HP ALM 最佳实践系列

面向 ALM 从业者

工作流最佳实践

文档发布日期：2016 年 5 月

法律声明

担保

HP 产品和服务的唯一担保已在此类产品和服务随附的明示担保声明中提出。此处的任何内容均不构成额外担保。HP 不会为此处出现的技术或编辑错误或遗漏承担任何责任。

此处所含信息如有变更，恕不另行通知。

受限权利声明

机密计算机软件。必须拥有 HP 授予的有效许可证，方可拥有、使用或复制本软件。按照 FAR 12.211 和 12.212，并根据供应商的标准商业许可的规定，商业计算机软件、计算机软件文档与商品技术数据授权给美国政府使用。

版权声明

商标声明

Microsoft® 和 Windows® 是 Microsoft Corporation 在美国注册的商标。

Oracle® 是 Oracle 和/或其附属公司在美国注册的商标。

文档更新

此文档的标题页包含以下标识信息：

- 软件版本号，用于指示软件版本。
  - 句点前面的数字标识主要发布号。
  - 句点后面的第一个数字标识次要发布号。
  - 句点后面的第二个数字标识次要的下一级发布号。
- 文档发布日期，该日期将在每次更新文档时更改。
- 软件发布日期，用于指示该版本软件的发布日期。

要检查是否有最新的更新，或者验证是否正在使用最新版本的文档，请访问：

https://softwaresupport.hpe.com/group/softwaresupport/search-result?doctype=manuals?keyword=

需要注册 HP Passport 才能登录此站点。要注册 HP Passport ID，请访问：

https://softwaresupport.hpe.com/

或单击“HP Passport”登录页面上的“New users - please register”链接。

此外，如果订阅了相应的产品支持服务，则还会收到更新的版本或新版本。有关详细信息，请与您的 HP 销售代表联系。

支持

请访问 HP 软件支持网站：

https://softwaresupport.hpe.com/

此网站提供了联系信息，以及有关 HP 软件提供的产品、服务和支持的详细信息。

HP 软件联机支持提供客户自助解决功能。通过该联机支持，可快速高效地访问用于管理业务的各种交互式技术支持工具。作为尊贵的支持客户，您可以通过该支持网站获得下列支持：

- 搜索感兴趣的知识文档
- 提交并跟踪支持案例和改进请求
- 下载软件修补程序
- 管理支持合同
- 查找 HP 支持联系人
- 查看有关可用服务的信息
- 参与其他软件客户的讨论
- 研究和注册软件培训

大多数提供支持的区域都要求您注册为 HP Passport 用户再登录，很多区域还要求用户提供支持合同。要查找有关支持访问级别的详细信息，请访问：

https://softwaresupport.hpe.com/web/softwaresupport/access-levels

要注册 HP passport ID，请访问：

https://softwaresupport.hpe.com/

目录

欢迎使用本指南

欢迎使用《HP 工作流最佳实践》指南。

本指南提供在各种组织中最佳实现工作流的概念、准则和实践示例。

关于工作流

新的技术、体系结构、业务趋势和最终用户预期正在改变应用程序的本质。因此，应用程序本身正在改变。新的工具和体系结构已经出现，加快和简化了复合应用程序、丰富的 Internet 应用程序和交互式 Web 2.0 服务的开发和交付。正在实现诸如敏捷开发这样的新流程，旨在更便于快速创建可适应的应用程序。

HP Application Lifecycle Management (ALM) 是一个涵盖软件存在期间所经历的所有阶段的完整解决方案。包含的 ALM 实践和解决方案使软件团队能够满足很高的业务预期和需求。HP ALM 套件可以为各种公司提供服务，根据行业分部、公司重点和流程、应用程序的数量及其类型等实现公司的特定需求。即使属于同一行业，处于类似环境，也没有两家公司以相同的方式运作。因此，以多种方式自定义 HP ALM 项目来满足您组织的业务流程需求的能力是每个实现的重要方面。

HP ALM 提供的最强大的工具之一便是内置的脚本编写功能，用于定义、控制和管理在项目内执行的业务流。ALM 项目管理员可以编写工作流脚本以自定义 HP ALM 用户界面及控制用户可以执行的操作。

本文档的目的是帮助 HP ALM 客户评估其当前自定义实践并使用 HP ALM 提供的高级功能成功构建和维护高效的工作流脚本。此过程的所有方面都已经使用最佳实践数据和来自各个来源（包括 HP 的操作系统管理员、HP 的专业服务组织、技术文档、来自业内专家的书籍以及来自许多客户测试组织的个人体验）的专业知识进行过研究。这些准则将帮助缩短初始创建时间并获得操作 HP ALM 的最大运营价值。

读者

本指南的目标读者是：

- 项目管理员
- 模板管理员
- 自定义专家

先决条件

为了使用本书，您应当熟悉软件开发生命周期 (SDLC) 的主要阶段，还应当熟悉实际 IT 组织中的业务流程。

HP ALM 的运营知识和管理特权是实现这些最佳实践的关键所在。

结构

本指南的组织方式如下：

- 工作流简介
- 工作流自定义准则
- 工作流事件
- 结论

反馈

如果您有任何问题或意见，或想要分享有价值的最佳实践信息，请将消息发送到以下电子邮件地址：

docteam@hpe.com

1 工作流简介

工作流的重要性

没有任何两家公司会以相同的方式运作，相反，它们往往采用不同的业务流程，身处不同的行业，运用迥异的发展方式，并使用不同的传统和现代技术，因此需要独特的 HP ALM 实现方式。在真实的 IT 组织中很难找到“一刀切”的理念。因此，每位 HP ALM 客户最终利用工作流脚本编写功能所提供的灵活性。

但是，工作流脚本对项目和整个站点的性能有重大影响。因此，开发符合逻辑、有条理的工作流代码极为重要。实现用于开发和维护工作流代码的合理流程也很关键。

以下步骤详述了正确的工作流步骤。

常见步骤

了解项目需求

创建或修改工作流代码之前，了解项目结构、项目的使用方法、组织过程和涉及的各个角色很重要。

要成功处理工作流代码，请先了解每个组或角色的需求，并确定考虑到所有组的工作流。标识共同点以创建满足全部需求的组合流程。

创建工作流需求文档

在开始编写代码之前，请先编写需求文档，此文档应包含计划的工作流自定义。此文档的目的是定义初始自定义。一旦工作流处于生产中，就必须定期更新此文档以包括实现的所有更改。

此文档应包含以下内容：

- 完整的工作流程
- 所需的工作流功能

  以下是一些示例：

  - 需求审阅过程 - 您的组织可能要求每个需求必须经过审阅和批准后才能链接到测试。
  - 用户或组可以根据其权限执行的操作。
  - 更改特定字段时发送电子邮件通知。
- 布局和格式

  例如，您可能需要确定新建缺陷时可用的字段，按用户组定义不同的字段集、不同选项卡中的字段位置等。

然后文档应获得所有干系人的批准。

编写工作流代码

工作流自定义文档获得批准后，开始在测试环境中编写工作流代码。有关测试环境的更多详细信息，请参见*测试环境* 部分。

测试工作流代码

邀请最终用户加入测试环境以验证您所做的更改。确保工作流实现符合他们的需求。

将工作流移动到生产

在客户端强制执行工作流。登录期间，会将自定义文件和工作流文件下载到您的本地客户端计算机上的以下目录下：`%temp%\TD_80`。

将工作流代码移动到生产环境后，需要注销并重新登录以访问最新的自定义和工作流修改。

管理工作流请求

要控制您的工作流代码，尤其是在涉及多名人员时加以控制，请定义一个系统来管理新的工作流请求。

此系统可用于跟踪请求，了解请求背后的业务需求、更改的影响、请求的重要性、请求范围（多少人需要此功能）等等。这样一个系统还应当具备发送有关请求进度的通知和状态的功能。

一个可能的解决方案可以是，针对管理新工作流请求的特定目的定义 ALM 项目。

2 工作流自定义准则

项目自定义选项

HP ALM 工作流脚本编写功能基于不同的自定义部分，如下所述。在编写代码之前，请在“项目自定义”中标识所有其他自定义需求。这些需求将用于实现工作流。

权限组

要防止项目受到未经授权的访问，ALM 支持将每个用户分配到一个或多个用户组。ALM 包含具有默认特权的预定义组。每个组都可以访问特定的 ALM 功能。

您可以基于现有组的特权创建新组。选择和您要创建的新用户组有相似访问特权的现有用户组会将您需要做的自定义级别最小化。

请注意，根据用户组设置权限不仅可用于强制执行可访问性，还可用于邮件操作、通知等。

HP **不建议**将一个用户分配到多个用户组。

项目列表

ALM 项目包含一组用于默认项目自定义的预定义列表，如缺陷状态和“是-否”列表。其中一些列表可进行自定义，以支持您组织中使用的各个流程。其他列表无法进行自定义，因为 ALM 依赖于其内部系统逻辑中的列表值。您也可以创建用户定义的列表，这些列表包含可输入到“查找列表”字段中的值。

项目实体

实体是任意 ALM 项目的构建块。实体包含用户为特定应用程序管理流程输入的数据，并且数据存储在表中。实体可以是任意工作对象，例如需求、测试、设计步骤、附件或缺陷。

通过“项目自定义”可设置 ALM 实体的特性和属性，例如，必填字段、只读和验证值。每个实体均包含称为*系统字段*的 ALM 默认字段。实体还可以包含您可创建的*用户字段*。用户字段的类型可以如下：

- 用户列表（项目中所有用户的列表）
- 列表
- 数字
- 日期
- 字符串

在“项目自定义”中，可以定义每个项目实体的属性，如哪些字段是用户必填字段，哪些字段的数据历史记录要记录。其中一些属性也可以使用工作流进行设置。建议使用“项目自定义”设置默认行为，仅在特殊情况下使用工作流脚本更改此行为。

每个实体的用户定义字段数的最大限制为 99。因此，HP **建议**所有干系人一起工作，以包括符合多数干系人的需求且短时间内不会变得冗余的字段。

需求类型

可以定义项目的需求类型。需求类型定义哪些字段可选，哪些用户定义的字段可用。这样您就能创建仅对特定类型需求可用的用户定义的字段了。

一般工作流规则

使用工作流代码可以进一步自定义项目。可以定义如下设置：

- 可见字段和必填字段
- 字段在对话框中的显示顺序
- 每个对话框选项卡中显示的字段
- 要分配给特定字段的列表
- 特定字段的默认值
- 字段值之间的依赖关系

可以根据用户组定义这些设置。

重要说明：

- 工作流代码将覆盖在“项目自定义”的特定自定义类别中定义的任何设置。
- 可以使用特定“项目自定义”页或通过工作流代码执行特定自定义，如定义用户组的转换规则或设置需求类型的字段属性。**建议**决定使用哪种方法进行此自定义，不要将两种方法结合起来使用。
- 使用自动邮件，ALM 允许您在指定的缺陷字段每次发生更改时，自动通过电子邮件通知用户。使用 `SendMail_AfterPost` 功能，工作流支持您定义所有项目实体的自动通知，添加复杂条件或将其用于特定用户或用户组。**建议**确保自动邮件和工作流功能之间不产生任何重叠。
- 工作流脚本可用于控制进入和退出模块的操作，以及限制模块访问。要阻止某用户组访问特定模块，HP **建议**使用“项目自定义”中的模块访问页。请勿使用工作流脚本阻止访问模块，因为此功能与“模块访问”功能冲突。

测试环境

在生产环境上实现工作流自定义之前，HP **建议**在反映您特定配置的测试环境中验证自定义功能。

什么是测试环境？

测试环境独立于生产环境，并准确反映后者。它模拟生产系统上安装的配置和应用程序，包括数据库服务器、软件和生产项目。通过在测试环境中测试工作流，可以获得所能取得结果的更好情形，同时识别和阻止任何对生产环境的潜在负面影响。

为什么设置测试环境？

工作流对项目的运行方式有重大影响。出于以下原因，HP **建议**设置测试环境：

- 在上线之前测试工作流是明智的。
- 如果工作流失败，由于测试环境独立于生产环境而不会造成实际损失。可能的生产损失是：数据丢失，工作流错误阻止功能，等等。
- 问题的及早识别和检测。
- 由计划流程的干系人验证。

调试工作流代码

可用多种不同方式调试工作流代码：

选项	工具功能	用于
通过 ALM 工作流添加 Msgbox	这是 ALM 工作流代码的内置功能。 可以将 `Msgbox`（消息框）添加到工作流中的任何位置，以便查看字段值、操作名称、代码中的位置，等等。 使用此方法时，建议对代码中的错误处理行进行注释： `On Error Resume Next`	调试代码中的特定位置
Dbgview	此免费软件监控器可调试您本地系统上的输出，并打印出所有应用程序事件。 若要只查看工作流事件，可以按 'workflow' 一词进行筛选。 请注意，此选项仅允许您查看内置的事件，而无法查看您添加的其他功能。 供应商：Microsoft (SysInternals) URL： *http://technet.microsoft.com/en-us/sysinternals/bb896647*	查看 ALM 所涉及的工作流过程

Microsoft Visual Studio	这是一种商业产品，允许您使用断点、检查变量等验证代码。 要将工作流代码附加到 Visual Studio，请在工作流代码中希望 Visual Studio 附加的点生成错误。为此，一种方法是调用不存在的子例程。安装 Visual Studio 后，脚本中的此类错误将弹出一个对话框，允许您附加到脚本。 请注意，要使用此选项，应对工作流代码中的 `On Error Resume Next` 进行注释。	• 调试代码中的特定位置 • 查看您已编写的代码流 • 观察运行时的代码

除上面的工具之外，您还可以实现自己的记录程序。HP **建议**您在使用自己的记录程序时，确保实现根据您的需求启用或禁用此记录程序的选项，以防止对性能造成负面影响。

工作流的可行操作和不可行操作

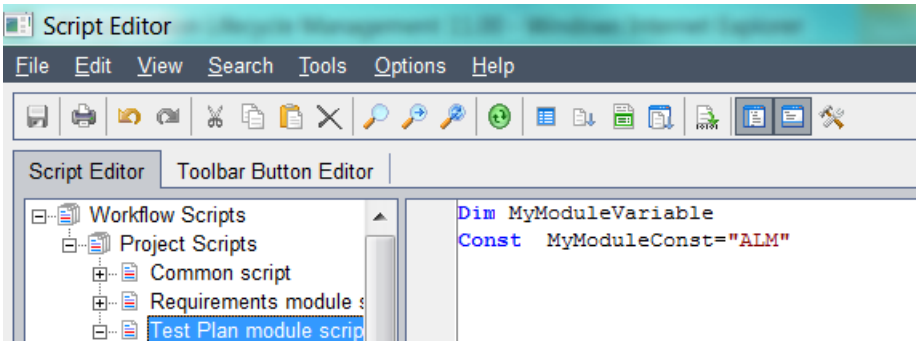
下面是可帮助您控制工作流的**建议的**常见实践。

需要做

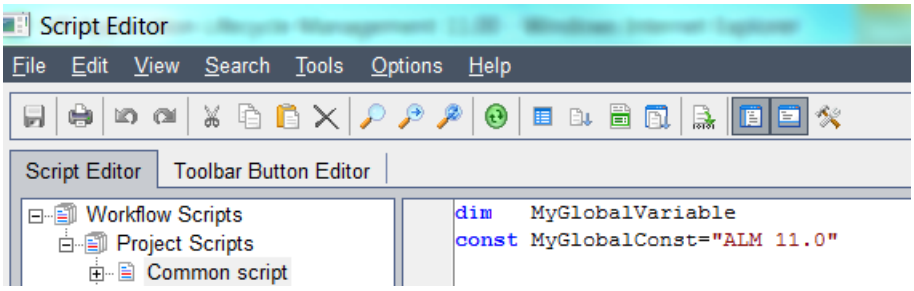
使用全局项

要在不同事件之间传递值，需要使用全局变量或全局常量。

变量可存在于“模块”级别。变量可用在“模块”事件中。



要在不同模块事件之间传递值，应在公用模块中定义全局变量。



优化代码

使用 VBScript 进行编程时，您可能会发现必须重复地对同一代码进行编程。这通常表示应使用函数来减少代码重复：

- 使用过程和函数代替冗余代码

  以下常见任务是独立函数或过程很好的主题的示例：

  - 设置字段属性
  - 在表单上设置字段
  - 设置列表依赖关系
  - 使用 OTA API 的任意过程

- 使用 Switch 语句代替重复的 ElseIf 语句

  此处的经验法则是，如果存在两个或更多 ElseIf 条件，则使用 Switch 语句。

改进代码可读性

除重用重复代码之外，也可以使用函数来改进代码可读性。下面是**建议的**良好实践，可帮助获得更好的代码可读性：

- 使用空行在逻辑上分隔相关代码块。
- 利用第一个变量声明中的引导（标头）注释和代码本身中最后一个声明的变量。
- 在所有注释前添加一空行。
- 使用两到四位的制表位缩进过程内的代码和注释。（默认情况下，Visual Basic 编辑器使用四位的制表位。）与空格一样，缩进用于在逻辑上组织代码并使其具有视觉吸引力。以下列表包含一些有关正确使用缩进以使代码更易读且更易维护的一般准则：

  - 将过程内的所有代码和注释缩进至少一个制表位。唯一不缩进的代码行是过程的开头和结尾以及与错误处理程序连接所使用的行标签。
  - 如果使用换行符设置过程的参数列表的格式，请使用制表符缩进各参数及其数据类型声明，以使其与列表中的第一个参数对齐。
  - 将声明的变量缩进一个制表位。一行上只声明一个变量。

- 将控制结构缩进至少一个制表位。如果一个控制结构嵌套在另一个控制结构内，则将嵌套的结构缩进一个制表位。将控制结构内的代码另外缩进一个制表位。
- 如果使用行继续符中断代码行，则将新行额外缩进一个制表位。这样将创建两行或更多行是一个整体的视觉提示。如果续行的下一行缩进的位数与续行相同，则为续行再添加一个制表位以将其与下一行区分开来。
- 将注释缩进到与注释引用的代码相同的级别。

为代码添加注释

可使用如下的注释模板记录代码：

```
'#############################################################
'#Date:
'#Designer:
'#Purpose:
'#############################################################
```

按名称访问

这是访问特定实体中的一个或多个字段的常见任务：

— 使用语句 <实体>_Fields.Field(“<字段名>”) 可按名称访问字段。

— 在 <实体>_Fields.FieldById(i) 上使用循环可访问集合中的所有字段。例如，这可用于重置字段的顺序。

这两种方法都允许使用*当前实体*的字段。

可通过以下方式定义当前实体：

— 当前实体

触发其当前事件的实体。几乎每个事件都指向可检索其字段的实体类型。例如，在 Defects_Bug_ 事件中，只能操作 Bug_Fields；在 TestPlan_DesignStep_ 事件中，只能访问 DesignStep_Fields。

— 聚焦项

在事件所定义的集合的所有实体中，只能检索当前聚焦实体的字段。例如，光标所在的测试，或手动运行器中的当前运行。

要检索相同或其他对象类型的其他对象的字段，请使用 OTA API。

在设置新布局之前重置

在设置*特定字段* 的布局（如 PageNo 和 ViewOrder）之前，请确保重置*所有字段* 的布局。

由于字段具有某种默认的预定义顺序，因此在定义新的自定义顺序之前重置此顺序很重要。否则，您得到的字段不会是顺序相同的所需字段，从而导致其余所有字段的顺序未知。

请参见本书后面的*工作流示例* 。

备份代码

HP **建议**执行工作流代码的定期备份。可以复制全部或部分工作流脚本并粘贴到外部文本文件，以保存在文件系统中。

使用全局常量与字段名

为提高代码可读性，请使用全局常量代替字段名。

您需要为每个字段名声明全局常量 - 请参见上面的*使用全局项* 部分。

示例：

```
If Bug_Fields.Field(Bug_Status).Value=” Closed”  then
       Bug_Fields.Field(Bug_Closed_In_Version).isRequired = true
End if
```

清理对象

确保在对象的范围结束时清理对象。对于每个对象实例，有必要清理未使用的对象。这样能提高 ALM 性能并帮助防止错误。

```
set myTDConnection = TDConnection
set myTDConnection = nothing
```

标准化

HP **建议**对所有项目应用标准化。负责多个项目的 ALM 项目管理员应在所有项目中使用公共约定。这样可改进代码的可读性和可维护性并支持跨项目的功能。

错误处理

影响工作流脚本质量的最重要的因素之一便是错误处理的正确实现。

通常，下面是一些**建议的**简单错误处理实践，可帮助更好地控制应用程序行为：

- 在每个过程和函数的开头使用 `On Error Resume Next` 语句
- 在每个过程或函数的结尾使用 `On Error GoTo 0`
- 在一些标准消息框中向用户显示错误

  应向每个工作流脚本（缺陷、测试计划等）添加一次代码，如下面的 `PrintError` 函数。

  出于此目的，可使用包含运行时错误的相关信息的 Visual Basic `Err` 对象。

```
Sub GetBug1
  On Error Resume Next
  Set Bug1 = TDConnection.BugFactory.item(1)
  PrintError(“GetBug1”)
End Sub
Sub PrintError(strFunctionName)
  If Err.Number <> 0 Then
    MsgBox "Error #" & Err.Number & ":" & Err.Description, _
    vbOKOnly, "Workflow Error in Function " & strFunctionName
  End If
End Sub
```

不可行操作

不要复制

不要在 `Entity_CanChange` 事件和 `EntityChange` 事件中复制您的代码。

可在 `Entity_canChange` 事件或 `Entity_Change` 事件中编写基于字段更改的代码。请确保您了解这些事件之间的差异，并在您的工作流代码中遵循下面的简单规则：

- 应在 `Entity_CanChange` 事件中编写处理权限（允许更改状态）的代码
- 应在 `EntityChange` 事件中编写处理依赖关系值或依赖关系列表的代码

不要放置太多代码

CanLogin 事件或 EnterModule 事件中的代码太多会影响性能。一个常见错误便是在 CanLogin 事件期间更新实体。

不要在可见之前设置其他属性

在设置字段的 IsRequired 或 IsReadOnly 属性之前，先设置 IsVisible 属性。

为屏幕上不可见的字段设置强制或只读属性没有意义，ALM 将忽略。因此，设置其中任何属性之前确保字段是可见的很重要。

请参见本书后面的*工作流示例*。

不要混淆工作流代码

尽管一个 ALM 项目可能要用几年，但是工作流代码仍应当清晰可扩展。

- 首选 Select 而非 If
- 使用函数

不要更新参数

不要更新工作流函数中的参数值。具体而言，不要更新 Entity_CanChange 事件中的 NewValue 参数。

不要在新项中修改

不建议在新的实体事件中修改操作，因为新的实体事件是在*创建*实体时调用的，而不是在新的实体对话框打开时调用。

常见用例是，当用户第二次打开新的实体对话框时。将不调用新的实体事件，因为在对话框第一次打开时已经创建实体。

不要在 MoveTo 中分配

不要向 MoveTo 事件中的字段分配值。此实践不好，因为 MoveTo 事件将锁定实体。

不要修改 After Post

不要对 After_Post 事件执行任何对象修改。

不要多次使用 API 调用

不要多次使用 API 调用，因为调用 API 将增加服务器与数据库之间的通信级别。每次调用 API 都将引起网络通信，从而导致脚本的运行时间变长。

例如，若要处理 100 个实体，请尝试在一个筛选器中获取所有实体，而不是单独检索每一个实体。

在工作流中使用 API

使用 ALM API 进行修改

ALM API 在用户界面（或使用用户界面的任何应用程序）与服务器逻辑之间提供一分离层。当使用 API 调用时，HP **建议**遵循以下规则：

- 使用预定义的 `TDConnection` 对象获取当前会话

  从 Visual Basic 或 Excel 之类的外部应用程序使用 OTA API 时，使用 OTA 的任意应用程序的第一步便是，创建 `TDConnection` 对象的实例，初始化与服务器的连接，并连接到数据库。但是，工作流中包含预定义的 `TDConnection` 对象（在这种情况下，`TDConnection` 不仅是类名，还是包含 `TDConnection` 实例的全局变量的名称），此对象指向当前用户所在的同一会话。这意味着始终可以从工作流中的任何位置访问所有 ALM 集合和对象。

- 由于以下潜在问题，请避免使用 `Command` 对象直接更新数据库：
  - 绕过服务器机制将导致：
    - 实体锁定
    - 历史记录丢失
    - 不需要的其他功能（设置邮件）
  - 高查询维护
  - 可导致数据损坏或不一致
- 使用 OTA 中可用的邮件发送方法将自定义邮件发送给用户

  OTA 允许访问 ALM 邮件，从而允许您：
  - 创建无法使用 ALM 的自动通知系统实现的自定义条件
  - 更改电子邮件的主题或文本

— 将电子邮件发送到特定的 ALM 组或用户

— 像自动邮件通知一样，从“admin”以外的特定用户发送电子邮件

可从诸如“缺陷”、“测试”等任何 ALM 对象或直接从 TDConnection 对象使用邮件发送方法。从 TestDirector 对象使用 Mail 方法，可发送包含该对象和自定义的主题和文本的电子邮件。

示例 —— 使用工作流对象，而不使用 SQL 语句

不要使用下面的命令：

```
Com.CommandText = "UPDATE TESTCYCL SET TC_TESTER_NAME =
'" & Cstr(ASSIGNED_TESTER) & "' " &_
"Where TC_CYCLE_ID = " & iTestSetId & " and TC_TESTER_NAME is
NULL"
Set UpdateRecSet = Com.Execute
```

改用下面的代码段：

```
Set tstestF = currentTestSet.tstestFactory
Set tsFilter = tstestF.Filter
tsFilter("TC_TESTER_NAME")= ""
Set tsTestList = filter.newList
For each tsTest in tsTestList
    tsTest.Field("TC_TESTER_NAME") = "admin"
Next
```

最大限度减少客户端上的活动

从服务器提取数据时，**建议**在服务器端而非在客户端筛选信息。在客户端上筛选的性能开销非常高。加载的记录太多也会影响服务器的性能。

示例 —— 使用历史记录时采用筛选

使用 Command 对象检查 HISTORY 表时，应通过实现 WHERE 条件在 SQL 中创建一个筛选，以便不会在客户端上产生所有记录集。

示例 —— 计算设计步骤

设计步骤具有一个保存步骤持续时间的用户定义字段。我们的目标是获取持续时间长于 30 分钟的设计步骤的数目。

下面的代码表示的实践不好：

```
For Each Test In TestLists
    Set DesStepF = Test.DesignStepFactory
    Set DSList = DesStepF.NewList("")
    For Each DStep In DSList
        If DStep.Field(“DS_USER_01”)>30 Then
          HowManyFound = HowManyFound + 1
        End If
    Next
Next
```

请改为尝试使用以下代码块：

```
Set TestF = TDConnection.TestFactory
Set TestList = TestF.NewList("")
For Each Test In TestList
  Set DesStepF = Test.DesignStepFactory
  Set DSList = DesStepF.NewList("select * from DESSTEPS WHERE
DS_USER_01>30 ")
  HowManyFound = HowManyFound + DSList.count
Next
```

3 工作流事件

在 ALM 用户会话中，当用户启动各种操作时，ALM 将触发事件过程。可以在这些过程中放置代码，来自定义相关用户操作的执行。事件过程可以是函数或子例程。

常规

下面给出了 HP ALM 中使用的事件函数和子例程以及命名约定的一些常规背景。

事件函数

这些过程由 ALM 触发，以检查是否应当执行用户的操作。可在这些函数中放置代码以确定 ALM 能否执行用户的请求。如果代码返回值 False，则 ALM 不继续操作。

例如，用户在*添加缺陷*对话框上单击“提交”按钮时，在将该缺陷发布到服务器上的数据库之前，ALM 调用函数 `Bug_CanPost`。可以将代码添加到 `Bug_CanPost` 函数以控制 ALM 是否发布缺陷。例如，可确保用户不能在没有添加注释的情况下拒绝缺陷。

事件子例程

事件发生时，将触发这些过程来执行操作。

例如，用户打开*添加缺陷*对话框时，ALM 调用子例程 `Bug_New`。可以将代码添加到 `Bug_New` 子例程，以执行用户打开该对话框时应当执行的操作。例如，如果用户不在 `QA Tester` 用户组中，可以将 ***Detection Mode*** 字段的值改为 `BTW`。

命名约定

HP ALM 中事件过程的命名约定如下：

<实体>_<事件>

注：某些事件过程名称（如 `GetDetailsPageName`）不包括实体名称。

实体

实体	描述
分析项	报告和图数据
分析项文件夹	报告和图文件夹数据
缺陷	缺陷数据
业务模型	业务模型数据
业务模型活动	业务模型活动数据
业务模型文件夹	业务模型文件夹数据
业务模型路径	业务模型路径数据
组件	业务组件数据
组件文件夹	业务组件文件夹数据
组件步骤	业务组件步骤数据
控制面板文件夹	控制面板文件夹数据
控制面板页面	控制面板页面数据
设计步骤	设计步骤数据
资源	测试资源数据
资源文件夹	测试资源文件夹数据
运行	测试运行数据
步骤	测试运行步骤数据
测试集	测试集数据
测试集测试	测试实例数据

工作流可能也支持某些扩展。

常用模块

CanLogin

触发此事件以检查指定用户可否登录指定项目。用于允许或禁止登录项目。此事件可用于更新用户。

主题	描述
语法	CanLogin(DomainName, ProjectName, UserName) 其中 DomainName 是域名，ProjectName 是项目名称，UserName 是用户名
类型	函数
返回	True 或 False
可用性	CanLogin（所有模块）

示例 —— 登录时通知用户

```
Function CanLogin(DomainName, ProjectName, UserName)
CanLogin = false
Call MsgBox("Hi " & User.UserName & “,” _
            & vbCrLf & "" _
            & vbCrLf & "Your project " & TDConnection.ProjectName
& " was upgraded to ALM 11.0" _
            & vbCrLf & "" _
            & vbCrLf & "The Project was moved to the server :
http://ALM:port/qcbin" _
            & vbCrLf & "" _
            & vbCrLf & "QC Admin" _
            , vbExclamation, "Important Message")
Exit function
End function
```

CanLogout

触发此事件以检查当前用户可否从当前项目注销。

主题	描述
语法	CanLogout
类型	函数
返回	True 或 False
可用性	CanLogout（所有模块）

示例 —— 注销前通知用户

```
Function CanLogout

Call MsgBox("Hi " & User.UserName & “,” _

            & vbCrLf & "" _

            & vbCrLf & "Your project " & TDConnection.ProjectName
& " will be upgraded to ALM 11.0  on 01/01" _

            & vbCrLf & "" _

            & vbCrLf & "The Project will be moved to the server:
http://ALM:port/qcbin" _

            & vbCrLf & "" _

            & vbCrLf & "QC Admin" _

            , vbExclamation, "Important Message")

End Function
```

ActionCanExecute

在 ALM 执行用户启动的操作之前触发此事件，以检查操作可否执行。可以将代码添加到此事件过程，以便在用户启动特定操作时执行操作，或在特定情况下禁止执行操作。

主题	描述
语法	ActionCanExecute(ActionName) 其中 ActionName 是用户已启动的操作 操作的格式是上下文.操作 用户定义的操作以前缀 UserDefinedActions 开始
类型	函数
返回	True 或 False
可用性	ActionCanExecute（所有模块）

示例 - 阻止删除缺陷

```
Function ActionCanExecute(ActionName)
On Error Resume Next
if ActionName= "Defects.DeleteDefect" then
       if Bug_Fields.Field("BG_STATUS").value  =” Closed”  then
              ActionCanExecute = true
       Else
Msgbox  “You don’t have enough credentials to perform Delete”
              ActionCanExecute = false
              Exit function
       End if
End if
 ‘……
End function
```

示例 - 查找操作名称

```
Function ActionCanExecute(ActionName)
On Error Resume Next
if user.Username=” Project_admin”  then
       MsgBox actionname
End if
End function
```

EnterModule

用户切换到 ALM 模块时，触发此事件。

可以将代码添加到此事件过程,使用户一切换到指定模块就执行操作。

主题	描述
语法	EnterModule
类型	子例程
返回	
可用性	EnterModule（所有模块）

示例 —— 隐藏按钮

```
Sub EnterModule
'hides the button Send Mail in the Defects grid
  On Error Resume Next
    Actions.action("Defects.SendByEmail").Visible= false
  On Error GoTo 0
End Sub

Sub DialogBox(DialogBoxName, IsOpen)
'hides the button Send Mail in the Defect details Dialog
'Use ActiveModule and ActiveDialogName to get the current context
On Error Resume Next
```

```
if (DialogBoxName="actBugDetails" or DialogBoxName="Details" or
DialogBoxName="Bug Details") and IsOpen=true  then
       Actions.action("BugDetails.SendByEmail").Visible= false
End if
  On Error GoTo 0
End Sub
```

ExitModule

用户退出指定模块时，触发此事件。

主题	描述
语法	ExitModule
类型	子例程
返回	
可用性	ExitModule（所有模块）

DialogBox

对话框打开或关闭时触发此事件。

主题	描述
语法	DialogBox(DialogBoxName, IsOpen) 其中 DialogBoxName 是对话框的名称，IsOpen 表示对话框是否打开
类型	子例程
返回	
可用性	DialogBox（所有模块）

示例 - 标识视图类型

此示例可帮助标识当前视图类型 - 网格、详细信息、新实体。此类型在名为 DialogIsOpen 的常用模块中的全局变量内进行维护。

```
Sub DialogBox(DialogBoxName, IsOpen)
On error resume next
If DialogBoxName="New Bug" and IsOpen=true then
         DialogIsOpen = "NEW"
             Else
                    DialogIsOpen =” OTHER”  ‘Details Or Grid
End if
 On Error GoTo 0
End sub
```

CanCustomize

用户试图打开“自定义”窗口时触发此事件，以检查用户可否自定义指定项目。

主题	描述
语法	CanCustomize(DomainName, ProjectName, UserName) 其中 DomainName 是域名，ProjectName 是项目名称，UserName 是用户名
类型	函数
返回	True 或 False
可用性	CanCustomize（所有模块）

示例 —— 禁止进入自定义

此示例可阻止未经授权的用户进入自定义。

```
Function CanCustomize(DomainName, ProjectName, UserName)

on error resume next

if  User.IsInGroup("TDAdmin”) then

  CanCustomize = true

else

    MsgBox User.FullName & vbcrlf & vbcrlf & "You don't have
enough privileges" &vbcrlf & vbcrlf &"Please Open a SR in Project
Center Admin", vbExclamation, "Not Allowed"

   CanCustomize = false

end if

On Error GoTo 0

End Function
```

Attachment_New

将附件添加到 ALM 时触发此事件。

主题	描述
语法	`Attachment_New(Attachment)` 其中 `Attachment` 是 `lAttachment` 接口
类型	子例程
返回	
可用性	Attachment_New（所有模块）

Attachment_CanOpen

在 ALM 从服务器打开附件之前触发此事件，以检查该附件可否打开。

主题	描述
语法	`Attachment_CanOpen(Attachment)` 其中 `Attachment` 是 `IAttachment` 接口
类型	函数
返回	True 或 False
可用性	`Attachment_CanOpen`（所有模块）

Attachment_CanPost

在 ALM 更新服务器上的现有附件之前触发此事件，以检查该附件可否更新。

主题	描述
语法	`Attachment_CanPost(Attachment)` 其中 `Attachment` 是 `IAttachment` 接口
类型	函数
返回	True 或 False
可用性	`Attachment_CanPost`（所有模块）

Attachment_CanDelete

在 ALM 从服务器删除附件之前触发此事件，以检查该附件可否删除。

主题	描述
语法	Attachment_CanDelete(Attachment) 其中 Attachment 是 IAttachment 接口
类型	函数
返回	True 或 False
可用性	Attachment_CanDelete（所有模块）

GetDetailsPageName

此事件由 ALM 触发，在以下对话框中检索 PageNum 中指定索引号的页面（选项卡）的名称：

- 实体的“详细信息”对话框
- 实体的“新建 <实体>”对话框

主题	描述
语法	GetDetailsPageName(PageName, PageNum) 其中 PageName 是默认的页面名称（例如，Page 1），而 PageNum 是页码。 注：页码是绝对页码，而不考虑相对于该对话框中其他显示的页面的页面相对位置
类型	函数
返回	包含页面名称的字符串
可用性	GetDetailsPageName（所有模块）

示例 - 更新选项卡

```
Function GetDetailsPageName(PageName,PageNum)
  On Error Resume Next
     Select Case activemodule
        Case "Requirements"
           Select Case PageNum
                  Case 1
                    GetDetailsPageName="Req_Details-First Tab"
                  Case 2
                    GetDetailsPageName="Req_Details-Second Tab"
                  Case 3
                    GetDetailsPageName="Req_Details-Third Tab"
           End select
        Case "Defects"
           Select Case PageNum
                  Case 1
                    GetDetailsPageName="Def_Details-First Tab"
                  Case 2
                    GetDetailsPageName="Def_Details-Second Tab"
                  Case 3
                    GetDetailsPageName="Def_Details-Third Tab"
           End select
     End select
  On Error GoTo 0
End Function

Function GetNewBugPageName(PageName,PageNum)
On Error Resume Next
  Select Case PageNum
         Case 1
           GetNewBugPageName="Def_Details-First Tab"
         Case 2
           GetNewBugPageName="Def_Details-Second Tab"
```

```
        Case 3
          GetNewBugPageName="Def_Details-Third Tab"
  End select
  On Error GoTo 0
End Function
```

实体模块

Entity_New

将对象添加到 ALM 时触发此事件。可以将代码添加到此事件过程，以便在添加新对象时执行操作。

主题	描述
语法	<实体>_New
类型	子例程
返回	
可用性	AnalysisItem_New AnalysisItemFolder_New Baseline_New Bug_New BusinessModelFolder_New BusinessModelPath_New Component_New ComponentFolder_New ComponentStep_New Cycle_New DashboardFolder_New DashboardPage_New DesignStep_New Library_New LibraryFolder_New Release_New ReleaseFolder_New

	Req_New Resource_New ResourceFolder_New Step_New Test_New TestConfiguration_New TestFolder_New TestSet_New TestSetFolder_New

Entity_MoveTo

用户将焦点从一个对象更改到另一个时，触发此事件。

可以将代码添加到此事件过程，以便在用户更改焦点时执行操作。

主题	描述
语法	<实体>_MoveTo
类型	子例程
返回	
可用性	AnalysisItem_MoveTo AnalysisItemFolder_MoveTo Baseline_MoveTo Bug_MoveTo BusinessModel_MoveTo BusinessModelActivity_MoveTo BusinessModelFolder_MoveTo BusinessModelPath_MoveTo Component_MoveTo ComponentFolder_MoveTo（以前的 MoveToComponentFolder） ComponentStep_MoveTo Cycle_MoveTo DashboardFolder_MoveTo DashboardPage_MoveTo DesignStep_MoveTo Library_MoveTo

	LibraryFolder_MoveTo Release_MoveTo ReleaseFolder_MoveTo Req_MoveTo Resource_MoveTo ResourceFolder_MoveTo Run_MoveTo Step_MoveTo Test_MoveTo TestConfiguration_MoveTo TestFolder_MoveTo TestSet_MoveTo TestSetFolder_MoveTo TestSetTests_MoveTo

示例 - 移动时更新设置

移到另一实体时更新设置。

```
Sub Bug_MoveTo
        Select Case Bug_Fields.Field("BG_STATUS").value
           Case “New”
               Setup_Status_New
           Case “Open”
               Setup_Status_Open
           Case “Fixed”
               Setup_Status_Fixed
           Case “Closed”
               Setup_Status_Closed
        End select
End sub
```

示例 - 依赖关系列表

以下代码演示如何根据其他字段的值更改与某字段关联的列表。

假定向需求实体中添加了名为 *SUB_AREA* (RQ_USER_01) 和 *TESTING_AREA* (RQ_USER_02) 的用户定义字段，并为每个测试区域添加了一个名为 *SUB_LIST_<测试区域>* 的用户定义列表。

应当在 <实体>_MoveTo 和 <实体>_FieldChange 事件中调用此代码。

```
Req_Fields.field(“RQ_USER_02”).List = Lists("SUB_LIST_" &
Req_Fields.field(“RQ_USER_01”).value)
```

Entity_ FieldCanChange

在 ALM 更改字段值之前触发此事件，以确定可否更改字段。

可以将代码添加到此事件过程以禁止字段在特定情况下更改。

主题	描述
语法	<实体>_FieldCanChange(FieldName, NewValue) 其中 FieldName 是字段的名称，NewValue 是字段值
类型	函数
返回	True 或 False
可用性	AnalysisItem_FieldCanChange AnalysisItemFolder_FieldCanChange Baseline_FieldCanChange Bug_FieldCanChange BusinessModel_FieldCanChange BusinessModelActivity_FieldCanChange BusinessModelFolder_FieldCanChange BusinessModelPath_FieldCanChange Component_FieldCanChange ComponentFolder_FieldCanChange ComponentStep_FieldCanChange Cycle_FieldCanChange DashboardFolder_FieldCanChange DashboardPage_FieldCanChange DesignStep_FieldCanChange Library_FieldCanChange LibraryFolder_FieldCanChange

	Release_FieldCanChange ReleaseFolder_FieldCanChange Req_FieldCanChange Resource_FieldCanChange ResourceFolder_FieldCanChange Run_FieldCanChange Step_FieldCanChange Test_FieldCanChange TestConfiguration_FieldCanChange TestFolder_FieldCanChange TestSet_FieldCanChange TestSetFolder_FieldCanChange TestSetTests_FieldCanChange

示例 - 允许或拒绝更改

此函数可允许或拒绝某些用户组根据当前值和新值更改缺陷中*状态*字段的权限。

```
Function Bug_FieldCanChange(FieldName, NewValue)
On Error Resume Next
if FieldName = "BG_STATUS" then
       if User.IsInGroup("QATester") then
              if Bug_Fields.Field("BG_STATUS").value ="Fixed" then
                 Select Case NewValue
                    Case "Fixed", "Closed"
                        Bug_FieldCanChange  = true
                    Case else
                        Bug_FieldCanChange  = false
                        Exit function
                     End select
              End if
       End if
End if
 On Error GoTo 0
End Function
```

Entity_ FieldChange

指定字段的值更改时，触发此事件。字段失去焦点时，值的每次更改都触发字段更改事件。

可以将代码添加到此事件过程，以便在特定字段的值更改时执行操作。例如，可以根据用户输入另一个字段中的值隐藏或显示一个字段。

主题	描述
语法	<实体>_FieldChange(FieldName) 其中 FieldName 是字段的名称
类型	子例程
返回	
可用性	AnalysisItem_FieldChange AnalysisItemFolder_FieldChange Baseline_FieldChange Bug_FieldChange BusinessModel_FieldChange BusinessModelActivity_FieldChange BusinessModelFolder_FieldChange BusinessModelPath_FieldChange Component_FieldChange ComponentFolder_FieldChange ComponentStep_FieldChange Cycle_FieldChange DashboardFolder_FieldChange DashboardPage_FieldChange DesignStep_FieldChange Library_FieldChange LibraryFolder_FieldChange Release_FieldChange ReleaseFolder_FieldChange Req_FieldChange Resource_FieldChange ResourceFolder_FieldChange Run_FieldChange Step_FieldChange Test_FieldChange

	TestConfiguration_FieldChange TestFolder_FieldChange TestSet_FieldChange TestSetFolder_FieldChange TestSetTests_FieldChange

示例 - 依赖关系值

将测试状态更改为 *To Automate* 值时，添加模板描述。

```
Sub Test_FieldChange(FieldName)

On Error Resume Next

  if Test_Fields.Field("TS_STATUS").Value="To Automate" then

     if Test_Fields.Field("TS_DESCRIPTION").value="" then

       myComments="<html><body><b>TO AUTOMATE-" & Now & "/ Checked
by " & user.UserName & "</b><br></body></html>"

       Test_Fields.Field("TS_DESCRIPTION").value= myComments

     Else

       myComments="<br><b>TO AUTOMATE-" & Now & "/ Checked by "&
       user.UserName & "</b><br>"

       Test_Fields.Field("TS_DESCRIPTION").value =
       Test_Fields.Field("TS_DESCRIPTION").value & "<br> "&
       myComments

     End if

  End if

 On Error GoTo 0

End Sub
```

示例 —— 更改时更新设置

字段（如缺陷状态）发生更改时更新设置：

```
Sub Bug_FieldChange(FieldName)

On Error Resume Next

If FieldName="BG_STATUS" then

   Select Case Bug_Fields.Field("BG_STATUS").value

          Case "New"

              Setup_Status_New
```

```
            Case "Open"
                Setup_Status_Open
            Case "Fixed"
                Setup_Status_Fixed
            Case "Closed"
                Setup_Status_Closed
  End select
End if
 On Error GoTo 0
End Sub
```

另请参见*依赖关系列表*示例。

Entity_ CanPost

在 ALM 将对象发布到该服务器之前触发此事件，以检查该对象可否发布。可以将代码添加到此事件过程以禁止对象在特定情况下发布。

主题	描述
语法	<实体>_CanPost
类型	函数
返回	True 或 False
可用性	AnalysisItem_CanPost AnalysisItemFolder_CanPost Baseline_CanPost Bug_CanPost BusinessModel_CanPost BusinessModelFolder_CanPost BusinessModelPath_CanPost Component_CanPost ComponentFolder_CanPost Cycle_CanPost DashboardFolder_CanPost DashboardPage_CanPost Library_CanPost

	LibraryFolder_CanPost Release_CanPost ReleaseFolder_CanPost Req_CanPost Resource_CanPost ResourceFolder_CanPost Run_CanPost Step_CanPost Test_CanPost TestConfiguration_CanPost TestFolder_CanPost TestSet_CanPost TestSetFolder_CanPost

示例 - 禁止更新

如果某个需求已完成但没有注释，则不允许用户提交该需求。

```
Function Req_CanPost
On Error Resume Next
if Req_Fields.Field("RQ_REQ_PRIORITY").IsModified then
   if Req_Fields.Field("RQ_DEV_COMMENTS").IsModified=false then
      Req_CanPost=false
      MsgBox "The priority was updated, you have to add a comment"
      Exit function
   End if
End if
On Error GoTo 0
End Function
```

Entity_CanDelete

在 ALM 从服务器删除对象之前触发此事件，以检查该对象可否删除。

主题	描述
语法	<实体>_CanDelete
类型	函数
返回	True 或 False
可用性	AnalysisItem_CanDelete AnalysisItemFolder_CanDelete Baseline_CanDelete Bug_CanDelete BusinessModel_CanDelete BusinessModelFolder_CanDelete BusinessModelPath_CanDelete Component_CanDelete ComponentFolder_CanDelete Cycle_CanDelete DashboardFolder_CanDelete DashboardPage_CanDelete Library_CanDelete LibraryFolder_CanDelete Release_CanDelete ReleaseFolder_CanDelete Req_CanDelete Resource_CanDelete ResourceFolder_CanDelete Test_CanDelete TestConfiguration_CanDelete TestFolder_CanDelete TestSet_CanDelete TestSetFolder_CanDelete

Entity_AfterPost

在对象发布到服务器之后触发此事件。在发布项目字段之后，不应当更改它们，否则数据库中就不会储存新值。

主题	描述
语法	<实体>_AfterPost
类型	子例程
返回	
可用性	AnalysisItem_AfterPost AnalysisItemFolder_AfterPost Baseline_AfterPost Bug_AfterPost BusinessModel_AfterPost BusinessModelFolder_AfterPost BusinessModelPath_AfterPost Component_AfterPost ComponentFolder_AfterPost Cycle_AfterPost DashboardFolder_AfterPost DashboardPage_AfterPost Library_AfterPost LibraryFolder_AfterPost Release_AfterPost ReleaseFolder_AfterPost Req_AfterPost Resource_AfterPost ResourceFolder_AfterPost Run_AfterPost Step_AfterPost Test_AfterPost TestConfiguration_AfterPost TestFolder_AfterPost TestSet_AfterPost TestSetFolder_AfterPost

示例 - 发送邮件

如果修改了“目标发布”字段，则将向需求作者发送通知邮件。要发送邮件，需要添加一个名为 sendreqmail 的自定义函数。

```
Sub Req_AfterPost

If Req_Fields.field("RQ_TARGET_RCYC").IsModified Then

   Sendreqmail Req_Fields.field("RQ_REQ_ID").Value,
   Req_Fields.field("RQ_REQ_AUTHOR").Value, "", "Target Cycle has
   changed", "Please Review"

End if

End sub

Sub sendreqmail(ReqId,Mto,cc,msubject,mcomment)

Dim tdc, bgf, bg

       Set tdc = TDConnection

       Set rf =  tdc.ReqFactory

       Set req = rf.Item(ReqId)

       req.Mail  mto , cc, 2, mSubject, mComment


       Set req = Nothing

       Set rf  = Nothing

       Set tdc = Nothing

End sub
```

工作流示例 - 定义设置

此工作流代码可更新字段属性：可见性、强制、只读和顺序。

将下面的函数添加到“缺陷模块”节点。

检查用户

检查用户是否位于特定组中以决定下一操作。

```
If User.IsInGroup("Developer") then

       Mygroup=” DEV”

End if
```

设置字段外观

此子例程可设置字段外观 —— 可见性、强制状态、页码以及在屏幕上的显示顺序。

```
Sub SetFieldApp( FieldName, Vis, Req, PNo, VOrder )
    With Bug_Fields(FieldName)
            .IsVisible = Vis
            .IsRequired = Req
            .PageNo = PNo
            .ViewOrder = VOrder
    End With
End Sub
```

重置为初始值

添加以下子例程可隐藏所有缺陷字段。

```
Sub ResetMetadata
       For i=0 to Bug_Fields.Count
           Bug_Fields.FieldById(i).IsVisible = false
       Next
End sub
```

设置状态

以下子例程可根据用户权限设置“New”状态。

您需要针对每个状态编写此子例程。

```
Sub Setup_Status_New
If User.IsInGroup("Developer") then
   Mygroup=” DEV”
ElseIf User.IsInGroup("QATester") then
   Mygroup=” QA”
ElseIf User.IsInGroup("Documentation") then
   Mygroup=” DOC”
End if
```

```
    Call ResetMetadata  'set to initial status
    Select case Mygroup
      Case " DEV"
         SetFieldApp "BG_ACTUAL_FIX_TIME", True, False, 0, 0
         SetFieldApp "BG_CLOSING_DATE", True, False, 0, 1
         SetFieldApp "BG_CLOSING_VERSION", True, False, 0, 2
         SetFieldApp "BG_DETECTED_BY", True, True, 0, 3
         SetFieldApp "BG_DETECTED_IN_RCYC", True, False, 0, 4
         SetFieldApp "BG_DETECTED_IN_REL", True, False, 0, 5
         SetFieldApp "BG_DETECTION_DATE", True, True, 0, 6
         SetFieldApp "BG_DETECTION_VERSION", True, False, 0, 7
         SetFieldApp "BG_ESTIMATED_FIX_TIME", True, False, 0, 8
         SetFieldApp "BG_PLANNED_CLOSING_VER", True, False, 0, 8
         SetFieldApp "BG_PRIORITY", True, False, 0, 10
    Case "  QA"
         SetFieldApp "BG_ACTUAL_FIX_TIME", True, False, 0, 0
         SetFieldApp "BG_CLOSING_DATE", True, False, 0, 1
         SetFieldApp "BG_CLOSING_VERSION", True, False, 0, 2
         SetFieldApp "BG_DETECTED_BY", True, True, 0, 3
         SetFieldApp "BG_DETECTED_IN_RCYC", True, False, 0, 4
         SetFieldApp "BG_DETECTED_IN_REL", True, False, 0, 5
         SetFieldApp "BG_DETECTION_DATE", True, True, 0, 6
         SetFieldApp "BG_DETECTION_VERSION", True, False, 0, 7
    Case " DOC"
         SetFieldApp "BG_ACTUAL_FIX_TIME", True, False, 0, 0
         SetFieldApp "BG_CLOSING_DATE", True, False, 0, 1
         SetFieldApp "BG_CLOSING_VERSION", True, False, 0, 2
         SetFieldApp "BG_DETECTED_BY", True, True, 0, 3
        SetFieldApp "BG_DETECTED_IN_RCYC", True, False, 0, 4
  End Select
End sub
```

4 结论

对运行良好的相关软件的需求推动了业务的创新与成功。软件对业务的重要程度不断增加，结合复杂的颠覆性趋势（如虚拟化和云）的出现，将继续推动流程改进需求。

HP ALM 可加强团队间的合作，整合战略和规划团队，提供最佳实践来激励创新和防止战术延迟，并连接运作组织关键的最后阶段，从而满足了现代应用程序生命周期的需求。HP ALM 具有可扩展性和动态性，准备迎接 ALM 为您带来的动态优势吧！它的灵活性支持涵盖从制药到汽车制造的各种行业，从经典的瀑布式到现代的敏捷式的各种开发类型，从平面到分层再到矩阵的各种组织结构，例子不胜枚举。

在许多方面，这是内置到产品中的许多自定义功能的结果，这些功能提供工具来区分对采用 ALM 的每个组织唯一的业务流程。项目管理员可以使用工作流脚本调整标准过程和屏幕以满足项目的特定需求。本文档深入探讨了使用模式，显示了不同编码方法的优缺点，详述了最有用的事件，并列举了许多可帮助编写代码的实用示例。

我们相信，本文档中列出的最佳实践将推动您在组织中正确采用 HP ALM 工作流。