Sun Java Enterprise System 2005Q4 部署规划指南

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95054
U.S.A.

文件号码 819–3450
2005 年 10 月

目录

表

图

前言

《Sun Java Enterprise System 2005Q4 部署规划指南》介绍如何基于 Sun Java™ Enterprise System 规划和设计企业部署解决方案。本指南介绍了部署规划和设计的基本概念和原理，对解决方案生命周期（概述了部署设计项目的所有阶段和任务）进行了讨论，并提供了使用 Java Enterprise System (Java ES) 规划企业范围部署解决方案时的高级示例和策略。

目标读者

本指南主要适用于负责分析和设计企业部署的部署结构设计师和业务规划员。对负责设计和实现企业应用程序各个方面的系统集成员及其他人员也有所帮助。

阅读本书之前

本指南假定您已掌握企业级应用程序的设计和安装，而且已阅读了《Sun Java Enterprise System 2005Q4 技术概述》。

本书的结构

本指南以一个说明部署规划不同阶段的解决方案生命周期为基础。第 1 章介绍解决方案生命周期。

Java ES 文档集

Java ES 文档集介绍部署规划和系统安装。系统文档的 URL 是 `http://docs.sun.com/coll/1286.1` 及 `http://docs.sun.com/coll/1382.1`。有关 Java ES 的简介，请参阅下表中按顺序列出的书籍。

表 P–1 Java Enterprise System 文档

文档标题	目录
《Sun Java Enterprise System 2005Q4 发行说明》	含有有关 Java ES 的最新信息，包括已知问题。此外，各组件都有对应的发行说明。
《Sun Java Enterprise System 2005Q4 文档汇总信息》	从系统及组件层面介绍了与 Java ES 有关的所有文档。
《Sun Java Enterprise System 2005Q4 技术概述》	介绍基本的 Java ES 技术和概念信息。描述组件、体系结构、过程和功能。
《Sun Java Enterprise System 2005Q4 部署规划指南》	介绍如何基于 Java ES 规划和设计企业部署解决方案。介绍部署规划和设计的基本概念及原理，讨论解决方案的生命周期，并提供基于 Java ES 规划解决方案时使用的高级示例和策略。
《Sun Java Enterprise System 2005Q4 安装规划指南》	帮助您形成 Java ES 部署的硬件、操作系统和网络方面的实施规范。介绍在安装和配置规划中要解决的一些问题，如组件依赖性问题。
《Sun Java Enterprise System 2005Q4 安装指南》	介绍在 Solaris 操作系统或 Linux 操作系统上安装 Java ES 的详细步骤。还讲述了在安装后如何配置组件，及如何确定各组件运行正常。
《Sun Java Enterprise System 2005Q4 安装参考》	介绍有关配置参数的更多信息，提供在配置规划中将使用的工作单，并列出参考材料（如默认目录和端口号）。
《Sun Java Enterprise System 2005Q1 部署示例系列：评估方案》	介绍如何在一个系统上安装 Java ES 、建立一组核心共享网络服务以及如何设置可访问所建立服务的用户帐户。
《Sun Java Enterprise System 2005Q4 升级指南》	说明如何在 Solaris 操作系统和 Linux 操作系统环境升级 Java ES。
《Sun Java Enterprise System 术语表》	定义在 Java ES 文档中使用的术语。

印刷约定

下表介绍了本书所采用的印刷约定。

表 P–2 印刷约定

字体	含义	示例
`AaBbCc123`	命令、文件和目录的名称；计算机屏幕输出	编辑 `.login` 文件。 使用 `ls -a` 列出所有文件。 `machine_name% you have mail.`
`AaBbCc123`	用户键入的内容，与计算机屏幕输出的显示不同。	`machine_name%` **`su`** `Password:`
AaBbCc123	保留未译的新词或术语以及要强调的词。要使用实名或值替换的命令行变量。	删除文件的命令是 `rm` *filename*。
新词术语强调	新词或术语以及要强调的词。	**高速缓存**是本地存储的副本。 **切勿**保存文件。
《书名》	书名	阅读《用户指南》的第 6 章。

命令中的 Shell 提示符示例

下表给出了默认的系统提示符和超级用户提示符。

表 P–3 Shell 提示符

Shell	提示符
UNIX 和 Linux 系统上的 C shell	`machine_name%`
UNIX 和 Linux 系统上的 C shell 超级用户	`machine_name#`
UNIX 和 Linux 系统上的 Bourne shell 和 Korn shell	`$`
UNIX 和 Linux 系统上的 Bourne shell 和 Korn shell 超级用户	`#`
Microsoft Windows 命令行	`C:\`

符号约定

下表说明本书中可能用到的一些符号。

表 P–4 符号约定

符号	说明	示例	含义
[ ]	包含可选参数和命令选项。	`ls [-l]`	`-l` 选项不是必需的。
{ | }	包含所需命令选项的一组选择。	`-d {y|n}`	`-d` 选项要求您使用 `y` 参数或 `n` 参数。
${ }	表示变量引用。	`${com.sun.javaRoot}`	引用变量 `com.sun.javaRoot` 的值。
-	结合同时发生的多个击键。	Control-A	按 A 键的同时按 Control 键。
+	结合相继发生的多个击键。	Ctrl+A+N	按 Control 键后松开，然后按后续各键。
→	表示图形用户界面中的菜单项选择。	“文件”→“新建”→“模板”	从“文件”菜单中选择“新建”。 从“新建”子菜单中选择“模板”。

联机访问 Sun 资源

通过 docs.sun.com^{SM} Web 站点可以联机访问 Sun 技术文档。您可以浏览 docs.sun.com 文档库或搜索具体的书名或主题。书籍以 PDF 和 HTML 格式的联机文件方式提供。行动不便的用户借助辅助技术也可以阅读这两种格式的文件。

要访问以下 Sun 资源，请转到 `http://www.sun.com`：

- Sun 产品的下载
- 服务和解决方案
- 支持（包括修补程序和更新）
- 培训
- 研究
- 团体（例如，Sun 开发者网络）

第三方 Web 站点引用

本文档引用第三方 URL，并提供其他相关信息。

注 – Sun 对本文档中提到的第三方站点的可用性不承担任何责任。对于此类站点或资源中的（或通过它们获得的）任何内容、广告、产品或其他材料，Sun 并不表示认可，也不承担任何责任。对于因使用或依靠此类站点或资源中的（或通过它们获得的）任何内容、产品或服务而造成的或连带产生的实际或名义损坏或损失，Sun 概不负责，也不承担任何责任。

Sun 欢迎您提出意见

Sun 致力于提高其文档的质量，并十分乐意收到您的意见和建议。要共享您的意见，请访问 `http://docs.sun.com`，然后单击“发送意见”(Send Comments)。请在联机表单中提供完整的文档标题和文件号码。文件号码包含 7 位或 9 位数字，可以在书的标题页或文档的 URL 中找到该号码。例如，本书的文件号码是 819-3450。提出意见时您还需要在表格中输入文件的英文文件号码和标题。本文件的英文文件号码是 819-2326，文件标题为《Sun Java Enterprise System 2005Q4 Deployment Planning Guide》。

第 1 章

部署规划简介

本章概括说明 Sun Java™ Enterprise System (Java ES)，讨论部署规划概念并介绍解决方案生命周期。该生命周期概述了规划和设计企业软件系统的各个步骤。本章包括以下部分：

关于 Java Enterprise System

Java Enterprise System 是一种软件框架结构，提供了可支持分布于网络或 Internet 环境中的企业级应用程序的一整套中间件服务。提供这些服务的 Java Enterprise System 组件均通过一个通用安装程序进行安装，在一组公用共享库上保持同步，并共享一个集成的用户身份和安全管理系统。

系统服务

Java Enterprise System 组件提供的主要基础结构服务可分为以下类别：

- **门户服务。**这些服务使移动办公员工、远程办公者、知识工作者、商业伙伴、供应商和客户能够从公司网络以外的任何地方通过 Internet 安全地访问各自的个性化企业门户。各用户团体可随时随地访问这些服务，同时还提供集成、聚合、个性化、安全性、移动访问及搜索功能。
- **通信和协作服务。**利用这些服务，可以在多样用户团体间安全地进行信息交换。具体功能包括用户业务环境上下文中的消息传送、实时协作以及日程安排。
- **网络身份认证和安全服务。**这些服务可确保在全局基础上对所有团体、应用程序和服务强制实施适当的访问控制策略，从而增强对公司主要信息资产的安全保护。这些服务使用信息库来存储和管理身份配置文件、访问权限以及应用程序和网络资源信息。

- **Web 和应用程序服务**。这些服务可使分布式组件之间相互通信，并支持范围广泛的服务器、客户机和设备的应用程序的开发、部署和管理。这些服务均基于 Java 2 Platform, Enterprise Edition (J2EE™) 技术。
- **可用性服务**。这些服务为应用程序和 Web 服务提供近乎连续的可用性和可伸缩性。

下表列出了上述服务类别，并指定了为每个类别提供服务的 Java Enterprise System 组件。

表 1–1 Java Enterprise System 服务类别

服务类别	Java Enterprise System 组件
门户服务	Portal Server
	Portal Server Secure Remote Access
	Access Manager
	Directory Server
	Application Server 或 Web Server
通信和协作服务	Messaging Server
	Calendar Server
	Instant Messaging
	Access Manager
	Directory Server
	Application Server 或 Web Server
网络身份管理服务	Access Manager
	Directory Server
	Web Server
Web 和应用程序服务	Application Server
	Message Queue
	Web Server
可用性服务	Sun Cluster
	Sun Cluster 代理

有关 Java Enterprise System 服务、组件和 Java Enterprise System 体系结构概念的更多信息，参阅《Sun Java Enterprise System 2005Q4 技术概述》。

内置服务和定制开发服务

基于 Java Enterprise System 的部署解决方案通常分为两大类：

- **80:20 部署**。这些解决方案主要由 Java Enterprise System 提供的各种服务组成。Java Enterprise System 提供约 80% 或更多的服务。
- **20:80 部署**。这些解决方案由大量定制开发服务和第三方应用程序组成。

80:20 和 20:80 类别是一种广义的概括。所提供服务类型的具体百分比并不重要。但是，这个百分比表示了解决方案中所包含的定制量。

Java Enterprise System 因拥有由 Java ES 提供的丰富服务集而特别适用于进行 80:20 部署。例如，使用由 Java Enterprise System 提供的服务进行企业范围通信系统或门户系统的部署相对来说比较容易。

对于需要定制开发的部署，Java Enterprise System 支持定制开发服务和应用程序的创建和集成。

第 17 页中的 “系统服务”中列出的多数服务类别都可用于提供 80:20 部署。例如，通信和协作服务可为最终用户提供电子邮件、日历和即时消息传送服务，使用户能够聚集和个性化内容。同样，可利用“网络身份管理”和“企业门户”服务类别安装和配置企业范围应用程序，而不必开发或集成各种定制服务。

对于需要定制开发 J2EE 平台服务的企业级解决方案，可以利用提供 Java Enterprise System Web 和应用程序服务的 Application Server、Message Queue 或 Web Server。

不同的企业部署所需的定制开发服务数量会有很大差异。由于各 Java Enterprise System 服务间可进行互操作，因此可根据企业的具体需要自行创建服务套件。

迁移到 Java Enterprise System

使用 Java Enterprise System 的企业解决方案的规划、设计和实现很大程度上取决于当前的部署策略。对于初次规划部署解决方案的企业，其规划、设计和实现主要由企业的具体需要驱动。但是，初次部署解决方案并不是很普遍。更多的解决方案是使用 Java Enterprise System 增强已有的企业解决方案或对 Java Enterprise System 组件的早期版本进行升级。

替换或升级现有解决方案时，必须采取额外的规划、设计和实现步骤，以确保保留现有数据并将软件顺利升级到当前版本。执行本指南中所述的分析和设计时，请勿忘记替换和升级现有软件系统所需的准备和规划工作。

有关升级到 Java Enterprise System 当前版本和从其他应用程序进行迁移的策略的更多信息，参阅 *Java Enterprise System* **升级与迁移指南**。

关于部署规划

部署规划是 Java Enterprise System 解决方案成功实现的关键步骤。每个企业都有其自身的一组目标、要求和优先级需要考虑。成功的规划开始于分析企业的目标，并确定满足这些目标的业务需求。然后将业务需求转变为技术要求，这些技术要求将用作设计和实现满足企业目标的系统的基础。

成功的部署规划是精心准备、分析和设计的成果。规划过程的任何环节出现错误或失策都可能导致系统在许多方面出现问题。系统规划不良可能引发严重的问题。例如，系统可能达不到性能要求、难于维护、运行成本高、浪费资源或无法扩展，无法适应不断增长的需要。

解决方案生命周期

下图所示的解决方案生命周期描述了基于 Java Enterprise System 的企业软件解决方案的规划、设计和实现步骤。生命周期是使部署项目不偏离轨道的有用工具。

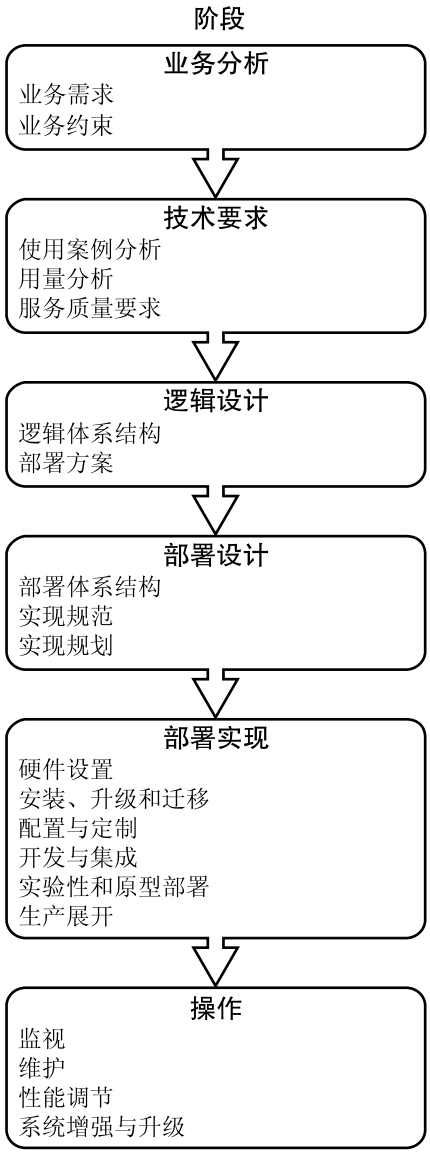


图 1–1 解决方案生命周期

生命周期由具有一定顺序的阶段组成。每个阶段都由相关任务组成，这些任务产生的输出将作为后续阶段的输入。每个阶段内的任务都是反复进行的，生成每个阶段的输出之前要求进行彻底的分析和设计。前面的阶段也可能反复进行。例如，在部署设计阶段，可能会发现前面阶段的分析不够充分，需要进行更多分析。

本章中的以下各节简要说明了生命周期的每个阶段。

业务分析阶段

业务分析阶段的任务是，定义部署项目的业务目标和阐述实现这些目标所必须满足的业务需求。阐述业务需求时，应将可能会对业务目标的实现能力产生影响的所有业务约束考虑在内。在整个解决方案生命周期中，业务分析阶段执行的分析将作为部署规划及最终的解决方案是否成功的标准。

在业务分析阶段创建业务需求文档，这些文档随后将用作技术要求阶段的输入。

有关业务分析阶段的更多信息，参阅第 2 章

技术要求阶段

技术要求阶段以业务分析阶段中形成的业务需求为起点，任务是将这些要求转化为可用来设计部署体系结构的技术规范。技术要求指定服务质量 (quality of service, QoS) 功能，如性能、可用性、安全性等。

在技术要求阶段要创建含有以下信息的文档：

- 用户任务和使用模式分析
- 模拟用户与规划系统间交互的使用案例
- 源自业务需求的服务质量要求，将用户任务和使用模式的分析结果考虑在内

使用分析、使用案例和 QoS 要求文档所产生的信息将作为解决方案生命周期逻辑设计阶段的输入。使用分析还在部署设计阶段发挥着重要的作用。

在技术要求阶段，可能还会指定作为随后创建的服务级别协议 (service level agreements, SLA) 基础的服务级别要求。服务级别协议规定为维护系统所必需提供的客户支持的条款，并且通常在部署设计阶段作为项目核准的一部分签署。

有关技术要求的更多信息，参阅第 3 章

逻辑设计阶段

在逻辑设计阶段中，以技术要求阶段的使用案例作为输入信息，确定实现解决方案所必需的 Java Enterprise System 组件。还需要确定对那些 Java ES 组件提供支持的组件，以及满足业务需求所必需的任何附加定制开发组件。然后在显示组件间相互关系的逻辑体系结构中映射组件。逻辑体系结构并不指定实现解决方案所需的硬件。

逻辑设计阶段的输出是逻辑体系结构。仅有逻辑体系结构还不足以开始部署设计，还需要来自技术要求阶段的 QoS 要求。逻辑体系结构和来自技术要求阶段的 QoS 要求形成一个部署方案。该部署方案是部署设计阶段的输入。

有关逻辑设计的更多信息，参阅第 4 章

部署设计阶段

在部署设计阶段，将逻辑体系结构中指定的组件映射到物理环境，从而生成一个高级部署体系结构。还要创建一个实现规范，该规范提供关于如何构建部署体系结构的初级详细信息。另外，还要创建一系列详细说明实现软件解决方案不同方面的规划和规范。

项目核准出现在部署设计阶段。在项目核准阶段对部署的成本进行评估。如果获得核准，将签署部署实现合同，并获取构建项目所需的资源。通常，在制定了详细的实现规范后才开始项目核准。但是，也可在部署体系结构完成后进行核准。

部署设计阶段的输出包括：

- **部署体系结构。**表示组件与网络硬件和软件间映射关系的高级设计文档。
- **实现规范。**用作构建部署蓝图的详细规范。
- **实现规划。**介绍实现企业软件解决方案各个方面的一组计划和规范。实现规划包括迁移规划、安装规划、用户管理规划、测试规划等。

有关部署设计的更多信息，参阅第 5 章

实现阶段

在实现阶段，从部署设计阶段创建的规范和规划着手，构建部署体系结构，并实现该解决方案。此阶段包括以下部分或全部任务，具体包含的任务取决于部署项目的性质：

- 安装和配置硬件基础结构
- 安装和配置软件
- 在 LDAP 目录设计中模拟用户和资源
- 根据用户管理规划从现有目录和数据库中迁移数据
- 在测试环境中创建和部署试验性和原型部署
- 设计和运行功能性测试来衡量与系统要求的符合度
- 设计和运行负载测试来衡量峰值负载下的性能
- 开发和集成定制企业应用程序
- 创建生产部署，可能需要分阶段部署到生产中

  部署进入生产阶段后，即可进入解决方案生命周期的操作阶段。

有关实现阶段的更多信息，参阅第 6 章

操作阶段

操作阶段包括确保实现部署顺利运行的必需任务。此阶段包括以下任务：

- 监视部署以确保系统按规划运行
- 性能调整以确保部署的软件处于最佳运行水平

- 提供平稳运行所需的定期维护和根据需要进行的随机维护
- 根据需要升级软件和硬件

有关操作阶段的更多信息不在本指南讨论范围之内。

第 2 章

业务分析

在解决方案生命周期的业务分析阶段，通过分析业务问题和确定实现业务目标的业务需求及业务约束来定义业务目标。

本章包括以下部分：

关于业务分析

业务分析的第一步是说明业务目标。接下来是分析必须解决的业务问题和确定实现业务目标所必须满足的业务需求。还应考虑会限制目标实现能力的任何业务约束。对业务需求和业务约束的分析将生成一组业务需求文档。

以生成的业务需求文档组为基础在技术要求阶段衍生出技术要求。在整个解决方案生命周期中，业务分析阶段执行的分析将作为判断部署规划及最终的解决方案是否成功的标准。

定义业务需求

不存在可以确定所有业务需求的简单公式。应根据与需要软件解决方案的风险承担者之间的合作、您自身对该业务领域的了解并发挥创造性思维来确定业务需求。

本节介绍定义业务需求时需要考虑的一些因素。

设置业务目标

业务分析应明确阐述部署项目的目标。清晰的目标有助于突出设计决策的重心，从而防止项目误入歧途。将业务目标与当前业务进行对比也有助于确定设计决策。

范围

业务需求应阐明部署项目的范围。确保您的目标区域能够找到解决方案，避免提出目标不明确或根本无法达到的没有结果的要求。范围定义不明确会导致部署设计不能充分满足业务需要或浪费资源。

优先级

制定目标的优先级可以确保优先实现部署中最重要的方面。受资源限制，可能需要延缓或修改某些目标。例如，大规模、复杂的部署通常要求将解决方案分阶段实现。为使部署设计能够通过风险承担者验收，可能需要作出某些决策，而通过阐明优先级，可帮助制定决策。

关键特质

确定成功的决定性因素，以使风险承担者和设计者集中精力拟定最重要的标准。

增长因素

设定业务目标时，不仅应考虑组织的当前需要，还要预见这些需要长期的变化和增长趋势。没有人想要一个可能过早被淘汰的解决方案。

保险余量

解决方案的设计以业务分析阶段所作的假设为基础。由于各种原因（如数据不足、判断错误或意料之外的外部事件），这些假设可能不正确。应确保在业务目标和整体规划中预留一定的保险余量，以使设计的解决方案可以应对意外事件。

了解用户需求

进行必要的调查，了解解决方案的目标用户类型、他们的需要以及预期可为用户带来的益处。例如，下表提供了一种对用户进行分类的方法：

- 仅限当前员工
- 当前和先前员工

- 管理员
- 活动客户
- 所有客户
- 成员站点
- 公众
- 受限访问

向用户明确阐述预期益处有助于制定设计决策。例如，以下是解决方案可为用户提供的一些益处：

- 远程访问公司资源
- 企业协作
- 简化日常任务
- 使远程团队得以共享资源
- 提高生产率
- 最终用户自我管理

制定操作要求

操作要求表述为一组具有明确目标的功能性要求。通常，可为以下领域创建操作规范：

- 最终用户功能
- 缩短响应时间
- 可用性和正常运行时间
- 降低错误率
- 信息存档和保留

将操作要求表述为所有风险承担者都可理解的可量化条目。避免含糊不清的语句，如“充足的最终用户响应时间。”操作要求的示例如下：

- 可在停电 10 分钟内恢复服务
- 可重放最近 48 小时内的传送的入站消息
- 高峰期间，可在 60 秒内完成在线事务处理
- 高峰期间，可在四秒内完成最终用户身份验证

支持现有使用模式

将现有使用模式表述为可清晰量化的目标。以下问题可帮助确定上述目标。

- 当前服务的使用情况如何?
- 使用模式是什么（例如，偶尔使用、经常使用或过量使用）？
- 用户通常连接哪些站点?
- 用户一般发送多大的消息?
- 用户每天或每小时通常完成多少事务?

研究访问您服务的用户。用户何时访问现有服务以及访问持续时间等因素是确定您的目标的关键。如果以您组织的经验不能确定这些模式，可研究类似组织的经验。

了解企业文化

要求分析应考虑企业文化和政策的各个方面。未充分考虑企业文化因素，可能会导致无法接受或难以实现解决方案。

风险承担者

确定能够从提议解决方案的成功中获取既得利益的个人和组织。所有风险承担者均应积极参与业务目标和要求的确定。如果风险承担者不参与或不了解计划的更改，则计划可能存在严重缺陷。这样的风险承担者甚至可能阻碍部署的实现。

标准和策略

确保了解需要解决方案的组织的标准和策略。这些标准和策略可能会影响设计的技术层面、产品选择及部署方法。

示例之一是可能由人力资源部门或部门经理所有和控制的人员数据的保密性。另一个例子是公司的更改管理流程。更改管理策略会极大地影响解决方案的接受，并影响实现方法和时间表。

监管要求

监管要求的差异极大，具体要求取决于业务的性质。研究和了解任何可能影响部署的监管要求。许多公司和政府机构要求符合可访问性标准。部署全球解决方案时应考虑外国的法律和法规。例如，许多欧洲国家对存储个人信息有严格的控制。

安全

您确定的某些目标可能存在需要重视的隐性安全问题。明确解决方案所必需的具体安全目标。例如：

- 专有信息的授权访问
- 以基于角色的方式访问机密信息
- 远程位置间的安全通信
- 在本地系统上调用远程应用程序
- 与第三方企业和组织的安全交易
- 安全策略的执行

站点分布

站点的地理分布和站点间的带宽可以影响设计决策。另外，某些站点可能要求本地管理。

这些地理方面的考虑会增加项目的培训成本、复杂性等。清楚阐明由站点的地理分布而产生的要求。确定对设计成功至关重要的站点。

使用渐增式方法

通常将软件解决方案视为一个完整的综合系统。不过，实际运作中往往会将完整系统的部署分为几个可衡量的步骤，以渐增方式完成。

采用渐增式方法时，通常要设计一个路线图，其中给出通往最终的综合解决方案的重大事件点。此外，可能还必须考虑制定针对综合解决方案随后实施的各方面的短期计划。

这种渐增式方法具有以下优势：

- 能够根据业务增长所带来的要求变化做出调整。
- 可在向最终部署实现过渡的过程中充分利用现有基础结构。
- 能够适应资本支出要求。
- 能够充分发挥少量人力资源的作用。
- 能够为集成合作伙伴方案做准备。

了解服务级别协议

服务级别协议 (Service Level Agreement, SLA) 指定了最低性能要求以及未能满足此要求时必须提供的客户支持级别和程度。服务级别协议基于业务分析过程中定义的业务需求，在随后的技术要求阶段，这些业务需求将被指定为服务级别要求。SLA 在部署设计阶段中的项目核准期间签署。

应根据正常运行时间、响应时间、消息传送时间和灾难恢复等领域制定 SLA。SLA 应说明系统概述、支持组织的角色和责任、如何衡量服务级别、更改请求等项目。确定您的组织对系统可用性的预期是确定 SLA 范围的关键。

定义业务约束

业务约束在确定部署项目的性质上发挥着重要作用。部署设计取得成功的关键是在受已知业务约束的情况下找到可以满足业务需求的最佳方法。业务约束可以是财政限制、物理限制（例如，网络容量）、时间限制（例如，在诸如下个年度会议等重要事件之前完成）或任何预见成为影响业务目标实现的因素的其他限制。

本节介绍了定义业务约束时需要考虑的几个因素。

迁移问题

通常，部署项目是对现有软件基础结构和数据的替换或补充。任何新的解决方案必须可将数据和过程从现有基础结构迁移到新的解决方案，并且通常保留与现有应用程序的互操作性。对现有基础结构的分析是确定迁移问题在提议解决方案中的作用程度的必要条件。

时间表要求

实现解决方案的时间表会影响设计决策。进取性的时间表可能导致降低目标、更改优先级或采用渐增式解决方法。时间表内可能还存在值得考虑的重大事件点。可以通过内部事件（如预定的服务展开）或外部事件（如学校某个学期的开学日期）设置重大事件点。

预算限制

大多数部署项目都必须遵守预算。构建提议解决方案的成本以及在特定生命周期内维护该解决方案所需的考虑事项包括：

- **现有硬件和网络基础结构。**对现有基础结构的依赖会影响系统的设计。
- **实现解决方案所需的开发资源。**如果开发资源（包括硬件、软件和人力资源）有限，便可能意味着要进行渐增式部署。可能必须在每个实现的渐增阶段重复使用相同的资源或开发团队。
- **维护、管理和支持。**分析可用于管理、维护和支持系统用户的可用资源。如果此类资源有限，可能会影响所作的设计决策。

拥有成本

除维护、管理和支持外，还应分析影响拥有成本的其他因素。例如，可能必要的硬件和软件升级，解决方案对电力网的影响、电信成本及影响现金支出的其他因素。指定解决方案可用性级别的服务级别协议也会要求增加冗余，从而影响拥有成本。

解决方案的实现应为该解决方案的投资带来收益。投资收益分析通常涉及资本支出所获财务收益的计算。

估算解决方案的财务收益涉及对要实现的目标与以其他方法实现这些目标，及与保持现状所需成本的详细比较分析。

第 3 章

技术要求

在解决方案生命周期的技术要求阶段，执行用量分析、确定使用案例并为提议的部署解决方案确定服务质量要求。

本章包括以下部分：

关于技术要求

技术要求分析以解决方案生命周期的业务分析阶段所创建的业务需求文档为起点。需要以这些业务分析为基础，执行下列步骤：

- 执行用量分析，以协助确定预计负载情况。
- 创建模拟用户与系统间交互的使用案例。
- 创建一组服务质量 (quality of service, QoS) 要求，用于定义部署的解决方案在响应时间、可用性、安全性及其他领域必须达到的性能。

  这些服务质量要求源自用量分析和使用案例，并考虑到先前确定出的业务需求和约束。

  服务质量要求随后要与逻辑设计阶段的逻辑体系结构相对应，以形成部署方案。该部署方案是解决方案生命周期部署设计阶段的主要内容。

  与业务分析阶段一样，技术要求分析阶段也不存在可生成用量分析、使用案例和系统要求的简单公式。进行技术要求分析要求对业务领域、业务目标及基础系统技术有所了解。

用量分析

用量分析的任务是确定要设计的解决方案的各个用户及确定这些用户的使用模式。收集的信息可用于估计系统的负载情况。在给使用案例指定加权时，用量分析信息同样有用，如第 33 页中的 “使用案例”中所述。

用量分析阶段的任务是：尽可能与用户面谈，对与使用模式有关的现有数据进行研究及与先前系统的制造者和管理员会谈。下表列出了进行用量分析时应考虑的因素。

表 3–1 用量分析因素

主题	说明
用户数量及类型	确定解决方案必须支持的用户数量，并在必要时对用户进行分类。 例如： ■ “企业对客户”(Business to Customer, B2C) 解决方案可能会有大量访问者，但可能只有少数用户会进行注册并参与商业交易。 ■ “企业对员工”(Business to Employee, B2E) 解决方案通常为每位员工提供服务，尽管有些员工可能需要从公司网络外部进行访问。在 B2E 解决方案中，经理可能需要获得普通员工不能访问的某些区域的访问权限。
活动和非活动用户	确定活动和非活动用户的使用模式和使用比率。 活动用户是指登录系统并与系统的服务进行交互的用户。非活动用户是指未登录的用户、虽然登录但不与系统组件进行交互的用户、或虽然数据库中存在该用户但从不登录的用户。
管理用户	确定对部署系统进行访问，以对部署进行监控、更新和支持的用户。 确定可能影响技术要求的任何特定管理使用模式（例如，从防火墙外部管理部署）。
使用模式	确定各类用户如何访问系统，并提供预期用量目标。 例如： ■ 是否存在因用量高涨而产生的高峰期? ■ 正常营业时间是什么? ■ 用户是否遍布全球? ■ 预计的用户连接持续时间是多少?
用户增长	确定用户群体规模是否固定，或部署是否预期有用户数量增长。 如果预期用户群体会有增长，请尽量作出合理的增长预测。

表 3–1 用量分析因素 （续）

主题	说明
用户事务	确定必须支持的用户事务类型。可将这些用户事务转化为使用案例。 例如： ■ 用户会执行哪些任务? ■ 用户登录后是否保持登录状态？他们通常会执行一些任务，然后注销? ■ 用户间的重要协作是否需要利用共用日历、召开 Web 会议及部署内部 Web 页?
用户研究和统计数据	利用现有用户研究和其他资源来确定用户行为模式。 企业或行业组织往往都会进行一些用户研究，可以从中汲取有用的用户信息。现有应用程序的日志文件中可能会包含一些统计数据，这些数据在做系统估量时会用到。

使用案例

使用案例模拟典型的用户与要设计的解决方案间的交互，以最终用户的视角说明操作的完整流程。在设计中对整个一组使用案例给予优先考虑可确保设计不偏离提供预期功能这一中心。使用案例是逻辑设计的主要输入。

为使用案例指定相对加权，加权最高的使用案例代表最常见的用户任务，这是一种常见的做法。为使用案例指定加权可让您将设计决策集中到使用最多的系统服务上。

可分两级对使用案例进行说明。

- **使用案例报告。**对各种使用案例（包括主要及备用事件流）的说明。
- **使用案例图。**描述参与者与使用案例间关系的示意图，提出较正式的事件流的组织方式。使用案例图有助于模拟长期或复杂的使用案例。通常使用“统一建模语言”(Unified Modeling Language, UML) 标准绘制使用案例图。

服务质量要求

服务质量 (quality of service, QoS) 要求是指定某些功能（如性能、可用性、可伸缩性和可维护性）的系统特性的技术规范。QoS 要求由业务需求阶段指定的业务需要驱动。例如，如果服务必须全年每天 24 小时可用，则可用性要求必须解决此业务需求。

下表列出了通常构成 QoS 要求基础的系统特性。

表 3–2 影响 QoS 要求的系统特性

系统质量	说明
性能	按用户负载条件对响应时间和吞吐量所做的度量。
可用性	对最终用户可访问系统资源和服务的频度的度量，通常以系统的**正常运行时间**来表示。
可伸缩性	随时间推移为部署系统增加容量（和用户）的能力。可伸缩性通常涉及向系统添加资源，但不应要求对部署体系结构进行更改。
安全	对系统及其用户的完整性进行说明的复杂因素组合。安全性包括用户的验证和授权、数据的安全以及对已部署系统的安全访问。
潜在容量	在不增加资源的情况下，系统处理异常峰值负载的能力。潜在容量是可用性、性能和可伸缩性特性中的一个因素。
可维护性	对已部署系统进行维护的容易度，其中包括监视系统、修复出现的故障以及升级硬件和软件组件。

各系统特性密切相关。对一个系统特性的要求可能会影响到对其他系统特性的要求和设计。例如，提高安全性级别可能会影响到性能，而性能又会影响到可用性。靠另外增加服务器来应对可用性问题会影响可维护性（维护成本）。

能否设计出既可满足业务需求，又能兼顾业务约束的系统的关键在于，了解各系统特性间的关联方式及必须作出的权衡。

以下各部分将对影响部署设计的各种系统特性做更深入的探讨，并就确定 QoS 要求时应考虑的因素提供相关指导。还包括一节介绍服务级别要求的内容，该要求是服务级别协议的基础。

性能

业务需求通常用指定响应时间的非技术术语表示性能。例如，基于 Web 访问的业务需求可能会做以下说明：

正常情况下，用户登录后会有一段合理的响应时间，这段时间通常不超过四秒。

从该业务需求入手，对所有使用案例进行研究，以确定在系统级体现该要求的方式。某些情况下，您可能希望将用量分析期间确定的用户负载情况也包括在内。以指定负载情况下的响应时间或响应时间加吞吐量来表示每个使用案例的性能要求。可能还要指定容错数。

下面是有关如何指定系统性能要求的两个示例。

- 以 15 分钟为间隔采样的 Web 页刷新响应时间在全天各时段均不得超过四秒，每百万事务错误数须少于 3.4 个。
- 在定义的高峰期间，系统必须能够每秒接受 25 个安全登录，而且任何用户的响应时间均不得超过 12 秒，每百万事务错误数须少于 3.4 个。

性能要求与可用性要求（故障转移对性能的影响）及潜在容量（可用于处理异常峰值负载的容量）关系密切。

可用性

可用性是指定系统正常运行时间的一种方法，通常以系统可供用户访问的时间占总时间的百分比来表示。系统不可访问时间（停机时间）可能是因硬件、软件、网络故障或任何其他因素（如断电）所致，这些因素会使系统停机。不将计划的服务（维护和升级）停机时间视为停机时间。按正常运行时间百分比计算系统可用性的基本公式是：

```
可用性 = 正常运行时间/(正常运行时间 + 停机时间) * 100%
```

通常以“九”的个数来衡量可达到的可用性。例如，99% 的可用性为两个九。指定更多的九会对部署设计产生很大影响。下表显示的是为某系统增加代表可用性的九后的非计划停机时间数值，该系统以 24x7 方式全年不间断运行（共计 8,760 小时）。

表 3–3 全年运行（8,760 小时）系统的非计划停机时间

九的个数	可用性百分比	非计划停机时间
2	99%	88 小时
3	99.9%	9 小时
4	99.99%	45 分钟
5	99.999%	5 分钟

容错系统

对可用性的要求达四个或五个九通常要求系统必须是一个容错系统。容错系统必须能够在硬件或软件出现故障时继续运行。通常，容错的实现手段是为提供关键服务的硬件（如 CPU、内存和网络设备）及软件配置冗余组件。

单一故障点是指没有备用的冗余组件的硬件或软件组件，而这些组件是重要路径的组成部分。该组件出现故障会使系统无法继续提供服务。设计容错系统时，必须确定并消除潜在的单一故障点。

容错系统的实现和维护成本高昂。请确保先了解业务可用性要求的本质，然后再考虑能够满足这些要求的可用性解决方案的策略和成本。

排定服务可用性优先级

在用户看来，可用性更多牵涉到的往往是逐个服务的可用性，而非整个系统的可用性。例如，如果即时消息传送服务不可用，通常情况下对其他服务的可用性几乎没有影响或无任何影响。但是，如果许多其他服务所依赖的服务不可用（如 Directory Server），则会有较大影响。较高的可用性规范应该明确引用要求更高可用性的特定使用案例和用量分析。

根据一组有序的优先级列出可用性需求会有帮助。下表按优先级顺序列出了不同服务类型的可用性。

表 3–4 不同优先级服务的可用性

优先级	服务类型	说明
1	关键任务	任何时候必须可用的服务。例如，应用程序的数据库服务（如 LDAP 目录）。
2	必须可用	必须可用，但可以较低性能获得的服务。例如，在某些业务环境中，消息传送服务的可用性可能并不关键。
3	可延迟	在特定时间段内必须可用的服务。例如，在某些业务环境中，日历服务的可用性可能并非不可或缺。
4	可选	可无限期延迟的服务。例如，在某些环境中，即时消息传送服务可能有用，但非必需。

服务丢失

可用性设计将考虑到可用性降低或组件丢失时所发生的情况。其中，要考虑连接的用户是否必须重新启动会话和一个区域内的故障对系统的其他区域的影响程度。QoS 要求应考虑这些方案并指定部署如何对这些情况作出反应。

可伸缩性

可伸缩性是增加系统容量的能力，从而使系统可以支持来自现有用户或扩大的用户群体的额外负载。可伸缩性通常要求增加资源，但不应要求对部署体系结构的设计进行更改，或因增加额外资源需要时间而中断服务。

与可用性一样，可伸缩性更多牵涉到的是系统所提供的各项服务，而非整个系统。不过，对于其他服务所依赖的服务（如 Directory Server），可伸缩性的影响可能会波及整个系统。

不必在 QoS 要求中指定可伸缩性要求，除非业务需求中对预测的部署增长做了明确说明。不过，在解决方案生命周期部署设计阶段，即使未指定可用性 QoS 要求，部署体系结构也应添加一定余量，用于扩展系统规模。

估计增长

估计系统的增长，确定可用性要求，做一些可能无法达成的预测、估计和推测。以下是开发可伸缩系统的三个关键点。

- **高性能设计策略。**在性能要求的确定阶段加入潜在容量，以处理可能会随时间推移而增长的负载。还要在预算限度内尽可能提高可用性。采用这一策略可使系统能够承担增长的负载，并可更从容地制订系统扩展的重大事件点。

- **渐增式部署。**采用渐增式部署有助于资源增加计划的制订。指定明确的系统扩展重大事件点。重大事件点通常是基于负载的要求以及评估可伸缩性的特定日期。
- **大范围性能监视。**对性能进行监视有助于确定向部署中增加资源的时机。监视性能的要求可为负责维护和升级的操作员和管理员提供指导。

下表列出确定可伸缩性要求应考虑的一些因素。

表 3–5 可伸缩性因素

主题	说明
分析使用模式	通过研究现有数据了解当前（或预测）用户群体的使用模式。如果缺少现时数据，可对行业数据或市场估计进行分析。
以最大的合理标度为目标进行设计	设计时以能够满足已知和潜在需求的最大必需标度为目标。 这往往是根据现有用户负载的性能评估和对未来负载的合理预期而作出的 24 个月估计。估计周期的长短在很大程度上取决于预测的可靠性。
设置合适的重大事件点	以渐增方式实现部署设计来满足短期要求，同时设立缓冲区来应对意外增长。设置增加系统资源的重大事件点。 例如： ■ 资本收购（如每季度或每年） ■ 购买硬件和软件的提前时间（如，一到六个星期） ■ 缓冲区（10% 到 100%，具体取决于增长预期）
融入新兴技术	了解新兴技术（如速度更快的处理器和服务器），及此技术会对基础体系结构的性能产生怎样的影响。

安全性要求

安全性是一个复杂的主题，涉及到部署系统的各个级别。开发安全要求围绕确定安全威胁和开发解决它们的策略进行。此安全分析包括以下步骤：

1. 确定关键资产
2. 确定对这些资产的威胁
3. 确定使组织暴露于可能带来风险的威胁的薄弱环节
4. 开发减轻组织风险的安全规划

分析安全要求应由您的组织的各方面风险承担者参与，包括管理员、业务分析师和信息技术人员。通常，组织会指定一个安全结构设计师来领导安全措施的设计和实现。

以下各节介绍安全规划包括的一些领域。

安全规划元素

规划系统安全是部署设计的一部分，对于设计的成功实现至关重要。规划安全时请考虑以下几点：

- **物理安全。**物理安全是对路由器、服务器、服务器机房、数据中心及基础结构中其他部分的物理访问。如果未经授权的人可以进入服务器机房然后拔掉路由器电源，则其他安全措施将毫无意义。
- **网络安全。**网络安全是通过防火墙、安全访问区、访问控制列表和端口访问对网络进行访问。对于网络安全，应开发针对未授权访问、篡改和拒绝服务 (denial of service, DoS) 攻击的策略。
- **应用程序和应用程序数据安全。**应用程序和应用程序数据安全包括通过验证和授权过程及策略访问用户帐户、公司数据和企业应用程序。此领域包括定义下列策略：
  - 密码策略
  - 访问权限，如对用户的委派管理（区别于管理员访问）
  - 帐户灭活
  - 访问控制
  - 加密策略，包括数据安全传输和使用证书签署数据
- **个人安全惯例。**组织范围的安全策略，定义工作环境和所有用户必须遵守的惯例，以确保其他安全措施按设计实行。通常的做法是编印安全手册并对用户进行安全惯例培训。要实现有效的总体安全策略，可靠的安全惯例必须成为企业文化的组成部分。

潜在容量

潜在容量是指在不增加资源的情况下，部署处理异常峰值负载用量的能力。通常不直接围绕潜在容量定义 QoS 要求，但该系统特性是确定可用性、性能和可伸缩性要求的一个因素。

可维护性要求

可维护性是指对部署系统进行维护的容易度，其中包括监视系统、修复出现的故障、向系统添加用户、从系统删除用户及升级硬件和软件组件等任务。

计划可维护性要求时应对下表所列的主题予以考虑。

表 3–6 可维护性要求主题

主题	说明
停机时间计划	确定必须使特定服务完全或部分不可用的维护任务。 某些维护和升级任务可在用户不知不觉中完成，而要执行其他类似任务则必须中断服务。尽可能与用户一起为要求停机的维护活动制订计划，使用户能够为停机时间作出准备。

表 3–6 可维护性要求主题　（续）

主题	说明
使用模式	确定使用模式，以确定适于安排维护的最佳时间。 例如，在通常于正常营业时间内出现用量高峰的系统中，适合在傍晚或周末执行维护任务。对于在地域上分散的系统而言，确定这些时段的难度可能更大。
可用性	可维护性往往是可用性设计思想的反映，因为最大限度缩短维护和升级停机时间策略是围绕可用性策略来制订的。要求具有高可用性的系统的维护、升级和检修停机时间较短。 处理可用性要求的策略会影响到对维护和升级的处理方式。例如，在地域上分散的系统中，维护方式可能取决于维护期内将工作负载发送到远程服务器的能力。 要求具有高可用性的系统可能还需要采用更为复杂的解决方案，使系统重新启动可以自动进行，而几乎不需要人为介入。
诊断和监视	定期运行诊断和监视工具来确定故障区域，可以提高系统的稳定性。 定期监视系统可以防患于未然，有助于根据可用性策略来平衡工作负载及改进维护和停机计划。

服务级别要求

服务级别协议 (service level agreement, SLA) 指定了最低性能要求以及未能满足此要求时必须提供的客户支持级别和程度。服务级别要求是指定 SLA 所基于的条件的系统要求。

与 QoS 要求一样，服务级别要求源自业务需求，并代表着对部署系统必须达到的整体系统特性的担保。服务级别协议被视为合同，所以必须明确规定服务级别要求。服务级别要求对要求的测试条件及不合要求的构成条件均有明确规定。

第 4 章

逻辑设计

在解决方案生命周期的逻辑设计阶段，设计指定解决方案逻辑组件间相互关系的逻辑体系结构。逻辑体系结构和来自技术要求阶段的使用分析一起组成了部署方案，作为部署设计阶段的信息来源。

本章包括以下部分：

关于逻辑体系结构

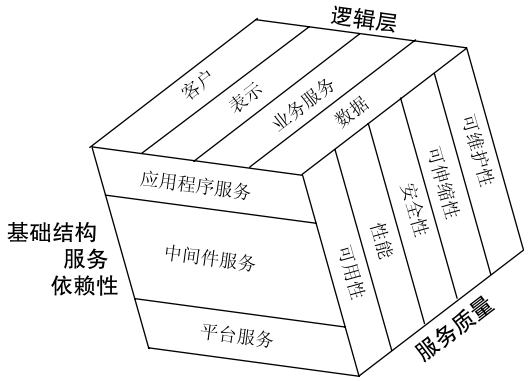
逻辑体系结构确定实现解决方案所需的软件组件，显示组件间的相互关系。逻辑体系结构和在技术要求阶段确定的服务质量要求一起组成一个部署方案。部署方案是进行下一阶段，即部署设计阶段，部署体系结构设计的基础。

开发逻辑体系结构时，不仅需要确定向用户提供服务的组件，还要确定提供必要中间件和平台服务的其他组件。基础结构服务依赖性和逻辑层提供了两种执行此分析的补充方法。

基础结构服务依赖性和逻辑层是 Sun Java™ Enterprise System 基于的解决方案体系结构三维中的两维。此三维在下面列出，同时显示在第 41 页中的 “关于逻辑体系结构”中。

- **基础结构服务依赖性。**提供企业服务的交互软件组件。这些软件组件需要一套允许分布式组件间相互通信和交互操作的底层基础结构服务。
- **逻辑层。**基于软件组件所提供服务的性质，表示软件组件逻辑和物理独立性的软件组件逻辑组织层。

- **服务质量。**系统服务质量，如性能、可用性、可伸缩性及其他体现软件解决方案设计和操作具体方面的要素。



注 – 有关 Java Enterprise System 体系结构概念的更多信息，参阅《Sun Java Enterprise System 2005Q4 技术概述》的“Java Enterprise System 体系结构”一章。

逻辑体系结构通过显示必要组件及其依赖性说明基础结构服务级别。逻辑体系结构还在逻辑层间分配组件。这些逻辑层表示可被客户层访问的表示、业务和数据服务。服务质量虽未在逻辑体系结构中建立模型，但与部署方案中的逻辑体系结构相对应。

设计逻辑体系结构

设计逻辑体系结构时，使用在技术要求阶段确定的使用案例来确定提供解决方案所需服务的 Java Enterprise System 组件。还必须确定向最初确定组件提供服务的所有组件。

根据 Java Enterprise System 组件所提供服务的类型，将其放在多层体系结构的上下文环境中。组件是多层体系结构的组成部分，这种理解有助于您确定对组件所提供的服务进行分布的方法，还有助于确定实现服务质量（如可伸缩性、可用性等）的策略。

另外，还可提供另一个将逻辑组件放在安全访问区内的逻辑组件视图。第 56 页中的 “访问区”一节中提供了一个安全访问区示例。

Java Enterprise System 组件

Java Enterprise System 由提供企业服务的交互软件组件组成，可用于构建企业解决方案。下图显示的是 Java Enterprise System 提供的关键软件组件。《Sun Java Enterprise System 2005Q4 技术概述》提供了关于 Java Enterprise System 组件及其所提供服务的更多信息。

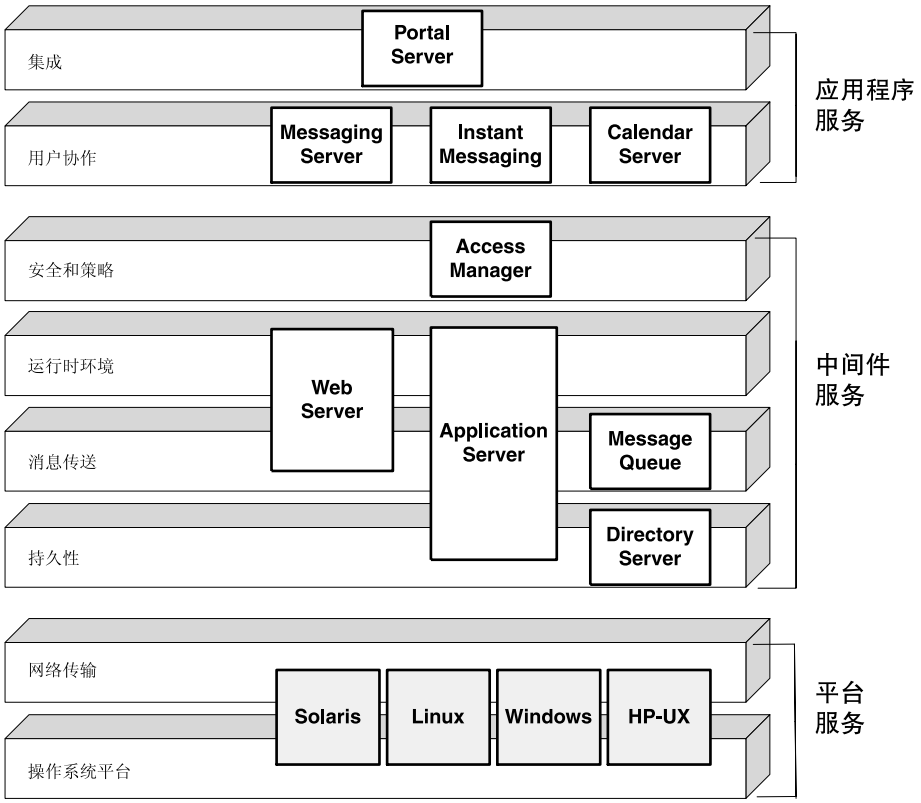


图 4–1 Java Enterprise System 组件

组件依赖性

确定逻辑体系结构的 Java Enterprise System 组件时，还需确定支持组件。例如，如果将 Messaging Server 确定为某个逻辑体系结构的必要组件，则该逻辑体系结构还必须含有 Directory Server 及可能含有 Access Manager。Messaging Server 依赖 Directory Server 提供目录服务，依赖 Access Manager 提供要求单点登录的解决方案。

下表列出了 Java Enterprise System 组件的依赖性。有关关键组件依赖性的图示，参阅第 43 页中的 “组件依赖性”。设计逻辑体系结构时，使用此表及所附数字确定设计中的依赖性组件。

表 4–1 Java Enterprise System 组件依赖性

Java Enterprise System 组件	所依赖的组件
Application Server	Message Queue
	Directory Server（可选）
Calendar Server	Messaging Server（用于电子邮件通知服务）
	Access Manager（用于单点登录）
	Web Server（用于 Web 接口）
	Directory Server
Communications Express	Access Manager（用于单点登录）
	Calendar Server
	Messaging Server
	Instant Messaging
	Web Server（用于 Web 接口）
	Directory Server
Directory Proxy Server	Directory Server
Directory Server	无
Access Manager	Application Server 或 Web Server
	Directory Server
Instant Messaging	Access Manager（用于单点登录）
	Directory Server
Message Queue	Directory Server（可选）
Messaging Server	Access Manager（用于单点登录）
	Web Server（用于 Web 接口）
	Directory Server

表 4–1 Java Enterprise System 组件依赖性　（续）

Java Enterprise System 组件	所依赖的组件
Portal Server	如果配置为使用 Portal Server 频道： Calendar Server Messaging Server Instant Messaging Access Manager（用于单点登录） Application Server 或 Web Server Directory Server
Portal Server Secure Remote Access	Portal Server
Web Server	Access Manager（可选，用于访问控制）

注 – 第 43 页中的 “组件依赖性” 中列出的 Java Enterprise System 组件间的依赖性并未列出全部的组件依赖性，其中未列出在规划安装时必须考虑的依赖性。有关 Java Enterprise System 依赖性的完整列表，参阅《Sun Java Enterprise System 2005Q4 安装指南》。

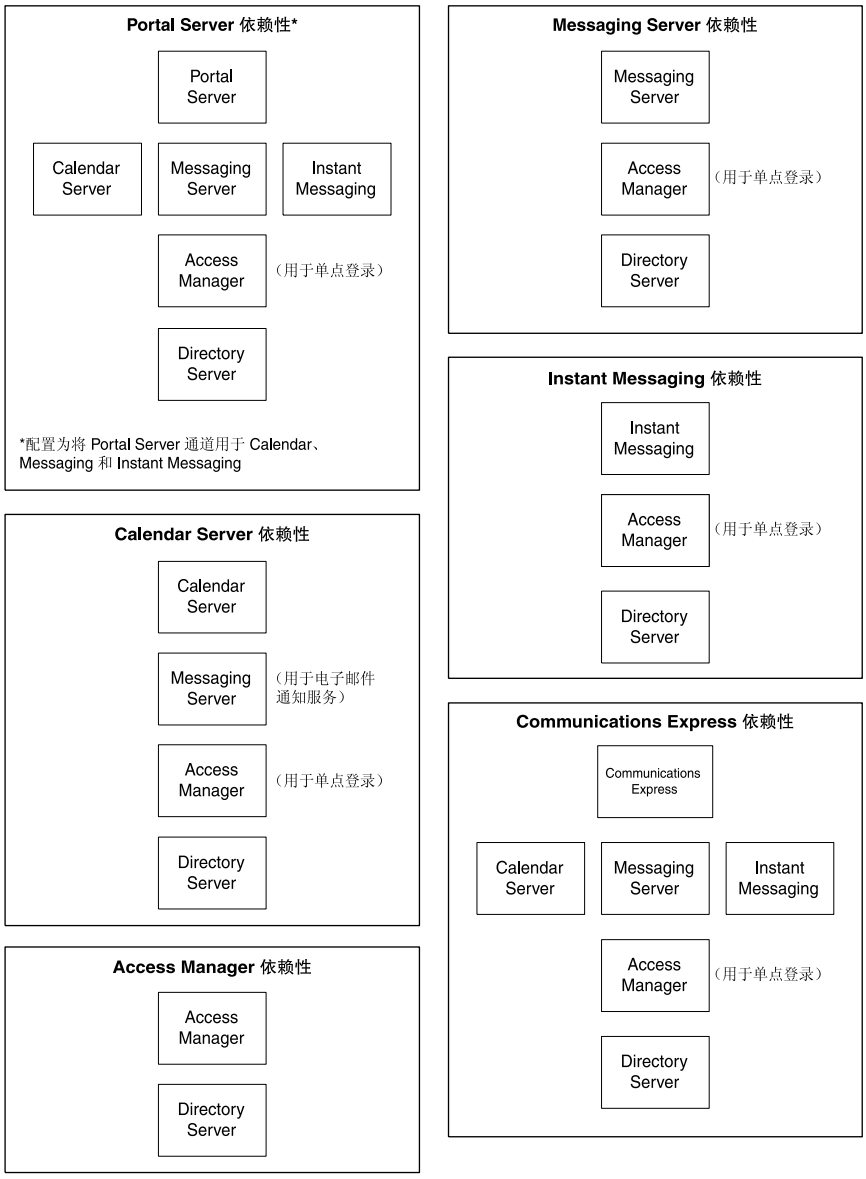


图 4–2 Java Enterprise System 组件依赖性

Web 容器支持

上一节第 43 页中的 “组件依赖性”中未提及 Portal Server 和 Access Manager 在其中运行的 Web 容器。此 Web 容器可由 Application Server、Web Server 或第三方产品提供。设计包含有 Portal Server 或 Access Manager 的逻辑体系结构时，请确保对这些组件所需的 Web 容器加以说明。

Messaging Server 提供的逻辑互异服务

Java Enterprise System Messaging Server 通过配置，可提供单独的实例，以提供以下逻辑互异服务：

- 消息传输代理
- 消息多路复用器
- Messenger Express 多路复用器
- 消息存储

Messaging Server 的这些不同配置提供了可在单独的物理服务器上部署并可在逻辑体系结构的不同层中表示的功能。由于 Messaging Server 的这些配置代表不同层中逻辑互异的服务，因此在设计逻辑体系结构时请将其视为逻辑互异组件。第 49 页中的 “逻辑体系结构示例”提供了一个逻辑互异组件的示例。

下表介绍了 Messaging Server 的逻辑互异配置。

表 4–2 Messaging Server 配置

子组件	说明
Message Transfer Agent (MTA)	支持通过处理 SMTP 连接发送电子邮件、路由电子邮件以及将消息传送到适当的消息存储。MTA 组件通过配置，可支持从企业外部发送（入站）或从企业内部发送（出站）电子邮件。
Message Store (STR)	提供电子邮件消息的检索和存储。
Message Multiplexor (MMP)	通过使用 IMAP 或 POP 协议访问电子邮件客户机的消息存储，支持电子邮件检索。
Messenger Express Multiplexor (MEM)	通过代表基于 Web (HTTP) 的客户机访问消息存储，支持电子邮件检索。

访问组件

Java Enterprise System 还含有提供对系统服务访问（通常从企业防火墙外部）的组件。Messaging Server 的某些配置还可提供网络访问，如为消息多路复用器配置的 Messaging Server。下表介绍了提供对系统服务远程访问的 Java Enterprise System 组件。

表 4–3 提供远程访问的 Java Enterprise System 组件

组件	说明
Directory Proxy Server	可为 Directory Server 实例提供增强的目录访问控制、模式兼容性、路由选择以及负载平衡。
Portal Server, Portal Server Secure Remote Access	提供从公司防火墙外部对 Portal Server 内容和服务（包括内部门户和 Internet 应用程序）的安全 Internet 访问。
Portal Server, Portal Server Mobile Access	提供从移动设备到 Portal Server 的无线访问和对 Portal Server 的语音访问。
Messaging Server Message Multiplexor (MMP)	通过代表基于 Web (HTTP) 的客户机访问消息存储，支持电子邮件检索。

提供远程访问的组件通常在安全访问区部署，如第 56 页中的 “访问区”一节中的示例所示。

多层体系结构设计

Java Enterprise System 非常适合多层体系结构设计。在多层体系结构中，服务根据其提供的功能放在不同层中。每个服务都是逻辑独立的，并且可由同层或不同层的服务访问。下图显示了企业应用程序的一个多层体系结构模型，介绍了客户层、表示层、业务服务层和数据层。

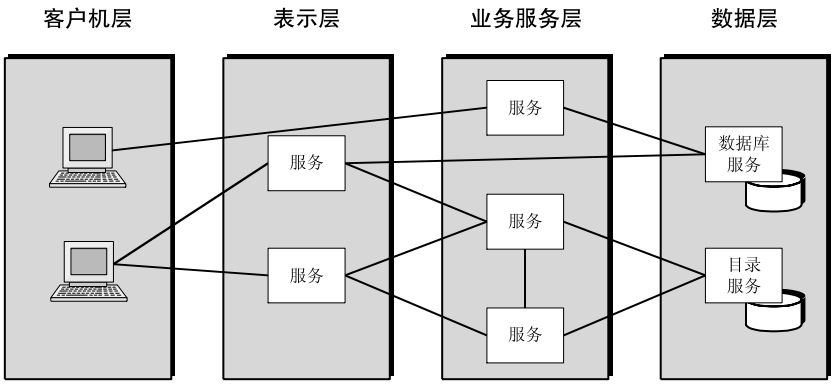


图 4–3 多层体系结构模型

下表是对第 48 页中的 “多层体系结构设计”中显示的逻辑层的说明。

表 4–4 多层体系结构中的逻辑层

层	说明
客户机层	含有向最终用户提供信息的客户机应用程序。对于 Java Enterprise System，这些应用程序通常是邮件客户机、Web 浏览器或移动访问客户机。
表示层	提供向最终用户显示数据的服务，允许用户处理和操作该表示。例如，Web 邮件客户机或 Portal Server 组件允许用户修改客户接收信息的表示。
业务服务层	提供后端服务，这些服务通常从数据层检索数据，以提供给表示层或业务服务层内的其他服务，或直接提供给客户层的客户机。例如，Access Manager 向其他 Java Enterprise System 组件提供身份认证服务。
数据层	提供由表示或业务服务层内的服务访问的数据库服务。例如，Directory Server 向其他服务提供 LDAP 目录访问。

多层体系结构设计具有若干优点。在部署设计阶段，根据多层体系结构中的功能布置服务有助于确定在网络中分配服务的方式。还可看到体系结构中的组件如何访问其他组件的服务。这种直观性有助于规划服务解决方案的可用性、可伸缩性、安全性和其他性质。

逻辑体系结构示例

本节提供了一些 Java Enterprise System 解决方案的逻辑体系结构示例。这些示例首先说明如何在多层体系结构的适当层布置逻辑组件，然后通过研究使用案例，分析组件间的关系。将本节中的逻辑体系结构示例用作了解 Java Enterprise System 解决方案中逻辑体系结构设计的基础。

第一个示例是一个基本的 Messaging Server 解决方案，说明了 Messaging Server 的逻辑互异组件如何与其他组件交互操作。第二示例说明一个基于身份的部署解决方案的逻辑体系结构，该方案可适用于拥有 1,000 到 5,000 名员工的中型企业。

Messaging Server 示例

下图说明了一个 Messaging Server 部署的基本逻辑体系结构。该逻辑体系结构仅显示 Messaging Server 要求的逻辑互异组件。后面的图中介绍了这些组件的关系。

注 – 通常，Messaging Server 的部署是包括其他 Java Enterprise System 组件的企业解决方案的组成部分，如第 53 页中的 “基于身份的通信示例”中所示。

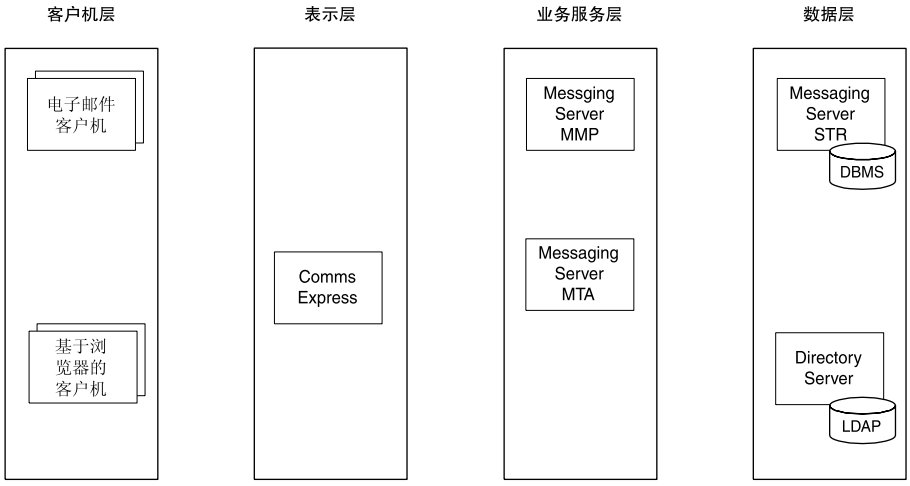


图 4–4 Messaging Server 部署的逻辑体系结构

下表介绍了第 49 页中的 “Messaging Server 示例”中显示的组件。

表 4–5 Messaging Server 逻辑体系结构中的组件

组件	说明
电子邮件客户机	用于阅读和发送电子邮件的客户机应用程序。
Messaging Server MTA	Messaging Server 配置为消息传送代理 (Message Transfer Agent, MTA)，以接收、路由、传输和发送电子邮件消息。
Messaging Server MMP	Messaging Server 配置为消息多路复用器 (Message Multiplexor, MMP)，以路由至合适的消息存储的连接，进行检索和存储。MMP 访问 Directory Server，查找目录信息，以确定适合的消息存储。
Messaging Server STR	Messaging Server 配置为消息存储，以进行电子邮件消息的检索和存储。
Directory Server	提供对 LDAP 目录数据的访问。

逻辑体系结构不为 Messaging Server 组件指定服务复制。例如，企业部署通常创建单独的入站和出站 MTA 实例，但第 49 页中的 “Messaging Server 示例”只显示一个 MTA 组件。将逻辑组件复制到多个实例是在部署设计阶段所作的设计决策。

Messaging Server 使用案例

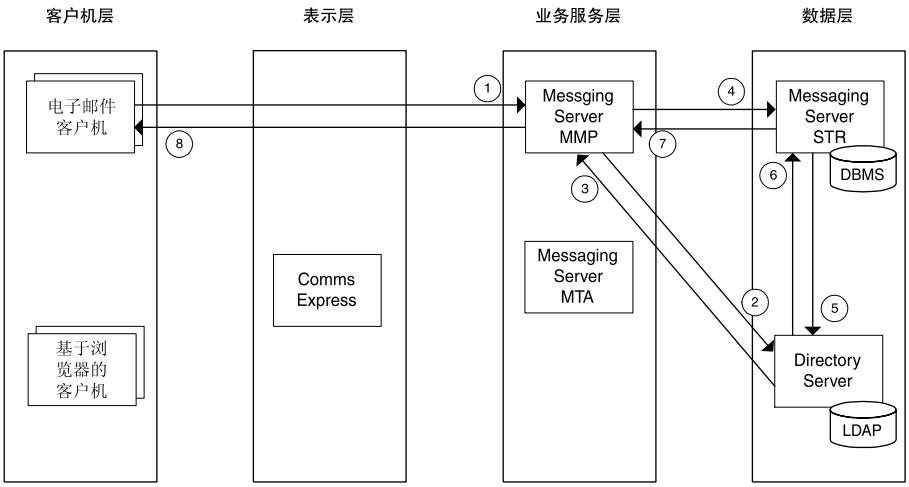
使用案例帮助确定体系结构中的逻辑组件间的关系。根据使用案例映射组件间的交互，从而获得一个有助于部署设计的组件交互视图。

通常，在部署设计之前分析每个使用案例，以确定组件的交互。下述三个使用案例是 Messaging Server 的典型使用案例，显示了逻辑组件间的交互。

▼ 使用案例 1：用户成功登录到 Messaging Server

步骤

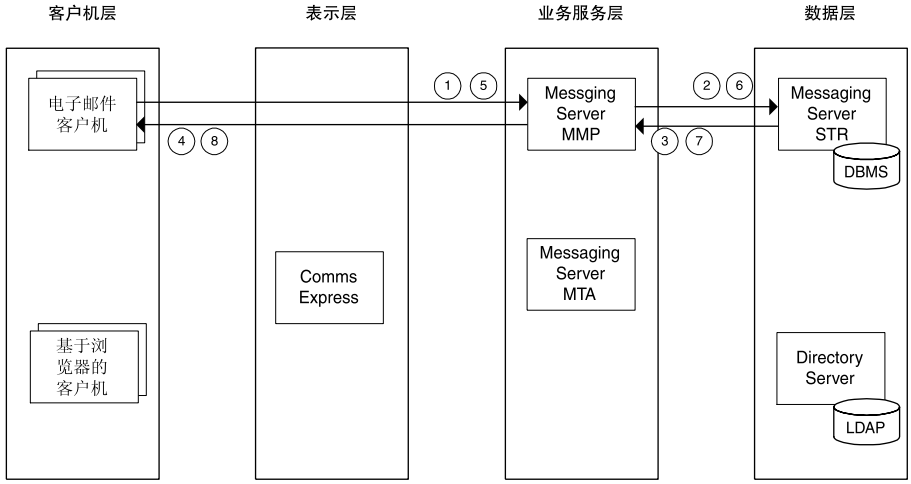
1. **电子邮件客户机将登录信息发送到 Messaging Server Multiplexor (MMP)。**
2. **MMP 向 Directory Server 请求用户 ID 和密码验证。**
3. **Directory Server 将验证返回给 MMP。**
4. **MMP 向 Messaging Server Message Store (STR) 请求消息列表。**
5. **STR 向 Directory Server 请求用户的 LDAP 记录。**
6. **Directory Server 将用户的 LDAP 记录返回给 STR。**
7. **STR 将消息列表返回给 MMP。**
8. **MMP 将消息列表转发给电子邮件客户机。**



▼ 使用案例 2：登录用户阅读和删除邮件

步骤 1. **电子邮件客户机向 Messaging Server Multiplexor (MMP) 请求要阅读的消息。**

2. **MMP 向 Messaging Server Message Store (STR) 请求消息。**
3. **STR 将消息返回给 MMP。**
4. **MMP 将消息转发给电子邮件客户机。**
5. **电子客户机将删除消息操作发送给 MMP。**
6. **MMP 将删除消息操作转发给 STR。**
7. **STR 将消息从数据库中删除，然后将确认发送给 MMP。**
8. **MMP 将删除确认转发给电子邮件客户机。**

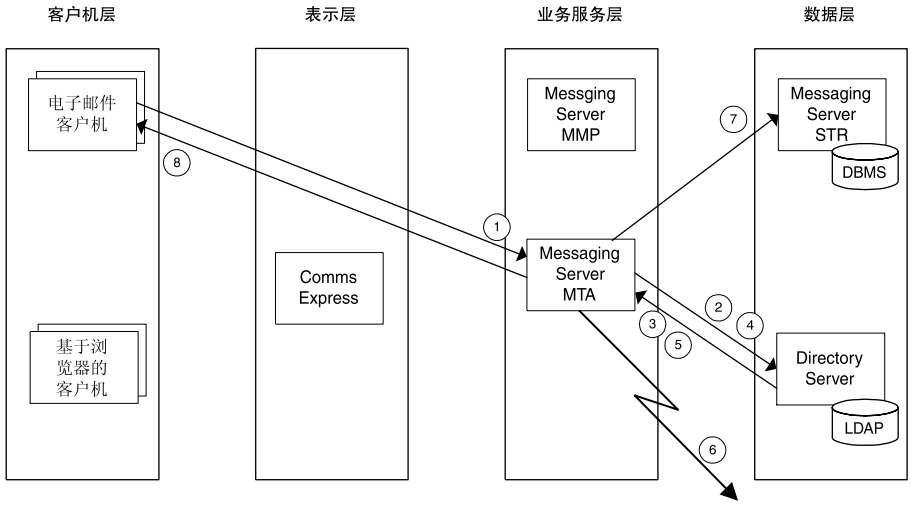


▼ 使用案例 3：登录用户发送电子邮件消息

步骤 1. **电子邮件客户机将在客户机中编写的消息发送给 Messaging Server Message Transfer Agent (MTA)。**

2. **MTA 向 Directory Server 请求用户 ID 和密码验证。**
3. **Directory Server 将验证返回给 MTA。**
4. **MTA 检查 Directory Server，获取每个收件人的目标域。**

5. **Directory Server 将每个收件人的目标域返回给 MTA。**
6. **MTA 将消息转发给每个收件人。**
7. **MTA 将消息转发给 Messaging Server Message Store (STR)，在发件箱中存储消息。**
8. **MTA 将确认发送给电子邮件客户机。**



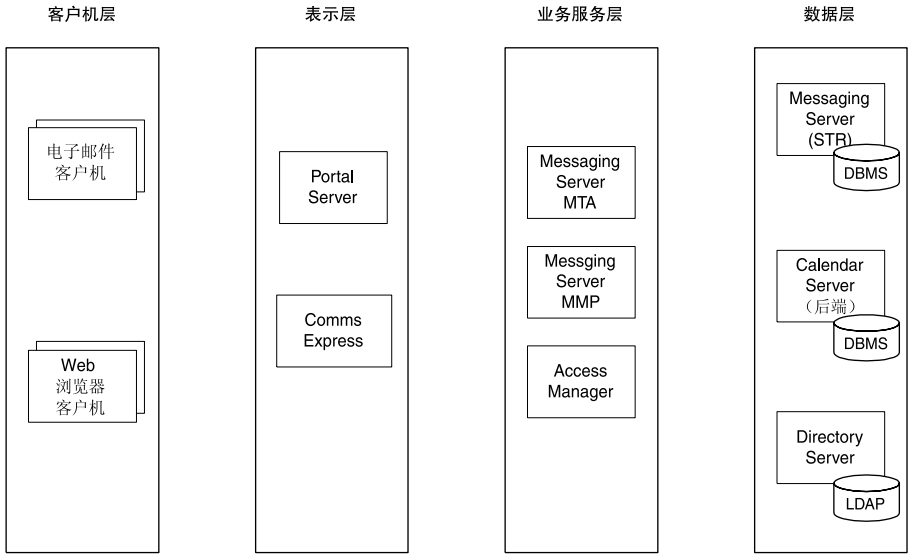
基于身份的通信示例

此示例说明了面向拥有约 1,000 至 5,000 名员工的中型企业的基于身份的通信解决方案。通常，设计逻辑体系结构时需要彻底的业务分析及随后的详细的技术要求分析。但是，这是一个理论性示例，假定已确定以下业务需求：

- 企业员工需要对内部站点、通信服务、日历服务及其他资源的个性化访问。
- 企业范围的验证和授权提供对内部站点和其他服务的访问。
- 可跨越所有企业服务对单个身份进行跟踪，从而启动了可提供对内部站点和其他服务访问的单点登录 (single sign-on, SSO)。

此示例的使用案例将详细说明登录程序、阅读电子邮件、发送电子邮件、个性化门户、同步日历以及其他类似用户活动。

下图说明了此种基于身份的通信解决方案的逻辑体系结构。



基于身份的通信示例的使用案例

对于此种性质的部署解决方案，通常存在大量概括说明用户与该解决方案所提供服务的交互作用的详细使用案例。此示例着重说明用户从浏览器客户机登录门户时组件间的交互。此示例将登录方案分为以下两个使用案例：

- 用户登录，通过验证，然后 Portal Server 检索用户的门户配置。
- Portal Server 检索要在 Web 客户机中显示的电子邮件和日历信息。

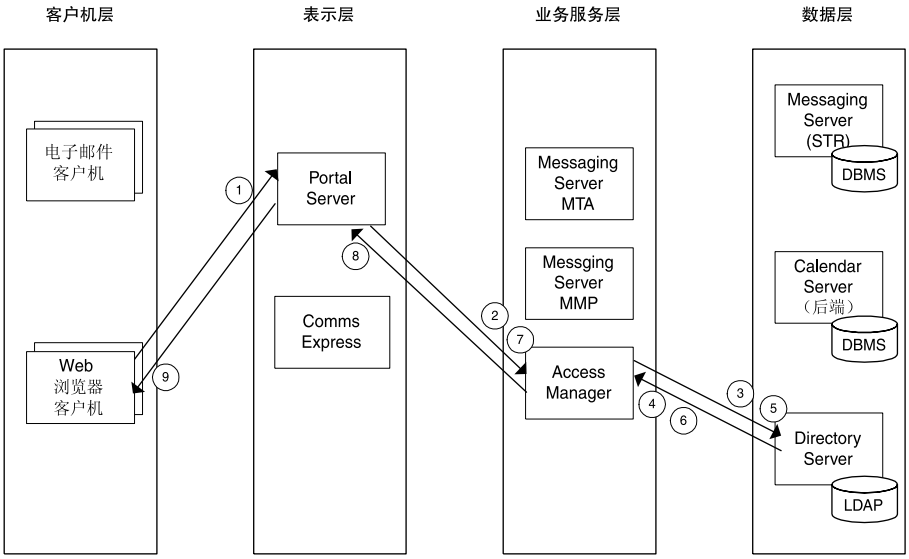
可将这两个使用案例视为一个扩展使用案例。但是在此示例中，为简单起见，单独对两个使用案例进行说明。

▼ 使用案例 1：用户成功登录后，门户检索用户的配置

步骤

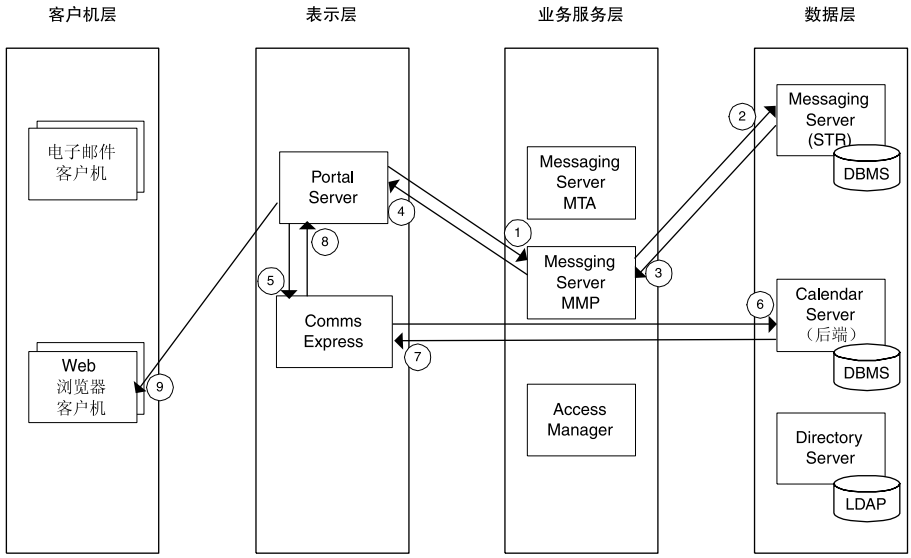
1. **Web 浏览器客户机将用户 ID 和密码发送到 Portal Server。**
2. **Portal Server 向 Access Manager 请求验证。**
3. **Access Manager 向 Directory Server 请求用户 ID 和密码验证。**
4. **Directory Server 验证用户 ID 和密码。**
5. **Access Manager 向 Directory Server 请求用户配置文件。**
6. **Directory Server 返回用户配置文件。**

7. **Portal Server 向 Access Manager 请求用户显示配置文件。**
8. **Access Manager 返回门户配置。**
9. **门户配置在浏览器客户机中显示。**



▼ 使用案例 2：门户服务器显示电子邮件和日历信息

步骤
1. **在成功登录、验证、检索门户配置后，Portal Server 向 Messaging Server MMP 请求电子邮件消息。**
2. **MMP 向 Messaging Server STR 请求消息列表。**
3. **STR 将消息列表返回给 MMP。**
4. **MMP 将消息标头转发给 Portal Server。**
5. **Portal Server 向 Communications Express 请求日历信息。**
6. **Communications Express 向 Calendar Server 后端请求日历信息。**
7. **Calendar Server 后端将日历信息返回给 Communications Express。**
8. **Communications Express 将日历信息转发给 Portal Server。**
9. **Portal Server 将所有频道信息发送给 Web 浏览器客户机。**



访问区

另一种表示逻辑体系结构组件的方式是将其置于访问区中，这些访问区显示体系结构如何提供安全访问。下图说明了部署 Java Enterprise System 组件的访问区。每个访问区显示了组件如何提供与 Internet 和内联网间的安全远程访问。

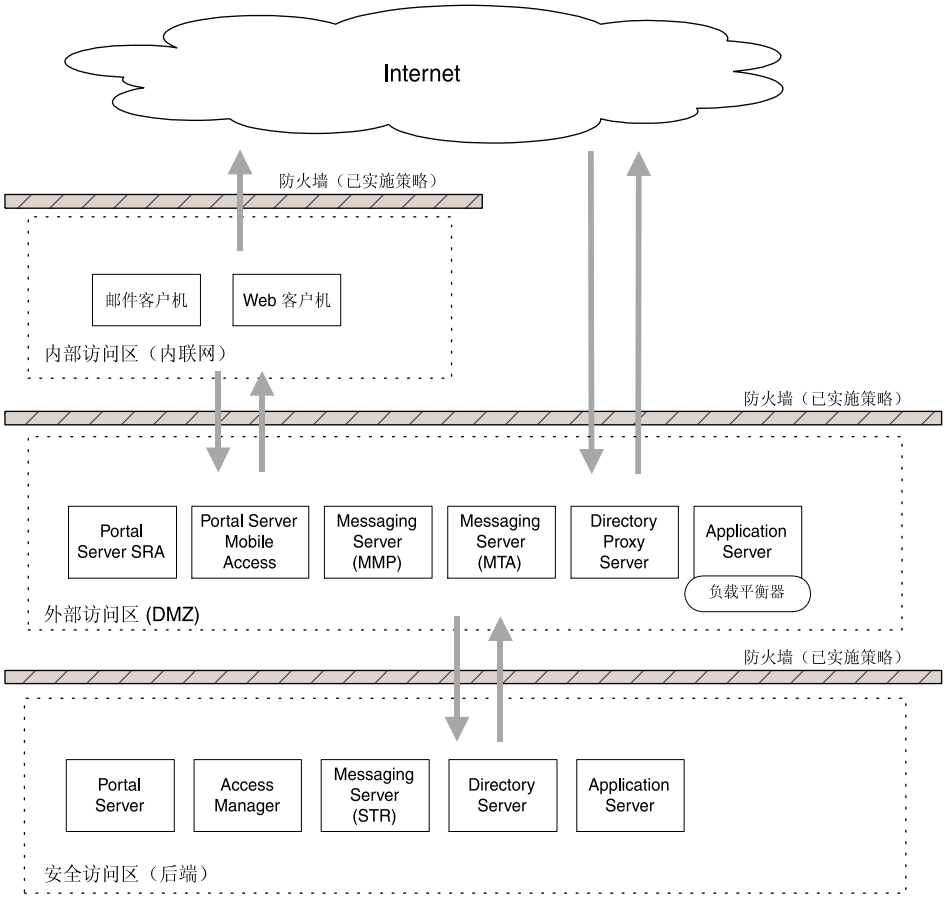


图 4–5 置于访问区的逻辑组件

下表介绍了第 56 页中的 “访问区”中显示的访问区。

表 4–6 安全访问区及置于其中的组件

访问区	说明
内部访问区 (Intranet)	通过由 Intranet 与 Internet 间的防火墙执行的策略访问 Internet。内部访问区通常用于最终用户进行 Web 浏览和发送电子邮件。 有些情况下，允许直接访问 Internet 进行 Web 浏览。但是，通常与 Internet 间的安全访问由外部访问区提供。
外部访问区 (DMZ)	提供与 Internet 间的安全访问，发挥关键后端服务的安全缓冲器的作用。

表 4–6 安全访问区及置于其中的组件 （续）

访问区	说明
安全访问区（后端）	提供对关键后端服务的受限访问，仅能从外部访问区访问这些服务。

第 56 页中的 “访问区”未显示前面示例中所述的逻辑层，而是着重强调了提供远程和内部访问的组件、这些组件与安全措施（如防火墙）间的关系，以及必须执行的访问规则的直观显示。将多层体系结构设计与显示访问区的设计结合使用，提供规划部署的逻辑模型。

部署方案

完成的逻辑体系结构设计本身还不足以进入解决方案生命周期的部署设计阶段。还需将逻辑体系结构与技术要求阶段确定的服务质量 (quality of service, QoS) 要求配对组合。逻辑体系结构与 QoS 要求间的相互对应共同组成了部署方案。部署方案是部署体系结构设计的起点，如第 5 章所述。

第 5 章

部署设计

在解决方案生命周期的部署设计阶段，需要设计一个高层部署体系结构和一个低层实现规范，并准备一系列实现该解决方案所必需的计划和规范。项目核准出现在部署设计阶段。

本章包括以下部分：

关于部署设计

部署设计始于解决方案生命周期的逻辑设计和技术要求阶段创建的部署方案。部署方案包括逻辑体系结构和解决方案的服务质量 (Quality of Service, QoS) 要求。通过在物理服务器和其他网络设备之间映射在逻辑体系结构中确定的组件，从而创建部署体系结构。要求对硬件配置提供了指导，以满足性能、可用性、可伸缩性和其他相关规范。

部署体系结构的设计是一个反复进行的过程。通常要复查要求和初步设计。要考虑不同要求之间的相互关系，对折衷和拥有成本问题进行平衡以获得最佳解决方案，最终满足该项目的业务目标。

项目核准

项目核准出现在部署设计阶段，通常在创建部署体系结构之后。使用部署体系结构（还可能用到下述的实现规范）对部署的实际成本进行估计，并提交给风险承担者进行核准。项目一经核准，即签署部署完成合同，并获取和分配项目实现所需的资源。

部署设计输出

在部署设计阶段，可能需要准备下列规范和计划：

- **部署体系结构。**描述逻辑体系结构域物理环境的映射关系的高层体系结构。物理环境包括内联网或 Internet 环境中的计算节点、处理器、内存、存储器设备及其他硬件和网络设备。
- **实现规范。**用作构建部署蓝图的详细规范。这些规范提供了计算机和网络硬件的细节，用于获取和描述部署的网络布局。实现规范还包括目录服务的规范，其中包括目录信息树的更多信息及为目录访问定义的组和角色。
- **实现计划。**包含实现企业软件解决方案各个方面的一组计划。实现计划包括以下各项：
  - **迁移计划。**描述迁移企业数据和升级企业软件的策略和进程。迁移的数据必须符合新安装的企业应用程序的格式和标准。所有企业软件必须为正确的发行版本级别，才能进行交互操作。
  - **安装计划。**源自部署体系结构，用于指定硬件服务器名称、安装目录、安装顺序、每个节点的安装类型以及安装和配置分布式部署所必需的配置信息。
  - **用户管理计划。**包括现有目录和数据库中的数据的迁移策略、考虑到部署体系结构中指定的复制设计的目录设计规范以及使用新内容置备目录的过程。
  - **测试计划。**描述测试已部署软件的过程，包括用于开发原型和试验性实现的具体计划、确定处理预测负载的能力的负载测试以及确定计划的功能是否按预期运行的功能性测试。
  - **展开计划。**描述实现从计划和测试环境转向生产环境的过程和时间表。将实现转向生产的过程通常出现在不同阶段。例如，第一个阶段可能是为有限的用户组部署软件，在随后的每一阶段逐渐增加用户群体，直到整个部署完成。分阶段实现还可包括按计划实现特定软件包，直到整个部署完成。
  - **灾难恢复计划。**描述如何将系统从意外的系统范围故障中恢复的过程。此恢复计划包括大规模和小规模故障的过程。
  - **操作计划（运行手册）。**描述监视、维护、安装和升级过程的操作手册。
  - **培训计划。**提供培训新安装的企业软件的操作员、管理员和最终用户的过程和步骤。

影响部署设计的因素

有几个影响部署设计过程中所作决策的因素。请考虑下列关键因素：

- **逻辑体系结构。**逻辑体系结构详细说明了提议解决方案中的功能性服务以及提供这些服务的组件之间的相互关系。将逻辑体系结构用作确定分配服务最佳方式的关键。部署方案包括逻辑体系结构及与之相对应的服务质量要求（如下所述）。
- **服务质量要求。**服务质量 (Quality of Service, QoS) 要求指定解决方案操作的各个方面。使用 QoS 要求有助于开发策略，以期达到性能、可用性、可伸缩性、可维护性及其他服务质量目标。部署方案包括逻辑体系结构（如前所述）及与之对应的服务质量要求。
- **用量分析。**在解决方案生命周期的技术要求阶段开发用量分析，从而提供有助于估计已部署系统负载的使用模式的信息。用量分析的使用有助于隔离性能瓶颈，开发出满足 QoS 要求的策略。
- **使用案例。**在解决方案生命周期的技术要求阶段开发使用案例，列出为某一部署确定的独特用户交互，通常确定最常见的使用案例。尽管使用案例已包含在用量分析中，但评估部署设计时，应参考使用案例，确保它们已妥善解决。
- **服务级别协议。**服务级别协议 (Service Level Agreement, SLA) 指定了最低性能要求以及未能满足此要求时必须提供的客户支持级别和程度。部署设计应能轻松满足服务级别协议中指定的性能要求。
- **总拥有成本。**在部署设计期间，分析能够解决可用性、性能、可伸缩性等 QoS 要求的潜在解决方案。但是，对于所考虑的每个解决方案，必须同时考虑该解决方案的成本及该成本影响总拥有成本的程度。请确保考虑决策中涉及到的折衷，并且已对资源进行了优化，能够在业务约束范围内达到业务需求。
- **业务目标。**业务目标始于解决方案生命周期的业务分析阶段，包括实现这些目标的业务需求和业务约束。最终将根据部署设计满足业务目标的能力对其进行评判。

部署设计方法

与部署规划的其他方面一样，部署设计不仅是一门科学，也是一门艺术，不能用特定的步骤和过程来详细规定。以往的设计经验、系统体系结构知识和特定领域知识的掌握以及发挥创造性思维，这些都是成功完成部署设计的因素。

部署设计通常围绕满足其他 QoS 要求的同时达到性能要求而展开。采用的策略必须对设计决策中的各种折衷进行平衡，以期优化解决方案。使用的方法通常涉及下列任务：

- **估计处理器要求。**部署设计通常从估计逻辑体系结构中每个组件所需的 CPU 数量开始。始于代表最大负载的使用案例，然后继续考虑每个使用案例。考虑对使用案例提供支持的所有组件上的负载，并相应地修改您的估计。还应考虑您在设计企业系统方面的任何以往经验。
- **估计安全传输的处理器要求。**研究要求安全传输的使用案例并相应修改 CPU 估计。
- **复制服务以实现可用性和可伸缩性。**对处理器估计感到满意后，请针对可用性和可伸缩性方面的 QoS 要求来修改设计。考虑能够解决可用性和故障转移事项的负载平衡解决方案。

  分析时应考虑设计决策的折衷问题。例如，可用性与可伸缩性策略对系统的可维护性（维护）有什么影响？这些策略的其他成本是什么？

- **确定瓶颈。**继续进行分析时，请查看部署设计，确定任何导致数据传输低于要求的瓶颈并进行调整。
- **优化资源。**查看资源管理的部署设计，考虑在满足要求的同时最小化成本的选项。
- **管理风险。**复查设计中的业务和技术分析，针对早期规划中可能未预见到的事件或情况进行修改。

估计处理器要求

本节讨论了估计支持部署设计中的服务所必需的 CPU 处理器数量和相应内存的过程。本节包括一个示例通信部署方案估计过程的演练。

估计 CPU 计算能力是一个不断反复的过程，要考虑下列方面：

- 逻辑组件及它们之间的交互（由逻辑体系结构中的组件依赖性表示）
- 已确定使用案例的用量分析
- 服务质量要求
- 以往部署设计和使用 Java Enterprise System 的经验
- 咨询具有设计和实现各类部署方案经验的 Sun 专业服务机构

估计过程包括下列步骤。这些步骤的执行顺序并非关键所在，这一顺序只是提供了考虑可对最终结果产生影响的因素的一种方式。

1. 给确定为用户的系统入口点的组件确定 CPU 数量估计底线。

   一个设计决策是完全加载还是部分加载 CPU。完全加载 CPU 可使系统容量最大化。要增加容量，需要添加额外的 CPU，从而增加了维护成本及停机的可能性。某些情况下，可以选择添加其他计算机来满足不断增长的性能要求。

   部分加载 CPU 为处理超额性能要求预留了余地，无需立即增加维护成本。不过，这种未充分利用的系统却附加了提前的费用开销。

2. 对 CPU 数量估计进行调整，将组件间的交互考虑在内。

   研究逻辑体系结构中组件之间的交互，确定相关组件要求的额外负载。

3. 研究特定使用案例的用量分析，确定系统的峰值负载，然后对处理峰值负载的组件进行调整。

   从最大加权的使用案例（要求最多负载）开始，然后考察每个使用案例，确保考虑了所有的预测使用方案。

4. 对 CPU 数量估计进行调整，将安全性、可用性和可伸缩性要求考虑在内。

此估计过程是确定所需实际处理能力的起点。通常，根据这些估计创建原型部署，然后对预期使用案例执行严格的测试。只有经过反复的测试，才能确定部署设计的实际处理要求。

示例的估计处理器要求

本节说明了一种对示例部署所需处理能力进行估计的方法。该示例部署基于员工数为 1,000 到 5,000 人的中型企业基于身份的通信解决方案的逻辑体系结构，如第 53 页中的 “基于身份的通信示例”一节所述。

本例中使用的 CPU 和内存数字都只是说明性的随意估计。这些数字基于本理论性示例所依据的任意数据。要估计处理器要求，有必要对各种因素进行彻底分析。此分析将包括（但不限于）下列信息：

- 基于彻底的业务分析的详细使用案例和用量分析
- 通过对业务需求分析而确定的服务质量要求
- 处理和网络硬件的特定成本和规范
- 实现类似部署的以往经验

注意 – 这些示例中提供的信息用于说明设计系统时可能使用的一种过程，并不代表任何具体的实现建议。

为用户入口点确定 CPU 数量估计底线

先估计处理每个作为用户入口点的组件的预期负载所需的 CPU 数量。下图显示了一种基于身份的通信方案的逻辑体系结构，先前在第 53 页中的 “基于身份的通信示例”一节中有所介绍。

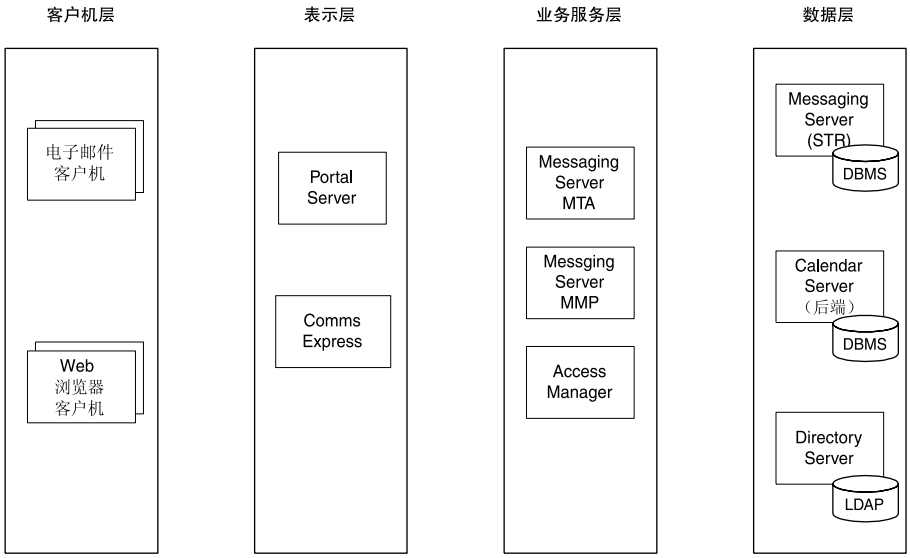


图 5–1 基于身份的通信方案的逻辑体系结构

下表列出该逻辑体系结构中与此部署的终端用户直接接口的表示层中的组件。该表包括 CPU 数量估计底线，这些估计源自技术要求分析、使用案例、特定用量分析以及此类型部署的以往经验。

表 5–1 包含用户入口点组件的 CPU 数量估计

组件	CPU 数量	说明
Portal Server	4	作为用户入口点的组件。
Communications Express	2	将数据发送到 Portal Server 消息传送和日历频道。

包括针对服务依赖性的 CPU 数量估计

提供用户入口点的组件需要其他 Java Enterprise System 组件的支持。为继续指定性能要求，进行性能估计时请将其他 Java Enterprise System 组件要求的支持考虑在内。设计逻辑体系结构时，应详细说明组件之间的交互类型，如第 49 页中的 “逻辑体系结构示例”一节中的逻辑体系结构示例所述。

表 5–2 支持组件的 CPU 数量估计

组件	CPU	说明
Messaging Server MTA（入站）	1	路由来自 Communications Express 和电子邮件客户机的入内邮件消息。
Messaging Server MTA（出站）	1	将出外邮件消息发送给收件人。
Messaging Server MMP	1	访问电子邮件客户机的 Messaging Server 消息存储。
Messaging Server STR (Message Store)	1	检索和存储电子邮件消息。
Access Manager	2	提供授权和验证服务。
Calendar Server（后端）	2	检索和存储 Calendar Server 前端 Communications Express 的日历数据。
Directory Server	2	提供 LDAP 目录服务。
Web Server	0	提供对 Portal Server 和 Access Manager 的 Web 容器支持。 （无需附加 CPU 周期。）

研究峰值负载用量的使用案例

返回使用案例和用量分析，确定峰值负载用量的区域，并对 CPU 数量估计进行调整。

例如，假设本例中确定如下峰值负载情况：

- 同时登录时用户数的初始激增
- 在指定期限内交换电子邮件

要考虑此峰值负载用量，应调整提供这些服务的组件。下表概述了考虑此峰值负载用量时可能要做的调整。

表 5–3 针对峰值负载进行的 CPU 数量估计调整

组件	CPU（调整后）	说明
Messaging Server 入站 MTA	2	为峰值外来电子邮件添加 1 个 CPU
Messaging Server 出站 MTA	2	为峰值出外电子邮件添加 1 个 CPU
Messaging Server MMP	2	为附加负载添加 1 个 CPU
Messaging Server STR（消息存储）	2	为附加负载添加 1 个 CPU
Directory Server	3	为附加 LDAP 查找添加 1 个 CPU

修改其他负载情况的估计

继续进行 CPU 数量估计，将影响负载的其他服务质量要求考虑在内：

- **安全。**从技术要求阶段开始，确定安全数据传输可能对负载要求的影响程度，并对估计进行相应修改。下一节第 67 页中的 “估计安全事务的处理器要求”中，介绍了进行调整的过程。
- **复制服务。**调整 CPU 数量估计，将用于可用性、负载平衡和可伸缩性的复制服务的情况考虑在内。下一节第 69 页中的 “确定可用性策略”中，讨论了对可用性解决方案的估量。第 76 页中的 “确定可伸缩性策略”一节中讨论了涉及目录服务可用性访问的解决方案。
- **潜在容量和可伸缩性。**根据需要修改 CPU 数量估计，为部署中的意外大型负载预留潜在容量。查看预计的扩展重大事件点及相应时段的预计负载增量，确保满足任何水平或垂直扩展系统的重大事件点的要求。

更新 CPU 数量估计

通常将 CPU 数量向上舍入到最近的偶数。向上舍入到偶数可在两个物理服务器间均匀分布分摊 CPU 数量估计，而且也为潜在容量增加了一个小因素。但是，应根据对复制服务的特定需要进行舍入。

一般规则是为每个 CPU 预留 2 GB 内存。实际所需的内存取决于您的特定用量，可以通过测试确定。

下表列出了该基于身份的通信示例的最终估计。这些估计值不包括任何额外计算能力（已在安全性和可用性中添加）。安全性和可用性的总数将在后续各节添加。

表 5–4 支持组件的 CPU 数量估计调整

组件	CPU	内存
Portal Server	4	8 GB
Communications Express	2	4 GB
Messaging Server（MTA，入站）	2	4 GB
Messaging Server（MTA，出站）	2	4 GB
Messaging Server MMP	2	4 GB
Messaging Server (Message Store)	2	4 GB
Access Manager	2	4 GB
Calendar Server	2	4 GB

表 5–4 支持组件的 CPU 数量估计调整　（续）

组件	CPU	内存
Directory Server	4	8 GB（从 3 CPU/6 GB 内存向上舍入）
Web Server	0	0

估计安全事务的处理器要求

安全数据传输涉及通过安全传输协议（如安全套接字层 (Secure Sockets Layer, SSL) 或传输层安全 (Transport Layer Security, TLS)）处理事务。通过安全传输处理的事务通常需要额外的计算能力先建立安全会话（称为握手），然后对传输的数据进行加密和解密。额外的计算能力要求可能相当大，这取决于所使用的加密算法（例如，40 位还是 128 位加密算法）。

为使安全事务具有与非安全事务相同水平的性能，必须对额外计算能力进行规划。安全事务可能需要高于非安全事务四倍（甚至更多倍）的计算能力，具体取决于事务的性质和处理事务的 Sun Java™ Enterprise System 服务。

估计处理安全事务的处理能力时，首先要分析使用案例来确定需要安全传输的事务所占的百分比。如果安全事务的性能要求与非安全事务相同，则需修改 CPU 数量估计，将安全事务所需的额外计算能力考虑在内。

在某些使用方案中，可能只有在验证时才需要进行安全传输。用户通过系统验证后，便不需要对数据传输采取额外安全措施。但在其他方案中，可能所有事务都需要安全传输。

例如，浏览在线电子商务站点的产品目录时，客户完成商品挑选并准备“付帐”前，所有事务都可以是非安全的。不过，某些使用方案（如银行或证券经纪行部署）要求大多数或全部事务必须为安全事务，并对安全与非安全事务有着相同的性能标准。

安全事务的 CPU 数量估计

本节将继续使用示例部署，说明如何计算得出既包括安全事务又包括非安全事务的理论性使用案例所需的 CPU 数量。

要估计安全事务的 CPU 数量要求，请进行以下计算：

1. 以 CPU 数量估计底线数（如上一节第 63 页中的 “示例的估计处理器要求”中计算的数字）作为起始数量。
2. 计算需要安全传输的事务的百分比，然后计算安全事务的 CPU 数量估计。
3. 计算非安全事务减少的 CPU 数量估计。
4. 将安全和非安全数量估计相加，计算出总 CPU 数量估计。
5. 将 CPU 数量估计向上舍入到最近的偶数。

第 67 页中的 “安全事务的 CPU 数量估计”显示了一个基于 Portal Server 的使用案例和用量分析的示例计算，其中假设：

- 所有登录均要求安全验证。
- 所有登录占 Portal Server 总负载的 10%。
- 安全事务的性能要求与非安全事务相同。

  要将处理安全事务的额外计算能力考虑在内，处理这些事务的 CPU 数量将增加为四倍。与本例中其他 CPU 数字一样，该倍数也只是说明性的随意倍数。

表 5–5 修改安全事务的 CPU 数量估计

步骤	说明	计算	结果
1	以所有 Portal Server 事务的估计底线作为起始数量。	第 65 页中的 “研究峰值负载用量的使用案例”中的估计底线为 4 个 CPU。	-----

2	计算安全事务的附加 CPU 数量估计。假设安全事务需要的 CPU 能力为非安全事务的四倍。	估计底线的百分之十需要安全传输：
		`0.10 x 4 CPU = 0.4 CPU`
		将安全事务的 CPU 能力增加为四倍：
		`4 x 0.4 = 1.6 CPU`
			1.6 个 CPU

3	计算非安全事务减少的 CPU 数量估计。	估计底线的百分之九十为非安全传输： `0.9 x 4 CPU = 3.6 CPU`	3.6 CPU
4	计算调整后的安全与非安全事务的 CPU 数量估计总数。	安全数量估计非安全数量估计总数： `1.6 CPU + 3.6 CPU = 5.2 CPU`	5.2 CPU
5	向上舍入到最近的偶数。	`5.2 CPU ==> 6 CPU`	6 CPU

根据本例中的安全事务计算，通过添加额外的两个 CPU 和 4 GB 内存，修改第 67 页中的 “安全事务的 CPU 数量估计”中的总 CPU 数量估计，得到以下用于 Portal Server 的总数。

组件	CPU	内存
Portal Server	6	12 GB

处理 SSL 事务的专用硬件

可以利用专用硬件设备（如 SSL 加速卡和其他装置）提供建立安全会话和加密与解密数据所需的计算能力。使用专用硬件进行 SSL 运算时，计算能力专用于 SSL 计算的特定部分，通常是建立安全会话的“握手”运算。

这种硬件可能会对最终部署体系结构有益。不过，由于此类硬件的专用化性质，最好先以 CPU 能力估计出安全事务的性能要求，然后再考虑使用专用硬件处理额外负载的益处。

使用专用硬件时应考虑的一些因素有：使用案例（例如，要求大量 SSL 握手运算的使用案例）是否支持使用该硬件及使用此类硬件给设计增添的复杂性。这种复杂性包括这些设备的安装、配置、测试和管理。

确定可用性策略

开发可用性要求策略时，应研究组件交互和用量分析，以确定要考虑的可用性解决方案。对组件进行逐个分析，以确定最适合可用性和故障转移要求的解决方案。

下列项目是为帮助确定可用性策略而需收集的信息类型的示例：

- 指定的可用性中有多少个九?
- 故障转移情况下的性能要求（例如，故障转移期间至少保持 50% 的性能）是什么?
- 用量分析是否区分高峰和非高峰使用时间?
- 地域考虑因素有哪些?

所选的可用性策略还必须考虑第 80 页中的 “设计最佳资源使用方案”中阐述的可维护性要求。避免需要太多管理和维护的复杂解决方案。

可用性策略

Java Enterprise System 部署的可用性策略包括以下各项：

- **负载平衡。**使用冗余硬件和软件组件来分流处理负载。负载平衡器把对某个服务的任意请求引导至该服务的多个对称服务实例之一。如果任一实例发生故障，其他实例可以承担更大的负载。
- **故障转移。**涉及对冗余硬件和软件的管理，在任何组件发生故障时提供对服务的不间断访问并保证关键数据的安全。

  Sun Cluster 软件为后端组件管理的关键数据提供了故障转移解决方案，比如 Messaging Server 的消息存储和 Calendar Server 日历数据。
- **复制服务。**复制服务为对同一数据的访问提供多个源。Directory Server 为 LDAP 目录访问提供多个复制和同步策略。

后续各节给出一些提供不同级别的负载平衡、故障转移和复制服务的可用性解决方案示例。

单服务器系统

将服务的所有计算资源置于单个服务器上。如果服务器发生故障，整个服务便终止运行。

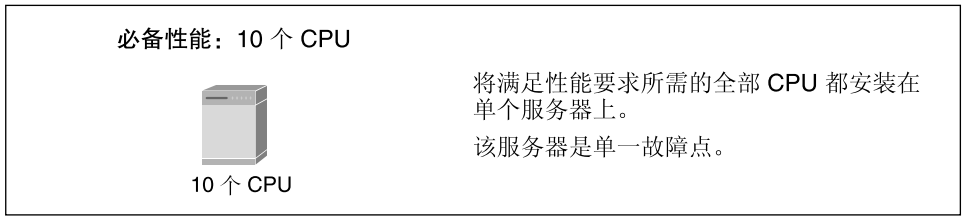


图 5–2 单服务器系统

Sun 提供具有下列优点的高端服务器：

- 系统运行中更换和重新配置硬件组件
- 可在服务器的故障隔离域中运行多个应用程序
- 不必重新引导系统即可升级容量、执行速度及 I/O 配置

一台高端服务器的价格通常高于具有可比性的多服务器系统。不过，单个服务器可节省对数据中心多台服务器的管理、监视和驻扎成本。多服务器系统在负载平衡、故障转移和解除单个故障点方面的灵活性更强。

水平冗余系统

利用可提供负载平衡和故障转移两种功能的平行冗余服务器提高可用性的方法有若干种。下图显示的是组成一个 N+1 故障转移系统的两台复制服务器。其中一台服务器发生故障时，N+1 系统通过一个额外服务器继续提供 100% 容量。

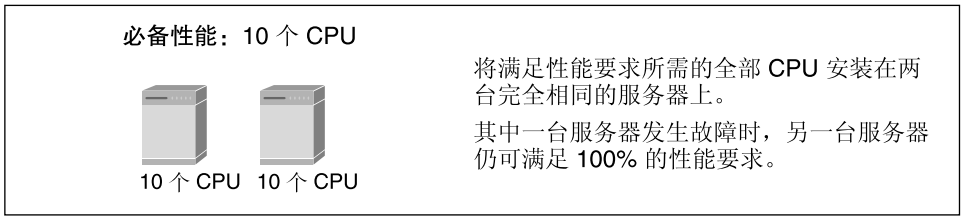


图 5–3 具有两台服务器的 N+1 故障转移系统

上面第 70 页中的 “水平冗余系统”中每台服务器的计算能力都完全相同。一台服务器即可满足性能要求，另一台服务器作为备份服务器调入服务时，也可提供 100% 的性能。

N+1 故障转移设计的优点是，故障转移情况下仍可达到 100% 的性能。缺点是增加了硬件成本，而总体性能却未得到相应提升（因为一个服务器只在发生故障转移的情况下使用，平时备用）。

下图所显示的是这样一种系统：通过在两台服务器间分配性能负载来实现负载平衡和故障转移。

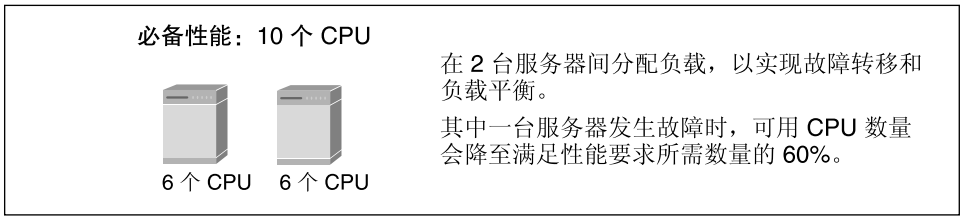


图 5–4 两台服务器间的负载平衡和故障转移

在上面第 70 页中的 “水平冗余系统”中所示的系统内，如果一台服务器发生故障，所有服务仍然可用，只不过性能只能达到完全性能的某一百分比。另一台服务器提供 6 CPU 计算能力，为要求的 10 CPU 的 60%。

这种设计的一个优点是，两台服务器均可用时有 2 个 CPU 的额外潜潜在容量。

下图显示的是，在多台服务器间分配负载来满足性能和负载平衡要求。

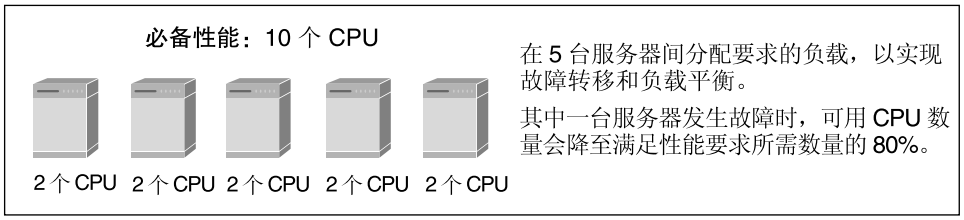


图 5–5 在 *n* 台服务器间分配负载

由于第 70 页中的 “水平冗余系统”所示的设计中有五台服务器，如果一台服务器发生故障，其余服务器可继续提供总计 8 个 CPU 的计算能力，达到 10 个 CPU 性能要求的 80%。如果在设计中增加一个具有 2 CPU 计算能力的服务器，实际得到的便是 N+1 设计。如果一台服务器发生故障，其余服务器可满足 100% 的性能要求。

这种设计具有下列优点：

- 单台服务器发生故障时的性能得到提升
- 即使不止一台服务器停机，仍然具有可用性
- 可轮换将服务器停机，以进行维护和升级
- 多台低端服务器的价格通常低于单台高端服务器

不过，增加服务器数量会使管理和维护成本大幅增加，还应考虑在数据中心驻扎服务器的成本。达到某一数量后，再增加服务器所得到的性能提升会越来越少。

Sun Cluster 软件

对于要求高可用性（如四或五个九）的情况，可以考虑将 Sun Cluster 软件纳入可用性设计。群集系统是冗余服务器、存储器及其他网络资源相结合的产物。群集中的服务器彼此间不间断地通信，如果其中一台服务器脱机，群集中的其余设备会将该服务器隔离，并将故障节点上的任何应用程序或数据故障转移到另一节点。这一故障转移过程所需时间较短，几乎不用中断为系统用户提供的服务。

配置、管理和维护 Sun Cluster 软件需要额外的专用硬件和专门技能。

可用性设计示例

本节包含两个可用性策略示例，基于员工数为 1,000 至 5,000 人的中型企业基于身份的通信解决方案，如先前在第 53 页中的 “基于身份的通信示例”一节中所述。第一个可用性策略说明了用于 Messaging Server 的负载平衡。第二个说明了使用 Sun Cluster 软件实现故障转移的解决方案。

Messaging Server 的负载平衡示例

下表中列出了逻辑体系结构中每个逻辑 Messaging Server 组件的 CPU 能力估计。此表重复了在第 66 页中的 “更新 CPU 数量估计”一节中计算出的最终估计数量。

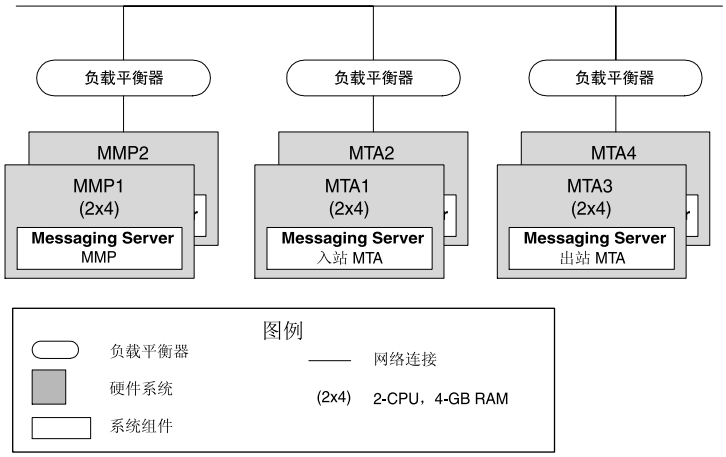
表 5–6 支持组件的 CPU 数量估计调整

组件	CPU	内存
Messaging Server（MTA，入站）	2	4 GB
Messaging Server（MTA，出站）	2	4 GB
Messaging Server (MMP)	2	4 GB
Messaging Server (Message Store)	2	4 GB

对于本例，假设在技术要求阶段指定了下列服务质量要求：

- **可用性。**总体系统可用性应为 99.99%（不包括计划停机）。单个计算机系统故障不会导致服务故障。
- **可伸缩性。**日常峰值负载情况下，任何服务器的使用量都不应超过 80%，而且系统必须能够适应每年 10% 的长期增长速度。

为满足可用性要求，应为每个 Messaging Server 组件提供两个实例，每一个都位于不同的硬件服务器上。如果一个组件的服务器发生故障，另一个组件可提供服务。下图显示了此可用性策略的网络示意图。



上图中，CPU 数量为原估计数量的两倍。CPU 数量加倍的原因如下：

- 在一台服务器发生故障的情况下，另一台服务器提供处理负载的 CPU 能力。
- 对于任何服务器在峰值负载下利用程度不超过 80% 的可伸缩性要求，添加的 CPU 能力可提供此保险余量。
- 对于适应年增长 10% 负载的可伸缩性要求，添加的 CPU 能力增加了潜在容量，在需要另外扩大规模之前可用于处理增长的负载。

使用 Sun Cluster 软件的故障转移示例

下图显示了一个对 Calendar Server 后端和 Messaging Server 消息存储实施故障转移策略的示例。Calendar Server 后端和消息存储都在单独的硬件服务器上复制，并且使用 Sun Cluster 软件配置了故障转移。Sun Cluster 中每台服务器上的 CPU 数量和相应内存都被复制。

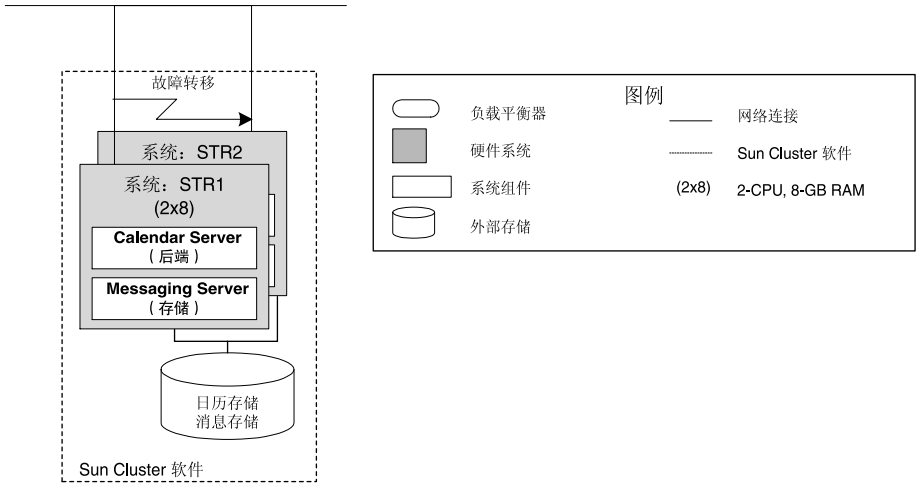


图 5–6 使用 Sun Cluster 软件的故障转移设计

目录服务复制示例

可对目录服务进行复制，以便在不同服务器间分配事务，从而提供高可用性。Directory Server 提供各种复制服务策略，其中包括：

- **多数据库**。在不同数据库中存储目录树的不同部分。
- **链锁和引用**。将分布数据链接到单个目录树。
- **单主复制**。为主数据库提供一个中心源，然后将该中心源分配到使用者副本中。
- **多主复制**。在多个服务器间分配主数据库。然后，这些主中的每一个都在使用者副本中分配其各自的数据库。

Directory Server 的可用性策略是个复杂的问题，不在本指南的讨论范围之内。以下两节，第 74 页中的 “单主复制”和第 75 页中的 “多主复制”提供了对基本复制策略的概括说明。有关详细信息，参见《Sun Java System Directory Server 5 2005Q1 Deployment Plannning Guide》的第 12 章 "Designing a Highly Available Deployment"。

单主复制

下图显示了一个说明基本复制概念的单主复制策略。

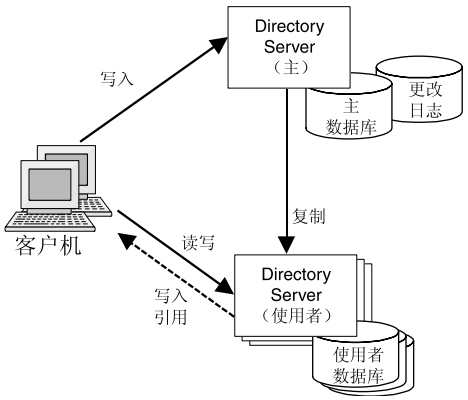


图 5–7 单主复制示例

在单主复制中，一个 Directory Server 实例管理主目录数据库，并记录所有变化。主数据库被复制为任意数量的使用者数据库。Directory Server 的使用者实例按读取和搜索操作进行了优化。使用者接收的任何写操作都被引用回主数据库。主数据库定期更新使用方数据库。

单主复制的优点包括：

- 为数据库读取和写入操作优化了单个 Directory Server
- 为读取和搜索操作优化了任意数量的 Directory Server 使用者实例
- Directory Server 使用者实例的水平可伸缩性

多主复制

下图显示了一个可用于全局分配目录访问的多主复制策略。

在多主复制中，一个或多个 Directory Server 实例管理主目录数据库。每个主数据库都有一个指定同步主数据库的过程的复制协议。每个主数据库都被复制为任意数量的使用者数据库。与单主复制相同，Directory Server 的使用者实例都按读取和搜索访问进行了优化。使用者接收的任何写操作都被引用回主数据库。主数据库定期更新使用方数据库。

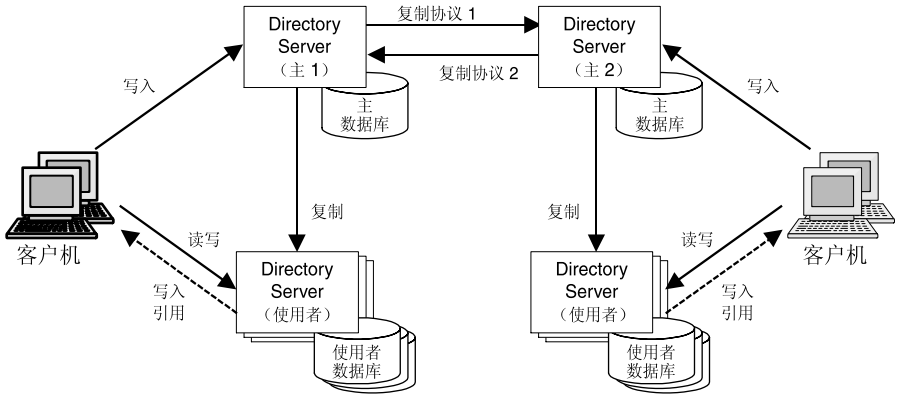


图 5–8 多主复制示例

多主复制策略除具有单主复制的所有优点之外，还提供了一个在更新主数据库时提供负载平衡的可用性策略。还可以实现一个对目录操作提供本地控制的可用性策略，这对于在全球分布数据中心的企业是一个很重要的考虑事项。

确定可伸缩性策略

可伸缩性是指增加系统容量的能力，通常是以增加系统资源但不改变部署体系结构的方式进行。在要求分析阶段，通常是根据业务需求和后续用量分析对预期增长作出预测。这些对系统用户数量和满足他们需要的系统容量的预测往往只是估计值，可能与部署系统的实际数字大相径庭。设计应具备足够的灵活性，考虑到预测中存在的偏差。

可伸缩的设计具有足够的处理增加负载的潜在容量，直到系统使用附加资源进行升级为止。可伸缩设计可以随时进行扩展，以处理持续增加的负载，同时无需重新设计系统。

潜在容量

潜在容量是可伸缩性的一个方面，即在系统中增加额外的性能和可用性资源，以便在出现异常峰值负载时能够从容应对。还可以对部署系统中潜在容量的使用案例进行监视，以协助确定何时要通过增加资源来扩展系统。潜在容量是给设计注入安全机制的一种方法。

分析使用案例有助于确定可能产生异常峰值负载的方案。利用对异常峰值负载的分析，并对可应对意外增长的因素加以考虑，便能够设计出可给系统注入安全机制的潜在容量。

系统设计应能处理一定的合理时间（通常为系统运行的前 6 到 12 个月）内的预测容量。可利用维护周期，根据需要增加资源或扩大容量。理想情况下应能安排定期对系统进行升级，但预测需要增加的容量往往很难。根据仔细的资源监视和业务预测来确定升级系统的时间。

如果计划在渐增式阶段中实现您的解决方案，则可以安排系统容量增长计划，使其与为每个渐增式阶段安排的其他改善相一致。

可伸缩性示例

本节中的示例说明对一个实现 Messaging Server 的解决方案进行水平和垂直扩展。对于垂直扩展，可向服务器添加额外的 CPU 以处理增长的负载。对于水平扩展，通过添加额外的服务器分摊负载来处理增长的负载。

本例的底线假设通过两个为负载平衡而分布的消息存储实例支持 50,000 名用户群体。每个服务器有两个 CPU，共有四个 CPU。下图显示如何扩展系统，为 250,000 名用户或 2,000,000 名用户处理增长的负载。

注 – 第 77 页中的 “可伸缩性示例” 显示垂直扩展和水平扩展的区别。本图未显示扩展时应考虑的其他因素，如负载平衡、故障转移和使用模式变化。

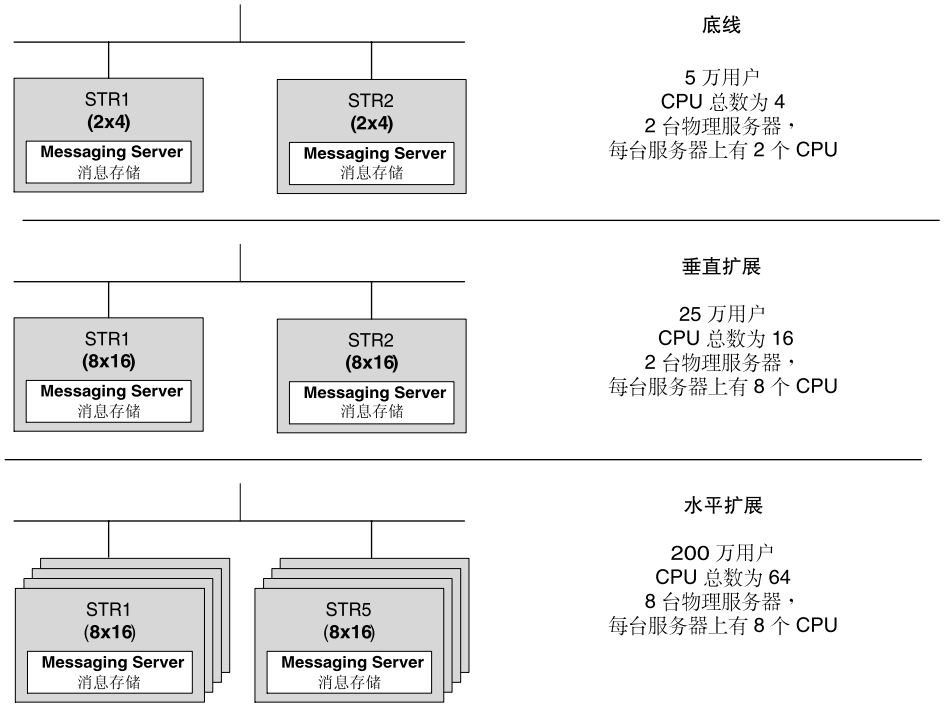


图 5–9 水平和垂直扩展示例

确定性能瓶颈

成功的部署设计的关键之一是确定潜在的性能瓶颈并开发出可以避免它们的策略。访问数据的速率不能满足指定的系统要求时，则出现性能瓶颈。

可以根据不同硬件的类别（例如，下表列出某一系统内的数据访问点）对瓶颈进行分类。此表还为每种硬件类别中存在的瓶颈提出了可能的补救措施。

表 5–7 数据访问点

硬件类别	相对访问速度	改善性能的补救措施
处理器	纳秒	垂直扩展：提高更多的处理能力，增强处理器高速缓存
		水平扩展：添加负载平衡的并行处理能力
系统内存 (RAM)	微秒	使系统内存专用于特定任务
		垂直扩展：添加额外内存
		水平扩展：创建附加实例，用于并行处理和负载平衡
磁盘读写	毫秒	使用磁盘阵列 (RAID) 优化磁盘访问
		使磁盘访问专用于特定功能，如只读或只写
		在系统内存中缓存频繁访问的数据
网络接口	随网络节点的带宽和访问速度的不同而不同	增加带宽
		传输安全数据时添加加速器硬件
		增强网络内节点的性能，以便数据更加可用

注 – 第 78 页中的 “确定性能瓶颈”按照相对访问速度列出硬件类别，表明低速访问点，如磁盘，更可能是瓶颈源头。但是，能力不足的处理器处理大量负载时也可能形成瓶颈。

部署设计通常从为部署中的每个组件以及它们的依赖性进行底线处理能力估计开始。然后确定如何避免与系统内存和磁盘访问相关的瓶颈。最后检查网络接口，确定潜在的瓶颈并集中精力于解决它们的策略。

优化磁盘访问

部署设计的一个关键组件是磁盘对经常访问的数据集（如 LDAP 目录）的访问速度。磁盘访问对数据的访问速度最低，很可能是性能瓶颈的源头。

优化磁盘访问的方法之一是将写入操作与读取操作分开进行。不仅写入操作比读取操作更加费时，读取操作（查找 LDAP 目录的操作）的出现频率也远远高出写入操作（更新 LDAP 目录中的数据）。

优化磁盘访问的另一种方法是将各个磁盘专用于不同类型的 I/O 操作。例如，为 Directory Server 日志记录操作（如事务日志和事件日志）与 LDAP 读写操作分别提供磁盘访问。

另外，还可考虑实现一个或多个专用于读写操作的 Directory Server 实例以及使用分布于各本地服务器的复制实例进行读取和搜索访问。链锁和链接选项也可用于优化对目录服务的访问。

《Sun Java System Directory Server 5 2005Q1 Deployment Plannning Guide》第 6 章 "Defining System Characteristics" 讨论了规划磁盘访问时的各种因素。本章包括以下主题：

- **最低内存和磁盘空间要求。**提供各种大小的目录所需的磁盘和内存的估计值。
- **估量缓存访问的物理内存。**为根据 Directory Server 的计划用量和总规划内存用量估计高速缓存大小提供指导。
- **估量磁盘子系统大小。**提供根据目录后缀和影响磁盘使用的 Directory Server 因素规划磁盘空间要求的信息。以及提供在磁盘（包括各种磁盘阵列变通方案）间分布文件的信息。

设计最佳资源使用方案

部署设计不只是对满足 QoS 要求所需的资源进行估计，部署设计期间，还对所有可用选项进行分析，选择出最大限度降低成本而又能满足 QoS 要求的最佳解决方案。必须分析每个设计决策中的平衡点，确保在某一方面获得的益处不会被在另一方面产生的成本抵消。

例如，针对可用性进行水平扩展可能会提升总体可用性，但代价是需要增加维护和维修成本。针对性能进行垂直扩展可能会以经济的方式提高计算能力，但某些服务对这些附加能力的使用效率不高。

完成设计策略前，请对决策进行检查，确保平衡利用资源，使提议解决方案总体受益。此分析通常是检查某一方面的系统性质如何对其他系统性质产生影响。下表列出某些系统性质及相应的资源管理考虑事项。

表 5–8 资源管理考虑事项

系统质量	说明
性能	对于将 CPU 集中分布在个别几台服务器上的性能解决方案，服务能否对计算能力加以高效利用？（例如，某些服务对可高效利用的 CPU 数量有上限。）
潜在容量	您的策略是否处理超出性能估计的负载？ 对于过载，是以在服务器上进行垂直扩展的方式，以负载平衡到其他服务器的方式，还是以这两种方式兼用的方式进行处理？ 在达到下一部署扩展重大事件点前，潜在容量是否足以处理出现的异常峰值负载？

表 5–8 资源管理考虑事项 （续）

系统质量	说明
安全	是否对处理安全事务所需的性能开销给予了充分考虑?
可用性	对于水平冗余解决方案，是否对长期维护开支给予了充分估计?
	是否已将系统维护所需的计划停机考虑在内?
	是否在高端服务器和低端服务器间求得了成本平衡?
可伸缩性	是否对部署扩展的重大事件点进行了估计?
	是否制定了可在达到部署扩展重大事件点前提供足够的潜在容量来处理预测的负载增长的策略?
可维护性	是否在可用性设计中考虑了管理、监视和维护成本?
	是否考虑了采用委派管理解决方案（允许终端用户执行某些管理任务）来降低管理成本?

管理风险

部署设计所依据的许多信息，如服务质量要求和用量分析，并不是基于经验的数据，而是基于从业务分析最终得来的估计和预测数据。由于许多原因（包括无法预料的商业气候环境、不完善的数据收集方式或者简单的人为错误），这些预测可能不准确。完成部署设计前，请复查设计所依据的分析，确保设计已将任何估计或预测值的合理偏差考虑在内。

例如，如果用量分析低估了系统的实际用量，便要面临这样的风险：所构建的系统无法应付其遇到的流量。未达到性能要求的设计当然是失败的设计。

反之，如果所构建系统的能力远超所需能力，便白白浪费了本可用在他处的资源。关键在于，在满足要求的基础上留出一定的保险余量，但要避免资源浪费。

资源浪费可导致设计失败，因为这种设计未能充分利用本可用于其他一些领域的资源。此外，浪费资源的解决方案还可能给风险承担者以未诚信履行合同的印象。

示例的部署体系结构

下图显示的是本白皮书前文中介绍的示例部署的完整部署体系结构。通过此图可大概了解部署体系结构的表示方法。

注意 – 下图中的部署体系结构只作说明之用，它不代表实际设计、构建或测试的部署，不应将其视为部署规划建议。

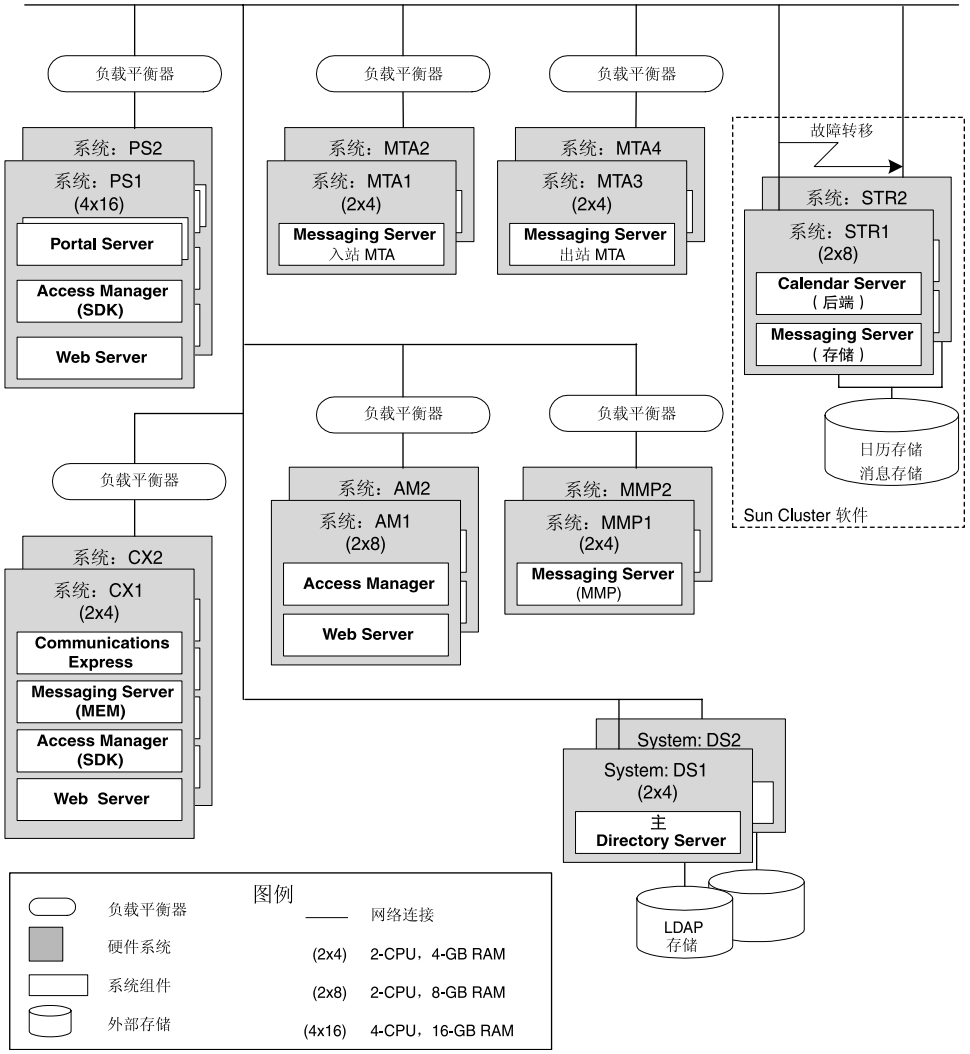


图 5–10 示例的部署体系结构

第 6 章

部署设计实现

在解决方案生命周期的实现阶段，从部署设计阶段创建的规范和规划着手，构建和测试部署体系结构，最终使部署进入生产阶段。虽然部署设计的实现不在本指南讨论范围之内，但本章将对此阶段进行概括说明。

本章包括以下部分：

关于实现部署设计

部署体系结构获得核准并已完成实现规范和规划后，即进入了解决方案生命周期的实现阶段。实现是一组复杂的过程和步骤，需要仔细规划才能确保成功。实现包括以下任务：

- 构建网络和硬件基础结构
- 按照安装规划安装和配置软件
- 将数据从现有应用程序迁移到当前解决方案
- 实现用户管理规划
- 根据测试规划在测试环境中设计和部署试验性或原型系统
- 根据测试规划设计和运行功能性测试和负载测试
- 根据展开规划将解决方案从测试环境向生产环境中展开
- 根据培训规划培训部署的管理员和用户

有关部署设计实现的更多信息不在本指南讨论范围之内。但以下各节提供了有关这些任务的概要信息。

安装和配置软件

面向分布式企业应用程序的 Sun Java™ Enterprise System 的安装和配置需要对大量任务和步骤进行规划和协调。在部署设计阶段，基于高级部署体系结构创建安装规划，该规划提供安装 Java Enterprise System 软件所需的安装和配置信息。

此安装规划的重点包括：

- 确定安装的顺序和类型
- 检查主机以前安装的软件和安装准备情况
- 收集每个要安装的 Java Enterprise System 组件的配置信息

《Sun Java Enterprise System 2005Q4 安装规划指南》提供了关于如何收集安装规划信息的详细说明。《Sun Java Enterprise System 2005Q4 安装参考》提供了详细的配置信息以及可用于记录此信息的工作表。《Sun Java Enterprise System 2005Q4 安装指南》提供了对涉及多个 Java Enterprise System 组件的通用安装方案的指导。有关更多信息，参阅《Sun Java Enterprise System 2005Q4 安装指南》中的第 1 章 “准备安装”。

开发试验性和原型系统

Java Enterprise System 部署通常分为两类：一类是主要基于 Java Enterprise System 所提供服务的部署；另一类是需要大量与 Java Enterprise System 服务集成的定制服务的部署。可将前者视作一种 80:20 部署（80% 的服务由 Java Enterprise System 提供），同样可将后者视作一种 20:80 部署。

对于 80:20 部署，通常需在实现阶段开发用于测试的试验性部署。由于 80:20 部署使用的是提供即用性功能的成熟的 Java Enterprise System 服务，因此相对而言，试验性部署从开发、测试和修改步骤，最终进至生产部署阶段的速度较快。试验性部署不仅验证解决方案的功能，还提供关于系统运行状况的信息。

相反，20:80 部署引入了新的定制服务，这些服务不具有 80:20 部署所具有的互操作性历史记录。因此需要创建原型部署，它们是概念验证式部署，通常需要经历更为严格的开发、测试、修改过程，才能进入生产阶段。原型部署可以确定提议解决方案在测试环境中解决问题的效果如何。一旦原型部署论证了其功能完全可以达到要求，即可继续进行更严格的测试，然后进入试验性部署。

注 – 实际企业部署所需的定制服务开发的数量会有很大差异。如何使用试验性部署和原型部署进行测试取决于部署的复杂程度和性质。

测试试验性部署和原型部署

测试试验性部署和原型部署的目的是，在测试条件下尽可能确定部署是否既能满足系统要求，又可实现业务目标。

理想情况下，功能性测试应基于所有已确定的使用案例模拟各种方案，并且应开发一套标准来衡量符合性。功能性测试还可包含将系统有限地部署给选定的一组试用版用户，以确定其能否满足业务需求。

负载测试衡量在峰值负载下的测量性能。这些测试通常使用一系列模拟环境和负载发生器来衡量数据吞吐量和性能。部署的系统要求通常是设计和通过负载测试的基础。

注 – 对于系统要求未经明确定义、没有可作为估量基础的先前实现且需要进行大量全新开发的大型部署，功能性测试和负载测试尤其重要。

通过测试能够发现部署设计规范存在的问题，并可能要经过若干次反复设计、生成和测试，才能向生产环境展开部署。测试原型部署时，可能会发现部署设计中存在问题。在这种情况下，可以重新返回解决方案生成周期的先前阶段，以解决存在的问题。

确保在进入试验性部署前已对部署设计进行了全面测试。试验性部署表示已在前面的一系列测试中验证了部署设计。在试验性部署测试中发现的问题通常必须在部署设计的参数中加以解决。

由于测试永远不可能完全模拟生产环境，并且已部署解决方案的性质会发生演进和改变，因此应继续监视部署的系统，以确定是否有需要调整、维护或修补的部分。

展开生产部署

试验性或概念验证部署符合测试标准后，即可向生产环境展开部署。向生产环境展开部署通常分阶段进行。分阶段展开对会影响大量用户的大型部署具有尤其重要的意义。

分阶段部署可以先向一小部分用户部署，然后逐步扩大用户范围，直至将其部署给所有用户。分阶段部署也可这样进行：先部署一定类型的服务，然后逐步引入其余类型的服务。分阶段升级服务有助于隔离、确定和解决服务可能在生产环境中遇到的问题。

索引

拥

用

优

预

原

运

灾

展

术

组