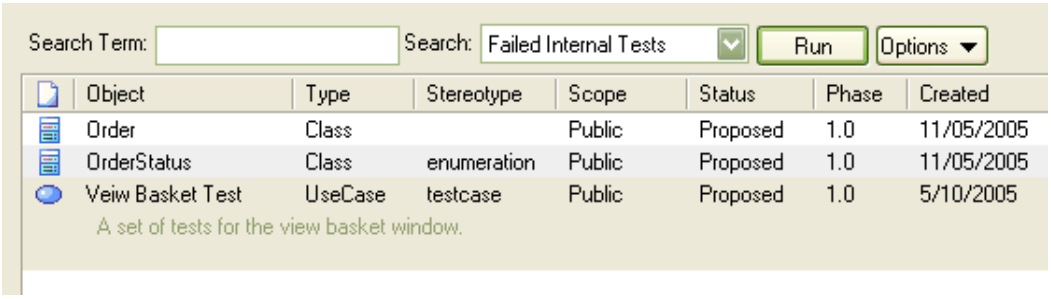


图 26 – 状态值为“Fail”的测试用例元素列表。

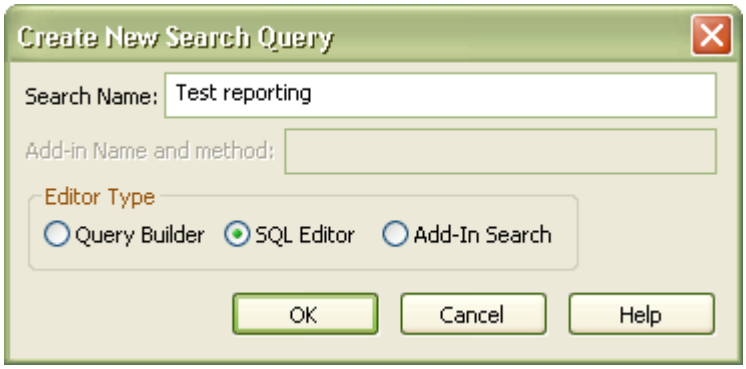
用户可以使用高级检索窗口（Advanced Search window）轻松创建检索。

SQL 编辑器检索

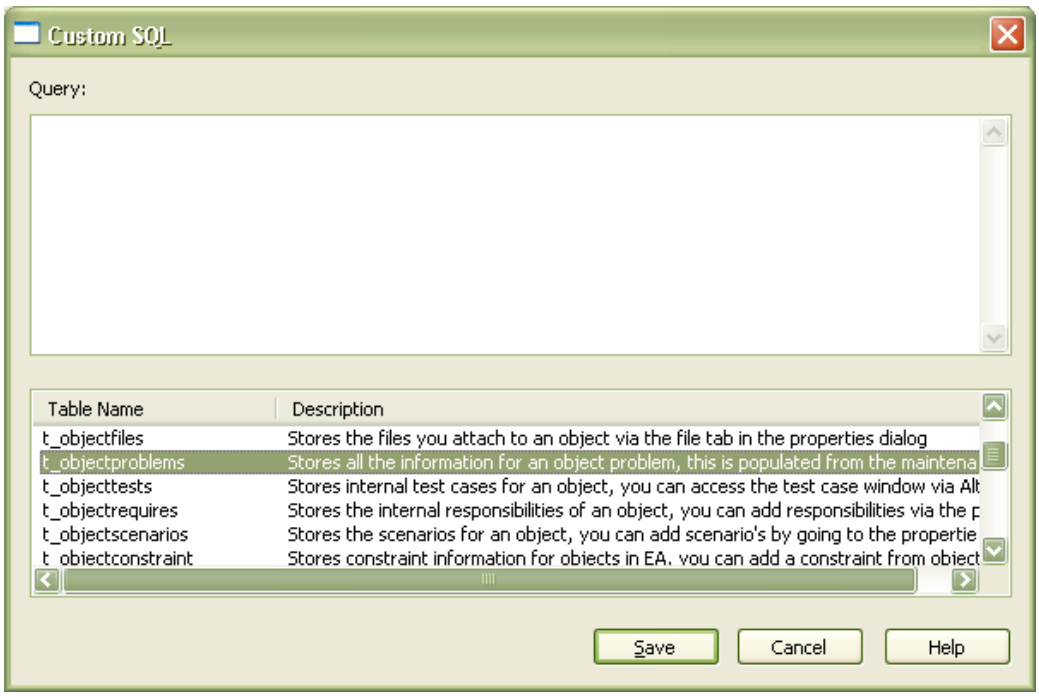
对于子元素级别的过滤和报告，检索工具有为子元素设置的过滤器(如一组元素 Element::TestCases)。 它需要使用对整个模型都有效的 SQL 语句，不能仅设置检索到模型层次结构的一部分。

下面是一个检索测试数据的实例:

1. 创建高级检索: Ctrl-F | Advanced | New Search
2. 在 Create New Search Query 对话框:
   - 输入: Search Name – 键入检索名称。
   - 选择: ⊙ SQL Editor。
   - 确认 OK。



这将打开 SQL 对话框



在 Query: 文本框中， 加入一个 SQL 语句:

- 例如: SELECT * FROM t_objecttests where status <> "Pass"
- 确认 Save
- 在 Find In Model 对话框，确认: Run Search

> **Tip:** 下面 SQL 语句提供了包名，元素名和测试细节:
>
> SELECT t_package.Name AS PackageName, t_object.Name AS ElementName, t_objecttests.Test, t_objecttests.TestClass, t_objecttests.TestType, t_objecttests.Status, t_objecttests.DateRun, t_objecttests.RunBy, t_objecttests.CheckBy, t_objecttests.Notes, t_package.Package_ID, t_object.Object_ID
> FROM (t_package INNER JOIN t_object ON t_package.Package_ID = t_object.Package_ID) INNER JOIN t_objecttests ON t_object.Object_ID = t_objecttests.Object_ID;

下面是上面 SQL 语句输出结果的部分视图:

Package...	ElementName	Test	Status	DateRun	RunBy	Notes
C# Model	OrderStatus	On new order - Ord...	Not Run	19/10/2005	Leo Burns	On creatin
C# Model	ShoppingBasket	Delete an Item	Not Run	27/10/2005	Leo Burns	On viewing
C# Model	ShoppingBasket	Alter Quantity	Not Run	27/10/2005	Leo Burns	The shopp
Changes	View Basket - ...	Update Cart Butto...	Deferred	19/10/2005	Leo Burns	This is to t
Changes	Create Accou...	Password Confirma...	Not Run	4/11/2005		1. Use cas
Changes	View Basket - ...	Alter Quantity	Not Run	5/10/2005	Leo Burns	The Alter C
Issues	Remove if qua...	Set to zero Quantity	Not Run	4/11/2005		In the shop
Issues	View basket to...	Check the Total C...	Not Run	4/11/2005	Ken Nielsen	The Total

对上述结果打印，需要单击它们，然后从上下文菜单选择打印选项。

SQL query builder 仅可以在整个模型内检索，使用 SQL 语句设置条件，然后会检索到一组明确的测试用例。

> **注意:类定义测试用例类**(单元，集成,等等). 它们在测试窗口有按顺序显示的数字定义. Unit = 1, Integration=2, 等等.

HTML 报告

HTML报告可以用EA生成。 HTML 报告生成器在Project | Documentation | HTML Report 或 Shift-F8。

下面是上述问题测试用例的数据显示:

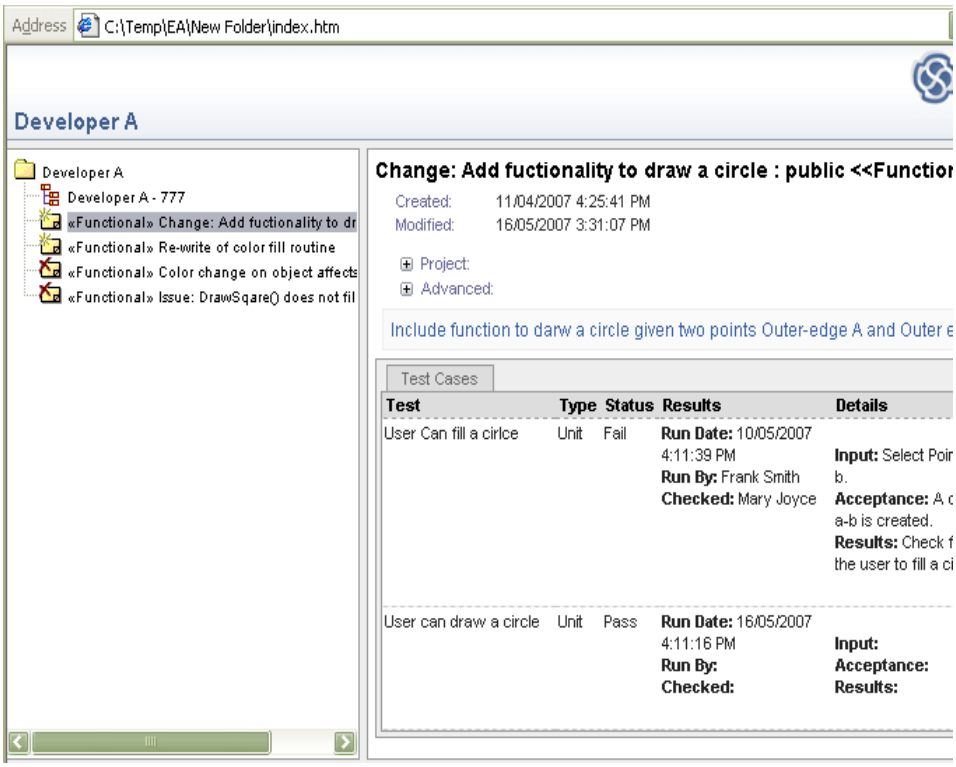


图 27 – 用浏览器查看生成的 HTML 视图

附录

从情况（Scenario），条件，需求和其它测试的导入

测试用例窗口支持从其它元素的数据导入。 这些元素包括脚本情况，条件，内部需求和其它测试用例。

这个功能的常规用途是为一个类或一个用例产生的问题而导入测试用例。下面是一组情况用例的实例:

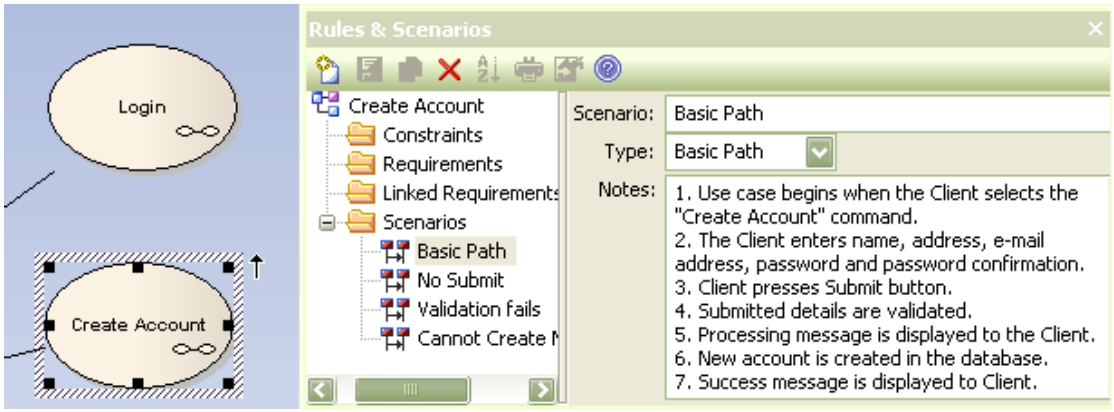


图 28 – 用例情况

开始导入过程:

1) 选择要导入数据的元素。
2) 打开测试窗口 (Alt+3)
3) 右键单击测试列表，从上下文菜单中选择导入类型。 下图显示了上下文菜单;

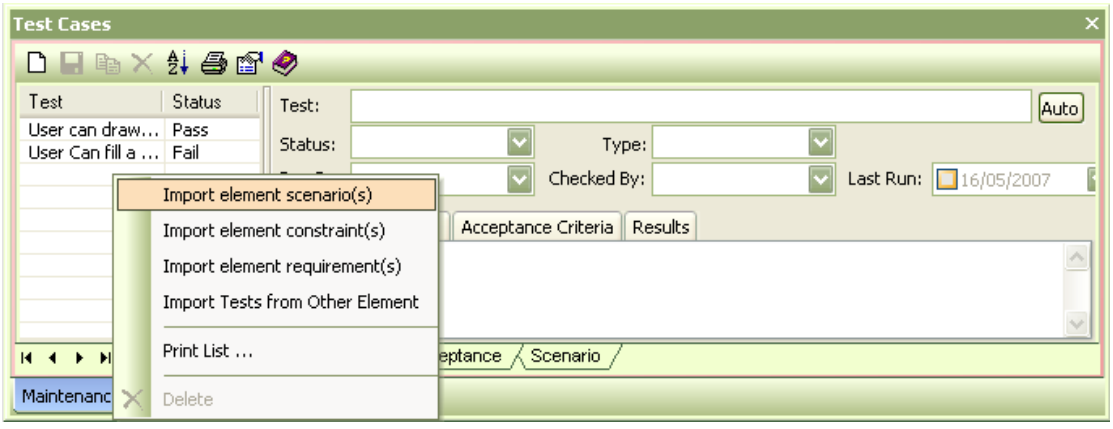
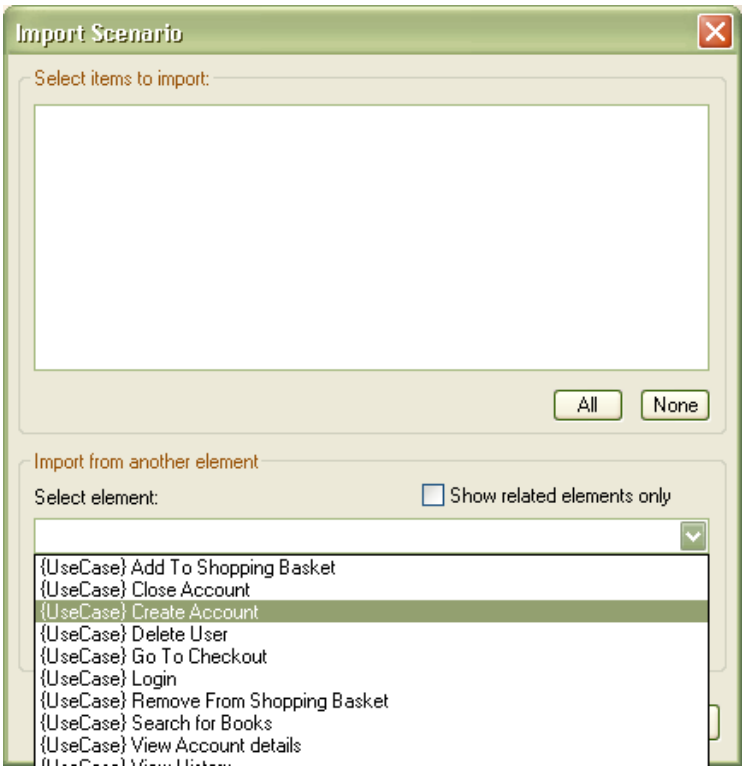
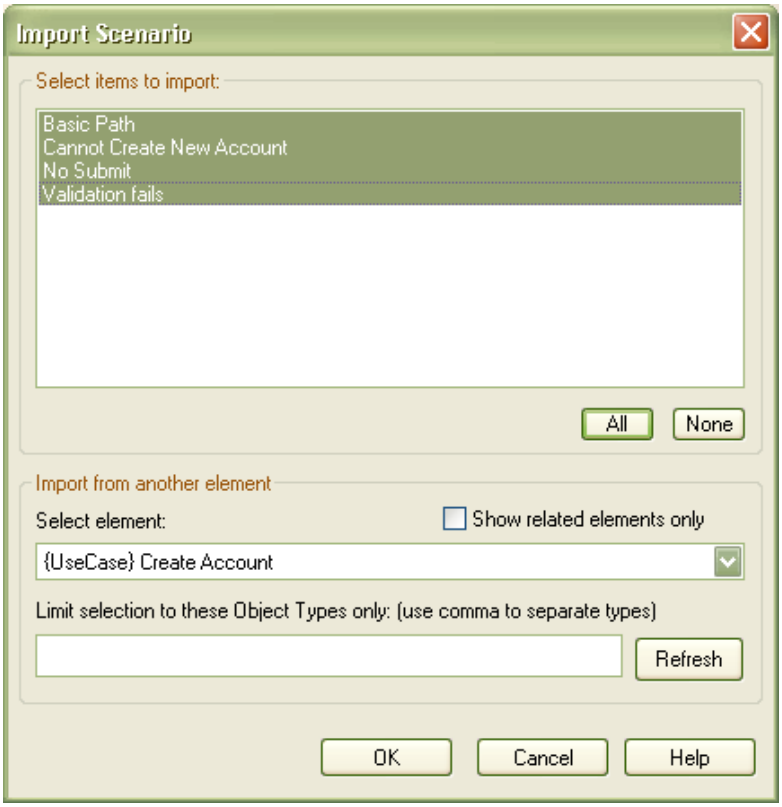


图 29 – 情况导入至测试用例的选项

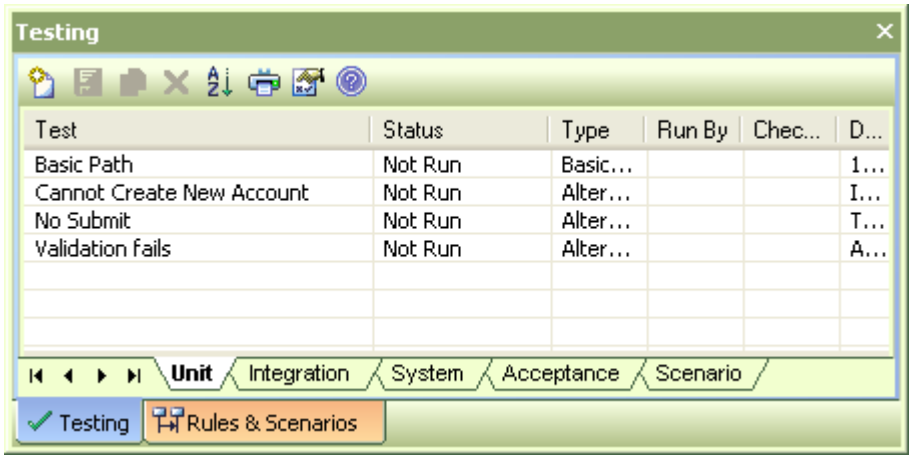
打开导入对话框。 使用 Select Element 下拉列表显示可能包含该情况的元素:



选中一个元素，适用的情况显示在Selected Items to Import中。我们可以选中一组输入，按钮 ：All – 选中所有输入导入。



选择 OK按钮，被选中的项目将会加入到当前测试用例中:



使用 Profile 定义元素属性

元素可以被预设来包含一组用户定义的属性。它们可用来记录用户的特定需求。尽管这些元素不能加入到 Element::Test Cases，但是它们可以加入到成组测试用例的属性中。例如，属性可用来存储数据，产品版本号，及错误报告关联的联系信息。

附加属性可以使用 *Model Template* 或 *Profile Definition 定义*。

简短比较下面的两个选项:

1. 使用模板是最简单的配置方式， 可是:
   a. 它预设了默认元素类型的特性（如:一个模板包中问题定义元素规定了所有为那个模型创建的未来问题元素）。
   b. 一个模板定义仅仅可以为一个模型生成一个给定元素类型。(只有问题类型可以被预设)。
2. Profile 设置相对复杂，它允许对一个给定类型提供多个扩展类型。

这两种方法都采用标签值定义。

> **注意:** 如果项目包含许多具有不同属性的同类元素，如：问题测试前（Issue-Pre-Test）和问题测试后（Issue-Post-Test），对此类项目 Profile 方法虽然设置复杂，但使用起来却更加方便。

关于使用模板的更多信息，请看: Help | Index: Template, Package, Settings。

定义标签值（Tagged Values）

标签值允许用户使用广泛的预定义或用户定义的数据类型来确定任何数量的字段。

在 Settings | UML | Tagged Value Types 选择设置标签值，然后打开下面的定义窗口。

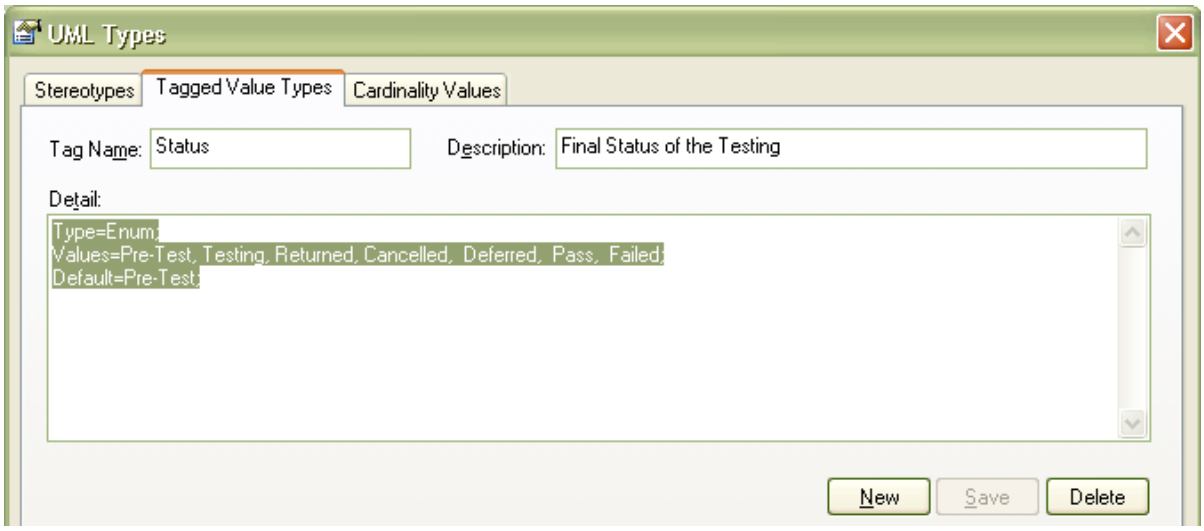


图 30 – 标签值定义窗口。

在上面例子中，选中的“Status”标签值，然后使用一个预设类型，它显示在可选项的下拉菜单中。给类型的细节包括:

Type=Enum;
Values=Pre-Test, Testing, Returned, Cancelled, Deferred, Pass, Failed;
Default=Pre-Test;

在标签值窗口浏览，可以看到一个下拉列表框:

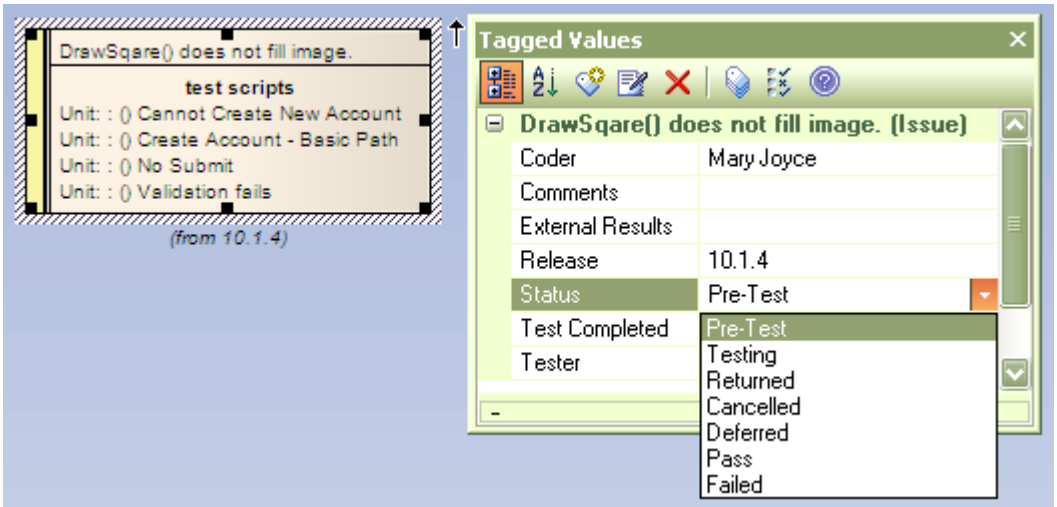


图 31 – 结合标签值的元素定义。

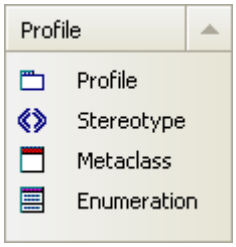
这里有许多可用的标准类型，如：数字和字符串类型，枚举列表（见上），日期，布尔数，和备忘录。关于 Enterprise Architect 中更多可用标准类型和列表的信息，请在帮助文档中使用‘Predefined Tagged Value Types’检索

使用 Profile 定义附加属性

Enterprise Architect 支持 Profiles 的创建。 Profiles 允许用户对标准元素使用标签值来定义一组扩展。请看下面的 Defining Tagged Values 。

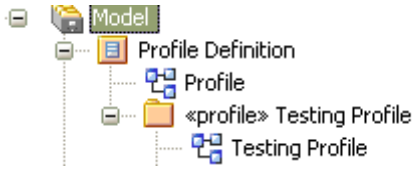
创建一个 Profile:

在项目浏览器中，设置一个特定包含Profile的包。
在该包下，为这个profile创建一个图
在 UML toolbox 打开这个 Profile



拖放 Profile 元素到图上。

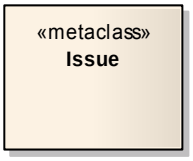
输入这个 profile 的名称，一个原型为«profile»的包*显示在* 图上，包的下面有子图。这个子图 (如下图中的'Testing Profile') 被用于向这个 profile 添加原型（stereotypes）。



添加一个新元素定义:

1. 打开系统创建并包含«profile»原型的包
2. 在profile上使用元类工具，创建一个元类元素（Metaclass element）。打开的对话框可以选择所创建的元素类型。
3. 为一个元素选择问题元素类型
4. 确认 OK。

元类 Metaclass 显示为:



Profile 包含定义标签值:

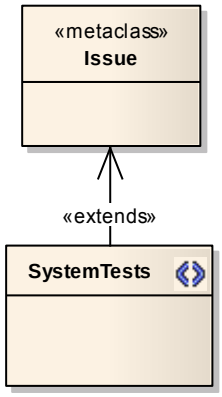
1. 从 Profiles 工具箱，拖动 Stereotype 工具到图上

2. 在属性对话框， 选中字段，在工具箱中为该元素键入名称， 如: ‘SystemTests’

3. 在属性对话框，确认OK

4. 从Profile 工具箱，单击扩展连接器 （Extension Connector）， 选中新的原型元 素 ‘Bug’ ，将鼠标拖至元类。这样在两个元素之间创建了一个扩展连接器。

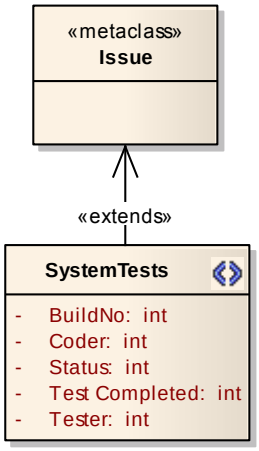
下面是一个被创建的元素和其连接的视图:



加入标签值

选择原型 (如: ‘SystemTests’)
从上下文菜单，选择 Attributes
在属性对话窗口 Attributes ，在文本区为每一个上面的，你想添加到‘SystemTests’元素中的标签值输入一个名称。

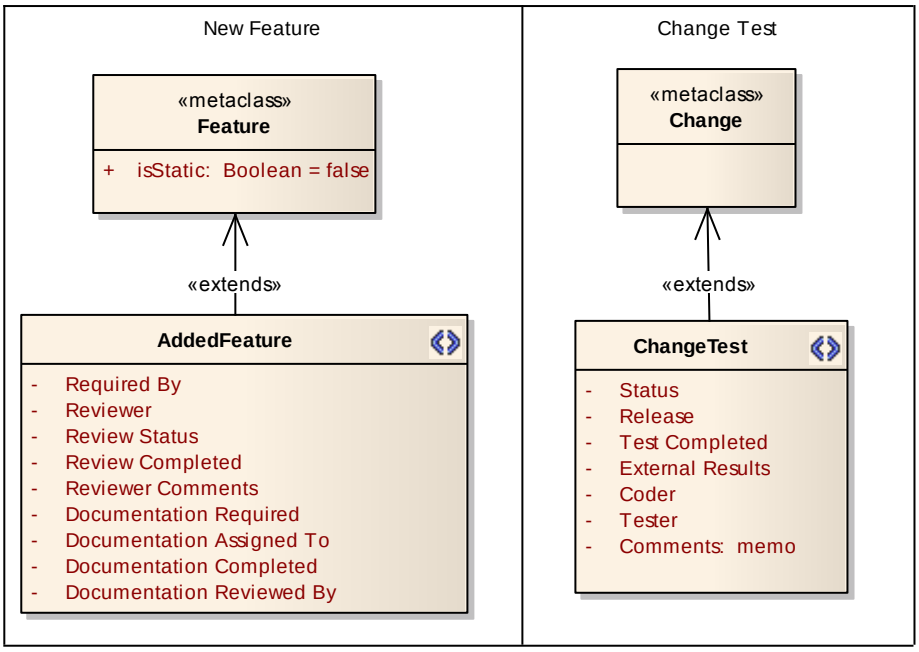
下面实例，是上面‘SystemTests’ 添加了许多定义的标签值
图 30 – 标签值定义窗口。



注意: 在创建简单字符串和布尔标签值的时候， 不需要在标签定义中定义，可以直接在‘SystemTests’ 元素的属性中输入.

一个 profile 可以设置任意数量的定义。每一个原型元素需要一个特定的名字。

下面是两个用户定义元素（泳道格式）的 Profile 实例， – 一个用于记录新功能，另一个用来记录相关测试细节的变化。 图 31 是测试变化的实例。



在工具箱中设置元素类型

1. 选择 «profile» 包
2. 右键点击，从上下文菜单选取 Save Package to UML Profile。
3. 保存 XMI 文件，设置文件名
4. 选择 Save
5. 打开 Resources 视窗
6. 从资源树状图，选择 UML Profiles
7. 右键单击，从上下文菜单，选择 Import Profile。

导入完毕，可使用下列步骤将其包含在工具箱中:

1. 选择最新导入的 profile
2. 右键单击在弹出的上下文菜单，选择: Show Profile in UML Toolbox。

带有新 profile 名的工具箱将加入到原工具箱中。现在你就可以创建你自己的常用功能，并且在任意包内使用工具箱拖放元素来更改元素。