
第 8 章 集成最佳实践

本章的目的是对 WebSphere Product Center 实现中的“集成最佳实践”进行总结。使用这些最佳实践将有助于在系统之间实现高质量并且可靠的集成。为了阐述集成的所有方面，本文档的编写采用了以下规则，对集成的每个不同方面标出相关联的最佳实践。

关键的集成元素：

- 设计原理
- 实现
- 验证
- 可视性

定义与首字母缩写词

集成因素：我们可以使用下面列示的因素来理解在 WebSphere Product Center 实现中遇到的各种类型的实现。文档的其余部分将着重说明哪些最佳实践或指南适用于实现的哪些因素或类型。

WebSphere Product Center 作为源或目标系统

最明显的因素是：WebSphere Product Center 是正在交换的信息的源系统还是目标系统。源系统对集成施加它自己的约束，最重要的约束是（a）执行增量联合的能力，（b）启动集成的能力，（c）接收有关所发送的数据的成功 / 失败通知以及执行适当操作的能力，（d）所支持的协议和格式以及对 EAI 基础结构的支持。

控制系统

我们将控制系统定义为根据集成的内部触发器来执行操作的系统。根据时间表以作业形式运行联合的 WebSphere Product Center 就是一个示例。另一个示例是由于添加项而触发 WBI 适配器的 SAP。

WebSphere Product Center 是集成的源系统还是目标系统与集成中的哪个系统是控制系统完全无关。有几种可能的情况。当涉及中介（如 FTP 服务器或 EAI 工具）时，源系统和目标系统都可以是控制系统：基于调度的旧系统将文件放在 FTP 服务器上，而基于调度的 WebSphere Product Center 检取该文件。WebSphere Product Center 作为受控目标系统（即，它等待外部信息来触发数据导入）的示例是：IBM WBI 通过调用程序来将消息公布至 WebSphere Product Center，例如，消息内容是来自 Transora 的项更新。WebSphere Product Center 作为受控源系统（即，它等待外部信息来触发数据导出）的一个示例是：IBM WBI 定期轮询 WebSphere Product Center 队列以了解是否已有可供检取的文件。

协议

在 WebSphere Product Center 实现小组中以及在客户资源中，协议、

格式和消息与基于文件的集成之间存在着概念混淆。因此，本文档的其中一个目标是确保我们已经为这些概念确定了公共的术语。协议示例包括文件传输协议（FTP）、超文本传输协议（HTTP）、简单消息传输协议（SMTP，即电子邮件）、Java 消息传递服务（JMS）和 IBM WebSphere Message Queuing（IBM WebSphere MQ）。协议定义诸如包络、数据（例如数字数组）的编码以及期望的响应之类的信息，但与所传送的内容无关。在所有集成中，我们都应该相当明确地了解所使用的协议，这是因为我们将始终使用至少一种协议。此外，集成的各个阶段可能实际上使用了不同的协议：SAP 中的 WBI 适配器可能通过 HTTP 将数据从 SAP 传送至 WBI Inter Connection Server（ICS），后者接着将该数据移交给 IBM MQ 队列管理器，以便将该数据传送给另一个作为 MQ 客户机与 WebSphere Product Center 相连接的队列管理器。

格式

数据的布局格式与协议无关。格式的示例包括逗号分隔的值（CSV）、竖杠定界、可扩展标记语言（XML）或仅仅是一些预定义的字段和记录结构（如对于电子数据交换（EDI）消息的情况）。每种格式都通过位置 / 长度参数或通过标记来定义字段。记住特定格式可能需要的编码十分重要。例如这样的事实：在实现中，常常会忘记需要在 XML 中对诸如尖括号（“<”和“>”）之类的字符进行转义或者内容实际上可能包含的是 CSV 中的逗号。

数据大小

人们常常将这个因素与“基于消息的”或“基于文件的”通信混淆，因此，弄清这个因素的含义十分重要。“基于消息的”集成通常被假定为涉及较小型的数据交换，属性与以下类似：

- 数据交换更频繁并且数据量更少，因此，与面向“批处理”的传统系统相比（在这些系统中，可能每星期执行一次导出 / 导入），传达更改的时间间隔要短得多。
- 两个系统（源系统和目标系统）相互联系以便对发送的消息进行处理并进行应答，而不是生成一个文件，该文件可能通过 FTP 被传送至其它位置或者在文件系统中停留了一星期才被检取以进行处理。

但是，没有明确的界线来区分基于消息的集成与基于文件的集成（即批处理集成），并且，定义一组明确的因素而不是相互混淆或重叠的因素十分重要。因此，与数据是被标记为“基于消息的”集成还是被标记为“批处理”集成相比，数据的整体大小应该是需要考虑的更重要的因素。

通信类型

另一个要考虑的因素是集成将要涉及的通信的类型。同步通信将特定操作的结果直接反馈给用户或系统。例如，通过使用 HTTP 来进行通信，就可以在发出操作后将结果自动反馈给系统或用户。另一方面，异步通信更多地使用“射后不理”策略。例如，如果集成涉及在 FTP 服务器上存储文件，以后，某个系统检取该文件时，不会将此操作的结果自动反馈给存储该文件的系统。

频率

与“数据大小”因素一起，这个“频率”因素捕获将需要定期处理的数据的总量。

集成线程

这个中间系统和基础结构因素捕获 EAI 基础结构是否正在使用。并且，在带有旧系统的集成中，有时可以编写中间程序以将数据上载或抽取到旧系统中。由于这些中间系统或程序通常是集成链中最弱的环节，因此，了解并记录这些中间系统或程序十分重要。

尤其是，在可能需要多次跳跃（例如，从 WebSphere Product Center 到 WBI 再到目标系统）的复杂集成中，非标准方法（如直接更新数据库）、多种协议或其它通信挑战（如通过防火墙进行通信）将提早建立单一工作线程或完整集成路径。这将标识问题并给予其它各方（如网络管理员或者在 IBM WBI 上工作的小组）充足的时间来并行地解决这些连接问题。

在 WebSphere Product Center 实现中，上面列示的集成因素应该成为用于描述集成的标准术语。PS 小组在分析 / 设计阶段提供的文档应该明确并一致地使用这些因素。

首字母缩写词

首字母缩写词	定义
EAI	企业应用程序集成
FTP	文件传输协议
HTTP	超文本传输协议
MQ	IBM 的消息排队中间件。由于所有连接解决方案现在都已纳入 WebSphere 品牌，所以 MQ 通常是指 IBM Websphere MQ。
ICS	IBM 的 WBI 互联服务器
WBI	IBM 的 WebSphere Business Integration 套件，即 IBM 的 EAI 套件。

设计原理

可重用性

可重用性是集成实现方法的总体基础。随着 WebSphere Product Center 的发展以及完成了越来越多的客户实现，我们就需要能够快速地调整

并解决与先前未集成的系统进行集成以及与那些在先前实现中已集成的系统进行集成的问题。为了满足这种需求，我们有必要使用可重用性来完成所有集成工作，以便在如果需要为另一个客户与同一系统集成时，我们能够以最高的效率做到这一点。

可重用性是通过以下方法实现的：（a）利用 EAI 工具（如 IBM WBI）及其一般业务对象模型，（b）选择独立于数据模型的格式，（c）编写独立于数据模型并且可以在其它实现中重用的 WebSphere Product Center 脚本（确认和轮询等）的库。

信息共享

通信作为集成方法

在概念上，可以将集成想象为仅仅是可以由控制系统 WebSphere Product Center 与受控系统之间的通信触发的一系列事件。可以通过系统间传递的消息、轮询内容或文件的自动过程或任何其它基本通信方法来触发这些事件。例如，通信内容将包括将要进行的更改的类型（添加、更新或删除）、唯一的通信标识（用于跟踪 / 确认）以及用来在 WebSphere Product Center 中或在整个系统中影响更改的相关内容。

可靠性衡量

除了在系统之间传递信息以传达更改之外，还应该有一种适当的方法来传达特定事务的成功或失败信息。可以通过同步通信形式来最直观地实现这种握手通信，并且，这种握手通信允许集成的系统跟踪在另一端接收失败时是否需要重新发送特定事务，从而改进并最终确保集成的可靠性。

信息格式

应该以很通用的方式来设计这些通信的特定格式，以便可以跨多个实现来重用该格式以及围绕它的处理功能。

当考虑用于系统之间的通信的一般格式时，重要的是根据下列需要来记录格式的可用性：

- 国际字符集和特殊字符（逗号，引号和尖括号等等）
- 复杂结构（即，内容和关系的层次结构）
- 能够处理内容或项占位符的多个实例并且每个实例具有不同的值

信息处理

虽然在系统之间发送的信息的格式在一定程度上应该具有一般性，但是，并非所有实现都能够满足预定义格式也是可以理解的，当我们要将集成看作 WebSphere Product Center 与整个系统之间的直接链接时尤其如此。为了避免由于差别（如数据模型差别）而需要在每个实现中对格式以及格式与 WebSphere Product Center 规范之间的映射进行重新处理，建议在 XML 格式与 WebSphere Product Center 规范之间使用可重用映射功能。

使用 EAI 平台

执行此操作的一种方式是使用 EAI 平台（如 WBI 或 webMethods 套件），这将允许我们构建可重用的连接器，例如，允许 WebSphere Product Center 通过完全可重用的单一消息格式（例如单一 XML DTD）进行通信的连接器。然后，WBI 可以对由于实现的具体情况的不同而导致的差别进行转换，而不是要求重新处理 WebSphere Product Center 功能以处理信息。由于不要求重新处理 WebSphere Product Center 功能，所以，可以跨多个实现使用同一功能。

其它选项

但是，需要考虑的另一个因素是，特定的客户可能需要重用在他们的企业内已用于其它系统的格式。这将使得 WebSphere Product Center 难以引入完全独立的 DTD（以后，将需要为企业中的其它系统转换此 DTD 以使其可以被理解而不是为 WebSphere Product Center 进行转换以使用已存在的 DTD）。在这种情况下，我们应该使用可重用功能来在 WebSphere Product Center 中的规范与 DTD 之间进行转换。

另一种可能性是，即使使用了 EAI 平台，但对于特定的客户来说，从理解将 WebSphere Product Center 管理的信息映射到它们的内部 DTD 的角度看来，这种方法对于交流信息来说也更有用，因此是更为理想的方法。

事件处理

理想情况下，WebSphere Product Center 中的一个自动进程将处理事件。例如，可以使用 WebSphere Product Center 发行版中引入的排队功能来处理消息的发送（出站队列）以及响应和入局消息（进站队列）的接收。然后，可以利用队列处理脚本来对那些消息进行实际处理，并因此实际地执行作为特定消息的结果而将要被触发的事件。

但是，事件处理不需要直接与 WebSphere Product Center 的特定功能或特定版本相联系。其它事件处理方法可以包括 WebSphere Product Center 中轮询 FTP 服务器的已调度作业、（通过文档库）在本地文件系统中查找文件的已调度作业、让基于调用程序的触发器脚本根据已公布的信息来激发事件或其它方法。选择的方法应最终取决于对特定集成的数据大小和频率因素需求的仔细考虑。

更改跟踪

为了能够在系统之间实现完全同步，在 WebSphere Product Center 中需要有一种方法来跟踪对某些内容和项所作的更改，那些内容和项可以被进一步地有效标记为是否传达给整个系统。例如，如果在（作为源系统的）WebSphere Product Center 中删除了一个项，我们有可能希望不仅能够触发将一条消息发送至目标系统（该消息通知目标系统：它应该删除同一个项），并且还能够跟踪该特定通信的成功或失败以使 WebSphere Product Center 可以了解是否从目标系统中实际地删除

了该项。

可重用的连接器

连接器资源库

随着实现的进步以及我们的合作伙伴的发展，我们将逐渐构建用于各种系统的可重用连接器的资源库。我们应尽一切可能来重用这些连接器，这是因为从特定实现的角度来看，只需要对通过这些连接器来处理项等的功能作小幅修改甚至不作修改就可以重用这些功能。当然，因为随着时间的推移会不断地发现并解决问题，所以，使用这些连接器将大大加快实现的总体执行速度并增强连接器以及使用那些连接器的实现的总体可靠性和稳定性。

当与尚未定义连接器的系统集成时，应该请一位集成专家参与工作，他可以快速地构建一个可重用连接器，然后，既可以将该连接器用于特定实现上的集成也可以将其存储在连接器资源库中，以后，万一我们需要与另一个实现上的系统集成时，可以使用该连接器。

连接器的使用

应该使用连接器，以使可能需要进行的任何修改是通过正在系统之间传递的信息的 EAI 层处理转换进行的。换言之，在重新编写 WebSphere Product Center 中任何用于处理通过 EAI 传递的信息的可重用功能之前，我们应该利用 EAI 平台的能力来执行必需的转换，以便不需要重新编写 WebSphere Product Center 中的任何功能。

实现

对实现进行缩放

最小集成

应该将大型的总体集成任务划分成小得多并且更易于管理的任务。例如，这可以通过将单个完整的集成划分为小得多的集成来完成 — 从每个项类型（规范）的“独立”集成，到每个容器（目录）的集成，一直到一组属性的集成（如果有必要的话）。一旦确信这些“最小集成”中的每一个都毫无缺陷地工作时，可以对它们进行组合以形成一个完整的集成。

功能的粒度

您应该对系统之间需要发生的集成的级别特别加以注意。例如，当将更改发送至目标系统时，用户可能希望能够发送自从特定日期之后的所有更改、仅发送特定目录的自从上次发送更改后的那些更改、仅发送对特定项组发生的更

改或者发送对所有项的特定属性发生的任何更改。特定的需求将依赖于实现，但是，在实现的设计过程中提早考虑所需的粒度以便适当地满足该需求十分重要。

性能调整

一般性能注释

不要将性能问题作为遗留问题。在最后阶段更改和修正集成的格式或其它方面很容易，但是，性能瓶颈可能需要进行重大的重新设计，有时还需要工程支持。在开发过程中的适当时候，请对脚本进行性能衡量。

衡量性能

在采用最小集成方法（在“实现”一节中详细描述）的情况下，应该在集成的每个步骤中通过测量每个最小集成任务所需的总时间来衡量性能。这样，可以在适当的粒度级别标识潜在的性能不佳区域，因而可以更方便地进行性能调节。

调节性能

在标识性能问题区域之后，应该进行详细的分析以确定操作缓慢的根本原因。可以使用诸如 WebSphere Product Center 的中间件概要分析以及作业详细信息屏幕上的性能选项卡之类的工具来进行详细的分析。然后，该分析可侧重于脚本或 SQL 查询的特定方面，然后，可以执行适当的操作 — 修改或重新编写脚本，或者进行“工程”以增强数据库查询。

验证

稳定性

最小集成的优点

通过提供集成显示为工作正常的所有区域的详细列表，实现最小集成（在“实现”一节详细描述）应该能够使您高度确信集成已成功。在看不到最小集成的情况下，不仅更加难以提供集成起作用的详细信息的证明，整个集成也更加有可能会遇到难以标识、诊断和调试的问题。因此，实现最小集成将增强集成的总体稳定性。

可伸缩性测试

最小集成的优点

通过实现最小集成（在“实现”一节详细描述），就可以在精细得多的详细信息级别进行集成测试，从而使得发生

的任何错误或问题不会被大量的（并且可能是不相关的）复杂事物掩盖。因此，如上所述，此方法应该能够大大加快诊断、调试和解决所找到的问题的过程。

代表环境与完整环境

应该对与最终环境具有相同的配置（相同的规范、验证规则、值规则和视图）但具有尽可能少的代表实体（语言环境、目录、类别树、项、类别、组织、用户和角色）的代表环境完成集成测试。与在全面环境中进行的测试相比，这应该会缩短测试的运行时间和屏幕的装入时间，并且，通常应该能够加快测试速度。所有测试和调试都应该在此环境中完成。

仅当在代表环境中完成了测试并且该环境中的所有内容都表现为工作正常之后，才应该在完整并且全面的环境中验证集成。但是，为了确保在代表环境中没有意外地忽略任何边缘方案，并且为了测试集成的产品级别性能，仍应该执行此步骤。

可伸缩过程测试

首先，只应该对非常少量（10 个或更少）的代表项运行任何可调度的作业（即导入和导出）。应该将此数目按比例增大到在实际处理这些项的脚本中获得的置信度级别。这种方法将确保不会有以下情况出现：一个大型的作业运行了数小时，但是运行它的用户最后才发现出现的错误情况没有使整个进程在运行过程中的最初几分钟内立刻失败。

只有在充分确信与作业相关联的脚本的操作之后，才在使用完整数据集的情况下运行该作业。与对于完整环境的建议相同，为了确保在较小型的作业运行中没有意外地忽略任何边缘方案，并且为了测试作业的产品级别性能，仍应该执行此步骤。

可视性

报告

最小集成的优点

实现最小集成（在“实现”一节中详细描述）允许进行更为详细的报告，这是因为集成块更小并且能够更快地实现。与在整个集成的级别报告实现进度相比，这个更精细的报告级别允许更加具体和定量地跟踪实现。

可以在一个图表中列示最小集成以及它们与更大的完整集成图形的关系，然后，可以根据最小集成任务的进度报告来方便地绘制出总体实现进度的准确图形。

所有权

即使是在与多个小组一起工作时，也应该将整个集成的所有权授予单一人员。这个人的工作是确保提早建立单一线程、确保各个小组根据本文档中的指南来工作以及确保跨（a）最小集成、（b）可伸缩过程测试和（c）代表环境与完整环境的增量式构建 / 测试环节在各小组之间同步。

文档

明确地标识格式和方法

当有多个小组在集成中工作时，请决定一条明确的执行路径，并明确地记录将要使用的任何格式。最普通的示例是，一个 WebSphere Product Center 小组正在从 WebSphere Product Center 中导出数据，而一个客户或 SI 小组正在将该数据上载至目标系统。在没有公共格式的规范之前，不要开始工作，并且，每天都应该保持此文档处于最新状态。这是项目管理员必须实施的绝对要求。

此方法并非与使用代表环境和执行最小集成不一致。两个小组都应该以增量方式进行构建和测试以确保进度稳定并且可视。

有关 WebSphere Product Center 集成的最重要的 10 项准则

使用明确并且公共的术语来描述集成

所有实现都应该使用“集成因素”一节中标识的因素。

可重用性

进行缩放的关键在于从先前的集成学习并在牢记“可重用性”一节所描述的可重用性原则的情况下封装集成。

可视性

确定报告进度的总体标准，并且，最起码每隔几天向项目管理员提供一次明确的状态更新。

最小集成

根据对一个大型集成有意义的各个因素（目录和属性）来对该集成的复杂性进行划分。每次都把注意力放在一个最小集成上，并且直接与可视性标准联系在一起。

代表环境与完整环境

维护易于调试和测试的代表环境。仅当确信脚本和规范的有效时，才

移至完整环境。将此项与可视性标准联系在一起。

可伸缩过程测试

在转到完整的数据集之前，对小型数据集测试所有作业以检查正确性。将此项与可视性标准联系在一起。

性能

在开发周期内的早期运行一些性能测试并在此之后定期地运行那些测试以标识问题，而不必担心逻辑或格式化的正确性。

提早建立单一线程

请提早建立单一工作线程，在需要多个跳跃、多种协议或非标准方法的复杂集成中尤其如此。

设计规范和文档

请定义并记录明确的执行路径并明确地记录将要使用的任何格式，当有多个小组在集成中工作时尤其如此。

单个所有者

即使是在与多个小组一起工作时，也应该将整个集成的所有权授予单一人员。

EAI 平台集成

方法

一般通信格式

每当有可能的时候，都应该设计一般通信格式或重用上一个项目的一般通信格式。格式越一般化，就会有越多的系统可以参与集成，而不需要对格式进行特殊的重新处理就可以让所有系统能够相互通信。当然，格式越一般，性能就会打折扣，因此，对一个项目正确的格式对另一个项目来说可能不是理想的选择。当确定所要使用的特定格式时，仍应该考虑“集成因素”。

内容映射

应该尽可能地通过可动态更新的方法来完成 WebSphere Product Center 中的内容模型与通过通信格式看到的通信模型之间的映射。此外，根据对“集成因素”的调查，特定的项目需求可能会指示这些映射的创建不能是完全可动态更新的 — 例如，因为对绝对最大处理吞吐量设置了高优先级。执行此操作的一种方式涉及使用类别树（例如，表

示一个 XML 结构)，那些类别树有单一的相关节点规范，该节点规范可以指示该类别树的特定节点在 WebSphere Product Center 的内容模型中映射至的属性的规范节点路径。然后，可以利用递归处理脚本来根据此类别树及其定义的映射处理项到 XML 文件的映射，甚至只需花很少的功夫就可以满足嵌套的多次出现的需要。

其它优点

信息转换 / 变换

在集成中涉及的系统本身应该不需要处理集成中的其它系统的信息或内容限制和需求。可以很方便地利用 EAI 平台来处理此内容转换和变换。例如，虽然 WebSphere Product Center 将 FLAG 的值存储为“TRUE”或“FALSE”，但是，我们与之集成的系统可能将该值存储为“Y”或“N”。可以使用 EAI 平台来执行这些转换，以便 WebSphere Product Center 始终可以发送 TRUE/FALSE 并假定将把它作为 TRUE/FALSE 发送，而集成系统始终可以发送 Y/N 并假定将把它作为 Y/N 发送。这确保即使集成涉及更多的系统，对于这些附加系统也完全不需要进行重新编码。

客户理解

由于我们可以重用客户有可能已熟悉的平台，所以，客户可以更加确信集成使用已知的功能 — 例如 EAI 平台。另外，如果已存在特定于客户机的通信格式并且将该格式用于 WebSphere Product Center 集成，则客户机端开发者几乎不需要参加其它培训就能够理解 WebSphere Product Center 将要映射到的通信格式。

通信灵活性和可靠性

大多数 EAI 平台提供了本机功能来允许通过各种协议进行通信并确保通过代理传递通信。这使 WebSphere Product Center 能够把工作重心放在仅仅生成所要传送的必需文档上，而不必担心是否支持其它方法来将此文档传送至各种系统，也无需关心跟踪此文档是否已被每个系统接收 — 这些问题成为 EAI 层和平台关注的问题，WebSphere Product Center 仅仅需要从总体集成线程的角度了解它们。