IBM WebSphere Business Integration Adapters

mySAP.com 适配器用户指南（适用于 SAP R/3 V3.x）

适配器版本 6.0.x

IBM WebSphere Business Integration Adapters

mySAP.com 适配器用户指南（适用于 SAP R/3 V3.x）

适配器版本 6.0.x

注意！

在使用本资料及其支持的产品之前，请阅读第 297 页的『声明』中的信息。

2004 年 9 月 30 日

本文档的此版本适用于 IBM WebSphere Business Integration mySAP.com 适配器（适用于 SAP R/3 V3.x（5724-H01））版本 6.0.x。

要向 IBM 发送有关 IBM WebSphere Business Integration 文档的意见，请将电子邮件发送至 ctscrcf@cn.ibm.com。我们欢迎您提出宝贵意见。

当您发送信息给 IBM 后，即授予 IBM 非专有权，IBM 可以按它认为合适的任何方式使用或分发此信息，而无须对您承担任何责任。

目录

关于本文档

IBM® WebSphere® Business Integration 适配器产品服务组合为领先的电子商务技术、企业应用程序和旧系统以及大型机系统提供集成连接。该产品集包括用于定制、创建和管理业务集成组件的工具和模板。

本文档包含的内容

本文档描述了 IBM WebSphere Business Integration mySAP.com 适配器的安装、连接器属性配置、业务对象开发和故障诊断。

本文档未包含的内容

本文档未描述部署度量和容量规划问题，例如，服务器负载均衡、适配器处理线程数、最大和最小吞吐量以及容错阈值。

这些问题对于每个客户部署都是唯一的，并且必须在部署适配器的环境中或者特别接近这样的环境中测量这些问题。您应该与 IBM 服务代表联系以共同讨论部署站点的配置。要了解规划和评估这些度量的详细信息，应提供您的特定配置。

读者

本文档面向在客户场所支持和管理 WebSphere Business Integration 系统的顾问、开发者和系统管理员。您应熟悉 SAP 和连接器开发。

相关文档

本产品提供的全套文档描述所有 WebSphere Business Integration 适配器安装的公共功能部件和组件，并包括关于特定组件的参考资料。

本文档包含对其它两个文档的许多引用: *《系统安装指南 Windows 版》*或*《系统安装指南 UNIX 版》*和 *System Implementation Guide for WebSphere InterChange Server*。如果选择打印本文档，可能还要打印这些文档。

可以在下列站点下载、安装和查看文档:

- 有关通用适配器信息、有关将适配器与 WebSphere Message Broker（WebSphere MQ Integrator、WebSphere MQ Integrator Broker、WebSphere Business Integration Message Broker）配合使用的信息以及有关将适配器与 WebSphere Application Server 配合使用的信息，请访问: http://www.ibm.com/websphere/integration/wbiadapters/infocenter
- 有关将适配器与 InterChange Server 配合使用的信息，请访问: http://www.ibm.com/websphere/integration/wicserver/infocenter
- 有关消息代理（WebSphere MQ Integrator Broker、WebSphere MQ Integrator 和 WebSphere Business Integration Message Broker）的更多信息，请访问: http://www.ibm.com/software/integration/mqfamily/library/manualsa/

- 有关 WebSphere Application Server 的更多信息，请访问: http://www.ibm.com/software/webservers/appserv/library.html

注: 关于本产品的重要信息可以在本文档出版后发行的技术支持说明和 Flash 中获得。可以在 WebSphere Business Integration Support Web 站点 http://www.ibm.com/software/integration/websphere/support/ 上找到它们。请选择您感兴趣的组件区并浏览技术说明和 Flash 部分。在 http://www.redbooks.ibm.com/ 处提供的 IBM 红皮书中还可以获得其它信息。

排版约定

本文档使用以下约定:

courier 字体	指示文字值，例如，命令名、文件名、您输入的信息或系统在屏幕上打印的信息。
粗体	指示第一次出现的新术语。
斜体，斜体	指示变量名或交叉引用。
蓝色边框	仅当您联机查看手册时才能看到蓝色边框，它指示交叉引用超链接。单击边框内部以跳至引用的对象。
{ }	在语法行中，花括号括起一组选项，您必须从这些选项中选择一个且只能选择一个选项。
[ ]	在语法行中，方括号括起一个可选参数。
...	在语法行中，省略号指示重复先前的参数。例如，option[,...] 意味着您可以输入多个用逗号隔开的选项。
< >	在命名约定中，尖括号括起名称的单个元素，以将它们互相区分开，如 <server_name><*connector_name*>tmp.log。
\	在本文档中，反斜杠（\）用作目录路径的约定。所有产品路径名都是相对于该产品在系统上的安装目录。
%text%	百分号（%）之间的文本表示 Windows™ text 系统变量或用户变量的值。
ProductDir	表示产品的安装目录。

命名约定

本文档中使用以下命名约定:

- SAP 连接器简称为连接器。
- “连接器”指可视连接器框架和连接器模块的组合。
- 对 ABAP 对象（如功能模块、程序和表）名称的所有引用对于支持 SAP R/3 V3.x 的连接器有效。

本发行版中的新增内容

此连接器和文档的本发行版包含以下新增信息:

发行版 6.0.x 中的新增内容

本文档的此发行版包含以下新增信息:

- 为了编辑上便于分类，已经调整了本指南中的某些部分和章节。
- 适配器现在与 WebSphere Business Integration Adapter Framework V2.6 一起运行。有关受支持的集成代理程序版本的详细信息，请参阅第 3 页的『代理程序兼容性』。
- 从版本 6.0.x 开始，mySAP.com 适配器在 Solaris 7 上不受支持，因此，本指南中已经删除了对该平台版本的引用。
- 现在，适配器在 Solaris 9、Windows 2003、Red Hat Enterprise Linux、SUSE Linux Enterprise Server 和 SUSE Linux Standard Server 上受支持。有关各个平台版本的详细信息，请参阅第 4 页的『适配器平台』。
- 现在，连接器使用的是支持所有 BAPI 调用的单个业务对象处理程序，而不是使用多个特定于 BAPI 的业务对象处理程序，其中每个处理程序都支持一个特定的 BAPI。有关详细信息，请参阅第 85 页的第 7 章，『BAPI 模块概述』。
- 连接器和 SAPODA 支持 BAPI 事务对象。有关详细信息，请参阅第 56 页的『BAPI 事务业务对象』和第 93 页的『BAPI 事务的业务对象结构』。
- SAPODA 不再为 BAPI 模块生成定制业务对象处理程序模板。要创建定制业务对象处理程序，必须由您自己根据所提供的模板来编写它。
- 连接器和 SAPODA 为 DB2 Information Integrator 提供了 ResultSet 支持，因此，使用新的标准连接器属性。有关详细信息，请参阅第 60 页的『ResultSet 业务对象』、第 93 页的『BAPI ResultSet 的业务对象结构』和第 261 页的附录 D，『连接器的标准配置属性』。
- SAPODA 对 RFC 节点中的搜索结果提供了高速缓存支持。每当您启动 SAPODA 时这种高速缓存服务就会在后台运行，而当您结束会话时，就会清除已高速缓存的搜索。有关详细信息，请参阅第 48 页的『展开节点和选择对象』。
- 传送文件现在安装在 `\connectors\SAP\dependencies\transports_31` 目录中。有关详细信息，请参阅第 190 页的『安装连接器传送文件』。
- 连接器支持将 IDoc 消息分成更小的单元，每一单元就是转换为更小的业务对象的 JMS-MQ 消息。有关详细信息，请参阅第 110 页的『事件处理组件』。

发行版 5.5.x 中的新增内容

2004 年 6 月

已通过编辑性说明来更新该手册。

2004 年 2 月

已添加具有快速配置的新附录。

2003 年 12 月

- 关于内存管理的 SAP 注意事项
- 关于返回状态处理的信息
- ALE MQ 消息头信息

发行版 5.0.x 中的新增内容

重要提示: 因为该连接器尚未国际化，所以如果您不能保证将只处理 ISO Latin-1 数据，则不要对 InterChange Server V4.1.1 运行它。

注: 补丁发行版将此连接器称为 mySAP.com R/3 连接器。

发行版 4.8.x 中的新增内容

IBM WebSphere Business Integration mySAP.com 适配器（SAP R/3 V3.x）包括 SAP 连接器。此适配器与 InterChange Server（ICS）和 WebSphere MQ Integrator（WMQI）集成代理程序一起运行。集成代理程序是一种应用程序，它执行不同种类应用程序的集成，并提供包括数据路由的服务。

此适配器包括:

- 一个特定于 SAP R/3 V3.x 的应用程序组件
- SAPODA
- 一个样本业务对象，它位于 \connectors\SAP\samples\ 中
- IBM WebSphere Adapter Framework，它由以下各项组成:
  - 连接器框架
  - 开发工具（包括业务对象设计器和连接器配置器）
  - API（包括 ODK、JCDK 和 CDK）

本手册提供关于将此适配器与两个集成代理程序（InterChange Server（ICS）和 WebSphere MQ Integrator（WMQI））配合使用的信息。

重要提示: 因为该连接器尚未国际化，所以如果您不能保证将只处理 ISO Latin-1 数据，则不要对 InterChange Server V4.1.1 运行它。

发行版 4.7.x 中的新增内容

注: 补丁发行版将此连接器称为 mySAP.com R/3 连接器。

连接器已使用 SAPODA 替换 CWSAPGEN 实用程序。有关更多信息，请参阅第 41 页的第 5 章，『使用 SAPODA 生成业务对象定义』。

发行版 4.6.x 中的新增内容

- 当连接器开始处理业务对象时，连接器支持对适当的功能模块调用 ABAP 调试器。有关更多信息，请参阅第 285 页的『ABAPDebug』。

- 连接器支持更新 ALE 事件的状态。有关更多信息，请参阅第 117 页的『配置 SAP 以更新 IDoc 状态』。
- 本手册描述如何在特定于应用程序的信息中指定 ALE 通信伙伴信息。有关更多信息，请参阅第 133 页的『父包装程序业务对象』。

发行版 4.5.x 中的新增内容

- 此版本的连接器包含对 ALE 模块的较小更改，该模块允许由 SAP 来验证连接器以进行集成。这些更改维持向后兼容生成的对象以供 ALE 模块的先前版本使用。
- 当以多线程方式运行连接器时，用 `MaxNumConnections` 属性分配的其中一个可用线程将专用于轮询操作。

连接器版本 4.4.x

- 已简化模块配置属性。新配置值更短并更直观。但是，为了向后兼容，仍接受先前值。
- 新的分层动态检索模块处理分层业务对象或平面业务对象，以响应来自配置为使用连接器的协作的请求。有关此模块的信息，请参阅第 165 页的第 17 章，『配置分层动态检索模块』。
- ALE 模块提供非入侵事件处理。有关更多信息，请参阅第 115 页的第 11 章，『配置 ALE 模块』。
- ALE 模块使用事务标识（TID）来保证将每段数据传递或处理一次并且只有一次。有关更多信息，请参阅第 115 页的第 11 章，『配置 ALE 模块』。
- 连接器支持多线程。有关更多信息，请参阅第 288 页的『NumberOfListeners』。

  重要提示: 在 IBM WebSphere InterChange Server SAP 连接器版本 4.3.0 之前生成的 BAPI 业务对象处理程序不是线程安全的。要在使用多线程时保证数据一致性和完整性，您必须重新生成这些业务对象处理程序。业务对象不需要进行任何更改。
- 在 Solaris 上运行的连接器现在使用 SAP 的 Java API 并完全支持所有连接器模块。因为 IBM WebSphere InterChange Server 不交付 Java API，所以您必须从 SAP 的 Web 站点下载它。有关更多信息，请参阅第 15 页的『安装 SAP JCo』。
- 连接器不再使用 CharacterEncoding 配置属性。

连接器版本 4.3.x

此文档包含 4.2.x 发行版中的所有补丁。

连接器版本 4.2.x

- 在 4.2.7 发行版中，IBM WebSphere InterChange Server SAP 连接器使用 SAP 的 Java 连接器（jCO）API。这仅限于 Windows。因为 4.2.7 版本的连接器不支持 UNIX jCO，所以从 4.2.7 发行版起，SAP 不再支持 UNIX jCO。
- 在 4.2.7 发行版中，第 289 页的『RefreshLogonCycle』特定于连接器的配置属性已更改，以便连接器在每次处理事件之后，都可以注销然并重新注册，或者根本不注销。

- Guide to the IBM WebSphere InterChange Server Connector for SAP Version 3.x 包含用于支持 3 个新连接器模块的新信息。第 3 部分描述 ALE模块，第 4 部分描述 BAPI 模块，第 5 部分描述 RFC 服务器模块。

  ALE 模块提供用于将 IDoc 发送至 SAP 应用程序的非入侵解决方案。BAPI 模块提供用于调用 SAP 应用程序中 BAPI 的非入侵解决方案。RFC 服务器模块使 RFC 支持功能能够调用连接器。

  ALE 模块仅启用消耗（服务调用请求）。BAPI 模块启用服务调用请求和事件通知。RFC 服务器模块启用实时非入侵事件通知。
- 在 ALE 模块中，现在以空白表示数据记录业务对象中使用 `CxIgnore` 或 `CxBlank` 的简单属性。以前将 CxIgnore 解释为正斜杠（/）。SAP 根据使用的功能以不同方式处理正斜杠（/）字符。有关更多信息，请参阅第 135 页的『属性：数据记录业务对象』。
- 介绍了支持新模块的特定于连接器的新配置属性。有关更多信息，请参阅 gwService、NumberOfListeners、PollQuantity 、RefreshLogonCycle、RfcProgramId 和 RfcTraceOn。

连接器版本 4.1.x

- 添加了新的标准连接器配置属性：代理程序 URL、匿名连接、CA 证书位置、GW 名称和侦听器端口。

第 1 部分 连接器概述和安装

第 1 章 连接器概述

mySAP.com 连接器是 WebSphere Business Integration mySAP.com 适配器的运行时组件。mySAP.com 适配器包括连接器、消息文件、配置工具和 Object Discovery Agent（ODA）。连接器使 WebSphere 集成代理程序能够与 SAP 应用程序交换业务对象。

连接器由特定于应用程序的组件和连接器框架组成。特定于应用程序的组件包含针对特定应用程序而定制的代码。连接器框架（其代码是所有连接器共用的）充当集成代理程序和特定于应用程序的组件之间的媒介。连接器框架在集成代理程序和特定于应用程序的组件之间提供以下服务:

- 接收和发送业务对象
- 管理启动消息和管理消息的交换

本文档包含关于特定于应用程序的组件和连接器框架的信息。它将这两个组件称为连接器。

关于集成代理程序与连接器的关系的更多信息，请参阅*《系统管理指南》*（如果 InterChange Server 是集成代理程序）、*Implementing Adapters with WebSphere Message Brokers*（如果消息代理是集成代理程序）或 *Implementing Adapters with WebSphere Application Server*（如果 WebSphere Application Server 是集成代理程序）。

mySAP.com 适配器环境

在安装、配置和使用适配器之前，您必须了解其环境要求。本节包含以下主题:

代理程序兼容性

此适配器与 WebSphere Business Integration Adapter Framework V2.6 一起运行，并且需要下列其中一项:

- WebSphere InterChange Server V4.2.2 和 V4.3
- WebSphere MQ Integrator V2.1
- WebSphere MQ Integrator Broker V2.1
- WebSphere Business Integration Message Broker V5.0.1

- WebSphere Application Server Enterprise V5.0.2，与 WebSphere Studio Application Developer Integration Edition V5.0.1 一起运行
- WebSphere Business Integration Server Foundation V5.1.1
- DB2 Information Integrator V8.2.3 – 仅受 WebSphere Business Integration mySAP.com、Peoplesoft 和 Siebel 适配器的支持。

请参阅*发行说明*以了解任何异常。

注: 有关安装集成代理程序及其必备软件的指示信息，请参阅以下文档。

对于 IBM WebSphere InterChange Server（ICS），请参阅《*系统安装指南 UNIX 版*》或《*系统安装指南 Windows 版*》。

对于 Message Broker（WebSphere MQ Integrator Broker、WebSphere MQ Integrator 和 WebSphere Business Integration Message Broker），请参阅 *Implementing Adapters with WebSphere Message Brokers* 和该消息代理的安装文档。可以在以下 Web 站点上找到该消息代理的某些信息:

http://www.ibm.com/software/integration/mqfamily/library/manualsa/

对于 WebSphere Application Server，请参阅 Implementing Adapters with WebSphere Application Server 和以下网址处的文档:

http://www.ibm.com/software/webservers/appserv/library.html

适配器平台

除了代理程序之外，此适配器还需要下列其中一个操作系统:

- 所有操作系统环境都需要 Java 编译器（对于 Windows 2000 为 IBM JDK 1.4.2），用于编译定制适配器
- **AIX**:
  维护级别为 4 的 AIX 5.1
  维护级别为 1 的 AIX 5.2。此适配器支持在 64 位平台上使用 32 位 JVM
- **Solaris**:
  Solaris 8（2.8），具有 2004 年 2 月 11 日发布的或者更高版本的 Solaris Patch Cluster
  Solaris 9（2.9），具有 2004 年 2 月 11 日发布的或者更高版本的 Solaris Patch Cluster。此适配器支持在 64 位平台上使用 32 位 JVM
- **HP-UX**:
  HP-UX 11.i（11.11），具有 2003 年 6 月发布的 GOLDBASE11i 和 2003 年 6 月发布的 GOLDAPPS11i 捆绑软件
- **Linux**:
  具有“更新 1”的 Red Hat Enterprise Linux AS 3.0
  具有“更新 1”的 Red Hat Enterprise Linux ES 3.0
  具有“更新 1”的 Red Hat Enterprise Linux WS 3.0
  具有 SP3 的 SUSE Linux Enterprise Server x86 8.1
  具有 SP3 的 SUSE Linux Standard Server x86 8.1

注: 在 Linux Red Hat 上不支持 WebSphere Business Integration Adapter Framework V2.6 的 TMTP（Tivoli Monitoring for Transaction Performance）组件。

- **Windows**:
  带有 Service Pack 4 的 Windows 2000（Professional、Server 或 Advanced Server）
  带有 Service Pack 1A 的 Windows XP（对于 WebSphere Business Integration Adapter Framework，仅限于管理工具）
  Windows 2003（Standard Edition 或 Enterprise Edition）

适配器支持的软件

适配器支持在 SAP 应用程序服务器版本 R/3 3.1I 上运行的 SAP 应用程序。

适配器相关性

在安装连接器 IBM WebSphere Business Integration mySAP.com 适配器之前:

- 在您正在安装连接器的同一机器上安装 SAP 客户机。
- 为您的 SAP 版本安装最新的 SAP 支持软件包。

  SAP 交付以下各项的支持软件包: 基本部件、R/3 应用程序、ABAP 和 HR。这些软件包为 SAP 应用程序中的 ABAP 代码提供了错误修订。使用已更新的 SAP 内核。该内核是用 C++ 语言编写的可执行文件，它们执行传输任务，与操作系统进行交互，与数据库通信并运行系统。
- 在 SAP 应用程序中设置 CPIC 用户帐户。对此帐户授予必需的特权以处理受连接器支持的业务对象所需要的数据。

  例如，如果连接器必须执行某些 SAP 业务事务，则 SAP 应用程序中连接器的帐户必须具有执行这些事务的许可权。您必须使用此帐户信息来设置特定于连接器的配置属性 `ApplicationUserName` 和 `ApplicationPassword`。有关如何设置这些属性的更多信息，请参阅第 261 页的附录 D, 『连接器的标准配置属性』。
- 使用安装和管理连接器的特权在 SAP 中设置用户帐户。该帐户应具有以下特征:
  - 有效的 SAP 用户名和密码
  - ABAP 开发者访问权
  - 表配置访问权
  - 对事务 SM21 和 SM50 的管理访问权，以管理和监视连接器
- 如果使用 ALE 模块，则请参阅第 115 页的『运行 ALE 模块的先决条件』以获取关于安装 MQSeries 队列的其它信息。

在安装连接器之后:

- 安装 SAP Java API。

  SAP 将其 Java API 称为 Java 连接器（SAP JCo）。SAP 适配器当前支持 SAPJCo V.2.1.3。如果不能从 SAP Service Marketplace 下载本文档中提及的 SAPJCo 版本，请联系您的 IBM 代表。

  SAP JCo 合用连接并与其连接用来执行请求的适配器通信。将在连接器配置文件中设置适配器的所有连接属性。JCo.PoolManager 管理 SAP 应用程序中合用的连接的所有配置。

  有关安装此连接器相关性的详细信息，请参阅第 15 页的『安装 SAP JCo』

有关连接器属性的详细信息，请参阅第 21 页的第 3 章，『配置连接器』、第 261 页的附录 D，『连接器的标准配置属性』和第 283 页的附录 E，『特定于连接器的配置属性』。

Common Event Infrastructure

此适配器与 IBM 的 Common Event Infrastructure 兼容，它是用于事件管理的一种标准，允许与其它 IBM WebSphere 事件生成应用程序进行相互操作。如果启用了 Common Event Infrastructure 支持，则与 Common Event Infrastructure 兼容的其它应用程序就可以接收（或使用）由适配器生成的事件。

有关更多信息，参阅本指南中的 Common Event Infrastructure 附录。

应用程序响应测量

此适配器与应用程序响应测量应用编程接口（API）兼容，该 API 允许对应用程序的可用性、服务级别协议和容量规划进行管理。ARM 检测应用程序可以参与 IBM Tivoli Monitoring for Transaction Performance，允许收集和查看与事务度量有关的数据。

有关更多信息，参阅本指南中的『应用程序响应测量』附录。

与语言环境相关的数据

连接器已国际化，因此对于非 Unicode SAP 系统，它可以支持多字节字符集。注意，这不适用于基于 Unicode 的 SAP 系统。

当连接器将数据从使用一个字符代码集的位置传送至使用不同代码集的位置时，它执行字符转换以保持数据的意义。

注： CrossWorlds Station 日志仅以英语提供。

Java 虚拟机（JVM）内的 Java 运行时环境以 Unicode 字符代码集表示数据。Unicode 包含大多数已知字符代码集（单字节和多字节）中字符的编码。IBM WebSphere Business Integration 系统中大多数组件都是以 Java 语言编写的。因此，当在大多数 IBM WebSphere Business Integration 组件之间传送数据时，不需要进行字符转换。

因为此连接器是以 Java 语言编写的，所以它不需要转换以本机编码编写的应用程序数据（包括 IDoc 文件中的数据）。在连接器处理这样的应用程序数据之前，SAP JCo 库将该数据转换为 Unicode。图 1 举例说明了数据转换中涉及的组件。

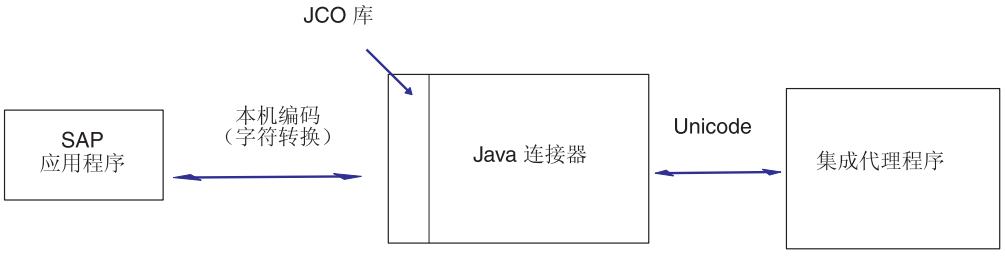


图 1. 配置代理程序属性

要将错误消息和参考消息以相应的语言并针对相应的国家或地区写入日志，请为您的环境配置 Locale 标准配置属性。关于这些属性的更多信息，请参阅第 261 页的附录 D, 『连接器的标准配置属性』。

重要提示: SAP 应用程序用于日语的字符代码集是 SAP-8000。此代码集不支持 MS932 字符。并且，SJIS 将某些字符映射至非标准的 Unicode 字符。因此，SAP JCo 库不能处理某些字符。如果 SAP 应用程序或 IBM WebSphere 业务对象中包括这些字符，则连接器使用 # 或 ? 字符替换它们。当数据包含这些字符时，则数据不会被正确处理，且连接器不会报告错误。

连接器组件

SAP 连接器是以 Java 语言编写的，它由两个部件组成: 可视连接器框架和连接器模块（特定于应用程序的连接器组件、连接器框架和业务对象处理程序）。可视连接器框架将元数据驱动的抽象层提供给由所有 WebSphere Business Integration 系统适配器使用的连接器框架。

可视连接器框架扩展适配器框架中的方法。连接器模块扩展可视连接器框架中的方法并与 SAP 应用程序通信。

图 2 举例说明了连接器的体系结构以及系统范围连接器框架和可视连接器框架的关系。visionConnectorAgent 类可以实现任何数目的连接器模块。

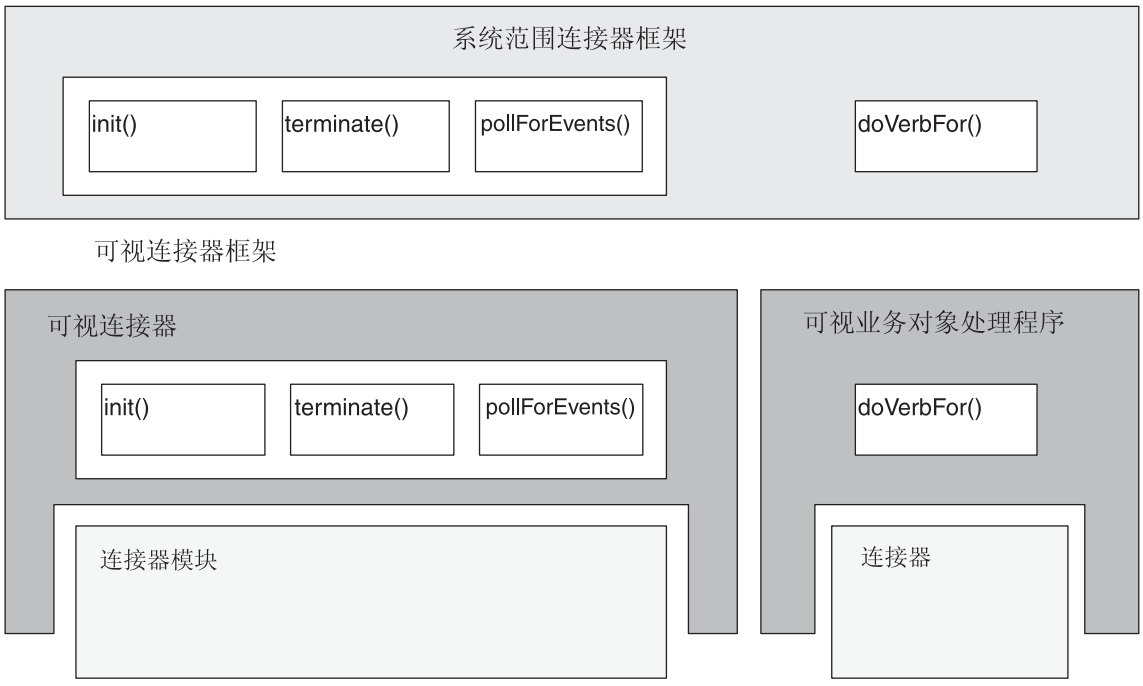


图 2. SAP 连接器的体系结构

可视连接器框架

可视连接器框架动态地将初始化、轮询和终止请求传递至连接器模块。它还动态地将业务对象传递至业务对象处理程序。业务对象处理程序是为支持业务对象而专门设计

的连接器模块。为了动态地传递请求和业务对象，连接器使用特定于应用程序的业务对象查询描述信息和某些特定于应用程序的连接器配置属性的值。

可视连接器框架由两个类组成: visionConnectorAgent 和 visionBOHandler。

图 3 举例说明了可视连接器框架及其与连接器模块的关联。

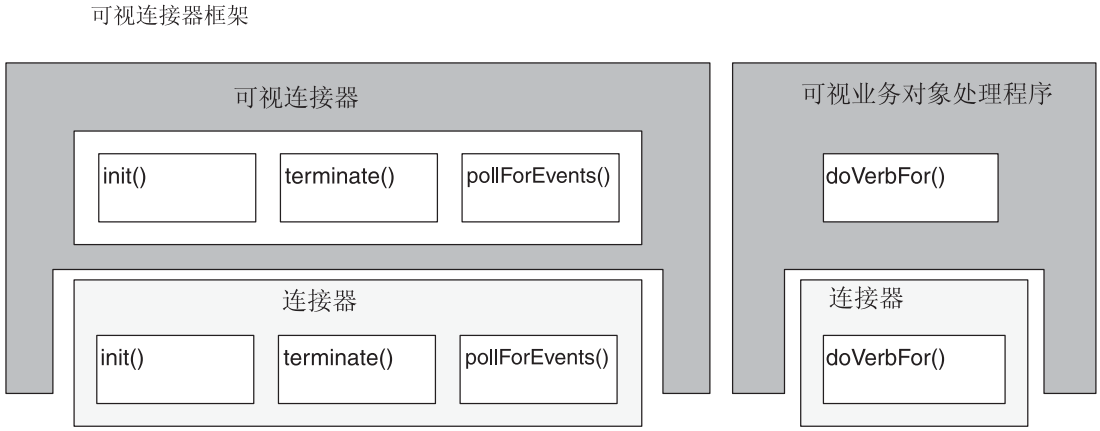


图 3. 可视连接器框架和连接器模块

可视连接器框架为连接器提供以下能力:

- 调用 `init()`、`pollForEvents()` 和 `terminate()` 方法的任何实现。
- 基于特定于应用程序的业务对象查询描述信息将业务对象传递至特定业务对象处理程序。

连接器模块

连接器模块是扩展可视连接器框架中方法的 Java 类。它通过提供特定功能（如登录到 SAP 应用程序、处理事件和业务对象以及终止与 SAP 应用程序的连接）来支持可视连接器框架。连接器模块执行可视连接器框架和 SAP 应用程序之间的请求。缺省情况下，可视连接器框架使用 `connectors\SAP` 目录作为连接器模块的根目录。

连接器模块可能并未使用所有框架方法。例如，一个模块可能使用 `init()` 和 `terminate()` 方法，而另一个模块仅使用 `pollForEvents()` 方法。但是，必须为每个连接器模块实现 visionConnectorAgent 和 visionBOHandler 类中的每个方法。必须作为哑元方法（即它们不执行操作就退出）实现连接器不使用的方法。

可视连接器框架的工作方式

连接器使用连接器模块与 SAP 应用程序交互。连接器模块调用 SAP 的本机接口并将数据（业务对象或事件数据）传递至 SAP 应用程序和从 SAP 应用程序中传递出来。连接器的灵活设计使不同模块能够用于不同任务，如初始化 SAP 应用程序的连接器或传递业务对象数据。

连接器和 SAP 应用程序之间的通信

连接器使用 SAP 的远程功能调用（RFC）库来与 SAP 应用程序通信。SAP 的 RFC 使外部程序能够调用 SAP 应用程序内的 ABAP 功能模块。

处理业务对象

连接器是元数据驱动的。在 WebSphere Business Integration 系统中，元数据是特定于应用程序的数据，它存储在业务对象中并有助于连接器模块与应用程序交互。元数据驱动的连接器模块基于业务对象定义中编码的元数据而不是连接器模块中硬编码的指令，来处理它支持的每个业务对象。

业务对象元数据包括业务对象结构、其属性特性设置和其特定于应用程序的信息的内容。因为连接器模块是元数据驱动的，所以它们可以处理新的或修改的业务对象而不必修改连接器模块代码。

可视连接器框架使用顶级业务对象中特定于应用程序的查询描述信息值，调用适当的连接器模块来处理业务对象。特定于应用程序的查询描述信息提供连接器模块的类名。

大多数顶级业务对象的特定于应用程序的查询描述信息必须标识连接器模块的类名。此特定于应用程序的查询描述信息的语法如下:

```
AppSpecificInfo = PartialPackageName.ClassName,
```

例如:

```
AppSpecificInfo = sap.sapextensionmodule.VSapBOHandler,
```

在此示例中，sap.sapextensionmodule 是部分包名，VSapBOHandler 是类名。完整的包名包括 com.crossworlds.connectors 前缀，WebSphere Business Integration 系统自动将该前缀添加至名称。换句话说，示例的完整文本为:

```
com.crossworlds.connectors.sap.sapextensionmodule.VSapBOHandler
```

注: 大多数特定于应用程序的顶级业务对象查询描述信息都必须在连接器类名后面使用逗号（**,**）定界符。但是，由 RFC 服务器模块使用的服务器查询描述却是由分号定界（**;**）。有关服务器查询描述的信息，请参阅第 147 页的『RFC 服务器模块的工作方式』和第 151 页的『受支持的查询描述』。

在下列情况下，您不必为特定于应用程序的查询描述信息指定包名和类名:

- 业务对象由 ALE 模块用来处理应用程序事件；但是，当您使用 ALE 模块来处理服务调用请求时，您必须指定包名和类名
- 业务对象由 ABAP 扩展模块使用，该模块使用缺省业务对象处理程序（sap.sapextensionmodule.VSapBOHandler）

重要提示: 为 RFC 服务器模块处理业务对象的客户生成的连接器模块必须指定一个完整的包名，该名称必须以 bapi 开头。例如，bapi.client.Bapi_customer_getdetail2。此示例中的完整包名是 bapi.client，类名是 Bapi_customer_getdetail2。

大多数业务对象处理是特定于每个连接器模块的。缺省情况下，连接器使用 ABAP 扩展模块。有关 ABAP 扩展模块的业务对象处理的更多信息，请参阅第 189 页的第 20 章，『安装和定制 ABAP 扩展模块』和第 199 页的『传递至 ABAP 处理程序的业务对象数据』。.

有关为 ALE 模块指定特定于应用程序的查询描述信息的详情，请参阅第 10 页的『事件处理』和第 140 页的『处理多个具有包装程序业务对象的 IDoc』。

处理多个并发交互

适配器框架可以创建不同的线程来处理应用程序事件和业务对象请求。当处理多个来自集成代理程序的请求时，它可以创建多个线程来处理多个业务对象请求。例如，当 InterChange Server 是集成代理程序时，连接器可以接收来自多个协作或来自一个多线程协作的多个业务对象请求。

图 4 举例说明了多线程体系结构。

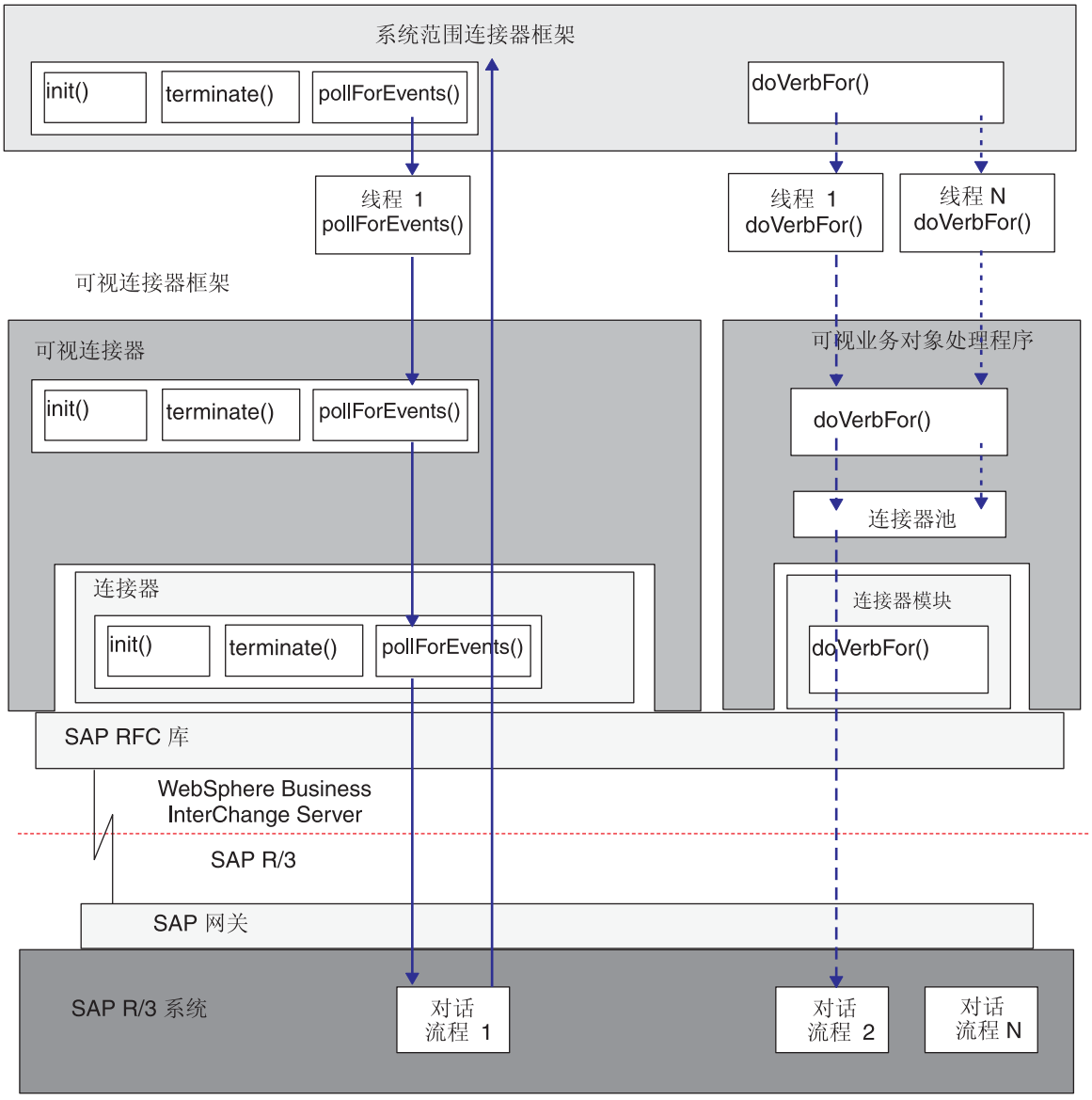


图 4. SAP 连接器的多线程体系结构

事件处理

连接器在处理轮询调用时执行以下步骤:

1. 适配器框架创建单个专用线程来处理轮询调用。此线程以在 `PollFrequency` 配置属性中指定的频率调用可视连接器框架的 `pollForEvents()` 方法。

2. 该线程轮询 SAP，SAP 使用对话进程来查找并返回事件。

   注: 如果连接器的 MaxNumberOfConnections 配置属性求值为大于 1 的数字，则可视连接器框架将一个连接专门用于 SAP 进行轮询。如果 MaxNumberOfConnections 求值为 1，则事件和服务调用请求处理共享与 SAP 的单个连接。

仅当连接器关闭时轮询线程才会停止运行。

注: 因为 RFC 服务器连接器代理程序将事件推送到 SAP 外部来代替轮询事件，所以它衍生自己的线程来代替使用由连接器框架创建的线程。因为 ALE 连接器代理程序使用 RFC 服务器连接器代理程序来访问事件，所以它也在处理事件时衍生它自己的线程来代替使用由连接器框架创建的线程。

请求处理

适配器框架与轮询无关，可以创建多个请求处理线程，每个请求业务对象使用一个线程。每个请求线程都会实例化适当的业务对象处理程序。

例如，当处理来自 InterChange Server 的业务对象请求时，业务对象处理程序的数目和类型取决于发送请求的协作的数目和类型:

- 如果多个协作发送业务对象，则每个请求线程都会实例化一个适当类型的业务对象处理程序。
- 如果一个多线程协作发送相同类型的多个业务对象，则请求线程会实例化相同数目的该类型业务对象处理程序。

如果连接器的 MaxNumberOfConnections 配置属性求值为大于 1 的数字，则可视连接器框架将一个连接专门用于 SAP 进行轮询，并将其余连接分配给仅用于请求处理的池。

如图 4 中所示，连接器在处理业务对象请求时执行以下步骤:

1. 适配器框架为每个业务对象请求创建不同的线程。每个线程调用可视业务对象处理程序的 doVerbFor() 方法。
2. 如果连接器的 MaxNumberOfConnections 配置属性求值为大于 1 的数字，则可视业务对象处理程序检查可视连接器框架的连接池以确定连接句柄是否可用。
   - 如果该句柄可用，则线程将请求发送至 SAP，SAP 使用对话进程来处理该请求。
   - 如果该句柄不可用，则线程将等至一个句柄变为可用为止。线程排序确定每个业务对象处理程序线程声明或等待可用连接句柄的顺序。

   如果连接器的 MaxNumberOfConnections 配置属性求值为 1，则可视业务对象处理程序与事件处理共享一个连接。
3. SAP 在完成处理并发送返回码之后释放对话进程。
4. 连接器在从 SAP 接收到返回码之后释放连接句柄。

设置可用连接数

使用 MaxNumberOfConnections 配置属性来指定最大可用连接句柄数。连接数不能超过对话进程数。

SAP 在处理交互时锁定对话进程，仅当交互完成时才释放它。因此，多个并发请求锁定相同数目的对话进程，直到处理完成为止。

重要提示: 在为 MaxNumberOfConnections 设置值之前，联系 SAP BASIS 管理员以确定一个适当的值，以使吞吐量最大而又不会对应用程序服务器的性能产生负面影响。

对多个连接的支持

缺省情况下，连接器支持多个线程。

第 2 章 安装连接器

本章描述 IBM WebSphere Business Integration mySAP.com 适配器（SAP R/3 V3.x）的连接器组件的安装和配置。本章假定当安装 WebSphere Business Integration 系统时已安装了所有必需的文件。

本章包含以下各节:

重要提示: 如果您正在升级连接器的版本，则您必须替换连接器 Jar 文件（`.jar`）。您还需要升级连接器传送文件以及您先前安装的任何业务对象传送包。根据对连接器所作的更改，您可能需要将 `SAPConnector.txt` 文件的新副本装入您的资源库。有关更多信息，请参阅发行说明。

安装任务

要为 mySAP.com 安装连接器，必须确认您的环境中是否存在必需的连接器先决条件、安装集成代理程序和运行连接器安装。

确认适配器先决条件

在安装适配器之前，确认您的系统上是否具有安装和运行连接器的所有环境先决条件。有关详细信息，请参阅第 3 页的『mySAP.com 适配器环境』。

安装集成代理程序

在代理程序的文档中描述了安装集成代理程序，这项任务包括安装 WebSphere Business Integration 系统和启动代理程序。有关连接器支持的代理程序的详细信息，请参阅第 3 页的『代理程序兼容性』。

有关安装代理程序的详细信息，请参阅您正在使用的代理程序的适当实现文档。

安装连接器

有关安装 WebSphere Business Integration 适配器产品的信息，参阅 *Installing WebSphere Business Integration Adapters* 指南，它位于以下站点的 WebSphere Business Integration Adapters Infocenter 中:

http://www.ibm.com/websphere/integration/wbiadapters/infocenter

安装连接器

在安装了 WebSphere Business Integration 系统之后，您可以在任何时候从产品 CD 安装其它适配器。为此，插入产品 CD，运行安装程序，并选择您要安装的适配器。

注: 除非另有指示，否则本章中其余各节适用于连接器的 Windows 和 UNIX 安装。

IBM WebSphere Business Integration mySAP.com 适配器连接器可以安装在 UNIX 或 Windows 机器上。连接器由三个需要安装的部件组成: 特定于应用程序的连接器组件、SAP 的 RFC 库和任何随产品交付且支持连接器而所需的 SAP 传送文件。

在安装必需的连接器文件之后，您必须下载和安装 Java 连接器（SAPJCo）文件。有关下载 SAPJCo 文件的更多信息，请参阅第 15 页的『安装 SAP JCo』。有关安装 SAPJCo 文件的更多信息，请参阅第 15 页的『安装 SAP JCo』。

在安装 UNIX 系统上安装

要在 UNIX 系统上安装连接器，运行 IBM WebSphere Business Integration 适配器的安装程序，并选择 IBM WebSphere Business Integration mySAP.com 适配器。表 1 列示在 UNIX 环境中运行的连接器所使用的文件。

表 1. WBIA: UNIX 文件结构

目录 / 文件名	描述
connectors/SAP/bapi/client	包含 BAPI 模块业务对象处理程序文件的目录
connectors/SAP/bapi/server	包含 RFC 服务器模块业务对象处理程序文件的目录
connectors/SAP/dependencies	包含所有特定于版本的传送文件的目录
connectors/SAPmessages	包含 SAPConnector.txt 文件的目录
connectors/SAP/samples	包含样本 ABAP 对象的目录
connectors/SAP/utilities	包含 generatedfiles 子目录的目录，您可以将 SAPODA 生成的文件放入该子目录
connectors/SAP/CWSAP.jar	连接器类文件
connectors/SAP/start_SAP.sh	连接器的系统启动脚本。 将从通用连接器管理器脚本调用此脚本。产品安装程序为此连接器管理器脚本创建定制的包装程序。 当连接器与 WebSphere InterChange Server 一起运行时，请使用此定制的包装程序来启动和停止连接器。当连接器与 WebSphere MQ 消息代理一起运行时，请使用此定制的包装程序来仅启动连接器；使用 mqsiremotestopadapter 来停止连接器
repository/SAP	包含 sap_idoccontrol.xsd 文件的目录
/lib	包含 WBIA.jar 文件
/bin	包含 CWConnEnv.sh 文件
/bin/Data/app	包含 SAPConnectorTemplate 文件

您必须从安装程序的“连接器配置”屏幕配置连接器，然后才能使用连接器。从此屏幕:

- 从“选择连接器名称”列表选择 SAP。
- 单击“安装”以便让安装程序生成定制的 SAP 安装程序 connector_manager_SAP。

注: 有关安装连接器组件的更多信息，参阅 *WebSphere Business Integration Adapters Installation Guide*。

在 Windows 系统上安装

要在 Windows 系统上安装连接器，运行 IBM WebSphere Business Integration 适配器的安装程序，并选择 IBM WebSphere Business Integration mySAP.com 适配器。安装程序安装与连接器相关的标准文件。表 2 列示 Windows 环境中安装的标准文件。

表 2. WebSphere Business Integration Adaptor: Windows 文件结构

目录 / 文件名	描述
connectors\SAP\bapi\client	包含 BAPI 模块业务对象处理程序文件的目录
connectors\SAP\bapi\server	包含 RFC 服务器模块业务对象处理程序文件的目录
connectors\SAP\dependencies	包含所有特定于版本的传送文件的目录
connectors\SAPmessages	包含 SAPConnector.txt 文件的目录
connectors\SAP\samples	包含样本 ABAP 对象的目录
connectors\SAP\CWSAP.jar	连接器类文件
connectors\SAP\start_SAP.bat	用来启动连接器的批处理文件
repository\SAP	包含 CN_SAPSAP.txt 文件的目录
\lib	包含 WBIA.jar 文件
\bin	包含 CWConnEvn.bat 文件

安装程序将特定于应用程序的连接器组件的菜单选项添加至 IBM WebSphere Business Integration 适配器菜单。要以最快的方式启动连接器，请在桌面上创建此组件的快捷方式。

注: 有关安装连接器组件的更多信息，参阅 *WebSphere Business Integration Adapters Installation Guide*。

安装 SAP JCo

在安装了连接器并且已将所有文件安装到适当目录中之后，必须下载并安装 SAP JavaAPI。这是 SAPODA 的必备软件，在第 41 页的第 5 章，『使用 SAPODA 生成业务对象定义』中对此进行了描述。

SAP 将其 Java API 称为 Java 连接器（SAP JCo）。SAP 的连接器当前支持 SAP JCo V.2.1.3。

1. 为您的连接器运行所在的操作系统下载 SAP JCo。可从 SAP 的 Web 站点下载 SAP JCo，网址为: http://service.sap.com/connectors。您必须具有 SAPNet 帐户才能访问 SAP JCo（如果您尚未具有该帐户，则联系本地 SAP Basis 管理员）。

   如果无法从 SAP Service Marketplace 下载连接器支持的 SAP JCo 版本，请检查受支持的最新版本 JCo 的当前适配器补丁说明，或者与 IBM 代表联系。
2. 将已解压缩的下列 SAP JCo 文件复制到您的环境中:

   UNIX:

   从压缩文件中解压缩可执行的 jar 文件（sapjco.jar）和运行时库（librfccm 和 libsapjcorfc）。

   如果您已遵循在安装 SAPODA 的同一机器上安装适配器的指示信息，则将这些文件从 \connectors\SAP 目录复制到 \ODA\SAP 目录中。如果您在与连接器不同的机器上安装 SAPODA，则在解压缩 SAP JCo 文件之后，将这三个文件复制到 \ODA\SAP 目录中。

Windows:

从 zip 文件中，解压缩可执行的 jar 文件（.jar 扩展名）和运行时库（.dll 扩展名）。如果您已遵循在安装 SAPODA 的同一机器上安装适配器的指示信息，则将这些文件从 \connectors\SAP 目录复制到 \ODA\SAP 目录中。

如果您在与连接器不同的机器上安装 SAPODA，则在解压缩 SAP JCo 文件之后，将这 3 个文件（librfc32.dll、sapjco.jar 和 sapjcorfc.dll）复制到 \ODA\SAP 目录中。对于 Windows，librfc32.dll 需要一个或多个 C 运行时 dll。C 运行时 dll 取决于正在使用的 SAP 发行版的版本。

在远程机器上安装连接器

您可以在远程机器上安装和运行连接器。在一台机器上安装集成代理程序，在另一台机器上安装连接器。建议（但不要求）两个机器在同一子网中。

安装多个连接器

要使集成代理程序能够同时处理多个 SAP 业务对象，您可能要为 SAP 系统安装和配置多个连接器组件，并定制每个连接器以处理特定业务对象。

每个连接器组件都可以预订某些业务对象，这取决于它们的类型（如客户或采购订单）。因为您可以让多个连接器来访问同一 SAP 应用程序，所以每个连接器可以处理多个事件并将它们传递至集成代理程序。另外，多个连接器可以同时支持多个业务对象请求。这会增加吞吐量并提高将数据输入和输出 SAP 应用程序的传送速度。

建议您为每个连接器组件选择一个唯一命名约定。例如，如果您正在使用两个连接器，则可以将它们命名为 SAP1Connector 和 SAP2Connector。

要安装并设置多个连接器组件，执行以下操作:

1. 按本章所述安装每个连接器。这包括连接器共享库文件。给您安装的每个连接器提供一个唯一名称，并验证您是否具有支持连接器文件。

   如果在同一机器上安装多个连接器，则您仅需要生成共享库文件的副本并重命名它们。您不需要再次安装传送包。
2. 创建启动脚本的副本:
   - 在 UNIX 上，生成用于启动连接器的现有 connector_manager_SAP 文件的副本，并重命名该文件以与连接器的名称匹配。
   - 在 Windows 上，生成 start_SAP.bat 文件的现有快捷方式的副本，并重命名该快捷方式文件以与连接器的名称匹配。添加连接器的名称作为连接器快捷方式的参数。
3. 生成连接器模板的副本，重命名它以与新的连接器名称匹配，然后将它复制到 repository 目录（如果 IBM WebSphere MQ Integrator 是集成代理程序），或将它装入 IBM WebSphere 资源库（如果 IBM WebSphere InterChange Server 是集成代理程序）。
4. 生成连接器类文件 CWSAP.jar 的副本并将它重命名为唯一的连接器名称，如 CWSAP1.jar。
5. 初始化连接器配置属性，以便所有连接器轮询同一 SAP 应用程序来获取事件。

6. 仅当 IBM WebSphere InterChange Server 是集成代理程序时，则添加每个连接器的映射引用。
7. 指定每个连接器所支持的业务对象。
8. 仅当 WebSphere InterChange Server 是集成代理程序时，才将协作指定给适当的连接器。目前，一个协作只能由一个连接器进行处理。如果已设置协作，则您可能需要停止它们并接着重新绑定端口。
9. 如果您要使用 ABAP 扩展模块来进行业务对象处理，则设置您安装的每个连接器的事件分布。使用 IBM CrossWorlds Station（传送包 /nYXR1）。有关为业务对象、集成代理程序和连接器的每个组合设置事件分布的指示信息，请参阅第 192 页的『设置事件分布』。

重要提示: 如果未将业务对象配置为转至特定连接器，则将把该业务对象发送至用于轮询事件的下一个连接器。如果将业务对象配置为转至特定连接器（例如，在测试阶段），但在生产阶段不使用连接器，那么连接器的事件队列将填满。要改正此情况，在“事件分布”窗口中删除连接器／业务对象配置（事务 YXRH）。

升级连接器

本节描述如何升级连接器:

- 『为 ALE 模块的 TID 管理升级连接器』
- 第 18 页的『升级至基于 Java 的连接器』

为 ALE 模块的 TID 管理升级连接器

ALE 模块为它从 SAP 应用程序接收的每个事务持久地存储 IDoc 对象和事务标识（TID）。在版本 4.8.x 之前的连接器发行版中，连接器使用 IBM WebSphere 协作、业务对象和映射来将数据存储在资源库中。版本 4.8.x 的连接器通过使用 MQSeries 队列来替换先前的 TID 管理方式。

警告: 要使 ALE 模块能够处理发送至 SAP 应用程序或从 SAP 应用程序中发送出来的 IDoc，您必须升级连接器。但是，在升级连接器之前，您必须允许当前 IDoc 处理循环完成。

在升级连接器以使 ALE 模块能够处理 IDoc 之前，您必须完成事件目录和 WIP 目录中当前文件的处理。并且，检查归档目录以查找失败的和未预订的事件。

要完成事件目录和 WIP 目录中当前文件的处理:

- 临时停止进出连接器的 IDoc 传输。
- 验证升级时以下目录中是否没有 IDoc（文件）:

UNIX

```
$CROSSWORLDS/connectors/SAP/ale/events
$CROSSWORLDS/connectors/SAP/ale/wip
```

Windows

```
%CROSSWORLDS%\connectors\SAP\ale\events
%CROSSWORLDS%\connectors\SAP\ale\wip
```

要完成任何失败事件和未预订事件的处理:

- 临时停止进出连接器的 IDoc 传输。
- 验证升级时以下目录中 IDoc（文件）的状态:

UNIX

```
$CROSSWORLDS/connectors/SAP/ale/archive
```

Windows

```
%CROSSWORLDS%\connectors\SAP\ale\archive
```

- 改正失败事件或未预订事件的任何错误。
- 将改正的文件移至事件目录进行处理。

注: 当使用事务 SM58 时，如果您在 SAP 系统中发现未成功处理的 IDoc，则等到升级连接器后重新提交这些 IDoc。在完成连接器的升级之后，改正错误并重新提交 IDoc，以便使用新的 TID 管理通过 MQSeries 队列来进行处理。

一旦清除这些目录，则应用升级并遵循以下章节中的配置指示信息:

- 第 283 页的『特定于连接器的配置属性』
- 第 109 页的第 10 章，『ALE 模块概述』.
- 第 115 页的第 11 章，『配置 ALE 模块』

升级至基于 Java 的连接器

将作为基于 Java 的连接器（连接器版本 4.0.0 和更高版本）交付 *IBM WebSphere Business Integration mySAP.com 适配器*。在前发行版中，连接器是以 C++ 语言编写的。目录结构是 \connectors\SAP。

以下过程用于将连接器的 C++ 版本升级至连接器的 Java 版本。

1. 重命名当前连接器目录。

   例如，\connectors\SAP 变为 connectors\SAP.old。
2. 重命名连接器消息文件。

   例如，\connectors\messages\SAPconnector.txt 变为 \connectors\messages\SAPconnector.txt.old。
3. 将新的连接器目录和文件复制到 \connectors 目录。
4. 将新的连接器消息文件复制到 \connectors\messages 目录。

Windows

修改连接器快捷方式以指向连接器目录中的 start_SAP.bat 文件。例如，如果您正在使用支持 SAP R/3 V4.x 的连接器，则修改快捷方式以指向连接器启动文件 \connectors\SAP\start_SAP.bat。

全面的安装和卸载信息

本节中的信息描述如何安装 WebSphere Business Integration 适配器。

通过运行安装程序的特定于平台的可执行文件来安装 WebSphere Business Integration 适配器。表 3 列示每个操作系统的安装程序可执行文件。这些安装程序可执行文件位于产品 CD 上的 WebSphereBI 目录。

表 3. WebSphere Business Integration 适配器安装程序的特定于平台的可执行文件

操作系统	WBIA 安装程序可执行文件
Windows	setupwin32.exe
AIX	setupAIX.bin
Solaris	setupsolarisSparc.bin
HP-UX	setupHP.bin

注: 这些过程假定您正在从产品 CD 进行安装。如果从 Passport Advantage 获取软件，则确保已下载它。请参阅 Passport Advantage 信息以获取那些下载指示信息。

注: 如果正在安装要与 InterChange Server 通信的适配器，则必须首先安装该代理程序。有关如何安装该代理程序的信息，请参阅适当平台上 InterChange Server 的安装指南。

重要提示: 确保在安装适配器之前作为 WebSphere Business Integration 系统管理员登录。当在 UNIX 计算机上安装时，将基于执行安装的用户帐户许可权来设置所创建的文件夹和文件的许可权。

重要提示: 您不得作为 root 用户安装 WBIA。当作为 root 用户安装时，添加至“对象数据管理器”（ODM）的条目会阻止您使用 SMIT 来卸载其它应用程序，因此不应该作为 root 用户安装 WBIA。

调用图形 WBIA 安装程序

图形 WBIA 安装程序向您提供了一个向导，该向导允许您对 WBIA 产品的安装作出选择。尽管该安装程序是基于 Java 的并因此独立于平台，但对于每种平台来说，调用该安装程序的方式各不相同。本节描述适合 Windows 和 UNIX 计算机的方法。

在 Windows 环境中调用安装程序

要在 Windows 环境中调用安装程序，浏览至产品 CD 的 WebSphereBI 目录并执行 setupwin32.exe。

在 UNIX 环境中调用安装程序

将通过位于 WebSphereBI 目录的特定于平台的 .bin 文件来调用 UNIX 环境中的 WBIA 安装程序。第 19 页的表 3 提供了每种平台的 .bin 文件的名称。

根据您使用 UNIX 计算机的方式，遵循以下其中一节中的步骤来调用安装程序:

- 『如果正在 UNIX 计算机上运行 CDE』
- 『如果正在通过 X 仿真软件连接至 UNIX 计算机』

如果正在 UNIX 计算机上运行 CDE: 如果正在运行“公共桌面环境”（CDE）并直接在 UNIX 计算机上操作，则可浏览至产品 CD 的 WebSphereBI 目录并双击特定于操作系统的 .bin 文件。

还可以浏览至产品 CD 的 WebSphereBI 目录并在命令行上执行 .bin 文件。以下示例显示如何在 Solaris 计算机上执行此操作:

```
# ./setupsolarisSparc.bin
```

如果正在通过 X 仿真软件连接至 UNIX 计算机: 如果正在使用 Windows 计算机来通过 X 仿真软件连接至 UNIX 计算机，则执行以下操作来调用安装程序:

1. 确定您正在用来连接至 UNIX 计算机的 Windows 计算机的 IP 地址。

   可以在 Windows 命令行界面中执行 ipconfig 命令来显示 Windows 计算机的 IP 地址。
2. 将 UNIX 计算机上的 DISPLAY 环境变量设置为在步骤 1 中确定的 IP 地址。

   必须确保在 IP 地址的后面加上冒号和 Windows 客户机计算机上的监视器或显示器的标识。如果 Windows 客户机计算机仅具有单个监视器，则显示值为 0.0。

   以下示例显示 DISPLAY 环境变量，该变量在 IP 地址为 9.26.244.30 的 Windows 计算机上设置为单个监视器:

   ```
   DISPLAY=9.26.244.30:0.0
   ```
3. 通过执行以下命令来导出 DISPLAY 环境变量:

   ```
   export DISPLAY
   ```
4. 在 Windows 计算机上启动 X 仿真客户机并连接至 UNIX 计算机。
5. 在 X 仿真客户机的命令行上浏览至产品 CD 的 WebSphereBI 目录。
6. 执行特定于操作系统的 .bin 文件。例如，如果 UNIX 计算机正在运行 AIX，则您将执行以下命令:

   ```
   # ./setupAIX.bin
   ```

   图形安装程序在用来连接至 UNIX 计算机的 Windows 计算机上启动。

第 3 章 配置连接器

本章描述 IBM WebSphere Business Integration mySAP.com 适配器（SAP R/3 V3.x）的连接器组件的安装和配置。本章假定当安装 WebSphere Business Integration 系统时已安装了所有必需的文件。

连接器配置器概述

连接器配置器允许您配置适配器的连接器组件以便与以下集成代理程序配合使用:

- WebSphere InterChange Server（ICS）
- WebSphere MQ Integrator、WebSphere MQ Integrator Broker 和 WebSphere Business Integration Message Broker，总称为 WebSphere Message Broker（WMQI）
- WebSphere Application Server（WAS）

如果适配器支持 DB2 Information Integrator，则使用 WMQI 选项和 DB2 II 标准属性（请参阅『标准属性』附录中的“注释”列）。

使用连接器配置器执行以下操作:

- 创建**特定于连接器的属性模板**以用于配置您的连接器。
- 创建**连接器配置文件**；您必须为您安装的每个连接器创建一个配置文件。
- 在配置文件中设置属性。
  您可能需要修改连接器模板中为属性设置的缺省值。您还必须指定受支持的业务对象定义以及（对于 ICS）映射以便与协作配合使用，并按需要指定消息传递参数、日志和跟踪参数以及数据处理程序参数。

根据您正在运行的集成代理程序，您运行连接器配置器的方式和使用的配置文件类型可能不同。例如，如果 WMQI 是代理程序，则您应直接运行连接器配置器，而不是从系统管理器中运行（请参阅第 22 页的『以独立方式运行配置器』）。

连接器配置属性包括标准配置属性（所有连接器都具有的属性）和特定于连接器的属性（连接器为了特定应用程序或技术而需要的属性）。

因为所有连接器都使用**标准属性**，所以您不必从头开始定义那些属性；只要您一创建配置文件，连接器配置器就会将它们合并到该文件中。但是，您需要在连接器配置器中设置每个标准属性的值。

标准属性的范围可能并非对于所有代理程序和所有配置都是相同的。某些属性仅在其它属性被提供特定值时才可用。连接器配置器中的“标准属性”窗口将显示可用于特定配置的属性。

但是，对于**特定于连接器的属性**，您需要首先定义这些属性，然后设置它们的值。通过为特定适配器创建特定于连接器的属性模板来执行此操作。系统中可能已设置了模板，在这种情况下，您只需使用它即可。如果没有设置模板，则遵循第 23 页的『创建新模板』中的步骤设置一个新的模板。

在 UNIX 上运行连接器

连接器配置器仅在 Windows 环境中运行。如果您要在 UNIX 环境中运行连接器，则在 Windows 中使用连接器配置器来修改配置文件，然后将该文件复制到 UNIX 环境。

连接器配置器中的某些属性使用目录路径，目录路径的缺省值为目录路径的 Windows 约定。如果您使用 UNIX 环境中的配置文件，则将目录路径修改为与这些路径的 UNIX 约定相匹配。在工具栏下拉列表中选择目标操作系统，以便将正确的操作系统规则用于扩展验证。

启动连接器配置器

您可以通过以下两种方式中的任何一种来启动并运行连接器配置器:

- 以独立方式独立地启动并运行
- 从系统管理器启动并运行

以独立方式运行配置器

您可以运行连接器配置器而不运行系统管理器，并且处理连接器配置文件，而不必考虑代理程序。

为此:

- 从**开始 > 程序**，单击 **IBM WebSphere InterChange Server > IBM WebSphere Business Integration 工具 > 连接器配置器**。
- 选择**文件 > 新建 > 连接器配置**。
- 当单击**系统连接集成代理程序**旁边的下拉菜单时，您可以选择 ICS、WebSphere Message Broker 或 WAS，这取决于您的代理程序。

您可以选择独立地运行连接器配置器来生成文件，然后连接至系统管理器以将它保存在“系统管理器”项目中（请参阅第 27 页的『完成配置文件』）。

从系统管理器运行配置器

您可以从系统管理器运行连接器配置器。

要运行连接器配置器:

1. 打开系统管理器。
2. 在“系统管理器”窗口中，展开**集成组件库**图标并突出显示**连接器**。
3. 从系统管理器菜单栏，单击**工具 > 连接器配置器**。“连接器配置器”窗口打开并显示**新建连接器**对话框。
4. 当单击**系统连接集成代理程序**旁边的下拉菜单时，您可以选择 ICS、WebSphere Message Broker 或 WAS，这取决于您的代理程序。

要编辑现有的配置文件:

- 在“系统管理器”窗口中，选择“连接器”文件夹中列示的任何一个配置文件并右键单击它。连接器配置器打开并显示该配置文件，其中集成代理程序类型和文件名显示在顶部。

- 从连接器配置器，选择**文件 > 打开**。从项目或从存储连接器配置文件的目录中选择该文件的名称。
- 单击“标准属性”选项卡以查看此配置文件中包括了哪些属性。

创建特定于连接器的属性模板

要为连接器创建配置文件，您需要特定于连接器的属性模板以及系统提供的标准属性。

您可以为连接器的特定于连接器的属性创建全新的模板，您也可以使用现有的连接器定义作为模板。

- 要创建新模板，请参阅第 23 页的『创建新模板』。
- 要使用现有文件，只需修改现有模板并以新名称保存它。您可以在 `\WebSphereAdapters\bin\Data\App` 目录中找到现有模板。

创建新模板

本节描述如何在模板中创建属性、为那些属性定义常规特征和值以及指定属性之间的任何相关性。然后保存该模板并使用它作为创建新连接器配置文件的基础。

要在连接器配置器中创建模板:

1. 单击**文件 > 新建 > 特定于连接器的属性模板**。
2. 出现**特定于连接器的属性模板**对话框。
   - 在**输入新模板名称**下的**名称**字段中输入新模板的名称。当您打开该对话框以从模板创建新的配置文件时，您将再次看到此名称。
   - 要查看任何模板中特定于连接器的属性定义，在**模板名称**屏幕中选择该模板的名称。该模板中包含的属性定义的列表出现在**模板预览**屏幕中。
3. 您可以使用其属性定义类似于您的连接器所需属性定义的现有模板，作为您的模板起始点。如果您未看到任何模板显示连接器使用的特定于连接器的属性，则您将需要创建一个模板。
   - 如果您打算修改现有的模板，则从**选择要修改的现有模板**: **查找模板**下面的**模板名称**表中的列表选择该模板的名称。
   - 此表显示所有当前可用的模板的名称。您还可以搜索模板。

指定常规特征

当单击**下一步**以选择模板时，出现**属性 - 特定于连接器的属性模板**对话框。该对话框具有用于已定义属性的“常规”特征和“值”限制的选项卡。“常规”屏幕具有以下字段:

- **常规**:
  属性类型
  属性子类型
  更新方法
  描述
- **标志**
  标准标志
- **定制标志**
  标志

当**属性类型**为“字符串”时，可以选择**属性子类型**。它是一个可选值，它会在您保存配置文件时提供语法检查。缺省值为空白，表示该属性尚无子类型。

在选择了属性的常规特征之后，单击**值**选项卡。

指定值

值选项卡使您能够设置属性的最大长度、最大多值数、缺省值或值范围。它还允许使用可编辑的值。为此:

1. 单击**值**选项卡。“值”的显示面板替换“常规”的显示面板。
2. 在**编辑属性**屏幕中选择该属性的名称。
3. 在**最大长度**和**最大多值数**字段中，输入您的值。

要创建新的属性值:

1. 在**编辑属性**列表中选择该属性并右键单击它。
2. 从对话框选择**添加**。
3. 输入新属性值的名称并单击“确定”。该值出现在右边的**值**面板中。

值面板显示一个具有三列的表:

值列显示您在**属性值**对话框中输入的值和您先前创建的任何值。

缺省值列允许您将任何值指定为缺省值。

值范围显示您在**属性值**对话框中输入的范围。

在已创建值且它出现在网格中之后，可以在表屏幕中编辑它。

要对表中的现有值作出更改，通过单击行号来选择整个行。然后在**值**字段中右键单击并单击**编辑值**。

设置相关性

当您已对**常规**和**值**选项卡作出更改后，单击**下一步**。出现**相关性 - 特定于连接器的属性模板**对话框。

从属属性是包括在模板中的一种属性，该属性仅当另一个属性的值满足特定条件时才在配置文件中使用。例如，仅当 JMS 是传输机制且 DuplicateEventElimination 设置为 True 时，PollQuantity 才会出现在模板中。
要将属性指定为从属的并设置它依赖的条件，执行以下操作:

1. 在**可用属性**屏幕中，选择将成为从属的属性。
2. 在**选择属性**字段中，使用下拉菜单来选择将控制条件值的属性。
3. 在**条件运算符**字段中，选择以下其中一项:

   ==（等于）

   !=（不等于）

   >（大于）

   <（小于）

>=（大于或等于）

<=（小于或等于）

4. 在**条件值**字段中，输入用于将该从属属性包括在模板中而需要的值。
5. 在**可用属性**屏幕中突出显示该从属属性后，单击一个箭头以将它移至**从属属性**屏幕。
6. 单击**完成**。连接器配置器将您输入的信息作为 XML 文档存储在连接器配置器的 \bin 安装目录中的 \data\app 下。

设置路径名

关于设置路径名的一些通用规则是:

- 在 Windows 和 UNIX 中，文件名的最大长度为 255 个字符。
- 在 Windows 中，绝对路径名必须遵循 [Drive:][Directory]\filename 这种格式: 例如, C:\WebSphereAdapters\bin\Data\Std\StdConnProps.xml
  在 UNIX 中，第一个字符应该为 /。
- 队列名可能没有前导空白或嵌入空白。

创建新的配置文件

当创建新的配置文件时，您必须命名它并选择一个集成代理程序。

- 在“系统管理器”窗口中，右键单击**连接器**文件夹并选择**创建新的连接器**。连接器配置器打开，并显示**新建连接器**对话框。
- 以独立方式: 在连接器配置器中，选择**文件 > 新建 > 连接器配置**。在“新建连接器”窗口中，输入新连接器的名称。

您还需要选择一个集成代理程序。您选择的代理程序确定将出现在配置文件中的属性。要选择代理程序:

- 在**集成代理程序**字段中，选择 ICS、WebSphere Message Broker 或 WAS 连接。
- 填写**新建连接器**窗口中的其余字段，如本章后面所述。

从特定于连接器的模板创建配置文件

一旦创建了特定于连接器的模板，您就可以使用它来创建配置文件:

1. 单击**文件 > 新建 > 连接器配置**。
2. 出现**新建连接器**对话框，该对话框具有以下字段:
   - **名称**

     输入连接器的名称。名称是区分大小写的。您输入的名称必须是唯一的，且必须与系统上安装的连接器的文件名一致。

     重要提示: 连接器配置器不会检查您输入的名称的拼写。您必须确保该名称是正确的。

   - **系统连接**

     单击 ICS、WebSphere Message Broker 或 WAS。

   - **选择特定于连接器的属性模板**

输入已为您的连接器设计的模板名称。可用模板显示在**模板名称**屏幕中。当您在“模板名称”屏幕中选择名称时，**属性模板预览**屏幕显示该模板中已定义的特定于连接器的属性。

选择您要使用的模板并单击**确定**。

3. 出现一个针对您正在配置的连接器的配置屏幕。标题栏显示集成代理程序和连接器的名称。您可以填充所有字段值以立即完成定义，您也可以保存该文件并在以后填充这些字段。
4. 要保存该文件，单击**文件 > 保存 > 至文件**或**文件 > 保存 > 至项目**。要保存到项目，系统管理器必须正在运行。
   如果您另存为文件，则会出现**保存文件连接器**对话框。选择 *.cfg 作为文件类型，验证“文件名”字段中名称是否拼写正确和大小写是否正确，浏览至您想要放置该文件的目录并单击**保存**。连接器配置器的消息面板中的状态屏幕指示已成功创建配置文件。

   重要提示: 您在此处建立的目录路径和名称必须与您在连接器的启动文件中提供的连接器配置文件路径和名称匹配。
5. 要完成连接器定义，在“连接器配置器”窗口的每个选项卡的字段中输入值，如本章中稍后所述。

使用现有文件

您可能具有一个使用以下一种或多种格式的现有文件:

- 连接器定义文件。
  这是一个文本文件，它列示特定连接器的属性和适用缺省值。某些连接器将这样的文件包括在其交付软件包中的 \repository 目录中（该文件通常具有扩展名 .txt; 例如，对于 XML 连接器为 CN_XML.txt）。
- ICS 资源库文件。
  连接器的先前 ICS 实现中使用的定义可以在该连接器的配置中以前使用的资源库文件中获得。这样的文件通常具有扩展名 .in 或 .out。
- 连接器的先前配置文件。
  这样的文件通常具有扩展名 *.cfg。

尽管其中任何文件源都可能包含您的连接器的大多数或全部特定于连接器的属性，但在您已打开连接器配置文件并设置属性之前，该文件仍将是不完整的，如本章后面所述。

要使用现有文件来配置连接器，您必须在连接器配置器中打开该文件，修改配置然后重新保存该文件。

遵循以下步骤从一个目录中打开 *.txt、*.cfg 或 *.in 文件:

1. 在连接器配置器中，单击**文件 > 打开 > 从文件**。
2. 在**打开文件连接器**对话框中，选择以下其中一种文件类型以查看可用文件:
   - 配置（*.cfg）
   - ICS 资源库（*.in 和 *.out）

   如果使用了资源库文件在 ICS 环境中配置连接器，则选择此选项。资源库文件可能包含多个连接器定义，当打开该文件所有定义都会显示。

- 所有文件（*.*）

  如果在连接器的适配器软件包中交付的是 *.txt 文件或者可用另一个扩展名提供定义文件，则选择此选项。

3. 在目录屏幕中，浏览至适当的连接器定义文件，选择它并单击**打开**。

遵循以下步骤以从“系统管理器”项目打开连接器配置:

1. 启动系统管理器。仅当已启动系统管理器时，才能从系统管理器打开配置或将配置保存到系统管理器。
2. 启动连接器配置器。
3. 单击**文件 > 打开 > 从项目**。

完成配置文件

当您从项目打开配置文件或连接器时，“连接器配置器”窗口显示配置屏幕，并显示当前属性和值。

工具栏中具有一个称为**目标系统**的下拉列表，该下拉列表允许您为属性的扩展验证选择目标操作系统。可用的选项有: Windows、UNIX、其它（如果不是 Windows 或 UNIX）和“无”（表示无扩展验证，也就是关闭扩展验证）。启动时的缺省值为 Windows。

配置屏幕的标题显示该文件中指定的集成代理程序名称和连接器名称。确保您具有正确的代理程序。如果没有正确的代理程序，则在配置连接器之前更改代理程序值。为此:

1. 在**标准属性**选项卡之下，选择 BrokerType 属性的值字段。在下拉菜单中，选择值 ICS、WMQI 或 WAS。
2. “标准属性”选项卡将显示与选择的代理程序相关联的连接器属性。表中显示了**属性名**、**值**、**类型**和**子类型**（如果“类型”是一个字符串的话）。
3. 您可以现在保存该文件或完成其余的配置字段，如第 30 页的『指定受支持的业务对象定义』中所述。
4. 在完成配置后，单击**文件 > 保存 > 至项目**或**文件 > 保存 > 至文件**。

   如果您要保存至文件，则选择 *.cfg 作为扩展名，为文件选择正确的位置并单击**保存**。

   如果打开了多个连接器配置，则单击**全部保存到文件**以将所有配置都保存到文件，或单击**全部保存到项目**以将所有连接器配置都保存到一个“系统管理器”项目。

   在保存该文件之前，连接器配置器检查是否已为所有必需的标准属性设置值。如果某个必需的标准属性缺少值，则连接器配置器显示一条验证失败的消息。您必须为该属性提供一个值才能保存配置文件。

设置配置文件属性

当创建并命名新的连接器配置文件时，或当打开现有的连接器配置文件时，连接器配置器显示一个配置屏幕以及必需配置值类别的选项卡。

连接器配置器要求以下类别的属性具有值以便连接器在所有代理程序上运行:

- 标准属性
- 特定于连接器的属性
- 受支持的业务对象
- 跟踪 / 日志文件值
- 数据处理程序（适用于将 JMS 消息传递与有保证的事件传递配合使用的连接器）

注: 对于使用 JMS 消息传递的连接器，为了配置将数据转换为业务对象的数据处理程序，可能会显示额外的类别。

对于在 **ICS** 上运行的连接器，以下属性的值也是必需的:

- 相关映射
- 资源
- 消息传递（在适用的任何地方）

重要提示: 连接器配置器接受使用英语或非英语字符集的属性值。但是标准的和特定于连接器的属性的名称以及受支持的业务对象的名称都必须只使用英语字符集。

标准属性与特定于连接器的属性不同，如下所示:

- 连接器的标准属性由特定于应用程序的连接器组件及其代理程序组件一起共享。所有连接器都具有一组相同的标准属性。每个适配器指南的附录 A 中描述了这些属性。您可以更改其中某些值，但并不能更改全部值。
- 特定于应用程序的属性仅适用于特定于应用程序的连接器组件，即，直接与应用程序交互的组件。每个连接器都具有其应用程序所独有的特定于应用程序的属性。其中有些属性提供缺省值，而有些属性则不提供；您可以修改某些缺省值。每个适配器指南的安装和配置章节都描述了特定于应用程序的属性和建议的值。

标准属性和**特定于连接器的属性**的字段是用颜色编码的，以显示哪些字段是可配置的:

- 背景为灰白色的字段指示标准属性。您可以更改值，但不能更改名称或除去该属性。
- 背景为白色的字段指示特定于应用程序的属性。这些属性根据应用程序或连接器的特定需要而不同。您可以更改值和删除这些属性。
- 值字段是可配置的。
- 将对每个属性显示**更新方法**字段。它指示激活更改的值是否需要重新启动组件或代理程序。您不能配置此设置。

设置标准连接器属性

要更改标准属性的值:

1. 单击您要设置其值的字段。

2. 输入一个值，或从下拉菜单（如果它出现的话）选择一个值。
3. 在为标准属性输入所有值之后，您可以执行以下其中一项操作:
   - 要废弃更改、保留原始值并退出连接器配置器，单击**文件 > 退出**（或关闭窗口），并在提示保存更改时单击**否**。
   - 要在连接器配置器中为其它类别输入值，选择该类别的选项卡。当您移至下一个类别时，将保留您为**标准属性**（或任何其它类别）输入的值。当您关闭窗口时，将提示您保存或废弃您在所有类别中作为整体输入的值。
   - 要保存修改的值，单击**文件 > 退出**（或关闭窗口）并在提示保存更改时单击**是**。或者，从“文件”菜单或工具栏单击**保存 > 至文件**。

要获取有关特定标准属性的更多信息，将鼠标移至“标准属性”选项卡式页面中该属性的“描述”列中的条目上。如果您安装了“扩展帮助”，则会打开一个“帮助”窗口并显示标准属性的详细信息。

有关“扩展帮助”文件的位置，参阅『标准属性』附录中的 AdapterHelpName 属性。

设置特定于连接器的配置属性

对于特定于连接器的配置属性，您可以添加或更改属性名、配置值、删除属性和对属性进行加密。缺省属性长度为 255 个字符。

1. 右键单击网格的左上部。将出现弹出菜单栏。单击**添加**以添加属性。要添加子属性，右键单击父行号并单击**添加子代**。
2. 为属性或子属性输入值。
3. 要对属性进行加密，选择**加密**框。
4. 选择保存或废弃更改，如第 28 页的『设置标准连接器属性』所述。

对每个属性显示的“更新方法”指示激活更改的值是否需要重新启动组件或代理程序。

重要提示: 更改预先设置的特定于应用程序的连接器属性名可能导致连接器发生故障。连接器可能需要某些属性名才能连接至应用程序或正确地运行。

连接器属性的加密

可以通过在“特定于连接器的属性”窗口中选择**加密**复选框来加密特定于应用程序的属性。要将值解密，单击以清除**加密**复选框，并在**验证**对话框中输入正确的值，然后单击**确定**。如果输入的值是正确的，则将解密并显示该值。

每个连接器的适配器用户指南都包含每个属性及其缺省值的列表和描述。

如果属性具有多个值，则**加密**复选框将对属性的第一个值显示。当您选择**加密**时，将对属性的所有值进行加密。要将属性的多个值解密，单击以对属性的第一个值清除**加密**复选框，然后在**验证**对话框中输入新的值。如果输入值是匹配的，则多个值将全部解密。

更新方法

参阅更新方法的描述，可以在*连接器的标准配置属性*附录的第 262 页的『配置属性值概述』中找到这些描述。

指定受支持的业务对象定义

使用“连接器配置器”中的**受支持的业务对象**选项卡来指定连接器将使用的业务对象。您必须指定通用业务对象和特定于应用程序的业务对象，并且您必须指定这些业务对象之间的映射关联。

注: 有些连接器要求将某些业务对象指定为受支持的，以便执行事件通知或对其应用程序进行额外配置（使用元对象）。有关更多信息，请参阅 *Connector Development Guide for C++* 或 *Connector Development Guide for Java*。

如果 ICS 是代理程序

要指定业务对象定义受连接器支持，或更改现有业务对象定义的支持设置，单击**受支持的业务对象**选项卡并使用以下字段。

业务对象名: 在系统管理器正在运行的情况下，要指定业务对象定义受连接器支持:

1. 在**业务对象名**列表中单击一个空字段。将显示一个下拉列表，显示出“系统管理器”项目中存在的所有业务对象定义。
2. 单击一个业务对象以添加它。
3. 为该业务对象设置**代理程序支持**（将在下面描述）。
4. 在“连接器配置器”窗口的“文件”菜单中，单击**保存至项目**。修改的连接器定义（包括为添加的业务对象定义指定的支持）将保存至系统管理器中的 ICL（集成组件库）项目。

要从受支持的列表删除业务对象:

1. 要选择业务对象字段，单击业务对象左边的数字。
2. 从“连接器配置器”窗口的**编辑**菜单，单击**删除行**。将从列表屏幕除去该业务对象。
3. 从**文件**菜单，单击**保存至项目**。

从受支持的列表删除业务对象将更改连接器定义，并使删除的业务对象不可用于此连接器的此实现。它不会影响连接器代码，它也不会从系统管理器除去业务对象定义。

代理程序支持: 如果某个业务对象具有代理程序支持，则系统将尝试使用该业务对象以通过连接器代理程序将数据传递至应用程序。

通常，连接器的特定于应用程序的业务对象受该连接器的代理程序支持，但通用业务对象则不受支持。

要指示该业务对象是否受连接器代理程序支持，选择**代理程序支持**框。“连接器配置器”窗口不验证您的代理程序支持选择。

最大事务级别: 连接器的最大事务级别是该连接器支持的最高事务级别。

对于大多数连接器，最大努力是唯一可能的选项。

您必须重新启动服务器以便事务级别的更改生效。

如果 WebSphere Message Broker 是您的代理程序

如果您以独立方式工作（未连接至系统管理器），则您必须手工输入业务对象名。

如果您已在运行系统管理器，则您可以选择**受支持的业务对象**选项卡中**业务对象名**列之下的空框。出现一个组合框，显示可从连接器所属的“集成组件库”项目获得的业务对象列表。从该列表选择您需要的业务对象。

消息集标识是 WebSphere Business Integration Message Broker 5.0 的可选字段，如果提供该标识，则它不必是唯一的。但是，对于 WebSphere MQ Integrator 和 Integrator Broker 2.1，您必须提供唯一**标识**。

如果 WAS 是代理程序

当选择 WebSphere Application Server 作为代理程序类型时，连接器配置器不需要消息集标识。**受支持的业务对象**选项卡仅对受支持的业务对象显示**业务对象名**列。

如果您以独立方式工作（未连接至系统管理器），则您必须手工输入业务对象名。

如果您已在运行系统管理器，则您可以选择“受支持的业务对象”选项卡中“业务对象名”列之下的空框。出现一个组合框，显示可从连接器所属的“集成组件库”项目获得的业务对象列表。从此列表选择您需要的业务对象。

相关映射（ICS）

每个连接器都支持业务对象定义列表及这些定义在 WebSphere InterChange Server 中当前活动的相关映射。在您选择**相关映射**选项卡时将出现此列表。

业务对象列表包含代理程序支持的特定于应用程序的业务对象和控制器发送至预订协作的相应通用对象。映射的关联确定哪个映射用来将特定于应用程序的业务对象变换为通用业务对象，或将通用业务对象变换为特定于应用程序的业务对象。

如果您正在使用为特定的源业务对象和目标业务对象唯一地定义的映射，则当您打开该屏幕时，这些映射将已经与其适当的业务对象相关，您将不需要（或无法）更改它们。

如果多个映射可供受支持的业务对象使用，则您将需要显式地将该业务对象与它应该使用的映射绑定在一起。

相关映射选项卡显示以下字段:

- **业务对象名**

  它们是受此连接器支持的业务对象，是在**受支持的业务对象**选项卡中指定的。如果您在“受支持的业务对象”选项卡下指定额外的业务对象，则在您通过从“连接器配置器”窗口的**文件**菜单选择**保存至项目**来保存更改之后，此列表中将反映这些业务对象。

- **相关映射**

  该屏幕显示已安装到系统以与连接器的受支持业务对象配合使用的所有映射。每个映射的源业务对象都显示在**业务对象名**屏幕中映射名的左边。

- **显式**

  在某些情况下，您可能需要显式地绑定相关映射。

仅当特定的受支持业务对象存在多个映射时，才需要显式绑定。当 ICS 引导时，它尝试自动将映射绑定至每个连接器的每个受支持的业务对象。如果多个映射使用同一业务对象作为其输入，则服务器尝试找到并绑定一个是其它映射超集的映射。

如果没有任何映射是其它映射的超集，则服务器无法将业务对象绑定至单个映射，并且您将需要显式地设置绑定。

要显式地绑定映射:

1. 在**显式**列，选择您要绑定的映射的复选框。
2. 选择您想要与业务对象相关的映射。
3. 在“连接器配置器”窗口的**文件**菜单中，单击**保存至项目**。
4. 将项目部署到 ICS。
5. 重新引导服务器以便更改生效。

资源（ICS）

资源选项卡允许您设置一个值，用于确定连接器代理程序是否将使用连接器代理程序来同时处理多个进程以及达到的处理程度。

并非所有连接器都支持此功能。如果您要运行在 Java 中旨在用于多线程的连接器代理程序，则建议您不要使用此功能，因为使用多线程通常比使用多进程更有效。

消息传递（ICS）

消息传递选项卡使您能够配置消息传递属性。仅当您已将 MQ 设置为 DeliveryTransport 标准属性的值并将 ICS 设置为代理程序类型时，消息传递属性才可用。这些属性影响连接器将如何使用队列。

验证消息传递队列

在可以验证消息传递队列之前，您必须执行以下操作:

- 确保安装了 WebSphere MQ Series。
- 使用主机上的通道和端口创建消息传递队列。
- 与主机建立连接。

要验证队列，使用“消息传递”选项卡上的“消息传递类型”和“主机名”字段右边的“验证”按钮。

安全性级别（ICS）

可以使用“连接器配置器”中的**安全性**选项卡来为消息设置各种隐私级别。仅当 DeliveryTransport 属性设置为 JMS 时才能使用此功能。

缺省情况下，“隐私”是关闭的。选择**隐私**框以启用它。

密钥库目标系统绝对路径名为:

- 对于 Windows:
  `<ProductDir>\connectors\security\<connectorname>.jks`
- 对于 UNIX:
  `opt/IBM/WebSphereAdapters/connectors/security/<connectorname>.jks`

此路径和文件应该与连接器配置器在同一系统上。

仅当目标系统是当前正在运行的系统时，您才能使用右边的“浏览”按钮。除非启用了**隐私**，并且菜单栏中的**目标系统**设置为 Windows，否则“浏览”按钮将变灰。

对于三种消息类别（“所有消息”、“所有管理消息”和“所有业务对象消息”），可以按如下所示设置**消息隐私级别**:

- “” 是缺省值；当尚未对消息类别设置隐私级别时使用。
- 无
  与缺省值不同：使用此选项来谨慎地为消息类别设置隐私级别“无”。
- 完整性
- 隐私
- 完整性和隐私

密钥维护功能允许您为服务器和适配器生成、导入和导出公用密钥。

- 当您选择**生成密钥**时，将出现“生成密钥”对话框，该对话框中显示了将生成密钥的密钥工具的缺省值。
- 密钥库值缺省设置为您在“安全性”选项卡上的**密钥库目标系统绝对路径名**中输入的值。
- 当您选择“确定”时，将验证条目、生成密钥证书并将输出发送至连接器配置器日志窗口。

在可以将证书导入适配器密钥库之前，必须从服务器密钥库中导出它。当您选择**导出适配器公用密钥**时，将出现“导出适配器公用密钥”对话框。

- 导出证书缺省设置为与密钥库相同的值，只不过文件扩展名为 <filename>.cer。
-

当您选择**导入适配器公用密钥**时，将出现“导入适配器公用密钥”对话框。

- 导入证书缺省设置为 *<ProductDir>*\bin\ics.cer（如果系统上存在该文件的话）。
- 导入证书关联应该是服务器名称。如果服务器已注册，则可以从下拉列表中选择它。

仅当 DeliveryTransport 的值为 IDL 时，才会启用**适配器访问控制**功能。缺省情况下，适配器使用 guest 标识登录。如果未选择**使用 guest 标识**框，则会启用**适配器标识**和**适配器密码**字段。

设置跟踪 / 日志文件值

当您打开连接器配置文件或连接器定义文件时，连接器配置器使用该文件的日志和跟踪值作为缺省值。您可以在连接器配置器中更改那些值。

要更改日志和跟踪值:

1. 单击**跟踪 / 日志文件**选项卡。
2. 对于日志记录或跟踪，您可以选择将消息写入以下其中一项或全部:
   - 至控制台（标准输出）:
     将日志或跟踪消息写至标准输出屏幕。

注: 对于在 Windows 平台上运行的连接器，您只能从**跟踪 / 日志文件**选项卡使用标准输出选项。

- 至文件:
  将日志或跟踪消息写入您指定的文件。要指定该文件，单击目录按钮（省略号），浏览至首选位置，提供文件名并单击**保存**。日志记录消息或跟踪消息将写入您指定的文件和位置。

  注: 日志记录文件和跟踪文件都是简单的文本文件。您在设置文件的文件名时可以使用您喜欢的文件扩展名。但是，对于跟踪文件，最好是使用扩展名 `.trace` 而不是 `.trc`，以避免与可能驻留在系统上的其它文件混淆。对于日志记录文件，`.log` 和 `.txt` 是典型的文件扩展名。

数据处理程序

仅当为 DeliveryTransport 指定了值 JMS 并且为 ContainerManagedEvents 指定了值 JMS 时，数据处理程序节才可用于配置。并非所有适配器都使用数据处理程序。

请参阅“附录 A，标准属性”中的 `ContainerManagedEvents` 下的描述以获取要用于这些属性的值。有关其它详细信息，请参阅 *Connector Development Guide for C++* 或 *Connector Development Guide for Java*。

保存配置文件

完成配置连接器后，保存连接器配置文件。连接器配置器以您在配置期间选择的代理程序方式保存该文件。连接器配置器的标题栏始终显示当前使用的代理程序方式（ICS、WMQI 或 WAS）。

将保存该文件作为 XML 文档。您可以三种方式来保存 XML 文档:

- 在系统管理器中，作为“集成组件库中”具有扩展名 `*.con` 的文件，或者
- 在您指定的目录中。
- 以独立方式，作为目录文件夹中具有扩展名 `*.cfg` 的文件。缺省情况下，将把该文件保存到 `\WebSphereAdapters\bin\Data\App`。
- 您还可以将它保存到 WebSphere Application Server 项目（如果您已设置一个项目）。

有关在系统管理器中使用项目的详细信息以及有关部署的进一步信息，请参阅以下实现指南:

- 对于 ICS: *Implementation Guide for WebSphere InterChange Server*
- 对于 WebSphere Message Broker: *Implementing Adapters with WebSphere Message Brokers*
- 对于 WAS: *Implementing Adapters with WebSphere Application Server*

更改配置文件

您可以更改现有配置文件的集成代理程序设置。这使您能够使用该文件作为模板来创建可以与不同代理程序配合使用的新配置文件。

注: 如果您切换集成代理程序，您将需要更改其它配置属性以及代理程序方式属性。

要在现有的配置文件中更改代理程序选择（可选的）:

- 在连接器配置器中打开现有的配置文件。
- 选择**标准属性**选项卡。
- 在“标准属性”选项卡的 **BrokerType** 字段中，选择适合于您的代理程序的值。当更改当前值时，属性屏幕上的可用选项卡和字段选择将立即更改，以仅显示与您选择的新代理程序有关的那些选项卡和字段。

完成配置

在为连接器创建配置文件并修改它之后，确保连接器在启动时可以找到该配置文件。

为此，打开用于连接器的启动文件，并验证用于连接器配置文件的位置和文件名是否与您为该文件提供的名称和您用来放置该文件的目录或路径完全匹配。

在全球化环境中使用连接器配置器

连接器配置器已全球化，并且可以处理配置文件和集成代理程序之间的字符转换。连接器配置器使用本机编码。当它写入配置文件时，它使用 UTF-8 编码。

连接器配置器在以下各项中支持非英语字符:

- 所有值字段
- 日志文件和跟踪文件路径（在**跟踪 / 日志文件**选项卡中指定）

CharacterEncoding 和 Locale 标准配置属性的下拉列表仅显示受支持值的子集。要将其它值添加至下拉列表，则您必须手工修改产品目录中的 \Data\Std\stdConnProps.xml 文件。

例如，要将语言环境 en_GB 添加至 Locale 属性的值列表，打开 stdConnProps.xml 文件并添加以下用粗体字显示的行:

```
<Property name="Locale"
isRequired="true"
updateMethod="component restart">
                <ValidType>String</ValidType>
            <ValidValues>
                                <Value>ja_JP</Value>
                                <Value>ko_KR</Value>
                                <Value>zh_CN</Value>
                                <Value>zh_TW</Value>
                                <Value>fr_FR</Value>
                                <Value>de_DE</Value>
                                <Value>it_IT</Value>
                                <Value>es_ES</Value>
                                <Value>pt_BR</Value>
                                <Value>en_US</Value>
                                <Value>en_GB</Value>

                    <DefaultValue>en_US</DefaultValue>
            </ValidValues>
    </Property>
```

第 4 章 运行连接器

本章描述 IBM WebSphere Business Integration mySAP.com 适配器（SAP R/3 V3.x）的正在运行的连接器组件。本章假定您已经安装了连接器，如第 13 页的第 2 章，『安装连接器』中所述。

本章包含以下各节:

- 『启动连接器』
- 第 38 页的『利用负载均衡』

重要提示: 如果您正在升级连接器的版本，则您必须替换连接器 Jar 文件（.jar）。您还需要升级连接器传送文件以及您先前安装的任何业务对象传送包。根据对连接器所作的更改，您可能需要将 SAPConnector.txt 文件的新副本装入您的资源库。有关更多信息，请参阅发行说明。

启动连接器

必须使用连接器的**连接器启动脚本**来显式启动连接器。在 Windows 系统上，启动脚本应该驻留在连接器的运行时目录中:

ProductDir\connectors*connName*

其中 *connName* 标识连接器。

在 UNIX 系统上，启动脚本应该驻留在 *ProductDir*/bin 目录中。

启动脚本的名称取决于操作系统平台，如表 4 中所示。

表 4. 连接器的启动脚本

操作系统	启动脚本
基于 UNIX 的系统	connector_manager
Windows	start_*connName*.bat

当启动脚本运行时，缺省情况下，它希望在 *Productdir* 中找到配置文件（请参阅下面的命令）。您将配置文件就放置在此目录中。

注: 如果适配器使用的是 JMS 传送包，则需要本地配置文件。

可以采用下列任何方式来调用连接器启动脚本:

- 在 Windows 系统上，从**开始**菜单中

  选择**程序 > IBM WebSphere Business Integration Adapters > 适配器 > 连接器**。缺省情况下，程序名为 “IBM WebSphere Business Integration Adapters”。但是，可以定制程序名。或者，可以对连接器创建桌面快捷方式。
- 从命令行
  - 在 Windows 系统上:

    start_*connName connName brokerName* [-c*configFile*]

– 在基于 UNIX 的系统上:

```
connector_manager -start connName brokerName [-cconfigFile ]
```

其中 *connName* 是连接器的名称，*brokerName* 标识集成代理程序，如下所示:

– 对于 WebSphere InterChange Server，对 *brokerName* 指定 ICS 实例的名称。

– 对于 WebSphere Message Broker（WebSphere MQ Integrator、WebSphere MQ Integrator Broker 或 WebSphere Business Integration Message Broker）或 WebSphere Application Server，对 *brokerName* 指定用来标识代理的字符串。

注: 对于 Windows 系统上的 WebSphere Message Broker 或 WebSphere Application Server，*必须*包括 `-c` 选项，再后跟连接器配置文件的名称。对于 ICS，`-c` 是可选的。

- 从 Adapter Monitor （仅当代理程序为 WebSphere Application Server 或 InterChange Server 时才可用），它是在您启动“系统管理器”时启动的

  可以使用此工具来装入、激活、取消激活、暂停、关闭或删除连接器。
- 从系统管理器（对于所有代理程序都可用）

  可以使用此工具来装入、激活、取消激活、暂停、关闭或删除连接器。
- 在 Windows 系统上，可以将连接器配置为作为 Windows 服务来启动。在此情况下，当引导 Windows 系统（对于“自动”服务）或者当您通过“Windows 服务”窗口启动服务时（对于“手工”服务）连接器就会启动。

有关如何启动连接器的更多信息（包括命令行启动选项），参阅下列其中一个文档:

- 对于 WebSphere InterChange Server，参阅*《系统管理指南》*。
- 对于 WebSphere Message Broker，参阅 *Implementing Adapters with WebSphere Message Brokers*。
- 对于 WebSphere Application Server，参阅 *Implementing Adapters with WebSphere Application Server*。

利用负载均衡

登录时的负载均衡通过以下方法增加定义的工作组的效率:

- 提高性能
- 减少系统资源的消耗
- 基于对工作组服务和负载灵敏度的需求在多个可用应用程序服务器之间分布用户

使用负载均衡功能启动连接器将启动与由 Hostname 属性指定的消息服务器的通信。然后该消息服务器查找具有最小负载的应用程序服务器。一旦确定了此应用程序服务器，消息服务器就会通过这一个应用程序服务器来路由所有将来与连接器进行的 RFC 通信。连接器被认为是一个与该消息服务器通信的对话用户。

负载均衡功能在 SAP 环境中性能最好，在该环境中，连接器处理少量的负载，而用户数目很大。对于较大的负载，考虑直接连接至您的一个较大的应用程序服务器。

有关配置连接器以获取负载均衡的信息，请参阅以下连接器属性的描述:

- 第 287 页的『ApplicationPassword』

- 第 287 页的『 ApplicationUserName 』
- 第 287 页的『 Client 』
- 第 287 页的『 Group 』
- 第 287 页的『 Hostname 』
- 第 290 页的『 SAPSystemID 』

第 5 章 使用 SAPODA 生成业务对象定义

本章描述 SAPODA（一个 Object Discovery Agent（ODA）），它生成 mySAP.com 适配器（R/3 V3.x）的业务对象定义。因为连接器使用基于 IDoc 类型的对象、BAPI、在 SAP 系统中定义的 RFC 支持功能模块和表示业务流程的 SAP 表，所以 SAPODA 使用这些对象来发现特定于其 SAP 数据源的业务对象需求。

注: 熟悉 IDoc 类型、BAPI、SAP 系统内 RFC 支持功能模块和 SAP 表，有助于了解 SAPODA 如何运行。

重要提示: 类似属性名或查询描述名的业务对象名必须仅使用在与美国英语语言环境（en_US）相关的代码集中定义的字符。有关其它信息，请参阅 *Business Object Development Guide*。

本章包含以下各节:

安装和使用

下节描述了 SAPODA 的安装和使用。

安装 SAPODA

要安装 SAPODA，使用 IBM WebSphere Business Integration Adapters 的安装程序。遵循 *Implementation Guide for WebSphere InterChange Server*、*Implementing Adapters with WebSphere Brokers*、*Implementing Adapter with WebSphere Interchange Server*、*Adapter for WebSphere MQ Integrator Broker*、*《系统安装指南 Windows 版》*或*《系统安装指南 Unix 版》*中的指示信息。当完成安装时，在系统上的产品目录中安装了以下文件:

- ODA\SAP\SAPODA.jar
- ODA\messages\SAPODAAgent.txt
- ODA\messages\SAPODAAgent_*ll_TT*.txt（特定于语言（*_ll*）和国家或地区（*TT*）的消息文件）
- ODA\SAP\start_SAPODA.bat（仅限于 Windows）
- ODA/SAP/start_SAPODA.sh（仅限于 UNIX）

注: 在本文档中，反斜杠（\）用作目录路径的约定。对于 UNIX 安装，用斜杠（/）替代反斜杠。所有产品路径名都是相对于该产品在系统上的安装目录。

在使用 SAPODA 之前

本节包含以下各节:

- 『在使用 SAPODA 为 ALE 或 ABAP 扩展模块创建定义之前』
- 『如何使用 SAPODA』

在运行 SAPODA 之前

您必须执行以下操作后才能运行 SAPODA:

- 具有 SAP 系统的有效登录标识
- 下载 SAP Java API, SAP 将该 API 称为它们的 Java 连接器（SAP JCo）。应该将此步骤作为连接器安装过程的一部分来执行，在第 15 页的『安装 SAP JCo』中描述了此步骤。

在使用 SAPODA 为 ALE 或 ABAP 扩展模块创建定义之前

您可以使用 SAPODA 来基于 IDoc（中间文件）生成 ABAP 扩展模块和 ALE 模块的业务对象定义:

- 解压缩至文件
- 在 SAP 系统中定义

重要提示: 在使用 SAPODA 来根据 SAP IDoc 定义文件生成业务对象定义之前，您必须为您想支持的每种 IDoc 类型创建 IDoc 定义文件。仅当使用解压缩的 IDoc 定义文件作为业务对象定义的模板时才需要此步骤。有关更多信息，请参阅第 129 页的『创建 IDoc 定义文件』。

如何使用 SAPODA

在安装 SAPODA 之后，您必须执行以下操作来生成业务对象:

1. 启动 ODA。
2. 启动业务对象设计器。
3. 遵循业务对象设计器中一个有 6 个步骤的过程来配置和运行 ODA。

以下各节详细地描述了这些步骤。

启动 SAPODA

您可以通过运行适当的文件来启动 SAPODA:

UNIX

start_SAPODA.sh

Windows

start_SAPODA.bat

使用业务对象设计器配置并运行 SAPODA。业务对象设计器使用代理程序的主机和端口来查找 ODA。在每个脚本或批处理文件的 AGENTNAME 变量中指定了代理程序的名称。此连接器的缺省 ODA 名称是 SAPODA。有关 ODA 和业务对象定义以及如何配置、启动和使用 ODA 的更多信息，请参阅 *IBM WebSphere Business Object Development Guide*。

使用错误消息文件和跟踪消息文件

错误消息文件和跟踪消息文件（缺省值为 SAPODAAgent.txt）位于产品目录下的 \ODA\messages\。这些文件特定于语言和国家或地区，并且使用以下命名约定:

*AgentName*Agent_ll_TT.txt

其中 _ll 是语言，_TT 是国家或地区。

例如，在中国大陆，该文件名将为:
SAPODAAgent_zh_CN.txt。

而在台湾，该文件名将为:
SAPODAAgent_zh_TW.txt。

业务对象设计器在选择消息文件时使用此信息。缺省搜索顺序是首先查找特定于语言环境的文件，该文件与业务对象设计器运行时所在的语言环境匹配。如果未找到该文件，则业务对象设计器缺省为美国英语（en_US ）版本。最后，业务对象设计器查找不带任何语言环境信息或语言信息的文件名。

如果您创建 ODA 脚本或批处理文件的多个实例并为每个提供的 ODA 提供一个唯一名称，尽管没有要求这样做，您也可以为每个 ODA 实例都提供一个消息文件。或者，您可以将使用同一消息文件的 ODA 命名为不同名称。有两种方式来指定有效的消息文件:

- 如果您更改 ODA 的名称而不为它创建消息文件，则您必须在业务对象设计器中更改消息文件的名称作为 ODA 配置的一部分。业务对象设计器为消息文件提供了一个名称，但并未实际创建该文件。如果显示为 ODA 配置的一部分的文件不存在，则更改该值以指向现有的文件。
- 您可以复制特定 ODA 的现有消息文件并按需要修改它。业务对象设计器假定您根据命名约定命名每个文件。例如，如果 AGENTNAME 变量指定 SAPODA1，则该工具假定相关消息文件的名称是 SAPODA1Agent.txt。因此，当业务对象设计器提供文件名进行验证作为 ODA 配置的一部分时，该文件名将基于 ODA 名称。验证是否正确命名了缺省消息文件，并按需要改正它。

如果您使用存在于 ODA 根目录中的部署描述符 odk_dd.xml 文件，则 MessageFile 属性不会显示在业务对象设计器的“配置代理程序属性”窗口中。

注: 如果需要非英语语言环境，则相同的命名约定仍然适用；例如，SAPODA1Agent_zh_TW.txt。

重要提示: 当您配置 ODA 时，未能正确指定消息文件的名称会导致它在不带消息的情况下运行。有关指定消息文件名的更多信息，请参阅第 45 页的『配置初始化属性』

在配置过程期间，您应指定:

- SAPODA 将错误和跟踪信息写入的文件的名称
- 消息文件的名称
- 跟踪级别，其范围在 0 至 5 之间。

表 5 描述跟踪级别值。

表 5. 跟踪级别

跟踪级别	描述
0	将所有错误写入日志
1	跟踪方法的所有进入和退出消息
2	跟踪 ODA 的属性及其值
3	跟踪所有业务对象的名称
4	跟踪所有衍生的线程的详细信息
5	• 指示 ODA 的所有属性的初始值 • 跟踪 SAPODA 衍生的每个线程的详细状态 • 跟踪业务对象定义转储

有关在何处配置这些值的信息，请参阅第 45 页的『配置初始化属性』。

在业务对象设计器中使用 SAPODA

本节描述如何在业务对象设计器中使用 SAPODA 来生成业务对象定义。有关启动业务对象设计器的信息，请参阅 *IBM WebSphere InterChange Server Business Object Development Guide*。

在启动 ODA 之后，您必须启动业务对象设计器来配置和运行它。为了使用 ODA 来生成业务对象定义，在业务对象设计器中提供了 6 个步骤。业务对象设计器提供了一个向导来指导完成其中每个步骤。

在启动 ODA 之后，执行以下操作以启动该向导:

1. 打开业务对象设计器。
2. 从“文件”菜单选择“使用 ODA 新建...”子菜单。

   业务对象设计器显示向导中的第一个窗口，它被命名为“选择代理程序”。第 45 页的图 5 举例说明了此窗口。

要选择、配置和运行 ODA，遵循以下步骤:

1. 第 44 页的『选择 ODA』.
2. 第 45 页的『配置初始化属性』.
3. 第 48 页的『展开节点和选择对象』.
4. 第 51 页的『确认对象选择』.
5. 第 51 页的『生成定义』和（可选）第 52 页的『提供其它信息』。
6. 第 69 页的『保存定义』.

选择 ODA

图 5 举例说明了业务对象设计器的 6 步骤向导中的第一个对话框。从此窗口选择要运行的 ODA。

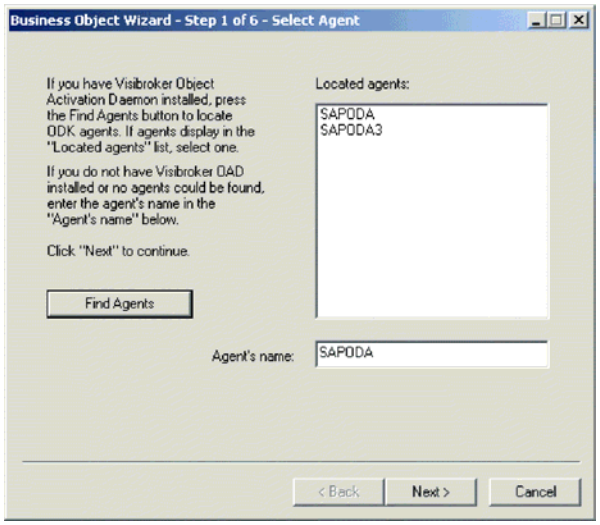


图 5. 选择 ODA

要选择 ODA:

1. 单击“查找代理程序”按钮以在“找到的代理程序”字段中显示所有注册的或当前正在运行的 ODA。

   注: 如果业务对象设计器未找到您期望的 ODA，则将主机和端口输入其各自的字段。

2. 从显示的列表中选择期望的 ODA。

配置初始化属性

业务对象设计器第一次与 SAPODA 通信时，它提示您输入一组初始化属性。您可以将这些属性保存在命名的概要文件中，以便您不必在每次使用 SAPODA 时都要重新输入它们。有关指定 ODA 概要文件的信息，请参阅 *IBM WebSphere Business Object Development Guide*。

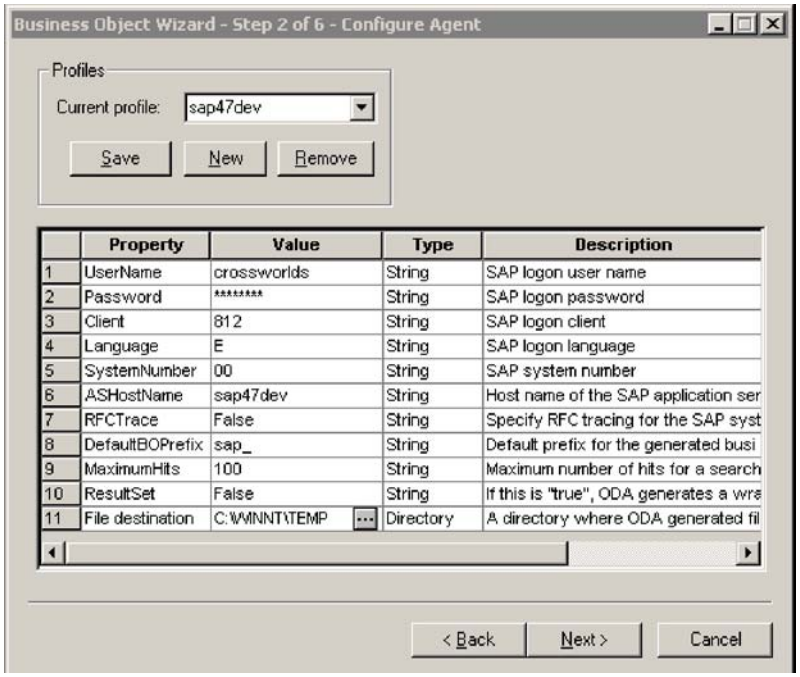


图 6. 配置代理程序属性

配置表 6 中描述的 SAPODA 属性。

表 6. SAPODA 属性

属性名	属性类型	描述
UserName	字符串	SAP 登录用户名（仅当从解压缩的 IDoc 定义文件生成定义时才不需要）
Password	字符串	SAP 登录密码（仅当从解压缩的 IDoc 定义文件生成定义时才不需要）
Client	字符串	SAP 登录客户机号（仅当从解压缩的 IDoc 定义文件生成定义时才不需要）
Language	字符串	SAP 登录语言（仅当从解压缩的 IDoc 定义文件生成定义时才不需要）。缺省值为 E，表示英语。
SystemNumber	字符串	SAP 系统号（仅当从解压缩的 IDoc 定义文件生成定义时才不需要）
ASHostName	字符串	SAP 应用程序服务器的主机名（仅当从解压缩的 IDoc 定义文件生成定义时才不需要）
RFCTrace	True/False 布尔值	SAP 系统的 RFC 跟踪
DefaultBOPrefix	字符串	附加到业务对象的名称之前以使该名称唯一的文本。 如果需要，您以后可以在业务对象设计器提示您输入特定于业务对象的属性时更改此属性。
MaximumHits	字符串	节点搜索期间返回的最大对象数。有关更多信息，请参阅第 48 页的『展开节点和选择对象』。 缺省值为: 100

表 6. SAPODA 属性 (续)

属性名	属性类型	描述
TraceFileName	字符串	跟踪文件的名称。如果该文件不存在，则 SAPODA 在 \ODA\SAP 目录中创建它。如果该文件已存在，则 SAPODA 追加至它。 SAPODA 根据命名约定命名该文件。例如，如果代理程序命名为 SAPODA，则它生成名为 SAPODAtrace.txt 的跟踪文件。 使用此属性来为此文件指定一个不同的名称。 **注意**：如果您使用存在于 ODA 根目录中的部署描述符 odk_dd.xml 文件，则“配置代理程序”屏幕不会显示此属性。
TraceLevel	整数	为 SAPODA 启用的跟踪级别。 关于跟踪的更多信息，请参阅第 43 页的『使用错误消息文件和跟踪消息文件』。
MessageFile	字符串	**注意**：如果您使用存在于 ODA 根目录中的部署描述符 odk_dd.xml 文件，则“配置代理程序”屏幕不会显示此属性。 错误和消息文件的名称。 SAPODA 根据命名约定命名该文件。例如，如果代理程序命名为 SAPODA，则它将消息文件命名为 SAPODAAgent.txt。有关更多信息，请参阅第 43 页的『使用错误消息文件和跟踪消息文件』。 **重要提示**：错误和消息文件必须位于 \ODA\messages 目录中。 使用此属性来验证或指定现有文件。
ResultSet	True/False 布尔值	**注意**：如果您使用存在于 ODA 根目录中的部署描述符 odk_dd.xml 文件，则“配置代理程序”屏幕不会显示此属性。 当设置为 True 时，SAPODA 将为 Information Integrator 支持生成包装程序业务对象。 如果您不想生成 ResultSet 对象，则设置为 False。
File destination	目录	用来存储 ODA 生成的文件（业务对象和类文件）的目录。注意，在生成了业务对象并将它放置在此目录中之后，可以显式地将它们保存到另一个目录中。 缺省值是 Windows 系统上的缺省目录。建议您将缺省设置更改为 \connectors\SAP\utilities\generatedfiles 目录。 **注意**：如果您正在业务对象设计器所在机器上运行 SAPODA，则不要使用 ODA\SAP 目录作为 File destination。业务对象设计器将此目录用作远程 ODA 的临时位置。

重要提示：如果业务对象设计器中显示的缺省值表示一个不存在的文件，则改正消息文件的名称。如果您从此对话框前进时此名称不正确，则业务对象设计器在启动 ODA 的窗口中显示一条错误消息。此消息不会在业务对象设计器中弹出。未能指定有效的消息文件会导致 ODA 在不带消息的情况下运行。有关更多信息，请参阅第 43 页的『使用错误消息文件和跟踪消息文件』。

展开节点和选择对象

在您配置 SAPODA 的所有属性之后，业务对象设计器显示一个具有以下初始节点的树:

- IDoc 类型 – 您可以:
  - 浏览解压缩的 IDoc 定义文件
  - 选择 SAP 系统中的 IDoc（基本 IDoc 类型和扩展类型）

    注: 扩展类型是客户定义的 IDoc 类型。
- BOR – 选择表示 SAP 应用程序中 BAPI 的对象
- RFC – 选择表示 SAP 应用程序中 RFC 支持功能的对象
- 动态事务和检索 – 选择表示动态事务和动态检索元数据表中对象的定义
  - HDR – 选择表示 SAP 事务的实体所需要的表，这些 SAP 事务由分层动态检索模块处理

名称前面有加号（+）的节点是可展开的。单击它们以显示更多节点或叶节点。SAPODA 仅从叶节点生成业务对象定义。

第 49 页的图 7 举例说明了此对话框初始显示时的外观并展开了一些节点。

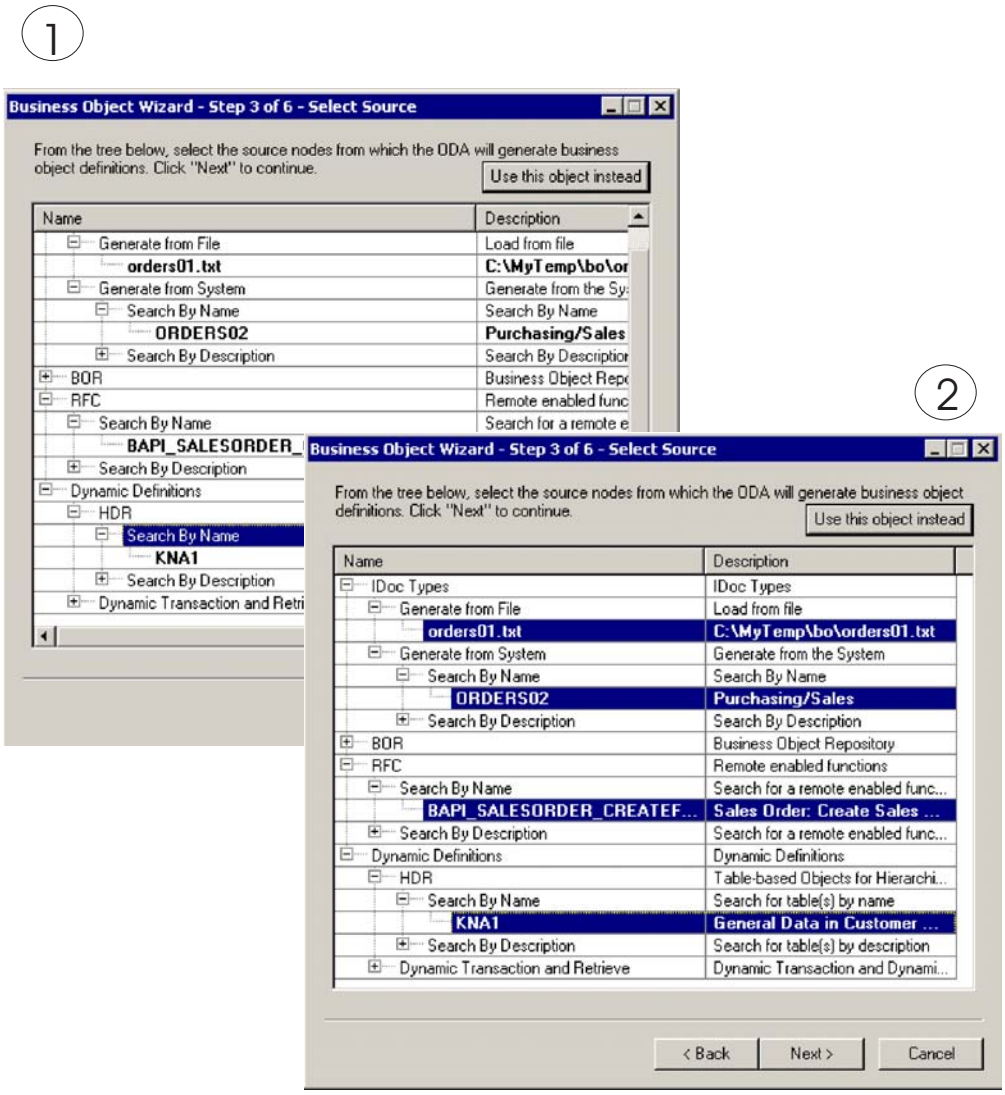


图 7. 具有展开节点的树

当叶节点的名称以粗体显示时，您可以选择该叶节点作为要生成的其业务对象的基础。使用标准 Windows 过程来选择多个叶节点。换句话说，当使用鼠标来选择多个叶节点时按住 CTRL 键。

重要提示: 在 Windows 系统上，如果业务对象设计器在 Path 环境变量中找不到必需的库文件，或这些文件不在系统上，则它显示一个空的“代理程序通知”窗口。有关这些文件的信息，请参阅第 42 页的『在运行 SAPODA 之前』。

SAPODA 使用多态节点类型，该类型允许您使平面文件与节点产生关联。该节点最初显示时没有任何叶节点。您可以浏览文件系统并选择要添加至该节点的文件。该节点之所以称为**多态的**，是因为当您使它与一个或多个文件产生关联时其性质从叶节点更改为枝节点。

注意，如果展开 RFC 节点，则会出现以下消息，指出对 RFC 节点中的搜索结果进行了高速缓存。这种高速缓存服务提供了少量叶节点数，从而 SAPODA 能够更高效地生成 ResultSet 和 BAPI 事务业务对象。会对搜索结果进行排序然后再显示出来。每当您启动 SAPODA 时这种高速缓存服务就会在后台运行，而当您结束会话时，就会清除已

高速缓存的搜索。可高速缓存的搜索结果数由在“配置代理程序属性”窗口中设置的 MaximumHits 属性值来确定，如第 46 页的图 6 中所示。

图 8. 高速缓存通知

图 9 举例说明了限制业务对象设计器返回的叶节点数的两种方式:

- 一个上下文相关菜单，它允许您打开窗口以浏览文件。从此窗口，您可以选择要产生关联的文件。
- 一个向导，它允许您在对象的名称或描述中指定搜索字符。

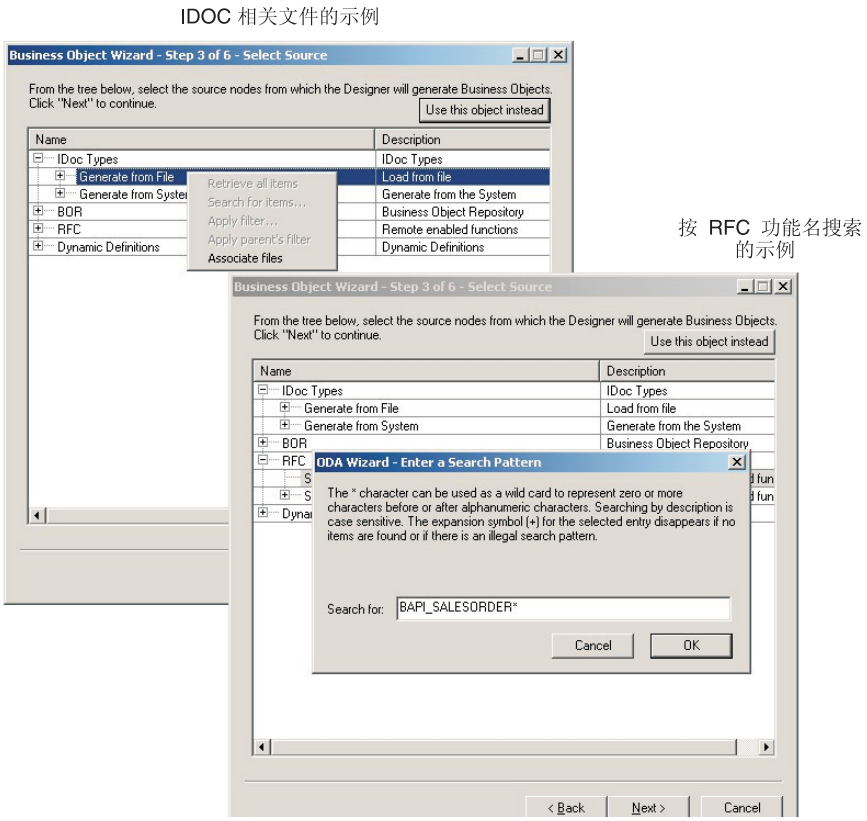


图 9. 使文件产生关联和输入搜索条件

在您为生成对象选择了所有期望的叶节点之后，单击“下一步”按钮。有关如何过滤返回的对象的信息，请参阅 *Business Object Development Guide*。

确认对象选择

在您标识所有要与生成的业务对象定义产生关联的对象之后，业务对象设计器显示该对话框，该对话框仅具有选择的叶节点及其节点路径。图 10 举例说明了此对话框。

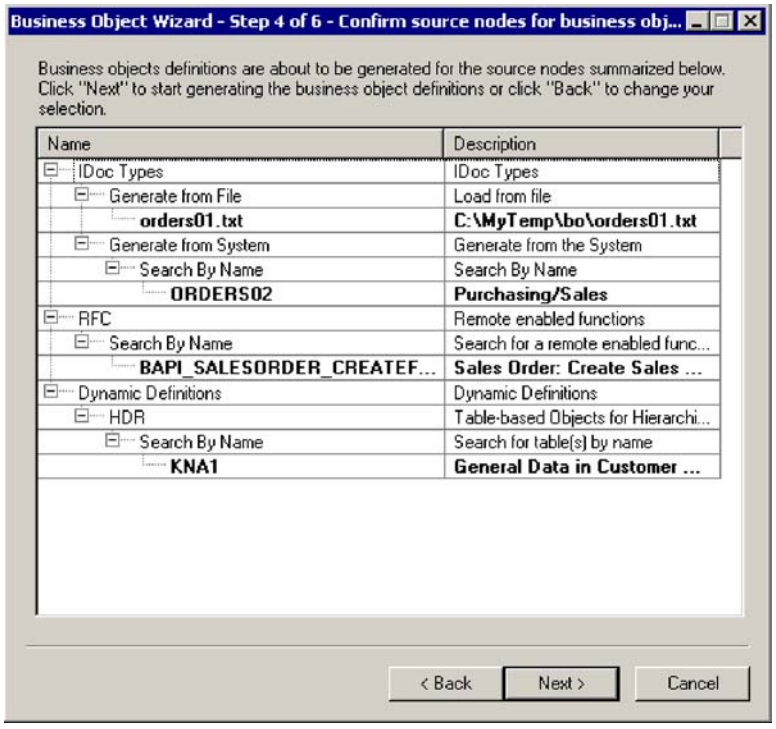


图 10. 确认节点和叶节点的选择

此窗口提供以下选项:

- 要确认选择的内容，单击“下一步”。
- 如果选择的内容不正确，则单击“上一步”以返回至前一窗口并进行必要的更改。当选择的内容正确时，单击“下一步”。

生成定义

在您确认选择的对象之后，下一个对话框通知您业务对象设计器正在生成定义。

第 52 页的图 11 举例说明了此对话框。

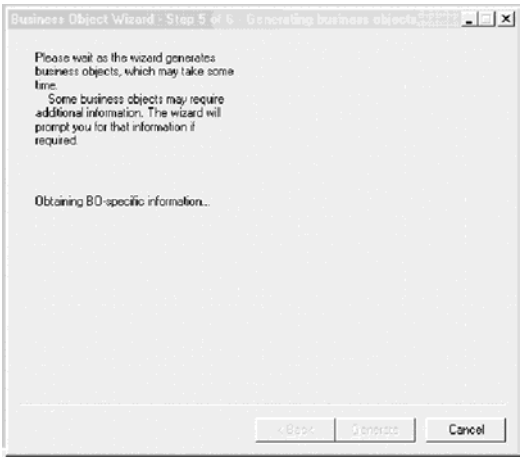


图 11. 生成定义

提供其它信息

SAPODA 提示输入其它信息。顶级节点的类型（IDoc 类型、BOR、RFC 或动态定义）确定:

- 业务对象设计器在 “BO 属性” 窗口中显示的属性集合。
- 业务对象设计器是否显示提示您输入其它对象生成信息的第二个窗口。

IDoc 类型: 提供其它信息

SAPODA 显示 “BO 属性” 窗口，以使您能够基于 IDoc 类型指定业务对象所需要的信息。此窗口中显示的属性会有所不同，取决于 IDoc 的源（一个解压缩的文件或 SAP 系统中的一个定义）以及是否已为 ABAP 扩展模块定义该定义。本节描述了以下主题:

- 『BO 属性窗口 – 公共属性』
- 第 53 页的『 “BO 属性” 窗口 – 在 SAP 系统中定义的 IDoc 的属性』
- 第 53 页的『业务对象属性窗口 – 为 ABAP 处理程序指定功能模块』

BO 属性窗口 – 公共属性: 无论 SAPODA 是根据 IDoc 文件还是根据在 SAP 系统中定义的 IDoc 生成业务对象定义，IDoc 类型的 “BO 属性” 窗口允许您指定或更改以下内容:

- 前缀信息

  前缀是附加到业务对象的名称之前以使该名称唯一的文本。如果您对在 “配置代理程序” 窗口中为 DefaultBOPrefix 属性输入的值满意，则您不必在此处更改该值。

- 模块类型

  模块类型选项是 ALE 或 “扩展”。因为 ALE 和 ABAP 扩展模块对其业务对象定义有不同的要求，所以指定哪个模块将使用业务对象很重要。

  注: 如果在顶级 IDoc 中有多个段，则当 SAPODA 为 ABAP 扩展模块生成业务对象定义时，它使用第一个 IDoc 段来表示顶级业务对象。SAPODA 将其它顶级段表示为子业务对象。

- UseFieldName

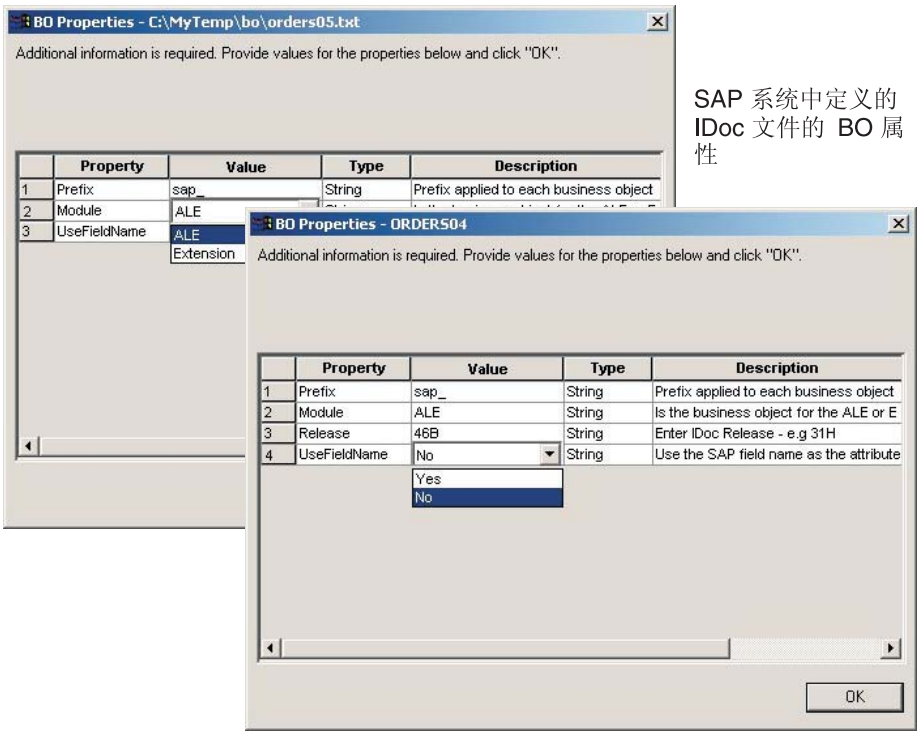
根据 SAP 字段名或 SAP 字段描述生成属性名，缺省情况下是根据 SAP 字段描述。

"BO 属性"窗口 - 在 SAP 系统中定义的 IDoc 的属性: 除前缀和模块属性之外，表示在 SAP 系统中定义的 IDoc 的 "BO 属性" 窗口还显示 Release 属性。您可以使用此属性来标识较早版本的 IDoc 类型。

重要提示: 如果较早版本的 IDoc 类型具有的段数比当前版本少，则 SAPODA 可能创建一个缺少段的定义，SAPODA 也可能显示一个错误，指示生成业务对象定义未成功。这种不一致是由于不同的 SAP 版本需要不同的 API 调用造成的。

图 12 举例说明了两个版本的 "BO 属性" 窗口，一个用于解压缩的 "IDoc 类型" 定义文件，另一个用于在 SAP 系统中定义的 IDoc。

Idoc 文件的 BO 属性



SAP 系统中定义的 IDoc 文件的 BO 属性

图 12. 提供 IDoc 类型业务对象的其它信息

业务对象属性窗口 - 为 ABAP 处理程序指定功能模块: 如果您选择扩展作为模块类型，则 SAPODA 提示您是否要为任何缺省查询描述输入功能模块名称。

缺省情况下，当生成 ABAP 扩展模块的定义时，SAPODA 指定以下文本表示顶级业务对象在业务对象级别的查询描述特定于应用程序的信息:

```
:Y_XR_IDOC_HANDLER
```

如果您已知道要传递至 ABAP 处理程序的功能模块名称，则在此提示处选择 `Yes`。SAPODA 显示第 54 页的图 13 中举例说明的窗口。

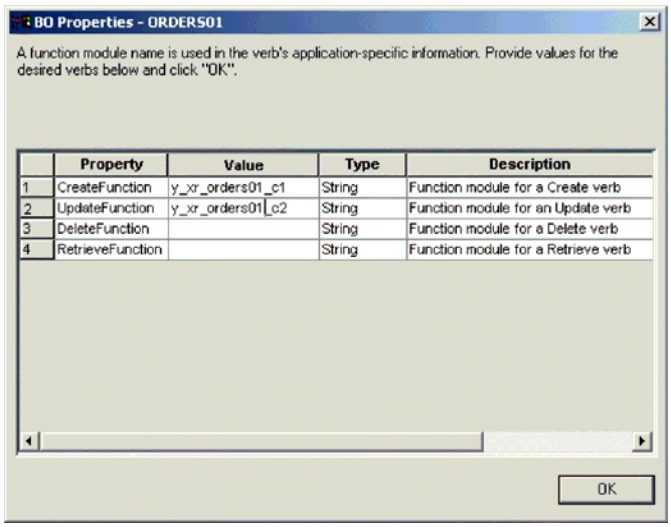


图 13. 为 ABAP 处理程序指定功能模块

图 13 举例说明了一个“BO 属性”窗口，在该窗口中已指定两个功能模块。

注: 如果定义文件中有许多 IDoc 类型，则将为文件中的每种 IDoc 类型都提供“功能模块 BO 属性”窗口。将仅提供“常规 IDoc 类型 BO 属性”窗口一次。

在您保存业务对象定义之后，业务对象设计器中的“常规”选项卡显示顶级业务对象在业务对象级别的特定于应用程序的必需信息。图 14 举例说明了这样一个具有两个指定的功能模块的窗口。

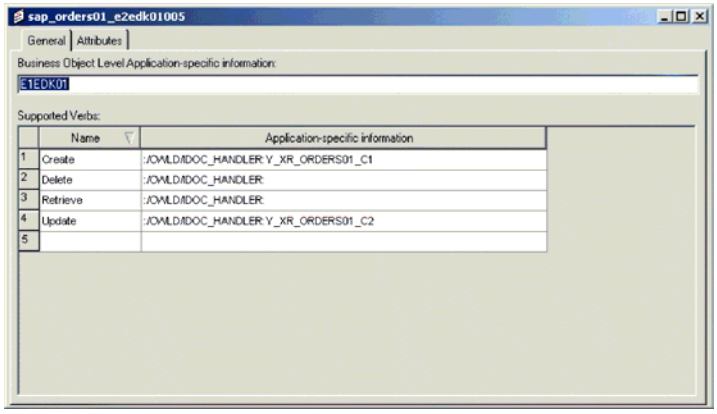


图 14. 在业务对象设计器中指定 ABAP 处理程序

有关 ABAP 处理程序的更多信息，请参阅第 199 页的『传递至 ABAP 处理程序的业务对象数据』。

BOR 或 RFC: 提供其它信息

SAPODA 将创建下列类型的对象:

- 单个 BAPI 业务对象
- BAPI 事务顶级业务对象
- ResultSet 业务对象

单个 BAPI 业务对象: 当“配置代理程序”窗口上的 ResultSet 属性设置为 False 时（如第 46 页的图 6 所示），可以使用 SAPODA 来创建单个 BAPI 事务业务对象或创建包含多个 BAPI 调用的 BAPI 事务业务对象。本节提供了有关单个 BAPI 调用的业务对象的详细信息。

有关为 BAPI 事务创建业务对象的详细信息，请参阅第 56 页的『BAPI 事务业务对象』。有关为 ResultSet 创建业务对象的详细信息，请参阅第 60 页的『ResultSet 业务对象』。

有两个“BO 属性”窗口用于 BOR 或 RFC 类型的单个 BAPI 对象。第一个窗口中显示的属性允许您指定或更改:

- 前缀 – 如果您对在“配置代理程序”窗口（第 46 页的图 6）中为 DefaultBOPrefix 属性输入的值满意，则您不必在此处更改该值。
- 查询描述 – 指定查询描述。
- 服务器支持 – 如果要为连接器的 RFC 服务器模块生成定义，则指定 yes。如果要为连接器的 BAPI 模块生成定义，则指定 no。
- UseFieldName – 根据 SAP 字段名或 SAP 字段描述生成属性名，缺省情况下是根据 SAP 字段描述。

在您单击“确定”以从第一个“BO 属性”窗口向前移动之后，SAPODA 会给您一个机会来减少生成的定义的大小。将提示您是否要从该定义中除去任何表示可选参数的属性。仅当存在要除去的可选参数时才显示此提示。减少定义的大小可以在以后连接器处理业务对象的实例时增强性能。

图 15 举例说明了为 BOR 或 RFC 类型对象显示的属性以及在您单击“确定”之后显示的提示。注意，对于您选择要创建单个 BAPI 调用对象的各个 BAPI 调用都会出现此提示。

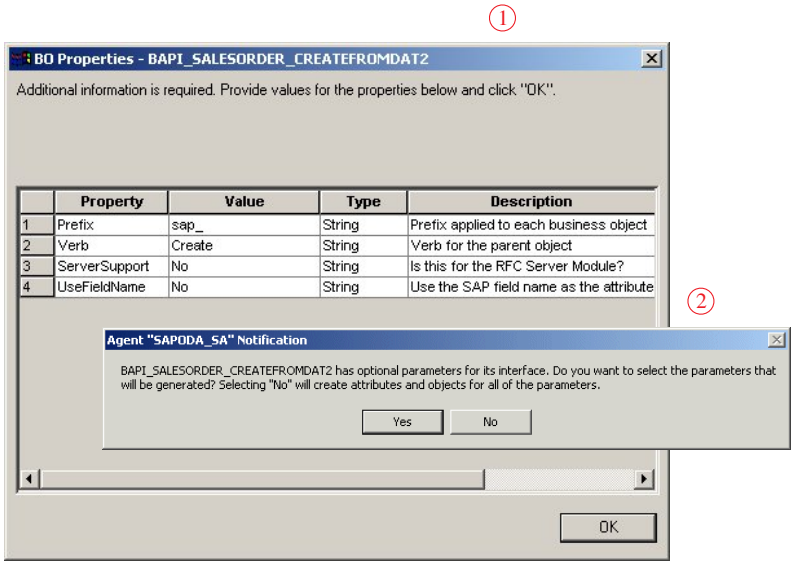


图 15. 提供 BOR 或 RFC 业务对象的其它信息

如果在以上举例说明的提示处单击**是**，则会显示第二个“BO 属性”窗口。可以通过将 BAPI/RFC 接口的每个可选参数的值从 Yes（在生成的定义中包括相应的属性）更改为 No（不包括属性），来指定除去每个可选参数。

如果在以上举例说明的提示处单击**否**，则会显示最终的向导。有关更多信息，请参阅第 69 页的『保存定义』。

图 16 举例说明了第二个“BO 属性”窗口。

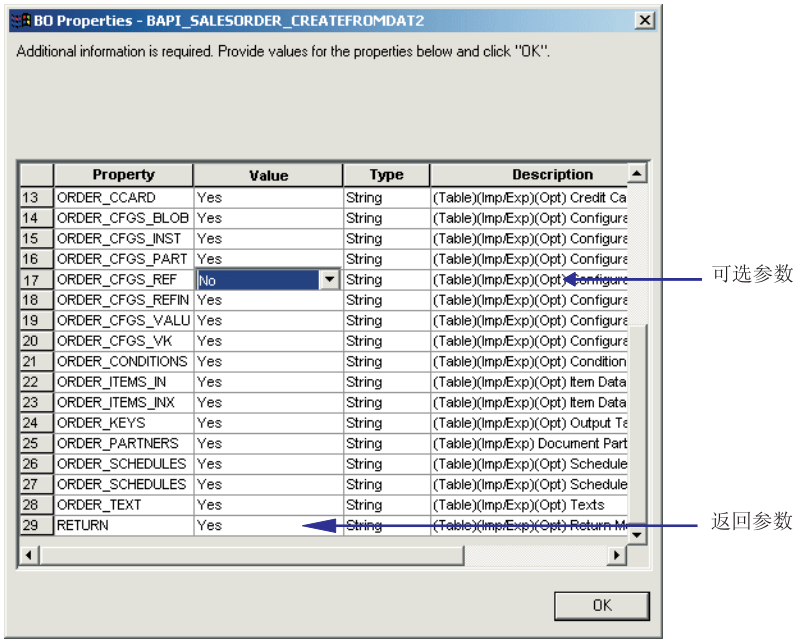


图 16. 指定要从定义中除去的属性

重要提示: 以“Bapi”开头的 RFC 支持功能的业务对象定义必须具有一个属性来表示对应于返回结构或表的业务对象。如果定义缺少这样的属性，则当编译其相应的生成代码时会出错。如果发生此编译错误，则检查 BAPI 以确定 SAP 是否在使用不同的返回结构。在这种情况下，更改生成的 Java 代码以指向正确的参数。

除了您在 SAPODA 中提供的说明以外，当为 RFC 服务器模块创建定义时，您可能还要在保存业务对象定义之后修改特定于应用程序的信息。有关更多信息，请参阅第 149 页的第 14 章，『为 RFC 服务器模块开发业务对象』。

BAPI 事务业务对象: 当“配置代理程序”窗口上的 ResultSet 属性设置为 False 时（如第 46 页的图 6 所示），可以使用 SAPODA 来创建 BAPI 事务业务对象。BAPI 事务业务对象包含多个 BAPI 业务对象。

当“配置代理程序”窗口上的 ResultSet 属性设置为 False 时，单击**下一步**以进入“高速缓存通知”窗口（第 57 页的图 17），然后单击**确定**。

图 17. 高速缓存通知

图 18 举例说明了出现的下一个窗口，该窗口允许您指定 SAPODA 将用来搜索和显示 BAPI 调用的条件。在本节使用的示例中，条件是以文本 “BAPI_SALESORDER” 开头的所有 BAPI 调用。

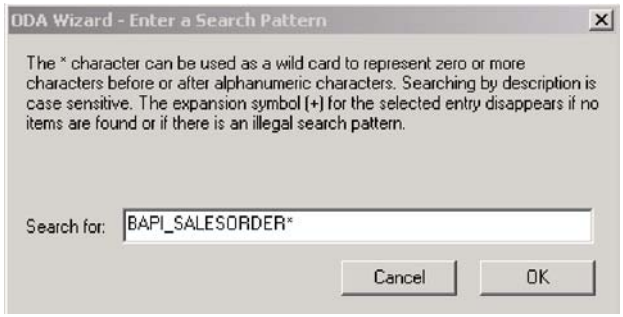


图 18. 输入搜索模式

在 “输入搜索模式” 窗口上指定搜索条件之后，单击**确定**以设置条件。图 19 举例说明了已展开 RFC 节点的搜索结果树。在此窗口上，选择 SAPODA 将用来创建 BAPI 事务业务对象的属性的 BAPI 调用。在此示例中，选择的 BAPI 调用是 BAPI_SALESORDER_CHANGE 和 BAPI_SALESORDER_CONFIRMDELVRY。

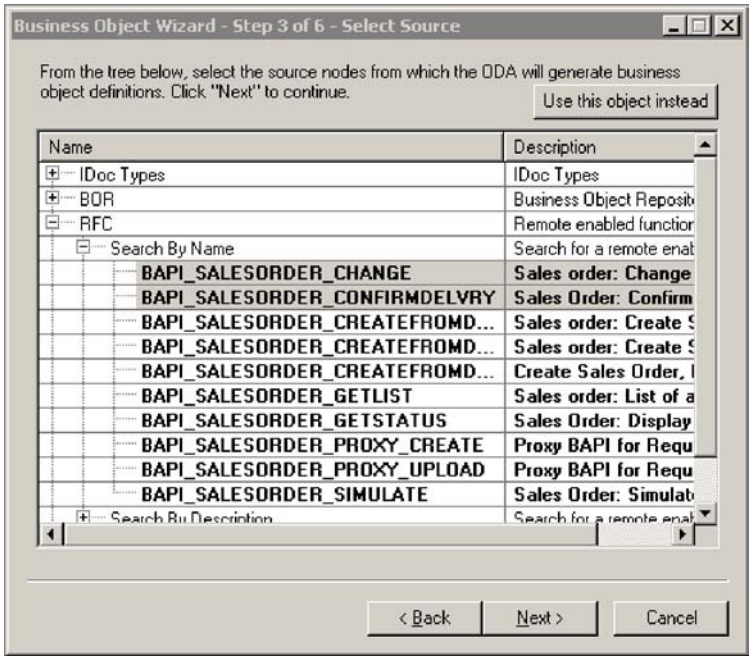


图 19. 为 RFC 节点展开的搜索结果树

单击**下一步**以进入确认窗口，如图 20 中所示。它列示了在第 57 页的图 19 中选择的两个 BAPI 调用。

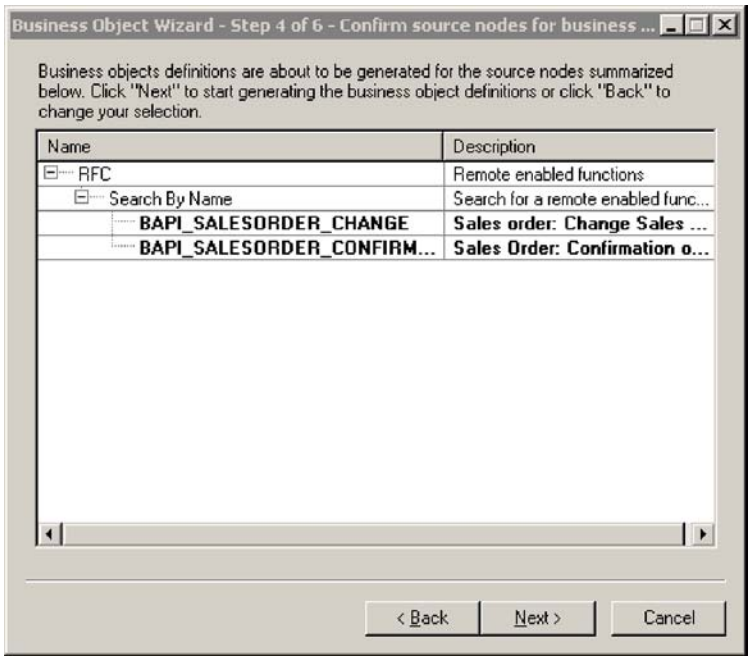


图 20. 确认事务中的 BAPI 调用的源节点

单击此屏幕上的**下一步**。出现一个消息窗口，通知您已经选择了多个 BAPI 调用。单击**是**以指示您的目的是从选择的多个 BAPI 调用创建 BAPI 事务业务对象。

图 21. 多个 BAPI 调用选择消息

下一个屏幕允许您为 BAPI 事务对象提供前缀和名称。在此示例中，输入的前缀为 `sap_`，事务对象的名称为 `salesorder_txn`。`UseFieldName` 属性确定将使用 SAP 中的字段名还是使用字段描述来生成属性名。

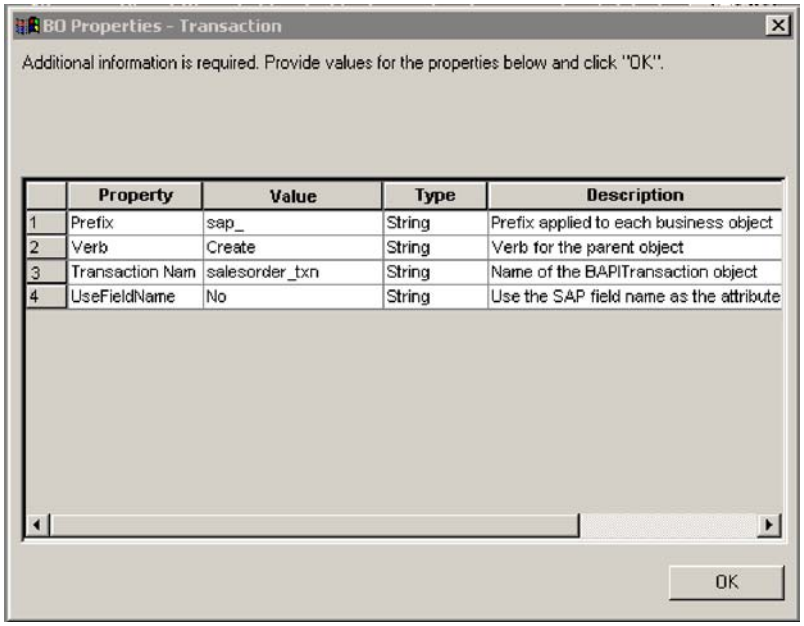


图 22. 为事务指定前缀和业务对象名

接下来，指示在处理事务对象时应采用何种顺序来执行所选择的 BAPI 调用。在此示例中，将首先执行 BAPI_SALESORDER_CHANGE 调用，接着执行 BAPI_SALESORDER_CONFIRMDELVRY 调用。在执行了事务中您想提交的任何 BAPI 之后，就可以应用 COMMIT 了。SAPODA 假定 COMMIT 是该事务的最终步骤。

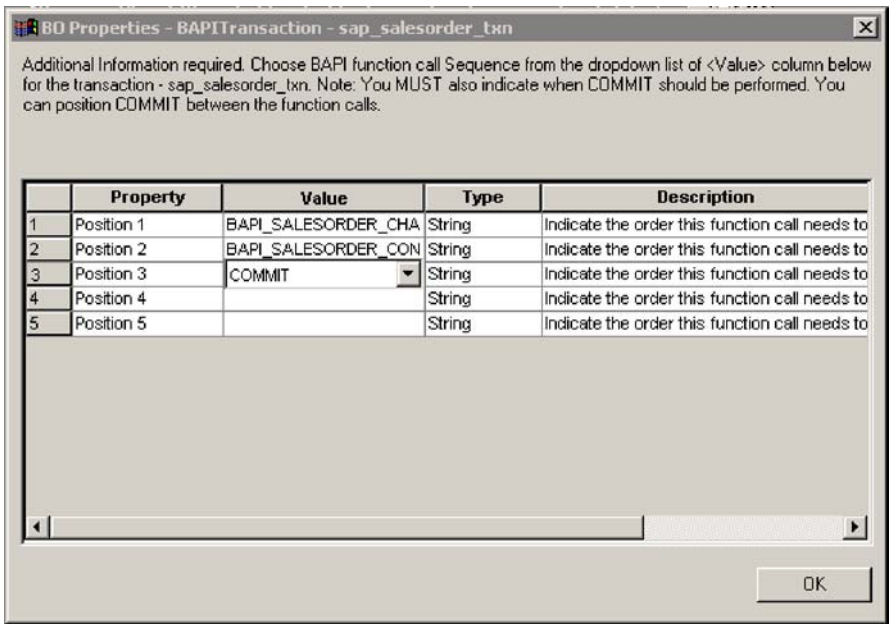


图 23. 指定 BAPI 事务对象中的 BAPI 调用顺序

对于具有可选参数的序列中的每个 BAPI 调用都会出现以下消息。第 60 页的图 24 举例说明了序列（BAPI_SALESORDER_CHANGE）中的第一个 BAPI 调用的此消息。

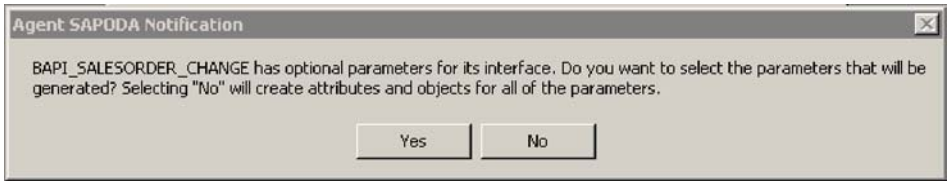
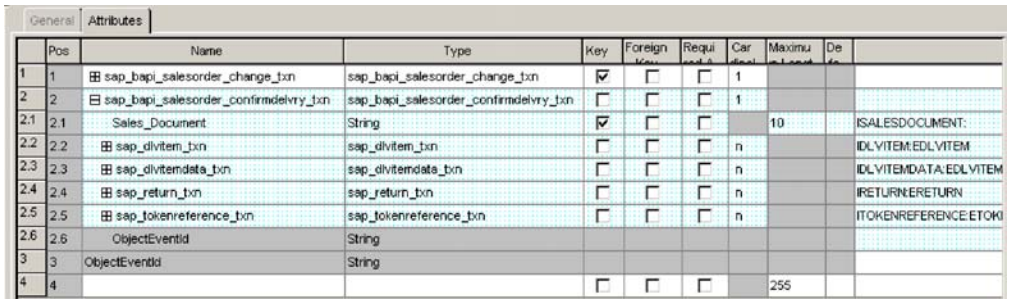


图 24. 可选的 BAPI 调用参数消息

单击**是**以选择您想添加哪些可选参数来作为将包含在 BAPI 事务对象中的个体 BAPI 业务对象的属性。如果单击**否**，则会将所有可选参数作为 BAPI 事务对象中的个体 BAPI 对象的属性来应用。

为 BAPI 事务对象中的每个 BAPI 调用创建 BAPI 对象属性之后，“业务对象”向导将显示 BAPI 事务对象树。图 25 举例说明了在此示例中创建的 sap_salesorder_txn 业务对象的**属性**选项卡。


General | Attributes

	Pos	Name	Type	Key	Foreign	Requi	Car	Maximu	De	
1	1	⊞ sap_bapi_salesorder_change_txn	sap_bapi_salesorder_change_txn	☑	☐	☐	1			
2	2	⊟ sap_bapi_salesorder_confirmdelvry_txn	sap_bapi_salesorder_confirmdelvry_txn	☐	☐	☐	1			
2.1	2.1	Sales_Document	String	☑	☐	☐		10		ISALESDOCUMENT:
2.2	2.2	⊞ sap_dlvitem_txn	sap_dlvitem_txn	☐	☐	☐	n			IDLVITEM:EDLVITEM
2.3	2.3	⊞ sap_dlvitemdata_txn	sap_dlvitemdata_txn	☐	☐	☐	n			IDLVITEMDATA:EDLVITEM
2.4	2.4	⊞ sap_return_txn	sap_return_txn	☐	☐	☐	n			IRETURN:ERETURN
2.5	2.5	⊞ sap_tokenreference_txn	sap_tokenreference_txn	☐	☐	☐	n			ITOKENREFERENCE:ETOKI
2.6	2.6	ObjectEventId	String							
3	3	ObjectEventId	String							
4	4			☐	☐	☐		255		

图 25. sap_salesorder_txn 业务对象的“属性”选项卡

注意下面这两个 BAPI 调用属性：sap_salesorder_change_txn 和 sap_salesorder_confirmdelvry_txn。每个属性都包含 BAPI 事务对象包装程序中的单个 BAPI 调用对象。sap_salesorder_change_txn 属性包含与 BAPI_SALESORDER_CHANGE 调用（在事务流中将首先执行该调用，如第 59 页的图 23 中指定的那样）相对应的业务对象。sap_salesorder_confirmdelvry_txn 属性包含与 BAPI_SALESORDER_CONFIRMDELVRY BAPI 调用相对应的业务对象。注意，这些属性都具有由 SAPODA 添加的后缀 _txn。此后缀确保在先前版本的连接器中创建的业务对象不会被可能同名的新业务对象所覆盖。

ResultSet 业务对象: 当“配置代理程序”窗口上的 ResultSet 属性设置为 True 时（如第 61 页的图 26 中所示），SAPODA 将创建顶级 ResultSet 业务对象。ResultSet 业务对象启用对 DB2 的 Information Integrator 支持。

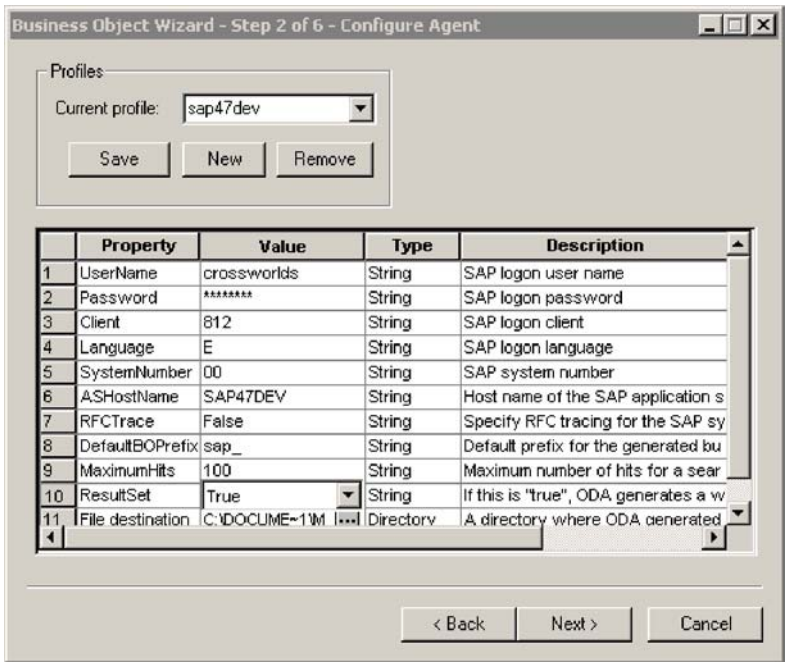


图 26. 将 ResultSet 属性设置为 True 的“配置代理程序”窗口

单击“配置代理程序”窗口上的**下一步**，然后单击“高速缓存通知”窗口（图 27）上的**确定**。

图 27. 高速缓存通知

该向导中的下一个窗口允许您指定 SAPODA 将用来搜索和显示 BAPI 调用的条件。在此示例中，星号（它是一个通配符）指示条件是以文本“BAPI_CUSTOMER_GET”开头的所有 BAPI 调用。

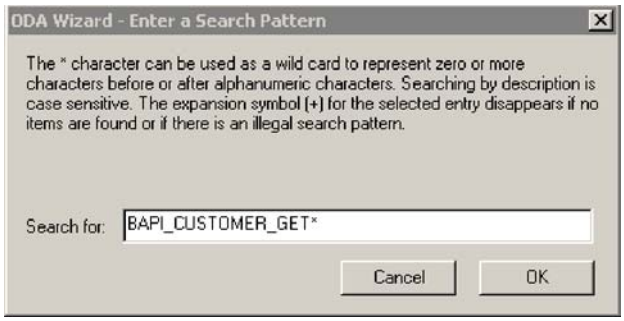


图 28. ResultSet 的搜索条件

单击**确定**以设置条件。图 29 举例说明了已展开 RFC 节点的搜索结果树。在此窗口上，选择 SAPODA 将用来创建 ResultSet 业务对象的属性的 BAPI 调用。

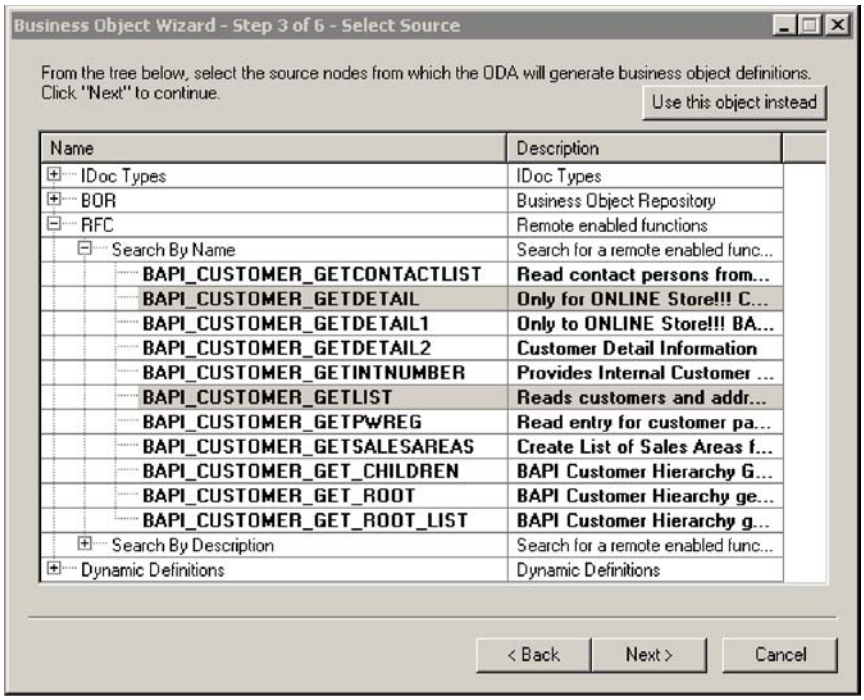


图 29. 为 RFC 节点展开的搜索结果树

ResultSet 对象具有两种属性: Query（用于类型查询对象）和 Result（用于类型结果对象）。Query 属性通常是从 GETLIST BAPI 调用生成的，而 Result 属性是从 GETDETAIL BAPI 调用生成的。

因此，如图 29 中所示，从展开的 RFC 节点中选择相应的 BAPI 调用，在此例中为 BAPI_CUSTOMER_GETDETAIL 和 BAPI_CUSTOMER_GETLIST。由于“配置代理程序”窗口上的 ResultSet 属性设置为 True（如第 61 页的图 26 所示），因此只允许您选择两个 BAPI 调用。

单击**下一步**以进入确认窗口（如第 63 页的图 30 中所示），该窗口允许您确认业务对象的属性的源 BAPI 调用。

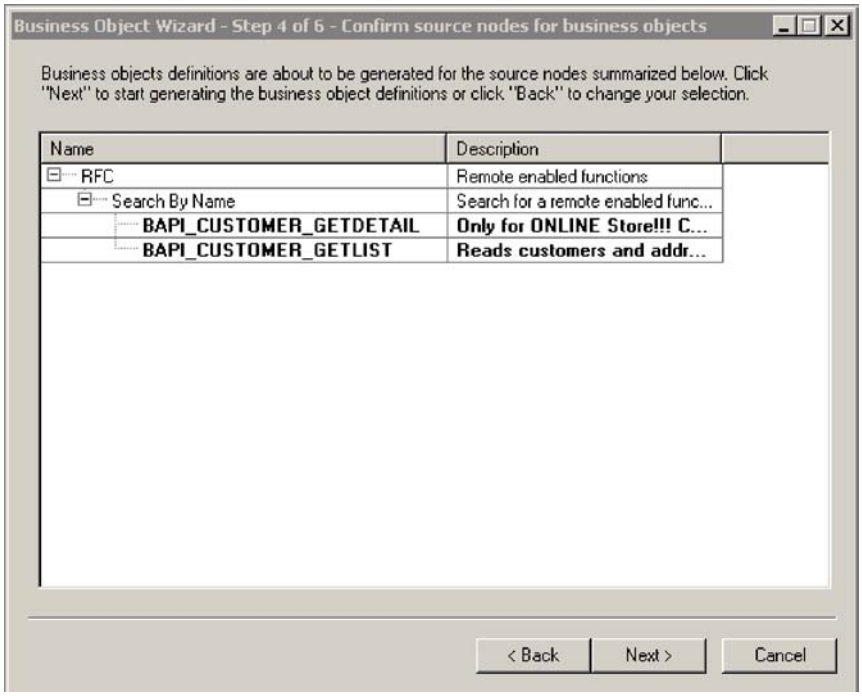


图 30. 确认属性的源节点

如图 31 所示，“业务对象”向导要求您指定 SAPODA 将为业务对象提供的业务对象名前缀（在此示例中为 `sap_`）和 BAPI ResultSet 对象的名称（在此示例中为 `customer_rs`）。`UseFieldName` 属性确定将使用 SAP 中的字段名还是使用字段描述来生成属性名。

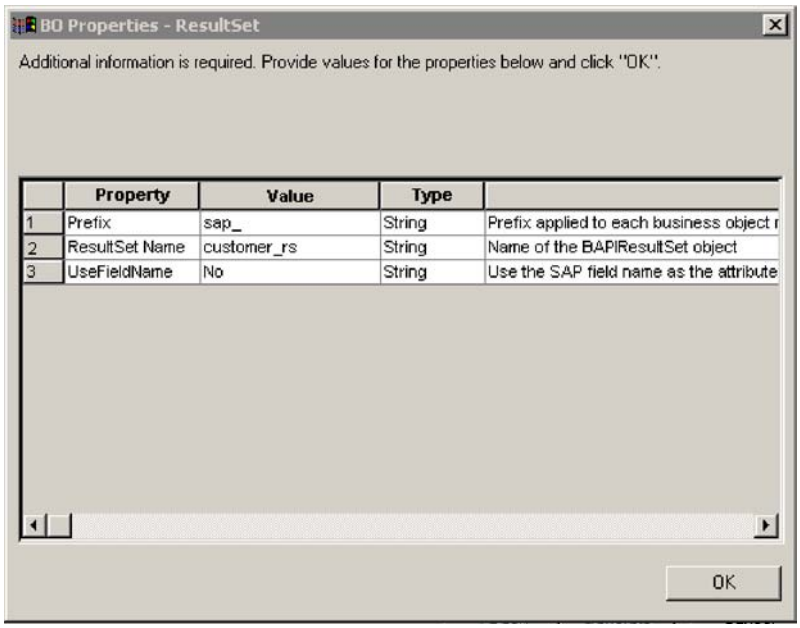


图 31. 提供业务对象名信息

“业务对象”向导还要求您指示应该将您在第 62 页的图 29 中选择的哪个 BAPI 调用用于 Query 属性。从下拉列表中选择 GETLIST BAPI 调用，如图 32 中所示。SAPODA 自动将所选择的其它调用视为 ResultSet 对象的 Result 属性。

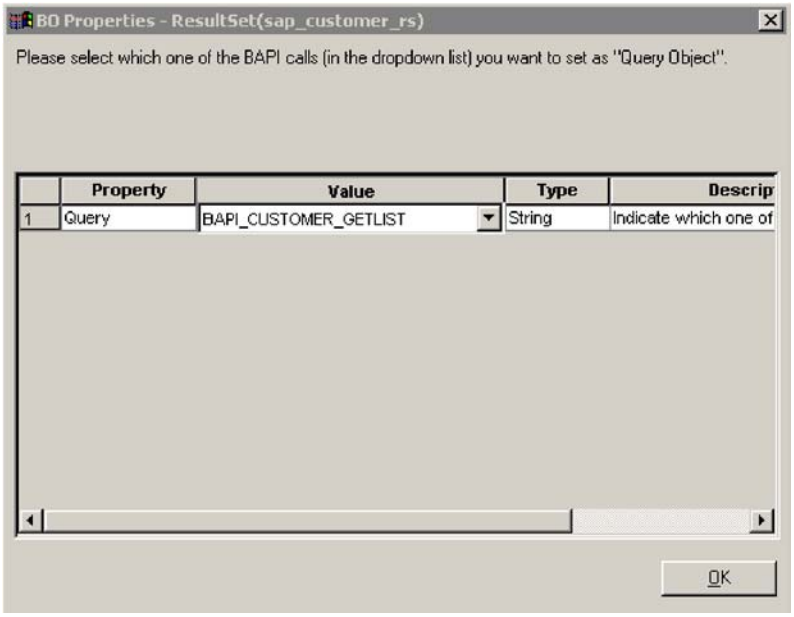


图 32. 指定业务对象的 Query 属性

单击**确定**以进入下一个窗口，在该窗口中指定您在图 32 中选择的 Query BAPI 的 Query 参数（主键）。在此示例中，BAPI 调用是 GETLIST。

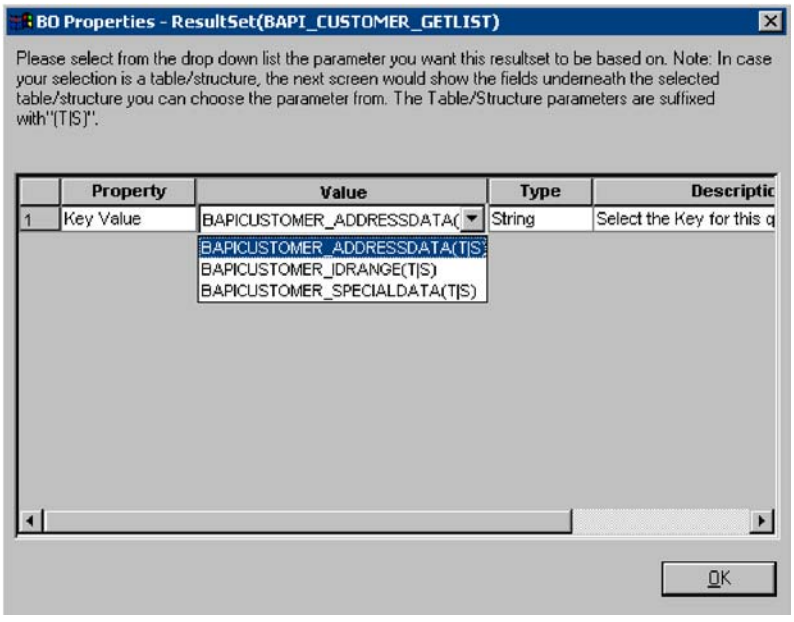


图 33. 选择主键

如果您在前一窗口中选择了一个是表 / 结构（由 T|S 标识）的值，则出现的下一个窗口允许您选择表 / 结构中的特定字段作为主键。在此示例中，选择的字段是 CUSTOMER。

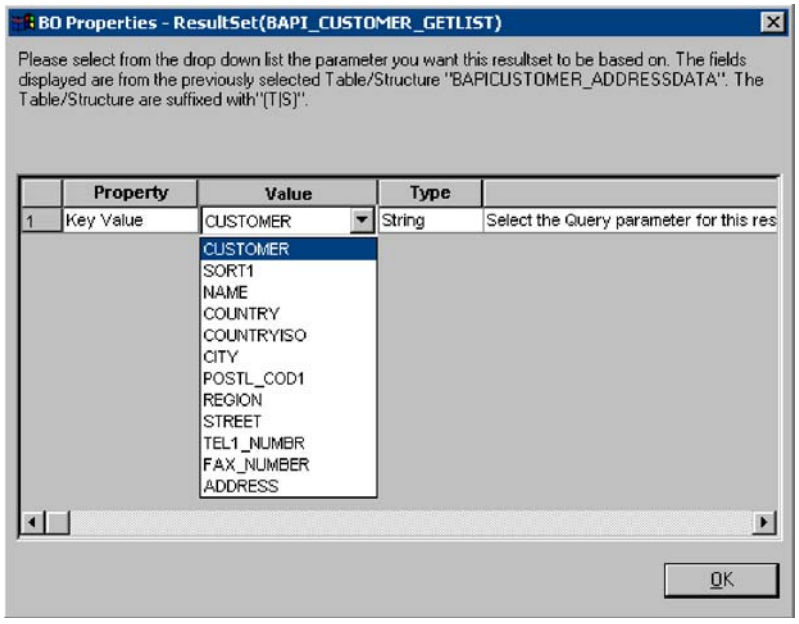


图 34. 选择表／结构上的字段

出现一个消息窗口，指示 Query 参数的完整路径，如图 35 中所示。该路径包括在前两个窗口中选择的 BAPI 调用参数，在此示例中为 BAPICUSTOMER_ADDRESSDATA 和 CUSTOMER。

图 35. Query 属性名的通知

还必须指定 ResultSet 对象的外键，如第 66 页的图 36 所示。外键建立 ResultSet 对象的 Query 属性与 Result 属性之间的关系。

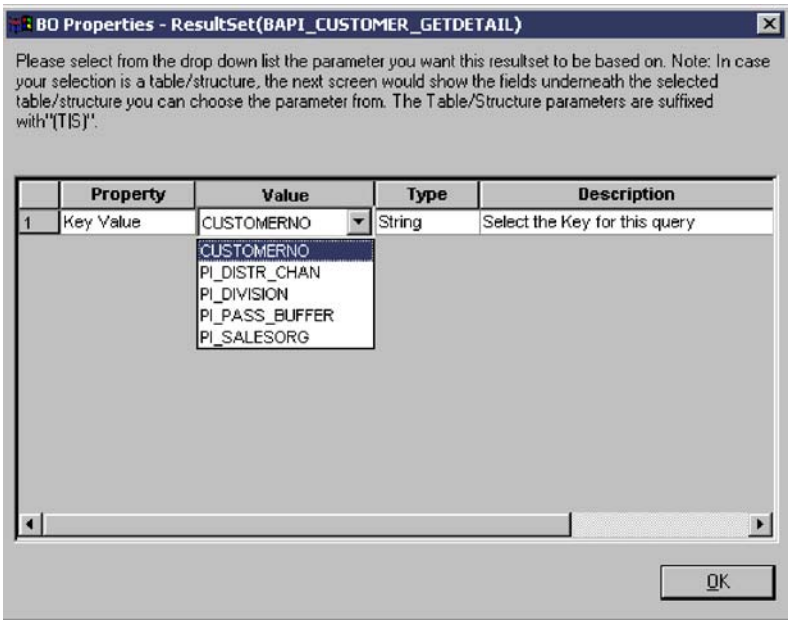


图 36. 选择外键

“业务对象”向导提供了一条消息，用来确认外键的完整路径。在此例中为 BAPI_CUSTOMER_GETDETAIL.CUSTOMERNO，如图 37 中所示。

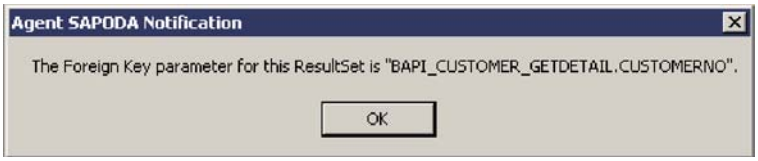


图 37. 外键路径确认

以下窗口指示 GETLIST BAPI 具有可选的参数，并且您可以选择这些可选参数来创建业务对象的相应属性。如果选择**否**（就像本示例中一样），则意味着向导将为所有参数生成业务对象属性。

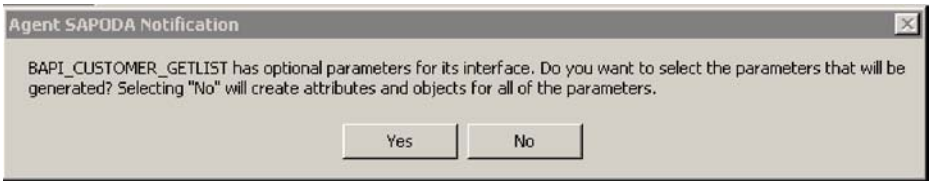


图 38. GETLIST BAPI 的可选参数通知

下一个屏幕（如第 67 页的图 39 中所示）允许您为 ResultSet 对象设置属性值。

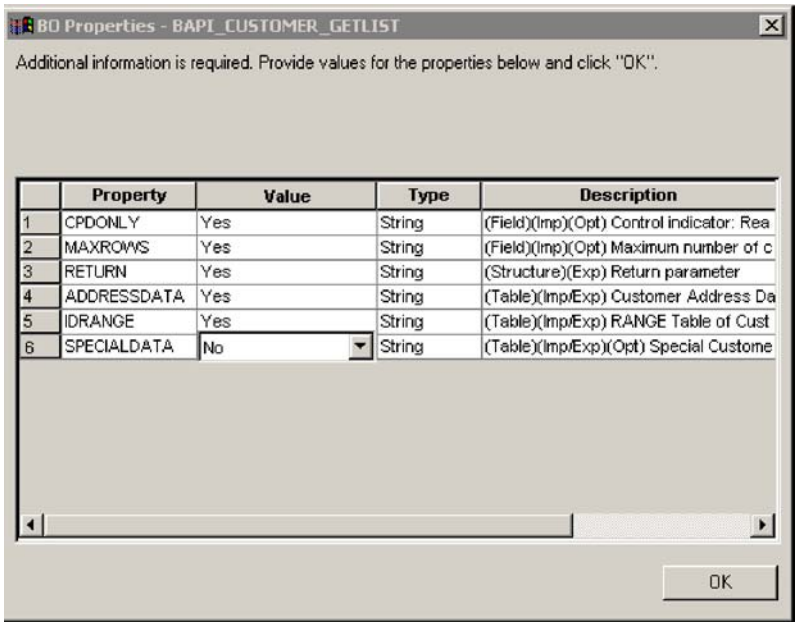


图 39. 为 ResultSet 对象设置属性值

以下窗口指示 GETDETAIL BAPI 具有可选的参数，并且您可以选择这些可选参数来创建业务对象的相应属性。如果选择**否**（就像本示例中一样），则意味着向导将为所有参数生成业务对象属性。

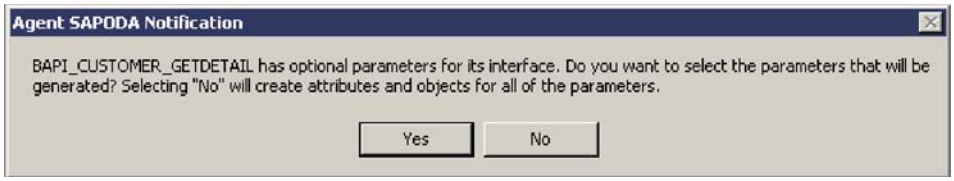


图 40. GETDETAIL BAPI 的可选参数通知

图 41 举例说明了业务对象设计器中的**属性**选项卡，该选项卡列示了 ResultSet 业务对象的两个属性：BAPI_Query 和 BAPI_Result。可以展开业务对象树以显示每个属性的层次结构。注意，这些属性都具有由 SAPODA 添加的后缀 _rs。此后缀确保在先前版本的连接器中创建的业务对象不会被可能同名的新业务对象所覆盖。

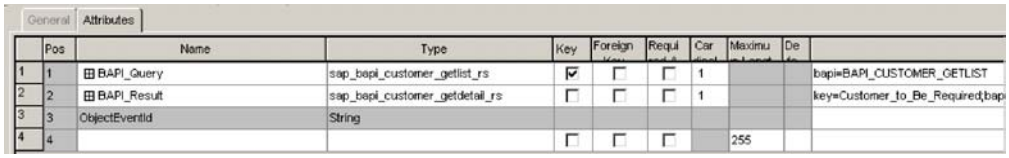


图 41. ResultSet 对象的“属性”选项卡

HDR: 提供其它信息

有两个“BO 属性”窗口用于基于 HDR 表的对象。第一个窗口中显示的属性允许您指定或更改业务对象的前缀。如果您对在“配置代理程序”窗口中为 DefaultBOPrefix 属性输入的值满意，则您不必在此处更改该值。

图 42 举例说明了此窗口。

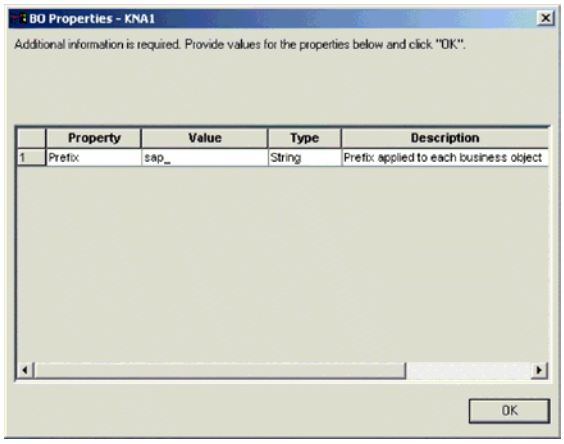


图 42. 提供 HDR 业务对象的其它信息

另外，只能返回表中 512 个字节的信息。当表返回 512 个以上的字节时，则将对您显示图 43 中的对话框。回答“否”将从表的开头返回属性（列描述），直到达到最大值 512 个字节。

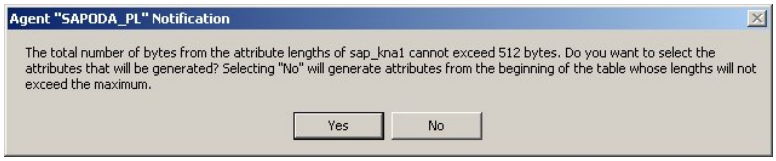


图 43. 512 个字节警告

回答“是”将显示在第 69 页的图 44 中说明的第二个“BO 属性”窗口。将在窗口描述中提供每个属性的长度（以字节计）。您可以通过在“是”和“否”之间切换属性的值来指定包括或排除业务对象的属性。

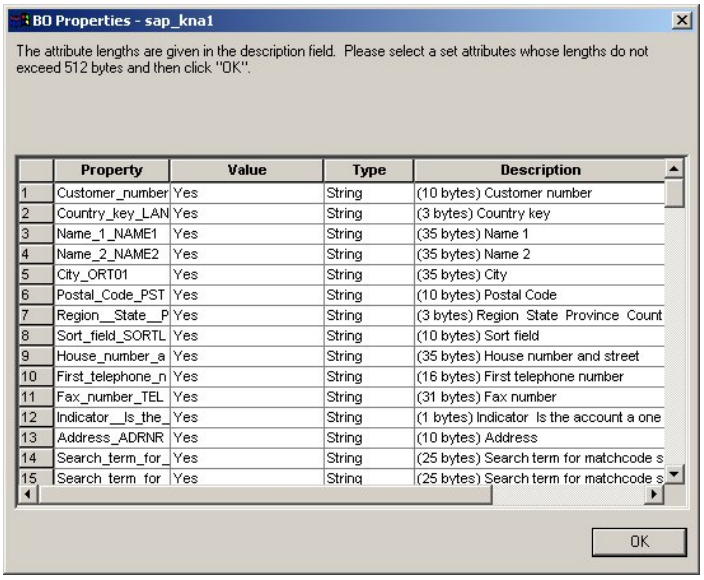


图 44. HDR 业务对象的 BO 属性的大小和类型

保存定义

当在“BO 属性”对话框中提供所有必需的信息并单击“确定”之后，业务对象设计器显示向导中的最后一个对话框。在该对话框中，您可以将定义保存至服务器或保存至文件，您也可以在业务对象设计器中打开该定义进行编辑。有关更多信息，以及为了作出进一步的修改，请参阅 *Business Object Development Guide*。

图 45 举例说明了此对话框。

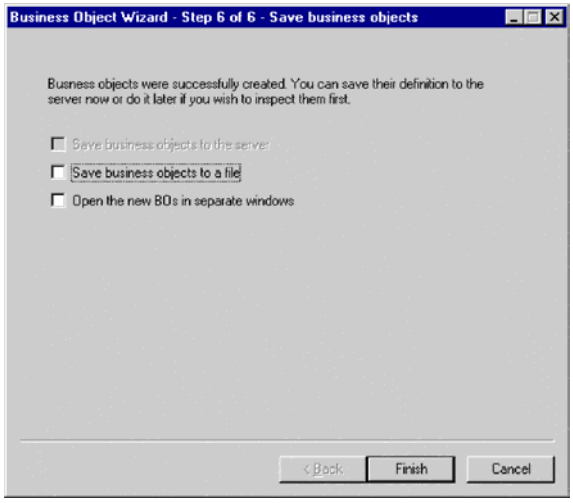


图 45. 保存业务对象定义

在使用 SAPODA 之后

对于 SAPODA 根据 IDoc 类型、BAPI、RFC 支持功能模块或表示业务流程的 SAP 表生成的业务对象定义，它可能并不包含连接器处理业务对象所需要的所有信息。因此，在 SAPODA 完成生成定义之后，您必须将所有必需的信息添加至该定义。使用业务对象设计器来检查和修改业务对象定义，并将修改的定义重新装入或复制到资源库。

当使用业务对象设计器为 ALE 模块将业务对象保存至文件系统时，您首先需要装入 `sap_idoccontrol` 对象。SAP ODA 已交付但未生成此对象，但在将父业务对象保存至文件系统之前需要此对象。

有关修改业务对象定义的更多信息，请参阅 *Business Object Development Guide*。有关特定连接器模块需要的业务对象定义以及在连接器可以处理该业务对象之前您必须进行的修改等信息，请参阅适当模块的文档:

- 第 205 页的第 22 章，『为 ABAP 扩展模块开发业务对象』
- 第 129 页的第 12 章，『为 ALE 模块开发业务对象』
- 第 91 页的第 9 章，『为 BAPI 模块开发业务对象』
- 第 149 页的第 14 章，『为 RFC 服务器模块开发业务对象』
- 第 167 页的第 18 章，『为分层动态检索模块开发业务对象』

第 6 章 连接器故障诊断

本章描述当启动或运行 mySAP.com 适配器（R/3 V3.x）的连接器组件时可能会遇到的问题。

本章包含以下各节:

- 『通用故障诊断』
- 第 72 页的『WBI 性能调整和内存管理』
- 第 73 页的『ABAP 扩展模块故障诊断』
- 第 75 页的『BAPI 模块故障诊断』
- 第 76 页的『RFC 服务器模块故障诊断』
- 第 78 页的『ALE 模块故障诊断』
- 第 80 页的『分层动态检索模块故障诊断』
- 第 82 页的『SAPODA 故障诊断』

通用故障诊断

本节描述当启动或运行 IBM WebSphere Business Integration mySAP.com 适配器的任何模块时可能会遇到的问题。它包括三个方面的故障诊断:

- 『启动问题』
- 第 72 页的『连接器停止运行』
- 第 72 页的『协作未预订业务对象（仅限于 WebSphere InterChange Server）』

启动问题

以下子节对常见启动问题提供了建议。

连接器不能启动

如果当尝试启动连接器时遇到困难:

- 检查以确保集成代理程序已启动并正在运行。
- 检查 SAP 应用程序是否正在运行。
- 验证是否正确设置了标准的和特定于连接器的配置属性。有关更多信息，请参阅第 21 页的第 3 章，『配置连接器』和第 71 页的第 6 章，『连接器故障诊断』。

连接器不能登录到 SAP 应用程序

如果连接器不能登录到 SAP 应用程序:

- 检查 SAP 应用程序是否可用。
- 通过检查 `Sysnr`、`Client`、`Hostname` 和 `Modules` 属性，确保您已正确设置标准的和特定于连接器的连接器配置属性。有关更多信息，请参阅第 21 页的第 3 章，『配置连接器』和第 261 页的附录 D，『连接器的标准配置属性』。
- 验证为连接器设置的用户名和密码是否具有适当级别的优先权。

连接器登录而会话关闭

如果连接器成功登录到 SAP 应用程序，而会话立即关闭，则可能是数据库有问题。检查 PSAPUSER1D 和 PSAPUSER1I 表空间是否分配了足够的空间。缺省情况下，SAP 系统为这两个表空间提供最小的空间。连接器需要比缺省空间量更多的空间。有关更多信息，请参阅第 193 页的『增加日志表空间大小』。

注: 此问题与除 RFC 服务器模块之外的所有连接器模块相关。

连接器停止运行

如果连接器停止运行，并显示一条消息“与 SAP 应用程序的连接已中断”，或者发生 RFC 系统异常，则可能是网络有问题。检查连接器用户的简短转储或发生错误的时间。使用 IBM CrossWorlds Station 工具或转至事务 ST22。如果您仍需要更多信息，则通过转至事务 SM21 来检查系统日志。

未设置缺省值

业务对象中已设置缺省值，但连接器未选取这些值。这是配置问题。为了使用缺省值，UseDefaults 连接器属性需要设置为 true，且需要缺省值的每个属性在业务对象定义中都必须标记为必需的。

协作未预订业务对象（仅限于 WebSphere InterChange Server）

如果协作未预订指定的 WebSphere InterChange Server 上的特定业务对象，则执行以下操作:

- 检查是否已配置协作来预订该特定业务对象。
- 验证协作是否在运行。
- 验证映射引用是否已指定正确的业务对象作为源业务对象。

对二进制数据编码（仅限于 MQ Integrator Broker）

对于具有二进制数据（SAP 系统中的 RAW 数据类型）的字段，适配器将以十六进制而不是 XML MQ 消息中更典型的基本 64 位（base64）编码来编码这些字段的值。另外，适配器还希望来自服务调用请求的数据在 XML MQ 消息中使用十六进制编码。

WBI 性能调整和内存管理

Java 虚拟机（JVM）使多个调整旋钮外部化，这些旋钮可用来提高 WebSphere Business Integration 应用程序的性能。这些旋钮控制与垃圾回收、堆大小、线程技术和锁定相关的问题。因为 ICS 服务器及其组件（映射和协作）以及大多数适配器都是用 Java 语言编写的，所以 JVM 的性能对由 ICS 应用程序提供的性能有显著影响。

有关 WebSphere Business Integration 的性能注意事项的详细描述，浏览至 IBM 软件支持站点并搜索“WBI 性能调整”，或者查阅以下文档，该文档将定期更新:

http://www-1.ibm.com/support/docview.wss?rs=203&context=SW000&q1=wbi+performance+tuning&uid=swg21173114&loc=en_US&cs=utf-8&lang=en

以下 URL 提供 JVM 选项的有用总结: http://java.sun.com/docs/hotspot/VMOptions.html

以下 URL 提供关于 HotSpot 引擎的有用 FAQ: http://java.sun.com/docs/hotspot/PerformanceFAQ.html#20

ABAP 扩展模块故障诊断

本节描述当启动或运行 ABAP 扩展模块时可能会遇到的问题。它包括三个方面的故障诊断:

- 『传送文件』
- 『启动问题』
- 第 74 页的『事件处理』
- 第 74 页的『Microsoft Windows 上的事件分布问题（仅限于连接器版本 4.2.7）』

传送文件

如果在安装 ABAP 扩展模块的适配器传送文件时发生错误:

- 验证您是否安装了正确的传送文件。传送文件安装在它们自己的目录中，该目录随 SAP 应用程序的版本不同而不同。有关传送文件及其安装目录的详细信息，请参阅第 189 页的『传送文件』。
- 验证您是否以正确的顺序安装了传送文件。某些传送文件具有相关性，如现有的表。

  例如，一个传送文件创建表的数据元素，另一个传送包为该数据元素创建表。如果首先未创建表，则系统返回错误。
- 验证是否正确安装了所有必需的传送文件。每个传送文件为连接器添加特定的功能。有关更多信息，请参阅第 189 页的『连接器传送文件安装』。

启动问题

如果连接器成功登录到 SAP 应用程序，但 SAP 应用程序中连接器的日志为空:

- 检查日志记录是否已打开。如果日志记录是关闭的，则使用 IBM CrossWorlds Station 打开它。缺省情况下，日志记录是打开的。有关更多信息，请参阅第 233 页的第 25 章，『管理 ABAP 扩展模块』。
- 检查连接器登录到的机器是否就是您正在查看连接器日志文件的同一机器。
- 检查 `Namespace` 配置属性是否设置为 `true`。如果您已从先前 `YXR` 环境升级至连接器的名称空间，则连接器仍可能登录到 `YXR` 环境。如果是这种情况，则将 `Namespace` 配置属性设置为 `true`。有关更多信息，请参阅第 283 页的『特定于连接器的配置属性』中的第 288 页的『Namespace』属性。
- 检查连接器日志中的号码范围是否同步。如果您已升级 NumberRange 传送号，则号码范围间隔可能不同步。验证号码范围对象号是否低于连接器日志中的第一个号码。

  要检查号码范围，转至 SNRO 并在“号码范围对象”字段中输入 `YXR_LOG`。单击“号码范围”按钮，单击“显示间隔”按钮，并记下号范围对象号。打开连接器日志并记下第一个条目的号码。如果此号码高于号码范围对象号，则需要将连接器日志中的日志条目号修改为一个更高的号码。有关更多信息，请参阅第 193 页的『验证传送对象的号码范围』。

Microsoft Windows 上的事件分布问题（仅限于连接器版本 4.2.7）

在 Windows 上升级至 IBM CrossWorlds SAP 连接器版本 4.2.7 之后，在以下情况下，事件保留在事件表中，且连接器不会选取和处理这些事件:

- 您在多个连接器之间配置了事件分布。
- 连接器正在装入了 NON 名称空间（yxr）的 SAP 3.x 系统上运行。

此问题是由 SAP 在其 java API（SAPJCo）中所作的更改引起的。

要修正此问题，装入仅更改由 IBM WebSphere Business Integration mySAP.com 适配器提供的事件请求和事件返回功能模块的补丁传送包。在不具有名称空间（/CWLD/）基础结构的 4.0 和 4.5 SAP 系统中装入此补丁传送包。

注: 名称空间 ABAP 基础结构没有此问题。

事件处理

以下子节对事件处理问题提供了建议。

预订业务对象未调用 ABAP 扩展模块

如果 ABAP 扩展模块不处理预订业务对象，则执行以下操作:

- 检查是否设置了可视连接器框架来调用 ABAP 扩展模块。Modules 属性必须设置为 Extension。
- 检查连接器是否预订该业务对象。

连接器不选取事件

如果您的连接器不从 SAP 应用程序中选取事件:

- 检查 SAP 应用程序中连接器的事件表以查看是否为您的连接器将该事件排队。
- 在多连接器环境中，如果未将该事件排队，则确保“事件分布”表（YXR_EVTDIS）中存在一个条目表示您的连接器和业务对象的组合。检查此组合是否是唯一的。

  如果您已让多个连接器预订同一业务对象，则一个连接器可能正在处理错误的事件。
- 如果 SAP 应用程序中事件具有锁定对象，则 SAP 应用程序可能不会完成该事件的保存过程的处理。

  检查 SAP 应用程序中连接器的事件表以查看该事件的状态是否为 L（已锁定）。如果状态为 L，则很可能是在 SAP 应用程序而不是在连接器中发生问题。
- 连接器在处理事件时可能已停止运行。检查该事件在 SAP 应用程序的连接器事件表中的状态。如果状态为 R（已检索），则尚未将事件移至归档表。如果事件的状态是 R，则验证事件是否未成功到达目标位置。

  如果事件未成功到达目标位置，则将状态从 R 更改为 Q（已排队）。连接器在下一个轮询时间间隔选取状态为 Q 的事件。要将状态从 R 更改为 Q，转至事件表、选择该事件并单击“编辑”按钮。在显示的窗口中，将“事件状态”字段从 R 更改为 Q。

业务对象未能处理

如果业务对象未能成功处理，则检查 SAP 应用程序中的连接器日志。失败事件的条目以红色出现。使用重新处理工具重新处理事件，该工具使您能够像您逐步执行事务一样在代码中设置断点。

警告: 不要在生产环境中使用重新处理工具，因为它会导致 WebSphere Business Integration 系统和 SAP 应用程序不同步。

事件表的死锁

当前事件表和将来的事件表可能会在一次添加许多事件时遇到死锁情况。如果由于数据库调整而未使用为事件表提供的索引，则会发行这种情况。调整通常在非高峰期间发生，这时事件表中只有很少的事件或没有事件。当数据库表没有或只有很少条目时，不使用索引来读取表效率会更高。要避免死锁情况，在运行数据库调整实用程序时排除当前事件表和将来的事件表。

大对象

大对象可能需要进行额外更改才能成功处理。在将数据传递至 SAP 应用程序之前，将把 ABAP 扩展模块对象转换为平面结构，或者在从 SAP 应用程序接收数据时将该对象从平面结构转换。有关更多信息，请参阅第 196 页的『业务对象至平面结构的转换』。此平面结构保持在内存中，对象实例的每个属性为该结构中的一行。对于每个属性，将在连接器和 SAP 应用程序之间传递 373 个字节的数据。属性数乘以 373 将给出平面结构的近似大小。另外，对象的实例也在内存中。因此，具有许多子对象（段）的对象可能需要在连接器的 Java 进程的启动脚本中对 Java 堆大小作出更改，以避免内存不够错误。

Windows

在 `start_SAP.bat` 脚本中，将 `-mx128m` Java 堆大小选项参数缺省值更改为足够处理平面结构和对象实例的值。大于运行 Java 进程的机器上可用内存的值也会导致内存不够错误。`128m` 表示最大 Java 堆大小为 128 MB。

Unix:

SAP 应用程序还可能需要更改 ABAP 超时参数以成功处理大对象。

BAPI 模块故障诊断

本节描述当运行 BAPI 模块时可能会遇到的问题。

请求过程处理

以下子节对常见请求过程处理问题提供了建议。

预订业务对象未调用 BAPI 模块

如果 BAPI 模块未在处理预订业务对象，则执行以下操作:

- 检查可视连接器框架是否设置为调用 BAPI 模块。`Modules` 属性必须设置为如下所示: `Bapi`。
- 检查连接器是否预订该业务对象。

- 检查是否任何定制业务对象处理程序文件都在 \bapi\client 目录中。如果类文件不在此目录中，则不会调用定制业务对象处理程序来处理业务对象。有关更多信息，请参阅第 156 页的『使用生成的业务对象和业务对象处理程序』。
- 检查特定于应用程序的业务对象查询描述信息中的定制业务对象处理程序名称是否正确。有关更多信息，请参阅第 96 页的『特定于应用程序的业务对象信息』。
- 确保在生成定制业务对象处理程序时，指定了适当的查询描述来与 BAPI 匹配。

业务对象未能处理

如果业务对象未能成功处理:

- 检查您正在使用的 BAPI 是否具有返回业务对象。BAPI 模块查看返回业务对象以获取具有关键字 e（错误）或 a（异常终止）的消息。如果该模块找到其中一个关键字，则这说明该事件已失败。如果 BAPI 不具有返回业务对象，则确保您实现自己的错误处理。
- 使用事务 SE37 来测试与失败事件相关的 BAPI。这应该使您能够重现该故障。

  如果这不起作用，则在从内部格式至外部格式的转换中可能发生问题。检查您是否正在以正确的格式来指定值。例如，对于日期，SAP 的内部格式为 YYYYMMDD，而您可能指定格式 MMDDYYYY。因为指定的格式不被识别，所以这会导致 BAPI 失败。
- 检查每个属性的特定于应用程序的信息是否正确。如果这些值不正确，则 BAPI 模块在将对象发送回 SAP 应用程序之前不会正确填充该对象。
- 检查是否正确指定了 I 和 E 参数。记住，I 标识导入参数，E 标识导出参数。有关更多信息，请参阅第 77 页的『业务对象未能处理』。

连接器似乎在轮询，但不选取事件

BAPI 模块包括 pollForEvents() 方法的哑元实现。因为连接器返回一条轮询消息，所以连接器似乎正在轮询。BAPI 模块不支持轮询，所以忽略这些消息。

如果您要为 BAPI 模块实现轮询，则必须在 ABAP 扩展模块中使用轮询能力。有关更多信息，请参阅第 179 页的第 19 章，『ABAP 扩展模块概述』。

RFC 服务器模块故障诊断

本节描述当启动或运行 RFC 服务器模块时可能会遇到的问题。本节包括:

- 第 71 页的『启动问题』
- 第 72 页的『连接器停止运行』
- 第 74 页的『事件处理』

启动问题

如果连接器不能向 SAP 应用程序注册:

- 检查 SAP 应用程序是否可用。
- 检查是否正确设置了标准的和特定于连接器的连接器配置属性。特别是，检查 gwService、Hostname、RfcProgramId 和 Modules 属性。有关更多信息，请参阅第 21 页的第 3 章，『配置连接器』、第 261 页的附录 D，『连接器的标准配置属性』和第 71 页的第 6 章，『连接器故障诊断』。

连接器停止运行

如果您的连接器停止运行，则执行以下操作:

- 检查 RFC 服务器模块是否正在衍生线程。验证是否正确设置了 NumberOfListeners 属性。有关更多信息，请参阅第 288 页的『NumberOfListeners』。
- 验证是否已设置 RFC 程序标识以便 RFC 服务器模块向 SAP 网关注册它自己。有关更多信息，请参阅第 289 页的『RfcProgramId』和第 159 页的『向 SAP 网关注册 RFC 服务器模块』。

事件处理

以下子节对常见事件处理问题提供了建议。

预订业务对象未调用 RFC 服务器模块

如果 RFC 服务器模块不处理预订业务对象，则执行以下操作:

- 检查是否已设置可视连接器框架来调用 RFC 服务器模块。第 288 页的『Modules』属性必须设置为如下所示: RfcServer。
- 检查连接器是否预订该业务对象。
- 检查 SAPODA 生成的特定于 BAPI 的业务对象处理程序类文件是否在 \bapi\server 目录中。如果该类文件不在此目录中，则不会调用 BAPI 业务对象处理程序来处理该业务对象。有关更多信息，请参阅第 156 页的『使用生成的业务对象和业务对象处理程序』。
- 检查特定于应用程序的业务对象查询描述信息中 BAPI 业务对象处理程序名称是否正确。有关更多信息，请参阅第 75 页的『业务对象未能处理』。
- 检查为特定于 BAPI 的业务对象处理程序指定的查询描述对于您需要的处理类型是否正确。特别是，确保您在生成业务对象处理程序时指定了适当的查询描述来与 BAPI 匹配。有关更多信息，请参阅第 156 页的『使用生成的业务对象和业务对象处理程序』。

业务对象未能处理

如果业务对象未能成功处理:

- 检查您正在使用的 BAPI 是否具有返回业务对象。RFC 服务器模块查看返回业务对象，以获取具有关键字 e（错误）或 a（异常终止）的消息。如果该模块找到其中一个关键字，则这说明该事件已失败。如果 BAPI 不具有返回业务对象，则确保您实现自己的错误处理。
- 使用事务 SE37 来测试与失败事件相关的 BAPI。这应该使您能够重现该故障。

  如果这不起作用，则在从内部格式至外部格式的转换中可能发生问题。检查您是否正在以正确的格式来指定值。例如，对于日期，SAP 的内部格式为 YYYYMMDD，而您可能指定格式 MMDDYYYY。因为指定的格式不被识别，所以这会导致 BAPI 失败。
- 检查每个属性的特定于应用程序的信息是否正确。如果这些值不正确，则 RFC 服务器模块在将对象发送回 SAP 应用程序之前不会正确填充该对象。
- 检查是否正确指定了 I 和 E 参数。I 参数标识导入参数，而 E 参数标识导出参数。有关更多信息，请参阅第 76 页的『业务对象未能处理』。

ALE 模块故障诊断

本节描述当启动或运行 ALE 模块时可能会遇到的问题。它包括以下主题:

启动问题

以下子节对常见启动问题提供了建议。

连接器不能登录到 SAP 应用程序或向它注册

如果连接器不能登录到 SAP 应用程序或向它注册:

- 检查 SAP 应用程序是否可用。
- 检查是否正确设置了标准的和特定于连接器的连接器配置属性:
  - 检查是否已创建必需的 MQSeries 队列及其相应的配置属性是否正确地指定其名称。
  - 对于请求处理，检查 Sysnr、Client、Hostname 和 Modules 属性。
  - 对于事件处理，检查 gwService、Hostname、RfcProgramId 和 Modules 属性。

  有关更多信息，请参阅第 21 页的第 3 章，『配置连接器』和第 71 页的第 6 章，『连接器故障诊断』。
- 验证为连接器设置的用户名和密码是否具有适当级别的优先权。

连接器不轮询事件

如果连接器不轮询来自 SAP 应用程序的事件:

- 检查特定于应用程序的查询描述信息，以查看是否已修改所需查询描述以具有正确的消息类型、消息代码和消息功能。
- 检查查询描述 AleOutboundVerbs 是否存在以及是否具有有效查询描述的列表。

连接器似乎在轮询，但不选取事件

- 检查是否已正确创建事件队列（SAPALE_Event_Queue 和 SAPALE_Wip_Queue），以及是否正在 event 队列上执行轮询。
- 验证以下各项在系统上是否正在运行:
  - MQSeries
  - TCP/IP
- 验证 SAP 应用程序内的 ALE 配置是否正确；有关更多信息，请参阅第 109 页的第 10 章，『ALE 模块概述』。
- 检查连接器是否至少执行了一次轮询调用；这样做将安装功能模块以执行事件处理。
- 检查是否已将一条消息写入 wip 队列并且已移至 event 队列。

事件处理

连接器将有关 JMS-MQ 事件消息（在 SAPALE_Event_Queue 配置属性中指定的队列中）中已成功处理的 IDoc 的信息写入 EventState.log 文件。此文件位于 AleEventDir 配置属性中指定的目录中。

注: 每条事件消息都可以包含多个 IDoc，每个 IDoc 表示一个业务对象。

如果连接器在处理当前事件消息中的所有 IDoc 之前当机，则它在恢复期间使用 EventState.log 文件以确保它只将每个 IDoc 发送至集成代理程序一次。

重要提示: 连接器在首次处理事件时不会自动创建该日志文件。在首次运行连接器之前，您必须创建此文件。

日志文件的格式为:

TID: OS, 1S, 2F, 3U

其中 <*TID*> 是正在处理的当前事务标识，每个数字表示事件消息中所有工作单元的序号。

例如，如果连接器已成功处理当前事件消息中前 4 个 IDoc 中的 3 个，第二个 IDoc 在处理时失败，并且连接器尚未完成处理当前事件消息，则 EventState.log 文件可能显示:

<TID> :: OS, 1F, 2S, 3S

如果连接器在处理整个事件消息之前当机，则连接器在启动时将使用该日志文件中的信息来在它停止处理的位置继续处理消息中的事件。连接器读取该日志以获取要恢复的事件的事务标识、最新的工作单元和每个工作单元的状态。然后连接器开始将一些业务对象发送至集成代理程序，这些业务对象表示事件消息中序号大于日志文件中最后一个号码的每个 IDoc。在先前示例中，连接器将处理当前事件消息中第 5 个 IDoc。

连接器将该日志文件的内容保留在内存中以增强性能。连接器访问磁盘上的该文件只是为了用新条目更新它。连接器仅在恢复时读取该日志文件。

有关连接器在恢复过程中如何使用 EventState.log 文件的信息，请参阅『故障恢复』。

故障恢复

注: 如果发生磁盘故障或磁盘已满，则这些恢复步骤不适用。

为了在事件通知期间从故障中恢复，连接器执行以下操作:

1. 连接器处理事件队列（在 SAPALE_Event_Queue 配置属性中指定）的 JMS-MQ 消息中的 IDoc。当它成功处理一个 IDoc 时，连接器将一个条目写入 EventState.log 文件。
   - 如果事件消息中没有任何工作单元在处理时失败，则连接器以 IDocProcessStatus 的 success 值将该事件消息移至归档队列。
   - 如果事件队列消息中有任何工作单元在处理时失败，则连接器将把该事件消息移至归档队列并将 IDocProcessStatus 更新为 partial 值。
2. 在连接器处理事件消息中的所有 IDoc 之后，它清除 EventState.log 文件并开始将条目写入它以获取下一个事件消息。

3. 如果连接器在处理事件消息中的所有 IDoc 之前当机，则它使用 EventState.log 中的信息来确定在恢复过程期间从何处开始处理。在恢复后，连接器检查该日志文件中是否存在任何条目。
   - 如果没有任何条目，则连接器将事件消息中的所有 IDoc 都发送至集成代理程序。
   - 如果存在条目，则连接器将使用此信息来在它停止处理的位置继续处理事件消息。连接器读取该日志以获取要恢复的事件消息的名称和最后一个 IDoc 序号。然后连接器将事件消息中序号大于日志文件中最后一个号码的每个 IDoc 发送至集成代理程序。在此示例中，将把该事件消息移至归档队列，并且根据 EventState.log 中每个工作单元的状态来更新 IDocProcessStatus。

     使用该日志文件来防止连接器将相同的 IDoc 多次发送至集成代理程序。连接器将该日志文件保留在内存中以增强性能。连接器访问磁盘上的该文件只是为了用新条目更新它，仅在恢复时才读取该日志文件。

     注: 如果事件消息中任何 IDoc 的序号都不大于日志文件中的最后一个号码，则连接器是在处理最后一个事件之后但在归档事件文件之前当机的。在这种情况下，将把该事件消息移至归档队列，并且根据 EventState.log 中每个工作单元的状态来更新 IDocProcessStatus。

从业务对象创建错误中恢复

如果连接器仅已创建 WIP 队列中消息的头部分，但未创建数据部分，则此过程将恢复该消息的数据部分。

1. 检查 SAP 连接器日志以获取与业务对象的名称、消息类型或查询描述相关的错误消息。
2. 对业务对象定义或连接器配置作出适当的修正。

   注: 配置更改可以包括对 MQSeries 队列的更改。有关更多信息，请参阅第 115 页的『运行 ALE 模块的先决条件』。
3. 重新启动连接器。

请求处理

如果 ALE 模块未在处理预订业务对象，则执行以下操作:

- 检查可视连接器框架是否设置为调用 ALE 模块。Modules 属性必须设置为 ALE。
- 检查连接器是否预订该业务对象。

分层动态检索模块故障诊断

本节描述当启动或运行分层动态检索模块时可能会遇到的问题。它包括以下几个方面:

- 第 81 页的『错误处理和日志记录』
- 第 82 页的『SQL SELECT 失败』

错误处理和日志记录

连接器在遇到导致检索失败的情况时将一条错误消息写入日志。当发生这样的错误时，连接器还会打印从集成代理程序接收到的失败业务对象的文本表示。它将该文本写入连接器日志文件或标准输出流，这取决于其配置。您可以使用该文本来查找错误源。

错误类型

表 7 描述了分层动态检索模块在每个跟踪级别输出的跟踪消息的类型。此外，这些消息还是 WebSphere Business Integration 系统体系结构（如 Java 连接器执行包装程序和 WebSphere MQSeries 消息接口）的任何跟踪消息输出。

表 7. 连接器跟踪消息

跟踪级别	跟踪消息
级别 0	标识连接器版本的消息。 此级别不执行其它跟踪。
级别 1	功能模块进入和退出消息。无论连接器执行线程何时进入或退出功能，都会写入这些消息。这些消息有助于跟踪连接器的进程流。
级别 2	包含诸如连接器在处理业务对象期间遇到或检索的数组和子业务对象等信息的业务对象处理程序消息。
级别 3	• 包含诸如连接器何时在业务对象中发现或设置外键等信息的外键处理消息。 • 提供关于业务对象处理的信息的消息。例如，当连接器在业务对象之间查找匹配、在一组子业务对象中查找业务对象或在检索期间除去子业务对象时，将传递这些消息。
级别 4	• 特定于应用程序的参考消息，例如，显示由功能返回的值的消息，该功能对业务对象的特定于应用程序的信息属性进行语法分析。 • 标识连接器何时进入或退出 Java 方法的消息，这些消息有助于跟踪连接器的进程流。 • SQL 语句。在此级别和以上级别，连接器打印它执行的所有 SQL 语句。 • 在检索期间对属性值的更改。在此级别和以上级别，连接器打印属性的名称及其新的值。
级别 5	• 指示连接器初始化的消息，例如，显示从集成代理程序检索到的每个配置属性的值的消息。 • 包含业务对象转储的消息。 • 包含连接器开始处理业务对象之前业务对象的表示（当连接器从集成代理程序接收到业务对象时显示业务对象的状态）和在连接器完成其处理之后业务对象的表示（当连接器将业务对象返回至集成代理程序时显示业务对象的状态）的消息。

连接器消息文件

连接器生成的错误消息存储在名为 `SAPConnector.txt` 的消息文件中。每个错误都有一个错误号，后跟错误消息。例如:

```
1210

SAP 分层动态检索模块无法初始化。
```

1211

SAP 分层动态检索模块未能找到...

未能调用 RFC_READ_TABLE

SAP RFC_READ_TABLE 功能不处理基于字符的数据类型。如果字段使用以下数据类型，则该模块在检索数据时可能失败:

- CURR
- DEC
- FLTP
- INT1
- INT2
- INT4
- LRAW
- RAW
- RAWSTRING

SQL SELECT 失败

如果 SELECT 语句失败，则检查标记为“键”或用作外键的任何简单属性是否包含单引号（'）。如果包含，则修改业务对象的映射以将单引号（'）转换为两个单引号（''）。

SAPODA 故障诊断

当使用 SAPODA 时您可能会遇到两个已知问题:

- SAPODA 运行但没有消息。

  如果为 ODA 指定的消息文件不存在，则 ODA 运行但没有消息。当业务对象设计器显示消息文件的缺省名称时，将在 ODA 的配置期间导致此问题。缺省名称遵循命名约定:

  *AgentName*Agent.txt

  如果实际消息文件的名称不遵循此约定，且未用实际值覆盖缺省值，则业务对象设计器在启动 ODA 的窗口中显示一条错误消息。此消息不会在业务对象设计器中弹出。

  有关更多信息，请参阅第 43 页的『使用错误消息文件和跟踪消息文件』。

- 在 Windows 系统上，如果业务对象设计器在 Path 环境变量中找不到必需的库文件，或这些文件不在系统上，则它在尝试获取树节点时显示一个 CORBA 异常。有关这些文件的信息，请参阅第 41 页的『在使用 SAPODA 之前』。

第 2 部分 BAPI 模块

第 7 章 BAPI 模块概述

本章介绍 IBM WebSphere Business Integration mySAP.com 适配器的 BAPI 模块。BAPI 模块使集成代理程序能够使用 BAPI 将业务对象发送至 SAP 应用程序。

BAPI 是 SAP 的标准商业应用编程接口，这些接口使第三方能够与 SAP 应用程序交互。它们是作为 SAP 业务对象方法的 RFC 支持功能模块实现的。

注: BAPI 只是 SAP 应用程序中的 RFC 支持功能。除了提供 BAPI 之外，BAPI 模块还可以用来支持任何 RFC 支持功能。

BAPI 模块组件

BAPI 模块是用 Java 语言编写的连接器模块，它直接支持对 SAP 应用程序的本机 BAPI 调用。它通过实现 VisionConnectorAgent 和 VisionBOHandler 类来扩展可视连接器框架。BAPI 模块使用以 Java 语言和 C 语言编写的 SAP RFC 库，这些库使外部程序能够与 SAP 应用程序通信。

图 46 举例说明了 BAPI 模块的完整体系结构。BAPI 模块由连接器框架、BAPI 的特定于应用程序的连接器组件、支持所有 BAPI 调用的单个 BAPI 业务对象处理程序以及 SAP RFC 库组成。除了 BAPI 模块提供的那个 BAPI 业务对象处理程序以外，您还可以创建定制业务对象处理程序，如第 101 页的『使用定制业务对象处理程序』中所述。

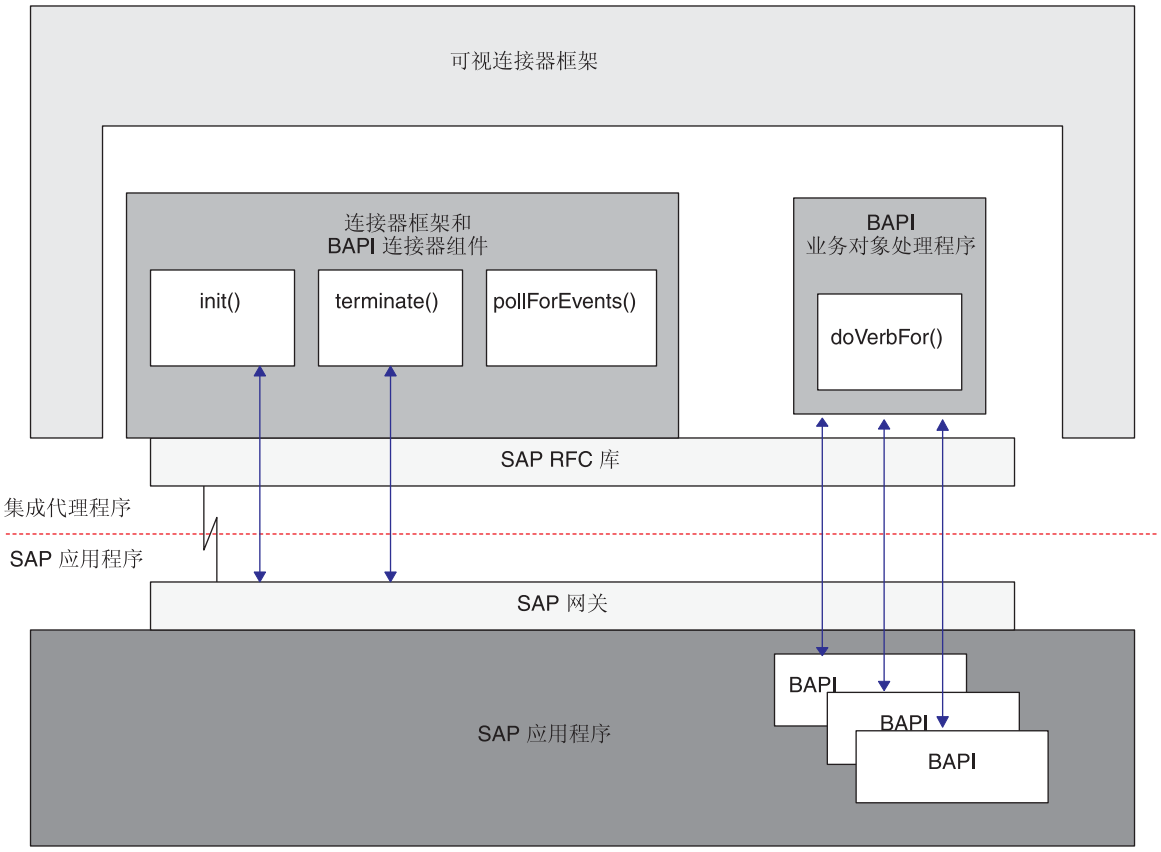


图 46. BAPI 模块体系结构

BAPI 模块组件:

- 使用 SAP RFC 库和 SAP 网关打开与 SAP 应用程序的 RFC 连接。
- 处理来自集成代理程序的请求并调用 SAP 应用程序中的 BAPI。
- 终止与 SAP 应用程序的连接。

BAPI 模块的工作方式

BAPI 模块实现 `init()`、`terminate()`、`pollForEvents()` 和 `doVerbFor()` 方法。但是，由于 BAPI 模块仅支持请求操作，所以未使用 `pollForEvents()` 方法。

初始化和终止

`init()` 方法通过 SAP 网关打开与 SAP 应用程序的 RFC 连接。如果连接器未能初始化，则它使用 `terminate()` 方法来终止。连接器通过断开与 SAP 网关的连接来终止。

业务对象处理

实现一次可视连接器框架的业务对象处理程序中的 `doVerbFor()` 方法可启动所有业务对象请求。可视业务对象处理程序处理在 BAPI 模块和集成代理程序之间传递的所有业务对象。在 BAPI 模块中，单个 BAPI 业务对象处理程序支持所有 BAPI 调用。

图 47 举例说明了 BAPI 模块的业务对象处理。

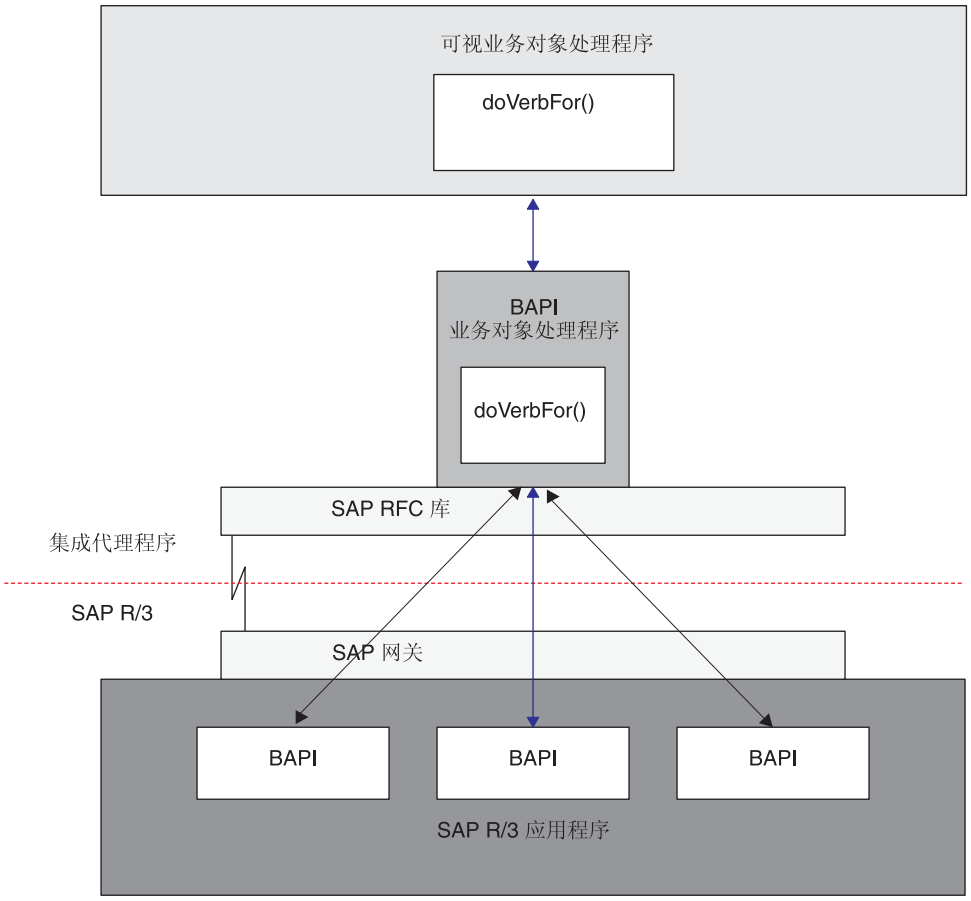


图 47. BAPI 模块的业务对象处理

一旦被可视业务对象处理程序调用，BAPI 业务对象处理程序就以下列方式执行:

1. 从可视业务对象处理程序接收 SAP 的 WebSphere 业务对象。
2. 用业务对象数据填充 BAPI 参数。
3. 使用 RFC 执行 BAPI 调用并将 BAPI 参数传递至 SAP 应用程序。业务对象处理程序等待返回业务对象数据。
4. 接收业务对象数据（BAPI 参数）。
5. 将 BAPI 参数重新转换为 WebSphere 业务对象数据。
6. 将业务对象传递至可视业务对象处理程序并最终传递至集成代理程序。

注: 如果 “BAPI 模块” 具有返回结构或返回表，则连接器检查消息类型 A（异常终止）和 E（错误）来确定是否成功处理了事件。消息类型 A 或 E 指示未能处理事件。如果 BAPI 模块不具有返回结构或返回表，则您必须实现您自己的错误处理。当消息代理是集成代理程序时，将把在连接器框架中失败的事件移至故障队列；而在消息代理环境中失败的事件则由 MQ 消息流处理。当 InterChange Server（ICS）是集成代理程序时，您可以使用 WebSphere InterChange Server 系统管理器（CSM）重新提交失败的事件。

支持 BAPI

IBM WebSphere Business Integration mySAP.com 适配器包括一个工具，即 SAPODA，它生成支持 BAPI 的业务对象定义。SAPODA 解释 BAPI 的接口，将其参数映射至业务对象属性，并添加每个属性的特定于应用程序的信息。

注: 某些 BAPI 没有与 WebSphere 业务对象中的简单属性对应的单一字段参数。连接器要求每个顶级业务对象都具有一个充当键属性的简单属性。因此，当从 BAPI 生成业务对象和业务对象处理程序而没有单一字段参数时，SAPODA 在顶级业务对象中创建一个名为 Dummy_key 的键属性，将它标记为键属性，并添加 dummy_key 作为此属性的特定于应用程序的信息。Dummy_key 为连接器提供一个键属性，以便它可以处理该业务对象。但是，连接器在修改应用程序数据时会忽略 Dummy_key 属性的值。

支持 BAPI 事务

连接器和 SAPODA 支持 SAP BAPI 事务（也称为“逻辑工作单元”）。BAPI 事务由一组 BAPI 组成，这些 BAPI 按顺序执行以便完成整个事务。使用同一 JCo 客户机连接来调用由多个 BAPI 组成的序列。

为了支持 BAPI 事务，SAPODA 生成充当一组子业务对象的包装程序的顶级业务对象，每个子业务对象都表示事务序列中的单个 BAPI 调用。BAPI 事务包装程序对象表示完整事务。每个第二级的子业务对象表示方法的结构参数或表参数。简单属性对应于方法的简单参数。

当成功处理了事务中的所有 BAPI 调用之后，BAPI 业务对象处理程序将返回 SUCCESS（成功）。业务对象处理程序还在发生故障时进行错误处理。如果未能处理事务中的 BAPI 调用，则事务中的后续调用将终止。并且，根据错误代码，BAPI_RETURN 将返回 FAIL 或 APPRESPONSETIMEOUT。

BAPI 接口不对事务提供回滚机制。可以通过下列其中一种方式来获得回滚:

- 终止所有后续 BAPI 调用，从而在进行 COMMIT 之前终止事务。当检测到错误时，连接器将终止事务。如果已调用的 BAPI 本身没有 COMMIT，则不需要执行进一步的步骤。
- 通过调用另一个 BAPI，该 BAPI 可以回滚在 BAPI 具有内在 COMMIT 情况下已提交的工作。

注意，需要由业务流程逻辑来处理回滚。

有关 BAPI 事务业务对象的结构的详细信息，请参阅第 91 页的第 9 章，『为 BAPI 模块开发业务对象』。

支持 BAPI ResultSet

连接器和 SAPODA 对 DB2 Information Integrator（DB2 II）提供了 ResultSet 支持。SAPODA 生成是包装程序业务对象的 ResultSet 对象。此对象包含 BAPI_Query 和 BAPI_Result 两种属性，表示 Query BAPI 和 Result BAPI 对象。有关 BAPI ResultSet 对象的结构的详细信息，请参阅第 93 页的『BAPI ResultSet 的业务对象结构』。

第 8 章 配置 BAPI 模块

- 『BAPI 模块目录和文件』
- 『BAPI 模块配置属性』

本章描述在安装连接器之后如何配置 BAPI 模块。有关安装连接器的更多信息，请参阅第 13 页的第 2 章，『安装连接器』。

BAPI 模块目录和文件

BAPI 模块目录和文件包含在 \connectors\SAP\ 目录中。表 8 列示 BAPI 模块使用的目录和文件。

表 8. BAPI 模块目录和文件

目录 / 文件名	描述
\bapi\client	包含连接器的运行时文件的目录。可以将您创建的任何定制业务对象处理程序复制到此目录中。
CWSAP.jar	连接器类文件

BAPI 模块配置属性

您必须配置 BAPI 模块，然后它才能开始运行。要配置 BAPI 模块，设置标准的和特定于连接器的连接器配置属性。有关配置连接器配置属性的更多信息，请参阅第 21 页的第 3 章，『配置连接器』和第 261 页的附录 D，『连接器的标准配置属性』。

第 9 章 为 BAPI 模块开发业务对象

本章描述由 BAPI 模块处理的三种业务对象: 单个 BAPI 调用的业务对象、BAPI 事务的业务对象和 BAPI ResultSet 对象。

本章还讨论业务对象生成实用程序 SAPODA 如何生成定义。本章假定您了解连接器如何处理业务对象。有关 BAPI 模块中业务对象处理的更多信息，请参阅第 85 页的第 7 章，『BAPI 模块概述』。

注: 本章描述支持 BAPI 的业务对象；但是，BAPI 模块可以用来支持任何 RFC 支持功能。

背景信息

开发 BAPI 模块的业务对象要求在业务对象级别配置特定于应用程序的信息。单个业务对象处理程序支持所有 BAPI。SAPODA 在为集成代理程序生成业务对象定义时使用 SAP 应用程序的本机定义作为模板。

SAP 支持许多方法，可以将这些方法映射至连接器支持的标准查询描述（创建、更新、删除和检索）。您可以开发业务对象和业务对象处理程序以支持由 BAPI 使用的任何方法。

SAPODA 有助于开发业务对象。SAPODA 在为 IBM WebSphere Business Integration mySAP.com 适配器生成业务对象定义时使用 SAP 应用程序的本机定义作为模板。

重要提示: SAPODA 必须能够访问 SAP 系统中的 BAPI 才能检索 BAPI 接口。

业务对象命名约定

BAPI 接口由简单参数、结构参数、返回参数和表参数组成，其中:

- 结构参数、简单参数和返回（导入）参数将传递至 BAPI。
- 结构参数、简单参数和返回（导出）参数将从 BAPI 中传递出来。
- 将向任一方向传递表（导出／导入）参数。

某些 BAPI 可能并不具有所有类型的参数。例如，BAPI 可能仅具有导入参数和表参数。

SAPODA 自动将 BAPI 结构和表参数映射至子业务对象，而将 BAPI 简单参数映射至 SAP 的 WebSphere 业务对象上相应的简单属性，如表 9 中所述。

表 9. 命名约定: SAP 的 WebSphere 业务对象

业务对象	BAPI 接口
顶级业务对象	BOprefix_BAPIname **注意**: 本章中的例子使用 SAP_ 或 sap_ 作为业务对象前缀。您在创建业务对象定义时，可以指定自己的有意义前缀。
属性	FieldDescription
子业务对象	BOprefix_BAPIParameterName

SAPODA 保证业务对象定义中的所有属性名都是唯一的。如果 BAPI 有多个参数具有相同字段描述，则 SAPODA 将计数器作为后缀添加至生成的属性名。

在生成业务对象定义之后，您可以在任何时间修改属性名。但是，当修改属性名时，确保不要修改特定于应用程序的信息。连接器使用此信息来标识该属性所对应的 BAPI 参数。有关特定于应用程序的信息的详情，请参阅第 97 页的『属性的 AppSpecificInfo』。

业务对象结构

连接器支持三种 BAPI 业务对象: 单个 BAPI 调用的业务对象、BAPI 事务的业务对象和 BAPI ResultSet 对象。

单个 BAPI 调用的业务对象结构

单个 BAPI 调用的业务对象反映 BAPI 接口上的方法。业务对象使用 BAPI 业务对象处理程序来将每个业务对象属性映射至 BAPI 参数。连接器、每个业务对象和 BAPI 业务对象处理程序都是元数据驱动的。对于每个业务对象和业务对象处理程序的元数据中提供的特定于应用程序的信息，它允许您为新业务对象及业务对象处理程序添加连接器支持而不必修改连接器代码。实际上是:

- 连接器使用特定于应用程序的顶级业务对象查询描述信息来实例化适当的业务对象处理程序。
- 业务对象处理程序使用查询描述 ASI 来确定要调用的正确 BAPI。

业务对象处理程序支持业务对象之间的单基数和多基数关系。

基于 BAPI 的业务对象不能包含多于两个级别的层次结构。因此，所有 BAPI 简单参数都对应于顶级业务对象的属性，并且 BAPI 结构参数和表参数对应于子业务对象。

表 10. BAPI 和 SAP 的业务对象之间的对应项

BAPI 接口参数	SAP 的 WebSphere 业务对象
简单字段	顶级业务对象的属性
结构	单基数子业务对象
表	多基数子业务对象

注: 导入参数和导出参数可以是简单字段或结构参数。

图 48 举例说明了业务对象和 BAPI 之间的关联。该图举例说明了 sap_bapi_salesorder_createfromdat2 业务对象的片段，该业务对象对应于 BAPI_SALESORDER_CREATEFROMDAT2 BAPI。

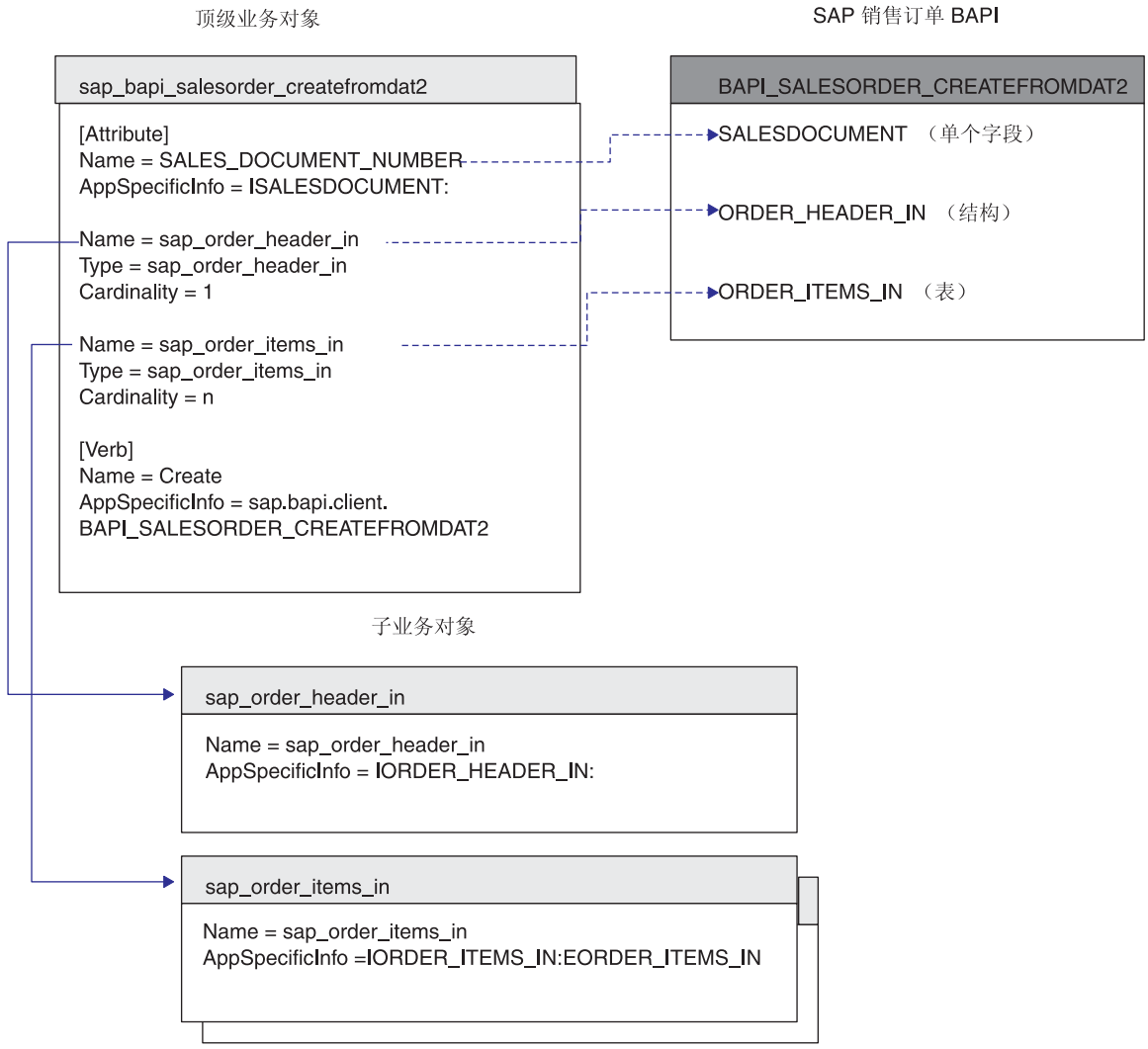


图 48. 在业务对象和 BAPI 之间的映射

BAPI 事务的业务对象结构

表示 BAPI 事务的业务对象是一个将多个 BAPI 对象作为子代的包装程序对象。包装程序 BAPI 事务对象中的每个子代 BAPI 对象都表示单个 BAPI 调用的参数。注意，SAPODA 始终会对 BAPI 事务对象名添加后缀 _txn。例如，sap_BAPI_salesorder_txn。

BAPI ResultSet 的业务对象结构

表示 BAPI ResultSet 的业务对象是一个用于对 DB2 Information Integrator 提供 ResultSet 支持的包装程序对象。当执行 RetrieveByContent 查询描述操作时，连接器会返回符合特定选择条件的多个结果业务对象。ResultSet 包装程序对象与这一组业务对象相对应。

包装程序业务对象包含对象类型的两种属性: BAPI_Query 和 BAPI_Result。BAPI_Query 属性表示 ASI 参数上的 “获取列表 BAPI”。例如，bapi=BAPI_CUSTOMER_GETLIST。BAPI_Result 属性表示 ASI 参数上的 “获取详细信息 BAPI”。注意，SAPODA 始终会对 BAPI_Query 和 BAPI_Result 的子业务对象的名称添加后缀 _rs。例如，sap_BAPI_addressdata_rs。

有关 BAPI ResultSet 对象的属性级别 ASI 的详细信息，请参阅第 99 页的『BAPI ResultSet 的属性级别 ASI』。

有关 ResultSet 处理的更多信息，请参阅 DB2 Information Integrator 文档。

受支持的查询描述

BAPI 模块支持由 WebSphere Business Integration 系统使用的标准查询描述（创建、更新、删除和检索）。对于每个受支持的查询描述，BAPI 可以具有一个相关的方法。查询描述的含义是由 BAPI 方法给定的。换句话说，BAPI 调用具有一些固有功能，跟与它相关联的查询描述无关。大多数 BAPI 支持下列其中一项操作: 创建、检索、更新和删除。

业务对象属性特性

根据属性是表示简单值还是一个子业务对象或一组子业务对象，顶级业务对象属性的特性有所不同。

- 表 11 列示并描述顶级业务对象的简单属性的特性。
- 表 12 列示并描述表示一个子业务对象或一组子业务对象的属性。

SAPODA 生成属性特性，如每个表中所述。

表 11. 简单属性特性: 顶级业务对象

属性名	描述
Name	源自 BAPI 参数的描述。SAPODA 用下划线替换特殊字符（如句点、斜杠和空格）。
Type	指定数据的类型。SAPODA 将其值设置为 String。
MaxLength	指定 BAPI 参数的字段长度。
IsKey	指定属性是否是键。业务对象的第一个简单属性缺省为键属性。对于单个 BAPI 对象，SAPODA 插入 Dummy_key 属性作为第一个属性，将它标记为键属性，然后设置适当的值。对于 BAPI 事务和 ResultSet，SAPODA 使用第一个属性作为键。有关更多信息，请参阅第 88 页的『支持 BAPI』。
IsForeignKey	SAPODA 将其值设置为 false。
IsRequired	指定属性是否必须包含值。SAPODA 将其值设置为 false。
AppSpecificInfo	包含对应于相关属性的 BAPI 参数的名称。格式为: I*ABAPFieldName*:E*ABAPFieldName*。有关特定于应用程序的信息的详情，请参阅第 96 页的『特定于应用程序的业务对象信息』。
DefaultValue	指定在没有运行时值时要赋予此属性的值。SAPODA 没有为此属性设置值。

表 12 列示并描述表示一个子业务对象或一组子业务对象的属性。SAPODA 生成下面描述的属性。

表 12. 表示一个或多个子代的属性的特性

属性名	描述
Name	此值是结构参数或表参数的名称。格式为: BOprefix_BAPIParameterName。业务对象名称中存在的所有特殊字符都将被替换为下划线字符 _。
Type	其值是子业务对象的类型; 即类型为 BOprefix_BAPIParameterName。
ContainedObjectVersion	SAPODA 将其值设置为 3.0.0。
Relationship	SAPODA 将其值设置为 containment。
IsKey	对于 BAPI 事务或 ResultSet, SAPODA 将第一个属性的值设置为 true, 而将所有其它属性的值设置为 false。
IsForeignKey	SAPODA 将其值设置为 false。
IsRequired	指定属性是否必须包含值。SAPODA 将其值设置为 false。
AppSpecificInfo	包含对应于相关属性的 BAPI 参数的名称。格式为: I*BAPIParameterName*:E*BAPIParameterName*。有关特定于应用程序的信息的详情, 请参阅第 97 页的『属性的 AppSpecificInfo』。
Cardinality	BAPI 结构参数具有单基数（1）, 而 BAPI 表参数具有多基数（n）。

重要提示: 简单属性可以具有两个特殊值: CxIgnore 和 CxBlank。当将业务对象作为服务调用请求发送至 BAPI 模块, 并且业务对象具有设置为 CxIgnore 或 CxBlank 的简单属性时, 这些属性对于 BAPI 模块好像是不可视的。但是, SAP 应用程序将此类属性初始化为其 ABAP 数据类型。BAPI 模块将所有返回的空白值转换为 CxIgnore。

初始化属性值

SAP 中的每个字段都具有初始值。当连接器接收到服务调用请求时, 业务对象处理程序使用表 13 中列示的值填充大多数 BAPI 接口参数。字符数据类型是一个例外。业务对象处理程序将业务对象属性中的 CxIgnore 转换为 SAP 字段中的空格。如果您想要将任何其它值转换为 CxIgnore, 则创建业务对象的组件必须执行该转换。例如, 当 ICS 是集成代理程序时, 修改映射以处理此转换。

表 13 提供由业务对象处理程序设置的初始值。

表 13. SAP 中的初始字段值

数据类型	描述	由业务对象处理程序设置的初始值
C	字符	空格
N	数字字符串	*000...*
D	日期（YYYYMMDD）	00000000
T	时间（HHMMSS）	000000
X	字节（十六进制）	X00
I	整数	0
P	压缩数字	0
F	浮点数字	0.0

特定于应用程序的业务对象信息

业务对象定义中特定于应用程序的信息（ASI）对 BAPI 模块提供关于如何处理业务对象的应用程序相关指示信息。这些指示信息是在下列级别指定的:

- 单个 BAPI 调用对象、BAPI 事务对象和 ResultSet 对象的业务对象级别
- 单个 BAPI 调用对象、BAPI 事务对象和定制业务对象处理程序（CBOH）的查询描述级别
- 下列属性的属性级别:
  - 简单属性
  - 表示子对象的属性
  - 表示一组子对象（ResultSet 对象）的属性

业务对象级别 ASI

应当为每种类型的对象设置业务对象级别 ASI，如表 14 中所述。

表 14. 业务对象级别 ASI

对象类型	特定于应用程序的信息
表示单个 BAPI 调用的对象	`type=bapi`
表示 BAPI 事务的对象	`type=bapi_txn`
表示 BAPI ResultSet 的对象	`type=bapi_resultset`

查询描述级别 ASI

应当为每种类型的对象设置查询描述级别 ASI，如表 15 中所述。注意，BAPI ResultSet 业务对象没有查询描述 ASI。

表 15. 查询描述级别 ASI

对象类型	查询描述 ASI
表示单个 BAPI 调用的对象	`verb ASI=`*`NameOfBAPI`*
表示 BAPI 事务的对象	`verb ASI=`*`NameOfBAPI1;NameOfBAPI2;NameOfBAPI3`*
表示定制业务对象处理程序的对象	`CBOH=`*`bapi.client.customBOHandlerName`*

表示单个 BAPI 调用的对象的向后兼容性

注意，连接器支持较早发行版中的业务对象的查询描述 ASI 格式，在这些发行版中，AppSpecific 属性的值将捕获特定于 BAPI 的业务对象处理程序的类名（`verb ASI=bapi.client.`*`BOHandlerName`*，其中 `bapi.client` 是特定于 BAPI 的业务对象处理程序名的 WebSphere Business Integration 限定符，*`BOHandlerName`* 是类的名称）。您必须在业务对象处理程序名称前面包括值 `client`，来标识业务对象处理程序充当客户机。注意，虽然连接器支持这些较早的格式，但是 SAPODA 不会自动生成它们，因此，必须在查询描述 ASI 中按名称显式指定它们。

例如，如果您要支持较早发行版中的 `SALES_ORDER_CREATEFROMDAT2` BAPI，则特定于应用程序的信息为如下所示:

```
AppSpecificInfo = bapi.client.sales_order_createfrom dat2
```

表示定制业务对象处理程序的对象的查询描述 ASI

对于定制业务对象处理程序，应显式设置查询描述 ASI（因为它不是由 SAPODA 生成的）并且用包名来完全限定它，其中 *bapi.client* 表示包名。

属性的 AppSpecificInfo

连接器使用一个属性的特定于应用程序信息的值来确定要使用哪些导入参数、导出参数和表参数。此属性的值包含前缀 I（表示导入参数）或 E（表示导出参数）。该前缀指示属性值是用来将数据传递至 SAP 应用程序还是从 SAP 应用程序中传递出来。

因为结构参数可以是导入或导出，所以它们在参数值前面使用 I 或 E。因为表参数可以将数据传递至 BAPI 并从 BAPI 返回数据，所以它们可以同时具有 I 和 E 参数值。

重要提示: 当使用 I 和 E 指定参数值时，应始终使用冒号（:）分隔符。如果仅指定导入值，则该值后面必须有冒号。如果仅指定导出值，则该值前面必须有冒号。如果同时指定两个值，则冒号在导入值的后面并在导出值的前面。

图 49 举例说明了业务对象和名为 BAPI_EXAMPLE 的示例 EAPI 之间的对应项。在此示例中，简单属性（Attribute_1、Attribute_2 和 Attribute_3）仅指定导入参数或导出参数。表示子业务对象（Child_1）的属性对应于导出结构参数。表示一组子业务对象（Child_2）的属性对应于表参数。

每个子业务对象都具有一个与相应结构或表的字段对应的简单属性（分别为 Attribute_11 和 aAttribute_14）。您可以通过查看 BAPI 的详细信息来找到这些字段。

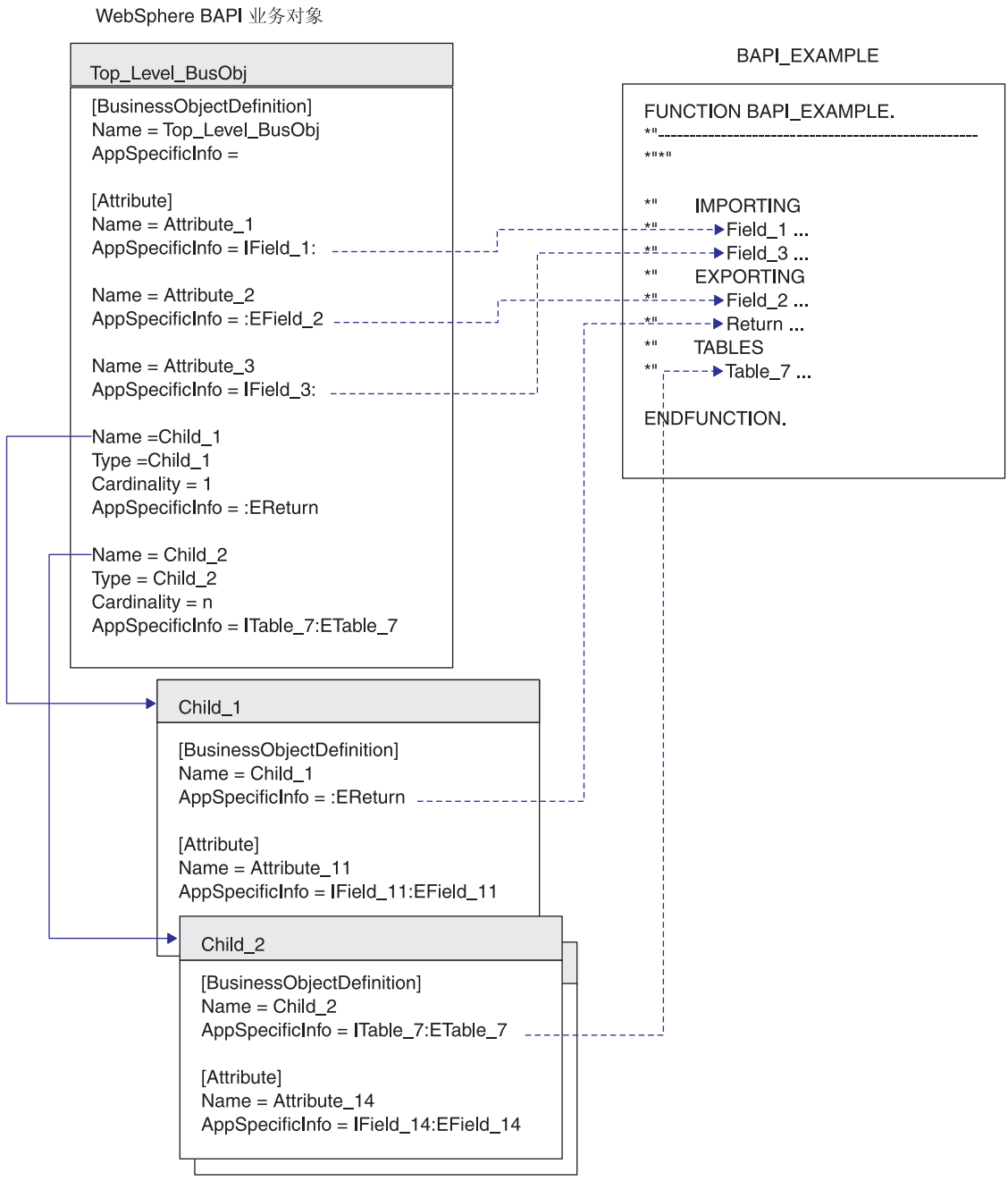


图 49. 业务对象和示例 BAPI 之间的对应项

表 16 标识特定种类属性的 ASI 的格式。

表 16. 特定种类的属性的 ASI 格式

AppSpecificInfo 格式	**属性类型**
I*ParameterName*:E*ParameterName*	简单
I*TableName*:E*TableName*	表示映射至表参数的子业务对象
I*StructureName*:E*StructureName*	表示映射至结构参数的子业务对象
I*FieldName*:E*FieldName*	表示映射至表参数或结构参数中字段的子业务对象的属性

SAPODA 自动为业务对象定义生成特定于应用程序的适当信息。建议不要更改生成的特定于应用程序的信息的参数名。

BAPI ResultSet 的属性级别 ASI

如第 93 页的『BAPI ResultSet 的业务对象结构』中说明的那样，BAPI ResultSet 是一个包含对象类型的以下两个属性的包装程序对象：BAPI_Query 和 BAPI_Result。BAPI_Query 表示“获取列表 BAPI”，而 BAPI_Result 表示“获取详细信息 BAPI”。表 17 中描述了这两个属性的属性级别 ASI。

表 17. ResultSet 的属性级别 ASI

ResultSet 属性	特定于应用程序的信息
BAPI_Query	bapi=*name_Of_GetListBAPI*
BAPI_Result	key=*key_field*; bapi=*name_Of_GetDetailBAPI*

BAPI_Result 对象的 Key 属性是使用属性级别 ASI key=*name_Of_KeyAttribute* 来标识的。业务对象处理程序使用此信息来进行性能优化。

BAPI Result 对象的 Foreign Key 属性将捕获 BAPI_Result 与 BAPI_Query 对象之间的关系。它是使用以下属性级别 ASI 来标识的：FK=BAPI_Query:*BAPI_Query 的包含“键属性:键属性名称”的子对象名称*。

例如，FK=Query:sap_addressdata:Customer_number。BAPI 业务对象处理程序使用此信息来将 BAPI 查询对象中的键值设置到 BAPI Result 对象键属性中，然后检索“详细信息结果”以找到每个键。

使用生成的业务对象定义

使用 SAPODA 来为您要支持的每个 RFC 支持功能生成业务对象定义。您可以使用生成的对象而不必进行任何修改。但是，可以手工编辑这些对象以改进此功能。

在生成对象之后，您必须将业务对象定义添加至 WebSphere Business Integration 系统的运行时环境。使用业务对象设计器将业务对象定义复制到资源库。或者，如果 InterChange Server（ICS）是集成代理程序，则您可以使用 repos_copy 命令来将该定义复制到资源库。

重要提示: 您可以修改生成的业务对象的名称及其子业务对象的名称。为此，您必须作为文本文件编辑该定义，而不是在业务对象设计器进行编辑。如果您更改业务对象的名称，则确保也修改所有引用到更改的名称的地方。

技巧和窍门

本节描述用于开发业务对象的下列技巧和窍门:

- 第 100 页的『多个业务对象包含相同的返回业务对象』
- 第 100 页的『生成的业务对象定义包含不必要的属性和子业务对象』
- 第 100 页的『生成的业务对象名太长或不符合命名约定』
- 第 101 页的『为表参数生成的 AppSpecificInfo 指定不是必需的参数』

多个业务对象包含相同的返回业务对象

大多数 BAPI 将相同名称用于返回对象。当 SAPODA 生成业务对象定义时，它创建一个子业务对象来表示此返回对象。如果多个业务对象定义包含同一个指定的子业务对象，则您只需一次将该子业务对象添加至资源库，或者可以将一个定义文件复制到资源库目录。

要使多个业务对象能够包含返回业务对象，您必须修改返回业务对象的名称，使其对于每个业务对象都是唯一的。

要重命名返回业务对象，修改包含它的每个业务对象定义的定义。子业务对象的定义与其父代的定义包含在相同的定义文件中。

要重命名子代，执行以下操作:

1. 在文本编辑器中打开顶级业务对象的定义文件。
2. 找到 `BOprefix_return` 子业务对象的定义。
3. 将该子代的名称更改为唯一的。例如，将一个数字追加至文本（`sap_return_2`）。
4. 更改定义中的所有引用以引用新命名的子代。例如，更改每个表示子业务对象的属性的 Type 属性值。
5. 保存更改后的定义文件。
6. 使用系统管理器（CSM）将新命名的子业务对象装入资源库中。

   注: 或者，如果 InterChange Server 是集成代理程序，则您可以使用 `repos_copy` 命令将该定义装入资源库。

生成的业务对象定义包含不必要的属性和子业务对象

SAPODA 解释所有 BAPI 接口参数，并且对于每个参数，它都会创建一个相应的子业务对象属性或子业务对象。要提高业务对象处理的性能，从业务对象定义除去所有不需要的属性和业务对象。

注: SAPODA 有助于在生成定义之前以图形方式除去所有可选属性和子业务对象。有关更多信息，请参阅第 52 页的『提供其它信息』。

要提高业务对象处理的性能，您还可以从特定于应用程序的信息除去所有不需要的导入和导出表参数值。

在生成定义之后，如果您需要进行其它更改，则可以使用业务对象设计器来手工编辑业务对象定义。但是，请注意仅除去以后绝对不会使用的属性。

生成的业务对象名太长或不符合命名约定

SAPODA 使用 BAPI 功能模块的名称来生成业务对象定义的名称。您可以使用文本编辑器来修改业务对象的名称。

重要提示: 如果您更改该名称，则确保您也修改该名称的所有引用。但是，不要修改生成的特定于应用程序的信息的参数名。

要更改生成的业务对象的名称:

1. 将定义保存至文件。
2. 使用文本编辑器来缩短或更改该名称。

3. 使用系统管理器（CSM）将新命名的子业务对象装入资源库中。

   注: 或者，如果 InterChange Server 是集成代理程序，则可以使用 `repos_copy` 命令将该定义装入资源库中。

为表参数生成的 AppSpecificInfo 指定不是必需的参数

表参数可以同时是导入参数和导出参数。如果您不要求导入或导出表参数的值，则您可以从特定于应用程序的信息中除去该参数。

例如，对于创建操作，如果您在创建操作完成之后不需要从 SAP 应用程序返回表数据，则您可以除去导出参数值（如 `E` *table name*）。

对于检索操作，您不需要指定任何导入表参数。因此，您可以除去导入参数值（如 `I` *table name*）。

注: 您必须从父代中某一表示子代的属性 `AppSpecificInfo` 中以及业务对象级别为子业务对象的 `AppSpecificInfo` 中除去不需要的值。不要除去冒号（`:`）。

例如，要除去第 98 页的图 49 中的 `ETable_7` 导出参数，您应执行以下操作:

1. 在 Top_Level_BusObj 业务对象的 Child_2 属性中，将该属性的 AppSpecificInfo 值更改为:

   ```
   ITable_7:
   ```

2. 在业务对象级别为 Child_2 业务对象的 AppSpecificInfo 中，将其值更改为:

   ```
   ITable_7:
   ```

3. 以 Attribute_14 为例，在子业务对象的每个属性的 AppSpecificInfo 中将其值更改为:

   ```
   IField_14:
   ```

使用定制业务对象处理程序

建议您将随连接器一起交付的 BAPI 业务对象处理程序用作模板（有关详细信息，请参阅第 86 页的『业务对象处理』），而不用编写您自己的业务对象处理程序。使用定制业务对象处理程序的原因如下:

- 实现定制错误处理。
- 支持在较早发行版中创建的业务对象，而不必在本发行版的 SAPODA 中重新生成它们。有关业务对象的向后兼容性的详细信息，请参阅第 96 页的『表示单个 BAPI 调用的对象的向后兼容性』。

创建定制业务对象处理程序

如果您决定创建定制业务对象处理程序，请将以下示例作为如何编写定制业务对象处理程序的指南。注意，SAPODA 不会为您生成代码。

```
package bapi.client ;


import AppSide_Connector.JavaConnectorUtil;
import CxCommon.BusinessObjectInterface;
import CxCommon.CxMsgFormat;
import CxCommon.CxStatusConstants;
import CxCommon.ReturnStatusDescriptor;
import com.crossworlds.connectors.sap.codegen.BapiBOHandlerBase;
import com.crossworlds.connectors.sap.codegen.exception.CwBoHandlerAppResponseTimeout;
import com.crossworlds.connectors.sap.codegen.exception.CwBoHandlerProcessingFailed;
import com.crossworlds.connectors.sap.visionframework.VisionBOHandlerInterface;
import com.crossworlds.connectors.sap.visionframework.VisionConnectorAgent;
import com.sap.mw.jco.IRepository;
```

```
import com.sap.mw.jco.JCO;


public class Bapi_customer_getlist extends BapiBOHandlerBase implements VisionBOHandlerInterface {

  private static VisionConnectorAgent vca = VisionConnectorAgent.getCWSapConnManager();
  private IRepository repository = getRepository();
  private JCO.Function bapicommit = new JCO.Function(repository.
     getFunctionTemplate("BAPI_TRANSACTION_COMMIT"));
  private int checkBapiReturn = CxStatusConstants.SUCCEED;

  public Bapi_customer_getlist()
  {
  }


  public int doVerbForVision(BusinessObjectInterface theObj, ReturnStatusDescriptor rtn)
  {
     JCO.Function function = new JCO.Function(repository.getFunctionTemplate("BAPI_CUSTOMER_GETLIST"));

     //Default processing to failure
     int    mReturnCode = CxStatusConstants.FAIL;
     JCO.Client theClient = null;

       try
     {
     traceMessage(JavaConnectorUtil.LEVEL1, 27025, CxMsgFormat.XRD_INFO, theObj.getName(), theObj.getVerb(),
       null, null);

     // get defaults and check for required fields for Create and Update only
     if ((theObj.getVerb().equalsIgnoreCase("Create")) || (theObj.getVerb().equalsIgnoreCase("Update")))
     {
       traceMessage(JavaConnectorUtil.LEVEL4, 27021, CxMsgFormat.XRD_INFO, theObj.getName(),
        null, null, null);
       JavaConnectorUtil.initAndValidateAttributes(theObj);
     }

     // get a new connection to Sap
     theClient = this.getClient();

     //   populate Rfc interface parameters
     mReturnCode = doBusObjtoRfcData(theObj, function, function.getImportParameterList(),"I",false);
     if (mReturnCode != CxStatusConstants.SUCCEED)
     {
     if (theClient != null) vca.releaseClient(theClient);
       return CxStatusConstants.FAIL;
     }

     // Execute Rfc Call
     this.callBapi(theClient,function);

  try {
     // After successful RfcCall: check Structure/Table RETURN for Bapi Return Codes E or A that
     // indicate failure
       this.checkBapiRc(function);

     // After successful Bapi Call: call BAPI_TRANSACTIOM_COMMIT
       this.callBapiCommit(theClient);

     } catch (CwBoHandlerProcessingFailed cwpf) {
        rtn.setErrorString(cwpf.getMessage());
       rtn.setStatus(CxStatusConstants.FAIL);
       checkBapiReturn = CxStatusConstants.FAIL;
     }
     // Now create CW Business Object
     mReturnCode = doRfcDatatoBusObj(theObj, function , function.getExportParameterList(), "E");
     if (mReturnCode != CxStatusConstants.SUCCEED || checkBapiReturn != CxStatusConstants.SUCCEED)
     {
     if (theClient != null) vca.releaseClient(theClient);
       return CxStatusConstants.FAIL;
     }

     // Clean up
     if (theClient != null) vca.releaseClient(theClient);

     // Finally, return success to ICS
     traceMessage(JavaConnectorUtil.LEVEL1, 27034, CxMsgFormat.XRD_INFO, theObj.getName(),
        null, null, null);
     return CxStatusConstants.VALCHANGE;
     }//end of try

     catch (CxCommon.Exceptions.SetDefaultFailedException sdfe)
     {
     if (theClient != null) vca.releaseClient(theClient);
       String msg = logMessage(20059, CxMsgFormat.XRD_INFO, theObj.getName(), null, null, null);
       rtn.setErrorString(msg);
       rtn.setStatus(CxStatusConstants.FAIL);
       return CxStatusConstants.FAIL;
```

```
    }
    catch (CxCommon.Exceptions.BusObjSpecNameNotFoundException c)
    {
    if (theClient != null) vca.releaseClient(theClient);
      String msg = logMessage(20059, CxMsgFormat.XRD_INFO, theObj.getName(), null, null, null);
      rtn.setErrorString(msg);
      rtn.setStatus(CxStatusConstants.FAIL);
      return CxStatusConstants.FAIL;
    }
    catch (CwBoHandlerAppResponseTimeout art)
    {
    if (theClient != null) vca.releaseClient(theClient);
      rtn.setErrorString(art.getMessage());
      rtn.setStatus(CxStatusConstants.APPRESPONSETIMEOUT);
      return CxStatusConstants.APPRESPONSETIMEOUT;
    }
    catch (CwBoHandlerProcessingFailed cwpf)
    {
    if (theClient != null) vca.releaseClient(theClient);
       rtn.setErrorString(cwpf.getMessage());
      rtn.setStatus(CxStatusConstants.FAIL);
      return CxStatusConstants.FAIL;
    }
  catch (Exception e)
    {
    if (theClient != null) vca.releaseClient(theClient);
       rtn.setErrorString(e.getMessage());
      rtn.setStatus(CxStatusConstants.FAIL);
      return CxStatusConstants.FAIL;
    }
    catch ( OutOfMemoryError merr ){                // CR 30185
      String msg = logMessage(23060, JavaConnectorUtil.XRD_ERROR,
        "doVerbForVision()", merr.toString(), null, null);
      rtn.setErrorString(msg);
      rtn.setStatus(CxStatusConstants.APPRESPONSETIMEOUT);
      return CxStatusConstants.APPRESPONSETIMEOUT;
    }
    catch ( StackOverflowError serr ){                // CR 30185
      String msg = logMessage(23060, JavaConnectorUtil.XRD_ERROR,
         "doVerbForVision()", serr.toString(), null, null);
      rtn.setErrorString(msg);
      rtn.setStatus(CxStatusConstants.APPRESPONSETIMEOUT);
      return CxStatusConstants.APPRESPONSETIMEOUT;
    }
    catch (Throwable ex) {
      // 23046 EXception raised in: {1} : ErrorMessage: {2}.
      String msg = logMessage(23046, JavaConnectorUtil.XRD_ERROR,
        "doVerbForVision()",ex.toString(), null, null);
      return CxStatusConstants.FAIL;
    }
  }// end of doVerbforVision

public int callBapiCommit(JCO.Client theClient) throws
    CwBoHandlerProcessingFailed, CwBoHandlerAppResponseTimeout
{

  int mStatus = CxStatusConstants.FAIL;
  try {
  traceMessage(JavaConnectorUtil.LEVEL3, 27032, CxMsgFormat.XRD_INFO, null, null, null, null);
       try
  {
     theClient.execute(bapicommit);
  }
     catch (JCO.Exception e)
  {
     String msg = logMessage(27019, CxMsgFormat.XRD_ERROR, null, null,
       null, null);
       throw new CwBoHandlerProcessingFailed(msg);
  }
  //  Read Return
  readBapiRc(bapicommit.getExportParameterList().getStructure("RETURN"));
  }// end of try
  catch (Exception e)
  {
     String msg = logMessage(27019, CxMsgFormat.XRD_ERROR, null, null,
       null, null);
     throw new CwBoHandlerProcessingFailed(e.getMessage());
  }

  // Return Success
  return CxStatusConstants.SUCCEED;
}


public void checkBapiRc(JCO.Function function) throws CwBoHandlerProcessingFailed
{

       try
```

```
  {
     JCO.ParameterList p = function.getTableParameterList();
       this.readBapiRc(p.getTable("RETURN"));
     }
     catch (CwBoHandlerProcessingFailed cw)
     {
       throw cw;
     }
  catch (Exception e)
     {
       try
       {
       JCO.ParameterList p = function.getExportParameterList();
       if (p != null)
       this.readBapiRc(p.getStructure("RETURN"));
  else
       {
       String msg = logMessage(27045, CxMsgFormat.XRD_INFO, null,null, null, null);
       throw new CwBoHandlerProcessingFailed(msg);
       }
     }//end try
     catch (JCO.Exception o)
       {
       String msg = logMessage(27045, CxMsgFormat.XRD_INFO, null,null, null, null);
       throw new CwBoHandlerProcessingFailed(msg);
       }
  }

}// end of checkBapiRc


public void readBapiRc(JCO.Structure Return) throws CwBoHandlerProcessingFailed
{
  String type = null;
  String no = null;
  String message = null;
  traceMessage(JavaConnectorUtil.LEVEL4, 27033, CxMsgFormat.XRD_INFO, null, null, null, null);
  if (Return.getString("TYPE") != null)
  {
     // Depending on RETURN ddic structure, number field is either "NUMBER" or "CODE".
       try
     {
       no = Return.getString("NUMBER");
     }
     catch (JCO.Exception o)
     {
       no = Return.getString("CODE");
     }
     message = Return.getString("MESSAGE");
     type = Return.getString("TYPE");
     if ((type.equalsIgnoreCase("A")) || (type.equalsIgnoreCase("E")))
     {
       String msg = logMessage(27015, CxMsgFormat.XRD_ERROR, type, no,
        message, null);
       throw new CwBoHandlerProcessingFailed(msg);
     }
  else
     {
     traceMessage(JavaConnectorUtil.LEVEL1, 27016, CxMsgFormat.XRD_INFO,
          type, no, message, null);
     return;
     }
  }
  // If structure Return is empty, it will be assumed that processing was
     successful
     traceMessage(JavaConnectorUtil.LEVEL1, 27036, CxMsgFormat.XRD_INFO, type, no, message, null);
}



public void readBapiRc(JCO.Table Return) throws CwBoHandlerProcessingFailed
{
  String type = null;
  String no = null;
  String message = null;
  String msg = null;
  String errorMsg = "";
  int mStatus = CxStatusConstants.SUCCEED;

  traceMessage(JavaConnectorUtil.LEVEL4, 27033, CxMsgFormat.XRD_INFO, null, null, null, null);
  for (int i=0; i<Return.getNumRows(); i++)
  {
       try
  {
  Return.setRow(i);
  if (Return.getString("TYPE") != null)
  {
```

```
    type = Return.getString("TYPE");
    // Depending on RETURN ddic structure, number field is either "NUMBER" or "CODE".
      try
    {
      no = Return.getString("NUMBER");
    }
    catch (JCO.Exception e)
    {
      no = Return.getString("CODE");
    }
    message = Return.getString("MESSAGE");
    }//end if
    }//end try
    catch (Exception o)
    {
  // Could not interprete Return structure ==> Failing event
      msg = logMessage(27043, CxMsgFormat.XRD_ERROR, type, no,
       message, null);
      throw new CwBoHandlerProcessingFailed(msg);
    }
    if  (type != null)
    {
    if ((type.equalsIgnoreCase("A")) || (type.equalsIgnoreCase("E")))
    {
      msg = logMessage(27015, CxMsgFormat.XRD_ERROR, type, no,
       message, null);
      mStatus = CxStatusConstants.FAIL;
    }//end if
  else
    {
    traceMessage(JavaConnectorUtil.LEVEL1, 27044, CxMsgFormat.XRD_INFO,
        type, no, message, null);
    }
    }//end if type != null
  }//end of for
  if (mStatus == CxStatusConstants.FAIL)
    {
      logMessage(27046, CxMsgFormat.XRD_ERROR, msg, null, null, null);
      throw new CwBoHandlerProcessingFailed(msg);
    }
  // If Return is empty, it will be assumed that processing was successful
  if (Return.getNumRows() <= 0)
    traceMessage(JavaConnectorUtil.LEVEL1, 27036, CxMsgFormat.XRD_INFO, type, no, message, null);
  else
  traceMessage(JavaConnectorUtil.LEVEL1, 27011, CxMsgFormat.XRD_INFO, null, null, null, null);
}//end of ReadBapiRc


};
```

以下示例举例说明了一个用于在 Windows 平台上编译定制 BOHandler 的脚本。注意，要运行此脚本，必须在您的机器上安装 JDK。

```
REM @echo off REM

REM init environment
call "%CROSSWORLDS%"\bin\CWODAEnv.bat
setlocal

REM set classpaths
set WBIA=%CROSSWORLDS%"\lib\WBIA\4.2.0\WBIA.jar
set CWLIB=%CROSSWORLDS%"\lib\CrossWorlds.jar
set AGENT="%CROSSWORLDS%"\ODA\SAP\SAPODA.jar
set JCO_JAR="%CROSSWORLDS%"\ODA\SAP\sapjco.jar

set JCLASSES=%AGENT%;%JCO_JAR%;%CWLIB%;%WBIA%
echo classpath = %JCLASSES%

REM compile the BAPI BOHandler passed as argument
javac -verbose -classpath %JCLASSES% %1

endlocal
pause
```

注: 连接器为使用生成的（非定制的）业务对象处理程序的现有安装提供了完全的向后兼容性。

第 3 部分 ALE 模块

第 10 章 ALE 模块概述

本章描述 mySAP.com 适配器（R/3 V3.x）的 ALE（应用程序链接启用）模块。ALE 是 SAP 的业务框架内集成层的一部分。ALE 模块在两个或多个 SAP 系统或者在 SAP 与外部系统之间启用业务流程集成和异步数据通信。

本章包含以下各节:

ALE 技术的概述

ALE 模块最适合于在本质上是异步的对象，如批处理对象。它使用要求有服务器来侦听事件的推送技术。称为注册和安装的进程将要侦听的内容和期望从何处获得信息通知服务器。注册涉及使用程序标识来对 SAP 网关提供具有侦听器线程的通信点（服务器）。服务器内的功能模块定义通过为从 SAP 推出的数据提供模板来解释该数据。

ALE 模块使用 RFC 服务器模块进行事件处理。ALE 模块将 MQ Series 队列用于事务标识（TID）和 IDoc 管理。连接器在处理从 SAP 到连接器的数据时检查预订，这导致在启动协作前事务保留在 SAP 中。

- 集成代理程序发送 SAP 的 WebSphere Business Integration 适配器业务对象。该业务对象的数据表示对连接器的处理请求。连接器将该业务对象转换为与 SAP 中间文件（IDoc）格式兼容的表格式。连接器使用 ALE 接口的“远程功能调用”（RFC）来将 IDoc 数据传递至 SAP 系统。
- 连接器以 IDoc 表格式从 SAP 接收表示应用程序事件的数据。它在将该数据发送至集成代理程序之前将该数据转换为 SAP 的 WebSphere Business Integration 适配器业务对象。连接器使用 ALE 模块的 RFC 来从 ALE 接口接收数据。

重要提示: 在版本 4.8.2 之前的连接器发行版中，连接器使用协作、业务对象和映射来将事务标识（TID）及其状态存储在资源库中，并使用本地文件系统来存储 IDoc 数据。版本 4.8.2 的连接器使用 MQSeries 队列来替换先前对 TID 和 IDoc 数据的管理。

注: 因为 ALE 模块使用异步通信，所以当需要进行交叉引用时不能使用。

ALE 模块组件

ALE 模块是用 Java 语言编写的，它扩展可视连接器框架。该模块由以下各项组成:

- 连接器框架
- 用于 ALE 的特定于应用程序的连接器组件
- 两个 ALE 业务对象处理程序类（一个用于事件处理，一个用于请求处理）
- SAP RFC 库
- SAP SAPJCo 连接器
- RFC 服务器的特定于应用程序的组件（仅用于事件处理）。

因为事件处理的相似性都支持直接来自 SAP 应用程序的 RFC 调用，所以 ALE 模块使用 RFC 服务器连接器组件。

SAP 以 Java 和 C 语言交付 RFC 库。Java 归档（JAR）文件将作为连接器交付并运行。

图 50 举例说明了 ALE 模块的体系结构。

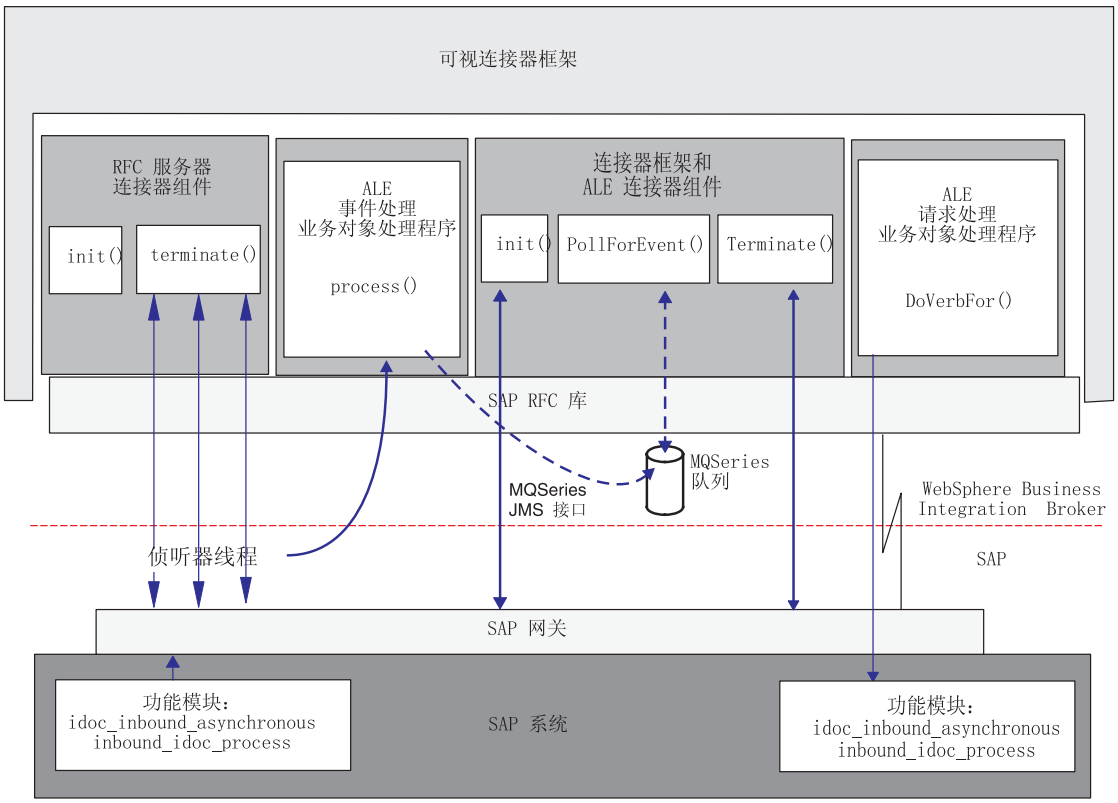


图 50. ALE 模块体系结构

事件处理组件

当处理来自 SAP 的事件时，连接器通过以下方式使用图 50 中举例说明的组件:

- 可视连接器框架启动 RFC 服务器连接器组件，该组件衍生一些侦听器线程。每个侦听器线程都使用 RFC 库和 SAP 网关来注册单个 SAP 应用程序的句柄。
- 侦听器线程处理来自 SAP 应用程序的事件。

  事件是将数据传送至侦听器的 ABAP 功能的执行过程。由 SAP 发送的事件数据可能表示一个或多个这样不同种类的执行过程。

  来自 SAP 的每个事件都被认为是一个事务。连接器使用带有事务标识（TID）的两步骤进程来处理每个事件，以保证将数据从 SAP 传递至连接器仅一次。
- MQSeries 队列持久地存储每个事件的 JMS-MQ 消息。每个 JMS-MQ 消息都会存储标识事件的 TID、TID 的状态、与事件相关的 IDoc 数据和 IDoc 的处理状态。

- 连接器的轮询进程根据存储的事件消息创建 WebSphere 业务对象，并将该业务对象发送至集成代理程序。连接器提供了对产生大型业务对象的大型 IDoc 消息的支持。

  为了获得更高效的处理性能，连接器将大型 IDoc 分成更小的部分，每一部分就是转换为更小的业务对象的 JMS-MQ 消息。这些消息中的每条消息都包含 MultiPartMessage 属性，该属性标识它具有更大消息的适当部分。例如，假定将原始的更大 IDoc 分成 8 条 JMS-MQ 消息，则每一部分的 MultiPartMessage 属性的值分别为 8 部分中的第 1 部分、8 部分中的第 2 部分，依次类推。为了使所有消息部分都互相关联，连接器将除了第一条消息之外的每个消息部分的 CorrelationID 头属性设置为第一部分的 JMSMessageID 属性的值。反过来，第一部分的 CorrelationID 属性始终设置为该属性在与原始大型 IDoc 相对应的第一条 JMS-MQ 消息中的值。有关 JMS-MQ 消息属性的详细信息，请参阅第 121 页的表 20。
- 业务集成系统跟踪未处理的事件，以在集成代理程序或连接器当机时处理这些事件的恢复。当集成代理程序或连接器恢复运行时，连接器自动重新提交这些事件。

请求处理组件

当处理来自集成代理程序的事件时，连接器通过以下方式使用图 50 中举例说明的组件:

- ALE 模块使用 SAP RFC 库和 SAP 网关来打开与 SAP 应用程序的 RFC 连接。
- ALE 请求处理业务对象处理程序处理来自集成代理程序的请求，将它们从业务对象格式转换为基于 SAP IDoc 格式的 IDoc 数据。
- 对于发送至应用程序的每个请求，ALE 模块都会将事务标识（TID）持久地存储在 TID 队列的一条 JMS-MQ 消息中。TID 可以保证传递该请求一次，且仅传递一次。但是，如果集成代理程序发送在事务标识属性中具有相同值的对象，则将再次处理此对象。一旦成功发送某个对象，将期望集成代理程序不会再次发送该对象。
- ALE 模块释放与 SAP 应用程序的连接。

侦听器线程

侦听器线程处理 ALE 模块与 SAP 应用程序之间所有特定于 ALE 的 RFC 调用。当连接器启动时，RFC 服务器模块的 init() 方法创建一个主线程，主线程衍生许多侦听器线程（该数目是可配置的）。每个侦听器线程都打开一个指向 SAP 网关的句柄。

侦听器线程执行以下操作:

- 使用程序标识来向 SAP 网关注册。
- 对 SAP 网关标识特定于 ALE 的 RFC 支持功能（它们支持这些功能）。这些功能为 idoc_inbound_asynchronous 和 inbound_idoc_process。
- 接收来自特定于 ALE 的功能的事件。
- 实例化事件处理 ALE 业务对象处理程序。

线程以同步方式连续侦听事件（这些事件来自它支持的特定于 ALE 的功能）。

事务标识

SAP 使用事务及其相应的标识来构造事件，以保证将每段数据从 SAP 传递一次，且仅传递一次。SAP 将事务标识（TID）随事件数据一起发送。为了集中管理 TID 以进行事件和请求处理，连接器在 MQSeries 队列上将每个 TID 存储为一条 JMS-MQ 消息。当处理事件时，它还将相关的 IDoc 数据存储为消息体。连接器将 TID、TID 状态和 IDoc 的处理状态存储在消息头中。

特定于 ALE 的业务对象处理程序

提供了两个特定于 ALE 的业务对象处理程序，一个用于事件处理，一个用于请求处理。

事件处理业务对象处理程序

侦听器线程实例化事件处理业务对象处理程序，该处理程序执行以下操作:

- 从 SAP 检索 RFC 事件数据。
- 创建 JMS-MQ 消息以持久地存储和管理 SAP 随事件一起发送的事务标识。
- 将从 SAP 接收的一个或多个 IDoc 的数据存储在 JMS-MQ 消息中。
- 通过 SAP 网关返回对特定于 ALE 的功能的响应。该响应指示已完成事务。

请求处理业务对象处理程序

可视连接器框架实例化 ALE 请求处理业务对象处理程序，该处理程序检查 SAP 的 WebSphere 业务对象中 TransactionId 属性的值。如果此值存在，它继续执行以下步骤。

1. 从 JMS-MQ 消息或从 SAP 获取 TID。
2. 将业务对象数据转换为 IDoc 数据格式（该格式由期望的功能模块接口对 SAP 中 RFC 调用所定义）。
3. 对 ALE 接口进行 RFC 调用。
4. 在 JMS-MQ 消息中更新此请求的 TID 状态。
5. 将成功响应返回到集成代理程序。

SAP 的业务对象的结构

SAP 的 WebSphere 业务对象将每个 IDoc 表示为包含两个子业务对象（控制记录业务对象和数据记录业务对象）的父包装程序业务对象。控制记录业务对象包含连接器处理业务对象所必需的元数据。数据记录业务对象包含将由 SAP 应用程序处理的实际业务对象数据以及元数据，连接器将需要该元数据以将其转换为 IDoc 结构以进行 RFC 调用。

连接器包括控制记录的业务对象定义。定义文件 `SAP_idoccontrol.xsd` 位于 `\repository\SAP` 目录中。

控制记录业务对象中的 `TABNAM` 属性指示父包装程序业务对象将调用哪个 SAP 功能模块:

- 值 `EDI_DC40` 表示 `idoc_inbound_asynchronous` 功能模块，连接器仅将该功能模块用于 SAP 4x。
- 值 `EDI_DC` 表示 `inbound_idoc_process` 功能模块，该功能模块是为了向后兼容 SAP 3x 而提供的。

另外，以下属性必须具有值，这样 SAP 才能够正确地在 ALE 中处理对象。这些值基于您的 ALE 配置:

- `Name_of_table_structure`
- `Client`
- `Name_of_basic_type`
- `Logical_message_type`
- `Partner_type_of_sender`

- Partner_number_of_sender
- Partner_type_of_recipient
- Partner_number_of_recipient

两个业务对象中的 DOCNUM 属性都会在数据记录业务对象与控制记录业务对象之间建立关系。

当处理服务调用请求时，ALE 模块可以处理单个业务对象中的多个 IDoc。但是，在它可以进行处理前，您必须在两个或更多父包装程序业务对象周围添加另一个多 IDoc 包装程序业务对象。这个顶级多 IDoc 包装程序业务对象包含一个表示一组父包装程序业务对象的属性。有关更多信息，请参阅第 133 页的『父包装程序业务对象』。

适配器包括一个业务对象生成工具 SAPODA。此工具使用 IDoc 定义文本文件来生成此 ALE 模块的业务对象定义。有关开发 ALE 模块的业务对象的更多信息，请参阅第 129 页的第 12 章，『为 ALE 模块开发业务对象』和第 41 页的第 5 章，『使用 SAPODA 生成业务对象定义』。

第 11 章 配置 ALE 模块

本章描述 ALE 模块的配置和使用。在执行本章中描述的配置任务之前，应安装 mySAP.com 适配器（R/3 V3.x）的连接器组件。

本章包含以下各节:

运行 ALE 模块的先决条件

要使连接器能够在事件处理期间持久地存储 TID 和 IDoc 数据，并能够在请求处理期间持久地存储 TID，您必须执行以下操作:

- 验证以下各项在系统上是否已安装并正在运行:
  - WebSphere MQ（未包括）
  - TCP/IP
- 对于事件处理，创建下列 WebSphere MQ 队列，其名称由相应的特定于连接器的配置属性指定:
  - 归档（SAPALE_Archive_Queue 属性）
  - 事件（SAPALE_Event_Queue 属性）
  - 正在进行的工作（WIP）（SAPALE_Wip_Queue 属性）
  - 错误（SAPALE_Error_Queue 属性）
  - 未预订（SAPALE_UnSubscribed_Queue 属性）
  - TID（SAPtid_Queue 属性）

  有关连接器如何使用这些队列的信息，请参阅第 118 页的『运行 ALE 模块』。
- 要使用 ALE 模块来处理很大的 IDoc 或 IDoc 信息包:
  - 增加 MQSeries 队列管理器及其队列的最大消息长度。此长度缺省为 4194304 个字节
  - 当创建队列管理器时，增加日志文件大小和日志文件的数目
  - 如果将通道用于 WebSphere MQ 队列管理器，则增大通道的最大消息长度

    有关配置日志文件的更多信息，参阅《MQSeries 系统管理》出版物。

ALE 模块目录和文件

第 116 页的表 18 列示由 ALE 模块使用的目录和文件。

表 18. ALE 模块目录和文件

文件名	事件	请求	描述
sap_idoccontrol.txt	是	是	控制记录业务对象定义文件。位于 \repository\SAP 目录中。
EventState.log 文件	是	否	位于在 AleEventDir 配置属性中指定的目录中，连接器将有关 JMS-MQ 事件消息中成功处理的 IDoc 的信息日志写入此文件。注意: 连接器在首次处理事件时不会自动创建该日志文件。在首次运行连接器之前，您必须创建此文件。

注: 在本文档中，反斜杠（\）用作目录路径的约定。对于 UNIX 安装，用斜杠（/）替代反斜杠（\）。所有文件路径名都是相对于该产品在系统上的安装目录。

配置 ALE 模块

在使用 ALE 模块之前，您必须执行以下操作:

- 将 ALE 模块的模块名添加至模块的属性。模块名是 ALE。
- 要启用具有 TID 管理的事件处理，您必须配置特定于连接器的适当属性。
- 要使连接器在 ALE 模块已检索 IDoc 进行事件处理之后更新标准 SAP 状态码，则配置 SAP 中逻辑系统的伙伴概要文件的特定属性和入站参数以接收 ALEAUD 消息类型。有关相关属性的更多信息和完整列表，请参阅第 117 页的『配置 SAP 以更新 IDoc 状态』。
- 设置其余必需的标准配置属性和特定于连接器的配置属性。

  要设置连接器配置属性，使用连接器配置器。有关设置连接器配置属性的更多信息，请参阅第 21 页的第 3 章，『配置连接器』和第 261 页的附录 D，『连接器的标准配置属性』。

重要提示: 连接器轮询是此模块在处理应用程序事件时正确管理错误所必需的。因此，不要将连接器的 PollFrequency 属性值设置为 key 或 no。在您已验证连接器的日志显示安装了必需的 RFC 功能部件之前，不允许 SAP 应用程序将事件触发至连接器。

检查 SAP 配置

在运行 ALE 模块之前，验证是否正确配置 SAP 系统来处理业务对象:

- 检查是否已对 SAP 系统和外部系统定义并分配逻辑系统（事务代码 SALE）。
- 检查是否已保留分布模型以及是否已将必需的消息类型添加至该模型（事务代码 BD64）。
- 检查逻辑系统或分布模型是否具有伙伴概要文件（事务代码 WE20）。

检查 MQ 配置

验证是否已正确配置消息队列。

对于事件处理:

- 检查 SAP 应用程序（事务代码 SM59）是否与在 RfcProgramId 配置属性中指定的程序标识匹配。有关设置 TCP/IP 端口的更多信息，请参阅第 159 页的『向 SAP 网关注册 RFC 服务器模块』。
- 检查 WIP（SAP_Wip_Queue）、事件（SAP_Event_Queue）、错误（SAP_Error_Queue）、未预订的（SAP_Unsubscribed_Queue）和归档的队列（SAP_Archive_Queue）是否已定义并正在 MQSeries 中运行。

对于请求处理，检查请求队列（SAPtid_Queue）是否已定义并正在 MQSeries 中运行。

配置 SAP 以更新 IDoc 状态

要使连接器在 ALE 模块已检索 IDoc 进行事件处理之后更新标准 SAP 状态码:

- 将 AleUpdateStatus 配置属性设置为 true，并设置 AleSuccessCode 和 AleFailureCode 配置属性的值。
- 配置逻辑系统的伙伴概要文件的入站参数以接收 ALEAUD 消息类型。

有关更多信息，请参阅第 122 页的『在 SAP 中更新 IDoc 状态』。

配置 SAP

配置逻辑系统的伙伴概要文件的入站参数以接收 ALEAUD 消息类型。将以下属性设置为指定的值:

表 19. 配置 SAP 以接收 IDoc 状态

SAP 属性	值
基本类型	ALEAUD01
逻辑消息类型	ALEAUD
功能模块	IDOC_INPUT_ALEAUD
进程代码	AUD1

设置特定于连接器的配置属性

设置以下特定于连接器的必需配置属性以返回 IDoc 状态:

- 第 286 页的『AleUpdateStatus』
- 第 286 页的『AleSuccessCode』
- 第 287 页的『AleFailureCode』

设置以下特定于连接器的必需配置属性以处理事件和请求:

- 第 290 页的『SAPtid_MQChannel』
- 第 290 页的『SAPtid_MQPort』
- 第 290 页的『SAPtid_QueueManager』
- 第 290 页的『SAPtid_QueueManagerHost』
- 第 290 页的『SAPtid_QueueManagerLogin』
- 第 290 页的『SAPtid_QueueManagerPassword』

您还可以设置以下特定于连接器的可选配置属性:

- 第 286 页的『AleSelectiveUpdate』
- 第 286 页的『AleStatusMsgCode』

- 第 287 页的『AleSuccessText』
- 第 287 页的『AleFailureText』

连接至远程队列管理器

为远程队列管理器设置以下特定于连接器的必需配置属性:

- 第 290 页的『SAPtid_MQChannel』
- 第 290 页的『SAPtid_MQPort』
- 第 290 页的『SAPtid_QueueManager』
- 第 290 页的『SAPtid_QueueManagerHost』
- 第 290 页的『SAPtid_QueueManagerLogin』
- 第 290 页的『SAPtid_QueueManagerPassword』

运行 ALE 模块

在处理应用程序事件时，ALE 模块接收 SAP 应用程序推送至连接器的事件。当处理请求时，ALE 模块接收来自集成代理程序的业务对象请求并将它们发送至 SAP 应用程序。

初始化和终止

`init()` 方法通过 SAP 网关打开与 SAP 应用程序的 RFC 连接。如果连接器未能初始化，则它使用 `terminate()` 方法来终止该连接。连接器通过与 SAP 网关断开连接来终止。

当处理应用程序事件或业务对象请求时，连接器的初始化进程执行以下任务:

1. 向 SAP 网关注册在 `RfcProgramID` 连接器配置属性中指定的程序标识。有关将程序标识设置为 TCP/IP 端口的信息，请参阅第 159 页的『向 SAP 网关注册 RFC 服务器模块』。
2. 打开与为连接器配置的队列的 MQSeries 会话。
3. 验证是否已创建进行事件和请求处理所必需的 MQSeries 队列。如果尚未创建它们，则该进程将终止连接器。

因为连接器支持多线程，所以当 ALE 模块处理来自集成代理程序的请求时，该模块使用 SAP 的 Java 连接器（SAPJCo）连接池来执行这种处理。

重要提示: 当您使用 ALE 模块来处理应用程序事件时，需要进行连接器轮询以正确地初始化该模块（以在服务器上安装 RFC 功能）以及为了该模块正确地管理错误。因此，不要将 PollFrequency 属性值设置为 `key` 或 `no`。在您已验证连接器的日志显示安装了必需的 RFC 功能部件之前，不允许 SAP 应用程序将事件触发至连接器。

处理业务对象

将通过事件处理或请求处理来启动 ALE 模块对 SAP 的 WebSphere 业务对象的处理。

当从 SAP 的 Java 连接器（SAPJCo）API 返回业务对象数据时，ALE 模块以下列格式接收 DATS 和 TIMS 字段的值: 对于 DATS 数据元素，该格式为 `YYYY-MM-DD`（包括连字号）；对于 TIMS 数据元素，该格式为 `HH:mm:ss`（包括冒号）。大写的 `HH` 表示

24 小时制时间，而不是 12 小时制时间。当处理事件时，ALE 模块更改这些格式以适合其相应业务对象属性的 8 个字符和 6 个字符的最大大小。连接器通过从日期数据除去连字号和从时间数据除去冒号来缩短值的长度。

事件处理

SAP 应用程序中两个 RFC 支持功能启动 ALE 模块的所有事件处理。ALE 用于事件处理的业务对象处理程序支持功能 `idoc_inbound_asynchronous` 和 `inbound_idoc_process`。

当处理事件时，此业务对象处理程序将业务对象持久地存储在 MQSeries 队列中。连接器保留与 RFC 调用相关的事务标识（TID）以保证将每段数据传递一次且仅传递一次。

重要提示: 单个 RFC 调用可以发送一个或多个 IDoc 的数据。因此，MQSeries 队列可以包含一条表示多个 IDoc（每个都表示一个业务对象）的 JMS-MQ 消息。每个 RFC 调用都与单个 TID 相关。

处理 MQSeries 队列中的事件: 图 51 举例说明了 ALE 模块如何处理 MQSeries 队列。

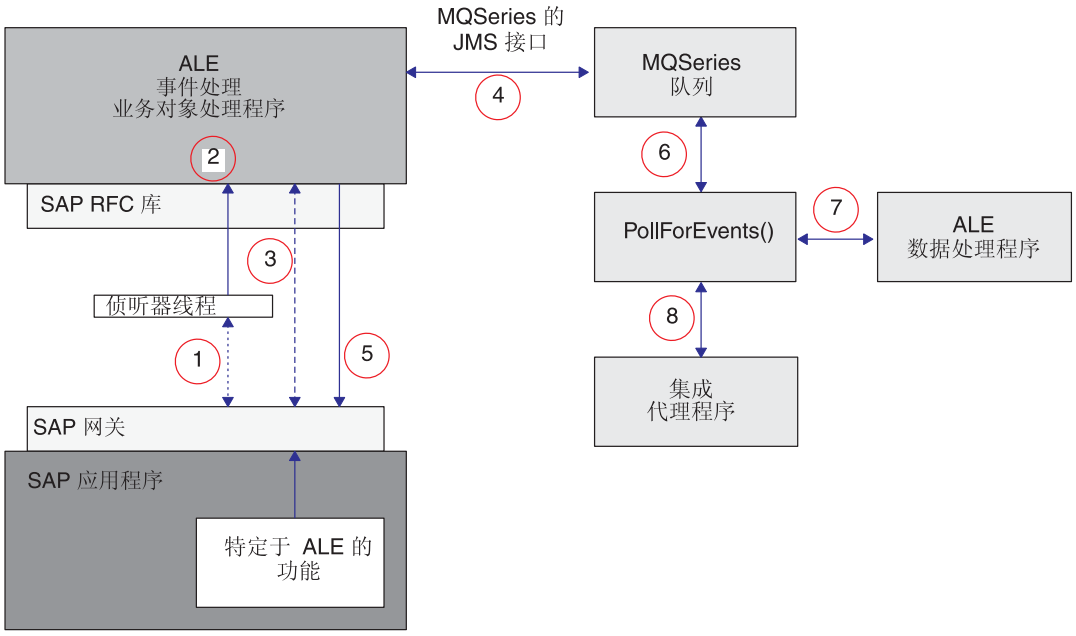


图 51. 业务对象事件处理

ALE 模块的业务对象事件处理以下列方式执行:

1. RFC 功能将事件数据推送至 SAP 网关，侦听器线程在该网关处选取事件。该线程检查与事件相关的 TID 以确定该 TID 的 JMS-MQ 消息是否存在:
   - 如果先前尚未发送该 TID，则连接器继续执行步骤 2。
   - 如果先前已发送该 TID，则连接器的行为取决于先前事务的状态。如果 TidStatus 是 `CREATED`，则连接器从消息除去 IDoc 数据。如果状态是 `ROLLBACK`，则连接器将状态更改为 `CREATED`。并且，如果 IDoc 数据在消息中存在，则连接器从消息除去 IDoc 数据。如果状态是 `EXECUTED`，则连接器将控制权返回给 SAP。
2. 侦听器线程实例化 ALE 事件处理业务对象处理程序，该处理程序从 SAP 网关检索 RFC 接口数据。

3. 业务对象处理程序将每个事务都格式化为一条 JMS-MQ 消息，将该消息持久地存储在由 SAPALE_Wip_Queue 配置属性指定的队列中。

   每条 JMS-MQ 消息都表示单个 RFC 调用。每个 RFC 调用都可以表示与单个 TID 相关的一个或多个业务对象。连接器将 TID 存储在消息的 CorrelationID 属性中，并将 TidStatus 设置为 CREATED，然后将 IDocProcessStatus 设置为 unknown。连接器使用消息体来存储 IDoc 数据。

   对于大对象，连接器会将对象分成多条消息，以便启用更高效的处理。有关如何启用此支持的详细信息，请参阅第 110 页的『事件处理组件』和第 121 页的表 20。
4. 在每个事务完成之后，连接器更改 TidStatus 的值，并将确认发送回 SAP 以指示事务已完成。在 SAP 接收到确认之后，它从 SAP 应用程序除去该 TID 及其相关数据。

   如果 AleUpdateStatus 配置属性求值为 true，则连接器在 SAP 中更新 IDoc 的状态。如果它检索到 IDoc 信息包，则它更新该信息包中所有 IDoc 的状态。有关更多信息，请参阅第 122 页的『在 SAP 中更新 IDoc 状态』。
5. 连接器将 JMS-MQ 消息从 WIP 队列移至由 SAPALE_Event_Queue 配置属性指定的队列。
6. ALE 模块的轮询线程从“事件”队列选取事件消息。
7. 连接器实例化 ALE 数据处理程序，该数据处理程序将把消息体中数据转换为业务对象以便记入集成代理程序。
8. 连接器尝试将每个业务对象都记入集成代理程序。如果集成代理程序是 WebSphere Interchange Server，则连接器首先检查是否预订了该业务对象。在处理消息体中的所有业务对象之后，将更新消息的 IDocProcessingStatus 和 BOProcessingStatus，并且将把该消息移至由 SAPALE_Archive_Queue 属性指定的队列。有关 IDocProcessingStatus 的更多信息，请参阅“创建归档消息”；有关 BOProcessingStatus 的更多信息，则请参阅“用于事件和归档处理的 JMS-MQ 消息的结构”。

当从“事件”队列读取消息时，ALE 模块使用 FIFO（先进先出）来维护处理顺序。

重要提示: 连接器轮询是此模块在处理应用程序事件时正确管理错误所必需的。因此，不要将连接器的 PollFrequency 属性值设置为 key 或 no。在您已验证连接器的日志显示安装了必需的 RFC 功能部件之前，不允许 SAP 应用程序将事件触发至连接器。

重新提交事件: 可以使用位于以下目录中的命令行实用程序（对于 Windows 为 BIA_AleEventUtil.bat，而对于 Unix 为 BIA_AleEventUtil.sh）来重新提交已放置在 SAPALE_Unsubscribed_Queue 和 SAPALE_Error_Queue 中的事件：*ProductDir*/connectors/SAP/utilities/ALEEventUtil/，其中 *ProductDir* 表示连接器的安装目录。有关更多信息，请参阅第 123 页的『用于事件处理的 ALE 模块队列管理实用程序』。

用于事件和归档处理的 JMS-MQ 消息的结构: 第 121 页的表 20 描述连接器发送至“事件”和“归档”队列的消息的结构。

表 20. 用于事件和归档处理的 JMS-MQ 消息的结构

JMS 消息头属性	描述
CorrelationId	连接器根据由 SAP 发送的事务标识（TID）来设置此属性的值。 当将大型 IDoc 分成更小的消息部分时，此属性标识该消息部分与哪个更大的消息有关。连接器将此值设置为集合中的第一部分的 JMSMessageID。注意，第一部分的 CorrelationID 始终是与大型 IDoc 相关联的第一条 JMS-MQ 消息的 CorrelationID。有关将 IDoc 分成更小的消息部分的详细信息，请参阅第 110 页的『事件处理组件』。
JMSMessageID	消息的唯一标识。当将大型 IDoc 分成更小的消息部分时，连接器将除了第一部分之外的所有部分的此属性的值设置为第一部分的 JMSMessageID。有关将 IDoc 分成更小的消息部分的详细信息，请参阅第 110 页的『事件处理组件』。
MutliPartMessage	当将大型 IDoc 分成更小的消息部分时，连接器使用此属性来标识具有更大消息的适当部分的消息。例如，假定将原始的更大 IDoc 消息分成 8 条 JMS-MQ 消息，则每一部分的 MultiPartMessage 属性的值分别为 8 部分中的第 1 部分、8 部分中的第 2 部分，依次类推。有关将 IDoc 分成更小的消息部分的详细信息，请参阅第 110 页的『事件处理组件』。
TidStatus	维护 TID 的状态。
IDocProcessStatus	在事件处理期间，维护 IDoc 对象的状态。
BOProcessingStatus	使用格式 <CID> :: <IDoc sequence number><Status symbol> 来维护消息中所有 IDoc 的状态。可能的状态符号为 S（表示成功）、F（表示失败）和 U（表示未预订）。例如，“<CID> :: 0S，1F，2U”表示，对于 CorrelationId = <CID>，第一个 IDoc 成功，第二个失败，第三个是未预订的。

表 21 描述在事件移至“归档”队列之后 IDocProcessStatus 属性的可能值。

表 21. IDocProcessStatus 消息属性的归档队列值

IDocProcessStatus 属性值	事件状态	描述
success	成功	在没有错误的情况下记入了消息中的所有业务对象。
partial	部分成功	在有错误的情况下记入了消息中的一个或多个业务对象但不是所有业务对象。如果集成代理程序是 WebSphere Interchange Server，则在有错误的情况下记入了消息中的一个或多个业务对象但不是所有业务对象，或者这些对象是未预订的。
unsubscribed	未预订	如果集成代理程序是 WebSphere Interchange Server，则消息中的所有业务对象都是未预订的。
fail	失败	在有错误的情况下记入了消息中的所有业务对象。

创建归档消息： 当将消息从“事件”队列移至“归档”队列时，将更新 IDocProcessingStatus 和 BOProcessingStatus。消息体仍保持不变。

例如，假定连接器处理一条具有 4 个 IDoc 的事件消息，它将每个 IDoc 转换或尝试转换为业务对象，在表 22 中举例说明了结果:

表 22. 归档消息创建

IDoc 或业务对象的状态	产生的归档消息
成功地转换第 1 个 IDoc，并且将该业务对象记入集成代理程序	IDocProcessStatus 更新为 success，并且 BOProcessingStatus 是 <CID> :: 0S
未能将第 2 个 IDoc 转换为业务对象	IDocProcessStatus 更新为 partial，并且 BOProcessingStatus 是 <CID> :: 0S，1F
成功地转换第 3 个 IDoc，并且将该业务对象记入集成代理程序	IDocProcessStatus 设置为 partial，并且 BOProcessingStatus 是 <CID> :: 0S，1F，2S
成功地转换第 4 个 IDoc，但集成代理程序中未预订创建的业务对象	• IDocProcessStatus 设置为 partial，并且 BOProcessingStatus 是 <CID> :: 0S，1F，2S，3U • 在处理最后一个 IDoc 之后，将消息从“事件”队列移至“归档”队列，并将该 IDoc 的 IDocProcessStatus 设置为 partial 以及将 BOProcessingStatus 设置为 <CID> :: 0S，1F，2S，3U

支持用于事件处理的多种消息类型:

可以将连接器的同一实例用于引用了同一 IDoc 类型的不同消息类型，但是，每种消息类型需要不同的业务对象定义。要创建不同的业务对象定义，复制并重命名 IDoc 类型的业务对象定义。务必在适当查询描述的查询描述 ASI 中配置正确的 MsgType。

在 SAP 中更新 IDoc 状态: 要使连接器在 ALE 模块已检索 IDoc 进行事件处理之后更新标准 SAP 状态码，您必须执行以下操作:

- 将 AleUpdateStatus 配置属性设置为 true，并设置 AleSuccessCode 和 AleFailureCode 配置属性的值。
- 配置逻辑系统的伙伴概要文件的入站参数以接收 ALEAUD 消息类型。

如果 AleUpdateStatus 求值为 true，则连接器将 ALEAUD IDoc 以及状态码信息和描述文本一起发送至 SAP。ALEAUD IDoc 调用 IDOC_INPUT_ALEAUD 功能模块。连接器支持将以下状态码发送至此功能模块:

- 已将 IDoc 完全记入业务集成系统中。

  特定于连接器的配置属性 AleSuccessCode 可以具有值 52 或 53。SAP 将此值转换为 41。
- 不能在 Business Integration 系统中处理 IDoc。

  特定于连接器的配置属性 AleFailureCode 可以具有值 68。SAP 将此值转换为 40。

在以上两种情况中，Business Integration 系统未发送将指示进一步处理的进一步状态码。

有关设置返回 IDoc 状态所必需的特定于连接器的配置属性的信息，请参阅:

- 第 286 页的『AleUpdateStatus』
- 第 286 页的『AleSuccessCode』
- 第 287 页的『AleFailureCode』

有关设置对于返回 IDoc 状态是可选的特定于连接器的配置属性的信息，请参阅:

- 第 286 页的『AleSelectiveUpdate』

- 第 286 页的『AleStatusMsgCode』
- 第 287 页的『AleSuccessText』
- 第 287 页的『AleFailureText』

用于事件处理的 ALE 模块队列管理实用程序

使用此命令行实用程序来维护由 WebSphere Business Integration mySAP.com 适配器（V.5.3.2）的 ALE 模块使用的 MQ 队列。此实用程序重新提交事件消息，将事件消息转储至文件系统以便查看并将消息归档至文件系统。

将在称为事务的工作单元中处理 IDoc。包含多个 IDoc 的 SAP 事务称为事务信息包。适配器通过使用 MQ 消息保存一个或多个 IDoc 来处理事务和事务信息包。适配器将 IDoc 转换为其相应的业务对象。ALE 模块在一个两步骤的进程中处理 IDoc: 先是从 SAP 至适配器，然后是从适配器至代理程序。每个步骤将以不同方式处理异常。

有关 MQ 消息的更多信息，请参阅 WebSphere Business Integration 库: http://www.ibm.com/software/integration/wmq/library/。

处理从 SAP 至适配器的 IDoc: 如果适配器检测到未预订的或不受支持的业务对象或在 IDoc 传输期间发生任何异常，则适配器将不能执行该 SAP 事务。可以查看失败的事务，并可以从 SAP 事务 SM58 重新提交这些事务。在重新提交该事务之前，解决以下异常:

- 不受支持: 为业务对象添加代理程序支持。
- 未预订: 重新启动业务对象的协作。
- 其它异常: 查看适配器日志以确定该异常然后进行必要的纠错。

一旦成功地执行了此步骤，就完成了 SAP 的事务。

重要提示: 要防止重复的事件传递，不要重新提交修正的 IDoc 事务或事务信息包内的各个 IDoc。

处理从适配器至代理程序的 IDoc: 如果 MQ 消息包含单个业务对象且该业务对象是未预订的，则将把该 MQ 消息移至未预订队列。事务信息包内每个未预订业务对象都将在未预订队列中作为它自己的 MQ 消息继续存在。原始 MQ 消息保持不变并包含各个 IDoc 的处理状态。一旦处理完 MQ 消息的事务信息包，将把该信息包移至归档队列。

在重新提交该事务之前，解决以下异常:

- 未预订: 重新启动业务对象的协作。
- 其它异常: 查看适配器日志以确定该异常然后进行必要的纠错。

在完成纠错之后，使用命令实用程序 `AleEventUtil` 将 MQ 消息移回事件队列，并重新提交该事件:

当 IDoc 包含格式不正确的数据或不包含数据时，将把该 IDoc 移至“错误”队列作为它自己的消息。

安装和配置 ALE 模块队列实用程序: ALE 模块队列实用程序与 SAP 适配器打包在一起。当安装时，它具有以下目录结构:

```
\Connectors\SAP\BIA_AleEventUtil.jar

\Connectors\SAP\BIA_AleEventUtil.bat
```

\Connectors\SAP\BIA_AleEventUtil_readme.txt

修改启动脚本文件 BIA_AleEventUtil.bat 以捕获以下参数。要创建本地队列管理器，您只需配置 MQQueueManager。

变量	描述	注释
MQQueueManager	队列管理器的名称	必需参数。
MQChannel	服务器连接通道名称	对于访问远程队列管理器是必需的。
MQPort	侦听通道所在的端口	对于访问远程队列管理器是必需的。
MQHost	正在运行队列管理器的主机名或 IP 地址	对于访问远程队列管理器是必需的。
MQUser	MQHost 上的有效用户名	对于访问远程队列管理器是必需的。
MQPassword	用户密码	对于访问远程队列管理器是必需的。值是未加密的。

运行 MQ 管理实用程序: 在安装并配置该实用程序之后，浏览至安装了 ALE 模块队列管理实用程序的目录。该实用程序的有效命令如下:

-c <choice>（有效的选项为 [move, archive, dump, replicate]）

-i <inputq>

-o <outputq>

-f <outputfile>

-d <date>

-u <unique message ID>

-n <replication count>

注: 当存在同名的现有文件时，archive 命令将发生异常，但 dump 命令将覆盖该文件。

要将消息的内容转储到文件，在命令提示符下进入安装了该实用程序的目录并运行以下命令:

```
BIA_AleEventUtil -cdump -i<QueueName> -f<OutputFileName>
```

要将消息从一个队列移至另一个队列，运行以下命令。此命令将移动队列中的所有消息:

```
BIA_AleEventUtil -cmove -i<FromQueue> -o<ToQueue>
```

要移动单个消息，使用与所需消息的消息标识对应的额外参数 MessageIdByte:

```
BIA_AleEventUtil -cmove -i<FromQueue> -o<ToQueue> -u<MessageIdByte>
```

要移动等于或早于指定日期的所有消息，添加 Date 参数:

```
BIA_AleEventUtil -cmove -i<FromQueue> -o<ToQueue> -d<date(YYYYMMDD)>
```

要将消息从队列归档至文件，并除去等于或早于指定日期的所有消息，则使用以下命令:

```
BIA_AleEventUtil -carchive -i<QueueName> -f<ArchiveFileName> -d<date(YYYYMMDD)>
```

请求处理

可视连接器框架使用顶级业务对象的查询描述 AppSpecificInfo 属性的值，来实例化 ALE 请求处理业务对象处理程序。请求处理业务对象处理程序中的 doVerbFor() 方法启动所有业务对象请求。

业务对象处理程序将业务对象数据转换为两个表，这两个表表示 IDoc 格式及其元数据组件（即控制记录）。一旦数据使用 IDoc 格式，则业务对象处理程序对适当的 SAP 功能模块执行 RFC 调用: idoc_inbound_asynchronous 或 inbound_idoc_process。因为 ALE 是异步的，所以连接器不会等待返回响应。

重要提示: 缺省情况下，由 SAPODA 生成的父包装程序业务对象包含 TransactionId 属性。此属性中的值导致连接器在处理服务调用请求时管理 TID。如果您不想对请求处理进行 TID 管理，则不要为此属性设置值。有关更多信息，请参阅第 133 页的『父包装程序业务对象』。

注: TransactionId 属性的值必须是唯一标识。该值不等于 SAP 应用程序中的 TID 。这些值存储在一个表中，该表在由 SAPtid_Queue 配置属性指定的队列的 JMS_MQ 消息内。

如果 TransactionId 属性不具有值，则 ALE 模块将请求直接发送至 SAP。如果 TransactionId 属性具有值，则 ALE 模块执行下列操作:

1. 连接器检查由 SAPtid_Queue 配置属性指定的队列中的 JMS-MQ 消息是否具有此值。
   - 如果业务对象的 TransactionId 属性值或 ObjectID 在 JMS_MQ 消息的表中不存在，则将在表中创建新条目。ObjectID 成为该表条目的键。然后连接器从 SAP 检索新的 TID，并将该 TID 分配给此 ObjectID。连接器还将此 ObjectID 的 TidStatus 设置为 CREATED。
   - 如果 ObjectID 在表中存在，则连接器的行为取决于此对象标识的 TidStatus。如果 TidStatus 是 CREATED，则连接器继续执行步骤 2。如果 TidStatus 是 ROLLBACK，则连接器将该值更改为 CREATED，并继续执行步骤 2。如果TidStatus 是 EXECUTED，则将除去并归档该键。
2. 连接器将业务对象转换为 RFC 表并对 SAP 执行 RFC 调用。
   - 如果调用已成功记入，则连接器将该键的 TidStatus 更新为 EXECUTED。
   - 如果调用未能记入 SAP 或发生异常，则连接器将该键的 TidStatus 更新为 ROLLBACK。
3. 在 SAP 确认接收到 RFC 调用之后，连接器从表中除去该键、归档该键并将成功状态返回到集成代理程序。

归档: 在成功处理服务调用处理之后，将把 SAPtid_Queue 中 JMS-MQ 消息的表中相应条目除去并归档至一个目录。对于 WINNT，将在 \ale\request 子目录中创建文件，对于 Unix 系统，将在 /ale/request 中创建文件。ale 子目录位于启动适配器的目录中。将使用已从表中除去的条目来创建新文件。文件名将具有以下格式:

<ObjectID>_<TID><timestamp>.executed，其中 ObjectID 是来自 TransactionId 属性的值，TID 是来自 SAP 的事务标识，而 timestamp 是创建文件时的时间戳记。

适配器本身使用连接器配置属性 ArchiveDays 来管理这些归档文件的删除。连接器配置属性 ArchiveDays 中的值确定这些归档文件将保留在 ale\request 子目录中的天数。将删除超过 ArchiveDays 中指定的天数的任何文件。如果未配置此属性，则 ArchiveDays 的缺省值为 7 天。还可以通过自己删除归档文件来手工管理这些文件。

重新提交失败请求: 对于集成代理程序指出的所有失败请求，检查是否为该请求创建了归档文件。如果该请求中的对象标识存在归档文件，则不要从集成代理程序重新提交该请求。如果该 ObjectID 没有归档文件，则重新提交请求。确保 ArchiveDays 连接器配置属性已设置为将允许验证重新提交的请求的值。

用于请求处理的 JMS MQSeries 消息的表中列: 表 23 描述连接器从 SAPtid_Queue 获取的 JMS-MQSeries 消息的列:

表 23. 用于请求处理的 JMS-MQ 消息的列

列名	描述
ObjectID	所请求业务对象的 TransactionID 属性中的值。此值用作表键。
TID	从 SAP 获取的事务标识
TidStatus	事务的状态

支持用于请求处理的多种消息类型:

对于事件处理，可以使用下列机制:

- 使用同一业务对象来表示 IDoc 类型，查询描述 ASI 元数据是使用 MsgType/MsgCode/MsgFunction 的不同组合来配置的。为每个查询描述指定的值的组合应该是不同的。例如，按如下所示为不同的查询描述配置 ASI:

  ```
  Verb=Create   VerbASI : MsgType=ORDERS; MsgCode=MC01;MsgFunction=MF01

  Verb=Update   VerbASI : MsgType=ORDERS;MsgCode=MC02;MsgFunction=MF02

  Verb=Delete    VerbASI : MsgType=ORDERS;MsgCode=MC03;MsgFunction=MF03
  ```

  注意，两个不同的查询描述不能使用 MsgType/MsgCode/MsgFunction 值的相同组合。

  或者，可以使每个查询描述具有不同的消息类型:

  ```
  Verb=Create  VerbASI : MsgType=ORDERS;MsgCode=;MsgFunction=

  Verb=Update   VerbASI :  MsgType=ORDCHG;MsgCode=;MsgFunction=

  Verb=Delete    VerbASI :  MsgType=;MsgCode=;MsgFunction=
  ```

- 如果需要对不同消息类型使用业务对象与查询描述的相同组合，则应使用不同的名称来创建同一 IDoc 类型业务对象的备份。例如，业务对象 sap_orders_05_ORDERS 和 sap_orders_05_QUOTES 都引用同一 IDoc 类型定义，它们是同一业务对象的备份。每个对象的 ASI 配置为如下所示:

  Verb ASI for sap_orders_05_ORDERS

  ```
  Verb=Create  VerbASI : MsgType=ORDERS;MsgCode=;MsgFunction=
  ```

Verb ASI for sap_orders_05_QUOTES

```
Verb=Create  VerbASI :  MsgType=QUOTES;MsgCode=;MsgFunction=
```

第 12 章 为 ALE 模块开发业务对象

本章描述 ALE 模块需要的业务对象。本章还讨论业务对象生成实用程序 SAPODA 如何生成定义。本章假定您熟悉连接器如何处理业务对象。有关 ALE 模块的更多信息，请参阅第 109 页的第 10 章，『ALE 模块概述』。

使用 SAPODA 生成此模块的业务对象定义。SAPODA 使用 SAP 应用程序的本机 IDoc（中间文件）定义作为 ALE 模块的业务对象定义的模板。在创建定义之后，可以使用业务对象设计器或文本编辑器来修改它们。您可以使用 SAPODA 来基于 IDoc 生成 ALE 模块的业务对象定义:

- 解压缩至文件
- 在 SAP 系统中定义

IDoc 必须遵循特定格式以便 SAP 正确处理它们。因此，当您为 ALE 模块开发业务对象定义时，确保定义遵循 SAP 中定义的 IDoc 结构。

有关使用 SAPODA 的信息，请参阅第 41 页的第 5 章，『使用 SAPODA 生成业务对象定义』。

本章包含以下各节:

- 『创建 IDoc 定义文件』
- 第 130 页的『业务对象结构』
- 第 139 页的『受支持的查询描述』
- 第 140 页的『处理多个具有包装程序业务对象的 IDoc』

创建 IDoc 定义文件

在使用 SAPODA 来根据 IDoc 定义文件生成业务对象定义之前，您必须为您要支持的每个 IDoc 创建 IDoc 定义文件。SAPODA 使用此文件作为输入。在 SAP 中使用事务 WE63 来创建 IDoc 定义文件。

注: 如果您使用 SAPODA 来根据 SAP 系统中定义的 IDoc 生成定义，则您不必创建此 IDoc 定义文件。

要创建 IDoc 定义文件:

1. 在 SAP 中，通过输入 /oWE63 来选择事务 WE63。
2. 取消选择“输出 IDoc 记录”复选框。
3. 选择“输出 IDoc 类型”复选框。
4. 在“IDoc 类型”字段中，输入基本 IDoc 类型或定制 IDOC 类型。
5. 选择“输出段字段”复选框。
6. 单击屏幕顶部的“执行”图标。IDoc 定义将显示在屏幕上。
7. 将该定义保存到本地目录。

重要提示: 您必须以英语登录到 SAP 系统以根据 IDoc 文件生成业务对象定义。因为 SAPODA 在 IDoc 的定义中使用文本字段来生成属性名，并且因为属性名必须是英语的，所以根据英语文件生成定义很重要。

业务对象结构

ALE 模块的 SAP 的 WebSphere 业务对象由一个顶级父包装程序对象和两个子对象（控制记录对象和数据记录对象）组成。本节描述以下内容:

举例说明业务对象结构

图 52 举例说明了 ALE 模块的 WebSphere 业务对象的结构。

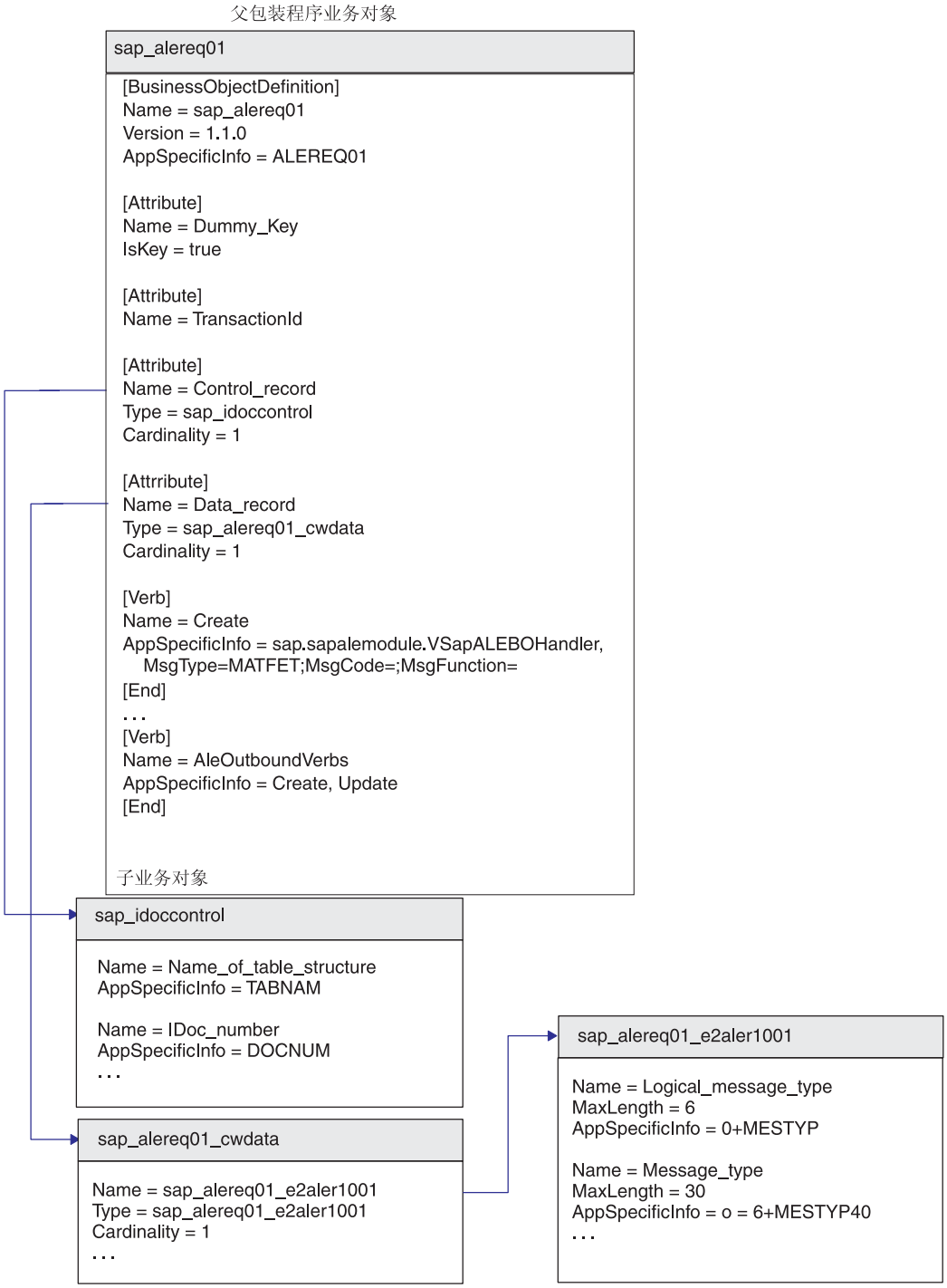


图 52. SAP 的 WebSphere 业务对象和 IDoc 的关系

业务对象命名约定

本节描述以下内容:

标准命名约定

ALE 模块要求其业务对象遵循表 24 中描述的命名约定。SAPODA 生成除控制记录业务对象以外的所有业务对象，它依照以下约定从 IDoc 定义中获得业务对象和属性名。

表 24. IBM WebSphere SAP 业务对象命名约定

IBM WebSphere 业务对象或属性	名称	类型
父包装程序业务对象	*BOprefix_BasicIDocType* **注**: 本章中的例子使用 SAP_ 或 sap_ 作为业务对象前缀。您在创建业务对象定义时，可以指定自己的有意义前缀。	不适用
控制记录业务对象	Control_record	sap_idoccontrol
数据记录业务对象	Data_record	*BOprefix_BasicIDocType*_cwdata
数据记录子业务对象	*BOprefix_BasicIDocType_ IDocSegmentName*	*BOprefix_BasicIDocType_ IDocSegmentName*
数据记录属性	*IDocFieldName* 或 IDoc 字段描述	当生成业务对象时，用户可以选择 IDoc 段字段名或字段描述作为业务对象属性名。

WebSphere Business Integration 系统中的组件名称仅支持字母数字字符和下划线字符（_）。因此，当在生成的业务对象定义中命名组件时，SAPODA 用下划线字符替换 IDoc 段字段描述或字段名中的特殊字符。例如，SAPODA 将以下 SAP 描述中的空格、圆括号和句点更改为相应属性名中的下划线:

```
Partner function (e.g. sold-to party, ship-to party)
```

SAPODA 在生成的业务对象定义中将以上定义表示为:

```
Partner_function__e_g__sold_to_party__ship_to_party__
```

SAPODA 保证业务对象定义中的所有属性名都是唯一的。如果 IDoc 有多个字段具有相同字段描述，则 SAPODA 将一个计数器后缀添加至生成的属性名。

当命名属性时，如果更改的属性名符合以下条件，则 SAPODA 将在属性名前面加上一个字符串:

- 以数字开头 - 在前面加上 A_
- 以下划线字符（_）开头 - 在前面加上 A

重要提示: 在生成业务对象之后，您可以在任何时间修改属性名。但是，当修改属性名时，不要修改其特定于应用程序的信息。连接器使用此文本来标识业务对象属性所对应的 IDoc 字段。有关更多信息，请参阅第 136 页的『特定于应用程序的信息: 数据记录业务对象』。

使用 IDoc 扩展 / 类型

您可以根据定制 IDoc 类型来生成业务对象定义，定制 IDoc 类型是从基本 IDoc 类型和扩展类型的组合中创建的。SAPODA 以下列格式为它生成的父包装程序业务对象命名:

sap_*IDocType*

重要提示: 当将 IDoc 扩展的业务对象定义装入 InterChange Server 资源库时，如果基本 IDoc 类型的业务对象定义在资源库中已存在且其名称与基本 IDoc 类型和扩展匹配，则您可能会遇到冲突。您必须手工解决这些冲突。

父包装程序业务对象

父包装程序业务对象的名称是前缀为用户定义前缀再后跟下划线（_）（如 sap_.）的基本 IDoc 类型。父包装程序业务对象包含四个属性: Dummy_key 、Control_record、Data_record 和 TransactionId。

IDoc 顶级对象 Dummy_key 属性用来将键字段从“控制”和“数据”记录映射至顶级对象中的 Dummy_key。连接器按下列方式处理 Dummy_key 映射:

1. Dummy_key 属性的属性级别 ASI 被配置为属性路径（值从该属性中设置）。换句话说，属性级别 ASI 被设置为属性（该属性映射至顶级对象）的业务对象树中路径。值对的定界符为 ;（分号）。从子代到键属性的路径的定界符是 :（冒号）。应该为外键（FK）指定绝对路径。

   例 如 ,
   DummyKey;FK=Data_record:sap_orders05_e2edk01005:IDOC_document_number"
2. 如果连接器在此路径中检测到多基数对象，则它会使用此容器中的第一个子实例。只要多基数对象出现在业务对象树中，它们就满足这种情况。
3. 如果 ASI 不正确或者如果映射的属性值是空的，则连接器将使事件失败并将它放置在 SAPALE_Error_Queue 中。当 ASI 配置为将对象类型值设置为 Dummy_key 时也是这种情况。注意，Dummy_key 属性只允许简单类型属性的值。

Control_record 和 Data_record 属性表示单基数子业务对象。

Control_record 属性的类型是 sap_idoccontrol。此业务对象定义是与 ALE 模块一起提供的。

Data_record 属性的类型是 *BOprefix_BasicIDocType*_cwdata。此业务对象定义包含一个或多个子业务对象，这取决于 SAP 应用程序中基本 IDoc 类型的 IDoc 段定义。

TransactionId 属性中的值确定连接器在处理服务调用请求时是否管理 TID。如果您不想对请求处理进行 TID 管理，则不要为 TransactionID 属性设置值。

父包装程序业务对象的特定于应用程序的信息指示以下内容:

- 要创建的 IDoc 的类型
- IDoc 扩展 – 仅当根据基本 IDoc 类型的定制来生成业务对象时才设置。有关生成 IDoc 定义文件的更多信息，请参阅第 41 页的『在使用 SAPODA 之前』。
- ALE 通信伙伴信息 – 仅当您的数据需要多个伙伴类型、伙伴号或伙伴功能时才设置。

语法

父包装程序对象的 AppSpecificInfo 属性具有以下语法:

```
BasicIDocType [,Pn=PartnerNumberOfRecipient [,Pt=
PartnerTypeOfRecipient[,Pf=PartnerFunctionOfRecipient
]]
```

语法说明

BasicIDocType
: 指定基本 IDoc 类型

Ext
: 指定扩展类型

Pn	指定接受者的伙伴号
Pt	指定接受者的伙伴类型
Pf	指定接受者的伙伴功能

示例

```
AppSpecificInfo = ALEREQ01,Pn=ALESYS2,Pt=LS,Pf=EL
```

控制记录业务对象

ALE 模块对所有 IDoc 都使用通用控制记录业务对象定义。它包含控制记录的 3.x 版本（SAP 结构 EDI_DC）和 4.x 版本（SAP 结构 EDI_DC40）中存在的属性的超集。控制记录业务对象定义是与 ALE 模块一起提供的，必须将它装入业务对象资源库中。使用业务对象设计器将该业务对象装入资源库。

注: 或者，如果 IBM WebSphere InterChange Server 是集成代理程序，则您可以使用 repos_copy 命令。

表 25 列示控制记录业务对象的简单属性特性。

表 25. 控制记录业务对象中简单属性的特性

属性名	描述
Name	Name 属性的值是 IDoc 定义中 TEXT 字段的修改值。SAPODA 用下划线替换特殊字符（如句点、斜杠和空格），以便名称仅包含字母数字字符和下划线字符（_），如第 131 页的『业务对象命名约定』中所述。
Type	指定数据的类型。SAPODA 将其值设置为 String。
MaxLength	SAPODA 从 IDoc 定义中的 LENGTH 字段获取 MaxLength 的值。
IsKey	SAPODA 在业务对象的第一个属性上将此属性设置为 true。
IsForeignKey	SAPODA 将其值设置为 false。
IsRequired	IsRequired 特性指定属性是否必须包含值。SAPODA 仅在控制记录对象中的 Name_of_table_structure 属性上将此特性设置为 true。
AppSpecificInfo	SAPODA 从 IDoc 定义中的 NAME 字段获取其值。
DefaultValue	指定在没有运行时值时要赋予此属性的值。SAPODA 没有为此属性设置值。

重要提示: 当控制记录业务对象中属性的值设置为 CxIgnore 或 CxBlank 时，连接器对 IDoc 控制记录将其值设置为空格。

数据记录业务对象

IDoc 定义文件具有关于 IDoc 结构、IDoc 段层次结构和一些组成段的字段的信息。SAPODA 使用 IDoc 作为输入来生成数据记录业务对象及其子业务对象。子代的数目取决于 SAP 应用程序中基本 IDoc 类型的 IDoc 段定义。

顶级数据记录业务对象对应于基本 IDoc 类型。此顶级业务对象包含一个属性，该属性表示一个子业务对象或一组子业务对象（每个 IDoc 段一个）。子业务对象的结构和层次结构与基本 IDoc 类型中 IDoc 段的结构和层次结构匹配。

使用 SAPODA 从系统创建 IDoc 时，将通过在 SAP 系统内部进行调用来创建数据记录对象及其子业务对象。本节将使用 IDoc 定义文件中的字段来帮助说明如何设置业务对象的不同属性。从系统生成 IDoc 将使用在 SAP 系统内部进行的调用中的相应字段。

本节描述:

- 『属性: 数据记录业务对象』
- 第 136 页的『特定于应用程序的信息: 数据记录业务对象』
- 第 137 页的『举例说明业务对象和 IDoc 之间的关系』

属性: 数据记录业务对象

表 26 描述数据记录业务对象中每个简单属性的特性。SAPODA 生成下面描述的属性。

表 26. 简单属性: 数据记录业务对象

属性名	描述
Name	Name 属性的值是 IDoc 定义中 NAME 或TEXT 字段的修改值。SAPODA 用下划线替换特殊字符（如句点、斜杠和空格），以便名称仅包含字母数字字符和下划线字符（_），如第 131 页的『业务对象命名约定』中所述。
Type	指定数据的类型。SAPODA 将其值设置为 String。
MaxLength	SAPODA 从 IDoc 定义中的 LENGTH 字段获取 MaxLength 的值。
IsKey	SAPODA 在每个业务对象的第一个属性上将此属性设置为 true。对于其它每个属性，SAPODA 将其值设置为 false。
IsForeignKey	SAPODA 将其值设置为 false。
IsRequired	指定属性是否必须包含值。SAPODA 将其值设置为 false。
AppSpecificInfo	SAPODA 将 AppSpecificInfo 属性的值设置为 IDoc 定义中 Name 字段的值，并在该值前面加上偏移值和 + 字符；例如，对于偏移为 40 的名为 SIGN 的段字段，它为 AppSpecificInfo 设置以下值: 40+SIGN。有关更多信息，请参阅第 136 页的『特定于应用程序的信息: 数据记录业务对象』。
DefaultValue	指定在没有运行时值时要赋予此属性的值。SAPODA 没有为此属性设置值。

重要提示: 数据记录业务对象中的简单属性可以具有两个特殊值: CxIgnore 和 CxBlank。设置为 CxIgnore 或 CxBlank 的简单属性由段数据字符串中的空格表示。SAP 通过将一个空格字符放置在应用程序字段中来处理这些属性。

表 27 描述表示一个子业务对象或一组子业务对象的数据记录业务对象中每个属性的特性。SAPODA 生成下面描述的属性。

表 27. 表示子业务对象的属性

属性名	描述
Name	SAPODA 将其值设置为 BOprefix_BasicIDocTypeIdocSegmentName; 例如，SAP_E2ALER1001
Type	SAPODA 将其值设置为: BOprefix_BasicIDocTypeIdocSegmentName
ContainedObjectVersion	SAPODA 将其值设置为 1.0.0。
Relationship	SAPODA 将其值设置为 containment。
IsKey	SAPODA 将其值设置为 false。
IsForeignKey	SAPODA 将其值设置为 false。

表 27. 表示子业务对象的属性 (续)

属性名	描述
IsRequired	IsRequired 属性指定子业务对象是否必须存在。如果 IDoc 定义中相应段的 STATUS 字段的值为 OPTIONAL，则 SAPODA 将该属性的值设置为 false。如果 IDoc 定义中 STATUS 字段具有值 MANDATORY，则 SAPODA 将此属性设置为 true 。
AppSpecificInfo	AppSpecificInfo 属性包含关于层次结构级别以及允许一个段出现的最小和最大次数的信息。有关更多信息，请参阅『表示子代的属性中特定于应用程序的信息』。
Cardinality	如果 IDoc 定义中的 LOOPMAX 字段的值是 1，则 SAPODA 将该属性的值设置为 1。如果 LOOPMAX 的值大于 1，则 SAPODA 将该属性的值设置为 n。

特定于应用程序的信息: 数据记录业务对象

本节描述连接器如何使用 AppSpecificInfo 属性的值:

- 『业务对象级别的特定于应用程序的信息』
- 『简单属性中特定于应用程序的信息』
- 『表示子代的属性中特定于应用程序的信息』

业务对象级别的特定于应用程序的信息: 连接器使用数据记录及其每个子代的业务对象级别的 AppSpecificInfo 属性值，来获取相关 Idoc 及其段的名称:

- 数据记录业务对象上特定于应用程序的信息的语法如下:

  *IDocType*_CWDATA

  例如，如果给出名为 ALERQ01 的 IDoc，则 SAPODA 将 AppSpecificInfo 属性的值创建为 ALERQ01_CWDATA。
- 数据记录业务对象的子代上特定于应用程序的信息值是相应的段名。例如，如果给出具有两个名为 E2ALER1001 和 E2ALEQ1 的段的 IDoc ALERQ01，则 SAPODA 自动将两个子业务对象的 AppSpecificInfo 属性值创建为:
  - 第一个子代: E2ALER1001
  - 第二个子代: E2ALEQ1

简单属性中特定于应用程序的信息: 连接器使用简单属性的 AppSpecificInfo 属性的值来获取 SAP 中的字段名及其在数据字符串中的位置（偏移）。

偏移值是数据字符串中属性值的第一个字符的位置。将通过从给定属性的 BYTE_FIRST 值中减去 IDoc 定义中第一个字段的 BYTE_FIRST 值中的值来计算偏移值。此值与 MaxLength 属性一起用来构建 IDoc 段的数据字符串。

简单属性的 AppSpecificInfo 属性的语法如下:

OffsetNumber+IDoc*FieldName*

例如，偏移为 40、名为 SIGN 的段字段具有以下 AppSpecificInfo 值:

```
40+SIGN
```

表示子代的属性中特定于应用程序的信息: 连接器使用表示一个子业务对象或一组子业务对象的属性的 AppSpecificInfo 属性值，来获取关于层次结构级别以及允许一个段

出现的最小和最大次数的信息。SAPODA 通过从 IDoc 定义中的 LEVEL、LOOPMIN 和 LOOPMAX 字段获取信息来为这些属性设置 AppSpecificInfo 属性。

举例说明业务对象和 IDoc 之间的关系

图 53 举例说明了 WebSphere 数据记录业务对象和 SAP 应用程序中的 IDoc 定义之间的关系。

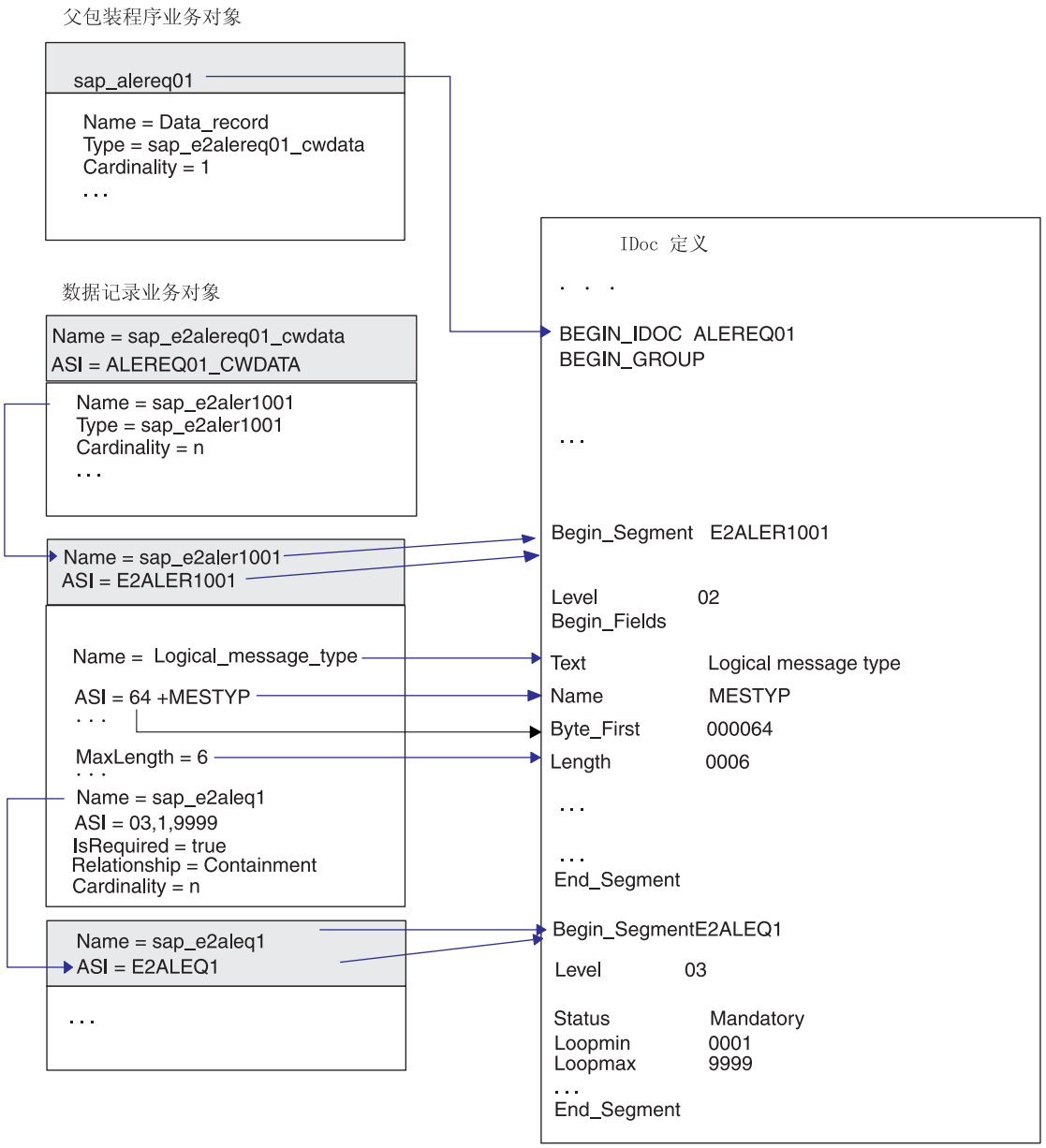


图 53. 数据记录业务对象和 IDoc 定义字段之间的关系

受支持的查询描述

ALE 模块的查询描述支持受 SAP 在其 ALE 接口中支持的查询描述限制。SAPODA 在业务对象定义中生成“创建”、“更新”、“删除”和“检索”查询描述。每种查询描述的实现都要求了解 SAP 内的 ALE 配置。

SAPODA 为查询描述生成 AppSpecificInfo，并在父包装程序业务对象上生成 AleOutboundVerbs 元查询描述。但是，它仅使用值填充 AppSpecificInfo 的其中一个参数：它指定要用于服务调用请求处理的业务对象处理程序。对于所有其它处理，您必须手工修改业务对象定义来添加或除去特定信息：

- 当将业务对象用于事件处理时，您必须为以下 AppSpecificInfo 属性指定值：
  - 父包装程序业务对象的查询描述 - 为唯一标识查询描述的那些参数指定值。根据 ALE 配置的要求，指定消息类型、消息代码和消息功能。在您将业务对象定义导入资源库之后再执行这些更改。

    重要提示： SAPODA 插入指定业务对象处理程序的 AppSpecificInfo 值，连接器将该处理程序仅用于请求处理。SAPODA 不会插入消息参数的值。如果您要将 ALE 模块用于事件处理，则您必须手工为消息参数添加值。
  - 父包装程序业务对象的 AleOutboundVerbs 元查询描述 - 用于事件处理的受支持查询描述的逗号分隔列表。
- 当将业务对象用于请求处理时，您必须为以下 AppSpecificInfo 属性指定值：
  - 父包装程序业务对象的查询描述 - 指定业务对象处理程序的包和类名，以便连接器可以确定适当的业务对象处理程序。SAPODA 将以下值插入每个标准查询描述的 AppSpecificInfo 属性中：AppSpecificInfo = sap.sapalemodule.VSapALEBOHandler。
  - 当使用包装程序业务对象来处理多个 IDoc 父业务对象时，您必须将业务对象处理程序的包和类名添加至多 IDoc 包装程序业务对象中每个查询描述的 AppSpecificInfo 属性。

  对于每个父包装程序业务对象，SAPODA 都会生成“创建”、“检索”、“更新”和“删除”查询描述。对于其中每个查询描述，它都会生成以下 AppSpecificInfo 值：

  ```
  sap.sapalemodule.VSapALEBOHandler,MsgType=;MsgCode=;MsgFunction=
  ```

AppSpecificInfo 属性：父包装程序查询描述

根据业务对象是表示应用程序事件还是表示服务调用请求，父包装程序业务对象查询描述的 AppSpecificInfo 属性的语法有所不同：

应用程序事件语法

[*BOHandler*],MsgType=*messageType*;MsgCode=[*messageCode*];MsgFunction=[*messageFunction*]

注： 连接器将控制记录中的值与查询描述的 AppSpecificInfo 属性中指定的值进行匹配来确定该查询描述。

服务调用请求语法

BOHandler[,MsgType=*messageType*;MsgCode=[*messageCode*];MsgFunction=[*messageFunction*]]

语法说明

BOHandler 指定请求处理业务对象处理程序；其值缺省为 sap.sapalemodule.VSapALEBOHandler

MsgType 指定已为 ALE 中的 IDoc 配置的消息类型

MsgCode 指定已为 ALE 中的 IDoc 配置的消息代码；仅当 MsgType 未唯一地标识查询描述时连接器才需要值；但是，如果 ALE 配置需要，则应指定一个值。

MsgFunction 指定已为 ALE 中的 IDoc 配置的消息功能；仅当 MsgType 和 MsgCode 未唯一地标识查询描述时连接器才需要值；但是，如果 ALE 配置需要，则应指定一个值。

AppSpecificInfo 属性: 父包装程序元查询描述

在父包装程序业务对象的 AleOutboundVerbs 查询描述的 AppSpecificInfo 属性中，指定连接器为了进行应用程序事件处理而应支持的那些查询描述，用逗号分隔各个查询描述。

重要提示: SAPODA 为“创建”、“检索”、“更新”和“删除”查询描述生成值。在生成了定义之后，您必须手工删除不想要连接器支持的那些查询描述。

以下示例指示连接器支持用于处理应用程序事件的“创建”和“更新”查询描述:

```
[Verb]
Name = AleOutboundVerbs
AppSpecificInfo = Create, Update
[End]
```

处理多个具有包装程序业务对象的 IDoc

注: 本节仅适用于服务调用请求处理。

当处理多个 IDoc 时，ALE 模块需要一个包装程序业务对象作为顶级业务对象。多个 IDoc 包装程序业务对象包含一个表示一组 IDoc 父包装程序业务对象的属性。

对于每个父包装程序业务对象，SAPODA 都会生成“创建”、“检索”、“更新”和“删除”查询描述。对于其中每个查询描述，它都会生成以下 AppSpecificInfo 值:

```
sap.sapalemodule.VSapALEBOHandler,MsgType=;MsgCode=;MsgFunction=
```

图 54 举例说明了顶级包装程序对象与其子 IDoc 业务对象之间的关系。

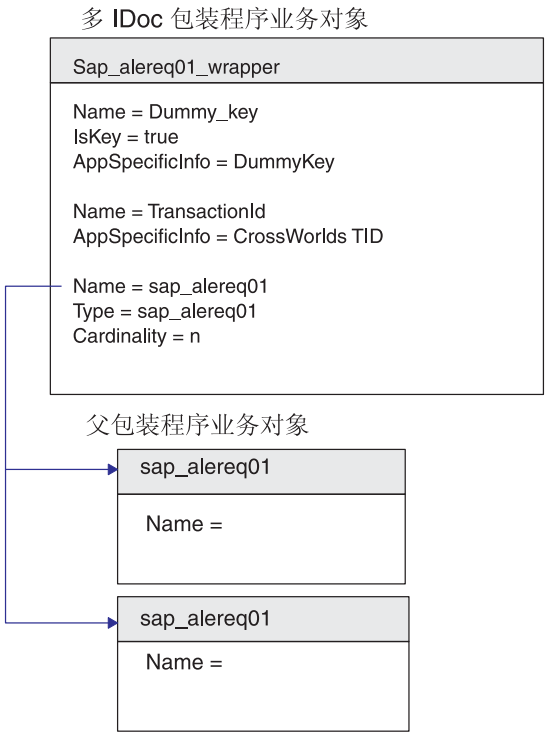


图 54. 包含子业务对象的包装程序业务对象

多 IDoc 包装程序对象示例

以下是一个多 IDoc 包装程序业务对象的样本定义:

```
[BusinessObjectDefinition]
Name = sap_alereq01_wrapper
Version = 1.0.0
AppSpecificInfo =

[Attribute]
Name = Dummy_key
Type = String
Cardinality = 1
MaxLength = 1
IsKey = true
IsForeignKey = false
IsRequired = true
AppSpecificInfo = DummyKey
DefaultValue =
[End]

[Attribute]
Name = TransactionId
Type = String
Cardinality = 1
MaxLength = 1
IsKey = false
IsForeignKey = false
IsRequired = false
AppSpecificInfo = CrossWorlds TID
DefaultValue =
[End]
```

```
[Attribute]
Name = sap_alereq01
Type = sap_alereq01
ContainedObjectVersion = 1.0.0
Relationship = Containment
Cardinality = n
MaxLength = 255
IsKey = false
IsForeignKey = false
IsRequired = false
AppSpecificInfo =
DefaultValue =
[End]

[Verb]
Name = Create
AppSpecificInfo = sap.sapalemodule.VSapALEBOHandler,MsgType=;MsgCode=;MsgFunction=
[End]

[Verb]
Name = Retrieve
AppSpecificInfo = sap.sapalemodule.VSapALEBOHandler,MsgType=;MsgCode=;MsgFunction=
[End]

[Verb]
Name = Update
AppSpecificInfo = sap.sapalemodule.VSapALEBOHandler,MsgType=;MsgCode=;MsgFunction=
[End]

[Verb]
Name = Delete
AppSpecificInfo = sap.sapalemodule.VSapALEBOHandler,MsgType=;MsgCode=;MsgFunction=
[End]
```

多 IDoc 包装程序: 表示子业务对象的属性

表 28 列示并描述了表示多 IDoc 包装程序业务对象中子业务对象的属性的特性。

表 28. 多 IDoc 包装程序: 表示子业务对象的属性

属性名	描述
Name	将其值设置为由 SAPODA 生成的父业务对象的名称。
Type	将其值设置为由 SAPODA 生成的父业务对象的名称。
ContainedObjectVersion	将其值设置为 1.0.0。
Relationship	子业务对象由父业务对象包含，因此，其值为 containment。
IsKey	将其值设置为 false。
IsForeignKey	将其值设置为 false。
IsRequired	将其值设置为 false。
AppSpecificInfo	此属性不用于表示 ALE 模块中子业务对象的属性。
Cardinality	将表示 IDoc 父业务对象的顶级包装程序业务对象中的属性值设置为基数 n。

第 4 部分 RFC 服务器模块

第 13 章 RFC 服务器模块概述

本章介绍 IBM WebSphere Business Integration mySAP.com 适配器（SAP R/3 V3.x）的 RFC 服务器模块。RFC 服务器模块使集成代理程序能够从支持 RFC 调用的 SAP R/3 V3.1H 或 3.1I 应用程序检索业务对象。它支持所有使用 RFC 支持功能的 SAP 应用程序（通过充当那些应用程序的服务器）。

RFC 服务器模块组件

RFC 服务器模块是用 Java 语言编写的连接器模块，它直接支持来自 SAP 应用程序的 RFC 调用。它通过实现 VisionConnectorAgent 类来扩展可视连接器框架。RFC 服务器模块使用以 Java 语言和 C 语言编写的 SAP RFC 库，该库使外部程序能够与 SAP 应用程序通信。

图 55 举例说明了 RFC 服务器模块的完整体系结构。RFC 服务器模块由连接器框架、RFC 服务器的特定于应用程序的连接器组件、特定于 RFC 服务器的业务对象处理程序、侦听器线程和 SAP RFC 库组成。

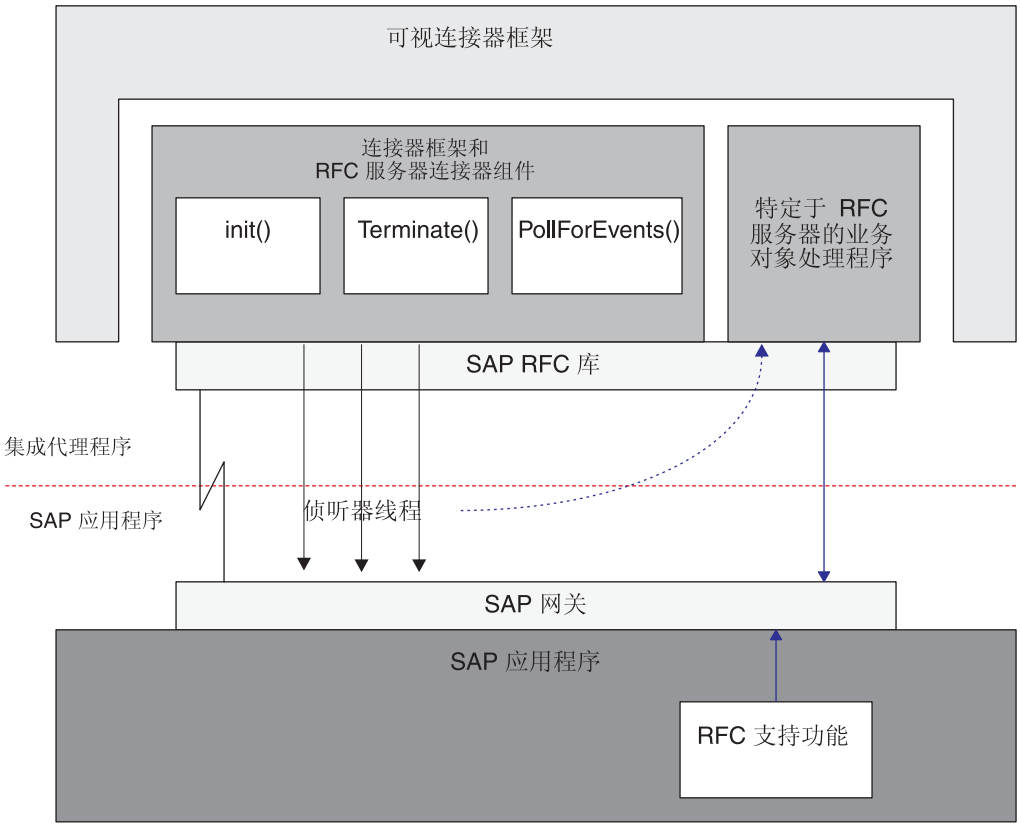


图 55. RFC 服务器模块体系结构

RFC 服务器模块组件:

- 衍生侦听器线程，这些线程使用 SAP RFC 库和 SAP 网关打开指向 SAP 应用程序的句柄。每个侦听器线程打开单个 SAP 应用程序的句柄。
- 处理来自 SAP 应用程序中 RFC 支持功能的请求。
- 终止与 SAP 应用程序的连接。

侦听器线程

侦听器线程处理 RFC 服务器模块与 SAP 应用程序之间的所有 RFC 调用。当连接器启动时，它衍生许多侦听器线程（该数目是可配置的）。每个侦听器线程都打开一个指向 SAP 网关的句柄。

侦听器线程:

- 使用程序标识向 SAP 网关注册。
- 对 SAP 网关标识它们支持的 RFC 支持功能。
- 使用第一个可用线程来从受支持的 RFC 支持功能选取事件。
- 基于相应的业务对象中服务器查询描述来实例化特定于 RFC 服务器的业务对象处理程序，然后从 SAP 网关检索事件数据。
- 用 RFC 事件数据填充业务对象，然后将返回的业务对象数据转换为 RFC 事件数据。
- 通过 SAP 网关返回对 RFC 支持功能的响应。

注: 线程以同步方式连续侦听事件（这些事件来自它支持的 RFC 支持功能）。

特定于 RFC 服务器的业务对象处理程序

特定于 RFC 服务器的业务对象处理程序是 SAP 应用程序中每个 RFC 支持功能所独有的。每个业务对象处理程序由侦听器线程实例化并调用相关的业务对象。

因为 RFC 服务器模块充当 SAP 应用程序的服务器，所以它将来自 SAP 应用程序的事件“推送”或发送至集成代理程序。此行为与其它模块完全不同，其它模块轮询应用程序以获取事件。由于此差别，所以特定于 RFC 服务器的业务对象处理程序执行与其它业务对象处理程序不同的任务。

一旦实例化，特定于 RFC 服务器的业务对象处理程序就会执行以下操作:

- 检索 RFC 事件数据并填充相关的 SAP 的 WebSphere 业务对象。
- 将该业务对象传递至集成代理程序，并相应地接收一个业务对象。

  业务对象处理程序使用业务对象的特定于应用程序的服务器查询描述信息来确定哪个协作应处理业务对象数据。

  – 当 InterChange Server（ICS）是集成代理程序时，业务对象的服务器查询描述必须指定一个有效协作。因为协作不能显式地预订推送至连接器的事件，所以特定于 RFC 服务器的业务对象处理程序必须确定适当的协作，然后实例化该协作。
  – 当消息代理是集成代理程序时，业务对象的服务器查询描述必须包含哑元值。
- 将返回的业务对象数据重新转换为 RFC 事件数据。
- 将 RFC 事件数据返回至 SAP 应用程序。

RFC 服务器模块的工作方式

RFC 服务器模块实现 init()、terminate()、pollForEvents() 和 process() 方法。

本节描述:

- 『初始化和终止』
- 第 147 页的『业务对象处理』
- 第 148 页的『支持 RFC 支持功能』

初始化和终止

init() 方法创建一个主线程，该线程衍生许多侦听器线程（该数目是可配置的），这些线程打开指向 SAP 网关的句柄。如果连接器未能初始化，则它使用 terminate() 方法来终止。连接器通过断开与 SAP 网关的连接来终止。

在初始化过程期间，RFC 服务器模块使用指定的程序标识向 SAP 网关注册。必须使用 RfcProgramID 连接器配置属性来设置此程序标识，且必须在 SAP 应用程序中将它设置为 TCP/IP 端口。有关设置 TCP/IP 端口的更多信息，请参阅第 159 页的『向 SAP 网关注册 RFC 服务器模块』。

业务对象处理

RFC 服务器模块的 WebSphere 业务对象的所有处理由 SAP 应用程序中 RFC 支持功能启动。在 RFC 服务器模块中，特定于 RFC 服务器的业务对象处理程序仅支持一个 RFC 支持功能；因此，对于 SAP 应用程序中每个受支持的功能，您必须具有一个特定于 RFC 服务器的相关业务对象处理程序。另外，对于每个特定于 RFC 服务器的业务对象处理程序，您还必须具有一个相关业务对象。

图 56 举例说明了 RFC 服务器模块的业务对象处理。

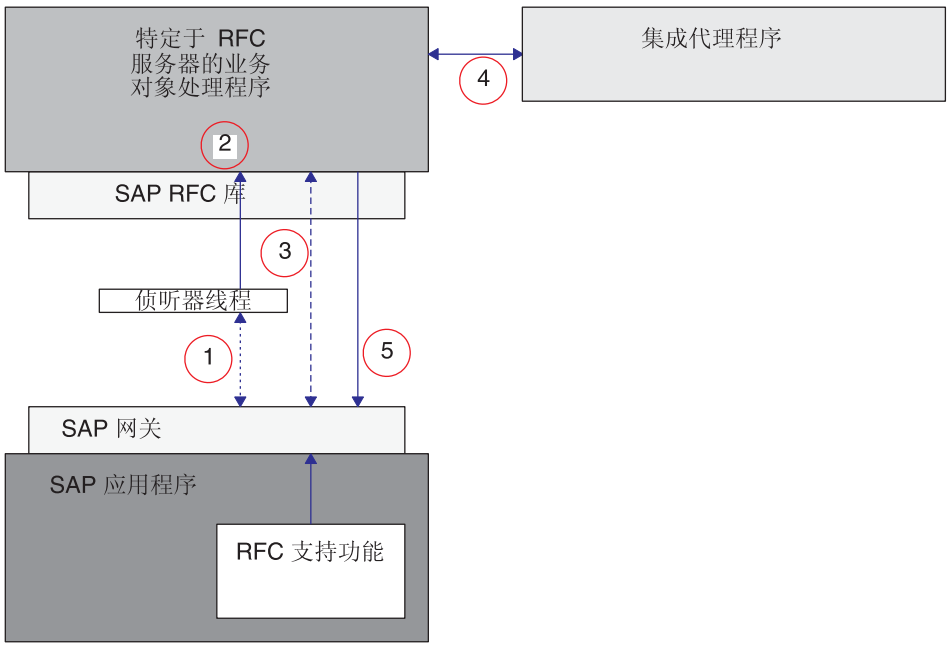


图 56. 业务对象处理

RFC 服务器模块的业务对象处理以下列方式执行:

1. 侦听器线程从 SAP 网关选取预订的事件，并使相应 RFC 支持功能的名称与特定于 RFC 服务器的业务对象处理程序匹配。
2. 侦听器线程基于来自 SAP 网关上 RFC 事件的数据来实例化特定于 RFC 服务器的适当业务对象处理程序，然后创建相应业务对象的实例。
3. 特定于 RFC 服务器的业务对象处理程序从 SAP 网关检索 RFC 接口数据，并填充 SAP 的 WebSphere 业务对象。
4. 特定于 RFC 服务器的业务对象处理程序将业务对象传递至集成代理程序。
5. 业务对象处理程序接收来自集成代理程序的返回业务对象，将它重新转换为 RFC 接口，然后将它返回至 SAP 网关。

RFC 服务器模块使用 SAP 网关来维护事件的处理顺序并维护事件的状态。因为侦听器线程执行同步调用，所以必须将事件返回至 SAP 网关后才能认为已成功处理该事件。

注: 如果 RFC 支持功能模块具有返回结构或返回表，则连接器检查消息类型 A（异常终止）和 E（错误）来确定是否成功处理了事件。消息类型 A 或 E 指示未能处理事件。如果 RFC 支持功能模块不具有返回结构或返回表，则您必须实现您自己的错误处理。

支持 RFC 支持功能

IBM WebSphere Business Integration mySAP.com 适配器包括一个工具，即 SAPODA，它基于 RFC 支持功能来生成业务对象定义。SAPODA 解释 RFC 支持功能的接口，将其接口参数映射至业务对象属性，并添加每个属性的特定于应用程序的信息。

对于每个业务对象定义，您必须生成一个特定于 RFC 服务器的相关业务对象处理程序，该处理程序调用相应的业务对象。有关开发业务对象和特定于 RFC 服务器的业务对象处理程序的更多信息，请参阅第 149 页的第 14 章，『为 RFC 服务器模块开发业务对象』。

注: 某些 RFC 支持功能不具有与 WebSphere 业务对象中简单属性对应的单一字段参数。连接器要求每个顶级业务对象都具有一个充当键属性的简单属性。因此，当从 RFC 支持功能生成业务对象和业务对象处理程序而没有单一字段参数时，SAPODA 在顶级业务对象中创建一个名为 Dummy_key 的键属性，将它标记为键属性，并添加 dummy_key 作为此属性的特定于应用程序的信息。Dummy_key 为连接器提供一个键属性，以便它可以处理该业务对象。但是，连接器在修改应用程序数据时会忽略 Dummy_key 属性的值。

第 14 章 为 RFC 服务器模块开发业务对象

本章描述 RFC 服务器模块所需要的业务对象和业务对象处理程序。它提供背景信息并讨论业务对象生成实用程序 SAPODA 如何生成定义。本章假定您熟悉连接器如何处理业务对象。有关 RFC 服务器模块中业务对象处理的更多信息，请参阅第 145 页的第 13 章，『RFC 服务器模块概述』。

注: 您一旦已创建业务对象和特定于 RFC 服务器的业务对象处理程序，您就必须确保向 SAP 网关注册该 RFC 服务器模块。有关更多信息，请参阅第 159 页的『向 SAP 网关注册 RFC 服务器模块』。

背景信息

RFC 服务器模块的业务对象开发包括为您要支持的每个 RFC 支持功能创建特定于应用程序的业务对象定义和特定于 RFC 的相关业务对象处理程序。因为 SAPODA 在为每个业务对象生成定义时使用 SAP 应用程序的本机定义作为模板，所以建议您使用 SAPODA 来生成这些定义。

注: SAP 支持许多方法，可以将这些方法映射至连接器支持的标准查询描述（创建、更新、删除和检索）。您可以开发业务对象和特定于 RFC 服务器的业务对象处理程序以支持由 RFC 支持功能使用的任何方法。

业务对象命名约定

RFC 支持功能接口由导入参数、导出参数和表参数组成，其中:

- 导入参数将传递至 RFC 支持功能。
- 将从 RFC 支持功能返回导出参数。
- 将向任一方向传递表参数。

某些 RFC 支持功能可能并不具有所有类型的参数。例如， RFC 支持功能可能仅具有导入参数和表参数。

SAPODA 自动将 RFC 支持功能的导入参数、导出参数和表参数映射至 IBM WebSphere 属性，如表 29 中所述。

表 29. 命名约定: SAP 的 WebSphere 业务对象

业务对象	RFC 支持功能接口
顶级业务对象	BOprefix_FunctionName**注意**: 本章中的例子使用 SAP_ 或 sap_ 作为业务对象前缀。您在创建业务对象定义时，可以指定自己的有意义前缀。
属性	字段描述
子业务对象	BOprefix_FunctionParameterName

SAPODA 保证业务对象定义中的所有属性名都是唯一的。如果 RFC 支持功能有多个参数具有相同字段描述，则 SAPODA 将计数器作为后缀添加至生成的属性名。

在生成业务对象定义之后，您可以在任何时间修改属性名。但是，当修改属性名时，确保不要修改特定于应用程序的信息。连接器使用此信息来标识该属性所对应的 RFC 支持功能的参数。有关特定于应用程序的信息的详情，请参阅第 154 页的『属性的 AppSpecificInfo』。

业务对象结构

连接器使用特定于 RFC 服务器的业务对象处理程序来将每个业务对象属性映射至 RFC 支持功能的参数。连接器、每个业务对象和每个特定于 RFC 服务器的业务对象处理程序都是元数据驱动的。对于每个业务对象和业务对象处理程序的元数据中提供的特定于应用程序的信息，它允许您为新业务对象及其处理程序添加连接器支持而不必修改连接器代码。实际上是:

- 连接器使用特定于应用程序的顶级业务对象查询描述信息，来实例化特定于 RFC 服务器的适当业务对象处理程序。

  重要提示: RFC 服务器模块与其它模块不同，它不轮询 SAP 来获取事件。而是 SAP 将事件数据推送至连接器。因为此模块不使用标准轮询过程，所以特定于 RFC 服务器的业务对象处理程序检查表示事件的每个业务对象，以获取将处理该事件的协作的名称。当 InterChange Server（ICS）是集成代理程序时，特定于 RFC 服务器的业务对象处理程序使用获得的值来实例化适当的协作。当消息代理是集成代理程序时，必须为特定于 RFC 服务器的业务对象处理程序提供哑元值才能成功处理事件。
- 业务对象处理程序使用每个业务对象的特定于应用程序的属性信息来在每个属性及其参数之间进行映射。

每个特定于 RFC 服务器的业务对象处理程序都支持业务对象之间的单基数和多基数关系。

基于 RFC 支持功能的 WebSphere 业务对象不能包含多于两个级别的层次结构。因此，所有简单参数都对应于顶级业务对象的属性，并且结构参数和表参数对应于子业务对象。

表 30. RFC 支持功能和业务对象之间的对应项

RFC 支持功能接口参数	SAP 的 WebSphere 业务对象
简单字段	顶级业务对象的属性
结构	单基数子业务对象
表	多基数子业务对象

注: 导入参数和导出参数可以是简单字段或结构参数。

图 57 举例说明了 WebSphere 业务对象和 RFC 支持功能（BAPI）之间的关联。该图举例说明了用户定义的 sap_bapi_po_create 业务对象的片段，该业务对象对应于 BAPI_PO_CREATE BAPI。

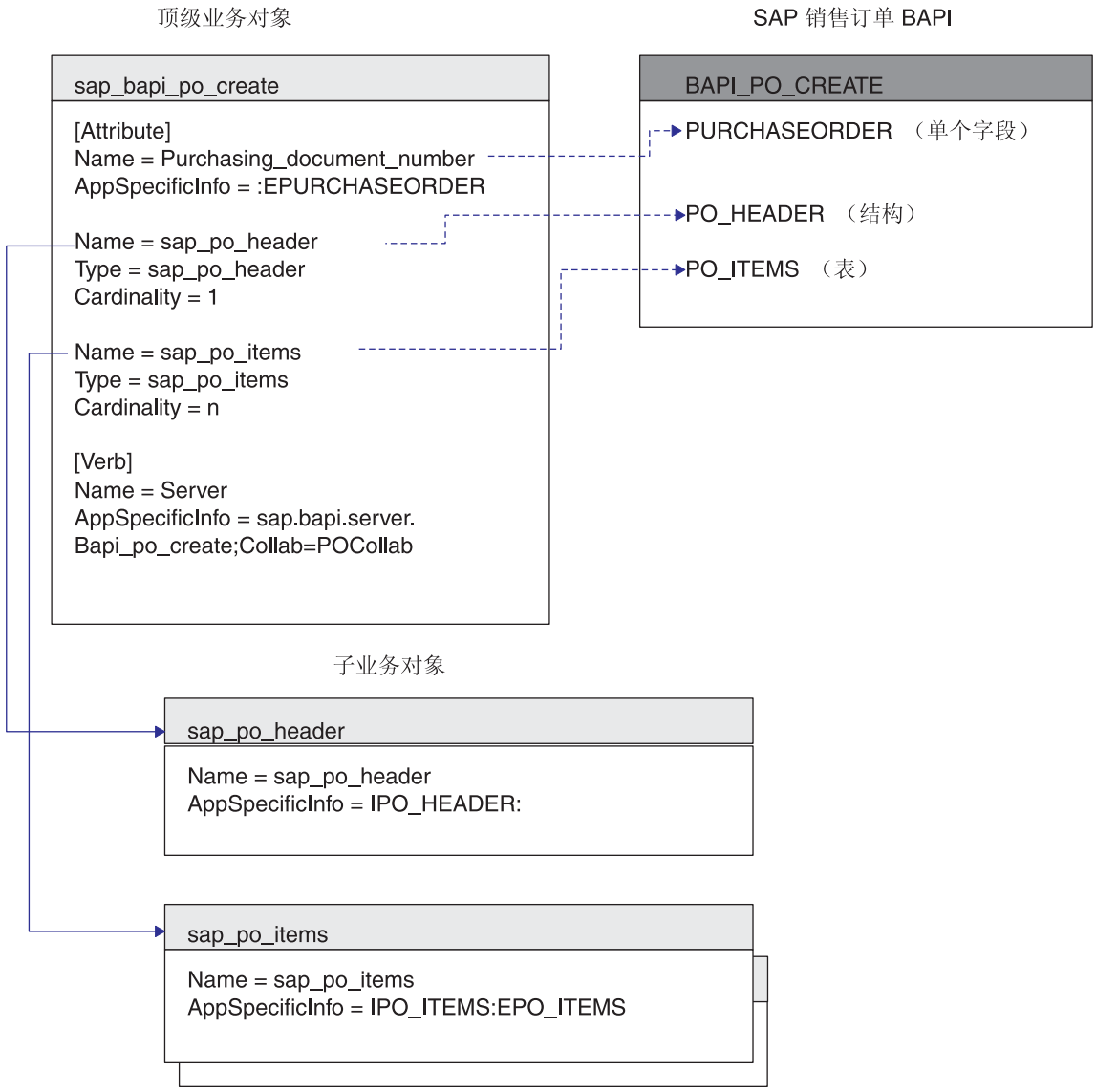


图 57. 业务对象和 BAPI 之间的映射

受支持的查询描述

RFC 服务器模块支持由 WebSphere Business Integration 系统使用的标准查询描述（创建、更新、删除和检索）。对于每个受支持的查询描述， RFC 支持功能可以具有一个相关的方法。大多数 RFC 支持功能支持下列其中一项操作: 创建、检索、更新和删除。

业务对象属性特性

顶级业务对象属性的特性根据属性是表示简单值还是一个子业务对象或一组子业务对象而不同。

- 表 31 列示并描述顶级业务对象的简单属性的特性。
- 表 32 列示并描述表示一个子业务对象或一组子业务对象的属性。

SAPODA 生成属性特性，如每个表中所述。

表 31. 简单属性：顶级业务对象

属性名	描述
Name	源自 RFC 支持功能参数的描述。SAPODA 用下划线替换特殊字符（如句点、斜杠和空格）。
Type	指定数据的类型。SAPODA 将其值设置为 `String`。
MaxLength	指定 RFC 支持功能参数的字段长度。
IsKey	指定属性是否是键。业务对象的第一个简单属性缺省为键属性。连接器不支持使用表示一个子业务对象或一组子业务对象的属性作为键属性。因此，如果该功能仅提供结构参数和表参数，则您必须插入一个简单属性作为第一个属性。SAPODA 插入 Dummy_key 属性作为第一个属性，将它标记为键属性，然后设置适当的值。不要修改这些值。有关更多信息，请参阅第 88 页的『支持 BAPI』。
IsForeignKey	SAPODA 将其值设置为 `false`。
IsRequired	指定属性是否必须包含值。SAPODA 将其值设置为 `false`。
AppSpecificInfo	包含 RFC 支持功能参数的名称，该名称对应于相关属性。格式为：`I`*`RFCFunctionParameterName`*`:E`*`RFCFunctionParameterName`*。有关特定于应用程序的信息的详情，请参阅第 153 页的『特定于应用程序的业务对象信息』。
缺省值	指定在没有运行时值时要赋予此属性的值。SAPODA 没有为此属性设置值。

表 32 列示并描述表示一个子业务对象或一组子业务对象的属性。SAPODA 生成下表中描述的属性。

表 32. 表示一个或多个子代的属性的特性

属性名	描述
Name	此值是结构参数或表参数的名称。格式为：`BOprefix_FunctionParameterName`
Type	其值是子业务对象的类型；即类型为 `BOprefix_FunctionParameterName`。
ContainedObjectVersion	SAPODA 将其值设置为 `1.0.0`。
Relationship	SAPODA 将其值设置为 `containment`。
IsKey	SAPODA 将其值设置为 `false`。
IsForeignKey	SAPODA 将其值设置为 `false`。
IsRequired	指定属性是否必须包含值。SAPODA 将其值设置为 `false`。
AppSpecificInfo	包含对应于相关属性的 BAPI 参数的名称。格式为：`I`*`FieldName`*`:E`*`FieldName`*。有关特定于应用程序的信息的详情，请参阅第 153 页的『特定于应用程序的业务对象信息』。
Cardinality	结构参数具有单基数（1），而表参数具有多基数（n）。

初始化属性值

SAP 中的每个字段都具有初始值，它们列示在表 33 中。当连接器接收到事件时，特定于 RFC 服务器的业务对象处理程序将这些值从每个 SAP 字段移至其相应的业务对象属性。业务对象处理程序保留来自 SAP 的初始值，但字符数据类型是一个例外。业务对象处理程序将 SAP 字段中的空格转换为业务对象属性中的 CxIgnore。如果您想要将任何其它值转换为 CxIgnore，则创建业务对象的组件必须执行该转换。例如，当 ICS 是集成代理程序时，修改映射以处理此转换。

表 33. SAP 中的初始字段值

数据类型	描述	由业务对象处理程序设置的初始值
C	字符	空格
N	数字字符串	*000...*
D	日期（YYYMMDD）	00000000
T	时间（HHMMSS）	000000
X	字节（十六进制）	X00
I	整数	0
P	压缩数字	0
F	浮点数字	0.0

特定于应用程序的业务对象信息

业务对象定义中特定于应用程序的信息对 RFC 服务器模块提供关于如何处理业务对象的应用程序相关指示信息。将在业务对象级别和属性级别（对于简单属性和表示子业务对象或一组子业务对象的属性）对查询描述指定这些指示信息。

顶级业务对象的服务器查询描述的 AppSpecificInfo

连接器使用顶级业务对象中特定于应用程序的服务器查询描述信息的值，来调用特定于 RFC 服务器的适当业务对象处理程序并确定用于事件处理的目标协作。服务器查询描述的 AppSpecificInfo 属性的值指定:

- 特定于 RFC 服务器的业务对象处理程序的包和类名
- 目标协作

格式如下:

```
AppSpecificInfo = bapi.server.BOHandler;Collab=CollaborationName
```

其中 BOHandler 是类名，*CollaborationName* 是目标协作的名称。

SAPODA 自动在顶级业务对象中添加服务器查询描述的特定于应用程序的信息。对于业务对象处理程序的类名的值，它使用 RFC 支持功能的名称。它没有为协作名参数提供值。因此，您必须手工添加协作的名称。

重要提示: 当消息代理是集成代理程序时，必须为协作名参数提供哑元值。特定于 RFC 服务器的业务对象处理程序需要此参数的值才能成功处理事件。

注: SAP 的 WebSphere 业务对象和特定于 RFC 服务器的业务对象处理程序之间存在一个一对一关系。业务对象处理程序类文件必须在 \connectors\SAP\bapi\server 目录中存在。

重要提示: 您必须在业务对象处理程序名称前面包括值 server，来标识特定于 RFC 服务器的业务对象处理程序充当服务器。

例如，如果您要支持 RFC 支持功能 BAPI_PO_CREATE，且目标协作称为 POCollab，则特定于应用程序的查询描述信息如下:

```
AppSpecificInfo =bapi.server.Bapi_po_create;Collab=POCollab
```

属性的 AppSpecificInfo

连接器使用特定于应用程序的属性信息的值来确定要使用哪些导入参数、导出参数和表参数。此属性的值包含前缀 I（表示导入参数）或 E（表示导入参数）。该前缀指示属性值是用来将数据传递至 SAP 应用程序还是从 SAP 应用程序中传递出来。

因为结构参数可以是导入或导出，所以它们在参数值前面使用 I 或 E。因为表参数可以将数据传递至 BAPI 并从 RFC 支持功能返回数据，所以它们可以同时具有 I 和 E 参数值。

重要提示: 当使用 I 和 E 指定参数值时，应始终使用冒号（:）分隔符。如果仅指定导入值，则该值后面必须有冒号。如果仅指定导出值，则该值前面必须有冒号。如果同时指定两个值，则冒号在导入值的后面并在导出值的前面。

图 58 举例说明了业务对象和名为 BAPI_EXAMPLE 的 RFC 支持功能示例之间的映射。在此示例中，简单属性（Attribute_1、Attribute_2 和 Attribute_3）仅指定导入参数或导出参数。表示子业务对象（Child_1）的属性映射至导出结构参数。表示一组子业务对象（Child_2）的属性映射至表参数。

每个子业务对象都具有一个映射至相应结构或表的字段的简单属性（分别为 Attribute_11 和 aAttribute_14）。您可以通过查看 BAPI 的详细信息来找到这些字段。

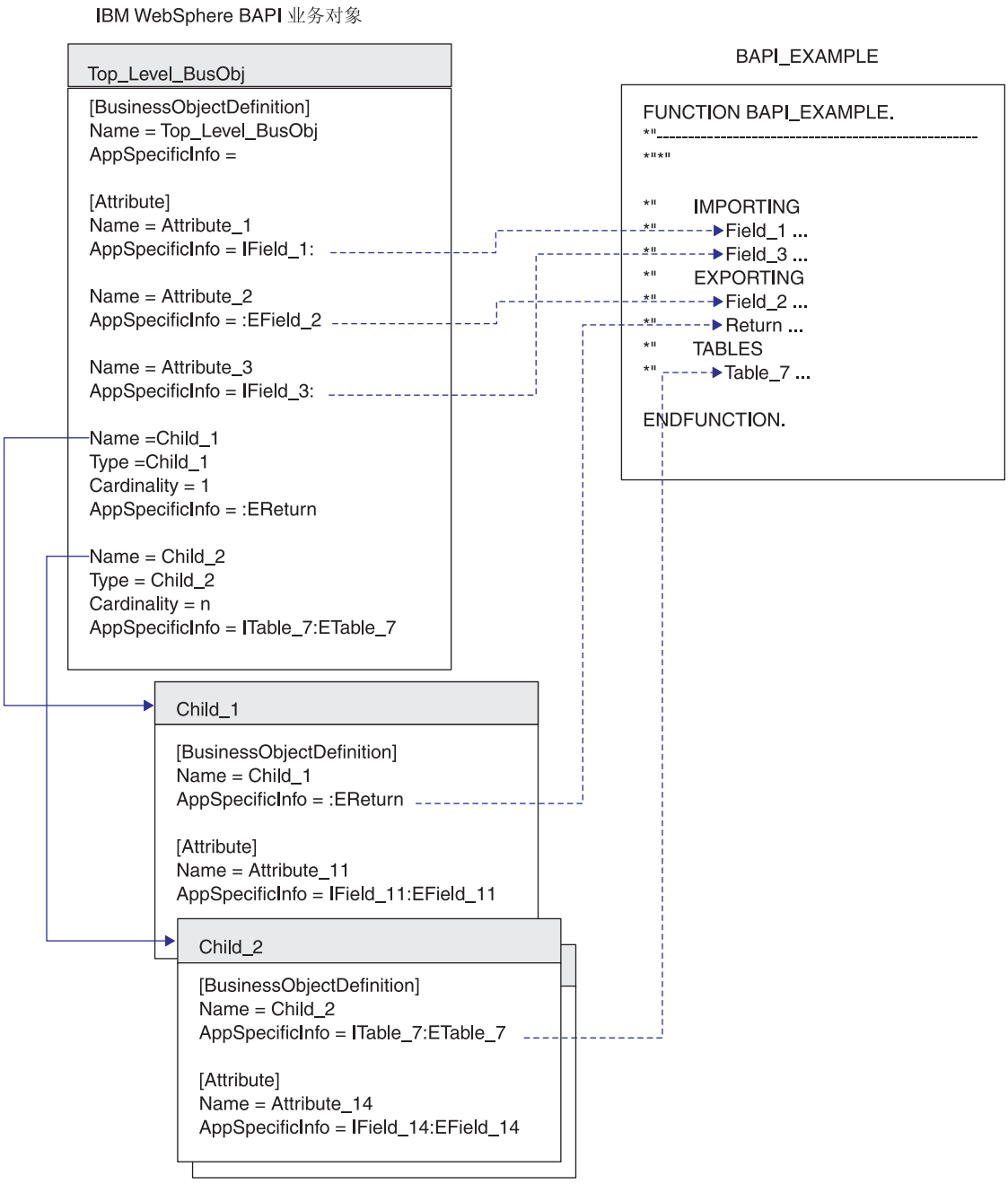


图 58. 业务对象和示例 BAPI 之间的映射

表 34 标识特定种类属性的特定于应用程序的信息格式。

表 34. 特定属性类型的 AppSpecificInfo 格式

AppSpecificInfo 格式	**属性类型**
I*ParameterName*:E*ParameterName*	简单
I*TableName*:E*TableName*	表示映射至表参数的子业务对象
I*StructureName*:E*StructureName*	表示映射至结构参数的子业务对象
I*FieldName*:E*FieldName*	表示映射至表参数或结构参数中字段的子业务对象的属性

SAPODA 自动为您的业务对象定义生成特定于应用程序的适当信息。建议不要修改生成的特定于应用程序的信息的参数名。

使用生成的业务对象和业务对象处理程序

使用 SAPODA 来为您要支持的每个 RFC 支持功能生成特定于 RFC 支持功能的业务对象定义和特定于 RFC 服务器的业务对象处理程序。您可以稍加修改就使用生成的文件。

您唯一必须进行的编辑，就是在服务器查询描述的特定于应用程序的查询描述信息中指定目标协作的名称。

- 因为协作不能显式地预订推送至连接器的事件，所以当 InterChange server（ICS）是集成代理程序时此信息是必需的。因此，特定于 RFC 服务器的业务对象处理程序必须根据业务对象的元数据确定适当的目标协作，然后实例化该协作。
- 当消息代理是集成代理程序时，需要为特定于 RFC 服务器的业务对象处理程序提供哑元值以正确处理事件。

重要提示: 如果您正在使用的 RFC 支持功能不包含简单字段属性，且 SAPODA 已创建 Dummy_key 属性作为键属性，则不要修改此属性的值。

在生成业务对象定义及其相应的特定于 RFC 服务器的业务对象处理程序之后，您必须将业务对象定义添加至 WebSphere Business Integration 系统的运行时环境。

- 使用业务对象设计器将业务对象定义装入资源库。

  注: 或者，如果 InterChange Server（ICS）是集成代理程序，则您可以使用 `repos_copy` 命令来将该定义装入资源库。
- 使用系统命令将特定于 RFC 服务器的业务对象处理程序文件复制到产品目录下的以下目录:

  ```
  \connectors\SAP\bapi\server
  ```

特定于 RFC 服务器的业务对象处理程序文件为:

- *RFC-EnabledFunctionName*`.java`
- *RFC-EnabledFunctionName*`.class`

例如，如果给出 RFC 支持功能 `BAPI_PO_CREATE` 和用户定义的前缀 `sap_`，则 SAPODA 生成以下各项:

- `sap_bapi_po_create`（包括所有子业务对象的业务对象定义）
- `Bapi_po_create.java`
- `Bapi_po_create.class`

重要提示: 您可以修改生成的业务对象的名称及其子业务对象的名称。为此，您必须作为文本文件编辑该定义，而不是在业务对象设计器进行编辑。如果您更改业务对象的名称，则确保也修改所有引用到更改的名称的地方。并且，如果您为业务对象处理程序修改生成的 `.class` 文件的名称，则您必须对相关业务对象的特定于应用程序的服务器查询描述信息保持这些更改。

技巧和窍门

以下是用于开发业务对象和特定于 RFC 服务器的业务对象处理程序的技巧和窍门:

- 『多个业务对象包含相同的返回业务对象』
- 『生成的业务对象定义包含不必要的属性和子业务对象』
- 『生成的业务对象名太长或不符合命名约定』
- 第 158 页的『为表参数生成的 AppSpecificInfo 指定不必要的参数』

多个业务对象包含相同的返回业务对象

大多数 RFC 支持功能将相同名称用于返回对象。当 SAPODA 生成业务对象定义时，它创建一个子业务对象来表示此返回对象。如果多个业务对象定义包含同一个指定的子业务对象，则您只需一次将该子业务对象的定义添加至资源库。

要使多个业务对象能够包含返回业务对象，您必须修改返回业务对象的名称，使其对于每个业务对象都是唯一的。

要重命名返回业务对象，修改包含它的每个业务对象定义的定义。子业务对象的定义与其父代的定义包含在相同的定义文件中。

要重命名子代，执行以下操作:

1. 在文本编辑器中打开顶级业务对象的定义文件。
2. 找到 `BOprefix_return` 子业务对象的定义。
3. 将该子代的名称更改为唯一的。例如，将一个数字追加至文本（`sap_return_2`）。
4. 更改定义中的所有引用以引用新命名的子代。例如，更改每个表示子业务对象的属性的 Type 属性值。
5. 保存更改后的定义文件。
6. 使用业务对象设计器将新命名的子业务对象装入资源库。

   注: 或者，如果 ICS 是集成代理程序，则您可以使用 `repos_copy` 命令将该定义装入资源库。

生成的业务对象定义包含不必要的属性和子业务对象

SAPODA 解释所有 RFC 支持功能接口参数，并且对于每个参数，它都会创建一个相应的 WebSphere 业务对象属性或子业务对象。要提高业务对象处理的性能，从业务对象定义除去所有不需要的属性和业务对象。

注: SAPODA 便于在生成定义之前以图形方式除去所有可选属性和子业务对象。有关更多信息，请参阅第 52 页的『提供其它信息』。

要提高业务对象处理的性能，您还可以从特定于应用程序的信息除去所有不需要的导入和导出表参数值。

在生成定义之后，如果您需要进行其它更改，则可以使用业务对象设计器来手工编辑业务对象定义。但是，请注意仅除去以后绝对不会使用的属性。

生成的业务对象名太长或不符合命名约定

SAPODA 使用 RFC 支持功能模块的名称来命名生成的业务对象。您可以使用文本编辑器来修改业务对象的名称。

重要提示: 如果您更改该名称，则确保您也修改该名称的所有引用。但是，不要修改生成的特定于应用程序的信息的参数名。

要更改生成的业务对象的名称:

1. 将定义保存至文件。
2. 使用文本编辑器来缩短或更改该名称。
3. 使用业务对象设计器将新命名的子业务对象复制到资源库。

   注: 或者，如果 ICS 是集成代理程序，则您可以使用 `repos_copy` 命令将该定义装入资源库。

为表参数生成的 AppSpecificInfo 指定不必要的参数

表参数可以同时是导入参数和导出参数。如果您不要求导入或导出表参数的值，则您可以从特定于应用程序的信息中除去该参数。

例如，对于创建操作，如果您在创建操作完成之后不需要从 SAP 应用程序返回表数据，则您可以除去导出参数值（如 `E` *table name*）。

对于检索操作，您不需要指定任何导入表参数。因此，您可以除去导入参数值（如 `I` *table name*）。

注: 您必须从父代中某一表示子代的属性的 `AppSpecificInfo` 中以及业务对象级别为子业务对象的 `AppSpecificInfo` 中除去不需要的值。不要除去冒号（`:`）。

例如，要除去第 155 页的图 58 中的 `ETable_7` 导出参数，您应执行以下操作:

1. 在 Top_Level_BusObj 业务对象的 Child_2 属性中，将该属性的 AppSpecificInfo 值更改为:

   ```
   ITable_7:
   ```

2. 在业务对象级别为 Child_2 业务对象的 AppSpecificInfo 中，将其值更改为:

   ```
   ITable_7:
   ```

3. 以 Attribute_14 为例，在子业务对象的每个属性的 AppSpecificInfo 中将其值更改为:

   ```
   IField_14:
   ```

第 15 章 配置 RFC 服务器模块

- 『RFC 服务器模块目录和文件』
- 『RFC 服务器模块配置属性』
- 『向 SAP 网关注册 RFC 服务器模块』

本章描述 RFC 模块的配置并假定在安装 IBM WebSphere Business Integration mySAP.com 适配器（SAP R/3 V3.x）时安装了所有必要的文件。有关安装连接器的更多信息，请参阅第 37 页的第 4 章，『运行连接器』。

RFC 服务器模块目录和文件

RFC 服务器模块目录和文件包含在 \connectors\SAP\ 目录中。表 35 列示 RFC 服务器模块使用的目录和文件。

表 35. RFC 服务器模块目录和文件

目录 / 文件名	描述
\bapi\server	包含连接器的运行时文件的目录。必须将所有特定于 RFC 服务器的 BOHandler 类文件复制到此目录。
CWSAP.jar	连接器类文件

RFC 服务器模块配置属性

您必须配置 RFC 服务器模块，然后它才能开始运行。要配置 RFC 服务器模块，设置标准的和特定于连接器的连接器配置属性。有关配置连接器配置属性的更多信息，请参阅第 21 页的第 3 章，『配置连接器』和第 261 页的附录 D，『连接器的标准配置属性』。

向 SAP 网关注册 RFC 服务器模块

在初始化期间，RFC 服务器模块向 SAP 网关注册。它使用为特定于连接器的配置属性 RfcProgramId 设置的值。此值必须与 SAP 应用程序中设置的值匹配。您必须配置 SAP 应用程序，以便 RFC 服务器模块可以创建它的一个句柄。

要将 RFC 服务器模块注册为 RFC 目标位置:

1. 在 SAP 应用程序中，转至事务 SM59。
2. 展开 TCP/IP 连接目录。
3. 单击“创建”（F8）。
4. 在“RFC 目标位置”字段中，输入 RFC 目标系统的名称。建议您使用 RFCSERVER。
5. 将连接类型设置为 T（通过 TCP/IP 启动外部程序）。
6. 输入新 RFC 目标位置的描述，然后单击“保存”。
7. 单击“激活”类型的“注册”按钮。
8. 设置程序标识。建议您使用与 RFC 目标位置（RFCSERVER）相同的值，然后单击“输入”。

重要提示: 确保特定于连接器的配置属性 RfcProgramID 的值设置为与 SAP 应用程序中的“程序标识”值相同。如果该值不匹配，则业务对象处理将失败。

第 5 部分 分层动态检索模块

第 16 章 分层动态检索模块概述

本章描述 IBM WebSphere Business Integration mySAP.com 适配器的分层动态检索模块。分层动态检索模块处理分层业务对象或平面业务对象。为了处理这些请求，连接器从 SAP 应用程序版本 4.0、4.5 和 4.6 检索数据。

本章包含以下各节:

分层动态检索模块组件

分层动态检索模块是用 Java 语言编写的，它扩展可视连接器框架。因为该模块没有自己的特定于应用程序的组件，所以它使用 BAPI 的特定于应用程序的组件。因此，该模块包含连接器框架、BAPI 的特定于应用程序的组件、vDynRetBOH 业务对象处理程序和 SAP RFC 库。 SAP 以 Java 和 C 语言交付 RFC 库。Java 归档（JAR）文件将作为连接器交付并运行。

图 59 举例说明了分层动态检索模块的体系结构。

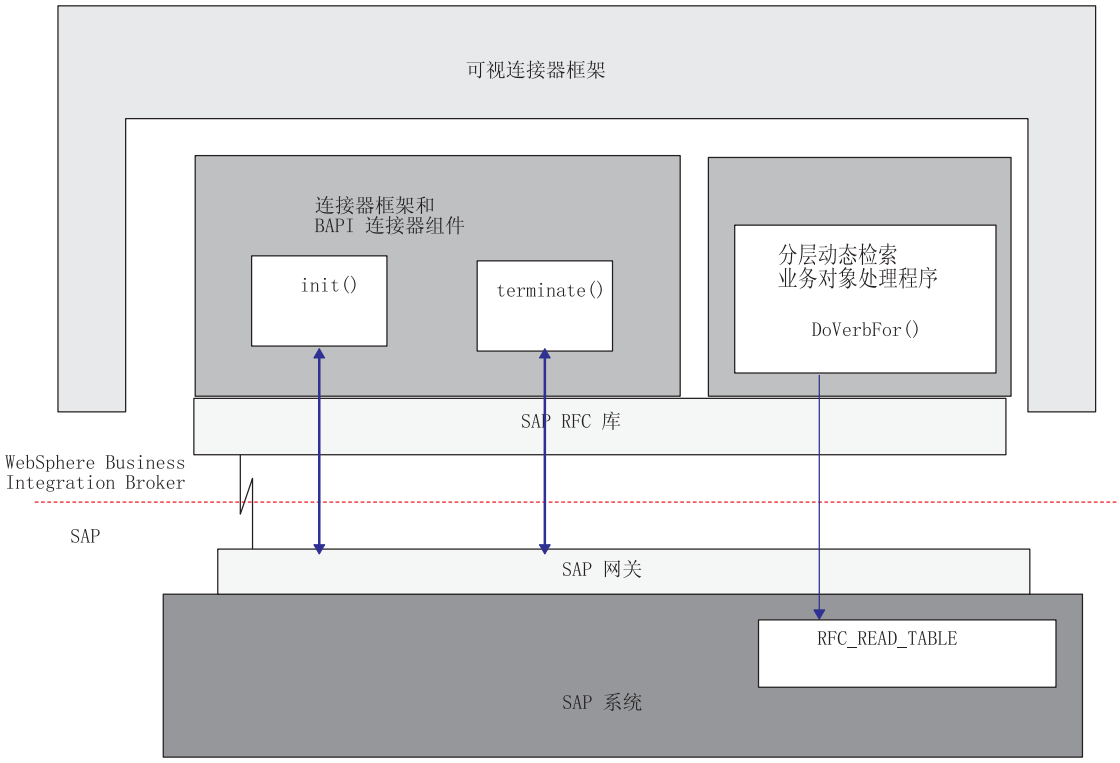


图 59. 分层动态检索模块体系结构

连接器的工作方式

连接器从业务对象中指定的元数据获取业务对象的处理信息，而不是从连接器内部硬编码的信息中获取处理信息。为了从业务对象获取处理信息，连接器作出关于以下各项的假设:

- 业务对象结构
- 父代和子代业务对象之间的关系
- 业务对象的可能数据库表示

有关信息，请参阅第 9 页的『处理业务对象』和第 167 页的第 18 章，『为分层动态检索模块开发业务对象』。

当连接器接收到来自要执行应用程序操作的集成代理程序的请求时，它从为顶级业务对象指定的查询描述中获取处理信息。

连接器递归地处理分层业务对象；即，它对每个子业务对象执行相同的步骤，直到它已处理所有个体业务对象。

注: **分层**业务对象这一术语指一个完整的业务对象，包括它在任何级别包含的所有子业务对象。**个体**业务对象这一术语指单个业务对象，独立于它可能包含或可能包含它的任何子业务对象。**顶级**业务对象这一术语指在层次结构的顶层的个体业务对象，它本身不具有父业务对象。

当集成代理程序发送具有“检索”查询描述的分层业务对象时，连接器尝试将业务对象返回至与该业务对象的当前数据库表示完全匹配的集成代理程序。换句话说，连接器返回的每个个体业务对象的每个简单属性的值与数据库中其相应字段的值匹配。而且，返回的每组业务对象中个体业务对象的数目都与该组业务对象的数据库中的子代数匹配（除非特定于应用程序的信息将这些子代限制于一个子集）。

为了执行这样的检索，连接器使用顶级业务对象中的主键值来递归地向下查找数据库中的相应数据。

第 17 章 配置分层动态检索模块

- 『分层动态检索模块配置属性』

本章描述 IBM WebSphere Business Integration mySAP.com 适配器（SAP R/3 V3.x）的分层动态检索模块的配置。在执行本章中描述的配置任务之前，应安装 SAP 连接器。有关安装该连接器的更多信息，请参阅第 37 页的第 4 章，『运行连接器』。

分层动态检索模块配置属性

必须设置标准的和特定于连接器的配置属性，然后才能运行分层动态检索模块。至少，必须将 BAPI 模块的名称添加至 `Modules` 属性。BAPI 模块的名称为 `Bapi`。

分层动态检索模块仅执行服务调用请求，因此，通过正在发送的业务对象中的元数据来调用业务对象处理程序。但是，通过将模块的值设置为 `Bapi` 来建立连接器线程，从而允许初始化和终止连接器。因此，如果在分层动态检索模块处理期间产生了任何问题，通过对正在运行的连接器线程调用 `terminate()` 方法就可以很容易地关闭连接器。

有关配置连接器配置属性的更多信息，请参阅第 21 页的第 3 章，『配置连接器』和第 261 页的附录 D，『连接器的标准配置属性』。

第 18 章 为分层动态检索模块开发业务对象

本章描述分层动态检索模块如何处理业务对象并描述连接器在检索数据时所作出的假设。您可以使用此信息作为修改现有业务对象的指南或作为实现新业务对象的建议。

有关分层动态检索模块的描述，请参阅第 163 页的第 16 章，『分层动态检索模块概述』。

业务对象结构

连接器假定每个个体业务对象由一个或多个数据库表来表示，并假定该业务对象内的每个**简单属性**（即，表示单值的属性，如 String、Integer 或 Date）由其中一个表中的列表示。以下情况是有效的:

- 数据库表具有的列数可能多于相应个体业务对象具有的简单属性数（即，未在业务对象中表示数据库中的某些列）。您的设计中仅包括业务对象处理所需要的那些列。
- 个体业务对象具有的简单属性数可能多于相应数据库表具有的列数（即，数据库中未表示业务对象中的某些属性）。在数据库中没有表示的属性不具有特定于应用程序的信息。
- 由于 SAP API 中的限制，由单个业务对象表示的每个表中所有期望列的字符总数不能超过 512。有关更多信息，请参阅第 170 页的『处理长数据行』。

SAP 的 WebSphere 业务对象可以是平面的或分层的。**平面**业务对象的所有属性都是简单的并表示一个单值。

分层业务对象具有表示单个子业务对象、一组子业务对象或两者组合的属性。同样，每个子业务对象都可以包含单个子业务对象或一组业务对象等等。

业务对象关系

表示一个子代或一组子代的属性的 Cardinality 特性确定父代和子代之间的关系类型:

- 当父业务对象中的属性用基数 1 表示子业务对象时，则发生**单基数关系**。
- 当父业务对象中的属性用基数 n 表示一组子业务对象时，则发生**多基数关系**。

连接器不会以与多基数关系不同的方式来处理单基数关系。但是，当数据库表具有单基数或多基数关系时，外键关系中存在结构差别:

- 在单基数关系中，外键由子代中的主键确定，该子代引用父代中的**非键**属性作为其外键。每个子代都必须至少具有一个引用其父代中的一个非主键属性作为外键的简单属性。图 60 提供了一个示例。
- 在多基数关系中，外键由子代中的主键确定，该子代引用父代中的**主键**属性。每个子代都至少具有一个包含父代的主键作为外键的简单属性。子代具有的外键属性数与父代具有的主键属性数相同。图 62 提供了一个示例。

在任一情况下，父代和子代业务对象之间的外键关系都由子业务对象键属性的特定于应用程序的信息指定。有关更多信息，请参阅第 172 页的『业务对象属性特性』和第 174 页的『简单属性的特定于应用程序的信息』。

下面各节描述业务对象之间的以下关系:

- 『单基数关系示例』
- 第 169 页的『多基数关系示例』

单基数关系示例

图 60 提供了一个为处理 SAP 中的客户对象而开发的简单 WebSphere 业务对象的示例。此 SAP_Customer 示例与它包含的地址对象示例具有单基数关系（addr_data[1] 属性具有基数 1）。子业务对象中的主键属性（address_id）引用父业务对象中的非主键（address_id）。

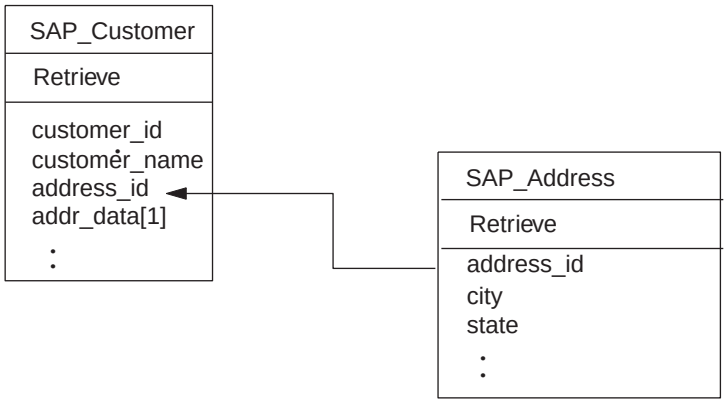


图 60. 客户和地址关系示例

以下 SELECT 语句及其输出举例说明了从以上业务对象表示的表中检索数据的过程:

```
SELECT * FROM KNA1

KUNNR          NAME1          ADRNR
-----          -------------  ----------
10254          JOE'S PIZZA          2208
10255          LARRY'S HARDWARE     2209

SELECT * FROM ADRC

ADDRNUMBER          CITY1          REGION
----------          ----           -----
2208                BURLINGAME     CA
2209                SAN FRANCISCO  CA
```

在以上示例中，每个客户（Joe's Pizza 和 Larry's Hardware）都具有单个地址。如果将 KUNNR 和 ADDRNUMBER 列分别定义为它们的表的主键约束，则以上结构确保每个客户都只能具有一个相关地址。

注: 为了简单起见，本文档中的举例说明不显示由连接器用来确定 SAP 应用程序数据库中表和字段的特定于应用程序的信息。

多基数关系示例

图 61 举例说明了多基数关系。在该示例中，ID=ABC 是具有父代的主键的简单属性，child[n] 是表示一组子业务对象的属性。

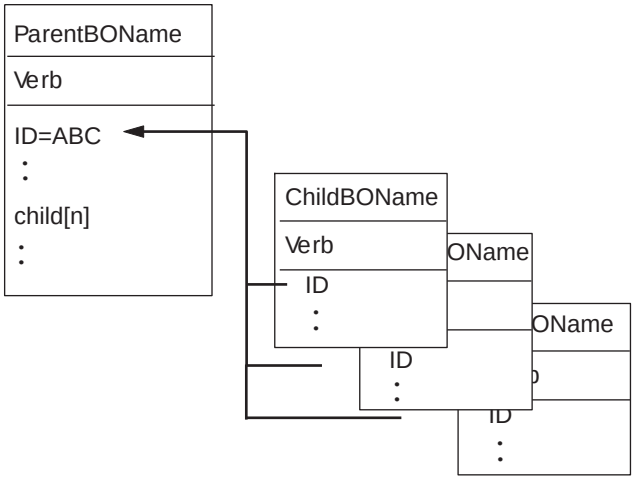


图 61. 多基数业务对象关系

图 62 提供了一个为处理 SAP 中的客户对象而开发的不同 WebSphere 业务对象的示例。此 SAP_Customer 示例与它包含的销售视图对象示例具有多基数关系（sales_view_data[n] 属性具有基数 n）。子业务对象中的主键属性（customer_id）引用父业务对象中的主键（customer_id）。

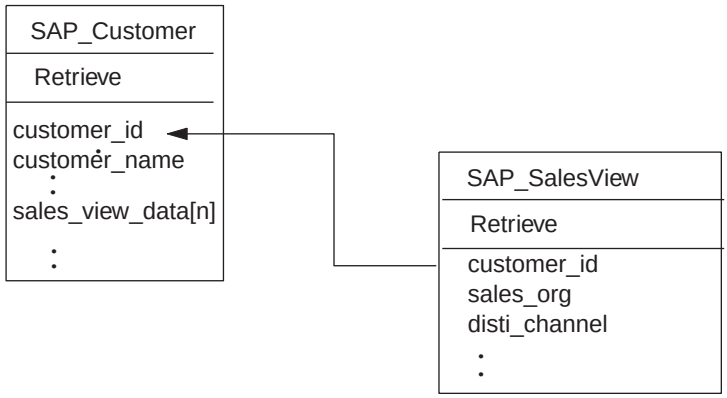


图 62. 客户和销售视图关系示例

以下 SELECT 语句及其输出举例说明了从其中每个表中检索数据的过程:

```
SELECT * FROM KNA1

KUNNR          NAME1
-----          -------------
10254          JOE'S PIZZA
10255          LARRY'S HARDWARE

SELECT * FROM KNVV

KUNNR          VKORG          VTWEG          SPART
----------     -----          -----          -----
10254          EURP          01             12
```

```
10255          EURP          01            09
10255          USA           01            13
10255          USA           01            14
```

在此示例中，Joe's Pizza 具有一条相关的销售视图记录，而 Larrys Hardware 具有三条相关的销售视图记录。以上结构允许每个客户具有 0 条或多条相关的销售视图记录。

处理长数据行

SAP 的 RFC_READ_TABLE 功能将数据检索限制为每行数据 512 个字节。许多 SAP 表的每行数据超过 512 个字节。但是，大多数业务对象都表示所有数据库字段的小子集。因此，一个业务对象中所有属性的总长度很少超过最大值 512 个字节。

在要求连接器从单个数据库表检索超过 512 个字节数据的那些情况下，必须在不同的单基数子业务对象中表示额外的字段。例如，如果业务对象必须表示单个表中 1500 个字节的数据，则顶级业务对象至少包含两个单基数子业务对象。父代或子代都没有其总长度（即它们的最大长度之和）超过 512 个字节的属性。

注: 如果一个业务对象表示多个数据库表，则表示每个表的属性中值的总长度不能超过 512 个字节。但是，此限制不适合于所有属性的值的总长度。例如，如果业务对象表示来自表（这些表存储关于客户和客户伙伴的信息）的数据，则用于表示客户的那些属性值不能超过 512 个字节，且用于表示客户伙伴的那些属性值也不能超过 512 个字节，但这些属性的组合值可以超过 512 个字节。

业务对象查询描述处理

本节概述连接器处理具有“检索”查询描述的业务对象请求所执行的步骤。连接器递归地处理分层业务对象；即，它对每个子业务对象执行相同的步骤，直到它已处理所有个体业务对象。

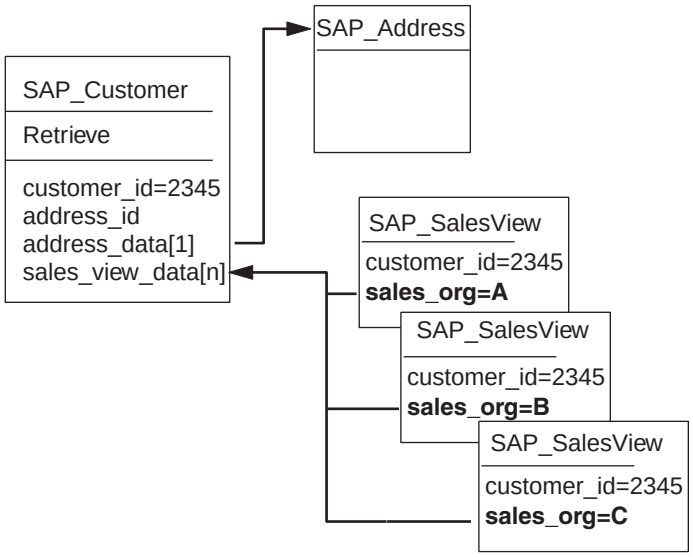
业务对象比较

当处理来自集成代理程序的检索请求时，连接器尝试返回与业务对象的当前数据库表示匹配的业务对象。换句话说:

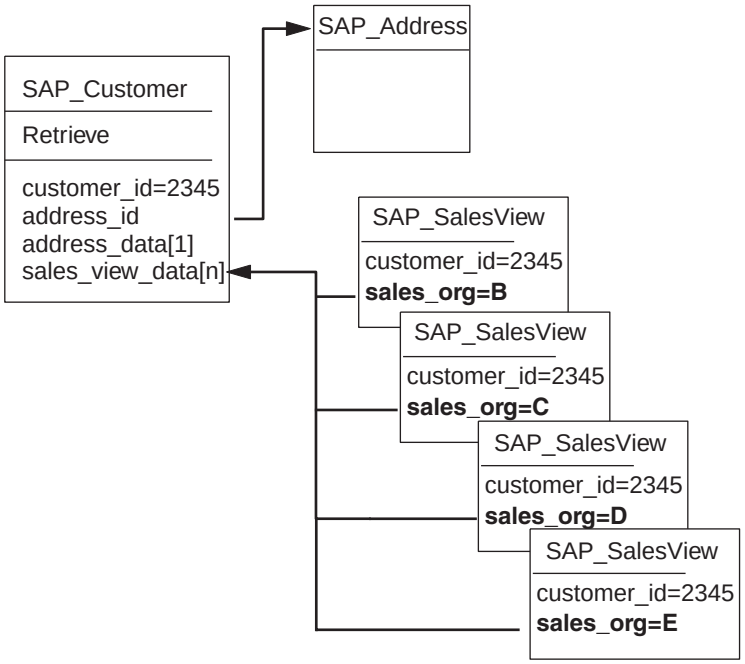
- 返回到集成代理程序的所有个体业务对象中每个简单属性的值都与数据库中其相应字段的值匹配。
- 返回的每组业务对象中个体业务对象的数目都与数据库中的相应子代数匹配。

因此，当分层动态检索模块接收到具有“检索”查询描述的业务对象请求时，它通过在应用程序中递归地向下查找整个对象并检索当前数据库表示来创建一个响应业务对象。为了执行该检索，连接器使用在顶级请求业务对象中指定的键值。因此，该响应业务对象（它包含该顶级父代的所有子代）可能具有与请求业务对象不同的简单属性值和不同的子业务对象。

例如，假设集成代理程序将以下 SAP_Customer 业务对象传递至分层动态检索模块:



在当前数据库表示中，如果 SAP_Customer 2345 包含的一组 SAP_SalesView 子业务对象不包括 sales_org A，则连接器的响应业务对象不包含该子代。但是，如果 SAP_Customer 2345 的当前数据库表示包括 sales_org D 和 sales_org E，则连接器在响应业务对象中包括那些子代。SAP 分层动态检索模块在结束检索时返回到集成代理程序的业务对象如下:



注: 如果连接器在创建特定响应业务对象时从多个表读取，则该业务对象不会与单个数据库对象匹配。而是，它与从指定的表中选择的字段匹配。

检索操作

当检索业务对象时，如果操作成功，则连接器返回 VALCHANGE 状态（而不管该操作是否导致了对业务对象的更改），或者，如果操作失败则返回 FAIL 状态。

连接器在检索分层业务对象时执行以下步骤:

1. 从它从集成代理程序接收到的顶级业务对象中除去所有子业务对象。
2. 调用 RFC_READ_TABLE 功能来从数据库检索**顶级业务对象**。

   连接器使用请求业务对象中的键值来构建 SELECT 语句的 WHERE 子句。检索结果导致以下其中一项操作:

   - 如果 SELECT 语句返回一条记录，则连接器继续处理子代并返回 VALCHANGE（而不管是否有任何属性更改了值）。
   - 如果 SELECT 语句未返回任何记录，则这表示该顶级业务对象在数据库中不存在，连接器返回 BO_DOES_NOT_EXIST。
   - 如果 SELECT 语句返回多条记录，则连接器继续处理子代并返回 VALCHANGE。
3. 递归地检索所有**子业务对象**（单基数和多基数）。

   连接器调用 RFC_READ_TABLE 功能，该功能使用适当的外键值来构建 SELECT 语句的 WHERE 子句。连接器以下列方式来处理标记为必需的属性:

   - 如果业务对象的定义指定需要子代，则检索必须返回记录。如果未返回记录，则连接器返回 FAIL。
   - 如果不需要子代且检索未返回记录，则这表示子代在应用程序中不存在，连接器使父代的属性保持为空。

   对于返回的每条记录，连接器执行以下操作:

   a. 创建正确类型的新的个体业务对象。
   b. 根据返回的行中的值设置当前业务对象的所有属性。
   c. 递归地检索当前业务对象的所有子代。

      警告: 如果单基数子代的检索返回多条记录，则连接器仅返回第一条记录。
   d. 将当前业务对象及其所有子代插入父代的适当单基数属性或数组属性。

注: 业务对象可以具有不对应于任何数据库列的属性，如占位符属性。在检索期间，连接器不会在顶级业务对象中更改这种属性；它们仍保持设置为从集成代理程序接收的值。这些属性的特定于应用程序的信息必须为空白。

业务对象属性特性

业务对象体系结构定义应用于属性的各种特性。本节描述连接器如何解释这些属性并描述当修改业务对象时如何设置它们。

Name 属性

每个业务对象属性必须具有一个唯一名称。

Type 属性

每个业务对象属性都必须具有类型 String 或一个子业务对象或一组子业务对象的类型。

Cardinality 属性

每个业务对象属性在此特性中都具有值 1 或 n。表示一个子业务对象或一组子业务对象的所有属性还具有 ContainedObjectVersion 属性（它指定子代的版本号）和 Relationship 属性（它指定值 Containment）。

Max length 属性

此连接器不使用此属性。尽管高级出站向导在生成业务对象时填充此属性，但它这样做的目的仅仅是为了提供信息。

Key 属性

每个业务对象中必须至少指定一个简单属性作为键。要将属性定义为键，将此特性设置为 true。

重要提示: 连接器不支持指定表示一个子业务对象或一组子业务对象的属性作为键属性。

如果对简单属性将 key 特性设置为 true，则连接器将该属性添加至它在处理业务对象时生成的 SELECT SQL 语句的 WHERE 子句。

为了使性能达到最高，建议您为尽可能多的键字段提供数据。

要从一组业务对象检索一个或多个子业务对象，连接器在 SELECT 语句的 WHERE 子句中使用外键。它不使用子业务对象中属性的 Key 属性。有关如何将子业务对象中的属性指定为外键的信息，请参阅第 174 页的『简单属性的特定于应用程序的信息』。

Foreign key 属性

此连接器不使用此属性。连接器从特定于应用程序的信息获取外键信息。有关更多信息，请参阅第 174 页的『简单属性的特定于应用程序的信息』。

Required 属性

Required 特性指定属性是否必须包含值。

- 如果用于表示一个子业务对象或一组子业务对象的属性标记为必需的，且连接器未能从应用程序检索到任何子代，则检索操作失败。
- 如果一个简单属性标记为必需的且连接器未能从数据库检索到相应行，则检索操作失败。例如，如果连接器从业务对象的多个表读取，而它未能检索到表示其中一个表中值的必需简单属性的一行，则整个检索失败。

AppSpecificInfo

有关此属性的信息，请参阅第 174 页的『简单属性的特定于应用程序的信息』。

Default value 属性

此属性指定连接器在生成 SELECT 语句的 WHERE 子句时使用的缺省值。此属性仅与已指定为键的简单属性相关。例如，要使连接器使用为 Language 属性指定的缺省值，您必须将 Language 属性指定为键。

简单属性的特殊值

业务对象中的简单属性可以具有特殊值 CxIgnore。当从集成代理程序接收到业务对象时，连接器忽略具有值 CxIgnore 的所有属性。好像那些属性对于连接器是不可见的一样。

当连接器从数据库检索数据且 SELECT 语句对某个属性返回空值时，缺省情况下，连接器将该属性的值设置为 CxBlank。

因为连接器要求每个业务对象至少具有一个键属性，所以应确保传递至连接器的业务对象至少具有一个未设置为 CxIgnore 的主键或外键。

特定于应用程序的业务对象信息

业务对象定义中特定于应用程序的信息对连接器提供关于如何处理业务对象的应用程序相关指示信息。此信息包括:

- vDynRetBOH 业务对象处理程序的类，在特定于应用程序的顶级业务对象查询描述信息中提供该类。此值对于此模块处理的所有业务对象都是相同的。
- 数据库和查询信息，在简单属性的特定于应用程序的信息中提供该信息。连接器对此信息进行语法分析来生成 SELECT 查询。

如果您扩展或修改特定于应用程序的业务对象，则确保业务对象定义中特定于应用程序的信息与连接器期望的语法匹配。

以下各节更详细地讨论了此功能。

特定于应用程序的顶级业务对象查询描述信息

顶级业务对象的查询描述指定 vDynRetBOH 业务对象处理程序的类。此特定于应用程序的信息应始终是以下内容:

```
sap.bapi.vDynRetBOH
```

简单属性的特定于应用程序的信息

属性的特定于应用程序的信息指定以下信息:

- 相应的数据库表的名称
- 相应的数据库列的名称
- 当前业务对象中的属性与父代或子代业务对象之间的外键关系
- 操作数

特定于应用程序的信息格式由 4 个名称 – 值参数组成，每个参数包括参数名及其值。将用冒号（:）定界设置的每个参数。

特定于应用程序的属性信息的格式如下所示。方括号（[ ]）括起可选参数。垂直条（|）分隔一组选项的成员。保留冒号作为定界符。

```
TN=TableName:CN=ColumnName:[FK=[..]fk_attributeName]:[OP=GT|GE|EQ|NE|LE|LT|LIKE]
```

图 60 描述每个名称 – 值参数。

表 36. 特定于应用程序的属性信息中的名称 - 值参数

参数	描述
TN=TableName	数据库表的名称
CN=ColumnName	数据库表列（字段）的名称
FK=[..]fk_attribute Name	此属性的值取决于外键关系是存储在父业务对象中还是存储在当前业务对象中: • *attributeName* - 指定当前业务对象中的属性；有关更多信息，请参阅第 175 页的『示例: 当前业务对象存储外键』 • ..attributeName - 指定父业务对象中的属性 如果属性不是外键，则不要在特定于应用程序的信息中包括此参数。
OP=GT|GE|EQ|NE|LE|LT| LIKE	操作数选项为: • GT - 大于 • GE - 大于或等于 • EQ - 等于（缺省选项） • NE - 不等于 • LE - 小于或等于 • LT - 小于 • LIKE - 类似 建议您指定 EQ 以使性能达到最高。如果未指定操作数，则连接器使用 EQ。

每个简单属性的必需参数是表名和列名。操作数缺省为 EQ（等于）。以下示例举例说明了基本格式:

```
TN=KNA1:CN=KUNNR
```

重要提示: 当为这些参数指定值时，大小写是有意义的。

允许业务对象内的简单属性不具有对特定于应用程序的信息字段指定的值（即 0 长度）。连接器忽略这样属性。这是最方便的方法，确保连接器不处理用来分隔相邻组的子业务对象的占位符属性。

如果业务对象的任何属性中没有任何特定于应用程序的信息来为连接器构建或执行查询提供足够的信息，则连接器返回故障。

示例: 当前业务对象存储外键

图 63 提供了一个具有两个外键的 WebSphere 业务对象示例，这两个外键引用业务对象内的属性。在这种情况下，业务对象在两个表中表示数据，一个表包含地址数据，另一个表包含州 / 省和国家或地区缩写的查找数据。为了处理此数据，连接器执行两次读表操作。

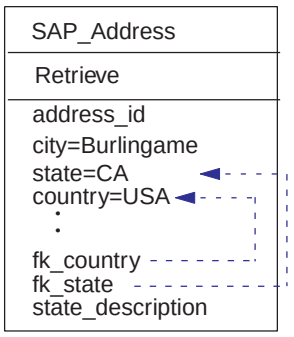


图 63. 示例: 当前业务对象存储外键

属性信息: 表 37 在示例 SAP_Address 中描述了每个属性的表名、列名、键和外键记录:

表 37. 示例业务对象属性的描述

属性	表名	列名	键	外键	缺省值
address_id	ADRC	ADDRNUMBER	true		
city	ADRC	CITY1	false		
state	ADRC	REGION	false		
country	ADRC	LAND1	false		
language	T005U	SPRAS	true		E
fk_country	T005U	LAND1	false	FK=country	
fk_state	T005U	BLAND	false	FK=state	
state_description	T005U	BEZEI	false		

特定于应用程序的属性信息: 如果给出表 37 中的信息，则 fk_state 属性的特定于应用程序的信息是:

```
TN=T005U:CN=BLAND:FK=state
```

fk_country 属性的特定于应用程序的信息是:

```
TN=T005U:CN=LAND1:FK=country
```

SQL 查询: 以下 SELECT 语句举例说明了 WHERE 子句，连接器构建该子句以从由 SAP_Address 表示的表中检索数据:

```
SELECT * FROM ADRC WHERE ADDRNUMBER = address_idValue
SELECT * FROM T005U WHERE SPRAS = 'E' AND LAND1 = countryValue
AND BLAND = stateValue
```

第 6 部分 ABAP 扩展模块

第 19 章 ABAP 扩展模块概述

本章描述 IBM WebSphere Business Integration mySAP.com 适配器的 ABAP 扩展模块。ABAP 扩展模块使集成代理程序能够将业务对象发送至 SAP 应用程序，也能够从这些 SAP 应用程序中接收事件。

ABAP 扩展模块组件

ABAP 扩展模块由 Java 和 ABAP 语言编写的组件组成。Java 组件由连接器模块和 SAP RFC 库组成。SAP 以 Java 和 C 语言交付它们的 RFC 库。ABAP 组件由多个 SAP 应用程序功能模块、数据库表和程序组成。其中某些 ABAP 组件是作为适配器的一部分开发和交付的，而某些 ABAP 组件则是每个 SAP 安装的本机组件。

图 64 举例说明了 ABAP 扩展模块的完整体系结构。

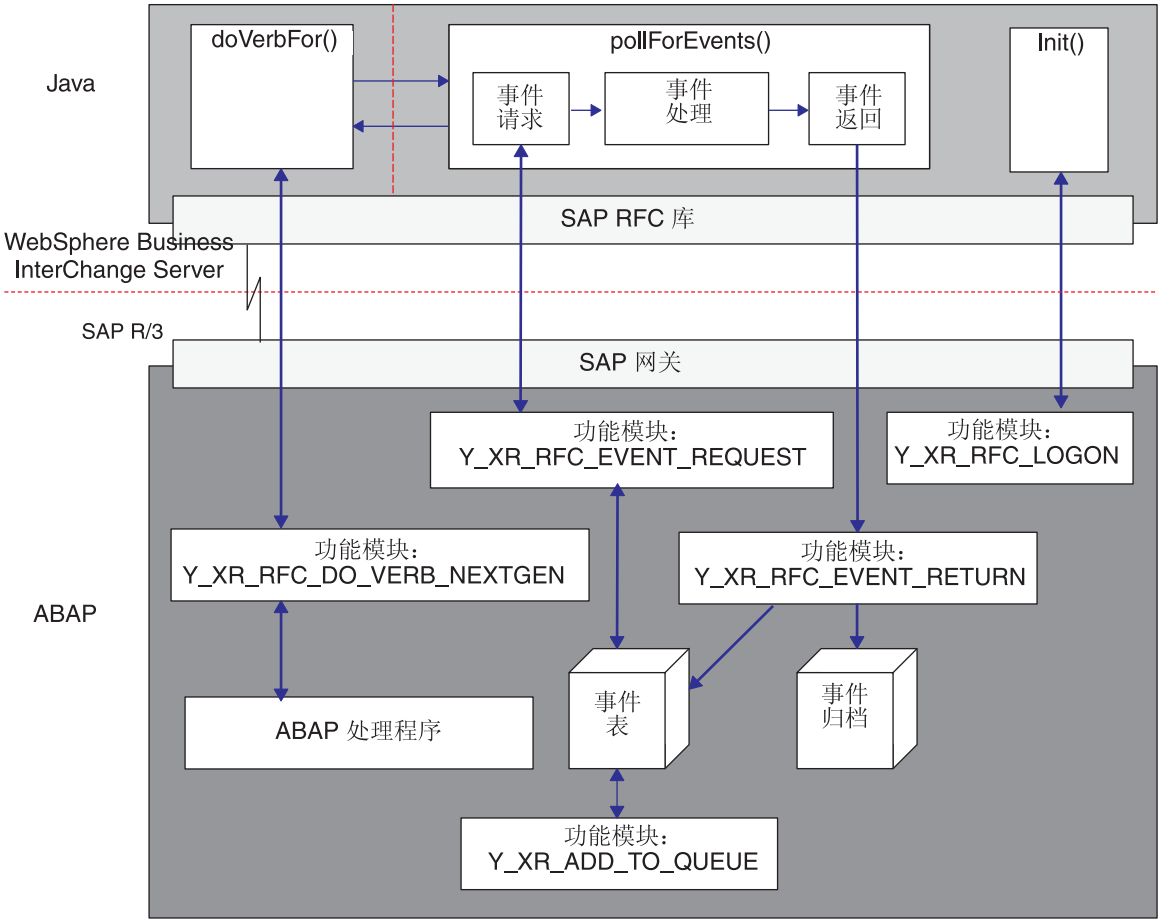


图 64. ABAP 扩展模块体系结构

Java 组件

将作为 Java 归档（JAR）文件交付和运行连接器。它处理事件传递和事件业务对象请求进程。同时 SAP RFC 库将作为 JAR 文件交付和运行。该库使外部程序能够执行 SAP 应用程序内的 ABAP 功能模块。

Java 组件:

- 使用 SAP RFC 库和 SAP 网关打开与 SAP 应用程序的 RFC 连接。
- 处理来自集成代理程序的请求并将这些请求传递至连接器的 ABAP 组件。
- 轮询 SAP 应用程序以获取事件。

ABAP 组件

连接器的 ABAP 组件有功能模块、程序和数据库表。这些元素处理由 Java 组件启动的事件传递和业务对象请求进程。将在要装入 SAP 应用程序中的连接器传送文件中交付 ABAP 组件；一旦装入这些组件，它们将作为 ABAP 资源库对象运行。

ABAP 组件:

- 通过调用旨在处理特定业务对象类型和查询描述的适当功能模块来处理来自 Java 组件的业务对象请求。
- 检测、触发和存储事件表中的事件。
- 处理来自 Java 组件的事件请求及其后续返回（事件状态更新）。

ABAP 扩展模块的工作方式

ABAP 扩展模块提供的大多数功能都在 SAP 应用程序内部执行。对于每个连接器必须实现的大多数虚拟功能，SAP 应用程序中存在一个对应的 ABAP 功能模块。但是，由于 SAP 不提供支持 `init()`、`doVerbFor()` 和 `pollForEvents()` 方法的特定需求的 ABAP 功能模块，因此这些功能模块是作为连接器模块的一部分开发和交付的。虽然 Java 组件提供了某些功能，但处理这些方法的主要工作由 SAP 应用程序中的 ABAP 组件来完成。

表 38 显示连接器模块实现的虚拟 Java 方法及其相应的 ABAP 组件。记住，此表不提供由连接器使用的 ABAP 组件的完整列表。

表 38. Java 组件及其相应的 ABAP 组件

Java 组件	ABAP 组件
`doVerbFor()`	`Y_XR_RFC_DO_VERB_NEXTGEN`
`getVersion()`	不需要实现
`getBOHandlerForBO`	不需要实现
`init()`	`Y_XR_RFC_LOGON`
`pollForEvents()`	`Y_XR_RFC_EVENT_REQUEST` `Y_XR_RFC_EVENT_RETURN`
`terminate()`	不需要实现

这些 ABAP 功能模块合起来就是 ABAP 扩展模块的核心。以下各节描述连接器初始化、业务对象处理和连接器如何处理事件通知。

本章的其余部分中讨论已实现的功能。

初始化

init() 方法调用 ABAP 功能模块 Y_XR_RFC_LOGON 来验证目标 SAP 应用程序是否正在运行以及 RFC 库是否可用于执行 ABAP 功能模块。如果该功能模块未成功执行，则连接器终止。

业务对象处理

SAP 的所有服务调用请求都由连接器模块的 Java 组件中的 doVerbFor() 方法启动。连接器的 ABAP 功能模块 Y_XR_RFC_DO_VERB_NEXTGEN 和连接器模块的 ABAP 组件中 ABAP 处理程序处理这些请求。

图 65 举例说明了业务对象处理。

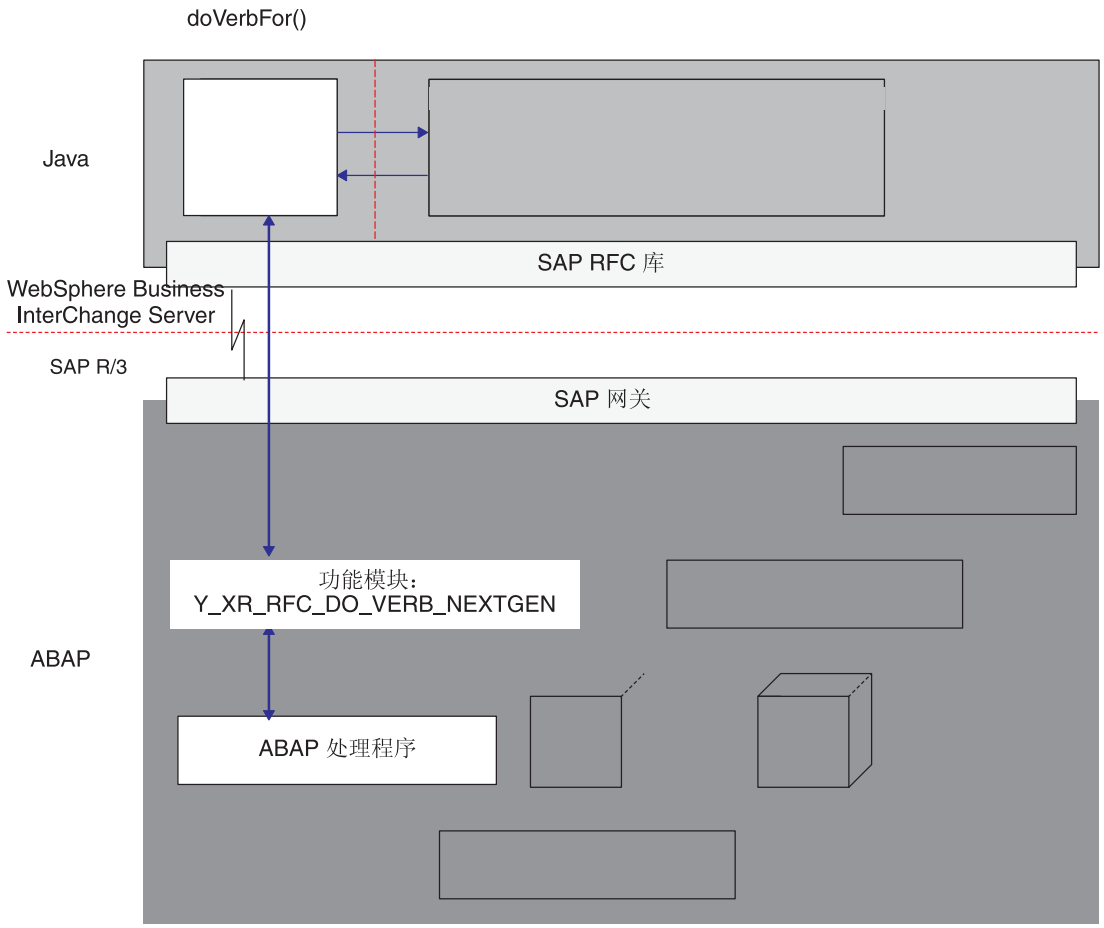


图 65. doVerbFor() 的业务对象处理

doVerbFor()

在连接器模块的 Java 组件中，单个业务对象处理程序实现的 doVerbFor() 方法处理来自集成代理程序的所有业务对象请求以及来自 pollForEvents() 方法的所有业务对象事件。在任一情况下，doVerbFor() 以下列方式执行:

1. 将 SAP 的 WebSphere 业务对象实例转换为包含业务对象数据的单个预定义平面结构。

2. 调用 ABAP 功能模块 Y_XR_RFC_DO_VERB_NEXTGEN，将业务对象数据传递至该模块，然后等待业务对象数据返回。
3. 将返回的业务对象数据重新转换为 WebSphere 业务对象。

doVerbFor() 方法将业务对象数据传递至功能模块 Y_XR_RFC_DO_VERB_NEXTGEN，然后根据返回的业务对象数据创建一种全新的业务对象结构。

Y_XR_RFC_DO_VERB_NEXTGEN

在连接器模块的 ABAP 组件中，连接器的 ABAP 功能模块 Y_XR_RFC_DO_VERB_NEXTGEN 负责SAP 应用程序中的所有 WebSphere 业务对象处理。特别是，它将业务对象数据传递至适当的 ABAP 处理程序。从这种意义上来说，可以认为功能模块 Y_XR_RFC_DO_VERB_NEXTGEN 是一个业务对象路由器。它始终以下列方式执行:

1. 接收业务对象。
2. 动态地调用 ABAP 处理程序来处理业务对象数据并以参数形式传递业务对象数据。
3. 从 ABAP 处理程序接收业务对象数据并将它返回到发出请求的调用。

Y_XR_RFC_DO_VERB_NEXTGEN 使用 ABAP 处理程序来完成每种对象类型和特定于查询描述的请求。Y_XR_RFC_DO_VERB_NEXTGEN 使用特定于应用程序的业务对象查询描述信息中值来确定要调用哪个 ABAP 处理程序。可以认为 Y_XR_RFC_DO_VERB_NEXTGEN 是从 doVerbFor() 方法至 ABAP 处理程序的路由器。

ABAP 处理程序

ABAP 处理程序是连接器模块所独有的，这些处理程序从连接器模块的 Java 组件扩展业务对象处理程序功能。ABAP 处理程序以 ABAP 功能模块的形式驻留在 SAP 应用程序中并直接与 Y_XR_RFC_DO_VERB_NEXTGEN 通信。将业务对象数据输入或输出 SAP 应用程序数据库需要 ABAP 处理程序。

图 66 举例说明了 ABAP 扩展模块的业务对象处理组件及这些组件的相互关系。注意，对于单个业务对象处理程序（doVerbFor()）和业务对象路由器（Y_XR_DO_VERB_NEXTGEN），存在多个 ABAP 处理程序。

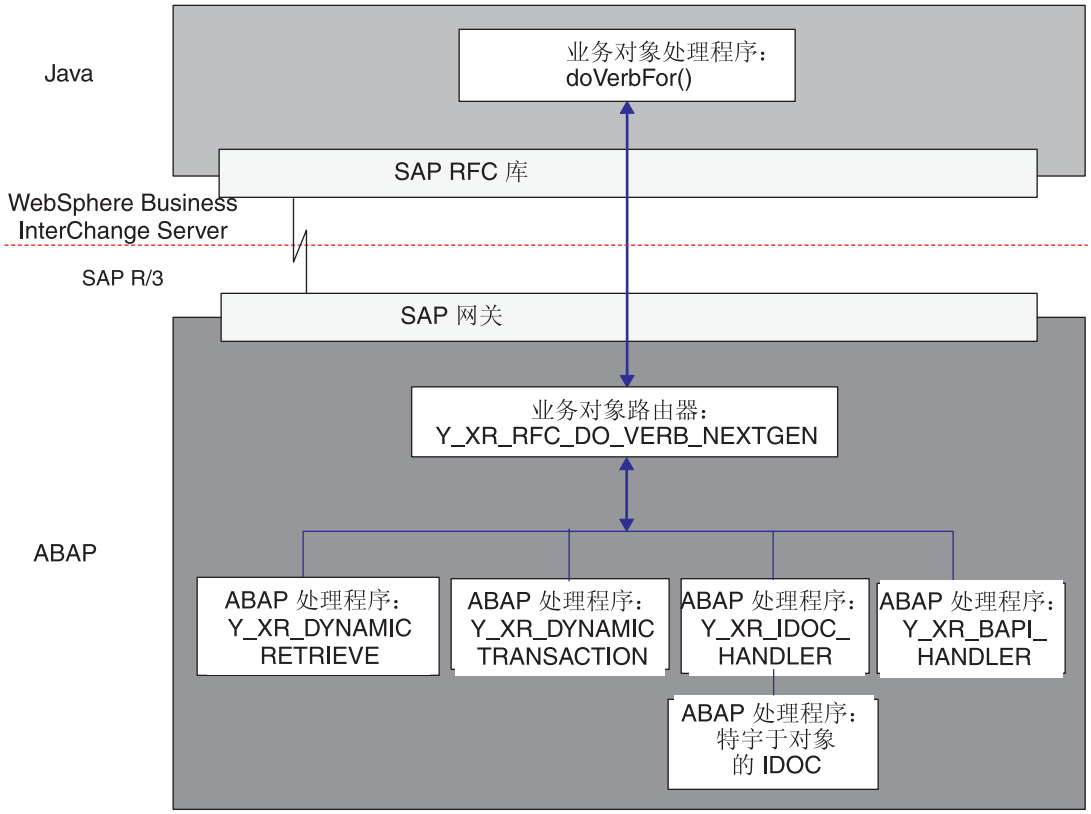


图 66. 适配器提供的业务对象处理组件

ABAP 处理程序负责将业务对象数据添加至 SAP 应用程序数据库（创建、更新或删除），或负责使用业务对象数据作为键来从 SAP 应用程序数据库检索数据（检索）。

适配器提供通用 ABAP 处理程序。例如，功能模块 `Y_XR_RFC_DYNAMIC_TRANSACTION` 支持对平面业务对象执行创建、更新、删除和检索操作。

WebSphere Business Integration 系统提供一个元数据资源库和一个通用 ABAP 处理程序来支持平面业务对象。适配器也提供一个 ABAP 处理程序（`Y_XR_IDOC_HANDLER`）来支持分层业务对象；但是，您必须为您需要支持的每个分层业务对象开发额外的特定于业务对象的 ABAP 处理程序。

WebSphere Business Integration 系统提供一些有助于开发过程的工具。有关开发业务对象和 ABAP 处理程序的更多信息，请参阅第 205 页的第 22 章，『为 ABAP 扩展模块开发业务对象』和第 41 页的第 5 章，『使用 SAPODA 生成业务对象定义』。

事件通知

事件通知指一些进程的集合，这些进程将 SAP 应用程序对象事件通知连接器。通知包括（但不限于）事件（对象和查询描述）的类型和外部系统检索相关数据所需要的数据键。

图 67 举例说明了使用 `pollForEvents()` 方法的事件通知进程。

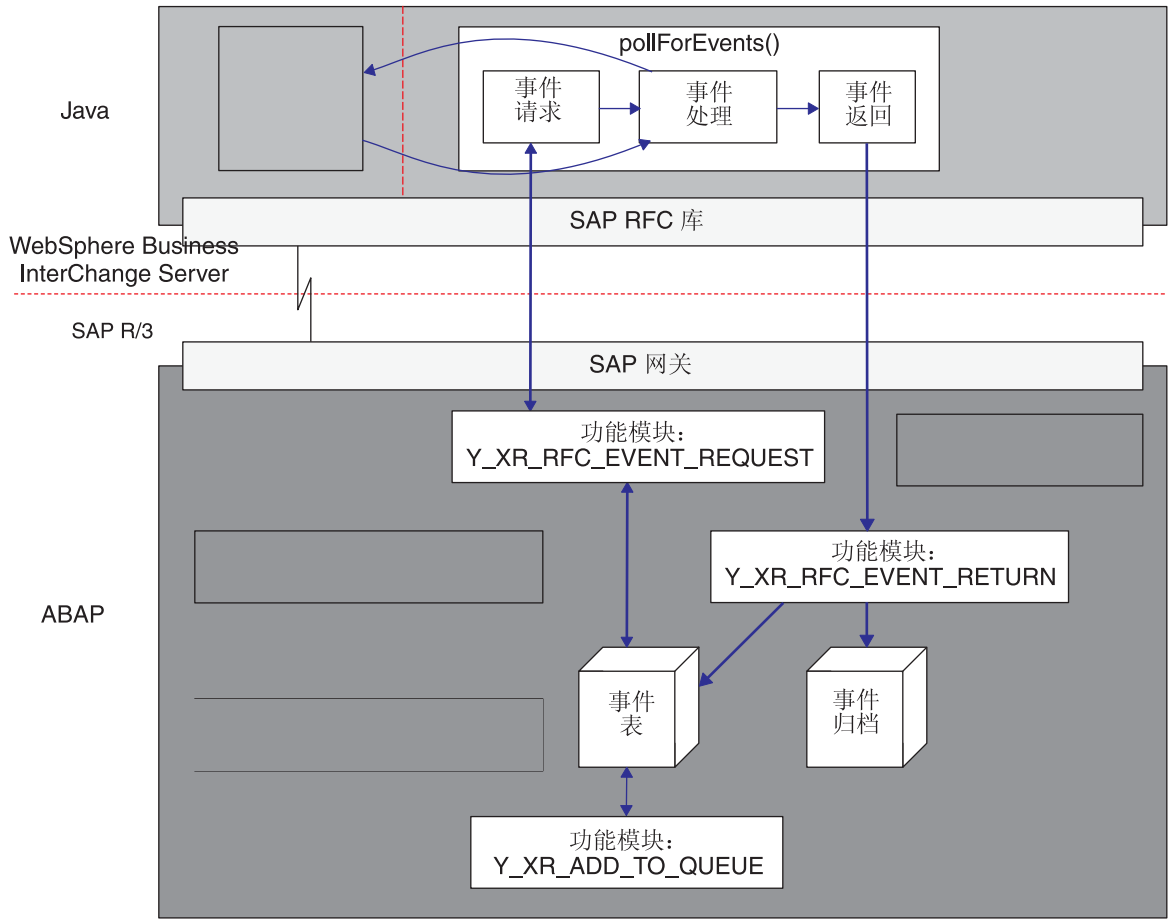


图 67. 事件通知进程

连接器的事件通知由两个功能组成:

- 『事件轮询』
- 第 186 页的『事件触发』

事件轮询

事件轮询由 `pollForEvents()` 方法执行的三个功能组成:

- 『事件请求』
- 第 186 页的『事件处理』
- 第 186 页的『事件返回』

注: 这些功能的角色分布在 Java 和 ABAP 组件中。但是，始终是 Java 组件启动事件轮询。

事件请求: 事件请求是在 SAP 应用程序中从事件表轮询和检索事件的过程。Java 组件的事件请求机制在 SAP 应用程序中具有一个对应的功能模块 `Y_XR_RFC_EVENT_REQUEST`。此功能从连接器的 ABAP 事件表 `YXR_EVENTS` 检索事件。

每个触发的事件进入事件表时，初始状态为预排队（在事件表中标记为 P 的状态），缺省事件优先级为 0。必须将事件的状态更改为已排队（事件表中的 Q），然后才能处理

事件。在连接器检索事件所表示的完整对象之前，事件的优先级必须为 0。有关事件优先级的更多信息，请参阅第 188 页的 『事件优先级』。

如果对创建事件和事件键的用户组合不存在数据库锁定，则该事件的状态从预排队更改为已排队。如果锁定存在，则事件的状态将设置为“已锁定”（事件表中的 L）并且将把该事件重新排队。ABAP 常量 C_MAXIMUM_REQUEUE 定义可以将事件重新排队的次数。如果达到最大次数（缺省为 100），则将把该事件归档至事件归档表。

注: 每次轮询将更新状态为预排队或已锁定的每个事件。当以批处理方式触发事件时，您可能会遇到性能问题。您可以使用 PollFrequency 配置属性来配置轮询频率。有关更多信息，请参阅第 261 页的附录 D, 『连接器的标准配置属性』。

在预处理所有预排队的事件之后，ABAP 功能模块 Y_XR_RFC_EVENT_REQUEST 选择要返回至连接器模块的 Java 组件中事件请求方法的事件（只能选择状态为已排队的事件）。特定于连接器的配置属性 PollQuantity（缺省为 20）确定为单个轮询返回的最大事件数。有关更多信息，请参阅第 37 页的第 4 章, 『运行连接器』。

事件请求机制分两步执行事件选择过程:

1. 选择专用于连接器和集成代理程序的事件。

   这些事件将专用于事件分布表中的特定集成代理程序。在此表中指定的集成代理程序的名称必须与在启动连接器的快捷方式中指定的名称匹配。例如，在 Windows 上运行的 SAP 连接器的标准快捷方式具有以下格式:

   ```
   ...\start_SAP.bat SAPconnectorName integrationBrokerName -cConfigFileName
   ```

   当 WMQI 是集成代理程序时，WebSphere Business Integration 系统通过从连接器的启动命令中获取值来标识在事件分布表中指定的集成代理程序:

   - integrationBrokerName 参数的值将启动命令中的代理程序实例链接至事件分布表中指定的代理程序。

     注: 产品的安装程序使用安装时指定的集成代理程序名称作为启动命令中 integrationBrokerName 参数的值。

   - ConfigFileName 参数的值标识为特定 WMQI 实例配置的队列管理器和队列。

2. 如果选择的事件数小于最大事件数，则从事件中获取未为事件分布配置的其余事件。

   例如，如果特定于连接器的配置属性 PollQuantity 保持在 20，并且有 8 个事件专用于特定连接器和集成代理程序，则该机制选择没有为事件分布配置的其它 12 个事件。

   当 WMQI 是集成代理程序且只配置了一个队列管理器时，队列的名称对于集成代理程序的每个实例必须是唯一的。当 WMQI 是集成代理程序且只配置了一个集群时，队列的名称对于集群内的每个集成代理程序必须是唯一的。

   如果愿意，您可以将代理程序的名称（在启动命令的 integrationBrokerName 参数中指定）或连接器的名称合并为队列的名称。例如，如果两个代理程序命名为 WMQI1 和 WMQI2，则其各自的 ADMINOUTQUEUE 可能分别命名为 ADMINOUTQUEUE_MQI1 和 ADMINOUTQUEUE_MQI2。

重要提示: 如果您设置要轮询多个连接器，则必须将每个事件配置为仅由一个连接器进行处理。否则，连接器可能发送重复的事件，或者可能将事件归档而不是检索它们。

事件处理: 事件请求功能根据 YXR_EVENTS 事件表生成一组要处理的事件。该功能将这些事件传递至事件处理功能，事件处理功能按以下方式每次处理一个事件:

1. 使用 object.verb 值来评估事件是否在连接器预订列表中。
   - 如果事件不在预订列表中，则将事件的状态设置为未预订。
   - 如果事件在预订列表中，则创建 parentObjectOnly.Retrieve 业务对象，并使用事件键值来设置第一个键属性的值。组合键被认为是单一的，且必须由 ABAP 业务对象处理功能模块解释
2. 调用 doVerbFor() 并将业务对象数据传递至它。一旦传递了业务对象，事件处理就等待业务对象数据返回。
3. 根据 doVerbFor() 处理更新事件数组的状态。
4. 如果已成功检索业务对象数据，则将它传递至集成代理程序。

事件返回: 在事件请求处理了每个事件之后，将使用功能模块 Y_XR_RFC_EVENT_RETURN 把每个事件返回至 SAP 应用程序。此功能模块生成已处理事件的副本，并将该副本添加至事件归档表（YXR_ARCHIV ），然后从事件表删除原始条目。

注: 在处理每个事件之后，将更新具有新状态的所有事件。

已归档的事件包括成功处理的事件、处理过但由于错误而终止的事件和未预订的事件。每个事件都具有一个状态，该状态可以指示以下其中一种情况:

- 业务对象已成功发送至集成代理程序。
- 事件从连接器生成了一个未知的 Java 返回码。
- 事件尝试从 SAP 应用程序检索数据时失败。
- 因为业务对象已锁定，所以事件超时。
- 没有任何协作预订事件 – 仅当 InterChange Server（ICS）是集成代理程序时才相关。

在 SAP 应用程序中使用 IBM CROSSWORLDS 连接器工具来管理事件归档表。连接器工具使管理员能够显示和截断归档表并重新提交事件进行处理。有关维护归档表和设置日志截断的更多信息，请参阅第 233 页的第 25 章，『管理 ABAP 扩展模块』。

事件触发

连接器是由事件驱动的。要从 SAP 应用程序获取事件，您需要为每个 IBM WebSphere 支持的业务对象实现事件触发机制。连接器的事件触发由三个功能组成:

- 『事件检测』
- 第 187 页的『事件触发器』
- 第 188 页的『事件持久性』

事件检测: 事件检测是标识在 SAP 应用程序中生成了事件的过程。连接器通常使用数据库触发器来检测事件。但是，因为 SAP 应用程序与 SAP 数据库紧密集成，所以 SAP 只具有非常有限的访问权来直接修改其数据库。因此，将在数据库上的应用程序事务层中实现事件检测机制。

IBM WebSphere Business Integration mySAP.com 适配器通常使用三种机制来在 SAP 应用程序中检测事件:

- 代码增强
- 批处理程序
- 业务工作流

注: 每种事件检测机制都具有优点和缺点，设计和开发业务对象触发器时需要考虑它们。有关实现事件检测机制的更多信息，请参阅第 221 页的第 23 章，『为 ABAP 扩展模块开发事件检测』。

注: 仅有几个事件检测机制示例。检测事件有许多不同的方式。

事件触发器: 无论使用哪种检测机制，都将使用事件触发器 Y_XR_ADD_TO_QUEUE 来触发所有事件。一旦事件由一种检测机制标识，则将把该事件将传递至事件触发器。IBM WebSphere Business Integration mySAP.com 适配器包括一个将事件提交至事件表（YXR_EVENTS）的事件触发器（Y_XR_ADD_TO_QUEUE）。特别是，它为表示事件的对象名、查询描述和键添加一行数据。

所有事件将使用 Y_XR_ADD_TO_QUEUE 添加将至当前事件表。除了将一行数据添加至 YXR_EVENTS 之外，还可以为以下各项设置 Y_XR_ADD_TO_QUEUE:

- 事件过滤
- 事件分布
- 事件优先级

事件过滤、事件分布和事件优先级将作为事件触发器的一部分执行，而不需要通过其它任何程序来执行。它们导致出现了事件的限制（过滤）或修改（事件分布和事件划分优先级）。

事件过滤

事件触发器可以用来过滤掉您不希望添加至事件表的事件。适配器提供了一个 ABAP 包含程序（YXRRESTR），该程序使您能够为此目的限制特定事件。

事件分布

负载均衡可以用来将事件处理分布在多个连接器上，从而允许您同时处理多个事件。事件触发器通过事件分布表（YXR_EVTDIS）提供此能力。您可以指定一些业务对象专供特定连接器检索。同时，事件分布可以选取单个事件并可以对连接器和集成代理程序的每个预订组合复制事件一次或多次。

图 68 举例说明了 SAP 应用程序内的事件触发功能。事件 E1、E2 和 E3 由事件触发器 Y_XR_ADD_TO_QUEUE 接收。E1 表示“客户”事件，E3 表示“订单”事件。事件分布设置为由 SAPconnector1 处理所有“客户”对象，由 SAPconnector2 处理所有“订单”对象。在此环境中，两个连接器都使用同一集成代理程序。因为 E1 是“客户”对象，所以它由 SAPconnector1 轮询; 因为 E3 是“订单”对象，所以它由 SAPconnector2 轮询。E2 是“库存”对象，用于限制库存对象的限制程序 YXRRESTR 中的代码将该对象从工厂 100 中过滤掉。

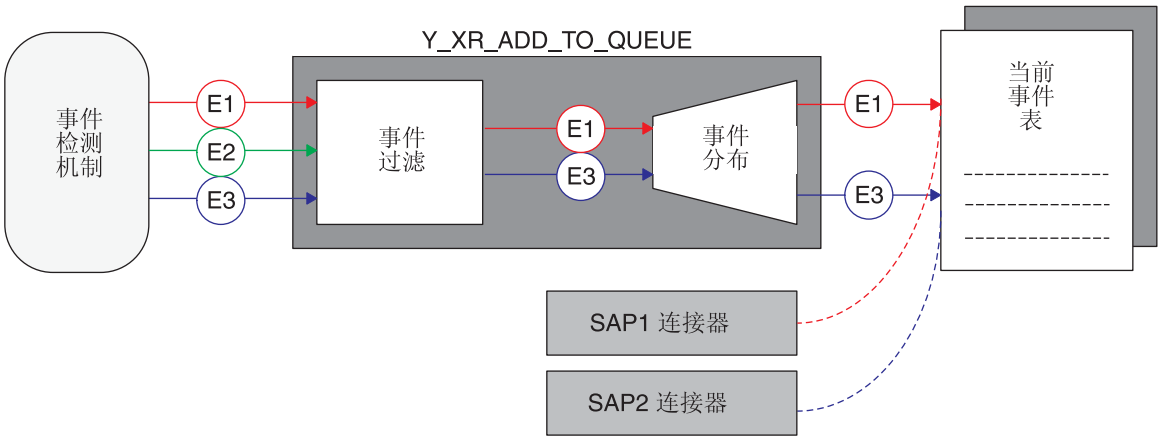


图 68. 使用功能模块 Y_XR_ADD_TO_QUEUE 的事件触发

警告: 如果您使用多个连接器进行轮询，则您必须将每个预订的事件用在特定连接器上。不这样做可能会导致传递重复事件。您必须保证专用于不同连接器的对象之间没有相关性，因为这可能导致不按顺序传递事件。

例如，假定您具有名为 CROSSWORLDS1 的单个集成代理程序，该集成代理程序预订了两个不同的业务对象 BO_A 和 BO_B。BO_A 业务对象较小，因此可以快速检索，而 BO_B 较大，因此需要更长的时间来检索。使用两个连接器轮询 SAPconnector1 和 SAPconnector2，您可以设置事件分布表，以便 SAPconnector1 检索 BO_A，SAPconnector2 检索 BO_B。SAPconnector1 可以连续轮询类型为 A 的小对象，而 SAPconnector2 主要用于类型为 B 的较大对象。

注: 有关 WebSphere Business Integration 系统如何标识 WMQI 集成代理程序的每个唯一实例的信息，请参阅第 184 页的『事件请求』。

重要提示: 如果没有为特定对象配置事件分布表，则为该对象触发的每个事件都可用于连接器和集成代理程序的任何组合。

事件优先级 您可以通过延迟事件检索来为业务对象、连接器和集成代理程序的每个组合设置事件优先级。事件的优先级指示选取事件进行传递之前需要进行的轮询次数。例如，如果您将事件的优先级设置为 10，则连接器在轮询事件表 10 次后才检索事件。每次连接器轮询时，优先级值将减去一，直到达到零为止。

缺省情况下，所有事件都获得优先级零。将在与事件分布相同的 ABAP 表中配置对象的优先级。

事件持久性: 一旦事件触发器将事件插入事件表，就会将事件提交至数据库并设置其事件分布和事件优先级值。此时，只有轮询才能修改该事件。当完成事件轮询进程时（这意味着已从 SAP 应用程序检索该事件并由连接器的 Java 组件处理它），将把已处理的事件的副本添加至事件归档表（YXR_ARCHIV）。然后将从事件表删除原始事件。

注: 您可以从归档表重新提交事件。记住，只会将该事件移至事件表，而不会再次触发它。特别是，它不会重新经历事件过滤、事件分布和事件优先级。

第 20 章 安装和定制 ABAP 扩展模块

本章仅描述 ABAP 扩展模块的安装和定制并假定您已安装和配置 IBM WebSphere Business Integration mySAP.com 适配器。有关安装和配置连接器的更多信息，请参阅第 37 页的第 4 章，『运行连接器』。定制连接器是可选的，但建议进行定制。

对于 Windows，可以在 `\connectors\SAP` 目录中找到连接器的所有组件；对于 UNIX，可以在 `/lib` 目录中找到连接器的所有组件。传送包安装在 SAP R/3 应用程序或数据库服务器上，如第 189 页的『连接器传送文件安装』中所述。

注: 在本文档中，反斜杠（\）用作目录路径的约定。对于 UNIX 安装，用斜杠（/）替代反斜杠。所有文件路径名都是相对于该产品在系统上的安装目录。

连接器传送文件安装

连接器的传送文件包含诸如表结构、功能和数据等各种对象。这些开发对象需要导入 SAP 安装中才能提供 ABAP 扩展模块所需要的特定功能。

传送文件

每个传送文件都包括在 `.zip` 文件中。例如，SAP R/3 V3.x 主传送包的传送文件位于 `Primary.zip` 文件中。可以在 `\connectors\SAP\dependencies\transports_31` 中找到这些文件。

IBM WebSphere Business Integration mySAP.com 适配器（SAP R/3 V3.x）所需要的修改通过一组必需的传送文件（共三个）以及每个业务对象的一个传送文件进行处理。要确保在添加那些表的数据之前已创建所有必需的表，必须以下列顺序导入传送文件:

1. `Primary`
2. `Infrastructure_1`
3. `Infrastructure_2`

一旦成功装入必需的传送文件，就可以任何顺序装入特定于业务对象的传送包。有关传送文件的详细信息，请参阅每个传送包 `.zip` 文件中包括的传送包说明。

Primary 此传送文件包含开发对象，仅应将该文件装入接收系统一次。它包含号码范围对象、YXR1 的开发类和限制包含，它们可以用来对触发逻辑作出特定于客户的更改。当将此传送文件应用于正在运行连接器的系统时一定要小心，因为该传送文件的内容将覆盖现有环境中的所有对象。

Infrastructure_1 此传送文件包含与连接器相关且不是特定于业务对象的程序和数据字典对象。

所有对象都以 YXR 开头，但具有以下例外:

- 在连接器的开发环境中使用的消息类为 YX。
- 有两个设置／获取内存参数 YXD 和 YXV，因为设置／获取参数都限制为三个字符。
- 因为 SAP 在所有区域菜单程序前面自动添加字符串 MENU 作为前缀，所以在两个产品区域菜单后面的程序为 MENUYXR0 和 MENUYXR1。区域菜单本身在我们的名称范围内。

  注意:
  对此传送文件中的对象所作的任何更改都需要在 SAP 外部详细地记录。更改将被下一个 IBM WebSphere 适配器基础结构传送包覆盖，将需要手工重新应用。

Infrastructure_2

此传送包包含连接器日志和用于维护连接器表的事务所必需的 SAP 定制数据。所有键都以 YXR 开头。连接器的表中的定制表条目不以 YXR 开头。

安装连接器传送文件

连接器传送文件通过导入程序和 IBM WebSphere Business Integration mySAP.com 适配器中包括的其它开发对象，对 SAP 作出所有必要的修改。它们不会改变任何 SAP 程序或修改用户出口。

在以下指示信息中，SID 是指 SAP 系统标识，*TransportFileName* 是指传送文件的名称。但是，组成传送文件名的字符在安装目录中出现的顺序不同于以参数形式将该名称传递到各种 tp 命令的方式。在 \usr\sap\trans\cofiles 目录中，传送文件名的格式是 K9*xxxxx*.*SID*，但当以参数形式传递该文件名时，它具有格式 *SID*K9*xxxxx*。例如，如果您的 *SID* 是 D30，则文件名 K912345.D30 将以参数形式传递为 D30K912345。

要安装传送包:

1. 作为 SAP 管理员 SIDadm 登录。
2. 将传送包复制到 SAP 数据库服务器。传送包由两种文件组成，因此应按如下方法复制:
   a. 将名称以 K 开头的文件复制到 \usr\sap\trans\cofiles 目录。
   b. 将其它文件复制到 \usr\sap\trans 数据目录。

      通过运行 tp connect 命令检查与数据库的连接并确定 tpparam 文件的路径:

      ```
      tp connect SID
      ```

      如果此命令失败，则尝试添加 tpparam 文件的路径作为第二个参数:

      ```
      tp connect SID pf = path_of_tpparam
      ```

      例如，如果 SID 为 P11，并且 tpparam 文件的路径为 \usr\sap\trans\bin\tpparam，则该命令将为:

      ```
      tp connect P11 pf = \usr\sap\trans\bin\tpparam
      ```

如果当指定 tpparam 文件的路径时 tp connect 成功，而当未指定该路径时失败，则您应在以下步骤 3 中所描述的命令中指定可选的 tpparam 路径。

3. 可以按以下两种方式的其中一种导入传送包:
   - 在 \usr\sap\trans\bin 中，以指定的顺序对每个传送包执行以下命令:

     tp addtobuffer *TransportFileName SID* pf = *path_of_tpparam*

     tp import *TransportFileName SID* u023689 CLIENT=*CLIENT#* pf = *path_of_tpparam*
   - 在传送管理系统（事务 STMS）中:
     a. 单击“导入概述”图标（F5）。
     b. 双击要更新的适当队列。
     c. 在菜单栏中，单击“额外的”，然后单击“其它请求”，接着单击“添加”。
     d. 填充“传送请求”字段，并单击选取标记以输入它。
     e. 当“添加传送请求”确认窗口出现时，单击“是”以将导入连接至队列。
     f. 将光标放置在刚添加的传送包上。
     g. 在菜单栏中，单击“请求”，然后单击“导入”。
     h. 填充“目标客户机”字段，并单击选取标记以导入它。

     重要提示: 按顺序安装传送包。
4. 当安装传送包时，更改开发类以遵循您的开发类的迁移路径。在 SAP R/3 V3.x 中，开发类是 YXR1。

   使用“IBM CrossWorlds 连接器工具”窗口（事务 YXR1）:
   a. 从“定制”菜单，单击“连接器”，并单击“再分配传输层”。
   b. 选择适当的传输层条目，然后单击“保存”按钮。

有关特别的特定于应用程序的业务对象的其它配置要求，请参阅该业务对象的参考页。

注意:
您对连接器传送包中的开发对象所作的任何更改都应在 SAP 外部详细地记录。更改可能被下一发行版的适配器的传送文件覆盖，所以将需要手工重新应用。

验证连接器传送文件安装

要验证是否已实际上将连接器传送文件移至 SAP 应用程序，通过以下两种方法中的其中一种来检查传送日志:

- 要验证是否已实际上将连接器传送文件移至 SAP 应用程序，通过以下其中一种方法来检查传送日志:

  使用传送组织程序（事务 SE01）:
  1. 用传送文件的名称填充号码字段。
  2. 单击“显示”以查看日志。

  使用“传送管理系统”图形界面（事务 STMS）:
  1. 单击“导入概述”图标（F5）。

2. 双击适当的队列。
3. 右键单击传送包号，然后选择“日志”。
4. 检查日志以了解安装是否成功。

- 要验证 SAP 是否已成功地生成对象:
  1. 转至事务 SE38
  2. 输入 YXR_CNST 作为程序。
  3. 选择“源代码”，然后单击“显示”。
  4. 从“程序”菜单，单击“生成”。
  5. 单击“全部选中”，然后单击“继续”（F2）。

     这会生成包括这些程序的所有适配器程序。

  如果您得到响应程序已成功生成，则您可以假定传送已成功。

为连接器启用 SAP 应用程序

在安装连接器并配置标准的和特定于连接器的配置属性之后，您可以选择修改事件处理和从 SAP 应用程序内部将连接器的能力写入日志。

设置事件分布

负载均衡将事件和业务对象请求处理分布在多个连接器之间。IBM WebSphere Business Integration mySAP.com 适配器一次只能处理一个事务。因此，如果您设置多个连接器来处理特定业务对象，则可以同时处理多个事件和业务对象。有关设置多个连接器的更多信息，请参阅第 16 页的『安装多个连接器』。

要设置多个连接器的事件分布:

1. 转至“IBM WebSphere InterChange Server 连接器工具”窗口（事务 YXR1）。
2. 从“定制”菜单，单击“连接器”，然后单击“事件分布”。
3. 单击“新建条目”按钮（F5），然后在“新建条目”窗口中输入业务对象名、连接器名和 InterChange Server 名。
4. 在计数器字段中为每个业务对象输入一个数字。业务对象和计数器的组合为事件分布表提供唯一键。计数器可以是长度不超过 6 位数的任何数字。

注: 在测试环境中，您可以让多个用户测试多个连接器预订的同一业务对象。如果每个用户都只需要该业务对象的某个特定事件，则您可以指定用户名以区分将把哪个事件传递至连接器和集成代理程序的哪个组合。在“用户”（事件触发器）字段中，为业务对象输入适当的用户名。

设置事件过滤

SAP 应用程序中的配置表并非能适应所有修改，因此 IBM WebSphere Business Integration mySAP.com 适配器提供了一个 ABAP 包含程序，可以修改该程序来过滤事件。将从事件触发器 Y_XR_ADD_TO_QUEUE 内部调用此程序（YXRRESTR）来启用其它的事件过滤。

注: 因为需要重新编译代码，所以您必须具有开发者特权才能进行更改。

要查看或修改包含程序 YXRRESTR:

1. 转至“IBM WebSphere InterChange Server 连接器工具”窗口（事务 YXR1）。
2. 从“定制”菜单，单击“事件触发”，然后单击“修改限制”（事务 YXRS）。

可以使用此程序来配置许多可能的事件过滤。

设置事件优先级

您可以根据事件重要性来设置要处理的事件的优先级。通过设置业务对象、连接器和集成代理程序的每个组合的优先级，可以延迟连接器的事件检索。例如，如果您将事件的优先级设置为 10，则连接器在轮询事件表 10 次后才检索事件。因此，如果连接器每 5 秒轮询事件表一次，则连接器在 50 秒后选取事件。每次连接器轮询时，优先级值将减去一，直到检索并处理事件为止。

要设置事件的优先级:

1. 转至“IBM WebSphere InterChange Server 连接器工具”窗口（事务 YXR1）。
2. 从“定制”菜单，单击“连接器”，然后单击“事件分布”。
3. 用 1 至 99 之间的值为适当的业务对象填充“优先级”列。

增加日志表空间大小

缺省情况下，IBM WebSphere Business Integration mySAP.com 适配器的日志表位于名为 PSAPUSER1D 的表空间中，而索引则位于表空间 PSAPUSER1I 中。PSAPUSER1D 和 PSAPUSER1I 是为了供客户使用而保留的 SAP 应用程序表空间，但通常很小。由于缺省大小较小，所以这些表空间可以快速填充，这取决于活动的级别和适配器安装的日志记录级别。

要查看这些表空间的当前大小，转至事务 DB02，然后单击“当前大小”按钮。IBM WebSphere Business Integration 系统捕获的大量事件确定这些表空间所需要的大小。

如果缺省大小太小，则请求安装的 SAP 数据库管理员修改它们。

验证传送对象的号码范围

适配器有三个对象，它们必须在 SAP 应用程序内具有一个适当的号码范围。当安装传送包时，将设置以下对象及其缺省号码范围:

- YXR_EVENT
- YXR_IDOC
- YXR_LOG
- YXR_OBJARC

仅必须验证是否正确设置了相关的号码范围。要查看号码范围:

1. 转至事务 SNRO。
2. 用对象名（例如，YXR_EVENT）填充“对象”字段。
3. 单击“号码范围”，然后单击“时间间隔”。

注: 如果您在已生成事件的安装中重新安装主连接器传送包，则可以使用现有的事件标识创建新的事件。要防止发生此问题，关闭日志记录（事务 YXRM），然后在重

新导入连接器传送文件之前完全截断日志。一旦已成功装入连接器传送文件，则重新打开日志记录。有关截断事件日志的更多信息，请参阅第 234 页的『设置事件日志的截断』。

第 21 章 ABAP 扩展模块中的业务对象处理

本章讨论 ABAP 扩展模块的业务对象处理。本章提供了连接器如何处理业务对象的详细描述。编写本章的目的是为了显示 Java 和连接器的 ABAP 组件之间业务对象的进度。

注: 对连接器的 ABAP 组件的所有引用都使用 SAP R/3 V3.x 命名约定。

IBM WebSphere Business Integration mySAP.com 适配器（SAP R/3 V3.x）的扩展模块的业务对象处理对于所有业务对象都是相同的，而与使用的特定本机 SAP API 无关。例如，如果您基于“调用事务”或 IDoc 来开发业务对象，则将以相同的方式处理业务对象数据。无论是以作为事件通知一部分执行的检索形式还是以业务对象请求的形式将业务对象发送至 SAP 应用程序，处理都是相同的。业务对象的查询描述也不会更改该处理。

图 69 举例说明了特定于应用程序的业务对象至平面结构并接着回复到特定于应用程序的业务对象的转换和处理。注意，从 SAP 应用程序传递出来的业务对象数据必须与传递进来的数据具有相同结构，但数据可以不必具有相同的值。

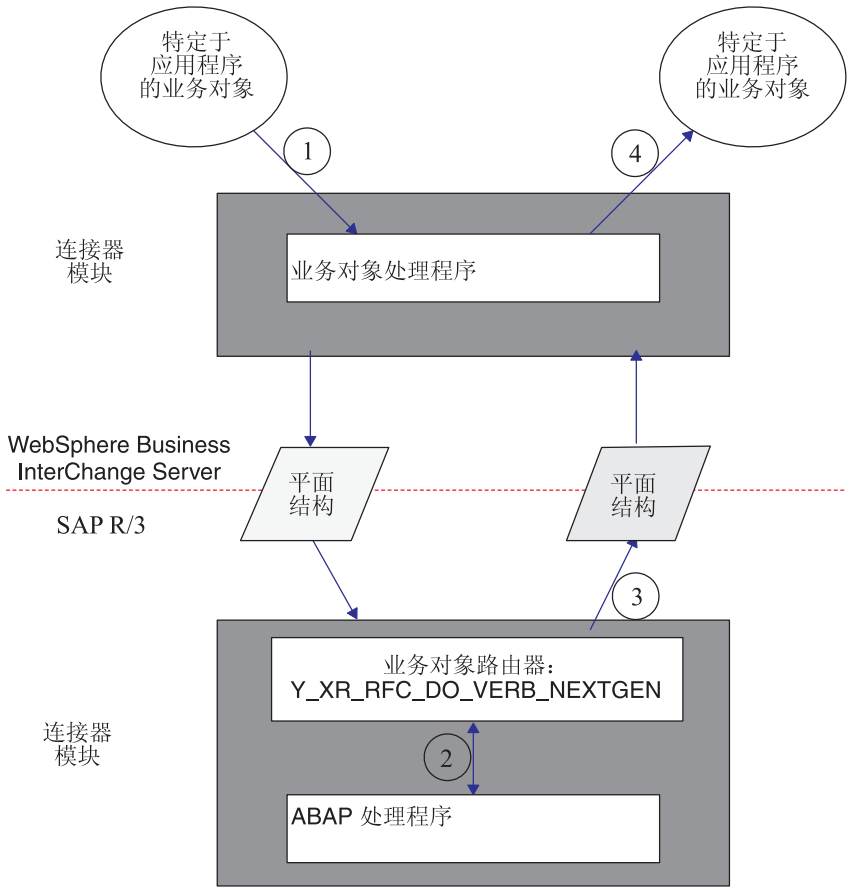


图 69. 业务对象处理

业务对象处理由四个步骤组成。下面列示的四个步骤对应于图 69 中的编号。

1. 连接器将特定于应用程序的业务对象转换为包含业务对象数据的平面结构，并将该数据传递至 SAP 应用程序。
2. 连接器的功能模块 Y_XR_RFC_DO_VERB_NEXTGEN 将业务对象数据动态地传递至 ABAP 处理程序。
3. ABAP 处理程序处理业务对象数据、生成业务对象响应数据并通过 Y_XR_RFC_DO_VERB_NEXTGEN 将新的业务对象数据返回至连接器。
4. 连接器接收新的业务对象数据，并使用该数据和特定于应用程序的业务对象的业务对象定义来创建新的业务对象以传递至集成代理程序。

业务对象至平面结构的转换

作为业务对象处理的第一步，连接器将业务对象转换为可以在 SAP 应用程序中处理的平面结构。平面结构的格式对于所有类型的业务对象（如基于事务调用的业务对象或基于 IDoc 的业务对象）都是相同的。平面结构是来自特定于应用程序的业务对象的重新格式化的数据。两种格式的数据之间的唯一差别是平面结构不保持父子业务对象关系。因此，连接器依靠一组规则来创建平面结构。

当将业务对象转换为平面结构时，连接器在内存中创建一个结构，然后用来自业务对象的数据填充该结构。在此过程中，连接器将以下数据从业务对象传递至 SAP 应用程序:

- 业务对象名
- 特定于应用程序的业务对象信息
- 业务对象查询描述
- 特定于应用程序的业务对象查询描述信息
- 属性名
- 属性特性 IsKey
- 属性特性 AppText
- 属性值

表 39 显示业务对象的通用平面结构。连接器在添加来自 WebSphere 业务对象的业务对象数据时使用此平面结构。

表 39. SAP 的 WebSphere 业务对象的通用平面结构表示

字段名	数据类型	长度	描述
ATTR_NAME	CHAR	32	属性名（例如 CustomerId）
BLANK1	CHAR	1	定界符
ATTR_VALUE	CHAR	200	属性值（例如 00000103）
BLANK2	CHAR	1	定界符
ISKEY	CHAR	1	1 = true, 0 = false; 仅限于属性
BLANK3	CHAR	1	定界符
ISNEW	CHAR	1	1 = 业务对象; 0 = 查询描述或属性
BLANK4	CHAR	1	定界符
PEERS	CHAR	6	指示一组业务对象中同级的数目
BLANK5	CHAR	1	定界符
OBJ_NUMBER	CHAR	6	未使用
BLANK6	CHAR	1	定界符
APPTEXT	CHAR	120	对象、查询描述或属性的特定于应用程序的信息
BLANK7	CHAR	1	定界符

注: BLANKn 字段名始终包含单个字符（CHAR）空格，且永远都不应填充。

要使数据转换能够正确进行，平面结构中的业务对象数据必须严格遵守一组规则。这些规则在以下初始数据转换步骤中进行定义:

- 每个业务对象属性按顺序放置在平面结构中，一行对应于一个属性。
- 分层业务对象将按先深度后宽度进行转换。

当连接器用业务对象数据填充平面结构时，连接器从顶级业务对象开始，对每个业务对象循环两次。

1. 在第一遍循环中，连接器设置所有简单属性。每个属性等同于平面结构中的一行。
2. 在第二遍循环中，连接器对每个子业务对象递归地执行步骤 1 中的相同处理。

表示子业务对象的属性不包括在它们的父代中。实际上，将作为完整的业务对象创建每个包含数据的子代。结果是按先深度后宽度来排序属性的单一列表。

图 70 举例说明了 SAP 的 WebSphere 业务对象的数据至平面数据结构的转换。数据转换始终遵循先深度后宽度的规则。在该示例中，顶级父业务对象 SAP_Order 具有两个子代 SAP_LineItem (1) 和 SAP_LineItem (2)，这两个子代被认为是同级的。SAP_LineItem (1) 具有一个子业务对象 SAP_ScheduleLines。

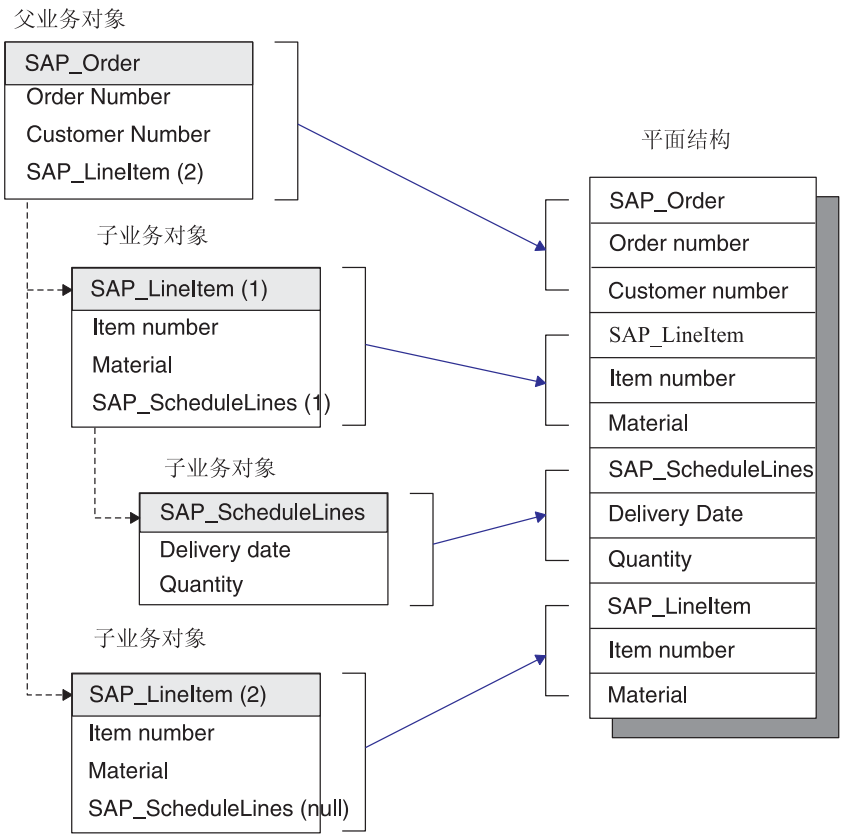


图 70. 从业务对象至平面结构的转换

当设计业务对象定义时，了解业务对象及其属性的排序很重要。下表举例说明了 WebSphere 业务对象至平面结构的转换结果。表 40 表示平面业务对象 SAP_Material 的平面结构，该对象的键值为 ItemID。在此示例中，该业务对象或任何属性没有特定于应用程序的信息。表 41 表示基于 IDoc 销售订单的分层业务对象的平面结构。

表 40. 平面业务对象 SAP_Material

ATTR_NAME	ATTR_VALUE	ISKEY	ISNEW	PEERS	OBJ_NUMBER	APPTEXT
BoName	SAP_Material	0	1	1	（空白）	（空白）
BoVerb	Retrieve	0	0	1	（空格）	:Y_XR_DYNAMIC_RETRIEVE
ItemID	000000000000001179	1	0	1	（空白）	（空白）
ShortDesc	CxIgnore	0	0	1	（空白）	（空白）
ObjectEventID	SAP_124	0	0	1	（空白）	（空白）

在此示例中，该业务对象或任何属性没有特定于应用程序的信息。

表 41. 基于 IDoc 销售订单的分层业务对象

ATTR_NAME	ATTR_VALUE	ISKEY	ISNEW	PEERS	OBJ_NUMBER	APPTEXT
BoName	SAP_Order	0	1	1	（空白）	YXRV4B01
BoVerb	Create	0	0	1	（空白）	[archive:methods]
Currency	USD	0	0	1	（空白）	E1EDK01:CURCY

表 41. 基于 IDoc 销售订单的分层业务对象 (续)

ATTR_NAME	ATTR_VALUE	ISKEY	ISNEW	PEERS	OBJ_NUMBER	APPTEXT
OrderId	CxIgnore	1	0	1	（空白）	E1EDK01:BELNR
ObjectEventId	SAP_124	0	0	1	（空白）	E1EDK01: ObjectEventId
BoName	SAP_LineItem	0	1	2	（空白）	Z1XRV40
BoVerb	Create	0	0	2	（空白）	（空白）
Createdby	User1	1		2	（空白）	Z1XRV40:ERNAM
ObjectEventId	SAP_125	0	0	2	（空白）	Z1XRV40: ObjectEventId
BoName	SAP_ ScheduleLines	0	1	1	（空白）	E1EDK14
BoVerb	Create	0	0	1	（空白）	（空白）
Qualifier	001	1	0	1	（空白）	Z1XRV40:QUALF
OrganizationId	1000	0	0	1	（空白）	E1EDK14:ORGID
ObjectEventId	SAP_126	0	0	1	（空白）	E1EDK14: ObjectEventId
BoName	SAP_LineItem	0	1	2	（空白）	Z1XRV40
BoVerb	Create	0	0	2	（空白）	（空白）
Createdby	User1	1	0	2	（空白）	Z1XRV40:ERNAM
ObjectEventId	SAP_127	0	0	2	（空白）	Z1XRV40: ObjectEventId

连接器将为每个业务对象添加前两行 BoName 和 BoVerb。BoName 和 BoVerb 是不能用作业务对象属性的关键字。

传递至 ABAP 处理程序的业务对象数据

一旦将业务对象数据转换为平面结构，将通过调用适配器的 ABAP 功能模块 Y_XR_RFC_DO_VERB_NEXTGEN 将业务对象数据传递至 SAP 内存。Y_XR_RFC_DO_VERB_NEXTGEN 不会处理业务对象数据；它只将该数据传递至适当的 ABAP 处理程序执行进一步的处理。在 Y_XR_RFC_DO_VERB_NEXTGEN 将业务对象数据传递至 ABAP 处理程序之后，它等待业务对象数据返回。

注: 记住，每个业务对象检索和请求是通过 Y_XR_RFC_DO_VERB_NEXTGEN 来处理的。

Y_XR_RFC_DO_VERB_NEXTGEN 使用特定于应用程序的业务对象查询描述信息来确定哪个 ABAP 处理程序处理业务对象数据。在运行时，Y_XR_RFC_DO_VERB_NEXTGEN 读取特定于应用程序的查询描述信息并将业务对象数据传递至指定的 ABAP 处理程序。

每个 ABAP 处理程序都必须一直使用连接器的特定于应用程序的查询描述信息。特定于应用程序的查询描述信息的格式如下:

:*function1*:*function2*:*function3*

其中 Y_XR_RFC_DO_VERB_NEXTGEN 执行 function1，并以参数的形式传递 function2 和 function3。例如，“客户更新”和“物料检索”仅执行 function1:

对于“创建”、“更新”或“删除”查询描述，指定 :Y_XR_RFC_DYNAMIC_TRANSACTION

对于“检索”查询描述，指定 :Y_XR_RFC_DYNAMIC_RETRIEVE

适配器提供的一个 ABAP 处理程序是功能模块 Y_XR_IDOC_HANDLER。此 ABAP 处理程序将平面结构的数据重新格式化为 IDoc 定义的实例，并将重新格式化的数据传递至另一个为处理该特定类型的 IDoc 而编写的 ABAP 处理程序。以下示例举例说明了 IDoc 处理程序 API 的用法:

销售订单更新 = :Y_XR_IDOC_HANDLER:Y_XR_ORDER_C2

销售订单检索 = :Y_XR_IDOC_HANDLER:Y_XR_ORDER_C4

在此示例中，将执行 Y_XR_IDOC_HANDLER 并传递第二个功能模块名称以及业务对象数据。Y_XR_IDOC_HANDLER 执行对第二个 ABAP 处理程序的调用，以将 IDoc 格式的业务对象数据传递至为处理订单对象而专门编写的 Y_XR_ORDER 功能模块。有关为 IDoc 处理程序设置查询描述支持的步骤，请参阅第 215 页的『使用 IDoc 开发业务对象』。

注: Y_XR_RFC_DO_VERB_NEXTGEN 仅使用 function1 的值。ABAP 处理程序可以使用 function2 和 function3。

要动态地调用 ABAP 处理程序，Y_XR_RFC_DO_VERB_NEXTGEN 要求每个 ABAP 处理程序的接口完全相同。这使 Y_XR_RFC_DO_VERB_NEXTGEN 能够发送和接收业务对象数据以及至任何 ABAP 处理程序的返回码和返回文本消息。有关功能模块接口的更多信息，请参阅第 207 页的『IBM WebSphere 功能模块接口』。

ABAP 处理程序如何处理业务对象数据

ABAP 处理程序的功能是将业务对象数据输入 SAP 应用程序数据库或从 SAP 应用程序数据库中输出。当处理业务对象数据时，ABAP 处理程序执行以下操作:

1. 解释业务对象数据。
2. 将数据与 SAP 本机 API 集成在一起。
3. 重新格式化从本机 API 返回的所有数据。

业务对象数据和 ABAP 处理程序

每个 ABAP 处理程序接收相同格式（平面结构）的业务对象数据。但是，每个 ABAP 处理程序都对业务对象具有特定要求，这些要求由 WebSphere 业务对象定义的复杂性、SAP 提供的本机 API 和 ABAP 处理程序提供的功能级别确定。由于这些原因，ABAP 处理程序可以通过对业务对象数据进行语法分析来将该数据解释为特定于业务对象的结构。这使 ABAP 处理程序能够更容易地处理数据。

注: 对数据进行语法分析不是必需的。但是，它简化了 ABAP 处理程序对业务对象的处理。

适配器提供了几个 ABAP 处理程序，如 IDoc 处理程序。IDoc 处理程序通过提供 ABAP 处理程序来利用 SAP 的 IDoc 技术，即，将业务对象数据重新格式化为基于 IDoc 的结构来解释业务对象数据，以便 ABAP 处理程序使用。

业务对象数据和 SAP 本机 API

一旦 ABAP 处理程序解释了业务对象数据，ABAP 处理程序就必须将该数据与 SAP 应用程序数据库集成在一起。它必须处理业务对象数据以使用 SAP 本机 API（如“调用事务”、BAPI 或 ABAP SQL）来将数据输入应用程序数据库或从应用程序数据库中输出。

创建、更新和删除处理

创建、更新或删除操作的目的是要修改 SAP 应用程序数据库。虽然给定业务对象的 SAP 应用程序数据库模式定义数据的结构，但由 SAP 提供的修改该数据的事务具有更广的影响范围。因此，直接修改 SAP 应用程序的应用程序数据库表可能会对应用程序的数据完整性造成灾难性的后果。

SAP 不会直接修改数据库表，而是为创建、更新和删除操作提供一个灵活的 ABAP API（调用事务）。“调用事务”是 SAP 提供的用于将数据输入 SAP 应用程序的功能。它通过使用联机用户将在事务中使用的相同屏幕来保证数据遵循 SAP 的数据模型。此过程通常称为屏幕搜集。

检索处理

如果查询描述是“检索”，则连接器使用 ABAP SQL 语句来从 SAP 应用程序数据库检索数据。当获取数据时，业务对象数据为 `where` 子句提供键。此检索数据的方法的困难在于必须以表示业务对象结构的格式表示检索的数据。将在 ABAP 处理程序 ABAP 代码中执行此操作。

重新格式化返回的业务对象数据

无论业务对象的查询描述是什么，连接器等待两种类型的确认:

- 返回码
- 返回的业务对象数据（仅限于成功情况，返回码 = 0）

如果 ABAP 处理程序返回非零代码，则不会将任何业务对象返回至连接器。如果 ABAP 处理程序处理成功，则连接器希望反映所执行操作的新业务对象数据。例如，在成功创建之后，返回的业务对象是最初发送进来的业务对象的精确副本，只是更新了键。同样，成功的检索将产生业务对象的结构完整实例。但是，创建、更新和删除操作对返回的业务对象具有与检索操作不同的需求。

当 InterChange Server（ICS）是集成代理程序时，需求的差别来自 WebSphere Business Integration 如何处理业务对象，尤其是如何处理映射期间对对象标识的动态交叉引用。当连接器在执行创建或更新操作之后将业务对象返回至 InterChange Server 时，映射基础结构尝试用新获得的对象标识来更新交叉引用表。将通过查找在将业务对象最初发送至连接器时设置的业务对象的 `ObjectEventId` 属性值来完成更新。

因为 ABAP 处理程序负责将对象标识与返回至连接器的业务对象“绑定”在一起，所以这对于 ABAP 处理程序很重要。通常，因为不存在相应的动态交叉引用，所以这对于检索操作不是问题。检索操作生成返回至连接器的全新业务对象。此业务对象与原始业务对象的结构没有任何直接关系。

ABAP 处理程序返回的业务对象数据必须处于与最初将该数据传递至功能模块 `Y_XR_RFC_DO_VERB_NEXTGEN` 时相同的平面结构格式。ABAP 处理程序只需要发送出简单类型属性以及每个属性的以下信息:

- 值
- 同级关系
- 特定于应用程序的信息

因为连接器仅使用特定于应用程序的信息来根据此数据创建业务对象，所以此时不需要属性名。将不使用也不应添加用于开始和结束业务对象或对象类型属性的标识。例

如，从 ABAP 处理程序返回的业务对象中将不使用 BoName 和 BoVerb 行。最初将它们传递至 ABAP 处理程序只是为了便于处理。

当 ABAP 处理程序用表示 WebSphere 业务对象的业务对象响应数据来填充平面结构时，它必须遵守下面的一组规则:

- 仅发送简单属性，不发送对象类型。
- 所有属性都必须在 WebSphere 业务对象定义中存在。
- 必须按属性在 WebSphere 业务对象定义中列示的顺序发送所有属性。
- 除非为子业务对象的父业务对象至少发送了一个属性，否则不能发送子业务对象的属性。
- 包含的业务对象必须告知它们具有的同级数。
- 属性名（字段 ATTR_NAME）不是必需的。

图 71 举例说明了平面业务对象（无对象类型属性）。

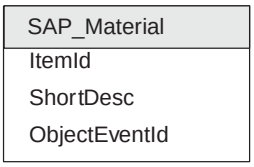


图 71. 平面业务对象 SAP_Material

表 42 表示平面业务对象 SAP_Material 的结构，该对象的键值为 ItemID。注意，字段 ATTR_NAME 不是必需的，APPTEXT 对于每个属性是唯一的。并且，由于此业务对象是平面的，所以 PEERS 字段可以保留为空白。

表 42. 平面业务对象 SAP_Material

ATTR_NAME	ATTR_VALUE	ISKEY	ISNEW	PEERS	OBJ_NUMBER	APPTEXT
（空白）	000000000000001179	（空白）	（空白）	（空白）	（空白）	ItemId
（空白）	Toaster 6000	（空白）	（空白）	（空白）	（空白）	ShortDesc
（空白）	SAP_124	（空白）	（空白）	（空白）	（空白）	ObjectEventId

图 72 举例说明了分层业务对象（包含对象类型）。

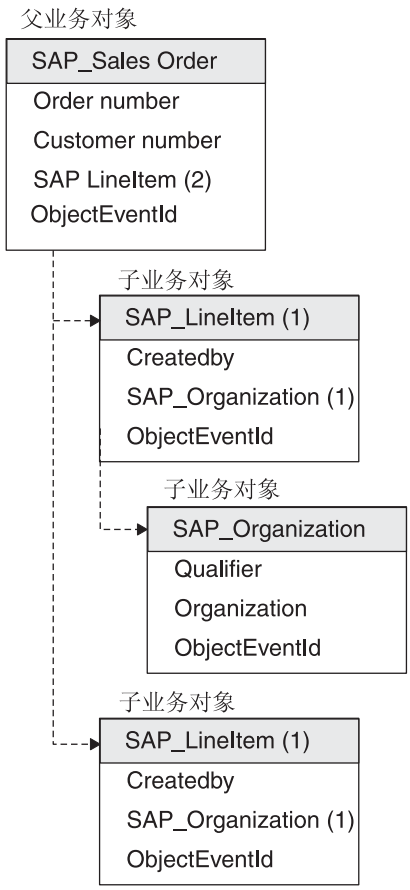


图 72. 分层业务对象 SAP 销售订单（IDoc）

表 43 显示基于 IDoc 销售订单的分层业务对象的平面结构表示。注意，字段 ATTR_NAME 不是必需的，APPTEXT 对于每个属性是唯一的。并且，由于此业务对象是分层的，所以 PEERS 字段列示适当的关系。

表 43. 基于 IDoc 销售订单的分层业务对象

ATTR_NAME	ATTR_VALUE	ISKEY	ISNEW	PEERS	OBJ_NUMBER	APPTEXT
（空白）	USD	0	0	1	（空白）	E1EDK01:CURCY
（空白）	0000000101	0	0	1	（空白）	E1EDK01:BELNR
（空白）	SAP_124	0	0	1	（空白）	E1EDK01: ObjectEventId
（空白）	User1	0	0	2	（空白）	Z1XRV40:ERNAM
（空白）	SAP_125	0	0	2	（空白）	Z1XRV40: ObjectEventId
（空白）	001	0	0	1	（空白）	Z1XRV40:QUALF
（空白）	1000	0	0	1	（空白）	E1EDK14:ORGID
（空白）	SAP_126	0	0	1	（空白）	E1EDK14: ObjectEventId
（空白）	User1	0	0	2	（空白）	Z1XRV40:ERNAM
（空白）	SAP_127	0	0	2	（空白）	Z1XRV40: ObjectEventId

平面结构至业务对象的转换

一旦用新的业务对象数据重新填充了平面结构，Y_XR_RFC_DO_VERB_NEXTGEN 就会将业务对象数据返回至调用连接器。记住，连接器是单线程的；因此，它一次仅传递一个业务对象。连接器现在必须将业务对象数据从平面结构转换为业务对象。当将平面结构中的数据处理为业务对象时，连接器必须完成下列操作:

1. 初始化原始业务对象。
2. 将业务对象数据从平面结构传送至业务对象。
3. 将业务对象传递至连接器基础结构。

业务对象初始化

连接器在填充从集成代理程序接收到的原始业务对象之前，将先初始化该业务对象。当初始化业务对象时，连接器将顶级业务对象中的每个属性都设置为空。对于对象类型属性，此操作递归地删除包含的每个业务对象，仅保留顶级业务对象。

连接器如何重建业务对象

在连接器初始化原始业务对象之后，保留下来的是包含业务对象名和业务对象查询描述的顶级业务对象，但没有属性值数据。必须通过 ABAP 处理程序从平面结构传送属性值数据。传送返回数据的逻辑很简单，但必须严格按照连接器期望数据的顺序来传送这些数据。

连接器使返回数据中特定于应用程序的信息与业务对象定义中属性的特定于应用程序的信息匹配。连接器尝试设置返回的业务对象数据中的每个属性。如果不能设置任何属性，则连接器将 FAIL 返回到连接器基础结构。

为了成功传送返回的数据，连接器希望返回的数据符合以下条件:

- 它仅包含简单属性，其中一行表示一个属性。
- 属性必须在 WebSphere 业务对象定义中存在。
- 必须按属性在 WebSphere 业务对象定义中排序的方式（先深度后宽度）来排序属性。
- 属性的特定于应用程序的信息将其对象的特定于应用程序的信息与在业务对象的定义内唯一标识该属性的另一个值链接在一起。
- 子属性必须出现在它的父对象的属性之后（永远不能在其父代之前和其祖父之后）。
- 属性必须告知其业务对象的同级数。

当连接器重建特定于应用程序的业务对象时，连接器从顶级业务对象开始，对该业务对象循环两次。

1. 在第一遍循环中，连接器设置所有简单属性。
2. 在第二遍循环中，它检查平面属性在子对象中是否存在。如果它存在，则连接器对子对象递归地执行相同处理。

警告: 如果平面结构至业务对象的转换失败，则连接器向集成代理程序报告故障。但是，数据已传送至 SAP 应用程序，因此，在此阶段不能回滚。尽管规则很简单，但实现具有许多属性的复杂分层业务对象可能难以管理。

一旦用新业务对象数据成功地重建业务对象，连接器就会将该业务对象返回至集成代理程序。

第 22 章 为 ABAP 扩展模块开发业务对象

本章讨论 ABAP 扩展模块的业务对象开发。本章提供了背景信息以及开发业务对象和 ABAP 处理程序的步骤。本章的内容应该用作业务对象开发的准则。您应该熟悉连接器如何处理业务对象。

注: 对连接器的 ABAP 组件的所有引用都使用 SAP R/3 V3.x 命名约定。

本章包含以下各节:

背景信息

ABAP 扩展模块的业务对象开发包括为您要支持的每个查询描述创建特定于应用程序的业务对象定义和相关的 ABAP 处理程序。

要开发特定于应用程序的业务对象，您必须创建支持您的业务需要的业务对象定义。IBM WebSphere Business Integration mySAP.com 适配器（SAP R/3 V3.x）包括一些工具，这些工具有助于在 SAP 应用程序中开发业务对象定义。尽管您可以使用业务对象设计器或文本编辑器来为 ABAP 扩展模块创建业务对象定义，但我们建议您最初使用适配器的业务对象开发工具。这些工具使用 SAP 应用程序的本机定义作为模板。

对于您可以开发的每个特定于应用程序的业务对象定义，您必须通过使用适配器提供的 ABAP 处理程序或通过开发定制 ABAP 处理程序来支持它。ABAP 处理程序是将数据输入和输出 SAP 应用程序数据库的机制。

注: 特定于应用程序的业务对象和 ABAP 处理程序依靠彼此的一致性，将数据传递至 SAP 应用程序和从 SAP 应用程序中传递出来。因此，如果您更改业务对象定义，则您必须更改 ABAP 处理程序以支持该业务对象定义。

以 ABAP 功能模块的形式实现适配器的 ABAP 处理程序。ABAP 处理程序是一个或多个功能模块，这些模块协同工作以完成来自业务对象路由器 `Y_XR_RFC_DO_VERB_NEXTGEN` 的业务对象检索或请求。ABAP 处理程序负责将业务对象数据传递至 SAP 应用程序和从 SAP 应用程序中传递出来。

注: 除了受 WebSphere Business Integration 系统支持的那些查询描述（创建、检索、更新和删除）以外，SAP 支持许多查询描述。您可以开发 ABAP 处理程序来支持任何查询描述。

要开发 ABAP 处理程序，您必须了解连接器如何将数据输入和输出 SAP 应用程序以及在此过程中该数据采用哪种格式。有关业务对象处理的进一步描述，请参阅第 179 页的

的第 19 章，『ABAP 扩展模块概述』。有关业务对象处理的详细描述，请参阅第 195 页的第 21 章，『ABAP 扩展模块中的业务对象处理』。

注: 当开发业务对象时，您必须确保在 SAP R/3 应用程序中将这些对象添加至连接器的 YXROBJ 表。如果未添加它们，则您将无法访问业务对象以进行定制（例如，设置对象以进行事件分布）。

SAP 本机 API

适配器提供的 ABAP 处理程序使用 SAP 本机 API，这些 API 使 ABAP 处理程序能够将数据传递至 SAP 应用程序和从 SAP 应用程序中传递出来。WebSphere Business Integration 系统已实现以下本机 API:

- 『ABAP SQL』
- 『调用事务』
- 『批处理数据通信（BDC）』

ABAP SQL

ABAP SQL 是 SAP 独有的 SQL 版本。它独立于数据库和平台，所以无论您编写哪一种 SQL 代码，您都可以在 SAP 支持的任何数据库和平台组合上运行它。ABAP SQL 的语法与其它版本的 SQL 类似并支持诸如更新、插入、修改、选择和删除之类的所有基本数据库表命令。有关 ABAP SQL、其用途、语法和功能的完整描述，请参阅 SAP 文档。

通过使用 ABAP SQL，ABAP 处理程序可以修改具有业务对象数据的 SAP 数据库表以执行创建、更新和删除操作，并同样可以在 ABAP select 语句中的 “where” 子句中将业务对象数据用作键。

注: WebSphere Business Integration 系统从不使用 ABAP SQL 来修改 SAP 表，因为这可能会损坏数据库的完整性。连接器使用 ABAP SQL 仅检索数据和修改适配器提供的数据库表。

调用事务

“调用事务”是 SAP 提供的用于将数据输入 SAP 系统的功能。“调用事务”通过使用联机用户在事务中看到的相同屏幕来保证数据遵循 SAP 的数据模型。此过程通常称为屏幕搜集。要使用“调用事务”，指定以下类型的指令:

- 开始 – 要调用的事务
- 导航 – 要处理的屏幕的顺序
- 映射 – 应填写到屏幕上每个字段的输入数据

“开始”将在“调用事务”调用中以单值参数的形式传递。“导航”和“映射”指令以特定格式在一个表中一起进行传递；此格式可用于为任何 SAP 事务调用“调用事务”。在此格式中，这些指令称为 BDC 数据、BDC 表或 BDC 会话。

批处理数据通信（BDC）

批处理数据通信（BDC）是一个指令集，SAP 可以遵循该指令集来执行事务而不需要用户介入。这些指令指示处理事务的屏幕的顺序和应该使用哪些屏幕上的数据来填充哪些字段。SAP 事务中对联机用户公开的所有元素都具有可以在 BDC 中使用的标识。这些元素如下:

- 屏幕 – 由程序名和屏幕号标识。
- 输入字段 – 通常由数据库表和该表引用的字段名标识。
- 事务中的命令 – 一些命令，如保存、新建项、详细信息和退出（由 1 至 8 个字符代码标识）

要获取屏幕的 BDC 标识，将光标放在屏幕上的任何字段中。按 F1 获取帮助，然后按 F9 获取技术信息。程序名和屏幕号列示在“屏幕数据”之下。

要获取输入字段的 BDC 标识，将光标放置在屏幕上您要输入数据的每个字段中。按 F1 获取帮助，然后按 F9 获取技术信息。如果有一个名为“批处理输入的字段描述”的框，则使用“屏幕字段”字段中的信息。如果此框不存在，则从“字段数据”框用连字号将表名和字段名连接在一起。

要获取某个命令的 BDC 标识，在菜单中突出显示该命令，并按 F1 获取帮助。使用“功能”字段中的值。

IBM WebSphere 功能模块接口

每个 ABAP 处理程序都必须实现相同的功能模块接口。功能模块接口保证业务对象路由器 Y_XR_RFC_DO_VERB_NEXTGEN 可以将业务对象数据传递至 ABAP 处理程序和从 ABAP 处理程序中传递出来。该接口为:

```
*"*"Local interface:
*"  IMPORTING
*"      VALUE(PROC_FUNC_1) LIKE  RS38L-NAME OPTIONAL
*"      VALUE(PROC_FUNC_2) LIKE  RS38L-NAME OPTIONAL
*"      VALUE(OBJECT_NAME) LIKE  YXR_LOG_H-OBJ_NAME OPTIONAL
*"      VALUE(OBJECT_VERB) LIKE  YXR_CHANGE-OBJ_VERB OPTIONAL
*"      VALUE(ARCHIVE) OPTIONAL
*"      VALUE(TEXT) LIKE  T100-TEXT OPTIONAL
*"  EXPORTING
*"      VALUE(RETURN_TEXT) LIKE  YXR_EVENT-OBJ_KEY
*"      VALUE(RFCRC) LIKE  YXR_RFCRC-YXR_RFCRC
*"  TABLES
*"      RFC_STRUCTURE STRUCTURE  YXR_RFC_S
*"  EXCEPTIONS
*"      NOT_FOUND
*"      ERROR_PROCESSING
```

在该接口的导入节中，您可以传送诸如 ABAP 处理程序名、业务对象名和业务对象等值。

该接口的导出节用来传送 ABAP 处理程序处理的结果。返回码 RFCRC 参数是用来确定连接器返回的代码的单个字段。可能的值有:

RC = 0（成功，VALCHANGE）

RC = 1（失败，FAIL）

RETURN_TEXT 参数是一个具有 120 个字符的自由文本字段，该字段由连接器写入或在返回状态描述符中记录为错误消息。如果 ABAP 处理程序没有为此参数提供值，则 Y_XR_RFC_DO_VERB_NEXTGEN 根据返回码提供缺省文本。

注: 该接口的异常节定义两个异常。建议您使用导出参数代替。

IBM WebSphere ABAP 处理程序 API

适配器包括几个 API，这些 API 有助于开发支持 SAP 的 WebSphere 业务对象的 ABAP 处理程序。这些 API 是作为“通用” ABAP 处理程序开发的，因为它们仅需要元数据就能支持其它任何类型的业务对象。适配器包括以下 ABAP 处理程序 API:

- 动态检索 – Y_XR_DYNAMIC_RETRIEVE
- 动态事务 – Y_XR_DYNAMIC_TRANSACTION
- IDoc 处理程序 – Y_XR_IDOC_HANDLER

适配器包括一组支持这些 API 的工具。可以在 IBM WebSphere InterChange Server 连接器工具（事务 YXR1）中找到这些工具。以下各节讨论了适配器提供的 API 并介绍了如何使用 IBM WebSphere InterChange Server 连接器工具（事务 YXR1）来为这些 API 开发业务对象的步骤。

使用动态检索开发业务对象

“动态检索”功能模块是一个带有动态 SQL 语句生成器的映射工具。此功能模块使用存储在 YXR_DISPLY 表中的元数据在运行时生成 SQL select 语句。“动态检索”获取这些 SQL 语句产生的字段并填充 WebSphere 业务对象的属性。当调用“动态检索”功能模块时，将执行下列步骤:

1. 从 YXR_DISPLY 中检索所有条目，其中，

   对象名 = *objectName*

2. 对于在 YXR_DISPLY 中指定的每个新表，将根据已标记为键的字段生成 SQL where 子句。业务对象属性中相应的字段名用来填充 where 子句中的值。如果在 YXR_DISPLY 中指定了缺省值，则将使用此缺省值。
3. 执行 SQL select 语句。产生的字段将复制到业务对象属性中相应的字段名中。

注: 您必须使用 IBM WebSphere InterChange Server 连接器工具窗口（YXR1）来创建 WebSphere 业务对象，然后才能生成业务对象定义。

提示

- 必须按表名来将 YXR_DISPLY 中的条目进行分组。所有相关键值都必须首先出现。
- 如果来自特定表的数据是可选的，则必须将该表的所有非键字段标记为可选的。
- where 子句的缺省操作数是等号。可以在 Rel 列中指定其它操作数。使用 F4 下拉列表以获取可能的操作数。
- 可以在“系统”列中指定诸如系统语言（LANGU）、当前日期（DATUM）或当前时间（UZEIT）等系统字段。按 F4 键以获取更多选项。这仅适用于键字段。
- 如果您要执行存在检查，则至少指定一个非键字段，即使您仅关心 select 语句的成功或失败。
- 从一个表读取的数据可以用于后一表的 where 子句中。

表 44 显示“动态检索”表的表条目。

表 44. 动态检索的表条目

字段名	描述	何时使用	技术名称
对象名	WebSphere 业务对象名	始终	OBJ_NAME

表 44. 动态检索的表条目 (续)

字段名	描述	何时使用	技术名称
计数器	计数器	始终	POSNR
表	要读取的表	始终	TABNAME
字段名称	表中的字段名	始终	NAME_FELD
键	指定表的键字段	每个表的第一个字段。用来构建 select 语句的 where 子句。	KEYFLAG
可选的	屏幕、字段或命令的自由文本描述	用于非键字段。如果所有非键字段都标记为可选的，并且如果 select 语句失败，则结果将为警告，而不是错误。	OPTIONAL
Rel	操作数（关系）	用来确定 where 子句中的关系。缺省情况下，where 子句中的操作数是“等号”，但可以通过在此字段中输入值来更改它。	YXR_OPERND
业务对象中的字段名	WebSphere 业务对象中的属性，用于提供输入值	• 键字段：用于构建 where 子句。 • 非键字段：用于将数据库字段返回至业务对象中的属性	SOURCEFLD
缺省值	当未在 WebSphere 业务对象中提供条目时要使用的静态缺省值	键字段	DEFLT_VAL
SY 字段	将用作缺省值（例如：DATUM）的动态系统字段	键字段	SYFIELD
长度	字符长度，从 0 到构建 where 子句时应使用的属性值的偏移位置。	键字段。仅当使用包含组合值的属性时才相关。	LENGTH
偏移	字符偏移，从构建 where 子句时应使用的属性值的 0 位置算起。	键字段。仅当使用包含组合值的属性时才相关。	YXR_OFFSET

要访问适配器的表驱动的连接器表以便显示:

1. 转至“IBM WebSphere InterChange Server 连接器工具”窗口（事务 YXR1）。
2. 从“定制”菜单，单击“动态读取”，然后单击“修改检索”。

图 73 显示使用嵌套 SQL select 语句的“动态检索”表。先前的 select 语句的结果用来在后续 select 语句上构建键。

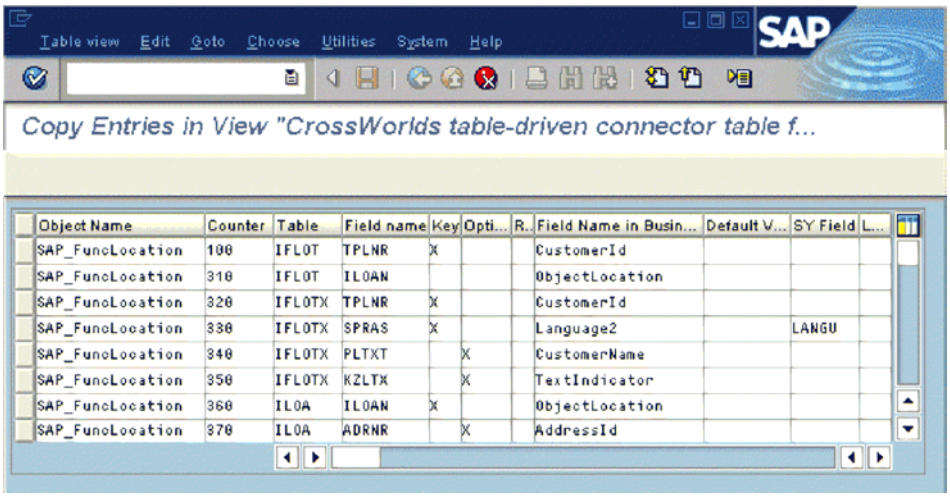


图 73. 使用嵌套 SQL select 语句的动态检索

通过使用“计数器”列作为要讨论的行号，您可以对“动态检索”表中的功能位置对象逐步执行 SAP_FuncLocation 示例。

100 表 IFLOT 只具有一个键字段，即 where 子句中 CustomerId 属性中的值。如果 **CustomerId** 中的值是 4711，则 where 子句为:

```
where TPLNR = '4711'
```

310 这是第一个非键字段。将在此处执行实际的 select 语句。select 语句为:

```
Select * from IFLOT where TPLNR = '4711'
```

然后将把产生的 ILOAN 值复制到 ObjectLocation 属性，对于此示例，该值为 ’5678’。

320 因为这是新表和键字段，所以将再次构建 where 子句。where 子句再次为:

```
where TPLNR = '4711'
```

330 表 TFLOTX 的另一个键。将使用以下内容扩展 where 子句:

```
and SPRAS = 'E'
```

如果 Language2 的值是 CxIgnore，则将从“缺省值”字段中获取 E。

340 这是表 IFLOTX 的第一个非键字段。将执行 select 语句。该语句为:

```
Select * from IFLOTX where TPLNR = '4711' and SPRAS = 'E'
```

产生的 PLTXT 值将复制到 CustomerName 属性。如果该 select 语句失败，则因为表 IFLOTX 的所有非键字段都标记为可选的，所以将发出一个警告。

350 来自先前 select 语句的 KZLTX 值将复制到 TextIndicator 属性。

360 因为这是新表和键字段，所以将再次构建 where 子句。where 子句的值将从属性 ObjectLocation 中获取，该属性是由先前的 select 语句填充的。如果 '5678' 是属性 ObjectLocation 中的值，则 where 子句为:

```
where ILOAN = '5678'
```

370 这是第一个非键字段。将执行 select 语句。该语句为:

```
Select * from ILOA where ILOAN = '5678'
```

产生的 ADRNR 值将复制到 AddressId 属性。

这将完成 SAP_FuncLocation 功能模块的构建以获取“动态检索”支持。

使用动态事务开发业务对象

“动事务索”功能模块是一个带有动态代码生成器的映射工具。它使用 SAP 的“调用事务”API 来将数据输入 SAP 应用程序。并且，它按对象／详细描述组合来存储“批处理数据通信”（BDC）会话的静态定义。在将 BDC 数据传递至“调用事务”之前，将把业务对象属性值映射至 BDC 会话。在完成调用事务时，将在业务对象的适当值中设置产生的键值，并将来自调用事务的所有消息写入日志。

注: 如果“调用事务”失败，则连接器不会将 BDC 会话存储在 SAP 中重新处理。废弃版本的连接器代理程序会存储 BDC 会话；但使用的是在交叉引用和请求处理中导致不一致的存储会话。

“动态事务”功能模块通过组合在“动态事务”表 YXR_CHANGE 中定义的 BDC 和来自入局业务对象的值，构建 BDC 会话来执行调用事务。当调用“动态事务”功能模块时，将执行下列步骤:

1. 从 YXR_CHANGE 中检索所有条目，其中:

   对象名 = *objectName*，查询描述 = *objectVerb*
2. 根据属性名将字段输入值从业务对象映射至 BDC 会话。
3. 已使用“调用事务”来处理 BDC 会话。
4. 捕获键值，将“调用事务”消息写入日志并在业务对象中设置键。

注: 您必须使用 IBM WebSphere InterChange Server 连接器工具窗口（YXR1）来创建 WebSphere 业务对象，然后才能生成业务对象定义。

提示

- 在初始屏幕上输入的数据可以是所有行项的缺省值，因而减少了需要的行项输入。
- 行项概述屏幕可以提供足够的输入，而不是向下查找到可能需要其它输入的详细信息屏幕。
- 确认消息在 BDC 中通常不需要回答；例如，Are you sure you want to save?
- 在您每次以更改方式进入和退出表维护时，对于每个对象和查询描述组合，计数器以 10 为增量重新编号。
- 在执行期间，“调用事务”使用用户设置，将日期格式化。确保连接器用户已设置为使用 YYYY-MM-DD 日期格式的变体。这是由 WebSphere Business Integration 系统使用的标准日期格式。同样，如果您要通过逐步执行事务来重新处理业务对象，则更改您自己的用户设置。

为业务对象编写 BDC 会话

编写 BDC 会话需要了解 SAP 事务的设计。SAP 事务允许以各种顺序和在不同屏幕上输入相同的数据。通常，每种顺序或流都会显示额外的功能。因此，在某些屏幕上将进行验证某种数据并且需要输入字段，而在其它屏幕上则不需要。难处在于要找出最省力的顺序。简单的 BDC 会话比复杂的 BDC 会话更稳定。

当 SAP 事务在后台进程中使用“调用事务”方法进行访问而不是联机进行访问时，它的行为可能会不同。例如，可能会出现不同的或额外的屏幕，或输入字段可能位于与显示的联机调查不同的屏幕上。出现这种差异的原因是因为在后台执行而不是联机执行时，事务的控制代码可能指示不同的行为。因此，与您逐步执行事务一样，当重新处理失败的业务事件时，联机测试可能起作用，但是，当处理同一对象时，连接器仍会失败。如果发生这种错误，则修改 BDC，以便它在后台进行处理。如果您修改 BDC，您可能会遇到 BDC 在后台进行处理但联机处理时仍失败的情况。

您在“动态事务”表中定义的 BDC 是静态的。如果某些输入数据导致弹出其它屏幕或其它字段在运行时变为必填字段，则它在事务期间不能作出反应。正确地调查事务的配置对于能否预测一致的行为非常重要。对事务进行几次试验，那么重复的行为可以成为您的准则。

一旦确定了屏幕流，则遵循以下步骤并将您收集的信息记录在电子表格中。

1. 转至支持您的对象的事务并标识事务代码。
2. 标识您需要的屏幕和输入字段的 BDC 元素。
3. 标识您继续处理到下一屏幕所需要的菜单命令。
4. 对每个必需的屏幕重复步骤 2 和 3。
5. 完成时记下保存事务的命令。

表 45 描述动态事务表 YXR_CHANGE 的列名。

表 45. YXR_CHANGE 表的动态检索条目

字段名	描述	何时使用	技术名称
对象名	WebSphere 业务对象名	始终	OBJ_NAME
查询描述	查询描述（创建、更新、删除或检索）	始终	OBJ_VERB
计数器	计数器	始终	POSNR
程序	与屏幕相关的程序	BDC 屏幕标识	PROG_NAME
屏幕号	与屏幕相关的屏幕号	BDC 屏幕标识	DYNPRO
开始	指定新的屏幕	BDC 屏幕标识	DYNBEGIN
BDC 字段名	BDC 输入字段名	BDC 输入字段	FNAM
缺省值	静态缺省值，该值在 IBM WebSphere 业务对象中未提供条目或者因为条目是命令值而使用 BDC_OKCODE 时使用	可能不会始终传递进一个值，而该值对于事务是必需的	DEFLT_VAL
SY 字段	将用作缺省值（例如：DATUM）的动态系统字段	未传递进值，或者应该由 SAP 系统字段确定值	SYFIELD
返回	一个 1 至 4 之间的数字，它标识哪个系统消息字段在事务完成时返回键值（sy-msgv#）	应接收键值的业务对象键属性	RETURNFLD
业务对象中的字段名称	WebSphere 业务对象中的属性，用于提供输入值	BDC 输入字段	SOURCEFLD
长度	字符长度，从应该用于输入的属性值的 0 位算起。	仅当使用包含组合值的属性时才相关	LENGTH

要定义或修改业务对象的元数据（将信息传送至 YXR_CHANGE）：

1. 转至“IBM WebSphere InterChange Server 连接器工具”窗口（事务 YXR1）。

2. 从“定制”菜单，单击“调用事务”，然后单击“修改”、“创建”或“更新”。

定义业务对象的元数据很简单。对于每个屏幕，第一个条目标识屏幕，后面的条目标识输入字段，最后一个条目必须是命令。此分组对每个屏幕重复。

图 74 显示 SAP CustomerMaster 业务对象用来创建客户（事务 XD01）的动态事务表。

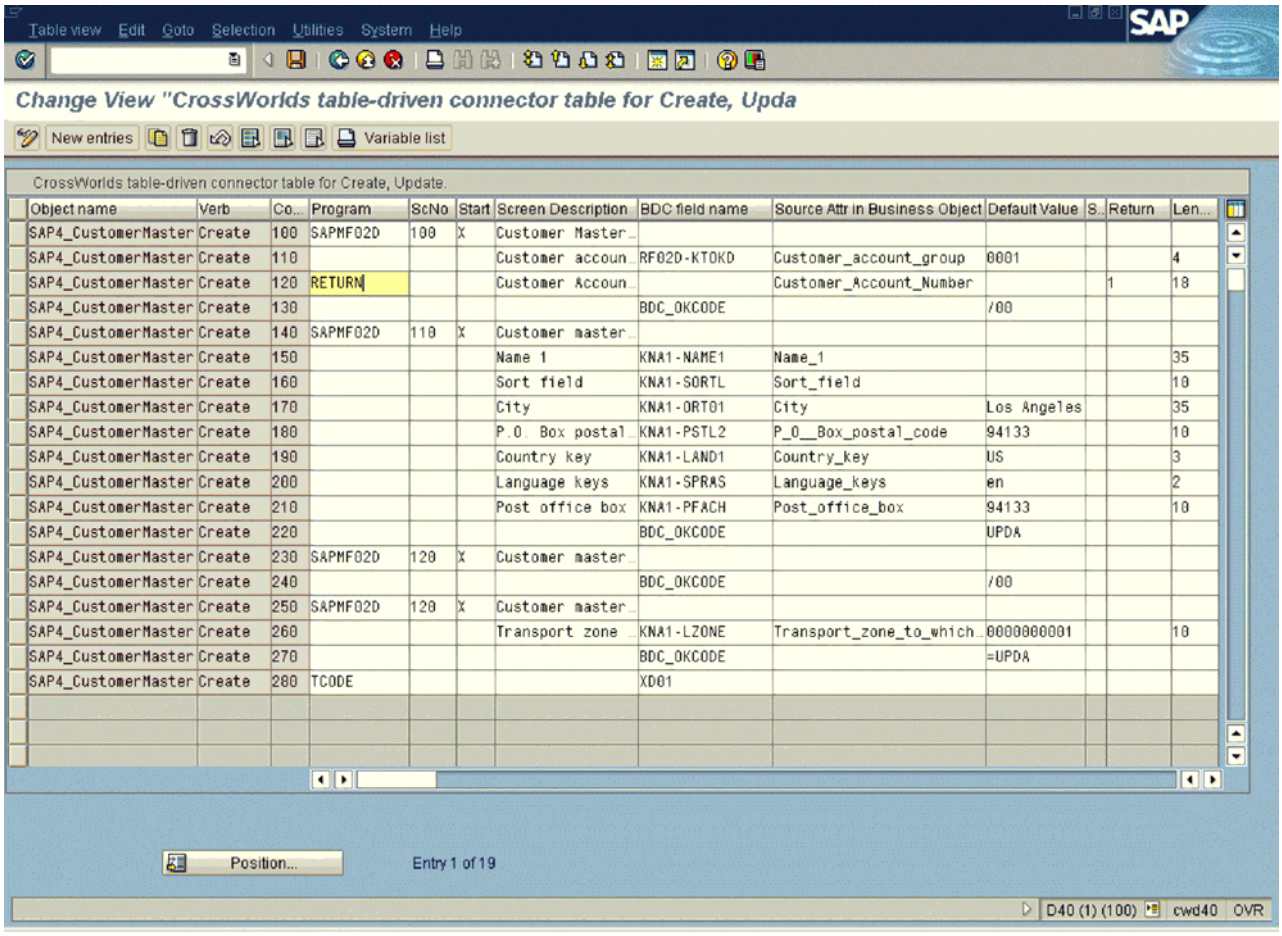


图 74. SAP CustomerMaster 业务对象的动态事务

通过使用“计数器”列作为要讨论的行号，逐步执行 SAP CustomerMaster 示例。

100 从程序 SAPMF02D 的屏幕号 100 开始。这是一个新屏幕，即第一个屏幕，因此在“开始”列中标记了它。

110 在屏幕 110 上，使用来自业务对象中 Customer_account_group 属性的值，并将它添加至“BDC 字段名”列（值为 RF02D-KTOKD）。将缺省值指定为 0001。如果 Customer_account_group 属性包含 CxIgnore，则“BDC 字段名”列接收缺省值 0001

120 Customer_Account_Number 属性是键值，因此在“调用事务”期间不设置它。SAP 在内部指定该键值，并仅在成功记入事务之后才使该键值可用。因此，将“BDC 字段名”列保留为空白，但应在表中包括一个条目，因为当结束“调用事务”返回 Customer_Account_Number 属性时，必须使用此键值来设置该属性。还应在 CustomerNumber 的“程序”列中输入单词 RETURN。

根据事务不同，SAP 在以下四个可能字段的其中一个字段中返回键值: SY-MSGV1、SY-MSGV2、SY-MSGV3 或 SY-MSGV4。要指定您想在特定属性中设置返回值，在“返回”列中输入一个 1 至 4 之间的数字。此数字对应于包含键值的 SY-MSGV# 字段。例如，图 74 举例说明了“返回”列中的 1，它指示 SY-MSGV1 包含客户号。

130 您完成了为第一个屏幕输入必要的值，因此在“缺省值”列中输入命令 /00 以模拟按 Enter 键。这使您进入下一个事务屏幕。在屏幕输入字段中输入命令 BDC_OKCODE，该字段是您输入事务代码的位置。

140 此时，您在下一个事务屏幕中。输入地址信息。因为它是新屏幕，所以在“开始”列中标记它。在此示例中，第二个屏幕和与初始屏幕相同的程序相关，只是屏幕号从 100 更改为 110。但情况并非总是如此。

150 -210 在业务对象中使用来自 Name_1、Sort_field、City、P_0_Box_postal_code、Country_key、Language_keys 和 Post_office_box 属性的值，并将相应的值添加至“BDC 字段名”列。

220 与行 130 类似，此屏幕的处理已完成。但是，输入命令值 UPDA 以保存事务，而不是只模拟 Enter 键。这使您进入下一个事务屏幕。

230 此时，您在第三个事务屏幕中，因此在“开始”列中标记它。因为您的业务对象不需要此屏幕中的数据，因此，您将在下一行中完成对此屏幕的处理。

240 与行 130 类似，此屏幕的处理已完成。输入命令值 /00 以模拟按 Enter 键。这使您进入最终的事务屏幕。

250 此时，您在最终的事务屏幕中。在“开始”列中标记它。

260 与行 150-210 类似。使用来自业务对象属性 Transport_zone_to_which_or_from_which_the_goods _are_delivered 的值，并将其相应值（KNA1-LZONE）添加至“BDC 字段名”列。

270 与行 220 类似，对此屏幕的处理已完成且事务也已完成，因此输入命令值 UPDA 进行保存。这是“调用事务”API 接收的最后一个操作。

280 任何业务对象的最后条目总是事务代码的说明。关键字 TCODE 在“程序”列，而事务代码在“BDC 字段名”列。

这样就完成了 SAP4_CustomerMaster 业务对象的 BDC 会话定义。

如果“调用事务”在失败时返回一条错误消息，您可能遇到以下描述的其中一个常见错误。

- SAP 应用程序调用了 BDC 不需要的屏幕，因此 SAP 应用程序返回消息没有可用于程序 XX 和屏幕 YY 的输入。如果发生这种错误，则将适当的条目添加至“动态事务”表来处理程序 XX 和屏幕 YY 的输入屏幕。
- SAP 应用程序根据 BDC 的指示来设置不存在的字段。最可能的情况是，SAP 应用程序执行了它自己的指令，而您未显式设置该指令。因此，您所在的屏幕不是预计的屏幕。如果发生这种错误，则重复此指令并仅添加使您进入适当屏幕的那部分。

使用 IDoc 开发业务对象

可以在 SAP 中将 ABAP 扩展模块的 WebSphere 业务对象定义为 IDoc。IDoc 是 SAP 的 EDI 解决方案（称为 ALE，应用程序链接启用）的一部分。它们的定义存储在 SAP 的 BOR（业务对象资源库）中，且可以在 SAP 系统中在全局范围访问它。IBM WebSphere Business Integration mySAP.com 适配器实现 ALE 的定义部分以解释 SAP 应用程序中的 WebSphere 业务对象并对它们进行语法分析，以准备好供 SAP 本机 API 使用。适配器提供了一个 IDoc 处理程序，该 IDoc 处理程序支持使用 IDoc 开发的业务对象。

IDoc 处理程序由两个功能模块组成。其它 ABAP 处理程序（如“动态检索”和“动态事务”）仅为单个功能模块。

Y_XR_DO_VERB_NEXTGEN 将业务对象数据传递至 IDoc 处理程序 Y_XR_IDOC_HANDLER。根据特定于应用程序的信息，Y_XR_IDOC_HANDLER 将业务对象数据重新格式化为由特定于应用程序的信息指定的 IDoc 的结构。在重新格式化之后，业务对象数据被传递至特定于对象的 IDoc 处理程序（基于业务对象的对象／查询描述组合），该过程处理与 SAP 本机 API 的集成。一旦特定于对象的 IDoc 处理程序完成对业务对象数据的处理，它以 IDoc 格式将业务对象数据返回至 Y_XR_IDOC_HANDLER。业务对象数据现在转换回其原始格式并返回至 Y_XR_DO_VERB_NEXTGEN。

图 75 举例说明了 IDoc 处理程序的基本体系结构。

IDoc 处理程序体系结构

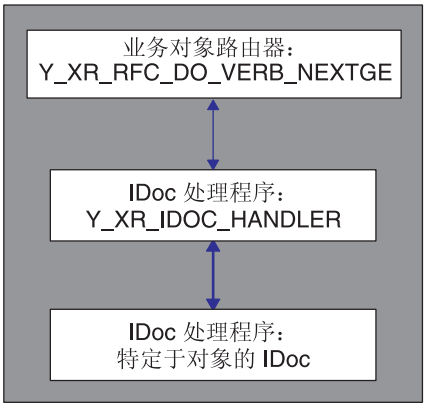


图 75. IDoc 处理程序体系结构

要使用适配器提供的 IDoc 处理程序，您必须在 SAP 应用程序中定义 IDoc。可以使用 SAP 交付的或客户构建的 IDoc。因为 IDoc 定义必须反映 SAP 的 WebSphere 业务对象的定义，所以适配器在 IBM WebSphere InterChange Server 连接器工具（事务 YXR1）中提供了一个工具，您可以使用该工具来基于 IDoc 生成 WebSphere 业务对象定义。

必须已在 SAP 应用程序中创建了 WebSphere 业务对象，然后才能生成业务对象定义。要基于 IDoc 创建业务对象定义:

1. 转至“IBM WebSphere InterChange Server 连接器工具”窗口（事务 YXR1）。
2. 从“定制”菜单，单击“维护对象”。
3. 创建新的对象名。

有关如何将特定于应用程序的信息用于查询描述功能的更多信息，请参阅第 199 页的『传递至 ABAP 处理程序的业务对象数据』。

在定义 IDoc 之后，为业务对象必须支持的每个查询描述创建一个功能模块。每个功能都应具有以下接口以确保 Y_XR_IDOC_HANDLER 可以调用它:

```
*"       IMPORTING
*"          VALUE(OBJECT_KEY_IN) LIKE  YXR_EVENT-OBJ_KEY OPTIONAL
*"          VALUE(INPUT_METHOD) LIKE  BDWFAP_PAR-INPUTMETHD
*"                              OPTIONAL
*"             VALUE(LOG_NUMBER) LIKE  YXR_LOG_H-LOG_NR OPTIONAL
*"       EXPORTING
*"             VALUE(OBJECT_KEY_OUT) LIKE  YXR_EVENT-OBJ_KEY
*"             VALUE(RETURN_CODE) LIKE  YXR_RFCRC-YXR_RFCRC
*"             VALUE(RETURN_TEXT) LIKE  YXR_EVENT-OBJ_KEY
*"       TABLES
*"              IDOC_DATA STRUCTURE  EDIDD
```

IDoc 处理程序和创建、更新和删除查询描述

支持创建、更新和删除操作的 IDoc 处理程序接收已格式化为 IDoc 的业务对象数据。这些操作的角色是将业务对象数据与 SAP 的“调用事务” API 集成在一起并生成对象键。仅会将对象键传递回 Y_XR_IDOC_HANDLER，而不会传递业务对象数据。Y_XR_IDOC_HANDLER 将业务对象数据存储在内存中，并将对象键插入父业务对象中标记为 isKey 的第一个属性中。然后，Y_XR_IDOC_HANDLER 将业务对象数据传递回连接器。

注: 当 InterChange Server（ICS）是集成代理程序时，因为映射基础结构需要保存 ObjectEventId 以进行动态交叉引用，所以维护业务对象数据非常重要。

下面的样本代码表示以下流:

1. 初始化全局数据。
2. 将 IDoc 解构到工作表中。
   - 因为并非所有对象都已发送到 SAP 应用程序，所以使用“/”（CxIgnore）来初始化目标结构。使用 YXRIFRM0 中的表单。
   - 使用 YXRIFRM0 中的表单来将数据从 IDoc 传送至内部表以在对象之间获取一致的行为。
3. 构建 BDC。使用 YXRIFRM0 中的表单来将数据从内部表传送至 IDoc 表以在对象之间获取一致的行为。
4. 创建“调用事务”。
5. 捕获对象键。

以下样本代码支持 SAP 销售报价创建:

```
*- Initialize working variables and internal tables
  PERFORM INITIALIZE_IN.

*- I01(MF): Begin IDoc interpretation
  PERFORM LOG_UPDATE(SAPLYXR1) USING C_INFORMATION_LOG TEXT-I01
                                     SPACE SPACE SPACE.

*- Interpret IDoc data structure
  IF NOT IDOC_DATA[] IS INITIAL.

*- Move IDoc to internal tables
    PERFORM INTERPRET_IDOC.

*- Check some of the input fields
```

```
    PERFORM CHECK_INPUT.

*- If key values were missing, exit function
    IF RETURN_CODE NE 0.
      EXIT.
    ENDIF.

*- E01(MF): No Idoc data lines sent for processing.
  ELSE.

    RETURN_CODE = 2.
    RETURN_TEXT = TEXT-E01.
    EXIT.

  ENDIF.

*- Build the BDC session for transaction VA21.
  PERFORM BUILD_BDC_VA21.

*- Call Transaction
  PERFORM LOG_UPDATE(SAPLYXR1) USING C_INFORMATION_LOG TEXT-I02
                                     'VA21' C_BLANK C_BLANK.

  CALL TRANSACTION 'VA21' USING BDCDATA
                           MODE INPUT_METHOD
                         UPDATE 'S'
                       MESSAGES INTO BDC_MESSAGES.

*- Capture return code and object key from transaction
  PERFORM PREPARE_RETURNED_MESSAGE.

ENDFUNCTION.
```

“创建”逻辑具有两个主要功能:

- 将 IDoc 数据转换为可管理的数据结构
- 执行“调用事务”

转换 IDOC 结构

“创建”逻辑的第一部分是执行转换任务，该任务将 IDoc 结构中的数据转换为工作数据结构。为此，您需要创建类似于下内容的代码:

```
loop at idoc_data.

    case idoc_data-segnam.
      when 'ZSQVBAK'.                  " Header Data
        move idoc_data-sdata to zsqvbak.

      when 'ZSQVBUK'.                  " Status Segment
        move idoc_data-sdata to zsqvbuk.

      when 'ZSQVBP0'.                  " Partner Header Level
        move idoc_data-sdata to zsqvbp0.

      when 'ZSQVBAP'.                  " Item Detail
        move idoc_data-sdata to zsqvbap.

      when 'ZSQVBA2'.                  " Item Detail Part 2
        move idoc_data-sdata to zsqvba2.

      when 'ZSQVBUP'.                  " Item Status
        move idoc_data-sdata to zsqvbup.

      when 'ZSQVBKD'.                  " Commerical data
        move idoc_data-sdata to zsqvbkd.
```

```
      when 'ZSQKONV'.                  " Condition
        move idoc_data-sdata to zsqkonv.

      when 'ZSQVBPA'.                  " Partner Item Level
        move idoc_data-sdata to zsqvbpa.

    endcase.

  endloop.
```

IDoc 处理程序和检索查询描述

支持“检索”查询描述的特定于对象的 IDoc 处理程序不接收来自初始 IDoc 处理程序的业务对象数据。Y_XR_IDOC_HANDLER 使用参数 OBJECT_KEY_IN 来传递业务对象数据中标记为 isKey 的第一个属性的值。IDoc 处理程序负责使用此键来使用 ABAP SQL 检索与对象的此实例相关的所有信息，并以适当的 IDoc 结构来格式化这些数据。

以下代码支持“销售报价”示例。“销售报价”对象必须从表 VBAK、VBUK、VBPO、VBAP、VBUP、VBKD、KNOV 和 VBPA 中检索数据。这些表遵循 IDoc 类型 ZSLSQUOT 的层次结构和基数。该代码执行下列操作:

1. 初始化全局数据。
2. 从 SAP 应用程序数据库返回业务对象数据。
3. 根据返回的数据构建 IDoc 并将该数据返回至 Y_XR_IDOC_HANDLER。

IDoc 类型 ZSLSQUOT 的代码如下:

```
*- Clear the interface structures.
  clear: g_text, object_key_out, return_code, return_text, idoc_data.
  refresh: idoc_data.

*  If no key value is specified, log it as an error and exit.
  if object_key_in is initial or
     object_key_in = c_cxignore_const.
    perform log_update(saplyxr1) using c_error_log text-e02
                                       space space space.
    return_code = 1.
    return_text = text-e02.
    exit.
  endif.

  perform initialize_global_structures.

  perform fill_internal_tables.
  if not return_code is initial.
    exit.
  endif.

* Build Idoc segments from internal tables
  perform fill_idoc_inttab.

  return_code = 0.
  return_text = text-s01.

  perform log_update(saplyxr1) using c_information_log text-s01
                                     space space space.
endfunction.
```

两个最重要的参数是表示入站键的 OBJECT_KEY_IN 和表示出站数据的 IDOC_DATA。注意，OBJECT_KEY_IN 甚至可能是一个多键， 这取决于您定义的约定。

VBAK 表驱动子表的选择标准，因此将把每个表都装入工作表中。您可以使用 VBAK 表来检索具有其它键的子表。因此，对于“销售报价”示例，代码如下所示:

```
form fill_internal_tables.

  * Get information from VBAK, VBUK, VBAP, VBKD, KONV, VBPA

  select single * from vbak
         where vbeln = object_key_in.

  if sy-subrc <> 0.
    perform log_update(saplyxr1) using c_error_log text-e01
                                       object_key_out c_blank c_blank.
    return_code = '1'.
    g_text = text-e01.
    replace '&' with order_number into g_text.
    return_text = g_text.

    exit.
  endif.

  select single * from vbuk
         where vbeln = vbak-vbeln.

  select * from vbap into table t_vbap
         where vbeln = vbak-vbeln.

  * Continue for other tables
```

以下代码用来将请求的数据从应用程序数据库复制到内部表和工作变量。然后，该代码创建直接对应于 WebSphere 业务对象定义的段并将这些段放置在 SAP 段结构中。

在某些情况下，为了使 IDoc 类型和工作结构之间的字段紧密匹配，您可以执行对应于 ABAP 移动的命令。在其它情况下，执行从工作表至 IDoc 类型表的手工移动更可取，因为只需要移动相对较少的字段而不是移动结构中的所有字段。不过，它只用来将数据从工作数据结构转换为 IDoc 结构，然后转换为平面数据字段。

代码为:

```
form fill_idoc_inttab.

perform fill_zsqvbak." Fill the Sales Quote Header
perform fill_zsqvbuk." Fill the Sales Quote Status
perform fill_zsqvbap." Fill Sales Quote Lines

endform." FILL_IDOC_INTTAB

*-- fill the Sales Quote Header
form fill_zsqvbak.

  clear idoc_data.
  clear zsqvbak.
  idoc_data-segnam = 'ZSQVBAK'.

  move-corresponding vbak to zsqvbak.
  move zsqvbak to idoc_data-sdata.
  append idoc_data.

endform." FILL_ZSQVBAK

*-- fill the Sales Quote Header Status
form fill_zsqvbuk.

  clear idoc_data.
  clear zsqvbuk.
```

```
  idoc_data-segnam = 'ZSQVBUK'.

  move-corresponding vbuk to zsqvbuk.
  move zsqvbuk to idoc_data-sdata.
  append idoc_data.

endform." FILL_ZSQVBAK

*-- fill the Sales Quote Line and the Line Child segments
form fill_zsqvbap.

  loop at t_vbap.
    clear idoc_data.
    clear zsqvbap.
    idoc_data-segnam = 'ZSQVBAP'.

    move-corresponding t_vbap to zsqvbap.
    move zsqvbap to idoc_data-sdata.
    append idoc_data.

    perform fill_zsqvba2.
    perform fill_zsqvbup.
    perform fill_zsqvbkd.
    perform fill_zsqkonv.
    perform fill_zsqvbpa.


  endloop.

endform.

*-- fill second part of vbap
form fill_zsqvba2.
" etc.
```

调用 ABAP 扩展模块和 ABAP 处理程序

连接器使用业务对象中特定于应用程序的查询描述信息值，来调用 ABAP 扩展模块中的适当 ABAP 处理程序。要调用 ABAP 扩展模块中的适当 ABAP 处理程序，您必须指定 ABAP 扩展模块的类名和业务对象使用的 ABAP 处理程序功能模块。例如，“动态检索” ABAP 处理程序的特定于应用程序的查询描述信息为:

```
AppSpecificInfo = sap.sapextensionmodule.VSapBOHandler,:Y_XR_DYNAMIC_RETRIEVE
```

注: 您必须在连接器模块（类名）和 ABAP 处理程序之间使用逗号定界符。

有关 ABAP 扩展模块的业务对象处理的更多信息，请参阅第 181 页的『业务对象处理』。

第 23 章 为 ABAP 扩展模块开发事件检测

事件检测是 ABAP 扩展模块的 ABAP 组件中事件触发进程的一部分。每个事件检测机制都必须调用一个事件触发器，该触发器获得检测到的事件并将它添加至事件表。关于触发事件的更多信息，请参阅第 186 页的『事件触发』。

注: 对连接器的 ABAP 组件的所有引用都使用 SAP R/3 V3.x 命名约定。

本章包括以下主题:

设计事件检测机制

您可以使用许多不同的机制来在 SAP 应用程序中检测事件。事件检测机制应该能够调用功能模块。IBM WebSphere Business Integration mySAP.com 适配器（SAP R/3 V3.x）已实现的三个事件检测机制如下:

- 代码增强 – 它是为业务流程实现的（通常是单个 SAP 事务），方法是在 SAP 事务内的适当位置插入事件检测代码
- 批处理程序 – 涉及开发一个包含生成事件的条件的 ABAP 程序
- 业务工作流 – 使用 SAP 本身的面向对象的事件检测能力

因为某些机制可能不可用于特定业务流程，所以您确定要为开发的每个业务对象实现的适当事件检测机制很重要。对于您想要实现事件检测的每个事务，您必须对特定业务流程的技术和功能方面有所了解。

当确定要为业务流程实现哪种事件检测机制时，查看以下实现注意事项。

可用性

哪些事件检测机制可用于此业务流程？这应该是您要考虑的一个首要问题。代码增强和批处理程序具有高可用性，而业务工作流不具有高可用性。

实时集成

需要同步检测事件吗？您需要同时检测大量事件吗？除批处理程序外的所有机制都适合于实时集成。

可靠性

当生成事件时，是否检测到此业务流程的所有数据更改？代码增强和批处理程序对捕获对象的所有事件提供最佳控制。业务工作流提供有限的可靠性。例如，在供应商事务更新期间，业务工作流不会检测地址更改。

灵活性

在触发事件之前，需要评估某个条件吗？需要在事务的某一时刻检测事件吗？因为您可以在提交事件数据之前在特定位置插入代码，所以代码增强最灵活。批处理程序具有适度的灵活性，而业务工作流在其实现中具有非常小的灵活性。

升级相关性

升级 SAP 应用程序会更改此业务流程检测事件的方式

吗？通常，这是未知的，但业务工作流受应用程序更改的影响最大，因为它在 SAP 的控制之下。

困难

问题是时间还是难度？每种机制都有各自不同的实现难度。通常，批处理程序最容易实现。代码增强和业务工作流较难实现。

此时，您应该对需要考虑的事件检测机制有所了解。在确定哪种机制可以用于您需要支持的每个业务流程时，使用表 46 作为一般准则。

表 46. 事件检测机制判定表

	代码增强	批处理程序	业务工作流
可用性	高	高	低
实时集成	是	否	是
可靠性	高	高	低
灵活性	高	中	低
升级相关性	低	低	中
困难	中	低	中

要注意的最后一个注意事项是站点的开发方法。也许仅使用业务工作流的事件检测是首选方法，而代码增强根本不能使用。

建议将代码增强用于事件检测，因为它可靠、灵活性大、同步且具有高可用性。与之比较而言，业务工作流机制通常并非可用于所有业务流程。当不希望进行实时集成时，通常使用批处理程序。

对于在业务流程中检测事件，每种事件检测机制都具有优点和缺点。以下各节给出了关于每种事件检测机制的更多详细信息，包括每种机制的主要优点和缺点。

代码增强

将在 SAP 事务的代码中的特定位置实现代码增强。通过利用用户出口，您可以将事件检测代码插入事务中最合理的位置。事件检测代码允许对条件进行评估来确定是否生成事件。

此机制的一般策略是在事务的数据即将提交至数据库时插入事件检测代码。

优点

- 可以访问事件检测进程的 SAP 事务信息
- 允许在事务的适当位置插入事件检测代码
- 提供同步事件检测
- 限制了对 SAP 功能的依赖性，因此维护和增强更容易

缺点

- 用户出口可能不会总是在事务的适当位置。
- 可能需要 SAP 修改功能部件。

批处理程序

当需要触发大量相同类型的事件（如客户订单）或业务流程需要大量处理时间时，批处理程序很有用。此机制不要求对 SAP 交付的代码进行任何修改；但是，您需要使用（编写）评估检测事件的条件的 ABAP 程序。

优点

- 可以对大多数业务流程实现
- 准确地检测事件
- 容易实现
- 如果运行时资源存在问题的话，可以安排在特定的时间运行

缺点

- 它不提供同步事件检测。
- 未提供 SAP 事务信息。
- 状态（创建、更新或删除）或状态更改检测不到或可能不容易检测。
- 如果创建后台作业来使批处理程序自动执行，则需要维护和监视额外的任务。

业务工作流

业务工作流是 SAP 应用程序中跨应用程序的工具，它使您能够将应用程序之间的业务任务整合在一起。此工具补充 SAP 应用程序的现有业务功能。可以使用业务工作流来调整 SAP 的标准功能，以满足所需业务功能的特定要求。业务工作流使用业务对象资源库（BOR），该资源库存储应用程序中每个 SAP 对象的定义。

优点

- 提供同步事件检测
- 利用 SAP 的面向对象的业务对象能力来将事件检测链接至 ABAP 功能模块
- 容易实现

缺点

- 对于每个业务流程，SAP 对象在 SAP BOR 中都不存在。
- 对于 SAP 对象，SAP 事件（如已创建或已删除）可能不存在。
- 它可能并不会检测业务流程中的所有更改。
- 它不会总是在适当的时间提供检测事件的灵活性。
- 它依赖于 SAP 提供的功能，这在 SAP 的各个版本之间可能有所不同。

实现事件检测机制

一旦确定了要支持的业务流程（例如，销售报价或销售订单）并确定了首选事件检测机制，则为您的业务流程实现该机制。

注: 当实现事件检测机制时，最好是在一个机制中支持业务流程的所有功能。这限制了对 SAP 应用程序的影响，并使事件检测更容易管理。

下列各节描述由 IBM WebSphere Business Integration mySAP.com 适配器实现的四种事件检测机制的实现过程。在适用的任何时候，均将提供示例以及样本代码。

代码增强

代码增强要求将一部分 ABAP 代码包括在定制功能模块中。事件检测代码将作为功能模块编写，以确保处理与事务保持分离。从事务中使用的任何表或变量都需要按值而不是按引用传递至功能模块。

要在检索事件时使锁定业务对象的影响最小，功能模块通常以更新任务方式执行。要避免不一致，当在处于更新任务方式下的进程内已调用功能模块时，不要使用更新任务。

要使事务中的影响最小，将功能模块放置在另一个包含程序内。使用包含程序将允许您对定制代码而不是 SAP 代码进行更改。

事件检测代码包含标识事件对象的逻辑。例如，销售订单事务处理许多类型的订单，但只有一种订单类型是必需的。此逻辑在事件检测代码中。放置此事件检测代码的一般策略是恰好在将数据提交至数据库之前插入它。包含事件检测代码的功能模块通常将作为业务对象功能组的一部分创建。

要实现代码增强以进行事件检测:

- 确定要支持哪些查询描述: 创建、更新或删除。这有助于定义要调查哪些事务。
- 确定事务的业务对象键。此键必须是唯一的，以允许连接器从数据库检索业务对象。可能需要组合键。
- 检查事务中 SAP 提供的用户出口是否具有检测事件所需要的所有信息。例如，因为此前已从数据库除去了该业务对象，所以用户出口可能无法实现“删除”查询描述。
- 如果不能使用某个用户出口，则确定事件检测代码的适当位置，然后使用 SAP 修改来添加事件检测代码。选择一个可以访问业务对象键和其它用来作出决定的变量的位置。

  通过在业务流程的事务所执行的代码中查找 “commit work statement” 来调查业务流程。您可以使用 ABAP 调试器在此时调查不同属性的值。
- 确定检测事件的条件。
- 创建包含事件检测代码的功能模块。
- 创建包含程序并将它添加至事务的代码。测试旨在检测事件的所有方案。

以下步骤描述使用“代码增强”事件检测机制来创建示例 SAP 销售报价的过程。步骤后面的代码是此过程的结果。

1. 在调查 SAP 销售报价事务时，找到事务 VA21 以支持期望的销售报价创建业务流程。
2. 销售报价号被确定为唯一键。销售报价号存储在表／字段 VBAK-VBELN 中。
3. 事务 VA21 在事务流中使用用户出口作为文档保存过程的一部分（Form Userexit_save_document）。在事务的此时刻，当执行用户出口时可获得报价号。
4. 用户出口属于其它业务流程，因此需要进行额外的编码才能将销售报价与其它类别的文档区分开来。VBAK-VBTYP 可用来确定文档类别。销售报价将以文档类别 B 保存在 SAP 数据库中。
5. 将包含语句添加至指向包含程序的用户出口。
6. 此时，需要创建包含程序和功能模块。

新的功能模块包含以下代码:

```
If VBAK-VBTYP = 'B'.
     C_OBJ_ORDER = 'SAP_SalesQuote'.
        TMP_OBJKEY = XVBAK-VBELN.
 TMP_EVENT = 'Create'.
 TMP_OBJTYPE = Space.

     CALL FUNCTION 'Y_XR_ADD_TO_QUEUE'
             EXPORTING
OBJTYPE = TMP_OBJTYPE
OBJNAME = C_OBJ_ORDER
OBJKEY = TMP_OBJKEY
EVENT = TMP_EVENT
GENERIC_RECTYPE = ''
IMPORTING
RECTYPE = TMP_RECTYPE
             TABLES
                  EVENT_CONTAINER = TMP_EVENT_CONTAINER
             EXCEPTIONS
                  OTHERS          = 1.

Endif.
```

批处理程序

要实现批处理程序作为事件检测机制，您必须编写评估数据库信息的 ABAP 程序。如果当 ABAP 程序执行时达到该程序中的条件，则将触发一个事件。

要实现批处理程序以进行事件检测:

- 确定要支持哪个查询描述: 创建、更新或删除。
- 确定事务的业务对象键。业务对象键必须是唯一的，以便可以从数据库检索业务对象。可能需要组合键。例如，为不同工厂的物料库存级别实现批处理程序需要键 `Material_key + Plant_key`。
- 确定检测事件的条件。您应了解与业务对象相关的数据库表。
- 创建一个包含生成事件的条件的 ABAP 程序。
- 确定是否需要后台作业来使批处理程序自动执行。如果批处理程序对系统资源有影响（这使得有必要在非高峰期间运行批处理程序），则后台作业是很有用的。

以下步骤描述创建批处理程序的过程，该批处理程序检测当天创建的所有销售报价的事件。步骤后面的代码是此过程的结果。

1. “创建”被确定为受支持的查询描述。
2. 报价号被确定为用来检索事件的唯一键。
3. 创建日期（VBAK-ERDAT）和文档类别（VBAK-VBTYP）需要检查。

以下样本代码支持 “SAP 销售报价”作为批处理程序:

```
REPORT ZSALESORDERBATCH.

tables: vbak.

parameter: d_date like sy-datum default sy-datum.

data: tmp_key like YXR_EVENT-OBJ_KEY,
      tmp_event_container like swcont occurs 0.

" retrieve all sales quotes for today's date
" sales quotes have vbtyp = B
```

```
select * from vbak where erdat = d_date
                     and vbtyp = 'B'.

tmp_key = vbak-vbeln.
TMP_OBJTYPE = space.

  CALL FUNCTION 'Y_XR_ADD_TO_QUEUE'
       EXPORTING
OBJTYPE = TMP_OBJTYPE
OBJNAME = 'SAP_SalesQuote'
OBJKEY = tmp_key
EVENT = 'Create'
GENERIC_RECTYPE = ''
IMPORTING
RECTYPE = r_rectype
TABLES
            EVENT_CONTAINER = tmp_event_container.

  write: / vbak-vbeln.
endselect.
```

业务工作流

业务工作流是一组或一系列在逻辑上相关的业务操作。工作流内的处理逻辑检测事件。业务工作流事件检测机制依靠 SAP 业务对象资源库（BOR），该资源库包含对象及其相关属性、方法和事件的目录。

要实现业务工作流以进行事件检测:

- 确定哪个业务对象表示您需要的功能。检查事件是触发、启动还是结束工作流。可以使用业务对象构建器（事务 SWO1）来搜索适当的业务对象。
- 创建此业务对象的子类型。子类型继承超类型的属性且可以定制后使用。
- 通过定制子类型来激活业务对象的事件（如 CREATED、CHANGED 和 DELETED）。

SAP 销售报价的以下示例可以用来实现使用业务工作流的事件触发器:

1. 搜索 BOR 以获取适当的销售报价业务对象。可以使用简短描述字段和字符串“*quot*”来执行搜索。BUS2031（客户报价）是返回的一个业务对象。
2. 进一步研究 BUS2031 后，确定键字段为 CustomerQuotation.SalesDocument（VBAK-VBELN）。
3. 将使用以下条目创建 BUS2031 的子类型:

   对象类型 - ZMYQUOTE

   事件 - SAP_SalesQuote

   名称 - SAP 销售报价

   描述 - SAP 销售报价子类型的示例

   程序 - ZMYSALESQUOTE

   应用程序 - V
4. 将通过把条目添加至“事件链接”表（事务 SWE3）来激活事件检测机制。将使用以下条目来激活创建事件:

   对象类型 - ZMYQUOTE

事件 – SAP_SalesQuote

接收器 FM – Y_XR_ADD_TO_QUEUE_DUMMY

接收器类型 FM – Y_XR_ADD_TO_QUEUE_WF

注: 接收器和接收器类型功能模块（FM）都指向 Y_XR_ADD_TO_QUEUE。因为 SAP 应用程序有时要求两个字段都要填充，所以仅使用了 DUMMY 功能模块。WF 功能模块将 SAP 标准接口转换为由 Y_XR_ADD_TO_QUEUE 使用的那个接口。

业务工作流事件检测机制已创建且是活动的。已设置它来检测已创建的所有 SAP 客户报价。

第 24 章 测试 ABAP 扩展模块的业务对象

一旦已开发特定于应用程序的业务对象和支持 ABAP 处理程序，则您必须进行单元测试以确保它们支持所需要的功能。IBM WebSphere Business Integration mySAP.com 适配器（SAP R/3 V3.x）提供了一些便于执行此测试的单元测试工具。这些工具独立于您的 IBM WebSphere Business Integration 系统运行，这意味着您不必运行此系统就能测试业务对象。

注: 这些工具不替换 IBM WebSphere Business Integration 系统中完全端到端测试。只打算将它们用于对各个业务对象和 ABAP 处理程序进行单元测试。

本章包括以下主题:

准备测试

所有业务对象处理起源于连接器的 Java 组件。这适用于所有对象和所有可能的查询描述。为了进行单元测试，适配器包括一个 ABAP 程序，该程序模拟业务对象请求中发送的连接器的操作。

特别是，该程序通过调用 ABAP 功能模块 Y_XR_RFC_DO_VERB_NEXTGEN 来模拟连接器的 Java 组件中的 doVerbFor() 处理。与 doVerbFor() 一样，测试程序需要一个业务对象作为输入以传递到 ABAP 功能模块 n。ABAP 测试程序使用文本文件作为其输入。

所有输入测试文件都具有相同的 ASCII 文本格式。根据此文件格式，测试程序重构数据以建立传递到 Y_XR_RFC_DO_VERB_NEXTGEN 的相似业务对象。以下规则适用于业务对象输入文件:

- 业务对象在文件中必须只具有一个父业务对象。
- 子业务对象首先按深度排序，然后按宽度排序。
- 属性和对象必须严格按它们在业务对象资源库定义中出现的顺序排序。
- 对于每个属性，表 47 中描述的信息必须按描述的格式和显示的顺序提供（将忽略“=”后面的前导空格）:

表 47. 属性特性和值

属性特性	描述或可能的值
Name	属性的名称
Value	属性的值或 CxIgnore = 'CxIgnore' 或 CxBlank = ' '
IsKey	指定属性是否是键的值: 0 = 否，1 = 是
Peers	表示一个整数的 NumberOfPeers 值，该值表示同一级别的子业务对象的总数，例如，如果“项”业务对象包含两个行项，则每个行项将具有值“2”。
AppInfo	特定于每个业务对象的特定于应用程序的信息

测试程序提供了“快速检索”功能来帮助生成测试输入文件。有关更多信息，请参阅第 230 页的『测试 ABAP 处理程序』。

单元测试问题

单元工具测试处理连接器的业务对象处理的所有 SAP 开发工作。并且，单元测试工具使您能够测试您的工作与连接器的 ABAP 组件的交互作用。这些测试工具仅允许您作为联机用户（实时）测试您的开发工作。

注: 了解作为联机用户测试连接器和作为后台用户测试连接器之间的差别，这很重要。

作为联机用户测试连接器或作为后台用户测试连接器之间的差别描述如下:

- 内存 – 当测试业务对象时，连接器必须登录 SAP 应用程序。

  连接器作为后台用户运行，所以它在单一内存空间中进行处理，在停止然后重新启动连接器之前，永远不会隐式刷新该空间（所以在完成处理后，清除内存对于业务对象开发非常重要）。因为您是联机用户，所以通常将在您执行的每个事务之后刷新内存。

  有关更多信息，请参阅第 205 页的第 22 章，『为 ABAP 扩展模块开发业务对象』。因而可能发生的任何问题（例如，从不初始化返回码）不使用此测试工具检测；仅对连接器的测试才会暴露这些问题。
- 屏幕流行为 – 仅当使用“调用事务”API 时才与屏幕流行为相关。用户与之交互的精确屏幕和屏幕序列通常由事务的代码在运行时确定。例如，如果用户选择扩充原料主记录以通过选择“销售视图”复选框来包括销售视图，则 SAP 会通过显示其它输入字段来询问用户以获取特定的销售组织信息。使用此方式，事务源代码在运行时根据用户输入的数据来确定特定的屏幕及其需求。虽然测试工具的确可以应付此类型的测试方案，但存在一个测试工具不能应付的相关方案。

  SAP 的事务代码可能对联机用户和后台用户显示不同的屏幕（通常是为了可用性和性能）。测试工具只能作为联机用户运行。连接器将仅作为后台用户运行。尽管存在此差别，单元测试还是应该在大多数测试情况下进行。

测试 ABAP 处理程序

要进行测试，您必须首先生成业务对象输入文件。此时，您可能需要修改该文件以包含属性值和特定于应用程序的适当信息。最后一个步骤是执行测试程序，并指向您的测试文件作为输入。

创建测试文件

要创建测试文件:

1. 转至“WebSphere InterChange Server 连接器工具”（事务 YXR1）。
2. 选择“测试程序”。
3. 执行“快速检索”以获取需要的业务对象:
   - 输入输出文件的名称。

- 选择“IDoc 检索”或“动态检索”。
- 输入业务对象的名称和对象键。
- 如果您选择了“IDoc 检索”，则输入 IDoc 类型，并在“方法 1”字段中输入 Y_XR_IDOC_HANDLER，然后在“方法 2”字段中输入“检索”功能模块的名称。
- 如果选择了“动态检索”，则在“方法 1”字段中输入 Y_XR_DYNAMIC_RETRIEVE。

4. 单击“执行”以将测试业务对象保存在您指定的输出文件中。
5. 在任何文本编辑器中编辑测试文件。您必须执行以下操作:
   - 修改特定于应用程序的查询描述信息以指向 ABAP 处理程序，有关语法，请参阅第 199 页的『传递至 ABAP 处理程序的业务对象数据』。例如，:function1:function2。
   - 验证父代上的适当属性是否标记为 isKey。
   - 按需要添加属性的输入值。

使用测试文件

要使用测试文件:

1. 转至“WebSphere InterChange Server 连接器工具”（事务 YXR1）。
2. 选择“测试程序”。
3. 在“输入文件”字段中输入您的输入文件的位置和文件名。
4. （可选的）为输出数据输入文件名和位置（可以与输入相同，但它将覆盖输入）。
5. 单击“执行”按钮。

当完成时，程序将显示处理期间出现的最后一条消息。另外，还将在屏幕上显示处理后的数据以进行验证。此信息与在步骤 4 的输出文件中生成的信息相同。

并且，您可以查看 IBM WebSphere InterChange Server ABAP 日志以获取其它详细信息。

第 25 章 管理 ABAP 扩展模块

IBM CrossWorlds 连接器工具（事务 YXR1）使您能够维护 IBM WebSphere Business Integration mySAP.com 适配器（SAP R/3 V3.x）事件处理。您还可以使用此工具来维护与 SAP 应用程序的连接。您可以查看连接器日志文件和 “SAP 网关服务” 连接。并且，您可以从连接器日志重新处理已归档的对象、查看正在等待处理的事件以及从归档表重新提交和删除事件。

本章包括以下主题:

管理连接器日志文件

SAP 应用程序中的连接器日志按年月日的相反顺序显示与连接器相关的所有事件和错误，如创建或更新操作或到达事件队列中的事件。日志文件列示每个日志条目的日期、时间和事件。日志文件将大大有利于开始对问题进行故障诊断。

设置日志选项

您可以将全局设置和用户设置设置为您希望写入到连接器日志文件的详细信息级别以及您希望显示的条目数和数据类型。要设置连接器的日志记录级别:

1. 转至 “IBM WebSphere InterChange Server 连接器工具” 窗口（事务 YXR1）
2. 从 “定制” 菜单，单击 “日志”，然后单击 “日志记录选项”。
3. 在 “日志记录级别” 下，从级别 0 至 3 中进行选择。日志记录的四个级别如下:
   - 0 – 关闭
   - 1 – 仅将警告和错误写入日志
   - 2 – 将每个事件以及很少的信息写入日志
   - 3 – 将每个事件详细地写入日志，包括每个业务对象的每个属性

   注: 建议不要使用日志记录级别 0。建议将日志记录级别 1 用于生产系统。建议将日志记录级别 3 用于开发或调试系统。

显示日志

要查看最近处理的对象和与它们相关的详细信息，显示连接器日志。要在 SAP 应用程序中显示连接器日志:

1. 转至 “IBM WebSphere InterChange Server 连接器工具” 窗口（事务 YXR1）。
2. 单击 “显示日志” 按钮，或从 “工具” 菜单，单击 “显示日志” （F8）。
   - 绿色 – 指示成功的事件
   - 黄色 – 指示警告消息

- 红色 – 指示错误
- 白色 – 指示已归档的对象
- 品红色 – 提供关于事件开始和结束的信息。

3. 单击任何箭头以链接至该业务对象的 SAP 的显示事务。

您可以更改显示的关于每个事件的详细信息量。要更改显示级别，根据您想要的详细信息级别单击“更多详细信息”或“较少详细信息”按钮。

过滤日志详细信息

您可以更改显示的关于每个事件的详细信息量。如果显示的数据量多于您当前需要的量，则缩小显示的信息。例如，您可以按用户、名称、日期或事件号来查看业务对象。

1. 单击“过滤数据”按钮。
2. 填充适当的字段以过滤日志文件。
3. 单击“过滤器”。

在“日志记录选项”屏幕中，您可以设置用户设置以便同时显示许多日志条目，并可以设置缺省日志记录显示级别。

设置事件日志的截断

SAP 保留连接器活动的事件日志。随着时间的过去，此日志可以占用大量磁盘空间。要节省磁盘空间，您可以将此日志设置为自动截断。当设置自动截断时，缺省情况下，SAP 将截断的条目打印到设置该作业的用户的缺省打印机。因此，您还可能要控制打印选项。

要设置截断选项:

1. 转至“IBM WebSphere InterChange Server 连接器工具”窗口（事务 YXR1）。
2. 单击“显示日志”按钮，或从“工具”菜单，单击“显示日志”（F8）。
3. 单击“日志选项”按钮，或从“转至”菜单，单击“日志选项”（F7）。
4. 在“全局设置”之下（对于所有用户），输入您想要在每次截断之后保留在日志中的日志条目数。例如，如果您指定 1000，则每次截断日志时，将截断日志中除最新的 1000 个以外的所有条目。
5. 要立即截断日志，单击“截断日志”按钮。

重要提示: 通过安排报告 YXRLOGT 来设置事件日志的自动截断。建议您定期运行此报告。

要安排 YXRLOGT 报告:

1. 从“系统”菜单，单击“服务”，单击“作业”，然后单击“定义作业”（事务 SM36）。
2. 为作业命名，并指定类（优先级）。仅当有多个应用程序服务器访问单个数据库服务器时，才需要填充目标主机。
3. 通过单击“开始日期”按钮来指定作业的开始日期和频率。
4. 指定您想要运行作业的频率。选择“定期作业”复选框，单击“周期值”按钮，然后通过单击“周期值”按钮来选择适当的按钮。

5. 单击“保存”按钮两次以返回到第一个屏幕。通过单击“步骤”按钮来定义要运行哪个程序。安排 ABAP 程序 YXRLOGT。不需要变体。
6. 缺省情况下，SAP 将日志中截断的条目打印到设置该作业的用户的缺省打印机。如果日志记录已打开且级别较高并且处理了许多事件，则此打印输出可能很长（几百页）。通过单击“打印规范说明”按钮来控制将在何处打印条目（或是否打印它们）。
7. 要取消激活已截断日志的自动打印，取消选择“立即打印”复选框。在这种方式下，打印输出将被假脱机并保留 8 天（缺省值），可以在需要时引用它，而不会浪费纸张。现在单击“保存”按钮两次。
8. 单击绿色的返回箭头。
9. 单击“保存”按钮，然后单击绿色的返回箭头。现在已安排您的作业。

要查看截断作业:

1. 从“系统”菜单，单击“服务”，单击“作业”，然后单击“作业概述”（事务 SM37）。
2. 输入作业名，然后按 Enter 键。如果作业处于“已释放”状态，则它是正常的。

注: 可以通过单击“连接器日志”按钮来访问连接器日志文件。

监视 SAP 网关服务连接

您可以监视连接器和 SAP 应用程序之间的“SAP 网关服务”连接。每个条目都显示诸如连接器主机名、用户名和连接器状态等信息。

要监视“SAP 网关服务”连接:

1. 转至“IBM WebSphere InterChange Server 连接器工具”窗口（事务 YXR1）。
2. 从“工具”菜单，单击“监视网关”。
3. 单击某个服务器名称以查看更多详细信息。

关闭连接器

建议您使用以下各项关闭连接器:

- IBM WebSphere InterChange Server 系统管理器（当 InterChange Server 是集成代理程序时）
- `mqsiremotestopadapter` 命令（当消息代理是集成代理程序时）。

重要提示: 不要使用“网关监视器”窗口。如果使用“网关监视器”窗口，则您的连接器可能不会正确关闭。

维护事件队列

您可以检查当前出局事件队列以获取连接器尚未处理的事件。

1. 转至“IBM WebSphere InterChange Server 连接器工具”窗口（事务 YXR1）。
2. 从“工具”菜单，单击“出局队列”，然后单击“显示队列”。
3. 单击“执行”或按 F8 键显示事件队列。

要限制显示的事件条目数，填充“当前事件选择”部分中的适用字段。例如，要限制特定业务对象的显示条目，在“对象名”字段中输入业务对象名。如果您不知道业务对象名的精确语法，则单击“对象名”字段，单击箭头按钮（F4），然后选择适当的业务对象名。

要查看关于事件的更多信息，双击事件字段。在正常情况下，将每隔几秒选取一些事件。如果显示了某个事件，则连接器尚未处理它。这可能指示连接器未在运行。

以下是事件队列的可能事件状态值的列表:

P - 预排队	当触发事件时，因为尚未确定是否锁定了业务对象，所以状态最初设置为预排队（P）。
L - 已锁定	当用户在 SAP 中创建或更新业务对象时，将锁定该业务对象。一旦已将业务对象提交至数据库，则 SAP 会除去该锁定。如果当业务对象已锁定时触发事件，则该事件保留在状态为已锁定（L）的事件队列中，直到除去锁定为止。
R - 已检索	当连接器检索事件时，则状态更改为已检索（R）。这是临时状态。一旦完成事件处理，则连接器更新该状态并将事件移至归档表。如果状态保持为“已检索”一段较长的时间，则表示在该事件的处理期间连接器和 SAP R/3 之间的连接可能已断开。
Q - 已排队	当业务对象不再处于锁定状态时，则状态更改为已排队（Q），并且事件随时可由连接器选取。事件保持为此状态，直到接收到检索确认为止。

维护归档表

通过使用 IBM WebSphere InterChange Server 连接器工具（事务 YXR1），您可以显示归档表并确定已归档事件的状态。在该表中，您可以标识当集成代理程序预订时需要重新提交以进行轮询的事件。

要显示归档表:

1. 转至“IBM WebSphere InterChange Server 连接器工具”窗口（事务 YXR1）。
2. 从“工具”菜单，单击“出局队列”，然后单击“显示归档”。
3. 单击“执行”或按 F8 键显示归档队列。

要限制显示的归档条目数，填充“已归档事件选择”部分中的适用字段。例如，要限制特定业务对象的显示条目，在“对象名”字段中输入业务对象名。如果您不知道业务对象的精确语法，则单击“对象名”字段，单击箭头按钮，然后选择适当的业务对象名。

要查看关于事件的更多信息，双击事件字段。

以下是归档表的可能事件状态值的列表:

0 - 成功	连接器已成功处理事件并将业务对象发送至集成代理程序。
1 - 在 SAP 中出错	连接器在 SAP 中检索此事件的业务对象时遇到错误。
2 - 未预订	集成代理程序未预订此事件的业务对象和查询描述的组合。

3 - 在 Java 中出错	在以下其中一项操作中，连接器遇到错误: • 从 SAP 接收业务对象 • 将 SAP 业务对象转换为 SAP 的 WebSphere 业务对象 • 将业务对象插入消息队列
4 - 最大重新排队	事件重新排队的次数超过重新排队常数 c_maximum_requeue 指定的最大次数（通常是 100）。如果事件的业务对象已锁定，则会将事件重新排队。
5 - 多个事件	某些业务对象在事件表中具有在检索时导致创建多个事件的单一事件。原始单一事件不创建业务对象，因此将使用此事件状态进行归档。
6 - 事件已删除	用户从事件表中手工删除了该事件。

重新提交归档表中的事件

您可以将归档表中的事件重新提交至事件队列以进行重新处理。根据您想要如何处理归档表中的事件，您可以选择重新提交单个事件或多个事件。记住，重新提交事件只是将事件从归档表移至事件表，因此，这些事件不会经历事件分布、事件限制或事件优先级。从“已归档事件”窗口执行以下步骤:

1. 转至“IBM WebSphere InterChange Server 连接器工具”窗口（事务 YXR1）。
2. 从“工具”菜单，单击“出局队列”，然后单击“重新提交事件”。
3. 选择要重新提交的一个或多个事件。
4. 单击“执行”按钮，或从“程序”菜单，单击“执行”（F8）。

将显示一条状态消息。您可以显示连接器日志以查看事件及其新状态。

从归档表删除事件

您可以手工删除归档事件或安排它们自动删除。

要手工删除归档事件:

1. 转至“IBM WebSphere InterChange Server 连接器工具”窗口（事务 YXR1）。
2. 从“工具”菜单，单击“出局队列”，然后单击“截断归档”。
3. 填充适用字段。
4. 单击“执行”按钮（F8）。

第 7 部分 附录

附录 A. 快速步骤

此附录补充《*mySAP.com 适配器用户指南*》中包含的信息。不打算用它替换用户指南中的信息。

注: 此处提供的快速步骤用于以独立方式与 WebSphere Message Broker 或 WebSphere Application Server 一起作为集成代理程序运行的适配器。

在开始这些步骤之前，您必须执行以下操作:

- 安装 WebSphere Message Broker 或 WebSphere Application Server 来作为代理程序。
- 安装 SAP JCo API。可从 SAP 的 Web 站点下载 SAP JCo，网址为: http://service.sap.com/connectors。您必须具有 SAPNet 帐户才能访问 SAP JCo（如果您尚未具有该帐户，则联系本地 SAP Basis 管理员）。将这些文件添加至 *ProductDir*\ODA\SAP 和 *ProductDir*\connectors\SAP 目录，其中 *ProductDir* 表示连接器的安装目录。
- 安装 JDK。
- 安装 WebSphere Business Integration 适配器运行时环境（适配器框架）。
- 配置标准 MQ 队列。

只要遵循这些快速步骤，您就可以创建自己的业务对象或者使用所提供的样本业务对象。可以在 *ProductDir*\connectors\SAP\samples 目录中找到这些样本，其中 *ProductDir* 表示连接器的安装目录。

公共配置属性

以下各表列示必须为 WMQI 代理程序维护的配置属性。使用 CN_SAP.txt 创建 SAP 配置文件。此文件位于 %CROSSWORLDS%\repository\SAP。使用连接器配置器打开该文件。

表 48. 标准配置属性

属性名	缺省值	需要的值
ApplicationName	无	SAPConnector
BrokerType	ICS	WMQI
AdminInQueue	/ADMININQUEUE	ADMININQUEUE
AdminOutQueue	/ADMINOUTQUEUE	ADMINOUTQUEUE
DeliveryQueue	/DELIVERYQUEUE	DELIVERYQUEUE
FaultQueue	/FAULTQUEUE	FAULTQUEUE
RequestQueue	/REQUESTQUEUE	REQUESTQUEUE
ResponseQueue	/RESPONSEQUEUE	RESPONSEQUEUE
SynchronousRequestQueue	/SYNCHRONOUSREQUESTQUEUE	SYNCHRONOUSREQUESTQUEUE
SynchronousResponseQueue	/SYNCHRONOUSRESPONSEQUEUE	SYNCHRONOUSRESPONSEQUEUE
MessageFileName	SAPConnector.txt	SAPCONNECTOR.TXT
RepositoryDirectory	C:\crossworlds\repository	*<业务对象说明的位置>*
Jms.MessageBrokerName	crossworlds.queue.manager	*<队列管理器名称>*

表 48. 标准配置属性 (续)

属性名	缺省值	需要的值
AgentTraceLevel	0	5

表 49. 特定于连接器的属性

属性名	缺省值	需要的值
ApplicationPassword	SOFTWARE	*<SAP 应用程序的密码>*
ApplicationUserName	CROSSWORLDS	*<SAP 应用程序的用户名>*
Client	无	*<客户机号>*
Hostname	无	*<SAP 应用程序服务器名称>*

BAPI 模块的快速步骤

在配置 BAPI 模块之前，配置以下特定于连接器的属性:

属性名	缺省值	需要的值
Modules	无	BAPI

在 BAPI 模块中生成业务对象

要为 BAPI 模块生成业务对象:

1. 启动 SAP ODA。
2. 启动业务对象设计器。
3. 在业务对象设计器中，选择“文件”>“新建”。向导启动。

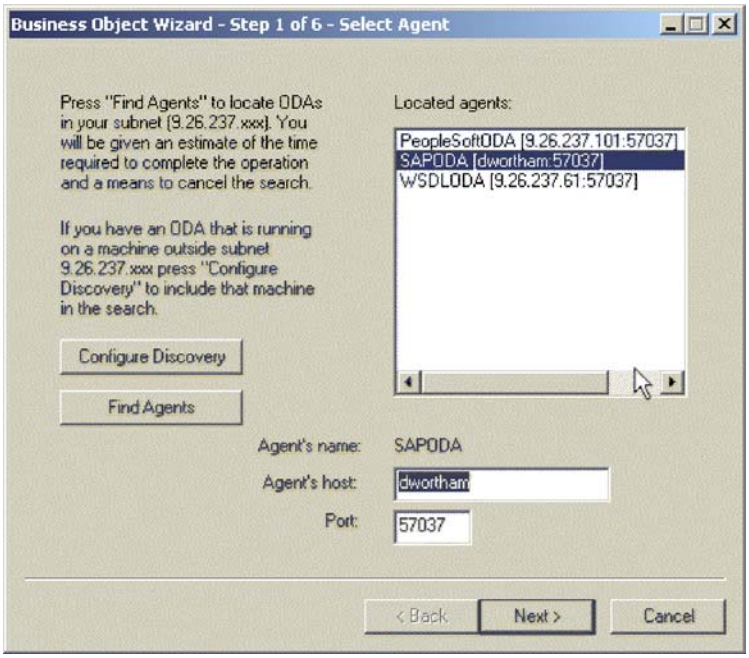


图 76. 业务对象向导 – 选择代理程序

4. 选择“配置发现”：
   a. 输入运行“发现”的机器的主机地址。
   b. 选择“添加主机”。
   c. 选择“确定”。
5. 选择“查找代理程序”。
   a. 突出显示“代理程序”。选择“下一步”。
   b. 填充 UserName、Password、Client、SystemNumber、ASHostName 和 FileDestination 的值。保存该概要文件。
6. 在向导的步骤 3 中，展开 RFC 节点。
   a. 右键单击“按名称搜索”。
   b. 输入 `bapi_customer_getdetail`。
   c. 突出显示 `bapi_customer_getdetail`。
   d. 选择“下一步”。

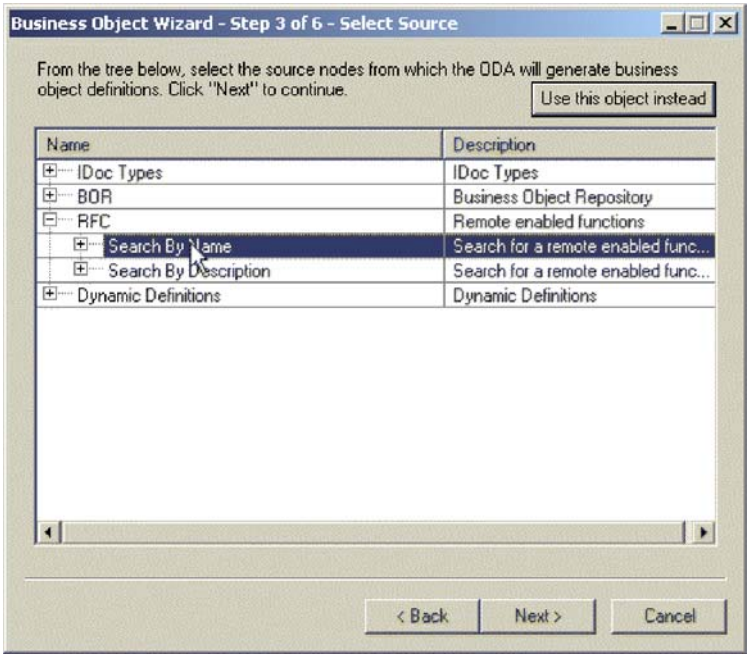


图 77. 业务对象向导 - 选择源

7. 选择“下一步”。
8. 将查询描述设置为**检索**，将“服务器支持”设置为**否**。选择“确定”。
9. 在“代理程序 SAPODA 通知”中，选择“否”。
10. 在不同窗口中打开业务对象。将有保证的业务对象说明保存到您在“资源库目录”标准属性值中指定的位置。

配置 BAPI 模块

在生成业务对象之后，将父对象名添加至配置文件的“受支持的业务对象”部分来继续配置 BAPI 模块。

准备 BAPI 模块以进行测试

要设置 BAPI 模块以进行测试，使用端口连接器:

1. 复制 SAP 配置文件。将复制的文件重命名 portconnector.cfg。
2. 在连接器配置器中打开 portconnector.cfg。
3. 在“标准”选项卡中更改以下属性:
   - 将 ApplicationName 更改为 PortConnector
   - 将 DELIVERYQUEUE 更改为 REQUESTQUEUE
   - 将 REQUESTQUEUE 更改为 RESPONSEQUEUE
4. 保存更改。关闭 portconnector.cfg。
5. 打开 sapconnector.cfg。
6. 保存更改。启动 mySAP.com。

测试 BAPI 模块

要测试 BAPI 模块:

1. 打开测试连接器。

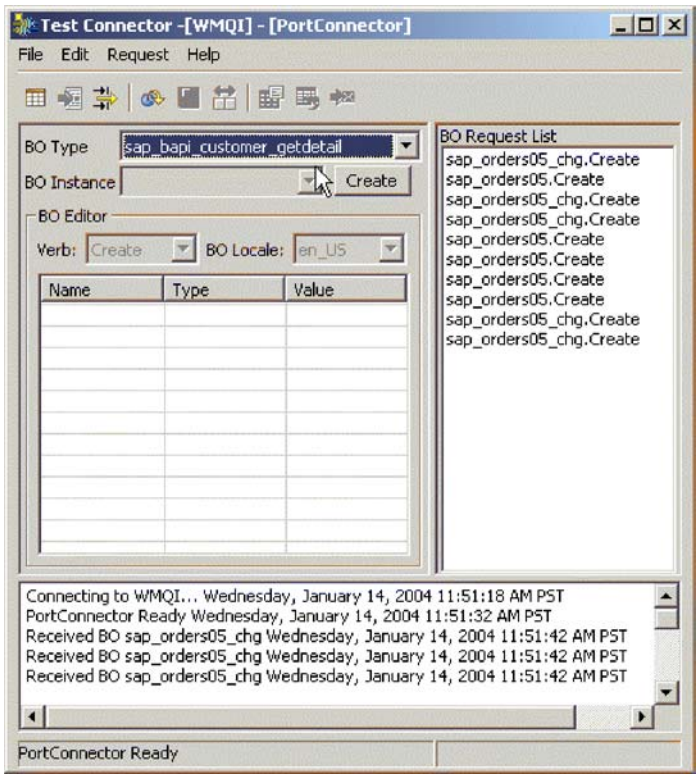


图 78. 测试连接器

2. 选择“文件” > “创建 / 选择概要文件”。
3. 选择“文件” > “新建概要文件”。

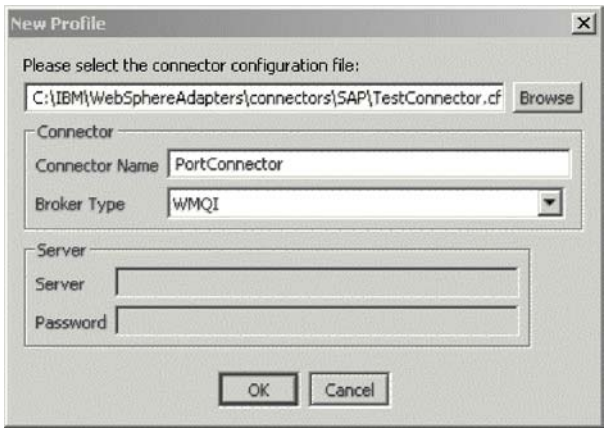


图 79. 测试连接器 – 新建概要文件

4. 选择“浏览”。
   a. 找到 portconnector.cfg。选择“打开”。
   b. 对“连接器名称”输入 PortConnector。
   c. 对“代理程序类型”输入 WMQI。
   d. 选择“确定”。
5. 突出显示 PortConnector。选择“确定”。
6. 选择“文件”>“连接”。
7. 创建业务对象实例:
   a. 对“BO 类型”选择 SAP_BAPI_customer_getdetail。
   b. 选择“创建”。
   c. 输入“新对象”。选择“确定”。
8. 将“查询描述”更改为“检索”。用现有的客户填充 Customer_to_be_required。
9. 选择“请求”>“发送”。
10. 检查日志文件以获取成功消息。

RFC 服务器模块的快速步骤

在配置 RFC 模块之前，配置下列特定于连接器的属性:

属性名	缺省值	需要的值
Modules	无	Rfcserver
RfcProgramId	CWLDSERVER	*<在 SAP 事务 sm59 中注册的程序标识>*

在 RFC 服务器模块中生成业务对象

要为 RFC 模块生成业务对象:

1. 启动 SAP ODA。
2. 启动业务对象设计器。
3. 在业务对象设计器中，选择“文件”>“新建”。向导启动。

4. 选择“配置发现”：
   a. 输入运行“发现”的机器的主机地址。
   b. 选择“添加主机”。
   c. 选择“确定”。
5. 在向导的步骤 3 中，展开 RFC 节点。
   a. 右键单击“按名称搜索”。
   b. 输入 bapi_customer_getdetail。
   c. 突出显示 bapi_customer_getdetail。
   d. 选择“下一步”。
6. 选择“下一步”。
7. 将“查询描述”设置为“检索”，将“服务器支持”设置为“否”。选择“确定”。
8. 在“代理程序 SAPODA 通知”中，选择“否”。
9. 在不同窗口中打开业务对象。选择“常规”>“设置 Collab = ″RFCCollab″”。
10. 将有保证的业务对象说明保存到您在“资源库目录”标准属性值中指定的位置。

配置 RFC 服务器模块

在生成业务对象之后，继续配置 RFC 服务器模块:

1. 将父对象名添加至配置文件的“受支持的业务对象”节。
2. 将生成的 BOHandler .class 文件从 ODA 配置属性中指定的定义复制到 %CROSSWORLD%\connectors\SAP\rfc\client。

为 SAP 服务器创建概要文件

要为 SAP 服务器创建概要文件:

1. 打开“SAP 登录”。
2. 选择“新建”。
3. 填充以下字段，然后选择“确定”：

描述	服务器的主机名
应用程序服务器	服务器的主机名
系统号	00
描述	主机名是标准的。输入您选择的描述。

4. 双击以打开您刚创建的概要文件。
5. 输入用户名和密码。选择“事务”>“类型 /nse37”。功能构建器打开。
6. 对“功能模块”输入 bapi_customer_getdetail。选择“功能模块”>“测试”>“单一测试”。
7. 对“RFC 目标系统”使用您在特定于连接器的属性中设置的 Rfcprogramid 值。并填充以下字段:

字段	示例
Customer Number	0000000001

PI_SALESORG	0001
PI_DISTR_CHAN	01
PI_DIVISION	01

测试 RFC 服务器模块

要设置 BAPI 模块以进行测试，使用端口连接器:

1. 复制 SAP 配置文件。将复制的文件重命名 portconnector.cfg。
2. 在连接器配置器中打开 portconnector.cfg。
3. 在“标准”选项卡中更改以下属性:
   - 将 ApplicationName 更改为 PortConnector
   - 将 REQUESTQUEUE 更改为 SYNCHRONOUSREQUESTQUEUE。

   保存更改并关闭窗口。
4. 打开 sapconnector.cfg。
5. 将 REQUESTQUEUE 更改为 SYNCHRONOUSREQUESTQUEUE。保存更改。
6. 启动连接器。选择“功能模块” > “执行”。
7. 在测试连接器中，在 BO 请求列表中查找该对象。突出显示该对象，并选择“请求” > “应答” > “成功”。
8. 检查日志以获取成功消息。

ALE 模块的快速步骤

在配置 ALE 模块之前，创建以下持久 WebSphere MQ 队列:

- SAPtid_Queue
- SAPtid_QueueManager
- SAPALE_Event_Queue
- SAPALE_Wip_Queue
- SAPALE_Archive_Queue
- SAPALE_UnSubscribed_Queue
- SAPALE_Error_Queue

参阅 MQ Series 文档以获取关于创建 MQ 队列的信息。

下一步，配置以下特定于连接器的属性:

属性名	缺省值	需要的值
Modules	无	Ale
AleEventDir	无	%CROSSWORLDS%\connectors\SAP\ale
SAPtid_QueueManager	无	*<队列管理器名称>*
SAPtid_Queue	无	*<队列名>*
SAPALE_Event_Queue	无	*<事件队列名称>*
SAPALE_Wip_Queue	无	*<WIP 队列名称>*
SAPALE_Archive_Queue	无	*<归档队列名称>*

属性名	缺省值	需要的值
SAPALE_UnSubscribed_Queue	无	*<未预订的队列名称>*
SAPALE_Error_Queue	无	*<错误队列名称>*
RfcProgramId	无	*<在 SAP 事务 sm59 中定义的程序标识名称>*
NumberOfListeners	1	1（对于单线程）

对于远程 WebSphere 队列，还应配置以下属性:

属性名	缺省值	需要的值
SAPtid_QueueManagerLogin	无	*<队列管理器登录名称>*
SAPtid_QueueManagerPassword	无	*<队列管理器密码>*
SAPtid_QueueManagerHost	无	*<队列管理器主机>*
SAPtid_MQPort	无	*<MQ 端口>*
SAPtid_MQChannel	无	*<MQ 通道>*

在 ALE 模块中生成业务对象

要在 ALE 模块中生成业务对象:

1. 启动 SAP ODA。
2. 启动业务对象设计器。
3. 在业务对象设计器中，选择“文件”>“新建”。向导启动。
4. 选择“配置发现”：
   a. 输入运行“发现”的机器的主机地址。
   b. 选择“添加主机”。
   c. 选择“确定”。
5. 在向导的步骤 3 中，展开“IDoc 类型”。
   a. 展开“从系统生成”。
   b. 展开“基本 IDoc 类型”。
   c. 右键单击“按名称选择...”
   d. 选择“搜索项...”
   e. 输入 orders03。选择“确定”。
6. 突出显示 ORDERS03。选择“下一步”。
7. 选择“下一步”。
8. 选择“确定”。将生成该业务对象。
9. 选择“将业务对象定义的副本保存到不同文件”并选择“在不同窗口打开新的业务对象定义”。选择“完成”。

编辑业务对象

要编辑业务对象:

1. 选择“常规”选项卡。
2. 将“创建特定于应用程序的参考消息类型”更改为 MsgType = ORDERS。

3. 打开 %CROSSWORLDS\repository\SAP\BO_SAPIDocControl.txt 并将它保存到 Repository 目录。
4. 将父对象名添加至配置文件的“受支持的业务对象”节。
5. 使用 SAP 事务 SM59 向 SAP 网关注册 RFC 服务器模块。
6. 确保:
   - 已对 SAP 系统和外部系统定义并分配逻辑系统（SALE）。
   - 已保留分布模型并已将必需的消息类型添加至该模型（事务代码 BD64）。
   - 逻辑系统或分布模型具有伙伴概要文件（事务代码 WE20）。
   - 为逻辑系统或分布模型定义了端口（事务代码 E201。

准备 ALE 模块以进行测试

要设置 ALE 模块以进行测试，使用端口连接器:

1. 复制 SAP 配置文件。将复制的文件重命名 portconnector.cfg。
2. 在连接器配置器中打开 portconnector.cfg。
3. 在“标准”选项卡中更改以下属性:
   - 将 ApplicationName 更改为 PortConnector
   - 将 DELIVERYQUEUE 更改为 REQUESTQUEUE
   - 将 REQUESTQUEUE 更改为 RESPONSEQUEUE
4. 保存更改。关闭 portconnector.cfg。
5. 打开 sapconnector.cfg。
6. 保存更改。启动 mySAP.com。

测试 ALE 模块的请求处理

要测试 ALE 模块:

1. 打开测试连接器。
2. 选择“文件” > “创建 / 选择概要文件”。
3. 选择“文件” > “新建概要文件”。
4. 选择“浏览”。
   a. 选择“打开”。
   b. 对“连接器名称”输入 PortConnector。
   c. 对“代理程序类型”输入 WMQI。
   d. 选择“确定”。
5. 突出显示 PortConnector。选择“确定”。
6. 选择“文件” > “连接”。
7. 创建业务对象实例:
   a. 对“BO 类型”选择 sap_order03。
   b. 选择“创建”。
   c. 在“输入名称”中，输入新对象。选择“确定”。
8. 将查询描述更改为“创建”。
9. 右键单击“控制记录”。选择“添加实例”。

10. 展开“控制记录”。填充以下字段:
    - IDoc_number
    - Sender_port
    - Partner_number_of_sender
    - Receiver_port
    - Partner_number_of_recipient
    - Client
    - SAP_Release
11. 启动连接器。
12. 在测试连接器中，选择“请求”>“发送”。检查日志以获取成功消息。

测试 ALE 模块中的事件处理

要测试 ALE 模块中的事件处理:

1. 转至事务 we19 “测试用于 IDoc 处理的工具”。
2. 用现有的 IDoc 填充字段。选择 IDoc >“创建”。
3. 选择 StandardOutboundProcessing 以将 IDoc 发送至测试连接器。
4. 在弹出窗口中，选择选取标记。
5. 要验证是否从 SAP 发送了 IDoc，检查 mySAP.com 连接器日志文件以获取成功消息。如果该事件在事务 sm58 中存在，则未正确发送它。
6. 查看已发送至 SAPALE_Archive_Queue 的消息以验证 ProcessingStatus 是否成功。如果未看到成功消息，则检查 SAPALE_Error_Queue 以查看是否发生故障。

HDR 模块的快速步骤

在配置 HDR 模块之前，配置以下特定于连接器的属性:

属性名	缺省值	需要的值
Modules	无	BAPI

在 HDR 模块中生成业务对象

要在 HDR 模块中生成业务对象:

1. 启动 SAP ODA。
2. 启动业务对象设计器。
3. 在业务对象设计器中，选择“文件”>“新建”。向导启动。
4. 选择“配置发现”:
    a. 输入运行“发现”的机器的主机地址。
    b. 选择“添加主机”。
    c. 选择“确定”。
5. 在向导的步骤 3 中，展开“动态定义”。
    a. 展开 HDR。
    b. 右键单击“按名称搜索...”。选择“搜索项”。

c. 输入 kna1。选择“确定”。

d. 突出显示 kna1。选择“下一步”。

e. 选择“下一步”。

f. 选择“确定”。

6. 在“通知”中，选择“否”。
7. 选择“在不同窗口打开新的业务对象定义”。选择“完成”。
8. 将新业务对象保存到 Repository 目录。

准备 HDR 模块以进行测试

要设置 HDR 模块以进行测试，使用端口连接器:

1. 复制 SAP 配置文件。将复制的文件重命名 portconnector.cfg。
2. 在连接器配置器中打开 portconnector.cfg。
3. 在“标准”选项卡中更改以下属性:
   - 将 ApplicationName 更改为 PortConnector
   - 将 DELIVERYQUEUE 更改为 REQUESTQUEUE
   - 将 REQUESTQUEUE 更改为 RESPONSEQUEUE
4. 保存更改。
5. 打开 sapconnector.cfg。
6. 将 REQUESTQUEUE 更改为 SYNCHRONOUSREQUESTQUEUE。
7. 保存更改。

测试 HDR 模块

要测试 HDR 模块:

1. 打开测试连接器。
2. 在“业务对象类型”中，选择 SAP_kna1。选择“创建”。
3. 在“输入名称”中，输入新对象。选择“确定”。
4. 将查询描述更改为“检索”。
5. 用现有的 SAP 客户号填充 customer_number_KUNNR。该号码的长度必须为 10 位，例如: 0000000001。
6. 选择“请求” > “发送”。
7. 检查日志文件以获取成功消息。

附录 B. Common Event Infrastructure

WebSphere Business Integration Server Foundation 包括 Common Event Infrastructure Server Application，必须具有它 Common Event Infrastructure 才能运行。WebSphere Application Server Foundation 可以安装在任何系统上（它不必与适配器安装在同一机器上）。

WebSphere Application Server Application Client 包括适配器与 Common Event Infrastructure Server Application 进行交互所必需的库。必须在安装适配器的系统上安装 WebSphere Application Server Application Client。适配器通过可配置的 URL 连接至 WebSphere Application Server（在 WebSphere Business Integration Server Foundation 中）。

使用此发行版支持的任何集成代理程序提供了 Common Event Infrastructure 支持。

必需的软件

除了适配器需要的必备软件以外，必须安装了下列软件 Common Event Infrastructure 才能运行:

- WebSphere Business Integration Server Foundation 5.1.1
- WebSphere Application Server Application Client 5.0.2、5.1 或 5.1.1。

  （WebSphere Application Server Application Client 5.1.1 是与 WebSphere Business Integration Server Foundation 5.1.1 一起提供的。）

注: Common Event Infrastructure 在任何 HP-UX 或 Linux 平台上都不受支持。

启用 Common Event Infrastructure

Common Event Infrastructure 功能是通过使用“连接器配置器”配置的标准属性 `CommonEventInfrastructure` 和 `CommonEventInfrastructureContextURL` 来启用的。缺省情况下未启用 Common Event Infrastructure。`CommonEventInfrastructureContextURL` 属性使您能够配置 Common Event Infrastructure 服务器的 URL。（有关更多信息，参阅本文档的『标准属性』附录。）

获取 Common Event Infrastructure 适配器事件

如果启用了 Common Event Infrastructure，则适配器将生成映射至下列适配器事件的 Common Event Infrastructure 事件:

- 启动适配器
- 停止适配器
- 应用程序对适配器代理程序产生的超时作出响应
- 从适配器代理程序发出的任何 `doVerbFor` 调用
- 从适配器代理程序发出的 `gotApplEvent` 调用

要使另一个应用程序（“用户应用程序”）接收由适配器生成的 Common Event Infrastructure 事件，该应用程序必须使用 Common Event Infrastructure 事件目录来确定适当事件及其属性的定义。必须在事件目录中定义事件，用户应用程序才能使用发送应用程序的事件。

本文档的『Common Event Infrastructure 事件目录定义』附录包含一些 XML 格式元数据，对于 WebSphere Business Information 适配器，这些元数据显示用户应用程序应该搜索的事件描述符和属性。

获取更多信息

有关 Common Event Infrastructure 的更多信息，参阅在以下 URL 提供的 WebSphere Business Integration Server Foundation 文档中包含的 Common Event Infrastructure 信息:

http://publib.boulder.ibm.com/infocenter/ws51help

参阅『Common Event Infrastructure 事件目录定义』以获取有关样本 XML 元数据的信息，该元数据显示用户应用程序应该搜索的适配器生成的事件描述符和属性。

Common Event Infrastructure 事件目录定义

Common Event Infrastructure 事件目录包含可供其它应用程序查询的事件定义。下面是典型适配器事件的使用 XML 元数据的事件定义样本。如果要编写另一个应用程序，则应用程序可以使用事件目录接口来查询事件定义。有关事件定义以及如何查询它们的更多信息，参阅联机 IBM WebSphere Server Foundation Information Center 提供的 Common Event Infrastructure 文档。

对于 WebSphere Business Integration 适配器，需要在事件目录中定义的扩展数据元素是业务对象的键。每个业务对象键都需要事件定义。因此，对于给定的任何适配器，各种事件（例如，启动适配器、停止适配器、适配器超时）以及任何 doVerbFor 事件（例如，创建、更新或删除）在事件目录中都必须具有相应的事件定义。

下列各节包含用于启动适配器、停止适配器以及事件请求或传递的 XML 元数据的一些示例。

XML 格式的“启动适配器”元数据

```
<eventDefinition name="startADAPTER"
      parent="event">
    <property name ="creationTime" //Comment: example value would be
 "2004-05-13T17:00:16.319Z"
         required="true" />
    <property name="globalInstanceId" //Comment: Automatically generated
 by Common Event Infrastructure
         required="true"/>
    <property name="sequenceNumber"    //Comment: Source defined number
for messages to be sent/sorted logically
         required="false"/>
    <property name="version"    //Comment: Version of the event
         required="false"
         defaultValue="1.0.1"/>
    <property name="sourceComponentId"
```

```
         path="sourceComponentId"
         required="true"/>
    <property name="application"   //Comment: The name#version of the
source application generating the event. Example is "SampleConnector#3.0.0"
         path="sourceComponentId/application"         required="false"/>
    <property name="component"   //Comment: This will be the name#version
 of the source component.
         path="sourceComponentId/component"
         required="true"
         defaultValue="ConnectorFrameWorkVersion#4.2.2"/>
    <property name="componentIdType"    //Comment: specifies the format
and meaning of the component
          path="sourceComponentId/componentIdType"
          required="true"
          defaultValue="Application"/>
    <property name="executionEnvironment"
 //Comment: Identifies the environment the application is running
 in...example is "Windows 2000#5.0"
          path="sourceComponentId/executionEnvironment"
          required="false" />
     <property name="location"    //Comment: The value of this is the
 server name...example is "WQMI"
          path="sourceComponentId/location"
          required="true"/>
    <property name="locationType"   //Comment specifies the format and
     meaning of the location
          path="sourceComponentId/locationType"
          required="true"
          defaultValue="Hostname"/>
     <property name="subComponent"     //Comment:further distinction
of the logical component
          path="sourceComponentId/subComponent"
          required="true"
          defaultValue="AppSide_Connector.AgentBusinessObjectManager"/>
     <property name="componentType"      //Comment: well-defined name
used to characterize all instances of this component
          path="sourceComponentId/componentType"
          required="true"
          defaultValue="ADAPTER"/>
     <property name="situation"   //Comment: Defines the type of
 situation that caused the event to be reported
          path="situation"
          required="true"/>
     <property name="categoryName="    //Comment: Specifies the type
of situation for the event
         path="situation/categoryName"
         required="true"
         defaultValue="StartSituation"/>
    <property name="situationType"    //Comment: Specifies the type
of situation and disposition of the event
         path="situation/situationType"
         required="true"
    <property name="reasoningScope" //Comment: Specifies the scope
 of the impact of the event
         path="situation/situationType/reasoningScope"
         required="true"
         permittedValue="INTERNAL"
         permittedValue="EXTERNAL"/>
    <property name="successDisposition" //Comment: Specifies the
 success of event
         path="situation/situationType/successDisposition"
         required="true"
         permittedValue="SUCCESSFUL"
         permittedValue="UNSUCCESSFUL" />
    <property name="situationQualifier"  //Comment: Specifies the
 situation qualifiers for this event
         path="situation/situationType/situationQualifier"
```

```
        required="true"
        permittedValue="START_INITIATED"
        permittedValue="RESTART_INITIATED"
        permittedValue="START_COMPLETED" />
</eventDefinition>
```

XML 格式的“停止适配器”元数据

除了下列不同之处以外，“停止适配器”与“启动适配器”的元数据是相同的:

- categoryName 属性的缺省值为 StopSituation:

```
<property name="categoryName="
 //Comment: Specifies the type
 of situation for the event
             path="situation/categoryName"
             required="true"
             defaultValue="StopSituation"/>
```

- situationQualifier 属性的允许值也有所不同，对于“停止适配器”，该属性的允许值为如下所示:

```
<property name="situationQualifier"
 //Comment: Specifies the situation qualifiers for this event
          path="situation/situationType/situationQualifier"
          required="true"
          permittedValue="STOP_INITIATED"
          permittedValue="ABORT_INITIATED"
          permittedValue="PAUSE_INITIATED"
          permittedValue="STOP_COMPLETED"
 />
```

XML 格式的“适配器超时”元数据

除了下列不同之处以外，“适配器超时”与“启动适配器”和“停止适配器”的元数据是相同的:

- categoryName 属性的缺省值为 ConnectSituation:

```
<property name="categoryName="
 //Comment: Specifies the type
 of situation for the event
             path="situation/categoryName"
             required="true"
             defaultValue="ConnectSituation"/>
```

- situationQualifier 属性的允许值也有所不同，对于“适配器超时”，该属性的允许值为如下所示:

```
<property name="situationQualifier"  //Comment: Specifies
 the situation qualifiers for this event
          path="situation/situationType/situationQualifier"
          required="true"
          permittedValue="IN_USE"
          permittedValue="FREED"
          permittedValue="CLOSED"
          permittedValue="AVAILABLE"
 />
```

XML 格式的“请求”或“传递”元数据

此 XML 格式的末尾是扩展数据元素。适配器请求和传递事件的扩展数据元素表示正在处理的业务对象中的数据。此数据包括业务对象的名称、业务对象的键（外键或本地键）以及父代业务对象的子业务对象。子业务对象又分成与父代相同的数据（名称、键和任何子业务对象）。此数据是在事件定义的扩展数据元素中表示的。此数据将随着正在处理的业务对象、键和子业务对象不同而不同。此事件定义中的扩展数据只是一个示例，它表示具有键 EmployeeId 的名为 Employee 的业务对象以及具有键 EmployeeId 的子业务对象 EmployeeAddress。对于特定业务对象，此模式可以处理所有存在的数据。

```
<eventDefinition name="createEmployee"     //Comment: This
 extension name is always the business object verb followed by the business
 object name
          parent="event">
    <property name ="creationTime"  //Comment: example value would be
"2004-05-13T17:00:16.319Z"
         required="true" />
    <property name="globalInstanceId" //Comment: Automatically generated
 by Common Event Infrastructure
         required="true"/>
    <property name="localInstanceId"    //Comment: Value is business
 object verb+business object name+#+app name+ business object identifier
         required="false"/>
    <property name="sequenceNumber"    //Comment: Source defined number
for messages to be sent/sorted logically
         required="false"/>
    <property name="version"  //Comment: Version of the event...value is
 set to 1.0.1
         required="false"
         defaultValue="1.0.1"/>
    <property name="sourceComponentId"
         path="sourceComponentId"
         required="true"/>
    <property name="application"    //Comment: The name#version of the
 source application generating the event...example is
 "SampleConnector#3.0.0"
         path="sourceComponentId/application"
         required="false"/>
    <property name="component"   //Comment: This will be the name#version
of the source component.
         path="sourceComponentId/component"
         required="true"
         defaultValue="ConnectorFrameWorkVersion#4.2.2"/>
    <property name="componentIdType"     //Comment: specifies the format
 and meaning of the component
         path="sourceComponentId/componentIdType"
         required="true"
         defaultValue="Application"/>
    <property name="executionEnvironment" //Comment: Identifies the
 environment#version the app is running in...example is "Windows 2000#5.0"
         path="sourceComponentId/executionEnvironment"
         required="false" />
    <property name="instanceId" //Comment: Value is business object
  verb+business object name+#+app name+ business object identifier
          path="sourceComponentId/instanceId"
          required="false"
    <property name="location"   //Comment: The value of this is the
server name...example is "WQMI"
          path="sourceComponentId/location"
          required="true"/>
     <property name="locationType" //Comment specifies the format and
 meaning of the location
          path="sourceComponentId/locationType"
          required="true"
          defaultValue="Hostname"/>
```

```
    <property name="subComponent"  //Comment:further distinction of the
 logical component-in this case the value is the name of the business
 object
          path="sourceComponentId/subComponent"
          required="true"/>
    <property name="componentType"      //Comment: well-defined name used
 to characterize all instances of this component
          path="sourceComponentId/componentType"
          required="true"
          defaultValue="ADAPTER"/>
    <property name="situation" //Comment: Defines the type of
situation that caused the event to be reported
          path="situation"
          required="true"/>
   <property name="categoryName"    //Comment: Specifies the type
 of situation for the event
         path="situation/categoryName"
         required="true"
         permittedValue="CreateSituation"
         permittedValue="DestroySituation"
         permittedValue="OtherSituation" />
   <property name="situationType"    //Comment: Specifies the type
of situation and disposition of the event
         path="situation/situationType"
         required="true"
   <property name="reasoningScope" //Comment: Specifies the scope
of the impact of the event
         path="situation/situationType/reasoningScope"
         required="true"
         permittedValue="INTERNAL"
         permittedValue="EXTERNAL"/>
   <property name="successDisposition" //Comment: Specifies the
 success of event
         path="situation/situationType/successDisposition"
         required="true"
         permittedValue="SUCCESSFUL"
         permittedValue="UNSUCCESSFUL" />
   <extendedDataElements name="Employee" //Comment: name of business
 object itself
             type="noValue"
             <children name="EmployeeId"
                  type="string"/>  //Comment: type is one of the
 permitted values within Common Event Infrastructure documentation
             <children name="EmployeeAddress"
                  type="noValue"/>
                     <children name="EmployeeId"
                         type="string"/>
                      -
                      -
                      -
   </extendedDataElements
</eventDefinition>
```

附录 C. 应用程序响应测量

此适配器与应用程序响应测量应用编程接口（API）兼容，该 API 允许对应用程序的可用性、服务级别协议和容量规划进行管理。ARM 检测应用程序可以参与 IBM Tivoli Monitoring for Transaction Performance，允许收集和查看与事务度量有关的数据。

应用程序响应测量检测支持

此适配器与应用程序响应测量应用编程接口（API）兼容，该 API 允许对应用程序的可用性、服务级别协议和容量规划进行管理。ARM 检测应用程序可以参与 IBM Tivoli Monitoring for Transaction Performance，允许收集和查看与事务度量有关的数据。

必需的软件

除了适配器需要的必备软件以外，必须安装了下列软件 ARM 才能运行:

- WebSphere Application Server 5.0.1（包含 IBM Tivoli Monitoring for Transaction Performance 服务器）。此软件不必与适配器安装在同一系统上。
- IBM Tivoli Monitoring for Transaction Performance V.5.2 修订包 1。必须将此修订包与适配器安装在同一系统上，并将此修订包配置为指向 IBM Tivoli Monitoring for Transaction Performance 服务器所驻留的系统。

使用此发行版支持的任何集成代理程序可提供应用程序响应测量支持。

注: 除了 HP-UX（任何版本）和 Red Hat Linux 3.0 之外，在此 IBM WebSphere Business Integration Adapters 发行版支持的所有操作系统上都支持应用程序响应测量检测。

启用应用程序响应测量

可通过将“连接器配置器”中的标准属性 TivoliMonitorTransactionPerformance 设置为“True”来启用 ARM 检测。缺省情况下未启用 ARM 支持。（有关更多信息，参阅本文档的『标准属性』附录。）

事务监视

当启用了 ARM 时，受监视的事务是服务事件和事件传递。对事务的测量是从服务请求或事件传递开始直到服务请求或事件传递结束。在 Tivoli Monitoring for Transaction Performance 控制台上显示的事务名称将以 SERVICE REQUEST 或 EVENT DELIVERY 开头。该名称的下一部分将为业务对象查询描述（例如，CREATE、RETRIEVE、UPDATE 或 DELETE）。该名称的最后一部分将为业务对象名称，例如，“EMPLOYEE”。例如，用于创建职员的事件传递的事务的名称可能为 EVENT DELIVERY CREATE EMPLOYEE。另一种可能是 SERVICE REQUEST UPDATE ORDER。

缺省情况下，将为每种类型的服务请求或事件传递收集下列度量:

- 最短事务时间
- 最长事务时间
- 平均事务时间

- 运行的总事务数

您（或者 WebSphere Application Server 的系统管理员）可以通过从 Tivoli Monitoring for Transaction Performance 控制台中为特定事务配置“发现策略”和“侦听器策略”，来选择要为哪些适配器事件显示哪些度量。（请参阅『获取更多信息』。）

获取更多信息

有关更多信息，参阅 IBM Tivoli Monitoring for Transaction Performance 文档。特别是，参阅 *IBM Tivoli Monitoring for Transaction Performance User's Guide* 以获取有关监视和管理由适配器生成的度量的信息。

附录 D. 连接器的标准配置属性

本附录描述 WebSphere Business Integration 适配器的连接器组件的标准配置属性。此信息包括与下列集成代理程序一起运行的连接器:

- WebSphere InterChange Server（ICS）
- WebSphere MQ Integrator、WebSphere MQ Integrator Broker 和 WebSphere Business Integration Message Broker，统称为 WebSphere Message Broker（在“连接器配置器”中显示为 WMQI）。
- Information Integrator（II）
- WebSphere Application Server（WAS）

如果适配器支持 DB2 Information Integrator，则使用 WMQI 选项和 DB2 II 标准属性（请参阅第 263 页的表 50 中的“注释”列。）

为适配器设置的属性取决于您使用的是哪个集成代理程序。使用连接器配置器选择集成代理程序。选择代理程序之后，连接器配置器将列示必须为适配器配置的标准属性。

有关特定于此连接器的属性的信息，请参阅本指南中的相关章节。

新增属性

在本发行版中添加了下列标准属性:

- AdapterHelpName
- BiDi.Application
- BiDi.Broker
- BiDi.Metadata
- BiDi.Transformation
- CommonEventInfrastructure
- CommonEventInfrastructureContextURL
- ControllerEventSequencing
- jms.ListenerConcurrency
- jms.TransportOptimized
- ResultsSetEnabled
- ResultsSetSize
- TivoliTransactionMonitorPerformance

标准连接器属性概述

连接器具有两种类型的配置属性:

- 标准配置属性，供框架使用
- 特定于应用程序或连接器的配置属性，供代理程序使用

这些属性确定适配器框架和代理程序运行时行为。

本节描述如何启动连接器配置器，并且描述所有属性共有的特征。有关特定于连接器的配置属性的信息，请参阅其适配器用户指南。

启动连接器配置器

您应从连接器配置器配置连接器属性，可以从系统管理器访问连接器配置器。有关使用连接器配置器的更多信息，参阅本指南中关于连接器配置器的章节。

连接器配置器和系统管理器仅在 Windows 系统上运行。如果您正在 UNIX 系统上运行连接器，则您必须具有一台已安装这些工具的 Windows 机器。

要设置在 UNIX 上运行的连接器的连接器属性，您必须在 Windows 机器上启动系统管理器，连接至 UNIX 集成代理程序，并启动连接器的连接器配置器。

配置属性值概述

连接器使用以下顺序来确定属性的值:

1. 缺省值
2. 资源库（仅当 WebSphere InterChange Server（ICS）是集成代理程序时才有效）
3. 本地配置文件
4. 命令行

属性字段的缺省长度为 255 个字符。对 STRING 属性类型的长度没有限制。INTEGER 类型的长度由运行适配器的服务器来确定。

连接器在启动时获得其配置值。如果您在运行时会话期间更改一个或多个连接器属性的值，则属性的更新方法确定更改如何生效。

属性的更新特征（即，对连接器属性的更改如何才能生效以及何时生效）取决于该属性的性质。

有四种更新标准连接器属性的方法:

- **动态的**
  在系统管理器中保存了更改之后，新值就会立即生效。但是，如果连接器采用独立方式（独立于系统管理器），例如，如果它正在与其中一个 WebSphere Message Broker 一起运行，则只能通过配置文件来更改属性。在这种情况下，动态更新是不可能的。
- **代理程序重新启动（仅限于 ICS）**
  仅当您停止并重新启动连接器代理程序之后，新值才会生效。
- **组件重新启动**
  仅当在系统管理器中停止并重新启动连接器之后，新值才会生效。您不需要停止并重新启动代理程序或服务器进程。

- **系统重新启动**
  仅当您停止并重新启动连接器代理程序和服务器之后，新值才会生效。

要确定如何更新特定属性，参阅“连接器配置器”窗口中的**更新方法**列，或查看第 263 页的表 50 中的“更新方法”列。

标准属性可以驻留在三个位置。某些属性可以驻留在多个位置。

- **ReposController**
  属性驻留在连接器控制器中，并且仅在该位置有效。如果您在代理程序端更改值，它不会影响控制器。
- **ReposAgent**
  属性驻留在代理程序中，并且仅在该位置有效。本地配置可以覆盖此值，这取决于属性。
- **LocalConfig**
  属性驻留在连接器的配置文件中，并且只能通过配置文件起作用。控制器不能更改属性的值，并且不知道对配置文件进行的更改，除非重新部署了系统以显式地更新控制器。

标准属性快速参考

表 50 提供标准连接器配置属性的快速参考。并非所有连接器都需要所有这些属性，并且不同集成代理程序的属性设置可能也不相同。

有关每种属性的描述，请参阅紧接在表后面的那一节。

注: 在表 50 中的“注释”列中，短语“RepositoryDirectory 设置为 <REMOTE> ”表示代理程序是 InterChange Server。当代理程序是 WMQI 或 WAS 时，资源库目录设置为 *<ProductDir>*\repository

表 50. 标准配置属性总结

属性名	可能的值	缺省值	更新方法	注释
AdapterHelpName	*<ProductDir>*\bin\Data\App\Help*Regional Setting*\ 中的其中一个有效子目录	如果有效，则为模板名，或者为空白字段	组件重新启动	受支持的区域设置。包括 chs_chn、cht_twn、deu_deu、esn_esp、fra_fra、ita_ita、jpn_jpn、kor_kor、ptb_bra 和 enu_usa（缺省值）。
AdminInQueue	有效的 JMS 队列名	*<CONNECTORNAME>*/ADMININQUEUE	组件重新启动	仅当 DeliveryTransport 的值为 JMS 时此属性才有效
AdminOutQueue	有效的 JMS 队列名	*<CONNECTORNAME>*/ADMINOUTQUEUE	组件重新启动	仅当 DeliveryTransport 的值为 JMS 时此属性才有效
AgentConnections	1 到 4	1	组件重新启动	仅当 DeliveryTransport 的值为 MQ 或 IDL，RepositoryDirectory 设置为 <REMOTE> 且 BrokerType 的值为 ICS 时此属性才有效。
AgentTraceLevel	0 到 5	0	如果代理程序为 ICS，则是动态的；否则需要进行组件重新启动	

表 50. 标准配置属性总结 (续)

属性名	可能的值	缺省值	更新方法	注释
ApplicationName	应用程序名	为连接器应用程序名称指定的值	组件重新启动	
BiDi.Application	下列双向属性的任何有效组合: 第 1 个字母: I 或 V 第 2 个字母: L 或 R 第 3 个字母: Y 或 N 第 4 个字母: S 或 N 第 5 个字母: H、C 或 N	ILYNN（5 个字母）	组件重新启动	仅当 BiDi.Transformation 的值为 true 时此属性才有效
BiDi.Broker	下列双向属性的任何有效组合: 第 1 个字母: I 或 V 第 2 个字母: L 或 R 第 3 个字母: Y 或 N 第 4 个字母: S 或 N 第 5 个字母: H、C 或 N	ILYNN（5 个字母）	组件重新启动	仅当 BiDi.Transformation 的值为 true 时此属性才有效。如果 BrokerType 的值为 ICS, 则该属性是只读的。
BiDi.Metadata	下列双向属性的任何有效组合: 第 1 个字母: I 或 V 第 2 个字母: L 或 R 第 3 个字母: Y 或 N 第 4 个字母: S 或 N 第 5 个字母: H、C 或 N	ILYNN（5 个字母）	组件重新启动	仅当 BiDi.Transformation 的值为 true 时此属性才有效。
BiDi.Transformation	true 或 false	false	组件重新启动	仅当 BrokerType 的值不是 WAS 时此属性才有效。
BrokerType	ICS、WMQI 或 WAS	ICS	组件重新启动	
CharacterEncoding	任何受支持的代码。 列表中显示了以下子集: ascii7、ascii8、SJIS、 Cp949、GBK、Big5、 Cp297、Cp273、Cp280、 Cp284、Cp037 和 Cp437。	ascii7	组件重新启动	此属性仅对于 C++ 连接器有效。
CommonEventInfrastructure	true 或 false	false	组件重新启动	
CommonEventInfrastructureURL	URL 字符串，例如, corbaloc:iiop:host:2809。	没有缺省值。	组件重新启动	仅当 CommonEvent Infrastructure 的值为 true 时此属性才有效。
ConcurrentEventTriggeredFlows	1 到 32,767	1	组件重新启动	仅当 RepositoryDirectory 的值设置为 <REMOTE> 并且 BrokerType 的值为 ICS 时此属性才有效。
ContainerManagedEvents	空白或 JMS	空白	组件重新启动	仅当“交付传输”的值为 JMS 时此属性才有效。
ControllerEventSequencing	true 或 false	true	动态的	仅当“资源库目录”值设置为 <REMOTE> 且 BrokerType 的值为 ICS 时此属性才有效。
ControllerStoreAndForwardMode	true 或 false	true	动态的	仅当“资源库目录”值设置为 <REMOTE> 且 BrokerType 的值为 ICS 时此属性才有效。

表 50. 标准配置属性总结 (续)

属性名	可能的值	缺省值	更新方法	注释
ControllerTraceLevel	0 到 5	0	动态的	仅当 RepositoryDirectory 的值设置为 <REMOTE> 并且 BrokerType 的值为 ICS 时此属性才有效。
DeliveryQueue	任何有效 JMS 队列名	<*CONNECTORNAME*> /DELIVERYQUEUE	组件重新启动	仅当“交付传输”的值为 JMS 时此属性才有效。
DeliveryTransport	MQ、IDL 或 JMS	当 RepositoryDirectory 的值为 <REMOTE> 时，此属性为 IDL，否则为 JMS	组件重新启动	如果 RepositoryDirectory 的值不是 <REMOTE>，则此属性的唯一有效值是 JMS。
DuplicateEventElimination	true 或 false	false	组件重新启动	仅当 DeliveryTransport 的值为 JMS 时此属性才有效。
EnableOidForFlowMonitoring	true 或 false	false	组件重新启动	仅当 BrokerType 的值为 ICS 时此属性才有效。
FaultQueue	任何有效的队列名。	<*CONNECTORNAME*> /FAULTQUEUE	组件重新启动	仅当 DeliveryTransport 的值为 JMS 时此属性才有效。
jms.FactoryClassName	CxCommon.Messaging.jms.IBMMQSeriesFactory 或 CxCommon.Messaging.jms.SonicMQFactory，或任何 Java 类名	CxCommon.Messaging.jms.IBMMQSeriesFactory	组件重新启动	仅当 DeliveryTransport 的值为 JMS 时此属性才有效。
jms.ListenerConcurrency	1 到 32767	1	组件重新启动	仅当 jms.TransportOptimized 的值为 true 时此属性才有效。
jms.MessageBrokerName	如果 jms.FactoryClassName 的值为 IBM，则使用 crossworlds.queue.manager。	crossworlds.queue.manager	组件重新启动	仅当 DeliveryTransport 的值为 JMS 时此属性才有效。
jms.NumConcurrentRequests	正整数	10	组件重新启动	仅当 DeliveryTransport 的值为 JMS 时此属性才有效。
jms.Password	任何有效密码		组件重新启动	仅当 DeliveryTransport 的值为 JMS 时此属性才有效。
jms.TransportOptimized	true 或 false	false	组件重新启动	仅当 DeliveryTransport 的值为 JMS 并且 BrokerType 的值为 ICS 时此属性才有效。
jms.UserName	任何有效名称		组件重新启动	仅当“交付传输”的值为 JMS 时此属性才有效。
JvmMaxHeapSize	以兆字节计的堆大小	128M	组件重新启动	仅当“资源库目录”值设置为 <REMOTE> 且 BrokerType 的值为 ICS 时此属性才有效。
JvmMaxNativeStackSize	以千字节计的堆栈大小	128K	组件重新启动	仅当“资源库目录”值设置为 <REMOTE> 且 BrokerType 的值为 ICS 时此属性才有效。
JvmMinHeapSize	以兆字节计的堆大小	1M	组件重新启动	仅当“资源库目录”值设置为 <REMOTE> 且 BrokerType 的值为 ICS 时此属性才有效。
ListenerConcurrency	1 到 100	1	组件重新启动	仅当 DeliveryTransport 的值为 MQ 时此属性才有效。

表 50. 标准配置属性总结 (续)

属性名	可能的值	缺省值	更新方法	注释
Locale	以下是受支持的语言环境的子集: en_US、ja_JP、 ko_KR、zh_CN、 zh_TW、fr_FR、 de_DE、it_IT、 es_ES 和 pt_BR	en_US	组件重新启动	
LogAtInterchangeEnd	true 或 false	false	组件重新启动	仅当“资源库目录”值设置为 <REMOTE> 且 BrokerType 的值为 ICS 时此属性才有效。
MaxEventCapacity	1 到 2147483647	2147483647	动态的	仅当“资源库目录”值设置为 <REMOTE> 且 BrokerType 的值为 ICS 时此属性才有效。
MessageFileName	有效文件名	InterchangeSystem.txt	组件重新启动	
MonitorQueue	任何有效的队列名	*<CONNECTORNAME>* /MONITORQUEUE	组件重新启动	仅当 DuplicateEventElimination 的值为 true 并且 ContainerManagedEvents 没有任何值时此属性才有效。
OADAutoRestartAgent	true 或 false	false	动态的	仅当“资源库目录”值设置为 <REMOTE> 且 BrokerType 的值为 ICS 时此属性才有效。
OADMaxNumRetry	正整数	1000	动态的	仅当“资源库目录”值设置为 <REMOTE> 且 BrokerType 的值为 ICS 时此属性才有效。
OADRetryTimeInterval	以分钟计的正整数	10	动态的	仅当“资源库目录”值设置为 <REMOTE> 且 BrokerType 的值为 ICS 时此属性才有效。
PollEndTime	HH = 0 到 23 MM = 0 到 59	HH:MM	组件重新启动	
PollFrequency	正整数（以毫秒计）	10000	如果代理程序为 ICS，则是动态的；否则需要进行组件重新启动	
PollQuantity	1 到 500	1	代理程序 重新启动	仅当 ContainerManagedEvents 的值为 JMS 时 此属性才有效。
PollStartTime	HH = 0 到 23 MM = 0 到 59	HH:MM	组件重新启动	
RepositoryDirectory	如果代理程序为 ICS， 则此属性为 <REMOTE>； 否则为任何有效的本地目录。	对于 ICS， 该值设置为 <REMOTE> 对于 WMQI 和 WAS， 该值为 *<ProductDir* \repository	代理程序 重新启动	
RequestQueue	有效的 JMS 队列名	*<CONNECTORNAME>* /REQUESTQUEUE	组件重新启动	仅当 DeliveryTransport 的值为 JMS 时此属性才有效
ResponseQueue	有效的 JMS 队列名	*<CONNECTORNAME>* /RESPONSEQUEUE	组件重新启动	仅当 DeliveryTransport 的值为 JMS 时此属性才有效。
RestartRetryCount	0 到 99	3	如果代理程序为 ICS，则是动态的；否则需要进行组件重新启动	

表 50. 标准配置属性总结 (续)

属性名	可能的值	缺省值	更新方法	注释
RestartRetryInterval	一个以分钟计的值，其范围是从 1 到 2147483647	1	如果代理程序为 ICS，则是动态的；否则需要进行组件重新启动	
ResultsSetEnabled	true 或 false	false	组件重新启动	仅供支持 DB2II 的连接器使用。 仅当 DeliveryTransport 的值为 JMS 且 BrokerType 的值为 WMQI 时此属性才有效。
ResultsSetSize	正整数	0（表示对结果集大小没有限制）	组件重新启动	仅供支持 DB2II 的连接器使用。 仅当 ResultsSetEnabled 的值为 true 时此属性才有效。
RHF2MessageDomain	mrm 或 xml	mrm	组件重新启动	仅当 DeliveryTransport 的值为 JMS 且 WireFormat 的值为 CwXML 时此属性才有效。
SourceQueue	任何有效的 WebSphere MQ 队列名	*<CONNECTORNAME>*/SOURCEQUEUE	代理程序重新启动	仅当 ContainerManagedEvents 的值为 JMS 时此属性才有效。
SynchronousRequest Queue	任何有效的队列名。	*<CONNECTORNAME>*/SYNCHRONOUSREQUEST QUEUE	组件重新启动	仅当 DeliveryTransport 的值为 JMS 时此属性才有效。
SynchronousRequest Timeout	0 到任意数目（以毫秒计）	0	组件重新启动	仅当 DeliveryTransport 的值为 JMS 时此属性才有效。
SynchronousResponse Queue	任何有效的队列名	*<CONNECTORNAME>*/SYNCHRONOUSRESPONSE QUEUE	组件重新启动	仅当 DeliveryTransport 的值为 JMS 时此属性才有效。
TivoliMonitorTransaction Performance	true 或 false	false	组件重新启动	
WireFormat	CwXML 或 CwBO	CwXML	代理程序重新启动	如果 RepositoryDirectory 的值不是设置为 <REMOTE>，则此属性的值必须为 CwXML。如果 RepositoryDirectory 的值设置为 <REMOTE>，则该值必须为 CwBO。
WsifSynchronousRequest Timeout	0 到任意数目（以毫秒计）	0	组件重新启动	仅当 BrokerType 的值为 WAS 时此属性才有效。
XMLNameSpaceFormat	short 或 long	short	代理程序重新启动	仅当 BrokerType 的值为 WMQI 或 WAS 时此属性才有效

标准属性

本节描述标准连接器配置属性。

AdapterHelpName

AdapterHelpName 属性是特定于连接器的扩展帮助文件所在的目录的名称。该目录必须位于 *<ProductDir>*\bin\Data\App\Help 中，并且必须至少包含语言目录 enu_usa。根据不同的语言环境，它可能包含其它目录。

如果它有效，则缺省值为模板名，否则为空白。

AdminInQueue

AdminInQueue 属性指定集成代理程序用来将管理消息发送至连接器的队列。

缺省值为 *<CONNECTORNAME>*/ADMININQUEUE

AdminOutQueue

AdminOutQueue 属性指定连接器用来将管理消息发送至集成代理程序的队列。

缺省值为 *<CONNECTORNAME>*/ADMINOUTQUEUE

AgentConnections

AgentConnections 属性控制当 ORB 初始化时打开的 ORB（对象请求代理程序）连接数。

仅当 RepositoryDirectory 的值设置为 <REMOTE> 并且 DeliveryTransport 属性的值为 MQ 或 IDL 时该属性才有效。

此属性的缺省值为 1。

AgentTraceLevel

AgentTraceLevel 属性设置特定于应用程序的组件的跟踪消息级别。连接器传递在设置的跟踪级别和更低级别适用的所有跟踪消息。

缺省值为 0。

ApplicationName

ApplicationName 属性唯一地标识连接器应用程序的名称。系统管理员使用此名称来监视集成环境。此属性必须具有值，然后您才能运行连接器。

缺省值为连接器的名称。

BiDi.Application

BiDi.Application 属性指定数据的双向格式是否以此适配器支持的业务对象形式从外部应用程序传送到适配器中。该属性定义应用程序数据的双向属性。这些属性有:

- 文本类型: 隐藏或可视（I 或 V）
- 文本方向: 从左到右或从右到左（L 或 R）
- 对称交换: 打开或关闭（Y 或 N）
- 塑形（阿拉伯语）: 打开或关闭（S 或 N）
- 数字塑形（阿拉伯语）: 印地语、上下文或名词性（H、C 或 N）

仅当 BiDi.Transformation 属性值设置为 true 时此属性才有效。

缺省值为 ILYNN（隐藏、从左到右、打开、关闭和名词性）。

BiDi.Broker

BiDi.Broker 属性指定数据的双向格式是发送至集成代理程序还是从集成代理程序接收。它定义代理程序数据的双向属性，BiDi.Application 中列示了这些属性。

仅当 BiDi.Transformation 属性值设置为 true 时此属性才有效。如果 BrokerType 属性为 ICS，则该属性值是只读的。

缺省值为 ILYNN（隐藏、从左到右、打开、关闭和名词性）。

BiDi.Metadata

BiDi.Metadata 属性定义元数据的双向格式或属性，BiDi.Application 中列示了这些双向格式或属性。

仅当 BiDi.Transformation 属性值设置为 true 时此属性才有效。

缺省值为 ILYNN（隐藏、从左到右、打开、关闭和名词性）。

BiDi.Transformation

BiDi.Transformation 属性定义系统在运行时是否执行双向转换。

如果属性值设置为 true，则 BiDi.Application、 BiDi.Broker 和 BiDi.Metadata 属性是可用的。如果属性值设置为 false，则会隐藏它们。

缺省值为 false。

BrokerType

BrokerType 属性标识您正在使用的集成代理程序类型。可能的值是 ICS、WMQI（适用于 WMQI、WMQIB 或 WBIMB） 或 WAS。

CharacterEncoding

CharacterEncoding 属性指定用来从字符（如字母表的字母、数字表示或标点符号）映射至数字值的字符代码集。

注: 基于 Java 的连接器不使用此属性。C++ 连接器对此属性使用值 ascii7。

缺省情况下，将只显示受支持的字符编码的子集。要将其它受支持的值添加至列表，则您必须手工修改产品目录（*<ProductDir>*）中的 \Data\Std\stdConnProps.xml 文件。有关更多信息，请参阅本指南中的『连接器配置器』附录。

CommonEventInfrastructure

Common Event Infrastructure（CEI）是一项简单事件管理功能，用来处理生成的事件。CommonEventInfrastructure 属性指定在运行时是否应该调用 CEI。

缺省值为 false。

CommonEventInfrastructureContextURL

CommonEventInfrastructureContextURL 用来获得对执行 Common Event Infrastructure（CEI）服务器应用程序的 WAS 服务器的访问权。此属性指定要使用的 URL。

仅当 CommonEventInfrastructure 的值设置为 true 时此属性才有效。

缺省值为空白字段。

ConcurrentEventTriggeredFlows

ConcurrentEventTriggeredFlows 属性确定连接器可以同时处理多少个业务对象以进行事件传递。将此属性的值设置为同时映射和传递的业务对象数。例如，如果将此属性的值设置为 5，则会同时处理 5 个业务对象。

如果将此属性设置为大于 1 的值，则允许源应用程序的连接器同时映射多个事件业务对象并将这些对象同时传递至多个协作实例。这样就可提高将业务对象传递至集成代理程序的速度，尤其是当业务对象使用复杂映射时效果更好。增加业务对象至协作的到达速率可以改进系统中的整体性能。

要对整个流程（从源应用程序至目标应用程序）实现并发处理，必须配置下列属性:

- 必须通过将协作的最大并发事件数属性设置为足够大以使用多个线程，来将协作配置为使用多个线程。
- 必须将目标应用程序的特定于应用程序的组件配置为同时处理请求。即，它必须是多线程的，或者它必须能够使用连接器代理程序并行性且可以对多个进程进行配置。必须将“并行处理度”配置属性设置为大于 1 的值。

ConcurrentEventTriggeredFlows 属性对连接器轮询无效，连接器轮询是单线程的并且是连续地执行。

仅当 RepositoryDirectory 属性的值设置为 <REMOTE> 时此属性才有效。

缺省值为 1。

ContainerManagedEvents

ContainerManagedEvents 属性允许 JMS 支持连接器在带有 JMS 事件库的情况下提供有保证的事件传递，在此过程中，将把事件从源队列中除去并将它作为一个 JMS 事务放置在目标队列上。

当此属性设置为 JMS 时，还必须设置下列属性以启用有保证的事件传递:

- PollQuantity = 1 到 500
- SourceQueue = /SOURCEQUEUE

您还必须使用 MimeType 和 DHClass（数据处理程序类）属性来配置数据处理程序。还可以添加 DataHandlerConfigMOName（元对象名，这是可选的）。要设置这些值，使用连接器配置器中的**数据处理程序**选项卡。

尽管这些属性是特定于适配器的，但以下还是提供一些示例值:

- MimeType = text\xml

- DHClass = com.crossworlds.DataHandlers.text.xml
- DataHandlerConfigMOName = MO_DataHandler_Default

仅当已将 ContainerManagedEvents 属性设置为值 JMS 时，**数据处理程序**选项卡中这些值的字段才会显示。

注: 当 ContainerManagedEvents 设置为 JMS 时，连接器不会调用它的 pollForEvents() 方法，从而禁用该方法的功能。

仅当 DeliveryTransport 属性的值设置为 JMS 时 ContainerManagedEvents 属性才有效。

没有缺省值。

ControllerEventSequencing

ControllerEventSequencing 属性在连接器控制器中启用事件排序。

仅当 RepositoryDirectory 属性的值设置为 <REMOTE>（BrokerType 为 ICS）时此属性才有效。

缺省值为 true。

ControllerStoreAndForwardMode

ControllerStoreAndForwardMode 属性设置连接器控制器在检测到特定于应用程序的目标组件不可用之后的行为。

如果此属性设置为 true，而特定于应用程序的目标组件不可用，则当事件到达 ICS 时，连接器控制器阻塞对该特定于应用程序的组件的请求。当特定于应用程序的组件变为可用时，控制器将该请求转发给它。

但是，如果目标应用程序的特定于应用程序的组件在连接器控制器将服务调用请求转发给它之后变为不可用，则连接器控制器将废弃该请求。

如果此属性设置为 false，则连接器控制器只要检测到特定于应用程序的目标组件不可用，就开始废弃所有服务调用请求。

仅当 RepositoryDirectory 属性的值设置为 <REMOTE>（BrokerType 属性的值为 ICS）时此属性才有效。

缺省值为 true。

ControllerTraceLevel

ControllerTraceLevel 属性设置连接器控制器的跟踪消息的级别。

仅当 RepositoryDirectory 属性的值设置为 <REMOTE> 时此属性才有效。

缺省值为 0。

DeliveryQueue

DeliveryQueue 属性定义连接器用来将业务对象发送至集成代理程序的队列。

仅当 DeliveryTransport 属性的值设置为 JMS 时此属性才有效。

缺省值为 *<CONNECTORNAME>*/DELIVERYQUEUE。

DeliveryTransport

DeliveryTransport 指定用于传递事件的传输机制。可能的值为 MQ（对于 WebSphere MQ）、IDL（对于 CORBA IIOP）或 JMS（对于 java 消息服务）。

- 如果 RepositoryDirectory 属性的值设置为 <REMOTE>，则 DeliveryTransport 属性的值可以为 MQ、IDL 或 JMS，缺省值为 IDL。
- 如果 RepositoryDirectory 属性的值是本地目录，则该值只能为 JMS。

如果 RepositoryDirectory 属性的值为 MQ 或 IDL，则连接器通过 CORBA IIOP 发送服务调用请求和管理消息。

缺省值为 JMS。

WebSphere MQ 和 IDL

除非您必须只具有一个产品，否则使用 WebSphere MQ 而不是 IDL 来作为事件交付传输。与 IDL 相比，WebSphere MQ 具有以下优点:

- 异步通信:
  WebSphere MQ 允许特定于应用程序的组件轮询并持久地存储事件，即使当服务器不可用时也是如此。
- 服务器端性能:
  WebSphere MQ 在服务器端提供更快的性能。在优化方式下，WebSphere MQ 仅将指向事件的指针存储在资源库数据库中，而实际事件则仍保留在 WebSphere MQ 队列中。这样就防止了将可能很大的事件写入资源库数据库。
- 代理程序端性能:
  WebSphere MQ 在特定于应用程序的组件端提供更快的性能。通过使用 WebSphere MQ，连接器轮询线程选取事件，将该事件放置在连接器队列中，然后选取下一个事件。它比 IDL 更快，IDL 执行的过程较复杂，即要求连接器轮询线程选取事件，通过网络转至服务器进程，将该事件持久地存储在资源库数据库中，然后选取下一个事件。

JMS

JMS 传输机制使用 java 消息服务（JMS）来启用连接器与客户机连接器框架之间的通信。

如果选择 JMS 作为交付传输，则诸如 jms.MessageBrokerName、jms.FactoryClassName、jms.Password 和 jms.UserName 的其它 JMS 属性将列示在连接器配置器中。jms.MessageBrokerName 和 jms.FactoryClassName 属性是此传输必需的。

如果您在以下环境中将 JMS 传输机制用于连接器，则可能存在内存限制:

- AIX 5.0

- WebSphere MQ 5.3.0.1
- ICS 是集成代理程序

在此环境中，由于 WebSphere MQ 客户机中使用的内存较多，所以您在启动连接器控制器（在服务器端）和连接器（在客户机端）时可能会遇到困难。如果您的安装使用小于 768MB 的处理堆大小，则设置以下变量和属性:

- 在 CWSharedEnv.sh 脚本中设置 LDR_CNTRL 环境变量。

  此脚本位于产品目录（<*ProductDir*>）下的 \bin 目录中。使用文本编辑器，在 CWSharedEnv.sh 脚本中添加下面这一行作为第一行:

  ```
  export LDR_CNTRL=MAXDATA=0x30000000
  ```

  此行将堆内存用量限制为最大值 768 MB（3 段 * 256 MB）。如果处理内存增长到超过此限制，则会发生页面交换，这对系统的性能有不利影响。
- 将 IPCCBaseAddress 属性的值设置为 11 或 12。有关此属性的更多信息，请参阅《*系统安装指南 UNIX 版*》。

DuplicateEventElimination

当此属性的值为 `true` 时，JMS 支持连接器可以确保不会将重复事件传递至传递队列。要使用此功能，在连接器开发期间，连接器必须已在特定于应用程序的代码中将唯一的事件标识设置为业务对象 ObjectEventId 属性。

注: 当此属性的值为 `true` 时，必须启用 MonitorQueue 属性以提供有保证的事件传递。

缺省值是 `false`。

EnableOidForFlowMonitoring

当此属性的值为 `true` 时，适配器运行时将把传入 ObjectEventID 标记为流监视的外键。

仅当 BrokerType 属性设置为 ICS 时此属性才有效。

缺省值是 `false`。

FaultQueue

如果连接器在处理消息时遇到错误，则它会将该消息（以及状态指示符和对问题的描述）移至在 FaultQueue 属性中指定的队列。

缺省值为 *<CONNECTORNAME>*`/FAULTQUEUE`。

jms.FactoryClassName

jms.FactoryClassName 属性指定要对 JMS 提供程序进行实例化的类名。如果 DeliveryTransport 属性的值为 `JMS`，则必须设置此属性。

缺省值为 `CxCommon.Messaging.jms.IBMMQSeriesFactory`。

jms.ListenerConcurrency

jms.ListenerConcurrency 属性指定 JMS 控制器的并发侦听器的数目。它指定控制器中同时访存和处理消息的线程数。

仅当 jms.OptimizedTransport 属性的值为 `true` 时此属性才有效。

缺省值为 `1`。

jms.MessageBrokerName

jms.MessageBrokerName 指定要用于 JMS 提供程序的代理程序名。如果您（在 DeliveryTransport 属性中）指定 `JMS` 作为交付传输机制，则必须设置此连接器属性。

当连接至远程消息代理时，此属性需要下列值:
`QueueMgrName``:`*`Channel`*`:`*`HostName`*`:`*`PortNumber`*
其中:
`QueueMgrName` 是队列管理器的名称。
`Channel` 是客户机使用的通道。
`HostName` 是队列管理器要驻留的机器的名称。
`PortNumber` 是队列管理器进行侦听将使用的端口号。

例如:

```
jms.MessageBrokerName = WBIMB.Queue.Manager:CHANNEL1:RemoteMachine:1456
```

缺省值为 `crossworlds.queue.manager`。当连接至本地消息代理时使用缺省值。

jms.NumConcurrentRequests

jms.NumConcurrentRequests 属性指定可以同时发送至连接器的最大并发服务调用请求数。一旦达到该最大值，新的服务调用将被阻塞，并且必须等待另一个请求完成之后才能继续。

缺省值为 `10`。

jms.Password

jms.Password 属性指定 JMS 提供程序的密码。此属性的值是可选的。

没有缺省值。

jms.TransportOptimized

jms.TransportOptimized 属性确定是否优化了 WIP（进行中的工作）。必须具有 WebSphere MQ 提供程序来优化 WIP。要使优化后的 WIP 能够运行，消息传递提供程序必须能够:

1. 读取消息而不将它移出队列
2. 删除具有特定标识的消息而不将整个消息传送至接收方的内存空间
3. 使用特定标识（需要使用它来进行恢复）来读取消息
4. 跟踪尚未读取的事件的出现位置。

不能将 JMS API 用于经过优化的 WIP，这是因为它们不满足上述的第 2 和第 4 个条件，但是 MQ Java API 满足所有这四个条件，因此，经过优化的 WIP 必须使用 MQ Java API。

仅当 DeliveryTransport 的值为 JMS 并且 BrokerType 的值为 ICS 时此属性才有效。

缺省值为 false。

jms.UserName

jms.UserName 属性指定 JMS 提供程序的用户名。此属性的值是可选的。

没有缺省值。

JvmMaxHeapSize

JvmMaxHeapSize 属性指定代理程序的最大堆大小（以兆字节计）。

仅当 RepositoryDirectory 属性的值设置为 <REMOTE> 时此属性才有效。

缺省值为 128M。

JvmMaxNativeStackSize

JvmMaxNativeStackSize 属性指定代理程序的最大本机堆栈大小（以千字节计）。

仅当 RepositoryDirectory 属性的值设置为 <REMOTE> 时此属性才有效。

缺省值为 128K。

JvmMinHeapSize

JvmMinHeapSize 属性指定代理程序的最小堆大小（以兆字节计）。

仅当 RepositoryDirectory 属性的值设置为 <REMOTE> 时此属性才有效。

缺省值为 1M。

ListenerConcurrency

当 ICS 是集成代理程序时，ListenerConcurrency 属性支持 WebSphere MQ 侦听器中的多线程。它允许将多个事件以批处理方式写入数据库，因此提高了系统性能。

此属性仅对于使用 MQ 传输的连接器有效。DeliveryTransport 属性的值必须为 MQ。

缺省值为 1。

Locale

Locale 属性指定语言代码、国家或地区和（可选）相关联的字符代码集。此属性的值确定诸如数据的整理和排序顺序、日期和时间格式以及货币规范中使用的符号等文化约定。

语言环境名称具有以下格式:

ll_TT.codeset

其中:
ll 是两个字符的语言代码（采用小写字母）
TT 是两个字母的国家或地区代码（采用大写字母）
codeset 是相关联的字符代码集的名称（这一部分是可选的）。

缺省情况下，只列示了受支持的语言环境的子集。要将其它受支持的值添加至列表，应修改 <*ProductDir*>\bin 目录中的 \Data\Std\stdConnProps.xml 文件。有关更多信息，参阅本指南中的『连接器配置器』附录。

如果连接器尚未国际化，则此属性唯一的有效值为 en_US。要确定特定连接器是否已全球化，请参阅用户指南中的那一章。

缺省值为 en_US。

LogAtInterchangeEnd

LogAtInterchangeEnd 属性指定是否将错误记录到集成代理程序的日志目标位置。

记录到日志目标位置还会打开电子邮件通知，当发生错误或致命错误时，电子邮件通知将为在 InterchangeSystem.cfg 文件中指定为 MESSAGE_RECIPIENT 值的收件人生成电子邮件消息。例如，当连接器丢失与应用程序的连接时，如果 LogAtInterChangeEnd 的值为 true，则会将一条电子邮件消息发送至指定的消息收件人。

仅当 RespositoryDirectory 属性的值设置为 <REMOTE>（BrokerType 的值为 ICS）时此属性才有效。

缺省值是 false。

MaxEventCapacity

MaxEventCapacity 属性指定控制器缓冲区中的最大事件数。此属性供流量控制功能部件使用。

仅当 RespositoryDirectory 属性的值设置为 <REMOTE>（BrokerType 的值为 ICS）时此属性才有效。

该值可以是 1 和 2147483647 之间的正整数。

缺省值为 2147483647。

MessageFileName

MessageFileName 属性指定连接器消息文件的名称。消息文件的标准位置是产品目录中的 \connectors\messages。如果消息文件不是位于标准位置，则以绝对路径指定消息文件名。

如果连接器消息文件不存在，则连接器使用 InterchangeSystem.txt 作为消息文件。此文件位于产品目录中。

注: 要确定连接器是否具有它自己的消息文件，请参阅各个适配器用户指南。

缺省值为 InterchangeSystem.txt。

MonitorQueue

MonitorQueue 属性指定连接器用来监视重复事件的逻辑队列。

仅当 DeliveryTransport 属性的值为 JMS 并且 DuplicateEventElimination 的值为 true 时此属性才有效。

缺省值为 *<CONNECTORNAME>*/MONITORQUEUE

OADAutoRestartAgent

OADAutoRestartAgent 属性指定连接器是否使用自动和远程重新启动功能。此功能使用 WebSphere MQ 触发的 Object Activation Daemon（OAD）来在异常关闭之后重新启动连接器或者从系统监视器启动远程连接器。

要启用自动和远程重新启动功能，此属性必须设置为 true。有关如何配置 WebSphere MQ 触发的 OAD 功能部件的信息，请参阅《*安装指南 Windows 版*》或《*安装指南 UNIX 版*》。

仅当 RespositoryDirectory 属性的值设置为 <REMOTE>（BrokerType 的值为 ICS）时此属性才有效。

缺省值为 false。

OADMaxNumRetry

OADMaxNumRetry 属性指定 WebSphere MQ 触发的 Object Activation Daemon（OAD）在异常关闭之后自动尝试重新启动连接器的最大次数。OADAutoRestartAgent 属性必须设置为 true，此属性才能生效。

仅当 RespositoryDirectory 属性的值设置为 <REMOTE>（BrokerType 的值为 ICS）时此属性才有效。

缺省值为 1000。

OADRetryTimeInterval

OADRetryTimeInterval 属性指定 WebSphere MQ 触发的 Object Activation Daemon（OAD）的重试时间间隔中的分钟数。如果连接器代理程序未在此重试时间间隔内重新启动，则连接器控制器要求 OAD 再次重新启动连接器代理程序。OAD 重复此重试过程的次数由 OADMaxNumRetry 属性指定。OADAutoRestartAgent 属性必须设置为 true，此属性才能生效。

仅当 RespositoryDirectory 属性的值设置为 <REMOTE>（BrokerType 的值为 ICS）时此属性才有效。

缺省值为 10。

PollEndTime

PollEndTime 属性指定停止轮询事件队列的时间。格式为 *HH:MM*，其中 *HH* 为 0 到 23 小时，*MM* 表示 0 到 59 分钟。

您必须为此属性提供一个有效值。缺省值为不带值的 HH:MM，您必须更改它。

如果适配器运行时检测到:

- 设置了 PollStartTime 而未设置 PollEndTime，或者
- 设置了 PollEndTime 而未设置 PollStartTime

它将使用为 PollFrequency 属性配置的值来进行轮询。

PollFrequency

PollFrequency 属性指定一个轮询操作的结束与下一个轮询操作的开始之间的时间量（以毫秒计）。它不是轮询操作之间的时间间隔。该逻辑而是如下所示:

- 进行轮询以获取由 PollQuantity 属性的值指定的对象数。
- 处理这些对象。对于某些连接器，此操作可能分成几部分在一些单独的线程上来执行，这些线程以异步方式执行到下一个轮询操作为止。
- 延迟由 PollFrequency 属性指定的时间间隔。
- 重复该循环。

此属性具有下列有效值:

- 轮询操作之间的毫秒数（正整数）。
- 单词 no，它使连接器不执行轮询。以小写输入该单词。
- 单词 key，它使连接器仅当您在连接器的“命令提示符”窗口中输入字母 p 时才执行轮询。以小写输入该单词。

缺省值为 10000。

重要提示: 某些连接器对于使用此属性存在限制。在存在这些限制时，它们记录在关于安装和配置适配器的章节中。

PollQuantity

PollQuantity 属性指定连接器从应用程序中轮询的项数。如果适配器具有一个用于设置轮询数量的特定于连接器的属性，则在该特定于连接器的属性中设置的值将覆盖标准属性值。

仅当 DeliveryTransport 属性的值为 JMS 并且 ContainerManagedEvents 属性具有值时此属性才有效。

电子邮件消息也被认为是事件。当轮询连接器以获取电子邮件时，连接器操作如下所示。

- 当第一次轮询时，连接器将检测消息的正文（称为附件）。因为没有为此 MIME 类型指定数据处理程序，所以它将忽略消息。
- 连接器处理第一个 BO 附件。数据处理程序可用于此 MIME 类型，因此，它将业务对象发送至可视测试连接器。
- 当第二次轮询它时，连接器处理第二个 BO 附件。数据处理程序可用于此 MIME 类型，因此，它将业务对象发送至可视测试连接器。
- 一旦接收了第三个 BO 附件，则应该传输它。

PollStartTime

PollStartTime 属性指定开始轮询事件队列的时间。格式为 *HH:MM*，其中 *HH* 为 0 到 23 小时，*MM* 表示 0 到 59 分钟。

您必须为此属性提供一个有效值。缺省值为不带值的 HH:MM，您必须更改它。

如果适配器运行时检测到:

- 设置了 PollStartTime 而未设置 PollEndTime，或者
- 设置了 PollEndTime 而未设置 PollStartTime

它将使用为 PollFrequency 属性配置的值来进行轮询。

RepositoryDirectory

RepositoryDirectory 属性是资源库的位置，连接器从该资源库读取存储业务对象定义的元数据的 XML 模式文档。

如果集成代理程序是 ICS，则由于连接器从 InterChange Server 资源库获得此信息，所以此值必须设置为 <REMOTE>。

当集成代理程序是 WebSphere Message Broker 或 WAS 时，缺省情况下，此值设置为 <*ProductDir*>\repository。但是，可以将它设置为任何有效的目录名。

RequestQueue

RequestQueue 属性指定集成代理程序用来将业务对象发送至连接器的队列。

仅当 DeliveryTransport 属性的值为 JMS 时此属性才有效。

缺省值为 *<CONNECTORNAME>*/REQUESTQUEUE。

ResponseQueue

ResponseQueue 属性指定 JMS 响应队列，该队列将来自连接器框架的响应消息传递至集成代理程序。当集成代理程序是 ICS 时，服务器发送请求并等待 JMS 响应队列中的响应消息。

仅当 DeliveryTransport 属性的值为 JMS 时此属性才有效。

缺省值为 *<CONNECTORNAME>*/RESPONSEQUEUE。

RestartRetryCount

RestartRetryCount 属性指定连接器尝试重新启动它自己的次数。当将此属性用于并行连接的连接器时，它指定特定于应用程序的主连接器组件尝试重新启动特定于应用程序的客户机连接器组件的次数。

缺省值为 3。

RestartRetryInterval

RestartRetryInterval 属性指定连接器尝试重新启动它自己的时间间隔（以分钟计）。当将此属性用于并行链接的连接器时，它指定特定于应用程序的主连接器组件尝试重新启动特定于应用程序的客户机连接器组件的时间间隔。

该属性的可能值的范围是从 1 到 2147483647。

缺省值为 1。

ResultsSetEnabled

当 Information Integrator 活动时，ResultsSetEnabled 属性启用或禁用结果集支持。仅当适配器支持 DB2 Information Integrator 时才可以使用此属性。

仅当 DeliveryTransport 属性的值为 JMS 并且 BrokerType 的值为 WMQI 时此属性才有效。

缺省值为 false。

ResultsSetSize

ResultsSetSize 属性定义可以返回到 Information Integrator 的业务对象的最大数目。仅当适配器支持 DB2 Information Integrator 时才可以使用此属性。

仅当 ResultsSetEnabled 属性的值为 true 时此属性才有效。

缺省值为 0。这表示结果集的大小不受限制。

RHF2MessageDomain

RHF2MessageDomain 属性允许您在 JMS 头中配置字段域名的值。当将数据通过 JMS 传送包发送至 WebSphere Message Broker 时，适配器框架写入 JMS 头信息以及域名和固定值 mrm。可配置的域名允许您跟踪 WebSphere Message Broker 处理消息数据的方式。

以下是一个示例头:

```
<mcd><Msd>mrm</Msd><Set>3</Set><Type>
Retek_POPhyDesc</Type><Fmt>CwXML</Fmt></mcd>
```

仅当 BrokerType 的值为 WMQI 或 WAS 时此属性才有效。另外，仅当 DeliveryTransport 属性的值为 JMS 并且 WireFormat 属性的值为 CwXML 时此属性才有效。

可能的值为 mrm 和 xml。缺省值为 mrm。

SourceQueue

SourceQueue 属性为连接器框架指定 JMS 源队列，以便为使用 JMS 事件库的 JMS 支持连接器提供有保证的事件传递。有关进一步的信息，请参阅第 270 页的『ContainerManagedEvents』。

仅当 DeliveryTransport 的值为 JMS 并且为 ContainerManagedEvents 指定了值时此属性才有效。

缺省值为 *<CONNECTORNAME>*/SOURCEQUEUE。

SynchronousRequestQueue

SynchronousRequestQueue 属性将需要来自连接器框架同步响应的请求消息传递至代理程序。仅当此连接器使用同步执行时，此队列才是必需的。使用同步执行时，连接器框架将消息发送至同步请求队列并等待来自同步响应队列上代理程序的响应。发送至连接器的响应消息具有与原始消息的标识相匹配的相关标识。

仅当 DeliveryTransport 的值为 JMS 时此属性才有效。

缺省值为 *<CONNECTORNAME>*/SYNCHRONOUSREQUESTQUEUE

SynchronousRequestTimeout

SynchronousRequestTimeout 属性指定连接器等待对同步请求的响应的时间（以毫秒计）。如果在指定的时间内未接收到响应，则连接器将原始同步请求消息（以及错误消息）移至故障队列。

仅当 DeliveryTransport 的值为 JMS 时此属性才有效。

缺省值为 0。

SynchronousResponseQueue

SynchronousResponseQueue 属性将代理程序用于应答同步请求的响应消息传递至连接器框架。仅当此连接器使用同步执行时，此队列才是必需的。

仅当 DeliveryTransport 的值为 JMS 时此属性才有效。

缺省值为 *<CONNECTORNAME>*/SYNCHRONOUSRESPONSEQUEUE

TivoliMonitorTransactionPerformance

TivoliMonitorTransactionPerformance 属性指定在运行时是否调用 IBM Tivoli Monitoring for Transaction Performance（ITMTP）。

缺省值为 false。

WireFormat

WireFormat 属性指定传输上的消息格式:

- 如果 RepositoryDirectory 属性的值是本地目录，则该值为 CwXML。
- 如果 RepositoryDirectory 属性的值是远程目录，则该值为 CwBO。

WsifSynchronousRequestTimeout

WsifSynchronousRequestTimeout 属性指定连接器等待对同步请求的响应的时间（以毫秒计）。如果在指定的时间内未接收到响应，则连接器将原始同步请求消息（以及错误消息）移至故障队列。

仅当 BrokerType 的值为 WAS 时此属性才有效。

缺省值为 0。

XMLNameSpaceFormat

XMLNameSpaceFormat 属性指定采用 XML 格式的业务对象定义的短名称空间或长名称空间。

仅当 BrokerType 的值设置为 WMQI 或 WAS 时此属性才有效。

缺省值为 `short`。

附录 E. 特定于连接器的配置属性

连接器具有两种配置属性：特定于连接器的配置属性和标准配置属性。本附录描述特定于 mySAP.com（SAP R/3 V3.x）连接器的属性，有关使用连接器配置器的信息，请参阅第 21 页的第 3 章，『配置连接器』。

标准配置属性提供所有连接器都使用的信息。请参阅第 261 页的附录 D，『连接器的标准配置属性』以获取这些属性的文档。注意，有几个标准配置属性具有 SAP 连接器所独有的问题，如表 51 中所述。

表 51. 特定于此连接器的属性信息

属性	注意事项
CharacterEncoding	此连接器不使用此属性。
Locale	因为此连接器已国际化，所以您可以更改此属性的值。请参阅适配器的发行说明以确定当前受支持的语言环境。
PollFrequency	如果使用 RFC 服务器模块或 ALE 模块进行事件处理，则不要将此属性的值设置为 key 或 no。将此值设置为 key 或 no 会阻止连接器在启动时实例化这些模块。

在运行连接器之前，您必须为 ApplicationName 配置属性提供值。

特定于连接器的配置属性

特定于连接器的配置属性提供连接器在运行时需要的信息。特定于连接器的属性还提供了一种方式，更改连接器框架和特定于应用程序的连接器组件内静态信息或逻辑而不必重新编码和重新构建连接器。

表 52 是特定于连接器的配置属性的快速参考。模块列包含使用相关属性的连接器模块的列表。

表 52. 特定于连接器的配置属性的快速参考

名称	可能的值	缺省值	模块
ABAPDebug	true 或 false	false	ABAP 扩展 BAPI HDR
AleEventDir	*path*		ALE
AleUpdateStatus	true 或 false	false	ALE
AleSelectiveUpdate	*IDocType:MessageType*		ALE
AleStatusMsgCode	*消息代码*		ALE
AleSuccessCode	52 或 53	52	ALE
AleFailureCode	68 或 58	68	ALE
AleSuccessText	*成功文本*		ALE
AleFailureText	*失败文本*		ALE
ApplicationPassword		SOFTWARE	全部
ApplicationUserName		CROSSWORLDS	全部

表 52. 特定于连接器的配置属性的快速参考 (续)

名称	可能的值	缺省值	模块
ArchiveDays			ALE
Client			全部
Group	*表示一组应用程序服务器的登录组的任何有效名称*		全部
gwService	*网关服务器标识*	sapgw00	RFC 服务器 ALE
Hostname	IP 地址或*服务器名称*		全部
Language		E	全部
MaxNumberOfConnections		2	ABAP 扩展、 ALE（仅限于请求处理）和 BAPI HDR
Modules	*模块名称*		全部
Namespace	true 或 false	true	ABAP 扩展
NumberOfListeners	*任何正整数*	1	RFC 服务器和 ALE
PollQuantity	*任何正整数*	20	ABAP 扩展和 ALE
RefreshLogonCycle	true	true	全部
RfcProgramId	*程序标识*	CWLDSERVER	RFC 服务器和 ALE
RfcTraceOn	true 或 false	false	全部
SAPALE_Archive_Queue	*任何有效的 MQ Series 队列名*		ALE
SAPALE_Event_Queue	*任何有效的 MQ Series 队列名*		ALE
SAPALE_Wip_Queue	*任何有效的 MQ Series 队列名*		ALE
SAPALE_Error_Queue			
SAPALE_Unsubscribed_Queue			
SAPSystemID	*SAP R/3 系统的逻辑名称*		全部
SAPtid_MQChannel	*任何有效的 MQ 通道*		ALE
SAPtid_MQPort	*任何有效的 MQ 端口*		ALE
SAPtid_Queue	*任何有效的 MQ 队列名*		ALE（仅限于请求处理）
SAPtid_QueueManager	*任何有效的 MQ 队列管理器名*		ALE
SAPtid_QueueManagerHost	*任何有效的 MQ 队列管理器主机名*		ALE
SAPtid_QueueManagerLogin	*任何有效的 MQ 队列管理器登录名*		ALE
SAPtid_QueueManagerPassword	*任何有效的 MQ 队列管理器密码*		ALE
Sysnr	*系统号*	00	全部
DateTimeFormat	*无或 legacy*		全部
TransIdCollabName			**不再受支持**

表 52. 特定于连接器的配置属性的快速参考 (续)

名称	可能的值	缺省值	模块
UpdateIDocStatus	true 或 false	True	ALE
IDocSuccessCode	12		ALE
IDocFailureCode	11		ALE
IDocSuccessText	分派成功		ALE
IDocFailureText	分派失败		ALE
UseDefaults	true 或 false	false	ABAP 扩展 ALE BAPI

ABAPDebug

指定连接器在开始处理业务对象时是否对适当的功能模块调用 ABAP 调试器。当此属性设置为 true 时，连接器对以下连接器模块打开 ABAP 调试器:

- ABAP 扩展 – 当处理 SAP 外部的事件和 SAP 内部的服务调用请求时
- BAPI – 仅当处理 SAP 内部的服务调用请求时
- 分层动态检索 – 当处理 SAP 内部的服务调用请求时

仅当您已执行以下操作时，连接器才调用 ABAP 调试器:

- 将**第 287 页的『ApplicationUserName』** 配置属性的值从 CROSSWORLDS 更改为具有适当用户权限的对话用户。
- 将 ABAPDebug 属性设置为 true。

注: 您只能在打开调试器之后添加断点。

重要提示: 在生产环境中，此属性应始终设置为 false。

缺省值是 false。

AleEventDir

指定由 ALE 模块用来将事件写入日志和恢复事件的 event 目录的根目录（\ale）位置。当连接器首次启动时，如果它在启动连接器的目录中未找到该根目录，则它会创建该目录和 event 子目录:

- 如果在此属性中指定了路径，则它使用该路径来创建目录。
- 如果未指定路径，则它在启动连接器的目录中创建该根目录。

例如，如果连接器位于 \connectors\SapConnector1（在产品目录中），则连接器创建以下目录:

```
\connectors\SapConnector1\ale
```

UNIX

如果您首次启动连接器时未在连接器的目录中，则连接器在您启动连接器的目录中创建该根目录，而不考虑此属性的值。

有关更多信息，请参阅第 109 页的第 10 章，『ALE 模块概述』。

缺省值为:

UNIX

$<ProductNameDir>/connectors/SAP/ale

Windows

%ProductNameDir%\connectors\SAP\ale

AleUpdateStatus

指定审计跟踪是否对于所有消息类型都是必需的。要使连接器在 ALE 模块已检索 IDoc 对象进行事件处理之后更新标准 SAP 状态码，必须将此属性设置为 `true`。

有关更多信息，请参阅第 109 页的第 10 章，『ALE 模块概述』。

缺省值是 `false`。

AleSelectiveUpdate

指定当配置连接器以更新标准 SAP 状态码时要更新哪些 IDocType 和 MessageType 组合。仅当 `AleUpdateStatus` 已设置为 `true` 时您才能为此属性定义值。

此属性的语法为:

`IDocType:MessageType``[,`*`IDocType:MessageType`*` [,...]]`

其中冒号（`:`）定界符分隔每个 IDocType 和 MessageType，逗号（`,`）定界符分隔集合中的条目。以下示例举例说明了两个集合。在该示例中，`MATMAS03` 和 `DEBMAS03` 是 IDoc，`MATMAS` 和 `DEBMAS` 是消息类型:

```
MATMAS03:MATMAS,DEBMAS03:DEBMAS
```

有关更多信息，请参阅第 109 页的第 10 章，『ALE 模块概述』。

AleStatusMsgCode

如果需要，则指定当连接器记入 `ALEAUD` 消息 IDoc（`ALEAUD01`）时要使用的消息代码。在接收伙伴概要文件中配置此消息代码。仅当 `AleUpdateStatus` 已设置为 `true` 时您才能为此属性定义值。

有关更多信息，请参阅第 117 页的『配置 SAP 以更新 IDoc 状态』。

AleSuccessCode

指定“已记入应用程序文档”的成功状态码。要使连接器在 ALE 模块已检索 IDoc 对象进行事件处理之后更新 SAP 成功状态码，您必须为此属性指定值（`52` 或 `53`）。SAP 将此值转换为状态 `41`（在接收系统中已创建应用程序文件）。

有关更多信息，请参阅第 109 页的第 10 章，『ALE 模块概述』。

AleFailureCode

指定分派失败的状态码。要使连接器在 ALE 模块已检索 IDoc 对象进行事件处理之后更新 SAP 失败状态码，您必须为此属性指定值（68 或 58）。SAP 将此值转换为 40。

有关更多信息，请参阅第 109 页的第 10 章，『ALE 模块概述』。

AleSuccessText

指定成功的“已记入应用程序文档”的描述文本。即使当您将 AleUpdateStatus 设置为 true 时，为此属性指定值也是可选的。

有关更多信息，请参阅第 109 页的第 10 章，『ALE 模块概述』。

AleFailureText

指定分派失败的描述文本。即使当您将 AleUpdateStatus 设置为 true 时，为此属性指定值也是可选的。

有关更多信息，请参阅第 109 页的第 10 章，『ALE 模块概述』。

ApplicationPassword

SAP 应用程序中连接器的用户帐户的密码。缺省值为 SOFTWARE。

ApplicationUserName

SAP 应用程序中连接器的用户帐户的名称。缺省值为 CROSSWORLDS。

ArchiveDays

ArchiveDays 连接器配置属性确定应在多少天之后从请求目录中删除“TID 管理”文件。内部保持的缺省值是 7 天。还可以指定部分日期值，例如，1.234。

Client

连接器登录所使用的客户机号，通常是 100。

Group

当配置连接器以获取负载均衡时，指定表示一组应用程序服务器的登录组的名称。有关更多信息，请参阅第 38 页的『利用负载均衡』。

gwService

网关服务器标识符；通常为 sapgw00。00 是运行 SAP 网关的服务器（通常是应用程序服务器）的系统号，如果您具有多个服务器，则可能不是 00。缺省值为 sapgw00。

Hostname

当配置连接器以获取负载均衡时，指定消息服务器的名称。当配置连接器以在没有负载均衡的情况下运行时，指定服务器登录的应用程序服务器的 IP 地址或名称。在这两种情况下，连接器假定网关主机的名称与为此属性指定的值相同。

Language

连接器登录所用的语言。缺省值为 E，表示英语。

MaxNumberOfConnections

连接器和 SAP 应用程序之间可能的最大并发交互数。这些交互包括轮询事件和处理服务调用请求。仅 ABAP 扩展、BAPI 和 ALE 模块使用此属性。ALE 模块仅将此属性用于服务调用请求。

因为每个交互都使用 SAP 应用程序服务器上的一个对话进程，所以连接数不能超过可用的对话进程数。有关更多信息，请参阅第 10 页的『处理多个并发交互』。

如果没有为此属性指定值，则连接器使用缺省值 2。

Modules

标识由连接器用来执行 init()、pollForEvents() 和 Terminate() 请求的模块。特别是，它指定由可视连接器框架使用的连接器模块。通过用逗号分隔每个值来指定多个连接器模块。不要添加空格。

受支持的连接器模块和指定它们的语法如下:

ABAP 扩展模块 - Extension

ALE 模块 - ALE

BAPI 模块 - Bapi

RFC 服务器模块 - RfcServer

分层动态检索模块 - Bapi

注: 当运行分层动态检索模块时，将值 Bapi 添加至此属性，以便在处理期间至少建立一个连接器线程，从而允许初始化和终止连接器。分层动态检索模块执行服务调用请求，因此，通过正在发送的业务对象中的元数据来调用业务对象处理程序。但是，通过将值 Bapi 添加至 Modules 属性建立了连接器线程，因此，如果在分层动态检索模块处理期间产生了任何问题，通过对正在运行的连接器线程调用 terminate() 方法就可以很容易地关闭连接器。

Namespace

指定连接器是否使用在连接器的名称空间 /CWLD/ 中定义的 ABAP 组件。该值必须设置为 true ，以便连接器使用在该名称空间中定义的 ABAP 组件。缺省值是 true。

NumberOfListeners

指定当初始化连接器时创建的侦听器线程数。侦听器线程一次可以处理一个请求。由于每个侦听器线程一次处理单个事件；因此，如果您具有多个侦听器线程，则连接器可以同时处理多个事件。缺省值为 1。

建议您不要让侦听器线程数超过 SAP 中可用的工作进程数。

PollQuantity

定义单个轮询选取的最大事件数。缺省值为 20。

RefreshLogonCycle

指定是否要对 SAP 客户机连接释放所有资源。缺省值为 false。

RfcProgramId

连接器在 SAP 网关中注册的标识，以便侦听器线程可以处理来自 RFC 支持功能的事件。此值必须与 SAP 应用程序（事务 SM59）中注册的程序标识匹配。缺省值是 CWLDSERVER。

有关在 SAP 应用程序中配置程序标识的更多信息，请参阅第 159 页的『向 SAP 网关注册 RFC 服务器模块』。

RfcTraceOn

指定是否生成详细说明每个侦听器线程的 RFC 活动的文本文件。您可以指定值 true 或 false。值 true 激活跟踪并生成一个文本文件。建议您仅在开发环境中使用这些文本文件，因为这些文件可能会快速增长。缺省值为 false。

SAPALE_Archive_Queue

指定在 ALE 模块处理完事件之后归档 TID 和 IDoc 数据的 MQ Series 队列。有关更多信息，请参阅第 109 页的第 10 章，『ALE 模块概述』。

没有缺省值。

SAPALE_Event_Queue

指定在 ALE 模块处理事件期间存储 TID 和 IDoc 数据的 MQ Series 队列。有关更多信息，请参阅第 109 页的第 10 章，『ALE 模块概述』。

没有缺省值。

SAPALE_Wip_Queue

指定当 ALE 模块构建事件队列的 MQ 消息时保存 TID 和 IDoc 数据的 MQ Series 正在进行的工作（wip）队列。连接器在接收到事件的所有数据之后，它将此队列中的数据移至 SAPALE_Event_Queue。有关更多信息，请参阅第 109 页的第 10 章，『ALE 模块概述』。

没有缺省值。

SAPALE_Error_Queue

定义一个队列来处理在 WIP 队列和事件队列之间失败的 MQ 消息。有关更多信息，请参阅第 109 页的第 10 章，『ALE 模块概述』。

SAPALE_Unsubscribed_Queue

定义一个队列来收集未预订的 IDoc 对象。未预订的 IDoc 对象先前放置在“归档”队列中。可以使用事件管理实用程序重新提交这些消息。现在，连接器在处理从 SAP 到

连接器的数据时检查预订，这导致在启动协作前事务保留在 SAP 中。有关更多信息，请参阅第 109 页的第 10 章，『ALE 模块概述』。

SAPSystemID

当配置连接器以获取负载均衡时，指定 SAP R/3 系统的逻辑名，它也称为 R3name。有关更多信息，请参阅第 38 页的『利用负载均衡』。

SAPtid_MQChannel

指定 MQ Series 队列管理器的客户机通道。有关更多信息，请参阅第 109 页的第 10 章，『ALE 模块概述』。

没有缺省值。

SAPtid_MQPort

指定用来与 MQ Series 队列管理器通信的端口，该队列管理器处理 ALE 模块的队列。有关更多信息，请参阅第 109 页的第 10 章，『ALE 模块概述』。

没有缺省值。

SAPtid_Queue

指定包含 TID 和 TID 状态的消息驻留的 MQ Series 队列。ALE 模块仅在处理请求时才使用此属性。有关更多信息，请参阅第 109 页的第 10 章，『ALE 模块概述』。

没有缺省值。

SAPtid_QueueManager

存储 TID 和 IDoc 数据的队列的 MQ Series 队列管理器名称。ALE 模块使用此属性来处理事件和请求。有关更多信息，请参阅第 109 页的第 10 章，『ALE 模块概述』。

没有缺省值。

SAPtid_QueueManagerHost

MQ Series 队列管理器驻留的主机的名称。ALE 模块使用此属性来处理事件和请求。有关更多信息，请参阅第 109 页的第 10 章，『ALE 模块概述』。

没有缺省值。

SAPtid_QueueManagerLogin

登录到 MQ Series 队列管理器的用户名。ALE 模块使用此属性来处理事件和请求。有关更多信息，请参阅第 109 页的第 10 章，『ALE 模块概述』。

没有缺省值。

SAPtid_QueueManagerPassword

登录到 MQ Series 队列管理器的用户的密码。ALE 模块使用此属性来处理事件和请求。有关更多信息，请参阅第 109 页的第 10 章，『ALE 模块概述』。

没有缺省值。

Sysnr

应用程序服务器的系统号。该值是一个两位数，通常为 00。缺省值为 00。

DateTimeFormat

保留与 DATE 和 TIME 字段值一起提供的定界符。如果设置为 Legacy，则连接器将保留 DATE 和 TIME 字段的定界符。否则，将除去定界符，并且值的长度将遵照属性定义长度。

TransIdCollabName

重要提示: 连接器不再支持此属性。

TransIdCollabName

重要提示: 连接器不再支持此属性。

UpdateIDocStatus

指出是否所有消息类型都需要审计跟踪。

IDocSuccessCode

分派成功的标准 IDoc 状态码。

IDocFailureCode

分派失败的标准 IDoc 状态码。

IDocSuccessText

与分派成功的 IDocSuccessCode 相关联的 IDoc 状态消息文本。

IDocFailureText

与分派失败的 IDocFailureCode 相关联的 IDoc 状态消息文本。

UseDefaults

在执行“创建”或“更新”操作时，如果 UseDefaults 设置为 true，则集成代理程序的适配器框架检查是否为标记为必需的每个业务对象属性提供了有效值或缺省值。如果提供了值，则“创建”或“更新”操作成功。如果该参数设置为 false ，则连接器仅检查有效值，并在未提供有效值时使“创建”或“更新”操作失败。缺省值为 false。

索引

[K]

[L]

[M]

[P]

[R]

[S]

[T]

[W]

[X]

[Y]

A

B

C

D

I

J

R

S

T

V

W

声明

IBM 可能并未在所有国家或地区提供本文档中讨论的产品、服务或功能特性。有关您当前所在区域的产品和服务的信息，请向您当地的 IBM 代表咨询。任何对 IBM 产品、程序或服务的引用并非意在明示或暗示只能使用 IBM 的产品、程序或服务。只要不侵犯 IBM 的知识产权，任何同等功能的产品、程序或服务，都可以代替 IBM 产品、程序或服务。但是，评估和验证任何非 IBM 产品、程序或服务，则由用户自行负责。

IBM 公司可能已拥有或正在申请与本文档内容有关的各项专利。提供本文档并未授予用户使用这些专利的任何许可。您可以用书面方式将许可查询寄往:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

本条款不适用英国或任何这样的条款与当地法律不一致的国家或地区:

International Business Machines Corporation “按现状” 提供本出版物，不附有任何种类的（无论是明示的还是暗含的）保证，包括但不限于暗含的有关非侵权、适销和适用于某种特定用途的保证。某些国家或地区在某些交易中不允许免除明示或暗含的保证。因此本条款可能不适用于您。

本信息中可能包含技术方面不够准确的地方或印刷错误。此处的信息将定期更改；这些更改将编入本资料的新版本中。IBM 可以随时对本资料中描述的产品和／或程序进行改进和／或更改，而不另行通知。

本信息中对非 IBM Web 站点的任何引用都只是为了方便起见才提供的，不以任何方式充当对那些 Web 站点的保证。那些 Web 站点中的资料不是 IBM 产品资料的一部分，使用那些 Web 站点带来的风险将由您自行承担。

IBM 可以按它认为适当的任何方式使用或分发您所提供的任何信息而无须对您承担任何责任。

本程序的被许可方如果要了解有关程序的信息以达到如下目的: （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，以及（ii）允许对已经交换的信息进行相互使用，请与下列地址联系:

IBM Burlingame Laboratory Director
IBM Burlingame Laboratory
577 Airport Blvd., Suite 800
Burlingame, CA 94010
U.S.A

只要遵守适当的条件和条款，包括某些情形下的一定数量的付费，都可获得这方面的信息。

本资料中描述的许可程序及其所有可用的许可资料均由 IBM 依据 IBM 客户协议、IBM 国际软件许可协议或任何同等协议中的条款提供。

此处包含的任何性能数据都是在受控环境中测得的。因此，在其他操作环境中获得的数据可能会有明显的不同。有些测量可能是在开发级的系统上进行的，因此不保证与一般可用系统上进行的测量结果相同。此外，有些测量是通过推算而估计的，实际结果可能会有差异。本文档的用户应当验证其特定环境的适用数据。

涉及非 IBM 产品的信息可从这些产品的供应商、其出版说明或其他可公开获得的资料中获取。IBM 没有对这些产品进行必要的测试，也无法确认其性能的精确性、兼容性或任何其他关于非 IBM 产品的声明。有关非 IBM 产品性能的问题应当向这些产品的供应商提出。

本信息可能包含在日常业务经营中使用的数据和报告的示例。为了尽可能完整地说明这些示例，这些示例中可能会包括个人、公司、品牌和产品的名称。所有这些名称都是虚构的，与实际商业企业所用的名称和地址的任何雷同纯属巧合。

所有关于 IBM 未来方向或意向的声明都可随时更改或收回，而不另行通知，它们仅仅表示了目标和意愿而已。

版权许可

本信息包括源语言形式的样本应用程序，这些样本说明不同操作平台上的编程方法。如果是为按照在编写样本程序的操作平台上的应用编程接口（API）进行应用程序的开发、使用、经销或分发为目的，您可以任何形式对这些样本程序进行复制、修改、分发，而无须向 IBM 付费。这些示例并未在所有条件下作全面测试。因此，IBM 不能担保或暗示这些程序的可靠性、可维护性或功能。

编程接口信息

编程接口信息（如果提供的话）用于帮助您创建使用本程序的应用软件。

通用编程接口允许您编写获取本程序工具的服务的应用软件。

但是，此信息可能还包含诊断、修改和调整信息。提供诊断、修改和调整信息是为了帮助您调试应用软件。

警告: 切勿使用此诊断、修改和调整信息作为编程接口，因为它随更改而变化。

商标和服务标记

下列各项是国际商业机器公司在美国和／或其他国家或地区的商标或注册商标:

IBM
the IBM logo
AIX
CICS
CrossWorlds
DB2
DB2 Universal Database
Domino
IMS
Informix
iSeries

Lotus
Lotus Notes
MQIntegrator
MQSeries
MVS
OS/400
Passport Advantage
SupportPac
WebSphere
z/OS

Microsoft、Windows、Windows NT 和 Windows 徽标是 Microsoft Corporation 在美国和／或其他国家或地区的商标。

MMX、Pentium 和 ProShare 是 Intel Corporation 在美国和／或其他国家或地区的商标或注册商标。

Java 和所有基于 Java 的商标是 Sun Microsystems, Inc. 在美国和／或其他国家或地区的商标。

Linux 是 Linus Torvalds 在美国和／或其他国家或地区的商标。

其他公司、产品或服务名称可能是其他公司的商标或服务标记。

WebSphere Business Integration Adapter Framework V2.6.0

中国印刷