IBM

eserver

iSeries

WebSphere® 开发工作室: 应用程序开发管理器用户指南

版本 5

SB84-0449-00

IBM

eserver

iSeries

WebSphere® 开发工作室: 应用程序开发管理器用户指南

版本 5

SB84-0449-00

注意！

使用本资料和它支持的产品之前，请务必阅读第251页的『注意事项』下的一般信息。

第 3 版（2001 年 5 月）

此版本适用于 IBM® WebSphere® Development Studio iSeries 版（程序 5722–WDS）的版本 5 发行版 1 修订版 0、“应用程序开发管理器”组件，以及所有后续发行版和修订版，除非新版本另有指示。本版本仅适用于“精简指令集计算机”(RISC) 系统。

此版本替换 SC09-2133-01。

通过当地的 IBM® 代表或 IBM 分部来订购出版物。出版物未存放在下面给定的地址处。

IBM 欢迎您提出意见。可以将您的意见寄往:

IBM Canada Ltd. Laboratory
Information Development
2G/KB7/1150/TOR
1150 Eglinton Avenue East
North York, Ontario, Canada M3C 1H7

也可以通过传真（注意: RCF Coordinator）或者以电子形式向 IBM 公司提出意见。有关发表意见方法的描述，参见『如何发表意见』。

当您向 IBM 发送信息时，就授予了 IBM 公司非专有权，IBM 可以它认为适当的任何方式来使用或分发这些信息，而不必向您负任何责任。

目录

关于本书

应用程序开发管理器，它是“应用程序开发工具箱”(ADTS) 组件的一个功能部件，它将方便您在 iSeries 400® 环境中开发应用程序。其活动分为两类：管理任务和开发任务。本书中包含有关下列方面的信息:

- “应用程序开发管理器”功能部件的功能
- “应用程序开发管理器”部件开发命令
- “应用程序开发管理器”管理命令
- 如何使用搜索路径和用户定义类型
- 如何构建、测试、运行、导出和分布应用程序
- 对远程系统的应用程序分发
- 安全性注意事项
- 如何使用“编程开发管理器”(PDM) 实用程序
- 如何管理 System/36™ 应用程序
- System/38™ 注意事项
- 如何使用 VARPG 部件

谁应使用本书

本书是为计划创建和管理为“应用程序开发管理器”功能部件定义的项目的项目管理员以及将使用此功能部件来开发应用程序的应用程序开发者编写的。

在使用本书之前，应熟悉下列各项内容:

- 关于此功能部件的一般概念。您应当已经阅读了 *ADTS/400: Application Development Manager Introduction and Planning Guide*。
- 工作站（也称为显示站）及其控件。
- 运行于 Operating System/400 (OS/400) 系统上的 IBM iSeries 400 系统和软件。
- “编程开发管理器”(PDM) 实用程序。有关此实用程序的信息，参见 *ADTS/400: Programming Development Manager*。

先决条件和相关信息

使用 “iSeries 信息中心” 来作为您了解 iSeries 和 AS/400e™ 技术信息的起点。您可以通过两种方法来访问 “信息中心”：

- 从以下 Web 站点访问:

  `http://www.ibm.com/eserver/iseries/infocenter`

- 从随 Operating System/400 订单一起交付的 CD-ROM 访问:

  iSeries 信息中心, SB84-0455-00. 此软件包还包括 PDF 版本的 iSeries 手册 *iSeries 信息中心: 手册补遗*, SB84-0456-00，它用来替换 “软拷贝库” CD-ROM。

“iSeries 信息中心”包含顾问程序和重要主题，如 CL 命令、系统应用程序编程接口 (API)、逻辑分区、群集、Java™、TCP/IP、Web 服务和安全网络。它还包括指向相关 IBM® 红皮书的链接以及指向其他 IBM Web 站点（例如 Technical Studio 和 IBM 主页）的因特网链接。

第249页的『文献目录』中列示了与“应用程序开发管理器”功能部件紧密相关的手册。

如何发送您的意见

您的反馈很重要，它可帮助我们提供最准确的、高质量的信息。IBM 欢迎您对本书或任何其他 iSeries 文档发表任何意见。

- 如果您喜欢通过邮件发送意见，可使用以下地址:

  IBM Canada Ltd. Laboratory
  Information Development
  2G/KB7/1150/TOR
  1150 Eglinton Avenue East
  North York, Ontario, Canada M3C 1H7

  如果从美国以外地区邮寄读者意见表，可以将此表交给当地的 IBM 分部或 IBM 代表，以进行邮资已付的邮寄。
- 如果您喜欢通过传真发送意见，可使用以下号码:
  - 1-416-448-6161
- 如果您喜欢用电子方式发送意见，可使用以下电子邮件地址之一:
  - 对这些书的意见:

    torrcf@ca.ibm.com

    IBMLink: toribm(torrcf)
  - 对“iSeries 信息中心”的意见:

    RCHINFOC@us.ibm.com

确保包括下列各项:

- 书名。
- 书号。
- 您的意见涉及的页号或主题。

样本项目

本书使用一个基于 Payroll 应用程序的样本项目来说明需要执行的计划以及创建可用应用程序开发环境的方法。但是，如何实现“应用程序开发管理器”功能部件的功能取决于企业的特殊需要和意愿、其过程，以及它计划开发的应用程序的性质。样本项目（如它在“应用程序开发管理器”库中所使用的那样）只说明了一种开发小组可能采用的方法。

定义

本书不包含词汇表。新术语是在第一次出现的地方定义的，并且以**粗体**字形式突出显示出来。为了便于参考，还在术语本身和题为定义的索引条目下为每个定义创建了索引。

获取联机信息

提供了下列联机信息:

- **有关控制语言 (CL) 命令的帮助**

  有两种类型的帮助可用: **上下文**帮助和**扩充**帮助。上下文帮助用于说明字段。扩充帮助用于说明显示或命令的目的。

  要查看关于 CL 命令的提示屏幕，输入该命令，然后按 F4=提示。要查看上下文帮助，输入该命令并将光标定位在字段上，然后按 F1=帮助。要查看扩充帮助，当您查看该命令的上下文帮助时按 F2=扩充帮助。

  要查看所有“应用程序开发管理器”命令的列表，输入:

  ```
  GO CMDADM
  ```

  要查看与项目相关的“应用程序开发管理器”命令的列表，输入:

  ```
  GO CMDPRJ
  ```

  要查看与组相关的“应用程序开发管理器”命令的列表，输入:

  ```
  GO CMDGRP
  ```

  要查看与部件相关的“应用程序开发管理器”命令的列表，输入:

  ```
  GO CMDPART
  ```

- **“编程开发管理器”屏幕的帮助**

  要查看上下文帮助，将光标定位在字段上，然后按 F1=帮助。要查看扩充帮助，可以将光标定位在上下文帮助可用的区域之外，然后按 F1=帮助; 也可以在查看上下文帮助时按 F2=扩充帮助。

第1章 应用程序开发管理器功能部件介绍

“应用程序开发管理器”是“应用程序开发工具箱”组件的一个功能部件，为在 iSeries 400 环境中工作的应用程序开发者提供了一种机制，使他们可以在整个应用程序有效期内有效地和高效地管理应用程序对象。此功能部件有两种类型的用户：项目管理员和应用程序开发者。**项目管理员**是负责定义项目层次结构和登记用户的人员。**应用程序开发者**是使用“应用程序开发管理器”功能部件来开发应用程序组件的应用程序员。

本章提供了关于下列主题的信息:

- 项目管理的概述
- 部件开发的概述
- 应用程序开发环境
- 用于应用程序开发的工具

项目管理的概述

为了使组织或企业使用此功能部件，某个人员（在此情况中为项目管理员）必须在“应用程序开发管理器”环境中创建项目及其框架。创建项目的人员即自动成为该项目的管理员，且具有在该项目中创建组和登记用户的必要权限。**组**是开发过程中处于同一阶段的部件的集合。它表示为库，其权限由“应用程序开发管理器”功能部件控制。**项目层次结构**是组织成一些层（每一层表示开发过程中的一个阶段）的组的集合。

管理员建立了应用程序框架，以使其他人员在定义项目层次结构时进行工作。要创建高效的层次结构，项目管理员必须非常了解“应用程序开发管理器”功能部件。另外，他还必须具有关于将处于此功能部件控制之下的信息的大量知识。项目管理员还必须了解组织在开发和维护应用程序时所遵循的过程。另外，具有操作系统和系统实用程序经验是很重要的。

项目管理员的角色

项目管理员的角色和活动的范围是由在项目层次结构中管理的应用程序确定的。首次使用此功能部件的项目管理员通常将执行第2页的图1中说明的五个步骤以便让应用程序开发者在“应用程序开发管理器”环境中开始他们的工作。

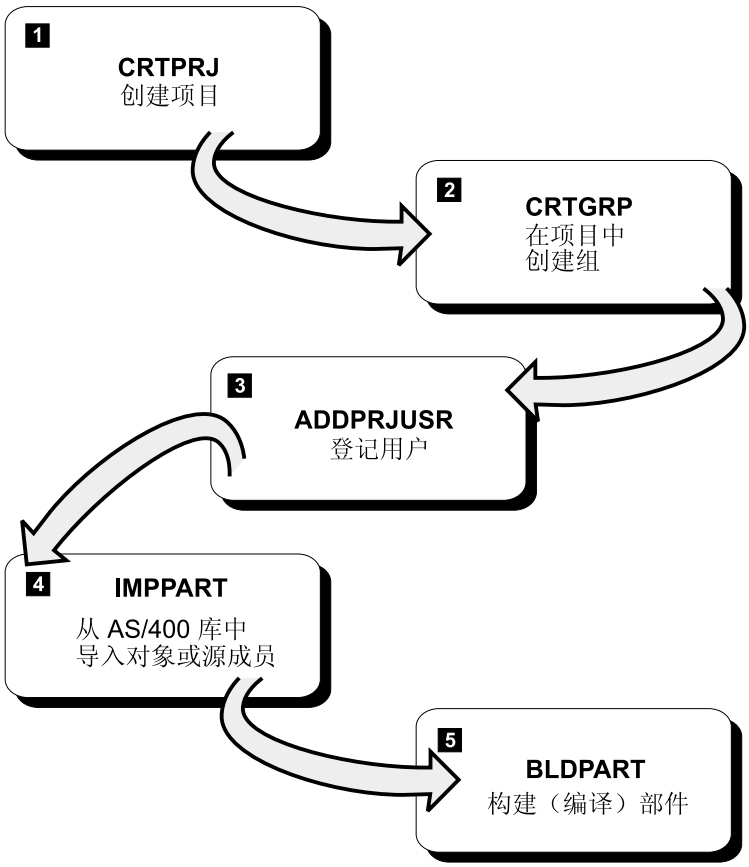

图 1. 开始使用此功能部件所需的五个基本步骤

另外，项目管理员的角色可能还需要:

- 在项目层次结构中提升项目的元素，例如，从测试层提升到主层
- 测试应用程序
- 将应用程序从“应用程序开发管理器”环境层次结构导出到生产或测试环境
- 在系统故障之后回收项目层次结构信息
- 开发应用程序的部件或组件

任何用户都可以创建项目。充当项目管理员的人员的用户简要表不需要特殊权限。创建项目的人员将自动成为该项目的项目管理员。一个项目的授权项目管理员不会还自动具有充当其他任何项目的项目管理员的权限。而是由每个项目的项目管理员将任何同事登记为项目管理员。在所有项目中，用户简要表 QSECOFR 实质上登记为项目管理员。即，QSECOFR 被当作所有项目中的项目管理员，而不管 QSECOFR 是否向各项目进行了登记。

项目管理员的角色与应用程序开发者的角色也是不同的。项目管理员对项目中的任何内容都具有更新访问权，而应用程序开发者只对特定的工作区或组具有更新访问权，如管理员所授权的那样。只是在应用程序开发活动的开始和完成时才进行基本的管理任务，而在整个开发周期中都要进行开发任务。并且，根据项目范围和开发小组规模的不同，项目管理员还可能会充当应用程序开发者。还要注意，项目管理员不需要是*系统管理员*。在 iSeries 400 上下文中，系统管理员负责与管理整个系统有关的各项工作。

部件开发的概述

应用程序开发就是对应用程序进行编码、编译、运行、调试和测试。应用程序开发者可以使用“应用程序开发管理器”功能部件来执行这些任务。此功能部件提供了一个框架，可以在其中管理组成应用程序的部件的开发。**部件**是一个对象（如文件或程序），或者是包含 RPG 程序的 RPG/400® 代码的源成员。部件具有系统提供的类型（如 RPGSRC、RPGINC、PGM 或 FILE），或者您可以使用“添加 ADM 类型”(ADDADMTYPE) 命令来定义您自己的类型。部件还可以具有由系统提供的语言（如 RPG 或 SQLRPG），并且您可以使用 ADDADMLANG 命令来为用户定义部件类型定义您自己的语言。

必须先存在项目层次结构，然后开发者才能使用“应用程序开发管理器”功能部件来执行部件开发任务。开发者还必须在项目中进行登记并对组具有更新权限。所有这些都是由项目管理员来组织的。

当部件需要更新时，开发者必须将部件**检出**到他们对其具有更新访问权的组中。要检出部件，该部件的适当版本被复制到开发者的组中，如果需要的话，还要进行锁定，以防止其他开发者更改它。当完成对部件的更新时，开发者会将该部件从他／她的开发组中提升到其父组中。

在任何时候，在一个项目中都可以存在一个部件的多个版本。例如，当应用程序第一个版本的工作完成之后，项目管理员就可以向项目层次结构中添加组，以便开发者可以开始开发应用程序的第二个版本。

与创建、更改或删除“应用程序开发管理器”项目、组和部件相关联的所有任务都应该使用“应用程序开发管理器”接口来完成。这些接口包括 CL 命令、“编程开发管理器”屏幕，以及 WRKPRJPDM、WRKGRPPDM 和 WRKPARTPDM 命令。如果管理员或开发者使用其他接口来更改或创建“应用程序开发管理器”信息，则结果将不可预料。构建过程可能不能识别出部件已更改，从而可能不能重新编译或处理它。这将导致各部件与其相应对象和成员之间的不一致性。

只能在您对其具有更新访问权的组或库中更改部件。如果您决定在“应用程序开发管理器”环境之外更改部件，则 CHGPARTINF 命令将更新该部件的内部信息。

应用程序开发者的角色

此功能部件使您能够在受控环境中开发不同部件的版本。另外，多个开发者也可以同时更新应用程序的部件。此功能部件可确保由一个开发者所作的更改不会与另一个开发者所作的更改发生冲突。部件的构建（编译）是自动进行的。

应用程序开发者执行的基本任务是:

- 创建新部件；从此功能部件的控件内复制部件，或者通过从其控件外部导入部件来复制它们；以及更改部件
- 使用特定的构建范围来构建部件，构建搜索路径，以及构建选项
- 在项目层次结构中将部件提升
- 在“应用程序开发管理器”环境内测试应用程序的部件，或者通过将部件导出至测试库来在环境外部测试部件

应用程序开发环境

图2 将在“应用程序开发管理器”功能部件控制之下的应用程序开发与在此功能部件的控制之外的应用程序开发进行比较。图中各框的流程说明了通常您可如何创建源代码、编辑它，然后编译它。图表左边显示了您使用的“应用程序开发管理器”部件开发命令。如果您不在此功能部件的控制之下开发部件，则图表右边显示了您使用的 OS/400 命令。

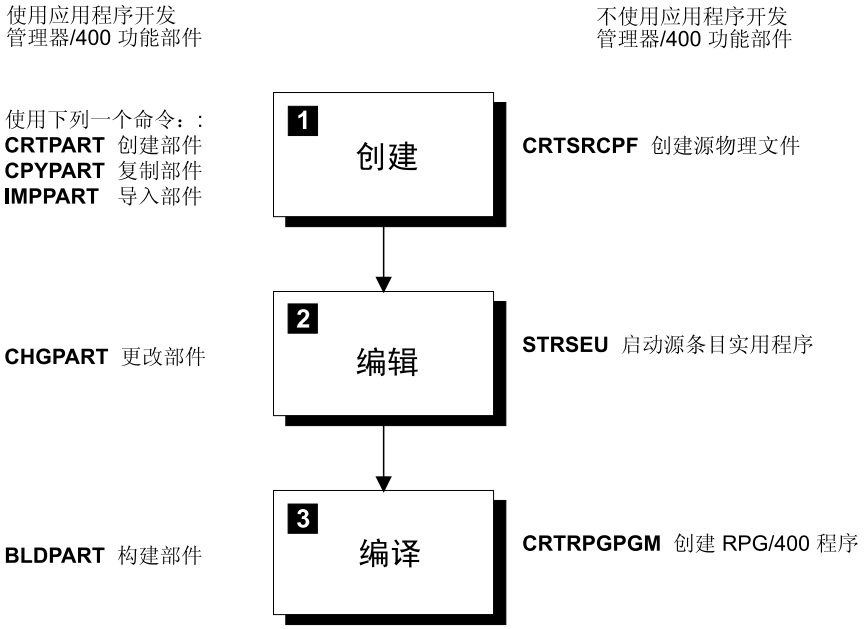

图 2. 应用程序开发环境

应用程序开发的工具

可使用下列任何一个接口来访问“应用程序开发管理器”功能部件中的功能:

- OS/400 命令行中的控制语言 (CL) 命令
- “应用程序开发工具箱”组件的“编程开发管理器”(PDM) 实用程序
- “应用程序开发工具箱”组件的“应用程序字典服务”功能部件
- iSeries 产品的“WebSphere 开发工具”
- IBM CoOperative Development Environment/400 (CODE/400) 组件
- VisualAge® RPG 工作站工具

使用应用程序开发工具箱实用程序

可以通过使用“编程开发管理器”(PDM) 实用程序或者使用命令行中的“应用程序开发管理器”控制语言 (CL) 命令来执行管理任务和开发任务。在本书中，每个命令的第一个示例都显示了 CL 命令及其在使用“编程开发管理器”实用程序时要遵循的等价步骤。附录A. 应用程序开发管理器控制语言命令 提供了这些 CL 命令的完整列表。

可以使用下列 PDM 屏幕来执行这些管理任务和开发任务:

用 PDM 来使用项目
用 PDM 来使用组
用 PDM 来使用部件

有关 PDM 实用程序的概述，参见第16章 使用编程开发管理器实用程序。

有几个配合“应用程序开发管理器”功能部件使用的其他 Application Development ToolSet/400 实用程序。这些实用程序包括“源条目实用程序”(SEU)、“屏幕设计辅助”(SDA)、“报告布局实用程序” (RLU)、“数据文件实用程序”(DFU)、“文件比较与合并实用程序”FCMU)。从“用 PDM 来使用部件”屏幕上的选项就可访问 SEU、SDA、RLU 和 DFU。

有关这些实用程序的 Application Development ToolSet/400 出版物的列表，参见文献目录。

使用应用程序字典服务功能部件

“应用程序字典服务”是“应用程序开发工具箱”组件的一个功能部件。“应用程序字典服务”功能部件是一个效果分析工具，可以用来评估对应用程序的潜在更改的效果。

如果您正在使用“应用程序字典服务”功能部件，则还可以访问“应用程序开发管理器”功能部件为您的应用程序开发环境提供的功能。

可以通过下列两种方法之一来访问“应用程序开发管理器”部件:

1. 指定您想在 Start AppDict Services/400 (STRADS) 屏幕上使用“应用程序开发管理器”
2. 指定您想在“使用字典”屏幕上使用“应用程序开发管理器”。

有关 Application Dictionary Services/400 出版物的列表，参见文献目录。

使用“WebSphere 开发工具”产品

“WebSphere 开发工具”产品包括下列工作站工具:

- CODE/400
- VisualAge RPG
- WebSphere Studio iSeries 版
- VisualAge for Java iSeries 版
- IBM Distributed Debugger

有关相关出版物的列表，参见文献目录。

使用 CODE/400 工具

可以使用 CODE/400 编辑器、“程序生成器”或“项目组织器”之一来通过使用图形用户界面访问库、文件和成员或者“应用程序开发管理器”项目、组及部件。

CODE/400 的“代码设计器”组件允许您打开和构建 DDSSRC 部件。而且，可以使用“调试工具”组件来调试程序部件。

有关 CODE/400 出版物的列表，参见文献目录。

CODE/400 编辑器

CODE/400 编辑器允许您使用图形用户界面来选择“应用程序开发管理器”部件。

要访问“应用程序开发管理器”部件:

1. 双击 CODE/400 编辑器图标。
2. 从*文件*菜单中选择*打开*。
3. “打开”窗口出现，可以在此窗口中选择“应用程序开发管理器”部件。

程序生成器

CODE/400“程序生成器”使您能够构建“应用程序开发管理器”部件。

要构建“应用程序开发管理器”部件:

1. 双击 CODE/400“程序生成器”图标。
2. “程序生成器”窗口出现，可以在此窗口中选择要构建的“应用程序开发管理器”部件。

注: 参考与需要对 BLDOPT 文件作的必要更改有关的 CODE/400 联机信息，以便启用 CODE/400 错误反馈和“调试工具”。

项目组织器

CODE“项目组织器”是一个功能强大的工具，它使您能够使用工作站来编辑、验证、编译和调试应用程序。它使您能够使用桌面上的 OS/400 对象。您可以使用面向对象的用户界面来处理库、文件、成员或“应用程序开发管理器”项目、组和部件。

使用 VisualAge RPG 工作站工具

VisualAge RPG (VARPG) 工作站工具为 VARPG 应用程序员提供了一种开发环境，使他们可以在具有主机接口的工作站上开发、维护和归档他们的可视应用程序。

“应用程序开发管理器”功能部件使您能够在主机系统上存储和管理“应用程序开发管理器”项目中的 VisualAge RPG 应用程序。有关详情，参见第209页的『第17章 使用 VisualAge RPG 部件』。

有关 VisualAge RPG 出版物的列表，参见文献目录。

第2章 使用项目

在“应用程序开发管理器”环境中，应用程序被称为**项目**。项目是由应用程序的所有相关组件组成的。而不论它们处于哪一开发阶段。因此，项目不是 OS/400 对象。它是存储在库（组）中的一组 OS/400 对象（部件）。

项目及其中的组为应用程序开发定义项目层次结构。“应用程序开发管理器”功能部件使用项目层次结构来确定其命令所需的任何库列表的次序。有关项目层次结构以及“应用程序开发管理器”命令如何找到该层次结构中的部件的详情，参见第3章 使用项目层次结构中的组。

本章提供了关于下列主题的信息:
- 创建项目
- 显示项目的列表
- 更改项目
- 打印关于项目的信息
- 删除项目
- 用“编程开发管理器”实用程序来使用项目

您应该花些时间来阅读 *ADTS/400: Application Development Manager Introduction and Planning Guide* 一节，它讨论了您在开始创建项目之前应该考虑的一些事项。

创建项目 (CRTPRJ)

要使用“应用程序开发管理器”功能部件，必须创建项目。使用“创建项目” (CRTPRJ) 命令来执行该任务。

任何用户都可以创建项目。一旦您创建了项目，您就成为该项目的项目管理员，可以用所有管理命令来使用它。您可以添加其他项目管理员。用户简要表 QSECOFR 是所有项目的管理员。

注

建议为您的每个应用程序都创建单独的“应用程序开发管理器”项目，而为对于几个应用程序为公共的任何部件创建另一个项目。使用单独的项目将提高“用 PDM 来使用部件”屏幕的性能，并且将使您能更好地控制应用程序。

创建项目时，您可以选择是否想要在重新构建物理文件时自动保存和恢复物理文件中的测试数据。此参数的缺省值是 SAVDTA(*YES)。构建过程将在构建之前保存测试数据，而在重新构建物理文件之后恢复这些数据。如果您指定 SAVDTA(*NO)，则在重新构建物理文件时您的数据会被删除。

创建项目时，您可以在 TEXT 参数上提供项目描述。例如，您可能想要在与项目相关联的文本中标识应用程序的名称和目的。

示例

此示例说明如何创建一个称为 PAYROLL 的项目，其项目简称为 PAY。项目简称最长可以为 4 个字符，用来为项目中的组创建唯一的库名。与物理文件相关联的任何数据是在进行构建过程时保存和恢复的。如 TEXT 参数所述，PAYROLL 项目是一个每周 Payroll 处理应用程序。

使用 CRTPRJ 命令

```
CRTPRJ PRJ(PAYROLL) SHORTPRJ(PAY) SAVDTA(*YES)
       TEXT('WEEKLY PAYROLL PROCESSING APPLICATION')
```

使用“编程开发管理器”实用程序

按“用 PDM 来使用项目”屏幕上的“F6=创建”。

使用项目

可以通过直接在任何命令行上输入 CL 命令来使用项目。还可以在 CL 程序、REXX 过程中使用命令，或者通过 QCMDEXC 来使用命令。项目命令为 CRTPRJ、QRYPRJ、CHGPRJ、PRTPRJ 和 DLTPRJ。

还可以通过使用“用 PDM 来使用项目”屏幕来通过“编程开发管理器”实用程序使用项目。

除了这些项目命令之外，此功能部件还支持“回收项目”(RCLPRJ) 命令。此命令解决了在发生系统故障或者项目管理命令异常结束之后项目中所存在的任何不一致性。有关此命令的详情，参见第178页的『回收项目 (RCLPRJ)』。

显示所有项目 (QRYPRJ)

使用“查询项目”(QRYPRJ) 命令来获取您在系统上登记的所有项目的列表。此列表上将只出现您有权读取或更新的项目。

还可以使用此命令来验证是否成功创建了项目，或者用来确定是否任何项目因系统故障而损坏或变得不一致。有关恢复项目的信息，参见第175页的『第15章 保护应用程序开发管理器信息』。

使用 OUTPUT 参数来显示应该将输出导向何处。缺省值为将报告假脱机以供打印，OUTPUT(*PRINT)。

示例

图3 说明了当您输入以下命令时假脱机文件的外观:

```
QRYPRJ OUTPUT(*PRINT)
```

```
5722WDS   V5R1M0              应用程序开发管理器  - 查询项目    05/08/01    11:59:48        页面 . . :   0001

                                      DBCS  保存
项目                                  简称  数据  数据  文本                                                  已破坏

PAYROLL                               PAY   否    是   每周 Payroll 处理应用程序                                 否
          * * * * * 列 表 结 束  * * * * *
```

图 3. 由 QRYPRJ 命令生成的假脱机文件的样本

另外，可以通过指定 OUTPUT(*OUTFILE) 来将输出导向某个输出文件。输出文件的记录格式与在库 QADM 中由系统提供的数据库文件 QALYQPRJ 中所使用的记录格式相同。

示例

库列表用来确定在何处存储称为 OUTFILE 的输出文件，文件中的第一个成员将接收输出。

```
QRYPRJ OUTPUT(*OUTFILE) OUTFILE(*LIBL/OUTFILE) OUTMBR(*FIRST)
```

更改项目 (CHGPRJ)

使用“更改项目”(CHGPRJ) 命令来更改与项目相关联的文本，或者更改当在项目中重新构建物理文件时是否想要自动保存和恢复您的测试数据。只有项目管理员才能使用此命令。

如果您指定 SAVDTA(*YES)，则构建过程将在构建之前保存测试数据，而在重新构建物理文件之后恢复这些数据。

示例

要更改与 PAYROLL 项目相关联的描述性文本，输入以下命令:

使用 CHGPRJ 命令

```
CHGPRJ PRJ(PAYROLL) SAVDTA(*SAME) TEXT('BIWEEKLY PAYROLL
       PROCESSING APPLICATION')
```

使用“编程开发管理器”实用程序

在“用 PDM 来使用项目”屏幕上选择选项 2（更改）。

此命令将不会更改任何登记信息或者更改谁对此项目具有权限。要更改登记信息，使用 CHGPRJUSR 命令，如第29页的『更改用户登记信息 (CHGPRJUSR)』一节中所述。

打印关于项目的信息 (PRTPRJ)

使用“打印项目”(PRTPRJ) 命令来查看项目的特征和项目层次结构内的各个组。可以在创建项目层次结构之后使用此命令来验证提升路径。项目管理员和应用程序开发者都可以使用此命令。

必须在 PRTPRJ 命令的 PRJ 参数上指定您已登记的项目的名称。

OUTPUT 参数的缺省值 OUTPUT(*PRINT) 将使报告假脱机以供打印。

报告显示了 PARTL 必需值，该值指示组是否要求在更改、创建或提升部件时都应给出原因。值 Y 指示必须给出原因。

示例

此示例说明如何创建第10页的图4中所显示的样本报告。

使用 PRTPRJ 命令

```
PRTPRJ PRJ(PAYROLL) OUTPUT(*PRINT)
```

使用“编程开发管理器”实用程序

从“用 PDM 来使用项目”屏幕上选择选项 6（打印）。

```
5722WDS   V5R1M0              应用程序开发管理器      - 打印项目   05/08/01    13:37:40          页面 . . :  0001

项目    . . . . . . . . . . . . . . . :   PAYROLL
项目简称. . . . . . . . . . . . . . . :   PAY
DBCS 数据 . . . . . . . . . . . . . . :   否
保存物理文件数据. . . . . . . . . . . :   是
文本  . . . . . . . . . . . . . . . . :   每周 Payroll 处理应用程序

                                                                                          PARTL
级别   组                                    简称   提升码                通知      请求   文本
 01    MASTER                                MAST   MASTER                *NONE        N   PAYROLL 项目的 MASTER 组（根）。
 02    TEST                                  TST    MASTER                *NONE        N   PAYROLL 项目的 TEST 组。
 03    DEVELOPER1                            DEV1   MASTER                *NONE        N   PAYROLL 项目的 DEVELOPER1 组。
 03    DEVELOPER2                            DEV1   MASTER                *NONE        N   PAYROLL 项目的 DEVELOPER2 组。
                  * * * * * 列 表 结 束  * * * * *
```

图 4. 由 PRTPRJ 命令生成的假脱机文件的样本

每个组的层次指示符帮助您确定项目层次结构。此层次指示符的目的是用来阐明项目内各个组的层次结构性质。图5 中表示了此报告中描述的项目层次结构。

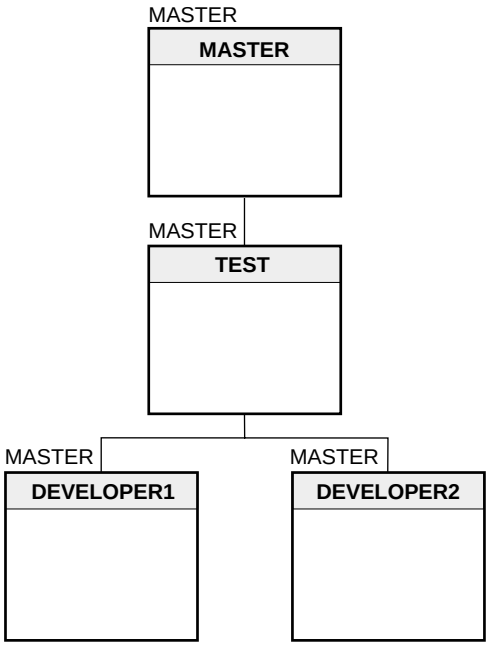

图 5. PRTPRJ 报告中描述的项目层次结构

图5 和样本报告都说明了一个具有四个组而其层次结构有三层的项目。列示于报告中的及出现在每个框外面上方的词 MASTER 表示为这些组定义的提升码。

另外，可以通过指定 OUTPUT(*OUTFILE) 来将输出导向某个输出文件。输出文件的记录格式与在库 QADM 中由系统提供的数据库文件 QALYPPRJ 中所使用的记录格式相同。

示例

库列表用来确定在何处存储称为 OUTFILE 的输出文件，文件中的第一个成员将接收输出。

```
PRTPRJ PRJ(PAYROLL) OUTPUT(*OUTFILE) OUTFILE(*LIBL/OUTFILE)
       OUTMBR(*FIRST)
```

删除项目 (DLTPRJ)

使用“删除项目”(DLTPRJ) 命令来从“应用程序开发管理器”功能部件中删除整个项目。删除项目时，项目内的所有组以及各组中的所有部件都会被删除，即使开发者正在那些组中工作。只有项目管理员才能删除项目。

必须在 PRJ 参数上指定项目名。

注

当删除项目时，归档库以及组库都会被删除。在“应用程序开发管理器”功能部件外部创建的库也会被删除（如果它的名称是项目中其中一个组的有效归档库名的话）。仅当所有者是 QPRJOWN 时才会删除归档库和组库，否则，将发出警告。有关有效归档库名的描述，参见第82页的『归档部件』。

示例

此示例说明如何从系统中除去项目 PAYROLL。

使用 DLTPRJ 命令

```
DLTPRJ PRJ(PAYROLL)
```

使用“编程开发管理器”实用程序

在“用 PDM 来使用项目”屏幕上选择选项 4（删除）。

当您使用选项 4 时，“编程开发管理器”实用程序会提供一个确认屏幕。如果直接从命令行使用此命令，就不会向您发出警告。

第3章 使用项目层次结构中的组

通常，一个项目由多个组组成。项目内的每个组都表示应用程序中代码的一个**版本**或层次。

例如，称为 MASTER 的组可表示整个应用程序的最新版本。另一个组 TEST 可表示仍然在测试的混合在一起的已更改及未更改代码。此组是在项目层次结构中的组 MASTER 下面创建的，如图6中所示。

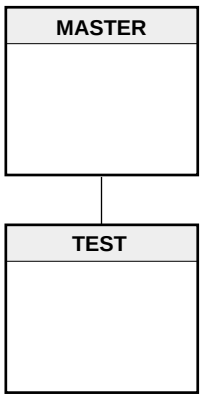

图 6. 具有两个组的项目层次结构

项目层次结构标识了搜索路径。**搜索路径**是各组的排列，它确定当在项目层次结构中查找部件时，“应用程序开发管理器”功能部件搜索各个组所采用的次序。当在组 TEST 中工作时，将首先使用在该组中找到的部件，因为它们表示应用程序中部件的最新更改版本。未在组 TEST 中找到的任何应用程序部件都在组 MASTER 中查找。有关搜索路径的详情，参见第9章 使用搜索路径。

本章提供了关于下列主题的信息:

- 创建组
- 创建项目层次结构
- 更改组
- 删除组
- 用“编程开发管理器”实用程序来使用组
- 共享开发组
- 对项目和组登记开发者和管理员
- 使用项目用户命令
- 创建样本项目层次结构

您应该花些时间来阅读 *ADTS/400: Application Development Manager Introduction and Planning Guide* 一节，它列示了您在开始定义项目中的组之前应该考虑的一些事项。

创建组 (CRTGRP)

使用“创建组”(CRTGRP) 命令来为项目创建组。只能在已创建项目之后才能执行此操作，并且您必须是该项目的项目管理员才能使用此命令。有关详情，参见第7页的『创建项目 (CRTPRJ)』。

当您创建组时，必须为它提供两个名称: 一个用于在所有与组相关的 CL 命令中使用的组，另一个是由五个字符组成的简称。此简称与项目简称组合起来，为项目中的每个组创建一个唯一名称。有关如何命名组的信息，参见附录C. 命名规则。

必须在 PARENT 参数上指定一个值，才能建立项目层次结构中各组之间的关系。当您正在创建项目层次结构中的第一个组时，您必须指定特殊值 *NONE，因为该组没有父组。**父组**就是项目层次结构分支中直接位于另一个组上面的组。当首次创建项目层次结构时，您不应更改提升码值。缺省值为 *PARENT。如果您正在创建项目层次结构中的第二个或后续的组，则必须指定该组处于其下的父组的名称。

编码字符集标识符 (CCSID) 可以与 CCSID 参数上的组相关联。此标识符为组中存储的部件确定字符集标识符。缺省 CCSID 值 *PARENT 使用与父组相关联的 CCSID。如果您正在创建项目层次结构中的第一个组，则与作业相关联的 CCSID 将充当缺省值。

可以通过指定 *JOB 来指示要使用与运行 CRTGRP 命令的进程或作业相同的 CCSID，或者通过指定 *HEX 来指示您不想转换源部件中的数据；或者您可以输入一个 1 到 65535 之间的整数值来标识特定的 CCSID。有关在支持多种国家语言的应用程序中如何使用 CCSID 的信息，参见第242页的『了解 CCSID』。

使用 NOTIFY 参数来通知人们: 在项目层次结构的一个分支中存在的某个部件正在被 CHKOUTPART 命令检出到同一项目层次结构的另一分支中的某个组中。有关此参数的详情，参考第20页的『为组设置通知』。

可以更改组的 PARTLREQ 参数。PARTLREQ 参数指定当在此组中更改部件时是否需要 PARTL 名称。若为特定组指定了 *YES，则当使用部件开发命令且将此组作为目标组时，必须给出原因。

创建组时，可以在 TEXT 参数上描述该组表示项目中的哪个阶段。

项目组和库列表

OS/400 操作系统将库列表的用户部分限制为 25 个库。在项目层次结构中的每个“项目－组”对，作为搜索路径的一部分，都表示一个库。项目层次结构中每个搜索路径中的组数受库列表限制，一定不能超过 25 个。但是，您不太可能在一个搜索路径中需要多于 10 个组。有关如何组合项目名和组名来形成唯一库名的信息，参见附录C. 命名规则。

创建项目层次结构

根据各个组在项目层次结构中的位置来进一步将组分类。**根组**在任何层次结构中都是第一个组或者最顶部的组。每个项目层次结构都只能有一个根组。通常，根组就是包含最先完成、更新和测试的应用程序版本的组。根组通常还是您将现存的应用程序导入至的组。

示例

此示例说明如何在项目 PAYROLL 中创建组 MASTER。图7 表示要创建的层次结构。在 PARENT 参数上指定 *NONE，以便向“应用程序开发管理器”功能部件指示 MASTER 是根组。

使用 CRTGRP 命令

```
CRTGRP PRJ(PAYROLL) GRP(MASTER) SHORTGRP(MST) PARENT(*NONE)
       TEXT('MASTER GROUP IN PROJECT PAYROLL')
```

使用“编程开发管理器”实用程序

在“用 PDM 来使用组”屏幕上按“F6=创建”。

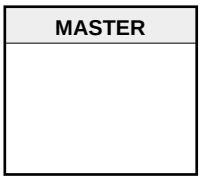

图 7. 具有组 MASTER 的项目 PAYROLL

要在项目层次结构中创建第二个层，可在下一个组的 PARENT 参数上指定 MASTER。

示例

以下命令将在项目 PAYROLL 中创建组 TEST，如图8中所述。

```
CRTGRP PRJ(PAYROLL) GRP(TEST) SHORTGRP(TST) PARENT(MASTER)
       TEXT('TEST GROUP IN PROJECT PAYROLL')
```

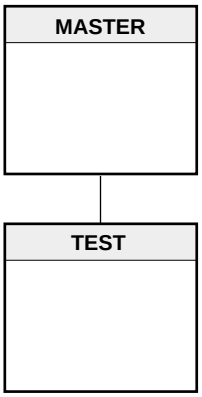

图 8. 具有两个组的项目 PAYROLL

因为项目层次结构具有组 MASTER 和 TEST，所以您需要定义一些组，使得开发者可以在其中执行部件开发任务。**开发组**就是项目层次结构中完成部件开发的组。

开发活动（如创建部件、检入或检出部件，以及更改或提升部件等）最好应该在项目层次结构的底部或者层次最低的组中进行。不应为开发者提供对诸如 TEST 和 MASTER 之类的组的更新访问权。只有在更改和测试部件之后，才应在层次结构中提升它们，以便与从其他组中提升的部件一起再次进行测试，以确保所有组件在一起正确地工作。因此，项目层次结构中较高层次的部件要比项目层次结构中较低层次的组中那些

部件稳定。其他注意事项是，如果开发者对项目层次结构的中间组中部件的副本进行了更改，则若另一开发者从更低层次的组提升同一部件，那么可以替换这样的更改。仅当绝对必要时才应进行这种更改，并且，如果要保存更改，则还应相应地更改层次结构中较低层次的相同部件的副本。

您可能想对这样清楚地标识它们的开发组指定名称。这些名称可以是在该组中工作的人员的用户简要表名，或者是更普通的名称，如 DEVELOPER1 或 DEVELOPER2。

示例

下列命令将在项目 PAYROLL 中创建两个开发组，开发者可以在其中进行部件开发活动。

```
CRTGRP PRJ(PAYROLL) GRP(DEVELOPER1) SHORTGRP(DEV1) PARENT(TEST)
       TEXT('DEVELOPER1 GROUP IN PROJECT PAYROLL')

CRTGRP PRJ(PAYROLL) GRP(DEVELOPER2) SHORTGRP(DEV2) PARENT(TEST)
       TEXT('DEVELOPER2 GROUP IN PROJECT PAYROLL')
```

图9表示项目 PAYROLL 具有四个组，如所出现的那样。

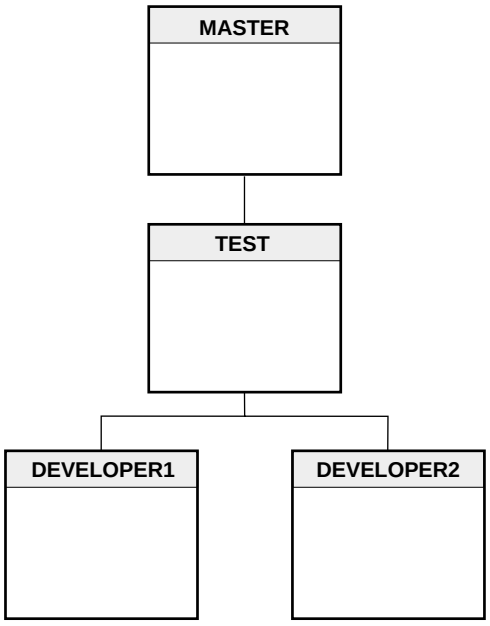

图 9. 具有四个组的 Payroll 项目

如果您想确保应用程序被构建或者准备好导出至 TEST 组，则您可以在 TEST 组与开发组之间创建一个组，例如，该组用来包含开发工作已完成的那些部件。可以将此组称为 INTEGRATION。创建此组的目的是在保留应用程序测试版本的同时，允许开发者从此组中检出部件，或者将部件提升到此组中。

为项目层次结构定义一个提升码

由于部件是在项目层次结构中开发的，因此，沿提升路径一次只能将一个部件组从层次较低的组提升至层次较高的组。**提升路径**是在包含必须提升的部件的组与对部件的工作完成时最终将包含部件的组之间的各个组的排列。

项目层次结构的提升路径是由您在创建组时在 PRMCODE 参数上指定的提升码确定的。**提升码**标识可以将部件提升至的组。此组称为**目标组**。提升路径控制部件在项目层次结构中可被提升的程度。

当您创建根组但未在 PRMCODE 参数上指定值时，该组的提升码将缺省设置为组名。“应用程序开发管理器”功能部件通过 PARENT 参数上的值确定您正在创建的组是否是根组。

当您创建不是根组的组时，PRMCODE 参数的缺省值将导致新组与在 PARENT 参数上指定的组具有相同的提升码。

项目 PAYROLL 的一个提升路径是通过接受 CRTGRP 命令的 PRMCODE 参数的缺省值来创建的。在此示例中，在开发组中开发的部件最终可被提升至组 MASTER，一次提升一层。您对您正在从其提升的组必须具有更新访问权。

可以指定要其与 CRTGRP 命令上的组相关联的提升码的名称。有关如何命名提升码的信息，参见附录C. 命名规则。在下一个示例中，下列任一命令将创建提升码为 MASTER 的组 TEST。

示例

在第一个示例中，使用的是缺省提升码，而在第二个示例中，指定了提升码。因为父组的提升码是 MASTER，所以 TEST 组的提升码也是 MASTER。

```
CRTGRP PRJ(PAYROLL) GRP(TEST) SHORTGRP(TST) PARENT(MASTER)
       TEXT('TEST GROUP IN PROJECT PAYROLL')

CRTGRP PRJ(PAYROLL) GRP(TEST) SHORTGRP(TST) PARENT(MASTER)
       PRMCODE(MASTER) TEXT('TEST GROUP IN PROJECT PAYROLL')
```

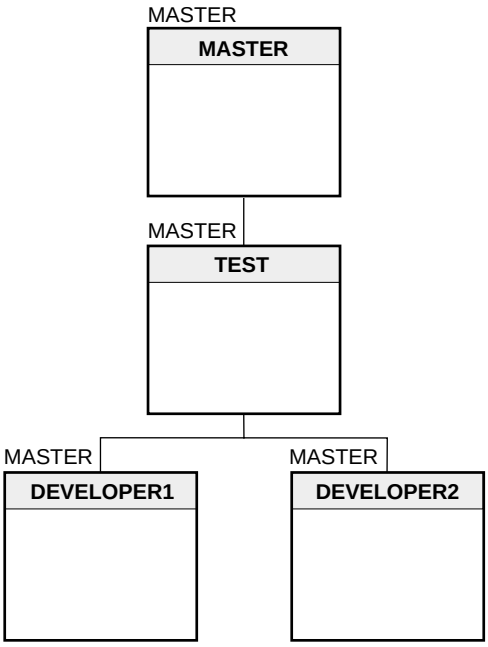

图 10. 具有提升码 MASTER 的 Payroll 项目

如果在项目层次结构中只有一个应用程序版本，或者如果您正在开始新的应用程序开发活动，则项目只有一个提升码是有效的。如果您的组织必须维护一个现存应用程序版本，同时更新同一应用程序的另一版本，则在项目层次结构内可能需要几个提升码。

构造项目层次结构

项目层次结构在一段时间之后可更改以便适应数个应用程序版本的开发工作。例如，当第一个应用程序版本的工作完成时，就可以开始开发应用程序的第二个版本。而且，可能需要第一个版本的开发工作来维护应用程序。图11 显示了这样的一个项目层次结构。

当第一个版本的工作完成时，项目层次结构的 V1 分支中的部件被提升至组 V1MASTER。V1FIX 组是为可能必需对组 V1MASTER 中的部件作的修正而创建的。

现在可以开始开发应用程序的第二个版本了。项目管理员向项目层次结构添加新的分支，该层次结构由组 V2MASTER、V2TEST、DEVELOPER3 和 DEVELOPER4 组成，所有组的提升码都为 V2。

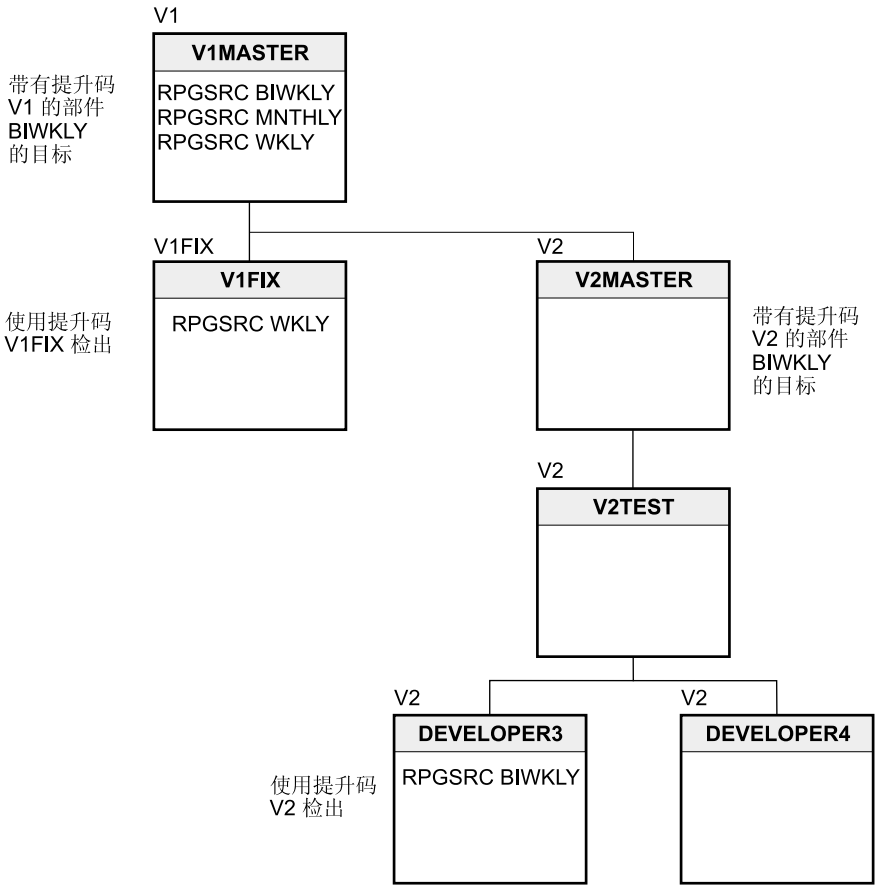

图 11. 具有三个提升码的 Payroll 项目

提升码

定义一个项目层次结构内的几个提升码与定义一个提升码的方法是相同的。可以在 CRTGRP 命令上使用 PRMCODE 参数来为组定义提升码。项目层次结构中各个组的提升码的组合就构成了提升路径。

示例: 下列 CL 命令有时用来创建一个具有六个组和三个提升码的项目层次结构，如第18页的图11中所示。在此示例中，假定已经使用 CRTPRJ 命令创建了项目。此项目层次结构假定第一个应用程序版本的工作已完成，且已将所有部件提升至组 V1MASTER。此层次结构保留第一个应用程序版本，而同时允许通过检出需要更改至 V1FIX 组中的部件来对此版本进行修正。它还允许更改应用程序的第二个版本。

```
CRTGRP PRJ(PAYROLL) GRP(V1MASTER) SHORTGRP(V1MST) PARENT(*NONE)
       TEXT('VERSION 1 GROUP IN PROJECT PAYROLL')

CRTGRP PRJ(PAYROLL) GRP(V1FIX) SHORTGRP(V1F) PARENT(V1MASTER) PRMCODE(V1FIX)
       TEXT('VERSION 1 FIX GROUP IN PROJECT PAYROLL')

CRTGRP PRJ(PAYROLL) GRP(V2MASTER) SHORTGRP(V2MST) PARENT(V1MASTER) PRMCODE(V2)
       TEXT('VERSION 2 GROUP IN PROJECT PAYROLL')

CRTGRP PRJ(PAYROLL) GRP(V2TEST) SHORTGRP(V2T) PARENT(V2MASTER)
       TEXT('VERSION 2 TEST GROUP IN PROJECT PAYROLL')

CRTGRP PRJ(PAYROLL) GRP(DEVELOPER3) SHORTGRP(DEV3) PARENT(V2TEST)
       TEXT('DEVELOPMENT 4 GROUP FOR VERSION 2')

CRTGRP PRJ(PAYROLL) GRP(DEVELOPER4) SHORTGRP(DEV4) PARENT(V2TEST)
       TEXT('DEVELOPMENT 5 GROUP FOR VERSION 2')
```

有三条提升路径: 一条路径使用 V1 提升码，第二条路径使用 V1FIX 提升码，第三条路径使用 V2 提升码。

项目层次结构中的每个组都只能有一个提升码与它相关联。项目可以具有许多个提升码。但是，当您创建组时，为该组指定的提升码一定不能破坏项目层次结构中较低层次的任何部件（其目标组在您正在创建的新组上面）的现存提升路径。例如，在第18页的图11中，不能在 V2TEST 组与较低层次的组之间创建一个提升码为 V1 的组（如果在那些组中有部件的提升码为 V2 的话）。这样做会破坏这些部件的 V2 提升路径。

如在第17页的图10中那样，在第18页的图11中用来定位部件的搜索路径是由开发者正在其中工作的组来确定的。例如，如果开发者正在组 DEVELOPER3 中工作，则将首先在该组中搜索部件。如果在该组中找不到所要的部件，则会搜索组 V2TEST。如果在该组中仍找不到该部件，则会搜索组 V2MASTER。如果在该组中还找不到部件，则会搜索组 V1MASTER。有了此搜索路径，在组 DEVELOPER3 中工作的开发者就*看不到*在组 DEVELOPER4 和 V1FIX 中的部件。

一个部件的两个版本

在第18页的图11中，PAYROLL 项目有三个部件。其中一个部件 BIWKLY 具有两个版本，因为它正在被添加新功能。这三个部件是 BIWKLY、MNTHLY 和 WKLY。假定已完成 Payroll 应用程序的第一个版本；但是，在这三个部件中发现了错误，需要进行一些附加的开发活动。部件 BIWKLY、MNTHLY 和 WKLY 被检出到组 V1FIX（其提升码为 V1FIX）中。

同时，开发应用程序下一个版本的开发者确定在部件 BIWKLY 中需要某些新功能。此开发者将 BIWKLY 检出到提升码为 V2 的组 DEVELOPER3 中。注意，不能将 BIWKLY 从组 V2MASTER 提升到组 V1，因为提升码不相同。

当在组 DEVELOPER3 中工作的开发者完成 BIWKLY 的工作时，部件被提升至组 V2TEST。在确保部件如预期般的运作后，管理员就会将该部件提升至组 V2MASTER。

现在，当构建了 Payroll 应用程序的第二个版本时，将使用组 V2MASTER 中的 BIWKLY 版本，而不是使用组 V1MASTER 中的版本。在第二个版本中，部件 MNTHLY 和 WKLY 未更改，因此，将使用在组 V1MASTER 中找到的这些部件的副本。

创建一个没有提升码的组

也可以创建一个提升码为 *NONE 的组。这意味着该组没有与它相关联的提升码。不能提升对提升码为 *NONE 的组创建的部件或复制至该组的部件。

可以为指定为测试应用程序的人员创建一个具有 PRMCODE(*NONE) 的组。然后，此测试人员仍然可以检出部件或者复制部件、更改部件并测试它，以及检入回部件，但是不能将已更改的部件提升回其在项目层次结构中的原来位置。

为组设置通知

当为诸如 V1MASTER 之类的组设置 NOTIFY 参数时，就会发出通知消息（例如，如果当前正在层次结构的 V2 分支上开发一个已被检出至 V1FIX 的部件的话）。如果部件已更改，则可能需要对该部件的多个版本进行更改。例如，当创建组时，可以使用 NOTIFY 参数来忠告用户：存在于项目层次结构的一个分支中的部件正在被 CHKOUTPART 命令检出到同一项目层次结构另一分支中的组中。如果部件已更改，则可能需要对该部件的其他版本进行同样的更改。

当指定了值 *DEVELOPER 或者特殊名称时，从在 CHKOUTPART 命令的 GRP 参数上指定的组中检出的任何部件都将导致发出如上所述的通知。将对其提升码与所检出部件不同的、且存在于项目层次结构其他分支的组中的特定部件和类型的所有版本发出通知消息。如果现存部件具有提升码 *NONE，则会发出通知，而不论用来检出部件的提升码是什么。

不会对存在于（从其中检出部件或者将部件检出到其中的）组中的同一部件的副本发出通知消息，或者如果部件是构建过程的输出（例如，类型为 PGM的部件），也不会发出通知消息。通常，项目管理员使用缺省值 NOTIFY(*NONE)，它意味着不应发出任何通知。

当完成第一个应用程序版本的所有开发工作，且所有部件都在组 V1MASTER 中时，项目管理员可为组 V1MASTER 更改 CHGGRP 命令上的 NOTIFY 参数，以便指定当检出该组中的部件时应该通知谁。参见第21页的图12。这可以帮助监控为应用程序的下一版本检出的部件以及为对应用程序当前版本的修正所检出的部件。

当从组 V1MASTER 中检出部件时，将发出下列类型之一的通知:

- 如果指定了值 *DEVELOPER，则在项目层次结构另一分支中检出了部件的开发者将接收到通知信息。检出该部件的开发者将接收到相同的消息。
- 如果指定了用户简要表名，则将通知用户简要表。检出该部件的开发者将接收到相同的消息。在项目层次结构另一分支中检出了部件的任何人也会接收到消息。

考虑下图，它说明了应用程序的三个版本。应用程序第一个版本的开发已完成，且所有部件都在组 V1MASTER 中。

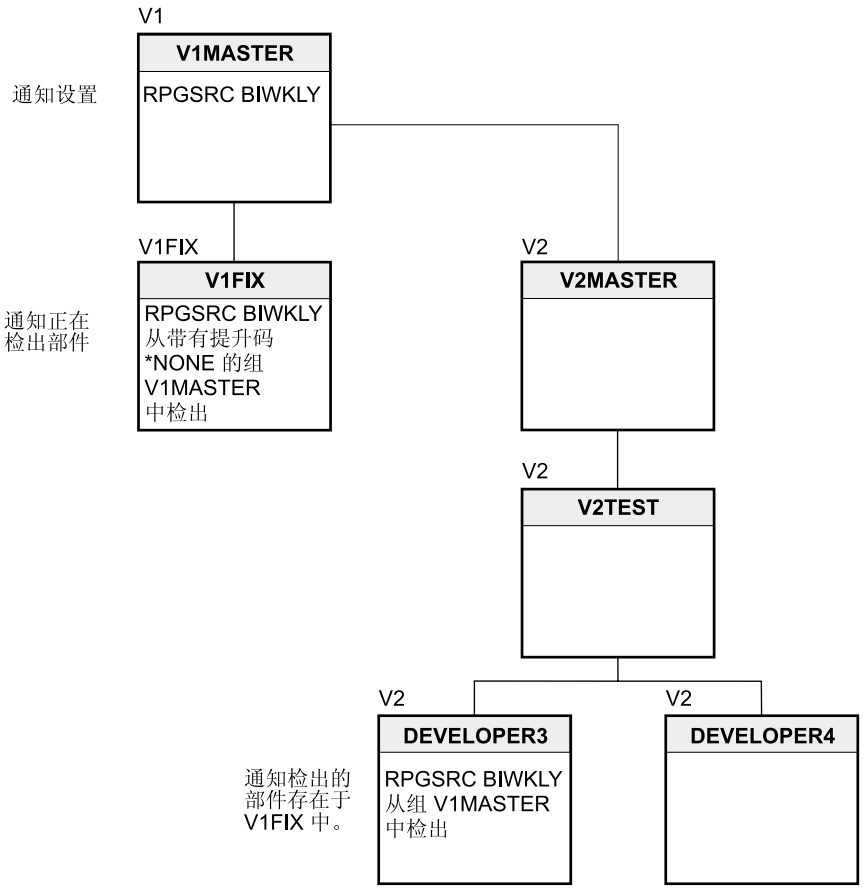

图 12. 通知示例

1. 项目管理员使用 CHGGRP 命令，并为组 V1MASTER 将 NOTIFY 参数设置为 *DEVELOPER。从此组中检出的任何部件现在都会导致发出通知消息。正在检出该部件的开发者将接收到相同的消息。
2. 组 V1FIX 中的开发者将部件 RPGSRC BIWKLY 从提升码为 *NONE 的组 V1MASTER 中检出。

   不会发出通知，因为存在的部件 RPGSRC BIWKLY 的唯一副本位于从其中检出部件的组 (V1MASTER) 以及将部件检出至的组 (V1FIX) 中。
3. 组 DEVELOPER3 中的开发者将部件 RPGSRC BIWKLY 从组 V1MASTER 中检出。现在，部件存在于三个组中: V1MASTER、V1FIX 和 DEVELOPER3。
4. 在组 V1FIX 中检出部件的开发者接收到通知消息，指示部件正在被检出到组 DEVELOPER3 中。正在检出部件的开发者接收到通知消息，指示部件已经被检出到组 V1FIX 中。
5. 进行修正工作的开发者在完成修正时就可以通知在组 DEVELOPER3 中检出部件的开发者了。该开发者可使用 CMPPART 命令来比较 DEVELOPER3 组和 V1FIX 组中的部件的两个版本。然后，如果需要的话，开发者可以使用 MRGPART 命令来合并修正。

如果部件仅存在于从中检出该部件的组或者存在于将它检出至的组中，则不会发出通知消息。

使用组

可以通过直接在任何命令行上输入 CL 命令来使用组。还可以在 CL 程序、REXX 过程中使用命令，或者通过 QCMDEXC 来使用命令。组命令为 CRTGRP、CHGGRP 和 DLTGRP。

除了这些组命令之外，还可以使用 PRTPRJ 命令来显示关于项目中的组的信息。有关 PRTPRJ 命令的信息，参见第9页的『打印关于项目的信息 (PRTPRJ)』。

还可以用“用 PDM 来使用组”屏幕通过“编程开发管理器”实用程序使用组。

更改组 (CHGGRP)

使用“更改组”(CHGGRP) 命令 来更改组的属性。只有项目管理员才能执行此操作。

可以使用此命令来更改与组相关联的文本。还可以使用它来更改组的父组。更改父组将允许您重新安排项目层次结构，或者在项目层次结构中插入新组。

可以更改与特定组相关联的提升码。您指定给该组的新提升码一定不能破坏该组以下组中部件的现存提升路径。

还可以更改当从组中检出部件时将发出的通知。您可以在 NOTIFY 参数上选择下列值之一: *SAME、*NONE、*DEVELOPER 或者用户简要表的名称。*SAME 是缺省值，表示已保存当前的通知设置。有关详情，参见第20页的『为组设置通知』。

PARTLREQ 参数指定当在此组中更改部件时是否 PARTL 名是必需的。若为特定组指定了 *YES，则当使用部件开发命令且将此组作为目标组时，必须给出原因。

更改项目层次结构中的组

第23页的图13说明当添加新组时，项目层次结构将如何更改。

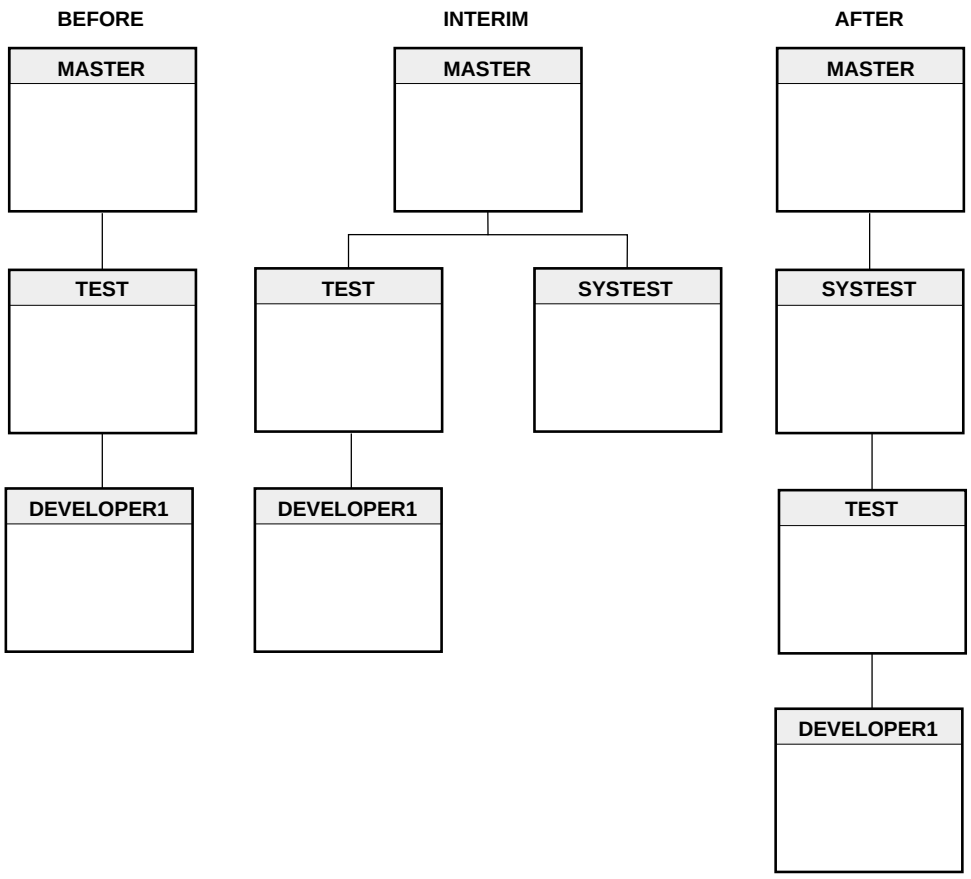

图 13. 更改项目层次结构

BEFORE 的项目层次结构有三个组。假定您想更改项目层次结构，以便在组 MASTER 与现存组 TEST 之间有一个新的“系统测试”组。第一步就是创建该新组。

示例: 以下 CL 命令将添加新组并创建 INTERIM 层次结构。

```
CRTGRP PRJ(PAYROLL) GRP(SYSTEST) SHORTGRP(SYST) PARENT(MASTER)
       TEXT('SYSTEM TEST GROUP IN PROJECT PAYROLL')
```

下一步就是将组 TEST 的父组从组 MASTER 更改为组 SYSTEST。

示例: 以下 CL 命令将更改组 TEST 的父组，以使项目层次结构看起来象图13中的“之后”项目层次结构。

使用 CHGGRP 命令

```
CHGGRP PRJ(PAYROLL) GRP(TEST) PARENT(SYSTEST)
```

使用“编程开发管理器”实用程序

在“用 PDM 来使用组”屏幕上选择选项 2（更改）。

如果您想通过创建新的根组来更改项目层次结构，只需如创建此项目中的任何其他组那样来创建该组，并在 CRTGRP 命令上指定 PARENT(*NONE) 即可。新的根组就成为先前根组的父组。

当一个组被添加至项目层次结构时，“应用程序开发管理器”功能部件将确保提升路径上层次较低的组中所有部件的提升路径不会受到破坏。如果您试图将组添加至项目层次结构，则将导致沿着“检出部件”(CHKOUTPART) 命令上所指示的另一提升路径来提升部件，并且 CHGGRP 命令将失败。

为了避免可能会破坏提升路径，您在将组插入项目层次结构中之前，应该要求所有开发者停止检入和检出部件。将所有部件都提升至它们的目标组也是一个好想法。

注意，一个组不能是它自己的父组。既不能将层次结构中较低层次的组标识为父组，也不能将层次结构中较高层次的组标识为父组。

更改项目层次结构中的提升路径

提升码必须表示现存项目层次结构中的有效提升路径。当为项目层次结构更改了提升路径时，此功能部件将检查是否不会阻止部件被提升至其目标组。

示例: 以下 CL 命令将更改项目 PAYROLL 中的组 DEVELOPER1 的提升码，导致项目如图14中所示。

```
CHGGRP PRJ(PAYROLL) GRP(DEVELOPER1) PRMCODE(*NONE)
```

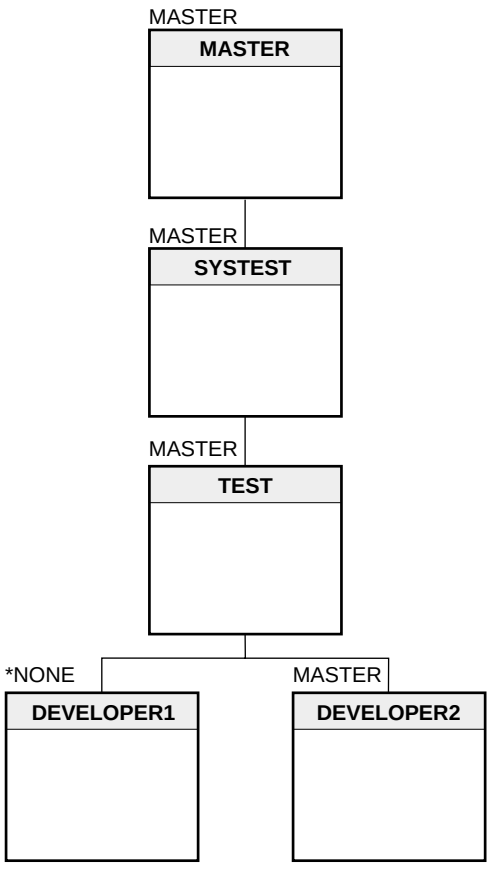

图 14. 具有已更改提升码的项目 PAYROLL

在此示例中，更改了与组 DEVELOPER1 相关联的提升码，使得在该组中工作的人员可以不再提升部件。如果输入 CHGGRP 命令时此组中的部件具有提升码 MASTER，则该命令将失败，因为部件不能再被提升回到目标组中。

如果更改了现存项目层次结构中组的提升码，则此功能部件还会确保该组中的部件不获取其提升路径上的另一目标组。例如，如果使用 CHGGRP 命令更改了图15中所说明的项目层次结构，以便为组 MASTER 指定 V1 提升码，则部件 BIWKLY 的提升路径被扩展，因为该部件将具有新的目标组。在此情况下，CHGGRP 命令将失败。

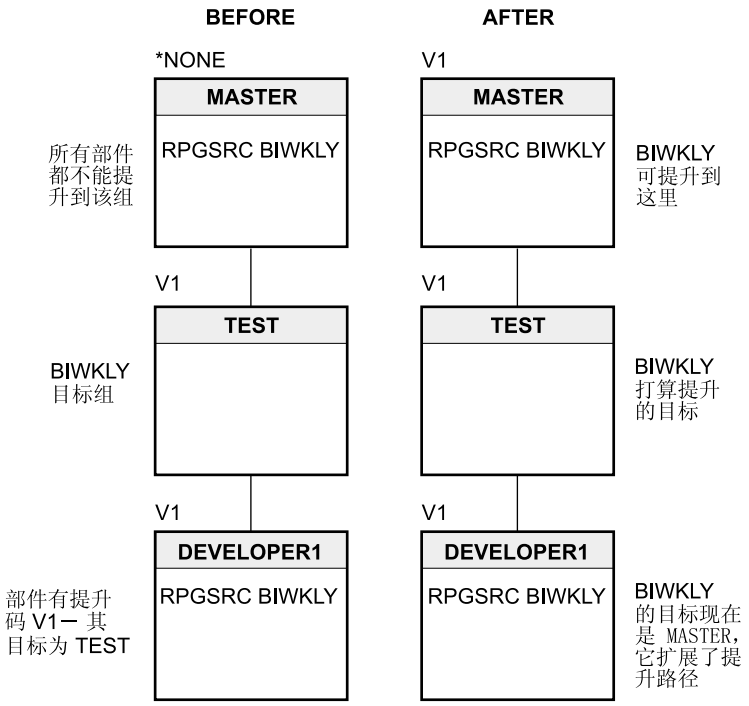

图 15. 显示为何不能扩展提升路径的一个示例

删除组 (DLTGRP)

使用“删除组”(DLTGRP) 命令来从项目层次结构中删除组，并删除与该组相关联的权限信息。只有项目管理员才能执行此操作。

在某人离开项目之后，您可以使用此命令来除去特定开发者的组，或者除去与该项目的旧版本相关联的某些组。例如，如果您正在进行第三个版本的工作，您可能不再需要项目第一个版本中的开发组。

注

当删除组时，归档库也会被删除。在“应用程序开发管理器”功能部件外部创建的库也会被删除（如果它的名称是有效归档库名的话）。仅当归档库和组库的所有者是 QPRJOWN 时，归档库和组库才会被删除，否则，会对被删除的组发出警告。有关有效归档库名的描述，参见第82页的『归档部件』。

示例

此示例说明如何从项目 PAYROLL 中删除组 DEVELOPER1。

使用 DLTGRP 命令

```
DLTGRP PRJ(PAYROLL) GRP(DEVELOPER1)
```

使用“编程开发管理器”实用程序

在“用 PDM 来使用组”屏幕上选择选项 4（删除）。

当您使用选项 4 时，“编程开发管理器”实用程序会提供一个确认屏幕。如果直接从命令行使用此命令，就不会向您发出警告。

当删除组时，DLTGRP 命令将检查下列情况:

1. 在项目层次结构底部删除的组中一定不能有提升码不是 *NONE 的任何部件。如果某个组包含的所有部件都具有提升码 *NONE，且没有任何一个部件被检出，则可以删除这样的组。
2. 在项目层次结构中的其他地方，下列句子必须为真。
   - 在该组中根本没有部件。
   - 在项目层次结构中没有任何部件使用此组来访问其目标组。
   - 如果组是根组，则它必须是空的。它必须是层次结构中唯一的组，或者它下面必须只有一个组成为新的根组。

如果您试图从开发组中删除部件，或者删除一个包含尚未提升至其目标的部件的组，则命令将失败。

如果使用选项 4 来输入 DLTGRP 命令，则“编程开发管理器”实用程序会提供一个确认屏幕。如果直接从命令行使用此命令，则不会接收到警告。

在项目层次结构中登记用户

一旦创建了项目层次结构，就需要给予开发者或管理员对它的访问权。项目管理员登记项目与组中的开发者和其他项目管理员。

您应该花些时间来阅读 *ADTS/400: Application Development Manager Introduction and Planning Guide* 一节，它列示了您在开始登记用户之前应该考虑的一些事项。

登记开发者和项目管理员 (ADDPRJUSR)

使用“添加项目用户”(ADDPRJUSR) 命令 来在项目中登记其他用户。只有项目管理员才能执行此操作。

用户简要表用来标识您想登记的人员。应在 ADDPRJUSR 命令的 USRPRF 参数上指定用户简要表名。用户简要表 QSECOFR 是所有项目的管理员。

可以通过记下在 USRTYPE 参数上提供的值来确定所登记用户的类型。例如，用户类型 *DEVELOPER 指示您正在登记的人员是一个应用程序开发者，且对项目层次结构中的所有组都具有读访问权。

通过在 ACCESS 参数上指定 *UPDATE，并在 DEVELOPGRP 参数上输入组名，来对特定开发组授予更新访问权。更新访问权表示开发者可以从他们的开发组提升部件。

这些参数一起使用来指示所登记开发者需要对指定的组具有更新访问权。当指定了 ACCESS(*READ) 时，DEVELOPGRP 参数就被忽略。

示例

此示例说明如何给予开发者对项目层次结构中所有部件的读访问权以及对一个开发组中所有部件的更新访问权。

使用 ADDPRJUSR 命令

```
ADDPRJUSR PRJ(PAYROLL) USRPF(SMITH) USRTYPE(*DEVELOPER) ACCESS(*UPDATE)
          DEVELOPGRP(DEVELOPER1)
```

使用“编程开发管理器”实用程序

在“用 PDM 来使用项目”屏幕上选择选项 41（添加用户）。

如果未在 DEVELOPGRP 参数上标识任何组，则开发者不能检入或检出部件或者提升项目层次结构中的部件。

示例

以下命令仅给予开发者对项目层次结构中所有部件的读访问权。

```
ADDPRJUSR PRJ(PAYROLL) USRPRF(SMITH) USRTYPE(*DEVELOPER) ACCESS(*READ)
```

用户类型 *ADMIN 指示您正在登记的人员是一个项目管理员，他对项目层次结构中的所有组都具有更新访问权，并且可以使用所有“应用程序开发管理器”命令。用户简要表 QSECOFR 是所有项目的管理员。

示例

以下命令说明如何登记项目管理员。

```
ADDPRJUSR PRJ(PAYROLL) USRPRF(TREMBLAY) USRTYPE(*ADMIN)
```

如果您指定 USRTYPE(*ADMIN)，则 ACCESS 和 DEVELOPGRP 参数会被忽略，因为隐含了对项目层次结构中所有组的更新访问权。

有关如何在项目中登记了开发者或管理员之后更改用户简要表的登记信息的信息，参见第29页的『更改用户登记信息 (CHGPRJUSR)』。

使用项目用户命令

可以通过直接在任何命令行上输入 CL 命令来使用项目用户的用户简要表。还可以在 CL 程序、REXX 过程中使用命令，或者通过 QCMDEXC 来使用命令。项目用户命令是 ADDPRJUSR、PRTPRJUSR、CHGPRJUSR 和 RMVPRJUSR。必须是项目管理员才能使用 ADDPRJUSR、GHGPRJUSR 和 RMVPRJUSR 命令。

所有项目用户命令都使用用户简要表。必须为每个命令提供用户简要表。

通过使用“用 PDM 来使用项目”屏幕，还可以用“编程开发管理器”实用程序来使用项目用户。

打印用户登记信息 (PRTPRJUSR)

使用“打印项目用户”(PRTPRJUSR) 命令来查看项目的特定用户简要表的登记信息。项目管理员和开发者都可以执行此命令。

必须在 PRJ 参数上指定您想显示关于哪个项目的用户登记信息。您可以在 USRPRF 参数上输入特定的用户简要表名或者使用 *ALL 来指示您想查看在项目中登记的所有人员的所有登记信息。

在最后一个 ADDPRJUSR 或 CHGPRJUSR 命令上授予的访问级别与用户类型值一起用来创建报告。如果您在此命令上指定的用户简要表是用于对某些组具有更新访问权的开发者的，则会列示他／她对其具有更新访问权的组。如果开发者对项目只有读访问权，或者如果您指定的用户简要表用于管理员的，则会在报告上打印 ALL 来代替组列表。

示例: OUTPUT 参数的缺省值 *PRINT 将使报告假脱机以供打印。如果您输入以下命令，则图16中显示的报告被假脱机。

使用 PRTPRJUSR 命令

```
PRTPRJUSR PRJ(PAYROLL) USRPRF(*ALL) OUTPUT(*PRINT)
```

使用“编程开发管理器”实用程序

在“用 PDM 来使用项目”屏幕上选择选项 46（打印用户）。

```
5722WDS   V5R1M0          应用程序开发管理器      - 打印项目用户   05/08/01     17:30:15          页面 . . :  0001

------------------------------------------------------------------------------

项目    . . . . . . . . . . . . . . . :   PAYROLL
用户  . . . . . . . . . . . . . . . . :   TREMBLAY
用户类型  . . . . . . . . . . . . . . :   *ADMIN
项目访问级别. . . . . . . . . . . . . :   *ALL


已授权的组
*** ALL ***
------------------------------------------------------------------------------

项目    . . . . . . . . . . . . . . . :   PAYROLL
用户  . . . . . . . . . . . . . . . . :   SMITH
用户类型  . . . . . . . . . . . . . . :   *ADMIN
项目访问级别. . . . . . . . . . . . . :   *ALL


已授权的组
*** ALL ***
------------------------------------------------------------------------------

项目    . . . . . . . . . . . . . . . :   PAYROLL
用户  . . . . . . . . . . . . . . . . :   ROBERTS
用户类型  . . . . . . . . . . . . . . :   *DEVELOPER
项目访问级别. . . . . . . . . . . . . :   *UPDATE


已授权的组
DEVELOPER1
------------------------------------------------------------------------------

项目    . . . . . . . . . . . . . . . :   PAYROLL
用户  . . . . . . . . . . . . . . . . :   LALONDE
用户类型  . . . . . . . . . . . . . . :   *DEVELOPER
项目访问级别. . . . . . . . . . . . . :   *READ


已授权的组
*** ALL ***
                  * * * * * 列 表 结 束  * * * * *
```

图 16. 由 PRTPRJUSR 命令生成的假脱机文件的样本

示例: 或者，可以通过指定以下命令来将输出导向输出文件。库列表用来确定在何处存储称为 FILE1 的输出文件，而文件中的第一个成员将接收输出。

```
PRTPRJUSR PRJ(PAYROLL) USRPRF(*ALL) OUTPUT(*OUTFILE)
          OUTFILE(*LIBL/FILE1) OUTMBR(*FIRST)
```

输出文件的记录格式与在库 QADM 中由系统提供的数据库文件 QALYPUSR 中所使用的记录格式相同。

如果您在此命令上指定的用户简要表是用于开发者的，则为开发者对其具有更新访问权的每个组创建一个输出文件记录。如果您是为管理员而指定的用户简要表，则仅创建一个输出文件记录。

更改用户登记信息 (CHGPRJUSR)

使用“更改项目用户”(CHGPRJUSR) 命令来更改项目的特定用户简要表登记信息。借助此命令，可以更改用户类型 (USRTYPE)、访问权限 (ACCESS) 或者开发者对其具有更新访问权的组 (DEVELOPGRP)。必须确保您更改其登记信息的用户没有将任何部件检出到他们的开发组中。只有项目管理员才能执行此操作。

您可以更改当前登记为开发者的某个人的用户简要表（如果该人员成为项目管理员的话），反之亦然。可以通过将开发者的访问权更改为 *READ，可以除去开发者对所有开发组的更新访问权。要从项目中整个地除去开发者或管理员的权限，可使用 RMVPRJUSR 命令。您在 DEVELOPGRP 参数上输入的组名充当开发者先前具有权限的组的替换列表。

示例: 假定先前输入了以下命令。

```
ADDPRJUSR PRJ(PAYROLL) USRPRF(ROBERTS) USRTYPE(*DEVELOPER)
          ACCESS(*UPDATE) DEVELOPGRP(DEVELOPER1)
```

下一个示例说明如何从组 DEVELOPER1 到组 DEVELOPER2 来更改开发者 ROBERTS 的更新访问权。

使用 CHGPRJUSR 命令

```
CHGPRJUSR PRJ(PAYROLL) USRPRF(ROBERTS) USRTYPE(*SAME)
          ACCESS(*SAME) DEVELOPGRP(DEVELOPER2)
```

使用“编程开发管理器”实用程序

在“用 PDM 来使用项目”屏幕上选择选项 42（更改用户）。

为允许开发者更新另一个开发组中的部件，您必须在 CHGPRJUSR 命令上指定所有组。例如，要为 ROBERTS 授予对 DEVELOPER1 和 DEVELOPER2 组的更新访问权，应输入以下命令。

```
CHGPRJUSR PRJ(PAYROLL) USRPRF(ROBERTS) USRTYPE(*SAME)
          ACCESS(*SAME) DEVELOPGRP(DEVELOPER1 DEVELOPER2)
```

如果现存的组列表太长，则首先按“F4=提示”以提示命令，并且在更改当前组列表之前检索它。

除去用户登记信息 (RMVPRJUSR)

使用“除去项目用户”(RMVPRJUSR) 命令来除去项目和该项目中各组的用户简要表登记。用户简要表 QSECOFR 是所有项目的管理员，可以从项目中除去所有项目管理员。只有项目管理员才能执行此操作。

当管理员或开发者离开项目或当您想从项目层次结构中整个地除去用户时，可使用此命令。

示例: 以下命令将从项目 PAYROLL 中除去 ROBERTS。

使用 RMVPRJUSR 命令

```
RMVPRJUSR PRJ(PAYROLL) USRPRF(ROBERTS)
```

使用“编程开发管理器”实用程序

在“用 PDM 来使用项目”屏幕上选择选项 44（除去用户）。

RMVPRJUSR 命令检入回部件，并且如果有任何部件被检出到在命令上指定的用户简要表中，则会发出信息性消息。管理员可以使用这些消息来解决此用户（无论是开发者还是管理员）在使用 RMVPRJUSR 命令之前正在对其进行工作的任何未解决工作项。有关在某个人离开项目时如何清理开发组的信息，参见第184页的『在用户离开项目时进行恢复』。

创建样本项目层次结构

本节描述了一系列命令，您可以用来:

- 创建类似于此处所描述的项目 TRAVEL 的项目
- 在项目层次结构中创建组
- 显示关于项目的信息
- 在其中登记用户
- 更改他们的一些登记信息

还可以使用“编程开发管理器”实用程序提供的屏幕来创建此项目层次结构。在此练习的末尾，您将已经创建了第31页的图17中所说明的项目层次结构。

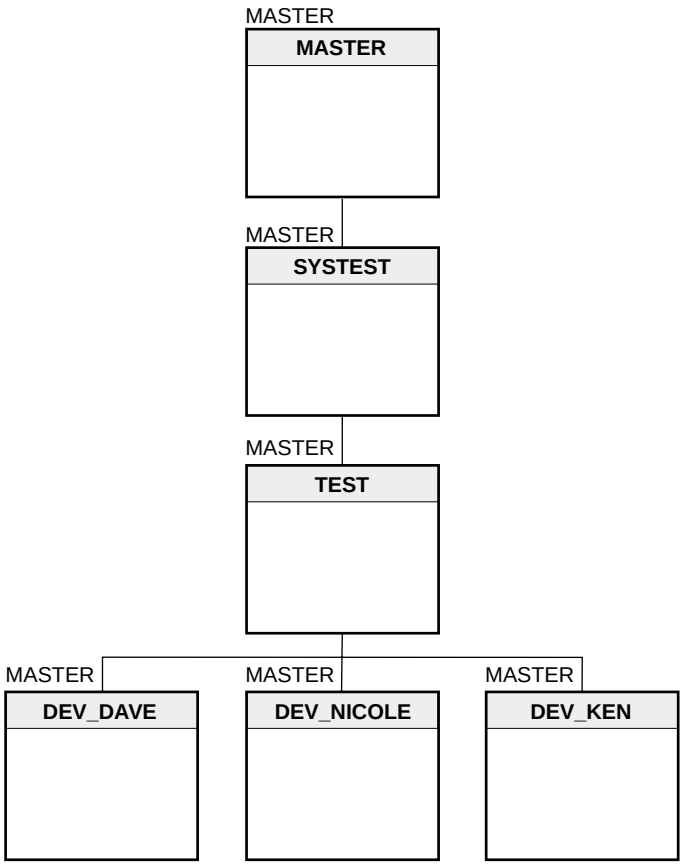

图 17. Travel 应用程序的项目层次结构

使用下列步骤来创建此新的项目层次结构:

1. 使用 CRTPRJ 命令来创建项目。以下命令将创建项目 TRAVEL。如果您愿意的话，可用另一个名称来创建该项目，以便您可以将它保存在系统上供将来练习使用。

   ```
   CRTPRJ PRJ(TRAVEL) SHORTPRJ(TRL) TEXT('TRAVEL DESTINATIONS APPLICATION')
   ```

   有关创建项目的详情，参见第7页的『创建项目 (CRTPRJ)』。
2. 通过使用以下命令来在项目 TRAVEL 中创建一个称为 MASTER 的根组。

   ```
   CRTGRP PRJ(TRAVEL) GRP(MASTER) SHORTGRP(MST) PARENT(*NONE)
          TEXT('MASTER GROUP IN PROJECT TRAVEL')
   ```

   最后，要此组包含应用程序的所有生产级别的部件。有关创建组的详情，参见第14页的『创建组 (CRTGRP)』。
3. 创建一个称为 SYSTEST 的系统测试组。用适当的简短组名和提升码 MASTER 来创建此组。在 TEXT 参数上描述此组。记住，将此组的父组指定为 MASTER。

   重复此步骤来创建组 TEST。此组也具有提升码 MASTER，但是其父组是组 SYSTEST。
4. 现在。您可以再次使用 CRTGRP 来添加您需要的开发组。在此示例中，我们需要三个开发组，分别为 Dave、Nicole 和 Ken。所有这三个组都将 TEST 组作为它们的父组，并且都具有提升码 MASTER。将 Dave 的组称为 DEV_DAVE， Nicole 的组称为 DEV_NICOLE，Ken 的组称为 DEV_KEN。

5. 使用 PRTPRJ 命令来验证项目层次结构是否是要的。有关打印项目信息的详情，参见第9页的『打印关于项目的信息 (PRTPRJ)』。您的报告看起来应该类似于第8页的图3中所显示的报告。如果您对结果感到满意，则转至下一步。如果您想更改项目层次结构的信息，尝试使用 CHGGRP 命令来更改关于其中一个组的文本描述。
6. 现在，需要将开发者登记到项目 TRAVEL 中，以便他们可以开始工作。目前，您想登记所有这三个开发者，使他们只对自己的组具有更新访问权。用 ADDPRJUSR 命令来完成此任务。因为需要有实际的用户简要表来使用此命令，所以选择您的三个同事来分别充当 Dave、Nicole 和 Ken 的部件。有关登记项目用户的详情，参见第26页的『登记开发者和项目管理员 (ADDPRJUSR)』。
7. 在对项目 TRAVEL 登记了 Dave、Nicole 和 Ken 之后，可以使用 PRTPRJUSR 命令来验证他们的登记信息。有关打印这种信息的详情，参见第27页的『打印用户登记信息 (PRTPRJUSR)』。您的报告看起来应该类似于第28页的图16中所显示的报告。注意，您的用户简要表也列示在报告上，原因是您已经创建了项目，因而您也是项目管理员。
8. 您可以决定您需要指定一个候补项目管理员。使用 CHGPRJUSR 命令来完成此操作。如果您需要有关如何执行此操作的详情，参见第29页的『更改用户登记信息 (CHGPRJUSR)』。再次通过使用 PRTPRJUSR 命令来验证您已完成的操作。

当您完成操作时，项目层次结构看起来应类似于第31页的图17中的层次结构。您可能想进一步地实践以获得使用这些命令的更多实践经验。当完成时，记住要使用 DLTPRJ 命令来删除该项目，以便其他人可使用同一示例。要删除项目，输入以下命令:

```
DLTPRJ PRJ(TRAVEL)
```

第4章 部件开发介绍

本章描述:
- 如何列示部件以及如何显示或打印它们的内容。
- 可以用来获得您对其登记的项目名称的命令，以及用来获得在那些项目内您对其具有权限的组名称的命令。这些命令帮助您了解您可以使用哪些部件。

要了解部件名是如何构造的以及与命名部件相关联的规则，参见附录C. 命名规则。

列示项目和组名以及您的访问权

要确定您在哪些项目中进行了登记以及您可以访问哪些组，使用 CL 命令或“编程开发管理器”实用程序。使用“查询项目”(QRYPRJ) 命令来获取您已对其登记的项目的列表。也可以使用 WRKPRJPDM 来查看您对其具有访问权的项目的列表。一旦您知道了项目名，就使用“打印项目用户”(PRTPRJUSR) 命令来打印有关对项目的用户简要表访问权的信息。“打印项目”(PRTPRJ) 命令允许您打印一份显示项目中所有组的报告。还可以使用 WRKGRPPDM 来查看项目中所有组的列表。

在第2章 使用项目和第3章 使用项目层次结构中的组中都对这些命令进行了描述。有关使用“编程开发管理器”实用程序来列示您在其中进行了登记的项目的详情，参考第16章 使用编程开发管理器实用程序。

查看部件列表 (QRYPART)

如果您知道您已对其登记的项目的名称，则您可以获得特定组中部件的列表或者项目中的搜索路径。

使用 QRYPART 命令

```
QRYPART PRJ(PAYROLL) GRP(DEVELOPER1) TYPE(RPGSRC) PART(BIWKLY)
```

使用“编程开发管理器”实用程序

选择 PDM 主菜单上的选项 6（使用部件）。在*项目*和*组*提示处指定项目和组的名称，然后按 Enter 键。将列示从指定的组一直到根组中的所有部件。还可以使用 WRKPARTPDM 来获得您对其具有访问权的组的列表。

必需参数

指定要查询的项目、组、类型和部件。

对于 TYPE 和 PART 参数，您可以指定类属类型和名称、所有类型和名称，或者特定类型和名称。

对于 GRP 参数，您可以指定特定名称或者 *ALL。当指定 GRP(*ALL) 时:
- QRYPART 命令上的 SCAN 和 SCHPTH 参数被忽略，而将搜索项目中的所有组
- “查询部件”报告中会出现重复的部件，因为列示了部件的所有版本

如果您使用类属类型或部件名 (**generic**)，则将查询与它匹配的所有部件和类型。通过使用星号 (*) 或问号 (?) 限定部件名或类型来指定所有部件或类型的子集，其中，* 表示 0 个或多个字符，而 ? 只表示 1 个字符。例如，如果您在 TYPE 参数上指定 *C*，则会处理在部件类型中任何地方具有字符 C 的类型的所有部件，例如 CBLINC 或 CINC。如果您在 PART 参数上指定 ?ABC，则将处理具有以 ABC 结尾的 4 字符名称的所有部件，例如 DABC。有关 QRYPART 命令允许的部件类型的列表，参见附录 B. 部件类型以及它们与命令的关系。QRYPART 命令也允许用户定义部件类型。

因为 *ALL 对于 PART 参数来说具有特殊含义，所以，使用 **ALL 来指定以 ALL 结尾的所有部件。

类属类型和名称在“编程开发管理器”实用程序中是以不同方式处理的。有关详情，参见第16章 使用编程开发管理器实用程序。

可选参数

可以在 QRYPART 命令上指定一些可选信息。您可以指示应将命令的输出发送至何处、是否应该扫描层次结构，以及应该使用哪条搜索路径。

ACCKEY 参数允许您查询检出到特定用户的所有部件。缺省值为 ACCKEY(*ALL)，它将查询所有部件而不管是谁将它们检出的。

使用 OUTPUT 参数来指示应该将输出发送至何处。缺省值为 OUTPUT(*PRINT)。对于此作业，报告被假脱机至打印设备。OUTPUT(*OUTFILE) 将输出导向输出文件。**输出文件**就是一个包含了某些处理结果的文件。输出文件的记录格式与在库 QADM 中由系统提供的数据库文件 QALYQPART 中所使用的记录格式相同。

示例

可以通过指定类似于以下的命令来将输出导向输出文件。

```
QRYPART PRJ(PAYROLL) GRP(DEVELOPER1) TYPE(RPGSRC) PART(BIWKLY)
        OUTPUT(*OUTFILE) OUTFILE(*LIBL/FILE1) OUTMBR(*FIRST *REPLACE)
```

示例

如图18所述，以下命令将在项目 PAYROLL 中查找部件 RPGSRC BIWKLY，首先搜索组 DEVELOPER1。在该组中未找到该部件，因此 QRYPART 命令将沿着缺省搜索路径（在此示例中为 DEVELOPER1、TEST、MASTER）来查找该部件，并在组 MASTER 中找到了它。图19中的报告被假脱机至打印设备。

```
QRYPART PRJ(PAYROLL) GRP(DEVELOPER1) TYPE(RPGSRC) PART(BIWKLY)
        OUTPUT(*PRINT) SCAN(*YES) SCHPTH(*DFT)
```

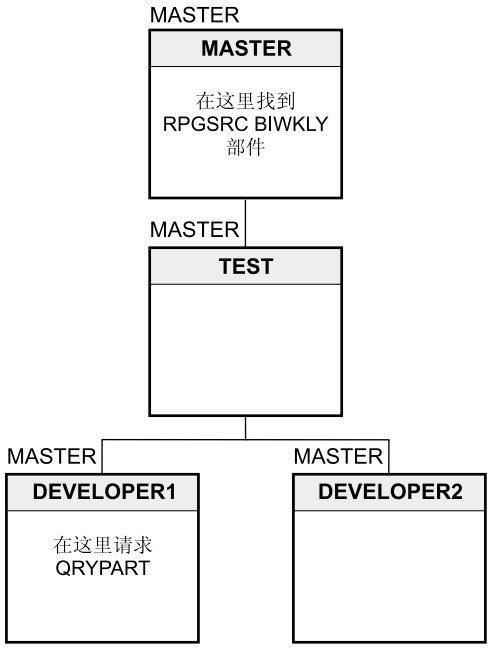

图 18. 查询部件

```
5722WDS   V5R1M0              应用程序开发管理器      - 查询部件      05/08/01    12:00:00          页面 . . :   0001

项目    . . . . . . . . . . . . . . . :   PAYROLL
组    . . . . . . . . . . . . . . . . :   DEVELOPER1
类型  . . . . . . . . . . . . . . . . :   RPGSRC
部件  . . . . . . . . . . . . . . . . :   BIWKLY
访问密钥（持有者）. . . . . . . . . . :   *ALL
扫描层次结构  . . . . . . . . . . . . :   *YES        1
搜索路径. . . . . . . . . . . . . . . :   *DFT        2

                                    满足搜索标准的部件

                                                                                上次更改
部件        类型        语言        组                                          的日期     持有者       文本

BIWKLY      RPGSRC      RPG         MASTER                                      05/08/01     SMITH       更新工资单
1                        3           4                                           5            6           7
                      * * * * * 列 表 结 束  * * * * *
```

图 19. 查询部件报告的样本

1 指定是否要在项目层次结构中搜索指定的部件。

2 在 SCHPTH 参数上指定的搜索路径。

3 部件的语言。

4 在其中找到部件的组。

5 上次更改部件内容的日期。

6 显示检出了部件的人员的用户简要表（如果有的话）。若为空白，则表示未检出部件。

7 显示部件的任何描述性文本。

显示和打印部件 (DSPPART)

使用“显示部件” (DSPPART) 命令来显示或打印部件的内容或特征。表1列示了 DSPPART 命令调用的命令以及各种部件类型。

示例

指定项目、组、类型和部件的名称。还可以选择想要将输出导向何处，以及如果在指定的组中未找到某个部件，是否使用缺省搜索路径来查找该部件。

```
DSPPART PRJ(PAYROLL)
```

TYPE 参数确定被调用来显示或打印部件的命令。

表 1. 被调用来显示给定类型部件的命令

部件类型	由 DSPPART 调用来显示部件的命令
BNDDIR	DSPBNDDIR
CLD	n/a
CMD	DSPCMD
DTAARA	DSPDTAARA
带有语言属性 DSPF 的 FILE	STRSDA（STRSDA 被调用，且 DSPFD 命令被调用来打印它们。）
具有语言属性 ICFF、LF、PF 或 PRTF 的 FILE	DSPFD
JOBD	DSPJOBD
JOBQ	WRKJOBQ
MENU (UIM)	DSPMNUA
MODULE	DSPMOD
MSGF	DSPMSGD
MSGQ	WRKMSGQ
OUTQ	WRKOUTQ
PARTL[1]	DSPPFM
PGM	DSPPGM
PNLGRP	n/a
SCHIDX	n/a
源，包含，搜索路径，构建选项部件，PRDDFN、PRDLOD 和 SYSTEML 部件	STRSEU
SRVPGM	DSPSRVPGM
VRPGBIN 和 VRPGTXT	DSPFD

注:

1. 有关显示类型为 PARTL 的部件的详情，参见第94页的『显示部件列表部件』。

OUTPUT 参数确定应该将输出发送至何处。使用缺省值 OUTPUT(*) 来使输出出现在屏幕上。OUTPUT(*PRINT) 将使输出假脱机至打印设备的输出队列上。

SCAN 参数确定是否使用缺省搜索路径来查找部件（如果未在指定的组中找到该部件的话）。如果您想只搜索在 GRP 参数上指定的组，则使用 SCAN(*NO)。如果您想搜索缺省搜索路径中的组，则使用 SCAN(*YES)。

第5章 创建部件

本章描述如何创建部件。可以创建新部件，也可以将现存部件复制至新部件。

还可以通过将 OS/400 对象或源成员导入到“应用程序开发管理器”控件中来创建部件。在第6章 导入应用程序中对此进行了描述。

部件类型

有几种类别的部件:

- 您可以编译的源
- 输出对象
- 包括源成员
- 对象（如，不是源且不是由编译源生成的数据区或消息文件）
- 包含指定了搜索路径的组列表的搜索路径部件
- 包含编译命令及其参数的列表的构建选项部件
- 包含其他部件的列表的部件列表部件
- 用来封装应用程序的产品定义和产品装入部件

表2 列示了您可以创建的部件的类型。右边的一列显示了部件的相关联语言。此表不包括通过发出 BLDPART、IMPPART 或 CPYPART 命令而创建的由系统提供的部件类型: PGM、FILE、CLD、 CMD、PNLGRP、MENU、MODULE 和 SRVPGM。 第52页的表5 列示了受“应用程序开发管理器”功能部件支持的 OS/400 对象类型。

不能用 CRTPART 命令创建可由 BLDPART 命令创建的那些类型的部件，除非 BLDPART 命令创建的部件是存储在源成员中的部件。

表 2. 部件类型和语言

部件类型	语言
BLDOPT	*NONE
BNDDIR	*NONE
BNDSRC	BND
CBL36INC	CBL36
CBL36SRC	CBL36
CBLINC	CBL, SQLCBL
CBLLEINC	CBLLE, SQLCBLLE
CBLLESRC	CBLLE, SQLCBLLE
CBLSRC	CBL, SQLCBL
CINC	C, CLE, SQLC, SQLCLE
CLDSRC	CLD
CLLESRC	CLLE
CLPSRC	CLP

表 2. 部件类型和语言 (续)

部件类型	语言
CLSRC	CL
CMDSRC	CMD
CSRC	C, CLE, SQLC, SQLCLE
DDSSRC	DSPF, ICFF, LF, PF, PRTF
DSPF36SRC	DSPF36
DTAARA	*NONE
FILE	SAVF
JOBD	*NONE
JOBQ	*NONE
MNU36SRC	MNU36
MSGF	*NONE
MSGF36SRC	MSGF36
MSGQ	*NONE
OCL36SRC	OCL36
OUTQ	*NONE
PARTL	*NONE
PNLINC	PNLGRP
PNLSRC	MENU, PNLGRP
PRDDFN	*NONE
PRDLOD	*NONE
REXXSRC	REXX
RPG36INC	RPG36
RPG36SRC	RPG36
RPGINC	RPG, SQLRPG
RPGLEINC	RPGLE, SQLRPGLE
RPGLESRC	RPGLE, SQLRPGLE
RPGSRC	RPG, SQLRPG
RPT36SRC	RPT36
SCHIDX	*NONE
SCHPTH	*NONE
SRT36SRC	SRT36
SYSTEML	*NONE
TXT36SRC	TXT36
TXTSRC	*NONE
VRPGBIN	VRPG
VRPGTXT	VRPG

创建新部件 (CRTPART)

使用“创建部件”(CRTPART) 命令来在项目层次结构的组中创建新部件。仅当组中尚未存在同名和同类型的部件，且没有具有同名称和同类型的部件可以提升至该组的缺省搜索路径时，才能创建给定名称和类型的部件。如果缺省搜索路径中已经存在名称和类型相同的部件，则当您创建该部件时必须指定提升码 *NONE。考虑图20:

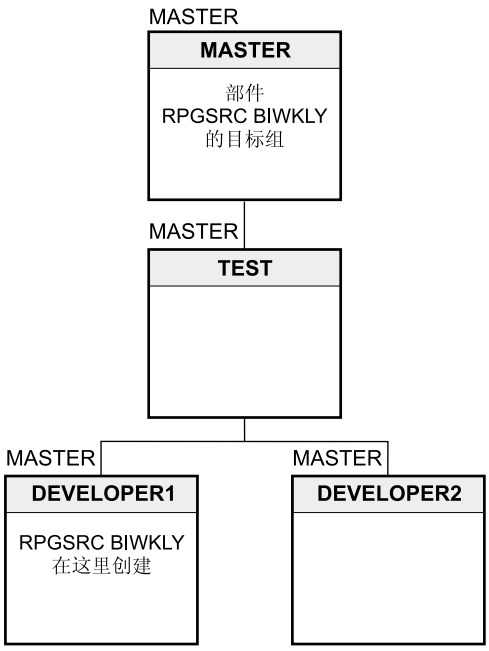

图 20. 创建部件

例如，如果已在DEVELOPER1 组中创建了一个称为 RPGSRC BIWKLY 的部件，而提升码为 MASTER，因而该部件的目标组是 MASTER，那么，您可以在组 DEVELOPER2 中创建名称和类型相同的部件的唯一方法就是指定提升码 *NONE。这样就避免了有两个具有相同名称和类型的部件都可能会被提升至同一组的问题。

必需参数

指定所创建部件的项目、组、类型和部件名。您必须对正在其中创建新部件的组具有更新访问权。有关 CRTPART 命令允许的部件类型的列表，参见附录B. 部件类型以及它们与命令的关系。

可以创建类型为 PARTL 的部件来跟踪您所作的更改，使它们具有一致的命名约定。仅在指定了 TYPE(PARTL) 时，才可指定 PART(*GENERATE)。生成的第一个部件列表名将为 nnnn000001，其中，nnnn 是项目简称。后续生成的部件列表名将为 nnnn000002、nnnn000003 等等。如果生成的名称已存在，则 CRTPART 命令将继续追加至名称的数字递增（最多到一个固定次数），直到找到不存在的部件列表名为止。

当创建部件时，它不会检出到用户简要表中。

可选参数

应在 LANG 参数上指定用于部件的语言。指定语言或者使用缺省值 LANG(*DFT)。*DFT 被替换为已经为所创建部件的类型定义的按字母顺序列示的第一种语言。如果不需要语言的话，则指定 LANG(*NONE)。（有关部件类型和相关联语言的列表，参见第39页的表2。）

如果您正在创建的部件不是源部件，例如，正在创建的是类型为 MSGF、DTAARA、JOBD 或 MSGQ 的部件，则可以指定是否想要提示适当的创建命令。第52页的表5 列示了受“应用程序开发管理器”功能部件支持的所有 OS/400 对象类型。

例如，如果您正在创建一个类型为 DTAARA 的部件，且您选择了 PRMPT(*NO)，则将使用“创建数据区”(CRTDTAARA) 命令缺省值。数据区是 *CHAR，且长度为 256 个字节。在创建数据区之后，就不能更改该数据区的属性。将自动提供输入（如对象名和库名）。

在此情况下，如果您选择 PRMPT(*YES)，则将提示 CRTDTAARA 命令。不能更改自动填充的信息，如对象名和库名。数据区可以为任何类型和大小。

如果您正在创建的部件的类型为 BLDOPT，则 CRTPART 命令将把系统提供的 BLDOPT 源成员复制到此新部件的源成员中。有关这些部件的信息，参见第138页的『使用构建选项』。

可以为保存文件创建类型为 FILE 而语言为 SAVF 的部件。

如果您正在创建的部件的类型为 SCHPTH，则 CRTPART 命令将把缺省搜索路径从指定的组放到此新部件的源成员中。有关这些部件的信息，参见第99页的『创建搜索路径部件』。

如果您正在创建的部件的类型为 SCHIDX，则 CRTPART 命令将用您正在创建的部件的名称来填充 TITLE 参数。如果您对此不满意，则应该在 CRTPART 命令上指定 PRMPT(*YES)，以便提示“创建搜索索引”(CRTSCHIDX) 命令。

应该在 PRMCODE 参数上指定您是否想要提升的部件。（并非所有部件都如此；例如，您可以决定只为测试目的而创建部件，并且不准备将它作为应用程序的部件来保存）。如果您不打算提升该部件，则选择缺省值 PRMCODE(*GRP)。这将导致把为 GRP 参数上指定的组而定义的提升码分配给新部件。如果您不想提升部件，则选择 PRMCODE(*NONE)，从而不能提升该部件。如果您需要查找某个组的提升码，可使用“打印项目”(PRTPRJ) 命令。

此功能部件将确保所创建的部件尚不存在，而且一直是唯一的。注意，PRMPART 命令上的 EXTEND 参数允许构建过程中的输出部件与相关联的源部件一起提升。输出部件的提升码为 *NONE。

SRCFILE 参数是指定源文件的地方，在源文件中，部件作为来源成员存储（如果适用的话）。选择 SRCFILE(*TYPE) 以便对该类型的部件使用由系统提供的缺省源文件，或者输入特定源文件的名称。如果您使用 CRTPART 或 IMPPART 命令来创建包含部件，且 SRCFILE 参数与 *TYPE 不同，则必须确保用源中的文件名来限定包含部件。表3 显示了部件类型以及在其中存储它们的相应 OS/400 缺省源文件。您还可以指定存储在源文件成员（在 ADDADMTYPE 命令上指定的 SYSTYPE(*MBR)）中的用户定义类型的

名称。

表 3. 部件类型及其相应的缺省源文件

类型	缺省源文件
BLDOPT	QBLDOPTSRC
BNDSRC	QSRVSRC
CBL36INC	QS36SRC
CBL36SRC	QS36SRC
CBLINC	QLBLSRC
CBLLEINC	QCBLLESRC
CBLLESRC	QCBLLESRC
CBLSRC	QLBLSRC
CINC（语言为 C、CLE、SQLC、SQLCLE）	H
CLDSRC	QCLDSRC
CLLESRC	QCLLESRC
CLPSRC	QCLSRC
CLSRC	QCLSRC
CMDSRC	QCMDSRC
CSRC（语言为 C、CLE、SQLC、SQLCLE）	QCSRC
DDSSRC	QDDSSRC
DSPF36SRC	QS36SRC
MNU36SRC	QS36SRC
MSGF36SRC	QS36SRC
PNLINC	QPNLSRC
PNLSRC（语言为 MENU）	QMNUSRC
PNLSRC（语言为 PNLGRP）	QPNLSRC
PRDDFN	QDFNSRC
PRDLOD	QLODSRC
OCL36SRC	QS36SRC
REXXSRC	QREXSRC
RPG36INC	QS36SRC
RPG36SRC	QS36SRC
RPGINC	QRPGSRC
RPGLEINC	QRPGLESRC
RPGLESRC	QRPGLESRC
RPGSRC	QRPGSRC
RPT36SRC	QS36SRC
SCHPTH	QSCHPTHSRC
SRT36SRC	QS36SRC
SYSTEML	QSYSTEMSRC
TXT36SRC	QS36SRC
TXTSRC	QTXTSRC

在 PARTL 参数中指定 PARTL 名将导致 CRTPART 命令自动将所创建的部件添加至您指定的部件列表部件中。如果您是系统管理员，则指定 PARTL(*PRV) 时将允许您使用用于此项目的最后一个部件列表部件的名称。但是，如果您是开发者，则此参数标识用于指定组的最后一个部件列表部件的名称。如果您正在使用 QSECOFR 用户简要表运行此命令，则此值无效。PARTL 参数的缺省值为 *NONE，它指示所创建的部件将不会被添加到部件列表部件中。

如果存在下列情况，则必须指定部件列表部件:
- 在命令上指定的组是用 PARTLREQ(*YES) 创建的
- 指定的类型不是 PARTL

如果在 CRTPART 命令上指定了部件列表部件，则:
- 部件列表部件必须存在于从在其中创建部件的组开始的缺省搜索路径中。
- 所创建的部件名被添加至指定的部件列表部件中（如果该部件名尚不存在的话）。（即使命令未能创建部件，也可能会发生此情况）。

如果指定的类型是 PARTL，则将忽略 PARTL 参数。在其他情况下，PARTL 参数将导致指定的部件列表部件被更新。

使用 TEXT 参数来为新部件输入一些描述性文本。如果不输入任何文本，则将使用缺省值 *BLANK，而描述保留为空白。

示例

下列示例说明如何创建称为 PAYROLL DEVELOPER1 RPGSRC BIWKLY 的部件。

使用 CRTPART 命令

```
CRTPART PRJ(PAYROLL) GRP(DEVELOPER1) TYPE(RPGSRC) PART(BIWKLY)   LANG(RPG)
PRMPT(*NO) PRMCODE(*GRP) SRCFILE(QRPGSRC) TEXT(*BLANK)
```

使用“编程开发管理器”实用程序

按“用 PDM 来使用部件”屏幕上的“F6=创建”。

部件被提升并作为源成员存储在 QRPGSRC 中，它是类型为 RPGSRC 的部件的缺省源文件。这与指定 SRCFILE(*TYPE) 相同。

以下命令将在项目 PAYROLL 的组 DEVELOPER2 中创建部件 MNTHLY。

```
CRTPART PRJ(PAYROLL) GRP(DEVELOPER2) TYPE(CSRC) PART(MNTHLY)    LANG(C)
PRMPT(*NO) PRMCODE(*NONE) SRCFILE(QCSRC) TEXT(*BLANK)
```

提升码 *NONE 指示部件不能被提升。部件作为源成员存储在 QCSRC 中，它是类型为 CSRC 的部件的缺省源文件。这与指定 SRCFILE(*TYPE) 相同。

复制部件 (CPYPART)

创建新部件的另一种方法就是将现存部件复制到指定的组和项目中。使用“复制部件”(CPYPART) 命令来执行此操作。

给定名称和类型的部件仅可复制到尚不包含有相同名称和类型的部件的组中。如果在要包含被复制部件的组的缺省搜索路径中已经存在相同名称和类型的部件，则当您复制该部件时必须指定提升码 *NONE。

您必须对包含所复制部件的组具有读或更新访问权，且对接收副本的组具有更新访问权。

在复制部件之后，它不会检出到用户简要表中。

当您复制逻辑文件部件时，必须首先复制它所基于的物理文件，或者该物理文件必须在接收组中已存在。如果您正在复制物理文件，则在 DATA 参数上指定您是否还想复制物理文件中的数据。

如果您想更改所复制部件的属性（如该部件的语言或文本），则在完成复制之后使用“更改部件信息”(CHGPARTINF) 命令。

您想复制的部件

在 CPYPART 命令上，指定*源*标准和*目标*标准。*源* 信息包括项目、组、类型和部件。使用 FROMPRJ、FROMGRP、FROMTYPE 和 FROMPART 参数。有关 CPYPART 命令允许的部件类型的列表，参见附录B. 部件类型以及它们与命令的关系。

如果您想复制所有类型的部件，则使用 FROMTYPE(*ALL)，如果您想复制在 FROMTYPE 参数上指定的类型的所有部件，则使用 FROMPART(*ALL)。

如果在您指定的组中找不到您想复制的部件，则使用 SCAN 参数。SCAN 参数指定如果在指定的组中找不到所复制的部件，是否搜索缺省搜索路径以找到所复制的部件。缺省值为 *YES；这将搜索从指定组直到根组的路径中的所有组。

您想创建的部件

TOPRJ、TOGRP、TOTYPE 和 TOPART 参数允许您指定所创建的部件的项目、组、类型和部件。如果您不填充这些参数，则将使用在 *from* 参数上指定的那些名称。如果您在 FROMTYPE 和 FROMPART 参数上指定了 *ALL，则必须使用缺省值 TOTYPE(*FROMTYPE) 和 TOPART(*FROMPART)。

如果原始部件是源成员，则 TOTYPE 参数必须是源类型。例如，如果原始部件的类型为 RPGSRC（源成员类型为 RPG），则可以将它复制到类型为 RPGSRC、CSRC 或 RPGINC 的部件中。

当将源部件复制到另一种源类型时，例如，将 RPGSRC 复制到 CSRC，或者将 TXTSRC 复制到 CBLSRC 时，应指定适当的语言。例如，如果您在 TOTYPE 参数上指定 CBLSRC，则语言 CBL 被指定给该部件；如果您在 TOTYPE 参数上指定 DDSSRC，则应指定语言 DSPF。对于用户定义类型，指定的缺省语言是已经为所创建部件的类型定义的、按字母顺序列示的第一种语言。

如果原始部件不是 OS/400 源成员，则新的部件类型必须存储在与在 FROMTYPE 参数上指定的部件类型相同的对象类型中。（有关 OS/400 源成员类型和相应的“应用程序开发管理器”部件类型的列表，参见第50页的表4。）有关 CPYPART 命令允许的部件类型的列表，参见『附录B. 部件类型以及它们与命令的关系』。CPYPART 命令还允许用户定义类型。

只要部件名保持不变，就可使用此命令将类型为 VRPGTXT 或 VRPGBIN 的部件从一个组复制到另一个组中，或者从一个项目复制到另一个项目中。

使用 PRMCODE 代码参数来指定是否要提升新部件。此参数适用于几个命令，且在称为第76页的『提升码』的一节中进行了描述。

使用 SRCFILE 参数来指定在其中部件被存储为源成员的源文件。此参数只适用于对第215页的『附录B. 部件类型以及它们与命令的关系』中所列示的 CPYPART 有效的部件类型。缺省值为 *SAME。源文件名与原始部件所使用的文件名相同。使用*名称*来标识特定的源文件，或者使用缺省值 *TYPE 来指定将使用由系统提供的缺省源文件。（有关缺省源文件的列表，参见第43页的表3。）

使用 DATA 参数来指定当复制物理文件时是否复制数据。缺省值为 *NO。

在 PARTL 参数中指定 PARTL 名将导致 CPYPART 命令自动将所创建部件添加至指定的部件列表部件中。如果您是系统管理员，则指定 PARTL(*PRV) 时将允许您使用用于此项目的最后一个部件列表部件的名称。但是，如果您是开发者，则此参数标识用于指定组的最后一个部件列表部件的名称。如果您正在使用 QSECOFR 用户简要表运行此命令，则此值无效。PARTL 参数的缺省值为 *NONE，它指示所创建的部件将不会被添加到部件列表部件中。

如果存在下列情况，则必须指定部件列表部件:
- 在命令上指定的组是用 PARTLREQ(*YES) 创建的
- 指定的类型不是 PARTL

如果在 CPYPART 命令上指定了部件列表部件，则:
- 部件列表部件必须存在于从复制目标组开始的缺省搜索路径中。
- 所创建的部件名被添加至指定的部件列表部件中（如果该部件名尚不存在的话）。（即使命令未能创建部件，也可能会发生此情况）。

如果指定的类型是 PARTL，则将忽略 PARTL 参数。在其他情况下，PARTL 参数将导致指定的部件列表部件被更新。

在第47页的图21中说明的项目层次结构中，部件 BIWKLY 从组 TEST 被复制到组 DEVELOPER3 中。

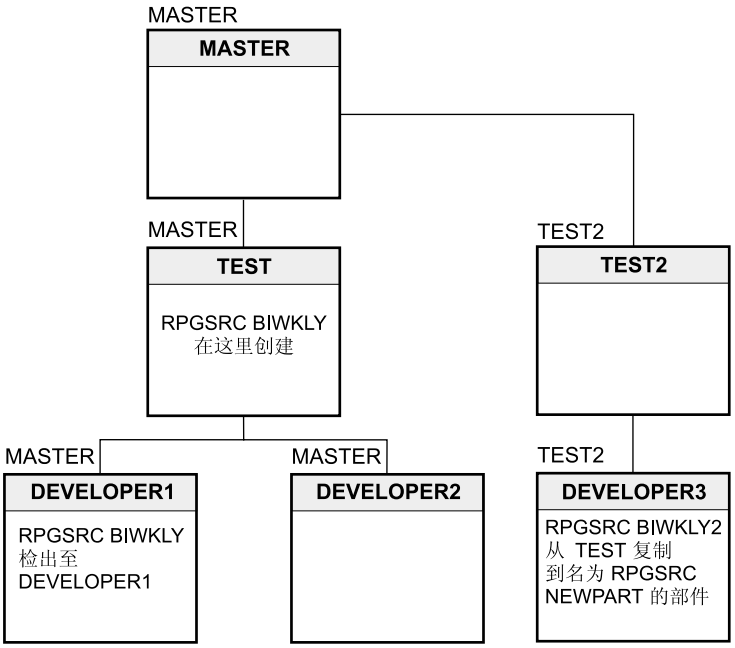

图 21. 当部件存在于几个组中时复制部件

示例

以下命令将复制部件 BIWKLY。

使用 CPYPART 命令

```
CPYPART FROMPRJ(PAYROLL) FROMGRP(TEST) FROMTYPE(RPGSRC)         FROMPART(BIWKLY)
TOPRJ(*FROMPRJ) TOGRP(DEVELOPER3)          TOTYPE(*FROMTYPE) TOPART(NEWPART)
SCAN(*YES)          PRMCODE(*GRP) SRCFILE(*TYPE)
```

使用“编程开发管理器”实用程序

在“用 PDM 来使用部件”屏幕上选择选项 3（复制）。

如命令中所指定的那样，部件 RPGSRC BIWKLY 从组 TEST 被复制到 组 DEVELOPER3 的部件 RPGSRC NEWPART 中。复制成功，因为在 DEVELOPER3 组中或者在 DEVELOPER3 组的提升路径中都不存在这种类型和名称的部件。如果此部件在提升路径中未存在，则它仅可被检出而不会被复制。例如，如果称为 RPGSRC NEWPART 的部件存在于组 MASTER 中，则会发生这种情况。

示例

以下命令将 RPGSRC BIWKLY 从组 DEVELOPER1 中复制到组 DEVELOPER3 中。

```
 CPYPART FROMPRJ(PAYROLL) FROMGRP(DEVELOPER1) FROMTYPE(RPGSRC) FROMPART(BIWKLY)
TOPRJ(*FROMPRJ) TOGRP(DEVELOPER3)          TOTYPE(*FROMTYPE) TOPART(*FROMPART)
SCAN(*NO)           PRMCODE(*GRP) SRCFILE(*TYPE)
```

新部件的类型和名称与所复制部件的类型和名称相同。在 PRMCODE 参数上指定了缺省值 *GRP。SRCFILE(*TYPE) 指定部件被存储在缺省源文件 QRPGSRC 中。此示例假定您对 DEVELOPER3 组具有更新访问权。

如果在组 DEVELOPER1 中找不到该部件，则命令将失败，因为指定了 SCAN(*NO)。如果部件存在于组 DEVELOPER3 （DEVELOPER3、TEST2 或 MASTER）的缺省搜索路径中，则使用 CHKOUTPART 命令，而不管所指定的提升码是什么。有关 CHKOUTPART 命令的详情，参考第61页的『检出部件 (CHKOUTPART)』。

第6章 导入应用程序

现存的应用程序、应用程序的各部分，以及个别成员或对象都可以置于“应用程序开发管理器”功能部件的控制之下。此功能部件允许开发者或管理员将对象或源成员**导入**到项目层次结构中。导入表示将应用程序的一个或多个组件从 OS/400 库复制到“应用程序开发管理器”项目层次结构中。使用“导入部件”(IMPPART) 命令来执行此操作。

本章描述如何将个别对象或整个应用程序导入到项目层次结构中，并提供了关于如何执行此过程的一些逐步指导信息。在导入应用程序之前，您应该花些时间来阅读 *ADTS/400: Application Development Manager Introduction and Planning Guide* 中有关导入应用程序的那一节，它列示了您应该考虑的一些事项。

导入到项目层次结构中

通常，管理员使用 IMPPART 命令来复制整个应用程序或者应用程序的一大部分，而开发者很可能只复制很少的源成员或对象。管理员可以将对象或源成员导入项目层次结构中的任何组中；开发者可以将它们导入到他们对其具有更新访问权的组中。

必须对您想导入的对象和文件具有足够的权限。需要对源文件具有 *USE 权限才能导入其成员，而必须对您想导入的任何对象具有 *USE 和“对象管理”权限。如果您对试图导入的对象没有正确的权限，则 IMPPART 命令将失败。

如果您的应用程序分散在几个库中，则可以首先导入一个库中的所有对象和成员，然后再继续下一个库，直到成功地导入应用程序的所有组件为止。

如果指定了 REPLACE(*NO)，则 IMPPART 命令如以前那样运作。如果在将包含被导入部件的组的缺省搜索路径中已经存在相同名称和类型的部件，则当您导入该部件时必须指定提升码 *NONE。部件的文本描述被更新，以反映所导入对象或源成员的文本描述。当导入部件时，它并未被检出到您的用户简要表中。

如果指定了 REPLACE(*YES)，且存在相同名称和类型的部件并且已经被同一用户在目标组中检出，则该部件将被导入的部件替换。如果该部件未被检出过，且它存在于缺省层次结构中，则该部件被检出而后被替换。因此，当导入完成时，检出的部件又会被检入回去。

如果随 REPLACE(*YES) 一起还指定了 ARCHIVE(*YES)，则部件的原始版本被归档。

要维护“应用程序开发管理器”的完整性，一旦导入的对象成为项目中的部件，则对所导入对象的所有权限就会被更改。对所导入对象的公用权限被设置为 *EXCLUDE。对所导入对象的专用权限是使用权限列表来设置的。有关详情，参见 *iSeries Security Reference*。

如果您使用“编程开发管理器”实用程序，则可以使用“使用组”屏幕上的选项 38（导入）来导入成员和对象。还可以使用称为 IM 和 IO 的用户定义选项来帮助您进行导入。IM 将导入成员，IO 将导入对象。有关如何执行此操作的信息，参见第208页的『使用用户定义选项』。

通常，不应导入由 BLDPART 命令创建的部件。BLDPART 命令需要能够编译导入的部件以了解已经导入的各部件之间的关系，进而重新创建输出部件。

如果您导入输出部件，则当 BLDPART 试图构建部件时，构建报告中将发出警告，即使在构建的搜索路径中可能存在同名的源部件时亦是如此。在此实例中，将需要删除输出部件，并再次发出 BLDPART 命令。

如果您想导入逻辑文件，则必须首先将该文件所基于的物理文件导入到同一组中。导入 *ALL 对象时不需要执行此步骤，因为 IMPPART 命令将确保物理文件是在逻辑文件之前复制的。

所导入的已记入日志的文件在项目层次结构中将不会被记入日志。

您想导入的文件或对象

在 IMPPART 命令上，必须指定 *from* 和 *to* 标准。*from* 信息由表示您想导入的信息的 OS/400 库和对象组成。如果您正在导入源文件，则还要指示成员名。使用 OBJ、OBJTYPE 和 MBR 参数来标识您想导入哪些 OS/400 对象。

在 OBJ 参数上指定库名。特殊值 *LIBL 和 *CURLIB 或者特定的库名是受支持的。

还要在 OBJ 参数上指定对象的名称，或者使用类属名。可以用格式 ABC* 来指定类属对象，这将处理以您指定的字符开头（如 ABC）的所有对象或文件。特定的对象名将导入一个对象。特殊值 *ALL 将导入在指定库中找到的所有对象。但是，要注意，不能通过在 IMPPART 命令上指定 OBJ(*LIBL/*ALL) 来导入在库列表中找到的所有对象。特殊值 *ALL 一次只能配合一个库使用。还会导入具有用户定义类型的任何对象以及为它们定义的适当 *CPY 操作。

OBJTYPE 参数指示您想导入哪种类型的 OS/400 对象。只能导入受“应用程序开发管理器”功能部件支持的 OS/400 对象类型。表4 列示了受支持的源成员类型，第52页的表5列示了受支持的对象类型。还可以导入用户定义部件类型。

表 4. 应用程序开发管理器功能部件支持的 OS/400 源成员类型

OS/400 对象类型	OS/400 成员类型	部件类型	部件语言
*FILE		VRPGTXT[1]	VRPG
*FILE	BND	BNDSRC	BND
*FILE	C[2]	CINC	C
*FILE	C[2]	CINC	CLE
*FILE	C[3]	CSRC	C
*FILE	C[3]	CSRC	CLE
*FILE	CBL[4]	CBLINC	CBL
*FILE	CBL[5]	CBLSRC	CBL
*FILE	CBL36	CBL36INC	CBL36
*FILE	CBL36	CBL36SRC	CBL36
*FILE	CBLLE[4]	CBLLEINC	CBLLE
*FILE	CBLLE	CBLLESRC	CBLLE
*FILE	CL	CLSRC	CL
*FILE	CLD	CLDSRC	CLD

表 4. 应用程序开发管理器功能部件支持的 OS/400 源成员类型 (续)

OS/400 对象类型	OS/400 成员类型	部件类型	部件语言
*FILE	CLLE	CLLESRC	CLLE
*FILE	CLP	BLDOPT[6]	*NONE
*FILE	CLP	CLPSRC	CLP
*FILE	CLP	PRDDFN[6]	*NONE
*FILE	CLP	PRDLOD[6]	*NONE
*FILE	CMD	CMDSRC	CMD
*FILE	DSPF	DDSSRC	DSPF
*FILE	DSPF36	DSPF36SRC	DSPF36
*FILE	ICFF	DDSSRC	ICFF
*FILE	LF	DDSSRC	LF
*FILE	MENU	PNLSRC	MENU
*FILE	MNU36	MNU36SRC	MNU36
*FILE	MSGF36	MSGF36SRC	MSGF36
*FILE	OCL36	OCL36SRC	OCL36
*FILE	PF	DDSSRC	PF
*FILE	PNLGRP[4]	PNLINC	PNLGRP
*FILE	PNLGRP	PNLSRC	PNLGRP
*FILE	PRTF	DDSSRC	PRTF
*FILE	REXX	REXXSRC	REXX
*FILE	RPG[4]	RPGINC	RPG
*FILE	RPG	RPGSRC	RPG
*FILE	RPG36	RPG36INC	RPG36
*FILE	RPG36	RPG36SRC	RPG36
*FILE	RPGLE[4]	RPGLEINC	RPGLE
*FILE	RPGLE	RPGLESRC	RPGLE
*FILE	RPT36	RPT36SRC	RPT36
*FILE	SQLC[2]	CINC	SQLC
*FILE	SQLC[2]	CINC	SQLCLE
*FILE	SQLC[3]	CSRC	SQLC
*FILE	SQLC[3]	CSRC	SQLCLE
*FILE	SQLCBL[4]	CBLINC	SQLCBL
*FILE	SQLCBL[5]	CBLSRC	SQLCBL
*FILE	SQLRPG[4]	RPGINC	SQLRPG
*FILE	SQLRPG	RPGSRC	SQLRPG
*FILE	SRT36	SRT36SRC	SRT36
*FILE	TXT	SCHPTH[6]	*NONE
*FILE	TXT	SYSTEML	*NONE
*FILE	TXT36	TXT36SRC	TXT36
*FILE		TXTSRC[7]	*NONE

表 4. 应用程序开发管理器功能部件支持的 OS/400 源成员类型 (续)

OS/400 对象类型	OS/400 成员类型	部件类型	部件语言

注意:

1. VRPGTXT 部件中每个成员的成员类型是由 VARPG 根据其内容来设置的，并且将由 IMPPART 命令来保留。
2. 如果您导入称为 H 的源文件，则将创建类型为 CINC 的部件。如果成员类型为 C，则导入的部件的语言将为 CLE。类似地，如果成员类型为 SQLC，则部件语言将为 SQLCLE。如果您想导入语言为 C 或 SQLC 的 CINC 部件，则必须显式地指定语言。
3. 如果成员类型为 C 且部件类型为 CSRC，则所导入部件的语言将为 CLE。如果您想导入语言为 C 或 SQLC 的此部件，则必须显式地指定语言。类似地，成员类型 SQLC 将具有语言 SQLCLE。
4. 可以导入包含，但是没有关于 OS/400 系统的任何信息来将源与包含区分开来。当导入 RPG、SQLRPG、CBL、SQLCBL 或 PNLSRC 成员时，“应用程序开发管理器”部件将采用与源成员对应的缺省部件类型。要将这些成员导入到类型为 RPGINC、RPGLEINC、CBLINC、CBLLEINC 或 PNLINC 的部件中，必须显式地指定部件类型，或者使用 CVTPART 命令来更改该部件的类型和语言。
5. 如果您导入类型为 CBL 或 SQLCBL 的成员，且它包含多个程序，则 BLDPART 命令将只编译第一个程序。其他程序会被忽略，并且在构建报告中将发出警告消息。
6. 由“应用程序开发管理器”创建的特殊源文件是 QBLDOPTSRC、QSCHPTHSRC、QDFNSRC、QLODSRC 和 QSYSTEMSRC。如果您从这些源文件中导入成员，则会创建适当类型的部件。例如，当从 QBLDOPTSRC 中导入源成员时，就创建了 BLDOPT 部件。
7. 在该表中未列示其类型的所有成员是作为类型为 TXTSRC、语言为 *NONE 的部件来导入的。当导入成员时，成员类型不会更改。

表 5. 受应用程序开发管理器功能部件支持的 OS/400 对象类型

OS/400 对象类型	OS/400 对象属性	部件类型	部件语言
*BNDDIR		BNDDIR	*NONE
*CLD		CLD	CLD
*CMD		CMD	CMD
*DTAARA		DTAARA	*NONE
*FILE	DSPF	FILE	DSPF
*FILE	ICFF	FILE	ICFF
*FILE	LF	FILE	LF
*FILE	PF	FILE	PF
*FILE	PF	PARTL	*NONE
*FILE	PF	VRPGBIN	VRPG
*FILE	PRTF	FILE	PRTF
*FILE	SAVF	FILE	SAVF
*JOBD		JOBD	*NONE
*JOBQ		JOBQ	*NONE
*MENU	UIM	MENU	*NONE
*MODULE	CBLLE	MODULE	CBLLE
*MODULE	CLE	MODULE	CLE

表 5. 受应用程序开发管理器功能部件支持的 OS/400 对象类型 (续)

OS/400 对象类型	OS/400 对象属性	部件类型	部件语言
*MODULE	CLLE	MODULE	CLLE
*MODULE	RPGLE	MODULE	RPGLE
*MSGF		MSGF	*NONE
*MSGQ		MSGQ	*NONE
*OUTQ		OUTQ	*NONE
*PNLGRP		PNLGRP	*NONE
*PGM	C	PGM	C
*PGM	CBL	PGM	CBL
*PGM	CBL36	PGM	CBL36
*PGM	CBLLE	PGM	CBLLE
*PGM	CLE	PGM	CLE
*PGM	CLLE	PGM	CLLE
*PGM	CLP	PGM	CLP
*PGM	RPG	PGM	RPG
*PGM	RPG36	PGM	RPG36
*PGM	RPGLE	PGM	RPGLE
*SCHIDX		SCHIDX	*NONE
*SRVPGM		SRVPGM	*NONE
*SRVPGM	CBLLE	SRVPGM	CBLLE
*SRVPGM	CLE	SRVPGM	CLE
*SRVPGM	CLLE	SRVPGM	CLLE
*SRVPGM	RPGLE	SRVPGM	RPGLE

在 OBJTYPE 参数上使用特殊值 *ALL，以显示要复制特定库中所有受支持的对象。此值不能与库列表同时使用，而且一次只能使用一个特定库。

特殊值 *SRC 指示您只想要复制作为源物理文件的对象。指定 *SRC 时，会导入具有对象属性 PF 且作为源物理文件的所有对象。表4显示 OS/400 成员类型如何与 “应用程序开发管理器” 部件类型对应。注意，还会导入表中未列出的成员类型的所有成员，但这些成员将作为类型为 TXTSRC 的部件。此值不能与库列表同时使用，而且一次只能使用一个特定库。

特殊值 *NONSRC 显示要复制未作为源物理文件的所有对象。指定 *NONSRC 时，会导入与第52页的表5中列出的对象类型相匹配的所有对象。此值不能与库列表同时使用，而且一次只能使用一个特定库。

可在 OBJTYPE 参数上指示特定对象类型。例如，可指定 OBJTYPE(*PGM) 以导入所有程序，或指定 OBJTYPE(*MSGF) 以导入所有消息文件。如果指定 OBJTYPE(*FILE)，则会导入类型为 *FILE 且具有属性 PF（数据和源物理文件）、LF、DSPF、PRTF、SAVF 或 ICFF 的所有对象。导入源文件的方式取决于在 TYPE 参数上指定的内容。如果指定 TYPE(*FILE)，则源文件会作为物理数据文件导入。要将源文件作为源导入，即，单独地导入每个成员，则使用缺省值 TYPE(*OBJTYPE)。

导入源物理文件时，可在 MBR 参数上指定特定源成员，或通过使用特殊值 *ALL 指示要导入源文件内的所有成员。还可使用类属名。可使用如 ABC* 之类的格式指定类属成员名，从而会处理以字符 ABC 开头的所有成员。

如果在 OBJTYPE 参数上指定了 *FILE、*SRC、或 *ALL，同时还指定了 OBJ(*ALL)，则需要值 MBR(*ALL)。如果只从源文件导入一个成员，则指定 OBJ（*文件名*）、OBJTYPE(*SRC)（或 OBJTYPE(*FILE)）和 MBR（*成员名*）。

想要创建的部件

必须指定一些信息来指示想要导入 OS/400 对象或源成员的位置。

必需参数

部件是在您指示的项目和组中创建的，以 OS/400 对象或源成员为基础。在 PRJ 参数上指定项目名，并在 GRP 参数上指定组名。这两个参数都是必需的。

在 TYPE 参数上指定“应用程序开发管理器”部件类型。如果指定特殊值 *OBJTYPE 且正在导入的对象不是源物理文件，则部件类型缺省为 OS/400 对象类型的“应用程序开发管理器”等效值。例如，如果导入 *PGM 对象并指定 TYPE(*OBJTYPE)，则“应用程序开发管理器”类型为 PGM。

如果正在导入的对象是源物理文件，则每个成员都会成一个独立部件。对于这些部件，在指定特殊值 *OBJTYPE 时该部件类型缺省为 OS/400 对象属性和成员类型的“应用程序开发管理器”等效值。参考表4和表5以查看 OS/400 类型与部件类型的相关性。

还可指定特定部件类型。但是，不能在 TYPE 参数上提供对导入的 OS/400 对象无意义的“应用程序开发管理器”部件类型。例如，不能将类型为 *PGM 的对象导入类型为 RPGSRC 的部件。如果试图这样做的话，IMPPART 命令会失败。

必须在 IMPPART 命令的 TYPE 参数上指定 VRPGTXT 或 VRPGBIN，才能导入 VARPG 源或二进制对象。如果导入了源文件，且指定了类属部件类型，则会个别导入源文件中的成员。如果导入了物理文件，且指定了类属部件类型，则该物理文件会作为类型为 *FILE 的部件导入。

示例: 以下命令将导入带有 C/400 源语句的源成员，并创建类型为 CSRC 的部件 HELLO。部件语言被设置为 CLE（如果 HELLO 的源成员属性为 C 的话）。

使用 IMPPART 命令

```
IMPPART OBJ(*CURLIB/CSRC) OBJTYPE(*SRC) MBR(HELLO) PRJ(PAYROLL)
        GRP(MASTER) TYPE(CSRC) PART(*NAME)
```

使用“编程开发管理器”实用程序

在“用 PDM 来使用组”屏幕上选择选项 38（导入）。

如果导入包含属性为 C 或 SQLC 的 C 源的源成员，会创建类型为 CSRC （除非在 TYPE 参数上明确指定了 CINC 类型）且部件语言分别为 CLE 或 SQLCLE 的部件。

在 PART 参数上指定部件的名称。选择部件名缺省值 *NAME 会导致源成员名或对象名成为新部件的名称。指定部件名（如果想要对正在导入的部件明确命名的话），或给定新名称（因为只导入带有该名称的对象或源成员）。

示例: 成员 BIWKLY 被导入并重命名为 EMPMST。

```
IMPPART OBJ(*CURLIB/QRPGSRC) OBJTYPE(*SRC) MBR(BIWKLY)
        PRJ(PAYROLL) GRP(MASTER) TYPE(RPGSRC) PART(EMPMST)
```

如果需要导入的对象的名称对“应用程序开发管理器”部件无效（如对象的名称中带有小写字符），则必须显式命名新部件。如果在此处使用 PART(*NAME)， IMPPART 命令将会失败。

如果通过在对象 (OBJ) 或成员 (MBR) 参数上指定 *ALL 来导入多个源成员或对象，则不能显式命名这些部件。而是，必须通过指定 PART(*NAME) 来让部件名缺省为源成员名或对象名。

示例: 以下命令导入当前库的源文件 QRPGSRC 中的所有源成员，并使用这些源成员的名称来创建部件。

```
IMPPART OBJ(*CURLIB/QRPGSRC) OBJTYPE(*SRC) MBR(*ALL)
        PRJ(PAYROLL) GRP(MASTER) TYPE(*OBJTYPE) PART(*NAME)
```

注意，使用 IMPPART 命令不能替换项目层次结构中的现存部件。如果该部件已经存在于您试图在其中创建它的组中，或者如果该部件存在于该组的搜索路径中，则 IMPPART 命令会失败。必须删除搜索路径中所有出现该部件的位置，并尝试再次导入该部件。

可选参数

在将对象导入项目层次结构时，除指定*源*和*目标*标准外，还可指示有关该部件的下列信息:

- 语言属性
- 提升码（参见第76页的『提升码』以获取详情）
- 想要存储部件的源文件
- 是否想要使用物理数据文件来复制数据
- 是否想要替换现存部件
- 要添加导入部件名的部件列表部件
- 是否想要将正在导入的部件的旧版本归档
- 是否想要与新部件相关联的描述性文本

使用参数 LANG、PRMCODE、SRCFILE、DATA、REPLACE、PARTL、ARCHIVE 和 TEXT 来分别指示此信息。

在 LANG 参数上指定该部件的语言属性。LANG 参数缺省值为 *ATTR。如果导入对象，对象类型和对象属性用来确定“应用程序开发管理器”语言属性。如果导入源文件，会使用成员类型。参考第50页的表4和第52页的表5以查看 OS/400 对象类型与“应用程序开发管理器”部件语言的相关性。

可在此参数上指定一种语言，或者可指定 LANG(*NONE)，指示没有语言属性。值 *NONE 仅对不需要语言属性的部件有效。

VRPGTXT 或 VRPGBIN 部件具有“应用程序开发管理器”部件语言 VRPG。

示例: 如果从源文件导入 RPG 源成员，但又想要创建类型为 CSRC 的部件，则必须在 IMPPART 命令中指定语言属性 C，如以下命令中所示。

```
IMPPART OBJ(*CURLIB/QRPGSRC) OBJTYPE(*SRC) MBR(BIWKLY) PRJ(PAYROLL)
        GRP(MASTER) TYPE(CSRC) PART(BIWKLY) LANG(C)
```

如果通过在对象 (OBJ) 或成员 (MBR) 参数上指定 *ALL 来导入多个源成员或对象，则必须使用缺省值 LANG(*ATTR)。

在 SRCFILE 参数上指定要存储新部件的位置。缺省情况是通过指定 SRCFILE(*FROMFILE) 以使用导入部件的源文件名。如果想要部件存储在特定源物理文件中，则输入源文件名称。特殊值 SRCFILE(*TYPE) 也是受支持的。这指示使用系统提供的缺省源文件。例如，类型为 RPGSRC 的部件存储在源物理文件 QRPGSRC 中。第43页的表3列出了“应用程序开发管理器”部件类型及其相应的 OS/400 缺省源文件。如果类型为 VRPGTXT 的话， SRCFILE 参数会被忽略。

在 DATA 参数上指定是否要复制与物理数据文件相关联的数据。缺省情况是不复制数据。

在将当前部件的版本替换为在“应用程序开发管理器”功能部件外部开发出的版本时，REPLACE 参数是非常有用的。此功能将允许您使用不同的提升码在不同组库中导入部件的不同的版本。这在“应用程序开发管理器”功能部件的前发行版中是不可能的。

如果指定了 REPLACE(*YES)，则将使用正在替换的部件的当前提升码和源文件，且 SRCFILE 参数将被忽略。如果正在导入的部件不是物理文件，且已经存在，则:
- 该部件将被检出至要导入它的组
- 该部件将被替换为导入版本
- 部件将被检入

如果部件不存在于目标组中，而是存在于更高级别的组中，则在导入操作期间会从较高级别组中检出该部件，并在导入结束时检入该部件。

使用 PARTL 参数指定是否想要将导入部件添加至部件列表部件。指定 PARTL(*PRV) 允许您使用用于此项目的上一个部件列表部件的名称（如果您是系统管理员的话）。但是，如果您是开发者的话，此参数会标识用于指定组的上一个部件列表部件的名称。如果使用 QSECOFR 用户简要表运行此命令，则此值无效。PARTL 参数的缺省值是 *NONE，指示正在导入的部件将不会添加至部件列表部件。

下列情况下必须指定部件列表部件:
- 在命令上指定的组是使用 PARTLREQ(*YES) 创建的
- 指定的类型不是 PARTL

如果在 IMPPART 命令上指定了部件列表部件，则:
- 部件列表部件必须存在于从导入该部件的组起始的缺省搜索路径中。
- 导入部件名被添加至指定部件列表部件，如果它还不存在的话。（即使命令未能导入该部件，也可能会发生此情况）。

如果指定的类型为 PARTL，则 PARTL 参数会被忽略。对于其他情况而言，PARTL 参数会导致指定部件列表部件更新。

使用 ARCHIVE 参数来指定正在替换的部件是否应归档。仅当指定 REPLACE(*YES) 且该部件是源成员部件时，此参数才有意义。有关对部件进行归档的详情，参见第82页的『归档部件』。

在 TEXT 参数上，对该部件指定描述性文本。可输入最多 80 个字符的文本。如果指定 TEXT(*TEXT) 的话，该描述将缺省为与该对象或源成员相关联的文本。特殊值 *BLANK 指示部件没有相关联的文本。

导入一个部件的方案

本节描述导入部件时要遵循的步骤。

1. 标识想要导入的 OS/400 对象或成员

   必须确定该对象类型是否受“应用程序开发管理器”功能部件支持。参见第52页的表5。还可导入用户定义的部件类型。

2. 导入源成员

   发出 IMPPART 命令以导入包含源代码的源物理文件。以下示例将称为 PROG1 的源成员导入组 DEVELOPER1 中称为 BIWKLY 的部件。

   ```
   IMPPART OBJ(*CURLIB/QRPGSRC) OBJTYPE(*SRC) MBR(PROG1) PRJ(PAYROLL)
   GRP(DEVELOPER1) TYPE(RPGSRC) PART(BIWKLY)
   ```

3. 必要时更改部件

   导入部件后，确保未使用库名限定源成员中的所有包含部件和外部引用。如果限定了的话，则将它们更改为 *LIBL，而不是特定库的名称。在构建过程中，这对于允许编译器在项目中查找该部件的正确版本来说是必需的。

   如果在 IMPPART 命令的 SRCFILE 参数上指定了不同于 *TYPE 的值，则应确保所有包含部件都是使用文件名限定的。

4. 构建部件

   必须构建部件才能确保所有相关部件已导入且是最新的。建议在导入部件后使用下列选项来构建它。

   PART(*ALL) 构建导入的所有部件

   TYPE(*ALL) 构建所有类型的部件

   如果只导入 OS/400 对象，则构建过程不知道该对象及其对应源之间的部件关系。还必须导入源，或构建过程会在尝试构建该对象时发出警告消息。参见第119页的『了解部件关系』以获取有关部件关系及构建过程的详情。

5. 验证部件是否以期望的方式运行

   参见第12章 测试和运行应用程序。

导入样本应用程序

本节描述在创建新“应用程序开发管理器”项目层次结构和从库 SAMPLE 中导入以下虚构应用程序时要遵循的一系列命令。（SAMPLE 库并不是随本产品提供的，您需要创建它。）它假定您主要导入源成员而不是程序对象。在导入应用程序后构建它就可创建您需要的所有程序对象。

表6列出了用来创建以下步骤中概述的新“应用程序开发管理器”项目层次结构的 OS/400 成员（及其相关联的对象类型和成员类型）。

表 6. OS/400 成员及其相关联的对象和成员类型

OS/400 成员	对象类型	成员类型
REFMST	*FILE	PF
CTLFIL	*FILE	PF
EMPMST	*FILE	PF
RSNMST	*FILE	PF

表 6. OS/400 成员及其相关联的对象和成员类型 (续)

OS/400 成员	对象类型	成员类型
TRWEEK	*FILE	PF
TRWEEKL	*FILE	LF
PRG03FM	*FILE	DSPF
PRG06RP	*FILE	PRTF
PROC3	*FILE	CLP
PRG03	*FILE	SQLRPG
PRG03A	*FILE	SQLRPG[1]

注:

1. 此成员实际上是一个包含成员。

表7列出了用来创建以下步骤中概述的新“应用程序开发管理器”项目层次结构的 OS/400 对象，及其相关联的对象类型和对象属性。

表 7. OS/400 对象及其相关联的对象类型和对象属性

OS/400 对象	对象类型	对象属性
MESSAGES	*MSGF	*NULL

1. 使用 CRTPRJ 命令创建项目。以下命令会创建项目 PAYROLL。

   ```
   CRTPRJ PRJ(PAYROLL) SHORTPRJ(PRL) TEXT('WEEKLY PAYROLL PROCESSING
          APPLICATION')
   ```

   有关创建项目的详情，参见创建项目 (CRTPRJ)。

2. 在项目 PAYROLL 中创建根组。应用程序会被导入根组。以后可在项目层次结构中添加其他组。以下命令在此项目中创建组 MASTER。

   ```
   CRTGRP PRJ(PAYROLL) GRP(MASTER) SHORTGRP(MST) PARENT(*NONE)
          TEXT('MASTER GROUP IN PROJECT PAYROLL')
   ```

   有关创建组的详情，参见创建组 (CRTGRP)。

3. 标识计划导入的所有应用程序，并确定哪些对象类型可导入项目 PAYROLL。参见第50页的表4和第52页的表5以获取受“应用程序开发管理器”功能部件支持的所有 OS/400 对象类型的列表。

   考虑使用“用 PDM 来使用成员”屏幕和“用 PDM 来使用对象”屏幕来列出作出应用程序的部件的 OS/400 源文件成员和对象。还可在这些屏幕中使用 IM 和 IO 用户定义选项。它们在第208页的『使用用户定义选项』中作了描述。

4. 使用 IMPPART 命令以导入包含源代码的所有源物理文件。使用以下命令来将库 SAMPLE 中发现的每个源文件内的所有成员导入项目 PAYROLL。在虚构应用程序中，注意成员 PRG03A 实际上是包含成员。

   ```
   IMPPART OBJ(SAMPLE/*ALL) OBJTYPE(*SRC) PRJ(PAYROLL) GRP(MASTER)
           TYPE(*OBJTYPE) PART(*NAME)
   ```

   如果源代码驻留在几个不同的库中，则对每个库使用此命令，将 SAMPLE 替换为实际库名。

5. 使用 IMPPART 命令以导入应用程序所需的所有对象。使用以下命令来导入库 SAMPLE 中的所有对象。在虚构应用程序中，只会导入一个对象，它被称为 MESSAGES。

```
IMPPART OBJ(SAMPLE/*ALL) OBJTYPE(*NONSRC) PRJ(PAYROLL) GRP(MASTER)
        TYPE(*OBJTYPE) PART(*NAME)
```

6. 为每个已导入的类型为 CMDSRC 部件创建类型为 BLDOPT 的部件。BLDOPT 部件应与类型为 CMDSRC 的部件同名，且必须包含此命令源所需的创建命令。参见使用构建选项以获取有关如何为应用程序创建和管理构建选项的详情。
7. 将使用库限定语句的所有部件更改为 *LIBL。
8. 如果在导入后您拥有的源部件必须转换为包含部件，则使用 CVTPART 命令。在此示例中，成员 PROG3A 实际上是包含源成员。在该成员导入时，会创建类型为 RPGSRC 的部件 PROG3A。此部件必须转换为类型为 RPGINC 的部件。有关导入包含部件的信息，参见第50页的表4。
9. 如果想要构建导入的部件，可在 BLDPART 命令上指定 GROUP(MASTER)、TYPE(*ALL)、和 PART(*ALL)，以构建刚刚导入的所有部件。
10. 将 BLDPART 命令与 SCOPE(*NORMAL) 配合使用以构建应用程序。以下命令会在项目 PAYROLL 的组 MASTER 中构建所有类型的所有部件。

    ```
    BLDPART PRJ(PAYROLL) GRP(MASTER) TYPE(*ALL) PART(*ALL)
            SCOPE(*NORMAL)
    ```

    参见第11章 构建应用程序以获取有关如何构建应用程序的详情。
11. 通过测试应用程序以验证它是否以期望的方式运行。查看第12章 测试和运行应用程序以获取有关进行此操作的信息。

第7章 使用部件

在使用部件时会执行的基本活动有:
- 检出部件
- 更改部件
- 比较部件
- 合并部件
- 检入部件
- 提升部件
- 重命名部件
- 删除部件
- 归档部件

还可对有关部件的信息执行下列活动:
- 更改部件信息
- 打印部件信息
- 检索有关部件的信息
- 将部件转换为不同类型的部件

注

与创建、更改或删除“应用程序开发管理器”信息（项目、组和部件）相关联的所有任务应使用“应用程序开发管理器”界面完成。这些界面包括 CL 命令和“编程开发管理器”屏幕（由 WRKPRJPDM、WRKGRPPDM 和 WRKPARTPDM 命令创建）。此外，还可通过 CODE/400 产品和“应用程序开发工具箱”的“应用程序字典服务”功能部件来访问“应用程序开发管理器”项目、组或部件。还可使用 CHGPARTINF 命令来同步部件时间戳记，以允许 BLDPART 命令识别已更改的部件。

检出部件 (CHKOUTPART)

部件必须完全由您控制，您才能更改它。此控制权是通过“检出部件” (CHKOUTPART) 命令获得的。

使用 CHKOUTPART 命令来将部件检出至您对具有更新访问权的组。从该部件驻留的组将其复制至指定组，而在目标组上设置该部件的低位锁定（项目层次结构中部件可提升至的最终组）。访问密钥被设置为您的用户简要表。设置访问密钥的目的是防止其他人更新该部件。当您使用“检入部件” (CHKINPART) 命令时，此访问密钥被复位为空白。**低位锁定**与存储的有关部件的信息相关联。它是包含出现该部件的最低位置的组的名称。

如果该部件在将其检出至的组中尚不存在，CHKOUTPART 命令会从项目层次结构中较高级别的组中复制它。系统会从指定组开始对项目层次结构进行扫描，以查找带有匹配名称和类型的部件。该部件必须存在于缺省搜索路径中才能被检出。

如果想要确定该部件是否已被检出至别的组，则使用 PRTPARTINF 命令以运行一个报告，显示让该部件检出的人员和将该部件检出至的组。查看第86页的图30以获取样本报告。

必需参数

在 CHKOUTPART 命令上指定项目、组、类型和部件。并非所有类型的部件都可以检出。

可选参数

使用 PRMCODE 参数来指定要在确定部件的提升路径时使用的提升码。此参数适用于几个命令，且在第76页的『提升码』中作了描述。注意，如果使用提升码 *NONE 检出部件，然后决定提升该部件，则必须删除该部件，并使用提升码 *GRP 再次检出它。

必须对如下类型的部件指定 PRMCODE(*NONE)：这些部件的类型为 PGM、FILE、CLD、CMD、MENU、PNLGRP、MODULE、SRVPGM 或是存储在由“构建部件”(BLDPART) 命令创建的 OS/400 对象中的用户定义类型。仅当在 PRMPART 命令上指定 EXTEND(*YES) 时才能提升这些类型的部件。注意，类型为 PGM 且语言为 CBL36 和 RPG36 的部件可使用任何提升码检出。

示例

本示例说明部件 RPGSRC BIWKLY（存在于组 TEST 中）如何被检出至组 DEVELOPER1。第63页的图22说明了此过程。

使用 CHKOUTPART 命令

```
CHKOUTPART PRJ(PAYROLL) GRP(DEVELOPER1) TYPE(RPGSRC) PART(BIWKLY)
PRMCODE(*GRP)
```

使用编程开发管理器实用程序

在“用 PDM 来使用部件”屏幕上选择选项 28（检出）。

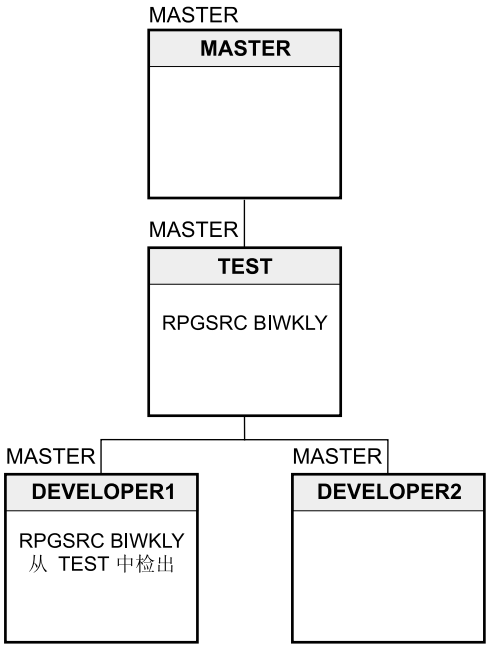

图 22. CHKOUTPART 命令和提升码

DEVELOPER1 的提升码为 MASTER，所以部件 RPGSRC BIWKLY 被指定为此提升码，表示它最终可提升回组 MASTER。

如果需要有关提升码的详情，查看第16页的『为项目层次结构定义一个提升码』以获取有关项目管理员如何设置提升码的描述。第74页的『提升部件 (PRMPART)』描述“提升部件 (PRMPART) 命令和提升码参数的使用方法。要查看组的提升码，发出“打印项目”(PRTPRJ) 命令。

数据安全性和完整性

部件的低位锁定阻止部件被检出至多个组。例如，如果部件已被检出至项目层次结构另一分支中使用同一提升码的的开发组，则会在目标组上对该部件设置其低位锁定。目标组是阻止您访问该部件的 TEST 组。

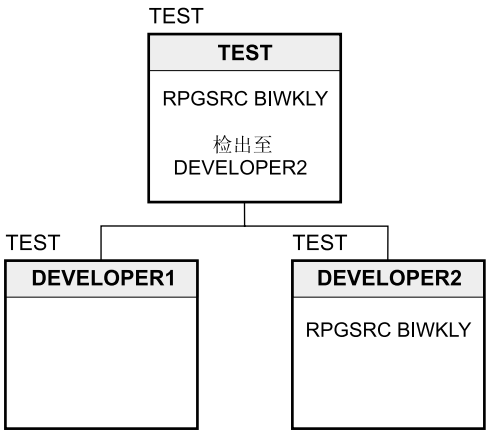

图 23. 部件上的低位锁定设置

在图23中，RPGSRC BIWKLY 存在于组 TEST 中。在组 DEVELOPER2 中工作的开发者已将部件检出至该组，并打算使用提升码 TEST 将该部件提升回组 TEST。因此，如果正在组 DEVELOPER1（它也具有提升码 TEST）中工作，则不能检出此部件。这样做意味着具有相同名称和类型的两个部件将被提升至同一组。低位锁定可防止这种情况发生。

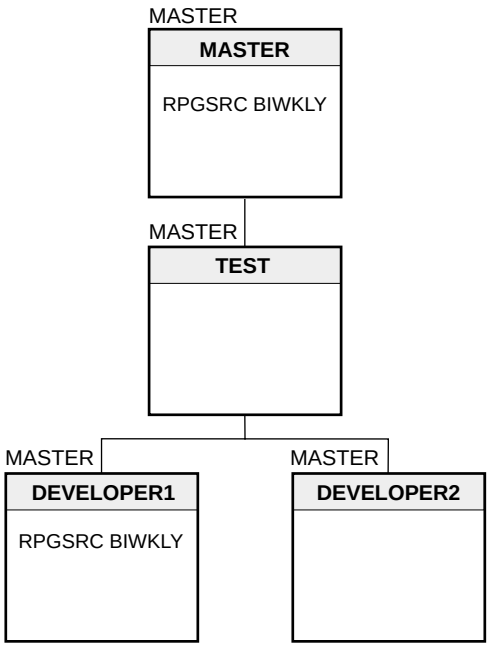

图 24. 部件上的低位锁定设置

在图24中，RPGSRC BIWKLY 存在于组 MASTER 中。它被检出至组 DEVELOPER1，并将被提升回至组 MASTER。RPGSRC BIWKLY 不能被检出至组 TEST，因为组 DEVELOPER1 的低位锁定会阻止此操作的进行。

部件的访问密钥阻止多个开发者检出该部件。例如，如果对组具有访问权的另一开发者已检出该部件，就对该部件设置了低位锁定，且该部件的访问密钥已被设置为此人的用户简要表。仅当部件的访问密钥被设置为空白时，您才能检出此部件。

更改部件 (CHGPART)

一旦您检出了部件，它就在您的控制之下，可使用“更改部件” (CHGPART) 命令来更改它。必须对包含想要更改的部件的组具有更新访问权。如果该部件已存在于指定组中，则 CHGPART 命令会首先将其检出。

注

如果对部件驻留的组具有 *UPDATE 访问权，则在“用 PDM 来使用部件”屏幕上的部件前输入选项 2（更改）。将会在该部件存在的组而不是在您工作的组中检出该部件并进行更改。

CHGPART 命令会运行 OS/400 命令，以更改指定部件的内容或特征。如果该部件是可编辑的，则会显示适当的编辑会话。如果该部件不能编辑，则会显示适当的更改命令以允许您对其特征作适当的更改。

必需参数

指定要更改的项目、组、类型和部件。并非所有类型的部件都可更改。查看附录B. 部件类型以及它们与命令的关系以获取 CHGPART 命令所允许的部件类型的列表。

可选参数

CHGCMD 参数确定要调用的更改命令。可在此参数上指定 *TYPE、STRSDA、STRRLU 或 UPDDTA。

如果选择缺省值 CHGCMD(*TYPE)，则部件类型和语言属性将用来确定要调用的更改命令。例如，如果更改类型为 RPGSRC 的部件，可将 SEU 编辑器与其语法检查功能配合使用以编辑该源部件中存在的源。

如果不能编辑该部件，系统会提示您输入想要更改的信息。例如，如果想要更改类型为 PGM 的部件，可以只更改程序部件的属性，如与其相关联的文本。

表8显示了在 CHGCMD 参数上使用缺省值 *TYPE 时要对特定部件类型调用的命令。

*表 8. 在指定 CHGCMD(*TYPE) 时调用的 OS/400 更改命令*

命令	要更改的部件类型
CHGCMD	CMD
CHGDSPF	语言属性为 DSPF 的 FILE
CHGDTA	PARTL
CHGDTAARA	DTAARA
CHGICFF	语言属性为 ICFF 的 FILE
CHGJOBD	用于类型为 JOBD 的部件
CHGLF	语言属性为 LF 的 FILE
CHGMNU	用于语言属性为 UIM 类型为 MENU 的部件
CHGMOD	MODULE
CHGMSGQ	用于类型为 MSGQ 的部件
CHGOUTQ	用于类型为 OUTQ 的部件
CHGPF	语言属性为 PF、VRPGBIN 的 FILE
CHGPGM	PGM
CHGPRTF	语言属性为 PRTF 的 FILE
CHGSAVF	语言属性为 SAVF 的 FILE
STRSEU	BLDOPT、BNDSRC、CBL36INC、CBL36SRC、CBLINC、CBLSRC、CBLLEINC、CBLLESRC、CINC、CLDSRC、CLLESRC、CLPSRC、CLSRC、CMDSRC、CSRC、DDSSRC、DSPF36SRC、MNU36SRC、MSGF36SRC、OCL36SRC、PNLINC、PNLSRC、PRDDFN、PRDLOD、REXXSRC、RPG36INC、RPG36SRC、RPGINC、RPGSRC、RPGLEINC、RPGLESRC、RPT36SRC、SCHPTH、SRT36SRC、SYSTEML、TXT36SRC、TXTSRC 和 VRPGTXT[1]
WRKBNDDIRE	用于类型为 BNDDIR 的部件
WRKJOBQ	用于类型为 JOBQ 的部件

*表 8. 在指定 CHGCMD(*TYPE) 时调用的 OS/400 更改命令 (续)*

命令	要更改的部件类型
WRKMSGD	MSGF
WRKSCHIDXE	用于类型为 SCHIDX 的部件

注:

1. 显示 VRPGTXT 部件中的所有成员的列表。可从此列表中选择和更改个别成员。

有关更改类型为 PARTL 的部件的详情，参考第92页的『更改部件列表部件』。

对于用户定义的部件类型，会启动在 CHGADMACN 命令的 ACTION 参数上定义的 *CHG 操作。可对用户定义的类型定义的操作在第115页的表10中作了描述。

可指定 STRSDA、STRRLU 和 UPDDTA，而不使用缺省值 CHGCMD(*TYPE)（如果您在系统上安装了“应用程序开发工具箱”组件的话）。

STRSDA SDA 用来更改部件。SDA 用于语言属性为 DSPF 的 DDSSRC 部件。如果使用通过 CHGPART 命令调用的“屏幕设计辅助”实用程序编译了部件，则构建过程可能不识别这些编译，也不会更新已更改的部件之间的关系。

STRRLU RLU 用来更改部件。RLU 用于语言属性为 PRTF 的 DDSSRC 部件。

UPDDTA DFU 用来更改部件。DFU 用于数据文件。

注: 在使用 STRSDA 或 STRRLU 命令之前，应使用 ADDPRJLIBL 命令将您正在工作的项目库添加至库列表。可通过“用 PDM 来使用部件”屏幕使用选项 45（添加项目库列表）或使用 PDM 用户定义选项 AP 而不是使用 ADDPRJLIBL 命令。这确保 SDA 和 RLU 将能够找到正在引用的文件中的所有字段引用。

在 SEU 或 SDA 中更改成员源类型不会导致对应部件的语言属性发生相应更改。但是，如果使用 CHGPARTINF 命令更改了源部件的语言属性，则对应源成员的成员源类型会发生更改以匹配语言属性。

注

应注意到如果在“应用程序开发管理器”的控制范围之外更改部件，则可能会导致不可预测的结果。构建过程可能不识别已更改的部件，也可能不能重新编译或处理它。这会导致部件及其对应 OS/400 对象和成员之间产生冲突。

在 PARTL 参数中指定 PARTL 名会导致 CHGPART 命令自动将已更改的部件添加至部件列表部件。指定 PARTL(*PRV) 允许您使用用于此项目的上一个部件列表部件的名称（如果您是系统管理员的话）。但是，如果您是开发者的话，此参数会标识用于指定组的上一个部件列表部件的名称。如果使用 QSECOFR 用户简要表运行此命令，则此值无效。PARTL 参数的缺省值是 *NONE，指示正在更改的部件将不会添加至部件列表部件。

下列情况下必须指定部件列表部件:

- 在命令上指定的组是使用 PARTLREQ(*YES) 创建的
- 指定的类型不是 PARTL

如果指定了部件列表部件，则:

- 部件列表部件必须存在于从更改这些部件的组起始的缺省搜索路径中。
- 更改部件名被添加至指定部件列表部件，如果它还不存在的话。（即使命令未能更改该部件，也可能会发生此情况）。

如果指定的类型为 PARTL，则 PARTL 参数会被忽略。PARTL 参数会导致指定部件列表部件被更新。

使用 ARCHIVE 参数来指定正在替换的部件是否应归档。有关对部件进行归档的详情，参见第82页的『归档部件』。

示例

以下命令会更改组 DEVELOPER1 中的部件 RPGSRC BIWKLY。

使用 CHGPART 命令

```
CHGPART PRJ(PAYROLL) GRP(DEVELOPER1) TYPE(RPGSRC) PART(BIWKLY)   CHGCMD(*TYPE)
```

使用“编程开发管理器”实用程序

在“用 PDM 来使用部件”屏幕上选择选项 2（更改）。

CHGCMD(*TYPE) 指定部件类型和语言属性用来确定被调用以更改该部件的 OS/400 更改命令。在此示例中，部件将放在 SEU 编辑会话中。

可使用您自己的编辑器来更改部件。可编写 CL 或 REXX 程序来检索对应于“应用程序开发管理器”部件名的 OS/400 源成员名，然后使用此信息来调用编辑器。查看第87页的『检索部件信息 (RTVPARTINF)』以获取有关如何进行此操作的说明。

使用您自己的编辑器编辑完部件后，应使用 CHKINPART 命令来检入部件。该部件的时间戳记和上次更改该部件的用户会被更新。这样，下一次发出 BLDPART 命令时就会构建该部件。

转换部件的类型和语言 (CVTPART)

“转换部件”(CVTPART) 命令在同一项目和组内将作为源成员的部件转换为不同类型的部件。导入之后，将类型为 RPGSRC 的部件转换为部件类型为 RPGINC 的包含部件，或是将类型为 CSRC 的部件转换为部件类型为 CINC 的包含部件是非常有用的。

如果要转换的部件不在 CVTPART 命令上指定的组中，则不会在项目层次结构的各个组中搜索该部件。

注: ILE RPG/400 提供它自己的转换命令: “转换 RPG 源”(CVTRPGSRC) 命令。该命令会将包含 RPG/400 源的源成员转换为包含 ILE RPG/400 的成员。包含结果成员的源文件的记录长度一定不能少于 112 个字节。因此，RPGLESRC 和 RPGLEINC 部件必须驻留在记录长度至少为 112 字节的源文件中。此外，一定要使用第68页的表9中列出的限定值。

通常，CVTPART 命令会创建新部件而不更改该部件。但是，如果原始部件是源成员，且您指定正在转换的部件的源文件与用来存储原始部件的源文件应是同一文件，则应删除原始部件。新的部件和类型表示原始成员。例如，检查称为 QRPGSRC 的源文件和

称为 A 且部件类型为 RPGSRC 的成员部件。如果将此源文件转换为部件类型 RPGINC 而不更改部件名，则 CVTPART 命令将删除原始部件 RPGSRC A 并将其复制至称为 RPGINC A 的新部件。

规则

下列规则适用:

1. TOTYPE 名必须不同于 FROMTYPE 名。
2. 不能转换类型为 FILE 的部件。
3. 类型为 PGM 或 SRVPGM 的部件不能转换为类型为 PGM 或 SRVPGM 的部件。
4. 作为源成员的部件只能转换为仍作为源成员的其他部件。
5. 类型为 TXTSRC 的部件可转换为仍作为源成员的任何类型的部件。
6. 存储在源成员中的任何类型的部件可转换为类型为 TXTSRC 的部件。

限制

除上述规则之外，还可以在 CVTPART 命令上同时使用下表中列出的 FROMTYPE 和 TOTYPE。

表 9. 用于 CVTPART 命令的 FROMTYPE 和 TOTYPE 的限定值列表

Fromtype	Totype
CLD	CLDSRC
语言为 CLLE 的 PGM	CLLESRC
语言为 CLP 的 PGM	CLPSRC
RPGSRC	RPGINC、RPGLEINC、RPGLESRC
RPGINC	RPGSRC、RPGLEINC、RPGLESRC
RPGLESRC	RPGLEINC
RPGLEINC	RPGLESRC

注: 在转换部件时，构建信息不会复制至新部件。

在“用 PDM 来使用部件”屏幕上使用选项 50（转换部件）时，此 CVTPART 功能也是可用的。

示例

执行下列步骤以将类型为 TXTSRC 的部件转换为类型为 PNLSRC 且语言为 PNLGRP 的部件。

1. 使用 CVTPART 命令将部件类型转换为 PNLSRC。这会将其语言设置为 MENU。
2. 使用 CHGPARTINF 命令将其语言更改为 PNLGRP。

现在，在缺省情况下，如果将类型为 TXTSRC 的部件转换为类型为 CSRC 或 CINC 的部件，部件语言会被设置为 CLE。所以，还可使用上述步骤将类型为 TXTSRC 的部件转换为:

- 类型为 CSRC 语言为 C 的部件
- 类型为 CINC 语言为 C 的部件
- 类型为 CSRC 语言为 SQLC 的部件
- 类型为 CINC 语言为 SQLC 的部件

必需参数

指定表示想要转换其类型的部件的项目、组和类型。如果想要转换所有部件，使用缺省值 PART(*ALL)。还必须指定新类型。

可选参数

可指定:

- 正在转换的部件的名称。缺省值为 TOPART(*FROMPART)。正在转换的部件的名称保持不变。
- 提升码。缺省值为 PRMCODE(*GRP)。会使用该组的提升码。
- 应将该部件作为源成员存储其中的源文件的名称。缺省值为 SRCFILE(*TYPE)。会使用缺省源文件。如果指定 SRCFILE(*SAME)，则还会使用原始部件使用的源文件名称。如果正在转换下列部件类型，则会采用值 *TYPE: PGM 转换至 CLPSRC 和 CLD 转换至 CLDSRC。查看第43页的表3以获取部件类型及其相应缺省源文件的列表。SRCFILE 参数还适用于存储在源文件成员中的用户定义类型。
- 转换部件被添加至的部件列表部件的名称。指定 PARTL(*PRV) 允许您使用用于此项目的上一个部件列表部件的名称（如果您是系统管理员的话）。但是，如果您是开发者的话，此参数会标识用于指定组的上一个部件列表部件的名称。如果使用 QSECOFR 用户简要表运行此命令，则此值无效。PARTL 参数的缺省值是 *NONE，指示正在转换的部件将不会添加至部件列表部件。

  下列情况下必须指定部件列表部件:
  - 在命令上指定的组是使用 PARTLREQ(*YES) 创建的
  - 指定的类型不是 PARTL

  如果在 CVTPART 命令上指定了部件列表部件，则:
  - 部件列表部件必须存在于从转换这些部件的组起始的缺省搜索路径中。
  - 转换部件的名称会被添加至部件列表部件（如果它们还不存在的话）。（即使命令未能转换该部件，也可能会发生此情况）。

  如果指定的类型为 PARTL，则 PARTL 参数会被忽略。对于其他情况而言，PARTL 参数会导致指定部件列表部件更新。

比较部件 (CMPPART)

“比较部件”(CMPPART) 命令会在报告中比较两个部件和文档之间的差异。这在您必须将修正迁移至部件的下一版本时会特别有用。

可对下列类型的部件进行比较:
- 存储在源文件成员中的部件
- 部件类型为 FILE 且语言为 PF（物理文件）和 VRPGBIN 的部件
- 部件类型为 PARTL（部件列表）的部件
- 部件类型为 VRPGTXT 的部件

要比较 VRPGTXT 部件的个别本机源成员，对存储该部件的本机源物理文件使用 CMPPFM 命令。

必需参数

指定表示正在比较的新部件的项目、组、类型和部件。新部件必须存在于项目和组中。不会搜索缺省组层次结构以查找此组中找不到的任何部件。如果想要比较所有部

件，则使用 NEWTYPE(*ALL) 和 NEWPART(*ALL)。还必须指定 OLDTYPE(*NEWTYPE) 和 OLDPART(*NEWPART)。

可选参数

指定正在比较的旧部件的组、类型和部件。会搜索缺省层次结构以查找此组中找不到的任何部件。

要指定要执行比较的类型，使用 CMPTYPE 参数。使用 CMPTYPE(*LINE) 来比较和标识插入的、删除的和更新的行；使用 CMPTYPE(*FILE) 来查看两个文件的内容是不同还是相同；并使用 CMPTYPE(*WORD) 来比较和标识添加的、删除的和更新的单词。

RPTTYPE 命令用来指定报告的类型。缺省值 RPTTYPE(*DIFF) 只会列出成员之间的差异。使用 RPTTYPE(*SUMMARY) 会列出比较结果的总结，而不显示详细的差异；使用 RPTTYPE(*CHANGE) 会提供与 *DIFF 报告类型相同的信息，及这些差异的前后十行；而使用 RPTTYPE(*DETAIL) 会列出整个新文件，指示差异，并提供结果总结。

使用 OUTPUT 参数指示输出应处于的位置。缺省值为 OUTPUT(*)。报告会显示在工作站上。如果命令是以批处理方式运行的，则输出将会假脱机至打印设备的输出队列。OUTPUT(*PRINT) 会将输出假脱机至打印设备。OUTPUT(*OUTFILE) 会将输出引导至输出文件。

OPTION 参数描述想要如何执行比较。每一次可指定最多 12 个选项。有关 OPTION 参数的选项的详情，参考 *ADTS/400: File Compare and Merge Utility* 中的CMPPFM 命令。

在包含带有比较伪指令语句的成员的 STMTFILE 参数上指定库名和文件名。有关比较伪指令语句的详情，参考 *ADTS/400: File Compare and Merge Utility*。

STMTMBR 参数指定包含比较语句的成员。

示例

以下命令会将组 DEVELOPER1 中的部件 RPGSRC BIWKLY与同一组中的部件 RPGSRC BIWEEK 进行比较。报告被假脱机至打印设备。

使用 CMPPART 命令

```
CMPPART PRJ(PAYROLL) NEWGRP(DEVELOPER1) NEWTYPE(RPGSRC) NEWPART(BIWKLY)
OLDGRP(DEVELOPER1) OLDTYPE(RPGSRC) OLDPART(BIWEEK)          OUTPUT(*PRINT)
```

使用“编程开发管理器”实用程序

在“用 PDM 来使用部件”屏幕上选择选项 54（比较部件）。

合并部件 (MRGPART)

“合并部件”(MRGPART) 命令会合并指定目标组的缺省搜索路径中的特定部件或所有部件。仅存储在源文件中的部件才可以合并。在您必须将更改从部件的修正版本迁移至部件的下一版本，或是要对给定部件合并供应商更改和您自己的更改时，这会特别有用。

MRGPART 命令会将每个目标部件和维护部件与其相应根部件进行比较。这些比较的结果用来确定是否发生了更新。

要合并 VRPGTXT 部件内发现的个别本机源成员，对存储该部件的本机源物理文件使用 MRGSRC 命令。

此命令使用 MRGSRC 命令。有关 MRGSRC 命令的详情，参见 *ADTS/400: File Compare and Merge Utility* 一书。

必需参数

目标项目、目标组、目标类型和目标部件名指定部件合并后要放置的位置。**目标**是一个源部件，也可能包含与维护部件合并后的更新。合并的结果会放入目标部件。必须对合并目标部件的组具有更新访问权。该目标部件将被检出至目标组。查看附录B. 部件类型以及它们与命令的关系以获取 MRGPART 命令所允许的部件类型的列表。

将搜索缺省层次结构以获取要合并的部件的列表。

如果指定了 TGTTYPE(*ALL)，则必须指定 MAINTTYPE(*TARGET)；如果指定了 TGTPART(*ALL)，则必须指定 MAINTPART(*TARGET)。

如果指定了 TGTTYPE(*ALL)，则必须指定 ROOTTYPE(*TARGET)；如果指定了 TGTPART(*ALL)，则必须指定 ROOTPART(*TARGET)。

可选参数

维护项目、维护组、维护类型和维护部件名指定要合并的已更改部件要保存的位置。**维护**是一个源部件，包含要合并到目标部件中的更新。

根项目、根组、根类型和根部件名指定要合并的原始部件要保存的位置。**根**是源部件的原始版本，它是这两组更新的基础。

在 PARTL 参数上指定可能会添加目标部件的部件列表部件。指定 PARTL(*PRV) 允许您使用用于此项目的上一个部件列表部件的名称（如果您是系统管理员的话）。但是，如果您是开发者的话，此参数会标识用于指定组的上一个部件列表部件的名称。如果使用 QSECOFR 用户简要表运行此命令，则此值无效。PARTL 参数的缺省值是 *NONE，它指示正在合并的部件将不会添加至部件列表部件。

下列情况下必须指定部件列表部件:

- 在命令上指定的组是使用 PARTLREQ(*YES) 创建的
- 指定的类型不是 PARTL

如果在 MRGPART 命令上指定了部件列表部件，则:

- 部件列表部件必须存在于从合并这些部件的目标组起始的缺省搜索路径中。
- 合并部件的名称会被添加至部件列表部件（如果它们还不存在的话）。

如果指定的类型为 PARTL，则 PARTL 参数会被忽略。对于其他情况而言，PARTL 参数会导致指定部件列表部件更新。

如果指定了 RPTONLY(*YES)，则 PARTL 参数也不是必需的。

使用 SELECT 参数来指定是否想要显示“分割合并”屏幕，以便您可在维护部件中选择要合并到目标部件中的个别更改。如果 MRGPART 命令是以批处理方式运行的，则 SELECT 会被忽略，且不会显示“分割合并”屏幕。

使用 RPTONLY 参数来指示是想要执行合并还是只生成报告。如果在 MRGPART 命令上指定了 SELECT(*NO)，则只会提示此参数。如果指定了 SELECT(*YES)，则必须指定 RPTONLY(*NO)。

示例

以下命令会将原始版本 1 部件、已更改的版本 1 部件与新的版本 2 部件合并。

使用 MRGPART 命令

```
MRGPART PRJ(PAYROLL) TGTGRP(V2) TGTTYPE(RPGSRC) TGTPART(BIWKLY)
MAINTGRP(V1CHANGE) ROOTGRP(V1)
```

使用“编程开发管理器”实用程序

在“用 PDM 来使用部件”屏幕上选择选项 55（合并部件）。

合并样本应用程序

假定您已从供应商处接收到包含三个部件: PART A、PART B 和 PART C 的应用程序。R1ROOT 组包含供应商提供的应用程序的第一个发行版。您已经更改了 PART A 和 PART B，并创建了一个新的部件 PART D。R1MAINT 组包含您更改的和添加的首发行版部件。经过一段时间之后，现在供应商又将应用程序的第二个发行版发送给您，该版本带有更新的 PART A 和 PART C 和一个新部件 PART E。 R2ROOT 组包含的部件包含应用程序的第二个发行版。项目层次结构显示在图25的 BEFORE 部分中。

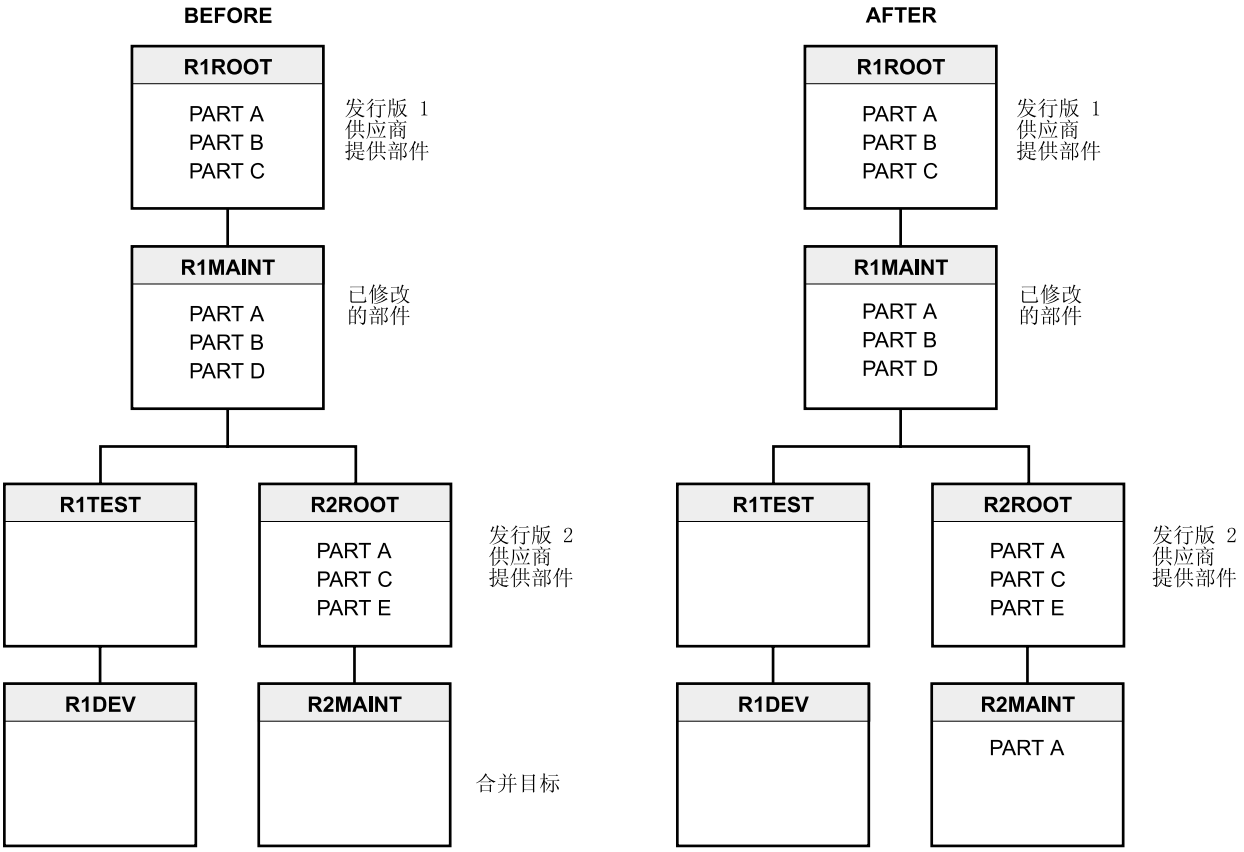

图 25. 显示合并前后的项目层次结构的示例

使用以下命令来合并 R2ROOT 中供应商的更改与您在 R1MAINT 中的更改，并将结果放在 R2MAINT 组中。

```
MRGPART PRJ(ACCTA) TGTGRP(R2MAINT) TGTTYPE(*ALL) TGTPART(*ALL)
MAINTGRP(R1MAINT) ROOTGRP(R1ROOT)
```

合并后的项目层次结构会显示在图25的 AFTER 部分中。注意，PART A 是目标组 R2MAINT 中获得检出并合并的唯一部件，因为此部件在 R1MAINT 和 R2ROOT 中分别为两个不同的版本。

执行合并之后，可能会决定在 R1MAINT 组中更改部件 PART C。如果这样的话，在更改之后，应使用上述合并过程将它与供应商为第二个发行版发送的部件 PART C 的副本合并。

为提高性能，应使用上述合并过程，除非在 MRGPART 命令上指定了 TGTPART(C)。

检入部件 (CHKINPART)

“检入部件”(CHKINPART) 命令通过将已检出部件的访问密钥复位为空白，来释放该部件。这允许共享同一开发组的其他人使用该部件。检入部件并不会释放其低位锁定，所以其他组中的开发者仍然不能访问它。仅当该部件被提升至目标组时，才会释放低位锁定。CHKINPART 命令不会将部件移至项目层次结构中的下一个组。

CHKINPART 命令还会确保更新部件的时间戳记，以与相应 OS/400 对象或成员的时间戳记相匹配。这会确保构建过程识别已更改的部件，并在下一次发出 BLDPART 命令时重新编译该部件。

检入部件的人员的用户简要表必须是检出该部件的简要表；但是，项目管理员可使用 CHGPARTINF 命令将任何部件的访问密钥复位为空白以检入该部件。（只有项目管理员才有权使用 CHGPARTINF 命令更改访问密钥。）

必需参数

指定项目、组、类型和部件。查看附录B. 部件类型以及它们与命令的关系以获取 CHKINPART 命令所允许的部件类型的列表。

如果想要同时检入几个部件，在 TYPE 或 PART 参数上使用选项 *ALL。这会检入某种类型的所有部件，或检入对您检出的所有部件。

示例

此示例说明如何检入组 DEVELOPER1 中的部件 RPGSRC BIWKLY。

使用 CHKINPART 命令

```
CHKINPART PRJ(PAYROLL) GRP(DEVELOPER1) TYPE(RPGSRC) PART(BIWKLY)
```

使用“编程开发管理器”实用程序

在“用 PDM 来使用部件”屏幕上选择选项 29（检入）。

提升部件 (PRMPART)

“提升部件”(PRMPART) 命令将指定部件以一次一组的方式从部件所在的组移至其父代组。您只能从您对其具有更新访问权的组中提升部件。提升的部件的访问密钥必须为空白或被设置为您的用户简要表。即，该部件必须是对您检出的。

如果要提升的部件被检出，在提升之前它会先自动检入。在检出部件时，该部件的访问密钥被设置为您的用户简要表，所以您对该部件具有独有的更新访问权。提升部件时，访问密钥被复位为空白，以便其他开发者可使用该部件。

VRPGTXT 及其关联 VRPGBIN 部件应使用 PRMPART 命令一起提升。这会维护 VRPG 客户机部件之间的完整性，因为 VRPGTXT 部件可能包含关联 VRPGBIN 部件的某些输出。

必需参数

指定项目、组、类型和部件。允许提升的部件类型将会得到提升（如果它们未将它们检出至其他人的话）。查看附录B. 部件类型以及它们与命令的关系以获取 PRMPART 命令所允许的部件类型的列表。

可选参数

通过构建源部件生成的部件是输出部件，且仅当您在 PRMPART 命令上指定 EXTEND(*YES) 时才能提升。EXTEND(*YES) 是能够提升带有提升码 *NONE 的部件

的唯一方法。在成功扩展提升后，PRMPART 命令将在较低组中删除 PGM 和 PF 部件。如果逻辑文件以正在提升的任何物理文件部件为基础，则在扩展提升后将不会从较低组中删除这些部件。

如果想要同时提升几个部件，可通过将 TYPE 或 PART 参数与 *ALL 选项配合使用来进行此操作。这允许您提升某种类型的所有部件，或未检出至其他人的所有类型的所有部件。一次提升几个部件的另一方法是创建部件列表部件（类型为 PARTL 的部件），并使用 PARTLOPT 参数来提升该部件列表部件。例如，如果指定 PARTLOPT(*LIST)，则会提升部件列表部件内列出的每个部件。

如果想要将已提升部件添加至部件列表部件，则应指定 PARTL 参数。指定 PARTL(*PRV) 允许您使用用于此项目的上一个部件列表部件的名称（如果您是系统管理员的话）。但是，如果您是开发者的话，此参数会标识用于指定组的上一个部件列表部件的名称。如果使用 QSECOFR 用户简要表运行此命令，则此值无效。PARTL 参数的缺省值是 *NONE，它指示正在提升的部件将不会添加至部件列表部件。

如果在该命令上指定的父代组是使用 PARTLREQ(*YES) 创建的，则必须指定部件列表部件。

在 PRMPART 命令上，可使用 PARTL 参数来指定 TYPE(PARTL)，因为此命令可展开部件列表部件，并分别处理部件列表中的每个部件。

如果在 PARTL 参数上指定了部件列表部件，则:

- 部件列表部件必须存在于从其提升这些部件的组起始的缺省搜索路径中。
- 提升部件会被添加至指定部件列表（如果它们还不存在的话）。（即使命令未能提升该部件，也可能会发生此情况）。

使用 ARCHIVE 参数来指定正在替换的部件是否应归档。有关对部件进行归档的详情，参见第82页的『归档部件』。

示例

此示例说明如何将部件 RPGSRC BIWKLY 从组 DEVELOPER1 提升至项目层次结构中的下一个组，即组 TEST。

使用 PRMPART 命令

```
PRMPART PRJ(PAYROLL) GRP(DEVELOPER1) TYPE(RPGSRC) PART(BIWKLY)
```

使用“编程开发管理器”实用程序

在“用 PDM 来使用部件”屏幕上选择选项 30（提升）。

提升由构建过程创建的输出部件

如果想要同时提升构建过程的输出部件和相关联的源部件，则使用 EXTEND 参数。

缺省值为 EXTEND(*NO)。只有在 PRMPART 命令上指定的部件才会被提升。如果指定 EXTEND(*YES)，则会同时提升指定部件及在上次构建该部件时创建的输出部件。此外，对象信息库 (OIR) 的服务数据区（存储用来编译该对象的源的源文件名、库名和成员名的位置）会被更新，以反映新的库名。由构建过程创建的部件的提升码为 *NONE。EXTEND(*YES) 是能够提升带有提升码 *NONE 的部件的唯一方法。

提升带有提升码 *NONE 的部件时，下列规则适用:

- 如果要提升至的组中已经存在提升码不同于 *NONE 的同一部件，则不能提升带有提升码 *NONE 的部件。
- 所有相关物理文件和逻辑文件必须位于同一组中。物理文件与其关联的逻辑文件一定要放在同一组中。
- 在提升该部件之后，该部件的提升码仍为 *NONE。项目管理员或对该部件被提升至的组具有 *UPDATE 访问权的任何用户都应对所有部件运行 BLDPART 命令以确保它们都是最新的。那些已经是最新的部件不会进行编译。

提升部件列表部件

如果正在提升部件列表部件，则 PRMPART 命令允许您指定提升它的方式。PRMPART 命令会以以下方式处理部件列表部件:

1. 缺省搜索路径用来搜索部件列表部件。如果未找到部件列表部件，则 PRMPART 命令失败。如果指定了 PARTLOPT(*PART)，则指定组中必须存在 PARTL 部件。
2. 如果在 PRMPART 命令上指定 PARTLOPT(*LIST)，则会提升部件列表部件中的每个部件。如果指定 PARTLOPT(*PART)，则会象提升任何其他部件一样提升部件列表部件。如果指定 PARTLOPT(*BOTH)，则部件列表部件及其内部列出的部件都会被提升。还可在 PRMPART 命令上指定类属部件名。有关详情，参见第8章 部件列表部件、原因控制和更改跟踪。

如果正在运行的命令异常结束，则可能是未处理其中的某些部件。为确保所有的部件都得到处理，应查看所有错误消息并再次运行该命令（如果必要的话）。

示例

以下命令将部件列表部件 PART001 从组 DEVELOPER1 提升至项目层次结构中的下一个组，即组 TEST:

```
PRMPART PRJ(PAYROLL) GRP(DEVELOPER1) TYPE(PARTL) PART(PART001)   PARTLOPT(*LIST)
```

提升码

本节描述提升码 (PRMCODE) 参数，它适用于 CHKOUTPART、CPYPART、CRTPART、IMPPART 和 RNMPART 命令。项目管理员会在创建特定组时定义它的提升码。

通常，在使用部件时只使用一个提升码。但是，项目管理员可能定义了几个提升路径，因此，一个项目层次结构内可能有几个提升码。如果您的组织在更新同一应用程序的另一版本时必须维护其现存版本，则可能会出现这种情况。查看第19页的『一个部件的两个版本』以获取这种情况的示例。

PRMCODE 参数指定要用来确定项目层次结构中部件的提升路径的提升码。可指定两个值。

***GRP** 会将在 GRP 参数（CPYPART 命令的 TOGRP 参数）上对组指定的提升码指定给该部件。

如果对 CHKOUTPART、CPYPART 或 IMPPART 命令指定了 *GRP，且部件的部件类型为
PGM、FILE、CLD、CMD、MENU、PNLGRP、MODULE、SRVPGM,

或是存储在由 BLDPART 命令创建的 OS/400 对象中的用户定义类型，则使用 PRMCODE(*NONE) 来处理该部件。除非使用 EXTEND 参数，否则这些类型的部件无法提升。EXTEND(*YES) 是能够提升带有提升码 *NONE 的部件的唯一方法。如果 BLDPART 命令创建的部件是存储在源成员中的部件，则由 BLDPART 命令创建的各类型的部件都可以提升。注意，类型为 PGM 且语言为 CBL36 和 RPG36 的部件的提升码可以是 *GRP。

***NONE** 不会对该部件指定任何提升码。该部件不能提升。

使用 CHKOUTPART、CHKINPART 和 PRMPART 命令

图26和第78页的图27使用样本项目层次结构来说明使用 CHKOUTPART、CHKINPART 和 PRMPART 命令前后部件发生的变化。

在图26的 BEFORE 部分中，RPGSRC 部件 BIWKLY 只存在于组 TEST 中。要对部件作一些更改。您使用 CHKOUTPART 命令将该部件检出至开发组 DEVELOPER1。这会在图的 AFTER 部分中说明。将 BIWKLY 复制至该组，并对该部件设置了低位锁定。访问密钥会设置为您的用户简要表，以阻止其他人更新该部件。

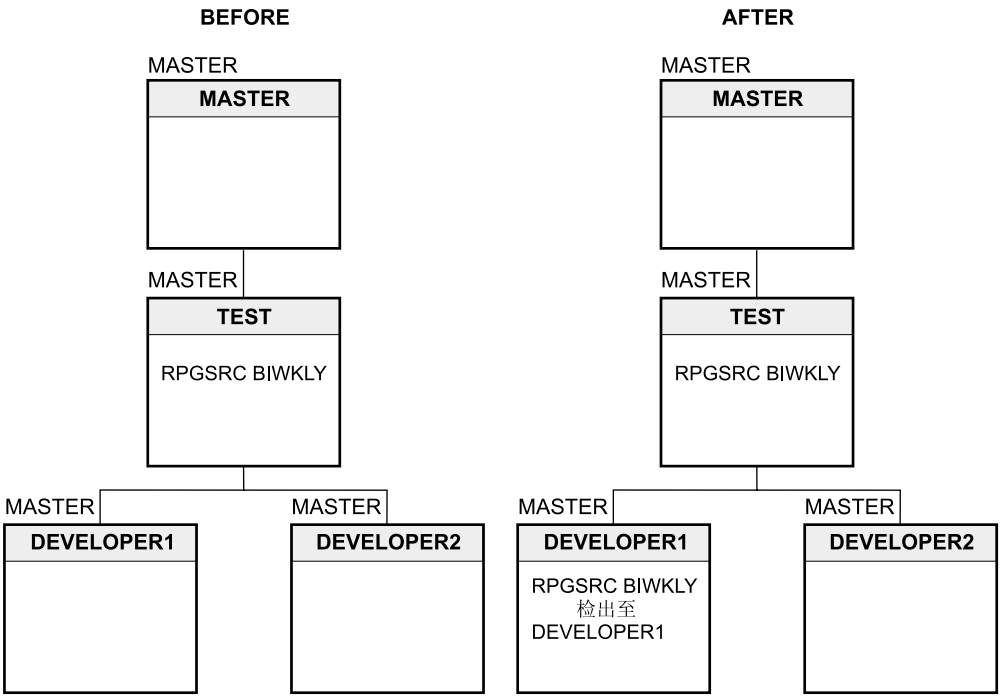

图 26. CHKOUTPART 命令

既然 BIWKLY 已检出至您的组，您就可以更改它。完成更改后，使用 CHKINPART 命令来检入该部件。该部件仍保存在 DEVELOPER1 中，如第78页的图27中 BEFORE 部分所示。如果这一组是由其他开发者共享的，则他们其中的一人可立刻检出该部件和更改它。

组 DEVELOPER2 中的开发者不能检出 BIWKLY，因为该部件已检出至组 DEVELOPER1（不管该部件是否被检出至组 DEVELOPER1 中的开发者)。该部件正等待被提升至组 TEST，且对组 DEVELOPER2 不可用。

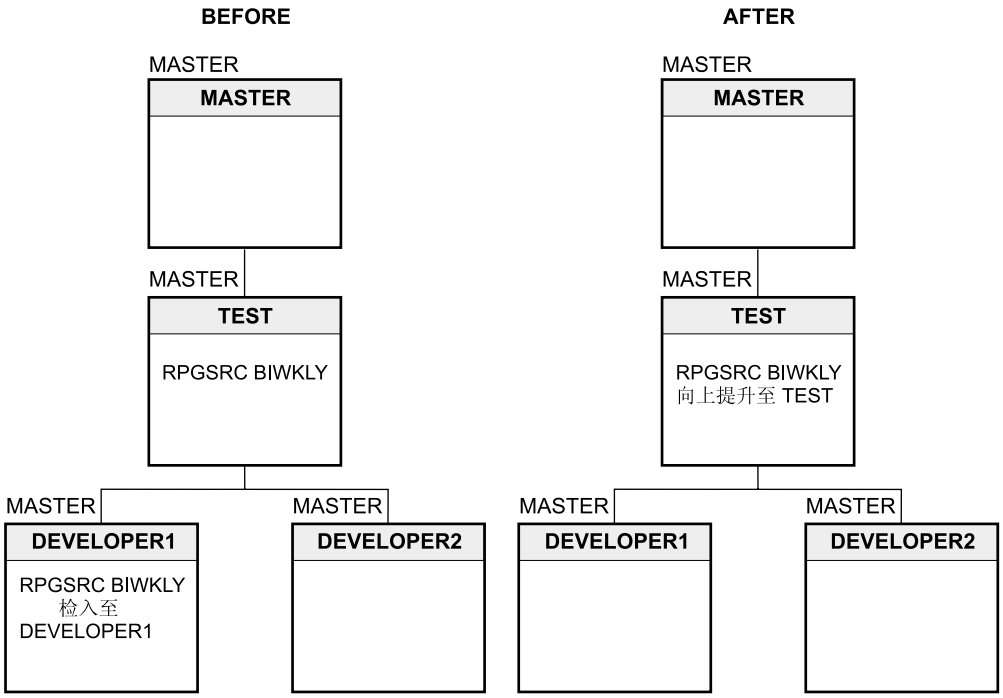

图 27. CHKINPART 和 PRMPART 命令

现在使用 PRMPART 命令来将其提升回至组 TEST。图27的 AFTER 部分显示 PRMPART 命令如何将该部件提升一个级别，移至组 TEST。如果该部件仍是对您检出的，则在将该部件提升回至组 TEST 之前， PRMPART 命令会先将该部件检入至 DEVELOPER1。将 BIWKLY 提升至组 TEST 会将 TEST 中该部件的旧副本替换为来自 DEVELOPER1 的新部件。

下列命令产生的操作显示在第77页的图26和图27中:

1. 要检出部件 RPGSRC BIWKLY，输入:

   ```
   CHKOUTPART PRJ(PAYROLL) GRP(DEVELOPER1) TYPE(RPGSRC) PART(BIWKLY)
   PRMCODE(*GRP)
   ```

2. 要更改部件 RPGSRC BIWKLY，输入:

   ```
   CHGPART PRJ(PAYROLL) GRP(DEVELOPER1) TYPE(RPGSRC) PART(BIWKLY)
   CHGCMD(*TYPE)
   ```

3. 要将部件 RPGSRC BIWKLY 检回至组 DEVELOPER1，输入:

   ```
   CHKINPART PRJ(PAYROLL) GRP(DEVELOPER1) TYPE(RPGSRC) PART(BIWKLY)
   ```

4. 要将部件 RPGSRC BIWKLY 提升回至组 TEST，输入:

   ```
   PRMPART PRJ(PAYROLL) GRP(DEVELOPER1) TYPE(RPGSRC) PART(BIWKLY)
   ```

重命名部件 (RNMPART)

使用“重命名部件”(RNMPART) 命令以在同一项目和组中将部件从一个名称重命名为另一个名称。

对于给定名称和类型的部件，仅当指定组中不存在带有新名称而类型相同的部件时，才能对其重命名。如果该组中已经存在相同名称和类型的部件，则会发出一条消息，且 RNMPART 命令失败。

重命名部件将不会维护可能已经存在的有关旧部件的构建信息。部件只会在指定组中重命名，而不会在整个项目中重命名。

必须对包含正在重命名的部件的组具有更新访问权。

如果重命名部件之前该部件已对您检出，则该部件将会重命名，但不会对您检出。如果该部件已检出至另一用户，则会发出一条消息，而 RNMPART 命令会失败。

如果想要更改复制部件的属性（如它的语言或文本），则在重命名完成后使用“更改部件信息” (CHGPARTINF) 命令。

类型为 VRPGTXT 或 VRPGBIN 的部件不能使用此命令来重命名。

必需参数

指定正在添加的部件的项目、组、类型和部件名，以及部件的新名称。必须对正在重命名的部件所在的组具有更新访问权。任何类型的部件都可使用 RNMPART 命令来重命名。

要重命名用户定义类型的部件，必须对该部件类型定义 *CPY 和 *DLT 操作。

NEWPART 参数允许您指定该部件的新名称。此名称一定要有别于在 PART 参数上指定的名称。

可选参数

使用 PRMCODE 参数来指定是否提升新部件。此参数适用于几个命令，且在第76页的『提升码』一节中作了描述。

可指定 PARTL 参数以将重命名部件添加至部件列表部件。指定 PARTL(*PRV) 允许您使用用于此项目的上一个部件列表部件的名称（如果您是系统管理员的话）。但是，如果您是开发者的话，此参数会标识用于指定组的上一个部件列表部件的名称。如果使用 QSECOFR 用户简要表运行此命令，则此值无效。PARTL 参数的缺省值为 *NONE，它指示正在重命名的部件将不会添加至部件列表部件。

下列情况下必须指定部件列表部件:

- 在命令上指定的组是使用 PARTLREQ(*YES) 创建的
- 指定的类型不是 PARTL

如果指定的类型为 PARTL，则 PARTL 参数会被忽略。对于其他情况而言，PARTL 参数会导致指定部件列表部件更新。

示例

以下命令会在 PAYROLL 项目的组 TEST 中将部件 RPGSRC BIWKLY 重命名为 RPGSRC BIWEEK。

使用 RNMPART 命令

```
RNMPART PRJ(PAYROLL) GRP(TEST) TYPE(RPGSRC) PART(BIWKLY)          NEWPART(BIWEEK)
PRMCODE(*GRP) PARTL(*NONE)
```

使用“编程开发管理器”实用程序

在“用 PDM 来使用部件”屏幕上选择选项 7（重命名）。

删除部件 (DLTPART)

使用“删除部件”(DLTPART) 命令来从组中删除部件。必须对该组具有更新访问权，且一定不能对其他人检出该部件。

必需参数

指定项目、组、类型和部件。查看第215页的『附录B. 部件类型以及它们与命令的关系』以获取可由 DLTPART 命令删除的部件类型的列表。

使用 TYPE(*ALL) 删除所有类型的部件，或使用 PART(*ALL) 删除所有部件。

必须先删除逻辑文件部件，才能删除相关联的物理文件部件。

如果正在删除的部件未被检出至其他人，则会删除该部件及“应用程序开发管理器”功能部件保存的有关该部件的任何信息。如果该部件已被检出，则不能删除它。

可选参数

如果想要从部件列表部件除去已删除的部件，则应指定 PARTL 参数。指定 PARTL(*PRV) 允许您使用用于此项目的上一个部件列表部件的名称（如果您是系统管理员的话）。但是，如果您是开发者的话，此参数会标识用于指定组的上一个部件列表部件的名称。如果使用 QSECOFR 用户简要表运行此命令，则此值无效。PARTL 参数的缺省值是 *NONE，它指示正在删除的部件将不会添加至部件列表部件。

注意，在 DLTPART 命令上， PARTL 参数是可选的，即使在该命令上指定的组是使用 PARTLREQ(*YES) 创建的。

如果指定了部件列表部件，且要删除的部件并非 PARTL 类型，则:

- 部件列表部件必须存在于从删除该部件的组起始的缺省搜索路径中。
- 已删除部件会从部件列表部件中除去。（即使命令未能删除该部件，也可能会发生此情况）。

如果指定的类型为 PARTL，则 PARTL 参数会被忽略。对于其他情况而言，PARTL 参数会导致指定部件列表部件更新。

使用 DLTARCHIVE 参数来指定是否应在删除该部件的同时删除该部件的归档版本。在删除所有归档成员时，如果源文件中没有任何其他成员，则还会删除该源文件。

示例

此示例说明如何从组 DEVELOPER1 中删除部件 RPGSRC BIWKLY。

使用 DLTPART 命令

```
DLTPART PRJ(PAYROLL) GRP(DEVELOPER1) TYPE(RPGSRC) PART(BIWKLY)
```

使用“编程开发管理器”实用程序

在“用 PDM 来使用部件”屏幕上选择选项 4（删除）。

低位锁定如何更改

如果删除了一个部件，而相同部件存在于多个组中，则低位锁定会更改为指向该部件当前检出至的组。考虑图28:

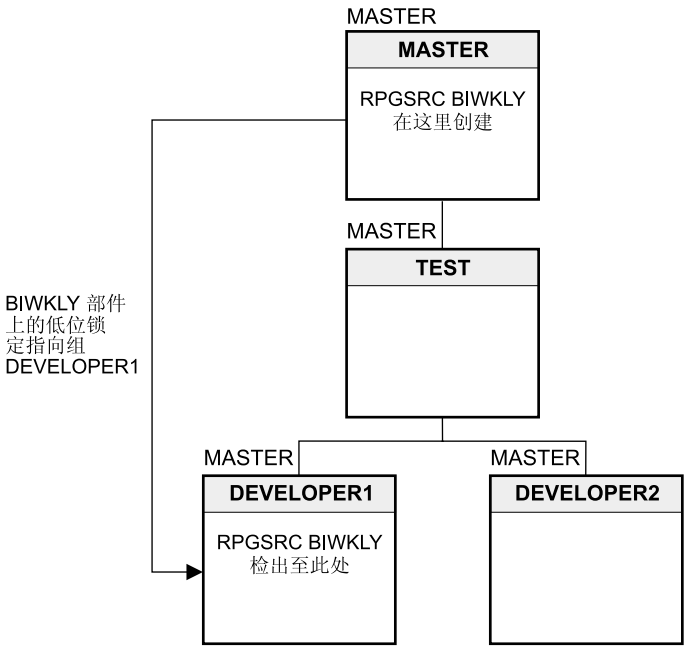

图 28. DLTPART 命令和低位锁定

假定将部件从组 DEVELOPER1 提升至组 TEST，然后将其再检出至组 DEVELOPER1。现在部件 BIWKLY 的副本存在于以下三个组中：DEVELOPER1、TEST 和 MASTER。如果在组 DEVELOPER1 中删除了该部件的副本，则低位锁定会指向组 TEST。这在第82页的图29中作了说明。

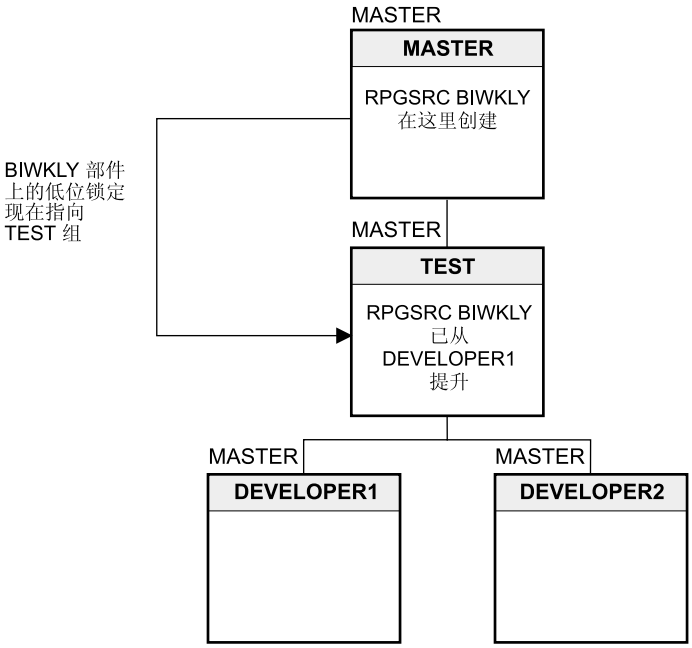

图 29. 低位锁定中的更改

项目管理员干预

因为一个部件可存在于多个组中，要从项目层次结构中彻底删除它可能需要项目管理员的干预。如果删除了一个部件，而该部件的副本仍存在于您对其没有更新访问权的另一个组中，就会出现这种情况。管理员必须删除部件的该版本。

归档部件

在“应用程序开发管理器”环境中，因为部件内容不断更改，您可能想要保存这些部件的先前版本。通过使用 ARCHIVE 参数，下列命令现在可在给定组中自动**归档**源部件的最多五个版本。

CHGPART
IMPPART
PRMPART

组归档库包含部件的先前版本，且该库是使用上述命令为该部件所驻留的组创建的。此库可包含其名称与其部件名完全相同的源文件，而每个库可包含最多五个先前版本成员。成员名为 "archive*n*"，其中 n=1 是最新版本，而 n=5 为最旧版本。

如果指定了 ARCHIVE(*YES):

- 创建组归档库（如果它还不存在的话）。组归档库的名称为 xxxx_yyyyy，其中 xxxx 是项目简称，而 yyyyy 是简短组名。组归档库的权限与组库的权限相同。
- 检查包含正在替换的部件的组的归档库，以查找带有该部件的名称的源文件。如果带有该名称的源文件不存在的话，就会创建它。
- 如果源文件包含成员 "archive5"，该成员会被删除。
- 余下的成员会重命名，以使成员名中的编号加 1。例如 "archive4" 成为 "archive5"。
- 正被替换的成员将复制至成员 "archive1"。

注: 在同一组中，可能会有两个部件使用同一名称（但具有不同类型）。这种成员将被归档至同一源文件，以便一个归档部件可能覆盖另一个。因此，明智的作法是使用唯一部件名（甚至对不同类型的部件也是这样），如果他们需要归档的话。

要检索或**回滚**归档部件，使用IMPPART 命令。在 LIBRARY 提示符上指定组归档库的名称，并在 MEMBER 提示符上指定归档成员的名称。记住以小写形式指定成员名，并用双引号引起来。可使用 REPLACE(*YES) 来将更改回滚至部件。

如果正在使用“用 PDM 来使用部件”屏幕或“用 PDM 来使用部件列表中的部件”屏幕，可在该部件旁输入选项 52（使用归档成员）以进至“用 PDM 来使用成员”屏幕，该屏幕会列出该部件的归档成员。然后，可使用用户定义选项 IM（导入部件）回滚更改。有关详情，参见第203页的『使用归档成员』和第206页的『用户定义选项』。

DLTPART 命令上的 DLTARCHIVE 参数允许您指定是否应在删除该部件本身的同时删除该部件的归档版本。在删除所有归档成员时，如果源文件中没有任何其他成员，则还会删除该源文件。

部件的关联信息

每个部件都有与其相关联的信息，您可以更改这些信息，也可以打印出来。使用“更改部件信息”(CHGPARTINF)、“打印部件信息”(PRTPARTINF) 和“检索部件信息”(RTVPARTINF) 命令来使用部件信息。

更改部件信息 (CHGPARTINF)

“更改部件信息”(CHGPARTINF) 命令可用来更改有关部件的下列信息（每次一个部件）：

- 部件的语言。第39页的表2列出部件类型及及相关联的语言。
- 持有该部件的个人的访问密钥（用户简要表）（只有项目管理员才能更改它）。
- 部件的文本描述。

如果使用 CHGPARTINF 命令来更改部件的语言，则部件的时间戳记会设置为当前时间，而上次更改该部件的人员的用户简要表会被设置为当前用户。

如果未使用 CHGPARTINF 命令更改部件的语言，则 CHGPARTINF 会确保部件的时间戳记等于该部件的对象或成员的上次更改的时间戳记。如果对语言、访问密钥和文本描述指定了 *SAME，则可能会发生这种情况。

“构建部件”命令 (BLDPART) 将不会构建类型为 CSRC、CINC 和 PGM，语言为 C 和 SQLC 的部件。为保持数据的完整性，这些部件将不会自动转换为语言 CLE 或 SQLCLE（如果这些类型的部件已存在的话）。为将语言更改为 CLE 或 SQLCLE，可使用 CHGPARTINF 命令。

必需参数

指定想要更改其信息的项目、组、类型和部件。查看附录B. 部件类型以及它们与命令的关系以获取可使用 CHGPARTINF 命令更改其部件信息的部件类型的列表。

可选参数

如果更改了部件的语言属性，且该部件的类型为第50页的表4列出的类型之一或用户定义类型，则包含该部件的对应 OS/400 源成员的源类型也会相应发生更改。用户定义源成员类型也会更改。

语言参数的缺省值为 LANG(*SAME)。它指定不更改当前语言。如果在 CHGPARTINF 命令上填入了所有必需的参数，则 *SAME 将替换为当前语言。如果不想要该部件与任何语言关联，则使用 LANG(*NONE)，或输入特定语言。查看第39页的表2以获取部件类型及其关联语言的列表。

如果想要将已更改部件添加至部件列表部件，则应指定 PARTL 参数。指定 PARTL(*PRV) 允许您使用用于此项目的上一个部件列表部件的名称（如果您是系统管理员的话）。但是，如果您是开发者的话，此参数会标识用于指定组的上一个部件列表部件的名称。如果使用 QSECOFR 用户简要表运行此命令，则此值无效。PARTL 参数的缺省值是 *NONE，它指示其部件信息正在更改的部件将不会添加至部件列表部件。

下列情况下必须指定部件列表部件:

- 在命令上指定的组是使用 PARTLREQ(*YES) 创建的
- 指定的类型不是 PARTL

如果在 CHGPARTINF 命令上指定了部件列表部件，则:

- 部件列表部件必须存在于从正在更改部件信息的组起始的缺省搜索路径中。
- 正在更改其部件信息的部件的名称会添加至部件列表部件，如果它还不存在的话。这将会进行更新，甚至在指定了 LANG(*SAME)、ACCKEY(*SAME) 和 TEXT(*SAME) 的情况下。（即使命令未能更改部件信息，也可能会发生此情况）。

如果指定的类型为 PARTL，则 PARTL 参数会被忽略。对于其他情况而言，PARTL 参数会导致指定部件列表部件更新。

要更改部件的文本描述，使用 TEXT 参数。使用 TEXT(*SAME) 保存相同描述；使用 TEXT(*BLANK) 将当前描述更改为空白；或使用 TEXT（*描述*）输入新描述。

“应用程序开发管理器”部件的文本描述最长可有 80 个字符，而与其相关联的本机对象的文本描述最长只能有 50 个字符。对象的文本描述会被截断为最长 50 个字符，而部件文本描述是使用 CHGPARTINF 命令更改的。

可使用 ACCKEY 参数来更改部件的访问密钥。这允许将新的用户简要表标识为持有部件的用户简要表，或者，可除去访问密钥锁定，以使该部件可供其他用户使用。但是，只有项目管理员才能更改 ACCKEY 参数。

示例: 以下命令会将部件 BIWKLY 的语言更改为 LF，并将其文本描述更改为空白。

使用 CHGPARTINF 命令

```
CHGPARTINF PRJ(PAYROLL) GRP(DEVELOPER1) TYPE(DDSSRC) PART(BIWKLY)
           LANG(LF) ACCKEY(*SAME) TEXT(*BLANK)
```

使用“编程开发管理器”实用程序

在“用 PDM 来使用部件”屏幕上选择选项 13（更改信息）。

打印部件信息 (PRTPARTINF)

“打印部件信息”(PRTPARTINF) 命令会打印关于部件特征的下列信息:

- 项目、组、类型和部件
- 用来编译该部件的语言
- 上次更改该部件的人员
- 上次更改部件内容的日期和时间
- 创建该部件的日期和时间
- 指定给该部件的提升码
- 低位锁定
- 访问密钥
- 对应的 OS/400 名称（库、对象、类型、成员）
- 包含在项目层次结构的该分支中出现该部件的最低位置，且提升码与此部件相同的组
- 让项目层次结构的该分支中出现该部件的最低位置上的部件检出，且提升码与此部件相同的人员的用户简要表
- 描述该部件的文本

必需参数

指定想要打印其信息的部件的项目、组、类型和部件名。查看附录B. 部件类型以及它们与命令的关系以获取可在 PRTPARTINF 命令上指定的部件类型的列表。

可选参数

使用 OUTPUT 参数指示输出应处于的位置。缺省值为 OUTPUT(*PRINT)。报告被假脱机至此作业的打印设备。OUTPUT(*OUTFILE) 会将输出引导至输出文件。

输出文件的记录格式与库 QADM 中的系统提供数据库文件 QALYPARTI 使用的格式相同。

示例: 可通过指定类似如下的命令将输出引导至输出文件。

使用 PRTPARTINF 命令

```
PRTPARTINF PRJ(PAYROLL) GRP(DEVELOPER1) TYPE(RPGSRC) PART(BIWKLY)
OUTPUT(*OUTFILE) OUTFILE(*LIBL/FILE1)          OUTMBR(*FIRST *ADD) SCAN(*YES)
```

使用“编程开发管理器”实用程序

在“用 PDM 来使用部件”屏幕上选择选项 26（打印信息）。

库列表用来确定要存储输出文件 FILE1 的位置。FILE1 中的第一个成员接收输出，并将输出数据添加至现存记录。

PRTPARTINF 命令上的 SCAN 参数指定是否搜索项目层次结构以查找该部件（如果在指定组中找不到它的话）。缺省值为 SCAN(*YES)。

示例: 以下命令打印 RPGSRC 部件 BIWKLY 的部件信息。

```
PRTPARTINF PRJ(PAYROLL) GRP(DEVELOPER1) TYPE(RPGSRC) PART(BIWKLY)
OUTPUT(*PRINT) SCAN(*YES)
```

报告被假脱机至此作业的打印设备。图30显示样本报告。

```
5722WDS   V5R1M0          应用程序开发管理器     - 部件信息报告  05/08/01   12:00:00  页面 . . :   0001

项目. . . . . . . . . . . . . . . :   PAYROLL
组. . . . . . . . . . . . . . . . :   DEVELOPER1
类型. . . . . . . . . . . . . . . :   RPGSRC
部件. . . . . . . . . . . . . . . :   BIWKLY

语言  . . . . . . . . . . . . . . :   RPG            1
上次更改（用户标识）. . . . . . . :   SMITH          2
上次更改日期. . . . . . . . . . . :   05/08/01   12:00:00    3
创建日期  . . . . . . . . . . . . :   05/08/01   12:00:00    4
提升码  . . . . . . . . . . . . . :   MASTER         5
低位锁定. . . . . . . . . . . . . :                  6
访问密钥（持有者）. . . . . . . . :                  7
系统名:
对象  . . . . . . . . . . . . . . :   QRPGSRC        8
库. . . . . . . . . . . . . . . . :     PAY.DEV1     9
类型  . . . . . . . . . . . . . . :   *FILE         10
成员. . . . . . . . . . . . . . . :   BIWKLY         11
检出至 PWS. . . . . . . . . . . . :   No             12
出现部件的最低位置的组. . . . . . :                  13
最低位置出现的部件的
持有者  . . . . . . . . . . . . . :                  14
文本  . . . . . . . . . . . . . . :                  15
                        * * * * *  列 表 结 束  * * * * *
```

图 30. 部件信息报告样本

下文描述部件信息报告中的信息:

1 部件的语言。

2 上次更改该部件的人员的用户标识。

3 上次更改该部件的日期和时间。上次更改该部件的日期和时间可能会比创建该部件的日期和时间要早。如果从项目层次结构中处于较高级别的组检出部件的话，就会出现这种情况。这样做的话，构建过程构建该部件就不会不成功。上次更改该部件的日期和时间就是从项目层次结构中处于较高级别的组中复制的部件的日期和时间。

4 创建该部件的日期和时间。

5 与该部件相关联的提升码。

6 该部件的低位锁定；为 *NONE 或组名。

7 持有该部件的人员的访问密钥或用户简要表。这是检出该部件的人员的用户简要表。如果某个人检出了一个部件，则其他任何人都不能更新它。

8 对应的 OS/400 系统对象名。

9 对应的 OS/400 系统库名。

10 对应的 OS/400 系统类型。

11 对应的 OS/400 系统成员名。如果该部件不是源成员的话，报告上就不会出现这一行。

12 指示是否将该部件检出至可编程工作站。

13 包含在项目层次结构的该分支中出现该部件的最低位置，且提升码与此部件相同的组。

14 检出项目层次结构的该分支中出现该部件的最低位置上的部件，且提升码与此部件相同的人员的用户简要表。

15 描述该部件的文本。

检索部件信息 (RTVPARTINF)

“检索部件信息” (RTVPARTINF) 命令会在 CL 或 REXX 程序内检索给定 “应用程序开发管理器” 四元名称的全限定 OS/400 对象或成员名。如果该部件是一个对象，则此命令会返回库名、对象名和对象类型。如果该部件是一个成员，则此命令会返回库名、文件名和成员名。此命令只能作为 CL 或 REXX 程序的部分运行。

在 RTVPARTINF 命令上，指定项目、组、类型和部件的名称，以及要接收库名、对象名、对象类型和成员名的 CL 变量。（查看附录B. 部件类型以及它们与命令的关系以获取可使用 RTVPARTINF 命令对其检索部件信息的部件类型的列表。）CL 变量必须以和号 (&) 开始，最多可有 10 个字符。

可使用 CL 或 REXX 程序检索对象或成员名，并使用您自己的工具对部件执行各个功能。例如，如果系统上未安装 “应用程序开发工具箱” 组件，则可以编写 CL 或 REXX 程序来检索对应于 “应用程序开发管理器” 部件的 OS/400 源对象或成员的名称，并使用此信息来调用您自己的编辑器。

第88页的图31显示样本 CL 程序 EDITPART，以说明在使用不同于 SEU 的编辑器时会如何更改源部件。样本程序会检出部件，检索有关该部件的系统信息，调用 SEU 编辑器来更改该部件，并检入该部件。需要使用 CHKOUTPART 命令才能使该部件处于 “应用程序开发管理器” 的控制之下。可创建一个类似的程序来以在以下程序中调用 SEU 编辑器的方式来调用您自己的工具。

样本程序使用 “应用程序开发管理器” 和消息。可在创建的 CL 或 REXX 程序中执行相同操作。消息记录将显示该程序已执行的操作。

```
/*********************************************************************/
/*                                                                   */
/* A sample program that illustrates how a source part could         */
/* be changed when using an editor other than SEU.  The project,     */
/* group, type, and part names are passed to the program. The        */
/* program issues the CHKOUTPART command, calls an editor (SEU is    */
/* used as an example), and then issues the CHKINPART command.       */
/*                                                                   */
/*********************************************************************/
            PGM        PARM(&PRJ &GRP &TYP &PART)
/*********************************************************************/
/*                                                                   */
/* The following variables are the parameters passed to this         */
/* program.  The variables resolve to the name of the project,       */
/* group, type, and part.                                            */
/*                                                                   */
/*********************************************************************/
            DCL        VAR(&PRJ) TYPE(*CHAR) LEN(32)
            DCL        VAR(&GRP) TYPE(*CHAR) LEN(32)
            DCL        VAR(&TYP) TYPE(*CHAR) LEN(10)
            DCL        VAR(&PART) TYPE(*CHAR) LEN(10)
/*********************************************************************/
/*                                                                   */
/* The following variables receive the information returned from     */
/* the RTVPARTINF command.                                           */
/*                                                                   */
/*********************************************************************/
            DCL        VAR(&OBJ) TYPE(*CHAR) LEN(10)
            DCL        VAR(&LIB) TYPE(*CHAR) LEN(10)
            DCL        VAR(&TYPE) TYPE(*CHAR) LEN(10)
            DCL        VAR(&MBR) TYPE(*CHAR) LEN(10)
/*********************************************************************/
/*                                                                   */
/* The part is checked out to the user running this program.         */
/* Any messages from the Application Development Manager             */
/* product are monitored.  The part must be checked out so that      */
/* it remains under the control of the Application Development       */
/* Manager feature.                                                  */
/*                                                                   */
/*********************************************************************/
            CHKOUTPART PRJ(&PRJ) GRP(&GRP) TYPE(&TYP) PART(&PART)
            MONMSG     MSGID(ADM0000)
```

图 31. 使用 RTVPARTINF 命令的样本 CL 程序 EDITPART (1/2)

```
/*****************************************************************/
/*                                                               */
/* The RTVPARTINF command retrieves the information              */
/* necessary to edit the part.                                   */
/*                                                               */
/*****************************************************************/
            RTVPARTINF PRJ(&PRJ) GRP(&GRP) TYPE(&TYP) PART(&PART) +
                         OBJLIB(&LIB) OBJ(&OBJ) OBJTYPE(&TYPE) +
                         MBR(&MBR)
            MONMSG     MSGID(ADM0000)
/*****************************************************************/
/*                                                               */
/* The editor that is being used can now be called. In this example */
/* it is assumed that the editor can be started from a command.     */
/* In any case, the library name is contained in &LIB, the   */
/* source file name is contained in &OBJ, and the     */
/* part name is contained in &MBR.  The line below would be         */
/* replaced with the actual call to the editor.                     */
/*                                                               */
/*****************************************************************/
            STRSEU     SRCFILE(&LIB/&OBJ) SRCMBR(&MBR) OPTION(2)
            MONMSG     MSGID(CPF0000)
/*****************************************************************/
/*                                                               */
/* When the editor is ended, the part must be checked in. This      */
/* informs the Application Development Manager     feature that     */
/* the part has been changed.                                    */
/*                                                               */
/*****************************************************************/
            CHKINPART  PRJ(&PRJ) GRP(&GRP) TYPE(&TYP) PART(&PART)
            MONMSG     MSGID(ADM0000)
            ENDPGM
```

图 31. 使用 RTVPARTINF 命令的样本 CL 程序 EDITPART (2/2)

第8章 部件列表部件、原因控制和更改跟踪

部件列表部件是部件类型为 PARTL 的部件，包含可作为逻辑分组处理的部件的列表。当要进行下列操作时会发现此部件类型非常有用:

- 一次提升一组特定部件
- 一次构建一组特定部件
- 一次导出一组特定部件
- 跟踪已更改的一组特定部件

本章提供有关下列主题的信息:

- 创建部件列表部件
- 更改部件列表部件
- 显示部件列表部件
- 打印部件列表部件
- 跟踪问题
- 更改 PARTL 参数缺省值

创建部件列表部件

可使用 CRTPART 命令创建部件列表部件。

示例

以下命令在项目 PAYROLL 的组 DEVELOPER1 中创建部件列表部件。创建的第一个部件将为 PAY000001。

```
CRTPART PRJ(PAYROLL) GRP(DEVELOPER1) TYPE(PARTL) PART(*GENERATE) LANG(*DFT)
TEXT('My first part-list part')
```

必须指定部件类型 PARTL 和语言 *DFT。如果创建类型为 PARTL 的部件，则缺省源文件参数不适用。

要实施原因控制，可使用一致命名约定自动生成的部件列表部件名称立刻创建这些部件。可将 CRTPART 命令与 PART(*GENERATE) 和 TYPE(PARTL) 配合使用来创建该部件。有关详细信息，参见第41页的『创建新部件 (CRTPART)』。

在部件列表部件内，创建想要作为逻辑分组处理的部件列表。例如，在上述部件列表部件中，可添加下列条目:

```
RPGSRC PROGRAM1        RPGSRC PROGRAM2        RPGSRC PROGRAM3
```

还可使用类属部件类型或部件名。星号 (*) 和问号 (?) 的使用规则在第33页的『必需参数』中作了描述。使用 *ALL 指示应使用所有名称。使用 **ALL 来指示应使用以 ALL 结束的所有名称。

以下部件列表条目将导致系统处理前一示例中显示的相同部件。

```
RPGSRC PROGRAM*
```

在使用部件列表部件时，记住下列规则:

- 可在部件列表部件内列出任何类型的部件，包括其他部件列表部件。
- 对于被列出多次的部件，只会处理一次。
- 部件可能会／也可能不会按它们出现在列表中的次序进行处理。处理取决于正在执行的命令和列表中的部件类型。
- 已列出但不存在的部件会被忽略。
- 如果指定了类属类型值（如 P*），而列出的部件集包含类型为 PARTL 的部件，则 PRMPART 和 BLDPART 命令将展开部件列表。
- 如果在 TYPE 参数上指定了 PARTL，且在 PART 参数上指定了类属部件值（如 ABC*）:
  - PRMPART 命令会根据在 PARTLOPT 参数上指定的 *LIST 或 *BOTH 值扩展部件。
  - BLDPART 命令会扩展类型为 PARTL 的部件内的部件列表。

查看 BLDPART、EXPPART 和 PRMPART 命令以获取如何通过这些命令使用部件列表部件的描述。对于所有其他部件开发命令，类型为 PARTL 的部件的处理与其他部件类型的处理方式相同；即，会处理 PARTL 部件本身，而不是其中列出的部件。查看第215页的『附录B. 部件类型以及它们与命令的关系』，它显示了允许在 TYPE 参数上指定 PARTL 的命令。

更改部件列表部件

如果想要使用CHGPART 命令更改类型为 PARTL 的部件，会调用系统提供的“数据文件实用程序”(DFU)，同时会出现以下屏幕。DFU 是与“应用程序开发管理器”功能部件配合使用的 Application Development ToolSet/400 实用程序。有关此实用程序的详情，参考*ADTS/400: Data File Utility*。

如果使用 CHGPART 命令来显示类型为 PARTL 的部件，会出现以下屏幕。

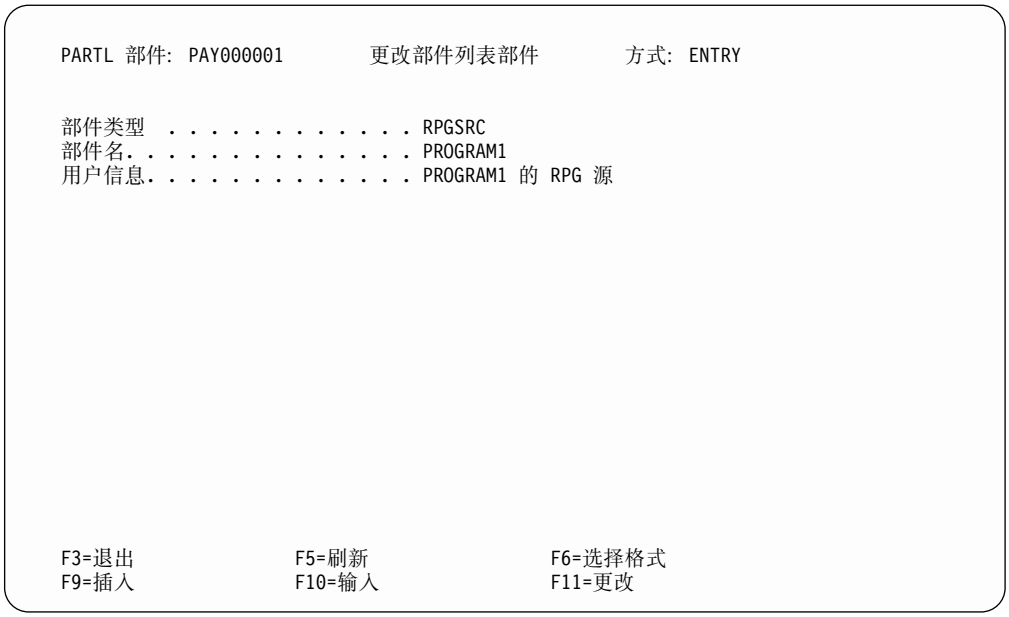

图 32. 更改部件列表部件屏幕

对*部件类型*和*部件名*字段输入值。如果指定的值在部件列表中已经存在，则会自动进入更改方式。这允许您更新所有现存部件列表条目，它等效于按 “F11=更改”。使用滚动键上下滚动以浏览部件列表部件条目。

按 “F1=帮助” 查看有关此屏幕的信息。

按 “F3=退出” 退出当前任务并返回至开始任务的屏幕。

按 “F5=刷新” 清除屏幕上的输入提示。

按 “F6=选择格式” 进至 “DFU 选择格式键” 屏幕。对于类型为 PARTL 的部件，只有一个记录格式。

按 “F9=插入” 或 “F10=输入” 将新部件名添加至部件列表。该部件不一定要存在就可以将其名称添加至部件列表。

按 “F11=更改” 更改与部件列表中的部件相关联的用户信息。

按 “F14=记录推进” 进至部件列表中的下一个部件。

按 “F21=状态” 查看此 DFU 程序的状态。

按 “F23=删除” 从部件列表中删除当前列出的部件的记录。再按 F23 确认删除操作。

还应将下列条目添加至 PAY000001 部件列表部件:

```
      RPGSRC PROGRAM2  PROGRAM2 的 RPG 源       RPGSRC PROGRAM3 PROGRAM3 的 RPG
源
```

尽管未对正在运行的系统提供 DFU 程序定义任何自动字段，但仍然对这些屏幕定义了下列功能键。

F18 从上次更改自动增加

F19 从文件结束自动递增

F20 自动记录推进

F22 自动复制

显示部件列表部件

如果使用 DSPPART 来显示类型为 PARTL 的部件，会出现以下屏幕。

```
                          显示物理文件成员
文件 . . . . . :   PAY000001           库  . . . . :   PAY.DEV11
成员 . . . . . :   QALYPRTL            记录  . . . :   1
控制 . . . . . :   _________________    列 . . . . . :   1
查找 . . . . . :   ____________________________________

RPGSRC    PROGRAM1   PROGRAM1 的 RPG 源
RPGSRC    PROGRAM2   PROGRAM2 的 RPG 源
RPGSRC    PROGRAM3   PROGRAM3 的 RPG 源
                   ****** 数据结束 ******

                                                              底部
F3=退出    F12=取消    F19=向左    F20=向右    F24=其余键
```

图 33. 显示物理文件成员屏幕

使用此屏幕查看类型为 PARTL 的部件列表部件中列出的部件。

打印部件列表部件

将 DSPPART 命令与 OUTPUT(*PRINT) 配合使用以打印类型为 PARTL 的部件。假脱机文件是使用名称 QPLYPRTL 创建的。

跟踪问题

更改管理的重要特性就是跟踪问题。这种问题跟踪又称为**原因控制**。项目管理员可对任何特定组打开原因控制，这要求开发者指定部件列表部件名（**原因**）以记录该组中正在进行处理的部件。无论何时在组中创建部件或更改部件，或者无论何时将部件提升至组，都需要指定部件列表（原因）。这将帮助管理员和审计员更有效地跟踪因请求的需求或增强功能而造成的更改。

下列“应用程序开发管理器”管理员命令提供原因控制支持:

更改组 (CHGGRP)

创建组 (CRTGRP)

打印项目 (PRTPRJ)

在部件开发命令上，在 PARTL 参数上指定部件列表部件名会导致对其执行部件命令的部件自动添加至指定部件列表，或者，如果是在 DLTPART 命令上进行此操作，部件会自动从部件列表中除去，从而减少部件列表部件的维护。如果正在处理的部件就是在 PARTL 参数上指定的部件，则此规则例外。在这种情况下，此部件列表部件的名称不会以同名（在 PARTL 参数上指定）添加至类型为 PARTL 的部件或从该部件中除去。

如果在创建或更改组时指定了 PARTLREQ(*YES)，则在该组中发出部件命令的开发者必须在命令上指定有效的部件列表部件名。这使得管理员能够确保针对给定需求和增强功能（原因）跟踪所有更改。如果命令失败，可能需要使用 CHGPART 命令从列表中删除该部件列表部件。

在 PARTL 参数中指定的部件列表部件必须存在于在命令上指定的组的缺省搜索路径中。如果是 CPYPART 命令则例外，在这种情况下，该部件必须存在于复制目标的组的缺省搜索路径中，作为正在该目标组中创建的新部件。

下列“应用程序开发管理器”部件开发命令提供使用 PARTL 参数的原因控制支持:

构建部件 (BLDPART)
更改部件 (CHGPART)
更改部件信息 (CHGPARTINF)
转换部件 (CVTPART)
复制部件 (CPYPART)
创建部件 (CRTPART)
删除部件 (DLTPART)
导入部件 (IMPPART)
合并部件 (MRGPART)
提升部件 (PRMPART)

多个开发者可通过在部件开发命令上的 PARTL 参数中指定相同部件列表部件以同时更新该部件。

更改 PARTL 参数缺省值

对于下列“应用程序开发管理器”部件开发命令，系统提供的 PARTL 参数缺省值是 *NONE:

构建部件 (BLDPART)
更改部件 (CHGPART)
更改部件信息 (CHGPARTINF)
转换部件 (CVTPART)
复制部件 (CPYPART)
创建部件 (CRTPART)
删除部件 (DLTPART)
导入部件 (IMPPART)
合并部件 (MRGPART)
提升部件 (PRMPART)
重命名部件 (RNMPART)

指定 PARTL(*PRV) 允许您使用用于指定项目的上一个部件列表部件的名称（如果您是项目管理员的话）。但是，如果您是开发者，指定 PARTL(*PRV) 允许您使用用于指定组的上一个部件列表部件的名称。如果使用 QSECOFR 用户简要表指定它的话，则 PARTL(*PRV) 无效。

对于 QSYS 库中的任何或所有上述部件开发命令，都可由系统管理员使用 CHGCMD 命令将 PARTL 参数的缺省值从 *NONE 永久更改为 *PRV。

第9章 使用搜索路径

本章描述:

- 搜索路径的基本原则
- 创建搜索路径部件之前要考虑的事项
- 如何创建搜索路径部件（类型为 SCHPTH 的部件），包括
  - 创建交叉项目搜索路径
  - 创建要包括用户库的外部搜索路径
- 如何处理搜索路径部件

了解搜索路径

在使用利用搜索路径的命令之前，应对它们进行了解。典型项目在项目层次结构中都组织有几个组。因此，典型项目都有几个版本的源和输出。就象早前定义的一样，搜索路径是确定“应用程序开发管理器”功能部件在项目层次结构中搜索部件所依照的次序的组的排列。有四种类型的搜索路径。

缺省搜索路径

搜索路径沿着项目的组层次结构的分支从特定组向上搜索，直至并包括根组。**根组**是项目层次结构中的第一个组或顶部的组。在第98页的图34中，从组 DEVELOPER1 开始的缺省搜索路径为 DEVELOPER1 至 TEST 至 MASTER。

备用搜索路径

通过组序列的搜索路径不同于通过项目层次结构中的分支的搜索路径。在第98页的图34中，备用搜索路径可定义为只包括组 MASTER，或包括组 TEST 和 MASTER，或者可能的话，包括 DEVELOPER3 和 MASTER 组。

交叉项目搜索路径

包括多个项目中的组的备用搜索路径。交叉项目搜索路径在部件被多个项目引用时非常有用。应设置一个单独的项目来保存这些公共部件，然后将此项目包括在引用公共部件的每个项目的搜索路径部件中。

外部搜索路径

外部搜索路径是一条备用搜索路径，包括一个或多个**外部库**。外部库是在“应用程序开发管理器”环境外部找到的用户库。在要一部分应用程序处于“应用程序开发管理器”的控制之外并仍然能够引用它的情况下，外部搜索路径非常有用。

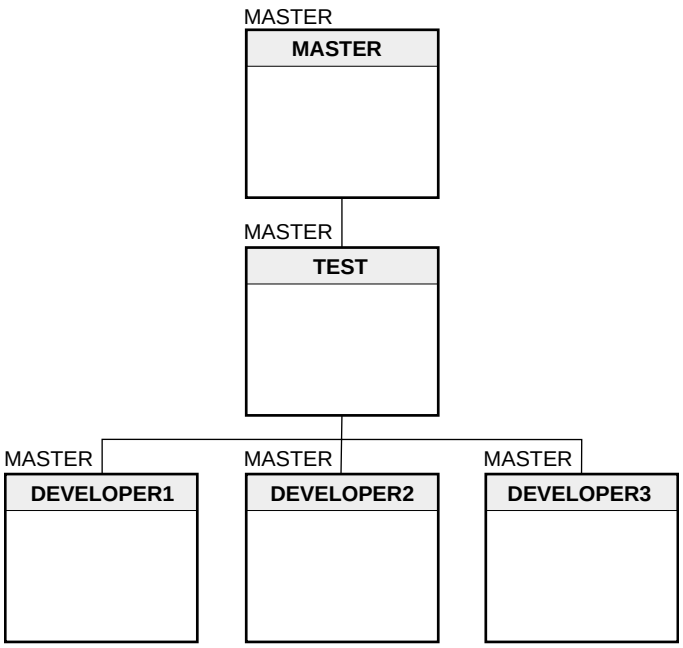

图 34. 说明搜索路径概念的样本项目层次结构

虽然一个项目只有一个项目层次结构，但一个项目可有多个搜索路径，而且因此会有多个搜索路径部件（类型为 SCHPTH 的部件）。SCHPTH 部件包含用来确定查找指定部件时搜索组的次序的组列表。

在编写下列命令以使用备用或交叉项目搜索路径来查找部件时，需要搜索路径部件:

添加项目库列表 (ADDPRJLIBL)
构建部件 (BLDPART)
导出部件 (EXPPART)
查询部件 (QRYPART)

在编写下列命令以使用外部搜索路径来查找“应用程序开发管理器”环境外部的部件时，也需要搜索路径部件:

添加项目库列表 (ADDPRJLIBL)
构建部件 (BLDPART)

ADDPRJLIBL 和 BLDPART 命令会更改当前作业的库列表的用户部分，以对应缺省搜索路径中的组的次序或搜索路径部件中列示的组的次序。库的用户部分中已经存在的所有项目库将被除去。组成搜索路径的项目库会被添加至库前面的库列表。项目层次结构中最低位置的组表示要搜索的第一个组，或者，使用库列表类推的话，将是库列表中的第一个库。BLDPART 命令会在命令完成时恢复库列表，而 ADDPRJLIBL 命令则不会。

QRYPART 和 EXPPART 命令不会更改库列表，但仍会使用搜索路径来查找部件。对于这些命令， SCHPTH 部件中发现的其他项目中的任何组都会被忽略。如果已经将带有多个分支的项目层次结构定义为根据语言或客户（基于定制）来管理应用程序的版本，则 SCHPTH 部件会非常有用。它们允许诸如 BLDPART 之类的命令搜索不一定在缺省搜索路径中的组中的部件的版本。

组及其与库列表的关系

项目层次结构内的搜索路径对应库列表。编译器使用库列表来查找包含文件和解析外部引用。图35说明项目层次结构与库列表之间的关系。

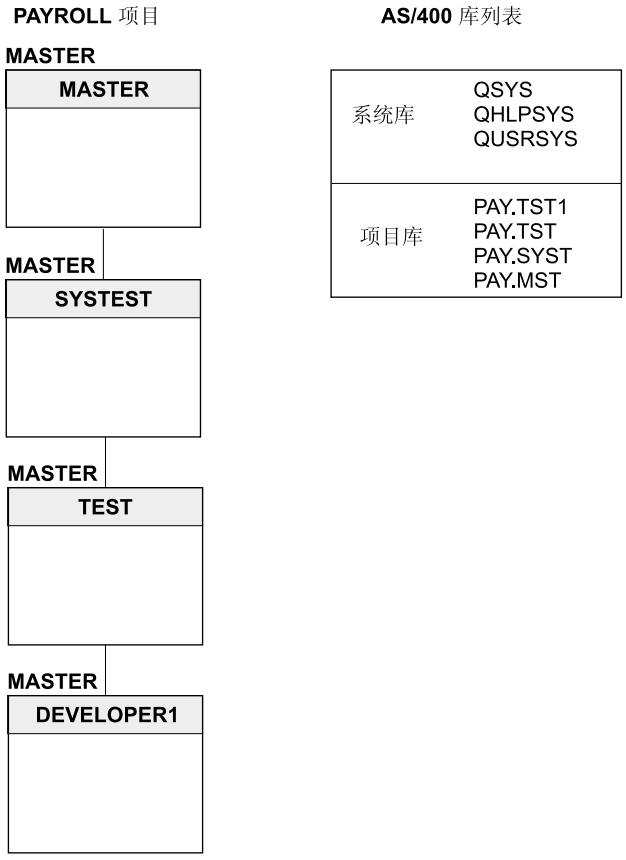

图 35. Payroll 项目及其对应的库列表

每一组都有一个库。此库的名称由用句点隔开的项目简称和简短组名组成，例如，PAY.SYST。

创建搜索路径部件

如果需要更改查找部件时搜索组的次序，则需要搜索路径部件；（即，如果需要使用备用搜索路径、交叉项目搜索路径或外部搜索路径）。搜索路径部件允许您创建项目层次结构的临时视图，该视图不同于项目层次结构本身创建的视图。

通常不需要创建搜索路径部件。但是，可能需要的一个搜索路径类型是－例如，交叉项目搜索路径。查看第101页的『创建交叉项目搜索路径』以获取有关此类型的搜索路径的详情。

类型为 SCHPTH 的部件的创建和管理方式类似于其他“应用程序开发管理器”部件，但是您不能构建它。要创建搜索路径部件，使用 CRTPART 命令并在 TYPE 参数上指定 SCHPTH。搜索路径部件最初包含构成缺省搜索路径的组列表，并以在 CRTPART 命令上指定的组开头。考虑图36:

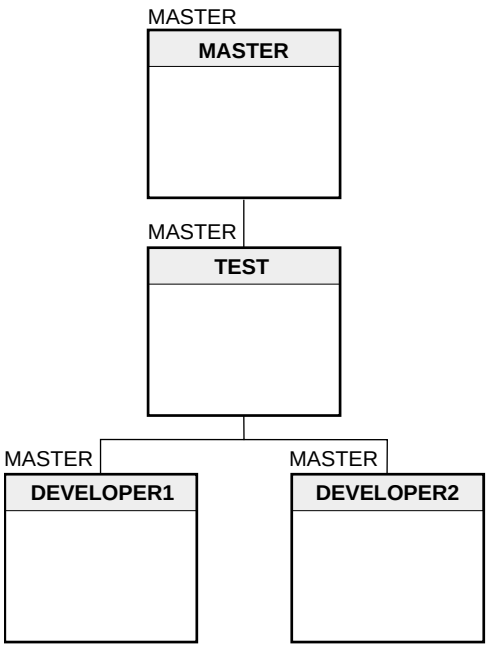

图 36. 缺省搜索路径

对于此项目层次结构，组 DEVELOPER1 中的缺省搜索路径为: DEVELOPER1 至 TEST 至 MASTER。这表示，如果正在组 DEVELOPER1 中工作，且要创建新的搜索路径部件，它会类似如下所示。

```
PAYROLL DEVELOPER1
PAYROLL TEST
PAYROLL MASTER
```

开发者不应将名称 QDFT 给予他们创建的搜索路径部件。项目管理员应创建搜索路径部件 QDFT。其目的是允许在带有搜索路径 (SCHPTH) 参数的命令上使用值 *DFT。

每一行由后跟组名的项目名组成。搜索路径部件中的列表会被从头至尾搜索，类似于库列表用来查找对象的方式。例如，如果在使用上面显示的 SCHPTH 部件的命令上指定组 DEVELOPER1，则首先会在组 DEVELOPER1，然后是 TEST，然后是 MASTER 中搜索该部件。如果在命令上指定组 TEST，则对该部件的搜索将会从 TEST 开始，如果找不到的话，则移至 MASTER。基于图36的备用搜索路径的搜索路径部件可能包含下列信息。

```
PAYROLL DEVELOPER2
PAYROLL MASTER
```

在编辑类型为 SCHPTH 的部件时，记住下列规则:

- 首先出现项目名。前导空格将被忽略。
- 项目名后必须跟一个或多个空格。
- 接着会出现组名。
- 该行上的所有剩余数据会被忽略。
- 不允许任何注释行。

记住创建类型为 SCHPTH 的部件时，这些部件可以象其他部件一样提升。这表示项目层次结构中可存在一个搜索路径部件的多个不同版本。构建部件时，可能会不小心使用到未知备用搜索路径或交叉项目搜索路径。

在“编程开发管理器”实用程序中使用搜索路径时，需要记住一些注意事项。参考第204页的『更改会话缺省值』以获取有关在使用此实用程序时指定搜索路径的详情。

创建交叉项目搜索路径

交叉项目搜索路径是另一种备用搜索路径。如果已经设置了一个项目以包含几个项目公共的文件和其他部件，则这种搜索路径会非常有用。通过使用交叉项目搜索路径，可以迅速地查找特定应用程序公共的那些部件。在类型为 SCHPTH 的部件内，可定义包括来自于另一项目的组的搜索路径。在图37中，在搜索部件时会使用两个项目层次结构。

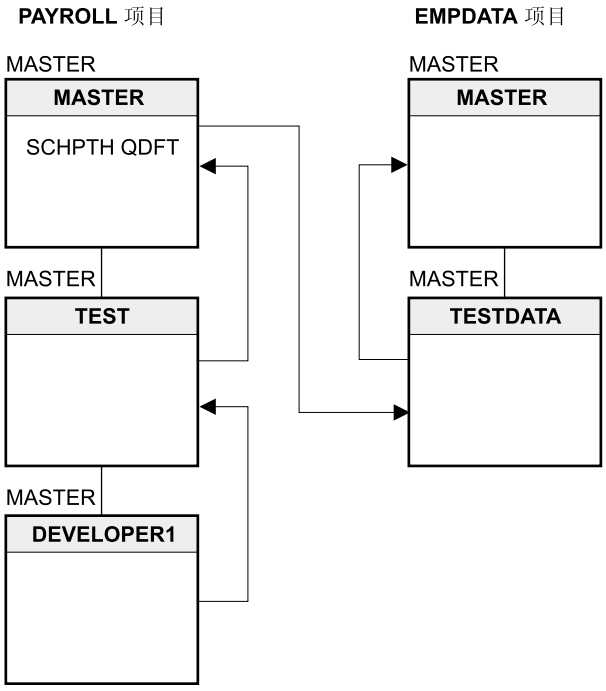

图 37. 交叉项目搜索路径

搜索路径部件 PAYROLL MASTER SCHPTH QDFT 现在包含下列搜索路径信息。

```
PAYROLL DEVELOPER1
PAYROLL TEST
PAYROLL MASTER
EMPDATA TESTDATA
EMPDATA MASTER
```

如果想在几个项目中共享*公共信息*，则交叉项目搜索路径会非常有用。公共信息可能包括多个应用程序中使用的高级语言程序、您的组织创建并定期更改的标准包含文件、多个应用程序公共的数据描述规范 (DDS) 或字段引用文件。

交叉项目搜索路径与备用搜索路径是以同一方式创建和处理的。但是，在使用交叉项目搜索路径时要记住一些规则。这些规则包括:

- 在处理 BLDPART 命令时，交叉项目组中找到的所有相关部件都会被添加至搜索路径。
- 构建过程会在交叉项目组中搜索和使用表示包含文件和外部描述文件的部件。
- 使用交叉项目搜索路径的 ADDPRJLIB 和 BLDPART 命令不会在共享项目中使用或搜索 BLDOPT 或 SCHPTH 部件。

- BLDPART 命令不会处理交叉项目组中找到的部件。例如，如果 DDSSRC 源部件已在共享项目中作了更改，就不会创建新的 FILE 部件。构建过程使用 FILE 部件的现存副本来确定是否必须编译引用 FILE 的部件。
- 如果部件在共享项目中已经是旧的，构建过程不会发出警告或错误消息。管理员应确保无论何时在共享项目中作出更改，都会使用 BLDPART 命令来构建已更改部件。

创建外部搜索路径

外部搜索路径是另一种备用搜索路径。如果想要访问应用程序中处于“应用程序开发管理器”环境之外的某些部件，则这种搜索路径会非常有用。以用户库中找到的供应商的软件应用程序（包含几百个部件）为例。您可能想要更改某些部件，但又不想让其他部件置于“应用程序开发管理器”的控制之下。这样的话，就可使用外部搜索路径来引用用户库中的未更改部件，就好象是“应用程序开发管理器”构建此应用程序一样。

将特殊值 *USRLIB 指定为项目名允许您将一个或多个外部库添加至搜索路径部件。如果对 *USRLIB 进行了编码，则在该条目上指定的库会被添加至紧跟在“应用程序开发管理器”项目库后面出现的库列表的用户部分。此算法确保构建会首先搜索“应用程序开发管理器”项目中的对象。仅当在这些项目中找不到对象时，才会搜索用户库。

例如，您可能想要将处于“应用程序开发管理器”控制之下的外部库添加至类似于第100页的图36中显示的项目层次结构。我们可以认为此外部库就是称为 SOFTLIB 的软件库。如果想要将 SOFTLIB 添加至库列表的用户部分，应将外部搜索路径部件更改为包含下列信息:

```
PAYROLL DEVELOPER1
PAYROLL TEST
PAYROLL MASTER
*USRLIB SOFTLIB
```

外部搜索路径与备用搜索路径是以同一方式创建和处理的。但是，在使用外部搜索路径时要记住一些规则。这些规则包括:

- 会首先搜索搜索路径部件来查找当前组。包括当前组后面出现的用户库的所有组都会被添加至库列表。建议所有 *USRLIB 条目应放在“应用程序开发管理器”项目库的所有条目之后。
- 如果发出带有搜索路径部件名的 ADDPRJLIBL 命令，则会将外部库添加至该搜索路径。如果在 *USRLIB 条目上指定的用户库已经存在于库列表之中，则 ADDPRJLIBL 将忽略该条目。
- RMVPRJLIBL 不能除去使用外部搜索路径的 ADDPRJLIBL 命令添加的用户库。
- 构建过程首先会搜索和使用“应用程序开发管理器”项目中的部件。仅当该项目中找不到这些部件时，构建才会在用户库中搜索这些部件。记住，仅当在 BLDPART 命令上指定的“应用程序开发管理器”项目中找到源部件时，构建过程才会构建它。

有关使用外部库的详情，参见第159页的『将外部库添加至库列表』。

如何处理搜索路径部件

搜索路径是使用 SCAN 和 SCHPTH 参数处理的。有三个命令会同时使用 SCAN 和 SCHPTH 参数（BLDPART 命令只使用 SCHPTH 参数）:

添加项目库列表 (ADDPRJLIBL)
导出部件 (EXPPART)
查询部件 (QRYPART)。

SCAN 和 SCHPTH 参数可相互配合使用。使用 SCAN 参数来指定是否搜索项目层次结构来查找部件（如果在指定组中找不到它的话）。SCAN(*YES) 会搜索搜索路径部件并使用在该部件中定义的搜索路径。如果找不到该部件的话，会使用缺省搜索路径。SCAN(*NO) 会将对部件的搜索范围限制为您指定的组。使用 SCHPTH 参数来指示搜索路径部件的名称。

如果指定 SCAN(*YES)，则用来查找指定部件的搜索路径会以如下方式确定:

1. 如果指定了缺省 SCHPTH(*DFT)，则此命令会在以给定组开头的缺省搜索路径中查找 SCHPTH QDFT 部件。
   - 如果找到了 SCHPTH QDFT 部件，则会使用在 SCHPTH QDFT 中定义的搜索路径来搜索该部件。
   - 如果 SCHPTH QDFT 在缺省搜索路径中不存在，则缺省搜索路径本身会用作该搜索路径。会从指定组至根组来搜索路径中的所有组，直到找到该部件为止。
2. 如果指定了 SCHPTH 部件的名称，则会在以给定组开头的缺省搜索路径中搜索该部件。
   - 如果找到该部件的话，在其中定义的搜索路径会用来搜索该部件。SCHPTH 部件必须存在。
   - 如果未找到部件 SCHPTH 的名称，则会接收到错误消息。

在搜索路径部件中指定组而不列出它们

要指示要在创建搜索路径部件时使用项目层次结构的缺省搜索路径，使用特殊值 *DFT 而不是组名。它指示要使用项目层次结构的缺省搜索路径。查看第204页的『更改会话缺省值』以了解搜索路径的讨论及使用“编程开发管理器”实用程序时它们所造成的影响。

示例

搜索路径 PAYROLL *DFT 会生成第100页的图36中描述的搜索路径（如果在您使用的命令上指定了 DEVELOPER1 的话）。 PAYROLL DEVELOPER1 PAYROLL TEST PAYROLL MASTER

要将所有组包括在项目层次结构的缺省搜索路径中而不列示每个组，使用 *DFT 值是一个简单的方法。对于 *DFT 值，会作如下解析:

1. 会扫描搜索路径部件内列出的搜索路径以查找特殊值 *DFT。
2. 对于找到的每个 *DFT，包含 *DFT 的行会替换为以下形式的项目和组对的名称:
   a. 从 *DFT 行开始，然后检查搜索路径部件中列出的前几行，会检查每一行以查找与 *DFT 相关联的项目的名称。
   b. 如果找到了相同项目名，则该项目和组对的缺省搜索路径会用来解析值 *DFT。
   c. 如果**未**找到相同项目名，会检查正在运行的命令以查看项目名是否匹配。

d. 如果在正在运行命令上指定了相同项目名，则当前与该项目相关联的组会用作要替换值 *DFT 的缺省搜索路径的起始点。如果找不到任何匹配的项目名，会发出错误消息。

3. 一旦解析了搜索路径部件中列出的所有 *DFT 值，搜索路径就会以在该命令上指定的项目 / 组对起始，后跟所有项目 / 组对，就象搜索路径部件中列示的一样。如果在该命令上指定的项目 / 组对不在解析列表中，则会发出警告。

示例

根据第101页的图37，假定已经创建了以下搜索路径，且已在 QRYPART 命令上指定了组 TEST。

```
PAYROLL DEVELOPER1
PAYROLL *DFT
EMPDATA TESTDATA
EMPDATA *DFT
```

依据上面列示的规则:

1. 在搜索路径部件中找到值 *DFT。
2. 第一个 *DFT 值只会替换为 PAYROLL 项目中的组 MASTER。

```
PAYROLL MASTER
```

3. 第二个 *DFT 值会替换为以 EMPDATA 项目中的组 TESTDATA 开头的缺省搜索路径。

```
EMPDATA TESTDATA
EMPDATA MASTER
```

结果是 QRYPART 命令会在以下搜索路径中搜索部件。

```
PAYROLL TEST
PAYROLL MASTER
EMPDATA TESTDATA
EMPDATA MASTER
```

第10章 处理用户定义类型

本章描述:

- 用户定义类型的概念及谁会使用它们
- 用来使用用户定义类型的命令 (ADDADMTYPE、RMVADMTYPE、PRTADMTYPE)
- 用来使用用户定义语言的命令 (ADDADMLANG、RMVADMLANG、CHGADMLANG、PRTADMLANG)
- 与用户定义类型相关联的操作 (CHGADMACN)

了解用户定义类型

可定义您自己的部件类型，它可使此功能部件对您更有用。它们称为用户定义类型。**用户定义类型**是您定义的，以便“应用程序开发管理器”功能部件可识别的部件类型。尽管“应用程序开发管理器”功能部件支持最常用的对象类型、成员类型和编译器，但仍有一些应用程序可能会包含不受支持的对象类型。您会发现，在要管理部件的多个版本，或是想要进行下列操作时，此功能将非常有用:

- 向当前不受支持的编译器语言添加用户定义源成员，如 PL/1 或 FORTRAN
- 向当前不受支持的对象类型添加用户定义对象类型
- 添加将不在“应用程序开发管理器”项目的对象或源文件成员中存储的用户定义类型，以允许需要管理其数据库模型的版本的 CASE 工具与“应用程序开发管理器”功能部件交互作用

添加用户定义源成员类型 (ADDADMTYPE)

使用“添加 ADM 类型”(ADDADMTYPE) 命令来添加源成员类型。为此，在 SYSTYPE 参数上指定 *MBR。如果使用的系统提供的或用户定义的编译器不受“应用程序开发管理器”功能部件支持的话，您可能想要添加源成员部件类型。您可通过如下方式来达到此目的:

- 要能够使用 CRTPART 和 BLDPART 命令创建和构建用户定义类型的部件，应使用“添加 ADM 语言”(ADDADMLANG) 命令将语言添加至用户定义类型，并定义 BLDPART 命令用来编译用户定义类型和语言的部件的命令。如果想要编译用户定义类型的部件，编译器必须写至 *System API Programming* 中所述的“应用程序开发管理器 API”。
- 如果编译器未写至“应用程序开发管理器 API”，您只能维护源的各个版本。您将无法构建用户定义类型的部件。

  但是，即使您仍然为用户定义类型定义类型和语言，您所拥有的编译器还是不能写至“应用程序开发管理器 API”。例如，可添加用户定义类型 ZPASSRC 并对该类型定义语言 PASCAL。

您可能需要添加用户定义类型的另一个原因是提供预处理。例如，您可能需要添加用户定义类型 RPT，以便能够将“自动报告程序”源存储在“应用程序开发管理器”项目中。在对此类型定义语言时，必须对“添加 ASDM 语言”(ADDADMLANG) 命令上的 BLDCMD 参数指定“创建自动报告程序”(CRTRPTPGM) 命令。因为 CRTRPTPGM 命

令会调用 CRTRPGPGM（它是针对“应用程序开发管理器”功能部件启用的），所以可使用 BLDPART 来编译 RPT 部件。有关详情，请参考第118页的『添加用户定义类型 RPT－示例』。

例如，如果您有 RPG/38 编译器，您可能想要添加 System/38 用户定义源成员类型。然后，可使用 CRTPART 命令创建用户定义类型的部件。不能在 ADDADMLANG 命令的 BLDCMD 参数上定义构建命令。不能对“应用程序开发管理器”功能部件启用 RPG/38 编译器。可维护源部件的不同版本，但要编译用户定义类型的部件，您必须执行下列其中一项任务:

- 使用 ADDPRJLIBL 命令将这些部件所驻留的组添加至库列表，然后调用指定想要存储输出 *PGM 对象的库的编译器。
- 在 EXPPART 命令的 SCAN 参数上指定 *YES，以将要编译的部件导出至库，然后调用指定要将这些部件导出所至库的名称的编译器。

添加用户定义对象类型

使用“添加 ADM 类型”(ADDADMTYPE) 命令以添加用户定义对象类型；例如，对象类型为 *CSI 的部件类型 ZCSI。在 SYSTYPE 参数上指定名称以添加用户定义对象类型。对于未受限于特定库（例如 QSYS 或 #LIBRARY）且不是 *FILE 的任何 OS/400 对象类型，都可以对其定义用户定义类型。

将 CHGADMACN 命令与 ADDADMTYPE 命令配合使用以定义一些操作，“应用程序开发管理器”功能部件必须用这些操作来使用用户定义类型的部件。

添加未存储在对象或源文件成员中的用户定义类型

使用“添加 ADM 类型”(ADDADMTYPE) 命令在“应用程序开发管理器”项目中添加未存储在 OS/400 对象或源文件成员中的用户定义类型。此功能部件会跟踪每个部件，但实体本身却存储在此功能部件的控制范围之外。

在 ADDADMTYPE 命令的 SYSTYPE 参数上指定 *NONE，以指示用户定义类型的部件不会存储在“应用程序开发管理器”项目的 OS/400 对象或源文件成员中。

如果在使用代码生成工具，您可能会想要添加这种用户定义类型。可创建用来存储源代码的用户定义类型。编译源代码时，可能会创建一个或多个成员。这些成员将对应系统提供的或用户定义的部件类型。

添加用户定义类型－基本步骤

要添加用户定义类型，需要遵循三个步骤:

1. 使用 ADDADMTYPE 命令添加用户定义类型。
2. 使用 ADDADMLANG 命令定义一种语言，以便能创建和构建具有用户定义类型的部件。
3. 定义“应用程序开发管理器”功能部件需要用来处理具有用户定义类型的部件的操作。为此，使用 CHGADMACN 命令。有时可能不需要使用 CHGADMACN 命令。如果类型储在已有部件类型（例如，成员和数据区）的对象中，则 ADDADMTYPE 命令会预填入操作。

注

在定义用户定义部件类型时，必须确保定义的操作是对对应指定部件的对象、成员或实体执行的。如果未能成功，则可能会导致不可预知的结果。

替换变量的正确使用将确保对相应的对象、成员或实体执行操作。查看附录D. 替换变量以获取各种替换变量的描述。

以下列表显示可用来定义和使用用户定义类型的所有命令:

ADDADMLANG	将语言添加至用户定义部件类型
ADDADMTYPE	添加用户定义类型
CHGADMACN	更改用户定义类型的操作定义
CHGADMLANG	更改用户定义部件语言
PRTADMLANG	打印用户定义部件语言信息
PRTADMTYPE	打印用户定义类型信息
RMVADMLANG	除去用户定义部件语言
RMVADMTYPE	除去用户定义类型

所有这些命令（PRTADMTYPE 和 PRTADMLANG 除外）都记录在项目 *ALL 的项目记录中。PRTPRJLOG 命令会更改为打印与请求的项目相匹配或项目值为 *ALL 的所有记录。

注: 所有这些命令（PRTADMTYPE 和 PRTADMLANG 除外）都需要 *ALLOBJ 权限。在“创建用户简要表”(CRTUSRPRF) 命令的 SPCAUT 参数上指定此权限。

使用用户定义类型

下列命令允许您使用用户定义类型:

ADDADMTYPE	添加用户定义类型
PRTADMTYPE	打印用户定义类型信息
RMVADMTYPE	除去用户定义类型

添加用户定义类型 (ADDADMTYPE)

使用“添加 ADM 类型”(ADDADMTYPE) 命令以将用户定义类型添加至“应用程序开发管理器”功能部件。添加之后，所有项目中的所有开发者都可使用该用户定义类型。

必须具有 *ALLOBJ 权限才能使用此命令。此值是在“创建用户简要表”(CRTUSRPRF) 命令的 SPCAUT 参数上指定的。

添加用户定义类型时，不会对该类型指定任何语言。必须使用 ADDADMLANG 命令来定义语言。查看第110页的『向用户定义类型添加语言 (ADDADMLANG)』以获取有关此命令的信息。

必需参数

指定用户定义类型及要在系统上存储用户定义类型的部件的方式。建议用户定义类型以字母 Z 开头。这会确保用户定义类型不会与任何将来系统提供的部件类型重名。如果将来系统提供的类型与用户定义类型重名，系统提供的类型将具有优先权。查看附录 C. 命名规则以了解部件类型的命名规则。

要存储用户定义类型的部件的对象类型可以是 OS/400 对象名、*MBR，或者，可指定用户定义类型的部件不存储在“应用程序开发管理器”项目的 OS/400 对象或源文件成员中，或是该对象类型可以是 OS/400 源文件成员。

如果在 SYSTYPE 参数上指定名称，用户定义类型的部件就会存储在 OS/400 对象类型中。允许不受限于特定库的所有 OS/400 对象类型。例如，不允许必须存在于库 QSYS 或 #LIBRARY 中的对象。而且，类型为 *FILE 的对象也不被允许。

如果在 SYSTYPE 参数上选择值 *NONE，用户定义类型的部件就不会存储在“应用程序开发管理器”项目内的 OS/400 对象或源文件成员中。此功能部件会跟踪每个部件，但实体本身却存储在此功能部件外部。用户定义类型的部件是由您使用 CHGADMACN 命令定义的操作来处理的。

如果选择值 *MBR，用户定义类型的部件就会存储在 OS/400 源文件成员中。

可选参数

可指定下列可选参数:

- 存储用户定义类型的部件的缺省源物理文件。DFTSRCF(QTXTSRC) 是作为源成员的用户定义类型的缺省值。可指定缺省源文件的名称。
- 用来存储用户定义类型的部件的任何源物理文件的缺省记录长度。DFTRCDLEN(92) 是缺省值。可对源物理文件指定缺省记录长度（13 至 32678 字节）。
- 其他部件是否能够包括该部件类型。缺省值为 ALWINC(*NO)；其他部件不能包括用户定义类型的部件。
- 该部件类型是否能包括其他部件。缺省值为 INCLUDES(*NONE)；用户定义类型的部件不能包括其他部件。如果指定了名称，则必须使用 ALWINC(*YES) 来定义用户定义类型，或该用户定义类型必须是作为包含的系统提供的类型。

BLDPART 命令会使用您对 ALWINC 和 INCLUDES 参数指定的值。

除去用户定义类型 (RMVADMTYPE)

使用“除去 ADM 类型”(RMVADMTYPE) 命令除去用户定义类型。仅当任何项目中都不存在用户定义类型的部件，且没有其他任何用户定义类型在 ADDADMLANG 命令的 BLDOUTTYPE 参数上定义有该特定类型时，这才会成功。

必须具有 *ALLOBJ 权限才能使用此命令。此值是在“创建用户简要表”(CRTUSRPRF) 命令的 SPCAUT 参数上指定的。

只要在 RMVADMTYPE 命令上输入要除去的类型即可。

如果想要获取具有想要除去的类型的部件的列表，在 WRKPRJPDM 命令上指定 *ALL，就会出现“用 PDM 来使用项目”屏幕。它会列示您有权使用的所有项目。对于列出的每个项目，使用 QRYPART 命令并在 TYPE 参数中输入用户定义类型。这会列示具有该用户定义类型的所有部件。

打印用户定义类型信息 (PRTADMTYPE)

使用 “打印 ADM 类型” (PRTADMTYPE) 命令以运行报告，此报告列出对 “应用程序开发管理器” 功能部件定义的任何或所有用户定义类型。此报告反映使用 ADDADMTYPE 命令添加用户定义类型时选择的所有参数值，对该类型定义的操作除外。使用 CHGADMACN 命令来定义这些值。查看图38以获取样本报告。

必需参数

指定想要打印其信息的部件类型。TYPE(*ALL) 会打印所有用户定义类型的信息。可指定特定部件类型的名称。

可选参数

使用 OUTPUT 参数指示应输出所至的位置。缺省值为 OUTPUT(*PRINT)。报告会假脱机至此作业的打印设备。OUTPUT(*OUTFILE) 会将输出引导至输出文件。

输出文件的记录格式与库 QADM 中的系统提供的数据库文件 QALYPTYP 中使用的记录格式相同。

示例: 可通过指定类似如下的命令将输出引导至输出文件。

```
PRTADMTYPE TYPE(ZPASSRC) OUTPUT(*OUTFILE) OUTFILE(*LIBL/FILE1)   OUTMBR(*FIRST
*ADD)
```

库列表用来确定要存储输出文件 FILE1 的位置。FILE1 中的第一个成员接收输出，并将输出数据添加至现存记录。

示例: 以下命令打印有关类型 PASSRC 的用户定义类型信息。

```
PRTADMTYPE TYPE(ZPASSRC) OUTPUT(*PRINT)
```

报告被假脱机至此作业的打印设备。图38显示样本报告。

```
5722WDS   V5R1M0        应用程序开发管理器      - 打印 ADM 类型
05/08/01     15:30:21   页面 . . :  0001

类型 . . . . . . . . . . . . . :       ZPASSRC      1
系统类型 . . . . . . . . . . . :       *MBR         2
缺省源文件 . . . . . . . . . . :       QTXTSRC      3
缺省记录长度 . . . . . . . . . :       00092        4
是否允许包括类型 . . . . . . . :       否           5
包括部件类型 . . . . . . . . . :       *NONE        6
*CRT 命令. . . . . . . . . . . :       QSYS/ADDPFM FILE(&L/&F) MBR(&ZN)      7

*DLT 命令. . . . . . . . . . . :       QSYS/RMVM FILE(&L/&F) MBR(&ZN)        8

*MOV 命令. . . . . . . . . . . :       *NONE        9

*CPY 命令. . . . . . . . . . . :       QSYS/CPYF FROMFILE(&L/&F) TOFILE(&ZI/&ZF) FROMMBR(&ZN) TOMBR(&ZJ) CRTFILE(*YES) MBROPT(*RE
                                       PLACE) FMTOPT(*MAP)                    10
*CHG 命令. . . . . . . . . . . :       QSYS/STRSEU SRCFILE(&L/&F) SRCMBR(&ZN) OPTION(2)       11

*DSP 命令. . . . . . . . . . . :       QSYS/STRSEU SRCFILE(&L/&F) SRCMBR(&ZN) OPTION(5)       12

*PRT 命令. . . . . . . . . . . :       QSYS/STRSEU SRCFILE(&L/&F) SRCMBR(&ZN) OPTION(6)       13
```

图 38. 打印 ADM 类型报告样本

以下列表描述 “打印 ADM 类型” 报告中的信息:

1 在 ADDADMTYPE 命令的 TYPE 参数上指定的用户定义类型。

2 在 ADDADMTYPE 命令的 SYSTYPE 参数上指定的系统类型。

3 在 ADDADMTYPE 命令的 DFTSRCF 参数上指定的缺省源文件。

4 在 ADDADMTYPE 命令的 DFTRCDLEN 参数上指定的缺省记录长度。

5 允许具有用户定义部件类型的其他部件包括该用户定义类型的部件。这是在 ADDADMTYPE 命令的 ALWINC 参数上指定的值。

6 允许用户定义类型的部件包括其他部件。这是在 ADDADMTYPE 命令的 INCLUDES 参数上指定的值。

7 在 CHGADMACN 命令的 CMD 参数上指定的值或创建命令。

8 在 CHGADMACN 命令的 CMD 参数上指定的值或删除命令。

9 在 CHGADMACN 命令的 CMD 参数上指定的值或移动命令。

10 在 CHGADMACN 命令的 CMD 参数上指定的值或复制命令。

11 在 CHGADMACN 命令的 CMD 参数上指定的值或更改命令。

12 在 CHGADMACN 命令的 CMD 参数上指定的值或显示命令。

13 在 CHGADMACN 命令的 CMD 参数上指定的值或打印命令。

使用用户定义语言

下列命令允许您使用用户定义语言:

ADDADMLANG	添加用户定义语言
CHGADMLANG	更改用户定义语言
PRTADMLANG	打印语言信息
RMVADMLANG	除去用户定义语言

向用户定义类型添加语言 (ADDADMLANG)

使用“添加 ADM 语言”(ADDADMLANG) 命令来将用户定义语言添加至用户定义类型。为了能够使用 BLDPART 命令创建和构建用户定义类型的部件，使用“添加 ADM 语言”(ADDADMLANG) 命令来添加语言并定义 BLDPART 命令将用来编译用户定义类型和语言的部件的命令。

在添加之后，所有项目中的所有开发者都可以使用该用户定义语言。只有表示非包含的成员或没有对象或成员的用户定义类型（即，在 ADDADMTYPE 命令的 SYSTYPE 参数上指定了 *NONE）才可以在 BLDCMD 参数上定义命令。

您必须具有 *ALLOBJ 权限才能使用此命令。此值是在“创建用户简要表”(CRTUSRPRF) 命令的 SPCAUT 参数上指定的。

必需参数

在 TYPE 参数上指定用户定义类型。在 LANG 参数上指定语言或 *NONE。

如果正在添加的语言是用于与带有属性的 OS/400 对象相对应的用户定义类型，建议您将语言名设置为与属性名相同。并且，如果正在添加的语言是用于与源成员相对应的用户定义类型，请指定与成员类型相同的语言。这能确保 IMPPART 命令在导入对象集或成员集时能够正确地导入对象和源。例如，语言为 PLI 的用户定义类型 ZPGM 与属性为 PLI 的对象类型 *PGM 相对应。

可选参数

可以指定下列可选参数:

- 构建过程应使用以编译具有您指定的类型与语言的组合的部件的命令。

  缺省值是 BLDCMD(*NONE)。这表示不能使用 BLDPART 命令来构建部件类型和语言。

  在下列情况中，必须在 BLDCMD 参数上指定 *NONE:

  – 如果正在添加的语言是 *NONE

  – 如果 TYPE 参数上指定的用户定义类型存储在对象中（即，如果在 ADDADMTYPE 命令的 SYSTYPE 参数上指定了 OS/400 对象类型的名称）

  – 如果用户定义类型是一个包含（是使用 ADDADMTYPE 命令上的 ALWINC(*YES) 定义的）

  如果用户定义类型是 *MBR 或 SYSTYPE *NONE，则可以指定要用来构建用户定义类型和语言的部件的命令字符串。命令字符串最长可达 250 个字符。不应在命令字符串中输入注释。不能提示 BLDCMD 参数。如果您定义的编译命令未替换构建过程的现存输出部件，则 BLDPART 命令在尝试构建 BLDPART 命令上指定的用户定义类型的部件时，它会失败。

  可以在命令字符串中使用替换变量。在发出命令之前，它们会被替换为实际的值。对 CHGADMACN 命令的 CMD 参数有效的所有替换变量也都对此参数有效。第227页的『附录D. 替换变量』描述了所有替换变量，并指示 BLDCMD 参数上允许哪些替换变量。

  在某些情况下，“应用程序开发管理器”功能部件不能确定用于替换替换变量的信息。例如，如果正在定义的操作为删除操作，则 &ZH 替换变量不能被替换为正在移动或复制的部件的名称。此时，使用替换变量 &ZH 没有任何意义。如果正在使用 BLDCMD 参数，则 &ZH 替换变量不能被替换为扫描值。在这种情况下，如果没有必需的信息可用，则不会解析此变量。

示例: 此示例假定您使用的是名为 ZRPG 的用户定义 RPG 编译器，且该编译器写至 *System API Programming* 中描述的“应用程序开发管理器 API”。

以下命令显示了 ADDADMLANG 命令的 BLDCMD 参数上定义的 CRTZRPGPGM 编译命令的示例。

```
0ADDADMLANG TYPE(ZRPGSRC) LANG(ZRPG) BLDCMD('CRTZRPGPGM PGM(&L/&ZN)
SRCFILE(&L/&F) SRCMBR(&ZN)') BLDOUTTYPE(PGM)              BLDOUTLANG(*INPUT)
SPLFNAME(*INPUT)
```

- **构建过程生成的输出对象的部件类型**

  如果类型是一个包含、存储在 OS/400 对象中的类型或未在 TYPE 参数上指定相关联对象或成员的用户定义类型，则此参数被忽略。如果在 LANG 和 BLDCMD 参数上指定了 *NONE，则不允许此参数。缺省值是 BLDOUTTYPE(*NONE)。这表示不会创建任何输出部件。

  在构建指定类型和语言的部件时，可以对构建过程生成的 *PGM 输出对象指定名称。

- **构建过程生成的输出的语言**

  如果类型是一个包含、存储在 OS/400 对象中的类型或未在 TYPE 参数上指定相关联对象或成员的用户定义类型，则此参数被忽略。如果在 LANG 和 BLDCMD 参数上指定了 *NONE，则不允许此参数。

缺省值是 BLDOUTLANG(*INPUT)；构建过程生成的输出部件的语言将与创建它的部件的语言相同。如果输出部件没有任何语言，则指定 BLDOUTLANG(*NONE)，或指定一名称来标识要指定给输出部件的语言。

- **构建过程保存的编译器列表的假脱机文件名称**

  如果编译器生成了编译器列表，且在 BLDPART 命令上指定了 SAVLST 参数。

  如果 TYPE 参数上的用户定义类型未存储在成员中，如果 LANG 参数上指定的语言是 *NONE，或者，如果用户定义类型是一个包含，则此参数被忽略。

  缺省值是 SPLFNAME(*INPUT)；编译器列表的假脱机文件与正在构建的部件同名。可指定编译器列表的假脱机文件将与构建过程生成的输出部件同名，即指定 SPLFNAME(*OUTPUT)；您可以指定在 BLDCMD 参数上定义的编译命令不会创建列表，即指定 SPLFNAME(*NONE)；或者，可以指定编译器列表的假脱机文件将具有的名称。（构建过程不检查或保存编译器列表。）

更改用户定义语言 (CHGADMLANG)

使用“更改 ADM 语言”(CHGADMLANG) 命令来更改 BLDPART 命令处理具有给定用户定义类型与语言的组合的部件的方式。只有表示非包含的成员或没有对象或成员的用户定义类型（即，在 ADDADMTYPE 命令的 SYSTYPE 参数上指定了 *NONE）才可以在 BLDCMD 参数上定义命令。

当使用 CHGADMLANG 命令来更改现存用户定义语言时，使用旧语言定义构建的任何部件都会失效。构建过程识别到语言已更改，并会在下次对该类型的部件发出 BLDPART 命令时再次构建该部件。

您必须具有 *ALLOBJ 权限才能使用此命令。此值是在“创建用户简要表”(CRTUSRPRF) 命令的 SPCAUT 参数上指定的。

必需参数

在 TYPE 参数上指定用户定义类型。对 LANG 参数上的语言指定用户定义语言的名称或 *NONE。

可选参数

可以指定下列可选参数:

- 构建过程应使用以编译具有您指定的类型与语言的组合的部件的命令

  如果 TYPE 参数上指定的类型的类型与语言的组合表示存储在 OS/400 对象中的类型，则此参数被忽略。

  在下列情况下，必须指定 *NONE:
  - 如果正在添加的语言是 *NONE
  - 如果在 TYPE 参数上指定的用户定义类型存储在对象中（而不是使用 ADDADMTYPE 命令上的 SYSTYPE(*NONE) 或 SYSTYPE(*MBR) 定义的）
  - 如果用户定义类型是一个包含（是使用 ADDADMTYPE 命令上的 ALWINC(*YES) 定义的）

  缺省值是 BLDCMD(*SAME)。使用当前的构建命令。当提示 CHGADMLANG 命令时，*SAME 被替换为构建命令。

  如果指定 BLDCMD(*NONE)，则不能构建该类型与语言的组合。

  可指定用来构建用户定义类型和语言的部件的命令字符串。命令字符串最长可达 250 个字符。不应在命令字符串中输入注释。

可以在命令字符串中使用替换变量。在发出命令之前，它们会被替换为实际的值。不能提示 BLDCMD 参数。第227页的『附录D. 替换变量』描述了所有替换变量，并指示 BLDCMD 参数上允许哪些替换变量。

- 构建过程生成的输出的类型

  缺省值是 BLDOUTTYPE(*SAME)。使用输出对象 (*PGM) 的部件类型。当提示 CHGADMLANG 命令时，*SAME 被替换为输出类型的名称。

  在构建 TYPE 和 LANG 参数上指定的类型和语言的部件时，可以对构建过程生成的 *PGM 输出对象指定部件类型的名称。

  如果指定 *NONE，则构建过程不会创建任何输出部件。

- 构建过程生成的输出的语言

  缺省值是 BLDOUTLANG(*SAME)。使用输出部件类型的当前语言。当提示 CHGADMLANG 命令时，*SAME 会被替换为实际的语言。

  可以为构建输出部件指定一个名称。

  如果指定 BLDOUTLANG(*INPUT)，则构建过程生成的输出部件与创建它的部件具有相同的语言。

  如果指定 BLDOUTLANG(*NONE)，则构建过程生成的输出部件没有任何语言。

- 构建过程用于它保存的编译器列表的假脱机文件的名称

  如果编译器生成编译器列表，且在 BLDPART 命令上指定了 SAVLST 参数，则此名称适用。

  缺省值是 SPLFNAME(*SAME)。使用编译器列表的假脱机文件的当前名称。当提示 CHGADMLANG 命令时，*SAME 会被替换为列表的名称。

  您可以为编译器列表的假脱机文件指定一个名称。

  如果指定 SPLFNAME(*INPUT) 或 SPLFNAME(*OUTPUT)，则编译器列表的假脱机文件分别与输出或输出部件同名。

  如果指定 SPLFNAME(*NONE)，则 BLDCMD 参数上定义的编译命令不会创建任何假脱机文件编译器列表。如果指定 *NONE，则构建过程将不处理或检查编译器列表。

除去用户定义语言 (RMVADMLANG)

使用“除去 ADM 语言”(RMVADMLANG) 命令来除去用户定义语言。仅当任何项目中都不存在用户定义类型与语言组合的部件，且没有其他用户定义类型具有 ADDADMLANG 命令的 BLDOUTLANG 参数上定义的特定语言时，此命令才会成功。

您必须具有 *ALLOBJ 权限才能使用此命令。此值是在“创建用户简要表”(CRTUSRPRF) 命令的 SPCAUT 参数上指定的。

只要输入要除去的用户定义类型和语言即可。

如果指定 *NONE，这表示会除去语言 *NONE。（语言 *NONE 指示不能构建该部件。）

如果您指定语言的名称，且您正在除去对该部件类型定义的唯一语言，则语言参数会复位为 *NONE。

打印用户定义语言信息 (PRTADMLANG)

使用“打印 ADM 语言”(PRTADMLANG) 命令来运行一个报告，此报告列示对用户定义类型定义的任何或所有用户定义语言。此报告显示与特定类型与语言的组合相关联

的类型、语言和构建命令。构建命令在 ADDADMLANG 命令上指定。它还会显示构建过程生成的部件的类型和语言以及假脱机文件的名称。

必需参数

指定要打印其语言信息的用户定义类型。

指定要从 TYPE 参数上指定的用户定义类型中除去的用户定义语言。您可以指定特定部件语言的名称；可以指定 LANG(*ALL) 以打印关于所有用户定义语言的信息；也可以指定 LANG(*NONE) 以打印关于用户定义语言 *NONE 的信息。

可选参数

使用 OUTPUT 参数指示应输出所至的位置。缺省值是 OUTPUT(*PRINT)。报告假脱机至此作业的打印设备。OUTPUT(*OUTFILE) 会将输出引导至输出文件。

输出文件的数据库格式与在库 QADM 中的系统提供的数据库文件 QALYPLNG 中使用的数据库格式相同。

示例: 通过指定类似如下的命令，可将输出引导至输出文件。

```
PRTADMLANG TYPE(ZPASSRC) LANG(*ALL) OUTPUT(*OUTFILE) OUTFILE(*LIBL/FILE1)
           OUTMBR(*FIRST *ADD) SCAN(*YES)
```

库列表用来确定在何处存储名为 FILE1 的输出文件。FILE1 中的第一个成员接收到输出，并将输出数据添加至现存记录。

示例: 以下命令打印对部件类型 ZPASSRC 定义的所有语言。

```
PRTADMLANG TYPE(ZPASSRC) LANG(PAS) OUTPUT(PGM)
```

与图39中的报告类似的报告将假脱机至此作业的打印设备。

```
5722WDS   V5R1M0        应用程序开发管理器                - 打印 ADM 语言          05/08/01      9:49:55   页面 . . :  0001

Type . . . . . . . . . . . . . . :       ZPASSRC     1

                                                                                  输出          输出          假脱机文件
语言          构建命令                                                            类型          语言          名称
PAS           CRTPASPGM PGM(&L/&ZN) SRCFILE(&L/&F) SRCMBR(&ZN)                    PGM           *NONE         *INPUT

2             3                                                                   4   5   6

                    * * * * *  列      表      结      束  * * * * *
```

图 39. 打印 ADM 语言报告的样本

以下列表描述“打印 ADM 语言”报告中的信息:

1. PRTADMLANG 命令上指定的用户定义类型，或正在打印的类型之一（如果指定了 *ALL 的话）
2. 对 ADDADMLANG 命令上的用户定义类型定义的语言
3. 对 ADDADMLANG 命令上的用户定义类型与语言的组合定义的构建命令
4. 对 PRTADMLANG 命令上的用户定义类型与语言的组合定义的输出类型
5. 对 PRTADMLANG 命令上的用户定义类型与语言的组合定义的输出语言
6. 对 ADDADMLANG 命令上的用户定义类型与语言的组合定义的假脱机文件名

定义要配合用户定义类型使用的操作 (CHGADMACN)

在将用户定义类型添加至“应用程序开发管理器”功能部件之后，可定义此功能部件必须用来使用该用户定义类型的部件的操作。

在使用 ADDADMTYPE 命令指定用户定义类型时，对于其 SYSTYPE 参数值为 *NONE 或系统提供的类型没有使用的值的那些部件类型，所有操作都被初始化为 *NONE。在定义部件类型时，如果其 ADDADMTYPE 命令上的 SYSTYPE 参数值已被系统提供的类型使用，则操作值的初始化方式与系统提供的类型相似。

“更改 ADM 操作”(CHGADMACN) 命令为用户定义类型指定操作。您必须具有 *ALLOBJ 权限才能使用此命令。此值是在“创建用户简要表”(CRTUSRPRF) 命令的 SPCAUT 参数上指定的。

第117页的『添加用户定义类型 ZPASSRC－示例』提供了 CHGADMACN 命令的示例。表10显示了要配合用户定义类型使用的操作，以及执行这些操作的相关“应用程序开发管理器”命令。

表 10. 操作及其相关命令

ACTION 参数上的值	操作	应用程序开发管理器命令
*CRT	创建	CRTPART
*DLT	删除	DLTPART、EXPPART（当指定 REPLACE(*YES) 时）、PRMPART（当在 ADDADMTYPE 命令上指定 SYSTYPE(*MBR) 时）和 RNMPART（用于任何类型）。
*MOV	通过项目层次结构移动部件	PRMPART
注: 您应该在 *MOV 操作的名称前面添加包含该命令的库的名称。这可以确保不使用被采用权限发出该命令的另一副本。应确保对移动操作定义的命令没有向别人提供输入命令的机会。		
*CPY	复制	CHKOUTPART、CPYPART、EXPPART、IMPPART、PRMPART（如果在 ADDADMTYPE 命令上指定了 SYSTYPE(*MBR)）和 RNMPART（用于任何类型）。
*CHG	更改	CHGPART
*DSP	显示	DSPPART
*PRT	打印	DSPPART

必需参数

在 TYPE 参数上指定用户定义类型，并在 ACTION 参数上对其指定操作。参见表10，以获取您可以在 ACTION 参数上选择的值以及与每个值相关联的“应用程序开发管理器”命令。

当在 CRTPART 命令的 PRMPT 上指定 *YES 时，会提示已定义的 *CRT 操作。如果在 ADDADMTYPE 的 SYSTYPE 参数上指定作为对象的用户定义类型的名称，则会提示已定义的 *CHG 操作。

当命令提示七种操作的任何一种时，对任何关键字参数的输入都是不允许的。在命令字符串之前插入问号 (?)。在命令字符串中的每个关键字参数前面插入问号和星号 (?*)。

不推荐您定义带有提示字符（？ 和 ?*）的操作。

可选参数

将 CMD 参数与 ACTION 参数配合使用。在 CMD 参数上，指定想要让 “应用程序开发管理器” 功能部件用来执行在 ACTION 参数上指定的操作的命令。不能提示 CMD 参数。

缺省值是 CMD(*SAME)。用于指定的操作的命令不会更改。使用 “应用程序开发管理器” 命令。当提示 CHGADMACN 命令时，*SAME 会被替换为命令（参见表10）。如果指定 CMD(*NONE)，这指示在 ACTION 参数上指定的操作不能对用户定义类型的部件执行。

可指定用于执行在 ACTION 参数上指定的操作的命令字符串。可在此处使用替换变量，在发出命令之前，替换变量会被替换为值。

有时，“应用程序开发管理器” 功能部件不能确定用来替换替换变量的信息。例如，如果正在定义的操作为删除操作，则 &ZJ 替换变量不能被替换为正在移动或复制的部件的名称。在这种情况下，使用替换变量 &ZJ 是没有意义的。如果没有必需的信息可用，则不解析替换变量。附录D. 替换变量描述了所有替换变量，并指示 CMD 参数上允许哪些替换变量。

如果指定任何 “应用程序开发管理器” 命令的 TEXT 参数，则在任何用户定义类型操作上指定的 TEXT 参数均被忽略。例如，可在 *CRT 操作上指定 TEXT 参数，但在创建此类型的部件时，将使用 CRTPART 命令上的 TEXT 参数。

如果用户定义类型操作找不到传送给它处理的部件，则应该会返回以下消息。

ADM9004 用户定义类型操作找不到部件。

“应用程序开发管理器” 功能部件会除去与正在处理的部件相关联的 “部件目录条目” 来对此消息作出响应。应当只会对使用 SYSTYPE(*NONE) 定义的用户定义类型操作返回此消息。

添加 RJE 对象类型 *CSI – 示例

下列步骤显示如何添加用户定义类型 RJE:

1. 使用 ADDADMTYPE 命令来添加用户定义类型。

   要将 RJE 对象类型 *CSI 定义为用户定义类型，指定以下内容。

   ```
   ADDADMTYPE TYPE(ZCSI) SYSTYPE(*CSI)
   ```

2. 使用 ADDADMLANG 命令来为用户定义类型指定语言，以便可创建和构建该部件。可以在命令字符串中使用替换变量。第227页的『附录D. 替换变量』描述了所有替换变量。

   要将语言添加至用户定义类型 CSI，指定以下内容。

   ```
   ADDADMLANG TYPE(ZCSI) LANG(*NONE)
   ```

   不能构建类型为 CSI 的部件，因此，指定 LANG(*NONE)。

3. 使用 CHGADMACN 命令来定义“应用程序开发管理器”功能部件需要用来使用具有用户定义类型的部件的操作。可以在命令字符串中使用替换变量。第227页的『附录D. 替换变量』描述了所有替换变量。

   要为 CSI 部件类型定义创建操作，指定以下内容。

```
CHGADMACN TYPE(ZCSI) ACTION(*CRT) CMD('QSYS/CRTCSI CSI(&L/&ZN)')
```

   要为 CSI 部件类型定义删除操作，指定以下内容。

```
CHGADMACN TYPE(ZCSI) ACTION(*DLT) CMD('QSYS/DLTCSI CSI(&L/&ZN)')
```

   要为 CSI 部件类型定义移动操作，指定以下内容。

```
CHGADMACN TYPE(ZCSI) ACTION(*MOV) CMD('QSYS/MOVOBJ OBJ(&L/&ZN)
          OBJTYPE(*CSI) TOLIB(&ZI)')
```

   要为 CSI 部件类型定义复制操作，指定以下内容。

```
CHGADMACN TYPE(ZCSI) ACTION(*CPY) CMD('QSYS/CRTDUPOBJ OBJ(&ZN)
          FROMLIB(&L) OBJTYPE(*CSI) TOLIB(&ZI) NEWOBJ(&ZJ)')
```

   要为 CSI 部件类型定义更改操作，指定以下内容。

```
CHGADMACN TYPE(ZCSI) ACTION(*CHG) CMD('QSYS/CHGCSI CSI(&L/&ZN)')
```

   要为 CSI 部件类型定义显示操作，指定以下内容。

```
CHGADMACN TYPE(ZCSI) ACTION(*DSP) CMD('QSYS/DSPCSI CSI(&L/&ZN)')
```

   要为 CSI 部件类型定义打印操作，指定以下内容。

```
CHGADMACN TYPE(ZCSI) ACTION(*PRT) CMD('QSYS/DSPCSI CSI(&L/&ZN)
          OUTPUT(*PRINT)')
```

添加用户定义类型 ZPASSRC – 示例

下列步骤显示如何添加用户定义类型 ZPASSRC。此示例假定用户有已对“应用程序开发管理器”功能部件启用的 PASCAL 编译器:

1. 使用 ADDADMTYPE 命令来添加用户定义类型。

   要定义要存储在 OS/400 源文件成员中的类型 ZPASSRC，指定以下内容。

```
ADDADMTYPE TYPE(ZPASSRC) SYSTYPE(*MBR)
```

2. 使用 ADDADMLANG 命令来为用户定义类型指定语言，以便可构建该部件。

   要将语言添加至用户定义类型，指定以下内容。

```
ADDADMLANG TYPE(ZPASSRC) LANG(PAS) BLDCMD('CRTPASPGM PGM(&L/&ZN)
           SRCFILE(&L/&F) SRCMBR(&ZN)') BLDOUTTYPE(PGM)
           BLDOUTLANG(*INPUT) SPLFNAME(*INPUT)
```

3. 使用 CHGADMACN 命令来定义“应用程序开发管理器”功能部件需要用来使用用户定义类型的部件的操作。可以在 CMD 参数上定义的命令字符串中使用替换变量。第227页的『附录D. 替换变量』描述了所有替换变量，并指示 CMD 参数上允许使用哪些替换变量。

   此示例使用成员类型 ZPASSRC。CHGADMACN 命令的 CMD 参数中已预先填入了操作命令。下面的示例显示了系统提供的预先填入的操作。您可能想更改这些预先填入的操作以符合您自己的需要。

   要为 ZPASSRC 部件类型定义创建操作，指定以下内容。

```
CHGADMACN TYPE(ZPASSRC) ACTION(*CRT) CMD('QSYS/ADDPFM FILE(&L/&F)
          MBR(&ZN)')
```

要为 ZPASSRC 部件类型定义删除操作，指定以下内容。

```
CHGADMACN TYPE(ZPASSRC) ACTION(*DLT) CMD('QSYS/RMVM FILE(&L/&F)
          MBR(&ZN)')
```

要为 ZPASSRC 部件类型定义移动操作，指定以下内容。

```
CHGADMACN TYPE(ZPASSRC) ACTION(*MOV) CMD(*NONE)
```

要为 ZPASSRC 部件类型定义复制操作，指定以下内容。

```
CHGADMACN TYPE(ZPASSRC) ACTION(*CPY) CMD('QSYS/CPYF FROMFILE(&L/&F)
          TOFILE(&ZI/&ZF) FROMMBR(&ZN) TOMBR(&ZJ) CRTFILE(*YES)
          MBROPT(*REPLACE) FMTOPT(*MAP)')
```

要为 ZPASSRC 部件类型定义更改操作，指定以下内容。

```
CHGADMACN TYPE(ZPASSRC) ACTION(*CHG) CMD('QSYS/STRSEU SRCFILE(&L/&F)
          SRCMBR(&ZN) OPTION(2)')
```

要为 ZPASSRC 部件类型定义显示操作，指定以下内容。

```
CHGADMACN TYPE(ZPASSRC) ACTION(*DSP) CMD('QSYS/STRSEU SRCFILE(&L/&F)
          SRCMBR(&ZN) OPTION(5)')
```

要为 ZPASSRC 部件类型定义打印操作，指定以下内容。

```
CHGADMACN TYPE(ZPASSRC) ACTION(*PRT) CMD('QSYS/STRSEU SRCFILE(&L/&F)
          SRCMBR(&ZN) OPTION(6)')
```

添加用户定义类型 RPT – 示例

下列步骤显示如何添加用户定义类型 RPT:

1. 使用 ADDADMTYPE 命令来添加用户定义类型。

   要为 RPT 包含定义部件类型，指定以下内容。

   ```
   ADDADMTYPE TYPE(ZRPTINC) SYSTYPE(*MBR) DFTSRCF(QRPGSRC) ALWINC(*YES)
   ```

2. 使用 ADDADMLANG 命令来为用户定义类型指定语言。

   要为 RPT 包含定义语言，指定以下内容。

   ```
   ADDADMLANG TYPE(ZRPTINC) LANG(RPG)
   ```

3. 使用 ADDADMTYPE 命令来添加用户定义类型。

   要为 RPT 源定义部件类型，指定以下内容。

   ```
   ADDADMTYPE TYPE(ZRPTSRC) SYSTYPE(*MBR) DFTSRCF(QRPGSRC)
              INCLUDES(ZRPTINC)
   ```

4. 使用 ADDADMLANG 命令来为用户定义类型指定语言，以便可创建和构建该部件。可以在命令字符串中使用替换变量。附录D. 替换变量描述了所有替换变量。

   ```
   ADDADMLANG TYPE(ZRPTSRC) LANG(RPG) BLDCMD('CRTRPTPGM PGM(&L/&ZE)
              SRCFILE(&L/&F) SRCMBR(&ZN)') BLDOUTTYPE(PGM) BLDOUTLANG(RPG)
   ```

第11章 构建应用程序

本章描述:
- 了解部件关系
- BLDPART 命令调用的编译器和预处理器
- BLDPART 命令如何确定何时编译部件
- 第一次构建
- 使用构建选项
- 构建过程如何搜索部件
- 使用 BLDPART 命令
- 特定部件类型的特殊构建处理
- 构建样本应用程序

了解部件关系

应用程序是有若干个组件构成的: 可以编译的源，以及通过编译源而生成的模块、程序或文件之类的输出。其中每一个组件都是一个“应用程序开发管理器”部件。当您使用 BLDPART 命令时，会在此功能部件中确定并存储部件关系。

有两种关系: *创建*关系和*相关*关系。当编译源部件以创建诸如程序之类的输出部件时，便建立了*创建*关系。*相关*关系有许多种。当您编译源部件，而它需要另一部件才能存在时，这就是相关。文件、子程序和包含都是*相关*部件的示例。每次使用 BLDPART 命令时，关于编译部件时所使用的相关部件的信息就会与该编译部件一起作为*相关*关系存储起来。BLDPART 命令使用这些关系来确定使用构建过程时要考虑哪些部件以及哪些部件需要编译。

以前从未构建过的部件不具有关系，因此，在调用 BLDPART 命令时，会构建这些部件。第一次构建过程会根据编译器返回的信息添加它们的关系。

在 BLDPART 命令成功地处理部件之后，它会将有关编译该部件时使用的部件之间的关系的信息保存下来。当部件的构建处理成功完成后，便认为该部件已构建。构建处理活动包括编译部件（如果有必要的话）和保存关于此部件与其他部件的关系的信息。

第120页的表11显示了“应用程序开发管理器”功能部件中的部件关系。表标题具有以下含义:

关系的源
源是高级语言源或 C/400® 程序或模块。

关系 源与关系目标之间的关系是*创建*或*相关*关系。

关系名 源与目标之间的关系的名称; *创建*、*绑定*、*包含*、*子程序*、*引用*、*基于*，*引用其中的记录*或*引用其中的字段*中的一种。

关系的目标
与关系的源相关的部件。

表 11. 应用程序开发管理器中的关系

关系的源	关系	关系名	关系的目标
BNDSRC	创建	创建	SRVPGM
CBLSRC、CBLLESRC、CLLESRC、CLPSRC、CSRC、RPGLESRC、RPGSRC	创建	创建	PGM
CBLLESRC、CLLESRC、CSRC（用于 CLE 语言）、RPGLESRC	创建	创建	MODULE
语言 SQLCBLLE 的 CBLLESRC、语言 SQLCLE 的 CSRC、语言 SQLRPGLE 的 RPGLESRC	创建	创建	MODULE、PGM、SRVPGM
CLDSRC	创建	创建	CLD
CMDSRC	创建	创建	CMD
DDSSRC	创建	创建	FILE
MODULE	创建	创建 绑定	PGM
语言 MENU 的 PNLSRC	创建	创建	语言 UIM 的 MENU
语言 PNLGRP 的 PNLSRC	创建	创建	PNLGRP
BNDDIR	相关	引用	BNDDIR、MODULE、SRVPGM
CBLINC、CBLLEINC、CBLLESRC、CBLSRC、CINC、CLLESRC、CLPSRC、CSRC、DDSSRC、RPGINC、RPGLESRC、RPGLEINC、RPGSRC	相关	引用	FILE
CMDSRC、DDSSRC、PNLSRC、PNLINC	相关	引用	MSGF
MODULE (CRTPGM)	相关	引用	BNDDIR、MODULE、SRVPGM
CBLINC, CBLSRC	相关	包含	CBLINC
CBLLEINC、CBLLESRC	相关	包含	CBLLEINC
CINC、CSRC	相关	包含	CINC
PNLINC、PNLSRC	相关	包含	PNLINC
RPGINC、RPGSRC	相关	包含	RPGINC
RPGLEINC、RPGLESRC	相关	包含	RPGLEINC
语言 LF 的 DDSSRC	相关	基于	FILE
CBLINC、CBLSRC、CBLLEINC、CBLLESRC、CINC、CLLESRC、CLPSRC、CSRC、DDSSRC、RPGINC、RPGSRC、RPGLESRC、RPGLEINC	相关	引用其中的记录	FILE
DDSSRC	相关	引用其中的字段	FILE
存储在成员中的用户定义类型（在 ADDADMTYPE 命令的 SYSTYPE 参数上指定了 *MBR）	创建	创建	存储在对象中的由系统提供的或用户定义的类型
存储在成员中的用户定义类型（在 ADDADMTYPE 命令的 SYSTYPE 参数上指定了 *MBR）	创建 / 包含	创建 / 包含	存储在对象中的由系统提供的或用户定义的类型

表 11. 应用程序开发管理器中的关系 (续)

关系的源	关系	关系名	关系的目标
并未存储在项目的对象或源文件成员中的用户定义类型（在 ADDADMTYPE 命令的 SYSTYPE 参数上指定了 *NONE）	创建	创建	存储在成员或多个成员中的由系统提供的或用户定义的类型

注意:

- 不能构建 C 和 SQLC 语言的 CSRC 和 PGM。
- 不能构建 System/36 类型。

图40说明了第120页的表11中的概念。部件存在于一个只包含一个组的小型项目中；因此，对于此示例，给出项目名、组名、类型名和部件名的全限定名称不是必需的。为了简单起见，只显示了部件类型和部件名。框外面的 **C** 表示*创建*关系，**D** 表示*相关*关系。

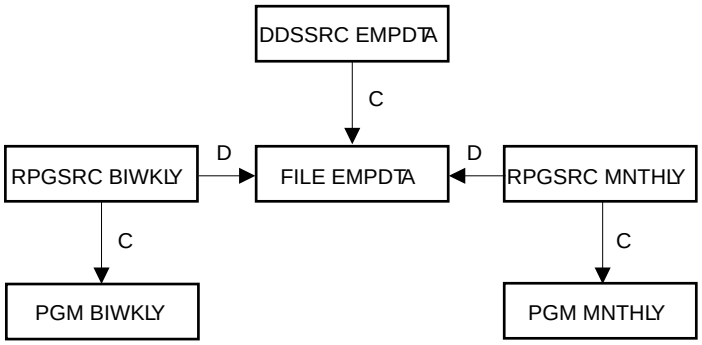

图 40. 组中的部件和它们的相关项

此项目含有下列部件:

DDSSRC EMPDTA-PF 已编译成 FILE EMPDTA 的 DDS 源物理文件。DDSSRC EMPDTA *创建* FILE EMPDTA。

FILE EMPDTA-PF 通过编译 DDSSRC EMPDTA 而创建的文件。

RPGSRC BIWKLY 已编译成 PGM BIWKLY 的源代码。RPGSRC BIWKLY 对 FILE EMPDTA 具有*相关*关系，对 PGM BIWKLY 具有*创建*关系。

PGM BIWKLY 通过编译 RPGSRC BIWKLY 而创建的程序。

RPGSRC MNTHLY 已编译成 PGM MNTHLY 的源代码。RPGSRC MNTHLY 对 FILE EMPDTA 具有*相关*关系，对 PGM MNTHLY 具有*创建*关系。

PGM MNTHLY 通过编译 RPGSRC MNTHLY 而创建的程序。

图41说明了与图40相同的概念，但使用类型为 CSRC 的部件。它显示了类型为 CSRC 且语言为 CLE 的部件如何使用“创建 C/400 模块”(CRTCMOD) 命令来创建类型为 MODULE 的部件。当使用“创建程序”(CRTPGM) 命令绑定类型为 MODULE 的部件时，它创建类型为 PGM 的部件。如果对类型为 CSRC 的部件使用“创建绑定 C 程

序”(CRTBNDC) 命令，则不会创建类型为 MODULE 的部件，而只创建类型为 PGM 的部件。

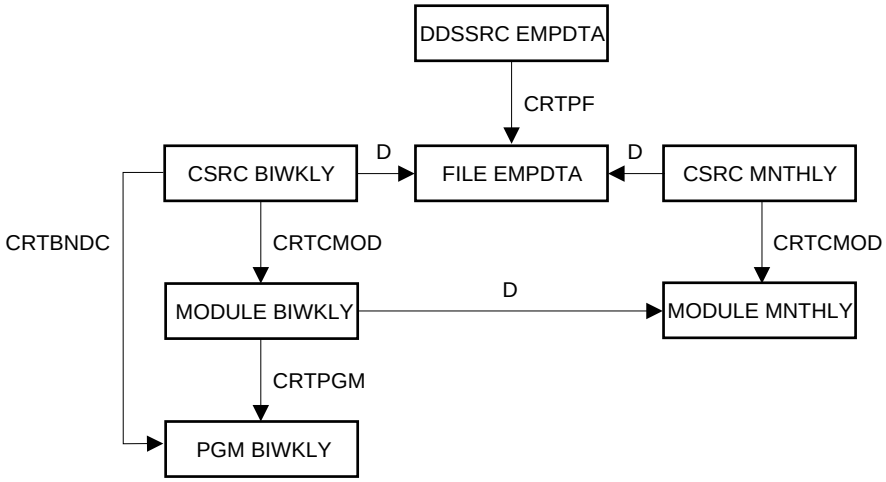

图 41. 组中的部件及其相关项. 类型为 CSRC 或 CINC 的部件带有语言 CLE。

BLDPART 命令调用的编译器和预处理器

表12列示了可以配合“应用程序开发管理器”功能部件使用的编译器和预处理器，并显示当在编译器或预处理器命令上指定 TGTRLS(*PRV) 时，是否可以使用特定编译器。可以使用 SBMJOB 命令以批处理方式发出 BLDPART 命令。然而，必须允许通过 BLDPART 命令调用的编译器以交互式方式运行。必须这样做，这是因为 BLDPART 命令是逐个处理部件的，且需要处理编译器返回的信息。

表 12. 可以与应用程序开发管理器功能部件配合使用的编译器和预处理器

编译器 / 预处理器语言	编译器 / 预处理器OS/400命令	在对“目标发行版”指定了 *PRV 时是否受支持
RPG/400	CRTRPGPGM	是
ILE RPG	CRTBNDRPG[1]，CRTRPGMOD	否
COBOL/400®	CRTCBLPGM	是
ILE COBOL	CRTBNDCBL[1]，CRTCBLMOD	否
ILE CL	CRTBNDCL[1]，CRTCLMOD	否
ILE C	CRTCMOD[1]，CRTBNDC	是
DDS[2]	CRTPF、CRTLF、CRTDSPF、CRTPRTF、CRTICFF	不适用
CL	CRTCLPGM	是
CLD	CRTCLD	是
CMD	CRTCMD	不适用
SQL SQL[2]	CRTSQLRPG、CRTSQLCBL、CRTSQLCI	是
SQL SQL[2]	CRTSQLRPGI、CRTSQLCBLI	否
CRTSRVPGM	CRTSRVPGM	是
CRTPGM	CRTPGM	是
MENU	CRTMNU TYPE(*UIM)	不适用

表 12. 可以与应用程序开发管理器功能部件配合使用的编译器和预处理器 (续)

编译器 / 预处理器语言	编译器 / 预处理器OS/400命令	在对 “目标发行版” 指定了 *PRV 时是否受支持
PNLGRP	CRTPNLGRP	不适用

注意:

1. BLDPART 命令使用缺省命令。
2. 根据部件类型和语言，使用适当的缺省编译器命令。

BLDPART 命令如何确定何时编译部件

本节描述在您发出 BLDPART 命令时构建部件的一些原因:

- 源部件以前未经编译。源部件不存在任何关系。
- 自从上次构建部件之后，BLDOPT 部件已更改。有关详情，参见第138页的『使用构建选项』。
- 用于编译给定部件的 BLDOPT 部件与先前构建所使用的部件不同。
- 先前构建使用的是 BLDOPT 部件，但现在使用的是缺省的编译命令，或相反。
- BLDPART 命令的 FORCE 参数被设置为 *YES。构建所有部件，无论它们是新的还是旧的。
- 搜索路径中找不到输出部件。
- 源部件或包含部件的时间戳记已更改。

构建报告描述了构建过程编译部件的原因。附录E. 构建报告消息列示构建报告的*构建原因*消息。

设置构建环境

项目管理员需要设置构建环境，以使每个开发者使用 BLDPART 命令时，可以一致地处理部件。项目管理员确定下列各项:

保存数据

在构建应用程序中的物理数据文件部件时是保存还是删除数据由 CRTPRJ 命令上的 SAVDTA 参数确定。如果该项目是使用 SAVDTA(*YES) 创建的，则您构建这些文件时，将保存与物理数据文件相关联的数据。如果项目是使用 SAVDTA(*NO) 创建的，则删除相关联的数据。

如果 CHGPRJ 命令上的 SAVDTA 参数值不再符合您的需求，则可以更改此值。管理员确定是否保存数据。开发者不能更改 SAVDTA 参数。

建立 BLDPART 准则

可以在您的组织中建立准则或规则，以在将开发者提升至项目层次结构中的下一个组之前，鼓励他们构建所有部件并验证部件是否能够象预期的那样工作。此外，还会建议使用特定 BLDPART 参数值。例如，可以鼓励开发者使用 SCOPE(*LIMITED) 来构建新部件。在构建操作成功后，可使用 SCOPE(*NORMAL) 构建该部件。

鼓励开发者以一致的方式构建部件的一种方法是在“编程开发管理器”实用程序中设置一个用户定义选项，以供所有开发者在准备好构建部件时使用。

示例

BLDPART 命令的以下用户定义选项指示在开发组中构建部件时要使用的特定搜索路径部件和构建范围。

```
BP = "BLDPART PRJ(&ZP) GRP(&ZG) TYPE(&ZT) PART(&ZN)
     SCHPTH(DEVGRP) SCOPE(*LIMITED)"
```

管理员可以通过创建类型为构建选项 (BLDOPT) 的部件来建立编译器处理标准。有关构建选项部件的详情，参见第138页的『使用构建选项』。

管理员可以创建供开发者使用的备用搜索路径。备用搜索路径是在类型为 SCHPTH 的部件中定义的。有关搜索路径的详情，参见第99页的『创建搜索路径部件』。

如果您的组织使用类型为 SCHPTH 和 BLDOPT 的部件，则不应鼓励开发者提升他们自己的 BLDOPT 或 SCHPTH 部件，除非他们得到管理员或小组长的许可。有关如何在项目层次结构中使用这些部件的详情，参见第145页的『项目层次结构中有多个构建选项部件』。

第一次构建

本节描述第一次构建部件。

假定您是在组 DEVELOPER1 中工作。在您发出 BLDPART 命令之前，部件之间没有相关关系。DEVELOPER1 中的下列部件是使用 BLDPART 命令缺省值构建的。对 PART 和 TYPE 参数指定了 *ALL。

在本章的示例中，如果必须标识部件的语言，则它会后跟部件名；例如，PGM BIWKLY-RPG。

RPGSRC BIWKLY Payroll 应用程序的部件；引用逻辑文件 FILE EMPID-PF 的 RPG 程序。

DDSSRC EMPID-PF 雇员标识；基于物理文件 FILE EMPDTA-PF 的逻辑文件。

DDSSRC EMPDTA-PF 雇员数据；DDS 源物理文件。

图42显示了在您发出 BLDPART 命令前组 DEVELOPER1 中的部件。

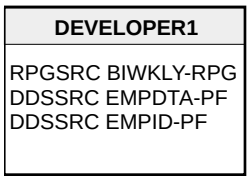

图 42. 在使用 BLDPART 命令前的组 DEVELOPER1

在运行 BLDPART 命令成功之后，就建立了部件之间的关系，而源部件已被编译成 DEVELOPER1 中的下列输出部件。

RPGSRC BIWKLY 已编译成 PGM BIWKLY-RPG

DDSSRC EMPID-PF 已编译成 FILE EMPID-PF

DDSSRC EMPDTA-PF 已编译成 FILE EMPDTA-PF

组 DEVELOPER1 现在包含图43中列示的部件。

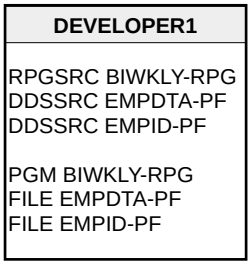

图 43. 使用 BLDPART 命令后的组 DEVELOPER1

下次使用 BLDPART 命令时，构建过程会检查相关关系，以确定需编译哪些部件以及这些部件是否需要编译。

图44显示了组 DEVELOPER1 中的部件之间的关系。同样，**C** 表示*创建*关系，而 **D** 表示*相关*关系。

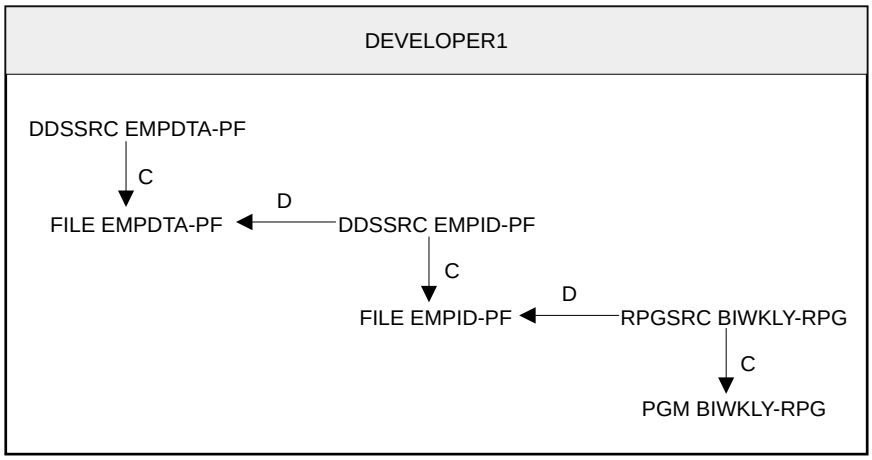

图 44. 组 DEVELOPER1 中的构建关系和构建过程的结果

您可以在 BLDPART 命令上构建源部件或输出部件。在必要时，会更新输出部件。例如，您还可以指定使用 *EXTENDED 构建范围来构建包含部件，将会构建与该包含部件相关的所有部件。

注: 当您正在引用包含部件或文件部件时，可以在源部件中指定文件名，但在这种情况下不要在源中使用特定库名。如果这样做，构建过程可能会在项目层次结构中找不到该部件的正确版本。

使用 *LIBL，而不是使用特定库的名称，或在需要的地方使用缺省值 *LIBL。如果编译时使用了相关项，而该相关项又不是“应用程序开发管理器”部件，则构建过程在构建报告中发出一条警告消息。

每次构建之后

在每次构建之后，您应当:

1. 检查构建完成消息，以了解是否有任何部件构建失败，或是否发出了任何警告。
2. 如果一个或多个部件构建失败，或发出了警告，则您应当检查构建报告。
3. 检查作业记录以确定可能的错误。
4. 检查编译器列表以确定可能的错误。

构建过程如何搜索部件

构建过程在搜索路径中查找部件的方式与编译器在系统上搜索库列表的方式相同。事实上，构建过程将库列表更改为包含搜索路径中的库。当您发出 BLDPART 命令时，会根据 SCHPTH 参数更改库列表。在构建过程之后，库列表返回构建过程之前它所处的状态。(有关搜索路径的详情，参见第97页的『第9章 使用搜索路径』。)考虑图45中的项目层次结构:

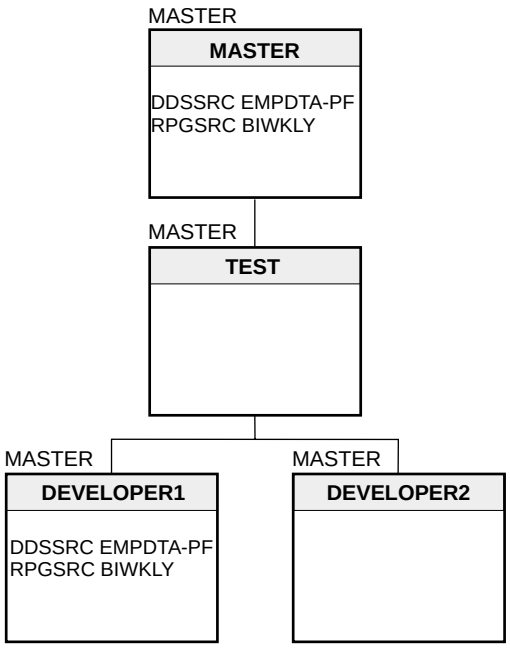

图 45. 构建过程和搜索路径

部件存在于组 MASTER 和 DEVELOPER1 中。将 DDSSRC EMPDTA-PF 和 RPGSRC BIWKLY 检出至组 DEVELOPER1，以便可以更改它们，但它们仍存在于组 MASTER 中。在更改它们之后，就从组 DEVELOPER1 发出 BLDPART 命令，该组的构建范围为 *NORMAL，搜索路径为 *DFT。

在构建成功之后，下列部件存在于组 DEVELOPER1 中:

DDSSRC EMPDTA-PF
FILE EMPDTA-PF
RPGSRC BIWKLY
PGM BIWKLY

现在，假定组 DEVELOPER2 中的某个开发者请求构建部件 RPGSRC BIWKLY。组 DEVELOPER2 不包含任何部件。构建过程在此组中搜索要构建的部件。在 DEVELOPER2 中找不到这些部件，因此构建过程在项目层次结构中的 DEVELOPER2 上面的组中查找，即查找组 TEST。此组不包含任何部件。这些部件是通过组 MASTER 构建的，输出 PGM BIWKLY 和 FILE EMPDTA-PF 被放入组 DEVELOPER2。这表示组 DEVELOPER2 中的开发者的构建部件的版本与组 DEVELOPER1 中的开发者构建的版本不同。

如果想要让构建过程通过不同于缺省搜索路径的组序列搜索部件，则您可能想创建您自己的搜索路径部件。对于发出的每个 BLDPART 命令，只会使用一个搜索路径部件。此部件的部件类型为 SCHPTH。BLDPART 命令上的 SCHPTH 参数有两个值: *DFT 和*名称*。第103页的『如何处理搜索路径部件』描述了这两个值。

因为运行 BLDPART 命令时 “应用程序开发管理器” 功能部件会使用库列表，所以，*引用相关部件时，不应在源代码中指定库名*。如果您这样做的话，当编译器在搜索路径中定义的库列表中搜索时，可能会使用不正确的部件版本。这可能会导致 BLDPART 命令发出警告消息。

如果您正在构建的包含或文件部件不存在于项目中，但存在于库列表中，则会在构建报告中发出警告。然而，有时构建会继续执行，并且不发出警告:

- 在 QCC/H 中的 C/400 包含文件，和在 QCLE/H 中的 ILE C/400 包含文件

  构建过程忽略系统包含文件。括在尖括号 (<...>) 中的文件名指示系统包含文件。括在双引号 ("...") 中的文件名指示用户包含文件。
- DB2/400 包含文件

  构建过程会忽略指定的 DB2® OS/400 版包含语句，如下所示:

  ```
  EXEC SQL INCLUDE SQLCA
  EXEC SQL INCLUDE SQLDA
  ```

  由于DB2 OS/400 版编译器不会从另一个成员或源文件读取源，所以这些语句不是真正的包含语句。
- 包含在 QRPG/QIRGINC 中的 RPG/400

BLDPART 命令只搜索搜索路径中的组来标识编译器所使用的消息文件。如果找不到 MSGF，或者它不是一个部件，则会发出警告。

使用构建部件命令

使用 BLDPART 命令可构建一个部件或整个应用程序。开发者和管理员都可以使用此命令。通常，项目管理员或小组长构建应用程序，而开发者构建应用程序的部件或组件。本节描述管理员和开发者如何控制构建过程的发生，根据 BLDPART 命令的构建范围、构建方式和构建强制参数上指定的值确定构建哪些部件。

必须在 BLDPART 命令的 PRJ 和 GRP 参数上指定项目名和组名。您必须对此组具有更新访问权。还必须指定部件类型和部件名。

要指示要构建特定类型和名称的部件，请在 PART 参数上输入名称，并在 TYPE 参数上输入类型。您可以指定 *ALL 以指示要构建所有类型的所有部件。例如，如果输入 TYPE(*ALL) PART(*ALL)，则会构建所有类型的所有部件。也可以对类型和部件指定类属名。查看第33页的『必需参数』，以了解类属名的使用。

如果要将构建输出部件添加至部件列表部件，则需指定 PARTL 参数。指定 PARTL(*PRV) 允许您使用用于此项目的上一个部件列表部件的名称（如果您是系统管理员的话）。但是，如果您是开发者的话，此参数会标识用于指定组的上一个部件列表部件的名称。如果您使用 QSECOFR 用户简要表运行此命令，则此值无效。PARTL 参数的缺省值是 *NONE，这指示不会将构建输出部件添加至部件列表部件。

如果命令上指定的组是使用 PARTLREQ(*YES) 创建的，则必须指定部件列表部件。

如果在 BLDPART 命令上指定了部件列表部件，则:

- 该部件列表部件必须存在于以构建这些部件的组起始的缺省搜索路径中。
- 构建期间创建的部件被添加至部件列表部件（如果它们还不存在的话）。（即使命令未能构建该部件，也可能会发生此情况）。

注

如果您正在使用部件列表部件来记录为修订更改或创建的所有部件（通过在用来执行修订的命令（例如，CHGPART、CRTPART，等等）的 PARTL 参数上指定该部件列表部件），您可能想要考虑在 BLDPART 命令的 PARTL参数上使用另一部件列表部件。这是因为 BLDPART 命令只将构建 OUTPUT 部件（即，非源部件）添加至部件列表。（例如，类型为 PGM 和 FILE 的部件。）如果只想导出非源部件，则可在稍后使用该部件列表来将修订导出至产品库。

如果您只想构建具有特定语言的部件，则使用 LANG 参数。缺省值 LANG(*ALL) 将构建所有语言的部件。此参数非常有用的一个示例是: 用于类型为 FILE 的部件。要构建所有打印文件，请指定 TYPE(DDSSRC)、PART(*ALL) 和 LANG(PRTF)。

如果您正在构建类型为 CBLSRC 且语言为 CBL 或 SQLCBL 的部件，且如果它包含多个程序，则 BLDPART 命令只编译第一个程序。其他程序被忽略，并在构建报告中发出警告消息。

不能使用 BLDPART 命令构建 System/36 应用程序，这是因为 System/36 编译器不返回构建过程所需的信息。有关如何编译 System/36 对象的详情，参见第212页的『构建 System/36 应用程序』。

示例

此示例说明 BLDPART 命令如何构建组 MASTER 中的所有部件。

使用 BLDPART 命令

```
BLDPART PRJ(PAYROLL) GRP(MASTER) TYPE(*ALL) PART(*ALL)
```

使用“编程开发管理器”实用程序

在“用 PDM 来使用部件”屏幕上选择选项 14（构建）。

先前 BLDPART 命令上使用的可选参数及其缺省值如下所示。

```
SCHPTH(*DFT) SCOPE(*NORMAL) FORCE(*NO) BLDMODE(*COND) SAVLST(*NO)
BINDSTEP(*YES) PARTL(*NONE)
```

下面各节更详细地说明了 SCOPE、FORCE、BLDMODE、SAVLST、BINDSTEP 和 PARTL(*NONE) 参数。有关 SCHPTH 参数的更多说明，参见第103页的『如何处理搜索路径部件』。

设置构建范围

BLDPART 命令上的 SCOPE 参数控制了构建过程期间要处理的部件的范围。构建范围指的是如何使用部件之间的关系来 确定考虑构建哪些部件。一定不会构建交叉项目部件，但构建过程会更新它找到的关系*目标*和*源*交叉项目部件。在 SCOPE 参数上，您可以从 4 个值中进行选择以确定要构建的部件: *NORMAL、*LIMITED、*DIRCHAIN 和 *EXTENDED。

如果没有在 BLDPART 命令上指定任何信息，则缺省情况是使用构建范围 *NORMAL。这就是在构建部件时使用的最典型的范围。正常构建范围使 BLDPART 命令上指定的部件成为当前部件。如果部件是使用用来创建它的所有源和相关部件的最新版本构建的，则该部件是**当前**部件。相反地，如果在上次构建部件之后已更改了源和相关对象，则该部件已**失效**。换而言之，如果使用构建范围 *NORMAL，则会构建 PART 参数上指定的部件和它所依赖的所有其他部件，以及那些部件所依赖的部件，等等。

例如，如果使用正常构建范围构建部件 RPGSRC BIWKLY，而此程序包含对部件 FILE BIWKLY 中的记录的引用，则会处理这两个部件。因此，这一个构建命令会导致创建两个部件，即 PGM BIWKLY 和 FILE BIWKLY。

当开发者在他们的开发组中测试相关的部件时，应当使用此值。当管理员在项目层次结构中的每个组中构建该应用程序时，也应使用它。

构建范围 *LIMITED 导致只处理 BLDPART 命令上指定的部件。对正在构建的部件中的其他相关项的引用被忽略。在创建新部件或充分更改现存部件之后，开发者应当使用此构建范围。构建范围 *LIMITED 允许您验证部件是否象预期的那样工作（以隔离方式）。

构建范围 *DIRCHAIN（或直接链）执行的操作与构建范围 *NORMAL 相同，但会执行进一步的构建处理。还会构建直接依赖于任何正在构建的部件的所有部件。

构建范围 *EXTENDED 执行的操作与构建范围 *DIRCHAIN 相同，但它会执行进一步的构建处理。还将构建直接或间接与构建的所有部件相关的任何部件。例如，在更改字段引用文件中的字段之后，管理员或开发者可使用构建范围 *EXTENDED 来构建应用程序。

第130页的图46说明了构建范围功能。DDSSRC EMPID-PF 是一个部件，围绕着此部件描述了构建范围 *LIMITED、*NORMAL、*DIRCHAIN 和 *EXTENDED。长竖线指示根据指定的范围在构建过程中涉及到的部件。虚竖线指示可能会构建在构建过程中涉及到的部件（根据指定的范围）。SCOPE(*NORMAL) 是缺省值。此示例假定所有部件都已失效。注意，可以在 BLDPART 命令上指定源 DDSSRC EMPID-PF 或输出 FILE EMPID-PF。这两者给出相同的结果。

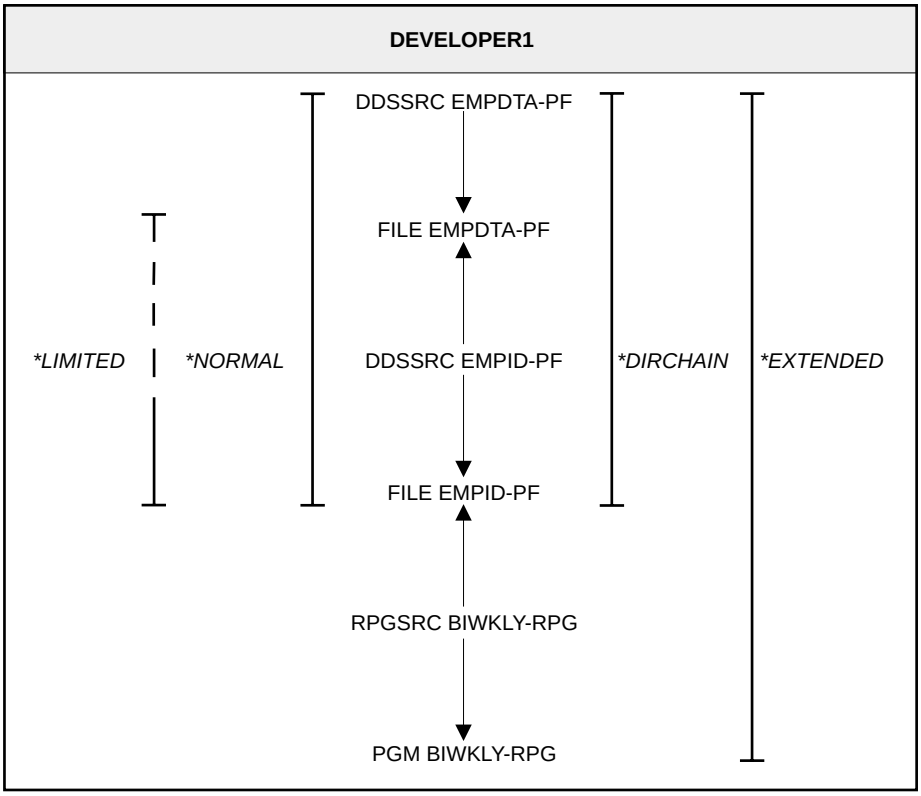

图 46. BLDPART 命令和 DDSSRC EMPID-PF 部件的构建范围

如果选择 *LIMITED、*NORMAL、*DIRCHAIN 或 *EXTENDED 构建范围，则上图中的部件按如下方式构建:

***LIMITED** 只构建 DDSSRC EMPID-PF。如果 FILE EMPDTA-PF 存在，则会重新创建 FILE EMPID-PF。更新 DDSSRC EMPID-PF 和 FILE DDSSRC-LF 之间以及 DDSSRC EMPDTA-PF 和 FILE EMPDTA-PF 之间的关系。

检查所有相关部件，了解它们当中是否有任何已更改但尚未构建。*LIMITED 构建范围在检查程序和文件的部件相关项时停止在第一层相关项上。作为 *LIMITED 构建范围一部分的虚竖线条显示该构建过程使用 FILE EMPDTA-PF 来确定是否需要构建 DDSSRC EMPID-PF。如果 FILE EMPDTA-PF 已更改，则构建 DDSSRC EMPID-PF。

***NORMAL** 构建 DDSSRC EMPID-PF 所依赖的部件。在示例中，它依赖于 FILE EMPDTA-PF，因此构建 DDSSRC EMPDTA-PF。

构建 BLDPART 命令上指定的部件，即 DDSSRC EMPID-PF。重新创建 FILE EMPID-PF。

***DIRCHAIN** 会对 *DIRCHAIN 构建范围构建已经为 *NORMAL 构建范围构建的部件。

***EXTENDED** 构建 DDSSRC EMPID-PF 所依赖的部件。在示例中，它依赖于 FILE EMPDTA-PF，因此构建 DDSSRC EMPDTA-PF。

构建 BLDPART 命令上指定的部件，即 DDSSRC EMPID-PF。重新创建 FILE EMPID-PF。

构建依赖于 FILE EMPID-PF 的部件。它是 RPGSRC BIWKLY。重新创建 PGM BIWKLY。

如果使用 SCOPE(*EXTENDED)，则检查所有相互关联的部件并相应地进行构建。在此示例中，因为所有部件都相互关联，所以在带有 SCOPE(*EXTENDED) 的 BLDPART 命令上指定任何部件都会得到相同的结果。

如果构建类型为 PGM、FILE、CLD、CMD MODULE 或 SRVPGM 的部件，则 BLDPART 命令会编译源（如果有必要的话），即使指定了 SCOPE(*LIMITED) 亦如此。可以在 BLDPART 命令上指定源或输出部件。第52页的表5列示了“应用程序开发管理器”功能部件支持的 OS/400 对象类型。

通过使用第130页的图46中使用的部件，让我们来构建编译过的 RPG 程序 BIWKLY。如果您选择 *LIMITED、*NORMAL、*DIRCHAIN 或 *EXTENDED 构建范围，则部件按如下方式构建:

***LIMITED** 只构建 PGM BIWKLY-RPG。我们请求 BLDPART 命令将构建过程限制为只构建 PGM BIWKLY-RPG。编译 RPGSRC BIWKLY-RPG。

使用 RPGSRC BIWKLY 直接依赖的所有部件来确定是否编译 RPGSRC BIWKLY。在第130页的图46中，这表示构建过程使用 FILE EMPID-PF 来确定是否需要构建 RPGSRC BIWKLY-RPG。如果 FILE EMPID-PF 或 RPGSRC BIWKLY 已更改，则会构建 BIWKLY，但不编译 FILE EMPID-PF。

***NORMAL** 构建 RPGSRC BIWKLY-RPG 所依赖的部件。在示例中，它依赖于 FILE EMPID-PF。先构建物理文件，然后再构建逻辑文件；因此，先构建 DDSSRC EMPDTA-PF，然后构建 DDSSRC EMPID-PF。

构建 BLDPART 命令上指定的部件，即 RPGSRC BIWKLY-RPG，并重新创建 PGM BIWKLY-RPG。

***DIRCHAIN** 为 *DIRCHAIN 构建范围构建已经为 *NORMAL 构建范围构建的部件。

***EXTENDED** 为 *EXTENDED 构建范围构建已经为 *NORMAL 构建范围构建的部件。

示例

下面四个命令说明基于第130页的图46的 *LIMITED、*NORMAL、*DIRCHAIN 和 *EXTENDED 构建范围的使用。

根据以下命令，只构建 DDSSRC EMPID-PF。

```
BLDPART PRJ(PAYROLL) GRP(DEVELOPER1) TYPE(DDSSRC) PART(EMPID)
        SCHPTH(*DFT) SCOPE(*LIMITED) FORCE(*NO)
        BLDMODE(*COND) SAVLST(*NO)
```

根据以下命令，构建 DDSSRC EMPDTA-PF 和 DDSSRC EMPID-PF。

```
BLDPART PRJ(PAYROLL) GRP(DEVELOPER1) TYPE(DDSSRC) PART(EMPID)
        SCHPTH(*DFT) SCOPE(*NORMAL) FORCE(*NO)
        BLDMODE(*COND) SAVLST(*NO)
```

根据以下命令，构建 DDSSRC EMPDTA-PF 和 DDSSRC EMPID-PF。这样的话，就会为 *DIRCHAIN 构建范围构建已经为 *NORMAL 构建范围构建的部件。

```
BLDPART PRJ(PAYROLL) GRP(DEVELOPER1) TYPE(DDSSRC) PART(EMPID)
        SCHPTH(*DFT) SCOPE(*DIRCHAIN) FORCE(*NO)
        BLDMODE(*COND) SAVLST(*NO)
```

根据以下命令，构建 DDSSRC EMPDTA-PF、DDSSRC EMPID-PF 和 RPGSRC BIWKLY-RPG。构建 RPGSRC BIWKLY-RPG 的条件是：它是以前构建的（因此与 DDSSRC EMPID-PF 建立了关系），并且现在已失效。

```
BLDPART PRJ(PAYROLL) GRP(DEVELOPER1) TYPE(DDSSRC) PART(EMPID)
        SCHPTH(*DFT) SCOPE(*EXTENDED) FORCE(*NO)
        BLDMODE(*COND) SAVLST(*NO)
```

示例

第133页的图47说明当包括名为 FILE MNTHLYL-LF 的逻辑文件时的构建范围功能。在 DDSSRC EMPDTA-PF 中，更改了 DDSSRC EMPID-PF 中引用的字段。DDSSRC EMPID-PF 是一个部件，围绕着此部件描述了构建范围 *LIMITED、*NORMAL、*DIRCHAIN 和 *EXTENDED。长竖线指示根据指定的范围在构建过程中涉及到的部件。虚竖线指示在构建过程中涉及到的，可能不会构建的部件（根据指定的范围）。此示例假定所有部件都已失效，并且，以前在目标组中已构建这些部件。 注意，可以在 BLDPART 命令上指定源 DDSSRC EMPID-PF 或输出 FILE EMPID-PF。这两者给出相同的结果。

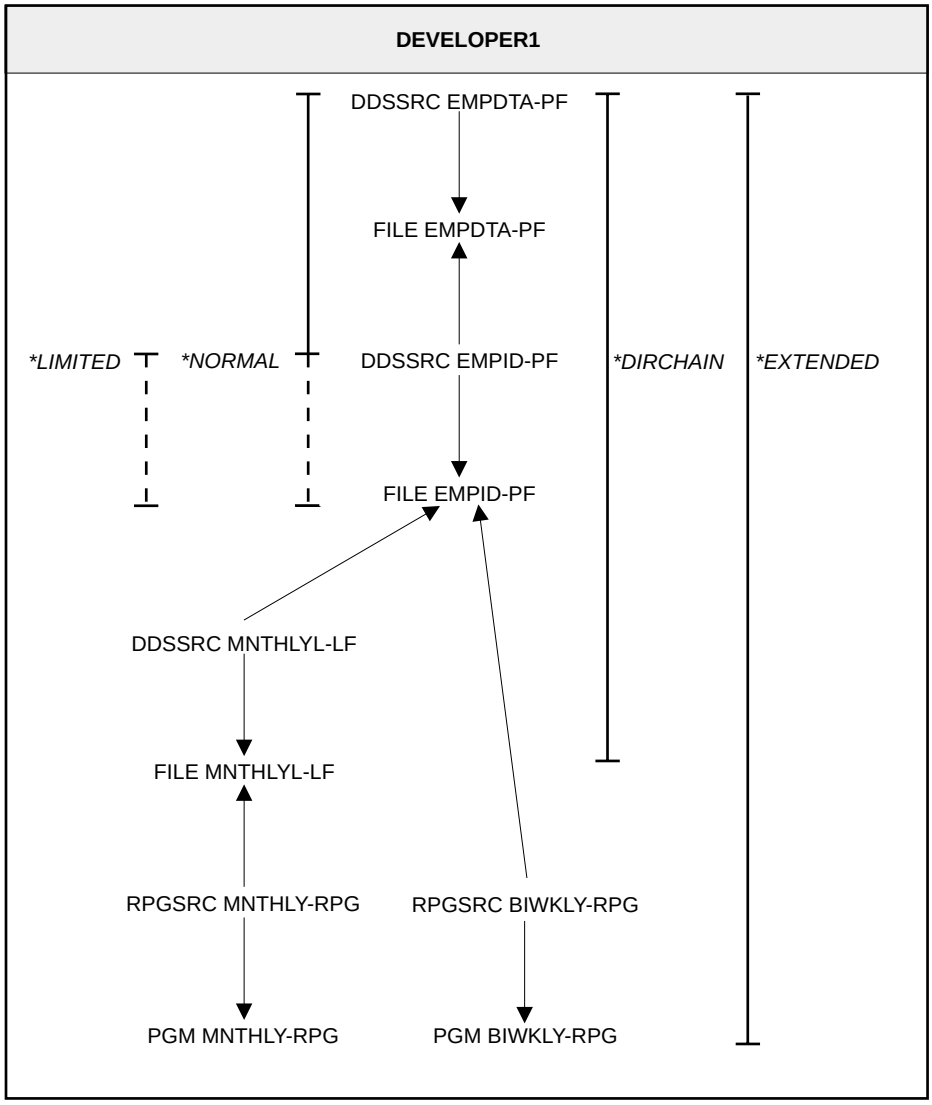

图 47. BLDPART 命令和 DDSSRC EMPID-PF 部件的构建范围

如果选择 *LIMITED、*NORMAL、*DIRCHAIN 或 *EXTENDED 构建范围，则前图中的部件按如下方式构建:

***LIMITED** 虚竖线指示不构建 DDSSRC EMPID-PF，并且不重新创建 FILE EMPID-PF，这是因为逻辑文件基于 FILE EMPID-PF。

***NORMAL** 构建 DSSSRC EMPID-PF 所依赖的部件。在示例中，它依赖于 FILE EMPDTA-PF，因此会构建 DDSSRC EMPDTA-PF。

不构建 BLDPART 命令上指定的部件（即 DDSSRC EMPID-PF），这是因为逻辑文件基于此文件。

***DIRCHAIN** 构建 DSSSRC EMPID-PF 所依赖的部件。在此示例中，它依赖于 FILE EMPDTA-PF，因此会构建 DDSSRC EMPDTA-PF。

构建 BLDPART 命令上指定的部件，即 DDSSRC EMPID-PF。

构建直接依赖于 FILE EMPID-PF 的部件。这意味着构建 DDSSRC MNTHLYL-LF。重新创建 FILE MNTHLYL-LF。

***EXTENDED** 构建 DDSSRC EMPID-PF 所依赖的部件。在示例中，它依赖于 FILE EMPDTA-PF，因此会构建 DDSSRC EMPDTA-PF。

构建 BLDPART 命令上指定的部件，即 DDSSRC EMPID-PF。重新创建 FILE EMPID-PF。

构建依赖于 FILE EMPID-PF 的部件。构建 DDSSRC MNTHLYL-LF，以重新创建 FILE MNTHLYL-LF。

构建 RPGSRC BIWKLY，以重新创建 PGM BIWKLY。

如果使用 SCOPE(*EXTENDED)，则检查所有相互关联的部件并相应地进行构建。在此示例中，因为所有部件都相互关联，所以在带有 SCOPE(*EXTENDED) 的 BLDPART 命令上指定任何部件都会得到相同的结果。

确定是否构建所有部件

在处理部件时，构建过程使用缺省值 FORCE(*NO)。这意味着只处理那些需要构建的部件。 这使构建命令不必处理不需要处理的部件。如果某个部件是当前部件，则不会再次构建它。

通过指定 FORCE(*YES)，可确保处理所有部件，即使有的部件尚未更改。您可能想要为确保更新应用程序中的所有部件而执行此操作。

注意，如果指定 BLDMODE(*COND)，并且未能构建某个相关部件，则即使 FORCE 是 *YES，也不构建任何依赖于它的部件。

假定您的组 DEVELOPER1 中有两个部件: RPGSRC BIWKLY（ 自发出上一个 BLDPART 命令之后，此部件未发生更改）和 PGM BIWKLY（它是 RPGSRC BIWKLY 的输出部件）。如果您在指定了 FORCE(*NO) 的情况下选择构建 RPGSRC BIWKLY，则不执行任何操作。因为 BIWKLY 是当前部件，所以不编译它。如果您在指定了 FORCE(*YES) 的情况下选择构建它，则会编译它。

检出、更改和提升部件时，在开发级别上方的组中的部件可能会很快地变得过时，这可能会导致在开发组中编译那些部件。

您可能会发现，让管理员在开发组上方的组中定期调用 BLDPART TYPE(*ALL) PART(*ALL) FORCE(*NO) 会显著减少在开发组中执行的编译。

确定构建方式

构建过程使用的方式或方法由 BLDMODE 参数上指定的值确定。三种可能的构建方式分别是“条件”、“无条件”和“纯报告”。

缺省值是 BLDMODE(*COND)。“条件”构建方式表示当您在 BLDPART 命令上指定一个部件时，如果因为任何原因未能构建此部件所依赖的那些部件，则不构建指定的部件。例如，部件 RPGSRC BIWKLY 包含对部件 FILE BIWKLY 的引用。如果未编译 DDSSRC BIWKLY，则既不创建 PGM BIWKLY 也不创建 FILE BIWKLY。

“无条件”构建方式 (BLDMODE(*UNCOND)) 指示即使部件未能成功地编译，也会编译依赖于它的部件。例如，即使未能编译部件 DDSSRC BIWKLY，但仍将编译 RPGSRC BIWKLY。

“纯报告”构建方式 (BLDMODE(*RPTONLY)) 指示您不想实际地编译部件，而只想生成一个报告，此报告列示当在 BLDMODE 参数上指定 *COND 时将会编译的部件。因为此选项阻止了部件的编译，所以构建过程只使用上次成功构建所生成的关于部件的关系信息。

保存构建报告和编译器列表

在您使用 BLDPART 命令时，总是会生成构建报告。使用 SAVLST 参数来将构建报告和编译器列表保存到输出组中的文件 QALYBLDREP 中。缺省情况是不保存它们。要指示您想保存它们，请指定 SAVLST(*YES)。

如果构建大量部件且您想自动清理 BLDPART 命令生成的假脱机文件，则使用 SAVLST(*DLT) 参数。此参数将删除所有成功编译的列表。如果所有报告都构建成功，未发出任何警告且未指定 BLDMODE(*RPTONLY)，则还将删除构建报告。

在创建一个组时，便会在新组中创建一个名为 QALYBLDREP 的物理文件。QALYBLDREP 包含两种类型的输出列表，即构建报告和编译器列表。构建报告存储在名为 QREPxxxxxx 的成员中，其中，xxxxxx 被替换为一串数字。成员类型是 REPORT。这些部件的描述由 BLDPART 命令上指定的部件的名称和类型组成。

编译器列表存储在名为 QLSTxxxxxx 的成员中，其中，xxxxxx 被替换为一串数字。它们的成员类型是 BLDPART 命令生成的报告的名称，即 QREPxxxxxx。这些成员的描述由编译的源部件的名称和类型组成。

注: 当 SAVLST 是 *YES 时，BLDPART 命令期望编译器生成缺省假脱机文件名。如果不是这样的话，则 BLDPART 命令不能保存编译器列表。在某些情况下，在编译器命令的 PRTFILE 参数上更改缺省名称会导致这种情况。

图48显示了构建报告的第一页，此页列示了 BLDPART 命令上指定的参数。

```
5722WDS   V5R1M0          应用程序开发管理器      - 构建报告    08/11/96    12:00:00          页面 . . :  0001

项目  . . . . . . . . . . . . . . . . . :  PAYROLL      1
组  . . . . . . . . . . . . . . . . . . :  DEVELOPER1   2
类型 . . . . . . . . . . . . . . . . . . :  RPGSRC       3
部件 . . . . . . . . . . . . . . . . . . :  *ALL         4
搜索路径 . . . . . . . . . . . . . . . . :  *DFT         5
构建范围 . . . . . . . . . . . . . . . . :  *NORMAL      6
强制构建 . . . . . . . . . . . . . . . . :  *NO          7
构建方式 . . . . . . . . . . . . . . . . :  *RPTONLY     8
保存列表 . . . . . . . . . . . . . . . . :  *NO          9
执行绑定步骤 . . . . . . . . . . . . . . :  *YES         10
使用的搜索路径部件 . . . . . . . . . . . :  *DFT         11
使用的搜索路径 . . . . . . . . . . . . . :  PAYROLL                    DEVELOPER1
                                        PAYROLL                    TEST          12
```

图 48. 构建报告的第一页 – 报告标题

1 在 BLDPART 命令上指定的项目。

2 在 BLDPART 命令上指定的组。

3 在 BLDPART 命令上指定的类型。

4 在 BLDPART 命令上指定的部件。

5 在 BLDPART 命令上指定的搜索路径。

6 在 BLDPART 命令上指定的范围。

7 在 BLDPART 命令上指定的强制。

8 在 BLDPART 命令上指定的方式。

9 在 BLDPART 命令上指定的 SAVLST 选项。

10 在 BLDPART 命令上指定的绑定步骤选项。

11 BLDPART 使用的搜索路径部件（如果使用了缺省搜索路径，则为 *DFT。）

12 BLDPART 使用的搜索路径。（这可以是搜索路径部件中的组列表，也可以是缺省搜索路径。）

图49显示了构建报告的第二页，此页列示构建过程中发现的警告和错误的消息，包括消息号和消息文本。查看附录E. 构建报告消息，以获取构建报告*构建原因*消息的列表和可能会出现的构建报告*警告*消息的列表。

```
5722WDS   V5R1M0          应用程序开发管理器      - 构建消息  05/08/01     12:00:00          页面 . . :  0002

消息标识  文本
-------  ------------------------------------------------------------------------------------------------------------------------
ADM4455   编译 PAYROLL DEVELOPER1 RPGSRC BIWKLY 失败。

                                    * * * * *  构     建     消     息     结     束  * * * * *
```

图 49. 构建报告的第二页 – 消息

图50是构建报告的第三页，此页显示构建过程创建的构建输出。这些输出包括通过编译源部件生成的程序和文件。本节按源类型分类。为所有编译的 CSRC 部件打印一个表，为所有编译的逻辑文件打印一个表，等等。此格式用于源类型 RPGSRC、CBLSRC、CSRC、CLESRC、CLDSRC、CMDSRC、DDSSRC（PF、LF、DSPF、PRTF 和 ICFF）、CLPSRC、PNLSRC 和 BNDSRC。

```
5722WDS   V5R1M0          应用程序开发管理器      - 构建输出   05/08/01     12:00:00               页面 . . :  0003

      13                                15
      DDSSRC-PF                   创建的                BLDOPT
部件          组                  文件         部件          组                  构建原因 18
----------    --------------------  ----------   ----------    --------------------  ------------------------------------------------------
EMPDTA        DEVELOPER1            EMPDTA       QDFT          DEVELOPER1            以前未构建源部件。
 14                                   16           17

      DDSSRC-LF                   创建的                BLDOPT
部件          组                  文件         部件          组                  构建原因
----------    --------------------  ----------   ----------    --------------------  ------------------------------------------------------
EMPID         DEVELOPER1            EMPID        *DFT          *NONE                 以前未构建源部件。
                                         * * * * *  构    建    报    告    结    束  * * * * *
```

图 50. 构建报告的第三页 – 构建输出

13 已编译的源部件的类型和语言；例如，DDSSRC-PF。

14 已编译的源部件的名称以及在其中找到该部件的组。

15 创建的输出的类型，例如，程序或文件。

16 创建的输出部件的名称。

17 使用的 BLDOPT 部件（如果有的话）的名称以及在其中找到该部件的组。如果报告中的 BLDOPT 字段对部件显示 *DFT，并对组显示 *NONE（就象图50中的报告的第二行那样），则表示构建过程使用了缺省编译器命令。没有使用 BLDOPT 部件中的编译器命令来编译该部件。

18 发生构建过程的原因。

绑定程序

如果您想让构建过程对类型为 MODULE 的部件执行“创建程序”(CRTPGM) 命令，请在*执行绑定步骤*参数使用缺省值 BINDSTEP(*YES)。如果不想执行绑定步骤，请指定 BINDSTEP(*NO)。图51是样本构建报告中的一部分，它显示指定 BINDSTEP(*YES) 时的构建过程的输出。

图51显示使用的入口模块源，使用的 BLDOPT 部件（如果有的话）以及执行构建过程的原因。

```
5722WDS   V5R1M0              应用程序开发管理器     - 构建输出   05/08/01    17:49:11          页面 . . :  0002

      CSRC                          创建的          BLDOPT
部件        组                    对象        部件        组                  构建原因
----------  --------------------  ----------  ----------  ------------------  ----------------------------------------------------
PGM2USRINC  USER      1           PGM2USRINC  *DFT        *NONE               *FORCE 设置为 *YES。

      MODULE                        创建的          BLDOPT
部件        组                    程序        部件        组                  构建原因
----------  --------------------  ----------  ----------  ------------------  ----------------------------------------------------
PGM2USRINC  USER                  PGM2USRINC  PGM2USRINC  USER                以前尚未构建源部件。
                                  2           3                               4
                        * * * * * 构   建   报   告   结   束 * * * * *
```

图 51. 显示绑定步骤的输出的样本构建报告

1 源部件和创建的对象

2 创建的类型为 PGM 的部件

3 使用的构建选项部件的名称

4 在每种情况下构建部件的原因

将部件添加至部件列表部件

可以使用 PARTL 参数来将部件添加至任何部件列表部件。指定部件列表部件的名称或使用缺省值 *NONE。

如果指定的组是使用 PARTLREQ(*YES) 定义的，则 PARTL 参数是必需的。如果指定了部件列表部件，则:

- 它必须存在于构建部件的组所在的提升路径中
- 将构建期间创建的输出部件的名称添加至部件列表部件（如果它们不存在的话）

构建较少部件

在开发活动最繁忙的时候，您可能想在独立于所有其他开发组的组（如集成和测试组）中使用 SCOPE(*NORMAL) 构建所有部件，且至少每晚执行一次。于是，集成和测试组中的所有构建输出部件（如 FILE、MODULE 和 PGM 都将是最新的，以后不需要重新构建。此外，在开发组中只有受更改影响的部件才会获得正常构建，因而改进了 BLDPART 命令的性能。

使用构建选项

本节讨论下列主题:

在创建构建选项部件之前要考虑的事项
创建构建选项部件
如何处理类型为 BLDOPT 的部件
项目层次结构中有多个构建选项部件
用于物理文件和逻辑文件的特殊构建选项命令

在创建类型为 BLDOPT 的部件之前需要考虑的事项

项目管理员需要确定使用类型为 BLDOPT 的部件的策略。项目管理员应考虑下列问题:

- 在处理程序时，您的组织是否只使用编译器缺省值？

  如果这样的话，则无需为应用程序创建类型为 BLDOPT 的部件，类型为 PGM（语言为 CLE、RPGLE、CBLLE 和 CLLE）的部件除外。

- 您的组织是否想确保在项目层次结构中跨应用程序一致地使用编译器选项？例如，您的组织是否要求在编译时优化所有程序？
- 您的组织是否想确保在每个开发组、在每个测试组以及在根组中一致地使用编译器选项？

  例如，在开发组中，您可以：创建类型为 BLDOPT 的部件以确保将程序编译为带有调试信息；生成编译器列表；以及在编译器列表中包括次级消息文本。在测试组中，可以创建类型为 BLDOPT 的部件，以确保将程序编译为不带调试信息，并生成编译器列表。在根组中，您可以创建类型为 BLDOPT 的部件以确保将程序编译为不带调试信息或不生成编译器列表，并进行优化。

- 是否将特殊值 QDFT 用作类型为 BLDOPT 的部件的名称？

  查看第144页的『如何处理类型为 BLDOPT 的部件』，以了解如何使用具有此名称的构建选项部件的说明。应当对包含要让您的组织在缺省情况下使用的缺省编译选项的任何构建选项部件使用此名称。

创建构建选项部件

在处理部件时，BLDPART 命令使用与编译器命令上的每个可选参数相关联的所有缺省值，但 CRTCMOD 和 CRTBNDC 命令是例外。*编译器选项*参数 (OPTION) 设置为值 *SYSINCPATH，因此，当您构建一个包含部件时，将构建该包含部件在项目层次结构的该特定分支中的最低版本。注意，*SYSINCPATH 不是此参数的缺省值。缺省值为 *USRINCPATH。第140页的图52显示了系统提供的构建选项部件。

如果您不想在构建部件时使用缺省值，则必须指定要在类型为 BLDOPT 的部件中处理的确切命令。例如，您可能想在 CRTRPGPGM 命令上指定 *NOLIST，以便不生成列表。

构建选项 (BLDOPT) 部件是一个“应用程序开发管理器”部件，它允许您指定编译器或处理命令。创建和管理此部件的方式与其他“应用程序开发管理器”部件类似，但不能构建该部件。

要创建构建选项部件，请使用 CRTPART 命令来象创建源部件那样创建该部件。当您创建构建选项部件时，CRTPART 命令自动地将缺省构建选项部件从与“应用程序开发管理器”功能部件一起安装的 QADM 库复制到类型为 BLDOPT 的部件中，以为您提供一个起始点。

此部件包含 BLDPART 命令在处理部件时使用的所有编译器或预处理器命令。此部件中指定的每个命令都对它的所有可选参数使用缺省值，但标有创建构建选项部件的本节开头处描述的 CRTCMOD 和 CRTBNDC 命令是例外。

必需参数支持替换变量。这表示从正在构建的部件派生适当的库、源文件名和对象名。

缺省 BLDOPT 部件中的每个 CL 命令都被注释掉。如果要使用特定的 CL 命令，则使用 CHGPART 命令来编辑您正在创建的构建选项部件。从要使用的命令中除去注释定界符（/* 和 */），指定要使用的编译选项，并且只删除您不需要的所有命令。在一个构建选项部件中，可以有许多编译器或预处理器命令。

当构建过程使用构建选项部件时，它不检查在该部件中找到的信息的语法。如果命令的语法中有错误，则不能构建该部件。构建选项部件的 OS/400 成员类型为 CLP；因此，您应当使用“源输入实用程序”(SEU) 检查部件的语法，以避免不正确地指定 CL 命令。有关此实用程序的详情，参见*ADTS for AS/400: Source Entry Utility*一书。

如果想要使用您自己的预处理器或后处理器，可使用一个与部件类型相同的标号，并在构建选项部件中指定任何命令。如果找到与正在构建的部件的部件类型同名的标号，则在该语句上运行 CL 命令，并且不进行任何语法检查。在运行这个带标号的 CL 命令之后，不发出任何其他编译器命令。替换变量在带标号的 CL 命令上是受支持的。如果发现多个对正在构建的部件有效的命令，则使用第一个带标号命令。如果在构建选项文件中的任何地方找到有效的带标号命令，则将其用于构建，即使它前面有另一无标号编译器命令。

示例

如果想要使用名为 PRCRPGSRC 的预处理器对 RPGSRC 部件 PART01 进行预处理，则可以创建构建选项部件 PART01 并添加 CL 命令，如下所示:

```
RPGSRC:  PRCRPGSRC OBJ(QTEMP/&ZE) SRCFILE(&L/&F) SRCMBR(&ZN)...
```

在这种情况下，将使用 PRCRPGSRC 处理器构建 RPGSRC PART01，而不是使用 CRTRPGPGM 命令，即使可能已对此 CL 命令取消注释。PRCRPGSRC 是一个预处理器，它应当调用 CRTRPGPGM 命令来通过它可能已创建的中间源创建 RPG 程序。因为 IBM 编译器会编写“应用程序开发管理器API”空间并提供构建信息，预处理器或后处理器将负责调用适当的编译器或使用 API 编写构建信息。有关详情，参见第152页的『确保构建过程了解部件之间的关系』。

如果在一行本身上找到标号，则 BLDPART 命令将使用在下一行上找到的命令。

注意，在构建选项部件中，BLDPART 命令不识别带有与部件类型不相同的标号的命令。通常，除了添加新参数之外，您不应改变示例命令。这能确保构建过程成功地运行该命令。图52显示了所有编译器和预处理器命令，这些命令被复制到您创建的构建选项部件中。

```
/*****************************************************************/
/*                                                               */
/* Application Development Manager     recognizes following      */
/* substitution variables in a build options part (part of       */
/* type BLDOPT):                                                 */
/*                                                               */
/*                                                               */
/* &F  - Source file name of the part being compiled.            */
/*                                                               */
/* &L  - Library name associated with the group containing       */
/*       the part being compiled.                                */
/*                                                               */
/* &N  - Object/member name of the part being compiled.          */
/*                                                               */
/* &O  - Name of the library where the output object of the      */
/*       BLDPART command is placed.                              */
/*                                                               */
/* &ZA - Language attribute of the source part being compiled,   */
/*       not of the output part.                                 */
/*                                                               */
/* &ZC - &ZC is replaced with BLDPART.                           */
/*                                                               */
/* &ZD - &ZD is replaced by the type of the output part created  */
/*       by the BLDPART command. If the source part has not      */
/*       been previously built, this is the same as the type     */
/*       of the source part.                                     */
/*                                                               */
/* &ZE - &ZE is replaced by the name of the output part being    */
/*       created by the BLDPART command. If the source part has  */
/*       not been previously built, this is the same as the name */
/*       of the source part.                                     */
/*                                                               */
/*       For a user-defined type of *NONE, &ZE is set to the     */
/*       first output member met by the BLDPART command, or the  */
/*       name of the part with the user-defined type of *NONE,   */
/*       if the part has not been processed before.              */
/*                                                               */
/* &ZG - Name of the group where the outputs of the BLDPART      */
/*       command are placed.                                     */
/*                                                               */
/* &ZN - Name of the part being compiled.                        */
/*                                                               */
/* &ZO - The value of the Build Scope parameter on the BLDPART   */
/*       command. The values are *NORMAL, *LIMITED, *DIRCHAIN, and */
/*       *EXTENDED.                                              */
/*                                                               */
/* &ZP - Name of the project containing the part being compiled. */
/*                                                               */
/* &ZS - The value of the Search Path parameter on the BLDPART   */
/*       command. This value is the name of the search path part. */
/*                                                               */
/* &ZT - Type of part being compiled.                            */
/*                                                               */
/* &ZY - Name of the part-list part specified on the PARTL       */
/*       parameter.                                              */
/*                                                               */
/*****************************************************************/
```

图 52. 系统提供的构建选项部件 (1/4)

```
/*****************************************************/
/* PGM compilers                                     */
/*****************************************************/
/* CRTRPGPGM  PGM(&O/&ZE) SRCFILE(&L/&F) SRCMBR(&ZN) +
              REPLACE(*YES) USRPRF(*USER) AUT(*EXCLUDE) */
/* CRTCBLPGM  PGM(&O/&ZE) SRCFILE(&L/&F) SRCMBR(&ZN) +
              REPLACE(*YES) USRPRF(*USER) AUT(*EXCLUDE) */
/* CRTCLPGM   PGM(&O/&ZE) SRCFILE(&L/&F) SRCMBR(&ZN) +
              USRPRF(*USER) REPLACE(*YES) AUT(*EXCLUDE) */
/*****************************************************/
/* FILE compilers                                    */
/*****************************************************/
/* CRTPF      FILE(&O/&ZE) SRCFILE(&L/&F) SRCMBR(&ZN) +
              AUT(*EXCLUDE) */
/* CRTLF      FILE(&O/&ZE) SRCFILE(&L/&F) SRCMBR(&ZN) +
              AUT(*EXCLUDE) */
/* CRTDSPF    FILE(&O/&ZE) SRCFILE(&L/&F) SRCMBR(&ZN) +
              AUT(*EXCLUDE) REPLACE(*YES) */
/* CRTPRTF    FILE(&O/&ZE) SRCFILE(&L/&F) SRCMBR(&ZN) +
              AUT(*EXCLUDE) REPLACE(*YES) */
/* CRTICFF    FILE(&O/&ZE) SRCFILE(&L/&F) SRCMBR(&ZN) +
              AUT(*EXCLUDE) REPLACE(*YES) */
/*****************************************************/
/* SQL preprocessors                                 */
/*****************************************************/
/* CRTSQLRPG  PGM(&O/&ZE) SRCFILE(&L/&F) SRCMBR(&ZN) +
              OPTION(*GEN) REPLACE(*YES) */
/* CRTSQLCBL  PGM(&O/&ZE) SRCFILE(&L/&F) SRCMBR(&ZN) +
              OPTION(*GEN) REPLACE(*YES) */
/*****************************************************/
/* CLD/CMD compilers                                 */
/*****************************************************/
/* CRTCLD     CLD(&O/&ZE) SRCFILE(&L/&F) SRCMBR(&ZN) +
              AUT(*EXCLUDE) REPLACE(*YES) */
/* CRTCMD     CMD(&O/&ZE) PGM(&O/&ZE) SRCFILE(&L/&F) +
              SRCMBR(&ZN) AUT(*EXCLUDE) REPLACE(*YES) */
/*****************************************************/
/* ADDPFM/ADDLFM                                     */
/*****************************************************/
/* ADDPFM FILE(&O/&ZE) MBR(MBRNAME) */
/* ADDLFM FILE(&O/&ZE) MBR(MBRNAME) */
/*****************************************************/
/* UIM                                               */
/*****************************************************/
/* CRTPNLGRP  PNLGRP(&O/&ZE) SRCFILE(&L/&F) SRCMBR(&ZN) +
              AUT(*EXCLUDE) REPLACE(*YES) */
/* CRTMNU     MENU(&O/&ZE) TYPE(*UIM) SRCFILE(&L/&F) +
              SRCMBR(&ZN) AUT(*EXCLUDE) REPLACE(*YES) */
/*****************************************************/
/* ILE compilers and preprocessors                   */
/*****************************************************/
/* CRTPGM     PGM(&O/&ZE) MODULE(&L/&ZN) REPLACE(*YES)   +
              USRPRF(*USER) AUT(*EXCLUDE) */
/* CRTSRVPGM  SRVPGM(&O/&ZE) SRCFILE(&L/&F) SRCMBR(&ZN)  +
              REPLACE(*YES) AUT(*EXCLUDE) */
/* CRTCMOD    MODULE(&O/&ZE) SRCFILE(&L/&F) SRCMBR(&ZN)  +
              REPLACE(*YES) AUT(*EXCLUDE)              +
              OPTION(*SYSINCPATH) */
```

图 52. 系统提供的构建选项部件 (2/4)

```
/* CRTSQLCI   OBJ(&O/&ZE) SRCFILE(&L/&F) SRCMBR(&ZN)     +
                OBJTYPE(*MODULE) OPTION(*GEN) REPLACE(*YES)*/
/* CRTBNDC    PGM(&O/&ZE) SRCFILE(&L/&F) SRCMBR(&ZN)     +
                REPLACE(*YES) USRPRF(*USER) AUT(*EXCLUDE)  +
                OPTION(*SYSINCPATH) */
/* CRTRPGMOD  MODULE(&O/&ZE) SRCFILE(&L/&F) SRCMBR(&ZN)  +
                REPLACE(*YES) AUT(*EXCLUDE)                */
/* CRTSQLRPGI OBJ(&O/&ZE) SRCFILE(&L/&F) SRCMBR(&ZN)     +
                OBJTYPE(*MODULE) OPTION(*GEN) REPLACE(*YES)*/
/* CRTBNDRPG  PGM(&O/&ZE) SRCFILE(&L/&F) SRCMBR(&ZN)     +
                REPLACE(*YES) USRPRF(*USER) AUT(*EXCLUDE)  */
/* CRTCBLMOD  MODULE(&O/&ZE) SRCFILE(&L/&F) SRCMBR(&ZN)  +
                REPLACE(*YES) AUT(*EXCLUDE)                */
/* CRTSQLCBLI OBJ(&O/&ZE) SRCFILE(&L/&F) SRCMBR(&ZN)     +
                OBJTYPE(*MODULE) OPTION(*GEN) REPLACE(*YES)*/
/* CRTBNDCBL  PGM(&O/&ZE) SRCFILE(&L/&F) SRCMBR(&ZN)     +
                REPLACE(*YES) USRPRF(*USER) AUT(*EXCLUDE)  */
/* CRTCLMOD   MODULE(&O/&ZE) SRCFILE(&L/&F) SRCMBR(&ZN)  +
                REPLACE(*YES) AUT(*EXCLUDE)                */
/* CRTBNDCL   PGM(&O/&ZE) SRCFILE(&L/&F) SRCMBR(&ZN)     +
                REPLACE(*YES) USRPRF(*USER) AUT(*EXCLUDE)  */
/*******************************************************/
/* CoOperative Development Environment/400 Compilers   */
/*******************************************************/
/* CRTRPGPGM  PGM(&O/&ZE) SRCFILE(&L/&F) SRCMBR(&ZN) +
                REPLACE(*YES) USRPRF(*USER) AUT(*EXCLUDE) +
                OPTION(*SRCDBG) */
/* CRTSQLRPG  PGM(&O/&ZE) SRCFILE(&L/&F) SRCMBR(&ZN) +
                OPTION(*GEN *LSTDBG) +
                REPLACE(*YES) */
/* CRTCBLPGM  PGM(&O/&ZE) SRCFILE(&L/&F) SRCMBR(&ZN) +
                REPLACE(*YES) USRPRF(*USER) AUT(*EXCLUDE) +
                OPTION(*SRCDBG) */
/* CRTSQLCBL  PGM(&O/&ZE) SRCFILE(&L/&F) SRCMBR(&ZN) +
                OPTION(*GEN *LSTDBG) REPLACE(*YES) */
/* CRTCLPGM   PGM(&O/&ZE) SRCFILE(&L/&F) SRCMBR(&ZN) +
                USRPRF(*USER) REPLACE(*YES) AUT(*EXCLUDE) +
                OPTION(*SRCDBG) */
/* CRTCMOD    MODULE(&O/&ZE) SRCFILE(&L/&F) SRCMBR(&ZN) +
                OPTION(*SYSINCPATH *EVENTF) +
                REPLACE(*YES) AUT(*EXCLUDE) +
                DBGVIEW(*SOURCE) */
/* CRTSQLCI   OBJ(&O/&ZE) SRCFILE(&L/&F) SRCMBR(&ZN)     +
                OBJTYPE(*MODULE) OPTION(*GEN *EVENTF) +
                REPLACE(*YES) DBGVIEW(*SOURCE) */
/* CRTBNDC    PGM(&O/&ZE) SRCFILE(&L/&F) SRCMBR(&ZN)     +
                REPLACE(*YES) USRPRF(*USER) AUT(*EXCLUDE)  +
                OPTION(*SYSINCPATH *EVENTF) DBGVIEW(*SOURCE) */
/* CRTRPGMOD  MODULE(&O/&ZE) SRCFILE(&L/&F) SRCMBR(&ZN)  +
                REPLACE(*YES) AUT(*EXCLUDE) +
                OPTION(*EVENTF) DBGVIEW(*SOURCE) */
/* CRTSQLRPGI OBJ(&O/&ZE) SRCFILE(&L/&F) SRCMBR(&ZN)     +
                OBJTYPE(*MODULE) OPTION(*GEN *EVENTF) +
                REPLACE(*YES) DBGVIEW(*SOURCE) */
/* CRTBNDRPG  PGM(&O/&ZE) SRCFILE(&L/&F) SRCMBR(&ZN)     +
                REPLACE(*YES) USRPRF(*USER) AUT(*EXCLUDE) +
                OPTION(*EVENTF) DBGVIEW(*SOURCE) */
```

图 52. 系统提供的构建选项部件 (3/4)

```
/* CRTCBLMOD  MODULE(&O/&ZE) SRCFILE(&L/&F) SRCMBR(&ZN)  +
                REPLACE(*YES) AUT(*EXCLUDE) +
                OPTION(*EVENTF) DBGVIEW(*SOURCE) */
/* CRTSQLCBLI OBJ(&O/&ZE) SRCFILE(&L/&F) SRCMBR(&ZN)     +
                OBJTYPE(*MODULE) OPTION(*GEN *EVENTF) +
                REPLACE(*YES) DBGVIEW(*SOURCE) */
/* CRTBNDCBL  PGM(&O/&ZE) SRCFILE(&L/&F) SRCMBR(&ZN)     +
                REPLACE(*YES) USRPRF(*USER) AUT(*EXCLUDE) +
                OPTION(*EVENTF) DBGVIEW(*SOURCE) */
/* CRTCLMOD   MODULE(&O/&ZE) SRCFILE(&L/&F) SRCMBR(&ZN)  +
                REPLACE(*YES) AUT(*EXCLUDE) +
                OPTION(*EVENTF) DBGVIEW(*SOURCE) */
/* CRTBNDCL   PGM(&O/&ZE) SRCFILE(&L/&F) SRCMBR(&ZN)     +
                REPLACE(*YES) USRPRF(*USER) AUT(*EXCLUDE) +
                OPTION(*EVENTF) DBGVIEW(*SOURCE) */
```

图 52. 系统提供的构建选项部件 (4/4)

注: CRTBNDC 和 CRTCMOD 命令上的*编译器选项*参数 (OPTION) 设置为值 *SYSINCPATH，因此，当您构建包含部件时，将构建该包含在搜索路径中的最低版本。注意， *SYSINCPATH 不是此参数的缺省值。缺省值为 *USRINCPATH。

构建选项部件中的替换变量 解析为下列各项:

&O 存放构建过程的输出的库的名称。这是 BLDPART 命令上指定的项目和组的库。永远不应更改 &O。

&ZE 正在创建的输出部件的名称。如果先前未构建源，则这是与源部件名相同的名称。必要时可以更改 &ZE。

&ZE 替换变量在构建时进行解析。可以将 &ZE 变量替换为下列任何一项。
*PGMID，仅用于 COBOL/400
*CTLSPEC，仅用于 RPG/400
名称

如果有可能的话，文件、模块和服务程序应当与创建它们的源部件同名。

如果源部件现在编译到同一输出组中名为 CLD B 的输出部件中，则 BLDPART 命令会自动删除输出部件 CLD A。

&L 包含源部件的组的库的名称。例如，可以发出以下命令:

```
BLDPART PRJ(PAYROLL) GRP(DEVELOPER1) TYPE(RPGSRC) PART(BIWKLY)
```

如果部件 RPGSRC BIWKLY 不在组 DEVELOPER1 中，而在名为 TEST 的组中，该组在项目层次结构中位于更高的一级，则 &L 解析为 TEST 组的库。这与 &O（存放构建输出的库）形成对照。永远不应更改 &L。

&F 包含正在编译的源部件的文件的名称。永远不应更改 &F。

&ZN 正在编译的源部件的名称。除了有可能对 CRTCMD 中的 PGM 更改 &ZN 之外，永远不应更改它。

要获取在“应用程序开发管理器”功能部件中可用的替换变量的完整列表，参见附录 D. 替换变量。

将 BLDOPT 部件用于 CODE/400

如果已安装 CoOperative Development Environment/400 (CODE/400) 产品，使用 CODE/400 构建“应用程序开发管理器”部件并想在“应用程序开发管理器”构建后将错误反馈给 CODE/400，则执行下列步骤:

1. 创建类型为 BLDOPT 的“应用程序开发管理器”部件，并使其具有第138页的『创建构建选项部件』中讨论的适当名称。
2. 确保此部件在搜索路径中。
3. 从系统提供的 BLDOPT 部件的 CODE/400 部分中除去对适当编译器命令的注释，如第140页的图52所示。

如何处理类型为 BLDOPT 的部件

因为您没有在 BLDPART 命令上指定要使用的 BLDOPT 部件，所以构建命令使用了设置的搜索次序。处理 BLDOPT 部件的搜索次序是:

1. 构建过程搜索 BLDOPT *part-name*（其中，*part-name* 是正在构建的部件的名称）并使用在其中找到的选项。例如，如果正在构建 RPGSRC BIWKLY，则构建过程搜索 BLDOPT BIWKLY。
2. 如果找不到 BLDOPT *part-name*，则构建过程搜索 BLDOPT *compiler-name*（其中，*compiler-name* 是要用来构建正在构建的部件的缺省编译器命令的名称）并使用在其中找到的选项。查看第122页的表12，以了解缺省编译器命令的名称。注意，如果有多个编译器命令对正在构建的部件有效，则构建过程只识别具有缺省编译器命令名的 BLDOPT 部件。然而，您可以从其他编译器命令中除去注释，以使构建过程使用它。例如，如果正在构建 RPGLESRC 部件，则 CRTBNDRPG 和 CRTRPGMOD 编译器命令都有效。然而，构建过程将只识别 BLDOPT CRTBNDRPG。您可以创建 BLDOPT CRTBNDRPG 部件，但从其中的 CRTRPGMOD 命令中除去注释。
3. 如果找不到 BLDOPT *compiler-name*，则构建过程搜索部件 BLDOPT QDFT。此部件便是缺省的构建选项部件。
4. 如果找不到 BLDOPT QDFT，则构建过程对正在使用的编译器命令使用缺省编译器选项。例如，如果正在编译的源部件类型是 DDSSRC-PF，则 BLDPART 命令按以上列表中描述的次序在 BLDOPT 部件中查找 CRTPF 命令。如果找不到，则使用缺省编译命令。如果 BLDOPT 部件中有许多 CRTPF 命令，则使用第一个。

这表示定义构建处理的人员必须了解项目层次结构中存在的构建选项部件，并确保执行正确的构建处理。需要记住的关于 BLDOPT 部件的其他一些要点包括:

- 搜索路径中的共享项目中的 BLDOPT 部件被忽略。
- 在 CRTPGM 命令的 ENTMOD 参数上指定的模块必须与在 CRTxxxMOD（其中，xxx 是编译器语言，如 C、CL、CBL 和 RPG）命令上指定的模块相同。

  如果正在使用与 MODULE 部件同名的 BLDOPT 部件，则必须在 CRTPGM 命令的 ENTMOD 参数上指定该名称。
- 在 CRTPGM 和 CRTSRVPGM 命令上的 MODULE 参数上，不允许值 *ALL 或 *generic**。
- 如果从部件 BLDOPT QDFT 中使用了多个编译器命令，且此部件已更改，这将导致重新构建以前使用 BLDOPT QDFT 部件中的任何命令构建的所有部件。仅当要更改 BLDPART 命令所使用的缺省编译器命令时，才使用部件 BLDOPT QDFT。您可能想使用 BLDOPT *compiler-name*。

- 在处理 BLDOPT 部件时，除非在 BLDOPT 部件中的任何地方找到具有与其部件类型相同的标号的命令，否则 BLDPART 命令会对它正在处理的部件的类型运行它找到的第一条命令。例如，当处理 BLDOPT 部件来构建 CSRC 部件时，BLDPART 命令运行它找到的第一条 CRTBNDC 或 CRTCMOD 命令。如果 CRTBNDC 和 CRTCMOD 命令都找不到，则将缺省的编译命令 CRTCMOD 用于类型为 CSRC 的部件。

BLDPART 命令只使用类型为 BLDOPT，称为 QDFT，或具有缺省编译器命令的名称，或与某部件同名的部件。例如，如果创建部件 BLDOPT MASTER，并且项目层次结构中没有名为 MASTER 的源部件，则永远不会使用此构建选项部件。

如果正在构建类型为 MENU 的部件，则 BLDPART 命令在构建选项部件中搜索 CRTMNU 命令。必须将该部件定义为 CRTMNU TYPE(*UIM)。

项目层次结构中有多个构建选项部件

图53中显示的项目层次结构带有两个类型为 RPGSRC 的部件和三个类型为 BLDOPT 的部件。每个源部件都有与其相关联的构建选项部件。除了特定于部件的构建选项部件之外，有一个名为 QDFT 的缺省构建选项部件。

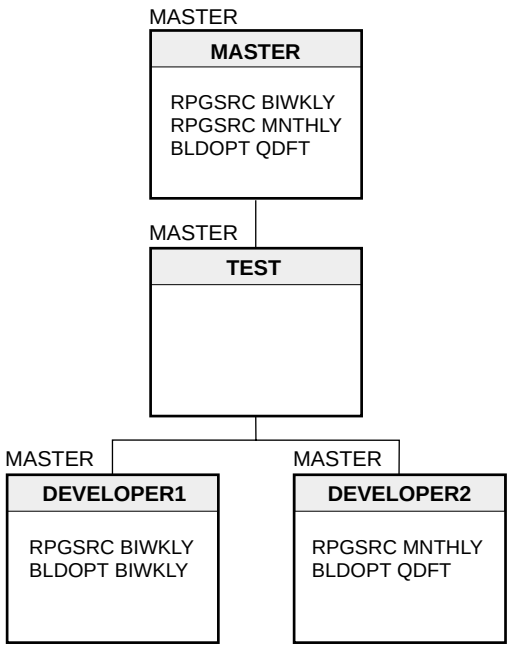

图 53. 带有类型为 BLDOPT 的部件的 Payroll 项目

项目管理员在每个组中使用以下 BLDPART 命令。

```
BLDPART PRJ(PAYROLL) GRP(group-name) TYPE(RPGSRC) PART(*ALL)
```

表13标识用来处理两个 RPGSRC 部件的构建选项部件。

表 13. 用来处理两个 RPGSRC 部件的构建选项部件

组	部件	使用的 BLDOPT 部件
MASTER	RPGSRC BIWKLY	PAYROLL MASTER BLDOPT QDFT

表 13. 用来处理两个 RPGSRC 部件的构建选项部件 (续)

组	部件	使用的 BLDOPT 部件
MASTER	RPGSRC MNTHLY	PAYROLL MASTER BLDOPT QDFT
TEST	RPGSRC BIWKLY	PAYROLL MASTER BLDOPT QDFT
TEST	RPGSRC MNTHLY	PAYROLL MASTER BLDOPT QDFT
DEVELOPER1	RPGSRC BIWKLY	PAYROLL DEVELOPER1 BLDOPT BIWKLY
DEVELOPER1	RPGSRC MNTHLY	PAYROLL MASTER BLDOPT QDFT
DEVELOPER2	RPGSRC BIWKLY	PAYROLL DEVELOPER2 BLDOPT QDFT
DEVELOPER2	RPGSRC MNTHLY	PAYROLL DEVELOPER2 BLDOPT QDFT

注意，在每个类型为 BLDOPT 的部件中，有关如何处理 CRTRPGPGM 命令的规范会有所不同。搜索路径以您在其中执行构建的组开始。如果在那里找到与正在构建的部件相匹配的类型为 BLDOPT 的部件，则使用那个 BLDOPT 部件。

要确保各个开发组的一致性，项目管理员可以在每个开发组中创建一个带有提升码 *NONE 的 BLDOPT QDFT 部件。同样，可以让 BLDOPT QDFT 的不同版本驻留在 TEST 和 MASTER 组中，以确保各个测试或主控级别中的构建操作一致。图54提供了此操作的示例。

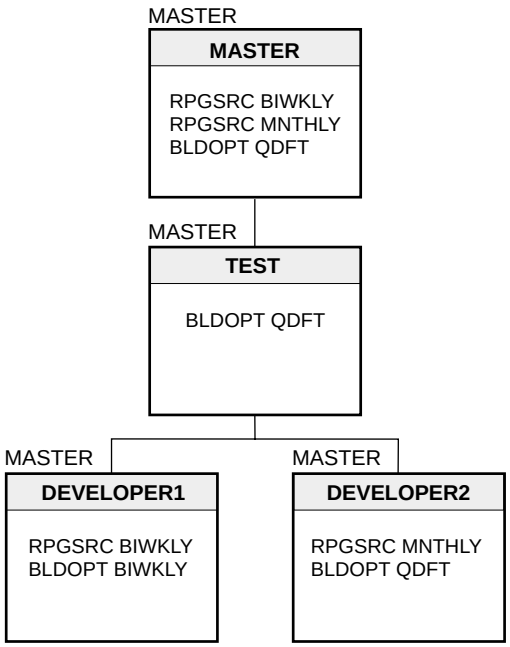

图 54. 带有类型为 BLDOPT 的部件的 Payroll 项目

在此图中，共有三个版本的 BLDOPT QDFT 部件，每一个都指示在项目层次结构中的每个级别所必需的处理信息。

如果开发者需要更改处理信息，他或她应创建一个特定于部件的 BLDOPT 部件或更改他或她的 QDFT 构建选项部件副本。不应将已更改的 QDFT 部件或 *part-name* BLDOPT 部件提升至 TEST 组。

用于物理和逻辑文件的特殊构建选项命令

通过使用“创建逻辑文件”(CRTLF) 和“创建物理文件”(CRTPF) 命令，或通过“添加逻辑文件成员”(ADDLFM) 和“添加物理文件成员”(ADDPFM) 命令，可以向物理和逻辑文件添加一个或多个成员。构建过程在 BLDOPT 部件中搜索这些命令，并在成功地编译物理或逻辑文件后运行 ADDPFM 或 ADDLFM 命令。例如，要让构建过程创建带有三个成员的物理文件（这三个成员简单地名为 A、B 和 C），则可以创建 BLDOPT 部件来包含下列命令:

```
CRTPF FILE(&O/&ZE) SRCMBR(&ZN) MBR(A) MAXMBRS(3) ADDPFM FILE(&O/&ZE) MBR(B)
ADDPFM FILE(&O/&ZE) MBR(C)
```

共有五种可能的方法来在 BLDOPT 部件中定义这些影响 BLDOPT 部件的处理方式的命令，如如何处理类型为 BLDOPT 的部件所述。以下列表描述指定 CRTPF 和 ADDPFM 命令的方法。以相同的方式指定 CRTLF 和 ADDLFM 命令:

1. 不在 BLDOPT 部件中定义 ADDPFM 命令。不在 BLDOPT 部件中定义 CRTPF 命令。

   使用 CRTPF 命令的缺省编译选项。
2. 不定义 ADDPFM 命令。定义 CRTPF 命令。

   只使用 CRTPF 命令。
3. 定义一个或多个 ADDPFM 命令。不定义 CRTPF 命令。

   构建过程使用 CRTPF 命令的缺省编译选项，并且如果部件编译成功，则在执行 CRTPF 命令后执行 ADDPFM 命令。
4. 定义一个或多个 ADDPFM 命令。定义 CRTPF 命令。

   构建过程使用 CRTPF 命令，并且如果部件编译成功，则在执行 CRTPF 命令后执行 ADDPFM 命令。如果 CRTPF 和 ADDPFM 命令都已定义，则它们必须存在于同一个 BLDOPT 部件中。
5. 将命令定义为带有标号 DDSSRC。

指定 OVRDBF 和 DLTOVR 命令

在 BLDOPT 部件中，可以指定要在构建过程期间发出的一个或多个“覆盖数据库文件”(OVRDBF) 命令和一个或多个“删除覆盖”(DLTOVR) 命令。这些命令允许您设置特定环境或在编译部件之前和之后执行特定功能。

您想让 BLDPART 命令运行的 OVRDBF 和 DLTOVR 命令必须与编译命令在同一个 BLDOPT 部件中。在编译源部件之前，会立即运行部件中的所有 OVRDBF 命令，并在编译源部件之后立即运行所有 DLTOVR 命令。DLTOVR 命令在运行 ADDPFM 和 ADDLFM 命令之前运行（如果指定了 ADDPFM 和 ADDLFM 的话）。

DLTOVR 命令用来关闭 OVRDBF 命令。如果命令不匹配，则其他部件的构建可能会受到影响。

这些命令可以包含替换变量。构建过程在运行命令之前解析变量。参考附录D. 替换变量以获取替换变量的列表。

如果任何 OVRDBF 命令失败，则不运行其余尚未运行的 OVRDBF 命令。该部件被认为已失败。无论是否有任何 OVRDBF 命令失败或源部件是否编译失败，都会运行 DLTOVR 命令。

用于特定部件类型的特殊构建处理

下列各节描述有关使用部件列表部件来构建物理和逻辑文件、MODULE、MENU 和 DB2 OS/400 版部件的特殊注意事项。

使用部件列表部件构建部件

在将部件列表部件与 BLDPART 命令配合使用时，以如下方式处理该部件列表部件:

1. 使用 BLDPART 命令上指定的搜索路径来搜索该部件列表部件。
2. 根据您在 SCOPE 参数上指定的值来构建该列表中的每个部件和任何其他部件。如果找不到该部件列表部件，则 BLDPART 命令失败。

构建物理和逻辑文件

当您构建逻辑文件时，该逻辑文件所基于的物理文件必须存在于同一组中。如果该物理文件不在同一组中而在搜索路径中，则 BLDPART 命令会在编译该逻辑文件之前将该物理文件复制到构建输出组中。如果该物理文件不在搜索路径中，则无法构建逻辑文件。

当要构建物理文件时，相关联的逻辑文件必须在构建范围中，否则不会构建该物理文件。

在调用编译器之前，BLDPART 命令自动删除物理或逻辑文件的对象。如果编译未能重新创建对象，则还会删除输出部件。

保存物理文件数据

如果“创建项目”(CRTPRJ) 命令上的 SAVDTA 参数设置为 *YES，则在重新创建物理文件之前，会保存其中的数据。在编译完成之后，无论编译成功与否，都会返回这些数据。这确保即使物理文件的编译未能成功地完成，也不会丢失数据。

如果不能将数据复制回文件中，您也可以找到以前包含在物理文件中的数据并将其复制回物理文件中。可以在输出组中的名为 QDTAxxxxxx（其中，xxxxxx 被替换为一串数字）的部件中找到物理文件中的数据。您应当将 QDTAxxxxxx *FILE 复制回该物理文件部件中并重新构建该物理文件部件。QDTAxxxxxx 部件的文本描述字段包含以前保存的物理文件的部件名。

类型为 MODULE 的部件和构建过程

仅当构建 MODULE 部件成功，且与该 MODULE 部件同名的 BLDOPT 部件包含 CRTPGM 命令时，构建过程才会执行 CRTPGM 命令。BINDSTEP 参数上指定的值必须是 *YES。

如果正在构建一个模块，则必须要有与模块部件同名的 BLDOPT 部件，且必须指定 BINDSTEP(*YES)，或使用 CRTBNDxxx（其中，xxx 是语言，如 C、CL、CBL 和 RPG）。要将几个模块绑定在一起，必须存在与作为入口点模块的模块部件同名的，包含 CRTPGM 的 BLDOPT 部件。

如果正在将几个部件一起绑定到 PGM 或 SRVPGM 中，则有几种不同的方法可引用模块:

1. 在 BLDOPT 部件中的 CRTPGM 或 CRTSRVPGM 命令的 MODULE 参数上指定模块。

2. 引用包含被引用模块的服务程序（类型为 SRVPGM 的部件）。
3. 通过类型为 BNDDIR 的绑定目录部件引用模块。

如果要构建 PGM 与 MODULE 部件之间的关系，则建议您使用上面提到的方法 1 和 2。这些方法提供了正确的构建关系信息，并且，如果任何模块发生了更改，会重新构建 PGM 或 SRVPGM。

对于服务程序:

1. 创建与服务程序和 BLDOPT 部件同名的绑定源 (BNDSRC) 部件。可以使用 RTVBNDSRC 命令来生成绑定程序语言，然后将此源导入到 ADM 项目中。
2. 取消 BLDOPT 部件中对 CRTSRVPGM 命令的注释。应将 CRTSRVPGM 命令的导出参数设置为 *SRCFILE（缺省值）。

如果使用 BINDSTEP(*YES) 成功地编译了 ILE 部件，则会创建类型为 MODULE 和 PGM 的部件。现在，如果您更改 BLDOPT 部件以更改 MODULE 部件的名称并重新构建 ILE 源部件，则会创建一个新的类型为 MODULE 的部件。原始的 MODULE 部件被认为是孤立的对象，将被删除。注意，要将新的 MODULE 部件构建到 PGM 中，必须要有与包含 CRTPGM 命令的新 MODULE 部件同名的 BLDOPT 部件。

类型为 CSRC、CINC 和 PGM 且语言为 C 和 SQLC 的部件和构建过程

构建部件不会构建类型为 CSRC、CINC 和 PGM 并带有 C 和 SQLC 语言的部件。

为了保持数据完整性，不会将这些部件自动转换为语言 CLE 或 SQLCLE。为了将语言更改为 CLE 或 SQLCLE，您可以使用 CHGPARTINF 命令。

类型为 MENU 的部件和构建过程

当您使用 CRTMNU 命令构建类型为 PNLSRC（语言为 MENU）的部件时，会创建类型为 MENU 的部件。如果您使用构建选项部件，则必须确保它包含 CRTMNU 命令，且 CRTMNU 命令上的 TYPE 参数包含值 *UIM。

DB2 OS/400 版部件和构建过程

DB2 OS/400 版部件是源部件，它们包含 DB2 OS/400 版语句并具有 DB2 语言属性;例如，具有语言属性 SQLRPG 的 RPGSRC 部件。下列源部件具有 SQL 语言属性:

表 14. 应用程序开发管理器源类型和语言

源类型	语言
CBLINC、CBLSRC	SQLCBL
CBLLEINC、CBLLESRC	SQLCBLLE
CINC、CSRC	SQLC[1]、SQLCLE
RPGINC、RPGSRC	SQLRPG
RPGLEINC、RPGLESRC	SQLRPGLE

注意:

1. 带有 SQLC 语言的类型为 CSRC 和 CINC 的源部件是不可构建的。

CRTSQLRPG、CRTSQLCBL、CRTSQLCI、CRTSQLRPGI 和 CRTSQLCBLI 命令的 OPTIONS 参数可以设置为 *GEN（缺省值）或 *NOGEN。“应用程序开发管理器”功能部件只支持 *GEN 选项。当您指定 *GEN 时，将创建 DB2 OS/400 版临时源文件成员，预处理器会调用 CRTRPGPGM、CRTCBLPGM、CRTBNDC、CRTCMOD、CRTBNDRPG、CRTRPGMOD、CRTBNDCBL、CRTCBLMOD 和 CRTSRVPGM 命令来创建程序。

如果某个 DB2 OS/400 版临时源文件成员是作为使用 *NOGEN 选项的结果而创建的唯一对象，则构建过程会考虑构建部件（如果该部件的编译成功的话）。如果您使用 DB2 OS/400 版语句引用文件，则 BLDPART 命令不能自动地构建这些文件，这是因为 DB2 OS/400 版语言不需要这些文件就能成功地对源代码进行预处理。

如果某个 DB2 OS/400 版包含部件引用文件，则使用该包含部件的源部件也可以引用该文件。在这种情况下，便存在从该包含部件到该文件以及从源部件到该文件的外部引用关系。

对每一个正在编译的部件显示的消息

本节说明在运行 BLDPART 命令时，同一消息为何会显示多次。仅当不是以批处理方式运行该命令时，才会显示这些消息。

逐个地构建部件，直到构建完所有部件为止。屏幕不断地更新，显示与正在编译的每个部件相对应的消息。一些消息的示例是:

```
正在编译 CSRC BIWKLY...
正在绑定 BIWKLY...
```

当构建过程从编译器了解到新部件关系时，它可能必须要向后跟踪以构建相关性并再次重新编译部件。因此，同一消息可能会显示多次。当一个部件更改为涉及到早先在构建过程中未涉及的另一部件时，必须进行**向后跟踪**。

构建样本应用程序

本节描述一系列用于表15中的名为 SAMPLE 的虚构应用程序的构建和提升命令。

表 15. SAMPLE 应用程序的部件

组	部件	部件类型
MASTER	REFMST	DDSSRC
MASTER	CTLFIL	DDSSRC
MASTER	EMPMST	DDSSRC
MASTER	RSNMST	DDSSRC
MASTER	TRWEEK	DDSSRC
MASTER	TRWEEKL	DDSSRC
MASTER	PRG06RP	DDSSRC
MASTER	MSGS	MSGF
TEST	PRG03FM	DDSSRC
TEST	PROC3	CLPSRC
TEST	PRG03	RPGSRC
TEST	PRG03A	RPGINC

应用程序的部件驻留在组中，如图55所述。

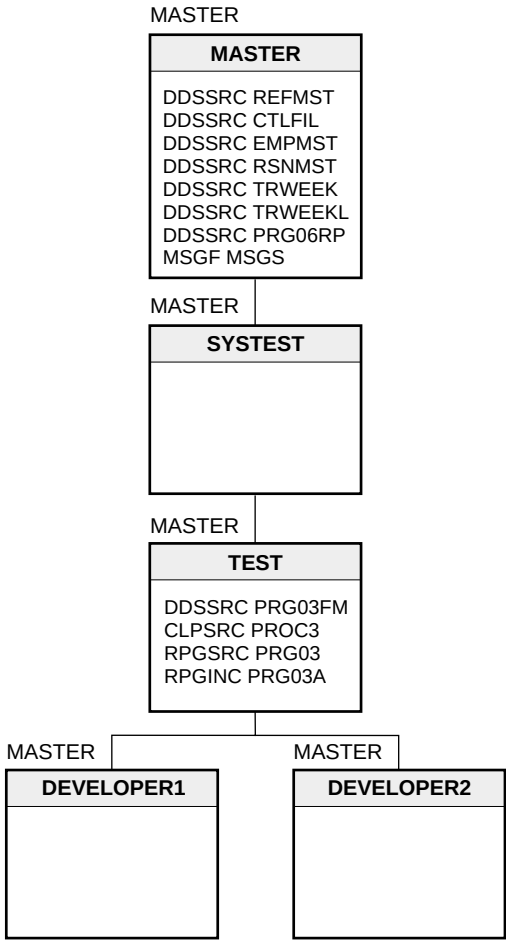

图 55. 样本应用程序中的部件

下例标识提升和构建样本应用程序中的部件需要执行的步骤:

1. 将所有部件从 TEST 组提升至 SYSTEST 组。
2. 在组 SYSTEST 中构建所有类型的所有部件。

   ```
   BLDPART PRJ(SAMPLE) GRP(SYSTEST) TYPE(*ALL) PART(*ALL)
   ```

   此构建命令使用正常构建范围以条件构建方式处理 SYSTEST 中的所有部件，并不强制处理当前部件。
3. 检查此作业的假脱机至输出队列的构建报告并确保处理部件时没有出错。如果发生了错误，则应通知对该部件负责的开发者并请求他们校正有错误的部件。可使用“打印项目记录”(PRTPRJLOG) 命令来确定上次更改该部件的人员。有关如何使用此命令的详情，参见第181页的『使用项目记录』。

   再次构建该部件，并验证更改是否起作用。
4. 运行应用程序，以查看各个部件是否能够象预期的那样合作。有关如何执行此操作的详情，参见第12章 测试和运行应用程序。
5. 检查构建报告，以确保处理部件时没有出错。

6. 运行应用程序，以验证各个部件是否能够象预期的那样合作。
7. 将所有部件提升至组 MASTER。

   ```
   PRMPART PRJ(SAMPLE) GRP(SYSTEST) TYPE(*ALL) PART(*ALL)
   ```

8. 在组 MASTER 中构建所有部件。

   ```
   BLDPART PRJ(SAMPLE) GRP(MASTER) TYPE(*ALL) PART(*ALL)
   ```

9. 检查构建报告，以确保处理部件时没有出错。
10. 运行应用程序，并验证它是否能够象预期的那样工作。如果是的话，则现在已经能够将该应用程序导出至生产环境了。有关如何执行此操作的详情，参见第13章 导出应用程序。

构建过程和用户定义类型

如果您已经创建归入下列两个类别之一中的用户定义类型，您将想要了解构建过程如何处理新类型的部件:

1. 作为配合不支持“应用程序开发管理器”功能部件的编译器使用的源成员的用户定义类型。
2. 未作为 OS/400 对象或源文件成员存储并在处理时生成一个或多个源文件成员的部件的用户定义类型。

本节描述:

- 为确保构建过程了解部件关系，您需要执行哪些操作
- 构建未存储在对象或成员中的部件
- 构建存储在源文件成员中的部件
- 如何将构建选项部件与您定义的新部件类型配合使用

确保构建过程了解部件之间的关系

要让编译器支持“应用程序开发管理器”功能部件，或者，如果您为编译器定义的新部件类型在您运行 BLDPART 命令时生成对象或源文件成员，则编译器需要在 BLDPART 命令上指定的输出组的库中创建对象或成员。(可以在 ADDADMLANG 命令的 BLDCMD 参数上指定 &L 替换变量以将此值传送至编译器。)必须使用 *System API Programming* 中描述的下列应用程序编程接口 (API):

- 获取空间状态 (QLYGETS)
- 读取构建信息 (QLYRDBI)
- 设置空间状态 (QLYSETS)
- 编写构建信息 (QLYWRTBI)

这些API能使“应用程序开发管理器”功能部件了解部件之间的关系，以便 BLDPART 命令可以正确地建立这些关系。

构建未存储在对象或成员中的部件

当您使用 ADDADMTYPE 命令添加用户定义类型时，可定义要将新类型的部件作为 SYSTYPE(*NONE) 存储的对象类型。这表示新类型的部件不存储在“应用程序开发管理器”项目中的 OS/400 对象或源文件成员中。当 BLDPART 命令运行时，会首先处理未存储在 OS/400 对象或成员中的部件。这些部件的输出是源文件成员。如果可以构建它们，则 BLDPART 命令会在处理完未存储在 OS/400 对象或成员中的部件之后处理它们。

构建过程创建的源文件成员部件的类型和语言由“应用程序开发管理器API”中的给定命令指定。

注

BLDPART 不删除孤立成员。为说明，假定 BLDPART 构建了一个未存储在对象或成员中的部件，并且构建输出是成员。如果再次调用 BLDPART 以构建同一部件，而构建输出是另一个成员，则原始构建输出变为孤立成员。BLDPART 不会除去此孤立成员。

构建存储在源文件成员中的部件

构建过程搜索对系统提供的类型（如 MSGF）的外部引用。然而，它不会搜索表示对象类型（如 *MSGF）的用户定义类型。如果以外部方式引用的对象不是系统提供的部件类型，则您可能会在构建报告中接收到一条警告消息。

如果定义了要将用户定义类型的部件作为 SYSTYPE(*MBR) 存储的对象类型，这表示部件存储在 OS/400 源文件成员中。如果这些部件是源部件，则它们由 BLDPART 命令构建，并且只能生成 *PGM 对象。如果这些部件是包含部件，则不构建它们，这是因为不能对它们定义构建命令。

如果创建的 *PGM 对象不具有第39页的表2中列示的其中一种相关语言，则需要为 *PGM 对象添加新类型，并为该类型添加新语言。

注: 如果程序对象可以有两个或多个与其相关联的部件，则第二个对象可能会覆盖第一个对象。

将构建选项部件与用户定义类型配合使用

当 BLDPART 命令构建具有用户定义类型的部件时，构建过程按以下次序程搜索用于构建部件的命令:

1. 在构建选项部件中搜索以在 ADDADMLANG 命令的 BLDCMD 参数上输入的第一个字符串记号开头的命令。
2. 如果在构建选项部件中找到相匹配的命令，则使用该命令来构建部件。（构建选项部件中的命令字符串包含的参数可以多于您在 ADDADMLANG 命令的 BLDCMD 参数上指定的命令字符串所包含的参数。）
3. 如果*找不到*相匹配的命令，则使用您在 ADDADMLANG 命令上指定的命令字符串构建该部件。

示例

下列命令对上述说明进行说明。假定您在 ADDADMLANG 命令的 BLDCMD 参数上指定了下列值。

```
BLDCMD('CRTPASPGM PGM(&L/&ZN) SRCFILE(&L/&F) SRCMBR(&ZN)')
```

然而，在构建选项部件中找到以下命令字符串。

```
CRTPASPGM PGM(&L/&ZN) SRCFILE(&L/&F) SRCMBR(&ZN)          TEXT('A sample Pascal
program')
```

将使用在构建选项部件中找到的命令字符串，这是因为它以 ADDLANG 命令的 BLDCMD 参数上指定的字符串开头。

定制的包含文件和 BLDPART 命令

如果正在使用定制的包含源文件且不想在构建期间为这些文件保存构建信息（关系），则可以将它们保存在那些名称以 'Q' 或 '#' 开头的系统库中。构建该部件时，不会保存这样的包含文件以及包含它们的任何部件的包含关系。例如，如果包含文件驻留在 QGPL 中，则不会保存与该包含文件的关系。

第12章 测试和运行应用程序

本章描述如何:
- 以*测试者*的角度创建项目层次结构
- 在“应用程序开发管理器”功能部件中测试应用程序
  - 为了进行测试而向库列表添加项目库
  - 在进行测试之后从库列表中除去项目库

如果不能在“应用程序开发管理器”功能部件中测试应用程序，则可以将该应用程序或其组件导出至库。有关如何在“应用程序开发管理器”功能部件的控制范围之外测试应用程序的详情，参见第161页的『导出应用程序以对它进行测试』。

对测试者定义带有组的项目层次结构

应用程序测试者是一个人，其主要的角色是运行应用程序并验证它的正确性。您的组织可能有专门扮演此角色的个人，也可以由应用程序开发者在应用程序开发周期中的特定点执行测试者的活动。

通常，测试者不执行部件开发活动。通常授予测试者对项目层次结构的读访问权，或指定给具有更新访问权的开发组。下面分别描述了这两种情况。

具有读访问权的测试者

授予测试者对整个项目层次结构的读访问权，但不授予对任何组的更新访问权。以下命令授予测试者对项目 PAYROLL 的权限。

```
ADDPRJUSR PRJ(PAYROLL) USRPRF(JONES) USRTYPE(*DEVELOPER) ACCESS(*READ)
```

在这种情况下，如果应用程序在运行时要更新文件，则测试者必须在项目层次结构外工作才能更新文件或对象。对于要在“应用程序开发管理器”控制范围之外测试的应用程序，必须将其导出。有关如何执行此操作的详情，参见第161页的『导出应用程序以对它进行测试』。

具有更新访问权的测试者

授予测试者对整个项目层次结构的读访问权，并向其提供一个开发组来在其中工作。测试者的开发组应该与开发者的开发组有所区别，即测试者不应该能够提升部件。下列命令在项目 PAYROLL 中创建组 TESTER1，并将 JONES 登记至新的测试组。

```
CRTGRP PRJ(PAYROLL) GRP(TESTER1) SHORTGRP(TST1) PARENT(TEST) PRMCODE(*NONE)
       CCSID(*PARENT) TEXT('JONES DEVELOPMENT GROUP - TESTER1')

ADDPRJUSR PRJ(PAYROLL) USRPRF(JONES) USRTYPE(*DEVELOPER)
          ACCESS(*UPDATE) DEVELOPGRP(TESTER1)
```

组 TESTER1 是使用提升码 *NONE 创建的，这意味着不能将其中的部件提升至另一个组。然而，JONES 可以检出部件、更改部件、复制部件和使用提升码 *NONE 创建新部件，并可以从此组中构建应用程序。

通常，测试者会创建执行测试时应用程序更新所需的文件。通常，不创建、复制或更改包含高级语言编程语句的部件。

图56中的项目层次结构说明定义带有测试者的开发组的项目层次结构的一种方法。

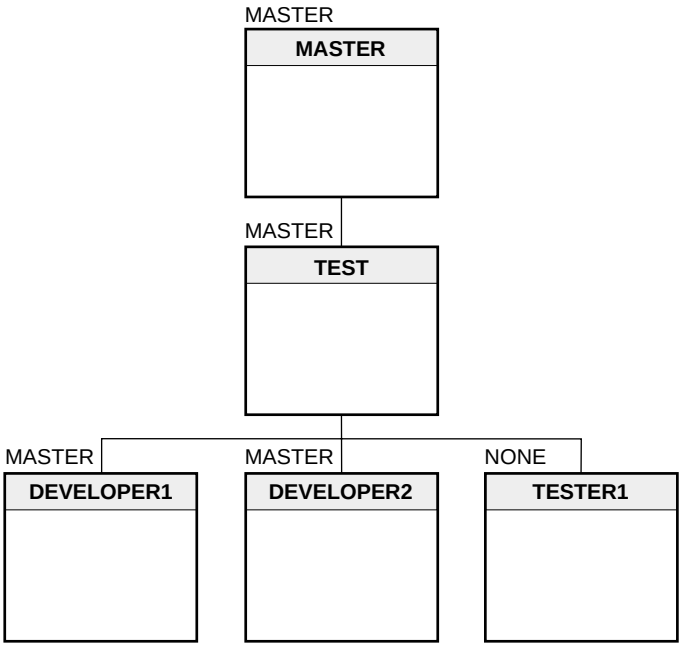

图 56. 具有测试者组的 Payroll 项目

测试者能够测试由在组 DEVELOPER1 和 DEVELOPER2 中工作的开发者提升至组 TEST 的部件。

注意，创建的并用来在组 TESTER1 中测试应用程序的数据文件对于同时正在进行的部件开发活动而言应该是最新的。如果在组 DEVELOPER1 中工作的开发者更改了部件中的记录格式，然后将部件提升至组 TEST，则可能会发生错误。还必须更新创建的用于测试此记录格式的数据文件。

为避免诸如上述之类的情况，管理员可以创建一个分阶段进行部件开发活动的项目层次结构，并向测试者提供一个更稳定的应用程序版本来供他们使用。第157页的图57显示了这样的项目层次结构。

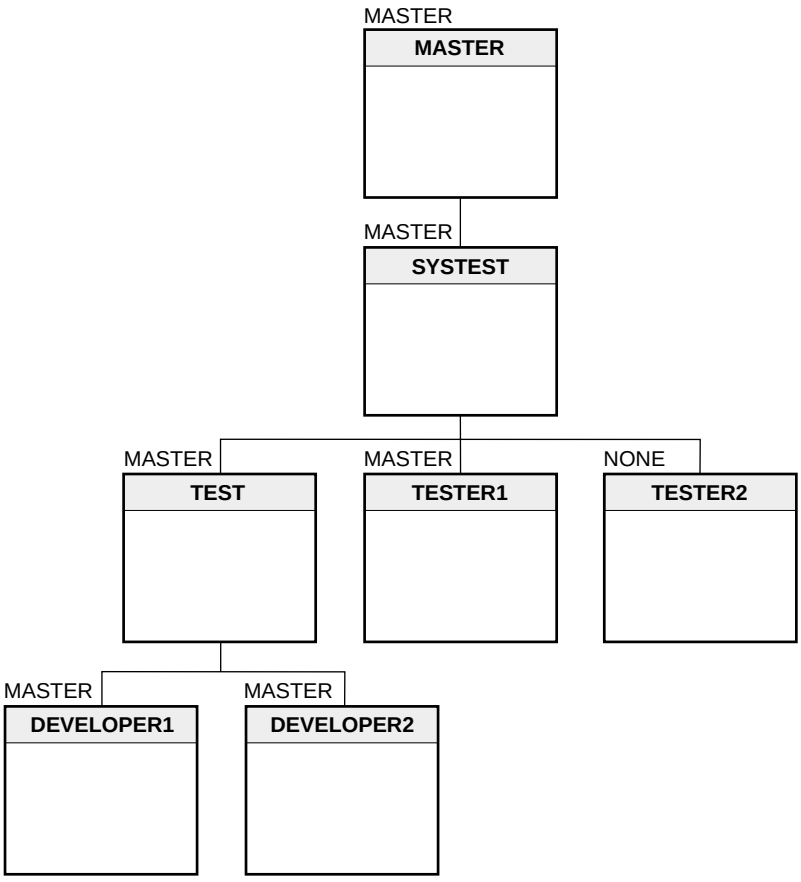

图 57. 具有测试阶段的 Payroll 项目

在此项目层次结构中，测试者在项目层次结构中的较高层工作。必须将部件从开发组提升至组 TEST。然后，管理员将部件提升至组 SYSTEST，并在该组中构建部件。这样，就可以在一致的应用程序版本中执行测试。

在项目层次结构中测试部件或应用程序

要使用“应用程序开发管理器”功能部件成功地测试部件或应用程序，必须根据部件或应用程序在生产环境中的工作方式设置基本库列表。查看第99页的『组及其与库列表的关系』，以获取组如何与库列表相对应的说明。

除非要测试部件或应用程序，否则，您不需要使用库列表。为此，您需要使用“添加项目库列表”(ADDPRJLIBL) 命令来添加所有表示指向库列表的用户部分的搜索路径的项目库。

将项目库添加至库列表

使用“添加项目库列表”(ADDPRJLIBL) 命令来将项目的搜索路径的所有项目库添加至库列表。与命令上指示的搜索路径相关联的所有库都被添加至库列表的用户部分。开发者和管理员都可以使用此命令。

必须在 PRJ 参数上指定项目名。还必须在 GRP 参数上指定组名。

SCAN 参数缺省为使用项目层次结构定义的搜索路径查找 SCHPTH 部件。此命令从 GRP 参数上指定的组开始，顺着缺省搜索路径搜索至根组。如果您选择 SCAN(*NO)，则 ADDPRJLIBL 命令只为 GRP 参数上指定的组添加库。

SCHPTH 参数缺省为 *DFT。如果缺省搜索路径中存在一个名为 SCHPTH QDFT 的部件，则使用该部件。如果 SCHPTH QDFT 不存在，则使用缺省搜索路径来确定要添加至库列表的项目库。您还可以在 SCHPTH 参数上指定特定搜索路径部件的名称。例如，可以创建并使用名为 COMMON 的 SCHPTH 部件来指示要添加表示指向库列表的交叉项目搜索路径的项目库。

如果在 SCHPTH 参数上输入搜索路径部件的名称，但指定 SCAN(*NO)，则搜索路径部件名被忽略。

示例

考虑第157页的图57中说明的项目层次结构。以下 ADDPRJLIBL 命令对部件 SCHPTH QDFT 中定义的搜索路径添加项目库。

使用 ADDPRJLIBL 命令

```
ADDPRJLIBL PRJ(PAYROLL) GRP(TESTER1) SCAN(*YES) SCHPTH(*DFT)
```

使用“编程开发管理器”实用程序

在“用 PDM 来使用组”屏幕中选择选项 45（添加项目库列表）或使用用户定义选项 AP。

部件 SCHPTH QDFT 包含下列项目和组的组合:

```
PAYROLL TESTER1
PAYROLL TEST
PAYROLL SYSTEST
PAYROLL MASTER
```

库列表更改为包括表示 SCHPTH QDFT 部件中定义的组的项目库，图58描述了新的库列表。

图 58. 新的库列表

当您使用 ADDPRJLIBL 命令时，不更改库列表的系统部分。用户部分更改为包括“应用程序开发管理器”项目库。当前库（如果已设置的话）仍是当前库。不除去已添加至库列表的其他库。然而，将项目库添加至此列表的顶部。参考下一幅图，以查看在运行 ADDPRJLIBL 命令之前和之后库列表的外观的示例。

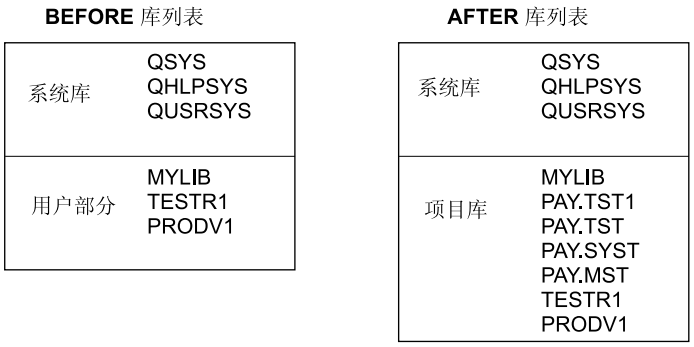

图 59. 在运行 ADDPRJLIBL 命令之前和之后的库列表

在此图中，当前库是 MYLIB。在输入 ADDPRJLIBL 命令之后，它仍是当前库。库 TESTR1 和 PRODV1 被移至库列表的底部。

当您需要测试“应用程序开发管理器”功能部件不支持的对象类型时，ADDPRJLIBL 命令非常有用。例如，在图59中，TESTR1 库中存在不受支持的对象类型。当您构建或运行部件时，部件需要访问该对象。ADDPRJLIBL 命令将项目库添加至库列表，并将用户部分中的库移至列表底部。当您构建或运行部件时，在 TESTR1 库中找到该对象。

添加项目库列表功能还会作为“选项 45”（添加项目库列表）出现在“用 PDM 来使用部件”屏幕上，或作为系统提供的用户定义选项 AP。有关详情，参见第208页的『添加和除去项目库』。

将外部库添加至库列表

可使用“添加项目库列表”(ADDPRJLIBL) 命令通过更新 SCHPTH QDFT 将一个或多个外部库添加至库列表，如下所示。为此，必须指定包含外部库的名称的搜索路径部件。

在下面的示例中，将把外部库添加至库列表。搜索路径部件包含下列信息:

```
PAYROLL TESTER1
PAYROLL TEST
PAYROLL SYSTEST
PAYROLL MASTER
*USRLIB SOFTLIB1
*USRLIB SOFTLIB2
```

当您对此搜索路径部件使用 ADDPRJLIBL 命令时，只会更改库列表的用户部分。

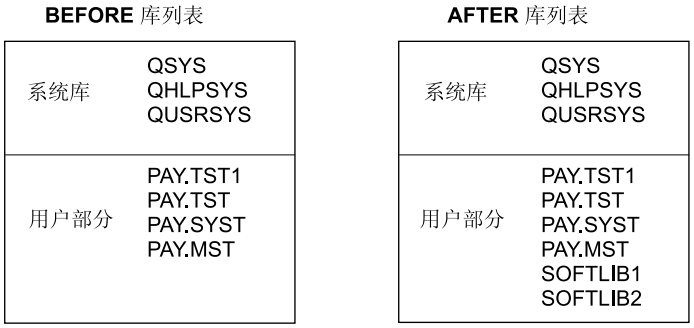

图 60. 在运行 ADDPRJLIBL 命令之前和之后的库列表。

第159页的图60显示在发出 ADDPRJLIBL 命令之后，SOFTLIB1 和 SOFTLIB2 库被添加至库列表的用户部分。它被添加至“应用程序开发管理器”项目之后，以确保构建先在“应用程序开发管理器”项目中搜索对象，然后再搜索用户库。

从库列表中除去项目库

使用“除去项目库列表”(RMVPRJLIBL) 命令来除去*所有*已使用ADDPRJLIBL 命令添加至库列表的项目库。

RMVPRJLIBL 命令上没有任何参数。只需要输入此命令的名称便可从库列表中除去所有项目库。因为 RMVPRJLIBL 命令上没有任何参数，所以不会除去外部库（这些库可能是由 ADDPRJLIBL 命令添加的）。

也可以从下列屏幕中使用除去项目库列表功能:

- 在“用 PDM 来使用组”屏幕中，作为功能键“F20=除去项目库列表”
- 在“用 PDM 来使用部件”屏幕中，作为选项 45

第13章 导出应用程序

此功能部件通过 EXPPART 命令提供**导出**功能。当您导出时，应用程序的一个或多个部件将从“应用程序开发管理器”项目层次结构复制到给定库中。

注: 只要整个项目驻留在同一备用搜索路径中，就可将“应用程序开发管理器”项目库和归档库移至备用搜索路径。

可导出应用程序的单个部件或整个版本。通常，为了关键的测试目的，开发者将导出应用程序的部件或组件，而管理员或小组长为了关键的测试目的导出整个应用程序，或将整个应用程序复制为产品。

本章描述:
- 为进行测试而导出应用程序
- 使用“导出部件”(EXPPART) 命令
- 导出应用程序后进行封装

另外，使用 EXPPART 命令来将部件导出至产品库或为了进行测试，也可使用它来将部件复制至您计划用来在另一系统上进行恢复的库。有关如何使用 EXPPART 和 IMPPART 命令来执行此操作的详情，参见第180页的『将信息复制至另一个 AS/400 系统』。

您应该花一些时间来阅读 *ADTS/400: Application Development Manager Introduction and Planning Guide* 中列示在移至生产环境之前需要考虑的事项的章节。

导出应用程序以对它进行测试

对于应用程序来说，如果出现下列任何一种情况，则可能必须将它或它的组件导出至测试库以对它进行测试:

- 如果应用程序必须覆盖文件。
- 如果需要测试某个用户是否对文件、命令或程序具有适当的访问权。
- 如果应用程序需要日志记录或如果正在测式备份过程。(日志和日志接收方是“应用程序开发管理器”功能部件不直接支持的 OS/400 对象。)
- 如果应用程序的部件需要唯一的库列表，或如果它更改库列表，则可能需要在项目层次结构之外测试。例如，如果它使用指定产品库的命令，则不能在此产品内测试它。必须导出它。
- 如果处理时，应用程序将作业提交至批处理子系统，则可从作业描述而不是从项目层次结构中得到这些作业的库列表。

使用导出部件命令

注: 交叉项目系统列表（包含驻留在多个项目中的部件）必须用 CROSSPRJ(*YES) 和 SCAN(*YES) 参数导出。而且，系统列表和搜索路径必须存在于 EXPPART 命令指定的项目中。

使用“导出部件”(EXPPART) 命令来将应用程序的部件或组件从“应用程序开发管理器”组复制至库。当您导出部件时，将在库中创建该部件的副本。使用当前权限来使用此命令。EXPPART 命令不会授予您附加权限。

在您导出至的库中，根据部件类型将部件复制为源文件成员或对象。例如，当正在导出类型为 RPGSRC 的部件时，如果库中不存在源物理文件，则将创建它，同时还将创建该源物理文件内的成员。如果源物理文件已存在，则将成员添加至其中。不替换源文件。如果物理数据文件已存在于库中，则不能替换它。可替换已存在于库中的成员和其他对象。

对于 IMPPART 命令，当您导出信息时，您需要指定*源*和*目标*标准。*源*标准指示要从项目层次结构中复制什么信息。*目标*标准指示您要将“应用程序开发管理器”信息复制至何处。

使用 EXPPART 命令来导出特定语言的部件。缺省值是 LANG(*ALL)。

EXPPART 命令的 CHGDATE 参数允许您只导出自特定日期之后更改的部件。CHGDATE(*BEGIN) 是缺省值，将导出所有部件，不管它们最后更改的日期如何。如果要导出今天更改的所有部件，指定 CHGDATE(*CURRENT)。

也可使用 CROSSPRJ 参数导出交叉项目部件。如果指定了 SCAN(*NO) 或如果交叉项目组在搜索路径中，则忽略此参数。有关详情，参见第101页的『创建交叉项目搜索路径』。为了能够从交叉项目中导出部件，您至少需要对该交叉项目具有 *READ 项目访问级别。

要导出的部件

必须在 EXPPART 命令中指定项目、组、类型和部件名。对于 TYPE 和 PART 参数，提供了一些特殊值。有关 EXPPART 命令允许的部件类型的列表，参见第227页的『附录D. 替换变量』。

在 TYPE 参数上，可输入特定部件类型。例如，TYPE(RPGSRC) 将导出类型为 RPGSRC 的部件。可通过使用特殊值 *ALL 或可使用类属名来指定要导出所有类型的部件。在第33页的『必需参数』中描述了类属的用法。您可以使用特殊值 *NONSRC 来指定要导出所有不安全的部件。如果指定 TYPE(*NONSRC)，则只导出存储在对象中的部件类型，如第52页的表5中列示的那些类型。不能导出其用户定义类型是使用 ADDADMLANG 命令的 SYSTYPE 参数的 *NONE 值定义的部件。此类型的部件已存在于“应用程序开发管理器”项目外部的库中。

在 PART 参数上，可输入特定部件名。例如，PART(BIWKLY) 导出部件 BIWKLY。指定特殊值 *ALL 以指示要导出所有部件，也可使用类属名称。第33页的『必需参数』描述了类属的使用。

示例

本示例说明如何将所有类型的所有部件从 PAYROLL 项目中的 TEST 组导出至当前库。

使用 EXPPART 命令

```
EXPPART PRJ(PAYROLL) GRP(TEST) TYPE(*ALL) PART(*ALL)
```

使用“编程开发管理器”实用程序

在“用 PDM 来使用部件”屏幕上选择选项 39（导出）。

示例

以下命令将名为 BIWKLY 的 CSRC 部件从 Payroll 项目中的组 DEVELOPER1 导出至当前库。

```
EXPPART PRJ(PAYROLL) GRP(DEVELOPER1) TYPE(CSRC) PART(BIWKLY)
```

要创建的对象或文件

IMPPART 与 EXPPART 命令之间的一个差别是，在导出操作上，*目*标标准具有缺省值。您必须在 TOLIB 和 SRCFILE 参数上指示想要存放被导出部件的位置，否则，将把它复制至缺省库和文件。如果该库中不存在文件，则会创建一个。

库的缺省值是当前库。如果不想使用缺省值，请在 TOLIB 参数上指示特定库。例如，TOLIB(TESTLIB) 使用 TESTLIB 库来存储所有已复制的部件。

库中文件名的缺省值由在项目层次结构中包含实际部件的源文件的名称确定。要使用缺省值，请指定 SRCFILE(*FROMFILE)。

当您使用 CRTPART、CPYPART 或 IMPPART 创建部件时，可以指定源文件名。如果不指定源文件名，则会在系统提供的与部件类型相匹配的源文件中创建该部件。SRCFILE 参数适用于具有第50页的表4中列示的其中一种部件类型或用户定义类型的部件。

示例

以下命令创建部件 BIWKLY 作为源物理文件 QRPGSRC 中的源成员。

```
CRTPART PRJ(PAYROLL) GRP(DEVELOPER1) TYPE(RPGSRC) PART(BIWKLY) LANG(RPG)
        SRCFILE(*TYPE)
```

如果您在创建部件时指定了源文件名，则导出该部件时，会在同名的文件中创建它。

示例

以下命令创建部件 BIWKLY 作为源物理文件 PAYSRC 中的源成员。

```
CRTPART PRJ(PAYROLL) GRP(DEVELOPER1) TYPE(RPGSRC) PART(MTHLY) LANG(RPG)
        SRCFILE(PAYSRC)
```

在导出部件 BIWKLY 时，它会复制该部件以作为当前库中的源物理文件 PAYSRC 中的源成员。

```
EXPPART PRJ(PAYROLL) GRP(DEVELOPER1) TYPE(RPGSRC) PART(BIWKLY)   TOLIB(PAY1)
SRCFILE(*FROMFILE)
```

如果不指定要将部件导出到同名的源文件中，可以指示要使用 SRCFILE(*TYPE)。此值在该类型的由系统提供的缺省源文件中创建该部件，即使项目层次结构内的部件存储在具有另一名称的文件中亦如此。查看第43页的表3，以获取缺省源文件的列表。也可以在 SRCFILE 参数上输入特定的源文件名。

示例

以下命令将 BIWKLY 部件导出至文件 PAYSRC 中的 PAY1 库。

```
EXPPART PRJ(PAYROLL) GRP(TEST) TYPE(RPGSRC) PART(BIWKLY) TOLIB(PAY1)
        SRCFILE(PAYSRC)
```

注意，如果在指定了 REPLACE(*YES) 的情况下导出已存在于*目标库*中的逻辑文件，并且已将逻辑文件的位于“应用程序开发管理器”环境外部的访问路径记入日志，则会在*目标库*中重新启动日志记录。

有关 SCAN 和 SCHPTH 参数的描述，参见第103页的『如何处理搜索路径部件』。注意，如果类型为 SCHPTH 的部件标识了交叉项目搜索路径，也不会导出其他项目中的部件。每次只能从一个项目导出。如果您想从共享项目导出部件，则必须输入另一个 EXPPART 命令并指定该项目。

随部件一起导出数据

EXPPART 命令允许您在导出应用程序时复制与物理数据文件相关联的数据。DATA 参数的缺省值是*不*复制数据。如果要复制数据，则指定 DATA(*YES)。

如果指定想要在导出操作时复制数据，且物理数据文件已存在于作为导出目标的库中，则*不替换文件*。这是为了避免错误地替换产品库中的基本数据或生产数据。

如果要替换作为导出目标的库中的物理数据文件，则需要进行人工介入，这是因为 EXPPART 命令不替换物理数据文件。您应编写一个 CL 程序，此程序保存物理文件的生产副本中的数据、删除该文件，然后调用 EXPPART 命令来导出该物理文件。作为最终的步骤，您的 CL 程序应将保存的数据恢复至新的物理文件。

如果是为了进行测试而导出，您可能想从项目层次结构中导出测试数据以继续测试。如果是导出至产品环境，大多数情况下您不会想复制测试数据。

示例

以下命令将 PAYROLL 项目中的所有部件导出至库 TESTLIB。还会复制与物理数据文件相关联的数据。

```
EXPPART PRJ(PAYROLL) GRP(TEST) TYPE(*ALL) PART(*ALL) SCAN(*YES)
        SCHPTH(*DFT) TOLIB(TESTLIB) SRCFILE(*FROMFILE) DATA(*YES)
        REPLACE(*YES)
```

导出部件时必需的授权

因为对象是在库中创建的，所以导出对象需要特定的权限。您必须具有必需的权限才能在 TOLIB 参数上指定的库中创建和删除对象和源文件成员。如果该对象或源文件成员已存在于指定的库中，而您指定了 REPLACE(*YES)，则您必须具有必需的权限才能创建和删除该对象或源文件成员。注意，不能替换物理数据文件。

除了对该库以及该库中的对象所必需的权限之外，EXPPART 命令还对创建或替换的对象指定所有者。所有者由 OWNER 参数上指定的值确定。缺省值 *TOOBJ 导致该命令使用您正在替换的对象的所有者。如果创建了该对象，则指定输入 EXPPART 命令的人员作为所有者。

在导出源成员时，现存源文件的所有权不会更改。如果 EXPPART 命令自动创建了新的源文件，则该源文件的所有者将是 OWNER 参数上指定的所有者。对此源文件的后续成员导出不会更改此源文件的所有权。

如果尝试更改现存对象的所有权，则必须具有执行此操作的权限。

通常，如果是为了进行测试而导出，则会导出至您拥有的测试库。在该库中，您应对库和对象具有所有必需的权限，并且应该已将您自己指定为所有者。

在将部件或应用程序导出至生产环境时，符合所有权限需求是非常重要的一点。如果您对需要的所有对象不具有适当的权限，则导出操作可能无法完成。

示例

以下命令将 PAYROLL 项目中的所有部件导出至测试库 TESTLIB。

```
EXPPART PRJ(PAYROLL) GRP(TEST) TYPE(*ALL) PART(*ALL) SCAN(*YES)
        SCHPTH(*DFT) TOLIB(TESTLIB) SRCFILE(*FROMFILE) DATA(*YES)
        REPLACE(*YES) OWNER(*TOOBJ) AUT(*TOOBJ)
```

假定这是您第一次导出此应用程序。您是 TESTLIB 中创建的对象的所有者。TESTLIB 中的对象的公共权限是由 AUT 参数值确定的。

示例

以下命令将组 DEVELOPER1 中的所有 RPGSRC 部件导出至 TESTLIB。

```
EXPPART PRJ(PAYROLL) GRP(DEVELOPER1) TYPE(RPGSRC) PART(*ALL) SCAN(*YES)
SCHPTH(*DFT) TOLIB(TESTLIB) SRCFILE(*FROMFILE) DATA(*NO)          REPLACE(*YES)
OWNER(*TOOBJ) AUT(*TOOBJ)
```

概括地说，EXPPART 命令不允许您对“应用程序开发管理器”环境外部您不具有权限的 OS/400 对象执行任何操作。有关对象和权限的详情，参考*iSeries Security Reference*。

导出部件列表部件

如果正在导出部件列表部件，则 EXPPART 命令允许您指定导出它的方式。缺省值 PARTLOPT(*LIST) 指示部件列表部件中列示的部件就是要导出的部件：不导出部件列表部件本身。如果指定 PARTLOPT(*PART)，则只导出部件列表部件本身：不导出部件列表部件中列示的部件。如果指定 PARTLOPT(*BOTH)，则同时导出该部件列表部件以及其中列示的部件。

EXPPART 命令按以下次序搜索要导出的部件列表部件：

1. 使用 EXPPART 命令上指定的搜索路径来搜索部件列表部件。
2. 如果在 EXPPART 命令上指定了 PARTLOPT(*LIST)，则根据您在 GRP、SCAN 和 SCHPTH 参数上指定的值搜索并导出每个部件。如果找不到部件列表部件，则 EXPPART 命令失败。如果指定了 PARTLOPT(*PART)，则以导出任何其他部件的方式导出部件列表部件本身。如果指定 PARTLOPT(*BOTH)，则同时导出该部件列表部件以及其中列示的部件。
3. 如果在部件列表部件中找到 PRDDFN 和 PRDLOD 部件，则 EXPPART 命令会执行一些额外的处理。有关 PRDDFN 和 PRDLOD 部件的信息，参见第166页的『导出和封装应用程序』。

使用系统列表部件导出

您可以使用 EXPPART 命令来将应用程序、应用程序修正或应用程序增强分布或发送至一个或多个远程系统。

系统列表部件是部件类型为 SYSTEML 的部件，它包含系统地址的列表，通过处理这些地址，可以将“应用程序开发管理器”部件发送至一个或多个远程系统。如果在 SYSTEML 参数上指定 *NONE，则 EXPPART 命令会将部件导出至同一系统上的指定库。

当指定系统列表部件名时，EXPPART 命令会将部件导出至系统列表部件中指定的远程系统。

要了解如何将应用程序分布至一个或多个远程系统，参见第169页的『第14章 将应用程序分布至远程系统』以获取更详细的信息。

当 SYSTEML 参数上指定了系统列表部件时，如果 QTEMP 库中有对象，则使用 EXPPART 命令进行的发送或分发操作将失败。

导出和封装应用程序

“应用程序开发管理器”功能部件提供了两种部件类型以使您能够使用 SystemView System Manager/400 产品封装应用程序。这两个部件类型称为**产品定义** (PRDDFN) 和**产品装入** (PRDLOD)。产品定义包含关于产品的一般信息。产品装入包含关于特定产品选件装入的信息。

我们期望您已经了解使用 SystemView® System Manager/400 产品封装应用程序所涉及的步骤。参见 *System Manager Use* 一书。

要在导出操作期间进行封装，必须能够在类型为 PARTL 的部件中找到类型为 PRDLOD 的部件。如果找到产品装入部件，则将部件列表部件中列示的部件封装为类型为 PRDLOD 的部件中指定的“创建产品装入”(CRTPRDLOD) 命令的“产品标识”参数上标识的产品。*System Manager Use* 一书中描述了 CRTPRDLOD 命令。

可将类型为 PRDDFN 和类型为 PRDLOD 的部件放在正在导出的部件列表部件中。类型为 PRDDFN 的部件必须包含有效的“创建产品定义”(CRTPRDDFN) 命令。类型为 PRDLOD 的部件必须引用在类型为 PRDDFN 的部件中指定的“创建产品定义”(CRTPRDDFN) 命令的“产品标识”参数上标识的产品。CRTPRDDFN 命令也在 *System Manager Use* 一书中作了描述。

创建产品定义和产品装入部件

要创建产品装入部件，请执行下列操作:

1. 使用 CRTPART 命令来创建该部件。在 TYPE 参数上指定 PRDLOD。
2. 使用 CHGPART 命令将 CRTPRDLOD 命令及其所有参数和值添加至 PRDLOD 部件。通过在正在更改的部件中输入此命令的名称并按“F4=提示”，可以很容易地完成此操作。出现的 CRTPRDLOD 命令参数已预先填写了缺省值。

要创建产品定义部件，请执行下列操作:

1. 使用 CRTPART 命令来创建该部件。在 TYPE 参数上指定 PRDDFN。
2. 使用 CHGPART 命令将 CRTPRDDFN 命令及其所有参数和值添加至 PRDDFN 部件。通过在正在更改的部件中输入命令的名称并按“F4=提示”，可以很容易地完成此操作。出现的 CRTPRDDFN 命令参数已预先填写了缺省值。

有关 CRTPRDLOD 和 CRTPRDDFN 命令的描述，参见 *System Manager Use* 一书。要查看这些命令的帮助，请在命令行上输入命令，按“F4=提示”以提示该命令，然后按“F1=帮助”。

创建部件列表部件以包含要封装的部件

您必须创建一个部件列表部件以包含应用程序中的所有部件的列表（为了成功地封装应用程序，需要包括这些部件）。有关如何创建和使用部件列表部件的说明，参见第91页的『第8章 部件列表部件、原因控制和更改跟踪』。您必须确保部件列表部件包含类型为 PRDLOD 的部件。它还可以包含类型为 PRDDFN 的部件。

为封装应用程序而导出部件列表部件

使用 EXPPART 命令来导出您创建的包含产品装入部件的部件列表部件。在 EXPPART 命令的 PARTLOPT 参数上指定 *LIST 以仅导出该部件列表部件中列示的部件。

EXPPART 命令认识到它已导出包含类型为 PRDLOD 的部件的部件列表。它从类型为 PRDLOD 的部件中抽取信息并使用该信息在所有其他已导出的部件中设置封装信息。如果存在类型为 PRDDFN 的部件，则它会从该部件中抽取信息并使用该信息设置产品定义信息。

注: 如果部件列表部件中列示了多个类型为 PRDLOD 的部件，则 EXPPART 命令会使用它找到的第一个此类型的部件。如果该部件列表部件只包含类型为 PRDDFN 和 PRDLOD 的部件，则封装结果是不可预测的。您应当创建一个不带 PRDDFN 和 PRDLOD 部件的部件列表部件，以使 EXPPART 命令和封装能够成功完成。

在导出所有部件之后，如果在部件列表部件中发现包含该命令的类型为 PRDDFN 的部件，则 EXPPART 命令会运行 CRTPRDDFN 命令。即使 CRTPRDDFN 命令因为 *PRDDFN 对象已存在而失败，EXPPART 命令仍会成功。

CRTPRDDFN 命令在产品定义部件中指定的 CRTPRDDFN 命令的“库”参数上指定的库中创建类型为 *PRDDFN 的对象。您应该在 CRTPRDDFN 命令的“库”参数上指定主产品库。

如果 CRTPRDDFN 命令无效，且未定义任何产品，则不执行封装。EXPPART 命令接着运行它在产品装入部件中发现的 CRTPRDLOD 命令。即使 CRTPRDLOD 命令因为 *PRDLOD 对象已存在而失败，EXPPART 命令仍会成功。

CRTPRDLOD 命令在产品装入部件中指定的 CRTPRDLOD 命令的“库”参数上指定的库中创建类型为 *PRDLOD 的对象。

现在，在 EXPPART 命令的 TOLIB 参数上指定的库包含部件列表部件中列示的所有部件的副本。

设置封装信息

EXPPART 命令按以下方式设置封装信息:

1. 对部件列表部件中列示的每一个已导出对象运行“更改产品对象描述”(CHGPRDOBJD) 命令，以更新对象描述中的产品信息。如果正在导出源文件成员中存储的部件，则对源文件运行 CHGPRDOBJD 命令。
2. 使用产品装入部件中指定的信息自动运行“封装产品选件”(PKGPRDOPT) 命令。此命令为指定产品选件封装一个或多个产品装入，从而允许使用“保存特许程序”(SAVLICPGM) 命令保存装入。

此时，使用 SAVLICPGM 命令保存构成特许程序的所有对象的副本。此命令以可使用“恢复特许程序”(RSTLICPGM) 命令进行恢复的形式保存特许程序。在使用

SAVLICPGM 命令之前，应验证产品库中是否列示了构成产品的所有对象。*System Manager Use* 一书描述了要让封装步骤成功完成所必须符合的条件。应确保符合这些条件，才能运行 SAVLICPGM 命令。

如果产品由多个库组成，您可能需要多次使用 EXPPART 命令。必须对运行的每个附加 EXPPART 命令创建一个部件列表部件。对于每个部件列表部件，还必须创建一个产品装入部件。*System Manager Use* 一书提供了 CRTPRDLOD 命令的 CL 命令示例，这些示例中提到了一个产品可以存在于多个库中的情况。

第14章 将应用程序分布至远程系统

系统列表部件是部件类型为 SYSTEML 的部件，它包含系统地址的列表，通过处理这些地址，可以将“应用程序开发管理器”分发发送至一个或多个远程系统。**分发**是已导出至远程系统的“应用程序开发管理器”部件。可以使用系统列表部件来将应用程序、应用程序修正和应用程序增强分布至远程系统。

本章提供有关下列主题的信息:
- 分布接收程序
- 构建前发行版应用程序
- 创建系统列表部件
- 将应用程序分布至远程系统
- 从远程系统接收对象

分布接收程序

为使用“接收部件”(RCVPART) 命令从远程系统接收分发，必须首先在远程系统上安装支持此命令的“应用程序开发管理器”程序的子集。远程系统的操作系统还必须是当前或前发行版级别。

如果远程系统的操作系统为当前发行版级别并安装有“应用程序开发管理器”功能部件，请跳过本节。

下列各节描述为接收分发而准备远程系统（未安装“应用程序开发管理器”）的过程。

不带应用程序开发管理器的当前系统

要将支持“接收部件”(RCVPART) 命令的接收程序分布至**未**安装“应用程序开发管理器”功能部件的当前系统，必须执行下列操作:

1. 在发送系统（已安装当前“应用程序开发管理器”功能部件的系统）上，通过输入以下命令以在 QADM 库中创建保存文件 QADMDIST:

   ```
   CALL QADM/QLYSAVDST
   ```

   注: 为成功使用此命令，您需要具有 *SECADM 和 *ALLOBJ 权限。

   然后，QLYSAVDST 程序将完成下列任务:

   a. 创建 QADMDIST 库

   b. 将表16中指定的对象列表从 QADM 复制到 QADMDIST 库中。表16还指示了对象是“机器可读信息”(MRI) 还是“机器可读材料”(MRM)。

表 16. 从 QADM 复制到 RISC 系统的 QADMDIST 库中的 MRI 和 MRM 对象

对象	对象类型	机器可读类型
QADMMSG	*MSGF	MRI
QHLYCMD	*PNLGRP	
RCVPART	*CMD	

表 16. 从 QADM 复制到 RISC 系统的 QADMDIST 库中的 MRI 和 MRM 对象 (续)

QLYCHKQT	*PGM	MRM
QLYCPP		
QLYCPYAA		
QLYDSFIL		
QLYDSTYP		
QLYEXEAA		
QLYIEXIT		
QLYINIT		
QLYMOVAA		
QLYRCVP		
QLYVRFY		

c. 在 QADM 库中创建名为 QADMDIST 的保存文件

d. 在“保存库”(SAVLIB) 命令的*目标发行版*参数上使用 *CURRENT 值，将 QADMDIST 库保存到保存文件中。

e. 删除 QADMDIST 库

注: 因为所有这些对象都是从 QADM 库复制的，所以机器可读信息 (MRI) 对象使用的将是创建保存文件的本地系统的主语言。如果想要创建包含使用任何其他语言的 MRI 对象的保存文件，则在本地系统上，您必须首先安装使用该语言的“应用程序开发管理器”功能部件作为辅助语言并以人工方式执行上述步骤。从辅助语言库 QSYS29xx（其中，xx 是语言标识符）复制“接收部件”(RCVPART) 命令、消息文件和屏面组对象。然后，将第169页的表16中列示的所有程序对象（或机器可读材料，即 MRM）从 QADM 库复制到 QADMDIST 库中。然后执行上述步骤1c至1e。

2. 将保存文件发送至用户想要使用“接收部件”(RCVPART) 命令的每个远程 RISC 系统。
3. 在接收系统上接收保存文件。
4. 在接收系统上，使用“恢复库”(RSTLIB) 命令将保存文件恢复到 QADMDIST 库中。

带有或不带应用程序开发管理器的前系统

要将支持“接收部件”(RCVPART) 命令的必需程序分布至操作系统的前发行版（已安装或未安装“应用程序开发管理器”功能部件），必须执行下列步骤:

1. 在发送系统（处于当前发行版并已安装“应用程序开发管理器”功能部件的系统）上，通过输入以下命令在 QADM 库中创建保存文件 QADMDISTP:

   ```
   CALL QADM/QLYSAVDSTP
   ```

   注: 为成功地使用此命令，您需要具有 *SECADM 和 *ALLOBJ 权限。

   然后，QLYSAVDSTP 程序将完成下列任务:

   a. 在 QADM 库中创建名为 QADMDISTP 的保存文件

   b. 通过使用“保存对象”(SAVOBJ) 命令的*目标发行版*参数上的值（例如，V4R4M0），将第171页的表17中指定的对象列表从 QADMDISTP 库复制到保存文件 QADMDISTP 中

表 17. 从 QADM 复制到 RISC 系统的 QADMDISTP 库中的 MRI 和 MRM 对象

对象	对象类型	机器可读类型
QADMMSG	*MSGF	MRI
QHLYCMD	*PNLGRP	
RCVPART	*CMD	

注: 因为一些 MRI 对象是从 QADM 库中复制的，所以它们将使用创建保存文件的本地系统的主语言。如果要创建包含使用另一语言的 MRI 对象的保存文件，则在本地系统上，您必须首先安装该语言的“应用程序开发管理器”功能部件作为辅助语言并以人工方式执行上述步骤。

不要替换 QADMDISTP 库中的“接收部件”(RCVPART) 命令。将消息文件和屏面组对象从辅助语言库 QSYS29xx（其中，xx 是语言标识符）复制到 QADMDISTP 库中。然后，完成步骤第170页的1a和第170页的1b。在创建保存文件之后，您可能想从 QADMDISTP 库中删除消息文件和屏面组对象。虽然将会使用本地系统的主语言保存“接收部件”(RCVPART) 命令，但它不应有任何显著影响，这是因为它不包含任何参数。在远程系统上，联机帮助以及“接收部件”(RCVPART) 命令发出的消息将使用辅助语言。

您可能想要保存 QADMDISTP 的主语言版本的副本，然后在创建 QADMDISTP 保存文件之后恢复它。

2. 将 QADMDISTP 保存文件从 QADM 库发送至用户想要在其中使用“接收部件”(RCVPART) 命令的每一个远程 RISC 系统。
3. 将 QADMPGM 保存文件从 QADMDISTP 库发送至用户想要在其中使用“接收部件”(RCVPART) 部件的每一个远程 RISC 系统。
4. 在接收系统上，接收这些保存文件。
5. 在接收系统上，创建 QADMDISTP 库。
6. 使用“恢复对象”(RSTOBJ) 命令，将保存文件 QADMDISTP 恢复到 QADMDISTP 库中。
7. 使用“恢复对象”(RSTOBJ) 命令，将保存文件 QADMPGM 恢复到 QADMDISTP 库中。

构建前发行版应用程序

如果想分布应用程序，以便在受编译器支持的前发行版的 OS/400 操作系统上运行它们，则必须首先在编译器命令的*目标发行版*上使用适当的值来构建应用程序。

对于标识的每个目标发行版:

1. 创建一个独立的组，使包含产品版本代码的组为其父组。
2. 在每一个这样的组中创建名为 QDFT 且带有提升码 *NONE 的构建选项部件。
3. 取消对所有必需编译器命令的注释。
4. 在*目标发行版*参数中添加适当的发行版值。
5. 在此组中使用前发行版编译器构建整个应用程序。

如果您使用的是任何其他具有特定名称的 BLDOPT 部件或 BLDOPT *compiler-names*，则必须对每个个别的部件重复上述步骤。

有关构建选项部件的详情，参见第138页的『使用构建选项』。

创建系统列表部件

可以使用系统列表部件来将应用程序、应用程序修正和应用程序增强分布至一个或多个远程系统。要创建系统列表部件，使用“创建部件”(CRTPART) 命令。

示例

以下命令在项目 PAYROLL 中的组 DEVELOPER1 中创建系统列表部件。该部件将名为 TESTSYS。

```
CRTPART PRJ(PAYROLL) GRP(DEVELOPER1) TYPE(SYSTEML) PART(TESTSYS) TEXT('Test
systems used for product A')
```

用于系统列表部件的缺省源文件是 QSYSTEMSRC。

使用“更改部件”(CHGPART) 命令来将远程系统地址添加至系统列表部件。例如，对于以上系统列表部件，可以添加下列条目:

```
TESTAS01        测试机器 001
TESTAS02        测试机器 002
TESTAS03        测试机器 003
```

您可以将分发发送至任何远程 iSeries 400 系统，即使它们不具有接收能力。如果远程系统不具有接收能力，这表示用户必须以人工方式接收这些分发。

SYSTEML 部件中的每个记录的头 8 个字符都必须是本地系统目录中定义的系统地址。每个记录的其余部分可以填入您认为有用的任何信息。在上面的示例中，给出了每个机器的简要描述。

每个“应用程序开发管理器”项目都可能创建多个系统列表部件。例如，对于项目中存储的应用程序的每个发行版，可能都会有一个系统列表部件。可以将一个系统列表部件用于当前发行版（包含当前发行版上的系统的列表），并对前发行版使用另一个系统列表部件。个别开发者可能还想设置一个系统列表部件，以便只包含他们想将他们的修正分布至的系统的名称。

将应用程序分布至远程系统

在“导出部件”(EXPPART) 命令的 SYSTEML 参数上指定系统列表部件的名称会导致它将部件导出至系统列表部件中列示的所有系统。确定您的用户简要表，并从系统目录获取相应的用户标识。分发被发送至系统列表部件中列示的每个系统地址上的该用户标识。

在“导出部件”(EXPPART) 命令的 SYSTEML 参数上指定 SYSTEML 名称将把分发导出至 SYSTEML 部件中列示的所有远程 iSeries 400 系统。不会将部件导出至本地 iSeries 400 系统。这些部件必须存在于从指定组到根组的缺省搜索路径中。

不在本地系统上处理“导出部件”(EXPPART) 命令上指定的 TOLIB、SRCFILE、DATA、REPLACE、OWNER 和 AUT 参数。而是将这些值与分发一起发送至系统列表部件中列示的每个远程系统地址，并在那里由“接收部件”(RCVPART) 命令处理。

如果您不在本地系统的目录中登记，则“导出部件”(EXPPART) 命令将失败。如果发送至的用户标识没有在远程系统的目录中登记，则分发将会失败。可以使用“添加目录条目”(ADDDIRE) 命令来在系统上登记该用户标识。

可使用 TGTRLS 参数来指定要将分发发送至的操作系统发行版。如果指定了 SYSTEML(*NONE)，则此参数被忽略。可以对 TGTRLS 参数使用缺省值 *CURRENT、*PRV 或上下文帮助中或提示屏幕上列示的任何选项。

从远程系统接收对象 (RCVPART)

使用“接收部件”(RCVPART) 命令来从一个或多个远程 iSeries 400 系统接收入局“应用程序开发管理器”分发。**入局分发**是使用“导出部件”(EXPPART) 命令从远程 iSeries 400 系统发送的部件。“接收部件”(RCVPART) 命令只处理那些与当前用户具有相同用户标识的用户发送的分发。

“接收部件”(RCVPART) 命令上没有任何参数。

对于每个分发，“接收部件”(RCVPART) 命令将入局部件移到在远程系统上发出的“导出部件”(EXPPART) 命令上指定的库中。另外，处理“导出部件”(EXPPART) 命令上指定的 SRCFILE、DATA、REPLACE、OWNER 和 AUT 参数的方式就象是正在从本地系统导出部件一样。

如果要以固定的时间间隔接收分发以接收修正，您可能想设置批处理作业来在特定的时间运行“接收部件”(RCVPART) 命令。

带有应用程序开发管理器的当前发行版系统

要从一个或多个已安装当前发行版的 Operating System/400 的远程 iSeries 400 系统接收入局“应用程序开发管理器”分发，使用“接收部件”(RCVPART) 命令。

不带应用程序开发管理器的当前发行版系统

要从一个或多个已安装当前发行版的 Operating System/400 的远程 iSeries 400 系统接收入局“应用程序开发管理器”分发:

1. 通过在任何 OS/400 命令行上输入 `ADDLIBLE QADMDIST` 来将 QADMDIST 添加至库列表的用户部分。
2. 运行“接收部件”(RCVPART) 命令。

不带应用程序开发管理器的前发行版系统

要从一个或多个支持“应用程序开发管理器”分发的远程 iSeries 400 系统接收入局“应用程序开发管理器”分发:

1. 通过在任何 OS/400 命令行上输入 `ADDLIBLE QADMDISTP` 来将 QADMDISTP 添加至库列表的用户部分。
2. 运行“接收部件”(RCVPART) 命令。

带有应用程序开发管理器的前发行版系统

要从一个或多个支持“应用程序开发管理器”分发的远程 iSeries 400 系统接收入局“应用程序开发管理器”分发:

1. 退出 PDM（如果您正在使用它的话）。
2. 通过在任何 OS/400 命令行上输入 CHGSYSLIBL QADMDISTP 来将 QADMDISTP 添加至库列表的系统部分。
3. 使用“回收资源”(RCLRSC) 命令回收资源。
4. 运行“接收部件”(RCVPART) 命令。
5. 通过在任何 OS/400 命令行上输入 CHGSYSLIBL QADMDISTP REMOVE(*YES)，从库列表的系统部分中除去 QADMDISTP。
6. 使用“回收资源”(RCLRSC) 命令回收资源。

任何其他 iSeries 400 系统

在不带“接收部件”(RCVPART) 命令的系统上，可以人工地从一个或多个远程 iSeries 400 系统接收入局“应用程序开发管理器”分发。

要人工地接收入局“应用程序开发管理器”分发:

1. 使用接收系统上的发送方的用户标识注册。
2. 在任何 OS/400 命令行上，输入 WRKNETF。入局分发的名称将是 QADMSAVF。
3. 将这些保存文件接收到库中。
4. 使用“恢复对象”(RSTOBJ) 命令恢复保存文件中的所有对象。

   注意，每个分发都将包含两个额外的数据区（QADMDATA 和 QADMTOLIB），它们不是分发的一部分，应删除它们。

限制

1. 因为分发算法使用库 QTEMP，所以不可能将部件发送至远程系统上的库 QTEMP。即，当在 SYSTEML 参数上指定系统列表部件时，不允许使用 TOLIB(QTEMP)。
2. 当 QTEMP 库中已有对象时，不可能发送或接收分发。“接收部件”(RCVPART) 或“导出部件”(EXPPART) 命令将失败。清除 QTEMP 库并重新发出该命令。
3. 您必须在系统目录中登记才能发送或接收“应用程序开发管理器”分发。

第15章 保护应用程序开发管理器信息

本章描述:

- “应用程序开发管理器”功能部件中的安全性
  - 其他项目安全性注意事项
- 备份和恢复策略
  - 系统灾难恢复
  - 异常结束“应用程序开发管理器”命令处理
  - 将项目信息移至另一个 iSeries 400 系统
- 使用项目记录
- 在用户离开时维护项目层次结构
- 当管理员作为开发者工作时的注意事项

注

因为“应用程序开发管理器”功能部件提供的安全性以 OS/400 系统提供的安全性为基础，所以，建议将“应用程序开发管理器”功能部件安装在安全级别为 30 或更高的系统上。

应用程序开发管理器功能部件中的安全性

在安装此功能部件之后，将创建一个名为 QPRJOWN 的用户简要表，此用户简要表具有 *USER 级别的权限，没有口令。此用户简要表是 ADM 创建的所有对象（程序除外）的所有者。

注: 您要了解，如果更改库 QSYS 中类型为 *USRPRF 的 QPRJOWN 对象的对象权限，则可能会发生不可预测的结果。通常，在用户运行其中一个“应用程序开发管理器”命令时创建的对象归此简要表所有。公共权限被设置为 *EXCLUDE，所有专用权限都被取消。例如，当运行 CRTGRP 命令来创建组及其相关的库时，该库归用户简要表 QPRJOWN 所有，并且不授予任何用户对该库的专用权限。因此，如果用户尝试在“应用程序开发管理器”控制之下采用对一个对象的所有者权限，他们会受到很大的限制。

管理员和开发者对 QPRJOWN 拥有的对象的访问是使用*权限列表*进行控制的。对于在项目层次结构中创建的每个组，将使用两个权限列表: 一个用来保护与该组相对应的库，另一个用来保护与该组中的部件相对应的 OS/400 对象。每个权限列表的名称都是项目简称与简短组名的组合。

每一个权限列表最初都只包含一个条目，此条目对 *PUBLIC 指定 *EXCLUDE 权限。当每个用户向项目登记时，向每个权限列表添加一个具有该用户的正确权限的条目。

用户具有的访问部件的权限取决于他或她如何在项目中进行登记。管理员对项目中的所有组拥有 *ALL 权限，而开发者对所有非开发组拥有 *USE 权限。开发者只有权在那些他们对其具有 *UPDATE 访问权的开发组中执行部件开发工作。

作为组简要表成员的用户通常授予该组中的其他用户对他们创建的对象的附加权限。“应用程序开发管理器”功能部件不允许发生这种情况。

另外，系统管理员通常在组简要表中登记用户，以将那些用户作为一个组来管理其权限。虽然“应用程序开发管理器”功能部件允许您向项目登记组简要表，但它不能成为该简要表中的成员。因此，不建议将组简要表登记到项目中。用户必须显式地登记到项目中才能对该项目中的部件执行“应用程序开发管理器”命令。

其他项目安全性注意事项

“应用程序开发管理器”功能部件引入了几种控制应用程序开发过程的方法。检出部件的操作确保每次只有一个开发者对该部件具有访问权。此控制是在使用“应用程序开发管理器”接口来对“应用程序开发管理器”信息执行操作时强加的。您应当使用“应用程序开发管理器 CL”命令或“编程开发管理器”实用程序来执行所有活动。

如果您决定使用非“应用程序开发管理器”接口来更改部件，则可能会发生不可预测的结果。例如，BLDPART 命令可能不识别部件已更改，因此不重新构建该部件。这可能会导致源部件与输出部件不一致。

另外，您不应该在“应用程序开发管理器”项目库中创建对象或文件。项目库中的所有信息都应该通过“应用程序开发管理器”功能部件来作为个别部件创建。如果要将文件或对象从一个库复制到项目层次结构中，使用 IMPPART 命令。

如果不首先使用应用程序开发管理器命令创建对象或文件，就不应该将那些对象或文件放到项目库中。否则，不会将这些对象或文件识别为“应用程序开发管理器”信息。

备份和恢复应用程序开发管理器信息

系统管理员应定期备份整个系统以确保组织可以在发生系统故障时恢复。“应用程序开发管理器”信息需要作为您的组织的常规备份和恢复过程的一部分进行备份。

除了执行常规系统备份和恢复（如果有必要的话）之外，有时，个别命令未完成处理，也需要恢复。命令处理可能会因为系统管理员取消了作业、发生电源故障或用户从系统请求菜单中选择选项 2 以结束先前请求而被中断。对于某些“应用程序开发管理器”命令，如果命令未正常地完成其处理，则信息可能会处于**不一致状态**。有关如何从类似于这样的情况进行恢复的信息，参见第179页的『从命令处理故障恢复』。

备份和恢复策略

使用『iSeries 信息中心备份、恢复和可用性』（在 URL http://www.ibm.com/eserver/iseries/infocenter 中）一节中描述的一种策略。必须保存下列各组对象:

- 权限列表，可用 SAVSYS 或 SAVSECDTA 命令保存
- ADM 创建的和 IBM 提供的用户库，可用 SAVLIB *ALLUSR 命令、SAVLIB *NONSYS 命令或 SAVE 菜单的选项 23 来保存。保存这些对象的最佳方法取决于您需要保存多少数据和您有多少时间。

有关保存对象的详细信息，参见“ iSeries 信息中心”和 *Backup and Recovery*, SC41-5304-05。可从 Infocenter 的主页中链接至此手册的联机版本。

如果用户简要表作为恢复过程的一部分进行恢复或重新创建，则当您尝试使用任何“应用程序开发管理器”命令时，可能接收到消息 ADM1804。要校正此情况，请作为 QSECOFR 注册。使用 RMVPRJUSR 命令除去所有项目用户，然后使用 ADDPRJUSR 命令将用户添加回到他们各自的项目中。

除了下列常规系统备份和恢复过程之外，在发生系统故障时，项目的管理员必须使用“回收项目”(RCLPRJ) 命令来确保在恢复库和对象之后正确地恢复所有“应用程序开发管理器”信息。因为 OS/400 系统命令恢复的信息不包括关于已删除或已移动的对象的信息，所以此附加的步骤是必需的。另外，不会保存“应用程序开发管理器”日志文件，而会保存它们的日志接收方。这会造成项目层次结构中的部件中与关于这些部件的信息不一致。

当在项目层次结构内恢复信息时，部件在项目层次结构中的位置也是您所需的信息的一部分。例如，假定将部件 PAYROLL DEVELOPER1 RPGSRC BIWKLY 提升到它的开发组之外的测试组中。如果出现的系统故障要求进行系统数据恢复，在您恢复“应用程序开发管理器”信息时，您将希望从组 DEVELOPER1 中除去该部件并在组 TEST 中恢复它。

当您定期从某些组中除去或删除部件时 – 例如，当您构建部件 PAYROLL DEVELOPER1 RPGSRC BIWKLY 时 – 创建的部件 (PAYROLL DEVELOPER1 PGM BIWKLY) 在未删除之前一直位于开发组中。在删除该部件之后，您不想让它在您恢复系统后又再出现。

使用 RCLPRJ 命令可以解决在恢复库和对象后“应用程序开发管理器”部件中存在的所有不一致性的问题（如前所述）。因为除了保存 OS/400 文件和对象之外，OS/400 保存和恢复命令还保存和恢复“应用程序开发管理器”功能部件创建和维护的关于那些文件和对象的信息，所以，项目层次结构会被恢复为其最初的或正确的状态。这就是使用建议的保存和恢复策略（而不是只保存和恢复特定库）的重要性所在。

迁移

有时，备份和恢复过程的目的是为了迁移至新的硬件或较新版本的操作系统。

当您备份系统时，应遵循本节中较早描述的策略之一。这将保存必需的与 ADM 有关的对象，包括 QUSRSYS 库中名为 QALY* 的数据库文件和 QLY* 日志和日志接收方。

每个组都有一个名称为 *proj.group* 的库和名称为 *proj.group* 和 *proj_group* 的权限列表，其中，*proj* 和 *group* 分别是项目简称和简短组名。

在迁移之前，以及在备份 QUSRSYS 库之前，使用下列命令来结束物理文件的日志记录:

```
ENDJRNPF FILE(*ALL) JRN(QUSRSYS/QLYJRN)
```

执行迁移，如下所示:

1. 确保恢复了所有 ADM 用户库、用户简要表和权限列表。
2. 此时，不要安装 ADM。如果已安装了 ADM，除去它，并删除库 QUSRSYS 中的所有 QALY* 和 QLY* 对象。这要求适当迁移先前的 ADM 环境。
3. 在安装 ADM 之前，恢复 QUSRSYS 中 QALY* 数据库文件的所有前发行版。
4. 安装最新版本的 ADM。
5. 作为 QSECOFR（安全主管）注册至 OS/400。

6. 在 OS/400 命令行上，输入下列命令:

   ```
   RCLPRJ PRJ(*ALL) OPTION(*TEST)
   ```

7. 查看报告，并备份要保存的所有孤立对象。
8. 在 OS/400 命令行上，输入下列命令:

   ```
   RCLPRJ PRJ(*ALL) OPTION(*RUN) DLTINCOBJ(*YES)
   ```

9. 对于报告中列示的每个对象，执行下列操作:
   a. 在 OS/400 命令行上，输入下列命令:

      ```
      PRTPRJUSR PRJ(name) USRPRF(*ALL)
      ```

   b. 对于项目中的每个用户，输入下列命令:

      ```
      RMVPRJUSR PRJ(name) USRPRF(name)
      ADDPRJUSR PRJ(name) USRPRF(name) USRTYPE(*DEVELOPER or *ADMIN)
        ACCESS(*READ or *UPDATE) DEVELOPGRP(groupnames)
      ```

现在就恢复了 ADM 环境。

回收项目 (RCLPRJ)

您必须是项目管理员才能回收项目。要求管理员在执行任何操作之前或在恢复系统之后，运行“回收项目”(RCLPRJ) 命令。在恢复日志之后，当第一次在 iSeries 400 系统上运行此命令时，将对所有项目的“应用程序开发管理器”数据库应用所有日志更改。但是，每个项目仍必须由其项目管理员回收。如果任何人试图使用需要回收的项目，则将发出错误消息，警告他或她该项目需要回收。

RCLPRJ 命令使用系统备份和恢复过程恢复的信息来确保将项目层次结构中的部件放回保存那些信息时它们所处的状态。换而言之，在处理 RCLPRJ 命令之后，可以在项目层次结构中精确地表示已提升或已删除的部件。

“应用程序开发管理器”功能部件创建和维护的关于文件和对象的信息称为**部件目录信息**。部件目录信息包含有关部件何时创建、上次更改时间以及它驻留在项目层次结构中的位置的信息。此信息与运行 RCLPRJ 命令时恢复的实际文件和对象相匹配。如果存在不一致性，则“应用程序开发管理器”功能部件会解决它们。

例如，假定部件目录信息指示一个部件不再存在于开发组中。然而，已在表示开发组的库中恢复了实际的对象。应删除该部件。RCLPRJ 命令允许您通过在 OPTION 参数上使用缺省值 *RUN 并在 DLTINCOBJ 参数上使用 *YES 来删除项目层次结构中没有对应部件目录信息的所有文件或对象。只删除具有系统提供的部件类型的对象或具有用户定义部件类型的对象。

注

如果已在“应用程序开发管理器”项目库中创建文件或对象，并且未使用应用程序开发管理器功能部件来执行此操作，则当您运行带有 OPTION(*RUN) 和 DLTINCOBJ(*YES) 的命令时，如果它们属于受“应用程序开发管理器”功能部件支持的类型，则会删除这些文件或对象。有关受支持的部件类型的列表，参见第50页的表4和第52页的表5。对于没有对应“应用程序开发管理器”类型（系统提供的类型或用户定义类型）的任何对象，既不报告也不删除它们。

您必须在 RCLPRJ 命令上指定项目的名称，或指示要回收您作为管理员对其具有权限的所有项目的项目信息。

如果要删除**不一致的对象**，请指定 DLTINCOBJ 参数。不一致的对象就是找不到相关部件目录条目的对象。

示例: 此示例说明如何回收样本 Payroll 项目的信息并删除此项目中没有关联部件目录信息的所有对象和成员。

使用 RCLPRJ 命令

```
RCLPRJ PRJ(PAYROLL) OPTION(*RUN)
```

使用“编程开发管理器”实用程序

在“用 PDM 来使用项目”屏幕上选择选项 37（回收）。

示例: 在运行 RCLPRJ 命令时，将生成一个报告，指示哪些对象或文件没有必需的部件目录信息。要打印 RCLPRJ 报告而不删除任何部件，请输入以下命令。

```
RCLPRJ PRJ(PAYROLL) OPTION(*TEST)
```

图61说明“回收项目”报告中提供的信息。在此示例中，根据部件目录信息，有 5 个部件未被识别。

```
5722WDS   V5R1M0              应用程序开发管理器      - 回收项目 05/08/01    10:35:57          页面 . . :  0001

项目  . . . . . . . . . . . . . . . . :   PAYROLL
选项  . . . . . . . . . . . . . . . . :   *RUN
删除不一致的对象  . . . . . . . . . . :   *NO
AS/400                  对象        源          成员        创建
库          对象        类型        成员        类型        日期      所有者        文本

PAY.PROD    ASSISTS     PGM                                 05/08/01  RICHARDS
PAY.PROD    GOALS       PGM                                 05/08/01  RICHARDS
PAY.PROD    PENALTIES   PGM                                 05/08/01  RICHARDS
PAY.PROD    POINTS      PGM                                 05/08/01  RICHARDS
PAY.TEST    QDDSSRC     FILE        NEWSTATS    PF          05/08/01  QPRJOWN
              * * * * *   列      表      结      束   * * * * *
```

图 61. RCLPRJ 命令生成的样本报告

如果您指示要回收关于您对其具有权限的所有项目的信息，则会为每个项目生成一个单独的报告。

RCLPRJ 命令既不报告也不删除在“应用程序开发管理器”项目库中找到的，尚未为它们定义系统提供的类型或用户定义类型的对象。

从命令处理故障恢复

当同时对部件目录信息和对象或文件执行操作的“应用程序开发管理器”命令在这两方面的步骤完成之前因某些原因而被中断时，项目信息便处于不一致的状态。这可能是因为系统操作员或安全主管在处理完成前取消了作业，这可能是因为用户通过系统请求菜单中的选项 2 结束了先前请求，或是因为发生电源故障造成的。

因为命令正常结束并向用户提供了错误消息，所以项目信息未处于不一致状态。例如，如果 CRTPRJ 命令返回一条消息，指示命令因为项目简称无效而失败，则所有项目信息仍一致。

当发现项目信息不一致时，处理部件开发命令的方式与处理项目管理命令的方式略有不同。

恢复部件开发命令

当未能彻底处理部件开发命令时，在发现不一致性之后，通常不需要进行人工介入就可以恢复部件。在开发者或管理员尝试使用或更改不一致的部件（此时，“应用程序开发管理器”功能部件自动解析错误状态）之前，不会发现不一致性。

即使不存在相对应的文件或对象，部件开发命令也允许部件目录信息存在。这表示即使 CRTPART 命令失败，部件目录信息页可以存在，尽管部件本身并不存在。当存在一个并不存在的部件（文件或对象）的部件目录信息时，管理员或开发者可以在 QRYPART 列表上*看到*此部件。

如果用户尝试对此部件执行一些操作，则“应用程序开发管理器”功能部件可以确定只存在部件目录信息，而部件本身并不存在。将删除该部件目录信息，命令在该组中找不到该部件。以后，如果输入 QRYPART 命令，便不再列示该部件。

如果作为命令操作目标的部件已损坏且不再可用，则需要使用 OS/400 备份和恢复命令恢复对象或文件的备份副本。有关如何执行此操作的详情，参见第176页的『备份和恢复策略』。

恢复项目管理命令

项目管理命令是通过再次输入该命令或通过输入 RCLPRJ 命令以人工方式恢复的。

如果 CRTPRJ、CHGPRJ 或 DLTPRJ 命令失败，则管理员应当再次输入同一命令。例如，如果输入了 CRTPRJ 命令，但此命令因为电源断电而失败，则任何尝试访问此项目的人员（管理员或开发者）都会看到一条指示项目不存在的消息。管理员应当再次输入同一个 CRTPRJ 命令。

如果下列其中一个命令在处理时失败，则必须运行指定了 OPTION(*RUN) 的 RCLPRJ 命令以使项目信息返回一致状态。

ADDPRJUSR	CRTGRP	RMVPRJUSR
CHGPRJUSR	DLTGRP	

将信息复制至另一个 AS/400 系统

如果要将项目层次结构和所有与其相关联的信息移动或复制至另一 iSeries 400 系统，请使用 EXPPART 和 IMPPART 命令。也可以将“应用程序开发管理器”项目库和归档库移至独立的辅助存储池 (ASP)，只要整个项目驻留在同一 ASP 中。有关这些命令的详情，参见第13章 导出应用程序和第6章 导入应用程序。

注意，不复制项目层次结构本身。管理员必须使用适当的“应用程序开发管理器”命令创建项目和该项目中的组。有关如何执行此操作的详情，参见第14页的『创建项目层次结构』。

还必须为新的项目层次结构创建访问或登记信息。有关如何执行此操作的详情，参见第26页的『在项目层次结构中登记用户』。

您可以使用下列两种方法来获取此信息：单版本应用程序和多版本应用程序。

单版本应用程序

如果应用程序没有多个版本的代码，或代码没有分布在多个组中，则此方法可能更好。

要复制的项目层次结构中的所有部件都应当在根组中。然后，可以使用 EXPPART 命令来将所有部件从此组导出到一个库中。可以在另一 iSeries 400 系统上保存和恢复此库。在第二个 iSeries 400 系统上，包含所有“应用程序开发管理器”信息的库与 IMPPART 命令配合使用。

在创建新的项目层次结构并添加用户登记信息之后，您应当将所有部件都导入到一个组中并在那个组中测试它们，以确保应用程序能够象预期的那样工作。

多版本应用程序

您可以使用此方法来移动或复制包含跨多个组分布的代码的项目层次结构或带有多个版本的应用程序。可以对项目的每个组使用带有 SCAN(*NO) 的 EXPPART 命令来将所有部件从此组导出到一个库中。您需要对项目的每个组重复此步骤，以便为每个组创建单独的库。现在，您可以在另一个系统上保存您创建的所有库并恢复它们。

在另一个系统上创建新的项目层次结构并添加用户登记信息之后，应当使用带有 REPLACE(*YES) 的 IMPPART 命令将所有部件从恢复的库导入到其相应的组中。您需要对项目层次结构的任何分支中从根组到节点组的每个组重复此步骤。

注意，即使是将带有多个版本的全新应用程序导入到“应用程序开发管理器”环境中，此方法也是很有用的。

使用项目记录

为了提供对项目的活动的审计跟踪，“应用程序开发管理器”功能部件会保持**项目记录**。项目记录通过记录创建或更改了哪些对象、活动何时发生以及操作由谁执行来跟踪开发者和管理员对项目执行的操作。给定 iSeries 400 系统上的所有“应用程序开发管理器”项目都记录在此项目记录中。

项目记录作为 iSeries 400 日志实现。此日志及其日志接收方驻留在 QUSRSYS 库中。项目记录的日志接收方必须驻留在此处才能被“打印项目记录”(PRTPRJLOG) 命令处理。没有任何用户（甚至是具有 QSECOFR 权限的用户）能够删除或更改个别日志条目。这能确保项目记录的安全性。

包含项目记录的日志的名称是 QLYPRJLOG。日志接收方的名称是 QLYPRJxxxx，其中，xxxx 是 1 到 9999 的数字。日志条目以用户定义的条目格式存储。

此产品自动监控日志接收方并在现存的日志接收方接近阈值限制时连接新的日志接收方。完成此操作时，将信息消息发送至 QSYSOPR 消息队列以通知操作员。您可以保存旧的日志接收方，并在以后删除它以释放存储器。建议的策略是每星期执行一次此项操作。有关日志记录的详情，参见 *Backup and Recovery*。

打印项目记录信息 (PRTPRJLOG)

请使用 PRTPRJLOG 命令来查看项目记录的内容。建议根据项目记录中要打印的信息的范围以批处理形式运行此命令。第182页的表18列示了记录的命令。

表 18. 打印项目记录中记录的命令

命令的类型	命令名
项目管理	ADDPRJUSR、CHGGRP、CHGPRJ、CHGPRJUSR、CRTGRP、CRTPRJ、DLTGRP、DLTPRJ、RCLPRJ RMVPRJUSR
部件开发	ADDADMLANG、ADDADMTYPE、BLDPART、CHGADMACN、CHGADMLANG、CHGPART、CHGPARTINF、CHKINPART、CHKOUTPART、CPYPART、CRTPART、CVTPART、DLTPART、EXPPART、IMPPART、MRGPART、PRMPART、RMVADMLANG、RMVADMTYPE

一些部件开发命令允许使用单一命令对许多部件执行操作。例如，可以使用 PRMPART 命令来提升一个组中的所有部件。除 BLDPART 命令之外，会为每一个作为这些命令的操作目标的部件创建一个记录条目。对于 BLDPART，只创建一个条目，此条目显示用户输入的 BLDPART 命令。

示例

PRTPRJLOG 命令上唯一必需的参数是要查看其条目的项目的名称。要打印描述所有开发者和管理员今天对 Payroll 项目使用“应用程序开发管理器”命令的活动的报告，请使用以下命令。图62说明了创建的报告。

```
PRTPRJLOG PRJ(PAYROLL)
```

```
5722WDS   V5R1M0            应用程序开发管理器     - 打印项目记录  05/08/01     10:29:07          页面 . . :  0001

项目  . . . . . . . . . . . . . . . . :   PAYROLL
开始日期  . . . . . . . . . . . . . . :   05/08/01
结束日期  . . . . . . . . . . . . . . :   05/08/01
搜索字符串  . . . . . . . . . . . . . :

用户简要表   日期       时间       命令

TREMBLAY     05/08/01    10:07:52   CRTPRJ PRJ(PAYROLL) SHORTPRJ(PAY) SAVDTA(*YES) TEXT('*BLANK')
TREMBLAY     05/08/01    10:08:42   CRTGRP PRJ(PAYROLL) GRP(PRODUCTION) SHORTGRP(PROD) PARENT(*NONE) PRMCODE(*PARENT) CCSID(*P
                                    ARENT) TEXT('*BLANK')
TREMBLAY     12/22/95    10:09:10   CRTGRP PRJ(PAYROLL) GRP(TEST) SHORTGRP(TEST) PARENT(PRODUCTION) PRMCODE(*PARENT) CCSID(*PA
                                    RENT) TEXT('*BLANK')
TREMBLAY     12/22/95    10:09:42   CRTGRP PRJ(PAYROLL) GRP(SMITH) SHORTGRP(SMITH) PARENT(TEST) PRMCODE(*PARENT) CCSID(*PARENT
                                    ) TEXT('*BLANK')
TREMBLAY     12/22/95    10:10:21   CRTGRP PRJ(PAYROLL) GRP(DOUG) SHORTGRP(DOUG) PARENT(TEST) PRMCODE(*PARENT) CCSID(*PARENT)
                                    TEXT('*BLANK')
TREMBLAY     12/22/95    10:11:38   ADDPRJUSR PRJ(PAYROLL) USRPRF(SMITH) USRTYPE(*DEVELOPER) ACCESS(*UPDATE) DEVELOPGRP(TANYA)
TREMBLAY     12/22/95    10:12:10   ADDPRJUSR PRJ(PAYROLL) USRPRF(DOUG) USRTYPE(*DEVELOPER) ACCESS(*UPDATE) DEVELOPGRP(DOUG)
SMITH        12/22/95    10:13:45   CRTPART PRJ(PAYROLL) GRP(SMITH) TYPE(DDSSRC) PART(MTHLY) LANG(PF) PROMPT(*NO) PRMCODE(*GRP
                                    ) SRCFILE(*TYPE) TEXT('*BLANK')
SMITH        12/22/95    10:14:07   CHKOUTPART PRJ(PAYROLL) GRP(SMITH) TYPE(DDSSRC) PART(MTHLY) PRMCODE(*GRP)
SMITH        12/22/95    10:15:06   CHGPART PRJ(PAYROLL) GRP(SMITH) TYPE(DDSSRC) PART(MTHLY) CHGCMD(*TYPE)
SMITH        12/22/95    10:15:25   BLDPART PRJ(PAYROLL) GRP(SMITH) TYPE(DDSSRC) PART(MTHLY) SCHPTH(*DFT) SCOPE(*NORMAL) FORCE
                                    (*NO) BLDMODE(*COND) SAVLST(*NO)
SMITH        12/22/95    10:18:11   PRMPART PROJ(PAYROLL) GROUP(SMITH) TYPE(DDSSRC) PART(MTHLY)
SMITH        12/22/95    10:19:45   DLTPART PROJ(PAYROLL) GROUP(SMITH) TYPE(FILE) PART(MTHLY)
TREMBLAY     12/22/95    10:20:44   BLDPART PRJ(PAYROLL) GRP(TEST) TYPE(*ALL) PART(*ALL) SCHPTH(*DFT) SCOPE(*NORMAL) FORCE(*NO
                                    ) BLDMODE(*COND) SAVLST(*NO)
SMITH        12/22/95    10:22:13   ADDADMTYPE TYPE(ZPASSRC) SYSTYPE(*MBR) DFTSRCF(QRPGSRC) DFTRCDLEN(00092) ALWI NC(*NO) INCL
                                    UDES(*NONE)
SMITH        12/22/95    10:23:59   ADDADMLANG TYPE(ZPASSRC) LANG(RPG38) BLDCMD(*NONE) BLDOUTTYPE(*NONE ) BLDOUTLANG(*INPUT) S
                                    PLFNAME(*INPUT)
                * * * * *  列      表      结      束   * * * * *
```

图 62. PRTPRJLOG 命令生成的样本报告

如果您不止想要查看今天的活动，则可以在 PRTPRJLOG 命令上指示 FROMDATE 和 TODATE。

在 FROMDATE 参数上，支持特殊值 *CURRENT 和 *BEGIN。缺省情况是使用 *CURRENT，即，今天的日期。如果指定 *BEGIN，则您将看到从项目记录的开始到

TODATE 参数上指定的日期的所有项目记录条目。也可以在 FROMDATE 参数上指定特定的日期。此日期必须使用系统值 QDATFMT 指定的格式，并且，如果使用分隔符，则使用系统值 QDATSEP 指定的格式。

在 TODATE 参数上，支持特殊值 *CURRENT。它是缺省值。也可以在 TODATE 参数上指定特定的日期。此日期必须使用系统值 QDATFMT 指定的格式，并且，如果使用分隔符，则使用系统值 QDATSEP 指定的格式。

也可以根据用户简要表打印条目。USRPRF 参数的缺省值是 *ALL。这会导致打印包含对项目具有访问权的所有用户的条目的报告。

如果您只想查看您自己在项目中的活动，请指定 USER(*USRPRF)。如果要查看特定人员的活动，请在 USRPRF 参数上指定用户简要表名。

如果要指示选择审计记录的标准，请使用 STRINGS 参数。您可以使用此功能来通过指定命令中包含的字符串限制审计记录报告。例如，您可以指定一个特定的命令名或命令中使用的特定组和部件名。

在 STRINGS 参数上，最多可以指定 4 个出现在审计记录中的字符串。每个字符串最多可以包含 32 个字符。所有比较都以大写完成。

选择的每个审计记录条目都将包含指定的**所有**字符串。

注: 效果相当于对所有字符串执行 AND 操作。

如果要打印包含任何一个字符串的报告，则需要通过指定每一个字符串单独地发出 PRTPRJLOG 命令。

示例

本示例说明如何打印带有用户简要表 ROBERTS 的当天的活动的报告。

使用 PRTPRJLOG 命令

```
PRTPRJLOG PRJ(PAYROLL) FROMDATE(*CURRENT) TODATE(*CURRENT) USRPRF(ROBERTS)
          STRINGS("RPGSRC")
```

使用“编程开发管理器”实用程序

在“用 PDM 来使用项目”屏幕上选择选项 36（打印记录）。

如果不想打印 PRTPRJLOG 命令生成的报告，则可以通过选择 OUTPUT(*OUTFILE) 来将输出引导至输出文件。必须指定用于接收输出的库和文件。以下命令就是如何完成此操作的一个示例。

```
PRTPRJLOG PRJ(PAYROLL) FROMDATE(*CURRENT) TODATE(*CURRENT) USRPRF(ROBERTS)
          OUTPUT(*OUTFILE) OUTFILE(*LIBL/OUTFILE) OUTMBR(*FIRST)
```

输出文件的记录格式与库 QADM 中由系统提供的数据库文件 QALYPRJLOG 中使用的记录格式相同。

在用户离开项目时进行恢复

雇员转至组织中的其他工作职位或完全离开单位并不常见。如果开发者离开处于应用程序开发周期中的部件位置，则管理员需要验证此人的工作是否已完成。根据开发者离开的环境的不同，可以将进行中的工作分配给另一个开发者、由管理员完成或从项目层次结构中删除。

列示开发组中的部件

第一步是评估开发者的开发组中有多少工作仍未完成。使用 “查询部件” (QRYPART) 命令来确定存在于此组中的信息。

示例

以下命令创建一个报告，此报告列示开发组 ROBERTS 中的所有部件。

```
QRYPART PRJ(PAYROLL) GRP(ROBERTS) TYPE(*ALL) PART(*ALL) OUTPUT(*PRINT)
        SCAN(*NO)
```

指定 SCAN(*NO) 选项的目的是，只列示 ROBERTS 组中的那些部件，而不列示此组的搜索路径中的所有部件。

删除不能提升的部件

可以删除类型为 PGM 的部件，这是因为它们是由构建过程使用 BLDPART 命令重新创建的。其他类型（例如，FILE、CLD、CMD 和 MODULE）的部件也可能会被删除，这是因为正常情况下不能提升这些部件。（通过使用带有 EXTEND(*YES) 参数的 PRMPART 命令并指定其相应的源部件，可以提升类型为 FILE、CLD、CMD 和 MODULE 的部件。）使用 DLTPART 命令来删除这些部件（如果您希望这样做的话）。

示例

以下命令删除 ROBERTS 开发组中类型为 PGM 的所有部件。

```
DLTPART PRJ(PAYROLL) GRP(ROBERTS) TYPE(PGM) PART(*ALL)
```

下一步是确定开发组中列示的哪些部件已检出至用户简要表 ROBERTS。（在此示例中，开发组的名称与用户简要表的名称是相同的。）

对于此组中余下的每一个部件，使用 “打印部件信息” (PRTPARTINF) 命令来确定访问密钥是否设置为 ROBERTS。对于 QRYPART 命令，在 PRTPARTINF 命令上使用 SCAN(*NO) 参数来限制对此组的搜索。

示例

以下命令生成一个报告，此报告列示关于部件 BIWKLY 的信息。

```
PRTPARTINF PRJ(PAYROLL) GRP(ROBERTS) TYPE(RPGSRC) PART(BIWKLY)
           OUTPUT(*PRINT) SCAN(*NO)
```

如果报告显示访问密钥设置为 ROBERTS，则需要进行人工介入。开发者离开单位的环境确定了如何处理下一步。必须使用 CHGPARTINF 命令将访问密钥更改为下列其中一个值:

- 将访问密钥设置为 *NONE，以便可以删除该部件
- 将访问密钥设置为另一个开发者的用户简要表（将工作传送给该开发者）
- 将访问密钥设置为您自己的用户简要表。

示例

如果您觉得不应保留部件中的更改，则使用以下命令将访问密钥更改为 *NONE。

```
CHGPARTINF PRJ(PAYROLL) GRP(ROBERTS) TYPE(RPGSRC) PART(BIWKLY)
           LANG(*SAME) ACCKEY(*NONE)
```

此命令释放该部件的访问密钥。现在，可以使用 DLTPART 命令删除此组中的副本。然而，请注意，对该部件所作的任何更改都会丢失。如果该部件是一个新部件，并且尚未提升，于是，当从组 ROBERTS 中删除它时，将彻底地从项目层次结构中除去它。如果尚未提升该部件，则另一个开发者可以将旧部件检出并在他或她的开发组中更改该部件。

如果您觉得应该对部件中的更改进行评估，请将访问密钥更改为您自己的用户简要表，或使用 CHGPRJUSR 命令来授予另一个开发者对组 ROBERTS 的更新访问权，并将访问密钥更改为该人员的用户简要表。这允许访问密钥上指定的人员更改该部件。

示例

下列命令给予 SMITH 必需的更新访问权并更改部件 BIWKLY 的访问密钥，以使 SMITH 可以使用它。

```
CHGPRJUSR PRJ(PAYROLL) USRPRF(SMITH) USRTYPE(*SAME) ACCESS(*UPDATE)
          DEVELOPGRP(SMITH ROBERTS)

CHGPARTINF PRJ(PAYROLL) GRP(ROBERTS) TYPE(RPGSRC) PART(BIWKLY)
           LANG(*SAME) ACCKEY(SMITH)
```

您必须对开发组 ROBERTS 中其访问密钥设置为 ROBERTS 的每一个部件使用 CHGPARTINF 命令。

在测试更改之后，便可以提升部件。

除去项目用户 (RMVPRJUSR)

下一步是使用“除去项目用户”(RMVPRJUSR) 命令除去开发者的访问权或登记信息。

示例

以下命令从项目中除去 ROBERTS。

```
RMVPRJUSR PRJ(PAYROLL) USRPRF(ROBERTS)
```

“应用程序开发管理器”功能部件确保项目中始终登记有一位项目管理员。如果尝试除去唯一的项目管理员，则会发出错误消息。

在唯一的项目管理员离开项目这种极少发生的情况下，某个人必须使用管理员的用户简要表或 QSECOFR 用户简要表注册到 iSeries 400 系统上，并使用 ADDPRJUSR 命令向项目登记新的项目管理员。然后，新的管理员可以使用 RMVPRJUSR 命令来除去先前管理员的登记信息。

删除组 (DLTGRP)

最后一步是从项目层次结构中除去组（如果其他开发者没有共享该组的话）。

示例

以下命令除去组 ROBERTS。

```
DLTGRP PRJ(PAYROLL) GRP(ROBERTS)
```

“应用程序开发管理器”功能部件确保正在删除的组是空的。例如，如果您尝试使用 DLTGRP 命令删除一个组，而它仍包含部件且那些部件的访问密钥设置为已指定给该组的用户，则此命令会失败。

使用日志进行审计跟踪

“应用程序开发管理器”使用日志来保存此产品的所有命令的审计跟踪。您可以使用 PRTPRJLOG 命令打印此审计跟踪。此审计跟踪是使用 QUSRSYS 中的日志 QLYJRN 实现的。QUSRSYS 中的另一个日志 QLYPRJLOG 用来记录产品的数据库文件的所有事务。这些事务用于回收项目。

日志对系统日期和时间敏感。如果因为任何原因您将系统日期更改为任何将来的日期和时间，请确保没有任何人使用此产品的任何命令，直到系统日期和时间返回正常为止。如果有人这样做的话，则您将无法按预期的那样使用 PRTPRJLOG 命令。PRTPRJLOG 命令以内部方式使用的 DSPJRN 命令不检索任何越过当前日期（PRTPRJLOG 使用的缺省值）的记录，因此不会返回任何在包含将来时间戳记的记录之后的记录。QLYPRJLOG 日志也可能会发生这种情况，回收项目时这将会导致类似的问题。

要解决此问题，您需要:

1. 查找并删除包含带有将来时间戳记的错误条目的日志接收方。如果条目在当前日志接收方中，请记下正在记入日志的文件的名称和当前日志接收方的名称（可以对 QUSRSYS/QLYJRN 使用 WRKJRNA 命令）。
2. 对正在对 QUSRSYS/QLYJRN 记入日志的所有文件发出 ENDJRNPF 命令。
3. 删除 QUSRSYS 中的 QLYJRN 日志。
4. 删除当前日志接收方。
5. 创建日志接收方 QLYRCVnnnn（其中 nnnn 是从 0001 开始的 4 位数），并指定 THRESHOLD(10000)。
6. 创建与 QLYRCV$_{NNNN}$ 日志接收方相关联的 QLYJRN 日志。
7. 使用 STRJRNPF 命令，对您早先记下其名称的物理文件启动日志记录。

对 QUSRSYS 库中的日志 QLYPRJLOG 重复上面的步骤 1 和步骤 3 至 6。

作为开发者的管理员的注意事项

同时作为开发者的项目管理员应该明白，某些部件开发命令必须谨慎使用。如果项目管理员忘记了他们可以对特定部件开发命令产生比开发者更广泛的影响这一事实，有时就有可能会产生问题。

通过使用 DLTPART 命令，管理员可以从开发组或从项目层次结构中的任何其他组中删除部件。当作为开发者工作时，管理员必须谨慎地使用此命令。

通过使用 CHGPARTINF 命令，管理员还可以更改部件的访问密钥。在更改已检出至开发者的部件的访问密钥之前，管理员应当确保开发者知道他们的介入。

第16章 使用编程开发管理器实用程序

作为 CL 命令的替代方法，您可以使用“编程开发管理器”实用程序来执行部件开发任务。本章讨论如何通过用“编程开发管理器”实用程序来使用项目、组和部件。

编程开发管理器实用程序概述

“编程开发管理器”屏幕列示可供您使用的项目、组或部件以及您可以对它们执行的操作。操作是以屏幕顶部的编号选项和屏幕底部的功能键形式呈示的。要执行操作，请在适当项目、组或部件旁边的“选项”字段中输入操作的编号，然后按“执行”键，或使用适当的功能键。

在有更多选项可用的屏幕上，要查看其余选项，请按“F23=其余选项”。如果要对列示的所有部件重复执行某个选项，请输入选项号，然后使用“F13=重复”。这将对第一个部件下面的所有部件重复该选项。在有更多功能键可用的任何屏幕上，要查看其余功能键，请按“F24=其余键”。

在某些屏幕上可以输入特定的名称或类属名。用于指定 **generic** 值的格式对所有屏幕都是一样的。如果使用类属值，则可以输入由星号 (*) 限定的部分名以显式特定子集。例如，类属名可以具有下列格式之一:

ABC*
显示以字符 ABC 开头的所有项的列表。

***ABC**
显示以字符 ABC 结尾的所有项的列表。

B
显示名称中的任何位置带有字符 B 的所有项的列表。

A*C
显示以字符 A 开头并以字符 C 结尾的所有项的列表。

"a*"
显示位于引号内并以 a 开头的所有项的列表。

****ALL**
显示以 ALL 结尾的所有项的列表。因为 *ALL 是用于显示所有项的列表的特殊值，所以这里需要双星号。

某些提示的缺省值为 *ALL，这些值会在您首次使用屏幕时显示。下一次使用屏幕时，您上次输入的输入将作为缺省值出现。要恢复先前使用的值，请按“F5=刷新”。有关如何使用屏幕的详情，参见*ADTS/400: Programming Development Manager*。

入门

要启动“编程开发管理器”实用程序，输入 STRPDM 命令。出现主菜单，如第188页的图63所示。

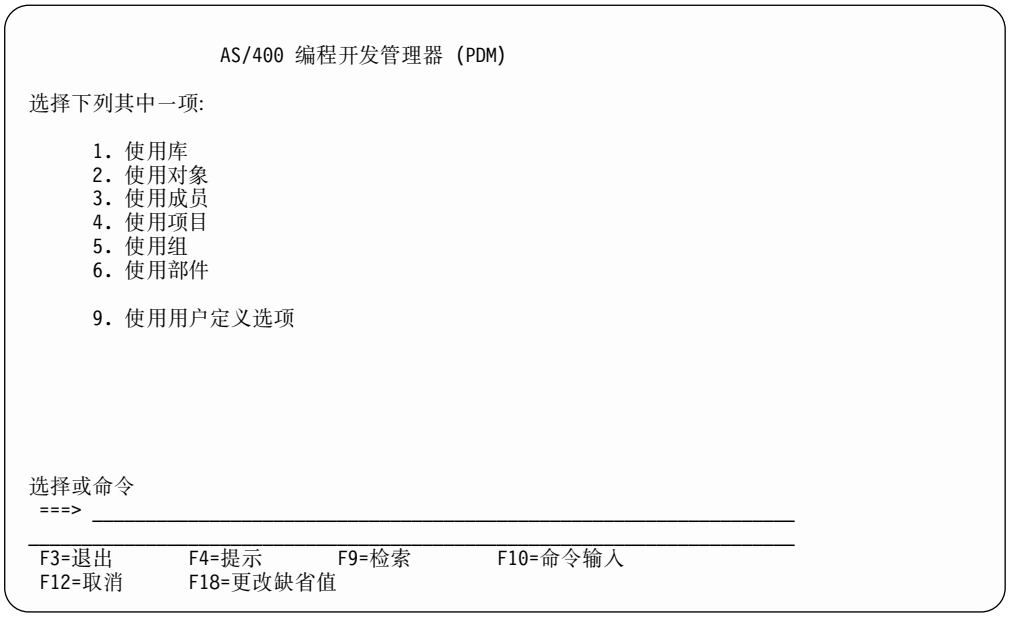

图 63. AS/400 编程开发管理器 (PDM) 主菜单

仅当已安装“应用程序开发管理器”功能部件时，选项 4、5 和 6 才会出现在屏幕上。

如果已知道包含您所需的部件的项目和组的名称，请选择选项 6。如果您知道项目但需要确定组，请选择选项 5。如果两者都需要确定，请使用选项 4。（选项 1、2 和 3 并不直接与“应用程序开发管理器”功能部件相关，因此本书不讨论它们。没有选项 7 和 8。）

指定项目

可以在“用 PDM 来使用项目”屏幕中指定要处理的项目。调用此屏幕的方法有两种:

1. 从主菜单中，选择选项 4（使用项目）。出现“指定要处理的项目”屏幕，如图64所示。

```
                          指定要处理的项目

输入选项，按“执行”键。

  项目 . . . . . .  *ALL_____________________________  *ALL，名称
                                                       *generic*









F3=退出      F5=刷新      F12=取消
```

图 64. 指定要处理的项目屏幕

a. 在*项目*提示中，如果您知道您要处理的项目，则输入特定的名称或类属名。如果要查看您已向其登记的所有项目的列表，请指定 *ALL。

b. 按“执行”键。“用 PDM 来使用项目”屏幕出现，此屏幕列示您在前一屏幕上指定的项目。参见图65。

```
                          用 PDM 来使用项目

定位至  . . . . . .  _____________________________

输入选项，按“执行”键。
  2=更改          4=删除          5=显示          6=打印
  12=使用         35=显示记录     36=打印记录     37=回收 ...

选项  项目                          文本
__    PAYROLL                       WEEKLY PAYROLL PROCESSING APPLICATION









                                                                      底部
参数或命令
===>_____________________________________________________________________
F3=退出         F4=提示         F5=刷新         F6=创建
F9=检索         F12=取消        F23=其余选项    F24=其余键
```

图 65. 用 PDM 来使用项目屏幕

通过使用此屏幕，您可以:

- 创建、删除或回收项目

- 更改、显示或打印项目信息
- 显示或打印项目的审计记录信息
- 添加、更改、除去、显示或打印关于项目用户的信息

有关审计记录的详情，参考第15章 保护应用程序开发管理器信息。

2. 从“编程开发管理器”实用程序外部，您可以使用 WRKPRJPDM 命令来直接转至“用 PDM 来使用项目”屏幕，而绕过主菜单和“指定要处理的项目”屏幕。在命令行上输入 WRKPRJPDM。

   如果按“执行”键，会出现该屏幕，并显示您已对其作了登记的所有项目。这等同于在“指定要处理的项目”屏幕上指定 *ALL。如果按“F4=提示”，则首先会显示提示。使用缺省值 *PRV（指示想要使用在前一会话中指定的内容），*ALL、*generic* 或输入名称，然后按“执行”键。于是，“用 PDM 来使用项目”屏幕出现，只显示您指定的项目。

在“用 PDM 来使用项目”屏幕上，在想要查看它的组的项目旁输入 12（使用）。此选项显示第191页的图67所示的“用 PDM 来使用组”屏幕。

指定组

您可以在“使用组”屏幕中指定要处理的组。调用此屏幕的方法有三种:

1. 从主菜单中，选择选项 5（使用组）以显示图66所示的“指定要处理的组”屏幕。

```
                          指定要处理的组

输入选项，按“执行”键。

  项目 . . . . . .   PAYROLL_____________________   名称

  组   . . . . . .   *ALL______________________   *ALL，名称

F3=退出      F5=刷新      F12=取消
```

图 66. 指定要处理的组屏幕

   a. 在*项目*提示中，输入包含要查看的组的项目的名称。在*组*提示中，如果知道想要使用的组，请输入特定的名称或类属名。如果您不知道组的名称，则按“F4=提示”以获取项目中所有组的列表。在此列表中，可选择想要使用的组。

b. 按“执行”键。“用 PDM 来使用组”屏幕出现，此屏幕列示您在前一屏幕上指定的组。使用“F11=显示层次结构”来以一种特定的方式显示组，在这种方式中，每一个组的级别与项目层次结构的其余部分的关系都将显示出来。目标组位于列表顶部。参见图67。

   如果您在前一会话中使用了 F11 键，则当“用 PDM 来使用组”屏幕出现时，组以分层结构的形式呈示。

```
                          用 PDM 来使用组

项目. . . . . .  PAYROLL_____________________

 输入选项，按“执行”键。
  2=更改          4=删除        12=使用        14=构建
  25=查找字符串   29=检入       30=提升        38=导入 ...

选项  级别   组
__    01    MASTER
__    02      TEST1
__    03        DEVELOPER1
__    03        DEVELOPER2
__    02      TEST2
__    03        DEVELOPER3


                                                                  底部
 参数或命令
===> ____________________________________________________________________________
F3=退出       F4=提示       F5=刷新       F6=创建
F9=检索       F10=命令输入  F23=其余选项  F24=其余键
```

图 67. 用 PDM 来使用组屏幕

在“用 PDM 使用组”屏幕上，按“F23=其余选项”可以访问下列附加选项。

```
输入选项，按“执行”键。
  39=导出          45=添加项目库列表 ...
```

图 68. 按 F23 键后出现的选项

而且，在“用 PDM 使用组”屏幕上，按“F24=其余键”可以访问下列附加功能键。按一次“F24=其余键”可以得到图69中说明的键，按两次可以得到第192页的图70中说明的键。

```
参数或命令
===> ____________________________________________________________________________
F11=只显示组          F12=取消     F13=重复     F14=显示文本
F18=更改缺省值        F23=其余选项              F24=其余键
```

图 69. 按一次 F24 键后出现的功能键

```
参数或命令
===> _____________________________________________________________________
F20=除去项目库列表                          F21=打印列表
F23=其余选项                                F24=其余键
```

图 70. 按两次 F24 键后出现的功能键

在此屏幕中，您可以执行下列任务:

- 更改组信息
- 从项目层次结构中删除组
- 构建、检入、提升、导入、导出组中的所有部件或或在其中查找字符串

2. 第189页的图65中显示的“用 PDM 来使用项目”屏幕中，在要查看它的组的项目旁输入 12（使用）。“用 PDM 来使用组”屏幕出现，此屏幕列示了此项目中的所有组。
3. 从“编程开发管理器”实用程序外部，您可以使用 WRKGRPPDM 命令来直接转至“用 PDM 来使用组”屏幕。这样的话，您就可以绕过主菜单和“指定要处理的组”屏幕。

   在命令行上输入 WRKGRPPDM。如果按“执行”键，则出现该屏幕，并显示包含在您访问的上一个项目中的所有组。（如果这是您的第一个会话，则不能确定任何项目，您将接收到一条错误消息。）

   如果您按“F4=提示”，则会首先显示提示。使用缺省值 *PRV 来指示想要使用在前一会话中指定的内容（假定这不是您的第一个会话），或输入名称，并按“执行”键。该屏幕出现，只显示您指定的组。

在“用 PDM 来使用组”屏幕上，在要查看它的部件的组旁输入 12（使用）。开发者对他们的项目中的所有组都具有读访问权。

使用部件

您可以在“使用组”屏幕中指定要使用的组。调用此屏幕的方法有三种:

1. 从主菜单中，选择选项 6（处理部件）以显示图71所示的“指定要处理的部件”屏幕。

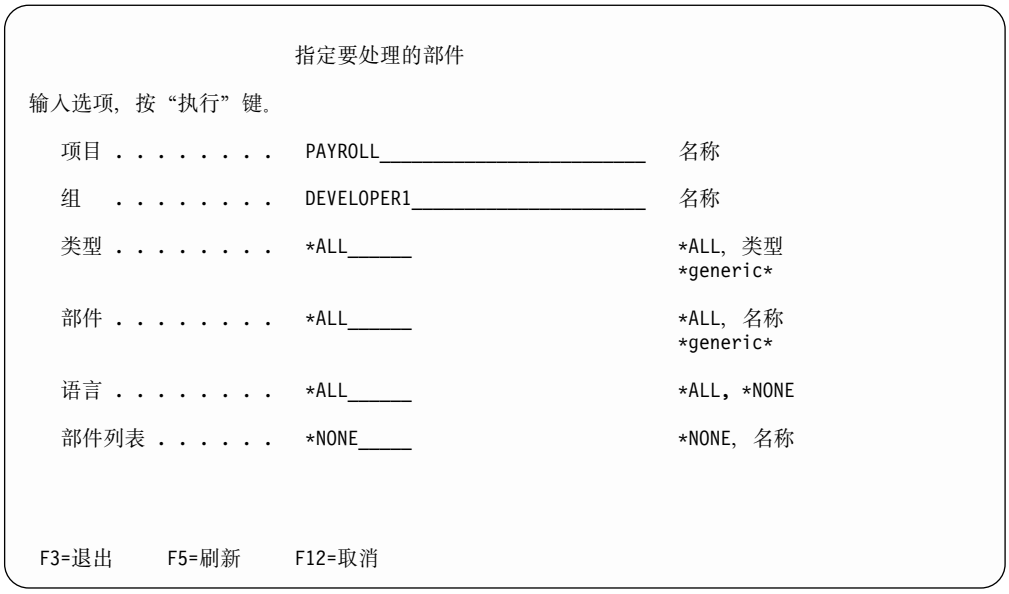

图 71. 指定要处理的部件屏幕

在*项目*和*组*提示中，输入包含要的部件的项目和组的名称。在*类型*、*部件*和*语言*提示中，输入特定名称或类属名作为要的部件的类型、部件名或语言，或者输入 *ALL（如果要查看所有部件的话）。

在*部件列表*提示中，输入部件列表部件的特定名称或 *NONE。

如果您指定 *NONE 并按“执行”键，则出现“用 PDM 来使用部件”屏幕。参见第194页的图72。

如果您指定部件列表部件的特定名称并按“执行”键，则出现“用 PDM 来使用部件列表部件”屏幕。参见第197页的图79。

或者，在“用 PDM 来使用部件”屏幕中，您可以指定 PL 用户定义选项以转至“用 PDM 来使用部件列表中的部件”屏幕。

```
PL  WRKPARTPDM PRJ(&ZP) GRP(&ZG) TYPE(*ALL) PART(*ALL) LANG(*ALL) PLIST(&N)
```

2. 在上述“用 PDM 来使用组”屏幕中，在要查看其部件的旁边输入 12（使用）。出现“用 PDM 来使用部件”屏幕。
3. 从“编程开发管理器”实用程序外部，您可以使用 WRKPARTPDM 命令来直接转至“用 PDM 来使用部件”屏幕，而绕过主菜单和“指定要处理的部件”屏幕。

   在命令行上输入 WRKPARTPDM。如果您按“执行”键，则该屏幕立即出现，并显示包含在您上次访问的组中的所有部件。（如果这是您的第一个会话，则不能确定任何项目和组，您将接收到一条错误消息。）

   如果您按“F4=提示”，则会首先显示提示。在*项目*和*组*提示中，输入 *PRV 以指示要使用您在前一会话中指定的内容（假定这不是您的第一个会话），或输入特定名称。在*类型*、*部件*和*语言*提示中，输入特定名称或类属名作为要使用的部件的类型、部件名或语言，或者输入*ALL（如果要查看所有部件的话）。按“执行”键。于是，出现该屏幕，显示包含在您指定的组中的所有部件。

用 PDM 来使用部件

“用 PDM 来使用部件”屏幕显示“更改会话缺省值”屏幕上指定的搜索路径中存在的部件。如果同一部件存在于多个组中，您只能看到存在于搜索路径中的最低层的组中的部件。您只能看到“用 PDM 来使用部件”屏幕上列示的同一部件的一个副本。部件按以下次序列示:

1. 源部件按类型以字母顺序列示。每种类型的部件按它们的部件名以字母顺序列示。
2. 包含部件。
3. 构建选项部件。
4. 搜索路径部件。
5. 所有其他对象类型。
6. 用户定义类型按类型以字母顺序列示。

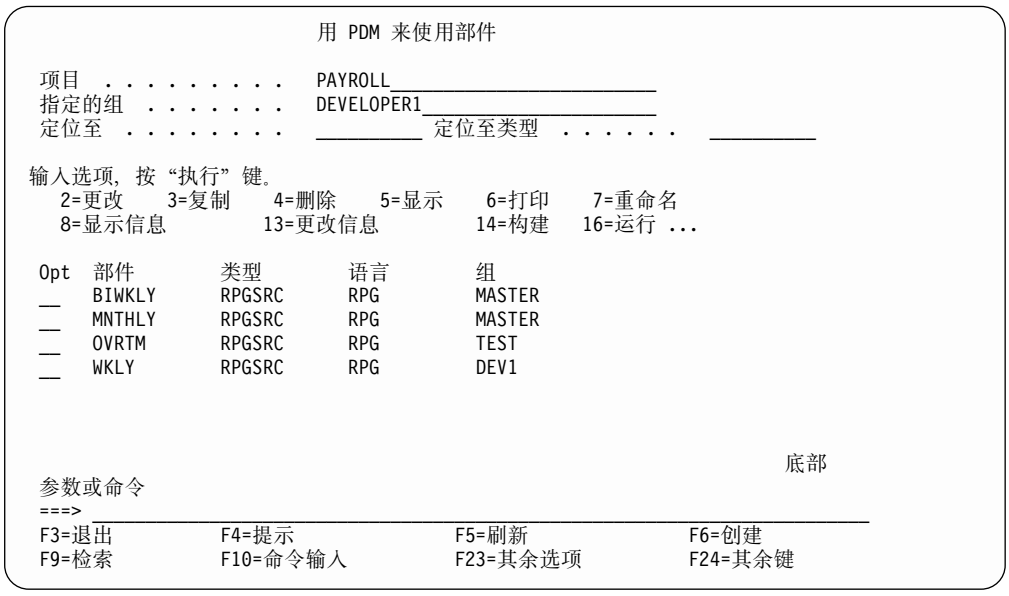

图 72. 用 PDM 来使用部件屏幕

在图72中，指定了组 DEVELOPER1 和搜索路径 *DFT; 然而，所有部件都显示为存在于组 MASTER 中。这表示 DEVELOPER1 中当前没有任何部件，但在组 MASTER 中发现有部件，该组在组 DEVELOPER1 的缺省搜索路径中。

如果想让“用 PDM 来使用部件”屏幕只显示符合特定标准的部件，请按“F17=子集”。出现“子集部件列表”屏幕，如图73所示。

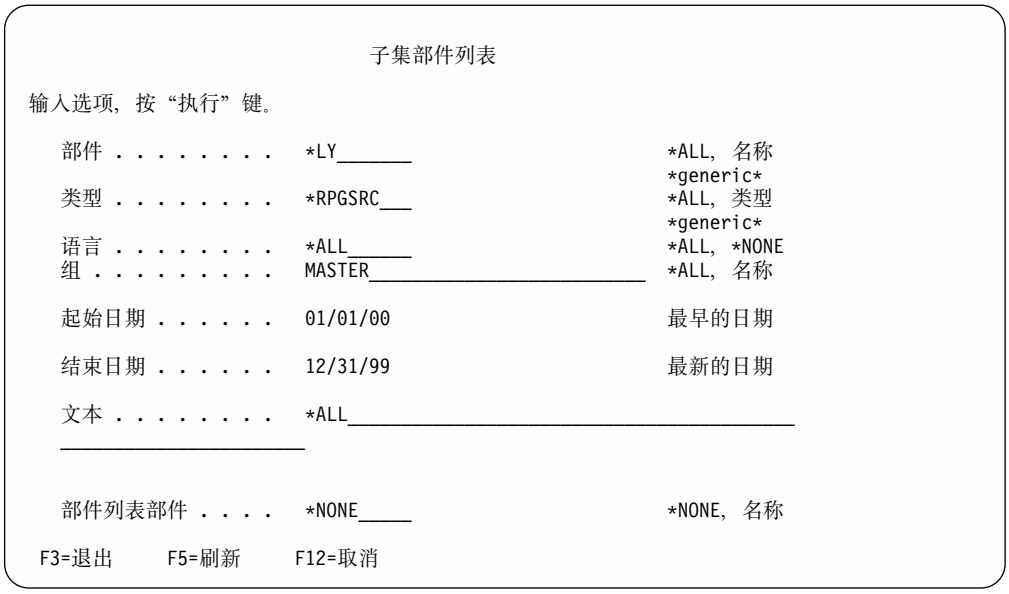

图 73. 子集部件列表屏幕

在此示例中，可指定只想查看类型为 RPGSRC 且名称以字符 LY 结尾并位于组 MASTER 中的部件。

按“执行”键。重新出现“用 PDM 来使用部件”屏幕，现在只显示您指定的那些部件。屏幕底部的消息指示该列表是一个较大列表的子集。参见图74。

还可以在*起始日期*和*结束日期*提示中输入日期以构成一个时间范围，以将子集限制为其上次更改日期在此范围内的那些部件。首次使用此屏幕时，缺省值是 01/01/00 和 12/31/99。您下次使用它时，将显示您在此会话中最近使用的日期。

```
                          用 PDM 来使用部件

 项目  . . . . . .   PAYROLL_____________________
 指定的组  . . . . . .   MASTER_____________________
 定位至  . . . . . .   _________  定位至类型  . . . . . . . . _____________

输入选项，按“执行”键。
   2=更改     3=复制     4=删除     5=显示     6=打印     7=重命名
   8=显示信息          13=更改信息          14=构建    16=运行 ...

 Opt  部件         类型         语言         组
 __   BIWKLY       RPGSRC       RPG          MASTER
 __   MNTHLY       RPGSRC       RPG          MASTER
 __   WKLY         RPGSRC       RPG          MASTER




                                                                    底部
 参数或命令
 ===>_____________________________________________________________________
 F3=退出           F4=提示           F5=刷新           F6=创建
 F9=检索           F10=命令输入      F23=其余选项      F24=其余键
这是一个子集化列表。
```

图 74. 部件列表的子集

如果要打印部件列表，使用“F21=打印列表”。报告被假脱机至此作业的打印设备的输出队列。打印的报告在报告标题中给出项目和组，并在报告主体中列示下列信息：部件、类型、语言、组、日期和文本。

在“用 PDM 来使用部件”屏幕上，有下列附加选项和功能键可用。

```
输入选项，按“执行”键。
  17=使用 SDA 进行更改   18=使用 DFU 进行更改    19=使用 RLU 进行更改
  25=查找字符串          26=打印信息             28=检出     29=检入 ...
```

图 75. 按一次 F23 键后出现的选项

```
输入选项，按“执行”键。
  30=提升     39=导出    45=添加项目库列表    50=转换
  52=使用已归档成员          54=比较          55=合并 ...
```

图 76. 按两次 F23 键后出现的选项

```
参数或命令
===>_____________________________________________________________________
F11=显示部件和类型        F12=取消          F13=重复
F14=显示日期和文本        F23=其余选项      F24=其余键
```

图 77. 按一次 F24 键后出现的功能键

```
参数或命令
===> ____________________________________________________________________________
F15=排序日期      F16=用户选项     F17=子集        F18=更改缺省值
F20=除去项目库列表        F21=打印列表    F24=其余键
```

图 78. 按两次 F24 键后出现的功能键

当您使用此屏幕时，功能键 F11、F14 和 F15 特别有用。它们允许您显示关于部件的不同信息，如日期和文本（而不是语言和组），部件的多个列，也可以按日期或名称排序。

用 PDM 来使用部件列表中的部件

“用 PDM 来使用部件列表中的部件”屏幕允许您处理指定项目和组中的部件列表部件中的部件。如果指定的部件列表部件在指定的组中找不到，则搜索缺省项目层次结构。

当在“指定要处理的部件”屏幕上指定了部件列表的名称时，会出现“用 PDM 来使用部件列表中的部件”屏幕。此屏幕与“用 PDM 来使用部件”屏幕类似。您在“用 PDM 来使用部件”屏幕上可执行的任何操作也可以在此屏幕上执行。

如果通过在此屏幕上的部件前输入选项来使用任何“应用程序开发管理器”命令，且该命令包含部件列表 (PARTL) 参数，则会自动使用您正在处理的部件列表部件的名称填充该参数。

```
                        用 PDM 来使用部件列表中的部件

项目  . . . . . .  PAYROLL_____________________
指定的组  . . . . . .  DEVELOPER1_____________________
定位至  . . . . . .  __________ 定位至类型  . . . . . .  _________

输入选项，按“执行”键。
  2=更改    3=复制    4=删除    5=显示    6=打印    7=重命名
  8=显示信息         13=更改信息         14=构建   16=运行 ...

Opt  部件        类型        语言        组
__   BIWKLY      RPGSRC      RPG         TEST1
__   BIWKL2      RPGSRC      RPG         DEVELOPER1
__   MNTHL2      RPGSRC      RPG         MASTER
__   OVRT2       RPGSRC      RPG         MASTER
__   WKL2        RPGSRC      RPG         MASTER
__   PAY000001   PARTL       *NONE       DEVELOPER1

                                                                      底部
参数或命令
===> ____________________________________________________________________________
F3=退出          F4=提示          F5=刷新          F6=创建
F9=检索          F10=命令输入     F23=其余选项     F24=其余键
```

图 79. “用 PDM 来使用部件列表中的部件”屏幕

注意，显示或刷新“用 PDM 来使用部件列表中的部件”屏幕所需的时间较长。保持部件列表简短可以缩短这段时间。

开发部件

本节提供一个样本方案，它指导您完成一些可以使用“编程开发管理器”实用程序执行的操作:

- 检出部件
- 更改部件
- 复制部件
- 重命名部件
- 转换部件
- 比较部件
- 合并部件
- 删除部件
- 构建部件
- 测试部件
- 检入部件
- 在部件中查找字符串
- 使用已归档成员
- 提升部件
- 更改会话缺省值
- 创建和使用用户定义选项

此示例假定您正在第194页的图72所示的“用 PDM 来使用部件”屏幕中工作，并且您对组 DEVELOPER1 和 DEVELOPER2 具有更新访问权。您将要使用的四个部件名为 RPGSRC BIWKLY、RPGSRC MNTHLY、RPGSRC WKLY 和 RPGSRC OVRTM。它们存在于图80所示的项目层次结构中的组 MASTER 中。

要在您自己的工作站上执行此处描述的操作，请使用项目中的适当部件和组。

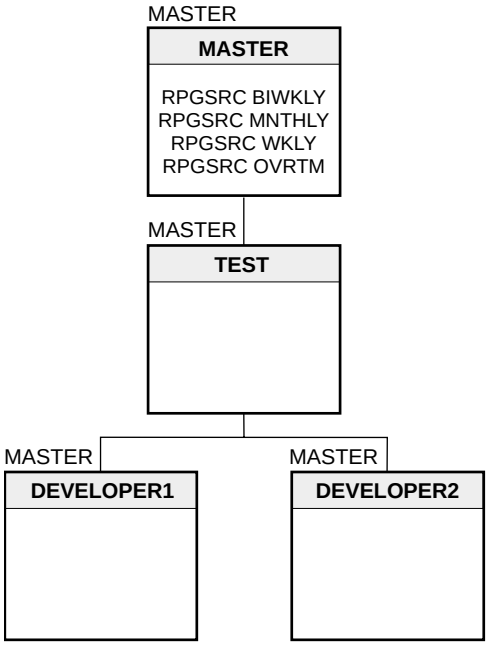

图 80. 样本方案中使用的项目层次结构

检出部件

在将部件检出开发组之前，您可能想对其检查以确保它是正确的。为此，在其名称旁边输入选项 5（显示）。

在此示例中，您将部件 RPGSRC BIWKLY 检出至组 DEVELOPER1。这就是必须在*指定的组*字段中指定的组。在其名称旁边输入 28（检出）。现在，屏幕显示正在将此部件检出至 DEVELOPER1，而且屏幕底部出现一条消息，以确认此操作。（参见图81。）现在，该部件已锁定至您的用户简要表和组。

```
                          用 PDM 来使用部件

 项目  . . . . . .  PAYROLL_____________________
 指定的组  . . . . . .  DEVELOPER1_____________________
 定位至  . . . . . .  __________  定位至类型  . . . . . . . . . _____________

输入选项，按“执行”键。
   2=更改     3=复制     4=删除     5=显示     6=打印     7=重命名
   8=显示信息          13=更改信息          14=构建    16=运行 ...

  Opt  部件         类型         语言         组
  __   BIWKLY       RPGSRC       RPG          DEVELOPER1
  __   MNTHLY       RPGSRC       RPG          MASTER
  __   OVRTY        RPGSRC       RPG          MASTER
       WKLY         RPGSRC       RPG          MASTER



                                                                      底部
  参数或命令
  ===> ____________________________________________________________________________
  F3=退出          F4=提示          F5=刷新          F6=创建
  F9=检索          F10=命令输入     F23=其余选项     F24=其余键
已对您检出部件 BIWKLY。
```

图 81. 用 PDM 来使用部件屏幕

更改部件

在部件 BIWKLY RPGSRC 旁边输入 2（更改）以更改它。如果尚未对您检出该部件，则将会立刻进行此操作。因为该部件已在指定的组中，所以不执行复制。

因为该部件是源部件，所以，您会进入 SEU 编辑会话。在您完成更改并进行保存之后，重新出现“用 PDM 来使用部件”屏幕，并显示一条消息，指示已成功地更改该部件。

如果要更改的部件不是源部件，则会显示适当的 AS/400 更改命令，而不是 SEU 会话。

如果不使用选项 2，则可使用选项 17、18 或 19 来通过使用 SDA、DFU 或 RLU 来更改部件:

- 如果该部件的类型为 DDSSRC，且语言属性为 DSPF，则使用选项 17（使用 SDA 进行更改）
- 如果该部件是一个数据文件，则使用选项 18（使用 DFU 进行更改）
- 如果该部件的类型为 DDSSRC，且语言属性为 PRTF，则使用选项 19（使用 RLU 进行更改）

复制部件

要将部件 BIWKLY 从组 DEVELOPER1 复制至组 DEVELOPER2，请在它的旁边输入 3（复制）。出现“复制部件”屏幕，如图82所示。

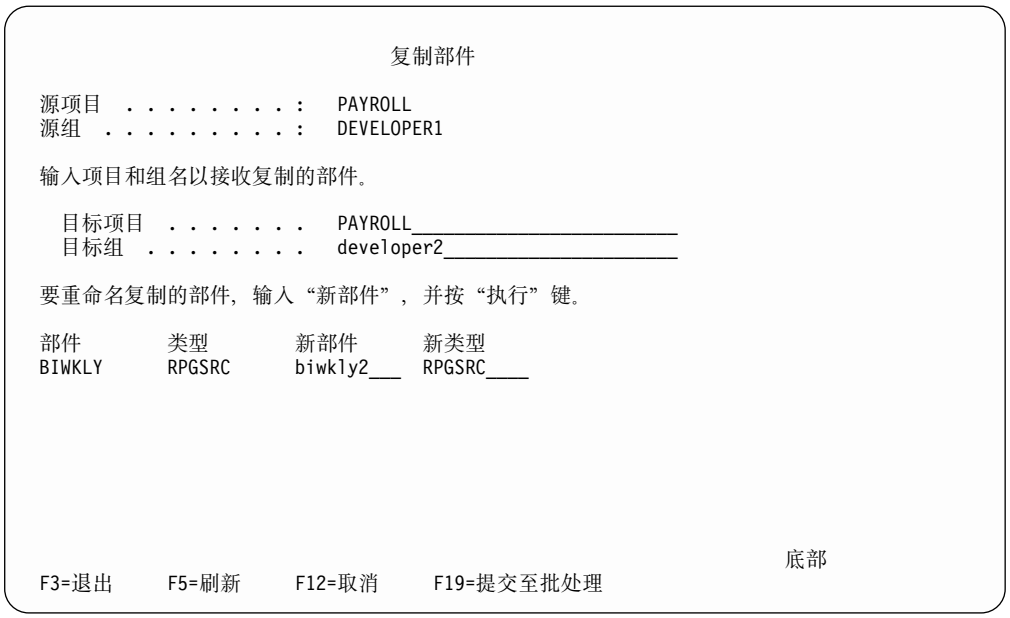

图 82. 指定了“新组和部件”的“复制部件”屏幕

*目标项目*和*目标组*提示的缺省值就是您在*源项目*和*源组*提示中看到的值。如果要将部件复制至另一个组或项目，请更改这些字段中的信息。您必须对正在向其复制部件的组具有更新访问权。

*部件*和*类型*字段标识正在复制的部件。*新部件*和*新类型*字段的缺省值就是您在*部件*和*类型*字段中看到的值。如果要对复制的部件重命名或对其指定另一类型，请更改这些字段中的信息。

在图82中，*目标组*提示已由 DEVELOPER1 更改为 developer2，而*新部件*提示已由 BIWKLY 更改为 BIWKLY2。按“执行”键。复制并重命名该部件。

屏幕底部出现一条信息，指示复制操作是否成功。如果您尝试复制的部件已存在于作为复制目标的组的提升路径中，并且没有对其指定新名称，则复制将失败。任何未能复制的部件都将以反白突出显示。

重命名部件

在“用 PDM 来使用部件”屏幕上，在要在同一个项目和组中重命名的部件旁输入 7（重命名）。

“重命名部件”屏幕列示要重命名的部件。在*新部件*字段中输入新名称，并按“执行”键。您必须对包含正在重命名的部件的组具有更新访问权。

转换部件

在要转换的部件旁边输入 50（转换部件）。

“转换部件”屏幕列示要转换的部件。因为只能转换同一个项目和组中的部件类型，所以您不能更改*项目*和*组*字段中的信息。在*目标类型*字段中输入新类型名，并按“执行”键。转换该部件，并重命名其类型。

比较部件

在“用 PDM 来使用部件”屏幕上，在要比较的部件旁边输入 54（比较部件）。

“比较部件”(CMPPART) 屏幕允许您指定要比较的旧部件的组、类型和部件名。您还可以指定比较时要使用的各种选项。比较结果被放在一个报告中，您可以在工作站上显示该报告，也可以将其假脱机至打印设备。

合并部件

在“用 PDM 来使用部件”屏幕上，在要作为合并目标的部件旁边输入 55（合并部件）。

“合并部件”(MRGPART) 屏幕允许您指定要合并的维护组、维护类型、维护部件、根组、根类型和根部件。

删除部件

假定您复制的部件不对。在 BIWKLY2 旁边输入 4（删除）以删除它。出现“确认删除部件”屏幕，如图83所示。

```
                              确认删除部件

项目  . . . . . :   PAYROLL
组  . . . . . :   DEVELOPER2

按“执行”键以确认您对“删除”的选择。
按“F12=取消”返回，以更改您的选择。

部件        类型        语言        组
BIWKLY2     RPGSRC      RPG         DEVELOPER2

                                                                  底部
F12=取消          F19=提交至批处理
```

图 83. 确认删除部件屏幕

此屏幕列示您已指示要删除的部件。按“执行”键以删除该部件，或按“F12=取消”。

如果正在删除许多个部件，则可使用“F19=提交至批处理”以将所有删除操作提交至多个批处理作业。这允许您在删除部件时执行其他任务。

构建部件

在要构建的部件旁输入 14（构建）。

如果正在对仅仅一个部件进行多次更改，并且您不想编译相关部件，则在“用 PDM 来使用部件”屏幕上按“F18=更改缺省值”以转至“更改缺省值”屏幕，然后将*构建范围*提示更改为“2=受限”，而不是使用缺省值（1=正常）。有关详情，参见第204页的『更改会话缺省值』。

在构建部件之后，您可以检查构建报告以获取构建警告消息（如果有的话）。第136页的图49中描述了构建报告。附录E. 构建报告消息列示了构建消息。

测试部件

在成功构建部件之后，您应对其进行测试。可使用 ADDPRJLIBL 命令将项目的各个库添加至会话库列表的用户部分的开头。有关 ADDPRJLIBL 命令的描述，参见第157页的『将项目库添加至库列表』。

选项 16（运行）将运行该部件。此选项对类型为 PGM、REXXSRC 和 CMD 的部件有效。

您可以使用在 AS/400 库列表中添加或除去项目库的两个用户定义选项。有关这两个选项的描述，参见第208页的『添加和除去项目库』。“使用组”和“使用部件”屏幕中的选项 45（添加项目库列表）和选项 20（除去项目库列表）用于在库列表中添加或除去项目库。

检入部件

在成功地创建或更改部件，构建部件和运行部件之后，您将重新将其检入，以使其可供对相同组具有更新访问权的其他开发者使用。要将其检入，请在它的名称旁边输入 29（检入）。您接收到一条消息，指示已成功地检入部件。

提升部件

在您使用完部件之后，请输入 30（提升）以将其提升至项目层次结构中的下一更高级别的组。

您接收到一条消息，指示已成功地提升该部件，并且，*组*字段更新为显示该部件目前所驻留的组的名称。如果尚未检入该部件，则会在提升该部件之前自动完成此操作。

在部件中查找字符串

“查找字符串部件”(FNDSTRPART) 命令允许您在“应用程序开发管理器”部件中搜索包含源或文件数据的字符或十六进制字符串。如果找到该字符串的匹配项，则您可以对该部件使用任何“编程开发管理器”选项或您自己的用户定义选项之一。FNDSTRPART 命令与“查找字符串 PDM”(FNDSTRPDM) 命令类似；然而，只能通过“用 PDM 来使用组”和“用 PDM 来使用部件”屏幕来使用它。

您可以从这两个屏幕中使用选项“25=查找字符串”，也可以从“编程开发管理器”实用程序外部使用 FNDSTRPART 命令。在“使用组”屏幕中，搜索指定组中的所有部件。在“使用部件”屏幕中，只搜索指定的部件。您在“更改缺省值”屏幕上的*扫描*

*层次结构*提示上指定的值确定是否搜索项目层次结构。查看第204页的『更改会话缺省值』以获取*扫描层次结构*提示值。（有关 FNDSTRPDM 命令的详情，参见 *ADTS/400: Programming Development Manager*。）

示例

以下命令在项目 PAYROLL 中从组 DEVELOPER1 开始在所有类型的所有部件中的列 2 至 4 中搜索字符串 'ADDR'。找到该字符串时，提示 CHECKOUT 命令，以便您可以编辑包含 'ADDR' 的部件。打印包含该字符串的部件的列表。包含该字符串的头两个记录以字符格式打印。不标记该字符串。如果记录的长度超过行的长度，则会将记录截断。

```
FNDSTRPART STRING('ADDR') PRJ(PAYROLL) GRP(DEVELOPER1) TYPE(*ALL)
           PART(*ALL) OPTION(*CHG *PROMPT) COL(2 4) PRTPARTLST(*YES)
           PRTRCDS(2 *CHAR *NOMARK *TRUNCATE) SCAN(*YES)
```

要获取“查找字符串”屏幕上的每个提示或 FNDSTRPART 命令上的每个字段的详细说明:

- 在“查找字符串”屏幕上，在您想了解关于其信息的任何屏幕区域上，按“F1=帮助”
- 提示 FNDSTRPART 命令（F4=提示），并在您想了解关于其信息的屏幕区域上按“F1=帮助”

使用归档成员

在“用 PDM 来使用部件”屏幕上，在要使用的源部件旁边输入 52（使用归档成员）。

“用 PDM 来使用成员”屏幕 (第204页的图84) 列示该部件的归档成员。这些成员是在更改此部件、将此部件导入此组或将其提升至此组时指定了 ARCHIVE(*YES) 的情况下创建的。

要恢复或回滚归档成员，请在要回滚的部件旁边输入 IM（导入部件）。

```
                          用 PDM 来使用成员

文件  . . . . . .   BIWKLY____
  库  . . . . . .     PAY_TST1___          定位至 . . . . . . . .  __________

输入选项，按“执行”键。
 2=编辑          3=复制  4=删除    5=显示          6=打印      7=重命名
 8=显示描述      9=保存 13=更改文本  14=编译      15=创建模块...

选项  成员        类型        文本
__    "archive1"  _________  ________________________________________________




                                                                        底部
 参数或命令
===> ____________________________________________________________________________
 F3=退出          F4=提示          F5=刷新          F6=创建
 F9=检索          F10=命令输入     F23=其余选项     F24=其余键
```

图 84. 用 PDM 来使用成员屏幕

注: 如果您在使用归档成员时更改了任何会话缺省值，则不会记下那些新的更改。当您返回至“用 PDM 来使用部件”屏幕时，会话缺省值与您输入选项 52（使用已归档成员）之前的缺省值没有任何不同。

更改会话缺省值

本节描述“更改缺省值”屏幕上的“*扫描层次结构*”、“*搜索路径*”、“*刷新部件列表*”和“*只显示源部件*”提示。它们允许您指示是否应该在项目层次结构中搜索部件、用来查找该部件的搜索路径以及要用来构建部件的构建范围。要更改它们，您必须更改会话缺省值。

使用“用 PDM 来使用部件”屏幕上的“F18=更改缺省值”转至“更改缺省值”屏幕，如第205页的图85和第205页的图86所示。

```
                              更改缺省值

输入选项，按“执行”键。

  对象库 . . . . . .  *SRCLIB___   名称，*CURLIB，*SRCLIB
  替换对象 . . . . . .  N           Y=是，N=否
  批处理编译 . . . . . .  Y         Y=是，N=否
  批处理运行 . . . . . .  N         Y=是，N=否
  保存会话缺省值 . . . . . .  Y     Y=是，N=否
  保存/恢复选项 . . . . . .  1      1=单个，2=全部
  作业描述 . . . . . .  QBATCH____  名称，*USRPRF，按 F4 以获取列表
    库 . . . . . .  *LIBL_____  名称，*CURLIB，*LIBL
  更改类型和文本 . . . . . .  Y     Y=是，N=否
  选项文件 . . . . . .  TESTFILE__  名称
    库 . . . . . .  *LIBL_____  名称，*CURLIB，*LIBL
  成员 . . . . . .  TESTMBR___  名称
  全屏幕方式 . . . . . .  N         Y=是，N=否
  扫描层次结构 . . . . . .  Y      Y=是，N=否
  搜索路径 . . . . . .  *DFT_____  名称，*DFT
  构建范围 . . . . . .  2           1=正常，2=受限，3=扩展，
                                    4=直接链
                                                          尚有...
F3=退出      F4=提示      F5=刷新      F12=取消
```

图 85. 更改缺省值屏幕 - 第 1 部分

```
                              更改缺省值

输入选项，按“执行”键。

  记录选项命令 . . . . . .  N       Y=是，N=否
  按“执行”键时退出列表 . . .  Y     Y=是，N=否
  刷新部件列表 . . . . . .  Y       Y=是，N=否
  只显示源部件 . . . . . .  Y       Y=是，N=否
```

图 86. 更改缺省值屏幕 - 第 2 部分

以下列表描述“更改缺省值”屏幕上的*扫描层次结构*、*搜索路径*和*构建范围*提示。*扫描层次结构*和*搜索路径*提示值由调用带有 SCAN 和 SCHPTH 参数的命令的任何选项使用。查看第103页的『如何处理搜索路径部件』以获取扫描和搜索路径参数如何配合工作以搜索部件的说明。

您在屏幕上选择使用搜索路径的选项时，将确认*搜索路径*提示中指定的值。您可能需要针对正在执行的特定任务更改该值。

扫描层次结构

此值确定哪些部件出现在“用 PDM 来使用部件”屏幕上，以及哪些组出现在“用 PDM 来使用组”屏幕上。您在这些屏幕上选择的各种选项也使用此值。缺省值是“Y=是”，这表示在缺省搜索路径中的组中搜索部件。

“N=否”指定只搜索指定的组。

搜索路径

此值确定“应用程序开发管理器”选项使用的搜索路径并显示指定搜索路径的组中包含的部件。缺省值是 *DFT，您也可以指定搜索路径部件的名称。

如果指定特定搜索路径，则可以查看无法通过缺省搜索路径提供的部件版本。如果您希望使用 SCHPTH QDFT 部件中指定的搜索路径来查找部件，则必须在此提示上指定 QDFT。如果在此提示上指定 *DFT，且有一个 SCHPTH QDFT 存在于层次结构中，则不会使用它来确定“用 PDM 来使用部件”和“用 PDM 来使用部件列表中的部件”屏幕的部件列表。然而，所有“应用程序开发管理器”部件开发命令都使用 SCHPTH QDFT。

如果指定跨项目执行搜索的搜索路径部件，则“编程开发管理器”实用程序忽略不属于搜索路径中指定的项目的任何组。对于“用 PDM 来使用部件”和“用 PDM 来使用部件列表中的部件”屏幕上列示的部件，当在此屏幕上更改搜索路径值时，它们会自动刷新。

指定的搜索路径并不确定“用 PDM 来使用组”屏幕上显示的组列表。

构建范围

缺省值是“1=正常”。此提示与 BLDPART 命令上的SCOPE 参数相对应。构建指定的部件、构建它所依赖的所有其他部件、构建那些部件所依赖的部件，等等。

有关详情，参见第129页的『设置构建范围』。

刷新部件列表

此提示指定在处理“处理部件”屏幕上的选项之后是否刷新部件列表。例如，如果在此提示上指定“Y=是”，并从显示的部件列表中删除了一个部件，则部件列表会再次显示，并且没有已删除的部件。

“Y=是”指定在处理“处理部件”屏幕上的选项之后需刷新部件列表。

“N=否”指定在处理“处理部件”屏幕上的选项之后不刷新部件列表。

只显示源部件

此提示指定是将指定项目和组中的所有部件都显示在部件列表中还是只将指定项目和组中的源部件显示在部件列表中。

“Y=是”指定只将指定项目和组中的源部件显示在部件列表中。

“N=否”指定将指定项目和组中的所有部件显示在部件列表中。

除非使用用户定义选项，否则*对象库*、*替换对象*和*更改类型和文本*提示被忽略。然而，无论您是否正在使用“应用程序开发管理器”功能部件，所有缺省值都可以随时更改。

用户定义选项

除了屏幕上的选项之外，您可以创建您自己的选项，即**用户定义选项**。这些选项允许您简单地通过在屏幕上输入一个选项来执行您频繁执行的操作，因而非常有用。

您选择的要与用户定义选项相对应的命令可以是任何 AS/400 系统或用户命令。它可以包含参数变量，以便可以对列表中的项执行该命令。这使您不必指定（与命令相对应的）选项、提示选项和为参数指定值；它还允许您执行您对列表中的项选择的命令。

创建用户定义选项

以下示例创建一个名为 EP 的用户定义选项，该选项调用一个名为 EDITPART 的 CL 程序。这就是第87页的『检索部件信息 (RTVPARTINF)』中的 CL 程序。该程序检出一个部件、检索编辑该部件所必需的 AS/400 信息、启动编辑器，然后再将该部件检入。

在创建此用户定义选项之后，您就可从“处理部件”屏幕中使用它:

1. 如果您在“用 PDM 来使用部件”屏幕上，请按“F16=使用用户选项”以转至“使用用户定义选项”屏幕。如果您在主菜单上，请选择选项 9（使用用户定义选项）以显示“指定要使用的选项文件”屏幕。指定要使用的选项文件，并按“执行”键。
2. 在“使用用户定义选项”屏幕上按“F6=创建”。出现“创建用户定义选项”屏幕，如图87所示。

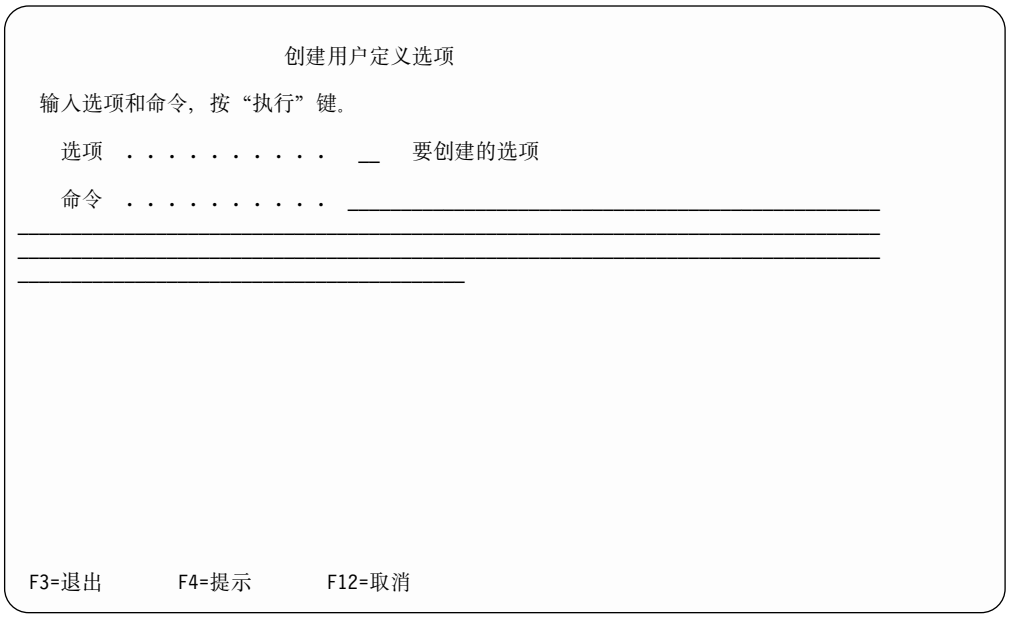

图 87. 创建用户定义选项屏幕

3. 在*选项*提示中，输入表示想要让新选项调用的命令的 1 或 2 个字符。第一个字符必须是字母；第二个字符可以是字母或数字。

   对于此示例，在*选项*提示中输入 EP。

4. 在*命令*提示中，请输入要在 EP 选项被选中时要调用的命令。如果您记不住命令的正确名称，请按“F4=提示”以显示一个菜单，该菜单允许您显示所有系统命令或特定类型的命令。如果您知道命令的名称，但想要查看其参数，则输入命令名，并按“F4=提示”查看其提示屏幕。

   对于此示例，在*命令*提示中输入以下命令。

   ```
   CALL EDITPART (&ZP &ZG &ZT &ZN)
   ```

   此命令将调用第87页的『检索部件信息 (RTVPARTINF)』中的 EDITPART CL 程序。

替换变量解析如下:

&ZP 项目的名称

&ZG 组的名称

&ZT 部件的类型

&ZN 部件的名称

您也可以使用单字符替换变量 &A、&B、&D、&F、&L、&N、&O、&R、&S、&T 和 &X 来创建用户定义选项。它们解析为与“使用”屏幕上列示的“应用程序开发管理器”项目、组或部件相对应的 AS/400 对象或成员。这允许您使用当前的用户定义选项来执行“应用程序开发管理器”功能。查看附录D. 替换变量以获取所有替换变量的描述。

使用用户定义选项

导入 OS/400 对象或源成员

IO（导入对象）和 IM（导入成员）这两个由系统提供的用户定义选项可以帮助您将 OS/400 对象导入“应用程序开发管理器”项目层次结构，在该层次结构中，它们变为部件。通过将这些选项与 “F13=重复”配合使用，一次可以导入许多对象。IO 和 IM 选项包括在库 QPDA 中由系统提供的用户定义选项文件 QAUOUSR 中。

从“使用对象”屏幕使用 IO。

```
IO  IMPPART OBJ(&L/&N) OBJTYPE(&T) TYPE(&S) PART(&N) TEXT(&X)
```

从“使用成员”屏幕使用 IM。

```
IM
   IMPPART OBJ(&L/&F) OBJTYPE(*FILE) MBR(&N) PART(&N) LANG(&S) TEXT(&X)
```

当您选择其中一个选项时，会出现一个提示，您可以在该提示上输入项目名、组、类型以及 IMPPART 命令需要的其他信息。

“使用组”屏幕中的“选项 38”（导入）提供与 IM 和 IO 用户定义选项相同的功能。

添加和除去项目库

另外两个由系统提供的用户定义选项是 AP 和 RP。它们用来在您测试部件时在库列表中添加和除去项目库。AP 和 RP 选项的外观为:

```
AP  ADDPRJLIBL PRJ(&ZP) GRP(&ZG) SCAN(&ZH) SCHPTH(&ZS)

RP  RMVPRJLIBL
```

要使用它们:

1. 在列表中的任何部件旁边输入 AP。

   通过使用 ADDPRJLIBL 命令，将搜索路径中的项目库添加至库列表。
2. 使用 16（运行）选项来运行部件。当该部件完成时，返回至“用 PDM 来使用部件”屏幕。
3. 在列表中的任何部件旁边输入 RP。

   通过使用 RMVPRJLIBL 命令，从库列表中除去项目库。

   “处理部件”屏幕上的选项 45（添加项目库列表）和“F20=除去项目库列表”分别提供与 AP 和 RP 相同的功能。

第17章 使用 VisualAge RPG 部件

“应用程序开发管理器”功能部件允许在“应用程序开发管理器”项目中存储和管理 VisualAge RPG 应用程序。为此，提供了两种部件类型: VRPGTXT 和 VRPGBIN。

VRPGTXT 部件类型存储在源物理文件中。该部件类型的语言将是 VRPG。“应用程序开发管理器”功能部件将把此部件视为单一实体。VisualAge RPG 文本对象是作为成员存储的，其成员类型设置为它们的工作站文件扩展名。

VRPGBIN 部件类型存储在数据文件中。该部件类型的语言将是 VRPG。此部件类型将包括 VisualAge RPG 二进制对象。

下列“应用程序开发管理器”部件开发命令支持这些部件类型:
- 更改部件 (CHGPART)
- 更改部件信息 (CHGPARTINF)
- 检入部件 (CHKINPART)
- 检出部件 (CHKOUTPART)
- 比较部件 (CMPPART)
- 复制部件 (CPYPART)
- 删除部件 (DLTPART)
- 显示部件 (DSPPART)
- 导出部件 (EXPPART)
- 导入部件 (IMPPART)
- 打印部件信息 (PRTPARTINF)
- 提升部件 (PRMPART)
- 查询部件 (QRYPART)
- 回收项目 (RCLPRJ)
- 检索部件信息 (RTVPARTINF)

下列“应用程序开发管理器”部件开发命令只支持 VRPGTXT 部件类型:
- 合并部件 (MRGPART)

虽然高性能文件系统 (HPFS) 上的目录（或 VisualAge RPG 对象的组件名）可能非常长，但在“应用程序开发管理器”环境中，只允许长度为 10 个字符的部件名。VisualAge RPG 用户可以使用部件文本字段来存储其他信息。

查看文献目录以获取 VisualAge RPG 出版物的列表。

导入 VisualAge RPG 部件

在“应用程序开发管理器”中创建 VisualAge RPG 部件的唯一方法是使用 IMPPART 命令。要导入 VisualAge RPG 部件，必须在 IMPPART 命令的 TYPE 参数上指定 VRPGBIN 或 VRPGTXT。这是必需的，原因是 IMPPART 命令在正常情况下将源文件中的个别成员作为部件导入。虽然 VRPGTXT 部件存储在源文件中，但它会作为单一实体导入。

要导入 VisualAge RPG 部件，需要使用“VisualAge RPG 客户机”功能部件将 VisualAge RPG 项目的一个组件保存到 iSeries 400 系统上的库中。您可以使用 QTEMP 库（如果您想临时保存的话）。然后，您可以发出带有 REPLACE(*YES) 的 IMPPART 命令来单独地导入所有 VisualAge RPG 源和二进制对象。导入功能将检出部件（如果有必要的话），然后重新将它们检入。如果某个部件不存在，导入功能将自动创建该部件。

由于 VisualAge RPG 功能部件施加的限制，您在导入 VisualAge RPG 部件时，不能更改它们的名称。

导出 VisualAge RPG 部件

如果要在工作站上恢复 VARPG 项目以进行查看或更改，则必须首先将适当的 VRPGTXT 和 VRPGBIN 部件导出到一个库中。可将 QTEMP 库用作 EXPPART 命令上的目标库。然后，可使用 VisualAge RPG 功能部件来在工作站上恢复 VARPG 部件。

如果只是想查看 VARPG 项目，则不需要在发出 EXPPART 命令之前检出这些部件。然而，如果您计划更改部件，则必须在发出 EXPPART 命令之前检出部件。

构建 VisualAge RPG 部件

要构建 VisualAge RPG 部件，请执行下列步骤:

1. 检出 VisualAge RPG 部件
2. 导出部件
3. 将部件恢复至工作站
4. 调用 VisualAge RPG 图形用户界面 (GUI) 来构建部件
5. 将部件保存至主机
6. 将经过更改的部件导入到“应用程序开发管理器”项目中，并指定 REPLACE(*YES)
7. 重新将部件检入

如果上面提到的任何步骤失败，则构建 VisualAge RPG 部件也会失败。

VisualAge RPG 部件开发

要维护 VisualAge RPG 部件的完整性，应同时更新 VRPGTXT 部件及其相关联的 VRPGBIN 部件。这表示您应当始终同时检出、构建、导出或导入 VRPGTXT 部件及其相关联的 VRPGBIN 部件。

如果您删除了一个 VisualAge RPG 部件，请记得还要删除它的相关部件。例如，如果您删除 VRPGBIN 部件，则还必须删除 VRPGTXT 部件，反之亦然。

第18章 使用 System/36 和 System/38 应用程序

本章描述:
- 使用 System/36 应用程序
- 构建 System/36 应用程序
- System/38 应用程序的注意事项

使用 System/36 应用程序

“应用程序开发管理器”功能部件直接支持 System/36 应用程序。您可以使用表19中显示的 System/36 部件类型，也可以定义您自己的部件类型。

表 19. System/36 部件类型和语言

System/36 部件类型	语言
CBL36INC	CBL36
CBL36SRC	CBL36
DSPF36SRC	DSPF36
MNU36SRC	MNU36
MSGF36SRC	MSGF36
OCL36SRC	OCL36
RPG36INC	RPG36
RPG36SRC	RPG36
RPT36SRC	RPT36
SRT36SRC	SRT36
TXT36SRC	TXT36

有关允许在 TYPE 参数上指定 System/36 部件类型的命令的描述，参见第215页的『附录B. 部件类型以及它们与命令的关系』。

您必须将上表中未提及的所有 System/36 语言的部件类型（如 BAS36）定义为用户定义部件类型。

部件类型 *PGM 可配合部件语言 CBL36 和 RPG36 使用。

以下“应用程序开发管理器”命令在 LANG 参数上支持 System/36 语言:
CHGPARTINF
CRTPART
EXPPART
IMPPART

构建 System/36 应用程序

不能使用 BLDPART 命令构建 System/36 对象，这是因为 System/36 编译器不会返回构建过程所需的信息。

要构建 System/36 应用程序，执行下列步骤:

1. 使用 ADDPRJLIBL 命令设置库列表。
2. 使用恰当的编译器命令来编译源部件。为此，您将需要指定要构建的源部件的全限定本机名。可在 CL 程序中使用 RTVPARTINF 命令，或在 WRKPARTPDM 屏幕上使用用户定义选项来检索此信息。
3. 必须在“应用程序开发管理器”控制范围之外的库中创建输出对象。
4. 使用带 REPLACE(*YES) 的 IMPPART 命令将输出对象导入项目中。
5. 使用 RMVPRJLIBL 来恢复库列表。

System/38 应用程序的注意事项

虽然，“应用程序开发管理器”功能部件不能直接支持 System/38 应用程序，但是，您可通过使用用户定义的部件类型来管理这些应用程序。一旦定义了这些类型，就可利用“应用程序开发管理器”功能部件的更改管理和版本控制功能。

除 BLDPART 命令外，所有其他“应用程序开发管理器”部件命令都可用于存储在具有用户定义部件类型的部件中的 System/38 源。

因为 System/38 编译器未为“应用程序开发管理器”功能部件提供构建关系信息，所以您将不能直接使用 BLDPART 命令构建这些应用程序。

附录A. 应用程序开发管理器控制语言命令

可在 OS/400 命令行中使用所有“应用程序开发管理器”控制语言 (CL) 命令。并且，还可通过在下列其中一个 PDM 屏幕上选择选项或按功能键来使用这些命令中的许多命令:

用 PDM 来使用项目
用 PDM 来使用组
用 PDM 来使用部件

表20列示了所有“应用程序开发管理器” CL 命令。有关使用 PDM 屏幕的信息，参见第16章 使用编程开发管理器实用程序。

表 20. 应用程序开发管理器 CL 命令

CL 命令	描述
ADDADMLANG[1]	添加用户定义类型的语言
ADDADMTYPE[1]	添加用户定义类型
ADDPRJLIBL	将项目库添加至 AS/400 库列表
ADDPRJUSR[2]	授予开发者或管理员对指定项目的权限
BLDPART	构建部件
CHGADMACN[1]	更改用户定义类型的操作定义
CHGADMLANG[1]	更改用户定义类型的语言
CHGGRP[2]	更改组的特征
CHGPART[3]	更改现存的部件
CHGPARTINF	更改有关现存部件的信息
CHGPRJ[2]	更改项目的特征
CHGPRJUSR[2]	更改对项目具有权限的开发者或管理员的特征
CHKINPART	释放检出的部件并使它可供其他开发者使用
CHKOUTPART	将部件复制至开发组（如果它不存在于开发组中的话），并锁定它以防止其他开发者访问它
CMPPART	比较两个或多个部件
CPYPART	创建基于现存部件的新部件
CRTGRP[2]	创建新组
CRTPART	创建新部件
CRTPRJ	创建项目
CVTPART	将部件类型和语言转换为另一部件类型和语言
DLTGRP[2]	删除组及其内容
DLTPART	从指定的组中删除部件
DLTPRJ[2]	删除项目及其内容
DSPPART	显示部件的内容或特征
EXPPART	将部件从项目层次结构复制到 OS/400 库中
FNDSTRPART	在一个部件或一组部件中查找特定字符串

表 20. 应用程序开发管理器 CL 命令 (续)

CL 命令	描述
IMPPART	复制 OS/400 成员或对象并将它作为“应用程序开发管理器”部件存储在项目层次结构中
MRGPART	合并对一个或多个部件所做的两组更新。
PRMPART	将部件提升至层次结构中的下一组
PRTADMLANG	打印用户定义类型的语言信息
PRTADMTYPE	打印有关用户定义类型的信息
PRTPARTINF	打印有关特定部件的信息
PRTPRJ	打印有关项目的信息
PRTPRJLOG	打印项目审计记录的内容
PRTPRJUSR	打印项目的授权开发者和管理员的列表
QRYPART	查询部件
QRYPRJ	查询项目
RCLPRJ[2]	在恢复操作之后或在系统故障之后回收项目
RCVPART	从远程系统接收入局部件
RMVADMLANG[1]	除去用户定义类型的语言
RMVADMTYPE[1]	除去用户定义类型
RMVPRJLIBL	从 OS/400 库列表中除去项目库
RMVPRJUSR[2]	从项目中除去开发者或管理员
RNMPART	重命名部件
RTVPARTINF[4]	检索特定“应用程序开发管理器”部件的 OS/400 成员或对象名
WRKGRPPDM[3]	用 PDM 来使用组列表
WRKPARTPDM[3]	用 PDM 来使用部件列表
WRKPRJPDM[3]	用 PDM 来使用项目列表

注意:

1. ADDADMLANG、ADDADMTYPE、CHGADMACN、CHGADMLANG、RMVADMLANG 和 RMVADMTYPE 命令需要 *ALLOBJ 权限。
2. ADDPRJUSR、CHGGRP、CHGPRJ、CHGPRJUSR、CRTGRP、DLTGRP、DLTPRJ、RCLPRJ 和 RMVPRJUSR 命令仅可供项目管理员使用。
3. 只能在交互式环境中处理此命令，当在命令输入提示上输入或从 QCMDEXC 中调用此命令时，它作为 CL 程序的一部分或 REXX 过程的一部分接受处理。
4. 在批处理或交互式环境中，此命令仅作为 CL 程序或 REXX 过程的部件运行。

附录B. 部件类型以及它们与命令的关系

使用下列各表来确定可对包含 TYPE 参数的“应用程序开发管理器”命令指定 受“应用程序开发管理器”支持的类型（y=是，可指定，<blank>=否，不能指定）。

表 21. 应用程序开发管理器类型以及它们与 CL 命令的关系（共 8 部分，第 1 部分）

命令	RPGSRC	RPGLESRC	RPGINC	RPGLEINC	CBLSRC	CBLLESRC	CBLINC
BLDPART	y	y	y[1]	y	y	y	y[1]
CHGPART	y	y	y	y	y	y	y
CHGPARTINF	y	y	y	y	y	y	y
CHKINPART	y	y	y	y	y	y	y
CHKOUTPART	y	y	y	y	y	y	y
CMPPART	y	y	y	y	y	y	y
CPYPART	y	y	y	y	y	y	y
CRTPART	y	y	y	y	y	y	y
CVTPART	y	y			y	y	
DLTPART	y	y	y	y	y	y	y
DSPPART	y	y	y	y	y	y	y
EXPPART	y	y	y	y	y	y	y
FNDSTRPART[2]	y	y	y	y	y	y	y
IMPPART	y	y	y	y	y	y	y
MRGPART	y	y	y	y	y	y	y
PRMPART[3]	y	y	y	y	y	y	y
PRTPARTINF	y	y	y	y	y	y	y
QRYPART	y	y	y	y	y	y	y
RNMPART	y	y	y	y	y	y	y
RTVPARTINF	y	y	y	y	y	y	y
WRKPARTPDM	y	y	y	y	y	y	y

注意:

下列各项适用于表中列示的部件类型:

1. 并不实际地构建消息文件和包含部件，但您正在构建的部件可以引用它们。对于类型为 BNDDIR 的部件，情况也是这样；然而，如果您在 BLDPART 命令上指定 SCOPE(*EXTENDED)，则可以构建它们。有关详情，参见第129页的『设置构建范围』。
2. FNDSTRPART 命令对存储在源文件中的由系统提供的类型或用户定义类型有效，并对 FILE(PF) 部件有效。
3. 一般说来，如果在 PRMPART 命令上指定 EXTEND(*YES)，则可以提升对象。

表 22. 应用程序开发管理器类型以及它们与 CL 命令的关系（共 8 部分，第 2 部分）

命令	CBLLEINC	CSRC (C)	C S R C (CLE)	CINC (C)	CINC (CLE)	CMDSRC	CLSRC
BLDPART	y		y		y	y	
CHGPART	y	y	y	y	y	y	y
CHGPARTINF	y	y	y	y	y	y	y
CHKINPART	y	y	y	y	y	y	y
CHKOUTPART	y	y	y	y	y	y	y
CMPPART	y	y	y	y	y	y	y
CPYPART	y	y	y	y	y	y	y
CRTPART	y	y	y	y	y	y	y
CVTPART		y	y		y		
DLTPART	y	y	y	y	y	y	y
DSPPART	y	y	y	y	y	y	y
EXPPART	y	y	y	y	y	y	y
FNDSTRPART[1]	y	y	y	y	y	y	y
IMPPART	y	y	y	y	y	y	y
MRGPART	y	y	y	y	y	y	y
PRMPART[2]	y	y	y	y	y	y	y
PRTPARTINF	y	y	y	y	y	y	y
QRYPART	y	y	y	y	y	y	y
RNMPART	y	y	y	y	y	y	y
RTVPARTINF	y	y	y	y	y	y	y
WRKPARTPDM	y	y	y	y	y	y	y

注意:

下列各项适用于表中列示的部件类型:

1. FNDSTRPART 命令对存储在源文件中的由系统提供的类型或用户定义类型有效，并对 FILE(PF) 部件有效。
2. 一般说来，如果在 PRMPART 命令上指定 EXTEND(*YES)，则可以提升对象。

表 23. 应用程序开发管理器类型以及它们与 CL 命令的关系（共 8 部分，第 3 部分）

命令	CLLESRC	CLPSRC	DDSSRC	REXXSRC	CLDSRC	TXTSRC	BLDOPT
BLDPART	y	y	y		y		
CHGPART	y	y	y	y	y	y	y
CHGPARTINF	y	y	y	y	y	y	y
CHKINPART	y	y	y	y	y	y	y
CHKOUTPART	y	y	y	y	y	y	y
CMPPART	y	y	y	y	y	y	y
CPYPART	y	y	y	y	y	y	y
CRTPART	y	y	y	y	y	y	y
CVTPART	y	y				y	
DLTPART	y	y	y	y	y	y	y
DSPPART	y	y	y	y	y	y	y
EXPPART	y	y	y	y	y	y	y
FNDSTRPART[1]	y	y	y	y	y	y	y
IMPPART	y	y	y	y	y	y	y
MRGPART	y	y	y	y	y	y	y
PRMPART[2]	y	y	y	y	y	y	y
PRTPARTINF	y	y	y	y	y	y	y
QRYPART	y	y	y	y	y	y	y
RNMPART	y	y	y	y	y	y	y
RTVPARTINF	y	y	y	y	y	y	y
WRKPARTPDM	y	y	y	y	y	y	y

注意:

下列各项适用于表中列示的部件类型:

1. FNDSTRPART 命令对存储在源文件中的由系统提供的类型或用户定义类型有效，并对 FILE(PF) 部件有效。
2. 一般说来，如果在 PRMPART 命令上指定 EXTEND(*YES)，则可以提升对象。

表 24. 应用程序开发管理器类型以及它们与 CL 命令的关系（共 8 部分，第 4 部分）

命令	SCHPTH	PGM	FILE	CLD	CMD	MSGF	DTAARA	BNDSRC
BLDPART		y[2]	y	y	y	y[1]		y
CHGPART	y	y	y		y	y	y	y
CHGPARTINF	y	y	y	y	y	y	y	y
CHKINPART	y	y	y	y	y	y	y	y
CHKOUTPART	y	y	y	y	y	y	y	y
CMPPART	y		y[5]					y
CPYPART	y	y	y	y	y	y	y	y
CRTPART	y					y	y	y
CVTPART		y						
DLTPART	y	y	y	y	y	y	y	y
DSPPART	y	y	y		y	y	y	y
EXPPART	y	y	y	y	y	y	y	y
FNDSTRPART[3]	y		y					y
IMPPART	y	y	y	y	y	y	y	y
MRGPART	y							
PRMPART[4]	y					y	y	y
PRTPARTINF	y	y	y	y	y	y	y	y
QRYPART	y	y	y	y	y	y	y	y
RNMPART	y	y	y	y	y	y	y	y
RTVPARTINF	y	y	y	y	y	y	y	y
WRKPARTPDM	y	y	y	y	y	y	y	y

注意:

下列各项适用于表中列示的部件类型:

1. 并不实际地构建消息文件和包含部件，但您正在构建的部件可以引用它们。对于类型为 BNDDIR 的部件，情况也是这样；然而，如果您在 BLDPART 命令上指定 SCOPE(*EXTENDED)，则可以构建它们。有关详情，参见第129页的『设置构建范围』。
2. 不能构建带有 C 语言的类型为 PGM 的部件。
3. FNDSTRPART 命令对存储在源文件中的由系统提供的类型或用户定义类型有效，并对 FILE(PF) 部件有效。
4. 一般说来，如果在 PRMPART 命令上指定 EXTEND(*YES)，则可以提升对象。
5. CMPPART 命令对存储在源文件中的由系统提供的类型或用户定义类型有效，并对 FILE(PF) 部件和 PARTL 部件有效。

表 25. 应用程序开发管理器类型以及它们与 CL 命令的关系（共 8 部分，第 5 部分）

命令	MODULE	SRVPGM	BNDDIR[1]	PNLSRC	PNLINC	PNLGRP	MENU (*UIM)
BLDPART	y	y	y	y	y	y	y
CHGPART	y	y	y	y	y		y
CHGPARTINF	y	y	y	y	y	y	y
CHKINPART	y	y	y	y	y	y	y
CHKOUTPART	y	y	y	y	y	y	y
CMPPART				y	y		
CPYPART	y	y	y	y	y	y	y
CRTPART			y	y	y		
CVTPART				y	y		
DLTPART	y	y	y	y	y	y	y
DSPPART	y	y	y	y	y		y
EXPPART	y	y	y	y	y	y	y
FNDSTRPART[2]				y	y		
IMPPART	y	y	y	y	y	y	y
MRGPART				y	y		
PRMPART[3]	y	y	y	y	y	y	y
PRTPARTINF	y	y	y	y	y	y	y
QRYPART	y	y	y	y	y	y	y
RNMPART	y	y	y	y	y	y	y
RTVPARTINF	y	y	y	y	y	y	y
WRKPARTPDM	y	y	y	y	y	y	y

注意:

下列各项适用于表中列示的部件类型:

1. 并不实际地构建消息文件和包含部件，但您构建的部件可以引用它们。对于类型为 BNDDIR 的部件，情况也是这样；然而，如果您在 BLDPART 命令上指定 SCOPE(*EXTENDED)，则可以构建它们。有关详情，参见第129页的『设置构建范围』。
2. FNDSTRPART 命令对存储在源文件中的由系统提供的类型或用户定义类型有效，并对 FILE(PF) 部件有效。
3. 一般说来，如果在 PRMPART 命令上指定 EXTEND(*YES)，则可以提升对象。

表 26. 应用程序开发管理器类型以及它们与 CL 命令的关系（共 8 部分，第 6 部分）

命令	SCHIDX	JOBD	JOBQ	MSGQ	OUTQ	PARTL	PRDDFN	PRDLOD
BLDPART						y		
CHGPART	y	y	y	y	y	y	y	y
CHGPARTINF	y	y	y	y	y	y	y	y
CHKINPART	y	y	y	y	y	y	y	y
CHKOUTPART	y	y	y	y	y	y	y	y
CMPPART						y	y	y
CPYPART	y	y	y	y	y	y	y	y
CRTPART	y	y	y	y	y	y	y	y
CVTPART								
DLTPART	y	y	y	y	y	y	y	y
DSPPART	y	y	y	y	y	y	y	y
EXPPART	y	y	y	y	y	y	y	y
FNDSTRPART[1]							y	y
IMPPART	y	y	y	y	y	y	y	y
MRGPART							y	y
PRMPART[2]	y	y	y	y	y	y	y	y
PRTPARTINF	y	y	y	y	y	y	y	y
QRYPART	y	y	y	y	y	y	y	y
RNMPART	y	y	y	y	y	y	y	y
RTVPARTINF	y	y	y	y	y	y	y	y
WRKPARTPDM	y	y	y	y	y	y	y	y

注意:

下列各项适用于表中列示的部件类型:

1. FNDSTRPART 命令对存储在源文件中的由系统提供的类型或用户定义类型有效，并对 FILE(PF) 部件有效。
2. 一般说来，如果在 PRMPART 命令上指定 EXTEND(*YES)，则可以提升对象。

表 27. 应用程序开发管理器类型以及它们与 CL 命令的关系（共 8 部分，第 7 部分）

命令	SYSTEML	CBL36SRC	CBL36INC	DSPF36SRC	MNU36SRC	MSGF36SRC	OLC36SRC
BLDPART							
CHGPART	y	y	y	y	y	y	y
CHGPARTINF	y	y	y	y	y	y	y
CHKINPART	y	y	y	y	y	y	y
CHKOUTPART	y	y	y	y	y	y	y
CMPPART	y	y	y	y	y	y	y
CPYPART	y	y	y	y	y	y	y
CRTPART	y	y	y	y	y	y	y
CVTPART	y	y	y	y	y	y	y
DLTPART	y	y	y	y	y	y	y
DSPPART	y	y	y	y	y	y	y
EXPPART	y	y	y	y	y	y	y
FNDSTRPART	y						
IMPPART	y	y	y	y	y	y	y
MRGPART	y	y	y	y	y	y	y
PRMPART[1]	y	y	y	y	y	y	y
PRTPARTINF	y	y	y	y	y	y	y
QRYPART	y	y	y	y	y	y	y
RNMPART	y	y	y	y	y	y	y
RTVPARTINF	y	y	y	y	y	y	y
WRKPARTPDM	y	y	y	y	y	y	y

注意:

1. 一般说来，如果在 PRMPART 命令上指定 EXTEND(*YES)，则可以提升对象。

表 28. 应用程序开发管理器类型以及它们与 CL 命令的关系（共 8 部分，第 8 部分）

命令	RPG36SRC	RPG36INC	RPT36SRC	SRT36SRC	TXT36SRC	VRPGBIN	VRPGTXT
BLDPART							
CHGPART	y	y	y	y	y	y	y
CHGPARTINF	y	y	y	y	y	y	y
CHKINPART	y	y	y	y	y	y	y
CHKOUTPART	y	y	y	y	y	y	y
CMPPART	y	y	y	y	y	y	y
CPYPART	y	y	y	y	y	y	y
CRTPART	y	y	y	y	y	y	y
CVTPART	y	y	y	y	y	y	y
DLTPART	y	y	y	y	y	y	y
DSPPART	y	y	y	y	y	y	y
EXPPART	y	y	y	y	y	y	y
FNDSTRPART							
IMPPART	y	y	y	y	y	y	y
MRGPART	y	y	y	y	y	y	y
PRMPART[1]	y	y	y	y	y	y	y
PRTPARTINF	y	y	y	y	y	y	y
QRYPART	y	y	y	y	y	y	y
RNMPART	y	y	y	y	y	y	y
RTVPARTINF	y	y	y	y	y	y	y
WRKPARTPDM	y	y	y	y	y	y	y

注意:

1. 一般来说，如果在 PRMPART 命令上指定 EXTEND(*YES)，则可以提升对象。

附录C. 命名规则

本节描述您在为项目、组、部件和提升码创建名称时需要牢记的规则。

项目和组名

在使用 CRTPRJ 命令（ “使用项目” 屏幕上的 “F6=创建” ）或 CRTGRP 命令（ “使用组” 屏幕上的 “F6=创建” ）创建项目或组时，为它提供两个名称。 您在 PRJ 或 GRP 参数上指定的 32 个字符的名称是在所有与项目或组相关的 CL 命令上使用的名称，用于标识此特定项目。一些有效的名称包括 PAYROLL、BIWEEKLYPAYROLL、PAY_ROLL_09.92 或 PAY$ROLL。一些有效的组名包括 VERSION1、V1PAY$ROLL 和 SYS_TEST_VERSION2。

在 SHORTPRJ 参数上，您指定一个项目简称，该名称用来为该项目中的组创建唯一的 AS/400 库名。

在 SHORTGRP 参数上，您指定一个简短组名，该名称用来为项目中的每个组创建唯一的 AS/400 库名。

项目简称与简短组名共同构成 OS/400 库名。例如，如果有一个简称为 PAY 的项目 PAYROLL 和一个简称为 MST 的组 MASTER，则 AS/400 库名将是 PAY.MST。如果在同一项目中创建另一个名为 TEST 的组，其简称为 TST，则它的库将是 PAY.TST。您不应使用以字符 Q 开头的项目简称，系统假定这样的库是系统库。项目简称最长可有 4 个字符。简短组名最长可有 5 个字符。第一个字符必须是以下其中一项:

- 字母 (A-Z)
- 美元符 ($)
- 数字符 (#)
- @ 符

项目简称的其余 3 个字符以及简短组名的其余 4 个字符可以是以下各项的任意组合:

- 字母 (A-Z) 或数字 (0-9)
- 美元符 ($)
- 数字符 (#)
- @ 符
- 下划线 (_)

项目简称和简短组名不能包含:

- 问号 (')
- 句点 (.)
- 嵌入空格
- DBCS 字符或任何其他特殊字符

一些有效的项目简称包括 PR$、PAYR 或 PR92。一些有效的简短组名包括 V1、V1PAY 或 TSTV2。

部件名

每个部件都由以下四部分信息的组合唯一标识:

项目 此部件所属的项目的名称（最长 32 个字符）。

组 此部件所属的组的名称（最长 32 个字符）。

类型 部件类型，例如 PGM、FILE 或 RPGSRC（最长 10 个字符）。第52页的表5列示“应用程序开发管理器”功能部件支持的 OS/400 对象类型。

部件 唯一标识特定项目、组和类型的部件的名称（最长 10 个字符）。

部件名的第一个字符必须是以下其中一项:
- 字母 (A-Z)
- 美元符 ($)
- 数字符 (#)
- @ 符

其余字符可以是以下各项的任意组合:
- 字母 (A-Z) 或数字 (0-9)
- 美元符 ($)
- 数字符 (#)
- @ 符
- 句点 (.)
- 下划线 (_)

部件名不能包含:
- 问号
- 嵌入空格
- DBCS 字符或任何其他特殊字符

因此，全限定部件名最长可以是 84 个字符加上用于进行分隔的空格。PAYROLL DEVELOPER1 RPGSRC BIWKLY 是全限定名称的一个示例。部件名为 BIWKLY; 它的类型为 RPGSRC; 位于项目 PAYROLL 中的组 DEVELOPER1 中。

给定名称和类型的部件只能存在于尚未包含具有相同名称相同类型的部件的组中。如果一个相同名称相同类型的部件已存在于要包含新部件的组的缺省搜索路径中，则创建、复制或导入该部件时，必须指定提升码 *NONE。

用户定义类型名

每个部件都由以下四部分信息的组合唯一标识:

项目 此部件所属的项目的名称（最长 32 个字符）。

组 此部件所属的组的名称（最长 32 个字符）。

类型 部件类型，例如 ZPGM、ZFILE 或 ZRPGSRC（最长 10 个字符）。建议所有用户定义部件类型名都以 Z 开头。

部件 用于唯一标识特定项目、组和类型的部件的名称（最长 10 个字符）。

部件名的第一个字符必须是以下其中一项:
- 字母 (A-Z)

美元符 ($)
数字符 (#)
@ 符

其余字符可以是以下各项的任意组合:
字母 (A-Z) 或数字 (0-9)
美元符 ($)
数字符 (#)
@ 符
句点 (.)
下划线 (_)

部件名不能包含:
问号
嵌入空格
DBCS 字符或任何其他特殊字符

因此，全限定部件名最长可以是 84 个字符加上用于进行分隔的空格。PAYROLL DEVELOPER1 RPGSRC BIWKLY 是全限定名的一个示例。此部件名为 BIWKLY; 它的类型为 RPGSRC; 位于项目 PAYROLL 中的组 DEVELOPER1 中。

给定名称和类型的部件只能存在于尚未包含具有相同名称相同类型的部件的组中。如果一个相同名称相同类型的部件已存在于要包含新部件的组的缺省搜索路径中，则创建、复制或导入该部件时，必须指定提升码 *NONE。

提升码名称

提升码值的命名限制与组名的命名限制相同; 即，最长 32 个字符，首字符必须是下列其中一项:
字母 (A-Z)
美元符 ($)
数字符 (#)
@ 符

其余 31 个字符可以是以下各项的任意组合:
字母 (A-Z) 或数字 (0-9)
美元符 ($)
数字符 (#)
@ 符
句点 (.)
下划线 (_)

附录D. 替换变量

“应用程序开发管理器”功能部件将几个字符串识别为替换变量。这与“编程开发管理器”实用程序为其用户定义选项提供替换的方式类似。

这些替换变量是以许多不同的方式识别的:

1. **“编程开发管理器”实用程序**

   在用来处理 PDM 用户定义选项的命令中。可以在下列“编程开发管理器”屏幕中使用替换变量:

   用 PDM 来使用项目
   用 PDM 来使用组
   用 PDM 来使用部件
   创建用户定义选项

   对每个变量返回的值要视您正在使用的屏幕而定。如果在使用库、对象或成员时使用这些变量中的任何一个，则该变换被替换为 *NULL。

2. **BLDOPT 部件**

   在您在构建选项部件（类型为 BLDOPT 的部件）中定义的编译命令中。

3. **CMD 参数**

   在您在 CHGADMACN 命令的 CMD 参数上定义的命令字符串中。

4. **BLDCMD 参数**

   在您在 ADDADMLANG 和 CHGADMLANG 命令的 BLDCMD 参数上定义的命令字符串中。

注: 在那些情况下，当替换变量不适用时，替换变量不会被替换。例如，如果 CHGADMACN 命令上定义的操作为删除操作，则 &ZJ 替换变量不能被替换为正在移动或复制的部件的名称。此时，使用替换变量 &ZJ 是没有意义的。如果没有必需的信息可用，则不解析替换变量。

表 29. 应用程序开发管理器识别的替换变量

替换变量	描述
&A 对象属性	**PDM** 如果您正在使用部件，则 &A 被替换为部件的 OS/400 对象属性。如果正在使用项目或组，则 &A 被替换为 *NULL。 **BLDOPT 部件** 不适用 **CMD 参数** 正在处理的部件的对象属性。 **BLDCMD 参数** &A 被替换为 *NULL。

表 29. 应用程序开发管理器识别的替换变量 (续)

替换变量	描述
&B 列表类型	**PDM** 如果正在使用项目列表，则 &B 被替换为 R。如果您正在使用组列表，则 &B 被替换为 G。如果您正在使用部件列表，则 &B 被替换为 P。 **BLDOPT 部件** 不适用 **CMD 参数** 不适用 **BLDCMD 参数** 不适用
&C 选项	**PDM** &C 被替换为用户定义选项代码。 **BLDOPT 部件** 不适用 **CMD 参数** 不适用 **BLDCMD 参数** 不适用
&D 上次更改日期	**PDM** 如果您正在处理部件列表，则 &D 被替换为上次更改部件内容的日期。 返回的值具有系统格式，带有分隔符。否则，&D 被替换为 *NULL。必须在单引号中使用此变量（即 '&D'），这是因为日期可能会包含会用作运算符的斜杠 (/)。 **BLDOPT 部件** 不适用 **CMD 参数** 正在处理的部件的上次更改日期。 **BLDCMD 参数** 正在编译的部件的上次更改日期。
&E 以批处理方式运行	**PDM** 如果在“更改缺省值”屏幕上的*以批处理方式运行*提示中指定了 Y，则 &E 被替换为 *YES，如果指定的是 N，则被替换为 *NO。 **BLDOPT 部件** 不适用 **CMD 参数** 不适用 **BLDCMD 参数** 不适用

表 29. 应用程序开发管理器识别的替换变量 (续)

替换变量	描述
&F 源文件名	**PDM** 如果正在处理部件，则 &F 被替换为存储该部件的文件的名称。此变量仅对第50页的表 4中列示的部件类型有效。 **BLDOPT 部件** 正在编译的部件的源文件名。 **CMD 参数** 正在处理的部件的源文件名。当在 CHGADMACN 命令的 ACTION 参数上指定了 *CPY 操作时，这是*来源*源文件的名称。 **BLDCMD 参数** 正在处理的部件的源文件名。 当源文件名不可用时，&F 被替换为 *NULL。
&G 作业描述库	**PDM** &G 被替换为“更改缺省值”屏幕中的作业描述库值。 **BLDOPT 部件** 不适用 **CMD 参数** 不适用 **BLDCMD 参数** 不适用
&H 作业描述名	**PDM** &H 被替换为“更改缺省值”屏幕中的作业描述值。 **BLDOPT 部件** 不适用 **CMD 参数** 不适用 **BLDCMD 参数** 不适用
&J 作业描述	**PDM** &J 被替换为“更改缺省值”屏幕中的作业描述值，具有库／作业描述格式。 **BLDOPT 部件** 不适用 **CMD 参数** 不适用 **BLDCMD 参数** 不适用

表 29. 应用程序开发管理器识别的替换变量 (续)

替换变量	描述
&L 库名	**PDM** 如果正在使用项目，则 &L 被替换为 *NULL。如果您正在使用组，则 &L 被替换为与该组相关联的库的名称。如果您正在使用部件，则 &L 被替换为包含该部件的组的库的名称。 **BLDOPT 部件** 与包含正在编译的部件的组相关联的库名。 **CMD 参数** 与包含正在处理的部件的组相关联的库名。 对于在 CHGADMACN 命令的 ACTION 参数上指定的 *CPY 操作，这是正在从中复制部件的库的名称。例如，如果该 IMPPART 命令是对 *CPY 操作定义的，则 &L 被替换为正在从中导入部件的库的名称。 当发出 EXPPART 命令时，这是对 CHGADMACN 命令的 ACTION 参数上指定的 *DLT 操作将部件导出至的库的名称。 **BLDCMD 参数** 与包含正在编译的部件的组相关联的库名。
&N 对象 / 成员名	**PDM** 如果正在使用项目，则 &N 被替换为 *NULL。如果您正在使用组，则 &N 被替换为与该组相关联的库的名称。如果您正在使用部件，则 &N 被替换为与该项目相关联的对象或成员的名称。 **BLDOPT 部件** 正在编译的部件的对象名。 **CMD 参数** 正在处理的部件的对象名。 **BLDCMD 参数** 正在编译的部件的对象名。
&O 对象库	**PDM** 如果正在使用项目，则 &O 被替换为 *NULL。如果您正在使用组，则 &O 被替换为与该组相关联的库的名称。如果正在使用部件，则 &O 被替换为与 “用 PDM 来使用部件” 屏幕上的*指定的组*提示中命名的组相关联的库的名称。 **BLDOPT 部件** 存放 BLDPART 命令的输出对象的库的名称。 **CMD 参数** 与包含正在处理的部件的组相关联的库名。 对于在 CHGADMACN 命令的 ACTION 参数上指定的 *CPY 操作，这是正在从中复制部件的库的名称。例如，如果该 IMPPART 命令是对 *CPY 操作定义的，则 &O 被替换为正在从中导入部件的库的名称。 当发出 DLTPART 命令时，这是部件所在的库的名称。 **BLDCMD 参数** 存放 BLDPART 命令的输出对象的库的名称。

表 29. 应用程序开发管理器识别的替换变量 (续)

替换变量	描述
&P 以批处理方式编译	**PDM** 如果在“更改缺省值”屏幕上的*以批处理方式编译*提示中指定了 Y，则 &P 被替换为 *YES，如果指定的是 N，则被替换为 *NO。 **BLDOPT 部件** 不适用 **CMD 参数** 不适用 **BLDCMD 参数** 不适用
&R 替换对象	**PDM** 如果正在使用项目、组或部件，则无论“更改缺省值”屏幕上的*替换对象*提示中指定的是什么值，&R 总是被替换为 *NO。 **BLDOPT 部件** 不适用 **CMD 参数** 不适用 **BLDCMD 参数** 不适用
&S 不带 '*' 的项类型	**PDM** 如果正在使用项目，则 &S 被替换为 *NULL。如果正在使用组，则 &S 被替换为 LIB。如果正在使用部件，则 &S 被替换为该部件的对象类型（不带 '*'）。 **BLDOPT 部件** 不适用 **CMD 参数** 正在处理的部件的 OS/400 对象类型，未前导以 '*'。 **BLDCMD 参数** 正在编译的部件的 OS/400 对象类型，未前导以 '*'。
&T 带有 '*' 的项类型	**PDM** 如果正在使用项目，则 &T 被替换为 *NULL。如果正在使用组，则 &T 被替换为 *LIB。如果正在使用部件，则 &T 被替换为该部件的对象类型（带有 '*'）。 **BLDOPT 部件** 不适用 **CMD 参数** 正在处理的部件的 OS/400 对象类型，前导 '*'。 **BLDCMD 参数** 正在编译的部件的 OS/400 对象类型，前导 '*'。

表 29. 应用程序开发管理器识别的替换变量 (续)

替换变量	描述
&U 用户定义选项文件	**PDM** &U 被替换为“更改缺省值”屏幕中的用户定义选项文件名。 **BLDOPT 部件** 不适用 **CMD 参数** 不适用 **BLDCMD 参数** 不适用
&V 用户定义选项库	**PDM** &V 被替换为“更改缺省值”屏幕中的用户定义选项库名。 **BLDOPT 部件** 不适用 **CMD 参数** 不适用 **BLDCMD 参数** 不适用
&W 用户定义选项文件成员	**PDM** &W 被替换为“更改缺省值”屏幕中的用户定义选项文件成员名。 **BLDOPT 部件** 不适用 **CMD 参数** 不适用 **BLDCMD 参数** 不适用
&X 对象文本描述	**PDM** 如果正在使用项目，则 &X 被替换为 *NULL。如果正在使用组或部件，则 &X 被替换为与该项相关联的对象或成员的文本（括在单引号中）。 **BLDOPT 部件** 不适用 **CMD 参数** 正在处理的部件的对象文本描述（50 个字符）。 **BLDCMD 参数** 正在编译的部件的对象文本描述（50 个字符）。

表 29. 应用程序开发管理器识别的替换变量 (续)

替换变量	描述
&ZA 语言	**PDM** &ZA 被替换为部件的语言属性。 **BLDOPT 部件** 正在编译的源部件的语言属性，而不是输出部件的语言属性。 **CMD 参数** 正在处理的部件的语言。 **BLDCMD 参数** 正在编译的部件的语言。
&ZC OS/400 命令名	**PDM** &ZC 被替换为正在处理的“应用程序开发管理器”命令（例如 IMPPART 或 BLDPART 命令）的名称。 **BLDOPT 部件** &ZC 被替换为 BLDPART。 **CMD 参数** 当前正在处理的 OS/400 命令名；例如，IMPPART 或 BLDPART 命令。 **BLDCMD 参数** &ZC 被替换为 BLDPART。
&ZD 构建输出部件类型	**PDM** 不适用 **BLDOPT 部件** &ZD 被替换为 BLDPART 命令正在创建的输出部件的类型。如果先前尚未构建源部件，则此变量与源部件的类型相同。 **CMD 参数** 不适用 **BLDCMD 参数** 不适用
&ZE 输出部件名	**PDM** 不适用 **BLDOPT 部件** &ZE 被替换为 BLDPART 命令正在创建的输出部件的名称。如果先前尚未构建源部件，则此变量与源部件的名称相同。 对于用户定义类型 *NONE，&ZE 设置为 BLDPART 命令遇到的第一个输出成员，或是具有用户定义类型 *NONE 的部件的名称（如果以前尚未处理该部件）。 **CMD 参数** 不适用 **BLDCMD 参数** 不适用

表 29. 应用程序开发管理器识别的替换变量 (续)

替换变量	描述
&ZF 源文件名	**PDM** 不适用 **BLDOPT 部件** 不适用 **CMD 参数** 与正在将部件复制至的组相关联的库名。 当您在 CHGADMACN 命令上定义 *CPY 操作时，可以使用 &ZF。 **BLDCMD 参数** 不适用
&ZG 组名	**PDM** &ZG 被替换为“用 PDM 来使用部件”屏幕上的 *指定的组*提示中给出的组的名称。对于 PRMPART 命令，这是正在从中提升部件的组。对于 CHKOUTPART 命令，这是正在将部件检出至的组。 **BLDOPT 部件** 存放 BLDPART 命令的输出的组的名称。 **CMD 参数** 包含正在处理的部件的组的名称。 **BLDCMD 参数** 包含正在编译的部件的组的名称。
&ZH 扫描关键字	**PDM** &ZH 被替换为“更改缺省值”屏幕中的 SCAN 关键字的值。这些值包括 *YES 和 *NO。 **BLDOPT 部件** 不适用 **CMD 参数** 不适用 **BLDCMD 参数** 不适用
&ZI 库名	**PDM** 不适用 **BLDOPT 部件** 不适用 **CMD 参数** 与正在将部件移动或复制至的组相关联的库名。 当在 CHGADMACN 命令上定义 *MOV 或 *CPY 操作时，可以使用 &ZI。 **BLDCMD 参数** 不适用

表 29. 应用程序开发管理器识别的替换变量 (续)

替换变量	描述
&ZJ TOPART 值	**PDM** 不适用 **BLDOPT 部件** 不适用 **CMD 参数** 正在使用复制操作移动或复制的部件的名称。 **BLDCMD 参数** 不适用
&ZL 组名	**PDM** &ZL 被替换为您在“用 PDM 来使用部件”屏幕上它的旁边输入了选项的组的名称。如果正在使用项目或组，则 &ZL 被替换为 *NULL。 **BLDOPT 部件** 不适用 **CMD 参数** 不适用 **BLDCMD 参数** 不适用
&ZN 名	**PDM** &ZN 被替换为正在使用的“编程开发管理器”列表中的名称。如果正在使用项目，则 &ZN 被替换为项目名。如果正在使用组，则 &ZN 被替换为组名。如果正在使用部件，则 &ZN 被替换为部件名。 **BLDOPT 部件** 正在编译的部件的名称。 **CMD 参数** 正在处理的部件的名称。 **BLDCMD 参数** 正在编译的部件的名称。
&ZO 构建范围	**PDM** &ZO 被替换为“更改缺省值”屏幕上的“构建范围”参数的值。 **BLDOPT 部件** BLDPART 命令上的“构建范围”参数的值。这些值包括 *NORMAL、*LIMITED、*DIRCHAIN 和 *EXTENDED。 **CMD 参数** 不适用 **BLDCMD 参数** ADDADMLANG 或 CHGADMLANG 命令的 BLDCMD 参数上的命令字符串中指定的值。这些值包括 *NORMAL、*LIMITED、*DIRCHAIN 和 *EXTENDED。

表 29. 应用程序开发管理器识别的替换变量 (续)

替换变量	描述
&ZP 项目名	**PDM** &ZP 被替换为您在使用组或部件时的项目名。 **BLDOPT 部件** 包含正在编译的部件的项目的名称。 **CMD 参数** 包含正在处理的部件的项目的名称。 **BLDCMD 参数** 包含正在编译的部件的项目的名称。
&ZS 搜索路径	**PDM** &ZS 被替换为“更改缺省值”屏幕中的 SCHPTH 关键字的值。此值是搜索路径部件的名称。 **BLDOPT 部件** BLDPART 命令上的“搜索路径”参数的值。此值是搜索路径部件的名称。 **CMD 参数** 不适用 **BLDCMD 参数** ADDADMLANG 或 CHGADMLANG 命令的 BLDCMD 参数上的命令字符串中指定的值。
&ZT 部件类型	**PDM** &ZT 被替换为您使用部件列表时的部件类型。 **BLDOPT 部件** 正在编译的部件的类型。 **CMD 参数** 正在处理的部件的类型。 **BLDCMD 参数** 正在编译的部件的类型。
&ZX 文本描述	**PDM** &ZX 被替换为某项的文本描述（80 个字符），这取决于正在使用的屏幕。 **BLDOPT 部件** 不适用 **CMD 参数** 正在处理的部件的文本描述（80 个字符）。 **BLDCMD 参数** 正在处理的部件的文本描述（80 个字符）。

表 29. 应用程序开发管理器识别的替换变量 (续)

替换变量	描述
&ZY 部件列表部件名	**PDM** &ZY 被替换为用来生成 “使用部件列表中的部件” 屏幕上的部件列表的部件列表部件名。在 PDM 中的任何其他列表上，它是 *NONE。 **BLDOPT 部件** PARTL 参数上指定的部件列表部件的名称。 **CMD 参数** PARTL 参数上指定的部件列表部件的名称。 **BLDCMD 参数** PARTL 参数上指定的部件列表部件的名称。

附录E. 构建报告消息

本附录列示构建报告中出现的 BLDPART 命令消息， 它们描述构建部件的原因。提供了第一级和第二级消息文本。

原因消息

下列消息是用于描述构建部件的原因的构建报告消息。在本节中，&1=*类型*，而 &2=*部件*。在构建报告中，*类型*和*部件*解析为实际名称。

ADM4213 &1 &2 已更改。

原因: 部件 &1 &2 的时间戳记已更改。

ADM4204 &1 &2 没有对象。

原因: BLDPART 命令发现输出部件没有相关联的对象。这是先前构建过程的结果，该构建过程除去了该对象，以允许在 FILE PF 和 FILE LF 情况下进行编译。

ADM4203 以前未构建 &1 &2。

原因: BLDPART 命令发现输出部件没有来自先前构建过程的信息。

ADM4206 &1 &2 不是由源部件创建的。

原因: 用来确定是否应该发生构建过程的输出部件先前是由另一个源部件构建的。

ADM4212 找不到 &1 &2。

原因: 先前构建过程使用了此部件，但在构建范围中找不到它。

ADM4226 BLDPART 处理不能继续。

原因: BLDPART 以内部方式使用的一个或多个工作区已被超出。如果正在构建大量的部件，则可能会发生这种情况。

ADM4209 BLDOPT 部件包含编译器命令。

原因: 此部件先前是使用缺省编译器命令构建的。此构建过程中使用的编译器命令来自 BLDOPT 部件。

ADM4211 BLDOPT 部件已更改。

原因: BLDOPT 部件的时间戳记已更改。

ADM4208 正在使用缺省编译器命令。

原因: 找不到 BLDOPT 部件，已使用缺省编译器命令。该部件先前是使用 BLDOPT 部件构建的。

ADM4210 找到不同的 BLDOPT 部件。

原因: 为了构建此部件而发现的 BLDOPT 部件与先前构建过程中使用的部件不同。

ADM4216 &1 &2 中的字段已更改。

原因: 指定的 FILE 中的字段已更改。

ADM4217 不再能够在 &1 &2 中找到字段。

原因: 指定的 FILE 中的字段不再存在。

ADM4205 *FORCE 设置为 *YES。

原因: BLDPART 命令上的 FORCE 参数设置为 *YES。

ADM4219 可能是因为已构建相关部件而构建。

原因: 此部件不需要构建，但一个相关部件可能需要构建，这可能导致此部件需要构建。

ADM4221 可能是因为构建了源或相关部件而被构建。

原因: 此部件不需要构建，但一个相关部件或创建此部件的源可能需要构建，这可能导致此部件需要构建。

ADM4220 因为必须构建源或相关部件而必须构建。

原因: 此部件不需要构建，但一个相关部件或创建此部件的源需要构建，这将导致需要构建此部件。

ADM4218 因为必须构建相关部件而必须构建。

原因: 此部件不需要构建，但一个相关部件必须构建，这导致此部件也将被构建。

ADM4202　　找到的输出部件在共享项目中。

原因: BLDPART 命令在交叉项目中找到输出部件。构建过程继续，就象没有输出部件一样，这表示将构建源部件。

ADM4201　　找不到来自先前构建过程的输出部件。

原因: BLDPART 命令找不到在先前构建过程中创建的输出部件。

ADM4214　　&1 &2 中的记录级别标识已更改。

原因: 指定的 FILE 中的记录级别标识已更改。

ADM4215　　不再能够在 &1 &2 中找到记录。

原因: 指定的 FILE 中的记录不再存在。

ADM4207　　源部件已更改。

原因: 自从执行先前构建过程之后，正在构建的源部件的时间戳记已更改。

ADM4225　　以前尚未构建源部件。

原因: 以前尚未使用 BLDPART 命令构建源部件。

ADM4227　　&1 &2 中的 SRVPGM 签名已更改。

原因: 指定的 SRVPGM 中的签名已更改。

ADM4228　　用户定义语言已更改。

原因: 已使用 CHGADMLANG 命令更改了用户定义语言。

警告消息

对于构建报告警告消息中的大多数，消息中的替换变量解析为下列各项:

&1　项目名

&2　组名

&3　部件类型

&4　部件名

对于一些消息，&1、&2、&3 和 &4 可以分别表示库、对象和成员名。要查看这些消息，请输入以下命令。

```
DSPMSGD RANGE(ADM4400 *LAST) MSGF(QADM/QADMMSG)
```

特殊警告消息

仅当 BLDPART 处理期间其中一个内部构建空间超过其容量的 80% 时，新的警告消息才会出现在构建报告中。在作业记录中，这些警告还伴随着低级别警告消息。请复查这些消息并执行这些消息的消息描述中提到的适当恢复操作。

附录F. 多语言支持

“应用程序开发管理器”功能部件允许您同时以多种国家语言创建和管理应用程序的多个版本，并允许您为需要翻译的组件管理所需的翻译活动。

本附录提供了一个样本项目层次结构，您可以创建此项目层次结构来管理多语言应用程序，本附录还说明了在开发此项目层次结构时需要考虑的问题。本附录没有讨论 IBM 为本产品提供的国家语言支持。请参考“iSeries 信息中心”以了解这方面的信息。

多语言项目层次结构

假设

本节讨论的方案作了下列假设:

1. 应用程序的开发由所有开发者以一种基本语言完成。(在此示例中，基本语言是英语。)
2. 必须翻译下列类型的应用程序组件中的某一些或全部:
   - 消息(类型为 MSGF 的部件)。
   - 显示文件(类型为 DDSSRC 的部件)。
   - 打印文件(类型为 PRTF 的部件)。
   - 数据区(类型为 DTAARA 的部件)。
   - 命令源(类型为 CMDSRC 的部件)。
   - 程序源(类型为 xxxSRC 或 xxxINC 的部件，其中，xxx 是本产品支持的编程语言之一。查看第50页的表4以了解这方面的信息。)

项目层次结构如何工作

第242页的图88显示了如何定义多语言项目层次结构来帮助管理翻译活动。

您象平常那样创建项目层次结构，并考虑所有附加 CCSID 需求。参考第242页的『了解 CCSID』以获取“应用程序开发管理器”命令如何使用 CCSID 的说明。

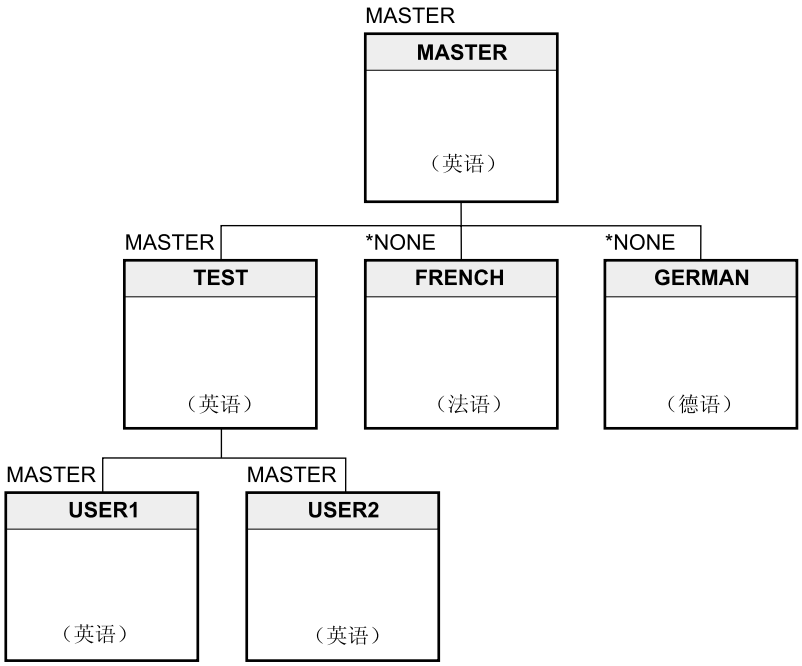

图 88. 用于翻译的项目层次结构

在创建项目时，它采用创建项目的系统上的双字节字符集 (DBCS) 特征。这表示如果系统启用了 DBCS，则该项目也会启动 DBCS，并且它的源文件将允许使用 DBCS 字符。

要使用基本语言来开发应用程序，请使用项目层次结构中具有与根组相同的提升码的组。在此示例中，根组的提升码是 MASTER。使用具有提升码 *NONE 的组来包含正在翻译的部件的副本。在此示例中，正在将应用程序转换成法语和德语。

注: 组 FRENCH 和 GERMAN 是使用提升码 *NONE 创建的，因此，不能提升这些组中的部件，也不能替换这些部件的英文版本。

当用户开发应用程序时，将不断地提升、构建和测试应用程序的部件，直至根组 MASTER。当应用程序的所有部件都在根组中时，您可以开始翻译对语言敏感的部件:

1. 使用 CHKOUTPART 命令以在组中为要翻译的语言创建正在翻译的部件的副本。在这些部件上指定 PRMCODE(*NONE)，以使它们永远不能被提升。
2. 在翻译了所有部件之后，在翻译组（例如组 FRENCH）中构建所有部件。
3. 在构建了组中的所有部件之后，导出所需语言版本的应用程序版本。例如，如果要创建英文版，则从根组导出整个应用程序，或者，通过在 EXPPART 命令上指定 GRP(FRENCH)，可以导出整个法文版的应用程序。如果您只想导出已翻译的部件，则可以在 EXPPART 命令上指定适当语言的组名和 SCAN(*NO)。有关 EXPPART 命令的详情，参考第13章 导出应用程序。

了解 CCSID

本节说明 “应用程序开发管理器” 命令如何使用 CCSID，以及 CCSID 是如何影响可转换源的。有关 CCSID 的讨论的详情，您应当参考 “iSeries 信息中心”。

下列 “应用程序开发管理器” 命令使用 CCSID:

- CRTGRP

当您创建一个组时，您以显式方式或以缺省方式指定用来在该组中创建所有源的 CCSID。CCSID 只影响源成员，它不直接影响其他对象。

如果 CCSID 与翻译部件的人员有关，则应使用该 CCSID 创建组。[1]

- CPYPART、IMPPART、CHKOUTPART、PRMPART 和 EXPPART

  当使用其中一个命令来将源成员从一个文件复制至另一个文件时，将把源成员中的十六进制值从复制源 CCSID 表示法转换为复制目标 CCSID 表示法。对于 CPYPART、IMPPART、CHKOUTPART 和 PRMPART 命令，如果用于接收成员的源文件不存在，则会使用接收部件的组的 CCSID 创建它。对于 EXPPART 命令，如果用于接收成员的源文件不存在，则会使用运行 EXPPART 命令的作业的 CCSID 创建它。

仅当考虑可转换源时，CCSID 才对您有所帮助。然而，它不影响消息文件本身，因为这些文件不能使用 CCSID 创建。

可以将基本语言组中的部件的十六进制值转换成其他语言所需的那些值，如下所述。参考图88，其中描述了如何将英文版部件转换成法文版和德文版部件:

- 消息

  要翻译消息文件，请使用 CHKOUTPART 命令将它们从根组检出到组 FRENCH 和 GERMAN。然后，必须以人工方式翻译每个文件。

- 屏面组和 UIM 菜单

  下列三个属性配合 PNLGRP 部件使用:

  – MSGID

    编译器从库 LIB 的库 MSGF 中抽取一级消息文本以获得消息 MSGID，并用文本替换此符号。检索到的消息文本不带任何替换变量。

    消息文件缺省为 PNLGRP 标志的 SUBMSGF 属性上指定的消息文件。如果未指定 SUBMSGF，则必须在符号中指定消息文件。

  – SUBMSGF

    这是 `&msg` 符号上未指定消息文件时要使用的缺省消息文件。

  – DFTMSGF

    这是当下列任何标记上未对消息标识符指定消息文件时要使用的缺省消息文件: CHECK、LISTDEF、TL、DATASLTC、PDFLDC、LISTACT 和 MENUI。

    对于其中任何一个标记，当对此标记指定 MSGID 属性时，如果没有在该标记上指定相对应的消息文件属性，则此属性是必需的。

- 显示文件

  您需要为 DDS 源考虑三种情况:

  – DDS 源包含它的所有可翻译文本的 MSGID 关键字。[2]

    如果是这样的话，您只需翻译以上描述的方法中的消息。这是因为消息和 MSGID 关键字都是在运行时解析的。

  – DDS 源包含它的部分或全部可翻译文本的 MSGCON 关键字。

1. 建议将同一 CCSID 用于项目层次结构中的所有组，作为用于开发基本语言应用程序的 CCSID。这样可以避免在项目层次结构中提升源时毫无必要地转换源的十六进制表示法。

2. 目前，MSGID 关键字只有在 DSPF DDS 源中才受支持。

请以上述方式翻译消息。然后重新编译组 FRENCH 和 GERMAN 中的文件。因为消息和 MSGCON 关键字是在编译时解析的，所以您必须这样做。使用 BLDPART 命令编译 DDS 源。

– DDS 源包含它的部分或全部可翻译文本的文字。

要翻译这些部件，必须将它们从根组检出至 FRENCH 和 GERMAN 组。然后，必须人工地翻译组中的每个部件。完成翻译后，使用 BLDPART 命令编译 DDS 源。

- 数据区

要翻译数据区，请首先将它们从根组检出至 FRENCH 和 GERMAN 组。然后人工地翻译每个数据区。

- 程序源

如果您计划将应用程序转换成其他语言，我们建议您不要在程序源代码中声明要翻译的文字。然而，即使程序源代码具有此特性，仍可通过执行下列各项来进行翻译。必须将部件从根组检出至组 FRENCH 和 GERMAN。然后，必须人工地翻译组中的每个部件中的文字。完成翻译后，使用 BLDPART 命令编译程序源代码。

- 命令源

要翻译命令源，首先将命令源从根组检出至组 FRENCH 和 GERMAN。然后人工地翻译每个命令源部件。完成翻译后，使用 BLDPART 命令编译命令源。

附录G. 提示与技巧

本附录包含用于改进一些“应用程序开发管理器”命令的效率和性能的建议。本附录提供了关于下列主题的信息:

- 性能改进
- 部件转换: 导入屏面组源
- 设置出口程序
- 其他提示与技巧

性能改进

本节提供了关于下列主题的信息:

- 改进 WRKGRPPDM、WRKPARTPDM 和 WRKPRJPDM 屏幕以及部件命令性能
- 改进 WRKPARTPDM 和部件命令性能

改进性能

要改进 WRKGRPPDM、WRKPARTPDM 和 WRKPRJPDM 屏幕上的部件命令的性能和响应速度，每晚发出下列命令来组织“应用程序开发管理器”使用的基本数据库文件:

```
RGZPFM FILE(QUSRSYS/QALYPART) KEYFILE(*FILE)
RGZPFM FILE(QUSRSSYS/QALYBLDMAP) KEYFILE(*FILE)
RGZPFM FILE(QUSRSYS/QALYFLDTBL) KEYFILE(*FILE)
RGZPFM FILE(QUSRSYS/QALYREL) KEYFILE(*FILE)
```

改进 WRKPARTPDM 和部件命令性能

要改进 WRKPARTPDM 屏幕上的部件命令性能以及响应速度，请发出下列命令:

1. 通过输入以下命令，了解部件目录文件的大小:

   ```
   DSPOBJD OBJ(QUSRSYS/QALYPART) OBJTYPE(*FILE) DETAIL(*FULL)
   ```

2. 创建子系统:

   ```
   CRTSBSD SBSD(QGPL/ADMPARTS) POOLS((1 *BASE) (2 xxx 1))
   ```

   其中，xxx 大于部件目录文件的大小。

3. 启动子系统:

   ```
   STRSBS (QGPL/ADMPARTS)
   ```

4. 清除池中的内存:

   ```
   CLRPOOL POOL(ADMPARTS 2)
   ```
   将清除池中的内存。

5. 设置对象访问权:

   ```
   SETOBJACC OBJ(QUSRSYS/QALYPART) OBJTYPE(*FILE) POOL(ADMPARTS 2)
   ```

部件转换: 导入屏面组源

要导入屏面组源作为类型为 PNLSRC 的部件（带有语言 PNLGRP），请执行下列步骤:

1. 将包含屏面组源的成员的成员属性更改为 PNLGRP。
2. 使用 IMPPART 命令导入成员，并在该命令上指定 LANG(*ATTR)。

此方法对于带有多种有效语言的任何部件类型都很有用。

如果在导入成员之前没有按预期更改成员属性，则可对创建的命令使用 CHGPARTINF 命令以将其当前语言转换为期望的语言。例如，如果您导入的屏面源成员的属性是 MENU，并且要创建类型为 PNLSRC 且具有语言 PNLGRP 的部件，则可以使用 CHGPARTINF 命令将其语言从 MENU 更改为 PNLGRP。

为附加处理设置出口程序

出口程序是在处理“应用程序开发管理器”命令时运行附加处理的程序。因为所有“应用程序开发管理器”功能都由命令驱动，所以很容易就可以为您所需的任何“应用程序开发管理器”功能设置出口程序。出口程序非常灵活，可以用来在处理“应用程序开发管理器”命令前后插入附加逻辑。

假定要对 DLTPART 命令添加一些额外的处理。您可以编写一个与图89中显示的出口程序类似的出口程序。

```
PGM        PARM(&CMD &PRJ &GRP &TYPE &PART &PARTL +
                  &DLTARCHIVE)

DCL        VAR(&CMD) TYPE(*CHAR) LEN(2)
DCL        VAR(&PRJ) TYPE(*CHAR) LEN(32)
DCL        VAR(&GRP) TYPE(*CHAR) LEN(32)
DCL        VAR(&TYPE) TYPE(*CHAR) LEN(10)
DCL        VAR(&PART) TYPE(*CHAR) LEN(10)
DCL        VAR(&PARTL) TYPE(*CHAR) LEN(10)
DCL        VAR(&DLTARCHIVE) TYPE(*CHAR) LEN(4)

/* Insert Preprocessing Logic Here  */

QSYS/DLTPART PRJ(&PRJ) GRP(&GRP) TYPE(&TYPE) +
             PART(&PART) PARTL(&PARTL) +
             DLTARCHIVE(&DLTARCHIVE)

MONMSG     MSGID(ADM0000) EXEC(SNDPGMMSG MSGID(ADM9003)     +
             MSGF(QADM/QADMMSG))

/* Insert Postprocessing Logic Here */

 ENDPGM
```

图 89. DLTPART 命令的样本出口程序。

以上出口程序尚未执行任何额外处理。然而，您很容易就可以在指示的位置处插入额外的语句。

如果您需要获取“应用程序开发管理器”部件的本机名，则可在出口程序中使用“检索部件信息”(RTVPARTINF) 命令来检索此信息。RTVPARTINF 命令在不同的参数中返回库、文件和成员值。

为了能够在开发者使用 DLTPART 命令时调用此程序，您需要创建 DLTPART 命令的副本，并将该副本放在这样一个库中，该库在库列表中的位置比 QSYS 库要高。您应当更改 DLTPART 命令的副本，以便出口程序可以对其进行处理。

BLDPART 命令提供了一种更容易的方法来编写出口程序，而不必重新编写命令接口。首先创建一个类型为 BLDOPT 的构建选项部件。可以在正在处理的部件类型的前面指定与其相同的标号来调用出口程序。标记的命令可以在构建选项部件中的任何位置编写。如果在构建选项部件中找到标记的命令，就会处理此命令，而不是处理源的编译器命令，即使取消对该编译器命令的注释也是这样。

图90中显示了 BLDOPT QDFT 部件的样本。

```
        CRTRPGPGM ...
          /*******************************************************/
          /* PGM compilers                                       */
          /*******************************************************/
             CRTRPGPGM  PGM(&O/&ZE) SRCFILE(&L/&F) SRCMBR(&ZN) +
                          REPLACE(*YES) USRPRF(*USER) AUT(*EXCLUDE)
          /* CRTCBLPGM  PGM(&O/&ZE) SRCFILE(&L/&F) SRCMBR(&ZN) +
                          REPLACE(*YES) USRPRF(*USER) AUT(*EXCLUDE) */
             CRTCLPGM   PGM(&O/&ZE) SRCFILE(&L/&F) SRCMBR(&ZN) +
                          USRPRF(*USER) REPLACE(*YES) AUT(*EXCLUDE)
                               .
                               .
                               .
CLPSRC:      CALL USRCRTCLP PARM(&O &ZE &L &F &ZN)

RPGSRC:      USRCRTRPG  PGM(&O/&ZE) SRCFILE(&L/&F) SRCMBR(&ZN) +
                          REPLACE(*YES) USRPRF(*USER) AUT(*EXCLUDE)
                               .
                               .
                               .
```

图 90. 样本 BLDOPT QDFT 部件。

在上面的示例中，BLDPART 命令调用出口程序命令 USRCRTRPG（而不是调用 CRTRPGPGM）来处理 RPGSRC 类型的部件。出口程序命令 USRCRTRPG（这是一个对源进行预处理的 CL 程序）通过调用 CRTRPGPGM 编译器来对源进行编译，最后对输出进行后处理。唯一的要求是出口程序必须调用适当的编译器，这是因为编译器会将构建关系信息传送至 BLDPART 命令。您必须记住 BLDOPT 部件的处理次序，并适当地修改它们。

其他提示与技巧

本节提供了关于下列各项的信息:

- 使用不同长度的源文件

使用不同长度的源文件

缺省情况下，“应用程序开发管理器”创建长度为 92 个字节的源文件。（QRPGLESRC 是一个例外，创建时它的长度为 112 个字节。）当任何部件开发命令要写至文件但却找不到它时，便会创建此源文件。如果部件开发命令找到源文件，则该命令会在其中添加、除去或替换其中的成员。

如果您使用不同于缺省长度（92 个字节）的源文件长度，请在创建组后就立即在每个组中创建具有期望长度的所有源文件。您还需要在归档组库中创建类似的文件。如果归档组库不存在，则归档至少一个存储在 92 字节的源文件中的源成员。这样就创建了归档库。

文献目录

本书目列示了您在使用“应用程序开发管理器”组件时可能会使用或感兴趣的各种书籍。

“应用程序开发管理器”库包含下列出版物:

- *ADTS/400: Application Development Manager Self-Study Guide*, SC09-2138-00
- *ADTS/400: Application Development Manager API Reference*, SC09-2180-00
- *ADTS/400: Application Development Manager Introduction and Planning Guide*, GC09-1807-00
- *ADTS: 应用程序开发管理器用户指南*, SB84-0449-00

“应用程序开发工具箱”库包含下列出版物:

- *ADTS/400: Advanced Printer Function*, SC09-1766-00
- *ADTS/400: Character Generator Utility*, SC09-1769-00
- *ADTS/400: Data File Utility*, SC09-1773-00
- *ADTS/400: File Compare and Merge Utility*, SC09-1772-00
- *ADTS/400: Interactive Source Debugger*, SC09-1897-00
- *ADTS/400: Programming Development Manager*, SC09-1771-00
- *ADTS for AS/400: Report Layout Utility*, SC09-2635-00
- *ADTS for AS/400: Screen Design Aid*, SC09-2604-00
- *ADTS for AS/400: Source Entry Utility*, SC09-2605-00
- *Introducing Application Development ToolSet for OS/400 and the AS/400 Server Access Programs*, GC09-2088-00

“应用程序字典服务”库包含下列出版物:

- *ADTS/400: Application Dictionary Services Self-Study Guide*, SC09-2086-00
- *ADTS/400: Application Dictionary Services User's Guide*, SC09-2087-00

VisualAge RPG 库包含下列出版物:

- *VisualAge RPG Parts Reference*, SC09-2450-04
- *VisualAge RPG Language Reference*, SC09-2451-03
- *Programming with VisualAge RPG*, SC09-2449-04
- *Getting Started with WebSphere Development Tools for AS/400*, SC09-2625-04

您可能会对下列与此功能部件相关的出版物感兴趣:

- *Backup and Recovery*, SC41-5304-05
- *CL Programming*, SC41-5721-04
- *COBOL/400 User's Guide*, SC09-1812-00
- *COBOL/400 Reference*, SC09-1813-00
- *ILE Application Development Example*, SC41-5602-00
- *ILE Concepts*, SC41-5606-05
- *ILE C/C++ Programmer's Guide*, SC09-2712-02
- *ILE C/C++ Language Reference*, SC09-4815-00
- *ILE C/C++ Compiler Reference*, SC09-4816-00
- *ILE COBOL Programmer's Guide*, SC09-2540-02
- *ILE COBOL Reference*, SC09-2539-02
- *ILE COBOL Reference Summary*, SX09-1317-02
- *ILE RPG Programmer's Guide*, SC09-2507-03
- *ILE RPG Reference*, SC09-2508-03
- *ILE RPG Reference Summary*, SX09-1315-02
- *iSeries Security Reference*, SC41-5302-04
- *System API Programming*, SC41-5800-00
- *System Manager Use*, SC41-5321-01
- *System Operation*, SC41-4203-00

如果您可以访问因特网，则您可以从以下 Web 站点获取当前的 iSeries 信息和出版物:

`http://www.ibm.com/eserver/iseries/infocenter`

对于 iSeries 出版物的 PDF 版本，参见 *iSeries 信息中心: 手册补遗*, SB84-0456-00 CD ROM。

注意事项

本资料是为在美国提供的产品和服务开发的。IBM 可能没有在其他国家提供本文档中讨论的产品、服务或功能部件。咨询本地的 IBM 代表，以获取有关您所在的地区当前可获得的产品和服务的信息。任何对 IBM 产品、程序或服务的引用并不打算声明或暗示只能使用 IBM 的产品、程序或服务。凡是同等功能的产品、程序或服务，只要不侵犯 IBM 的任何知识产权，都可以使用。然而，评估和验证任何非 IBM 的产品、程序或服务的操作均由用户自行负责。

IBM 可能已经申请或正在申请与本文档有关的各项专利权。提供本文档并不表示允许您使用这些专利。您可以用书面形式将许可查询寄往:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

下列短文不适用于英国或这种规定与当地法律不一致的任何其他国家: 国际商业机器公司“按原样”提供此出版物，但不作任何明确或暗示的保证，包括但不限于对不侵权、销售能力或特定用途的适应性的暗示保证。某些国家不允许在特定事务中否认明确或暗示的保证，因此，此声明可能不适用于您。

本资料可能会包含技术性错误或印刷错误。会定期对本书中的信息进行更改；这些更改将编入本出版物的新版本中。IBM 可能在任何时间在不通知的情况下改进和／或更改本出版物中描述的产品和／或程序。

本资料仅为方便您参考而提供了对非 IBM Web 站点的任何引用，这些引用并不对那些 Web 站点提供任何形式的保证。那些 Web 站点上的资料不是本 IBM 产品资料的一部分，使用那些站点所带来的风险由您自己负责。

为了以下目的: (i) 允许在独立创建的程序和其他程序（包括本程序）之间进行信息交换 (ii) 允许对已经交换的信息进行相互使用，而希望获取本程序有关信息的许可证持有者应与以下地址联系:

IBM Canada Ltd.
Department 071
1150 Eglinton Avenue East
North York, Ontario M3C 1H7
Canada

只要遵守适当的条款和条件，包括某些情形下一定数量的付款，都可获取这方面的信息。

本资料中描述的特许程序和可用于该特许程序的所有特许资料均由 IBM 在“IBM 用户协议”、“IBM 国际程序许可证协议”或我们之间的任何等效协议的条款下提供。

本资料中包含了用于日常商务处理的数据或报表的示例。为了尽可能完整地说明问题，这些示例包含了个人、公司、品牌和产品的名称。所有这些名称都是虚构的，如与实际商业企业所使用的名称和地址相似，纯属巧合。

商标和服务标记

以下各项是国际商业机器公司在美国和或其他国家的商标:

Application System/400
AS/400
AS/400e
C/400
COBOL/400
DB2
DB2
e（特别设计的）
IBM
Integrated Language Environment
iSeries
Operating System/400
OS/400
RPG/400
System/36
System/38
SystemView
VisualAge
WebSphere

Java 和所有基于 Java 的商标和徽标是 Sun 公司在美国和／或其他国家的商标。

其他公司、产品和服务的名称可能是其他公司的商标或服务标记。

索引

[A]

[B]

[C]

[D]

[F]

[G]

[H]

[J]

[K]

[L]

[M]

[P]

[Q]

[R]

[S]

[T]

[W]

[X]

[Y]

[Z]

A

B

C

D

E

F

I

L

程序编号：5722-WDS

中国印刷

SB84-0449-00