

[日] NTT DATA集团 著

[杉原健郎 吉田一幸 岩崎贤治]
[三浦广志 吉田佐智男]

杨文轩 译

图灵程序设计丛书

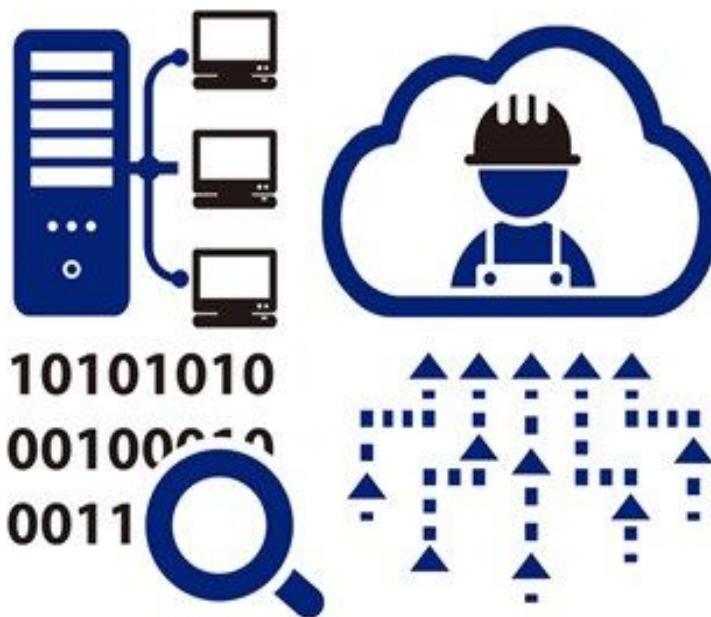
TURING

构建高可用性、高可扩展性、高安全性的IT系统

图解基础设施 设计模式

256张图表讲透
127个设计模式

集群 · 虚拟机管理器 · 实时迁移 · 快照
高可靠性核心交换机 · VLAN · 广域集群
DMZ · IDS · IPS · 生物认证 · 无共享
共享磁盘 · 访问控制 · 构成管理 · 监控
任务管理 · 时钟同步 · 报表 · BI
数据仓库 · 服务总线 · 数据中心
云 · 自动伸缩 · 备份站点



中国工信出版集团



人民邮电出版社
POSTS & TELECOM PRESS

版权信息

书名：图解基础设施设计模式

作者：[日] NTT DATA集团

译者：杨文轩

ISBN：978-7-115-38992-3

本书由北京图灵文化发展有限公司发行数字版。版权所有，侵权必究。

您购买的图灵电子书仅供您个人使用，未经授权，不得以任何方式复制和传播本书内容。

我们愿意相信读者具有这样的良知和觉悟，与我们共同保护知识产权。

如果购买者有侵权行为，我们可能对该用户实施包括但不限于关闭该帐号等维权措施，并可能追究法律责任。

图灵社区会员 张海川 (zhanghaichuan@ptpress.com.cn) 专享 尊重
版权

[版权声明](#)

[译者序](#)

[前言](#)

[本书特点](#)

致谢

第1章 什么是基础设施设计模式

1.1 什么是基础设施

系统 = 应用程序 + 基础设施

基础设施 = 硬件 + 操作系统 + 中间件

基础设施构建的难点

1.2 基础设施故障引发的后果

ANA 的事例

索尼电脑娱乐公司的事例

NTT DoCoMo 的事例

东京证券交易所的事例

First Server 的事例

1.3 基础设施构建中哪个阶段最重要

需求定义

设计

构建

测试

需求定义最重要

1.4 需求定义的重要性与难点

需求定义为什么很重要？

功能性需求与非功能性需求

基础设施需求定义的难点

1.5 非功能性需求等级

克服非功能性需求定义困难的工具

网罗一般的非功能性需求

定义非功能性需求的“程度”

1.6 基础设施设计模式

研究能够满足需求的设计方式的工具

特点

基于模式设计的步骤

风险因素的确认步骤——谨防遗漏

需求变更的应对步骤

1.7 本书的内容安排

第 2 章 可用性需求的实现策略：防止系统宕机

2.1 可用性策略的基础

预防故障

快速恢复

编写文档与持续改善

完整、彻底的确认

2.2 Web/AP 服务器的高可用性设计方式

容错服务器 / 大型机模式

会话共享的负载均衡模式

会话非共享负载均衡模式

备用机模式

各模式的比较结果与选择标准

注意点

2.3 DB 服务器的可用性设计方式

容错服务器 / 大型机模式

并列 DB 集群模式

N+1 集群结构模式

双机互备集群结构模式

各模式的比较结果与选择标准

注意点

2.4 虚拟服务器冗余的设计方式

集群软件模式

虚拟机管理器 HA 功能模式

实时迁移模式

各模式的比较结果和选择标准

注意点

2.5 LAN 的可用性设计方式

高可靠性核心交换机模式

动态路由模式

VLAN 模式

双机热备模式

各模式的比较结果与选择标准

注意点

2.6 WAN 的可用性设计方式

双网双工模式

双网热备模式

单网双链路模式

互联网 VPN 备用模式

ISDN 备用模式

各模式的比较结果与选择标准

注意点

2.7 互联网连接的设计方式

BGP 多宿主模式

双链路模式

单链路模式

各模式的比较结果与选择标准

注意点

2.8 数据备份的可用性设计方式

存储复制模式

SAN 存储复制模式

NAS 存储复制模式

SAN 快照模式

NAS 快照模式

业务服务器备份模式

- 业务 LAN 备份模式
 - 各模式的比较结果与选择标准
 - 注意点
 - 网络或存储装置的选择
- 2.9 灾害应对策略的设计方式
- 广域集群模式
 - 双系统热备 DR 模式
 - 降级热备 DR 模式
 - 备份转移模式
 - 远程镜像方式类型
 - 备份数据传送类型
 - 选择类型时的注意点
 - 各模式的比较结果与选择标准
 - 注意点

2.10 总结

第 3 章 安全性需求的实现策略：保护系统不受威胁

3.1 安全性策略的基础

- 攻击方式
- 加密
- 认证
- 访问控制

3.2 非法访问应对策略的设计方式

- 单防火墙 DMZ 模式
- 双防火墙 DMZ 模式
- DMZ + IDS 模式
- DMZ + IPS 模式

- 各模式的比较结果与选择标准
- 注意点

3.3 身份认证的设计方式

生物认证模式
一次性密码模式
IC 卡 + 密码模式
IC 卡 / 令牌模式
ID 密码模式
各模式的比较结果与选择标准
注意点

3.4 ID 管理和维护的设计方式
单点登录模式
统一 ID 的个别登录模式
个别 ID 的个别登录模式
各模式的比较结果与选择标准
注意点

3.5 信息泄露应对策略的设计方式
瘦客户端模式
外存管理模式
数据加密模式
数据分割模式
通信加密模式
各模式的比较结果与选择标准
注意点

3.6 总结

第 4 章 性能与可扩展性需求的实现策略：防止系统性能下降

4.1 性能与可扩展性策略的基础
规模调整的重要性
规模调整的可扩展性
规模调整的步骤
4.2 可扩展性策略的设计方式
扩容模式

升级模式
功能分割模式
集群模式
无共享模式
共享磁盘模式
各模式的比较结果与选择标准
注意点

4.3 超负荷应对策略的设计方式

访问控制模式
资源分割模式
并发数控制模式
资源结构变更模式
网络带宽控制模式
各模式的比较结果与选择标准
注意点

4.4 总结

第 5 章 运用与维护性需求的实现策略：不放过系统故障

5.1 运用与维护性策略的基础

系统监控

任务管理
备份管理
运维管理

5.2 运用与维护体制的设计方式

服务级别提升模式
服务级别管理模式
定期监控模式
各模式的比较结果与选择标准
注意点

5.3 构成管理的设计方式

隔离网络模式
代理软件构成管理模式
工具软件构成管理模式
各模式的比较结果与选择标准
注意点

5.4 系统监控的设计方式
可用监控模式
代理监控模式
资源信息保存监控模式
无代理监控模式
各模式的比较结果与选择标准
注意点

5.5 任务管理的设计方式
专业任务管理工具模式
附带任务管理功能模式
OS 任务管理功能模式
手动任务管理模式
各模式的比较结果与选择标准
注意点

5.6 时钟同步、杀毒软件更新的设计方式
后端网络自动同步模式
前端网络自动同步模式
手动同步模式
各模式的比较结果与选择标准
注意点

5.7 总结

第 6 章 基础设施构成的设计方式
6.1 Web 系统的网络构成的设计方式
4 网段构成模式

- 3 网段构成模式
- 2 网段构成模式
- 5 网段构成模式
- 各模式的比较结果和选择标准
- 注意点
- 6.2 存储设备构成的设计方式
 - SAN 模式
 - NAS 模式
 - DAS 模式
 - 存储设备连接形态的类型
 - 连接形态类型的注意点
 - 各模式的比较结果和选择标准
 - 注意点
- 6.3 报表生成的设计方式
 - 报表实时生成模式
 - 报表异步生成模式
 - 报表批处理生成模式
 - 各模式的比较结果和选择标准
 - 注意点
- 6.4 报表输出的设计方式
 - 报表服务器打印模式
 - 用户终端直接打印模式
 - 电子文档输出模式
 - 各模式的比较结果和选择标准
 - 注意点
- 6.5 报表基础设施配置的设计方式
 - 分布式管理 + 分布式印刷模式
 - 集中管理 + 分布式印刷模式
 - 集中管理 + 集中印刷模式

各模式的比较结果和选择标准

注意点

6.6 数据使用和信息分析的设计方式

数据集市 + BI 构成的数据仓库模式

BI 构成的数据仓库模式

EUC 模式

各模式的比较结果和选择标准

注意点

6.7 基础设施交互结构的设计方式

服务总线模式

数据中心模式

P2P 模式

各模式的比较结果和选择标准

注意点

6.8 总结

第 7 章 使用云计算服务的实现策略

7.1 云服务中性能与可扩展性的设计方式

服务器纵向伸缩模式

磁盘资源量增减模式

自动伸缩模式

任务应对横向伸缩模式

计划应对横向伸缩模式

手动横向伸缩模式

各模式的比较结果

注意点

7.2 云服务中备份的可用性设计方式

快照模式

数据中心复制模式

地区间高速数据传输模式

虚拟服务器复制模式

系统复制模式

各模式的比较结果

注意点

7.3 云服务中虚拟服务器的可用性设计方式

虚拟服务器冗余模式

路由变更模式

备份站点切换模式

固定 IP 替换模式

NAT 服务器冗余模式

虚拟磁盘替换模式

各模式的比较结果

注意点

7.4 总结

第 8 章 基于模式的设计实践

8.1 基础设施构成的讨论步骤

- ① 确认业务需求和涉众需求
- ② 提取功能性需求和非功能性需求
- ③ 识别风险和注意点，讨论对策
- ④ 设计系统的概念构成
- ⑤ 设计系统的逻辑构成

8.2 地理信息系统

- ① 确认业务需求和涉众需求
- ② 提取功能性需求和非功能性需求
- ③ 识别风险和注意点，讨论对策
- ④ 设计系统的概念构成
- ⑤ 设计系统的逻辑构成

8.3 综合 DB 系统

- ① 确认业务需求和涉众需求

- ② 提取功能性需求和非功能性需求
- ③ 识别风险和注意点，讨论对策
- ④ 设计系统的概念构成
- ⑤ 设计系统的逻辑构成

8.4 总结

版权声明

INFRA DESIGN PATTERN by NTT DATA Corporation, Takeo Sugihara, Kazuyuki Yoshida, Kenji Iwasaki, Hiroshi Miura, Sachio Yoshida

Copyright © 2014 NTT DATA Corporation

All rights reserved.

Original Japanese edition published by Gijyutsu-Hyoron Co., Ltd., Tokyo

This Simplified Chinese language edition published by arrangement with Gijyutsu-Hyoron Co.,Ltd., Tokyo in care of Tuttle-Mori Agency, Inc., Tokyo

本书中文简体字版由 Gijyutsu-Hyoron Co., Ltd. 授权人民邮电出版社独家出版。未经出版者书面许可，不得以任何方式复制或抄袭本书内容。

版权所有，侵权必究。

译者序

初识模式，是 2007 年在书店闲逛时无意间看到书架上陈列着的 Gof《设计模式》一书。书中四位作者以他们的睿智与经验总结出了 23 种设计模式，扩展了面向对象设计思路，令我爱不释手。2008 年赴大阪出差时亦随身带着此书，现在此书仍然摆在我的办公桌上，以便随时查阅。之后，随着项目类型的改变和自己在项目中角色的转变，又购买和学习了《重构与模式》《企业应用架构模式》。个人并非痴迷于模式，也反对模式污染，只是经验丰富的从业者总结和归纳出的模式，经常可以为我解决项目中遇到的困难指明方向。

但是以上书籍中所学到的设计模式只适用于应用程序，要想构建高可用性、高可扩展性、高安全性的 IT 系统，除了应用程序外，基础设施的设计也是非常重要的。特别是在编写 IT 集成解决方案时，对于经验不足的新手来说，设计和构建满足客户需求，且具有可用性、高可扩展性、高安全性的基础设施是非常困难的。

为了解决这个问题，来自日本 NTT 集团的五位著者调查和研究了几百个 IT 系统的基础设施，并整理出 118 个设计模式和 9 种设计方式，内容涵盖了可用性设计、性能与可扩展性设计、安全性设计、运用与维护性设计等各个方面，甚至还包括了使用时下最流行的云服务构建基础设施的设计方式。

想必大家还记得 2014 年 9 月末曝出的 Shellshock 漏洞吧。继 Heartbleed 漏洞之后，这个漏洞再次将 Linux 推向了网络安全的风口浪尖。我所就职的公司也经历了这场风波。该漏洞曝出后，某客户的 IT 系统部随即对其所有系统都进行了检查，发现 10 年前我们为该客户构建的一个系统中有此漏洞，并要求我们提交应对策略。显然，此时对应用程序层进行修改没有意义，必须对基础设施层做出改变。得益于本书，我和组员在查阅了当初的基础设施设计图之后，除了提出“修补 Shellshock 漏洞”建议外，还参考本书中的“安全性需求实现策略”章

节中的内容，提出了若干网络安全方面的改善建议，并针对每项建议详细地说明其优缺点，得到了客户的好评。

本书最大的特点是对所有设计模式和设计方式都配有示意图，读者可以很直观地理解各设计模式的特点。此外，对相同类型的设计模式，著者还非常体贴地以表格的形式整理出它们的选择标准和异同，读者只需逐一确认各项选择标准是否符合需求即可轻松地选择合适的设计模式。当然，与应用程序层的模式滥用会增加应用程序的复杂性一样，基础设施层的设计模式也有缺点。效果越好的设计模式，往往建设成本和运维成本也越高。著者在本书中多次强调，在选择设计模式时一定要考虑性价比，希望读者朋友们注意。

模式既是起点，也是终点。基础设施设计新手可以以模式为起点，学习各个模式中涉及的相关知识；老手则可根据自己的专业知识和经验，总结和归纳出设计模式。愿本书中的设计模式能为您的工作和学习有一些帮助。

在翻译过程中，身边许多人给予了我莫大的支持和鼓励。我的同事邵聪在 IT 系统基础设施构建与运维方面有着丰富的经验，为我理解本书内容提供了很多帮助。刚着手翻译本书时，女儿雨菓出世了，妻子徐文和父母、岳父母替我分担了照顾女儿的重任，使我有更多的时间来翻译本书。最后，还要感谢图灵出版社的编辑，正是他们指出了译文中行文的不足和内容的错误，才确保了本书的高质量。感谢你们，没有你们就不可能有本书。

杨文轩

2014 年 12 月 14 日

前言

基础设施设计模式将基础设施技术者们创造出的设计方式概念化、抽象化，同时将其命名，并记载了其特点与注意事项。目的是防止在基础设施开发中发生问题，提高基础设施的开发效率以及培养基础设施开发人才。基础设施设计模式特别适合对基础设施缺乏经验的技术者（也包括应用程序开发者）、负责系统构建的项目经理、在企业的信息系统部门中对系统规划与引进具有决策权的人，以及负责评估和比较供应商提案和设计的人。

现在，关于系统与基础设施开发的信息有很多，这些信息也很容易获取。因此，很容易让人觉得即使是不太具备基础设施开发技巧的人，也可以根据这些信息轻松地完成开发工作。但是大多数情况下，事实却并非那么简单。如果不具备足够的基础设施开发技巧，例如没有与用户等相关人员进行充分的沟通，就会开发出难以使用的系统。最终的结果就是增加了投资过剩，以及因需求定义不充分而导致系统运行后问题频发的风险。

高级技术者参与需求定义等上游设计，对于防止以上事态的发生是非常有效的。高级技术者对于各种需求都进行过类似的设计，同时还具备解决各种问题的经验。在大多数情况下，他们都可以应用从这些经验中总结出的技巧，高效地推进研讨进度。

那么怎样才能高效地、站在更高的层面上学习高级技术者的技巧呢？能够参与需求定义这种上游设计的人很少，能够从中吸取经验的人也很有限。此外，单纯地通过积累经验来提高技能也是有限度的。在计算机系统的适用范围飞速扩大，且越来越复杂和多样的今天，如何向刚刚从事基础设施开发工作的新人们系统地传授上游设计的技巧是一个需要解决的问题。

而解决这个问题的方法之一就是“设计模式”。设计模式是为了解决经常发生的问题而总结出的典型方法。在介绍设计模式的书籍中，最著名的莫过于《设计模式：可复用面向对象软件的基础》¹一书了。合理使用该书中介绍的设计模式，就可以提高开发质量与生产效率。而本书

将要介绍的基础设施设计模式，就是将这种适用于软件开发的设计模式技巧应用到基础设施的开发中去。

¹ 原书名为 *Design Patterns: Elements of Reusable Object-Oriented Software* , Eric Gamma、Ralph Johnson、Richard Helm、John Vlissides 著。4位合著者常被称为 GoF (Gang of Four) 。

笔者所属的 NTT DATA 集团每年研究与调查 100 多个系统中使用的技巧，并进行基础设施设计模式的开发，这些基础设施设计模式已经在数百个项目中被广泛使用了。实际上，我们也从用户那里得到了“很容易理解一般的基础设施应该有哪些选择项”“在确认设计是否妥当时发挥了很大作用”“能够防止在设计研讨过程中有遗漏并提高了开发质量”“对提高设计研讨速度有很大贡献”等赞誉，且 90% 以上的用户向我们反映使用基础设施设计模式有提高品质、回避风险、减少研讨工时等效果。

通过学习围棋与将棋中的“定式”²、空手道与柔道中的“型”³，可以采取基本措施应对各种状况。在基础设施开发中也是一样，以基础设施设计模式为基础进行设计，就像是高级基础设施技术者们在进行设计研讨一样。

² 定式是指经过棋手们长久以来的经验累积，从而形成的在某些情况下双方都会依循的固定下法。——译者注

³ 型是指前人根据其自身经验修炼而成的攻防招式，是对基础技能的应用动作，在固定的演武线上表现出攻击、防守、反击等技术。——译者注

本书特点

本书内容有以下两大特点。

- 所有的设计模式均不依赖于特定的产品

- 本书编写的基础是在需求定义等上游设计阶段中使用设计模式

关于第一点，本书将基础设施设计模式总结成为更加概念化的东西。这样，以基础设施设计模式为基础进行的设计，可以通过各种供应商的产品来实现。可以说，站在完全中立的立场，会更加容易讨论或者是评价、比较提案与设计是否完全符合需求。而且，虽然在计算机系统的业界，各种各样的产品不断地改变与升级，但是设计模式并不太容易因此而受到影响，具有长久的生命力。本书将介绍的就是具有这些特点的 127 个设计模式（119 个模式 + 8 种类型）。

关于第二点，需求定义等上游设计阶段对整个系统的质量会产生很大的影响。我们认识到了需求定义对于系统开发的重要性，并以提升需求定义的质量为目标来组织与编写本书。

为了可以在需求定义的非功能性需求研讨中使用设计模式，在本书中，我们按照可用性、性能与可扩展性、安全性、运用与可维护性等不同的非功能性需求，与云计算、网络结构、存储结构等不同的基础设施结构来分别讲解它们的设计模式。

为了更容易把握设计模式的特点，我们为所有的设计模式都配上了简洁的示意图。关于计算机系统，有许多术语似是而非。通过将本书作为“会话工具”使用，即便参加需求定义阶段的用户与供应商等相关人员都具有不同背景知识，也可以很容易达成共识。当全员达成共识后，就更容易针对需求讨论出最合适的设计和最适合的结构。即使是在完成需求定义后，由于环境变化等原因导致需求自身也发生了变化的情况下，因为需求与设计方式的关系很明确，也可以迅速地讨论出对策。

而且，对于各种设计模式，我们不仅阐述了它们可以“做什么”，也注明了它们“不能做什么”和“需要注意的事项”。根据这些内容，可以很容易地在需求定义等上游设计阶段中提前整理出风险，并提前制定出对

策。另外，通过使用本书来评审需求定义的结果，还可以确认研讨结果是否妥当、有无遗漏。

致谢

本书的撰写得到了敝公司的小林武博、寺西浩之给予的全面支持。此外、敝公司的风见纯、田村真浩、大西高史、西川治、三井吾朗、高桥宏明、冈安一将、吉田尚志、神谷慎吾、山下裕介、安田隆浩、堀江幸纪、中山伸、村山弘城、佐藤野英留、高桥优辅、蛇名高嗣、小田中忠雄、藤原慎和其他同事，以及包括普华永道业务咨询公司的酒井健一先生在内的许多技术顾问都给予了我们很大的帮助。没有他们，我们不可能完成本书的撰写工作。真的非常感谢他们。

为本书担任编辑工作的技术评论出版社的稻尾尚德先生指出了本书中的很多错误与不足之处。正是稻尾先生给予的莫大帮助，让本书的质量有了飞跃性的提高。这里要再次表达我们的谢意。

全体作者

2014年1月

本书中涉及的信息是在第一版发行时编写的，您在使用本书时，这些信息可能已经发生了变化。

本书作者、技术评论出版社、人民邮电出版社以及译者对于您应用本书内容所产生的后果概不负责。

本书中记载的公司名、产品名都是一般的商标，没有以™、©、®等符号表示出来。

请您在同意以上内容的基础上使用本书。如果没有阅读以上注意事项而联系我们，本书作者、技术评论出版社、人民邮电出版社以及译者

都不会予以回答，望请谅解。

第1章 什么是基础设施设计模式

自20世纪90年代开始，计算机系统呈现出开放的发展趋势，系统的应用领域也随之不断地飞速扩大。另一方面，由于越来越多的企业活动和社会活动离不开计算机系统，导致系统也变得越来越复杂和多样。在这种背景下，基础设施作为系统之根基，如何对其进行良好的设计与构建就显得尤为重要。

本章首先讲解什么是基础设施以及基础设施的重要性，然后介绍基础设施设计的辅助工具——“非功能性需求等级”与“基础设施设计模式”。

1.1 什么是基础设施

系统 = 应用程序 + 基础设施

计算机系统大体可分为“应用程序”与“基础设施”两部分（图1.1）。



图 1.1 系统的组成部分

应用程序既包括用于文字处理与表格计算的软件，也包括财务管理系统和销售管理系统等为特定的企业业务开发的软件。应用程序多以系统自身的名称来命名，大家平时操作得很多，理解起来应该不困难。

那么基础设施是什么呢？“基础设施”一词来源于英文单词 Infrastructure，表示基础结构的意思。在计算机系统中，基础设施指的就是系统的基础结构，即支撑起整个系统的部分。

同样使用了基础设施的术语是“社会基础设施”。社会基础设施指的是公路、铁路等交通网络，以及自来水管道、电力系统、燃气管道等关系国计民生的设施。“社会基础设施”作为支撑我们日常活动的基础，为我们创造了便利的生活环境。如果发生了交通被阻断、水电被切断等社会基础设施故障，将给我们的日常生活带来很大的影响。

计算机系统的基础设施也一样，它创造了可以让应用程序高效、稳定运行的环境。如果基础设施发生了故障，则会给整个系统带来很大的影响。

基础设施 = 硬件 + 操作系统 + 中间件

基础设施由服务器和网络设备等“硬件”、Windows 等“操作系统”以及数据库（以下称为 DB）等“中间件”构成。

正如操作系统也被称作“基础软件”，它只具有最基本的功能。因此，名为中间件的软件被加入到操作系统与应用程序之间，为应用程序提供了操作系统所不具备的功能。典型的中间件有 RDBMS (Relations Database Management System, 关系数据库管理系统)，这是一种可以创建出 DB 的中间件，在 DB 中可以很容易地对大量数据进行检索、提取、更新、删除等操作。通过使用 RDBMS，应用程序可以轻松地从大量数据中提取和更新所需数据。

基础设施构建的难点

近年来，在计算机系统业界中，硬件与中间件的规格正在逐渐被公开和标准化。得益于这种趋势，我们可以自由地组合各厂商的硬件与中间件来构建我们的基础设施。

虽然标准化工作一直在持续推进，但各厂商的产品仍然有其各自的特点与缺点，并非无论怎样组合这些产品都能够确保系统无故障运行。当然，简单的基础设施最近已经能够很容易地构建并运行，但是对于需要在全公司都引入的大型系统和具有很大社会影响力的系统，其构建要复杂得多，即使尝试着将这些产品组合使用，也会时常发生没有如预想那样正常工作的情况。

为了找出没有正常工作的原因，需要花费大量的时间来进行各种测试与确认。即使现在找出了原因，以后也可能会发生其他问题。因此，要想构建合适的基础设施，构建者必须要有专业的知识、高超的技能和丰富的经验，但这类人才非常稀少。更糟糕的是，基础设施的结构每年都在变得越来越复杂。超过半数的企业活动无法在没有系统的情况下进行，有的企业甚至在机房中放置着几百台服务器，运行着很多系统。各系统又与其他多个系统之间通过数据交互进行协作业务处理，而这些交互系统中既有老系统，也有最近引入的新系统。不仅如此，这些系统的基础设施中还使用了各种各样的产品，产品的版本也各不相同。总之，要针对以上情况来构建与运行基础设施是一件非常困难的事情。

1.2 基础设施故障引发的后果

如前所述，一旦基础设施发生故障，会对系统运行造成很大的影响。如果该系统在企业的业务活动中承担了重要的任务，那么损失就会相当巨大。而如果是具有很大社会影响力的系统发生故障，甚至会引发社会问题，降低企业的信用度。

ANA 的事例

2008 年 9 月 14 日，全日空航空公司（ANA）的国内航线旅客登机系统发生了故障。根据日本国土交通省发表的消息，“9 月 14、15 日这两天的时间里，共造成 63 个航班停飞，356 个航班延误，约 7 万旅客的行程受到影响”¹。根据 ANA 发表的调查结果，本次故障的原因是加密认证超过了有效期限²。

¹ <http://www.mlit.go.jp/common/000023462.pdf>

² http://www.ana.co.jp/ana-info/ana/csr/report/pdf2009/CSR2009_all.pdf

加密认证是系统安全性策略之一，主要由基础设施负责处理。因此，本次故障可以认为是基础设施的不完善引起的。

索尼电脑娱乐公司的事例

2011 年 4 月 27 日，索尼电脑娱乐公司（Sony Computer Entertainment）发表声明，“由于 PlayStation Network 和 Qriocity 遭到非法入侵，2011 年 4 月 17 日至 19 日，用户的账号信息可能已经被泄露”³。根据这份声明，泄露（被非法入侵者截取）的账号信息包括姓名、住址、电子邮件地址、出生年月日、密码以及线上 ID 等。

³ http://cdn.jp.playstation.com/msg/sp_20110427_psn.html

防止非法入侵也是系统安全性策略之一，该事件很可能也是由基础设施的原因引起的。

NTT DoCoMo 的事例

2012 年 1 月 26 日，鉴于 NTT DoCoMo 提供的移动电话服务中发生的事故，日本总务省对 NTT DoCoMo 进行了行政指导。根据总务省发表的声明，在招致行政指导的事故中，2012 年 1 月 1 日发生的智能手机

收发邮件困难的事故，其原因是“向保存了用户邮箱信息的服务器发送的查询请求数量超过了允许同时访问的上限值，导致服务器的处理能力大幅下降”⁴。

⁴ http://www.soumu.go.jp/menu_news/01kiban05_02000017.html

事故原因是服务器的处理能力（性能）不足，因此，我们推测这也是基础设施的原因引起的。

东京证券交易所的事例

2012年8月7日，东京证券交易所（东证）金融衍生品交易系统的网络设备发生了硬件故障，导致衍生品交易不能继续进行。根据东京证券交易所发表的声明，“系统中发生故障的网络设备采用了冗余配置，当生产环境中的1号机发生故障时，会自动地切换到备用的2号机上以保证业务可以继续进行，但是当时自动切换处理并没有正常工作”⁵。

⁵ <http://www.tse.or.jp/news/30/b7gje6000002s6l2-att/b7gje6000002s6o9.pdf>

设备发生故障时确保系统能够继续提供服务的工作主要由基础设施来负责。因此，本次故障也是由基础设施的原因引起的。

First Server 的事例

2012年6月20日，雅虎子公司First Server所提供的服务器托管服务中，发生了大量顾客数据消失的事故。根据第三方调查委员会的调查报告显示，“事故的直接原因是工作人员在进行系统升级的时候没有遵守公司制定的操作手册，而是使用了自己制作的更新程序”⁶。第三方调查委员会还指出最根本的原因是“开发与运维体制的问题”。

⁶ <http://support.fsv.jp/urgent/pdf/fs-report.pdf>

运维体制也是基础设施的重要因素之一。也就是说，该故障也是由基础设施的原因引起的。

还有其他许多事例也是因为基础设施发生故障而引发了社会问题。即使是那些所谓的“大企业”，想要在复杂的系统中构建基础设施并保持其高效稳定地运行，也是非常困难却又非常重要的事情。

1.3 基础设施构建中哪个阶段最重要

怎样才能构建出可以为系统创造高效稳定运行环境的基础设施呢？前文说过需要构建者具备专业的知识、高超的技能和丰富的经验，但是这样的高级技术人员很稀少。为了找到解决办法，首先我们得来看看系统的开发方式。

与建造房屋的方式非常类似，系统的构建也大致分为以下 4 个阶段。

需求定义

第一个阶段是“需求定义”。在此阶段中，明确（= 定义）想要什么样的系统（= 需求），并定义必要的构成要素（设计方式）。以建造房屋来打比方的话，需求定义就是明确想要建造什么样的房屋的阶段，比如“想要有书房”“想要有露天浴池”“想要有宽敞的客厅”等，可以尽情地说出自己的想法。但是，考虑到有限的预算，需要将需求分为“必要需求”与“可选需求”，以确保成本不会超出预算。

设计

第二个阶段是“设计”。这个阶段决定系统的具体构造和规格，使系统能够满足需求。以建造房屋来打比方的话，这个阶段就是通过画出详细的设计图、制作建筑模型等方式来确认房屋是否与需求相符，并决定其结构。在这个阶段中，还要讨论墙壁与地板的材料，在厨房与浴室、卫生间等处都要使用哪个厂家的产品。

构建

第三个阶段是“构建”。采购机器，按照设计的构造进行组装，安装软件并进行相关的设定。以建造房屋来打比方的话，这就是“施工”阶段。

测试

第四个阶段是“测试”。如果在测试阶段没有问题，系统就可以在生产环境中运行了。以建造房屋来打比方的话，就是在竣工（施工完成）后，进行工程验收（房屋是否与合同、设计相符，是否有施工缺陷等）。如果没有问题，则将房屋交付给委托方。

需求定义最重要

总而言之，计算机系统就是通过整理想实现的内容（需求定义）、描绘设计图（设计）、组装（构建）、确认是否完成了想实现的内容（测试）这 4 个阶段来构建的，基础设施也是在这 4 个阶段中构建的。

其中最重要的是“需求定义”阶段，也称作“上游设计”阶段。如果不能正确地定义想实现的内容，不论是下游设计还是构建都无法进行。也就是说，如果在上游设计阶段发生了遗漏，是不可能构建出符合需求的系统与基础设施的。

下面，我们就来讲解一下需求定义为什么很重要，并介绍在不具备专业知识、高级技能、丰富经验的情况下，也能在一定程度上妥当进行基础设施需求定义的辅助工具“非功能性需求等级”与“基础设施设计模式”。

1.4 需求定义的重要性与难点

需求定义为什么很重要？

需求定义为什么很重要呢？

系统的构建与运维涉及许多不同职责的相关人员，包括系统的直接用户、设计与构建系统的专家、监控系统运行状况并在发生故障时进行紧急处理的运维管理人员等。设计与构建者、运维管理者如果在未与用户达成共识的情况下进行构建、运维工作，就无法确保系统可以高效稳定地运行。

即使是在用户内部，大家的立场也各不相同。其中有系统的直接用户，也有使用系统执行业务的负责人。由于这些用户使用系统的目的与所处立场不同，所以他们对系统会有不同的看法，有些看法甚至互相矛盾。系统的直接用户可能会希望系统的操作尽可能地简单，但业务负责人可能会觉得操作上难以理解一点也无妨，一定要优先确保业务的执行效率。如果用户内部对系统的要求都各不相同，那么系统的设计与构建者、运维管理者就更不知如何是好了。

为了使系统相关人员就“到底需要什么样的系统（=需求）”达成共识并使构建、运维工作能够顺利进行，需要确保大家对于需求的理解一致并在文档中明确记录下来，这项工作就是需求定义。

根据日本信息服务产业协会的调查，在影响系统质量的关键因素中，排在第一名的就是“需求定义”（图 1.2）。也就是说，系统的质量取决于定义系统需求的这个阶段。对系统而言，需求定义是最重要的工作。

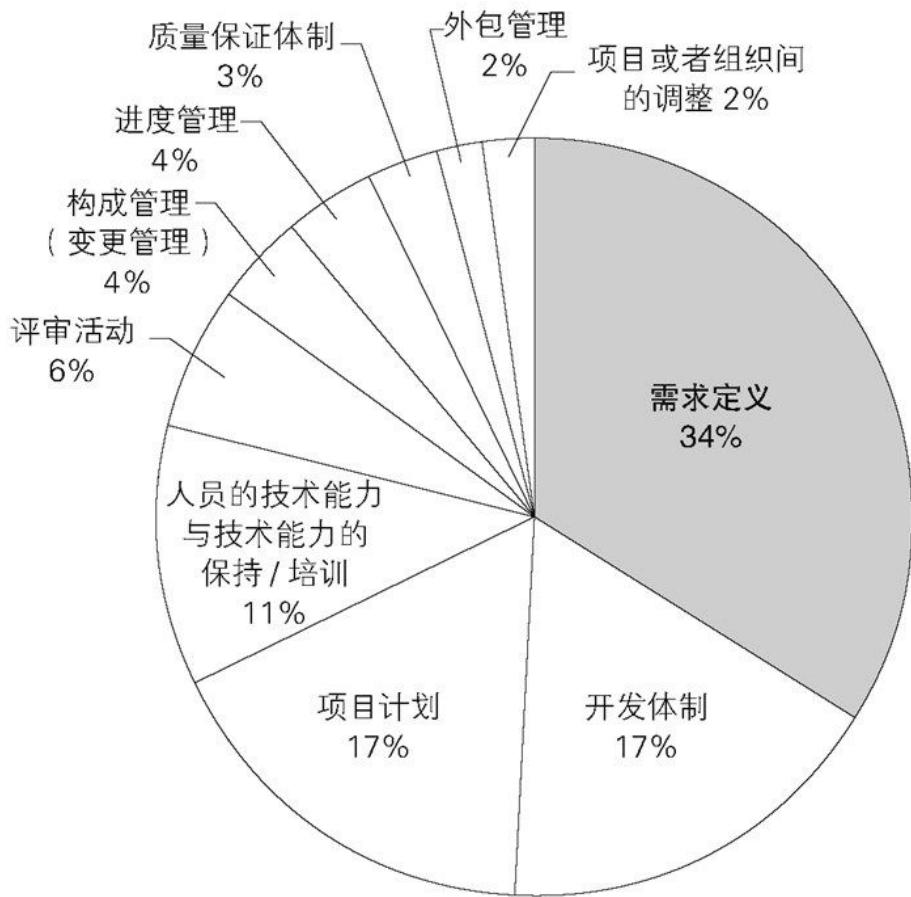


图 1.2 影响质量高低的关键因素

2006 年，在《JISA 会报 No.83》中刊登了信息服务产业协会技术委员会软件工程部编写的《信息服务产业中软件开发实际情况的问卷调查结果概要》，我们将其中 P.79 的数据以饼状图的方式展现出来

功能性需求与非功能性需求

如前所述，需求对系统质量有很大影响。为了定义需求，首先要将用户对系统的“功能性需求”与“非功能性需求”提取出来。

功能性需求是指关于“想通过系统做什么”的需求。例如“想要扫描商品的条形码，价格就会在终端上显示出来”“想要统计昨天每个小时的销售

额”等内容都是功能性需求。因为功能性需求与客户的日常工作有直接联系，所以很容易定义。

而所谓非功能性需求，顾名思义，指的是并非功能性的需求，即功能性需求以外的所有需求。在构建系统时，与相关人员仅仅就“想通过系统做什么”达成共识是不够的。如系统的运行时间段（系统提供服务的时间段）、保护系统不受非法攻击的策略（安全性策略）等需求，虽然不属于功能性需求，但对系统而言也是非常重要的。

以大型国际体育活动的票务预订系统为例，“可以预订比赛门票”“座位不能被重复预订”“分配预订座位时不能浪费座位”等想通过票务预订系统完成的工作就是功能性需求。而“可以 24 小时预订（票务预订系统提供服务的时间段为 24 小时）”“对用户进行身份识别以防止恶意预订”“预订处理的响应时间（预订按钮按下后到预订处理完成的时间）不能超过 10 秒”等功能性需求以外的系统特性需求都是非功能性需求。

在系统中，主要由应用程序实现功能性需求，由基础设施实现非功能性需求。因此也可以认为，非功能性需求是应用程序高效稳定的保证。

基础设施需求定义的难点

定义系统的非功能性需求是一项困难的工作。功能性需求是“想通过系统来做什么”，因为与日常工作有着直接的联系，所以包括用户在内的系统相关人员很容易定义。但是非功能性需求是为了能使系统高效稳定地运行而假想出来的条件，不容易完整地整理出来。即使毫无遗漏地列举了所有需求，还必须要准确地表达出“达到什么程度”。从系统用户的角度看，他们很容易提出“尽可能”不让系统停止和“大致”实现用户希望的响应速度等不明确的需求。“尽可能”和“大致”究竟是什么程度，即使对是同一个系统，不同的人也有不同的理解，这种情况就很难达成共识。如果继续向他们追问“具体是什么程度”，也几乎没有人能回答出来。如果提取出的需求太低，很容易在系统运行后产生重大问题；

而如果提取出的需求太高，则会导致预算增加。恰当地把握需求的程度是一件非常困难的工作。

以前文提到的票务预订系统为例，如果不能很准确地定义允许同时访问的人数上限，可能会因访问人数过多而导致系统宕机。此外，还必须决定当访问的用户数量超过允许的人数上限时应当如何处理。如果仅仅是将用户转送到“当前浏览网站的用户数量过多，网站负荷较大，请稍后再访问，谢谢谅解”的画面，是无法改变许多用户仍然无法预订的状况的。如果不能解决根本问题，只会让想要预订的用户更加焦急。诸如类似的情况，即使注意到了这些非功能性需求，如果不采取相应的策略（基础设施的设计方式），也还是不能解决根本问题，只会导致问题的恶化。

1.5 非功能性需求等级

克服非功能性需求定义困难的工具

实现非功能性需求的是基础设施。如果不能完整并准确地定义非功能性需求的程度，就不能构建合适的基础设施。那么怎样才能完整并准确地定义非功能性需求呢？为了解决这个问题，我们开发了“非功能性需求等级”工具。

非功能性需求等级是由 NTT DATA、富士通、日本电气、日立制作所、三菱电机信息系统、冲电气工业等 6 家日本 SI (System Integration, 系统集成) 业务公司在“系统基础发包方需求可视化的非功能性需求等级讨论会”⁷ 上得出的成果，现已在 IPA (Information-technology Promotion Agency Japan, 日本信息处理推进机构) 上进行管理，并免费公开⁸。

⁷ <探讨“实现客户的业务系统时，就对系统的实现水平有很大影响的响应速度等性能需求，以及发生故障时的容错性需求等典型非功能性需求与用户、供应商达成共识的方法”的讨论会。
<http://www.nttdata.com/jp/ja/nfr-grade/> >

⁸ 非功能性需求等级的中文版下载地址：

<http://www.ipa.go.jp/sec/softwareengineering/reports/20130329.html>。——译者注

网罗一般的非功能性需求

非功能性需求等级定义了 236 项非功能性需求指标。由于是 6 家日本 SI 业务公司利用他们的丰富经验定义的指标，所以具有一定的全面性。如果能逐一确定每个指标，就可以防止在讨论中发生遗漏。我们将这 236 个非功能性需求指标分为大项目、中项目、小项目，并进行了整理。这样，即使对非功能性需求不熟悉的用户也可以很容易地理解其中的内容。其中大项目分为“可用性”“性能与可扩展性”“运用与维护性”“可移植性”“安全性”“系统环境与生态环境”。各大项目的需求范例参见表 1.1。

表 1.1 非功能性需求等级的 6 大项目

大项目	内容	范例
可用性	关于系统服务运行与停止的需求	<ul style="list-style-type: none">● 关于系统运行时间与停止时间的需求● 发生故障导致业务停止时，在哪里、做什么、做到什么程度能使系统服务恢复相关的需求● 关于数据保护的需求
性能与可扩展性	关于系统的处理能力及其可扩展性的需求	<ul style="list-style-type: none">● 关于 Web 系统每秒可以处理的请求数量的需求● 关于可以同时访问系统的用户数量的需求
运用与维护性	关于维持系统长时间运行所要进行的运维工作，与维持系统服务质量所必需的监控工作的需求	<ul style="list-style-type: none">● 关于运维制度的需求● 关于可以进行运维处理的时间段与运维处理所需时间的需求

大项目	内容	范例
可移植性	关于将旧系统的资源移植到新系统的需求	<ul style="list-style-type: none"> ● 关于移植设备与移植数据量的需求 ● 关于移植代码行数的需求
安全性	关于系统中所处理的信息的安全性需求	<ul style="list-style-type: none"> ● 关于限制信息读写的需求 ● 关于是否允许从外部访问系统的需求
系统环境与生态环境	关于系统物理环境与生态环境的需求	<ul style="list-style-type: none"> ● 耐震/防震、重量/空间、温度/湿度、噪音等物理环境相关的需求 ● 系统的二氧化碳排放量、能源消耗等生态环境相关的需求

定义非功能性需求的“程度”

非功能性需求等级不仅完整地列举了非功能性需求指标，还将各非功能性需求指标的实现级别划分为从 0 到 5 的 6 个等级，并整理出每个等级的具体内容。非功能性需求等级并非采用提问形式让用户“从零开始考虑”，而是采用选择方式，让用户从 6 个选项中选出与自身需求相适应的等级来定义非功能性需求。表 1.2 中记录了大项目中可用性、中项中连续性、小项目中运维计划的指标的实现等级。

表 1.2 实现等级范例

指标	等级					
	0	1	2	3	4	5
运行时间（通	无规定	工作时间内 (9点~17)	仅夜间停 止(9点	约1小时的停止 时间(9点~次	短暂停止(9 点~次日早8	24 小

常)		点)	~21点)	日早8点)	点55分)	时 运 行
运行时间 (指定日)	无规定	工作时间内 (9点~17点)	仅夜间停止 (9点~21点)	约1小时的停止时间 (9点~次日早8点)	短暂停止 (9点~次日早8点55分)	24小时运行
有无停止计划	有停止计划 (可变更运维计划)	有停止计划 (不可变更运维计划)	无停止计划	—	—	—

使用非功能性需求等级可以很容易全面、正确地定义非功能性需求。也许有人会觉得，“236个非功能性需求指标”数量太多，而且对于自己能否正确地从中选出适合自己系统的等级感到不自信。但是请放心，在我们设计这套非功能性需求等级内容的时候，就已经考虑到让所有用户都能做出恰当的选择。在 IPA 上公开的说明资料“非功能性需求等级使用向导《使用篇》”⁹ 中，也有关于“能让用户做出恰当选择的设计方法”的详细说明。

⁹ <http://www.ipa.go.jp/files/000026809.pdf>

在使用非功能性需求等级全面地、正确地定义非功能性需求后，还必须将定义好的非功能性需求正确地落实到基础设施的设计中。

1.6 基础设施设计模式

研究能够满足需求的设计方式的工具

为了能让非功能性需求落实到基础设施的设计中，NTT DATA 开发了基础设施设计方式集合，也就是本书谈及的基础设施设计模式。基础设施设计模式是给基础设施设计人员过去创造出的设计方式命名（模式名），并总结其特点与注意事项而形成的集合体。基础设施设计模

式被应用在 NTT DATA 集团的许多项目中，具有提高质量、回避风险、降低讨论工时的效果。我们也从用户那里得到了“很容易理解一般的基础设施应该有哪些选择”“在确认设计是否妥当时发挥了很大作用”“能够防止在设计研讨过程中有遗漏并提高了开发质量”“对提高设计研讨速度有很大贡献”等赞誉。

本书将介绍多个与非功能性需求（非功能性需求等级中的大项目）策略、基础设施构成要素相关的常用设计方式。例如在非功能性需求策略中，有可用性策略、安全性策略、性能与可扩展性策略等。而在基础设施的构成要素中，有 Web 系统的构成与网络的构成、存储的构成等。

基础设施设计模式中的设计方式与住宅的设计方式类似。住宅设计中，在基本确定了户型之后，就可以选择必需的建筑构件。这些建筑构件就是建筑公司根据以往建造住宅的经验制作出的模型（模式）。在这些模式中浓缩了建筑公司长年积累的技术与智慧，通过选择与组合这些模式，就可以建造高品质的住宅。基础设施设计模式也一样，是长期从事基础设施构建的设计人员根据他们的经验提炼出的模型。在基础设施设计模式中，浓缩了设计人员的技术与智慧，通过组合这些设计模式，就能以低廉的成本构建出高质量的基础设施。

基础设施设计模式有以下几个特点。

特点

一目了然

在基础设施设计模式中，我们将基础设施的设计方式以示意图展现出来。通过查看示意图，任何人都能很容易地理解设计方式的构造与特点。

而且我们还以“○、×”等表现方式做出了与其他设计方式进行比较的表格，使人很容易理解它们之间的不同。

记载了选择标准

在基础设施设计模式中，我们记录了众多设计者在选择设计方式时所讨论的项目（选择标准），以及各种设计方式可以多大程度地符合标准。即使是对基础设施构建缺乏经验的人，也能借助众多设计者的智慧结晶来选择设计方式。

另外，设计方式的比较项目与选择标准项目都采用了非功能性需求项目，因此很容易据此选择合适的设计方式来实现非功能性需求。例如，Web/AP 服务器的可用性设计方式的选择标准采用了“故障发生时对系统服务有无影响”“允许在多长时间内恢复业务”“在业务完全恢复前性能下降的程度”等非功能性需求项目。如果系统需求是“不允许故障导致服务停止和性能下降”，那么根据比较表可以选择“容错服务器 / 大型机模式”的设计方式来实现该需求。

不依赖于特定的产品

各种设计方式都不依赖于特定的产品。有许多产品以及产品组合都可以实现基础设施设计模式，因此设计模式不会受产品变化的影响，用户可以选择最合适（可以满足条件且性价比高）的产品来实现需求。

记载了缺点与注意事项

在基础设施设计模式中，不仅记载了设计方式的优点，也记载了其缺点与注意事项。因此，用户很容易知道所采用的设计方式有何风险，并能尽早制定出风险的应对策略。

基于模式设计的步骤

根据基础设施设计模式中记载的选择标准，设计人员可以很容易地讨论出基础设施的构成。而这项工作就称为“基于模式设计”（图 1.3）。



图 1.3 基于模式设计的步骤

① 确认需求

根据用户的要求，通过使用非功能性需求等级等方法整理出非功能性需求并确认其内容。同时还要确认成本、交付日期、相关法律法规、技术标准和数据中心地点等制约条件。

② 选择设计方式

选择适合需求的设计方式，并讨论所选的设计方式是否满足制约条件。要想选择出合适的设计方式，不能只参考选择标准，还需要事先通过示意图和比较表来理解设计方式的特点与注意事项。

如果难以选择出设计方式，则应该根据选择标准中的项目对需求进行更深入的讨论。另外，请事先记录下所选择的设计方式与需求和制约条件的关系，以便能更容易地追踪设计阶段之后发生的需求变更。

③ 讨论基础设施构成

将所选择的设计方式组合起来，并绘制基础设施构成图。因为组合设计方式时可能会有需要注意的地方，因此一定要确认好各方式的特点与注意事项。

④ 确认基础设施构成

从以下问题出发，与相关人员确认基础设施构成的讨论结果。

- 所考虑的制约条件是否妥当
- 是否完全反映了需求和制约条件
- 经营战略与信息战略是否匹配
- 能否很容易地看出根据什么需求或者制约条件选择了现在的设计方式
- 能否从理论上说明为什么现在的这种设计方式是最好的
- 是否确认了故障应对策略与备份功能
- 是否提取了风险因素并充分讨论了风险应对策略

关于风险因素的提取方法请参考以下内容。如果确认结果是还需要对设计进行变更或是继续讨论，可以回溯到之前的步骤重新进行讨论。

风险因素的确认步骤——谨防遗漏

即使是设计方式与产品的组合，也会有不能完全满足基础设施需求的情况。而且，采用了某种设计方式后，在构建和运维过程中也会有需要特别注意的地方。因此，我们在基础设施设计模式中记载了各种设计方式的缺点与注意事项。这样，通过使用基础设施设计模式来构建基础设施，可以有效地防止讨论时产生遗漏（图 1.4）。



图 1.4 风险因素的确认步骤

① 提取风险因素

参考比较表、选择标准中的注意事项，提取设计方式的缺点。模式比较表中用“△”与“×”标记的项目表示该项目不能满足模式选择标准。为了防止遗漏，即使只有一点相关的事项也要提取出来。

然后讨论提取出来的事项中有哪些可能成为风险。在提取风险因素的过程中，可以通过向经验丰富、具有专业知识的人员咨询意见来防止遗漏。

② 整理风险

将提取出来的风险因素整理成为设计阶段应当讨论的风险与后续工作阶段应当注意的事项两部分。在设计阶段必须讨论的风险包括：设计方式的组合不能满足需求的风险、与可行性有关的风险（在某些情况下实施困难的可能性）、使用的产品价格过高以及构建和运维成本增加等与成本相关的风险、构建和运维工作效率下降的风险等。后续工作阶段应当注意的事项包括在产品的选择、构建、系统运行后的运维等各个工作阶段需要注意的事项。

③ 讨论风险应对策略

针对整理出的风险，讨论采取改变设计方式的组合等方法来回避、减轻这些风险可能带来的影响。与提取风险一样，可以通过向经验丰富、具有专业知识的人员咨询意见来制定出更有效的应对策略。

需求变更的应对步骤

系统的业务环境是瞬息万变的，随之而来的是需求的变更。而当需求发生变更时，就需要重新审视基础设施了。基础设施设计模式通过明确需求、制约条件与基础设施构成要素之间的关系，可以更容易地应对业务环境变化带来的需求和制约条件的变更（图 1.5）。

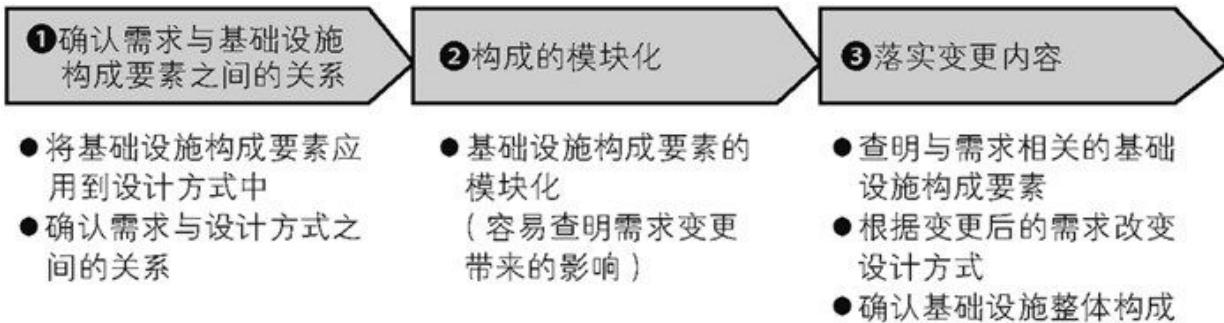


图 1.5 需求变更的应对步骤

① 确认需求与基础设施构成要素之间的关系

确认需求和制约条件与基础设施构成要素之间的关系。如果关系很模糊，可以使用基础设施设计模式进行整理。

首先，将变更对象的基础设施构成要素应用到设计方式中。通过比较基础设施构成图与设计方式示意图，找出合适的设计方式，并将基础设施分解为构成要素。

接下来，根据比较表与选择标准中记载的设计方式的特点等设想基础设施构成要素的需求，并与需求定义书中的需求进行匹配。如果设计方式不符合任何需求，可以根据当时的业务环境等推测并添加需求（添加的需求必须在需求定义书上与其他需求区分开）。

② 将构成模块化

归纳出基础设施构成要素中紧密关联的要素，并将它们视为基础设施构成中的“模块”（模块化）来应用，如“Web 服务器”“应用程序服务器（以下简称 AP 服务器）”“DB 服务器”“备份”“系统监控”“作业管理”“网络”“报表”等。

通过将基础设施的构成模块化，相关人员将更容易确定受到需求变更影响的范围。

③ 落实变更内容

查出与发生变更的需求相关联的基础设施构成要素，并改变设计方式以适应新的需求。改变设计方式后需要再确认一遍基础设施的整体构成。例如当发生变更的部件是“备份”模块时，需要讨论现在的备份产品是否满足新的需求。

1.7 本书的内容安排

本书从第 2 章至第 5 章，介绍非功能性需求（非功能性需求等级中的“大项目”）策略的设计方式。第 6 章介绍网络和存储、报表等基础设施构成要素的设计方式。第 7 章介绍云计算的基础设施设计模式。最后，第 8 章介绍使用前几章讲解的设计方式进行系统设计的几个实例。

此外，关于非功能性需求等级中包含在大项目中的可移植性策略，虽然最困难的是数据移植，但是只要在早期阶段采取应对措施就可以有效地提高数据移植的质量，所以本书基本不会提及。另外，在系统环境与生态环境方面，构建耐震 / 防震的系统环境、使用 CO₂ 排放量小和节能的数据中心、设备等是很有效的策略，所以本书也基本不会提及。

第 2 章 可用性需求的实现策略： 防止系统宕机

可用性需求是指与系统的运行效率相关的需求。在一定的时间内，系统实际运行的时间所占的比率称为运行效率。系统对社会活动和企业商业活动的影响越大，所要求的运行效率就越高。

有许多策略可以提高系统的运行效率，例如硬件冗余和引入运维工具以防止误操作等。究竟哪种策略最适合我们的系统，则必须根据可用性需求来正确地从中进行选择。另外，在实施策略时，无论是硬件冗余还是引入运维工具，都会产生额外的费用，因此，还需要考虑费用与效果的性价比。

在本章中，我们将要讲解可用性策略的基本思考方法和设计方式。

2.1 可用性策略的基础

我们将发生故障时可导致系统整个停止的部件称为 SPOF (Single Point Of Failure, 单点故障)。排除 SPOF 是提高系统可用性的典型策略，通过排除 SPOF，即使某处单独发生了故障，也可以避免系统停止、无法继续提供服务（系统宕机）的状况。

但是，仅仅排除 SPOF 还不足以完全避免系统宕机。根据《日经电脑》的调查，引起系统故障发生的原因多种多样，其中运维和设定错误（粗心的错误）占 49%、软件缺陷占 30%、硬件故障占 12%¹。由此可看出，要想提高系统的可用性，仅仅针对基础设施领域制定策略是不够的，还需要针对应用程序和系统运维方面来综合地制定策略。

¹ 消除粗心的错误，日经电脑，日经 BP 出版社，2008 年 7 月 15 日，第 39 页。

制定可用性策略的方法有很多种，但是还需要考虑如图 2.1 中所示的费用与效果的性价比。基本方针是不要一开始就采用高成本的策略，而是优先实施低成本且比较有效果的策略。以容灾策略为例，这种策略是指为了使系统能在灾害发生时继续提供服务，在异地部署一套备用系统。与其达到的效果相比，成本是很高的。因此应当优先考虑引入作为国际最佳实践的 ITIL (Information Technology Infrastructure Library, 信息技术基础架构库) 系统运维服务和获取可靠备份等其他策略。

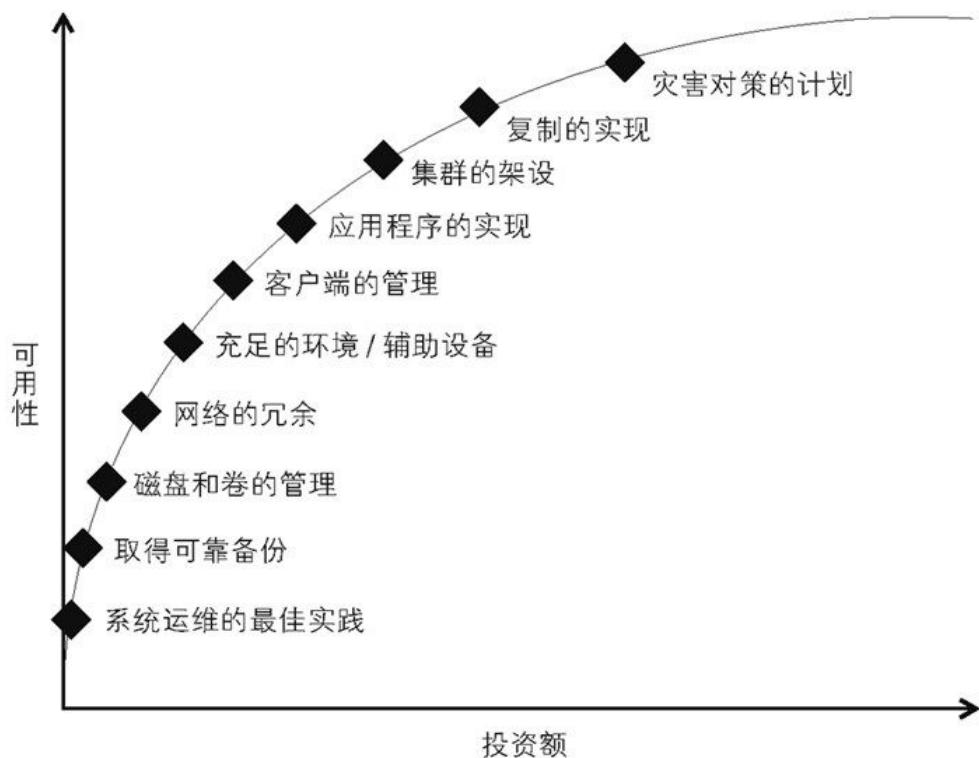


图 2.1 可用性策略的级别与投资额的相关性

出处：Evan Marcus、Hal Stern, Blueprints for High Availability Second Edition, Wiley Publishing, Inc., 2003, p.474（著者译）²

² 中文版名为《高可用性系统设计》，清华大学出版社于 2005 年出版，汪青青、卢祖英译。
——译者注

为了提高系统的可用性，在进行“预防故障”（防患于未然）和“快速恢复”（事后应对）的准备的同时，还必须提高和改善作为可用性根基的系统构建与运维过程的质量，例如编写文档并持续改善，以及进行完整、彻底的确认等（图 2.2）。通过这些影响系统可用性因素的作用，可以有效地将系统整体的可用性维持在一个高水平上。

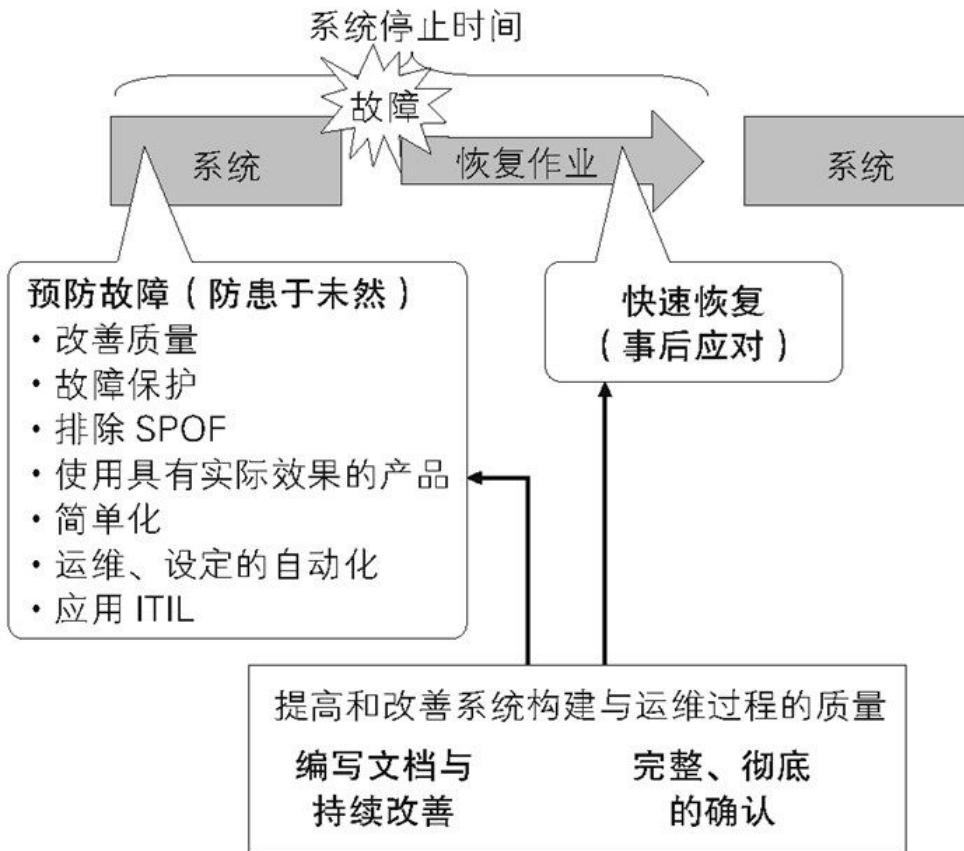


图 2.2 可用性策略的全局图

预防故障

预防故障是指尽可能地防止故障发生，是提高系统可用性的根本策略。在本策略中也包含了将故障影响范围缩至最小的策略。为了预防故障，需要采取“改善质量”“故障保护”“排除 SPOF”等措施。

改善质量

从设计阶段开始，就不断地对应用程序设定质量目标，并逐步实现这些质量目标，以这种方式来进行系统构建是非常重要的。因为如果问题在测试阶段才暴露出来，会导致返工或者无法抽出时间和精力来提高应用程序质量等情况的发生。

在设计阶段将应当实现的项目设定为质量项目并落实成规格说明书（设计书）是非常重要的。质量项目不仅在评审活动中可以用作评审标准，还可以在测试阶段作为编写测试项目的参考。

故障保护

故障保护（Fail Safe）原本是指设定安全阀门来避免在发生意外故障时产生灾难性后果（对操作者或设备而言）。在计算机系统中，也可以设计并实现故障保护。例如，可以在应用程序中限制从数据库中读取数据的最大记录数，来确保不会出现因应用程序使用过多内存而导致系统停止的状况。

排除 SPOF

排除 SPOF 在防止因硬件故障导致系统宕机方面是非常有效的。服务器和磁盘、网络设备、线缆等许多 SPOF 都可以通过硬件冗余来防止系统宕机。

使用具有实际效果的产品

具有实际效果的软件、硬件以及它们的组合，由于用户数量多，严重的问题都已经被发现并修复了，版本也较为稳定，因此可以有效地确保高可用性。

简单化

系统越复杂，不仅引发故障的因素会越多，而且会更加容易诱发因对系统理解不充分而引起的误操作，导致系统宕机。例如，可以统一各系统间作业管理的基础设施使协作作业简单化，来有效地防止各系统间进行协作作业时可能发生的错误。

运维、设定的自动化

积极地使用、开发运维管理工具可以有效地减少手工作业内容，避免误操作。另外，为了防止配置错误，在配置验证工具与配置文件的自动生成等细小环节上下一些工夫也是非常重要的。

应用 ITIL

ITIL 网罗了与运维管理相关的业务过程并将它们梳理成体系，从整体思考方法到小技巧均有记载。因此，可以考虑将其作为最佳实践应用于系统的运维管理活动中。

快速恢复

实现快速恢复，将故障发生对业务的影响降到最低，是与预防故障同样重要的策略。要实现快速恢复，不仅要优化备份与恢复方式，还要准备并优化联络体制和运维步骤（检测、记录、诊断、修理、恢复、复原）。

此外，还需要事先制定包括灾害等意外情况发生时的“应急预案”³ 在内的 BCP (Business Continuity Plan, 业务连续性计划) ⁴ 和 BCM (Business Continuity Management, 业务连续性管理) ⁵。

³ 指的是为了将灾害造成的损失降到最低而事先制定的、灾害发生后的应对方法和行动步骤的相关计划。

⁴ 指的是发生灾害时能尽可能地降低其对公司业务的影响，利用有限的经营资源，继续开展最低限度的业务活动，并能够在目标恢复时间内让业务再次展开的行动计划。相对于应急预案，BCP 计划更加注重保持业务的连续性。

⁵ 指的是包含了从 BCP 的制定到运维、重新评估的整个管理系统。

运维体制的准备

事先准备好联络体制与应对体制，对于发生问题时将其对业务造成的影响降到最低是非常重要的。不仅要准备监控、维护的体制，还需要

准备包括与企业所有者进行联络在内的其他体制。此外，还需要事先讨论好当问题不能如预想的那样顺利解决时需要汇报的对象⁶。

⁶ 指的是在问题解决过程中依赖其下达指示和进行判断的人，或者是具有很高的技术能力，可以帮助解决问题的人。典型的汇报对象包括管理者与责任者，或者是在专业上造诣更深的技术人员等。

运维步骤的准备

主要通过事先准备、确认以下几个项目的相关步骤，在发生故障时就可以实现迅速对应和防止二次灾害。

- 影响最小化（分离故障机器等）
- 对导致故障的变更进行回退（卸载补丁文件等）
- 为了确定合适的故障对应人员而对故障原因进行初步分析（判断原因是属于应用程序层还是属于基础设施层等）

对于典型的故障，通过讨论其“对于业务的影响”，并事先定义“紧急度”“目标恢复时间”，当实际故障发生时就可以在现场迅速地为故障设定相应的优先级。

业务连续计划 / 业务连续管理（BCP/BCM）

通过准备业务连续计划（BCP）以及业务连续管理（BCM），可以管理危机发生时其对业务产生的影响，并将影响降至最低。然后可以迅速地采取具体的方法和手段来恢复系统。在 BCM 中制定 BCP，并持续地进行关于 BCP 的训练、维持和更新活动。

错误日志

应用程序的错误日志中记录的信息对于解析故障发生时的系统状态有着重要的作用。具体来说，向系统维护人员传达以下信息是很重要

的。

- 可以查明错误发生位置的信息（错误发生时，运行中的程序的位置、负责的业务、调用的方法等）
- 导致问题的数据项目与参数信息，以及可以查找出这些数据的主键等

除了在错误日志中记录能够区分故障原因的必要信息以外，还需要将错误日志设计得通俗易通，使运维人员可以很容易地理解故障的影响范围。

基础设施开发与应用程序开发一样，需要记录可以区分故障原因的有效信息（脚本中错误代码的位置等）。

调整监控项目

为了迅速地查明故障原因并采取应对策略，必须设定充分的监控项目（可用监控、进程监控、SNMP（Simple Network Management Protocol，简单网络管理协议）自陷监控等）和监控对象。

备份、恢复的设计和调优

为了缩短 MTTR（Mean Time To Recovery，平均恢复时间），必须采取合适的设计与调优措施，例如定期获取系统备份、将恢复方式多样化、减小恢复前滚量等。

编写文档与持续改善

为了提高系统构建与运维过程的质量，将过程编写为文档（可视化）是非常重要的。编写的文档类型既包括“操作手册”等描述具体内容的文档，也包括“标准化方针”等记载着基本思想这些抽象内容的文档。

另外，编写文档的优点不仅在于通过它，我们可以习得技能和将技能标准化，而且编写文档自身也可以成为各种过程质量改进活动的基础，间接地提高服务的可靠性。

但是，文档需要经常更新并进行整合，这也会增加成本。因此从系统的整个生命周期出发考虑性价比，制定文档编写的方针是非常重要的。

站在读者的角度编写文档

编写文档的过程中很重要的一点就是要站在运维人员等系统用户的角度来编写文档。因为只有试着去理解读者会被置于什么状况中、他们需要什么样的指导来帮助他们完成任务，才能够编写出最适合他们的文档。在定义术语时要使用清晰明白的语言、不能使用模模糊糊的用语，并要根据读者的水平给难以理解的内容配上示意图来进行说明。另外，站在读者的角度编写文档对于为文档读者提供合适的、必要的信息是非常重要的。

通过编写《文档编写要领》和《术语集》等标准化指南文件，并任命文档管理者来检查文档的质量及各文档之间的一致性，可以有效地消除因文档作者不同而带来的混乱感，提高文档质量。

持续改善的过程

为了能够提高系统价值，使系统可以应对环境变化，故障应对等持续改善的过程是非常必要的。

在 ITIL V3 中，将持续改善的过程定义为以下 7 个步骤⁷。

⁷ ITIL 虽然非常注重 PDCA（Plan：计划。Do：执行。Check：评价。Act：改善）周期，但是在 ITIL V3 中特意将 Check-Act 部分独立出来，更加强调通过 PDCA 周期来进行改善活动的重要性。

① 定义应该测量什么

为了达到改善目的，确定应当将哪些项目提高到什么水平

② 定义能够测量什么

对于选择要进行测量的项目，讨论它们能否被测量

③ 收集数据

确定进行测量所必需的数据的收集方法，并开始实际收集

④ 处理数据

处理收集到的数据，并整理出需要的信息。

⑤ 分析数据

分析这些信息，测量目标达成情况，并在没有达成目标时分析其原因

⑥ 展示并使用信息

向相关人员提出分析结果报告

⑦ 实施改善措施

根据报告进行判断，在需要采取改善措施的时候，进行相应的改善

改善成功的重要因素包括了“管理者的参与”“分工合作以提高改善活动的效率”以及“任命推进改善活动的责任人”等，也就是说进行改善活动的主体应该是一个组织，而非个人。

“管理者的参与”指的是由董事长等高级管理者设定改善活动的目标并确保这个目标的实现。目标明确并有高级管理者作为后盾，推进责任人与改善活动参与者就会更加积极地进行改善活动。

“分工合作以提高改善活动的效率”是指通过指定专人专项，使其全身心投入改善活动中，从而加快改善周期，提高改善效率。

“任命推进改善活动的责任人”是指通过明确改善活动的责任人，加快做出决定的过程，确保各项工作的一致性。

完整、彻底的确认

要想切实提高服务的可用性，不能带着“一定没问题的”这样乐观的设想来开展工作，而是必须对所有的开发过程都逐一进行彻底的确认。

要想实现这个目标，除了采用检查项目清单（Check List）等工具外，还可以通过创造能保持、激发现场经理、工作人员能动性的工作环境，促使他们积极主动地开展完整、彻底的确认活动。

而要想保持与激发能动性，可通过“明确现场可自主酌情处理的范围”“增加报告确认进度与讨论内容的机会”等，加深当事者对项目的理解，增强他们对职位、工作任务的责任感。

此外，进行工作报告和评审活动时，区分开现实情况与前提条件会更容易察觉到风险。

至此，我们讲解了可用性策略的基本思考方法。接下来，我们将分类讲解可用性策略的设计方式。

2.2 Web/AP 服务器的高可用性设计方式

现在，全世界到处都有通过互联网提供的各种各样的服务，例如电子商务、网络银行服务、网络证券交易、在线电影，等等。这些服务或业务的用户接口和流程的控制、业务逻辑的处理都是在称为 Web 服务器以及 AP 服务器的基础设施中进行的。

在 Web/AP 服务器的可用性策略中，我们将讨论如何设计服务器结构，才能在担当如此重任的 Web/AP 服务器在发生硬件故障时，将故障对系统服务的影响降到最低。通过服务器的 CPU、内存等部件和磁盘阵列柜的冗余，可以提高系统的可用性。此外，通过在 Web/AP 服务器之前引入负载均衡器，可以自动检测到发生故障的服务器，并将其从系统中分离，将故障对系统服务的影响降到最低。

本书基本上是按照质量优先到成本优先的顺序来介绍设计模式的。

容错服务器 / 大型机模式

通过使用 CPU、内存、硬盘等部件全部冗余的容错（FT, Fault-Tolerant）服务器⁸ 和大型机⁹，可实现在发生硬件故障时，硬件的更换和修理工作对系统服务和性能不会造成任何影响（图 2.3）。

⁸ 是指硬件的部件全部实现冗余，在故障发生时也可保持系统运行的、具有极高可用性的服务器。

⁹ 在银行的结账系统等企业主干业务中使用的大型机。与容错服务器一样，是可以实现硬件冗余等故障应对策略的机器。

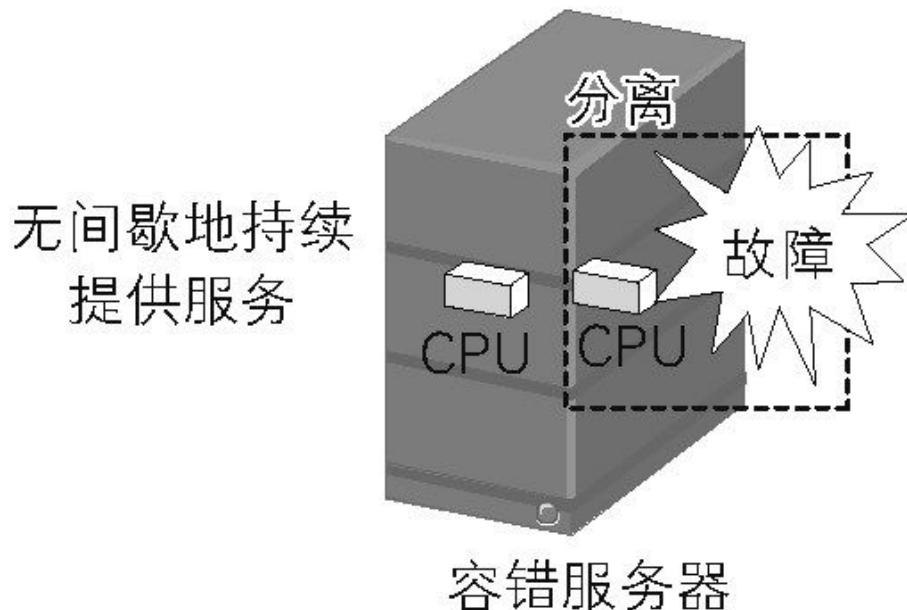


图 2.3 容错服务器 / 大型机模式

该模式适合因社会影响力很大等原因而不能停止服务，且必须一直保持服务稳定性的系统。

会话共享的负载均衡模式

该模式使用负载均衡器来分散多台 Web/AP 服务器之间的处理负荷。将服务器集群，当有用户访问时将服务器上的信息（会话）共享给其他服务器。这样，当故障发生时，就可以将正在故障机上进行的处理转移到同群集内的其他服务器上继续进行，进而保证系统继续提供服务。由于会话被转移，事务¹⁰也得以继续。用户无需再次输入必要的信息（图 2.4）。

¹⁰ 指的是进行某项业务和处理时所必需的一系列处理。以电商网站为例，“用户登录→选择商品→选择送货地址→……→结算完成”这样一系列处理就是事务。事务不能继续指的是这一系列处理不能在中途断开后继续进行，而是必须从第一个处理重新再来。

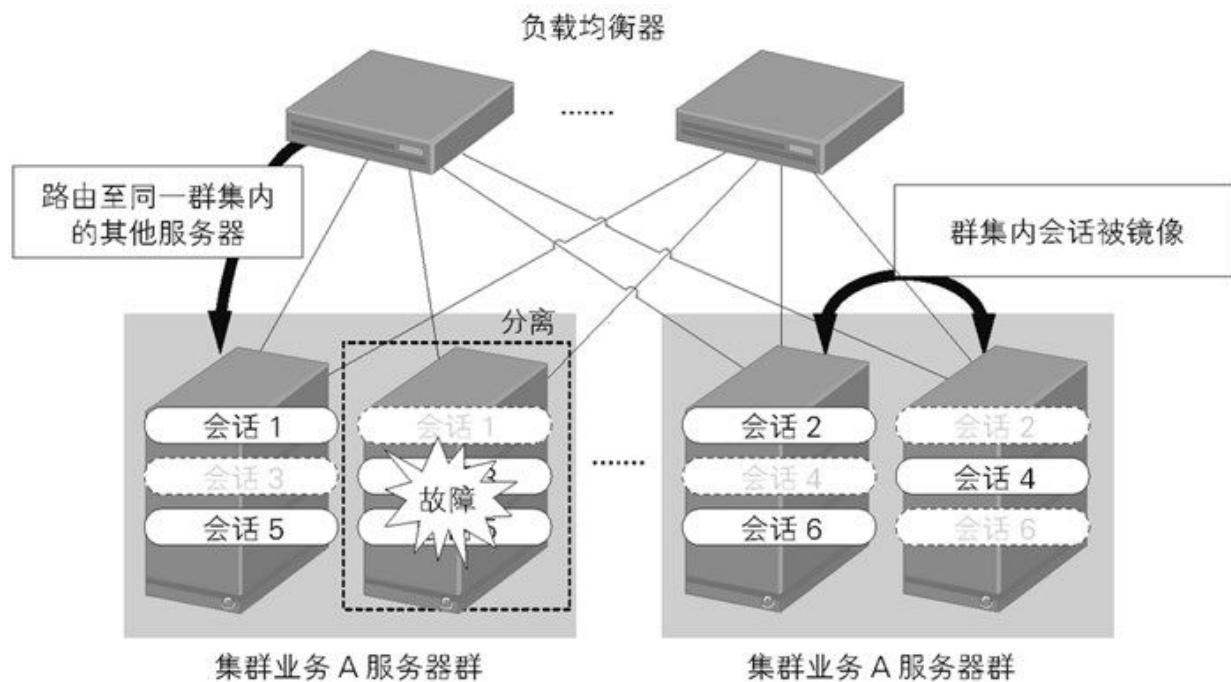


图 2.4 会话共享的负载均衡模式

通过负载均衡器的设定，还可以将发生故障的服务器迅速从系统中分离出来。但是，整个系统的性能也会因为缺少一台服务器而有所下降。另外，分离出发生故障的服务器后，如果系统只剩一台服务器提供服务，那么系统的可用性将无法维持。

会话非共享负载均衡模式

该模式以具有相同功能的多台服务器来运行系统。故障发生时会分离发生故障的服务器。但是由于没有共享会话，故障发生时正在进行的处理会消失，事务不能继续（图 2.5）。

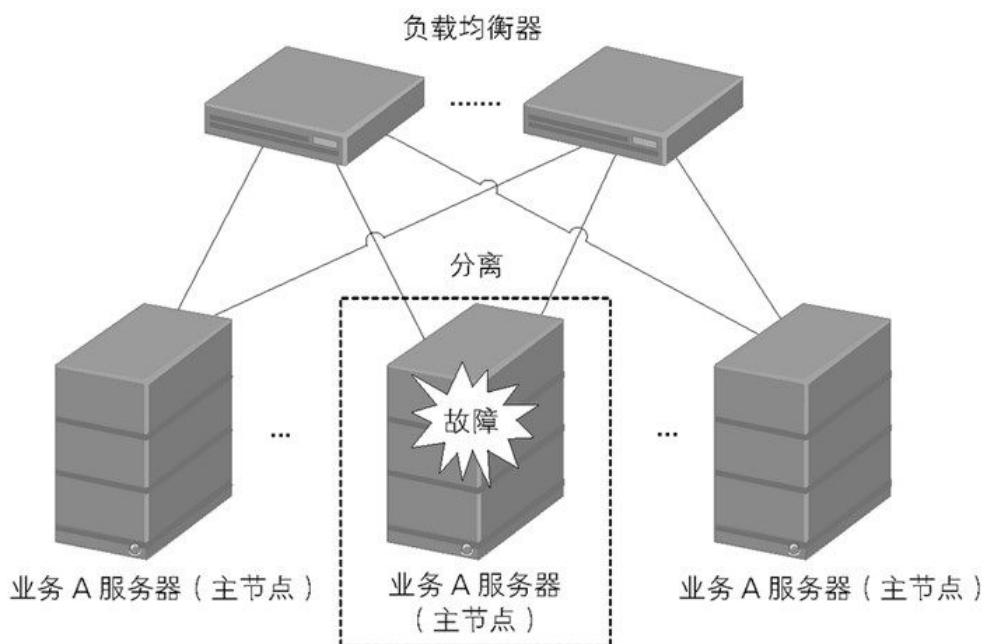


图 2.5 会话非共享负载均衡模式

故障发生时其对性能和可用性的影响与会话共享负载均衡模式相同。

备用机模式

该模式为使用中的服务器准备功能相同的备用机。发生故障时可以通过手动切换到备用机来恢复服务（图 2.6）。

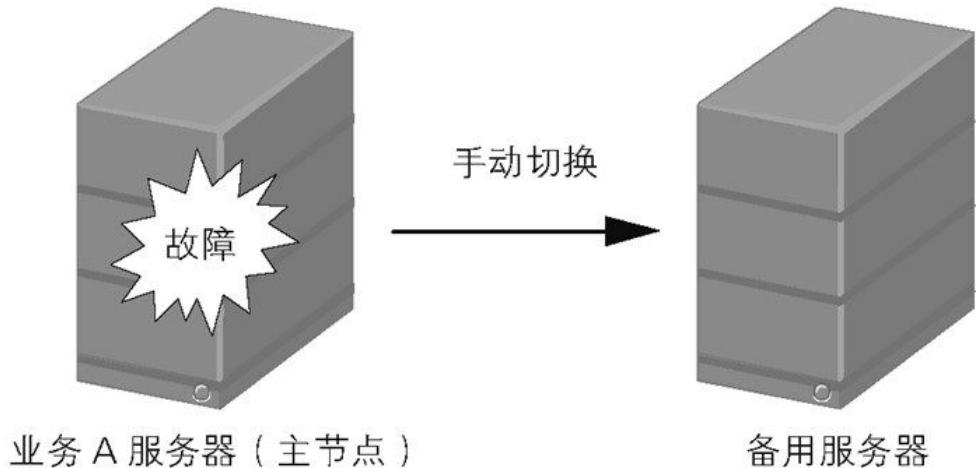


图 2.6 备用机模式

由于需要手动作业，所以除了手动作业的时间外，还需要加上赶往故障现场的时间，因此整个恢复过程大约需要 1 天左右的时间。而且在复原工作完成之前，系统一直处于停止状态。

各模式的比较结果与选择标准

我们将各模式的比较结果与选择标准分别总结在表 2.1 和表 2.2 中。

表 2.1 Web/AP 服务器的可用性设计方式中各模式的比较结果

比较项目	容错服务器/大型机	会话共享的负载均衡	会话非共享的负载均衡	备用机
服务的切换时间 (检测时间除外)	◎： 不停止 通过所有硬件部件的冗余，可在不中断服务的情况下切换服务	○： 数分钟以内 通过将发生故障的服务器上的会话转移到集群内的其他服务器上，可以继续提供服务。事务也可继续	○： 数分钟以内 通过负载均衡器将发生故障的服务器分离，可继续提供服务。但是由于没有共享会话，事务不能继续	×： 1天左右 手动将发生故障的服务器切换到具有相同功能的备用服务器上以恢复服务
有无 SPOF	◎： 无			×： 有 服务器发生故障时服务停止

是否会话共享	-: 不需要 由于硬件全部冗余,发生故障时会话也可以继续	○: 是 由于会话共享,发生故障时会话也可以继续	✗: 否 由于会话没有共享,发生故障时会话消失
可否使用备用资源(正常运行时)	✗: 不能使用 发生故障时才会运行备用机器	○: 可以使用	✗: 不能使用 发生故障时才会运行备用机器
可扩展性	○～✗: 取决于产品 可以Scale Up(纵向扩展)、Scale Out(横向扩展)以及不可扩展的产品都有,需要根据需求选择	○: 高 Scale Up(纵向扩展)与Scale Out(横向扩展)都可以	△: 普通 仅可以Scale Up(纵向扩展)

◎○△✗是设计方式的比较,并非绝对标准(以后的表格中也是如此)

不能仅仅考虑可用性需求,还要结合性能需求来决定服务器结构。例如当一台服务器即使发生故障也要确保一定性能的情况下,实际需要运行的服务器数量要比事先根据性能需求而决定的服务器数量多一台

表 2.2 Web/AP 服务器的可用性设计方式中各模式的选择标准

模式	故障发生时对系统服务有无影响	允许在多长时间内恢复业务	在业务完全恢复前性能下降的程度*1	成本
容错服务器/大型机	无 适用于不允许故障对服务造成影响的情况	无 适用于不允许故障导致事务中断的情况	无 适用于追求服务质量长期稳定的情况	高 容错服务器、大型机等将部件全部冗余的设备成本很高

会话 共享的负载均衡	无* ² 适用于不允许故障对服务造成影响的情况	无* ² 适用于不允许故障导致事务中断的情况	中 适用于允许故障导致一定程度的性能下降的情况	中 需要引入负载均衡器和进行集群设定作业
会话 非共享的 负载均衡	有 适用于允许故障对服务产生一定影响的情况，如用户在故障时输入的信息消失	短 适用于需要在故障发生后，数分钟内恢复事务的情况		
备用机		长 适用于允许在故障发生后1天左右的时间内恢复事务的情况	大 适用于允许在业务完全恢复前暂时不能提供服务的情况	低 虽然购买备用机会发生费用，但是与容错服务器和引入负载均衡器相比成本更低

*1 这里记载的选择标准的前提是在通常情况下可以满足性能需求，但是在发生故障时允许系统进入降级模式。如果当一台服务器发生故障时仍然要满足性能需求，则不限于以上的选择标准

*2 但是在服务的切换过程中用户可能会感觉系统响应变慢

注意点

DB 服务器的模式的组合

在 Web 系统中除了 Web/AP 服务器以外，一般还配置有 DB 服务器。这里需要注意的是，我们要根据系统的用途和需求的特性来考虑如何组合 Web/AP 服务器和 DB 服务器。图 2.7 中给出了一个 Web/AP 服务器可用性设计方式和 DB 服务器可用性设计方式的组合示例，以供大家参考。

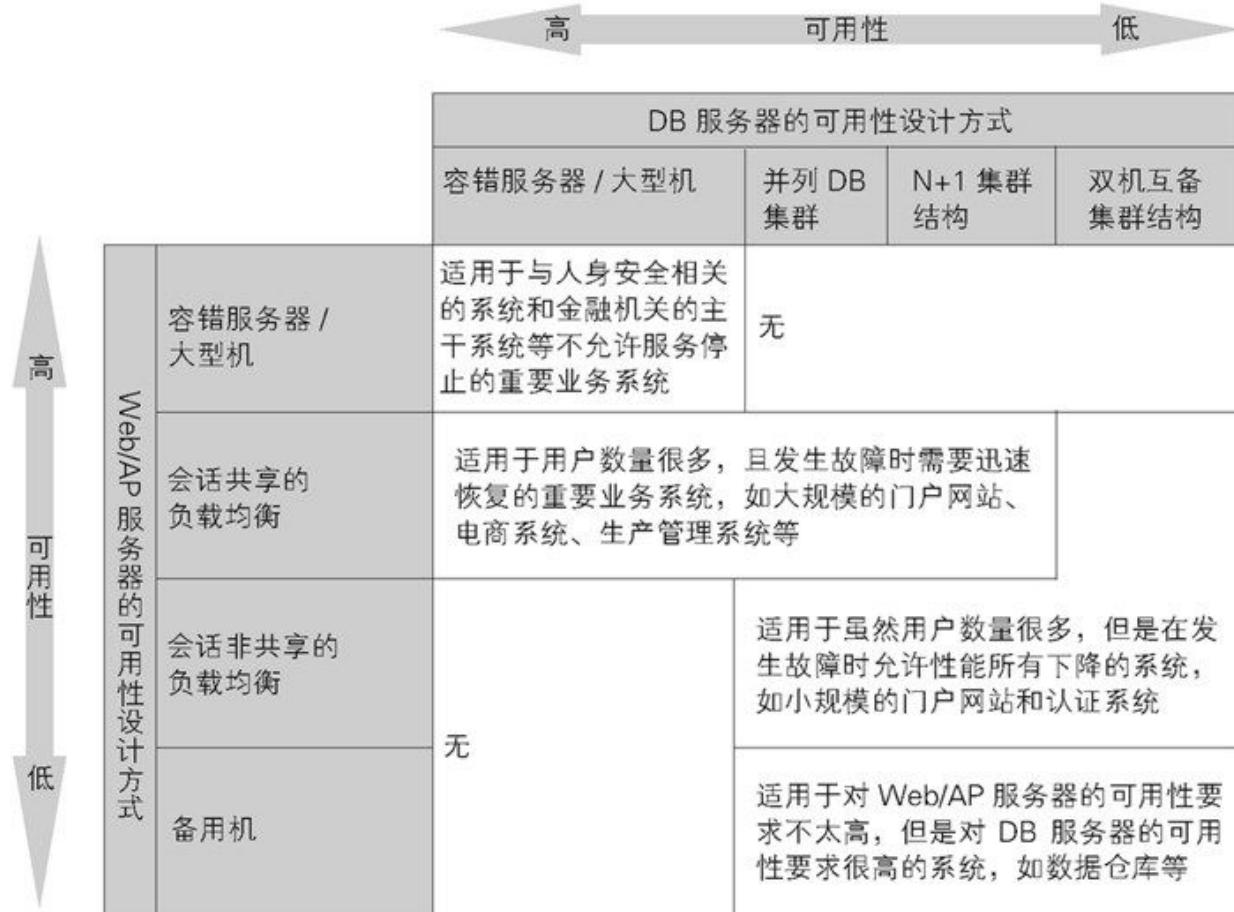


图 2.7 与 DB 服务器的可用性设计方式进行组合

“无”表示只有 Web/AP 服务器或者 DB 服务器一方使用了容错服务器 / 大型机模式，因此不会提高全体系统的可用性

负载均衡的实现方式

我们在会话共享的负载均衡模式、会话非共享的负载均衡模式中介绍了使用负载均衡器来分散负荷的方式。除此之外，仅在在服务器上使用软件也可以实现负载均衡。

通过维护服务确保可用性

如果可用性需求不太高，如允许 1 天以上的服务停止时间等，那么通过维护服务就足以确保系统的可用性，而无需采用本节介绍的可用性策略。

操作系统与中间件的选择

不仅是本节中介绍的服务器的可用性，运行在服务器上的操作系统和中间件的可用性对系统整体的可用性的影响也很大。例如在选择操作系统和中间件的时候，就要特别留意有无内存泄露的缺陷，有的话就会导致服务需要定期重启，不能实现长期的持续运行。

2.3 DB 服务器的可用性设计方式

DB 服务器保存着企业提供系统服务和业务所必需的数据，并要在需要的时候提供必要的信息，是非常重要的基础设施。DB 服务器的故障或停止与服务的停止有着直接的联系，会导致企业错失商业机会和企业价值下降。

DB 服务器可用性策略的目的是使 DB 服务器能稳定运行，为企业活动打下牢固基础。DB 服务器的可用性策略是通过对系统配件，如服务器的 CPU 等部件、磁盘阵列柜以及服务器所连接的共享磁盘等进行冗余来实现的。需要注意的是，集群的构成方式不同，发生故障时，服务的切换时间与降级运行时的性能都会所有不同。

容错服务器 / 大型机模式

参见 2.2 节中介绍的容错服务器 / 大型机模式。

并列 DB 集群模式

该模式使用多台具有相同功能的 DB 服务器共同工作，通过共享磁盘来进行数据共享（图 2.8）。

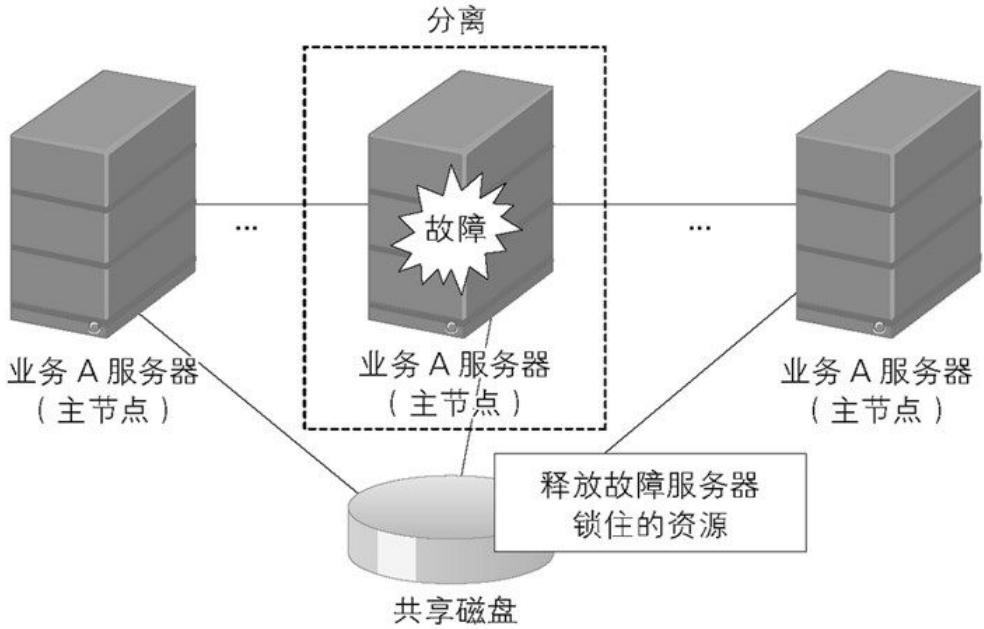


图 2.8 并列 DB 集群模式

多数情况下，该模式可在 1 分钟左右分离出发生故障的服务器，但是在故障发生时，系统也会因分离出的那台服务器而损失掉相应的性能。

N+1 集群结构模式

除了将 N 台正在运行的 DB 服务器作为主节点外，还准备了 1 台 DB 服务器作为备用节点。当故障发生时，可以在几分钟至几十分钟的时间内从故障服务器切换到备用服务器上（图 2.9）。切换后，可以维持与正常工作时一样的服务质量。

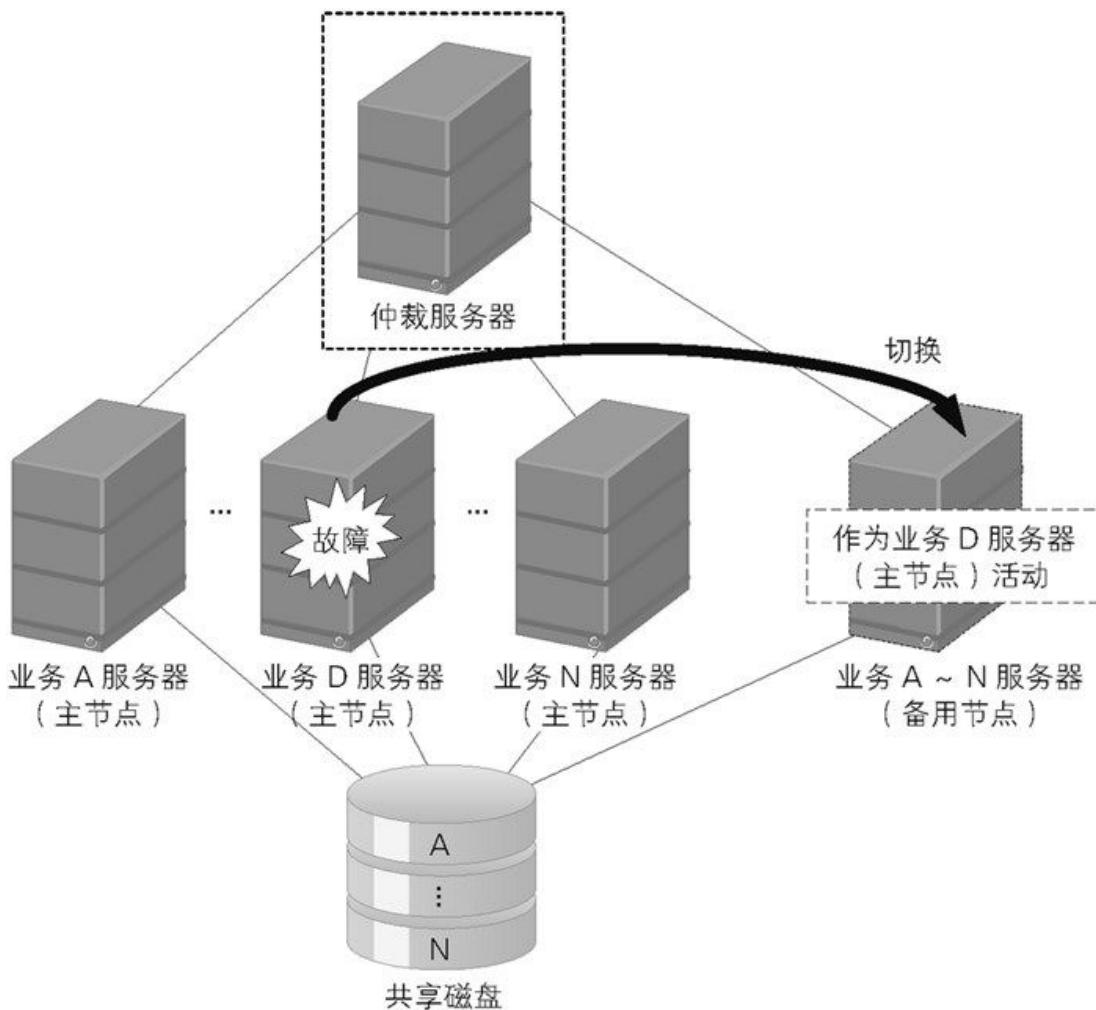


图 2.9 N+1 集群结构模式

双机互备集群结构模式

由两台服务器组成集群结构。集群的构成方式是在业务 A 中，服务器 1 是主服务器，服务器 2 是备用服务器；但在业务 B 中服务器1是备用服务器，服务器 2 是主服务器（图 2.10）。当 1 台服务器发生故障时，可以在数分钟至数十分钟的时间内切换到备用机上，继续提供服务。但是原本两台同时工作的服务器中进行的处理现在只能在一台服务器上进行，所以在故障发生时，系统的性能几乎下降了一半。

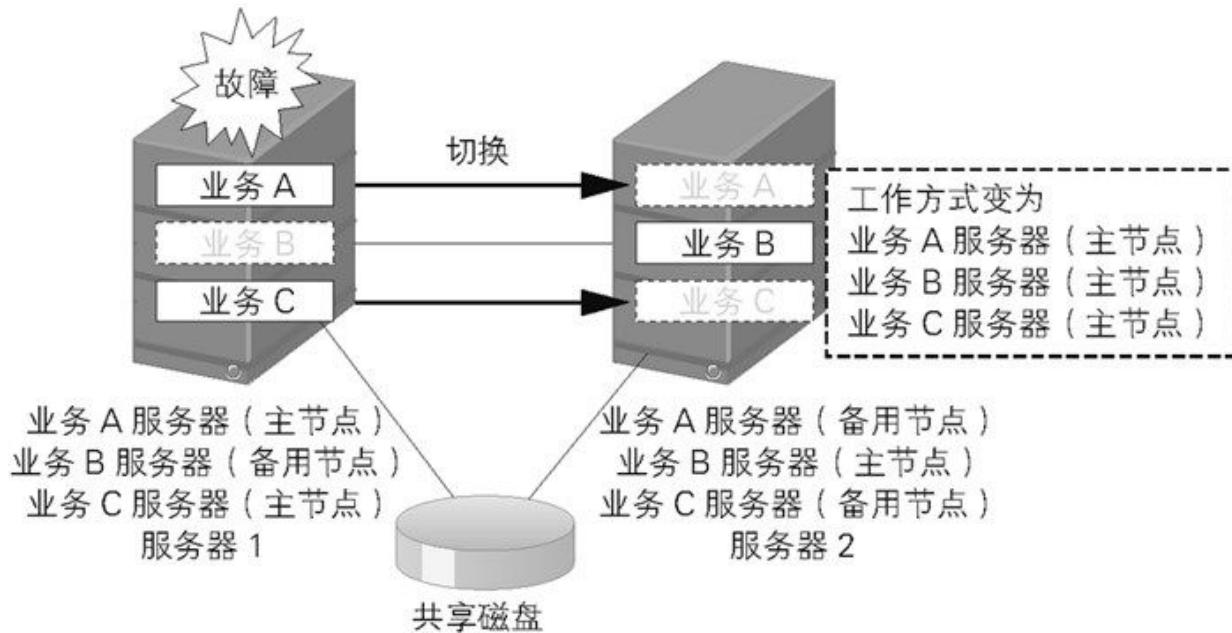


图 2.10 双机互备集群结构模式

各模式的比较结果与选择标准

我们将各模式的比较结果与选择标准分别总结在表 2.3 和表 2.4 中。

表 2.3 DB 服务器的可用性设计方式中各模式的比较结果

比较项目	容错服务器/大型机	并列DB集群	N+1集群结构	双机互备集群结构
服务的切换时间 (检测时间除外) ^{*1}	◎： 不停止 通过所有硬件部件的冗余，可在不中断服务的情况下切换服务	○： 1分钟以内 通过释放发生故障的服务器的资源并分离该服务器，继续提供服务	△： 数分钟以至数十分钟 分离发生故障的服务器并切换至备用服务器运行，继续提供服务	△： 数分钟以至数十分钟 分离发生故障的服务器并将该服务器上提供的服务转移到其他服务器上，继续提供服务
有无SPOF	◎： 无 通过所有硬件部件的冗余来避免因单	△： 共享磁盘 通过共享磁盘的冗余以及连接共享磁盘与服务器的光纤信道 ^{*2} 接口的冗余，可以消除SPOF		

	点硬件故障引起的 服务中断			
可否 使用 备用 资源 (正 常运 行 时)	x: 不能使用 发生故障时才会运行备用机器	o: 可以使 用	x: 不能使用 发生故障时才会运行 备用机器	o: 可以使用
可扩 展性	o~x: 取决于产品 可以Scale Up (纵 向扩展)、Scale Out(横向扩展)以及 不可扩展的产品都 有，需要根据需求 正确选择	o: 高 Scale Up (纵向扩 展) 与 Scale Out(横向扩 展)都可以	△: 普通 可以Scale Up (纵 向扩展)。如果要 Scale Out(横向扩 展)，在修改DB物理 层面的设计时还需要 修改结构	△: 普通 仅可以Scale Up (纵向扩展)

不能仅仅考虑可用性需求，还要结合性能需求来决定服务器结构。例如一台服务器即使发生故障也要确保一定性能的情况下，实际需要运行的服务器数量要比事先根据性能需求决定的服务器数量多一台

※1 这里记载的切换时间为推测值，实际的切换时间取决于具体的设计

※2 使用光纤和同轴线来实现高速数据通信的一种数据通信方式

表 2.4 DB 服务器的可用性设计方式中各模式的选择标准

模式	允许在多长时间内恢复业务	在业务完全恢复前性能下降的程度
容错服务 器/大型机	无 适用于不允许故障导致服务器中断的情 况	无 适用于追求服务质量长期稳 定的情况

模式	允许在多长时间内恢复业务	在业务完全恢复前性能下降的程度
并列DB集群	短 适用于需要在故障发生后，数分钟内恢复服务的情况	小 适用于允许故障导致一定程度性能下降的情况
N+1集群结构	中 适用于需要在故障发生后，数分钟至数十分钟内恢复服务的情况	无 适用于追求服务质量长期稳定的情况
双机互备集群结构	中 适用于需要在故障发生后，数分钟至数十分钟内恢复服务的情况	中 适用于允许故障导致性能减半的情况

注意点

业务恢复的时间

在 N+1 集群结构模式、双机互备集群结构模式中，DB 服务器发生故障时所需要的业务恢复时间不仅包括将服务切换到备用节点上的时间，还需要考虑 DB 连接池¹¹的恢复时间。

¹¹ Web/AP 服务器与 DB 服务器进行通信时，并不是每次通信都与 DB 服务器建立连接，而是使用在初次访问 DB 服务器时就已经建立的连接，这样可以减轻服务器的处理负荷。储存初次访问 DB 服务器时建立的连接的地方就称为连接池。

业务恢复时间 = 服务切换时间 + 数据库连接池的恢复时间

如果对业务恢复时间要求很高，就还需要注意在服务切换过程中使 DBMS 日志重新生效的相关参数的设计等细节。

考虑服务器内部部件级别的冗余

此外，还需要考虑包括系统磁盘、NIC（Network Interface Card，网络适配器）¹²等在内的服务器内部部件级别的冗余。

¹² 将个人计算机和服务器连接至网络所必需的扩展卡。当网络适配器发生故障时，服务器与个人电脑将不能连接网络。

考虑故障的检测方法

如果考虑了可能发生的故障却不考虑故障的检测方法，那么可能会导致无法检出故障，甚至是检测错误的情况发生。例如在集群结构中，是通过检测软件的响应来判断是否有故障的。但是当服务器超负荷运行时，可能会导致软件的响应暂时中断。如果将这种现象认为是故障那就是检测错误。但是如果检测响应中断，又很容易漏掉真正的故障。因此，我们需要充分考虑“什么才是真正的故障”，并制定相应的检测方法。

明确各供应商之间的责任分界点和责任划分方法

如果硬件、中间件、软件是由不同供应商提供的，那么必须要事先明确各供应商之间的责任分界点和责任划分方法。

2.4 虚拟服务器冗余的设计方式

通过虚拟化技术，可以将处理器、硬盘等服务器的组成设备虚拟化，在1台物理机器上搭建出多台虚拟服务器。由于这种技术可以减少购买硬件的费用、数据中心的费用以及电费等IT相关支出，因此被许多系统中所采用。

但是一旦物理机器发生故障，就会导致搭建的虚拟服务器停止运行。特别是当在1台机器上搭建了多台虚拟服务器的时候，影响可能会非常大。因此在引入虚拟服务器的时候，必须根据可用性需求来讨论冗余方式。

虚拟服务器的冗余可以根据冗余方式的组合分为以下这些模式。后文中会将运行虚拟服务器的物理机器简称为“主机”。

集群软件模式

当硬件发生故障或是虚拟服务器上的操作系统和应用程序发生故障时，使用集群软件将服务切换到其他主机上的备用虚拟服务器上（失效转移¹³），使业务继续进行（图 2.11）。

¹³ 在多台服务器之间共享数据，发生故障时自动将服务转移到无故障的服务器上继续进行行业务处理的功能。由于是自动处理，所以用户几乎感觉不到发生了故障。

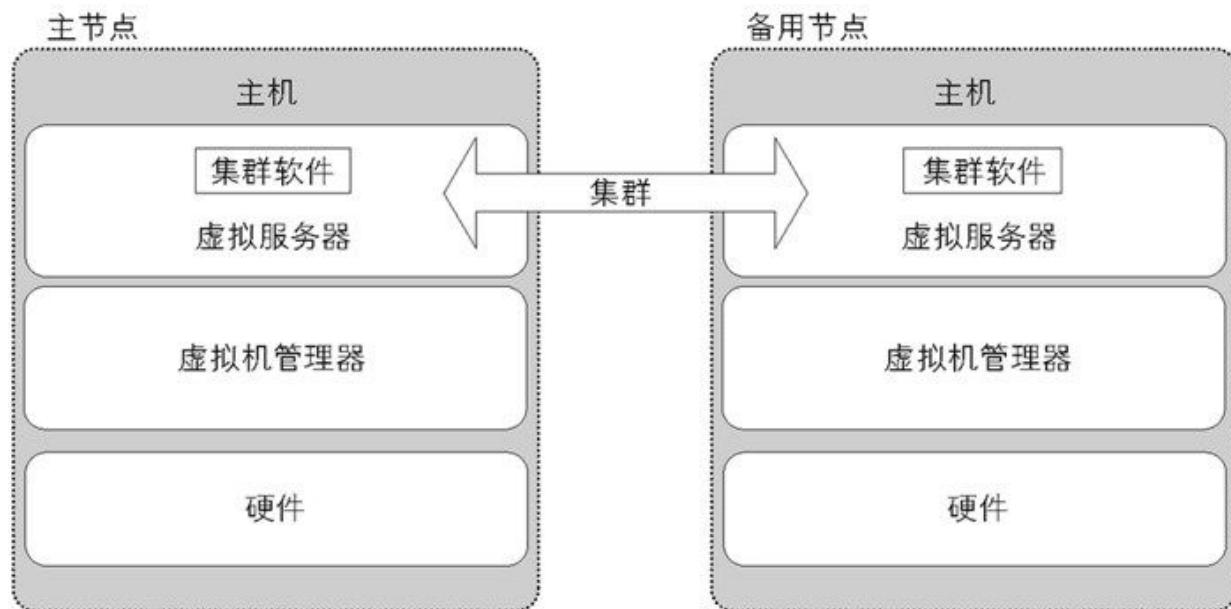


图 2.11 集群软件模式

不过，有些集群软件只能用于物理服务器，不支持虚拟服务器，因此在选择集群软件时需要特别注意。此外，还必须阅读供应商提供的集群软件说明书，确认其中的制约事项，并根据规格说明来构筑基础设施。

虚拟机管理器 HA 功能模式

利用主机上的虚拟机管理器（Hypervisor）¹⁴的功能，可以将因硬件故障停止运行的虚拟服务器自动切换到其他主机上（失效转移），使业务继续进行（图 2.12）。我们将该功能称为虚拟机管理器 HA（High Availability，高可用性）功能¹⁵。虚拟机管理器 HA 功能是通过多台主机上的虚拟机管理器之间的调度来实现的。

¹⁴ 介于主机和虚拟服务器之间，具有控制多台虚拟服务器功能的软件。

¹⁵ 为了实现虚拟服务器的高可用性而在虚拟机管理器上实现的功能。

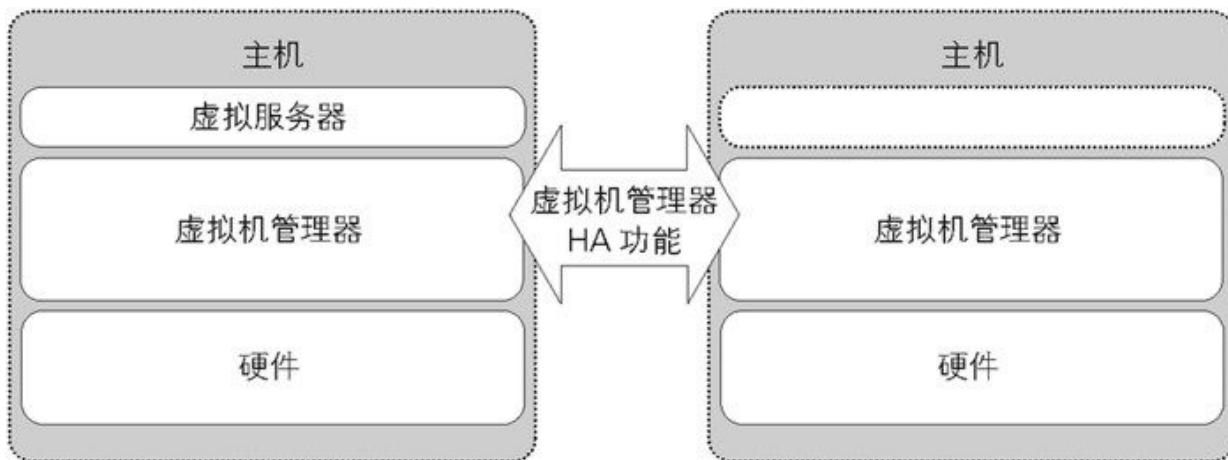


图 2.12 虚拟机管理器 HA 功能模式

正是由于虚拟机管理器 HA 功能在发生故障时可以自动地切换主机，所以无需事先启动备用虚拟服务器并使其保持在待机状态，可以将这部分资源分配给使用中的虚拟服务器。但是如果在这种情况下发生了故障，会使备用虚拟服务器的可用资源减少，根据资源的情况也可能会造成备用虚拟服务器性能的下降。因此，需要根据故障时的性能需求来平衡分配资源。此外，如果虚拟服务器上运行的虚拟服务很多，自动切换可能会需要很长时间，运维上需要格外注意。

实时迁移模式

为了确保系统的高可用性，在发生故障时除了要进行业务恢复处理以外，有时还需要实现能在不停止服务的情况下进行定期维护与零部件更换工作。

该模式利用主机上虚拟机管理器的实时迁移功能（Live Migration），来实现在继续提供服务的同时移动主机间的虚拟服务器（图 2.13）。

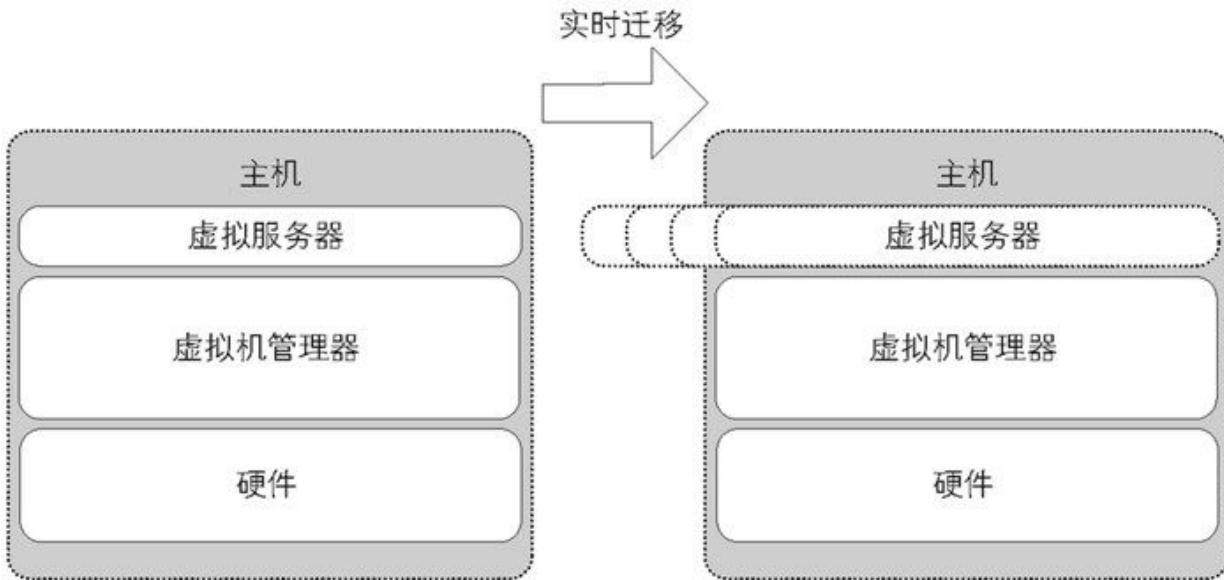


图 2.13 实时迁移模式

该模式主要用于主机的维护工作和提高正常状态下虚拟服务器的资源利用率。由于网络连接也同时被转移过来，所以用户完全感觉不到到虚拟服务器已经被转移到新的主机上。

例如，想要增加内存容量或进行预防性维修而更换零部件等情况时，需要有计划地停止硬件的运行。这时，就可以先通过实时迁将虚拟服务器暂时转移到其他主机上，再来停止主机的运行并进行维护，结束后再将虚拟服务器还原到这台主机上。这样就实现了不停机运维。

由于实时迁移并非总是同步，所以不具备恢复到故障前的业务点的功能。

各模式的比较结果和选择标准

我们将各模式的比较结果与选择标准分别总结在表 2.5 和表 2.6 中。

表 2.5 虚拟服务器冗余设计方式中各模式的比较结果

比较项目	集群软件	虚拟机管理器HA功能	实时迁移
可应对的故障	○：范围广 业务应用程序、虚拟服务器、硬件的故障都可以应对	△：有局限 只能应对硬件故障	—：用于维护 维护作业中可在不停止虚拟服务器的情况下更换硬件
操作系统、虚拟服务器切换时间 ^{*1}	○：短 几分钟即可完成切换	△：长 ^{*2} 几分钟～十几分钟可完成切换	—
业务切换时间 ^{*1}	—：取决于具体业务 可在几分钟～几十分钟左右（取决于具体业务）完成切换		
系统运行中所消耗的资源	×：多（使用中和备用的所有服务器都会消耗） 除了使用中的虚拟服务器外，还需要为备用虚拟服务器分配必要的资源（CPU、内存）	○：普通 (仅使用中的虚拟服务器会消耗) ^{*3} 仅需要为使用中的虚拟服务器分配资源 (CPU、内存)	—：无 仅在迁移时需要临时分配资源
许可	×：高	△：高～	○：低

证费用	需购买集群软件的许可证	低 部分软件可能需要购买其他许可证 (需要确认)	多数虚拟机管理器的许可证都包含了实时迁移功能
构建、运维费用	<ul style="list-style-type: none"> ○：必需（与虚拟服务器数量有关） <p>使用集群软件构建（设计～测试）集群以及集群的运维都需要费用。特别是由于主机和虚拟服务器有多种组合，所以需要充分地进行测试</p>	<ul style="list-style-type: none"> ○：必需（与主机数量有关） <p>构建虚拟机管理器HA功能（设计～测试）以及运维都需要费用</p>	<p>△：需要事前讨论并编写操作文档</p> <p>由于实时迁移中作为转移源和转移目的地的虚拟服务器和主机都有限制条件，所以需要事前讨论并进行设计和验证，这些都需要花费费用。此外，如果是手动操作进行实时迁移，还需要为运维人员编写操作手册。</p>

- ※1 表格中的切换时间为推测值。由于环境不同，切换时间可能会超过表格中的时间
- ※2 与集群软件类似，部分虚拟机管理器产品在操作系统或应用程序发生故障时也可以进行切换。同样，从切换虚拟服务器后到应用程序开始运行的时间也可能超过表格中的时间
- ※3 虽然仅需要为使用中的虚拟服务器分配资源，但是从性能角度考虑，还是有必要为备用节点分配资源的

表 2.6 虚拟服务器冗余设计方式的选择标准

模式	使用目的	故障时对业务的影响
集群软件	●故障应对策略 ●提高可用性	几乎不允许有影响 适用于需要系统持续稳定运行的情况
虚拟机管理软件HA功能		允许有一定程度的影响 适用于故障时允许业务暂时停止的情况
		—

实时迁移	定期维护 特别适用于需要在不停止服务的情况下进行运维工作的情况	
------	------------------------------------	--

注意点

关于资源分配的考虑

在集群软件模式、虚拟机管理器 HA 功能模式和实时迁移模式中转移虚拟服务器的时候，要注意转移目的地的主机是否有足够的空闲资源。如果空闲资源不足，可能会造成转移失败或者转移后虚拟服务器性能低下的状况。因此，必须在转移前就考虑到至少需要多少资源才能确保转移工作顺利完成。

例如要应对单点故障时，虚拟机管理软件 HA 功能模式如果要运行 N 台虚拟服务器，那么至少需要准备可以让 $N+1$ 台虚拟服务器运行的资源，而在集群软件模式中，由于必须保持备用虚拟服务器一直处于运行状态，因此至少需要准备可以让 $2N$ 台虚拟服务器运行的资源。

存储设备与网络的冗余策略

除了虚拟服务器冗余策略外，还需要考虑用于存储虚拟服务器数据的存储设备以及用于连接虚拟服务器与外部客户端、外部系统的网络的冗余策略。

2.5 LAN 的可用性设计方式

计算机系统由多台服务器与网络设备构成，它们之间通过 LAN (Local Area Network, 局域网) 连接。如果 LAN 的可用性很低，那么一部分设备发生故障就会影响到整个系统，甚至可能会导致系统服务的中断。

LAN 的可用性可以通过网络中的路由器¹⁶冗余和交换机¹⁷冗余来提高。

¹⁶ 主要用于连接各网络的设备。可用于连接 LAN 和 WAN。

¹⁷ 主要用于将建筑物中或建筑区域内的多台设备连接到同一个网络中。L2（二层）交换机可用于 LAN 内部的通信，L3（三层）交换机可用于 LAN 之间的通信。

高可靠性核心交换机模式

该模式在采用两台以上高可靠性核心交换机进行冗余配置的同时，还引入了可实现一至多台虚拟交换机和路由器的设备。因此，可以搭建出物理配线少，又具有冗余路由路径和设备的网络环境（图 2.14）。

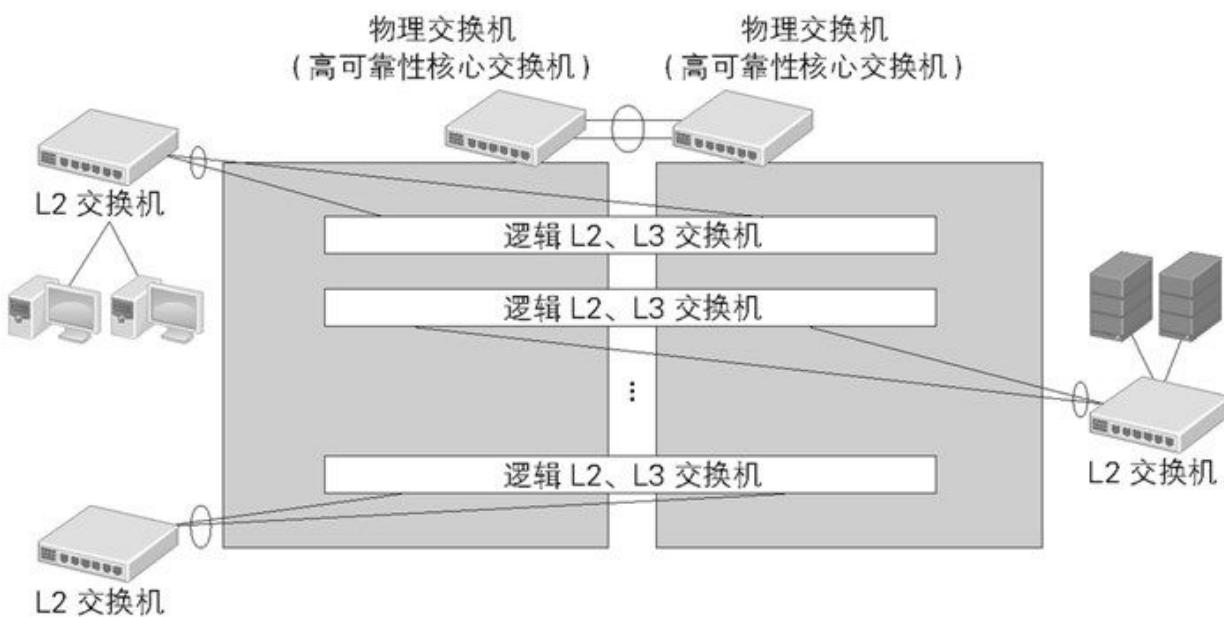


图 2.14 高可靠性核心交换机模式

由于 L2、L3 交换机在逻辑上被集中在一起，因此在发生故障时无需交换各种设备与路由信息，即可瞬间恢复通信。但是因为核心交换机功能非常强大，所以价格也非常昂贵。

动态路由模式

该模式通过交换机物理冗余和路由器物理冗余来搭建网络，并在网络通信中使用动态路由协议（图 2.15）。

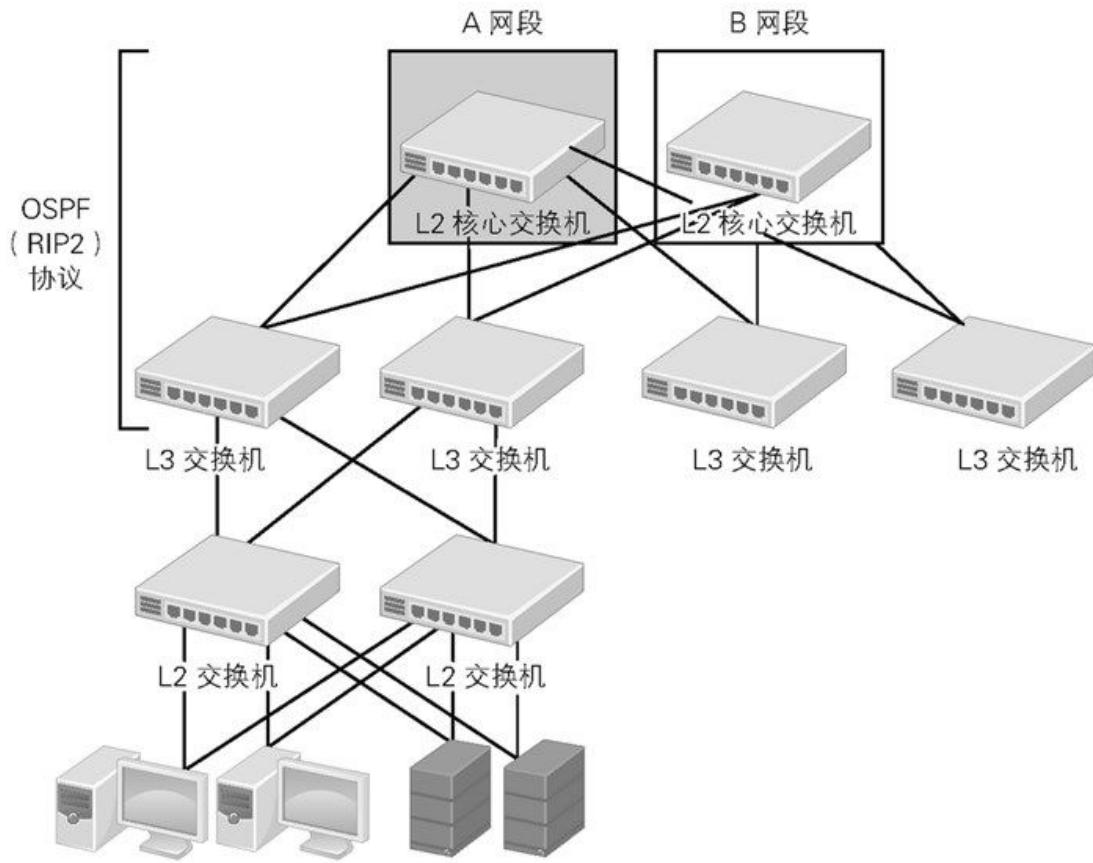


图 2.15 动态路由模式

故障发生时，该模式会自动在经过物理冗余的其他交换机中检索可以使用的路径来恢复通信。根据所选择的路由协议，可以在 1 秒左右自动检测故障并恢复业务。

其缺点是物理交换机冗余和物理路由器冗余易导致成本增加。

VLAN 模式

该模式使用具有 VLAN (Virtual Local Area Network, 虚拟局域网) 功能¹⁸ 的交换机（图 2.16），将多个端口汇聚成一个逻辑分组来实现冗余。通过使用 VLAN 功能将交换机的多个端口在逻辑上连接在一起，

当其中一个端口发生故障时，就可以使用同一 VLAN 分组中的其他网络路径继续通信。

¹⁸ 将一个物理路由器分割为多个逻辑路由器的功能。

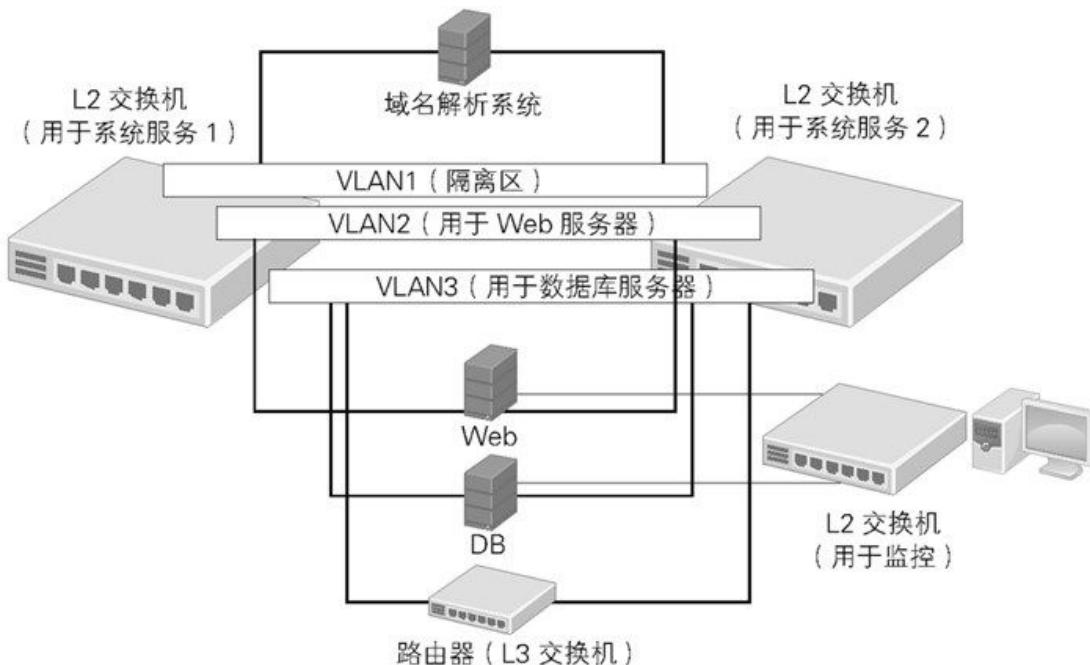


图 2.16 VLAN 模式

例如，有一台服务器是通过交换机 A 的端口 1 和交换机 B 的端口 4 连接在一起，当交换机 A 发生故障时，可在数秒内改由交换机 B 进行通信，使业务可以继续进行。

不过，该模式可能会产生循环回路，因此需要使用实现了 STP (Spanning Tree Protocol, 生成树协议) 等可避免通信无限循环的协议的交换机。

与动态路由模式相比较，VLAN 模式可以用较少的交换机搭建出高可用性的网络。

双机热备模式

准备一台备用交换机并保持机器处于待机状态，这样当发生故障时就可以自动地将通信切换到备用交换机上。虽然切换时间可能受到交换机性能的影响，但一般都可以在一分钟左右完成自动切换，恢复业务（图 2.17）。

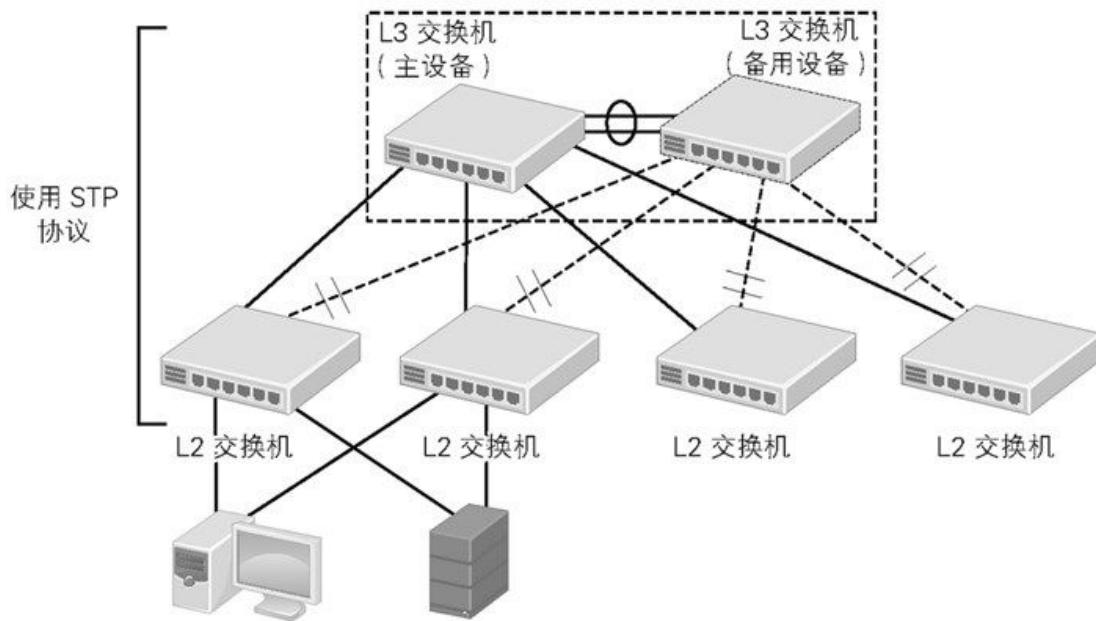


图 2.17 备用交换机模式

该模式适用于小规模网络，可以以较低廉的成本搭建网络。但是在大规模网络中，几乎每次增设设备都需要重新审视网络设计，因此不适合使用该模式。

各模式的比较结果与选择标准

我们将各模式的比较结果与选择标准分别总结在表 2.7 和表 2.8 中。

表 2.7 LAN 的可用性设计方式中各模式的比较结果

比较项目	高可靠性核心交换机	动态路由	VLAN	双机热备
------	-----------	------	------	------

冗余方式		线路冗余		设备冗余
网络的切换时间	◎：1秒以内	○：1秒左右	△：3~5秒左右	×：最多50秒左右 (RSTP ^{*1} 只需数秒)
可扩展性	○：高 通过增加设备可以实现横向扩展，还可减少网络设备的使用（还可应对子网增加、网络流量的突增）	○：高 通过增加设备可以实现横向扩展（还可应对子网增加、网络流量的突增）	△：一般 可纵向扩展（可以很容易地应对子网增加，但若想应对网络流量的突增，只能依靠横向扩展）	△：一般 通过增加设备可以实现横向扩展，但布线会变得很复杂（很难应对网络流量的突增）
解析问题的难度	○：较简单 逻辑上形成的是很简单的网络，不需要专业的网络协议知识。此外，使用供应商提供的综合管理工具可以轻松地确认故障状况，解析问题	×：难 运维中需要专业的网络协议知识 ^{*2} 。需要排查所有设备，查出问题原因	△：一般 由于使用了VLAN，必须参考物理布线与逻辑设计来排查问题	◎：易 4种模式中，该模式的网络构成最简单，最容易查找故障原因
设备的整合度	◎：非常高 将核心的L2、L3交换机在逻辑上整合到两个物理交换机上机来实现逻辑网络，还可以减少物理布线	△：中等 虽然全部使用了物理设备，但是通过路由协议可将使用了多台物理交换机的线路整合为一条逻辑线路	○：高 通过在物理交换机上使用VLAN，可将L2交换机整合到物理交换机上	×：低 全部使用物理设备
网络故障对业务	○：小 可在1秒内完成网络切换，使服务继续。用户几乎察觉不到	△：中 可在数秒内完成网络切换，使服务继续		×：大 完成网络切换需要1分钟左右的时间

和 服 务 的 影 响				
成 本	<ul style="list-style-type: none"> ○： 高 需要多功能/高成本的设备 	<p>△： 较高 需要设备冗余 (综合比较的话 要比高可靠性核心交换机模式便宜些)</p>	<ul style="list-style-type: none"> ○： 一般 	<p>◎： 低 如果交换机型 号相同，仅需 少量备用设备 即可</p>

*1 RSTP (Rapid Spanning Tree Protocol, 快速生成树协议)：在 IEEE802.1W 中定义的快速生成树协议，是 STP 的改进版。当发生故障导致 LAN 发生变化的时候，使用 RSTP 协议可有效缩短线路的切换时间 (STP 的收敛时间在 50 秒以内，但 RSTP 的在 1 秒以内)

*2 使用动态路由模式需要具备 OSPF (Open Shortest Path First, 开放式最短路径优先) 的路由知识，在 4 种模式的运维工作中是最需要专业知识的模式

表 2.8 LAN 的可用性设计方式的选择标准

模式	允许在多长时间内恢复业务	运维的负荷
高可靠性核心交换机	短 适用于需要瞬间恢复通信的情况	中等 减少了需要管理的网络设备和物理布线，适用于想减轻运维负荷的情况
动态路由	短 适用于需要在故障发生后 1 秒左右即恢复通信的情况	重 由于有许多物理布线、路由器、交换机需要管理，适用于具有足够的运维技巧与资源的情况

模式	允许在多长时间内恢复业务	运维的负荷
VLAN	中等 适用于需要在故障发生后数秒左右恢复通信的情况	中等 减少了交换机和路由器，适用于想减少运维所需资源的情况
双机热备	长 适用于允许故障后50秒左右恢复通信的情况	轻 适用于想大幅减少运维所需资源的情况

注意点

可用性策略的讨论顺序

在讨论可用性策略时，应当优先从 L3 等对网络整体的逻辑构成与体系结构有较大影响的高层开始讨论，并在选择策略时注意 L2 等底层的冗余不要与高层的策略相冲突。

讨论切换回原设备的程序

我们不仅要考虑故障时将网络切换到备用设备上，还要考虑当故障修复后再切换回原来的设备上的程序。

当故障发生后，无论是为了排查原因而暂时恢复网络还是真正修复了网络，都需要先在测试环境上进行测试和确认。

准备好设计和运维体制

仅仅采用昂贵的网络结构并不足以提高可用性，还需要充分准备设计和运维体制。

2.6 WAN 的可用性设计方式

现在许多企业在海外都有业务，这就需要系统能够向不同地理位置的用户持续提供同样品质的服务。也就是说，现在的系统还要能够与远距离的其他系统连接并进行数据交换。

实际上，许多系统都是通过 IP（Internet Protocol，互联网协议） - VPN（Virtual Private Network，虚拟专用网络）¹⁹ 和互联网等网络线路与其他系统进行连接和交互的。由于许多系统都使用了 WAN（Wide Area Network，广域网）²⁰，因此 WAN 的可用性也与系统的可用性息息相关。

¹⁹ 是在网络运营商的宽带网络上设定一块他人无法访问的虚拟封闭的网络，可以确保不同物理地点之间的相互通信具有很高安全性的一种服务。

²⁰ 与同一物理地点内的局域网不同，是连接不同物理地点的网络。

WAN 的冗余不仅可以提高其可用性，在发生故障时确保通信不中断，而且万一通信中断了，也可以缩短恢复所需的时间。

双网双工模式

该模式会搭建两套 WAN 并保持它们都处于工作状态（图 2.18）。在本书中，我们简称为“双网双工”。当其中一套 WAN 发生故障时，可在 1 分钟左右将通信切换到另外一套 WAN 上，恢复系统服务。但是由于故障时只有一套 WAN 可以使用，因此性能（通信速度）会比正常工作时有所下降。

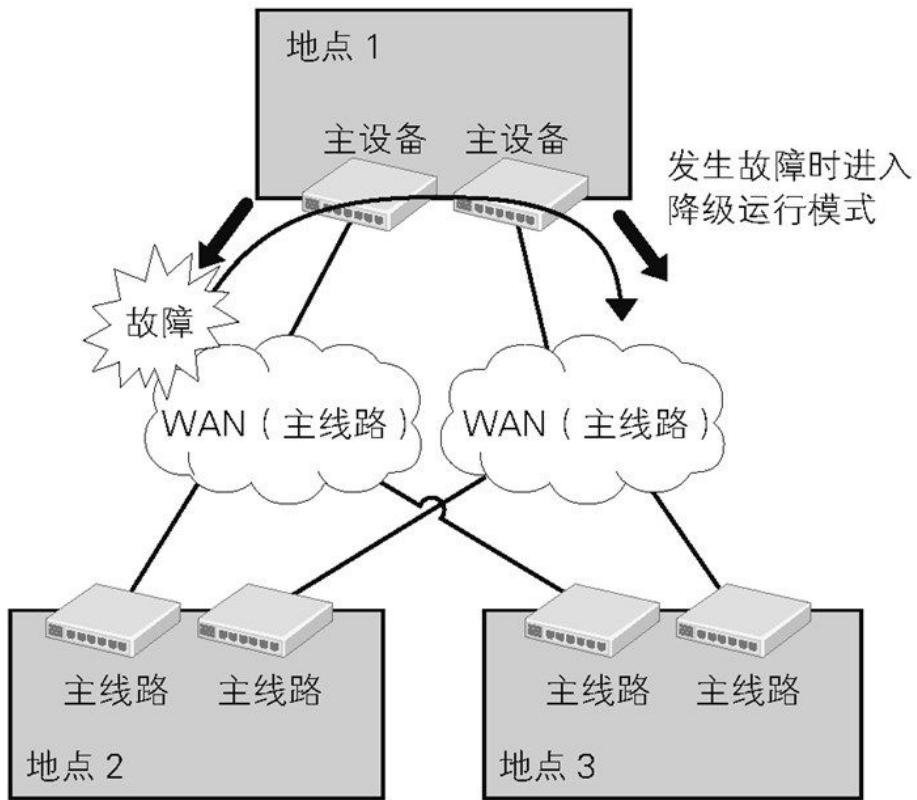


图 2.18 双网双工模式

双网热备模式

该模式搭建两套 WAN，一套主线路，一套备用线路（图 2.19）。在本书中，我们简称为“双网热备”。当主 WAN 线路发生故障时，可在 1 分钟左右将通信切换到备用 WAN 线路上。

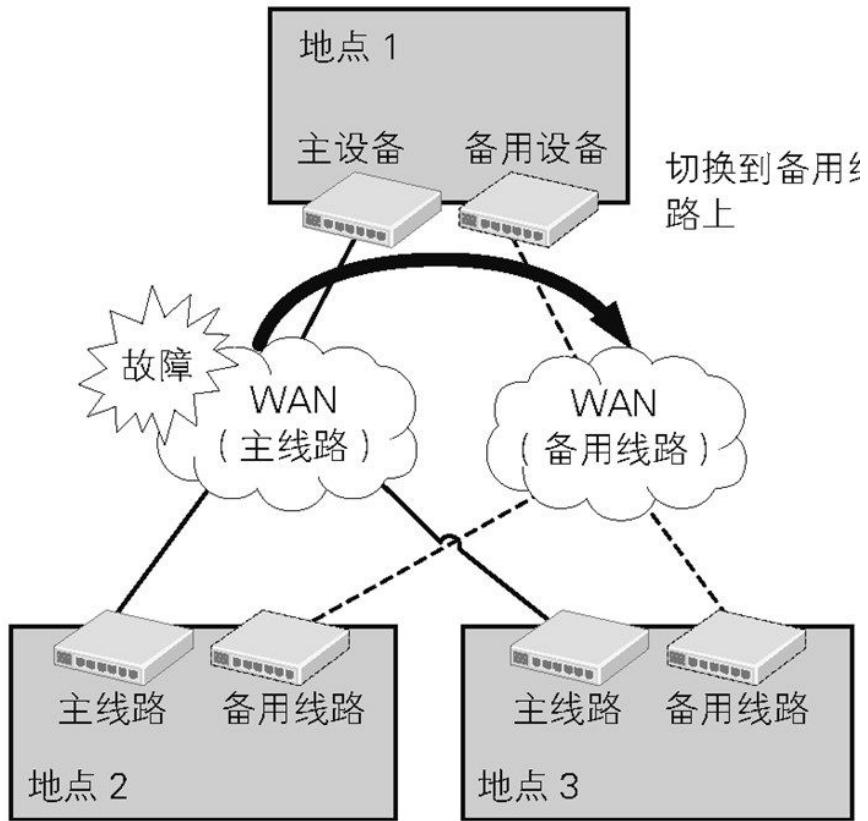


图 2.19 双网热备模式

单网双链路模式

该模式为一套 WAN 准备了两条（主用和备用）链路（图 2.20）。主用链路发生故障的话就将通信切换到备用链路上，即可在数分钟内将系统服务恢复到故障前的水平。

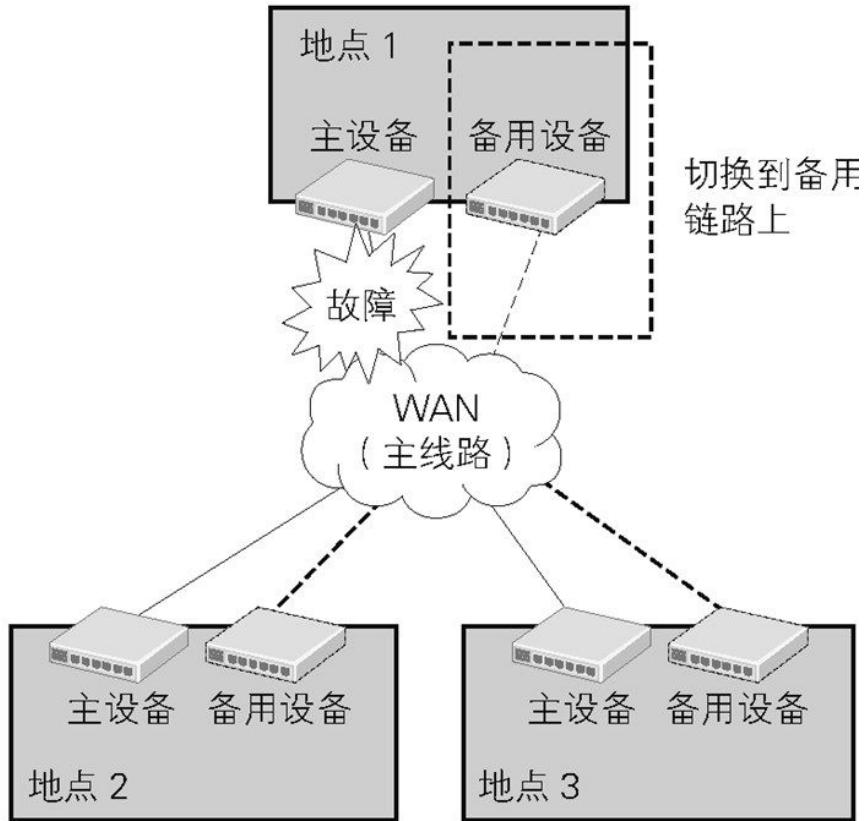


图 2.20 单网双链路模式

但是如果 WAN 自身发生了故障，将无法继续提供服务。

互联网 VPN 备用模式

该模式也是搭建两套 WAN，一套主 WAN 线路，一套备用 WAN 线路。使用互联网 VPN²¹作为备用 WAN 线路，可以降低运维费用（图 2.21）。与双机热备模式一样，当主用 WAN 线路发生故障时，可在 1 分钟左右的时间里切换到备用 WAN 线路上，使通信恢复。但是由于使用了 VPN，通信会转为尽力服务（Best Effort）模式。

²¹ 在互联网上设定一个他人无法访问的虚拟封闭网络，使特定的不同物理地点之间可以相互通信的一种服务。

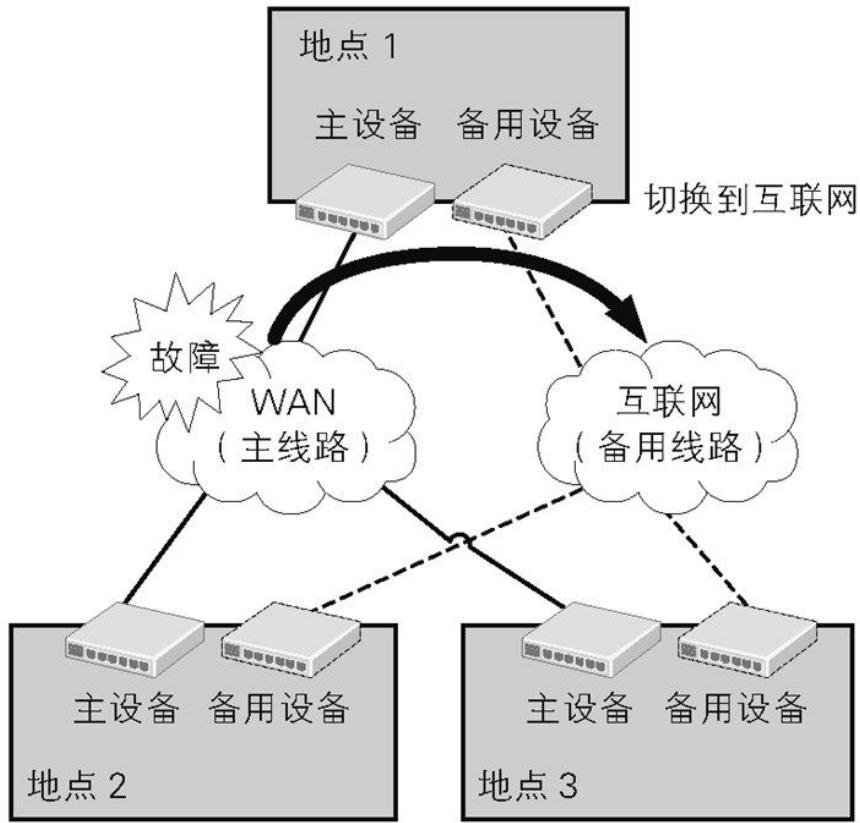


图 2.21 互联网 VPN 备用模式

ISDN 备用模式

该模式搭建两套 WAN，一套主 WAN 线路，一套备用。WAN 线路通过使用 ISDN（Integrated Services Digital Network，综合业务数字网）作为备用 WAN 线路，可以降低运维费用（图 2.22）。与双机热备模式一样，当主 WAN 线路发生故障时，可在数分钟左右切换到备用 WAN 线路上并恢复服务，但由于使用的是 ISDN 线路，所以通信速度会显著下降。

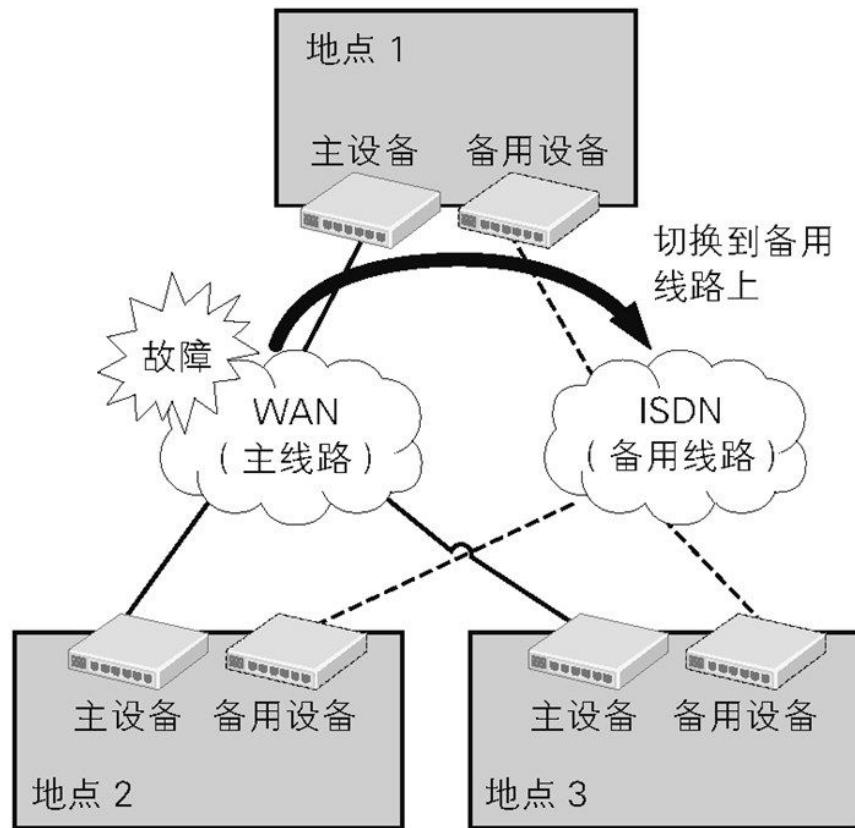


图 2.22 ISDN 备用模式

各模式的比较结果与选择标准

我们将各模式的比较结果与选择标准分别总结在表 2.9 和表 2.10 中。

表 2.9 WAN 的可用性设计方式中各种模式的比较结果

比较项目	双网双工	双网热备	单网双链路	互联网VPN备用	ISDN备用
构成(主用、备用)	主线路: WAN×2	主线路: WAN 备用线路: WAN (不同的运营商)	主线路: WAN 备用线路: WAN (同一运营商)	主线路: WAN 备用线路: 互 联网VPN	主线 路: WAN 备用线 路: ISDN
有无SPFO	○: 无		×: WAN	○: 无	

RTO ^{*1}	◎： 1分钟以内	○： 一分 钟左右	△： 数分 钟	○： 一分钟左 右	△： 数 分钟
网络发生故障时的性能 ^{*2}	△： 正常时的1/2	○： 与正常时性能相 同		△： 尽力服务 模式	×： ISDN通 信速度
运维性（网络设计、路由器设定的难易度）	△： 中 使用VRRP ^{*3} 可较容易地进行网络设计与路由器设定。	×： 难 故障的测试很复杂	○： 易	△： 中 由于使用了互联网VPN，需要注意网络安全	△： 中 故障测试和路由选择很复杂

*1 RTO (Recovery Time Objective, 恢复时间目标)：灾害导致业务停止后，恢复到预定水平所需要的时间

*2 上述双网双工、双网热备、单网双链路模式的比较前提是主线路与备用线路选择的是相同性能的线路（双网双工模式中主用线路与备用线路是同一线路）

*3 VRRP (Virtual Router Redundancy Protocol, 虚拟路由器冗余协议)：是可将一台虚拟路由器的责任动态分配到多台物理路由器上的通信协议。当一台物理路由器发生故障时，可自动地切换到其他无故障的物理路由器上

表 2.10 WAN 的可用性设计方式的选择标准

模式	允许在多长时间内 恢复业务	通信速度的下降	成本
双网 双工	短 适用于需要在故障后1分钟以内恢复业务的情况	约下降一半 适用于允许故障时的通信速度下降到正常时的一半左右的情况	非常高 需要一直保持宽带连接，由于两套网络同时工作，所以成本非常高
双网 热备	短 适用于需要在故障后1分钟左右恢复业务的情况	无 适用于需要始终保持一定通信速度的情况	高 需要一直保持宽带连接，成本高
单网 双链	长 适用于需要在故障		一般 需要一直保持宽带连接，

路	后数分钟左右恢复业务的情况		但因为是同一运营商，所以有折扣
互联 网 VPN 备用	短 适用于需要在故障后1分钟左右恢复业务的情况	尽力服务模式 适用于故障时允许通信进入尽力服务模式（互联网VPN）的情况	低 需要一直保持宽带连接，但因为是借助于互联网，所以成本较低
ISDN 备用	长 适用于需要在故障后数分钟左右恢复业务的情况	降低到ISDN水平 适用于允许故障时的通信速度降低到ISDN水平的情况	低 因为是窄带，所以成本低

注意点

考虑 WAN 的服务内容

使用某些网络运营商的服务，可能需要用到托管路由器²²，原则上用户是无法对路由器进行设置的。这种情况可能无法使用本节中所讲解的设计模式，因此需要事先和网络运营商确认。

²² 由网络运营商负责运维的路由器。有些运营商会要支付额外的运维费用。

考虑 WAN 的连接服务种类

有些网络连接可能在人口密度较低的地区无法使用，需要特别注意。

不同网络运营商构成 WAN 的连接的可用性也不同，还有可能需要执行骨干网²³冗余等冗余策略。特别是在核心业务²⁴中使用的网络服务，一定要事先向网络运营商确认服务内容是否能满足需求。

²³ 指的是连接多个地区的网络中的核心大容量网络。主要被用作 ISP 之间、国家之间和企业的重要地区之间的网络。

²⁴ 指的是那些一旦停止就会对社会产生很大影响，必须 365 天 24 小时不间断运行的业务。例如金融机关和交通机关的系统。

表 2.11 中列举了连接不同物理地点的连接服务的种类和它们各自的特点。此外，即使是采用的是同一种连接，不同的网络运营商所能提供的带宽、安全性策略以及可用性策略等服务的水平也不相同，因此需要根据需求定义考虑使用哪家网络运营商的服务。

表 2.11 网络连接服务的种类及其特点

特点	VPN (IP-VPN/互联网VNP)	窄带专用网	宽带专用网	广域以太网
使用的连接	可使用专线、ISDN、ADSL (Asymmetric Digital Subscriber Line, 非对称数字用户线路)、FTTH (Fiber To The Home, 光纤到户) 等种类丰富的网络连接	ISDN和数字专用服务等	ATM数据通信网、SONET (Synchronous Optical Networking, 同步光纤网络) / SDH (Synchronous Digital Hierarchy, 同步数字体系) 专线等	运营商的光纤等
带宽/保证	ADSL: 数Mbit/s~54Mbit/s FTTH: 10Mbit/s~10Gbit/s 互联网VPN: 尽力服务模式 IP-VPN: 带宽保证	ISDN: 64kbit/s ~ 128kbit/s 数字专用服务: 64kbit/s ~ 6Mbit/s (带宽保证)	ATM: 64kbit/s~600bit/s (带宽保证) SONET/SDH专线: 45Mbit/s~40Gbit/s (带宽保证)	广域以太网: 0.1Mbit/s ~ 1Gbit/s

2.7 互联网连接的设计方式

许多企业和组织不仅在互联网上浏览资料，还在互联网上运行着各种业务系统和进行各种交易。如今，连接互联网已经成为企业活动中不

不可缺少的一部分。特别是对于在互联网上进行重要业务活动的企业，如果不能连接互联网，造成的损失是难以估量的。

不仅仅是公司内部的路由器和 L2 交换机，当 ISP（Internet Service Provider, 互联网服务提供商）²⁵ 提供的服务发生故障时，公司内部的终端以及为公司外部提供服务的系统都会连接不上互联网，导致企业活动停滞。由于 ISP 提供的服务并不能确保完全不发生故障，因此应当事先制定故障时的可用性策略。

²⁵ 指的是提供互联网连接服务的组织。

可以通过 ISP 冗余或者将连接 ISP 的线路冗余来提高互联网连接服务的可用性。

BGP 多宿主模式

指的是使用 BGP（Border Gateway Protocol, 边界网关协议）²⁶ 实现多宿主²⁷ 的模式（图 2.23）。该模式由于使用了多个 ISP，即使其中一个 ISP 发生了故障，也需要约 1 分钟左右即可恢复网络连接服务，且通信速度不会降低。

²⁶ 指的是在互联网这种多个网络相互连接的环境中，用于各网络间交换路由信息的一种通信协议。

²⁷ 使用多条线路连接互联网等。

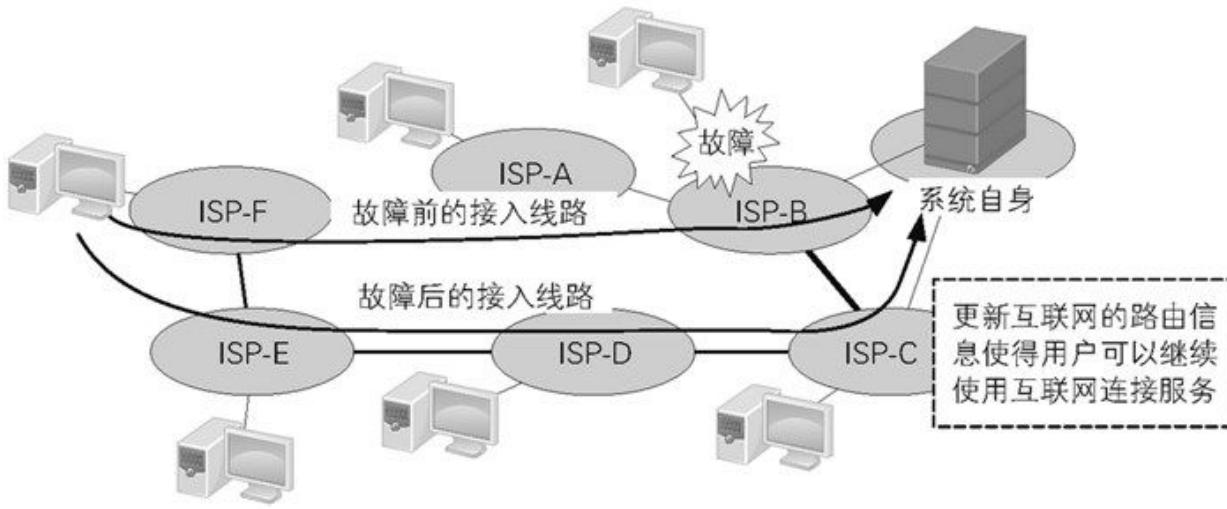


图 2.23 BGP 多宿主模式

但是为了实现 BGP 多宿主，必须要将系统自身也作为一个 ISP 来管理。因此，需要用到一般系统中并不需要的昂贵设备与专业知识来进行相关设定，所需时间和费用都会大幅增加。

双链路模式

该模式通过对连接互联网的线路进行冗余配置（图 2.24），可实现当主线路故障时，用 1 分钟左右的时间切换到备用线上，进而恢复互联网连接，并且连接质量不会降低。



图 2.24 双链路模式

但是当 ISP 自身发生故障时，系统将无法连接互联网。

单链路模式

该模式并不会对 ISP 和连接 ISP 的链路进行冗余配置（图 2.25），因此当故障发生时，用户将无法使用互联网连接服务。

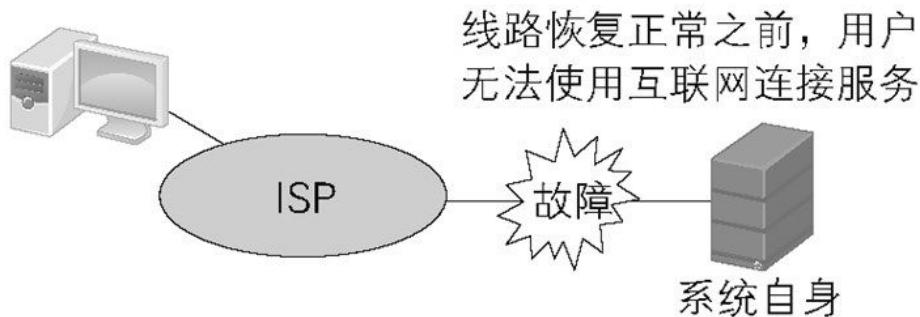


图 2.25 单链路模式

各模式的比较结果与选择标准

我们将各模式的比较结果与选择标准分别总结在表 2.12 和表 2.13 中。

表 2.12 互联网连接的设计方式中各模式的比较结果

比较项目	BGP多宿主	双链路	单链路
连接形态	使用多家ISP的连接服务形成冗余	使用2条线路连接一家ISP	使用1条线路连接一家ISP（无冗余）
IP地址转移性 ^{*1}	◎：有 可以在多家ISP中使用单独获取的IP地址	✗：无 只能使用ISP分配的IP地址	
有无SPOF	○：无	△：ISP	✗：ISP以及路由器
RTO	◎：1分钟以内	○：1分钟左右 ^{*2}	✗：直至SPOF恢复
通信路径的选择 ^{*3}	可自主在系统上选择路径	取决于ISP工作人员	

*1 双链路模式和单链路模式只能使用各家 ISP 分配的 IP 地址。因此，为了确保高可用性而使用（切换）多家 ISP 提供的互联网连接服务时，也必须切换 IP 地址。相反，BGP 多宿主模式虽然也使用多家 ISP 进行连接，但并不需要切换 IP 地址

*2 发生灾害时，如果提供网络连接服务的 ISP 发生了故障，则连接服务无法恢复

*3 指的是当 ISP 的连接因发生故障而需要进行切换时，如何选择通信路径。例如，有可能是路径中途的 ISP 发生故障导致了无法通信

表 2.13 互联网连接的设计方式的选择标准

模式	恢复业务的时间宽限	通信速度的下降
BGP多宿主	短 适用于故障发生后，需要用1分钟左右的时间恢复服务的情况	小 适用于总是需要保持一定通信速度的情况
双链路	短 适用于故障发生后，需要用1分钟左右的时间恢复服务的情况*1	无 适用于总是需要保持一定通信速度的情况
单链路	取决于ISP 在ISP恢复之前无法通信	无法通信 在ISP恢复之前无法通信

*1 但是，如果在发生灾害时 ISP 发生了故障，则连接服务无法恢复

注意点

这里需要注意的是 ISP 冗余的成本（表 2.14）。

表 2.14 ISP 冗余的成本

比较项目	BGP多宿主	双链路	单链路
成本	x: 高 需要与多个ISP签订特殊的合同。 此外，还需要准备只有ISP才使用的、昂贵的BGP路由器	o: 中 需要与ISP签订2条连接的合同。此外，还需要为每一个ISP连接都准备路由器	◎: 低 需要与ISP签订1条连接的合同

2.8 数据备份的可用性设计方式

现今，企业活动越来越依赖于IT系统，系统中保存的数据量也是越来越庞大。这些数据对于系统服务非常重要，如果数据丢失，会给企业带来很大的影响。例如，银行的金融数据、交易对象的数据、顾客数据等数据丢失的话，就会给企业的业绩和信用带来负面影响。

因此，我们必须要考虑如何对系统数据进行备份，将灾害和误操作等原因导致系统数据丢失的影响降到最小。

这里的备份指的是通过复制数据，来达到当备份对象机器发生了物理故障²⁸ 和逻辑故障²⁹ 时，系统也可以恢复的目的³⁰。

²⁸ 指的是硬盘等物理设备发生故障所导致的系统故障。

²⁹ 并非物理设备发生故障，而是操作系统和软件故障所导致的系统故障。

³⁰ 后面将会讲解的快照模式仅适用于逻辑故障。

本节中，我们将系统用来提供服务的数据进行备份的过程称为“一次备份”，将“一次备份”的数据再备份到其他存储设备上的过程称为“二次备份”。

此外，备份方式有 D2T（Disk To Tape，磁盘到磁带）方式³¹ 和 D2D（Disk To Disk，磁盘到磁盘）方式³²。D2T 方式已经使用了十几年了，但是近十年越来越多的解决方案采用磁盘存储，因此 D2D 方式也逐渐被广泛利用。此外，还有 D2D 方式与 D2T 方式组合形成的 D2D2T（Disk To Disk To Tape，磁盘到磁盘再到磁带）³³ 方式。磁带存储与备份服务器之间使用 SCSI（Small Computer System Interface，小型计算机系统接口）电缆和光纤信道电缆进行连接。

³¹ 将磁盘存储设备上的数据备份到磁带存储设备上的备份方式。

³² 将磁盘存储设备上的数据备份到其他磁盘存储设备上的备份方式。

³³ 先用 D2D 方式进行一次备份，将数据备份到磁盘存储设备上，然后再进行二次备份，将磁盘存储设备上的数据备份到磁带存储设备上。

存储复制模式

所谓存储复制是指总是保持某个存储设备上的内容与另外一个存储设备上的内容完全同步的功能。“一次备份”是将需要备份的数据同步到辅助卷上，“二次备份”则是对辅助卷上的数据再进行备份（图 2.26）。由于系统使用主卷上的数据来提供服务，所以“二次备份”对业务没有影响。存储设备与备份服务器之间使用 SAN（Storage Area Network，存储区域网）³⁴ 进行通信。

³⁴ 用于连接外部存储设备之间、存储设备与电脑之间的存储器专用高速网络。

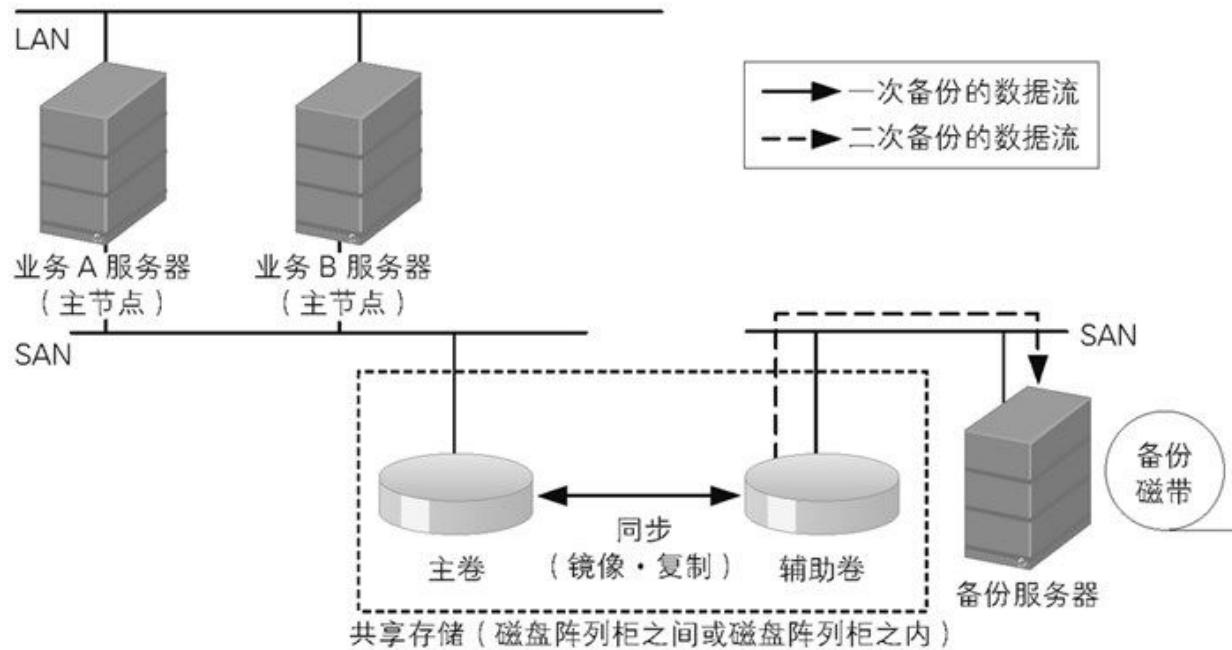


图 2.26 存储复制模式

对响应速度要求很高的系统和核心任务系统可以考虑采用该模式。

SAN 存储复制模式

该模式在存储设备与备份服务器之间使用 SAN 进行通信，通过一次备份将系统内存储设备中的数据完整复制下来（图 2.27）。

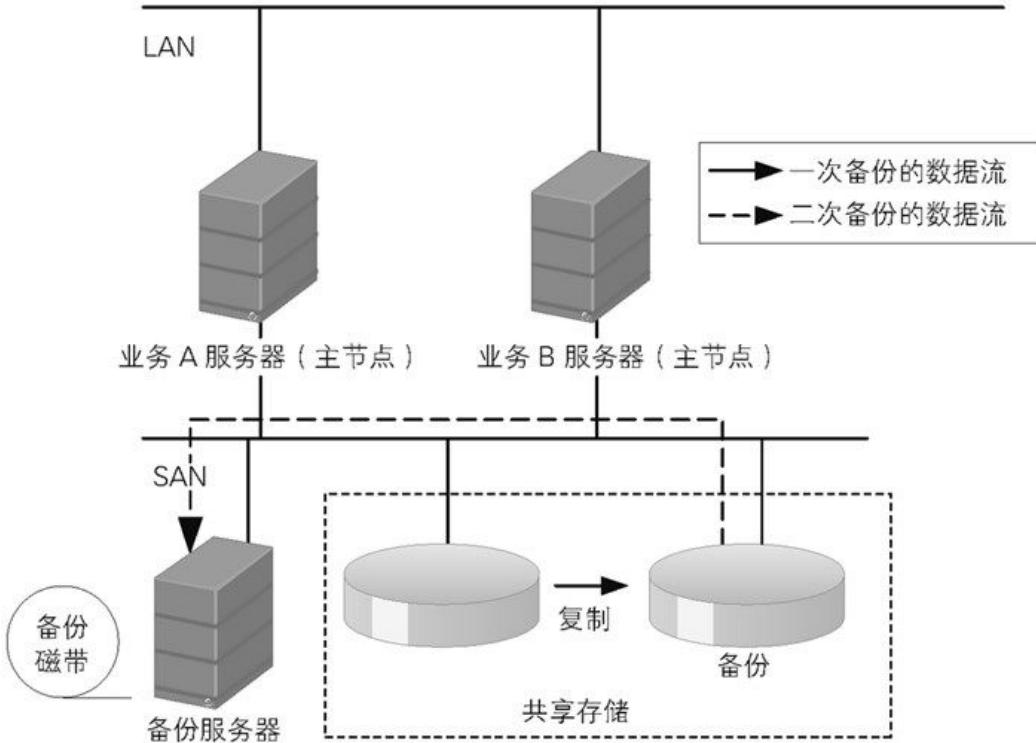


图 2.27 SAN 存储复制模式

由于一次备份采用了 D2D 方式，所以能够实现高速备份与快速恢复。但是，由于一次备份的数据与备份对象数据的容量几乎相同，所以需要很大的存储空间。二次备份是将一次备份的数据再备份到磁带存储设备上。

NAS 存储复制模式

与 SAN 存储复制模式不同，该模式是使用 NAS（Network Attached Storage，网络附加存储）³⁵ 进行备份的（图 2.28）。由于 NAS 使用 LAN 进行数据通信，所以一次备份和二次备份过程中 LAN 可能会对业务造成影响，这点也与 SAN 存储复制模式不同。

³⁵ 指的是通过连接 LAN 来进行存储的装置。

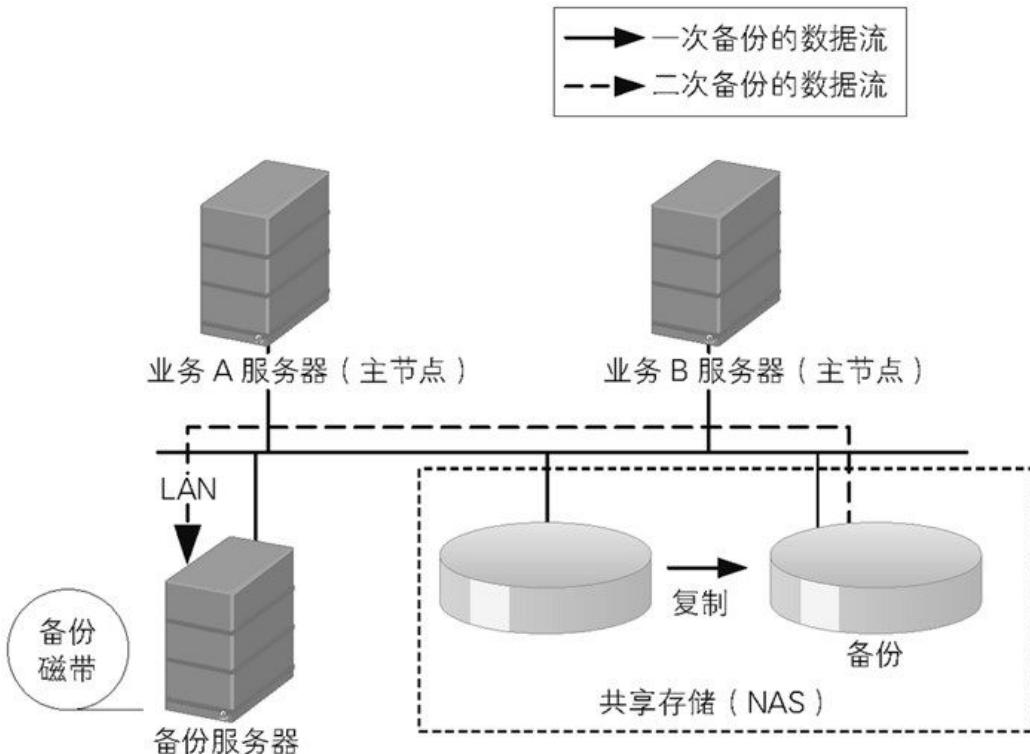


图 2.28 NAS 存储复制模式

SAN 快照模式

该模式是指取得快照³⁶后，在快照中进行数据备份。存储设备与服务器（业务以及备份）之间使用 SAN 进行通信（图 2.29）。由于是在快照中进行备份处理，所以对处理中的业务几乎没有影响。

³⁶ 在某个特定的时间点从正在运行的存储设备中提取出磁盘镜像。

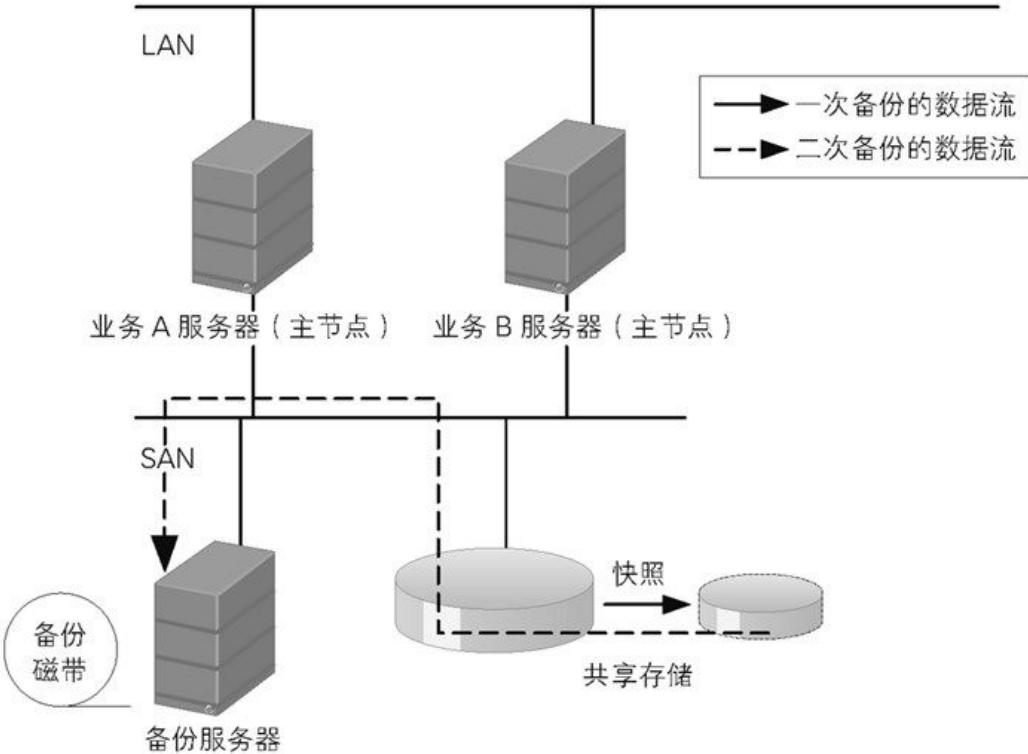


图 2.29 SAN 快照模式

对响应速度要求很高的系统和核心任务系统可以考虑采用该模式。但是，如果快照与作为备份对象的数据不匹配，则会导致数据无法恢复数据，因此还需要定期备份数据。此外，该模式还可以使用 RAID (Redundant Arrays of Independent Disks，磁盘阵列) 来应对故障。

NAS 快照模式

该模式是指在快照模式中不使用 SAN，而是使用 NAS 进行备份（图 2.30）。由于 NAS 使用 LAN 进行数据通信，所以备份过程中 LAN 可能会对业务造成影响，这点与 SAN 快照模式并不相同。

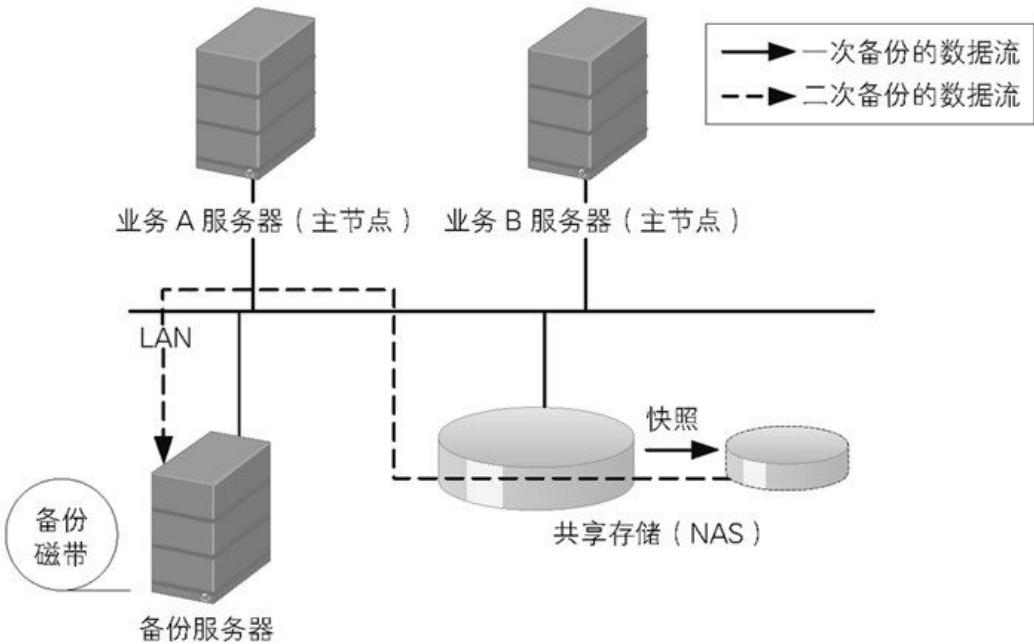


图 2.30 NAS 快照模式

业务服务器备份模式

业务服务器管理着文件服务器等备份对象数据，该模式就是通过使用业务服务器的资源（CPU、内存）等来进行备份的（图 2.31）。虽然业务服务器与存储设备之间使用 SAN 可实现高速通信，但是因为备份是直接在业务服务器上进行的，所以业务服务器的处理性能会有所下降。这种性能下降可能会对业务造成很大影响，例如系统服务器无法正常执行批处理等。因此，可以考虑在对响应速度要求很高，但不涉及核心业务的系统上采用该模式。

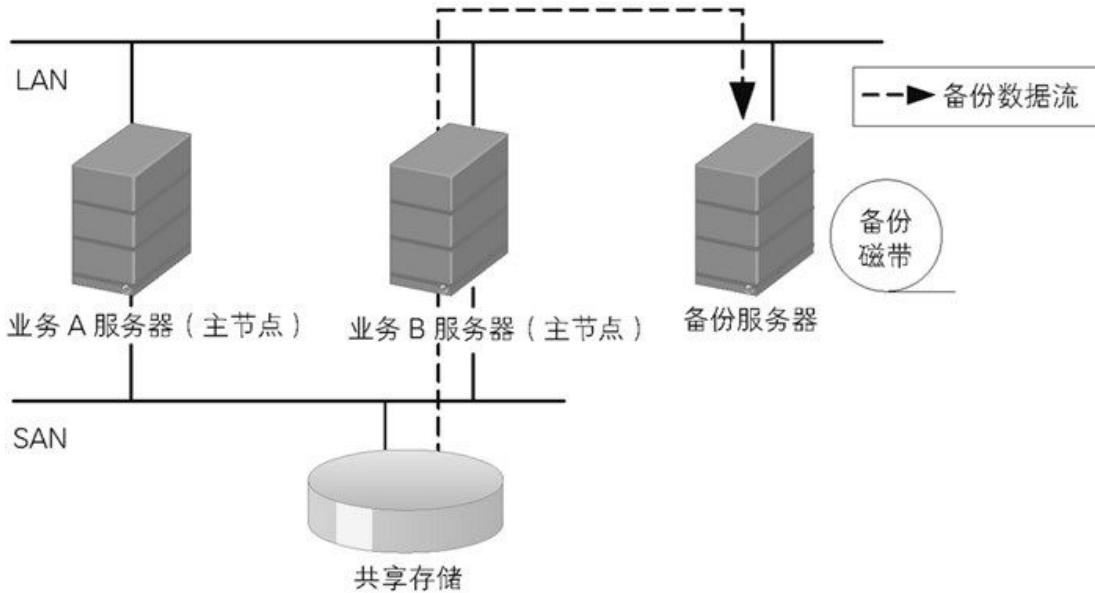


图 2.31 业务服务器备份模式

业务 LAN 备份模式

该模式使用连接着共享存储（NAS）的 LAN 来进行备份（图 2.32）。进行备份时会占用 LAN，可能会对业务造成影响。此外，如果在备份过程中数据发生了更新，则无法确保备份数据的一致性。由于该模式的数据恢复与 SAN 快照模式一样，需要数十小时，因此可以考虑在备份频率低，且几乎不涉及核心业务的系统上采用该模式。

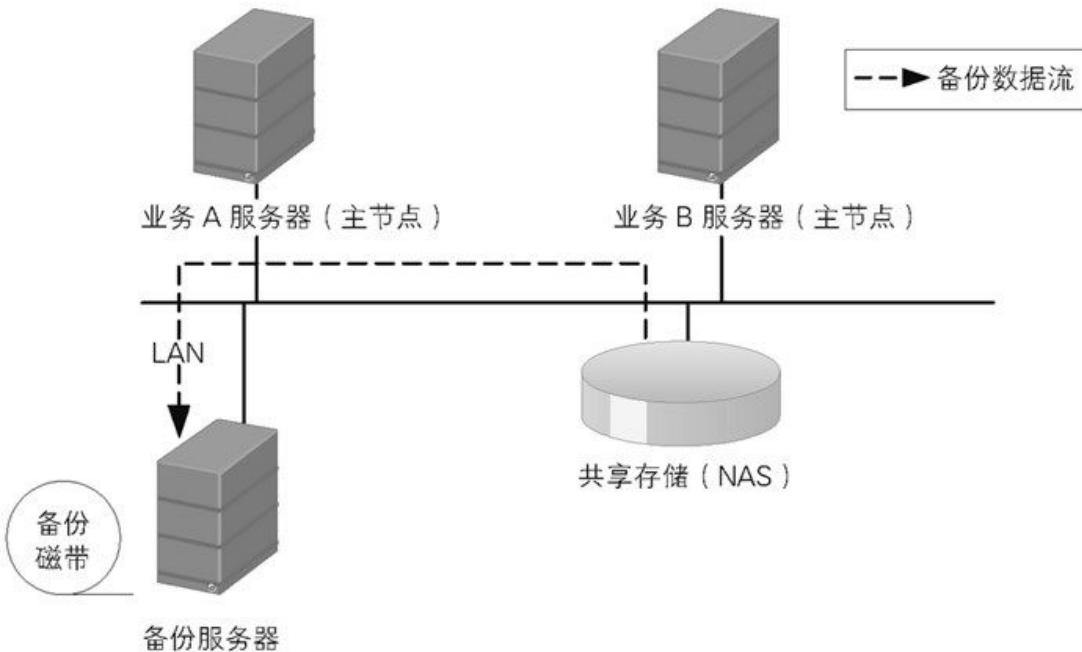


图 2.32 业务 LAN 备份模式

各模式的比较结果与选择标准

我们将各模式的比较结果与选择标准分别总结在表 2.15 和表 2.16 中。不过，使用 SAN 的模式与使用 NAS 的模式并无太大差异。在使用这些模式时，请根据实现这些模式的产品的成本和特点来进行选择。

表 2.15 数据备份的可用性设计方式中各模式的比较结果

比较项目	存储复制	SAN存储拷贝	NAS存储拷贝	SAN快照	NAS快照	业务服务器备份	业务 LAN备份
RTO	◎：数秒	○：数分钟～数小时		×：数十小时以上 ※1		×：数十小时以上	
RPO ^{※2}	◎：故障前的状态	×：数十小时前 最后取得备份数据的时间点		○：数分钟前 取决于取得快照的时间间隔		×：数十小时以前 最后取得备份数据的时间点	
一次备份的速度	◎：超快 在存储磁盘阵列柜内进行一次			— 快照可随时记录备		×：慢 从业务服务	△：一般

度	备份		份对象的变更历史		器直接备份	通过 LAN进 行备份
一次备份过程中对业务的影响	◎：无 对网络、业务服务器几乎没有影响	○：小 对存储设备自身有些许影响，但对网络、业务服务器几乎没有影响	△：中 备份时对磁盘和网络有影响。文件写入会增加I/O负荷	△：中 备份时对磁盘和网络有影响。文件写入会增加I/O负荷	×：大 由于对存储设备、网络、业务服务器的负荷比较集中，所以对业务会有较大影响	△：中 备份时使用了业务LAN，所以对业务有影响
二次备份过程中对业务的影响	◎：无 用于二次备份的SAN与业务SAN是独立的，因此对业务几乎没有影响	○：小 虽然备份时使用了业务SAN，但是对业务几乎没有影响	△：中 备份时使用了业务LAN，所以对业务有影响	○：小 虽然备份时使用了业务SAN，但是对业务几乎没有影响	△：中 备份时使用了业务LAN，所以对业务有影响	— 本模式中没有二次备份

在使用了 NAS 的模式中，还记载着备份过程与业务处理使用了同一个 LAN 的情况

*1 指的是发生物理故障时的 RTO。如果发生的是逻辑故障，数分钟即可恢复

*2 RPO (Recovery Point Objective, 恢复点目标)：指的是故障恢复时，可以恢复到多久之前的数据

表 2.16 数据备份的可用性设计方式的选择标准

模式	允许在多长时间内恢复业务	备份时对业务的影响
存储 复制	短 适用于在发生物理故障、逻辑故障时需要瞬间恢复提供服务的情况	不允许有影响 适用于需要性能稳定的情况
SAN 存储		

复制 NAS 存储 复制		
SAN 快照 NAS 快照	中等 存储设备发生物理故障时无法恢复。适用于存储设备发生逻辑故障后，需要在数分钟内恢复提供服务的情况	允许一定程度影响 适用于在备份时允许业务会有一定程度的延迟2的情况
业务 服务 器备 份	低 适用于当存储设备发生物理故障、逻辑故障时，允许服务停止一天以上的情况	允许有影响 适用于在备份时允许业务会有延迟的情况
业务 LAN 备份		允许一定程度影响 适用于在备份时允许业务会有一定程度的延迟的情况

在使用了 NAS 的模式中，还记载着备份过程与业务处理使用了同一个 LAN 的情况

注意点

各模式的备份方法

我们将各模式备份方法的差异记录在表 2.17 中。

表 2.17 各模式备份方法的差异

模式	存储 复制	SAN存储 复制	NAS存储 复制	SAN 快照	NAS 快照	业务服务器 备份	业务LAN 备份
SAN/NAS	SAN		NAS	SAN	NAS	SAN	NAS
是否使用 快照	无			有		无	
方式	D2D2T					D2T	

网络或存储装置的选择

由于数据备份过程中会产生大量的数据通信，可能会对其他通信造成影响。因此，选择数据备份中使用的网络或存储装置（SAN、NAS）时，不仅仅要考虑备份速度，还要考虑对系统内其他通信的影响。关于 SAN 与 NAS 的详细区别，请参考 6.2 节。

异地管理

我们还需要考虑异地保管备份媒介的相关策略，这样可以避免灾害发生时生产环境站点和其备份媒介同时被破坏。

备份次数与保存期限

在考虑备份次数与保存期限的时候，不能只从发生故障时的角度来考虑，还要追溯过去的内部管理信息等进行综合考虑。

备份对象

备份对象包括以下三种。

- 完全备份

备份全部文件与文件夹

- 差异备份

只备份上次完全备份后发生变化的部分

- 增量备份

只备份上次备份（包括完全备份和其他备份）后发生变化的部分

我们在图 2.33 中展示了数据的推移变化，并将各备份方法的备份对象整理在表 2.18 中。表中还记录了在第 5 天发生故障时恢复数据所需要的文件。

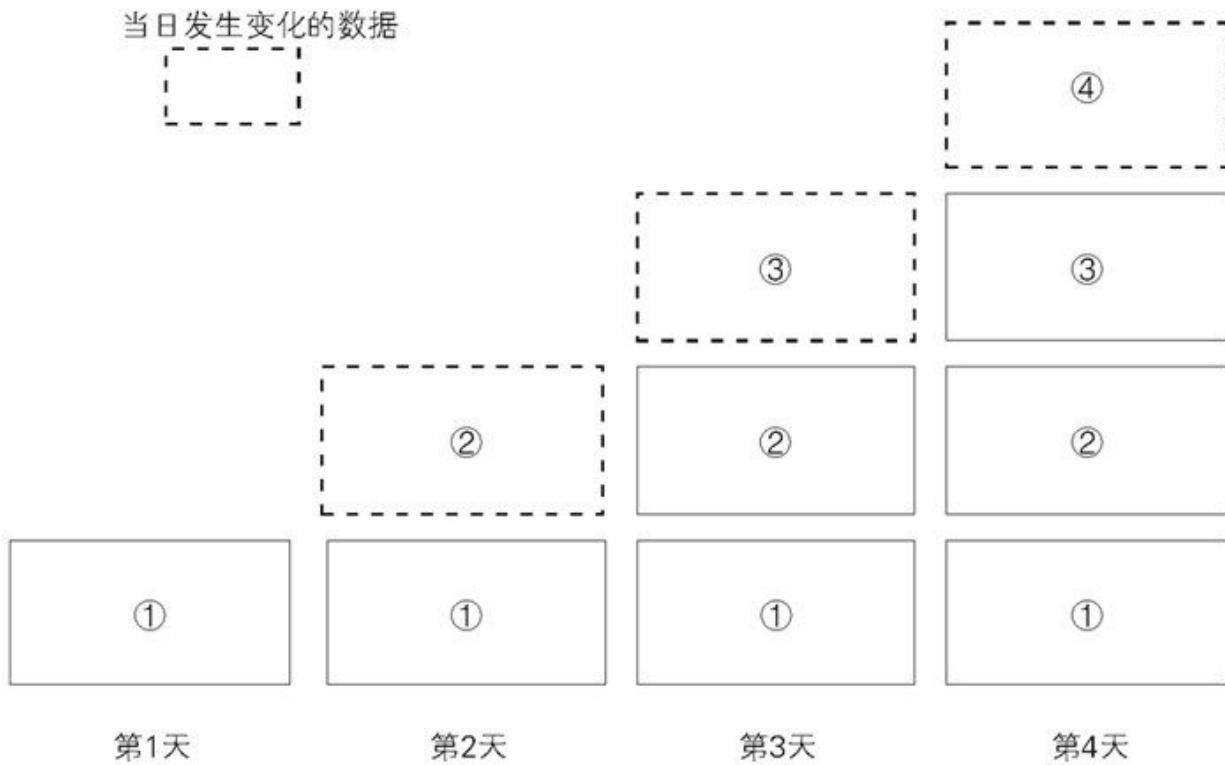


图 2.33 数据的推移变化

表 2.18 各备份方法的备份对象和恢复数据时所需要的文件

日期	完全备份	差异备份	增量备份
第1天 (备份对象)	① (完全备份)		
第2天 (备份对象)	①+②	②	
第3天 (备份对象)	①+②+③	②+③	③
第4天 (备份对象)	①+②+③+④	②+③+④	④
第5天 (恢复时需要的文件)	第4天取得的备份	第1天和第4天取得的备份	第1天~第4天取得的所有备份

此外，我们还将备份方法的选择标准整理在表 2.19 中。

表 2.19 备份方法的选择标准

备份方法	备份处理时间	RTO
完全备份	长 适用于可以确保有足够的时间进行备份处理的情况（例如每天数小时）	短 适用于需要尽可能缩短故障后的恢复时间的情况
差异备份	中等 适用于可以确保有一定的时间进行备份处理的情况（例如每天1小时）	中等 适用于需要在一定程度上缩短故障后的恢复时间的情况
增量备份	短 适用于几乎无法确保备份处理时间的情况	长 适用于允许恢复时间较长的情况

2.9 灾害应对策略的设计方式

不仅仅是企业活动，社会基础设施也越来越依赖于 IT 系统，因此系统一旦停止，就会对企业和社会造成很大的影响。例如，如果某生产商的生产管理系统因发生故障而停止运行，不仅仅会影响该企业交货的时间，还会影响其客户的各种企业活动；如果银行等金融系统停止运行则会影响资金流转；交通机关等管制系统停止运行则会影响人们的出行。

为了能确保企业活动能够继续进行、社会基础设施能够持续运转，企业和政府除了应当考虑应对系统设备故障外，还应当针对灾害和事故等意外状况制定 BCP（业务持续计划）。为了在发生灾害时能够继续进行业务活动和保护现场数据，我们需要异地搭建备用系统作为 BCP 站点，并进行数据备份。像这样的灾害应对策略我们称之为 DR（Disaster Recovery，容灾）。

我们将正常情况下提供服务的系统称为“生产环境站点”，将灾害发生时提供服务的系统称为“BCP 站点”。本节中，我们将介绍通过异地搭建 BCP 站点来将灾害时系统停止给社会带来的影响降至最低的设计方式。

广域集群模式

在 BCP 站点搭建与生产环境站点完全相同的系统，并在生产环境站点中设置自动检测和自动切换功能，当生产环境站点出现故障时，可自动切换（失效转移）到 BCP 站点（图 2.34）。

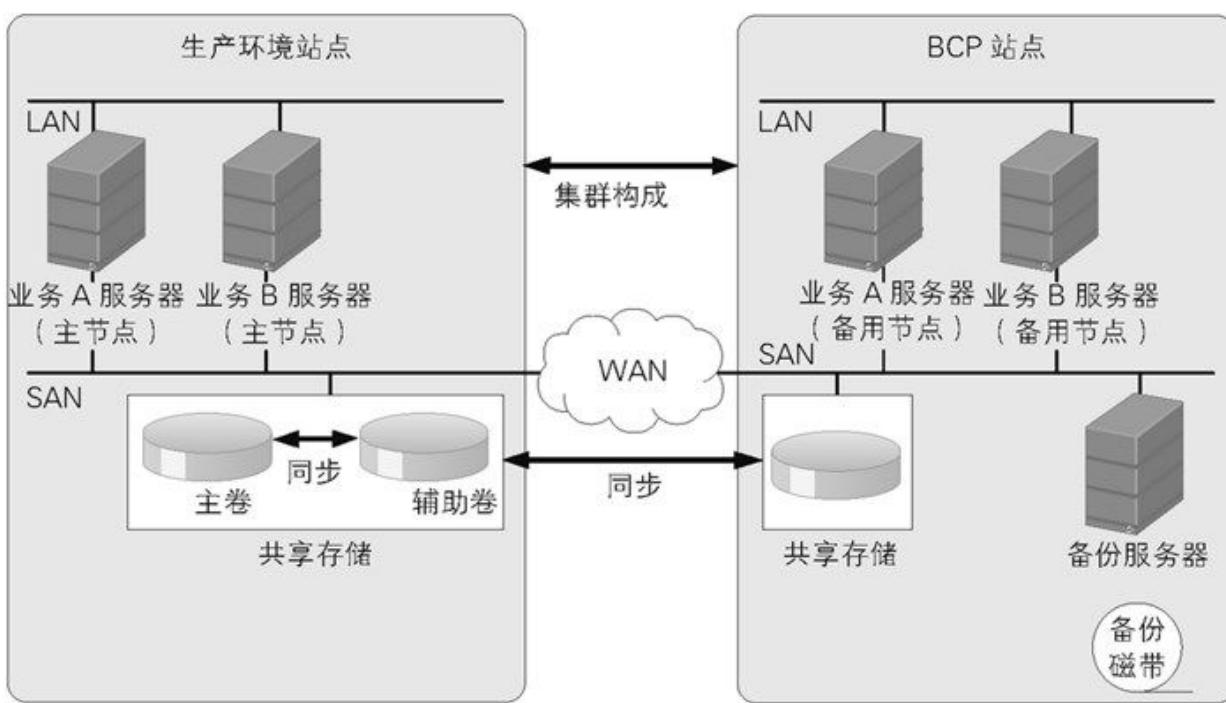


图 2.34 J域集群模式

本图展现的是广域集群的双系统热备构成方式。也可以以双系统双工方式和双系统互备的方式来实现广域集群

该模式可在短时间内恢复全部系统服务，适用于涉及人身安全、重要经济活动的系统。

双系统热备 DR 模式

在 BCP 站点搭建与生产环境站点完全相同的系统，通过远程镜像方式在 BCP 站点和生产环境站之间共享数据（图 2.35）。

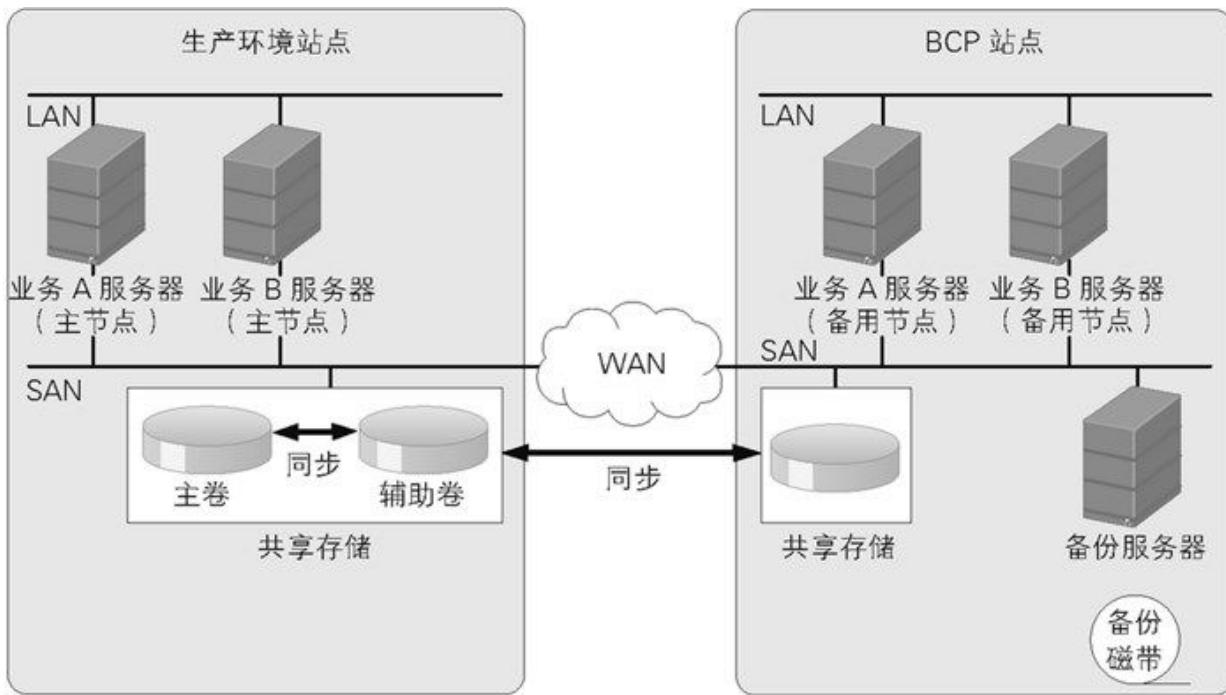


图 2.35 双系统热备 DR 模式

在正常情况下，BCP 站点作为备用站点，但当灾害发生时，可手动从生产环境站点切换到 BCP 站点上，恢复所有业务。

降级热备 DR 模式

在 BCP 站点中仅搭建部分系统功能或者是性能稍差的系统（图 2.36）。通过之前介绍的远程镜像方式在 BCP 站点和生产环境站点之间共享数据。与在 BCP 站点中搭建与生产环境站点完全相同的系统相比，费用较低。

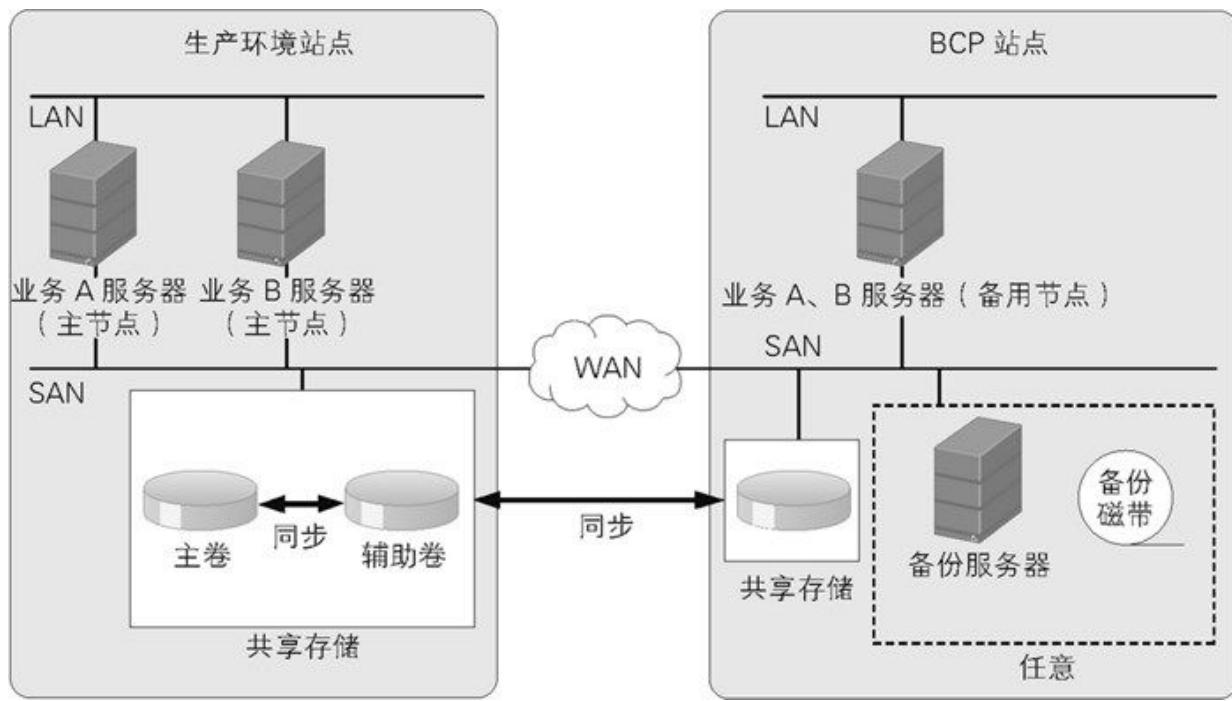


图 2.36 降级热备 DR 模式

发生故障时，可手动从生产环境站点切换到 BCP 站点。切换后，系统进入降级运行³⁷ 模式。

³⁷ 指的是只运行系统的一部分功能或者是运行系统时降低系统的性能、可用性等方面的质量。

备份转移模式

在 BCP 站点准备备用机器（图 2.37），当故障发生时，在 BCP 站点的备用机器中还原生产环境站点的备份数据以恢复系统服务。由于需要使用备用机器来构建 BCP 站点的运行环境，根据系统规模和数据量的不同，可能需要较长时间，甚至可能需要 1 天以上的时间来恢复系统服务。

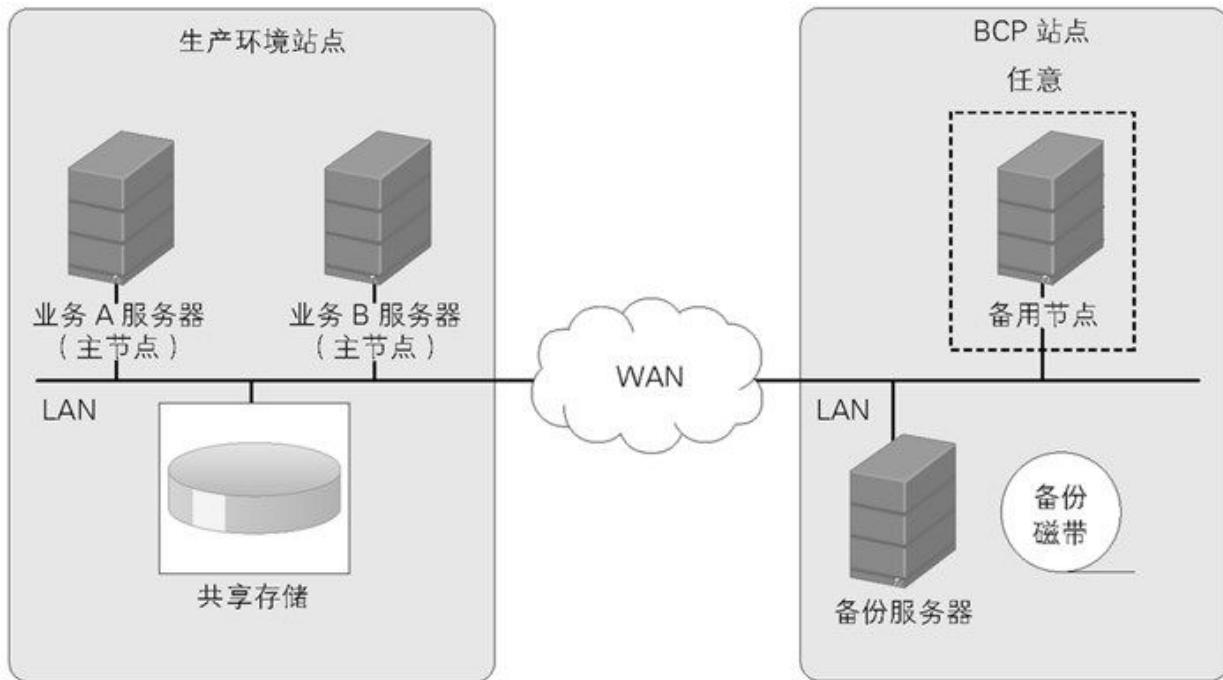


图 2.37 备份转移模式

远程镜像方式类型

有多种方法可以实现生产环境站点与 BCP 站点之间的数据共享。尽管其中也有方式可以恢复故障发生之前的数据，但是成本太高。因此，需要综合考虑经济价值和备用运维方案来进行选择。在这里，我们就介绍一下远程镜像方式类型和备份数据转移方式类型。

首先讲解远程镜像方式类型。

存储设备远程镜像类型

使用存储设备在生产环境站点、BCP 站点之间共享数据（图 2.38）。有些产品不仅支持异步方式（不必等待 BCP 站点发来文件写入完成的消息，就可以直接在生产环境站点将数据写入到 BCP 站点），还支持同步方式（收到 BCP 站点发来的文件写入完成的消息后，再将生产环境站点的数据写入到 BCP 站点）。

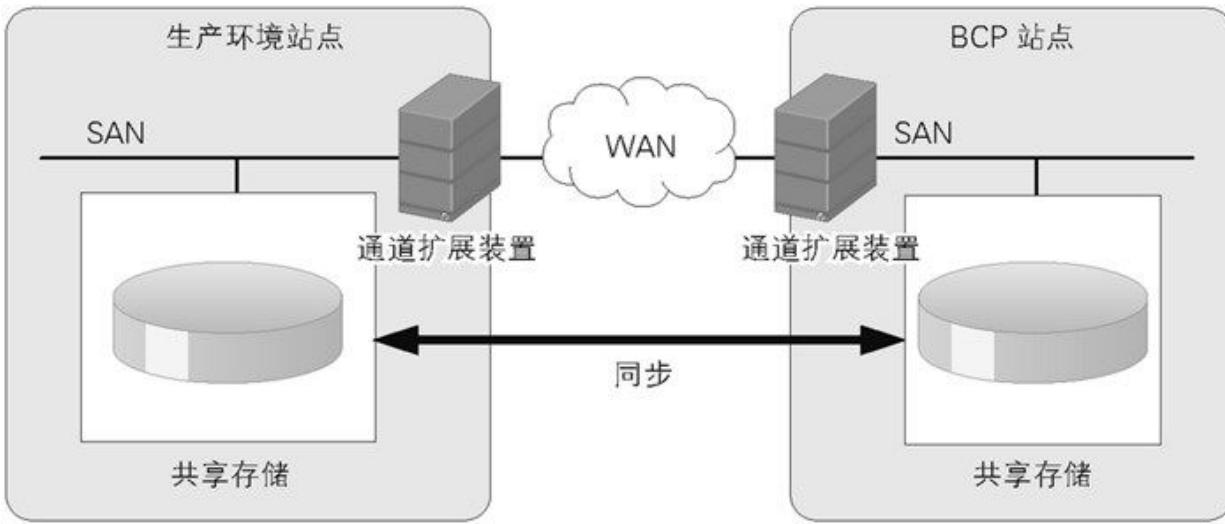


图 2.38 存储设备远程镜像类型

通道扩展装置是指为了能够延长两地之间光纤通道连接而使用的装置。由于在 SAN 内是通过光纤通道协议进行通信，因此不能用通过 IP 协议进行通信的 WAN 直接与其连接，而是需要使用通过光纤通道扩展装置来进行光纤通道协议和 IP 协议之间的相互转换，从而实现远距离异地通信。

采用同步方式实时共享数据时，原则上是可以将数据恢复到生产环境站点发生故障之前的状态。由于两个站点之间需要实时交换数据，因此需要保持 BCP 站点热备（备用机处于工作状态）。

软件复制类型

使用中间件的功能实现生产环境站点与 BCP 站点之间的数据共享，可实现同步处理和异步处理（图 2.39）。

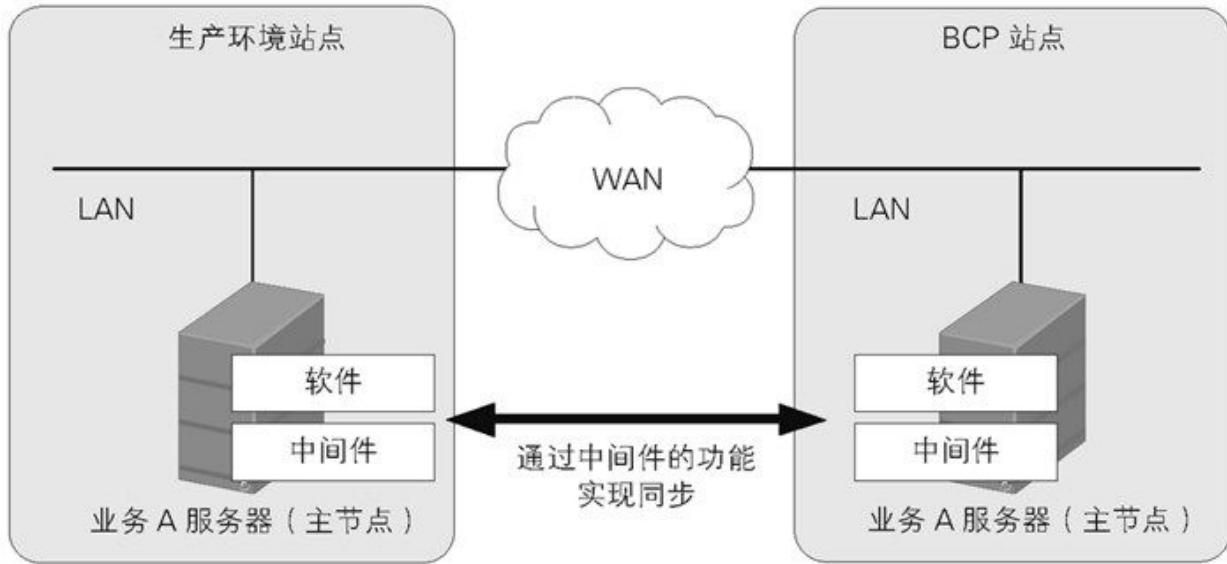
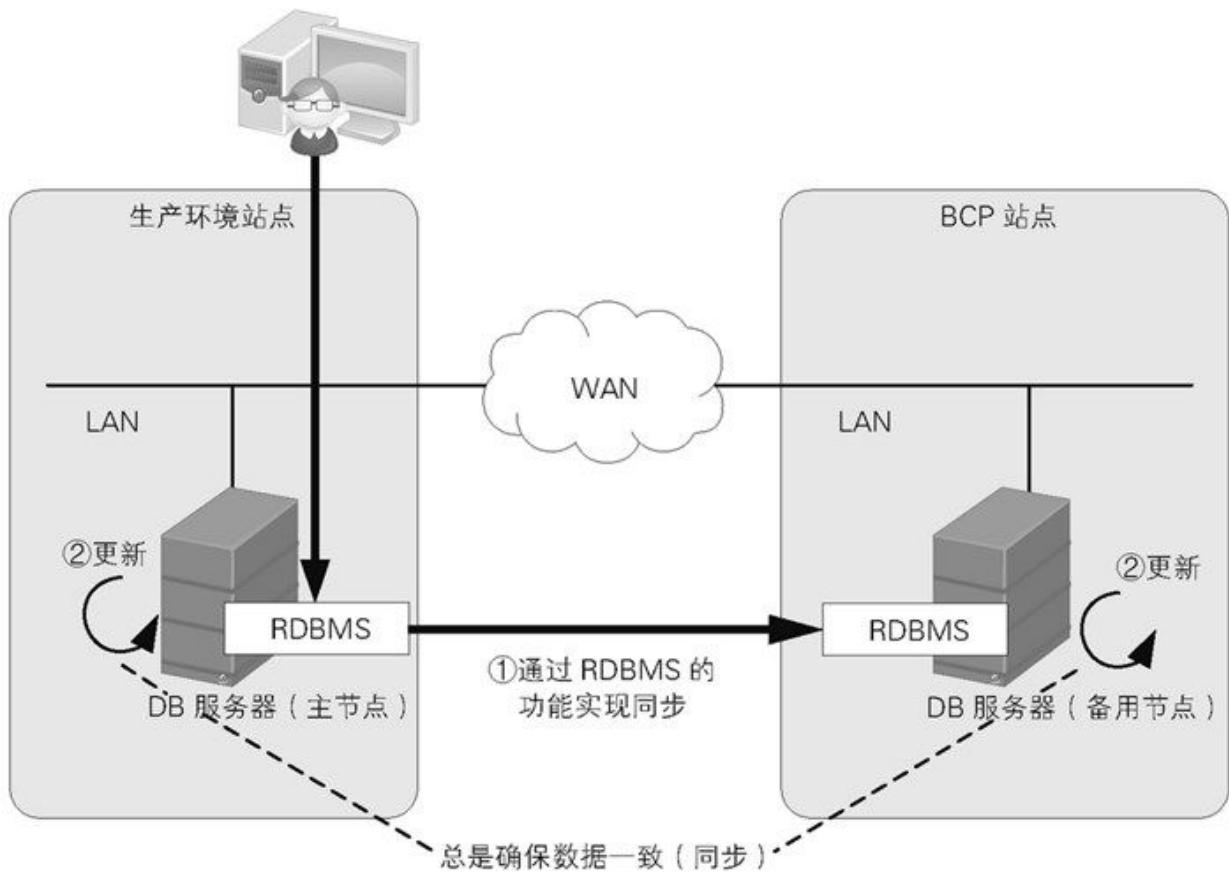


图 2.39 软件复制类型

采用同步方式实时共享数据的时候，原则上是可以将数据恢复到生产环境站点发生故障之前的状态。

RDBMS 复制类型

使用 DB 的功能实现生产环境站点与 BCP 站点之间的数据共享（图 2.40）。需要考虑采取同步方式还是异步方式来选择相应的产品。



上图为采用同步方式的示意图

图 2.40 RDBMS 复制类型

RDBMS 日志传送类型

使用 DB 在生产环境站点执行数据更新处理的时候，同时也在 BCP 站点对相同的数据执行相同的处理，以此实现生产环境站点与 BCP 站点之间的数据共享（图 2.41）。将生产环境站点执行数据更新处理后产生的日志存储一段时间（数分钟～数小时）后，通过网络传送至 BCP 站点。由于生产环境站点执行数据更新处理后再传送日志，所以反映到 BCP 站点时会有时间差，因此这属于异步方式。

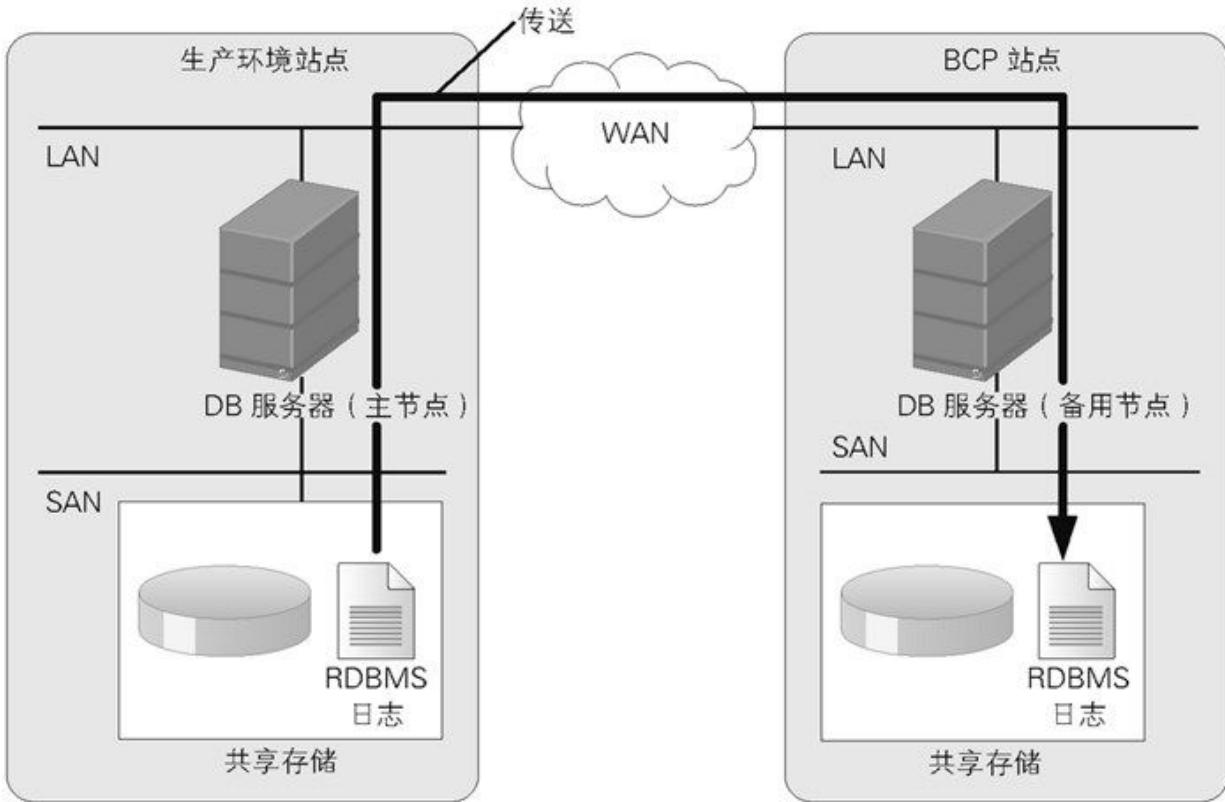


图 2.41 RDBMS 日志传送类型

备份数据传送类型

接下来讲解备份数据传送方式。

文件传送类型

每隔一定时间（例如一天一次）去获取业务应用程序等生成的数据文件，并通过网络传送实现异步数据共享（图 2.42）。

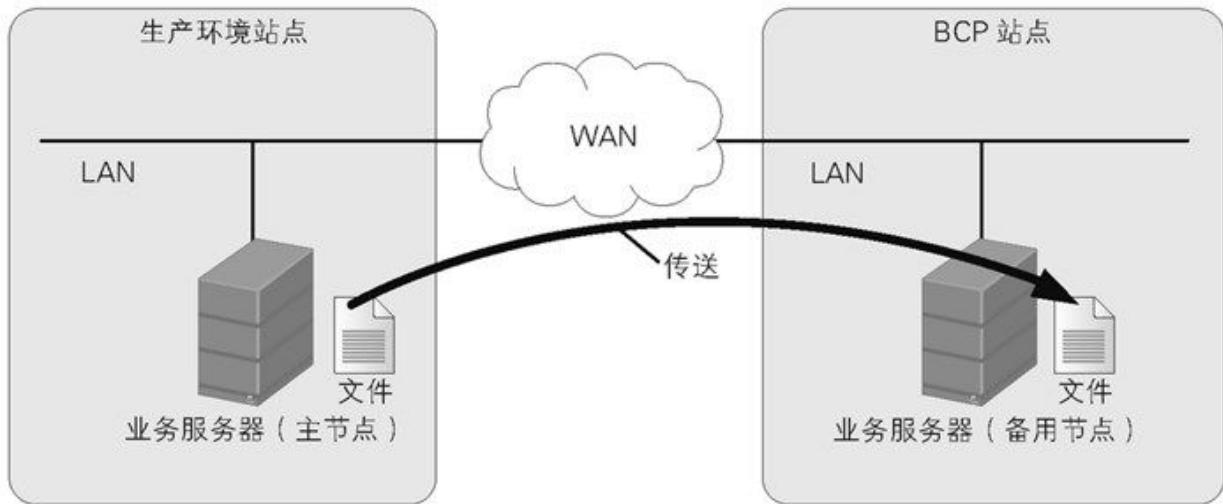


图 2.42 文件传送类型

远程磁带备份类型

将生产环境站点的数据通过 WAN 备份到 BCP 站点的备份磁带上（图 2.43）。当灾害发生时，可以通过将备份数据加载到 BCP 站点上来实现其与生产环境站点的数据同步。

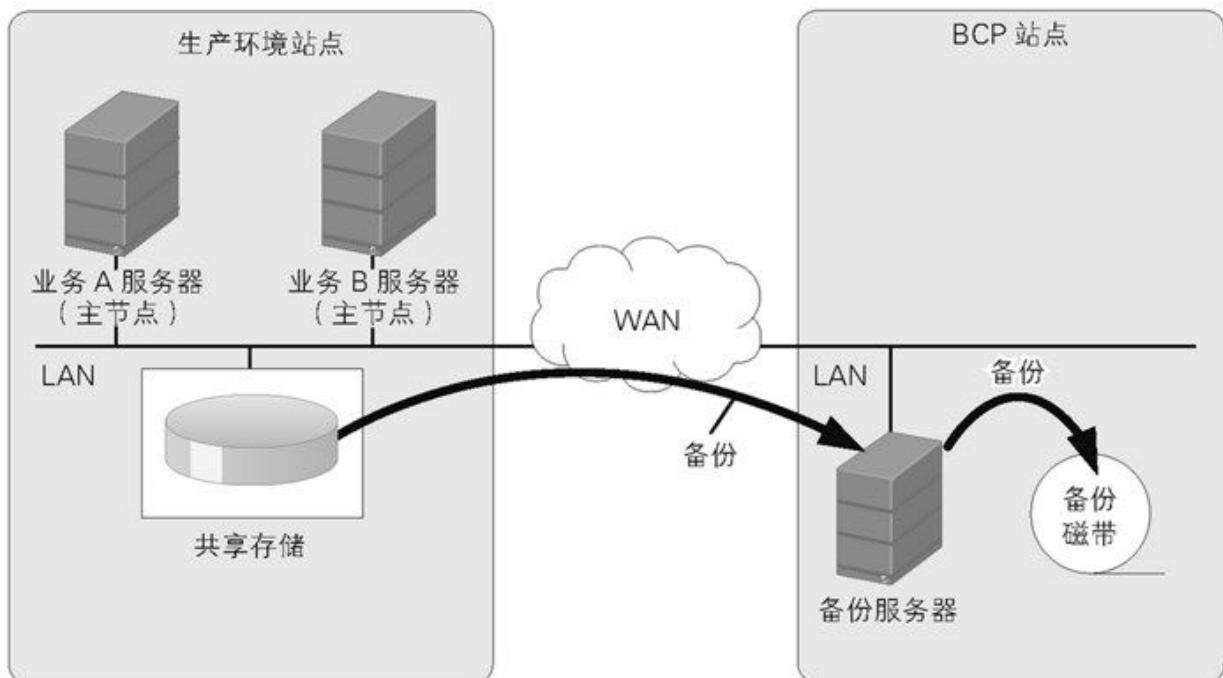


图 2.43 远程磁带备份类型

选择类型时的注意点

主要从 RPO（业务可以恢复到故障前哪个时间点）、RTO（业务恢复需要多少时间）考虑选择哪种类型（图 2.44）。一般来说，RPO 和 RTO 指标越高，成本也会越高，因此需要根据需求选择合适的类型。

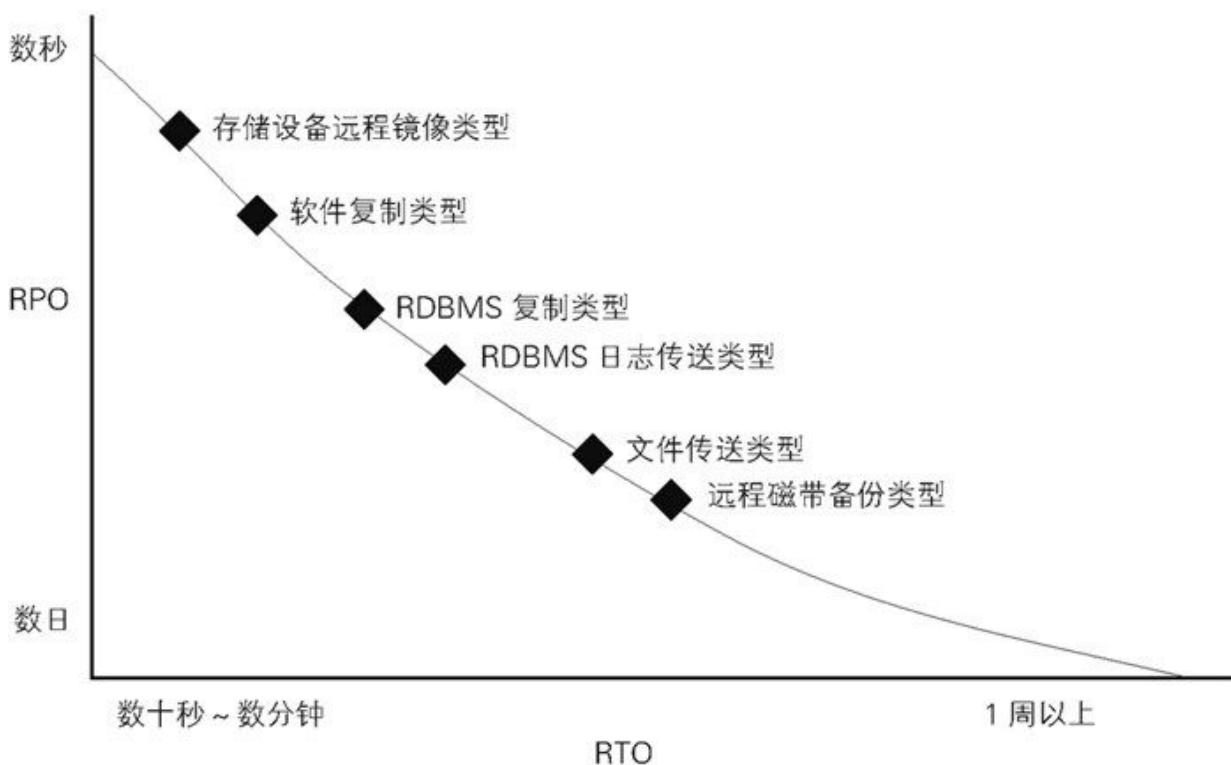


图 2.44 各数据共享方式中 RTO 与 RPO 的关系

各模式的比较结果与选择标准

我们将各模式的比较结果与选择标准分别总结在表 2.20 和表 2.21 中。

表 2.20 灾害应对策略的设计方式中各模式的比较结果

比较项目	广域集群	双系统热备DR	降级热备DR	备份转移

受灾时的措施	通过集群构成检测到生产环境站点受灾后，自动激活BCP站点的所有系统	在判断需要切换后，手动进行切換作业，激活BCP站点的所有系统	在判断需要切换后，手动进行切換作业，激活BCP站点的部分系统	将备份数据恢复到事先在BCP站点准备好的备用机器上
RTO	◎： 数十秒 ~ 数分钟 自动切换	○： 数十分钟~数小时 需要手动切换		△： 1 天以上 需要在生产环境站点恢复后进行数据复原或者是在BCP站点搭建系统环境
RPO	◎～△： 数十秒 ~ 数小时 根据远程镜像方式以及同步、异步方式有所不同		×： 1 天以上 取决于备份的时间间隔。但是由于对网络负荷较大，所以很难1天进行数次频繁备份	
RLO ^{*1}	◎： 可恢复所有的业务		○： 可恢复一部分业务（或降级运行）	
切换作业的负荷	◎： 小 灾害发生时自动进行切換作业，系统可确保生产环境站点与BCP站点的一致性	○： 中 灾害发生时需要手动切换生产环境站点与BCP站点。切换时需要遵从事先制定的切換作业程序，进行相关的设置作业		△： 大 灾害发生时需要基于备份数据搭建BCP站点的系统环境

*1 RLO (Recovery Level Objective, 恢复水平目标)：在 RTO 内，可以将系统恢复到什么程度

表 2.21 灾害应对策略的设计方式的选择标准

模式	RTO	恢复业务对象	切 换 方 法	适用系统
广域集群	数十秒 ~ 数分钟	所有业务	自动	适用于灾害发生后需要在数分钟以内恢复所有业务的系统

			切换	
双系统热备DR	数小时	部分业务或者部分资源（降级运行）	手动切换	适用于灾害发生后需要在数小时内恢复所有业务的系统
降级热备DR				适用于灾害发生后需要在数小时内恢复部分业务（或者全部业务）的系统
备份转移	1天以上			适用于灾害发生后允许花费数天时间恢复的系统

注意点

WAN 连接的选择

在同步生产环境站点与 BCP 站点时，必须注意选择高品质的 WAN 连接以防止通信中断。另外，必须事先估算可能会发生的通信流量并基于此选择合适的带宽。虽然带宽、距离与 WAN 连接成本成正比，但是没有足够的带宽就无法正确同步，因此需要尽可能准确地估算通信流量。

两地之间的距离

如果两地之间的距离过远，会导致网络延迟时间变长，可能会影响系统的处理时间。因此，在选择同步方式的时候，需要考虑地点之间的距离。

确保数据一致性

在正常情况下，如果系统与其他系统有数据交互，那么如果要在故障发生时继续这种交互，就必须考虑如何在灾害发生时确保数据的一致性。

运维方面的注意点

运维方面有以下注意点。

- 如果需要在生产环境站点与 BCP 站点之间手动切换，需要提前制定操作手册并进行演练，以防止人为操作失误。
- 在生产环境站点执行的 OS 升级、软件升级和应用程序的修改等维护作业也需要在 BCP 站点执行，以确保切换后 BCP 站点能正常工作。
- 如上条所述，生产环境站点进行的所有维护作业也都要在 BCP 站点执行，因此在 BCP 站点也会产生维护成本。
- 为了能在灾害发生时准确地判断是否需要切换系统以继续提供服务，必须事先明确灾害时做出决定和传达信息的流程，或是围绕 RTO 与系统运维服务供应商就运维制度的准备、维护等签订服务合同并适时进行调整等。

2.10 总结

本章我们讲解了可用性策略的基本思考方法和设计方式。系统宕机会使用户不能完成他们想做的事情，降低用户的满意度，导致企业错失商业机会、降低自身的信誉度。在可用性策略中，虽然考虑性价比很重要，但是如果只注重削减成本，没有采取最合适的策略，可能会导致系统运行后出现严重问题。因此，充分讨论可用性需求与策略之间的关系是非常重要的。

第 3 章 安全性需求的实现策略： 保护系统不受威胁

安全性需求是指与数据和系统的安全性相关的需求。如果系统的安全性需求不清晰或者没有制定实现安全性需求的策略，可能会发生预想不到的系统宕机和信息泄露，轻则给企业和机构带来经济损失和社会信誉损失，重则导致直接影响企业的存亡问题。因此，企业和机构必须充分重视 IT 系统的安全性策略。

在本章中，我们将讲解安全性策略的基本思考方法和设计方式。

3.1 安全性策略的基础

安全性策略是指从“预防”“检测”“恢复”的角度出发，想办法尽可能降低或是消除系统中使用的设备、业务数据、软件等各种信息资产受到损害的可能性。

首先，可通过制定系统运维守则和增加系统安全功能来防止安全事故发生。但预防的范围和程度都是有限的，因此当事故发生或者有事故先兆时，进行准确的检测是非常重要的。另外，当检测到事故发生后，如果事故的应对处理太慢，会导致事故影响扩大，因此还需要考虑如何能尽早使系统恢复到正常状态。

安全领域的技术发展日新月异，没有任何策略可以一劳永逸，定期地评审安全性策略非常重要。如有必要，还要重新制定安全性策略以应对新的安全风险。要想了解最新的安全性对策，可以阅读 IPA 上公开的《信息安全性策略实践资料》¹ 和其他相关文章、演讲资料²。

¹ <http://www.ipa.go.jp/security/awareness/awareness.html>

² <http://www.ipa.go.jp/security/read>

在讲解安全性策略之前，我们先来了解一下都有哪些攻击方式。在理解了这些攻击方式后，我们再讲解加密和认证等安全性策略。

攻击方式

恶意用户（Cracker，解密者）所采取的攻击方式中，有以客户终端为目标的，有以服务器为目标的，有以网络为目标的，还有以人（用户）为目标的。随着攻击技术的进步，以后可能还会有新的目标，但多数情况下的攻击都是利用了安全性策略中的漏洞发起的。我们将主要的攻击方式和其对其采取的策略整理在表格 3.1 中。

表 3.1 主要的攻击方式和策略

攻击目标	攻击方式举例	策略
客户终端	恶意软件 ^{*1} 等	杀毒软件
服务器 (应用程序层)	SQL注入 ^{*2} 、XSS ^{*3} 、CSRF ^{*4} 等	安全程序、 Web应用程序 防火墙 ^{*5} 等
服务器 (OS、中间件层)	端口扫描、Dos (Denial of Service, 拒绝服务) 攻击、恶意软件等	防火墙、恶意 访问检测、杀 毒软件等
网络	DDos (Distributed Denial of Service, 分布式拒绝服务) 攻击、DNS (Domain Name System, 域名解析系统) 缓存污染、窃听等	恶意访问检 测、加密等
用户	偷看密码、尝试登录攻击、钓鱼邮件等	安全培训、设 定密码策略等

*1 指的是计算机病毒、间谍软件、蠕虫和木马等带有恶意行为的程序和代码。

※2 将恶意的 SQL 命令插入到 Web 页面中的输入域或请求的查询字符串中，使其在 DB 系统中执行来进行攻击的方式

※3 XSS (Cross-Site Scripting, 跨站脚本攻击)：在根据用户输入的数据动态生成页面的 Web 站点中，在用户的浏览器上执行恶意代码进行攻击

※4 CSRF (Cross-Site Request Forgery, 跨站点请求伪造)：当用户访问事先设置了脚本的 Web 站点时，该脚本就被执行，向另一个 Web 站点发送伪造请求（如向论坛发帖和向电子商务网站发送订单）的攻击方式

※5 指的是通过直接监视外部网络与 Web 应用程序之间的交互内容（发送给程序的内容和 URL 等），可以防止非法入侵者利用 Web 应用程序的安全漏洞进行非法攻击的防火墙

加密

加密是指将明文数据根据既定算法转换为无法直接解读的其他数据，通过加密数据可以防止第三者窃听和篡改数据。反之，将加密后的数据还原的过程称为解密。

现在，通常情况都是根据已经公开的算法进行加密，这种方式已经被许多技术人员和数学家证明了几乎是无法破解的，可安全使用。而在许多遗产系统（Legacy System）中都使用了私有算法进行加密，其中大部分私有算法的安全性都无法确保，容易被窃听和篡改，所以现在已经不再推荐使用这种加密方式。我们在表 3.2 中记录了主要的加密方式和其特点。

表 3.2 主要的加密方式

加密方式	公开算法加密方式		私有算法加密方式
	对称密钥加密方式	非对称密钥加密方式	
特点	○：加密处理速度快 ×：密钥更新困难	○：密钥更新容易 ×：加密处理速度慢	×：解密方法确定 ×：安全性无保证
算法示例	Triple DES	RSA	XOR等

	AES Blowfish等	DSA ELGamal等	
--	------------------	-----------------	--

此外，加密的强度不仅依赖于加密算法的种类，还与密钥长度和密钥管理有关。即使使用了复杂的算法，但是如果密钥长度短、密钥管理不细致，加密强度就会变弱（图 3.1）。



即使是推荐的加密方式，也需要根据系统提供服务的时间来进行正确的选择

图 3.1 加密强度

认证

所谓认证，是指确认对象所持有的识别信息是否真实和合法。而在 IT 系统中，则是指为了防止他人非法访问而采取的、确保访问者和访问者所使用的设备的合法性的技术。根据认证对象不同，可以分为身份认证、服务器和终端认证、数据认证三类（表 3.3）。

表 3.3 认证对象的分类

分类	内容	实例
----	----	----

分类	内容	实例
身份认证	进行个人身份识别，确认是否为本人	通过在ATM机上输入取款密码进行身份认证
服务器和终端认证	确认向系统发送请求的服务器和终端是否是非法	通过移动电话的来电号码进行终端认证
数据认证	确认数据是否被第三者非法篡改	通过数字证书进行数据完整性认证

身份认证根据认证因素又可以分为信息认证、所有物认证、生物认证三类（表 3.4），也可将这些不同类型的认证因素相组合以实现更高的系统安全性。

表 3.4 身份认证的分类

分类	内容	实例
信息认证	通过只有本人才知道的信息和记忆进行身份认证	个人识别码、密码、密码保护问题等
所有物认证	通过只有本人才持有的物品进行身份认证	通过IC卡、硬件令牌进行一次性密码认证等
生物认证	通过人类身体或是动作特征进行身份认证	虹膜、手的静脉等

访问控制

所谓访问控制是指限制用户和设备，使其只能访问他们被赋予的权限所允许访问的资源。为了确保系统的保密性，必须为系统设计和实现合适的访问控制功能。IT 系统的访问控制方式主要分为任意访问控制、角色访问控制、强制访问控制三类。

任意访问控制

由资源的所有者来设定资源访问权限的控制方式。Windows 和 Linux 等普通操作系统的文件系统一般采用这种访问控制方式，可根据资源所有者的判断设定访问权限。不过，虽然这种方式能够灵活地提供资源，但是由于访问权限的设定都交由资源所有者负责，所以资源的保密性降低了。

角色访问控制

按照用户的职责（角色）设定资源访问权限的控制方式。在 Windows 和 PostgreSQL 等 OS 和中间件的用户账号管理功能中一般采用这种控制方式。通过将用户设定为“普通用户”“备份用户”“系统管理员”等角色，使他们只能进行其角色所被允许的操作。由于不是以用户为单位，而是以角色为单位进行权限控制，当用户发生人事变动时，只需要改变该用户对应的角色即可实现访问控制的变更。需要注意的是，采用这种访问控制方式时，必须事先缜密、细致地确定各种角色可以访问哪些资源和禁止访问哪些资源。

强制访问控制

资源的访问权限并非由资源的所有者决定，而是由系统决定。系统对访问主体（用户和设备）、访问对象（所访问的资源）设定各自的安全级别，通过比较他们之间的级别是否相匹配来判断是否可以对资源进行读取和写入操作。这种方式下，即使是资源所有者也无法变更权限的判定方式，可以提高资源的保密性。近年来，在 Linux 企业版和

Windows Server 等适用于企业的操作系统中，都带有强制访问控制或者与之类似功能。

网络安全性

为了防止内网中的系统受到来自外网（互联网）的非法访问，需要在内网和外网的边界采取安全措施。网络设计中典型的安全性策略有：引入防火墙、构建 DMZ（Demilitarized Zone，隔离区）和引入 IDS（Intrusion Detection System，入侵检测系统）等（图 3.2）。

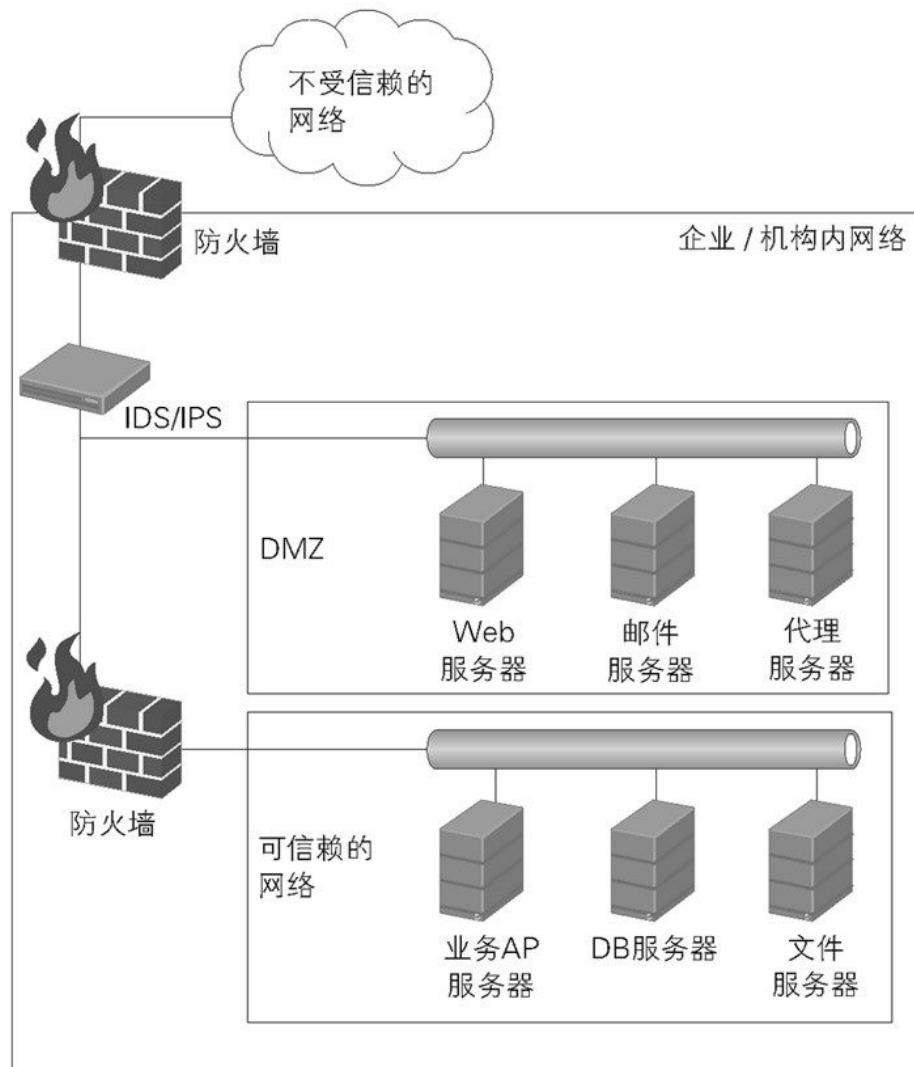


图 3.2 网络设计中的安全性策略

引入防火墙

防火墙的方针是明确拒绝所有不被允许的通信，能够对非法访问防患于未然，因此在内网和外网的交界处设置防火墙，可有效防止针对内网的非法访问。但是对于伪装成正常通信的非法访问，防火墙则无法检测出来。只有通过分析防火墙所记录的庞大日志才能发现问题。此外，在防火墙的运维过程中，需要依据供应商提供的信息定期地检查防火墙的相关设置。

构建 DMZ

在内网和外网之间，为 Web 服务器和邮件服务器等对外公开的服务器构建一块 DMZ。这样，即使对外公开的服务器遭遇了非法访问，也可以防止非法访问者以这些对外公开的服务器作为踏板继续入侵内部网络，造成二次损害。

引入 IDS

IDS 可以实时监测网络上的资源，一旦发现非法访问立即发送通知。IDS 还可检测出防火墙无法拦截的伪装成正常通信的非法访问。另外，通过对 IDS 进行扩展可实现 IPS（Intrusion Prevention Systems，入侵保护系统），在检测到非法访问后会自动地拦截通信。当然，在 IDS/IPS 的运维过程中也需要定期地更新非法访问的检测规则。

至此，我们讲解了安全性策略的基本思考方法。接下来，我们分类讲解安全性策略的设计方式。

3.2 非法访问应对策略的设计方式

如果放任非法访问对系统的入侵，可能会导致系统被破坏、无法使用，甚至是内部设备上保存的信息被泄露出去，后果非常严重。更有

甚者，非法访问者还可能入侵系统后将其作为踏板，使用系统发布恐怖袭击预告和发送垃圾邮件，实施犯罪行为。这种情况下，系统管理者不仅是受害者，同时也是加害者。如果是企业遭遇这种情况，不仅需要花费巨额款项用于入侵事件的调查和损害赔偿，还会造成社会信誉度急剧下降，给企业发展带来严重影响。因此，对于引入了系统的所有企业来说，考虑非法访问的应对策略是非常重要的。

防火墙模式

在内网和外网交界处设置防火墙，限制外部访问（图 3.3）。如果对外公开服务器遭到入侵，有可能入侵者会以它们为踏板继续入侵非公开服务器，造成二次损失。因此，该模式适用于将与外网的连接限定在来自客户端 PC 的 Web 浏览上，或即便是公开服务器也要限定允许外部访问的内容（例如仅允许访问静态 HTML）的情况。

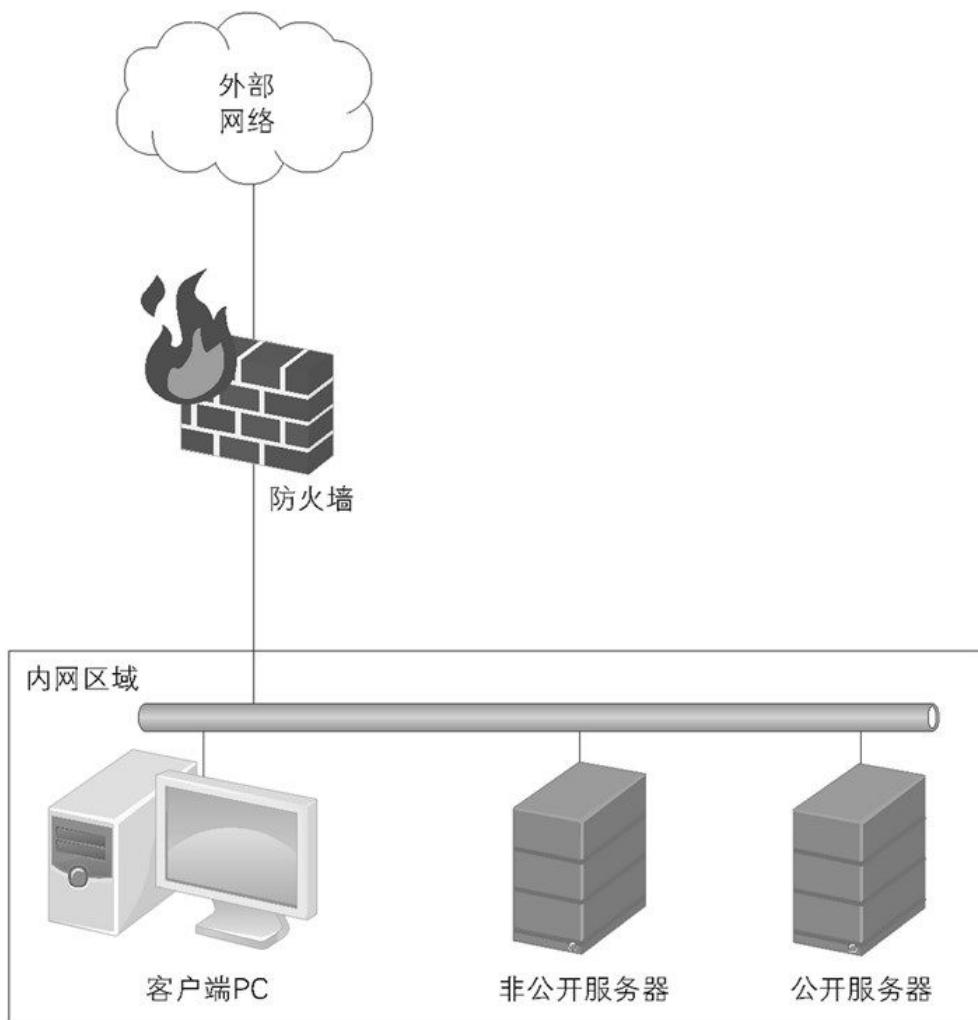


图 3.3 防火墙模式

单防火墙 DMZ 模式

通过防火墙搭建公开服务器专用的网络区域（DMZ）（图 3.4），可以拦截所有外网对内网的直接访问，并严格限制外网经由 DMZ 对内网的访问。这样，即使公开服务器遭到了入侵，也可以减小对内网节点中非公开服务器的影响。

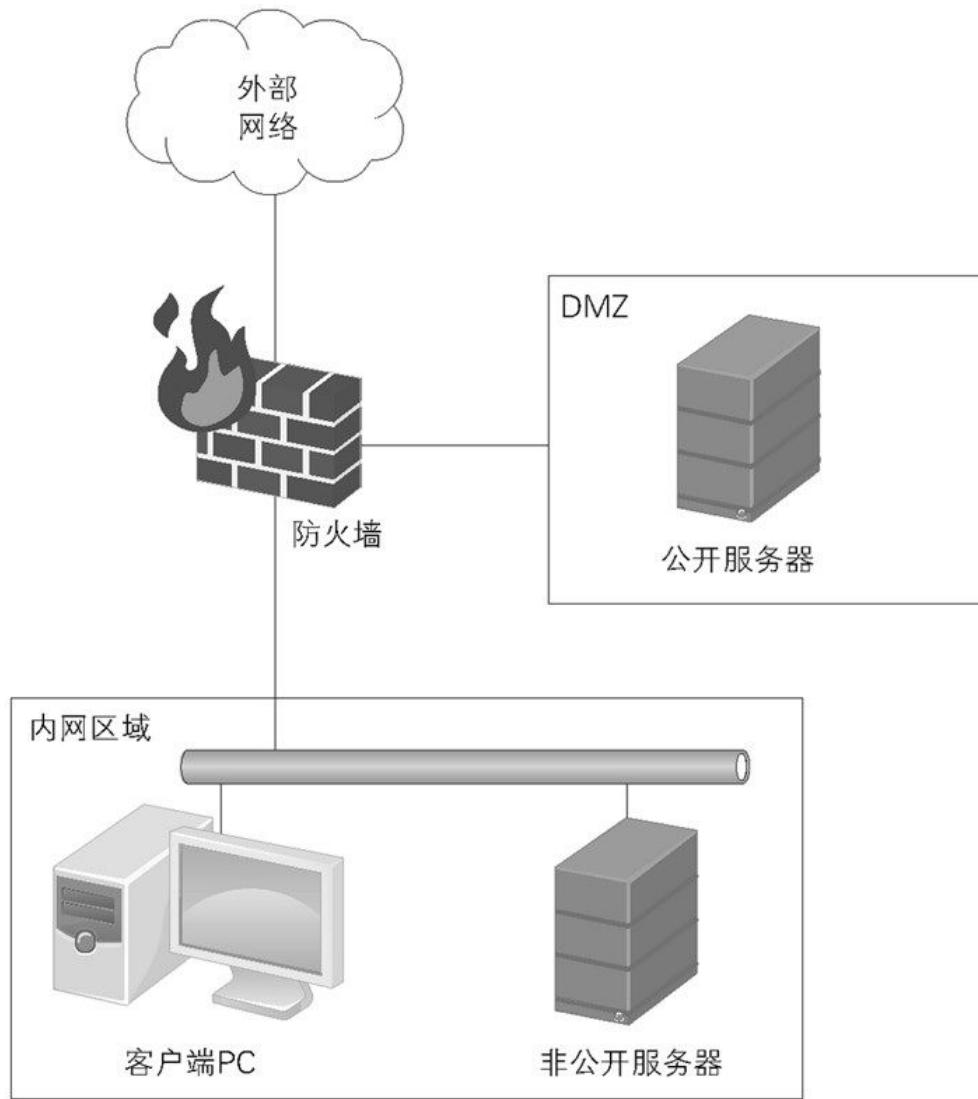


图 3.4 单防火墙 DMZ 模式

双防火墙 DMZ 模式

通过两台防火墙包围公开服务器，构建专用的网络区域（DMZ）（图 3.5）。与单防火墙 DMZ 模式相同，即使公开服务器遭到了入侵，也可以减小对内网区域中的非公开服务器的影响。虽然 2 台防火墙成本上会有所增加，但使用两家不同供应商提供的防火墙可以降低因防火墙自身的故障和缺陷导致被入侵的风险。

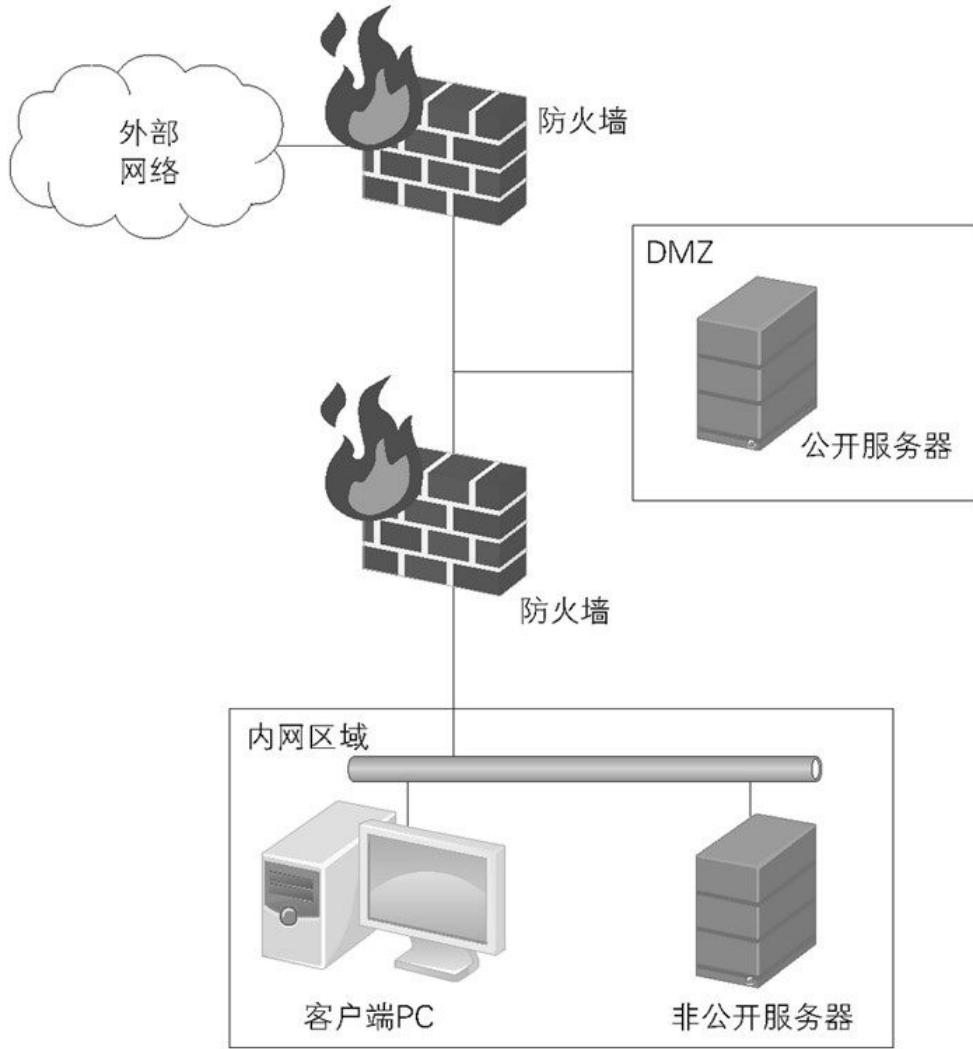


图 3.5 双防火墙 DMZ 模式

DMZ + IDS 模式

除了在 DMZ 中设置公开服务器外，还引入了 IDS（图 3.6）的模式，这样可以检测出防火墙无法拦截的伪装成正常通信的非法访问，并向管理员发出警报。为了避免将正常访问误报为非法访问或是漏报非法访问，需要持续对特征（Signature）³ 和描述⁴（Profile）进行调优。

³ 检测非法入侵和攻击数据包的规则和模型。

⁴ 检测异常的规则。例如，事先设定通信阻塞的阈值，这样当通信数据量超过阈值时就可检测出通信异常。

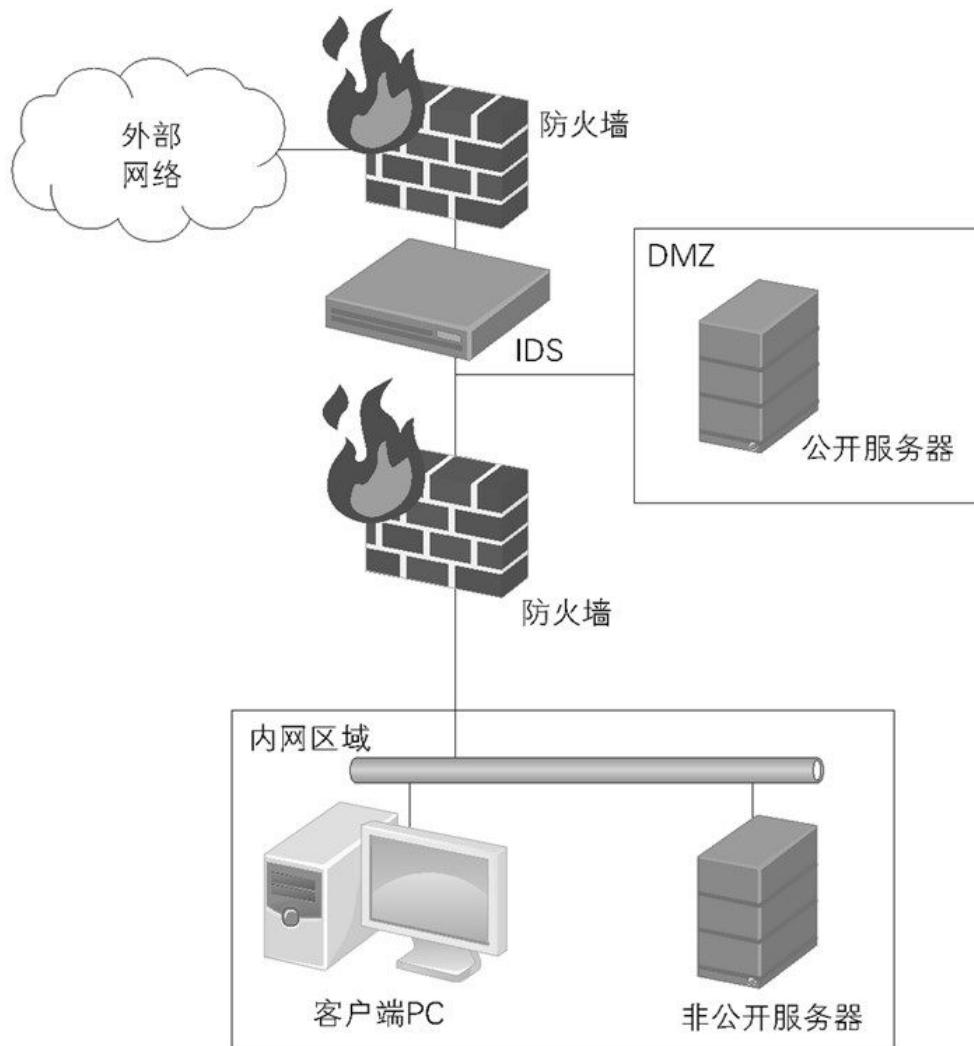


图 3.6 DMZ + IDS 模式

DMZ + IPS 模式

除了在 DMZ 中设置公开服务器外，还引入了 IPS（图 3.7）的模式，除了具有 IDS 的检测和警报功能外，IPS 还可自动阻止通信，可降低因警报后处理不及时而遭受损失的风险。但同时，误报引起的通信中断也可能导致系统服务停止。所以与 IDS 一样，需要持续对特征（Signature）和描述（Profile）进行调优。

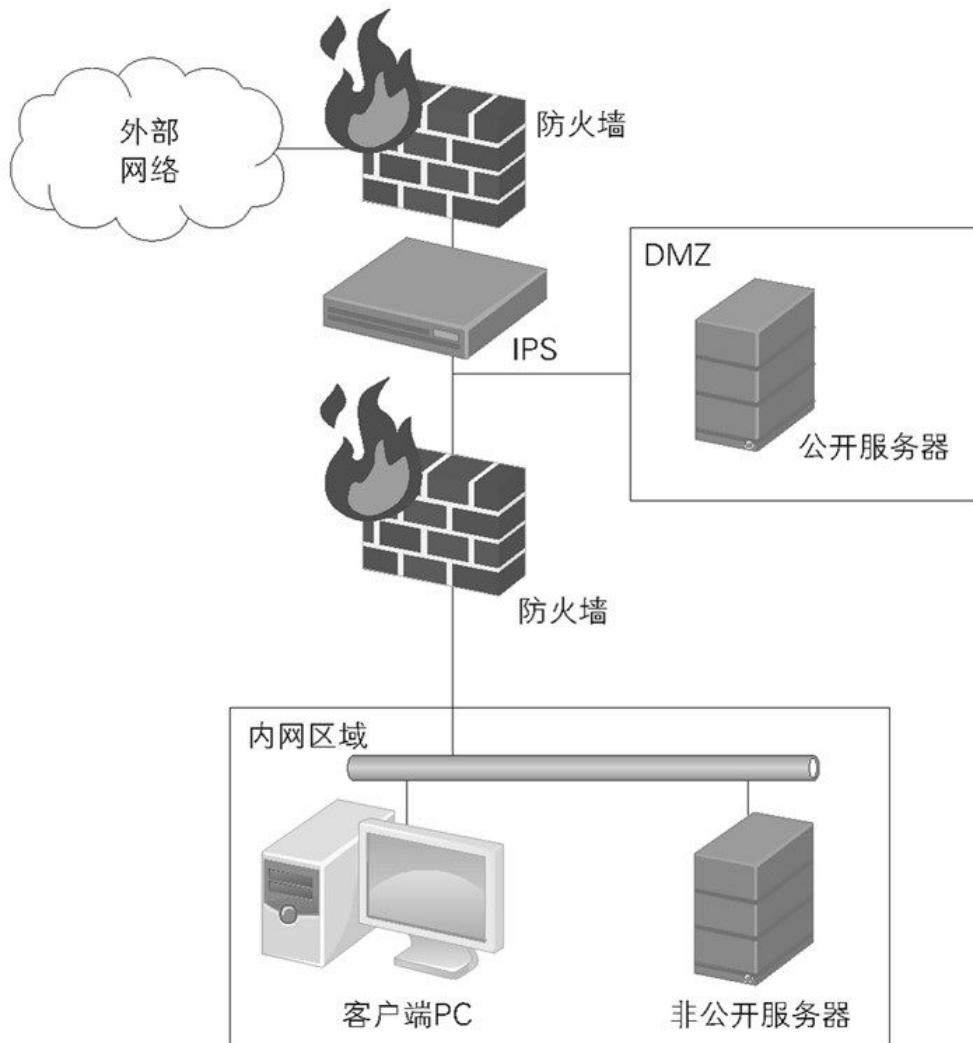


图 3.7 DMZ + IPS 模式

各模式的比较结果与选择标准

我们将各模式的比较结果与选择标准分别总结在表 3.5 和表 3.6 中。

表 3.5 非法访问应对策略的设计方式的比较

比较项目	防火墙	单防 火墙 DMZ	双防 火墙 DMZ	DMZ + IDS	DMZ + IPS
非法入侵	x: 低 一旦公开	○: 高		即使公开服务器被入侵，内网的非公开服务器被攻击的风险也很	

的防御力	服务器被入侵，内网的非公开服务器被攻击的风险很高	低	
对伪装成正常通信的攻击的适应性	<p>△： 低 不具备检测和防御伪装成正常通信的攻击的能力</p>	<p>○： 高 可以检测伪装成正常通信的攻击，并向管理员发出警报</p>	<p>◎： 非常高 可以检测伪装成正常通信的攻击，并向管理员发出警报，且自动阻止通信</p>
运维性	<p>◎： 容易 根据供应商提供的防火墙信息来更新防火墙策略与修补漏洞</p>	<p>△： 困难 除了防火墙模式、单防火墙DMZ模式、双防火墙DMZ模式的运维特点以外，为了防止误报，还需要持续对特征和描述进行调优</p>	<p>×： 非常困难 除了防火墙模式、单防火墙DMZ模式、双防火墙DMZ模式的运维特点以外，为了防止误报，还需要持续对特征和描述进行调优。此外，还需要准备相应的体制以应对因误报而导致的通信阻断</p>

表 3.6 非法访问应对策略的设计方的选择标准

模式	构建成本	信息的机密程度	互联网连接
防火墙	低	低 适用于处理非保密信息的情况	无 适用于不连接互联网的情况
单防火墙DMZ		中 适用于处理较保密的信息的情况	有 适用与互联网连接的情况
双防火墙DMZ	中		
DMZ + IDS	高	高 适用于处理高保密性的信息的情况	

DMZ + IPS	非常高	非常高 适用于绝对机密的信息的情况
-----------	-----	----------------------

注意事项

防火墙和 IDS 等无法防止 SQL 注入和 XSS 等针对 Web 应用程序的非法访问。要想防止这些攻击，必须针对 Web 应用程序的安全漏洞制定安全策略。可以考虑引入 Web 应用程序防火墙来确保系统具有更高的安全性。

3.3 身份认证的设计方式

近年来，发生了许多某些人通过钓鱼网站和间谍软件非法盗取互联服务的用户账号和密码的事件。如果账号被盗，不仅这些人可以非法使用这些互联网服务，可能还会导致姓名、地址等个人信息、保密信息泄露。要想应对这些风险，除了用户自身需要加强注意以外，系统服务的提供者也需要加强对用户身份的认证。

生物认证模式

生物认证指的是使用指纹和虹膜等人体特征、步行方式和习性等动作特征来进行身份认证（图 3.8）。由于这些人体生物特征长期不易改变，且他人难以复制和模仿，可以确保非常高的安全性。

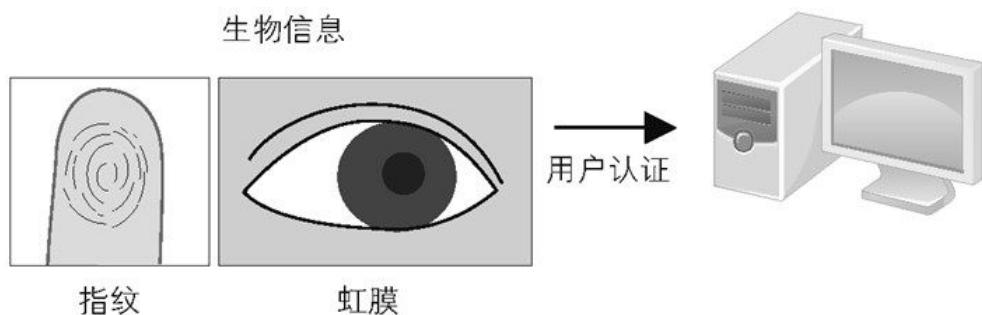


图 3.8 生物认证模式

一次性密码模式

通过安全令牌（以下简称“令牌”）定期生成不同的密码来进行身份认证（图 3.9）。由于认证时使用的密码到期就会失效，即使密码在某个时间点被窃取了，也难以用于非法行为，可以确保较高的安全性。但是由于令牌也存在被窃取和非法使用的风险，因此该模式的安全强度也取决于用户保存令牌的态度。

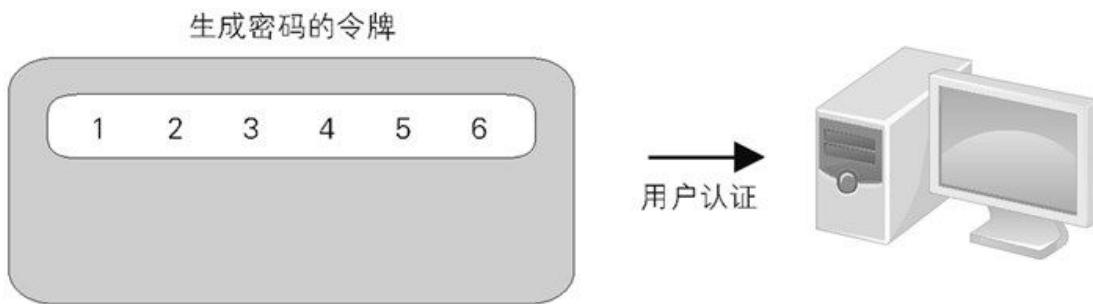


图 3.9 一次性密码模式

IC 卡 + 密码模式

通过在终端读取 IC 卡，并要求用户输入密码来进行身份认证（图 3.10）。该模式被日本银行协会作为基本认证方式应用于日本的 ATM 机中。由于不是在读取 IC 卡的终端上进行认证，而是通过网络将 IC 卡中保存的用户信息和密码发送到认证服务器上进行核对，因此可以有效防止伪造 IC 卡。同样，该模式的安全强度也取决于用户保存 IC 卡的态度。

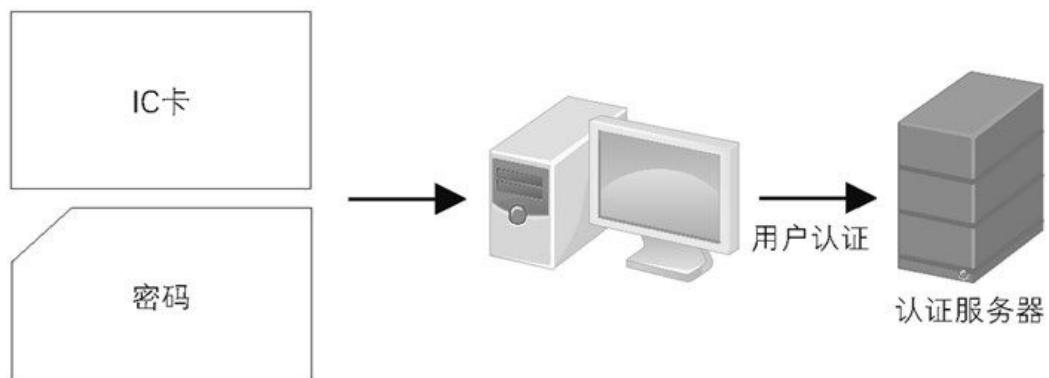


图 3.10 IC 卡 + 密码模式

IC 卡 / 令牌模式

通过在终端读取 IC 卡和 USB 令牌来进行身份认证（图 3.11）。由于该模式将 IC 卡和 USB 令牌的持有者视为合法用户，因此当 IC 卡和安全令牌被借出或是遗失时可能会导致安全问题。因此，该模式的安全强度也取决于用户保存 IC 卡和 USB 令牌的态度。此外，由于该模式仅通过 IC 卡和 USB 令牌进行认证，因此相比于 IC 卡 + 密码模式，安全强度较低。如果需要保护高保密性的信息，应当选择其他模式。

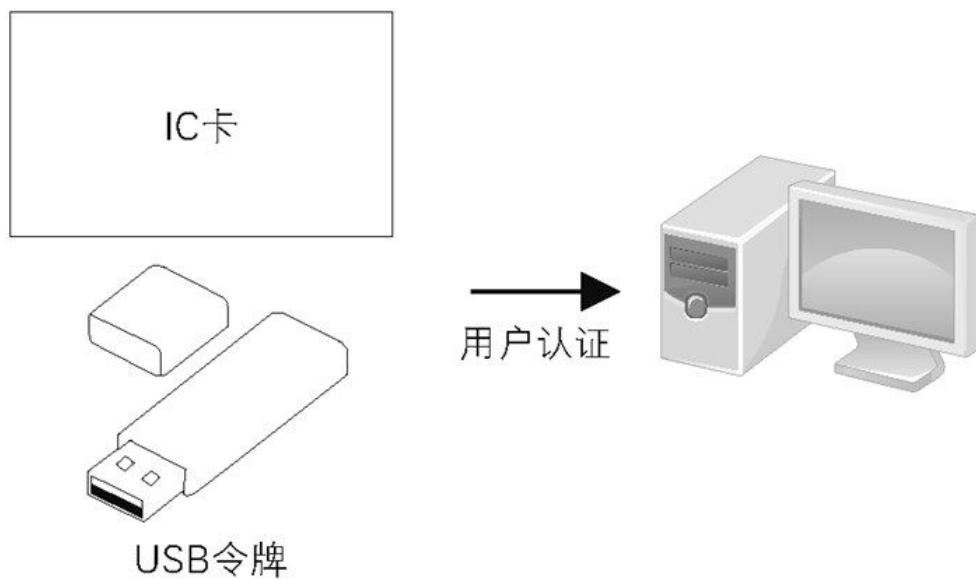


图 3.11 IC 卡 / 令牌模式

ID 密码模式

通过核对分配给用户的 ID 和用户自己设定的密码来进行身份认证（图 3.12）。如果用户因担心遗忘 ID 和密码而将它们写在浮签上并贴在终端上，或是与他人共享自己的认证信息则会导致安全问题。同样，该模式的安全强度也取决于用户的态度。如果需要保护高保密性的信息，应当选择其他模式。

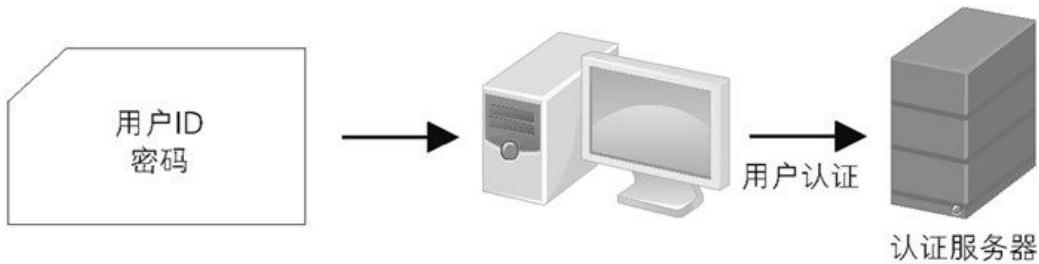


图 3.12 ID 密码模式

各模式的比较结果与选择标准

我们将各模式的比较结果与选择标准分别总结在表 3.7 和表 3.8 中。

表 3.7 身份认证方式的设计方式的比较结果

比较项目	生物认证	一次性密码	IC卡 + 密码	IC卡/令牌	ID密码
安全强度	◎：强 ID信息的伪造和被盗的风险很低	◎：强 与普通的密码认证相比，密码被盗的风险很低	◎：强 因为是在服务器上管理认证信息，所以比较安全	○：一般 伪造ID信息风险很低，但是密码被盗的风险较高	△：弱 可能因窃听导致ID信息泄露
认证要素	指纹等生物信息	用户ID 一次性密码	IC卡 密码 (个人识别码)	IC卡 USB令牌	用户ID 密码 (个人识别码)
认证所必要的设备	生物信息感应器	令牌	IC卡读卡器		无

表 3.8 身份认证方式的设计方式的选择标准

模式	信息的保密性	认证的严密性	适用系统
生物	高	非常高	适用需要处理个人信息等高保密性信息的系统

认证			
一次 性密 码	高	适用于需要在一定程度上完全管理用户ID/密码，而且需要处理个人信息等高保密性信息的系统	
IC卡 +密 码		适用于需要完全管理用户ID/密码，而且需要处理个人信息等高保密性信息的系统	
IC卡 / 令牌	中	中	适用于需要管理用户ID，而且存在需要保密的信息的系统
ID密 码		低	适用于需要完全管理用户ID/密码，而且被保护的信息不属于高保密性信息的系统

注意点

制定认证服务器的可靠性策略

由于认证服务器是 SPOF，因此存在所有需要认证的系统都无法使用认证服务器的风险，因此必须事先针对认证服务器制定可用性策略。

引入备用认证方式

近年来，虽然生物认证的识别率在逐渐升高，但是依然存在识别错误和运行不正常等情况。此外，用户的 IC 卡和令牌也可能会遗失。为了防止这些风险导致不能使用认证系统，应当考虑引入备用认证方式。例如即使没有 IC 卡也可以使用责任人认可的特定 ID 登录系统的认证方式。

3.4 ID 管理和维护的设计方式

如果用户共用系统的 ID，就无法对用户单独设定访问权限，无法知道“谁”“在什么时候”“访问了什么资源”。为了确保系统的安全性，实现企业内部的统一管理，不应当让用户共用 ID，而是对每个用户都分配一个设定了相应权限的 ID。

本节将会介绍 ID 管理和维护方式的设计模式。

单点登录模式

将需要认证才能使用的多个系统的 ID 统一为一个，只需认证一次即可访问所有的系统（图 2.13）。用户只需要记住一组 ID 与密码的组合即可，使用上方便很多。不仅如此，用户不必再将 ID 和密码记录在浮签上，可以有效降低安全风险。

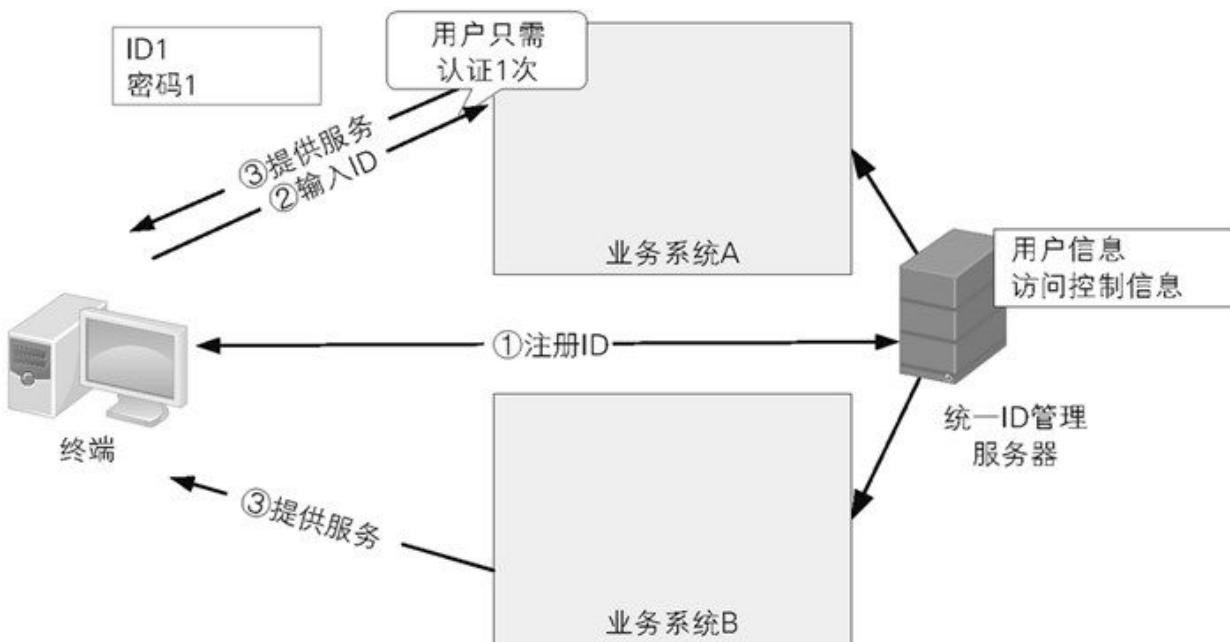


图 3.13 单点登录模式

统一 ID 的个别登录模式

将需要认证才能使用的多个系统的 ID 统一为一个，并使用这个 ID 来访问各个系统（图 3.14）。用户只需要记住一组 ID 与密码的组合即可，使用上方便很多，但是访问每个系统时都需要单独登录一次。

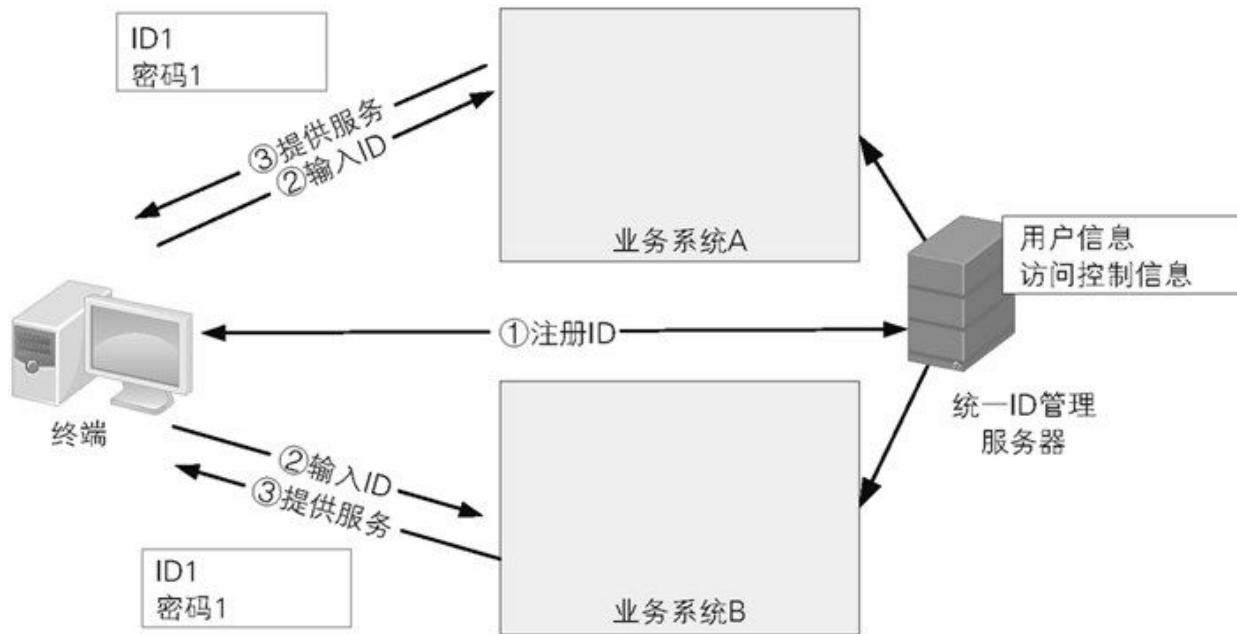


图 3.14 统一 ID 的个别登录模式

个别 ID 的个别登录模式

对每个需要认证才能使用的系统分别设定和注册 ID 和密码（图 3.15）。用户需要管理多个 ID 和密码，并且访问每个系统都需要登录一次。

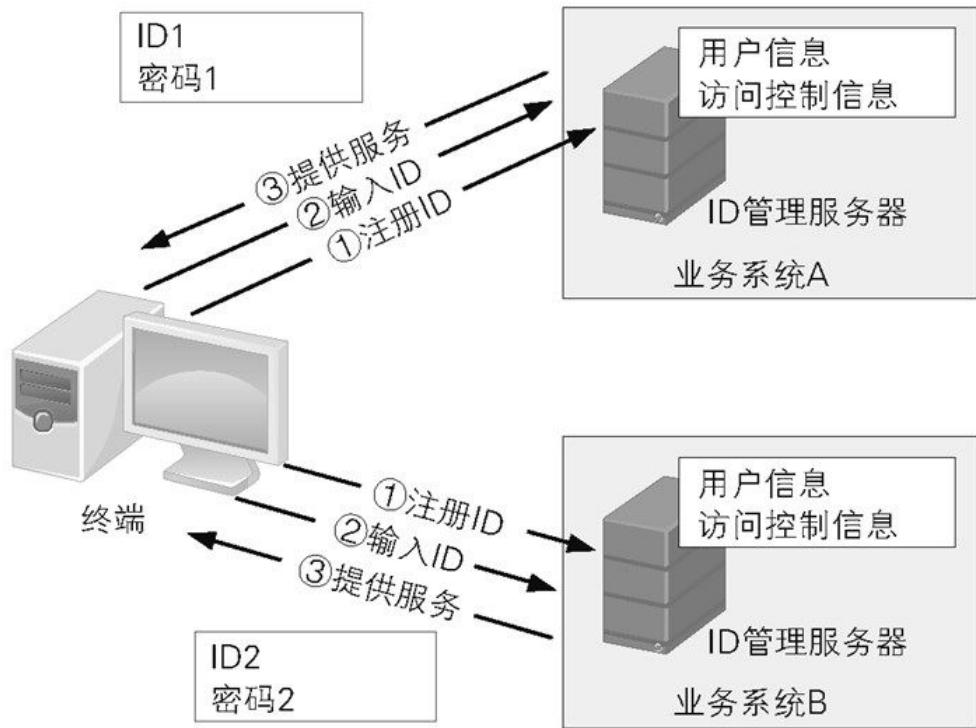


图 3.15 个别 ID 个别登录模式

各模式的比较结果与选择标准

我们将各模式的比较结果与选择标准分别总结在表 3.9 和表 3.10 中。

表 3.9 ID 管理和维护的设计方式的比较

比较项目	单点登录	统一ID个别登录	个别ID个别登录
用户方便程度	○: 高 不需要分开管理多个系统的ID, 一次登录即可访问所有系统	△: 一般 不需要分开管理多个系统的ID, 但访问每个系统都需要单独登录	×: 低 需要将多个系统的ID分开管理, 并且访问每个系统都需要单独登录
系统管理员的ID管理难易度	○: 容易 当发生人事变动时, 可以很容易地增加、变更、删除ID		×: 烦杂 当发生人事变动时, 需要对所有系统都增加、变更、删除ID

表 3.10 ID 管理和维护的设计方式的选择标准

模式	用户登录的频率	人事变动的频率
单点登录	多 适用于需要登录的系统多，想要减少登录次数的情况	高 适用于人事变动多，用户的属性频繁变更的情况
统一ID个别登录	少	
个别ID个别登录	适用于不需要频繁登录系统或是需要登录的系统少的情况	低 适用于人事变动少，用户的属性变更少的情况

注意点

没有特别需要注意的地方。

3.5 信息泄露应对策略的设计方式

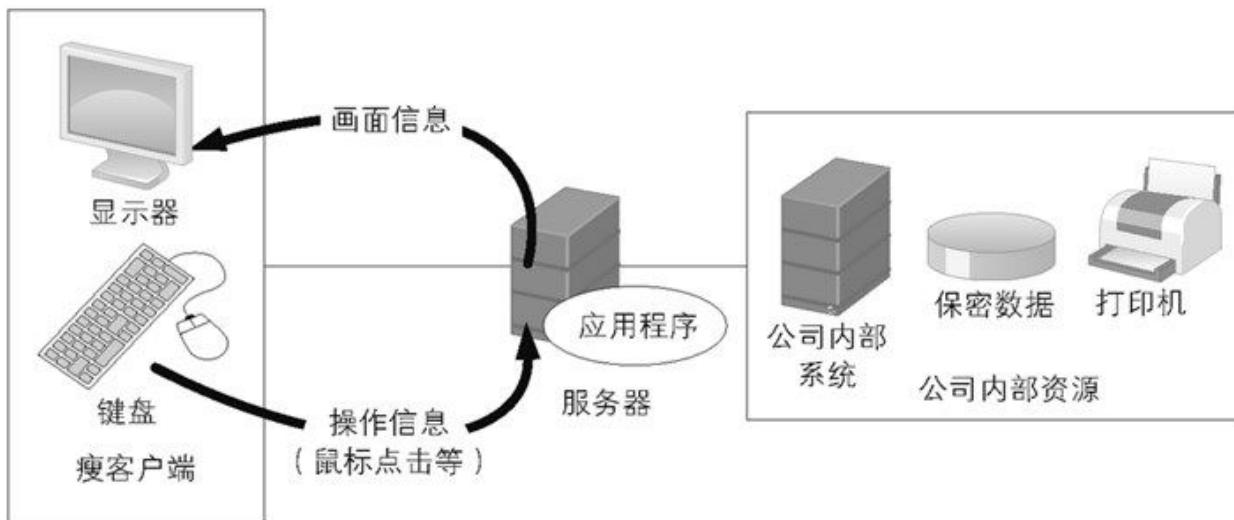
现在许多系统中都保存着用户的个人信息和产品的生产方法等保密信息。如果这些信息泄露出去，会导致企业丧失信息资产和社会信誉，直接影响到企业存亡。

因此，针对防止信息泄露制定相应的策略和管理规范，对企业来说是非常重要的企业活动。在制定防止 IT 系统信息泄露的策略时，主要从减少和防止企业内部人员将数据非法带出企业（防止带出）和防止外部人员进入企业获取信息（防止获取）两个方面来考虑采取必要的技术手段提高安全性。

瘦客户端模式

在业务中使用不带磁盘的瘦客户端（图 3.16）。瘦客户端只带有基本的输入输出设备（键盘、鼠标、显示器），所有的业务数据的处理，都是通过网络连接到服务器上集中进行的。服务器基于用户的输入信息进行相应的处理，并将处理结果反馈到工作站的显示器上，用户感

觉不到与使用普通 PC 进行业务处理有所不同。由于数据不会保留客户端上，因此降低了从客户端窃取信息的风险。



瘦客户端模式不仅只有图中展示的共享 OS 类型，还有为每个用户都提供 OS 并在工作站进行业务处理的网络启动（Network Boot）类型

图 3.16 瘦客户端模式

外存管理模式

在业务终端引入管理软件来管理与业务终端接口（USB 端口等）相连的外存设备（闪存、外接硬盘等）（图 3.17）。由于需要在每台业务终端上都安装管理软件，并设置“可读 / 可写”“可读 / 不可写”“不可读 / 不可写”等权限规则，所以如果终端数量太多，会加重运维负担，还可能导致部分终端漏装管理软件或者权限规则设置错误等问题。

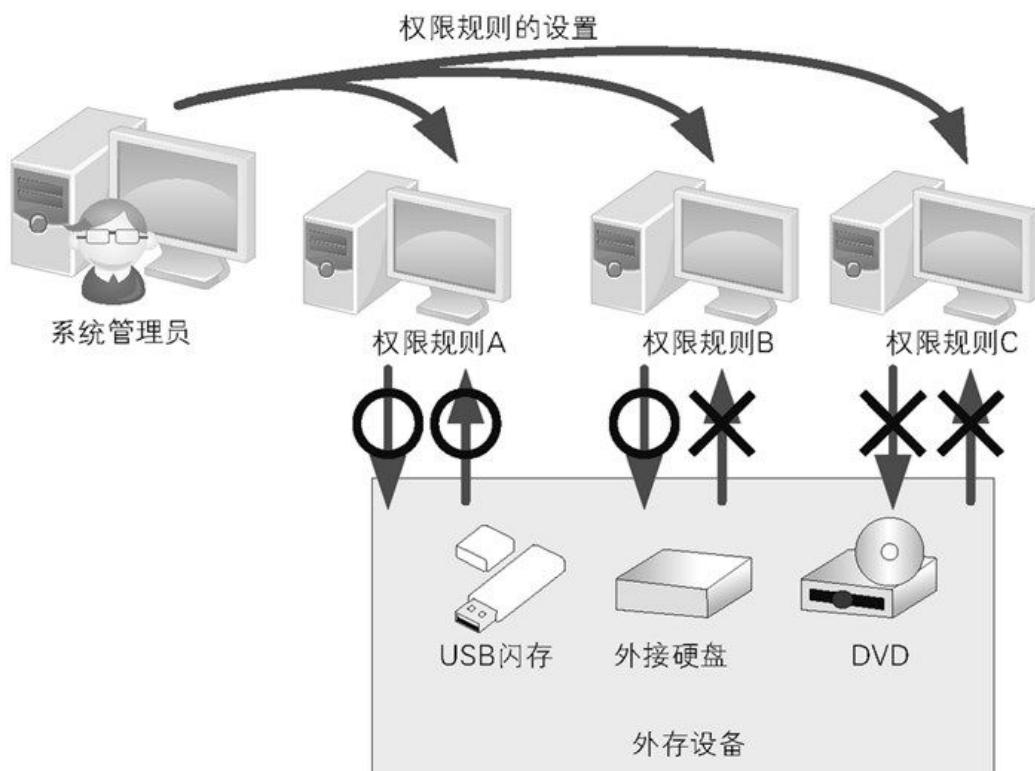


图 3.17 外存管理模式

数据加密模式

通过专用加密软件和 OS 的附加功能对需要保存的数据进行加密（图 3.18）。加密后的数据无法直接读取，只能通过加密时生成的解密密钥解密后才能读取。

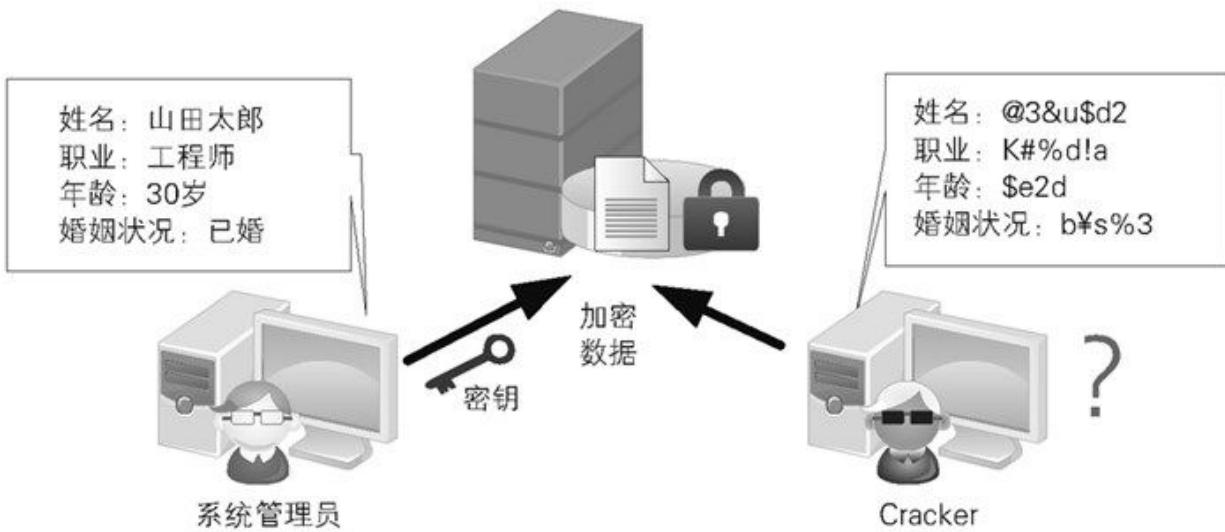


图 3.18 数据加密模式

数据分割模式

将数据分割，并分别保存在 USB 闪存和 DVD 等移动存储媒介中（图 3.19）。这样，即使其中一部分存储媒介遗失或者被盗，也无法根据这部分数据还原出完整的数据。因此，在保存不再更新的数据或是进行数据转移时可以有效地防止信息泄露。

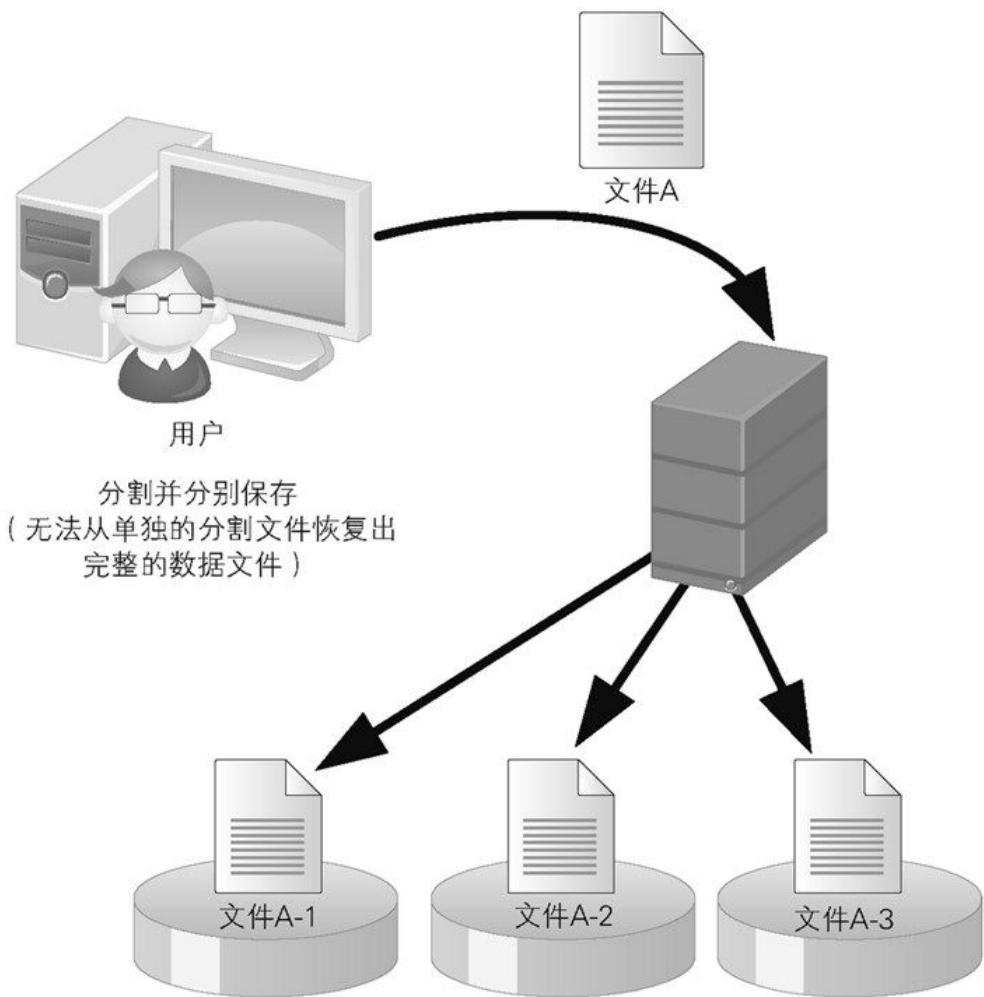


图 3.19 数据分割模式

通信加密模式

通过对通信数据进行加密，可以防止因窃听等导致的信息泄露（图 3.20）。在进行保密数据通信时，可考虑采用该模式。

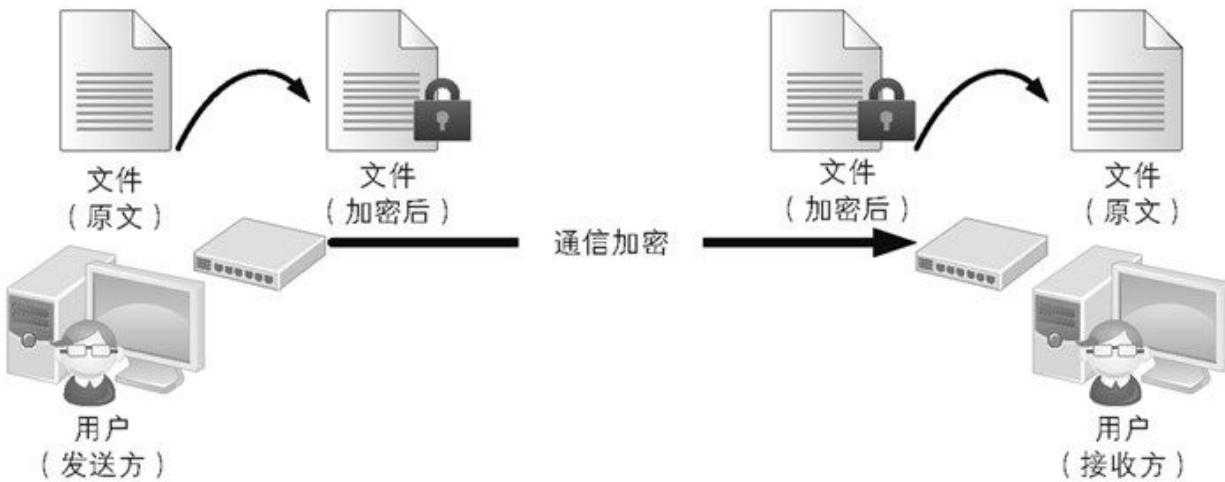


图 3.20 通信数据加密模式

各模式的比较结果与选择标准

我们将各模式的比较结果总结在表 3.11 中，将“防止带出”的选择标准和“防止获取”的选择标准分别总结在表 3.12 和表 3.13 中。

表 3.11 信息泄露应对策略的设计方式的比较结果

比较项目	防止带出技术		防止获取技术		
	瘦客户端	外存管理	数据加密	数据分割	通信加密
对象设备	办公室的业务终端	办公室的业务终端	磁盘	USB闪存和DVD等可移动存储媒介	通信线路中的路由器和交换机
实现方法	引入瘦客户端系统	在业务终端中安装专用的外存管理软件	使用专用的加密软件和OS附带功能	引入专用的数据分割软件	通过路由器和交换机的附带功能对通信数据包进行加密

表 3.12 信息泄露应对策略的设计方式的选择标准（防止带出技术）

模式	所处理信息的保密性	运维的负荷	业务终端数	适用系统
瘦客户端	高 适用于处理个人信息等高保密信息	低	多	适用于终端数量多，运维工作量大的系统
外存管理	中 如需处理个人信息等高保密信息，推荐与其他模式组合使用	高	少	适用于能够逐台管理终端，并对每台终端设定相应权限规则的系统

表 3.13 信息泄露应对策略的设计方式的选择标准（防止获取技术）

模式	信息泄露应对策略的对象设备
数据加密	磁盘 适用于需要在磁盘上保存个人信息等高保密信息的情况
数据分割	USB闪存和DVD 适用于需要通过USB闪存和DVD移动个人信息等高保密信息的情况
通信加密	通信路径（通信路径中的交换机等） 适用于需要通过网络传输个人信息等高保密信息的情况

注意点

本节介绍的具有逻辑性和系统性的信息泄露应对策略实现成本比较高，因此，还可以一并考虑以下这些物理策略，寻求更高的性价比。

- 使用安全链给 PC 加锁
- PC 的 USB 端口加锁、网络设备端口加锁
- 设置特定的安全办公区域，并进行严格的出入管理

3.6 总结

本节讲解了安全性策略的基本思考方法和设计方式。为了降低和排除企业活动和信息资产的安全威胁，需要实施全面、立体的安全性策略。此外，安全技术的发展日新月异，安全风险也在随之不断变化，随时可能有新的安全风险出现，因此，应当定期重新审视安全策略。

第 4 章 性能与可扩展性需求的实现策略：防止系统性能下降

性能与可扩展性需求是指与系统当前的处理能力相关的需求，以及与该处理能力将来能够轻松提升到何种程度相关的需求。如果不考虑性能需求就着手开发系统，会导致系统投入运行后发生性能不足或性能浪费等问题，或者是无法弹性地应对将来可能发生的性能不足问题。更有甚者会导致系统无响应和页面显示时间过长。为了避免这些情况的发生，必须在进行系统开发之前制定性能与可扩展性策略。

在本章中，我们将讲解性能与可扩展性策略的基本思考方法和设计方式。

4.1 性能与可扩展性策略的基础

性能与可扩展性策略的基础是“规模调整”。规模调整是指整理系统的性能需求，并确定合适的硬件及其数量来实现需求。即使是已经投入运行的系统，当其发生性能下降的情况时，通过增加硬件来进行应对也属于规模调整。在进行规模调整时，不能只考虑服务器、存储设备、网络设备等单一系统构成要素，还要站在系统整体的角度，找出所有对性能有影响的位置（性能瓶颈）。

规模调整的重要性

如果不确定系统规模就着手开发系统，会导致系统投入运行后发生性能不足或性能浪费等问题（图 4.1）。

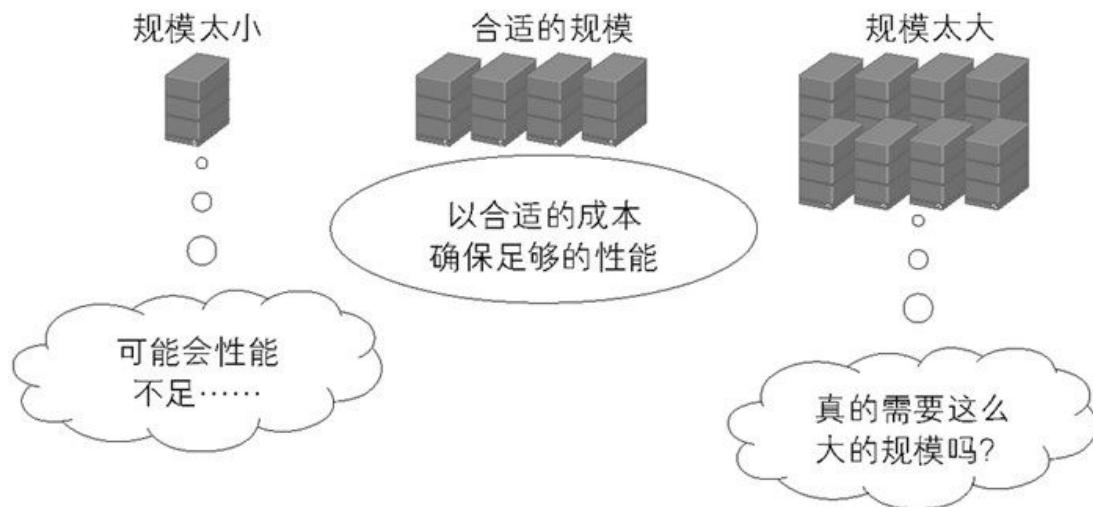


图 4.1 规模调整的重要性

如果系统性能不足，就无法满足“在多少秒以内完成响应”“夜间批处理程序在早上几点之前全部执行完毕”等与系统处理速度相关的非功能性需求。而如果是在测试阶段和投入运行后才发现有很多性能不足的问题，就需要再回到上游设计阶段重新调整和设计，导致整个项目进度延期和成本增加。

反之，如果性能浪费，虽然不会有前述的性能不足问题，但是浪费在设备上的投资导致成本增加。引入 IT 系统的目的就是为了能节省成

本，如果系统构建成本过高，其结果是可能达不到当初预期的效果。

规模调整的可扩展性

根据性能需求调整规模的时候，还需要同时考虑系统的可扩展性。例如，如果预计未来每年业务量都会增加，将来系统的性能状况可能会无法满足届时的性能需求，那么就需要事先进行可扩展性设计，使系统可通过增加部分设备，以最小的成本来应对这种变化。如果没有充分考虑可扩展性设计，当这种情况发生时，有可能需要花费巨额成本来修改整个系统。因此，在调整规模的时候，必须同时考虑可扩展性设计（图 4.2）。

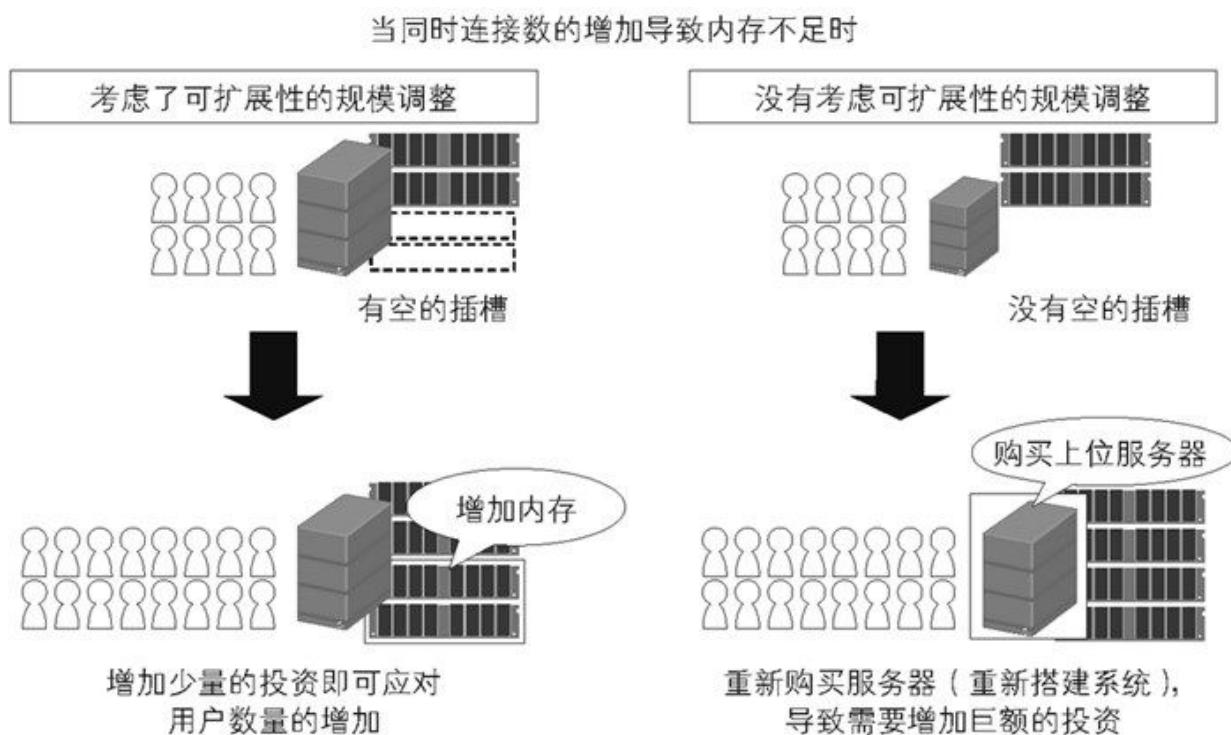


图 4.2 可扩展性设计的重要性

规模调整的步骤

通常情况下，规模调整的步骤如图 4.3 所示。

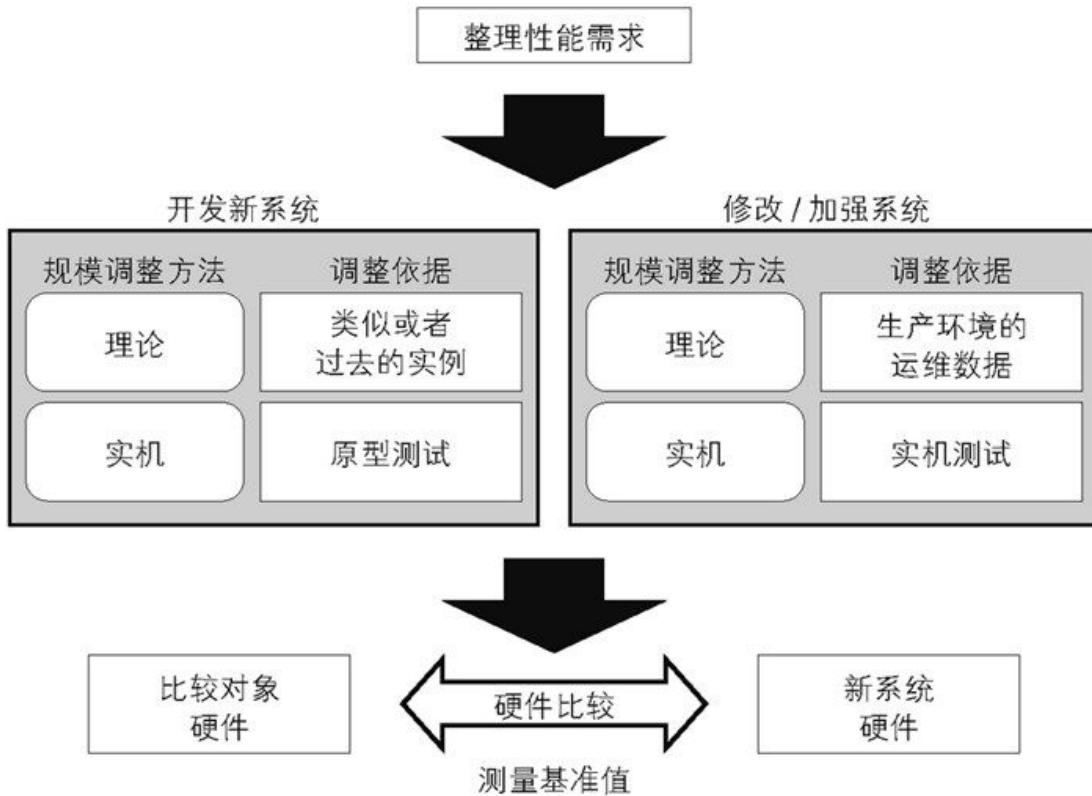


图 4.3 规模调整的步骤

① 整理性能需求

首先要整理性能需求。如果性能需求不明确，需要基于业务特点（用户数量、高峰期负荷倾向等），参考非功能性需求等级讨论性能需求。业务特点和性能需求的典型示例请参见表 4.1。

表 4.1 业务特点和性能需求示例

种类	示例
业务特点	用户数量、高峰期负荷倾向、业务量增加比率、提供服务时间（365天24小时等）

种类	示例
在线处理系统的性能需求	访问数/秒、同时连接数、允许的响应时间
批处理系统的性能需求	待处理的数据量、处理时间、并行情况

② 理论调整

在理论层面讨论实现性能需求的硬件构成，本书中称为“理论调整”。

如果是开发新系统，可以参考类似系统案例，结合应用程序、业务特点、环境的差异来进行理论调整。如果没有可以参考的案例，可依据基准值来进行理论调整。我们将在“④硬件的比较和选择”中说明其中需要注意的问题。由于理论调整很难实现精确的模拟，因此确定安全率¹就显得尤其重要。

¹ 又称为安全系数。为了防止性能不足，需要与在理论调整中算出的性能值相乘的数值。需要注意的是，安全率过高会导致性能浪费。

如果是对现有的系统进行修改和加强，也可以参考类似系统案例和基准值来进行理论调整。此外，一般系统修改都都会沿用大部分以前的业务内容，因此可以通过分析现有系统开发时的性能测试结果和运维中收集的相关数据（现在的业务量和各硬件的资源使用率等）来提高理论调整的精度。

在分析资源使用率的时候，不应当仅着眼于各硬件的使用率，还要以业务为单位分析资源使用率。如果一台服务器处理多个业务而又不了解每个业务的资源使用率，就无法进行可扩展性设计中的规模调整

(图 4.4)。这种情况下，可通过回归分析和差异分析来提取各业务的资源使用率。在下面的实机调整阶段则会进行更详细的分析。

CPU 使用率：90%

业务A：？%
业务B：？%
业务C：？%

虽然知道服务器的 CPU 的使用率为 90%，
但是不知道业务 A 的 CPU 使用率具体是多少

图 4.4 服务器 L 处理多个业务时的资源使用率

③ 实机调整

除了需要进行理论调整外，还需要在实际硬件上模拟业务，讨论可实现性能需求的硬件构成。在本书中，我们称之为“实机调整”。

如果是新开发系统，通常是事先准备应用程序的原型，然后在测试服务器上进行性能测试。这种情况下，原型与实际生产环境中运行的应用程序的差异在很大程度上决定了调整的精度。因此，随着开发阶段的深入进行，为每个阶段都准备一套接近真实规格的原型程序并反复进行性能测试，对于提高调整精度是非常重要的。

如果是对现有的系统进行修改和加强，且可以在生产环境系统上进行测试，那么就直接在现有系统上进行性能测试。和理论调整一样，先分析业务的混合比率和资源使用率，然后依据分析结果进行调整。在分析资源使用率的时候，去除低负荷和高负荷的部分，只针对负荷增量比较稳定的部分分析业务和资源的关系，就可提高调整精度（图 4.5）。

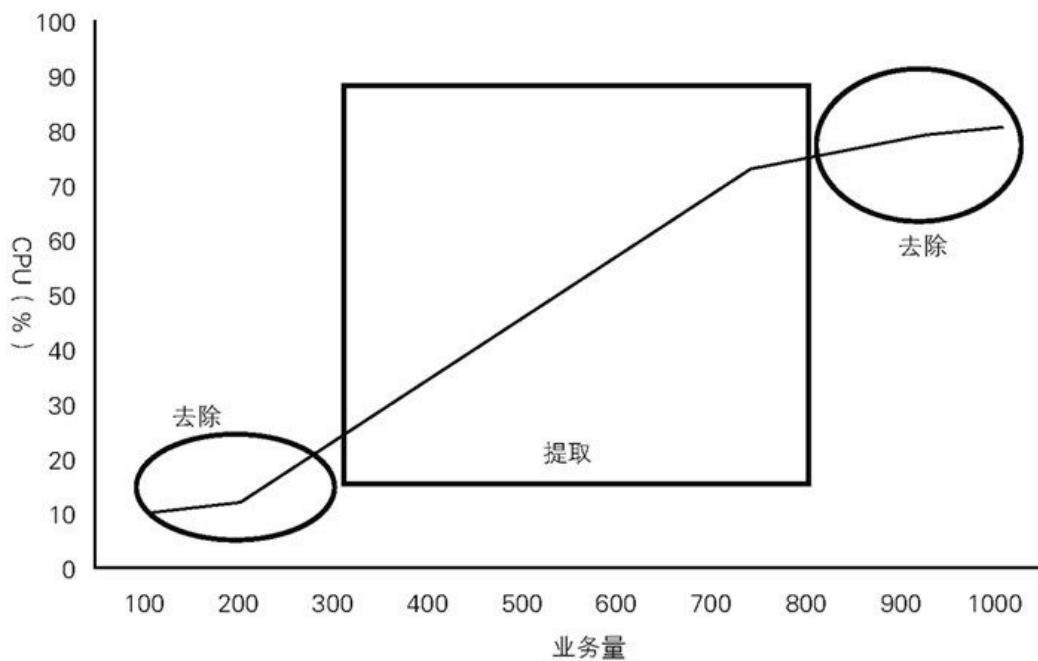


图 4.5 实机调整中的资源使用率分析

如果无法在生产环境系统上进行测试，那么就在测试环境进行性能测试。

④ 硬件的比较和选择

在通过理论调整和实机调整进行了性能测试，推定出性能值后，就可以着手进行硬件的比较和选择了。通常情况下都是依据基准值来比较硬件的性能。

基准值依据其制定者可以大致分为 3 类（表 4.2）。可以通过比较多个基准值来提高调整精度。

表 4.2 不同制定者制定的基准值的优缺点

基准值的制定者	优点	缺点
独立的业界团体	可以横向比较不同硬件供应商的硬件产品	制定基准值时用来参考的硬件数量经常会偏向某个硬件供应商
软件供应商	如果使用了软件供应商提供的产品，则该基准值具有高可靠性。此外，还可以比较不同硬件供应商的硬件产品	由于基准值是在使用软件供应商提供的产品的前提下制定的，如果要使用的软件产品还没有确定，或是公司自行开发软件，那么很难以基准值作为参考
硬件供应商	可以方便地比较同一硬件供应商提供的不同硬件产品	很难比较不同硬件产品供应商的硬件

基准值除了以单个硬件的处理性能作为评测对象以外，还可以以系统整体的处理性能作为评测对象。需要注意的是，参考系统整体处理性能的基准值是有前提条件的（业务特点、系统构成、性能瓶颈）。例如，如果没有理解随着业务量的增加，存储设备的处理能力也需要提高，而仅仅增加了存储设备的容量，就会导致存储设备的处理能力变成整个系统的性能瓶颈，引发新的性能问题。

此外，在实际系统中，不应当局限于依据基准值简单地进行扩展。特别是在多 CPU 系统中，还要参考实机测试结果和以往的类似事例。

在与现有系统进行比较时，事先将过去的数据换算为现在的数据并准备基准值。最好是使用与现有系统的 CPU 结构相同，且处理器数量、核心数、线程数、时钟周期尽可能一致的基准值。如果是 CPU 数和时钟频率不同，则需要通过 CPU 数或是通过时钟频率进行换算，这时容易产生偏差，需要特别注意。

至此，我们讲解了性能与可扩展性应对策略的基本思考方法。接下来我们讲解可扩展性的设计方式和与系统超负荷运行（超过了规模调整时预想的处理量）的应对策略相关的设计方式。

4.2 可扩展性策略的设计方式

在系统开发中，依据预想的业务量计算系统所需要的性能，然后据此选择和购买硬件。但是，随着业务范围的扩大，业务量和处理量都会增加，可能会导致系统性能不足。如果重新开发系统去解决这个问题，成本、时间和人力资源消耗很大，效率很低。为了高效地应对性能不足的问题，需要充分考虑系统的可扩展性设计。

扩容模式

已经确定性能瓶颈的原因是 CPU 或内存等硬件组件（部件）的时候，可以通过更换或是增加该部件来提升性能（图 4.6）。但是在增加 CPU 或内存时，需要注意服务器中是否还有空的插口和插槽。

如果每次性能不足时，都通过增加 CPU 和内存等组件来应对，最后的成本可能比新购入一台服务器更昂贵。因此，该模式适用于不需要性能提升太高，且扩展后不会再有大幅提升性能的需求的情况。

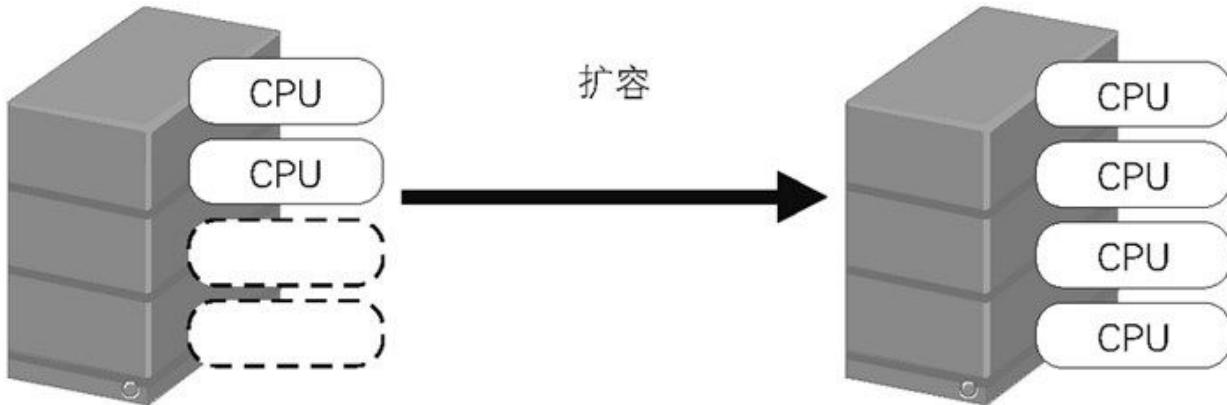


图 4.6 扩容模式

升级模式

该模式并不升级 CPU 和内存等组件，而是升级服务器（图 4.7）。选择处理能力更强的型号的服务器，可以大幅度提升系统性能。一般而言，服务器升级的同时，可扩展性也会提高，因此该模式适用于用户数增加等导致需要提升性能的情况。同时，也适用于没有空余的插座和插槽来扩容或是因 EOS (End Of Sales, 断货)² 等原因无法扩容的情况。但是，由于升级后硬件环境发生了变化，可能会导致系统出现问题，因此必须谨慎地进行系统升级。

² 指的是销售结束日期。原则上，供应商在销售结束日期后不会再销售该产品。

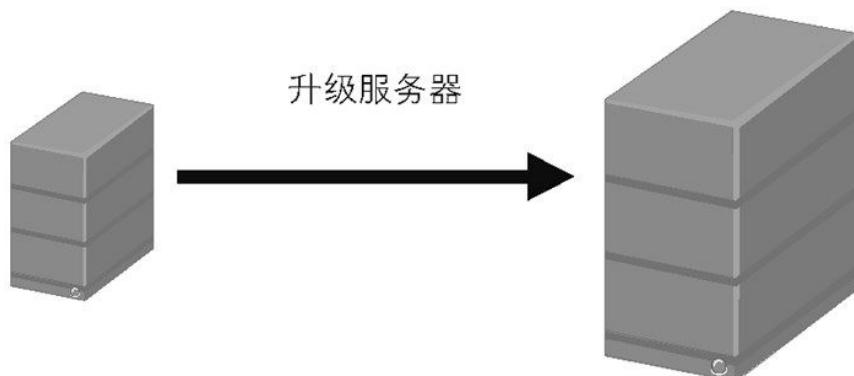


图 4.7 升级模式

功能分割模式

按照业务将 Web/AP 服务器分开，减轻每台服务的处理负荷来确保高性能（图 4.8）。将导致性能瓶颈的功能和处理转移到其他服务器上，确保每个功能和处理的性能。

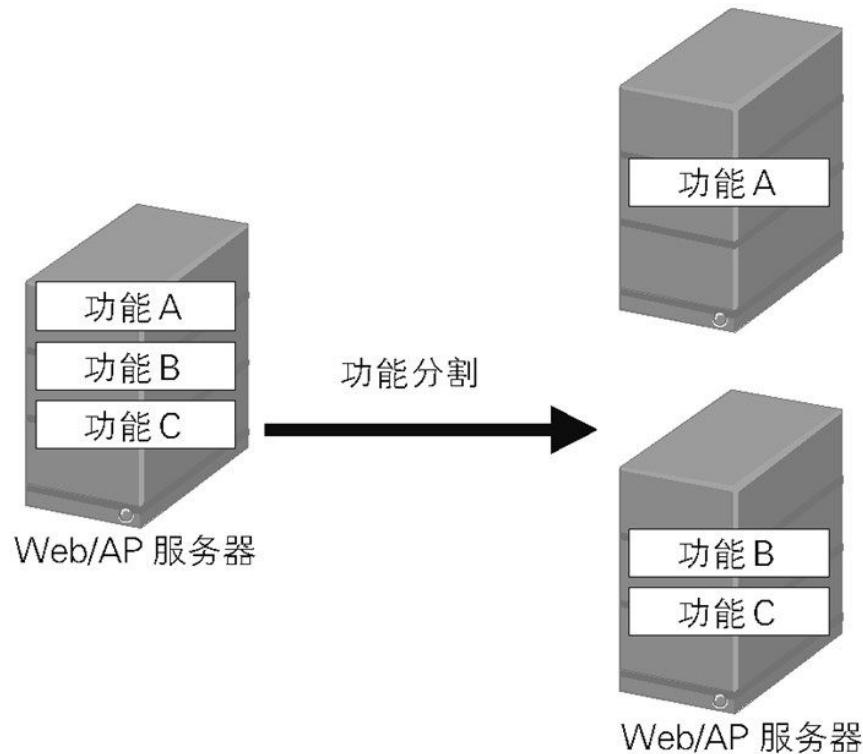


图 4.8 功能分割模式

集群模式

通过负载均衡器将向系统请求的事务分配给多台 Web/AP 服务器上来确保高性能（图 4.9）。有多种分散方式，既有将事务按顺序分配给 Web/AP 服务器的“轮询调度方式”，也有将事务分配给正在处理的事务数量最少的 Web/AP 服务器的“最小连接方式”。此外，负载均衡器还可以检测服务器故障，当服务器故障后，不会再将事务分配给它处理，提高了系统的可用性。

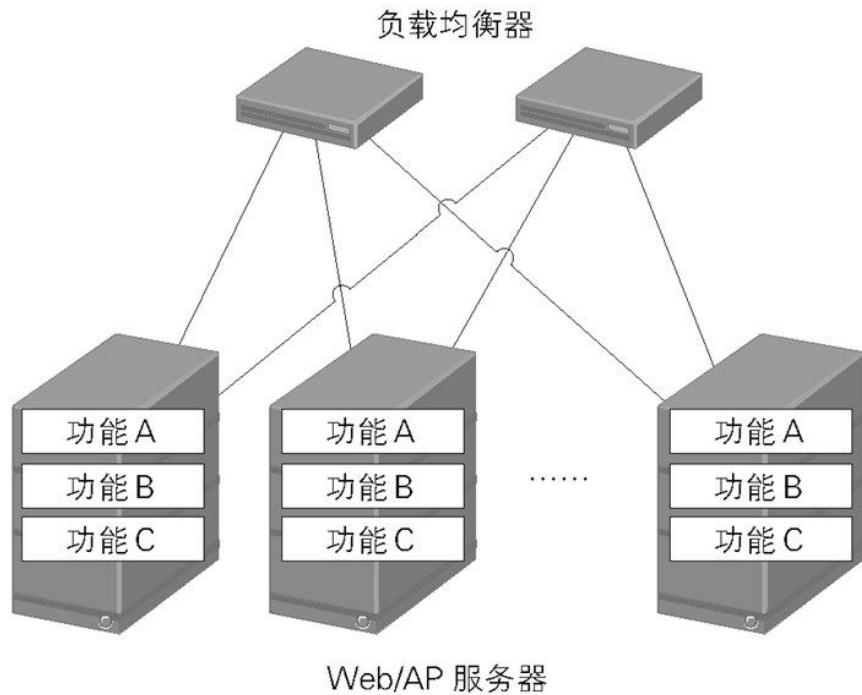


图 4.9 集群模式

无共享模式

多个 DB 服务器之间并不共享磁盘、会话和其他资源，而是互相独立（图 4.10）。性能与服务器数量成正比。因此，与下面要介绍的共享磁盘模式相比，该模式容易确保可扩展性，但同时需要考虑 DB 数据分割的设计。

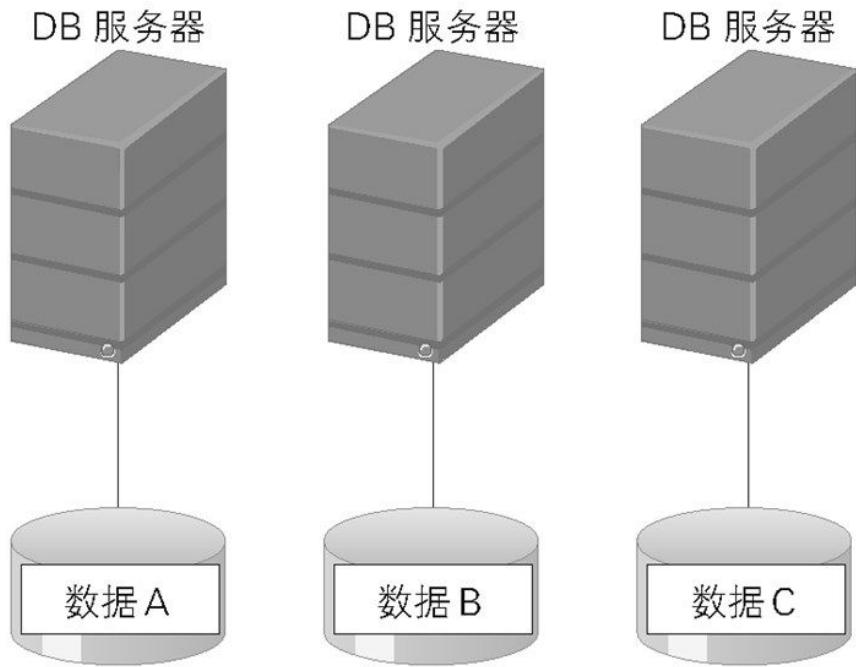


图 4.10 无共享模式

共享磁盘模式

多个 DB 服务器之间共享磁盘（图 4.11）。由于 DB 数据没有被分割保存，可以通过增加 DB 服务器数量来提高性能。但是，由于 DB 冲突时的处理和共享磁盘的 IO 会产生额外系统开销，所以与无共享模式相比，该模式的可扩展性较差。

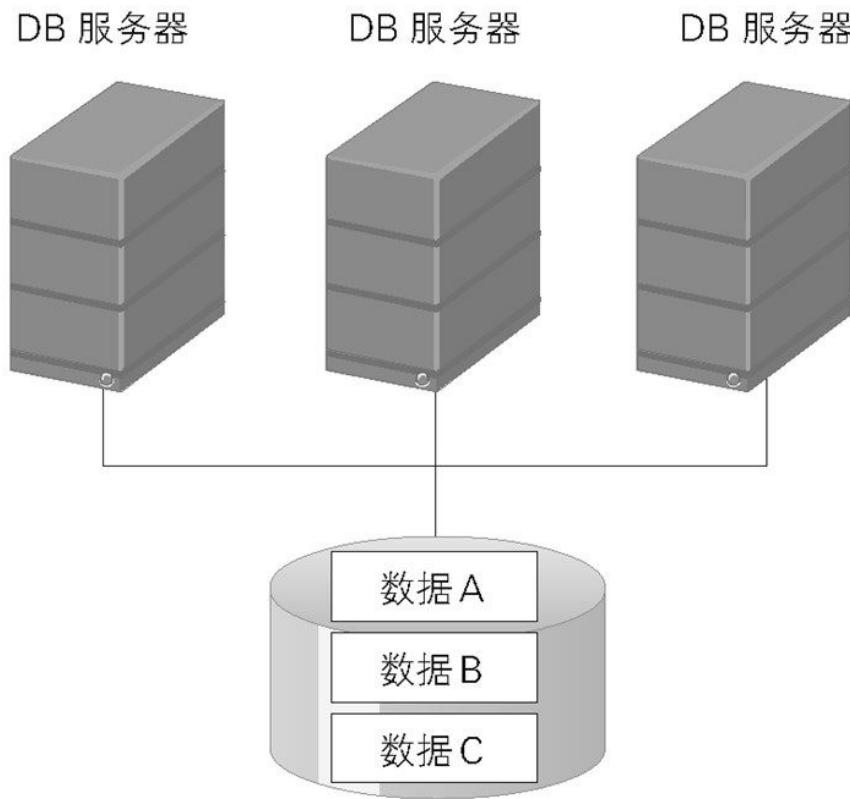


图 4.11 共享磁盘模式

各模式的比较结果与选择标准

下面按照纵向扩展方式和横向扩展方式分别介绍各模式的比较结果和选择标准。

纵向扩展方式是指通过增加现有服务器的 CPU 或内存等来提高服务器的处理性能。扩容模式和升级模式就属于纵向扩展方式。我们将这两种模式的比较结果与选择标准分别总结在表 4.3 和表 4.5 中。

表 4.3 可扩展性应对策略的设计方式的比较结果（纵向扩展方式）

比较项目	扩容※1※2	升级
成本	<ul style="list-style-type: none"> ○：低 因为只增加必需的硬件组件，成本较低 	<ul style="list-style-type: none"> ×：高 所增加的服务器价格昂贵，搭建系统环境也会产生费用，因此成本较高
服务停止时间	<ul style="list-style-type: none"> ×：长 扩容的时候需要关闭服务器电源，因此必须要停止服务。但是，如果采用主/备冗余结构，可以通过切换备用机器和生产环境机器来将服务停止的时间缩短到最小 	<ul style="list-style-type: none"> ○：短 增加的服务器和工作中的服务器并行，可以在不停止服务的情况下进行性能扩展
扩展界限	<ul style="list-style-type: none"> ×：低 直至用完所有的空余插座和插槽 	<ul style="list-style-type: none"> △：中 直至超出服务器所支持的集群数量

※1 也可事先在服务器上预装备用的硬件部件，在有需要的时候激活它们，即所谓的“按需扩容”方式

※2 在扩容的时候，需要通过 MP (Multi Processor, 对称多处理器) 系数（对称多处理器系统中增加 CPU 后机器处理性能提高的比率）和基准值来进行调整

横向扩展方式是指通过增加服务器的数量来提高整体系统的处理性能。适用于很难预测系统业务量或是实际业务量比预测大幅增加的情况。功能分割模式、集群模式、无共享模式、共享磁盘模式属于横向扩展方式。我们将这些模式的比较总结在表 4.4 中，选择标准则分为 Web/AP 服务器和 DB 服务器分别总结在表 4.6 和表 4.7 中。

表 4.4 可扩展性应对策略的设计方式的比较结果（横向扩展方式）

比较项目	功能分割	集群	无共享	共享磁盘
服务器对象	Web/AP服务器		DB服务器	
实施方法	Web/AP服务器（业务逻辑）和DB服务器（DBMS）的功能分割 动态页面生成功能（AP服务器）和静态页面（单纯的HTML和图片服务器等）的分割	通过负载均衡器分散事务（还可实现服务器之间会话同步）	使用DBMS功能	使用DBMS功能 共享存储设备 使用文件系统
扩展时的注意点	与功能分割系统相关的其他系统的接口（连接方式）需要变更	实现会话同步的情况下，如果增加服务器会导致同步负荷加重，可能会导致无法大幅提高性能	需要重新设计数据分割（Partitioning）（也可不分割，在磁盘间进行数据复制）	多次扩展后会导致同步负荷加重，可能会导致无法大幅提高性能

表 4.5 可扩展性应对策略的选择标准（纵向扩展方式）

模式	业务量预测值与实际值的偏差
扩容	小 适用于容易预测业务量，且业务量的变化不大的情况

模式	业务量预测值与实际值的偏差
升级	<p>大</p> <p>适用于业务量与预想相比有大幅增加的情况</p>

表 4.6 可扩展性应对策略的选择标准（横向扩展方式、Web/AP 服务器）

模式	性能瓶颈位置
功能分割	<p>个别处理</p> <p>适用于系统的性能瓶颈在于个别功能或处理的情况（可以按照功能进行扩展）</p>
集群	<p>通用处理</p> <p>适用于不是提高个别功能或处理的性能，而是要确保系统整体事务处理性能的情况</p>

表 4.7 可扩展性应对策略的选择标准（横向扩展方式、DB 服务器）

模式	可扩展性要求	扩展时的难度
无共享	<p>高</p> <p>适用于计划对DB服务器进行扩展，预计业务量会有很大增加等需要大幅提高性能的情况</p>	<p>高</p> <p>需要重新设计 DB数据分割</p>
共享磁盘	<p>中</p> <p>适用于计划对DB服务器进行扩展，但只需要一定程度提高性能即可的情况</p>	<p>中</p> <p>不需要修改现有DB</p>

注意点

没有需要特别注意的地方。

4.3 超负荷应对策略的设计方式

近年来，越来越多的系统都通过网络提供服务，这样就加大了服务器和网络设备的负荷。特别是对于像 Web 站点这样的用户数量大且不稳定的系统，很难准确预测其业务量，因此需要注意防止其系统超负荷运行。

如果系统超负荷运行，服务器的处理速度和网络通信速度就会下降，从而导致“系统无响应”“Web 页面的显示很慢”“不能收发邮件”等问题。

为了防止这些问题的发生，应当事先针对防止系统超负荷运行以及万一系统超负荷运行时，如何缩小影响范围制定“超负荷应对策略”。

访问控制模式

当系统即将超负荷运行时，将用户的访问转移到其他服务器上进行处理，防止系统进入超负荷状态（图 4.12）。具体而言，就是如果用户要访问超负荷运行的服务器，负载均衡器会检测到超负荷状态，并自动将访问转发给紧急服务器处理。紧急服务器通常会在页面上向用户表示“目前系统繁忙，请稍后再试”等提示信息。通过这种方式可以让用户知道系统现在处于超负荷状态，需要过一段时间才能再次访问。像互联网上公开的 Web 站点这种用户数量大且不稳定，很难准确预测其业务量的系统中大多采用该模式。

访问控制模式中，也可以不使用紧急服务器，而是从网络设备中直接返回错误信息。此外，在大规模复杂系统中，系统内部还需要对针对通信阻塞对通信流量进行控制。

访问控制模式的种类参见表 4.8。

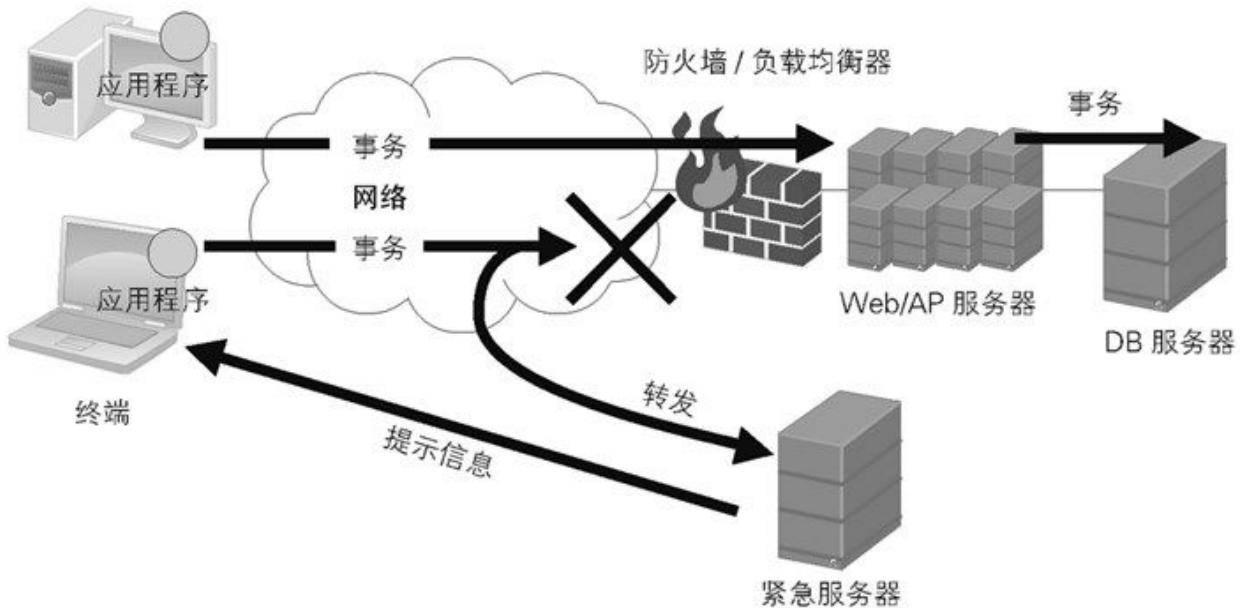


图 4.12 访问控制模式

表 4.8 访问控制的种类

方式	超负荷时的表现	控制单位	成本	设计上的注意点
无访问控制	可能导致系统整体性能下降或者处理出错	x: 无	○: 低 不需要访问控制的设计和测试的成本	事先确认业务需求，确保事务量不能超过系统容量
总量访问控制	当事务总量超过所规定的阈值时，将事务转送到紧急服务器上进行处理，返回提示信息	△: 总量	△: 中 访问控制的设计和测试成本较低	各事务使用的资源量差别很大的时候，需要谨慎地决定阈值

方式	超负荷时的表现	控制单位	成本	设计上的注意点
根据服务种类进行访问控制	当事务量超过各服务所规定的阈值时，将事务转送到紧急服务器上进行处理，返回提示消息	○：服务	×：高 访问控制的设计和测试成本较高	由于很难有效利用资源，请勿将各服务的阈值设计得过小

资源分割模式

以业务和功能为单位分割服务器资源，万一发生超负荷运行的情况，可以将影响范围缩至最小（图 4.13）。在该模式下，即使某台服务器超负荷运行，也只会对那台服务器所负责的业务和功能产生影响，对其他业务和功能不会有任何影响。因此，该模式适用于系统中既存在重要业务和功能，也存在不重要的业务和功能的情况。

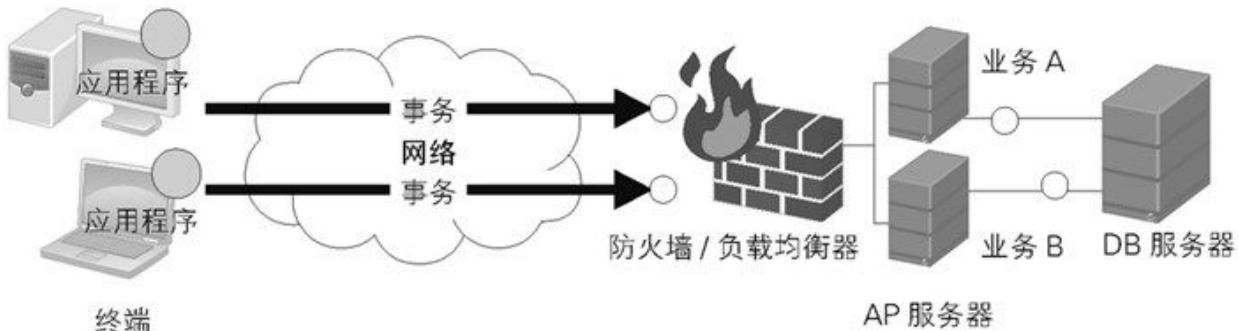


图 4.13 资源分割模式

分割方法有物理服务器分割、逻辑服务器分割、中间件分割等。

并发数控制模式

通过中间件控制系统的并发业务数（图 4.14）。在系统中预先确定并发业务数的上限值，当访问超过这个上限值时，这次访问就会排列到系统内的中间件所管理的队列中，并按照进入队列的顺序等待处理。

还可以通过中间件创建多个队列，并按照业务类别分别管理这些队列。由于超过上限值的事务系统不会处理，因此可以避免系统陷入超负荷状态。而同时，业务虽然进入等待状态，但并不会丢失，因此该模式适用于需要确保任何时候事务都必须被处理的系统。

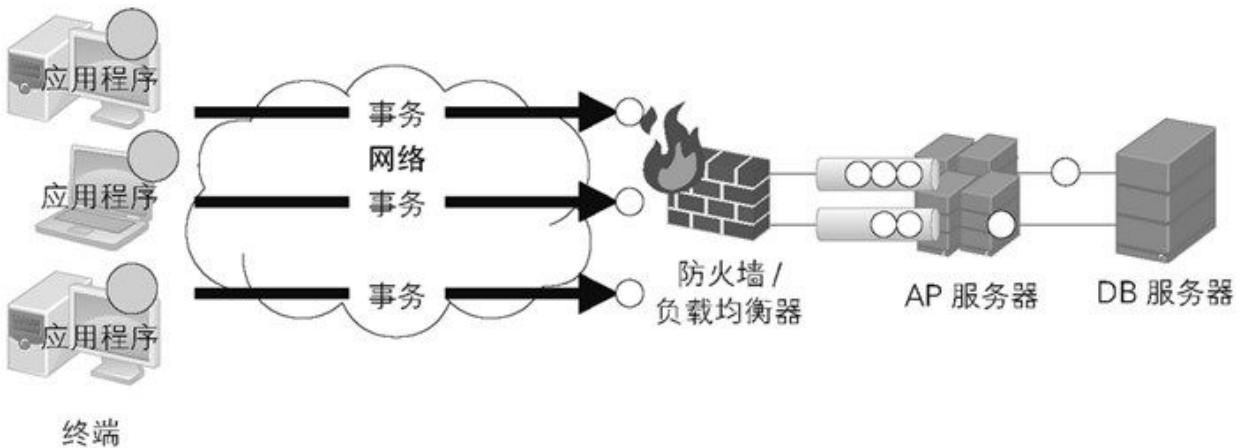


图 4.14 并发数控制模式

并发数量的设计，除了考虑系统吞吐量和响应时间以外，还需要综合考虑各业务的重要性。

资源结构变更模式

当系统超负荷运行时，通过改变系统资源结构来应对通常情况下系统无法处理的那部分负荷（图 4.15）。由于需要改变系统结构，因此该模式适用于可以预测系统负荷高峰期和负荷峰值的情况。

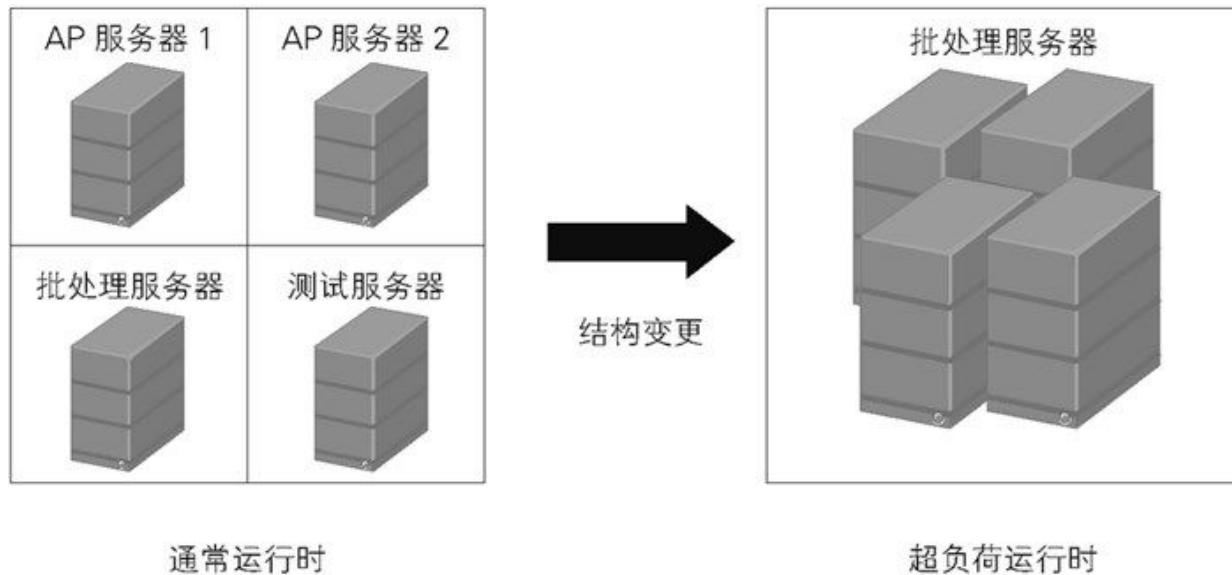


图 4.15 资源构成变更模式

此外，当变更资源结构时，需要注意不要违反中间件的授权协议。

网络带宽控制模式

控制网络通信数据量（图 4.16），可以防止低优先级业务的通信数据量过大而影响到高优先级业务的数据通信。具体而言，就是通过控制通信数据包的优先级，来确保重要业务有足够的通信带宽可以使用。

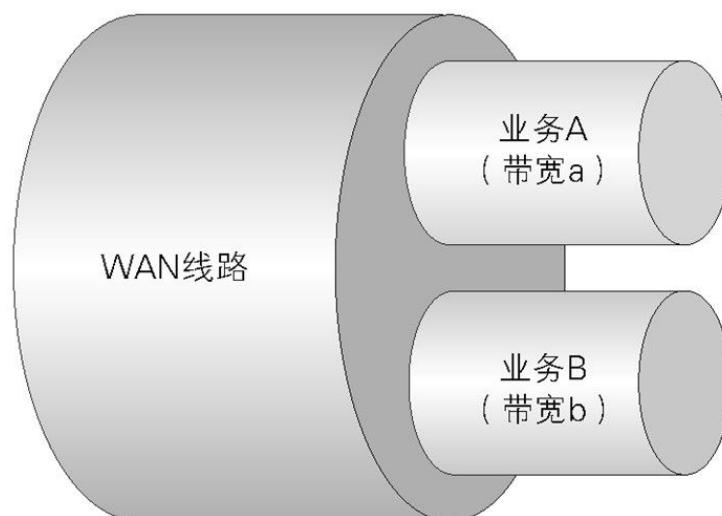


图 4.16 网络带宽控制模式

各模式的比较结果与选择标准

我们将各模式的比较结果与选择标准分别总结在表 4.9 和表 4.10 中。此外，组合这些模式还可以取得更好的效果。

表 4.9 超负荷应对策略的设计方式的比较结果

比较项目	访问控制	资源分割	并发数控制	资源结构变更	网络带宽控制
目的	防止系统整体超负荷运行	万一系统超负荷运行，将影响范围缩至最小	防止服务器超负荷运行（资源不足）	优化资源结构以应对预测到的负荷高峰	确保处理特定通信的性能
方法	限制系统的访问数量，当超过上限时，将事务转发到紧急服务器上进行处理	按照业务和功能单位分割资源	通过中间件等控制事务并发数	变更服务器等的资源结构，应对超负荷	实行网络优先控制和带宽限制
应对突发高负荷	○：可以应对			×：难以应对 资源结构变更必须是有计划地进行的，因此很难应对突发高负荷	○：可以应对
优化对象	访问量	无优化	事务并发数	无优化	网络通信的优先级和带宽

表 4.10 超负荷应对策略的设计方式的选择标准

模式	特点
访问控制	适用于用户数量多且不稳定，业务量难以预测的系统
资源分割	适用于既有重要的用户/业务，也有不重要的用户/业务的系统
并发数控制	适用于必须确保所有事务都被处理的系统
资源结构变更	适用于可以预测业务量和负荷高峰时期的系统
网络带宽控制	适用于既有重要的通信数据，也有不重要的通信数据的系统

注意点

没有什么特别需要注意的地方。

4.4 总结

本节讲解了性能与可扩展性策略的基本思考方法和设计方式。如果不制定性能与可扩展性策略，可能会因严重性能问题引起系统宕机和响应速度变慢。反之，如果采取过多的性能与可扩展性策略，则可能导致投资浪费，因此需要慎重考虑。

第5章 运用与维护性需求的实现策略：不放过系统故障

运用与维护性需求是与维持系统正常运行的作业（监控、改进或修改、故障恢复等）相关的需求。系统的生命周期可分为两大部分：开发阶段和运维阶段。为了确保系统能在运维阶段稳定运行，需要在开发阶段就设计“运维功能”和制定“运维管理”。

本章中，我们将讲解运用与维护性策略的基本思考方法和设计方式。

5.1 运用与维护性策略的基础

本书中将检测系统故障的方法和减轻运维作业负担的方法称为“运维功能”。表 5.1 中所列举的运维功能中，许多都是需要在开发阶段就整合到基础设施中。接下来，我们将主要讲解“系统监控”“任务管理”“备份管理”等典型运维功能实现过程中的要点。

表 5.1 主要的运维功能

功能	内容
系统监控功能	<ul style="list-style-type: none">● 实时观测系统运行状态，检测系统故障并通知系统管理员和运维人员● 收集、保存系统资源信息，使运维人员可以进行相关分析
任务管理功能	<ul style="list-style-type: none">● 运行批处理等处理内容固定的任务，并管理它们的运行● 运维功能并非设计批处理任务的处理方式，而是设计和实现批处理任务的运行管理功能
备份功能	<ul style="list-style-type: none">● 发生物理故障和逻辑故障时保护和恢复数据
日志管理功能	<ul style="list-style-type: none">● 收集日志、规范日志内容、统一管理和保存日志，使运维人员容易理解和分析日志

功能	内容
构成管理功能（发布管理）*1	●为了改善系统的故障恢复功能和性能，确保发生修改和变化的软件和硬件能被应用于各种环境中

*1 后面的运维管理中也包括构成管理（发布管理）

本书中将在运维阶段实践开发阶段中制定的运维原则和决定事项的活动称为“运维管理”。在考虑运维管理时，需要特别注意表 5.2 中的 3 个 P。

表 5.2 维护管理的 3 个 P

3个P	内容
People（体制、职责）	实施运维作业的人和体制，以及业务上的职责和责任
Process（程序）	运维上必要的管理和作业，以及它们的程序
Product（产品）	用于支持运维工作和提高运维工作效率的工具

运维功能和运维管理之间有着紧密的联系，如果忽略了这点，很容易导致运维阶段作业负担加重和服务品质下降等问题。因此，在开发阶段设计“运维功能”和制定“运维管理”时，需要时刻注意它们之间在内容上是否有不相符或者矛盾的地方。

系统监控

首先，我们来讲解系统监控的要点。

监控对象和监控项目

考虑到实现系统监控功能的性价比和使用效率，我们不应当随意增加监控对象和监控项目，而是要严格控制它们。因此，在最初设计时就应当根据监控对象的重要度和特点，整理和制作监控对象和监控项目的矩阵表（表 5.3），并随着设计的深入逐步细化该矩阵表。

表 5.3 监控对象和监控项目的矩阵表示例

监控对象	可用监控	日志监控	进程监控	资源监控（阈值监控）	资源监控（信息保存）	详细监控
Web服务器	○					×
AP服务器						
DB服务器	○					
备份服务器						
运维管理服务器	○				×	
网络设备	○	×	—	○ SNMP监控		○ SNMP 自陷
共享存储设备	×		—	× 由服务器监控		

监控消息的重要度

当系统监控检测到异常现象时，会向监控人员发送监控消息。如果每次都向全体监控人员发送监控消息，并要求商议处理对策，会非常烦杂且低效。因此，系统监控中需要有一种功能，让监控人员可以根据所检测出的事件的严重程度来设置消息和应对的优先级。为此，预先对所有监控消息设置重要度是很重要的。表 5.4 中列举了设置监控消息重要度的示例。

表 5.4 设置监控消息重要度的示例

重要度	事件	例
致命	服务停止	进程停止
警告	如果不立即处理会导致服务停止	磁盘使用率达到90%以上
注意	虽然不会立即导致服务器停止，但必须处理	磁盘使用率达到80%以上
信息	虽然不需处理，但需要确认	批处理程序正常结束

监控消息的通知对象和通知方法

我们还要考虑当系统监控检测到异常现象时，“如何”发送通知以及向“谁”发送通知。

通知对象一般有监控中心的监控人员、系统管理员、客户等。需要根据通知对象适当地选择监控消息的重要度和内容。例如，向客户发送重要度为“致命”以上的监控消息，向系统管理员发送重要度为“警告”以上的监控消息等。此外，有些情况下白天和夜晚的通知对象也会有所不同。

常用的通知方法有：对在监控中心内的运维人员通过监控画面和旋转灯鸣叫进行通知；对在监控中心外的相关人员通过发送电子邮件进行通知。由于在系统故障时可能会产生大量的通知消息，旋转灯的持续鸣叫可能会导致漏掉其他故障，而发送大量电子邮件可能会导致邮件系统误认为是垃圾邮件而自动将其屏蔽，这些都会给系统监控带来困难，因此在设计通知方法时需要特别留意。

消息过滤

在大型系统中，可能会产生大量的监控消息，监控画面也会比较凌乱，给系统监控带来困难。因此，需要尽可能在画面上过滤掉不必要的监控消息，简化画面的显示。例如，可以仅显示高重要度的监控消息或是将多条相同的监控消息合成为一条显示。

消息重要度的变换

由于对 OS 和中间件的日志设置的重要度可能和监控消息的重要度不同，因此如果直接将这些日志作为监控消息发送给相关人员，可能导致遗漏重要的监控消息或是发送了实际上不需要处理的监控消息（例如，当主要进程停止，服务出现致命错误时，OS 却将其当作“信息”级别的重要度进行处理等）。所以，在监控日志时，需要将 OS 和中间件的日志重要度与监控消息的重要度进行恰当的变换。

测试和监控的优化

在系统监控测试中，需要确认监控设计是否符合系统的运用与维护性需求。如果有不相符的地方，则需要进行优化。这里所说的优化是指将在测试中发现的遗漏的监控项目增加到监控对象中，或反之将不需要监控的项目从监控对象中剔除。

特别是对于与系统异常相关的监控消息，在监控设计阶段毫无遗漏地将它们列举出来是非常困难的。因此，可参考其他基础设施测试中的日志消息，进行适当优化后用于本系统中。有些情况下，即使系统服务上线后，也需要继续进行这种优化。

任务管理

接下来讲解任务管理的要点。

自动化的范围

在设计任务管理时，首选需要确定自动化作业的范围，包括“允许有多少手动作业”“线上处理中的异步处理是否需要作为任务运行”等。

自动化的范围可能会对系统的设备构成和其他非功能性需求有影响，因此需要特别注意。例如，如果将电源的开关自动化，则需要引入 UPS (Uninterruptible Power Supply, 不间断电源)。另外，如果线上处理中执行异步任务，则需要通过任务管理功能控制任务的最大并发数，并确保不会对线上处理的运行速度造成影响。

判断是否引入任务管理软件

根据之前确定的自动化范围，决定是使用 OS 附带的标准功能 (Cron¹ 等) 来实现目标自动化范围，还是引入专业的任务管理软件来实现。如果所执行的处理需要在多台服务器之间进行交互或是需要确保任务执行的顺序，推荐引入专业的任务管理软件。

¹ UNIX 或者类 UNIX 系统中使用的自动执行任务的程序。

任务的统一管理

在系统中只设置一台任务管理服务器。如果设置多台任务管理服务器，任务执行顺序和执行结果的统一管理会变得困难，容易引起系统故障或加大任务管理难度。

当然在有些复杂的大型系统中，任务的运行性能非常重要，必须设置多台任务管理服务器。但即使是这样，也不要并用多种任务管理功能 (Cron 等)，而应当引入单一的专业任务管理软件进行统一管理。

任务管理服务器的可用性

如果需要通过运行任务来进行线上处理的关闭² 和预关闭³ 等整体系统的控制，那么任务管理服务器发生故障就可能会导致严重系统故障。

因此，应当将任务管理服务器组成 HA 集群或是采用容错服务器来确保其高可用性。

² 并不停止整个系统，而是仅仅停止线上处理。

³ 由于即将停止（关闭）线上处理，需要拒绝新的处理请求，仅按照顺序处理请求队列中剩余的请求。就像餐馆即将打烊一样，不接待新的顾客，只等待店里的顾客结账离开。

任务设计规范

为了整理任务和任务集（Job Net）⁴ 的结构，使维护作业更加容易，需要制定任务的设计规范。表 5.5 展示了任务设计规范的示例。

⁴ 定义了执行顺序的多个任务的集合。

表 5.5 任务设计规范的项目示例

项目	内容
层级	定义任务集的层级和用途（按照功能区分、按照业务区分等）。层级太深会导致运维上的不便，因此应当尽量控制在5层以内
结束代码	决定任务所调用的Shell脚本的结束代码，通过结束代码控制后续任务的执行 例： 正常结束：0~39：执行后续任务 警告结束：40~99：执行后续任务 异常结束：100~255：不执行后续任务
任务执行的脚本	决定Shell脚本的语言、文字编码、环境变量和脚本的日志文件。 此外，还需要决定脚本的详细编码规范

项目	内容
任务ID的分配标准和命名规范	决定各任务和任务集的ID和命名规范
恢复方针	决定任务失败时的恢复方针 决定再次执行任务的粒度、中止线上服务的标准等

备份管理

接下来讲解备份管理的要点。

设想物理故障和逻辑故障

存储设备的 RAID (Redundant Arrays of Independent Disks, 独立冗余磁盘阵列) 技术可以防止磁盘故障等物理故障导致的数据损坏，但无法防止因误操作或是系统的处理错误等逻辑故障导致的数据损坏。与其相反，存储快照技术可以防止逻辑故障引发的数据损坏，却无法防止物理故障引发的数据损坏。因此，要想同时防止物理故障和逻辑故障导致数据损坏，需要同时采用两种技术，即在不同的机柜和媒体中，对具有一致性的数据进行复制。

上游设计阶段确定设备构成

由于备份管理功能的设计对整体系统的设备构成和成本有很大影响，因此在上游设计阶段尽早展开讨论就显得非常重要。具体而言，服务器和存储设备等备份装置、LAN/SAN 的网络设备、备份软件的授权等都是会对系统成本有影响的设备。在上游设计阶段讨论这些设备构成时，主要是通过备份数据量和所允许的备份时间来计算备份时所必需的数据传输速度。

根据备份对象进行设计

备份对象不同，备份频率和保存期间也不同，这一点在设计时需要注意。例如，保存期间不同的数据如果保存在同一个磁带媒体中，则无法循环利用该媒体（Media Rotation）⁵，因此必须设计使用不同的存储媒体分别保存这两种数据。我们在表 5.6 中展示了主要的备份对象。

⁵ 覆盖已经超过保存期间的媒体的数据。

表 5.6 主要的备份对象

比较项目	数据备份		系统备份
	业务数据	日志文件	
数据的备份频率	高 因为数据经常被更新		低 在系统构成发生变化时获取备份即可
数据量	多 共享磁盘上的DB等	少 访问日志的文件大小左右	多 整个本地磁盘
保存期间	短 很少需要恢复到以前的状态	长 需要保存数年	保存数次系统更迭 一般与保存期间无关，都是保存数次系统更迭
恢复需求	严格 需要迅速恢复数据	宽松 主要用于调查和审查，需求不严格	严格 系统处于停止状态，需要迅速恢复系统

备份功能的统一

为了提高系统的运维性，应当集中和统一用于备份的设备和网络，而不是按照服务器分别搭建。具体而言，就是可搭建备份专用网络，使所有服务器的备份数据均通过备份专用网络传输，并与其他需要获取备份数据的服务器共享备份设备等。

此时，还应当考虑如何控制备份时间以防止备份设备超负荷运行，以及备份功能的可扩展性以弹性地应对将来备份数据量的增加。

备份设备的选择

备份装置一般使用磁带设备或者磁盘设备。在选择备份设备时，需要考虑表 5.7 中所记载的备份设备特点来做出最合适的选择。

表 5.7 磁带和磁盘的差异

比较项目	磁带设备	磁盘设备
性能	○ 顺序存取 ^{*1} (DB完全恢复等) 快	△ 随机存取 ^{*2} (单个文件恢复等) 快。较依赖于设备构成，难以做出预算
可用性	△ (与磁盘相比) 容易发生故障的机械动作较多	○ 通过组成RAID可以降低机械动作导致的故障发生几率
运维性	✗ 需要清洁、更换等定期维护作业	○ 基本无需维护作业
可移动性	○ 容易移动、可异地保存	✗ 难以异地保存
可扩展性	○ 容易增加和取出磁带进行维护	✗ 容量不足时需要增加磁盘

*1 从前往后依次读取存储单元

※2 通过索引信息直接读取想要访问的存储单元

备份软件的选择

获取备份数据，是为了在物理故障和逻辑故障导致数据损坏时能够有迹可循，进行数据恢复和审查，所以在获取备份数据时需要确保备份数据的完整性和正确性。因此，是否支持系统所采用的 OS 和 RDBMS 等中间件是备份软件选择时的一个重要基础。此外，如果是备份软件最近才支持或是主版本号升级后才支持 OS 和 RDBMS 等中间件，那么在选择备份软件前一定要进行充分的验证。

运维管理

接下来讲解表 5.2 中介绍的 3 个 P (People、Process、Product) 的要点。

People (体制、职责)

为了能够平稳地进行运维管理，需要明确系统相关人员的职责。表 5.8 展示了典型的运维阶段各人员的职责。

表 5.8 运维阶段的职责示例

组织/部门	职责示例
服务的用户	
系统用户	实际使用系统
用户部门	站在系统用户的角度，与运维组织一同进行调整
服务的提供者	
运营组织	按照运营规则进行固定的系统运营作业，并处理系统用户提出的咨询
维护	接手开发完成的系统，进行修复应用程序或软件、硬件，应对各种故障等

管理组织	内容不固定的维护作业。有时该职责由开发组或外部供应商担任
统一管理组织	统一负责向使用部门报告和进行各种调整。也可由其他组织兼任

此外，还需要考虑以何种体制来履行这些职责。虽然系统不同，各相关人员担当的职责也不尽相同，但我们还是在图 5.1 中列举了一些典型运维体制。在小型系统中，也可将多个职责交给同一个组织承担。

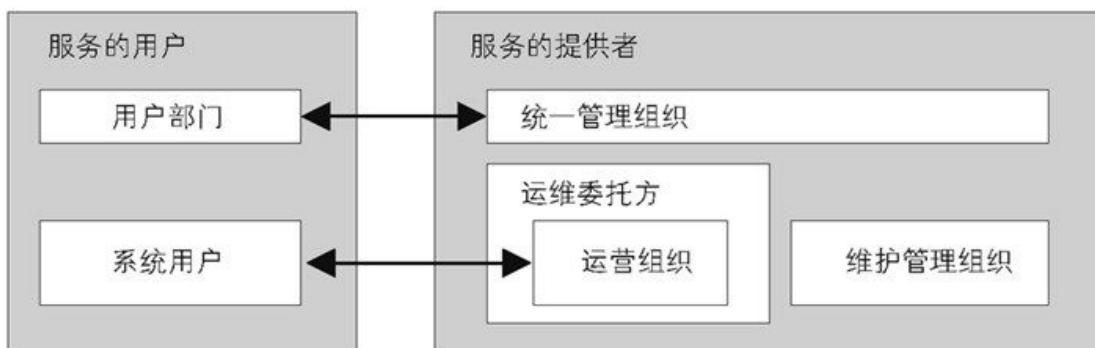


图 5.1 典型的运维体制示例

Process (程序)

运维管理程序分为“管理程序”和“作业程序”。

管理程序是指系统发生故障时对应的流程、在生产环境实施作业时的授权流程等以组织为单位进行的运作程序。表 5.9 列举了管理程序中应当规定的主要管理项目。

表 5.9 主要管理项目

项目	内容

项目	内容
事件管理	故障信息、客户的意见、供应商提供的信息、防止再次发生故障的措施等，受理这些对服务有影响的事象，并统一管理。当服务中断时迅速处理，尽快恢复服务
问题管理	深究对服务有影响的事象或问题的根本原因，根据情况进行临时处理以及制定和提供最终解决方案
变更管理	评估系统配置项的变更给服务提供带来的影响，在开发环境和测试环境进行验证，并确保只有被授权的变更落实到了生产环境中，将变更风险减至最小
发布管理	将客户授权的系统配置项的变更作为发布对象进行管理，按照发布作业手册中制定的作业程序进行增加、修改、撤去等作业
构成管理	通过正确地管理系统配置项的信息，为正确评估增加、修改、撤去等作业给系统带来的影响提供相关信息
服务级别管理	设置包含组织内部目标在内的SLA (Service Level Agreement, 服务水平协议)，通过反复监控、分析、汇报和改善问题来维持服务水平，并通过持续改善来提供有效和高效的服务
性能管理	掌握系统的使用情况和性能情况，预测未来的变化倾向，分析系统中的设备的处理能力和必要的设备数量
可用性管理	收集、分析和评估与可用性相关的数据，并编写和提供报告资料，并向承担制定策略和进行改善活动的SLA组织进行反馈

项目	内容
服务持续性管理	将严重灾害对服务的影响缩至最小，并在与客户协商的时间内恢复服务
信息安全管理	防止系统发生信息安全事故，确保信息资产的保密性、完整性和可用性
培训、演练管理	为了能维持、提高服务级别，明确系统管理员所需的能力，如果能力欠缺，可以通过培训和演练来提高

这些管理程序不仅应当在提供服务的组织内部，与设备供应商和客户之间，就发生故障时的应对和进行系统变更时的规则也需要达成一致。此外，为了防止管理程序的陈腐化和形式化，还需要定期地评估服务级别并采取必要的改善措施。

作业程序是指运维作业人员在进行系统作业时需要遵守的程序。作业程序很大程度上依赖于系统运维功能的实现方式。

如果不制定充分的系统作业程序，容易发生因操作失误导致系统故障，或者因交接不充分导致运维成本增加的情况。因此，可以在系统测试阶段就让运维成员加入，与开发成员一起编写系统作业手册，或进行走查（Walk Through），共同努力提高作业程序的质量。

当系统构成发生变更导致需要修改作业程序时，除了需要遵守管理程序中的变更程序来修改作业程序，还需要通知相关人员，并进行培训和演练。

Product (产品)

我们需要考虑引入支持运维管理的产品（工具）。

例如，在小型系统的运维管理中，可以使用表格计算软件制作的管理文档，并使用电子邮件和电话来与相关人员进行沟通。但是在大规模系统的运维管理中，一般都需要通过导入专用的运维管理工具（供应商提供的产品或是开源工具）来综合管理运维信息，实现系统变更时的授权工作流程，高层管理者也可以掌握系统运维的整体状况，使运维工作更专业、更高效。

如果准备购买的工具性价比不高，或是工具提供的功能无法满足需求，可以考虑定制工具或是自己开发工具。

至此，我们讲解了运用与维护性策略的基本思考方法。接下来我们将分类讲解运用与维护性策略的设计方式。我们首先会介绍运维体制的设计方式，然后讲解运维功能的构成管理、系统监控、任务管理的设计方式。最后，我们将介绍时钟同步、杀毒软件更新的设计方式。

5.2 运用与维护体制的设计方式

为了能够持续地向客户提供稳定的服务，需要系统具有高可用性。虽然可以采取第2章中介绍的冗余等策略来尽可能地防止故障发生，但是“检测到故障先兆时，如何采取措施避免故障发生”和“万一发生故障时，如何迅速恢复”对于提高系统可用性也是非常重要的。要想解决这些问题，需要考虑如何建立系统监控、设备的更换与修理等运维体制。

服务级别提升模式

该模式是指进行系统监控，基于ITIL进行运维管理，并派遣专业运维人员常驻生产环境现场负责运维工作（图5.2）。系统监控可实时监测

系统故障，并找到故障原因。

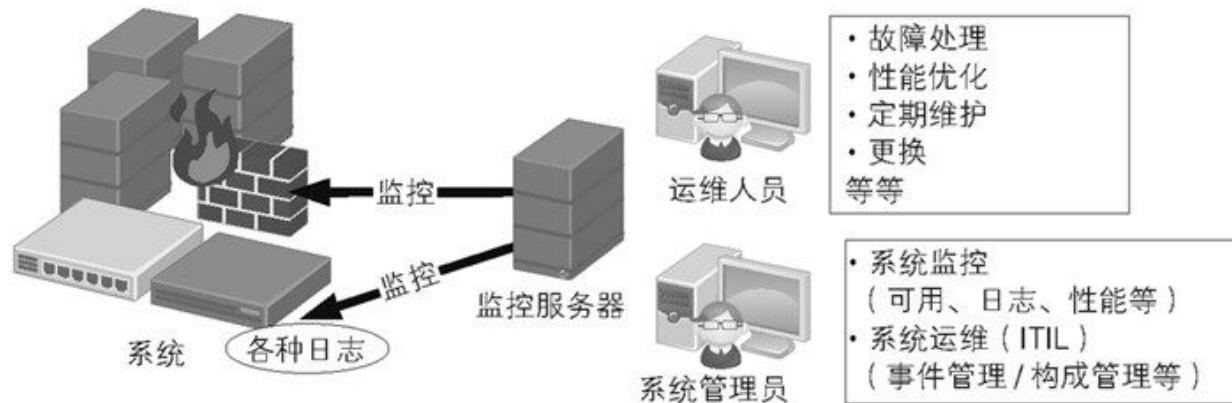


图 5.2 服务级别提升模式

系统管理员基于系统负荷变化和系统事件状况等运维信息来管理系统，确保向用户提供的服务级别满足要求。

运维人员与系统管理员相互配合，随时进行性能优化和硬件更换等维护工作。维护工作不仅可以将故障风险降至最低，还可以持续地提高服务质量。

服务级别管理模式

进行系统监控，并基于 ITIL 进行运维管理（图 5.3）。与“服务器级别提升模式”相比，除了缺少运维人员常驻现场所带来的益处外，其他与“服务器级别提升模式”相同。

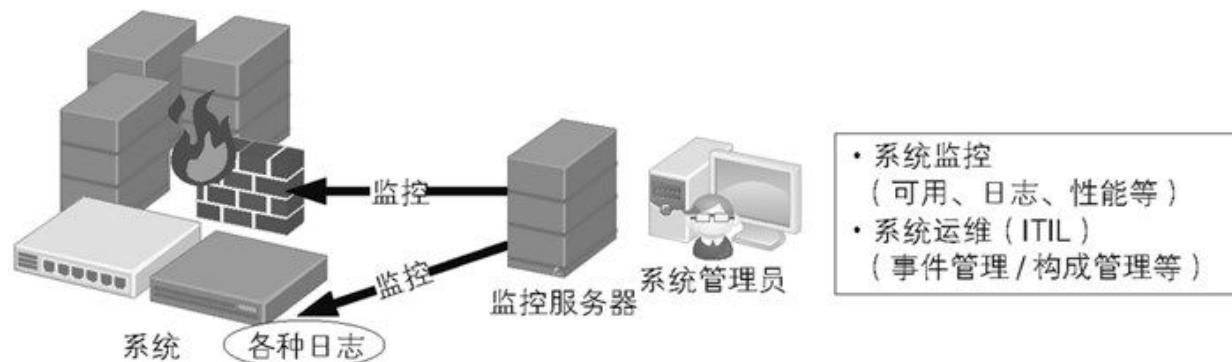


图 5.3 服务级别管理模式

定期监控模式

不监控系统，只获取系统各构成部分输出的日志（图 5.4）。

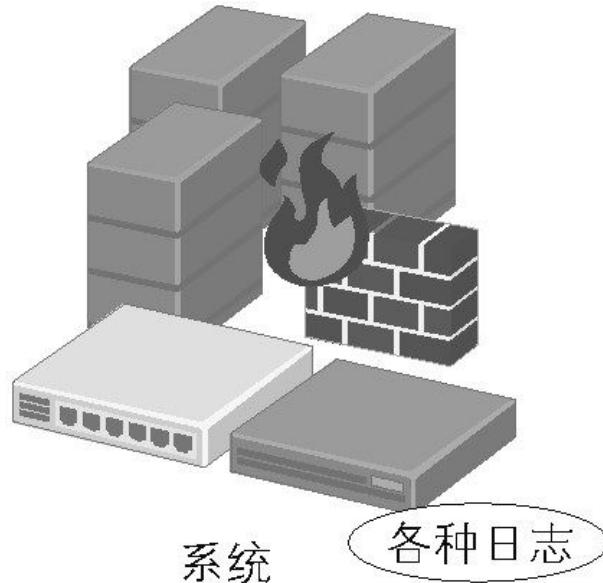


图 5.4 定期监控模式

系统管理员定期检查所获取的日志，判断系统运行是否正常和系统性能是否需要提高。

除了因故障导致系统服务停止或是在检查日志时发现系统异常，该模式基本上无法检测出其他异常，因此故障处理速度相比其他两种模式会慢很多。

各模式的比较结果与选择标准

我们将各种模式的比较结果与选择标准分别总结在表 5.10 和表 5.11 中。

表 5.10 运用与维护性策略的设计方式的比较结果

比较项目	服务级别提升	服务级别管理	定期监控
故障处理速度	◎：数小时 既进行系统监控也进行维护作业，因此可快速处理故障	○：1天左右 系统监控可确定故障发生原因，因此可快速处理故障	△：数日 虽然在日志中记录了故障信息，但确定故障原因需要花费时间
故障预防	○：可 通过检测故障发生先兆并及时更换硬件等处理可以预防故障		×：不可 很难及时检测异常，基本无法预防故障（但是系统管理员可通过定期分析日志检测到异常）
内容	除了服务级别管理模式中的内容以外，还有以下内容 ●软件故障处理 ●硬件故障处理 ●性能优化 ●定期维护 ●预防维护 ●设备更换 ●安全策略	除了定期监控模式中的内容以外，还有以下内容 ●系统运行监控 ●系统构成管理/变更管理 ●系统性能管理 ●系统操作（异地备份、任务维护） ●计算机资源管理 ●系统安全监控和维护（运行安全补丁程序）	内容如下 ●获取性能日志 ●获取通信日志 ●获取访问日志

表 5.11 运用与维护体制设计方式的选择标准

模式	服务质量的重要性	运维成本
服务级别提升	高 适用于系统所提供的服务质量是系统竞争力的源泉，需要持续地提高服务质量的情况	高 适用于有可以进行运维监控作业的工程师和维护作业人员（签订维护合同）的情况

模式	服务质量的重要性	运维成本
服务 级别 管理	中等 适用于需要确保系统所提供的服务的质量达到一定的水平情况	中等 适用于有可以进行运维监控作业的工程师的情况
定期 监控	低 适用于不需要确保较高服务质量的情况	低 适用于想将运维人员和运维成本减小到最少的情况

注意点

365 天 24 小时运维作业成本高昂，因此需要根据系统的重要性来选择合适的运用与维护体制。

5.3 构成管理的设计方式

如果业务终端用户的安全意识和遵守规则的意识很差，就容易引发安全问题。典型的安全问题包括用户在业务终端上安装了文件传输软件等可疑软件，却没有安装应当安装的安全软件，导致业务终端感染计算机病毒或是遭到非法入侵，使得其中的保密信息和个人信被泄露出去。此外，还有在终端中安装了盗版软件和禁止用于商业用途的软件导致第三者遭受损失的情况。

发生以上这些情况会给企业活动和企业信誉带来严重影响，必须考虑对安装在业务终端上的构成要素进行恰当管理的“构成管理”。

隔离网络模式

通过隔离服务器获取终端的构成信息（图 5.5）。当终端试图访问业务系统时，认证路由器会将第一次的访问先转发给隔离服务器进行处理。隔离服务器会检查终端中是否有非法构成要素（恶意软件和盗版软件等），如果检查通过则认证交换机允许该终端访问业务系统。如果终端无法通过检查就不能访问业务系统，因此只要用户要想使用系统，就可自动获取其使用的终端信息。

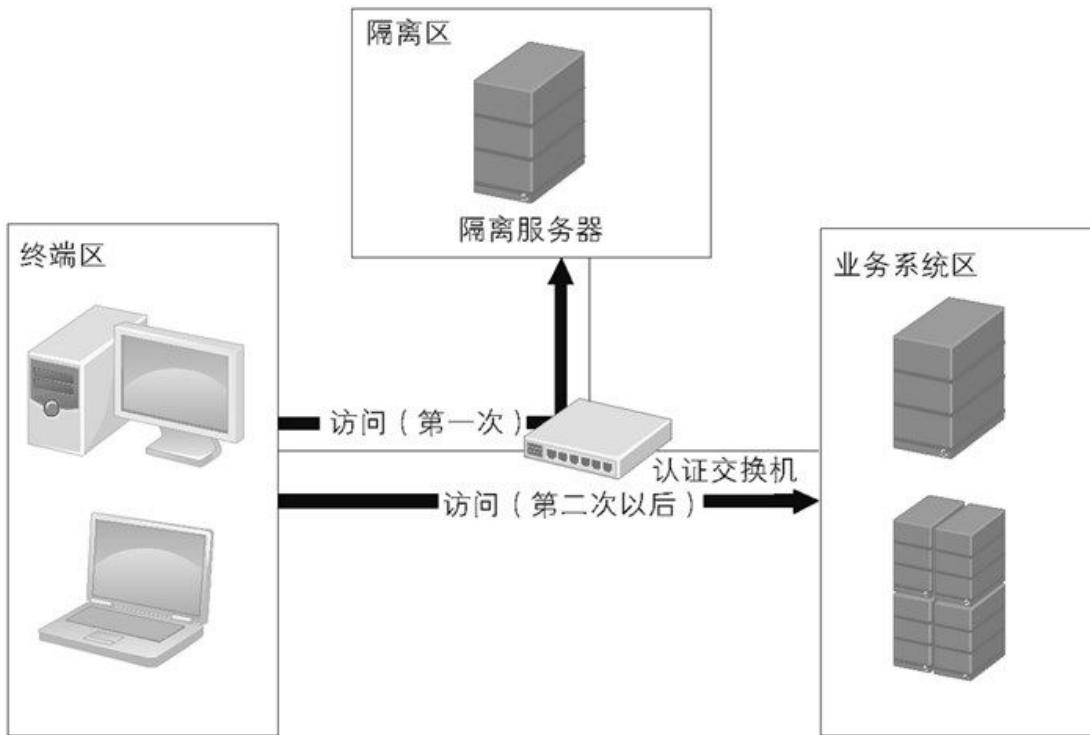


图 5.5 隔离网络模式

代理软件构成管理模式

在终端上安装用于收集软件安装信息的代理软件，向构成管理服务器发送终端的构成信息（图 5.6）。系统管理员检查构成管理服务器中收集到的信息，确认其中是否有可疑终端或是包含非法软件的终端。

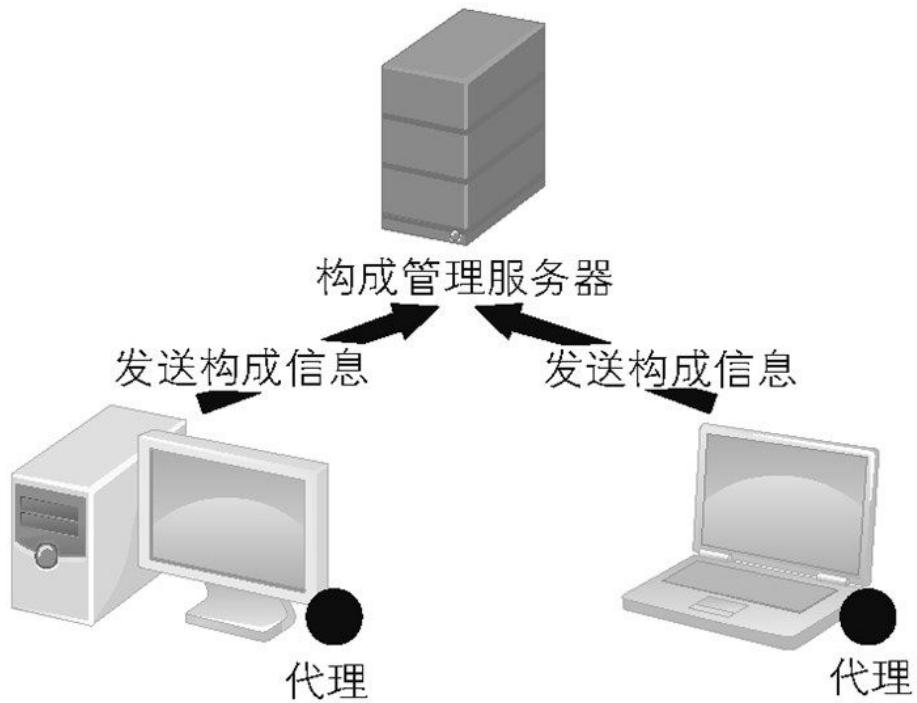


图 5.6 代理软件构成管理模式

终端的用户只需要在第一次访问业务系统时安装代理软件，之后即可自动收集终端的软件安装信息。

工具软件构成管理模式

在终端上安装用于收集软件安装信息的工具软件，向构成管理服务器发送终端的构成信息（图 5.7）。系统管理员检查构成管理服务器中收集到的信息，确认其中是否有可疑终端或是包含非法软件的终端。

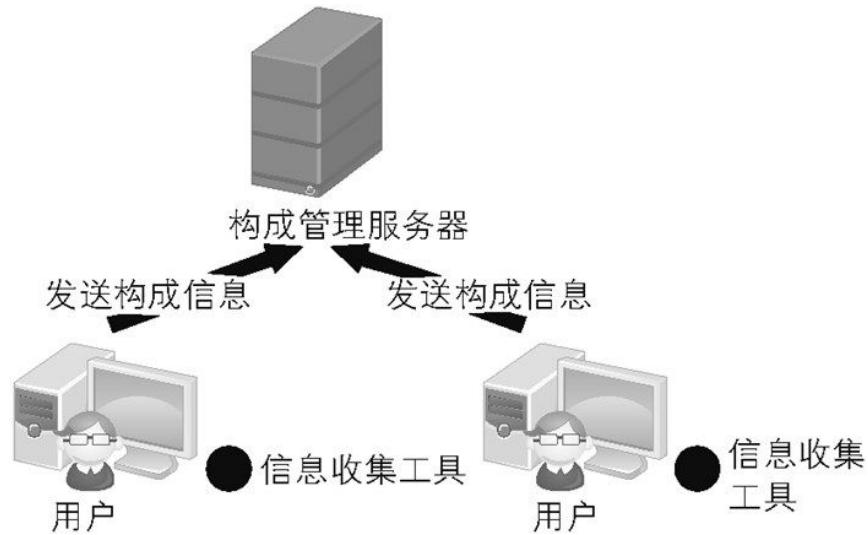


图 5.7 工具软件构成管理模式

终端的用户需要在每次访问业务系统时都运行工具软件。

各模式的比较结果与选择标准

我们将各种模式的比较结果与选择标准分别总结在表 5.12 和表 5.13 中。

表 5.12 构成管理设计方式的比较结果

比较项目	隔离网络模式	代理软件构成管理模式	工具软件构成管理模式
软件安装信息信息收集遗漏的风险	○：低 但是，如果管理对象终端的用户不使用系统就无法进行收集	○：低 但是，如果用户没有给管理对象终端安装代理软件就无法进行收集	×：高 只要用户不运行工具软件就无法收集

比较项目	隔离网络模式	代理软件构成管理模式	工具软件构成管理模式
软件安装信息收集的自动化程度	◎：高 自动收集软件安装信息，无需用户操作（访问业务系统时自动收集）	○：普通 自动收集软件安装信息，但需要用户安装代理软件	△：低 每次收集软件安装信息都需要用户运行工具软件
安全性	◎：强 管理对象终端访问业务系统前，通过隔离服务器检查终端，可以隔离可疑终端	○：一般 可实时检测可疑终端。有些代理软件还可以控制所安装的软件	△：弱 如果用户不运行工具软件就无法检测可疑终端

表 5.13 构成管理设计方式的选择标准

模式	用户依赖程度	管理对象数量
隔离网络	低 适用于用户比较消极的情况（不依赖于用户的操作，可自动获取软件安装信息）	多 适用于管理对象终端数量多，无法抽出时间和精力进行构成管理的情况
代理软件构成管理	中等 适用于用户希望在其所使用的终端中安装代理软件的情况	中等 适用于可以抽出时间和精力在管理对象终端上安装代理软件的情况
工具软件构成管理	高 适用于用户希望在访问业务系统时都运行工具软件的情况	少 适用于管理对象终端数量少的情况

注意点

发送终端的构成信息给构成管理服务器时会增加网络通信的负荷。在用户设定代理软件和运行工具软件时，需要通过提示等方式进行调整，防止构成信息的发送对业务通信造成影响。

5.4 系统监控的设计方式

现在，越来越多的系统要求 24 小时不间断运行。为了实现高可用性，要求系统管理员必须能快速检测出故障先兆，并进行处理。但是，随着系统形态越来越复杂，系统的构成要素越来越多样，通过人眼监控系统运行状态变得非常困难。因此，必须在运维作业中引入系统监控功能，这其中还包括 CPU 使用率的监控、磁盘使用率的监控等性能监控。如果不监控服务器资源的使用情况，CPU 超负荷就会引发系统处理能力不足，磁盘空间不足则会引发系统停止。

可用监控模式

检测设备是否停止运行（图 5.8）。该模式并不能防止设备停止，仅能在设备停止后检测出故障，适用于系统中重要性不高，且不想花费较高监控成本的设备。

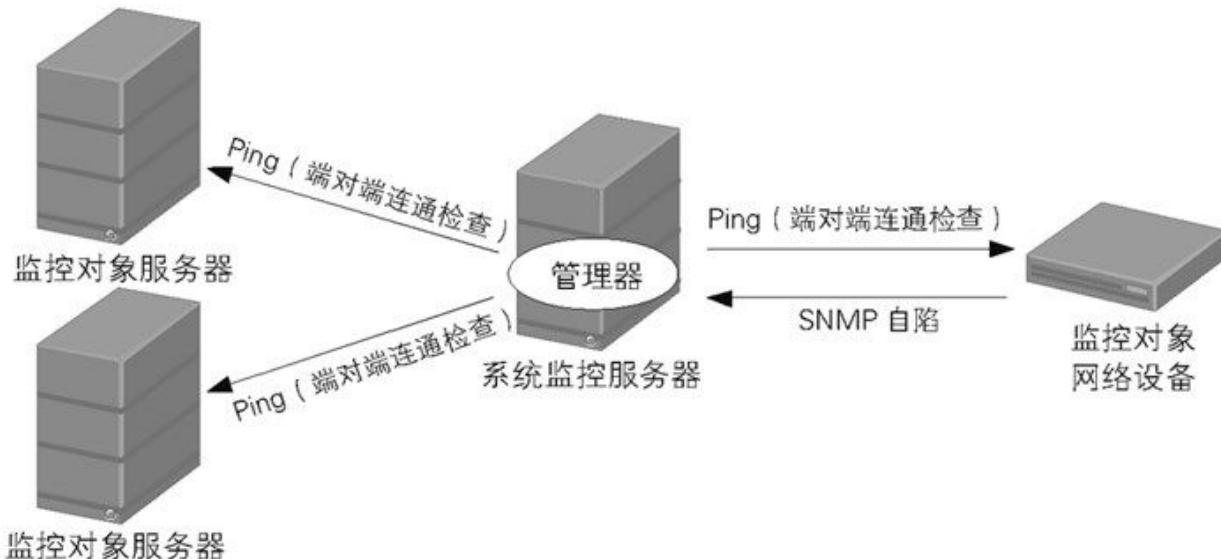


图 5.8 可用监控模式

代理监控模式

在监控对象服务器上安装代理软件，由代理软件接收系统监控服务器发送的轮询⁶消息，检测监控对象服务器是否发生故障（图 5.9）。在监控代理软件中，可以设置系统中的各种运行信息作为监控项目。此外，即使系统监控服务器宕机或是网络瘫痪，代理软件依然会在监控对象服务器内部进行监控，当与系统监控服务器的网络连接恢复后再向其发送消息。因此，该模式适用于系统中重要性较高，需要实时掌握其运行情况的设备。

⁶ 这里是指向监控对象服务器查询是否有需要向系统监控服务器发送的数据。

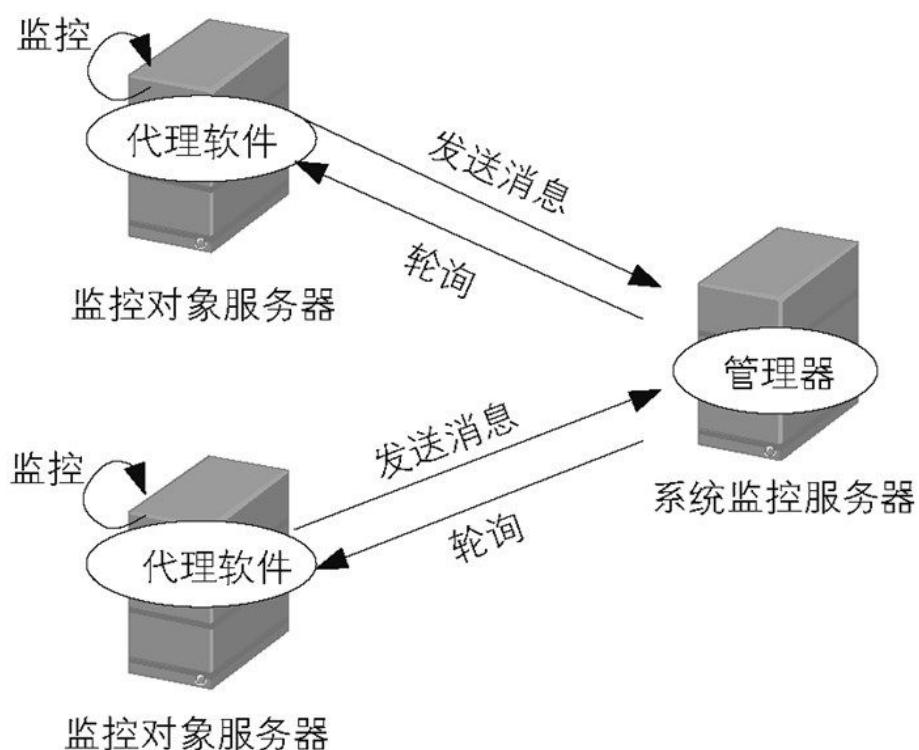


图 5.9 代理监控模式

资源信息保存监控模式

在监控对象服务器上安装代理软件以定时保存资源信息（图 5.10）。必要时可以通过系统监控服务器访问这些资源信息。

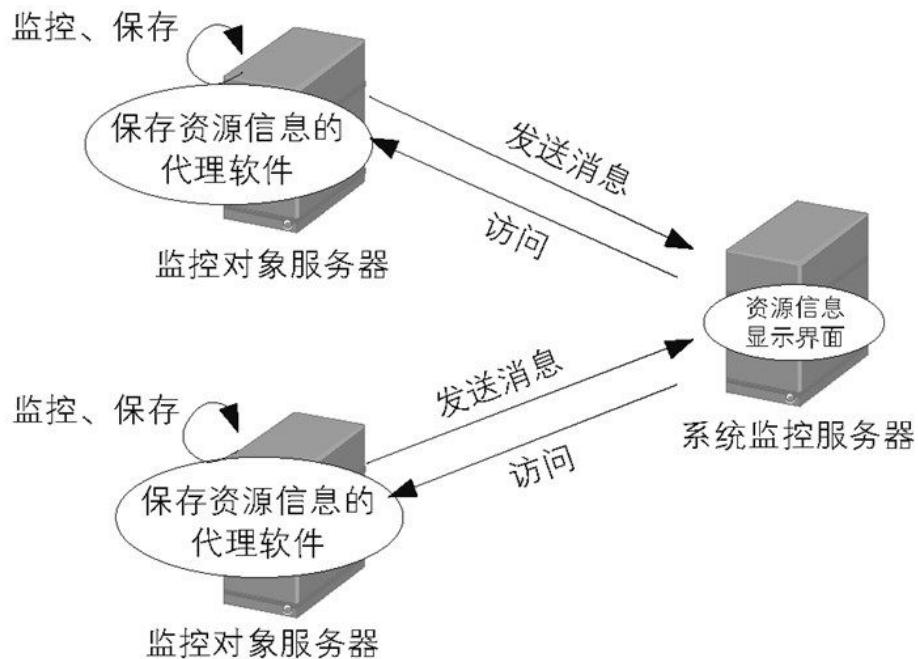


图 5.10 资源信息保存监控模式

无代理监控模式

不在监控服务器上安装代理软件，而是采用标准协议来检测监控对象服务器是否发生故障（图 5.11）。与代理软件监控模式相比，可监控的项目较少，且系统监控服务器宕机或是网络瘫痪后无法监控，因此适用于系统中重要性不高的设备。

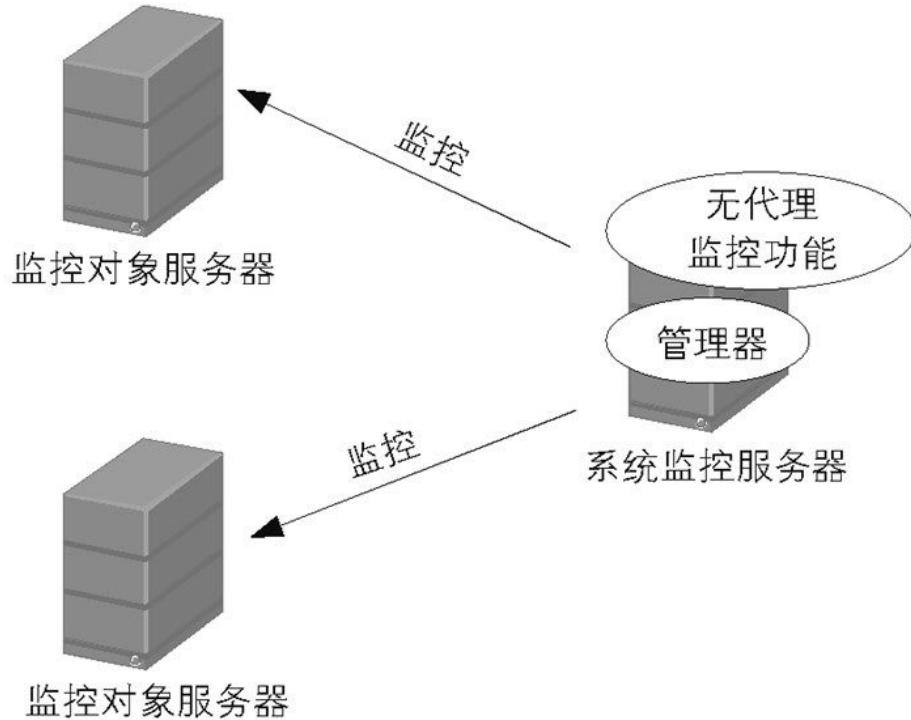


图 5.11 无代理监控模式

各模式的比较结果与选择标准

我们将各种模式的比较结果与选择标准分别总结在表 5.14 和表 5.15 中。

表 5.14 系统监控的设计方式的比较结果

比较项目	可用监控	代理监控	资源信息保存监控	无代理监控

比较项目	可用监控	代理监控	资源信息保存监控	无代理监控
监控方式	监控管理器通过Ping和SNMP来监控服务器和网络设备是否可用	在各监控对象服务器上安装代理软件进行监控，发生故障时向管理器发送消息。不保存性能信息和资源信息	在各监控对象服务器上安装可保存资源信息的代理软件来进行监控和保存资源信息	监控管理器通过标准协议来远程监控服务器的运行状况
监控项目	△：少 只能监控设备是否可用※1	◎：多 可以获取监控对象的所有运行信息	△：有限 仅可监控资源信息	○：多 监控管理器产品可能无法监控部分运行信息
资源信息的保存	✗：不可 不可保存	✗：不可 也有部分监控管理器产品可以保存	◎：可 由于是在监控对象服务器上进行监控和保存，系统监控服务器宕机或是网络瘫痪时，也可进行监控和保存工作	✗：不可 也有部分监控管理器产品可以保存

※1 除了设备的启动停止外，还想监控更多项目的时候，推荐代理监控模式或是无代理监控模式

表 5.15 系统监控的设计方式的选择标准

模式	监控对象设备的重要度	成本

模式	监控对象设备的重要度	成本
可用监控	低 适用于设备停止也不会导致系统停止的情况	低 仅需监控管理器的授权费用
代理监控	中高 适用于需要365天24小时监控的情况	中 每套代理软件的授权费用
资源信息保存监控	中高 适用于需要保存和分析资源信息的情况	高 和代理监控模式组合使用
无代理监控	中低 适用于系统维护时，允许监控中断的情况	低 与引入代理监控模式相比，授权费用较低，因此适用于服务器数量很多的情况

注意点

监控服务器的冗余与分层

如果监控管理器停止运行就会导致无法掌握系统整体的运行情况，因此需要考虑监控服务器的冗余和分层。特别是综合监控多个系统的时候，需要具备很高的可用性。

考虑监控网络

系统监控的通信流量虽然很小，但是总是在发生。因此，推荐将系统服务的网络与监控网络分离开来。此外，还需要考虑设置防火墙等策略，以防止在通往 DMZ 的监控路径中产生安全漏洞。

考虑监控消息

监控消息的过滤与阈值的优化是非常重要的。如果向运维人员发送了大量的无用信息，可能会导致运维人员漏掉重要信息和增加运维成本。建议在基础设施测试阶段和系统测试阶段也要恰当提取和设置错误提示信息。

考虑通知方法

考虑如何将检测到的事象通知给运维人员。常用方法有在运维监视终端的画面显示、旋转灯的鸣叫和发送邮件等。同时，还需要考虑运维体制和发生故障时的运维流程。

考虑其他注意事项

与运维相关的其他设计，包括运维手册（其中记载了确认通知消息的方法等）的设计、保存的处理数据的汇总、分析和向客户的报告的设计、数据的导出和备份的设计等，这些设计与其实现方式也是需要慎重考虑的。

5.5 任务管理的设计方式

系统中有着各种任务，有每天运行一次的、每周运行一次的、每月运行一次的以及每年运行一次的。它们的运行条件也各不相同，有人为决定是否运行的任务，有像“每个月最后一个工作的 22 点启动”这样按日期时间运行的任务，也有满足“当任务 A 运行完成后再启动”等特定条件时才运行的任务。

如果不对这些任务加以管理，理应顺序执行的任务的开始时间就可能会发生误差，导致数据紊乱，影响整体服务的质量。因此，需要根据任务的特点和内容，考虑任务的管理方法。

任务的管理方法需要从是否定期启动、处理的数据量、任务之间的交互等角度进行综合考虑。

专业任务管理工具模式

使用专业任务管理工具进行任务管理（图 5.12）。适合于系统重要性高，且需要谨慎管理各项任务的情况。此外，该模式还适用于综合管理大型系统以及统一管理多个系统的情况。

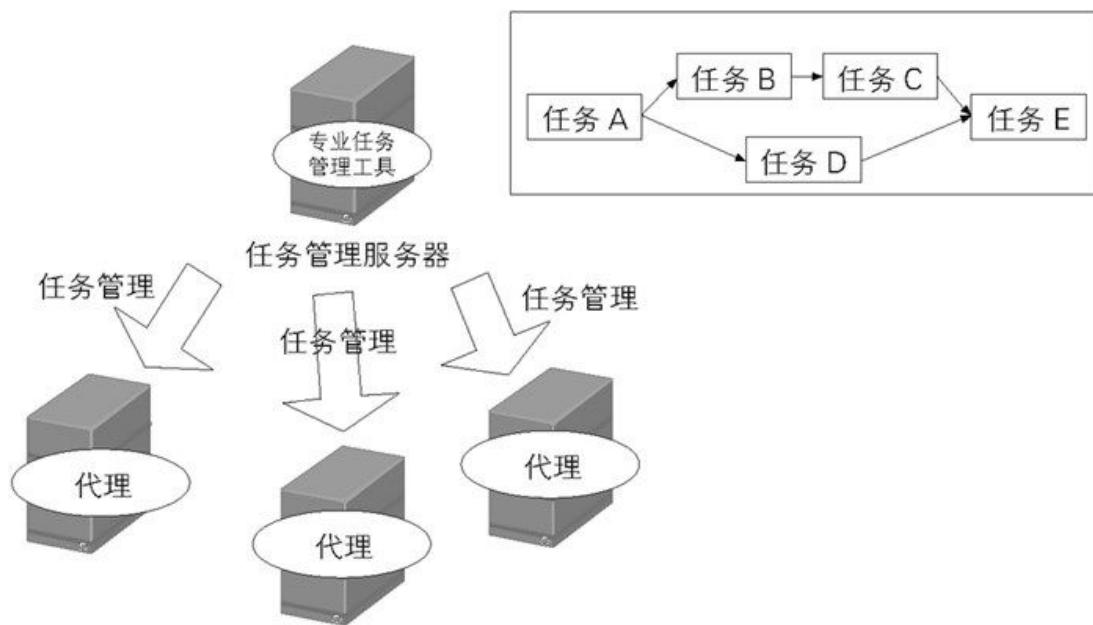


图 5.12 专业任务管理工具模式

附带任务管理功能模式

通过 DB 和集群软件等中间件的附带功能来进行任务管理（图 5.13）。可考虑在比较重要、但不需要统一管理任务的小型系统中采用该模式。



图 5.13 附带任务管理功能模式

OS 任务管理功能模式

通过 UNIX 的 Cron 等 OS 附带功能来进行任务管理（图 5.14）。如果要想使用该模式来管理大量任务以及任务之间的交互，需要编写复杂的代码和 Shell 脚本。因此，该模式适合于需要管理的任务少，且任务之间交互比较简单的情况。对于重要性不高，仅需进行简单任务管理的系统可考虑采用该模式。

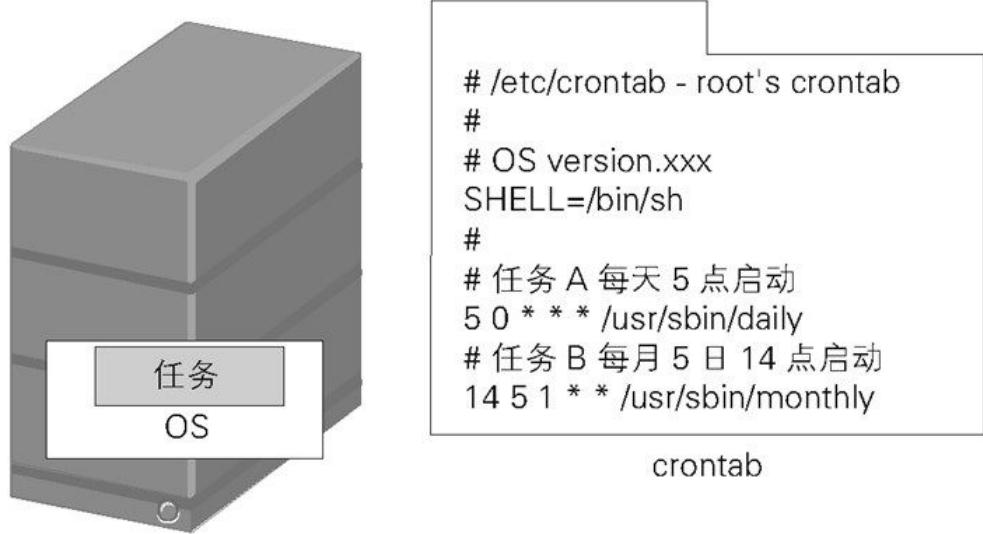


图 5.14 OS 任务管理功能模式

手动任务管理模式

手动管理任务，并根据需要由相关人员手动启动任务（图 5.15）。如果无法采用以上三种模式，可以考虑采用该模式。

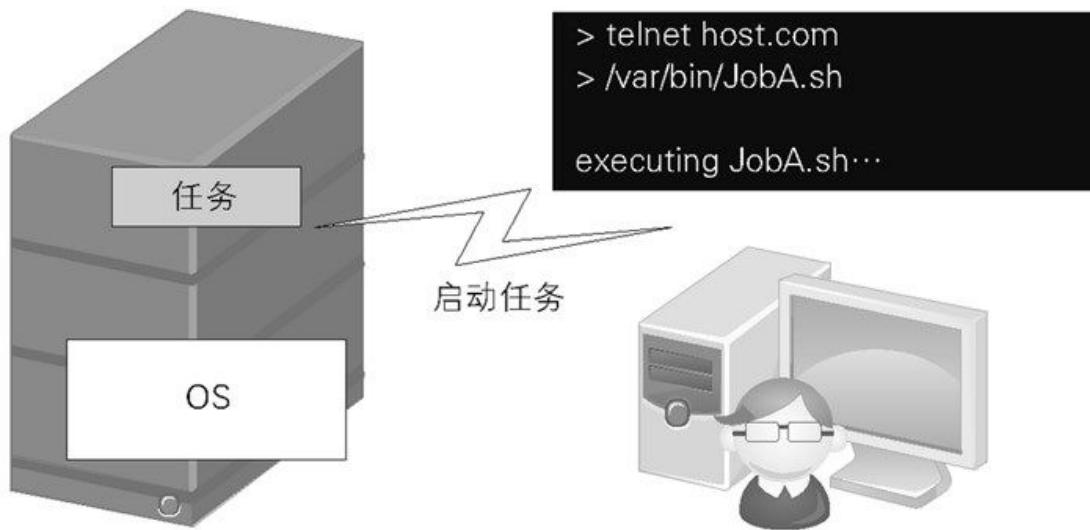


图 5.15 手动任务管理模式

各模式的比较结果与选择标准

我们将各模式的比较结果和选择标准分别总结在表 5.16 和表 5.17 中。

表 5.16 任务管理的设计方式的比较结果

比较项目	专业任务管理工具	附带任务管理功能	OS任务管理功能	手动任务管理
定期启动	○： 容易 以OS的时间为标准，按照既定的计划启动任务			×： 困难 操作员手动启动
可管理的任务量	○： 多 使用中间件管理任务，可以管理很多任务		△： 少 通过操作员的作业管理任务，可管理的任务数量少	
系统综合管理·监控	○： 容易 一部分工具中包含本功能，容易进行综合管理和监控		△： 困难 各任务都是单独管理，难以综合管理	
任务的交互	○： 容易 任务的交互是专业任务管理工具的标准功能		△： 困难 编写代码或Shell脚本才能实现任务交互	△： 困难 操作员手动实现任务交互

表 5.17 任务管理的设计方式的选择标准

模式	系统的重要性	任务综合监控
专业任务管理工具	高 适用于支付系统和财务系统等非常重要的系统	有 适用于需要统一管理多个系统的任务的情况
附带任务管理功能	中等 适用于各个部门内部的分析工具等具有一定重要性的系统	无 适用于所需管理的任务数少，不需要综合监控的情况
OS任务管理功能	低 适用于部门内部的企业网等不太重要的系统	
手动任务管理	— 适用于无法采用以上三种模式的情况	

注意点

管理大量任务时的注意事项

在管理大量任务时请考虑采用专业任务管理工具模式。如果同时采用附带任务管理功能模式或任务管理功能模式，就无法统一管理任务，会导致任务管理运维作业变得复杂，容易出错。

根据需求选择任务管理工具

如果选择专业任务管理工具模式，需要注意不同的工具可管理的任务数量不同。如果所需管理的任务数接近工具所允许的上限值，会导致管理工具性能下降。

如果任务集规模较大，或是任务之间依赖关系复杂、任务流程复杂，需要根据需求选择合适的任务管理工具。

5.6 时钟同步、杀毒软件更新的设计方式

构成系统的服务器和网络设备都管理着它们各自内部的时间信息。如果各设备的时间信息不一致，可能会导致在各设备之间进行交互的批处理程序无法按时运行，从而引发问题和故障。此外，当发生故障时，也很难根据各设备的日志文件中的时间信息来定位故障原因。因此，系统的时钟同步的设计非常重要。

同样，定期更新系统中安装的杀毒软件的病毒库文件也很重要。如果疏于更新病毒库文件，则无法防止和隔离病毒变种的新病毒，导致系统感染计算机病毒的风险增加。

但是，时钟同步和杀毒软件更新都需要从系统外部（互联网等）获取信息，所以在考虑如何获取这些信息时就需要谨慎对待了。

后端网络自动同步模式

在后端额外搭建一套网络用于从外部官方机构、杀毒软件供应商获取正确的时钟信息、最新的病毒库文件，并将它们发送给内部服务器以及网络设备（图 5.16）。

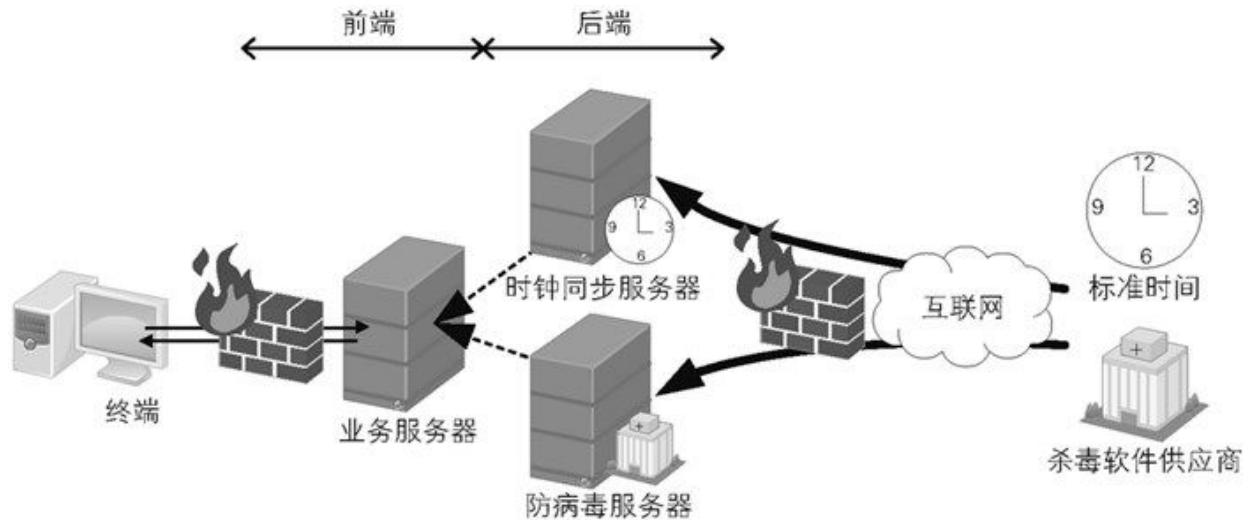


图 5.16 后端网络自动同步模式

通过这套后端网络，可将业务服务器所处的网络与时钟同步服务器和防病毒服务器所处的网络隔离开来，确保系统的高安全性。

前端网络自动同步模式

通过业务服务器所处的网络（前端）来获取正确的时钟信息以及最新的病毒库文件，并将它们发送给内部服务器以及网络设备（图 5.17）。

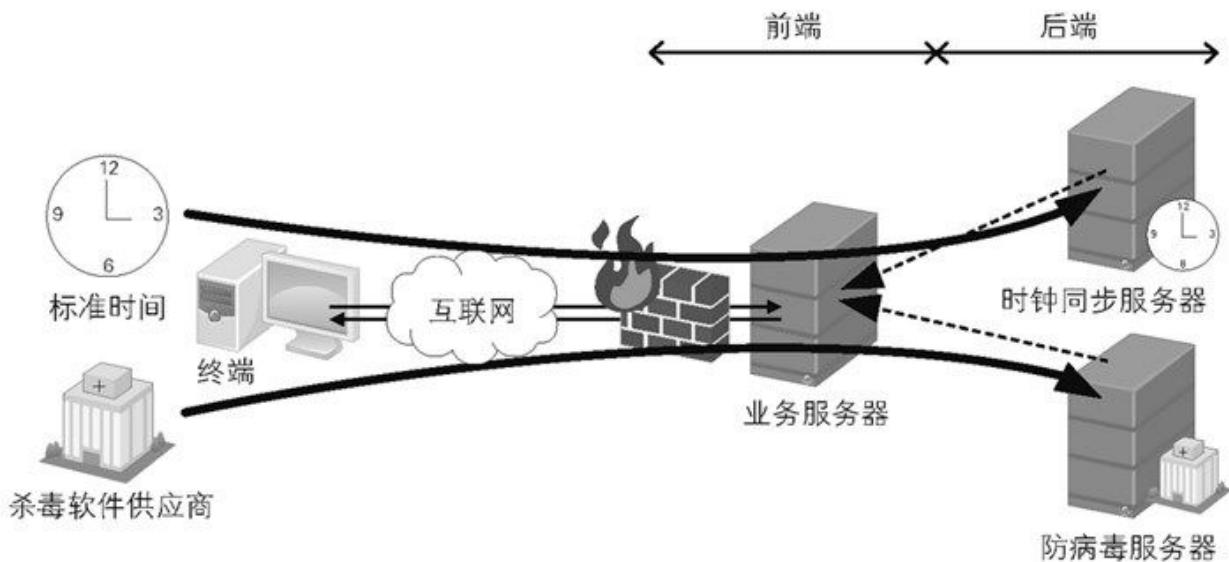


图 5.17 前端网络自动同步模式

与后端网络自动同步模式不同，时钟同步服务器和防病毒服务器与业务服务器处于同一网络中。因此，该模式适用于不需要确保高安全性的系统。

手动同步模式

系统管理员手动检查和更改时钟同步服务器的时间以及更新杀毒软件的病毒库文件（图 5.18）。与前两种模式相比，同步和更新的即时性以及正确性都较低。此外，系统管理员将获取的病毒库文件通过 OA 终端或是其他媒体复制给内部服务器和网络设备，安全性也较低。

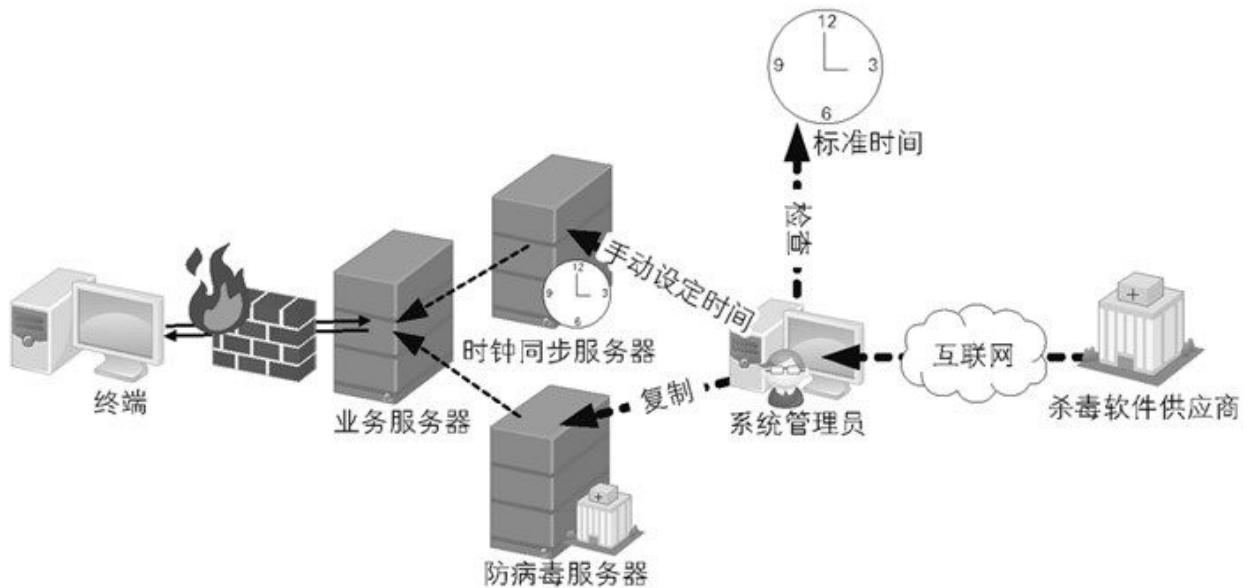


图 5.18 手动同步模式

各模式的比较结果与选择标准

我们将各模式的比较结果和选择标准分别总结在表 5.18 和表 5.19 中。

表 5.18 时钟同步、杀毒软件更新的设计方式的比较结果

比较项目	后端网络自动同步	前端网络自动同步	手动同步
可靠性 (即时性、正确性)	○：高 可以实现自动在适合的时间点获取正确的标准时间信息和最新的病毒库文件		✗：低 需要系统管理员手动修改时间和获取最新的病毒库文件，因此在即时性、正确性方面不如自动同步方式
安全性	○：高 为获取标准时间信息和最新的病毒库文件搭建专用网络，并与前端网络隔离开来，因此安全性很高	△：稍低 与使用专用网络获取标准时间信息和最新病毒库文件相比，在防火墙的设置上出现错误等导致安全事故的风险较高	✗：低 需要在OA终端上拷贝病毒库文件，因此安全性很低

成本 (初期成本、运维成本)	x: 高 由于搭建了专用网络，需要额外的网络设备和联网线路的费用。运维上虽然是自动更新，不需要人力成本，但是需要维护和保养网络设备	○: 低 共用网络设备和联网线路可以减少初期成本。自动更新也不需要人力成本，因此运维成本也较低	△: 稍高 不需要使用其他设备，可以减少初期成本，但是运维需要人力成本
-------------------	----------------------------------------------------------------------	----------------------------------------------------	----------------------------------------

表 5.19 时钟同步、杀毒软件更新的设计方式的选择标准

模式	业务服务器和网络的连接	初期成本
后端 网络 自动 同步	不可 为了处理保密信息需要确保高安全性，因此适用于不允许业务服务器直接连接互联网的情况。但是时钟同步、防病毒服务器可以连接互联网	高 适用于有足够的预算准备专用网络的情况
前端 网络 自动 同步	可 适用于Web系统和其他在互联网上对外公开的系统等业务服务器可与互联网直接连接的情况	低 适用于无法花费太高初期成本的情况
手动 同步	不可 需要确保高安全性以处理保密信息，因此适用于不允许业务服务器直接连接互联网的情况	低 适用于无法花费太高初期成本的情况，但是会发生运维成本

注意点

时钟同步时通常使用 NTP (Network Time Protocol, 网络时间协议)。

也有使用电子表、GPS (Global Positioning System, 全球定位系统)、电话线路等进行时钟同步的专用设备⁷。时钟同步专用设备与手

动同步模式一样，适用于业务服务器无法与互联网直接连接，需要处理保密信息的情况。

⁷ 具有特定功能或进行特定处理的计算机。

5.7 总结

本章讲解了运用与维护性策略的基本思考方法和设计方式。运用与维护性策略对系统的可用性和安全措施等其他需求都有影响，因此为了能使系统安全、稳定地运行，必须仔细考虑运用与维护性策略。

第 6 章 基础设施构成的设计方式

第 2 章至第 5 章讲解了主要的非功能性需求（可用性、安全性、性能与可扩展性、运用与维护性）的策略和设计方式。在构建基础设施时，除了非功能性需求策略外，还需要考虑基础设施的构成要素。不过要注意的是，需要结合多个非功能性需求来选择合适的基础设施构成要素。

在本章中，我们将讲解基础设施的设计方式。

6.1 Web 系统的网络构成的设计方式

在网络基础设施（LAN）中，需要根据网络安全需求以及网络通信流量对网络进行分段。

系统内部的网络流通着业务信息和个人信息、监控信息和备份信息等各种不同保密级别的信息。如果让它们在同一网络中传输，运维人员就有可能看到业务信息，导致信息泄露。因此，建议根据各种信息的用途和保密级别将网络分段，提高网络安全性。

此外，网络的通信量有限（带宽），如果在某一网段中的通信量到达极限会导致整个网络通信发生延迟。因此，如果网络构成设计不好，在发生大量数据通信时会导致系统处理变慢，或因通信超时而引起处理中断，给整个系统带来严重影响。例如，由于在备份过程中会发生大量的数据通信，因此不应该让备份过程与其他大量数据通信或是重要的数据通信在同一网段中同时进行。可通过分离网段使得监控信息和备份等需要大量数据通信的处理不会对系统服务造成影响。

安全性与网络通信流量并非是对立的，通过组合各种模式可以构建出同时满足两方面需求的系统。

接下来，我们将网络根据其用途分为 Web 处理网段、AP 处理网段、DB 处理网段、运维监控处理网段分别进行讲解。

4 网段构成模式

按照功能将网络划分成 4 个网段：Web 处理网段、AP 处理网段、DB 处理网段和运维监控处理网段（图 6.1）。与之后的 5 网段模式不同的是，该模式没有 DB 备份处理网段。因此，对于可以在提供服务时间段以外进行备份的系统，或是备份处理不通过 LAN 通信的系统，以及 DB 的备份处理负荷很小的系统，可以考虑采用该模式。

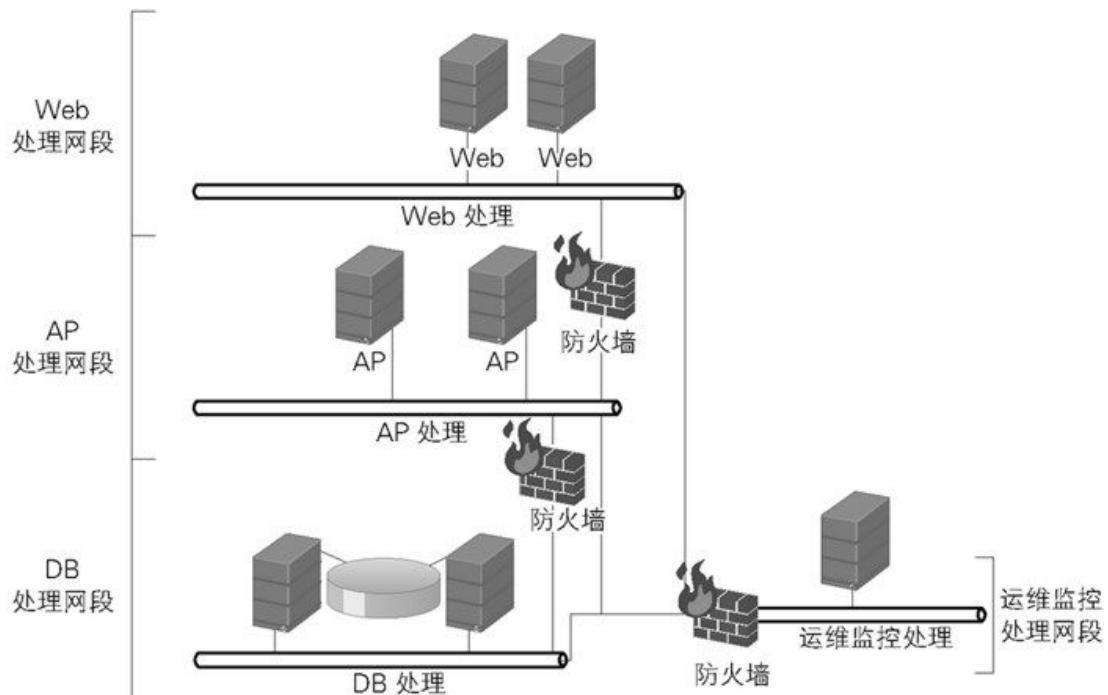


图 6.1 4 网段构成模式

该模式将提供 Web 处理、AP 处理、DB 处理服务的三个功能分成不同的服务器群（Web 服务器群，AP 服务器群、DB 服务器群），并称之为 Web 三层结构。

Web 三层结构中，可以根据性能需求的变化，对每层进行适当的扩展（有时还需要对各层进行移动），以此来维持服务器资源能够得到高效利用的系统构成。但该模式下即使是最精简的构成也需要多台服务器，因此适用于大中型系统。

此外，与外部网络相连接的 Web 层与程序所处的 AP 层、保密信息所处的 DB 层相分离，且各层还设置了防火墙，能够多重保护数据安全，因此提高了系统的安全性¹。

¹ 关于设置了防火墙的系统构成，请参见第 2 章。

3 网段构成模式

该模式将网络划分成 3 网段：Web/AP 处理网段、DB 处理网段和运维监控处理网段（图 6.2）。Web/AP 网段虽与外部网络相连接，但与保存顾客信息和业务信息的 DB 网段相分离。

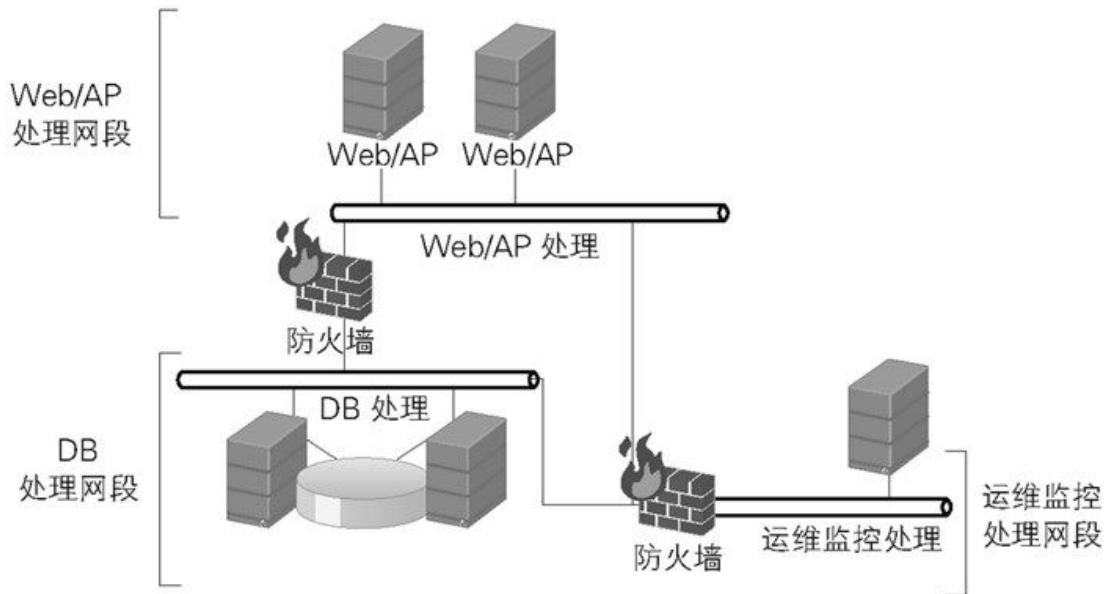


图 6.2 3 网段构成模式

该模式将提供服务的三种功能（Web、AP、DB）分为两个服务器群：Web/AP 服务器群和 DB 服务器群。前者负责接受用户的访问，并与 DB 交互，后者则负责保存数据。整体称为 Web 双层结构。

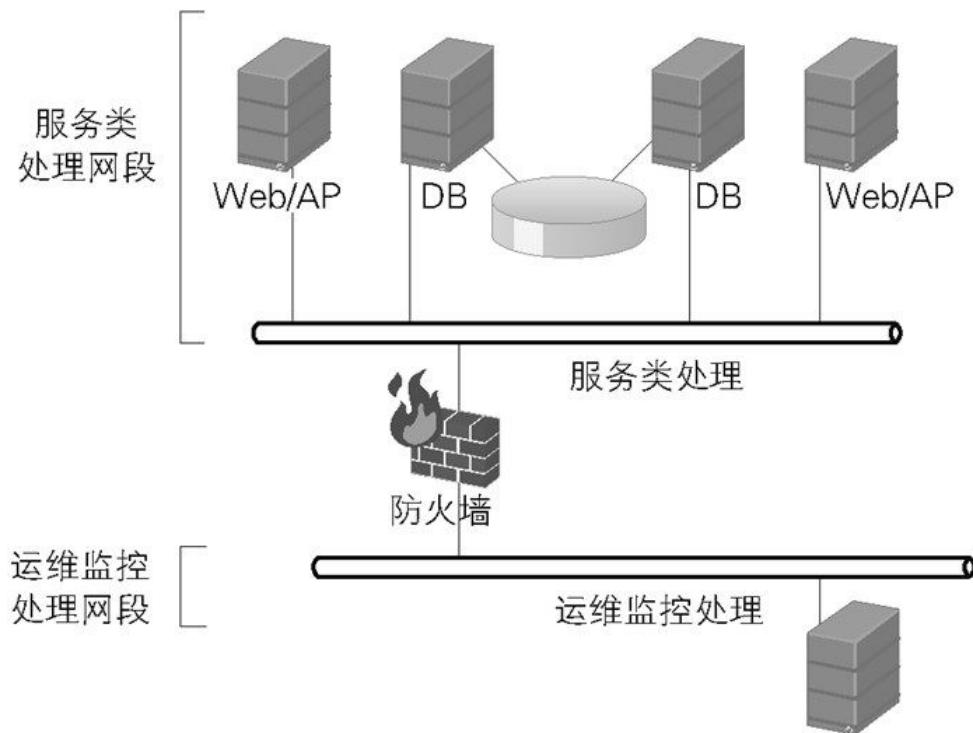
与 Web 三层结构不同，接受用户的访问的 Web 层和实际处理用户请求，并与 DB 交互的 AP 层被合并为一层。但由于是在 DB 层进行检索、更新、删除等处理，所以性能上并不会下降。

通过将保存顾客信息和业务信息的 DB 层分离，可在 Web/AP 层和 DB 层之间设置防火墙，提高系统的安全性。

与 4 网段构成模式相比，Web/AP 功能被部署在同一台服务器上，无法单独扩展 Web 功能或是 AP 功能，可扩展性较低。

2 网段构成模式

将 4 网段构成模式和 3 网段构成模式中分离出来的 Web 处理网段、AP 处理网段、DB 处理网段统一为一个网段，仅与运维监控处理网段分离开来（图 6.3）。由于 DB 处理网段也直接与外部网络相连接，因此与其他模式相比，顾客信息和业务信息的泄露风险较高。此外，除运维监控外，其他服务都在同一网段中通信，因此该模式并不适用于通信数据量很大的大型系统，可考虑在一天内只有少量用户数次访问的小系统中采用该模式。



图中为 Web/AP 服务器群和 DB 服务器群的 Web 双层结构配置示意图。也可以配置成 Web 服务器群、AP 服务器群、DB 服务器群的 Web 三层结构，或是将 Web/AP 功能和 DB 功能全部部署在一台服务器上

图 6.3 2 网段构成模式

5 网段构成模式

与 4 网段构成模式相比，除了 Web 处理网段、AP 处理网段、DB 处理网段、运维监控处理网段以外，还增设了 DB 备份处理网段（图

6.4)。当某个网段内的通信数据量突然增大时 (DB 备份等通信负载很高的处理), 不会对其他网段产生影响, 这样就可以将突发数据量的影响范围缩至最小。

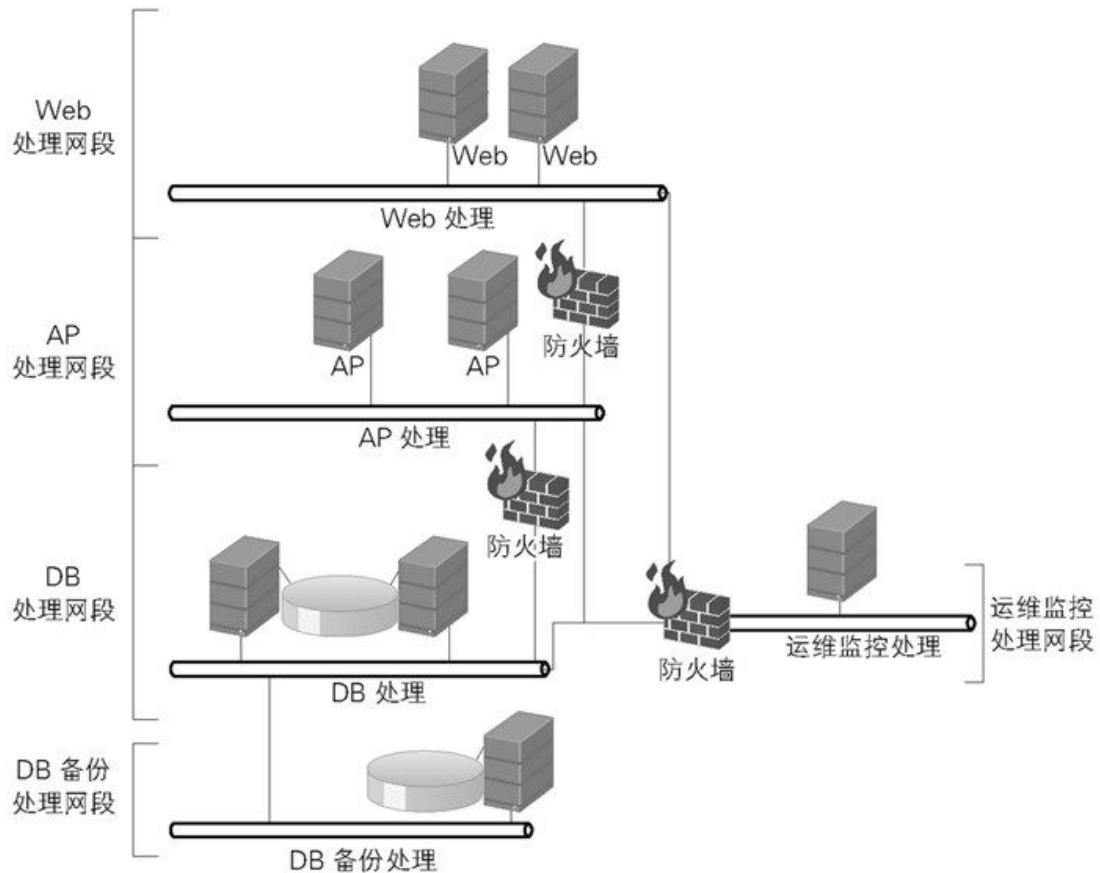


图 6.4 5 网段构成模式

各模式的比较结果和选择标准

4 网段构成模式、3 网段构成模式、2 网段构成模式都是从安全性级别角度来划分网段, 而 5 网段构成模式则是从网络数据通信的角度来划分网段。我们按照划分网段的角度不同, 将各模式的比较结果总结在表 6.1、表 6.2 中, 将各模式的选择标准总结在表 6.3、表 6.4 中。

表 6.1 Web 系统网络构成的设计方式的比较结果 (从安全性级别角度划分网段)

比较项目	4网段构成	3网段构成	2网段构成
该模式中的网段	将网络分为4个网段 Web处理网段 AP处理网段 DB处理网段 运维监控处理网段	将网络分为3个网段 Web/AP处理网段 DB处理网段 运维监控处理网段	将网络分为2个网段 服务类处理网段 运维监控处理网段
安全性级别	○：高 从外部网络访问DB处理网段需要经过Web处理网段和AP处理网段两个网段，具有高安全性	△：中 从外部网络访问DB处理网段需要经过Web/AP处理网段，具有一定的安全性	×：低 从外部网络可直接访问包含DB在内的几个网段，安全性较低
可运维性	△：中	○：易	◎：易 网段少，网络的设置和管理都很容易

表 6.2 Web 系统网络构成的设计方式的比较结果（从数据通信角度划分网段）

比较项目	5网段构成	4网段构成（对比）
------	-------	-----------

比较项目	5网段构成	4网段构成（对比）
该模式中的网段	将网络分为5个网段 <ul style="list-style-type: none"> ● Web处理网段 ● AP处理网段 ● DB处理网段 ● DB备份处理网段 ● 运维监控处理网段 	将网络分为4个网段 <ul style="list-style-type: none"> ● Web处理网段 ● AP处理网段 ● DB处理网段 ● 运维监控处理网段
性能	○：高 除了4网段构成模式中的网段外，还搭建了DB备份专用网段。这样DB备份时的数据通信不会对其他网段的数据通信造成影响	△：中 按照功能将网络分为Web处理网段、AP处理网段、DB处理网段和运维监控处理网段，以此将它们之间的数据通信分隔开
可运维性	✗：难 5网段的运维需要专业的网络运维知识。但是当发生故障时很容易判断故障发生在哪个网段	△：中

表 6.3 Web 系统网络构成的设计方式的选择标准（从安全性级别角度划分网段）

模式	安全性	系统例
4网段构成	高 适用于系统处理个人信息和高保密信息，需要具有高安全性的情况	处理顾客信息的EC站点等

模式	安全性	系统例
3网段构成	中 适用于系统中存在需要保护的信息，需要确保一定安全性的情况	进行企业介绍的Web系统等
2网段构成	低 适用于允许运维管理员查看系统内的信息等不需要高安全性的情况	无保密信息的移动电话站点等

表 6.4 Web 系统网络构成的设计方式的选择标准（从数据通信角度划分网段）

模式	通信数据量	系统例
5网段构成	DB备份数据量大 适用于备份大量数据的系统	(拥有大量数据的) 银行的主干系统等
4网段构成 (对比)	中 适用于系统处理个人信息和高保密信息，需要具有高安全性的情况	处理顾客信息的EC站点等

注意点

考虑网络使用率

LAN 中服务器数量的增加会导致通信数据量增加，通信性能下降，出现通信错误等情况。想要避免此问题，需要以高负荷下系统的网络使用率为电缆传输速度的 50% (平均 15 分钟) 或是 20% ~ 30% (平均 1 小时) 左右为标准设计 LAN。

主要的网络电缆种类与其传输速度和可延伸距离参见表 6.5。

表 6.5 主要的网络电缆的传输速度和可延伸距离

比较项目	10BASE-T	100BASE-TX	1000BASE-T	1000BASE-SX、1000BASE-LX	10GBASE-R面向LAN的规范	10GBASE-R
使用的电缆	双绞线 (cat3以上) *1	双绞线 (cat5)	双绞线 (cat5、cat5e、cat6)	光纤 (SX、LX) *2	光纤 (SR、LR、ER) *3	双绞线 (cat6e、cat7)
传输速度	10Mbit/s	100Mbit/s	1Gbit/s		10Gbit/s	
可延伸距离	100m	100m (推荐90m)	100m	SX: 550m LX (SMF) : 5km LX (MMF) : 550m	SR (MMF) : 300m LR (SMF) : 10km ER (SMF) : 40km	cat6e: 55m cat7: 100m

*1 将两根绝缘的铜导线互相绞在一起。双绞线按照传输速度分为 cat3、cat6、cat6a 等

*2 SX (Short eXchange) 、LX (Long eXchange) 分别是 1000BASE-SX、1000BASE-LX 的简写，符合 IEEE802.3z 千兆以太网规范。SX 的最大传输距离为 550m，LX 的最大传输距离为 5km

*3 SR (Short Reach) 、LR (Long Reach) 、ER (Extended Reach) 分别是 10GBASE-SR、10GBASE-LR、10GBASE-ER 的简写，符合 IEEE802.3ae 千兆以太网规范。SR 可以传输 300m，LR 可以传输 10km，ER 可以传输 40km

使用 L3 交换机划分子网

在 LAN 设计中考虑划分子网，并管理通信范围，这样既能有效利用网络带宽，还可提高安全性。

使用具备镜像端口的交换机

使用镜像端口²可以在不增加监控对象通信负荷的情况下进行监控工作。因此，在设计 LAN 时可以考虑使用具备镜像端口的交换机。

² 指地是用于接收和发送与指定端口所接收和发送内容完全相同内容的端口。用于端口监控。

考虑 IP 地址的可扩展性

在分割子网时，如果无法为增加的服务器分配 IP 地址则需要重新分割子网，重新考虑和设置网络构成，从而导致成本增加。因此，需要考虑 IP 地址的可扩展性充分分割子网。

考虑服务器 NIC 多路径（组合）

通过服务器 NIC 组合³，即使 NIC 发生故障，也可以继续保持通信。

³ 将多个物理网卡绑定成一个虚拟网卡，可分散网络通信流量以及在故障时自动切换网卡，提高了可用性。

其他注意事项

在设计网络时，除了考虑本节介绍的系统内部网络划分以外，还需要考虑与外部进行通信的 DNS 服务器、邮件服务器、NTP 服务器的通信数据量与通信内容来进行分段设计。

6.2 存储设备构成的设计方式

存储设备是指 DB 和 Office 文件服务器等保存企业和组织重要信息的设备。

存储设备与服务器、终端的连接方式和形态多种多样，而且对业务会造成何种影响、是否有多台服务器和终端使用存储设备、使用的服务器和终端数量等都会影响存储设备的设计方式。如果没有选择合适的存储设备连接方式，那么访问存储设备就会对业务造成影响。如果没有选择合适的存储设备连接形态，那么在增加新的服务器和终端时，就不得不对系统进行改造。

SAN 模式

搭建存储设备专用网络，使多台服务器可以使用存储设备的数据（图 6.5）。为了实现高速通信，一般都会使用光纤通道（Fibre Channel）技术。由于搭建了存储设备专用网络，因此该模式适用于存储过程中需要处理大量数据，且不能对其他业务通信产生影响的情况。

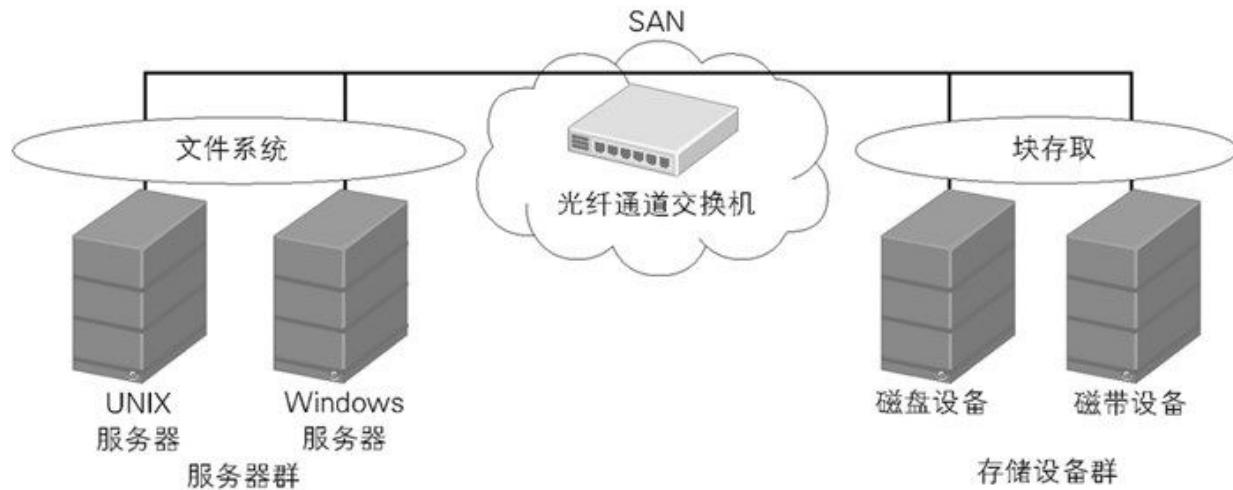
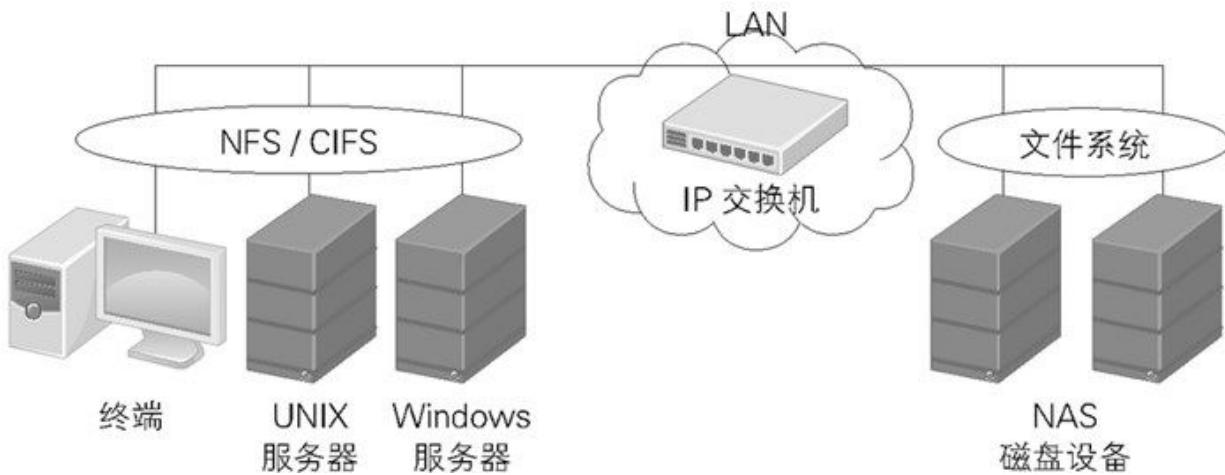


图 6.5 SAN 模式

NAS 模式

使用 LAN 与存储设备进行数据通信（图 6.6）。既可以与业务服务器和网络设备共用 LAN，也可以搭建专用 LAN。可实现多台服务器和终端访问存储设备中的数据。与 SAN 模式相比，当与业务服务器和网络设备共用 LAN 时，会对业务的数据通信造成影响。



NFS (Network File System, 网络文件系统)、CIFS (Common Internet File System, 通用互联网文件系统) 是访问 NAS 上的共享文件时使用的协议，通过使用这些协议，可以像访问本地磁盘中的文件一样访问 NAS 上的文件。NFS 多用于 UNIX 系统和类 UNIX 系统，CIFS 多用于 Windows 系统。

图 6.6 NAS 模式

DAS 模式

使用 DAS (Direct Attached Storage, 直接连接存储)⁴ 连接特定的服务器和存储设备（图 6.7）。使用该模式时，其他服务器无法使用存储设备中的数据，因此该模式适用于只有与存储设备相连接的服务器可以访问存储设备中的数据的情况。

⁴ SAN 和 NAS 是通过网络连接服务器与存储设备，DAS 则是直接连接服务器与存储设备。



图 6.7 DAS 模式

存储设备连接形态的类型

关于存储设备的连接形态，可以采用以下类型或者它们的组合。

P2P 方式

将存储设备与服务器、终端直接连接，进行 P2P（Peer to Peer，端对端）⁵ 通信（图 6.8）。由于独享连接路径，可以确保高速通信，但是所连接的服务器和终端数量受制于存储设备上的端口数量，因此该连接类型不适用于需要将存储设备与许多服务器和终端进行连接的情况。

⁵ 在网络上各终端之间相互直接连接并进行数据通信的方式。

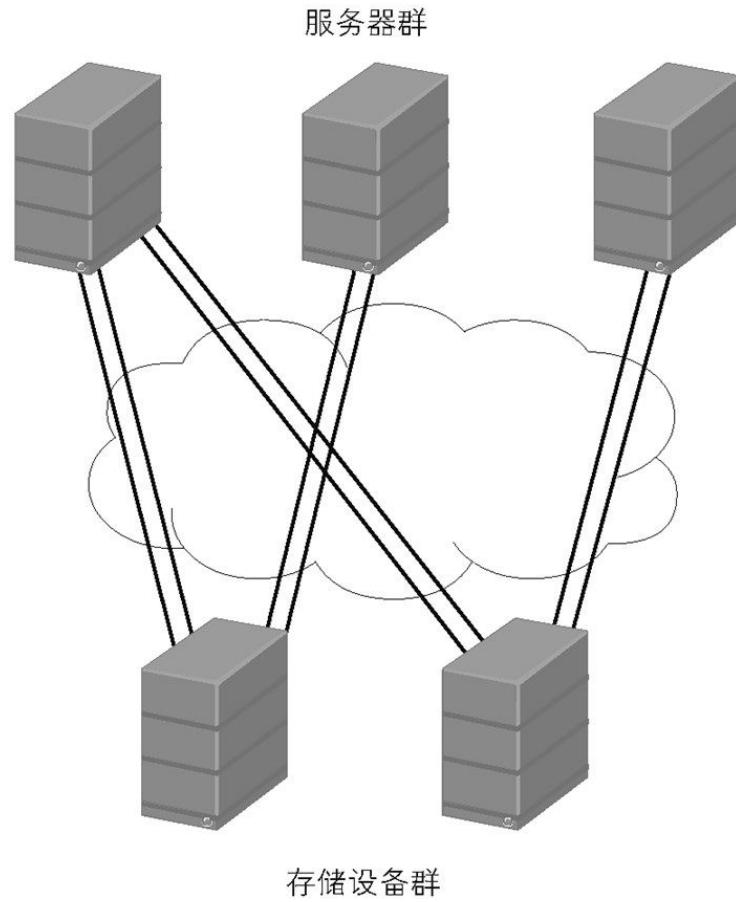


图 6.8 P2P 类型

集线器类型

通过将存储设备连接到交换机上，使得多台服务器和终端都可以访问存储设备（图 6.9）。适用于计划将来访问存储设备的服务器和终端数量会有所增加的情况。

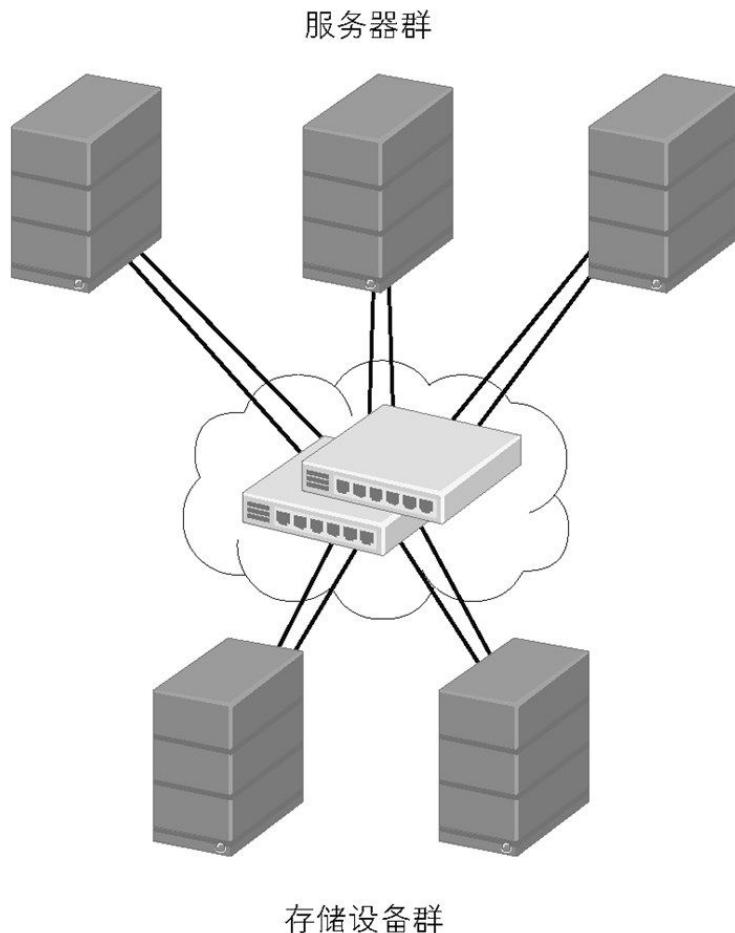


图 6.9 集线器类型

连接形态类型的注意点

我们将各种连接形态类型的比较结果与选择标准分别总结在表 6.6 和表 6.7 中。

表 6.6 存储设备连接形态类型的比较结果

比较项目	P2P	集线器

比较项目	P2P	集线器
性能	<ul style="list-style-type: none"> ○: 高 独占与各连接对象的连接线路，不受其他连接影响，可以确保最高性能 	<ul style="list-style-type: none"> ×: 低 由于通过交换机与其他连接对象共享连接线路，因此交换机容易成为性能瓶颈^{*1}
可扩展性	<ul style="list-style-type: none"> ×: 低 需要通过端口与服务器和终端连接，因此可以连接的服务器和终端的数量受制于设备上的端口数量 	<ul style="list-style-type: none"> ○: 高 可以以最小的端口数量连接许多服务器和终端，因此很容易增加服务器和终端，可扩展性高

*1 例如使用交换容量为 8Gbps 的交换机连接服务器和存储设备，当它们之间的通信数据量大于 8Gbps 时则无法通信

表 6.7 存储设备连接形态类型的选择标准

方式	性能	连接存储设备的服务器和终端数量
P2P	<p>高 适用于存储设备与服务器、终端之间有大量数据交互，且能够应对数据量激增等需要确保高性能通信的情况</p>	<p>少 适用于存储设备与DB服务器等少量特定服务器、终端连接的情况</p>

方式	性能	连接存储设备的服务器和终端数量
集线器	低 适用于存储设备与服务器和终端之间没有大量数据交互的情况。通信数据量的激增可能会导致通信性能下降，例如发生通信延迟	多（不特定） 适用于不特定的多台服务器和终端使用存储设备（如文件服务器）的情况，例如Office文件服务器等

各模式的比较结果和选择标准

我们将各模式的比较结果与选择标准分别总结在表 6.8 和表 6.9 中。

表 6.8 存储设备结构的设计方式的比较结果

比较项目	SAN	NAS	DAS
网络	○：存储设备专用 使用专用网络，不受LAN的影响，可实现高速通信	△：共享 使用LAN访问存储设备，进行备份等作业时可能会影响业务处理的性能 ^{*1}	○：存储设备专用 内置硬盘等直接连接服务器，可实现高速通信
容量的使用效率	○：高 多台服务器共享存储设备，使用效率高	○：高 多台服务器和终端共享存储设备，使用效率高	×：低 多台服务器无法共享存储设备，使用效率低

比较项目	SAN	NAS	DAS
运维性	<p>△：稍低 增加设备时需要相对专业的知识来设定SAN^{*2}</p>	<p>○：高 增加设备时只需要重新设置LAN即可，无需特别专业的知识</p>	<p>△：稍低 虽然该模式的引入很容易，但当增加服务器时，管理负担会加重</p>
备注	高端存储设备具有高性能、高可用性和高可扩展性，适用于大型系统。但是，需要确认不同种类设备之间是否可以连接	各终端可共享文件，因此适用于文件服务器	适用于服务器之间不需要共享文件的小型系统。但是可扩展性有限

*1 指的是使用业务 LAN 时的情况。如果是与业务 LAN 分离，则无此影响

*2 指的是使用光纤通道时的情况。在 TCP/IP 网络上使用 SCSI 命令（服务器与存储设备等服务器外围设备进行通信的指令）进行交互的时候，请参照 NAS 模式

表 6.9 存储设备结构的设计方式的选择标准

模式	对业务的影响	多台服务器和终端的使用
----	--------	-------------

模式	对业务的影响	多台服务器和终端的使用
SAN	无 适用于不允许访问存储设备（处理大量数据时）所带来的网络负荷对业务产生影响的情况	可 适用于需要通过多台服务器对存储设备上的数据进行增删改查的情况
NAS	有 适用于允许访问存储设备所带来的网络负荷对业务产生影响的情况	可 适用于通过多台服务器和终端对存储设备上的数据进行增删改查的情况
DAS	无 适用于不允许访问存储设备所带来的网络负荷对业务产生影响的情况	不可 适用于仅有特定的服务器和终端使用存储设备中保存的数据的情况

注意点

RAID 的种类和特点

应对磁盘故障的策略之一是 RAID。主要的 RAID 种类和特点请参见表 6.10。

表 6.10 RAID 的种类

比较项目	RAID0	RAID1	RAID1+0	RAID5
方式	将数据分散保 存在多个磁盘 中。也称为数 据分割 (Striping)	在其他磁盘上保 存相同的数据 (冗余)。也称 为镜像 (Mirroring)	RAID1和 RAID0的组 合。至少需 要4个磁盘 驱动器	将称为同位元 ^{*1} 的用于恢 复数据的编码和数据分 散保 存在多个磁盘中。至 少 需 要3个磁盘驱动器
对单	<input checked="" type="checkbox"/> 无效	<input type="radio"/> 有效		

一故障是否有有效 ^{*2}				
性能	○：高 可分散I/O	×：低 不可分散I/O	○：高 可分散I/O	○：高 ^{*3} 可分散I/O
效率	○：高 所需容量为 100%	×：低 所需容量为200%		△：稍高 所需容量为 $(n+1)/n\%$ ^{*4}
主要用途	几乎不在企业 系统中使用	系统磁盘等	适用于保存 DB日志文件 等顺序存取 文件的磁盘	适用于保存DB日志文件 等大容量磁盘

※1 在磁盘中写入 0 或 1 的序列。同位元信息表示了在多个磁盘中保存的数据之和是奇数还是偶数。当其中一个磁盘发生故障时，通过确认同位元就可以知道磁盘中写入的值是 0 还是 1。例如，发生故障以外的数据的和为偶数，而同位元为奇数的时候，就可以确定发生故障的数据为 1

※2 针对单一故障进行了可用性设计时，如果发生冗余故障，数据就会丢失。因此，为了尽量缩短发生单一故障后系统恢复到高可用性状态所需的时间，需要准备热备用磁盘（通电并待机的磁盘）和冷备用磁盘（故障发生后进行通电并待机的磁盘）。因为平时处在未通电状态，因此与热备用相比虽然切换需要花费时间，但是维护费用比较便宜）

※3 恢复单一故障时会设置同位元，部分产品的 I/O 速度可能会暂时下降

※4 n 为除去同位元的磁盘后剩余的磁盘数量

当磁盘发生故障时，尽早在存储设备机柜内的备用磁盘上恢复数据，可以减低二次故障发生的几率。

I/O 优化的产品

有些产品通过使用硬盘和 SSD (Solid State Drive, 固态硬盘)⁶ 的组合，使存储设备分层，达到优化 I/O 的目的。

⁶ 指的是使用闪存的存储设备。与硬盘不同的是，由于没有使用磁盘，所以缩短了读取数据的时间，可以实现高速 I/O。

6.3 报表生成的设计方式

社会中广泛使用着金融机关的转账申请表和移动电话的话费单、公共机构的申请表等报表，相关的业务系统每天会根据 DB 数据生成大量的报表。报表的生成是否操作简便、报表生成速度是否很快对于实现性能需求和成本需求非常重要。

报表生成中，既有适用于定期生成大量报表的方式，也有适用于仅在必要时机生成必要报表的方式，因此结合所生成报表的特点就可以优化系统性能的使用和运维。

本节将介绍报表生成的设计方式。

报表实时生成模式

当用户想要生成报表时，向报表服务器发送请求，即可实时生成报表（图 6.10）。



图 6.10 实时报表生成模式

生成大量报表会造成报表服务器负荷增大，可能会导致其他用户无法实时生成报表。因此，该模式适用于处理少量报表的情况。

报表异步生成模式

当用户需要生成报表时，向批处理服务器发送请求。批处理服务器会在指定的批处理运行时刻向报表服务器发送命令，以此来生成报表（图 6.11）。

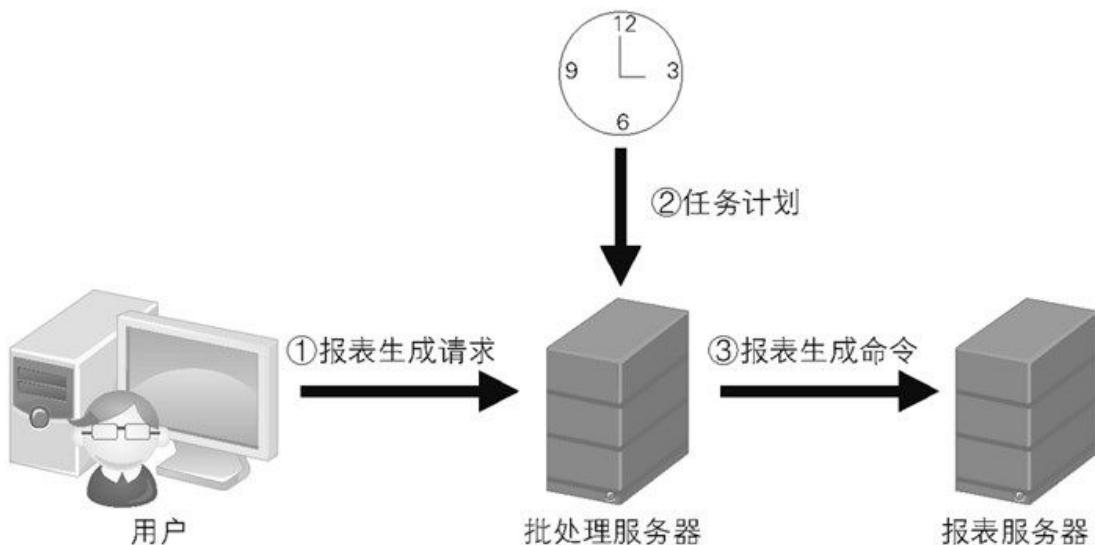


图 6.11 报表异步生成模式

需要注意当用户向批处理服务器发送报表生成请求后，并不会立即生成报表。此外，由于在报表服务器没有被使用的时候向其发送要求生成报表的命令可以充分使用服务器资源，因此适用于处理中等数量的报表的情况。

报表批处理生成模式

在批处理任务运行时向报表服务器发送命令，生成报表（图 6.12）。该模式适用于工资单等需要在特定日期生成大量报表的情况，可作为不与其他批处理任务相冲突，预先确保服务器充足资源的有效手段来使用。

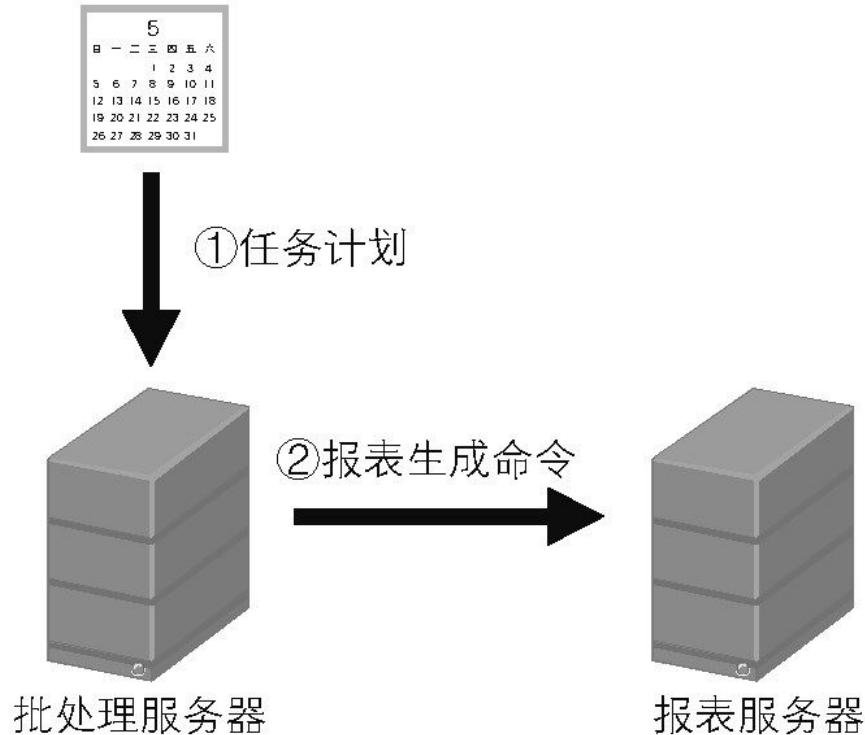


图 6.12 报表批处理生成模式

各模式的比较结果和选择标准

我们将各模式的比较结果与选择标准分别总结在表 6.11 和表 6.12 中。

表 6.11 报表生成的设计方式的比较结果

比较项目	报表实时生成	报表异步生成	报表批处理生成
响应速度	◎：快 实时生成	△：慢 取决于批处理运行时刻	×：慢 取决于批处理运行时刻
输出数量	△：少量	○：中量	◎：大量

比较项目	报表实时生成	报表异步生成	报表批处理生成
主要的报表对象	估价表 工作日志 证明 账单	工程成本 模拟报告	月结账表 工资单

表 6.12 报表生成的设计方式的选择标准

模式	报表的生成者	允许的报表生成时间	报表数量
报表 实时 生成	用户 适用于用户在有需要时才生成报表的情况	短 适用于需要在用户要求生成报表后能够实时生成报表的情况	少量 适用于一次生成少量报表的情况
报表 异步 生成		数分钟至1小时 适用于需要在用户要求生成报表后能够在一定时间内生成报表的情况	中量 适用于一次生成中量报表的情况
报表 批处 理生 成	部门（会计部门、财务部门等） 适用于需要定期地、有计划地生成月度报告和财务相关报告、收入证明和董事会议的通告等报表的情况	— 适用于需要在指定时间内生成报表的情况	大量 适用于一次生成大量报表的情况

注意点

如果需要报表可重复打印，则需要考虑 SPOOLING⁷ 等技术的设计。

⁷ Simultaneous Peripheral Operation On-Line，外部设备联机并行操作。在进行打印的时候，将所需打印的数据保存在终端或者服务器硬盘中的技术。

6.4 报表输出的设计方式

金融机关的转账申请表和移动电话的话费单、产品的估价表和收据等我们常见的报表中，除了电子报表外，还有不少是需要打印出来的。

报表的输出（打印）方式会对作业人员的操作是否方便以及每次可打印报表的数量产生影响，因此需要根据需求选择合适的报表输出方式。另外，是选择系统自动打印报表还是让用户自己操作打印报表会对系统的安全级别产生影响。

本节将介绍报表输出的设计方式。

报表服务器打印模式

画面上显示的信息并不保存为电子文档，而是直接输出为报表（图 6.13）。由于不需要保存电子文档，减少了文件管理工作的负担。

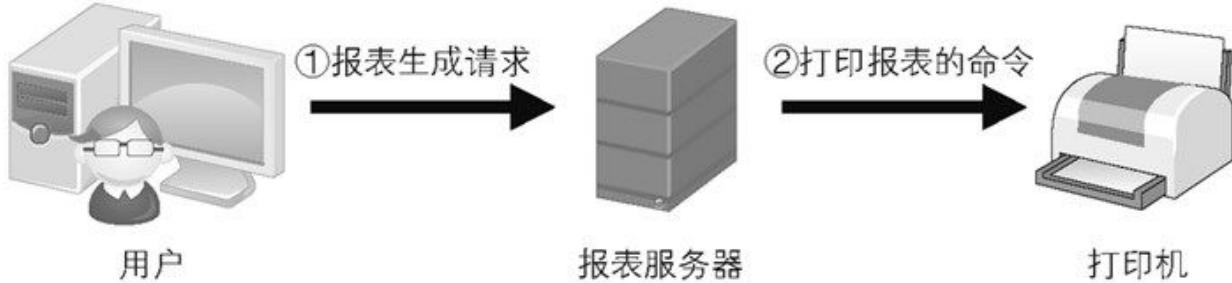


图 6.13 报表服务器打印模式

由系统决定使用哪台打印机和设置打印尺寸、打印布局等相关打印选项。当需要打印报表时，用户只需要向系统发送生成报表的请求即可，无需其他繁琐操作。

然后系统就会在报表服务器上生成打印数据，并直接执行发送给打印机的打印报表命令，因此该模式适合打印大量报表的情况。

用户终端直接打印模式

与报表服务器打印模式一样，画面上显示的信息并不保存为电子文档，而是直接输出为报表。不同的是，是在用户终端上决定使用哪台打印机和设置打印尺寸、打印布局等相关打印选项（图 6.14）。如果想要再次打印报表，必须再次打开想要打印的报表重新选择和设置。

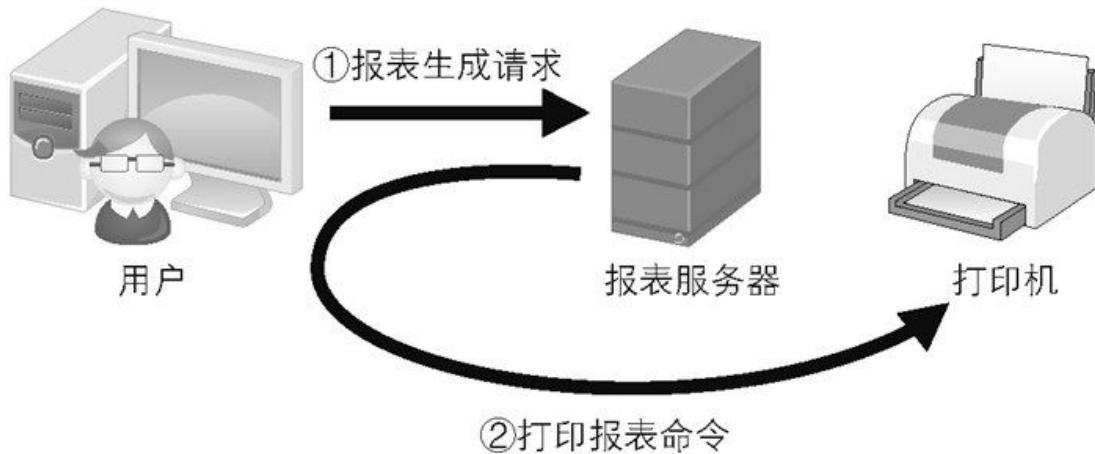


图 6.14 用户终端直接打印模式

该模式下用户的操作比较繁琐，因此不适合大量打印报表的情况。

在选择报表软件产品时要根据需求确认打印失败后是否不用重新生成报表即可再次打印，以及是否可以设置打印页码范围等打印选项。此外，有些时候必须在客户端安装报表软件，这时需要确认报表软件是否会对系统安全造成影响。

电子文档输出模式

将要输出的报表保存为电子文档并进行打印（图 6.15）。由于将报表保存为了电子文档形式，所以需要考虑安全性策略以防止电子文档泄露，同时还需要考虑电子文档的保存和管理方法。

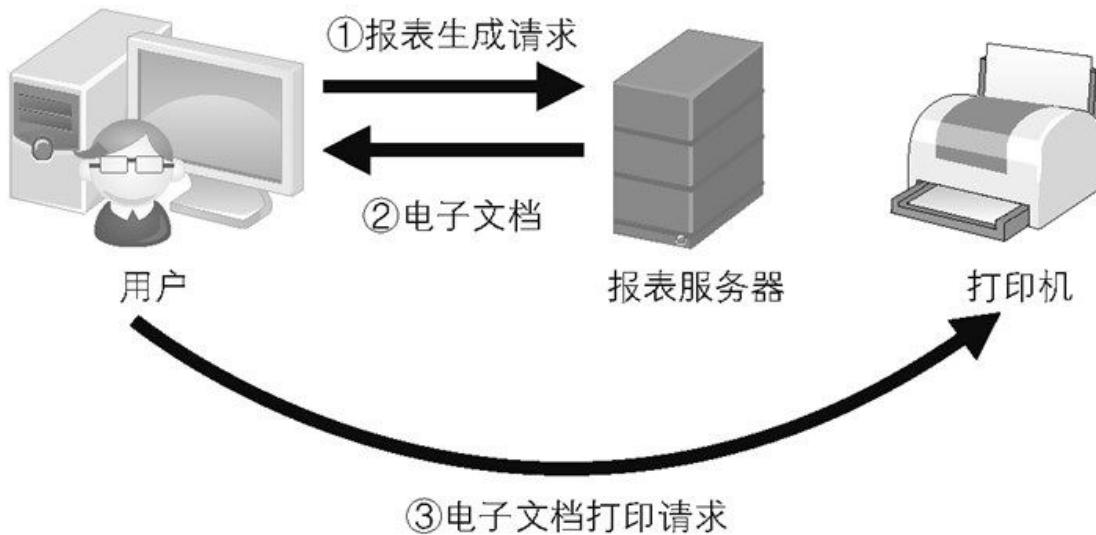


图 6.15 电子文档输出模式

如果业务中可能会需要重复打印或者是在 Web 上阅览报表，可以考虑采用该模式。与用户终端直接打印模式一样，需要在客户终端设置打印选项，因此该模式适合于少量打印报表的情况。

各模式的比较结果和选择标准

我们将各模式的比较结果与选择标准分别总结在表 6.13 和表 6.14 中。

表 6.13 报表输出的设计方式的比较结果

比较项目	报表服务器打印	用户终端直接打印	电子文档输出
安全性	◎： 极高 不会在终端生成电子文档，数据不容易被泄露	○： 高 通常情况下终端上不会残留电子文档，数据不容易被泄露	△： 低 终端上会残留电子文档，需要考虑安全性措施 ^{*1}

比较项目	报表服务器打印	用户终端直接打印	电子文档输出
可用性	○：高 用户无需在意打印机的型号和各种打印选项的设置，操作简便	△：中 可使用在OA系统中已经设置完毕的打印机，但是与报表服务器打印模式相比，用户操作略显繁琐	✗：低 需要频繁设置电子文档的打印选项，用户操作繁琐

*1 通过设置密码、加密保存数据、加密通信数据可以提高安全性

表 6.14 报表输出的设计方式的选择标准

模式	用户的操作负荷	打印数量
报表服务器打印	少 通过报表服务器直接打印，用户几乎无需操作	大量 适用于需要打印大量报表的情况
用户终端直接打印	中 需要通过显示画面确认报表并进行打印等操作	少
电子文档输出	多 需要保存为电子文档，然后再打开电子文档进行打印等操作	适用于需要打印少量报表的情况

注意点

考虑输出格式

报表系统除了打印纸质报表外，有时还需要输出 CSV (Comma Separated Value, 逗号分隔值) 格式的电子报表。因此在设计系统时需要考虑如何支持各种输出格式。

考虑 WAN 线路

打印数据量通常要比原始数据量大，因此为了能够缩短打印时间，需要考虑采用高速 WAN 设备和确保充足的带宽。

6.5 报表基础设施配置的设计方式

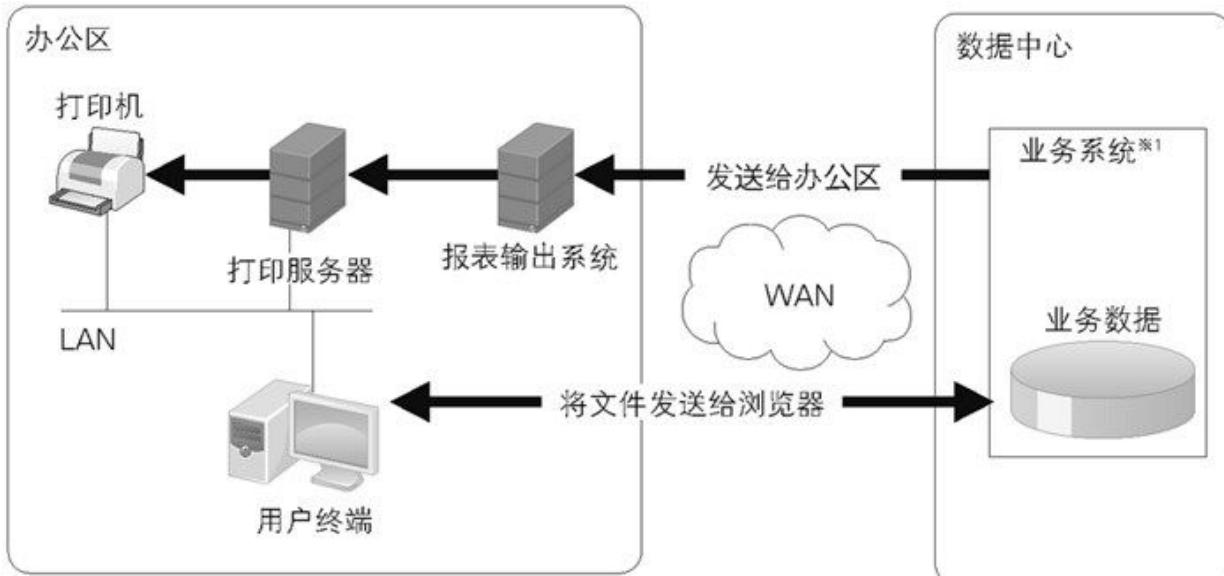
由于商业和业务需求，电信运营商所打印的话费明细单和产品生产商所打印的发货单每年要有数千甚至数亿张之多。所需打印的报表越多，越难在同一系统中同时满足报表生成需求和报表输出需求。因此，许多情况下，报表生成系统和报表输出系统需要配置在不同的机房或是数据中心来进行管理。

在不同的机房或是数据中心分别进行报表的生成和打印，与在同一个机房或是数据中心生成和打印报表相比，安全性较低，而且大量报表数据的通信会可能会对业务数据通信造成影响。另一方面，如果在同一个机房或是数据中心生成报表和打印报表，又很难针对不同地区的用户设置不同的打印选项。因此，需要根据安全性、数据量等需求，为报表系统选择合适的物理配置方式和它们之间的交互方式。

本节将介绍报表系统的物理配置方式。

分布式管理 + 分布式印刷模式

在数据中心生成必要的数据发送给各个办公区，然后在各个办公区使用这些数据来生成报表数据并打印报表（图 6.16）。



※1 该系统为报表输出系统生成和打印报表提供原始数据（本节以后的图中不再特别注明）

图 6.16 分布式管理 + 分布式印刷模式

该模式适合于需要为每个办公区域定制报表输出格式的情况。由于在每个办公区都有报表服务器，所以在安全管理上需要特别注意。

集中管理 + 分布式印刷模式

在数据中心生成必要的数据并使用这些数据生成报表数据，报表输出系统会通过网络在各个办公区的打印机上打印各办公区需要的纸质报表（图 6.17）。

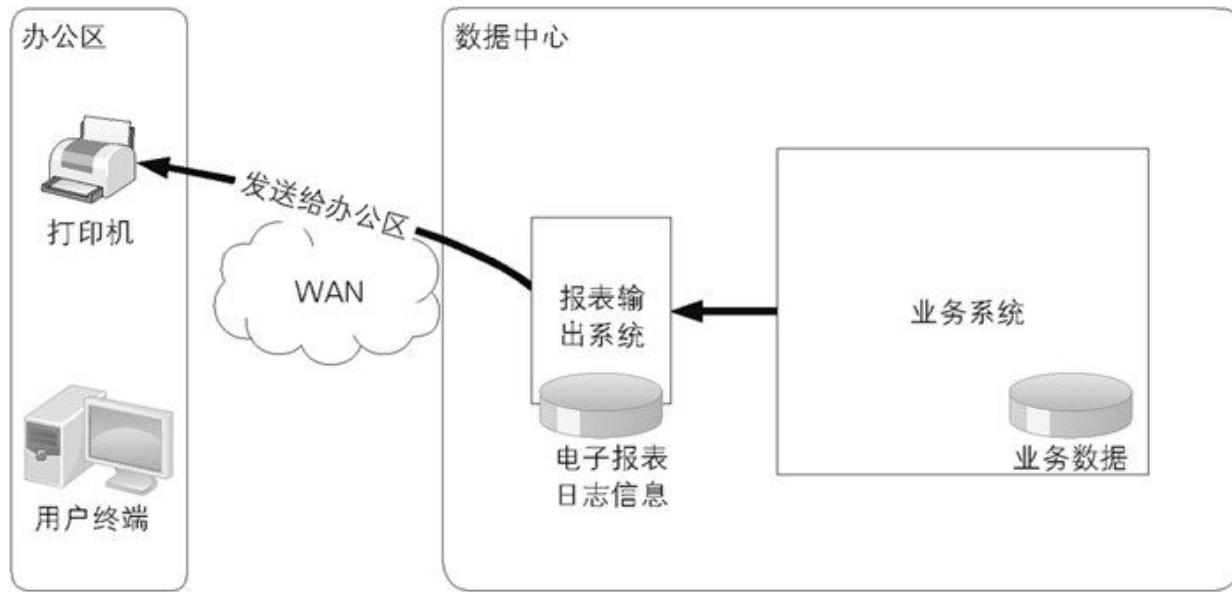


图 6.17 集中管理 + 分布式印刷模式

与分布式管理 + 分布式印刷模式相比，该模式适用于想要统一变更和管理报表模板的情况。

由于所有的报表数据都是在数据中心生成，因此可以在数据中心统一管理用户在什么时间打印了什么报表，提高了安全性。

但是，从数据中心送往各办公区域的报表数据量较大，打印大量报表时会加重网络通信负荷，因此该模式不适合打印大量报表的情况。此外，还需要考虑图片数据的传送时间段以防止对业务通信造成影响。而且，在打印的时候必须给打印机插上电源，而电源管理是在各个办公区进行的，因此数据中心必须与各办公区就打印时的电源问题进行协调。

集中管理 + 集中印刷模式

报表的生成和打印都在数据中心进行（图 6.18）。由于打印也是在数据中心进行的，因此如果各个办公区需要纸质报表时，必须通过交通工具将报表从数据中心运输到办公区。但由于管理也是在数据中心集

中进行的，因此用户无法从各办公区直接连接报表系统进行打印，因此提高了安全性。

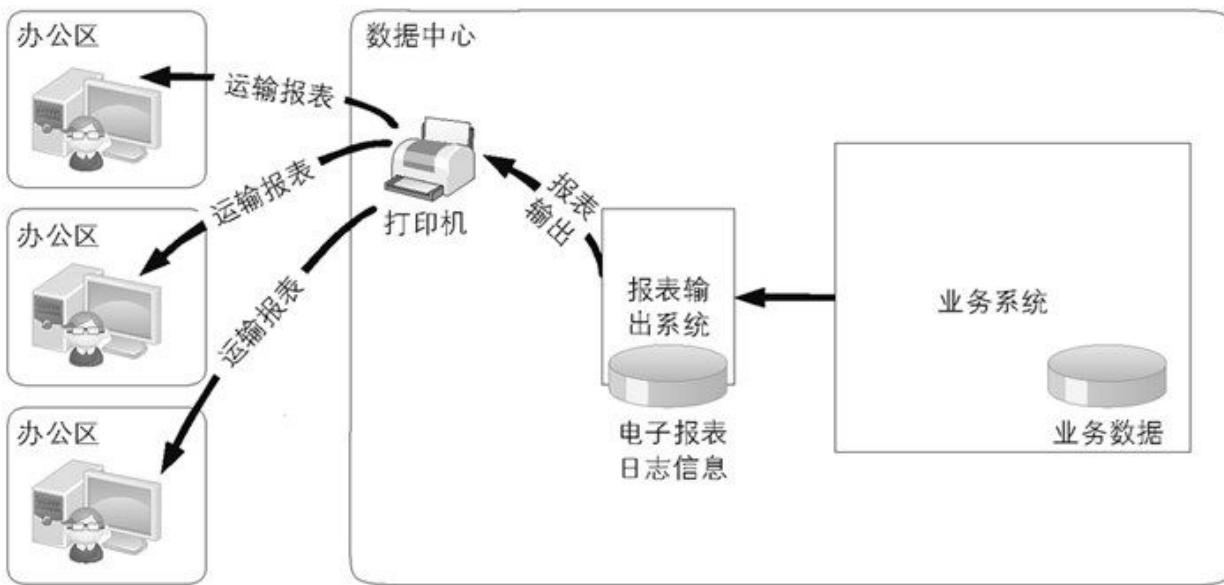


图 6.18 集中管理 + 集中印刷模式

此外，由于是在数据中心集中生成和打印报表，所以能够统一管理打印机。因此，只要准备高性能的打印机，就可以完成打印大量报表的任务。

各模式的比较结果和选择标准

我们将各模式的比较结果与选择标准分别总结在表 6.15 和表 6.16 中。

表 6.15 报表基础设施配置的设计方式的比较结果

比较项目	分布式管理 + 分布式打印	集中管理 + 分布式打印	集中管理 + 集中打印
安全性	○：高 可以在每个办公区各自的打印服务器	◎：高 可以统一管理报表服务器的访问权限和访问日志	

	上管理报表输出 (打印)			
报表系统的统一管理	<p>△: 困难 在多个办公区都有报表输出系统，需要对所有的报表输出系统进行统一管理</p>	<p>○: 容易 可以通过报表输出系统统一管理所有办公区的报表</p>		
可用性	<p>○: 高 每个办公区都可定制报表模板。纸质报表在每个办公区的打印机处进行打印，该办公区内的人员随时可以拿到</p>	<p>△: 中 报表模板在数据中心进行统一管理，因此如果要针对某个办公区定制模板需要与其他办公区进行协调。纸质报表在每个办公区的打印机处进行打印，该办公区内的人员随时可以拿到</p>	<p>×: 低 在数据中心统一管理报表输出系统，如果要针对某个办公区进行定制，则需要与其他办公区进行协调。纸质报表在数据中心打印，需要运输到每个办公区</p>	
网络负荷	<p>△: 中 在报表输出系统和报表生成系统之间传输的信息基本只有生成报表所必需的文本数据</p>	<p>×: 高 需要将数据中心生成的报表传输给各办公区</p>	<p>○: 无 报表的生成和打印都是在数据中心进行的。纸质报表则通过交通工具进行运输</p>	

表 6.16 报表基础设施配置的设计方式的选择标准

模式	各办公区定制的便利性	报表数量
分布式管理 + 分布式打印	<p>高 适用于想在各办公区定制报表模板的情况</p>	<p>中等 适用于需要一次打印中等数量的报表的情况</p>
集中管理 + 分布式打印	<p>低 适用于不需要为各办公区定制报表模板的情况</p>	
集中管理 + 集中打印		<p>大量 适用于需要一次打印大量的报表的情况</p>

注意点

打印机的限制

不同的报表软件所支持的打印机不同，需要根据所使用的报表软件来选择合适的打印机。另外，在处理大量报表时，如果使用的是不同型号和不同性能的打印机，运维成本和打印速度可能会有很大区别。因此，需要根据需求来选择合适的打印机。

字符编码的限制

不同的报表软件所支持的字符编码集⁸不同，在选择报表软件时需要考虑系统中所使用的文字编码。如果交互系统较多，且交互系统中使用了不同的文字编码，则需要考虑增加文字编码转换功能。

⁸ 字符编码是指英文、汉字等字符的二进制编码。字符编码集的规格各不相同，有些情况下它们之间无法互相转换。

6.6 数据使用和信息分析的设计方式

随着IT技术的飞速发展，系统中所收集和保存的数据量越来越大。因此，为了能够从大量数据中提取有效信息，数据挖掘系统和BI（Business Intelligence，商业智能）⁹工具被广泛使用。现在，“大数据”这一关键词越来越受到业内人士所关注。

⁹ 通过统计、整理、分析企业保存的数据，为企业决策提供支持的工具。

为了能使系统保存和处理大量数据，需要在构建系统时综合考虑性能问题和安全性问题。

本节中将介绍数据分析的设计方式。关于最近流行的大数据处理技术，全面定义其基础设施设计模式还为时尚早，我们仅总结了部分设计模式。

数据集市 + BI 构成的数据仓库模式

该模式构建数据仓库¹⁰ 来收集和保存所有部门和用户使用的各种信息，并构建 ETL（Extract/Transform/Load）¹¹ 对保存的信息进行加工，使信息更容易被使用。根据使用数据的目的和部门相应地引入数据集市¹² 和 BI 工具（图 6.19）。

¹⁰ 指的是在主干系统之外再构建一套大型 DB。通过保存和分析主干系统中的业务信息，为企业的决策提供支持。

¹¹ 提取系统保存的信息并进行转换，使其可供数据仓库等使用，并将它们保存在 DB 中的软件。

¹² 可从数据仓库中提取符合需求的数据和功能的系统。

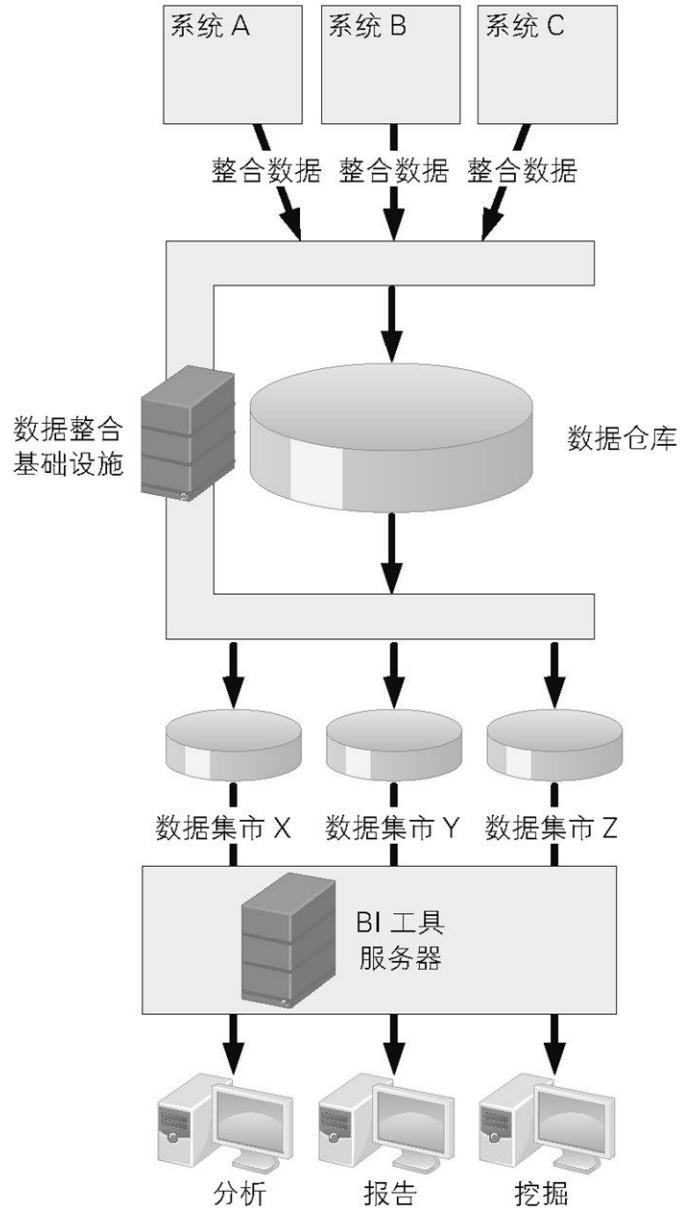


图 6.19 数据集市 + BI 构成的数据仓库模式

各数据集市的处理由于只涉及所必需的那部分数据，所以可以确保高性能。此外，由于可以根据部门和用户设置数据集市的访问权限，因此可以根据需要设置安全等级。

需要特别注意的是，数据集市分布式保存数据仓库中的数据，因此降低了整体系统的可维护性。

此外，ETL 中对数据的加工非常重要。例如，门栋地址相同但小区名字被省略的情况下以及“1-101”和“1 号楼 101 室”等实际地址相同但写法不同的情况下，系统可能会将它们识别为不同的地址导致分析结果有误。这时，应当制定相应的数据规则让系统可以正确识别它们，这个过程称为“数据清洗”。

BI 构成的数据仓库模式

构建数据仓库收集和保存所有部门和用户使用的各种信息，并构建 BI 工具来使用这些数据（图 6.20）。

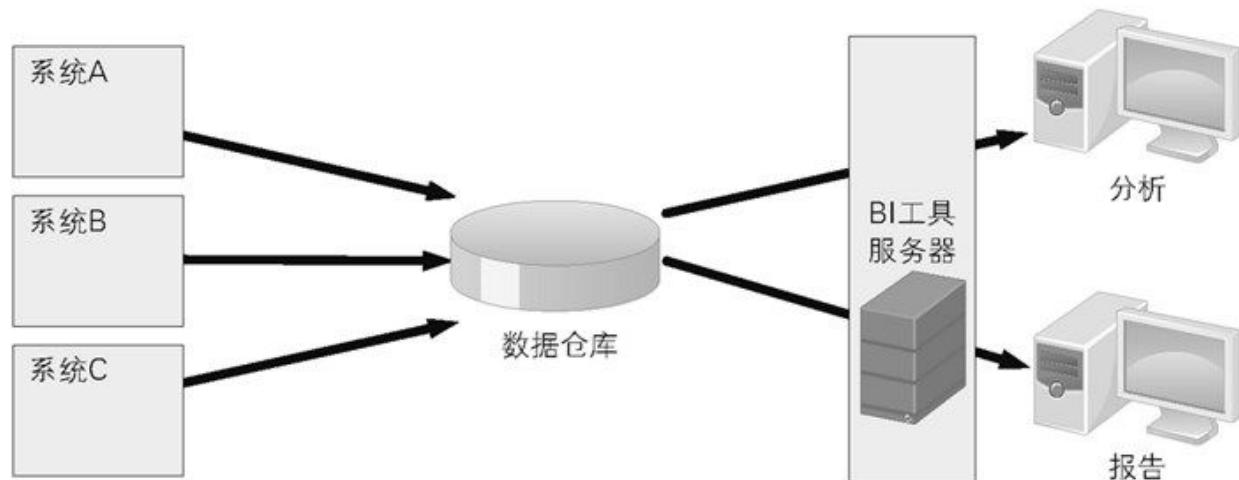


图 6.20 BI 构成的数据仓库模式

查询处理和计算处理在数据仓库或是 BI 工具上进行，但是需要先对数据进行加工，可以考虑采用 ETL 来完成数据加工工作。

由于没有构建数据集市，处理速度上会比“数据集市 + BI 构成的数据仓库模式”慢一些。安全性方面，可以通过 BI 工具设置各部门和各用户访问数据的权限。

EUC 模式

由用户自己从目标系统中查询出想要的数据，或者利用查询出的信息进行分析（图 6.21）。

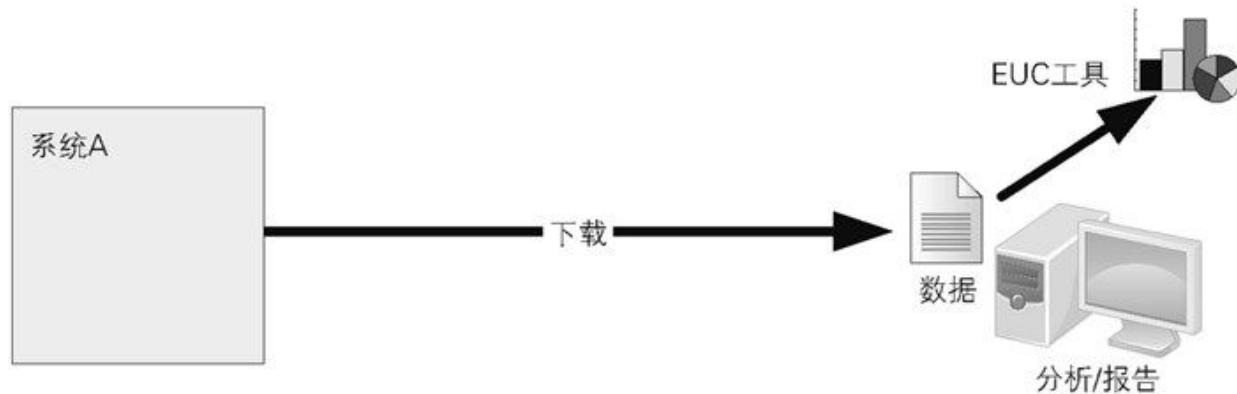


图 6.21 EUC 模式

我们将不是由系统管理员，而是由用户自身进行系统构建、管理和使用的方式成为 EUC (End User Computing, 终端用户计算)。由于是用户自身进行分析处理，可处理的信息量和处理速度都非常有限。

此外，该模式下数据的安全性也取决于用户自身，因此容易导致信息泄露。

各模式的比较结果和选择标准

我们将各模式的比较结果与选择标准分别总结在表 6.17 和表 6.18 中。

表 6.17 数据使用和信息分析的设计方式的比较结果

比较项目	数据集市 + BI 构成的数据仓库	BI 构成的数据仓库	EUC
处理的数据量	◎：数千万以上 TB数量级以上的数据	○：数十万~数千万 最多TB数量级的数据	△：数万件 最多MB数量级的数据
可否应对各种数据混	◎：容易 通过数据整合基础设	○：可应对 可以通过加工处理应对混合数据	-：无法应对

合的情况	施可以整合各种混合数据		
安全性	○：高 可通过BI工具进行访问控制。可通过数据仓库、数据集市分层进行访问控制	△：中 可通过BI工具进行访问控制	×：低 访问控制困难
处理速度	○：快 使用了数据集市，处理速度快	△：普通 如果正在进行大负荷处理，会导致系统整体处理速度降低	×：慢 用户需要手动从系统中提取必要的数据，处理速度慢
使用的工具	ETL 数据集市 BI工具	BI工具	EUC工具
可分析、加工性	◎：高 BI工具服务器中可进行详细分析和自由加工	○：普通 BI工具服务器中可以进行分析。但是可处理的数据量和处理速度均不如数据集市+BI构成的数据仓库模式	△：依赖用户 依赖于用户的IT技能

表 6.18 数据使用和信息分析的设计方式的选择标准

模式	可应对的业务复杂度	作为分析对象的业务数量	访问权限的设置
数据集市+BI构成的数据仓库	高 适用于短期的市场调查等需要快速分析大量信息的情况	多 适用于从许多业务系统中收集数据的情况	可在数据仓库中设置 不依赖于所分析的业务系统的访问权限，而是根据所分析的内容在数据仓库中设置访问权限
BI构成的数据仓库	中 适用于进行商品的长期销售情况分析等期限较长的分析	中 适用于从一定数量的业务系统中收集数据的情况	可在数据仓库中设置 不依赖于所分析的业务系统的访问权限，而是根据所分析的内容在数据仓库中设置访问权限
EUC	低 适用于业务简单、处理时间长，用户（个	少 适用于从少量	依赖各业务系统 用户只能收集和分析业务系统允许访问的数据

	人) 就可以进行分析的情况	业务系统中收集数据的情况	
--	---------------	--------------	--

注意点

考慮性能策略

在审计处理中由于每次都会运行数据统计处理，可能导致处理速度变慢。为了确保高性能，可以考虑引入数据仓库专用设备或是大数据解决方案。

特别是在数据集市 + BI 构成的数据仓库模式、BI 构成的数据仓库模式中，在进行数据传输和分析处理时，网络性能和磁盘 I/O 性能可能会成为性能瓶颈，因此必须构建能够确保充足性能的网络和存储设备。

考慮数据更新方法

从作为信息提供源的各个系统对数据仓库、数据集市进行更新的方法需要综合考虑各业务系统的负荷趋势、数据量和成本来确定。此外，如果对数据更新时间有要求，可能会导致成本增加，需要谨慎考虑。

其他注意事项

如果业务系统与数据仓库的字符编码不同，则需要进行字符编码的转换。字符编码转换处理会给数据仓库造成较大的负荷，因此实在在业务系统中进行还是在数据整合基础设施中进行可能对系统构成产生很大影响。

补充：元数据管理

数据仓库中不仅数据量大，而且各种数据之间关系复杂，因此为了高效提取所需要的数据，必须进行元数据¹³管理（图 6.22）。可考虑引入元数据管理产品帮助管理数据种类和数据之间的关系。

13 表示数据的种类和数据之间的关系等数据属性信息的数据。

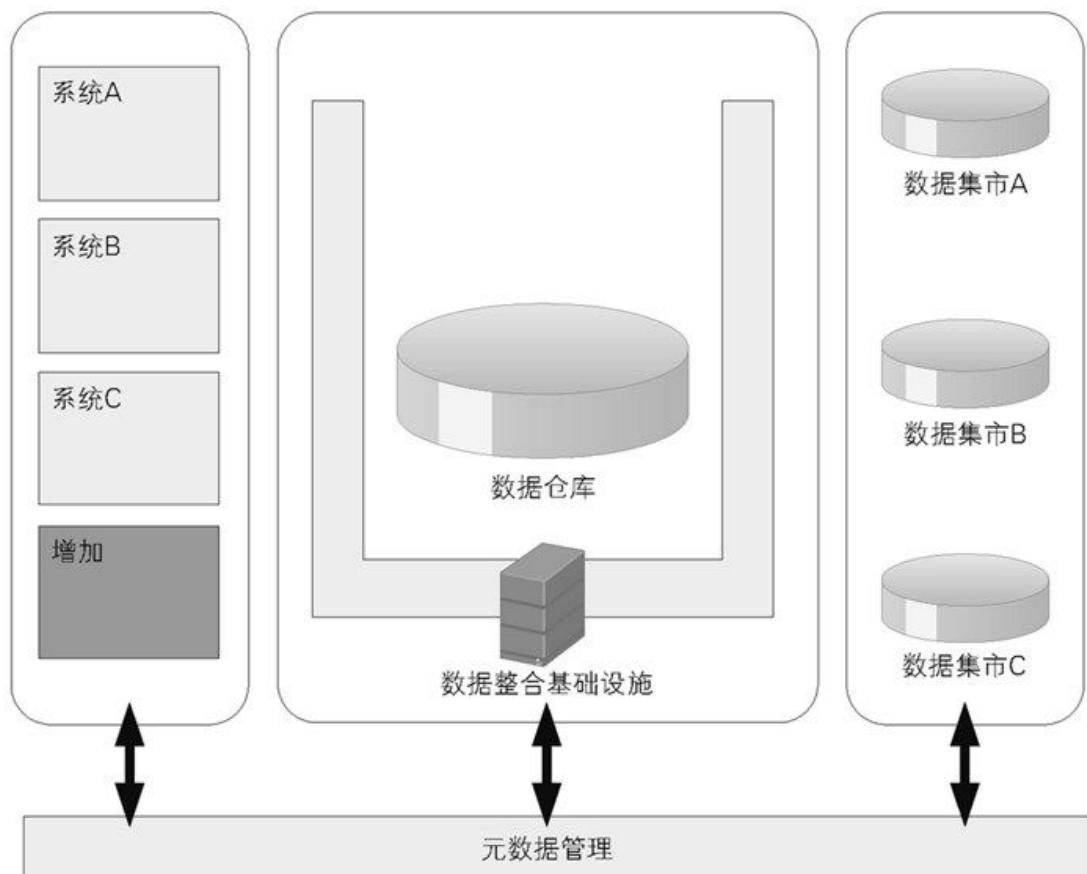


图 6.22 元数据管理

补充：数据仓库设备

数据仓库设备优化了磁盘的输入输出性能，实现了高速输入输出，可高效处理大量数据。特别是对于非结构化的数据，采用数据仓库设备可实现高效信息分析。

但是需要特别注意数据仓库设备与 BI 工具的兼容性。如果已经使用了某个 BI 工具而且想继续使用的时候，选择数据仓库设备时就需要注意是否与该 BI 工具相兼容。

6.7 基础设施交互结构的设计方式

企业活动中 IT 系统很少单独工作，经常需要与其他系统进行交互，例如电商网站中的订单管理系统需要与库存管理系统进行交互。但是由于这些系统构建时的目的和理念不同，所使用的技术也不相同，因此它们之间的交互非常困难。

本节将介绍如何设计基础设施间的交互方式，使得既能确保交互数据的正确性和完整性，又可以灵活地应对商业需求的变化。

服务总线模式

采用 SOA (Service-Oriented Architecture, 面向服务架构) 模式¹⁴，构建服务总线来处理各系统的字符编码和事件，支持系统间的交互（图 6.23）。将数据转换为符合交互规范的处理过程集中在服务总线中进行，服务总线与各系统相互连接。

¹⁴ 在面向服务架构中，每个系统都被看作一个服务。为了能够实现交互，各系统都带有标准接口，也因此能够很方便地增加和移除系统。不仅如此，每个系统都被看作一个服务还有助于分散各系统的职责，防止重复开发，节省投资。

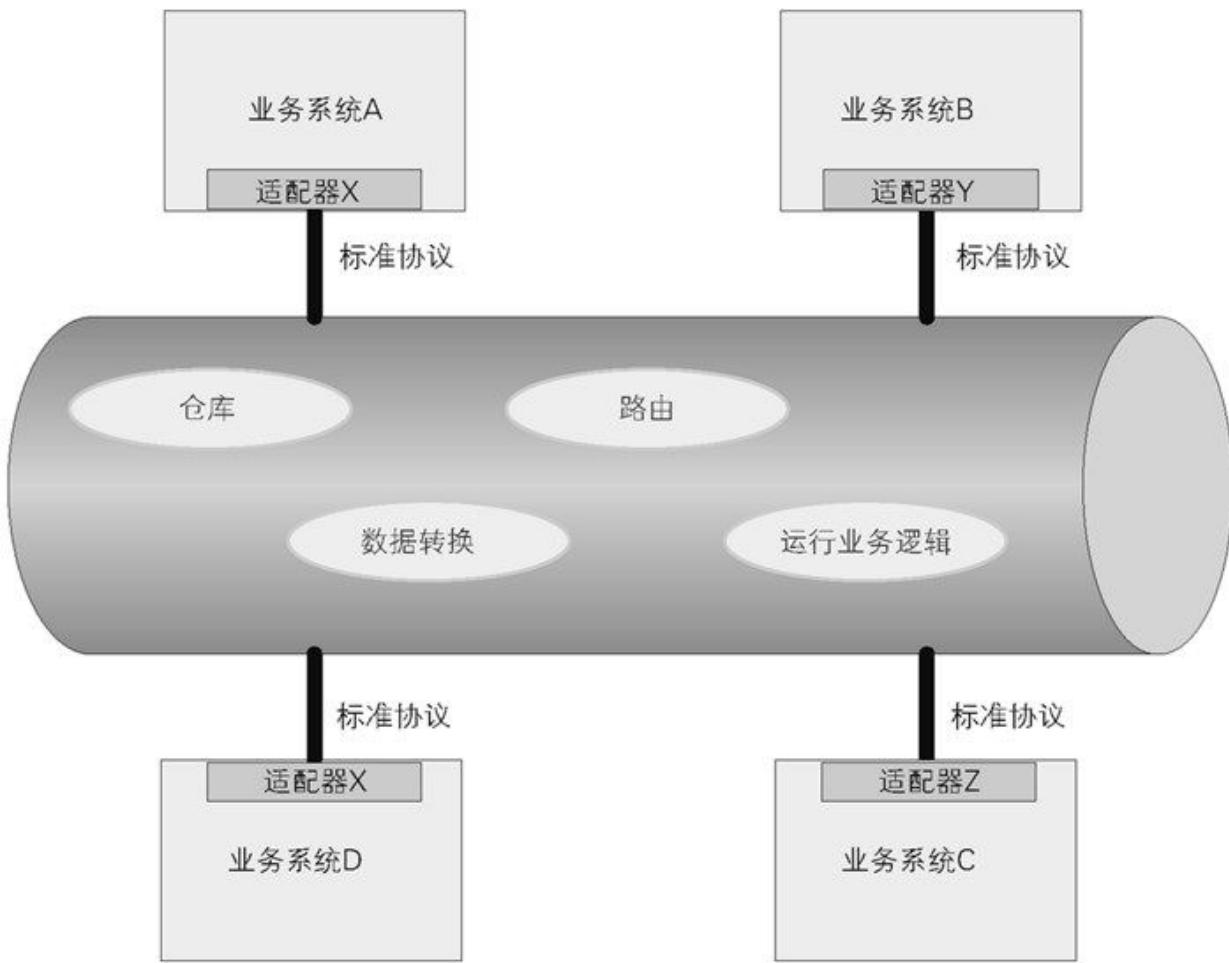


图 6.23 服务总线模式

虽然各系统可以很容易地通过服务总线与其他系统进行交互，但是需要先通过适配器将与服务总线交互的接口改为标准协议。

数据中心模式

采用 EAI (Enterprise Application Integration, 企业应用集成)¹⁵ 整合企业内部处理的数据。通过被称为“数据中心” (Data Hub) 的信息管理专用系统管理保存在各系统中的数据 (图 6.24)。当某个系统需要获取其他系统的数据时，向数据中心发送请求。数据中心会从其他系统获取数据，进行适当的数据转换处理后发送给该系统。因此，如果系统数量过多会导致数据中心负荷加大。

¹⁵ 集成企业内部多个系统的数据和处理过程以实现它们之间的交互。

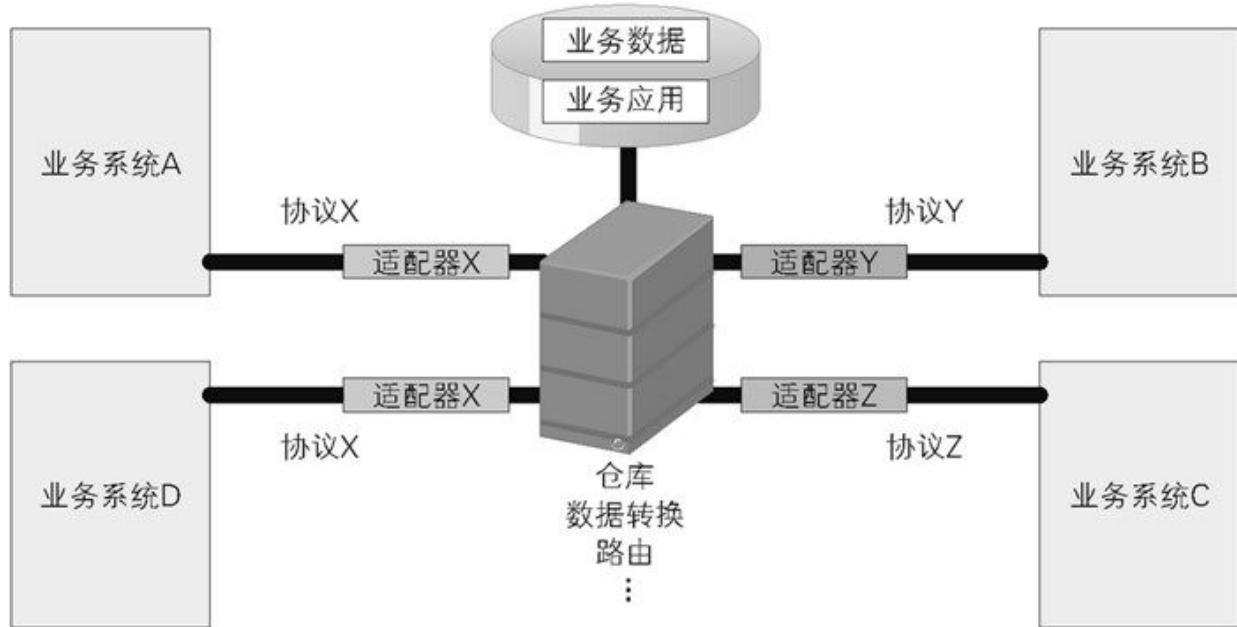


图 6.24 数据中心模式

当有新系统需要通过数据中心与其他系统进行交互时，现有系统不需做任何改变。

P2P 模式

采用 MQ (Message Queuing, 消息队列)¹⁶ 等实现小型系统间的交互。当系统间需要交互时，只需构建接口即可。交互只能通过接口进行（图 6.25）。

¹⁶ 不同应用程序间进行交互时，将传输数据保存在称为“队列”的存储空间中，不必等待接收方应用程序的响应即可进行下一个处理。适用于在不使用服务总线和数据中心模式的情况下实现系统间 N 对 N 的直接交互。

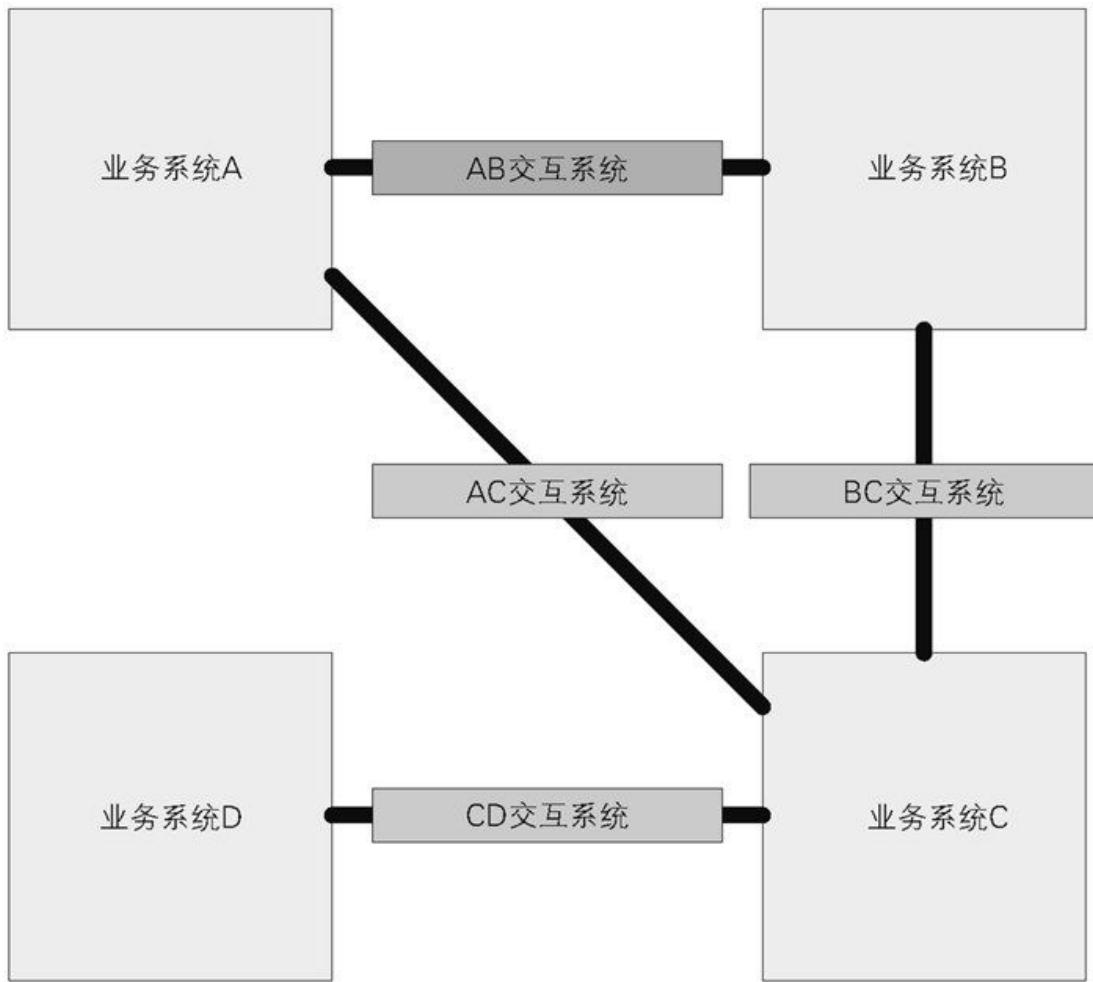


图 6.25 P2P 模式

当系统数量较少时，可以以比服务总线模式和数据中心模式更低的成本实现系统交互。设计时需要注意系统数量，交互系统数量越多，成本越高。因此该模式适用于预计将来不会增加较多交互系统的情况。

各模式的比较结果和选择标准

我们将各模式的比较结果与选择标准分别总结在表 6.19 和表 6.20 中。

表 6.19 交互基础设施的设计方式的比较结果

比较项目	服务总线	数据中心	P2P
交互方法	引入服务总线功能，通过进行各系统间字符编码和协议的转换以及业务逻辑的处理实现系统间交互	将各系统中使用的信息集成到数据中心。各系统通过向数据中心发送数据和从数据中心接收数据实现系统间的交互	根据需要额外开发交互系统实现系统间的交互
性能	<p>△：普通 必须通过服务总线进行数据交互，会增加服务总线负荷</p>	<p>△：普通 由于是辐射型集中处理，会增加数据中心的负荷</p>	<p>◎：非常 可以确保高性能</p>
接口可复用性	<p>○：高 各系统的接口（适配器）采用标准技术</p>	<p>△：普通 各系统的接口（适配器）可能会使用特殊的技术，与其他数据中心进行交互时可能很难复用</p>	<p>×：低 基本上所有系统间的接口都不相同，很难复用</p>
系统间交互的难度	<p>○：易 需要进行交互时通过服务总线交换数据即可，不会发生数据中心模式下的数据不正确、不完整的问题。此外，由于使用了标准技术，系统间具有高互通性</p>	<p>△：普通 由于数据交换时需要在数据中心临时保存数据和进行数据转换，因此需要注意如何确保数据的正确性和完整性。此外，有可能需要考虑对外系统的互通性</p>	<p>×：困难 需要开发交互功能，很难用于多个系统间交互</p>

表 6.20 交互基础设施的设计方式的选择标准

模式	交互系统数量	可扩展性
服务总线	多 适用于一个系统需要从其他许多系统获取信息（DB中的数据）等交互系统很多的情况	高 适用于决定将来要增加交互系统等需要高可扩展性的情况
数据中心	中 适用于一个系统需要从其他若干系统获取信息（DB中的数据）等交互系统较多的情况	中 适用于预计会增加交互系统等需要一定的可扩展性的情况
P2P	少 适用于一个系统只需要从一个或几个特定的系统中获取信息等交互系统很少的情况	低 适用于今后不会增加交互系统，不需要可扩展性的情况

注意点

性能测试

采用服务总线模式和 P2P 模式进行系统交互时，性能较差的系统可能会导致整体性能的下降。为了确保不同规格的系统进行交互时整体性能不受影响，必须进行充分的设计和测试。

交互系统数量与成本的关系

图 6.26 展示了交互系统数量与成本的关系。当交互系统数量超过一定数量时，构建交互基础设施的优势将会非常明显。

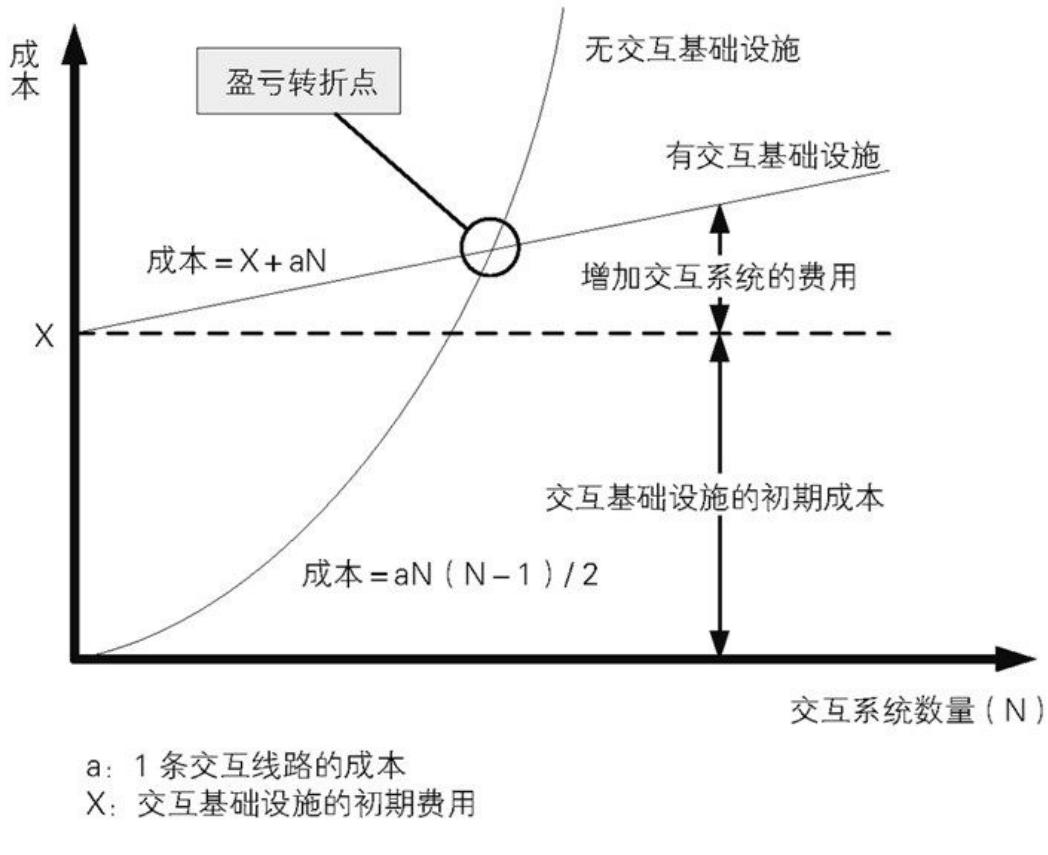


图 6.26 是否引入交互基础设施与系统间交互成本的关系

其他注意事项

当不断增加或是移除交互系统时，需要考虑服务总线模式中的服务总线和数据中心模式中数据中心的运维和管理。

如果没有考虑系统间的交互问题就直接构建和运维系统，那么当以后该系统需要与其他系统进行交互时，就会花费额外的成本与人力。因此，事先制定统一的规格和规范，并按照规格和规范去构建和运维系统，在实现系统交互时设计可复用的接口和确定共通 DB 规格等就会轻松得多。

6.8 总结

本章讲解了基础设施的设计方式。在设计基础设施时，需要综合考虑各种需求才能得到最佳的设计方式。因此，必须在充分理解基础设施的各种设计方式和其特点后，从系统整体的角度选择合适的设计方式。

第 7 章 使用云计算服务的实现策略

云计算一般是指通过网络按需分配计算资源的计算方式。很早以前业界就在讨论这种计算方式，随着近几年计算机技术的革新，越来越多的企业开始提供云计算服务（以下简称云服务）。

云服务提供商通过在大型数据中心设置大量服务器，构建可以通过网络远程使用计算资源的系统。用户通过网络使用云服务提供商的计算资源，并支付相应的服务费用。由于用户共享计算资源，大部分云服务可以实现规模利益，降低成本。

云服务有 SaaS（Software as a Service，软件即服务）、PaaS（Platform as a Service，平台即服务）、IaaS（Infrastructure as a Service，基础设施即服务）等类型。SaaS 是指用户可以通过网络使用软件的服务形态。PaaS 是指用户可以通过网络使用服务器平台（OS 和中间件）的服务形态，该平台用于构建并运行软件。IaaS 是指用户可以通过网络使用虚拟服务器、网络、CPU、存储设备、OS、中间件等基础设施的服务形态。通过熟练使用云服务，可以简单、低廉地实现 IT 系统需求，享受云服务的便利。

本章中，我们将特别介绍使用了基础设施云服务的 IaaS 的典型设计方式。

此外，由于本章介绍的设计方式使用了云服务提供的功能，因此其实现可能性依赖于所使用的云服务，所以我们并没有总结各模式的选择标准。在选择模式时，除了需要充分了解各模式的特点外，还需要确认云服务是否提供模式中所需的功能。

7.1 云服务中性能与可扩展性的设计方式

大部分的云服务中可以快速且灵活地增加或减少计算资源（CPU 和内存、虚拟服务器等）。因此，云服务的规模调整并非在上游设计阶段进行，而是在运维过程中不断优化的，这样就可以节省规模调整的工作量，避免资源浪费。从而既可以确保系统的高性能与高可扩展性来应对企业所追求的商业速度和环境变化，又能够缩减系统的构建和运维成本。

服务器纵向伸缩模式

根据需要变更虚拟服务器的配置（CPU、内存等）（图 7.1）。

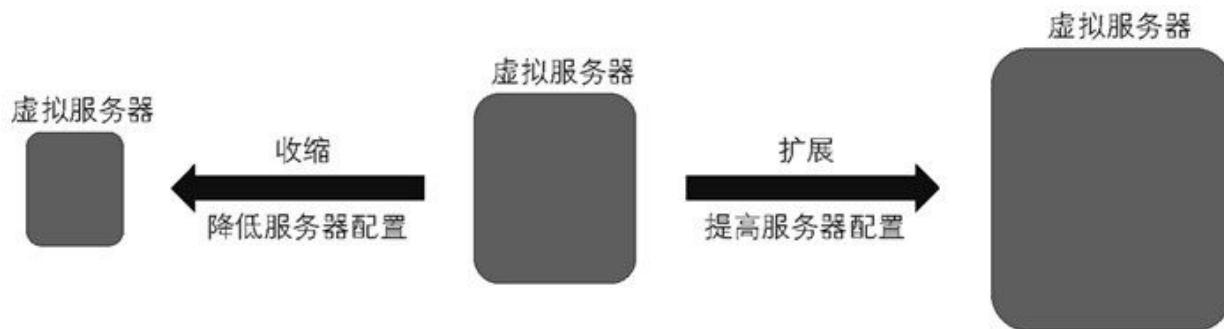


图 7.1 服务器纵向伸缩模式

许多云服务都允许随时切换某台虚拟服务器的规格，因此企业用户可以随时根据系统资源使用情况来调整服务器配置，从而有效防止服务器资源不足导致的系统性能下降或是性能过剩导致的投资浪费。

磁盘资源量增减模式

根据需要变更磁盘的数量和容量（图 7.2）。

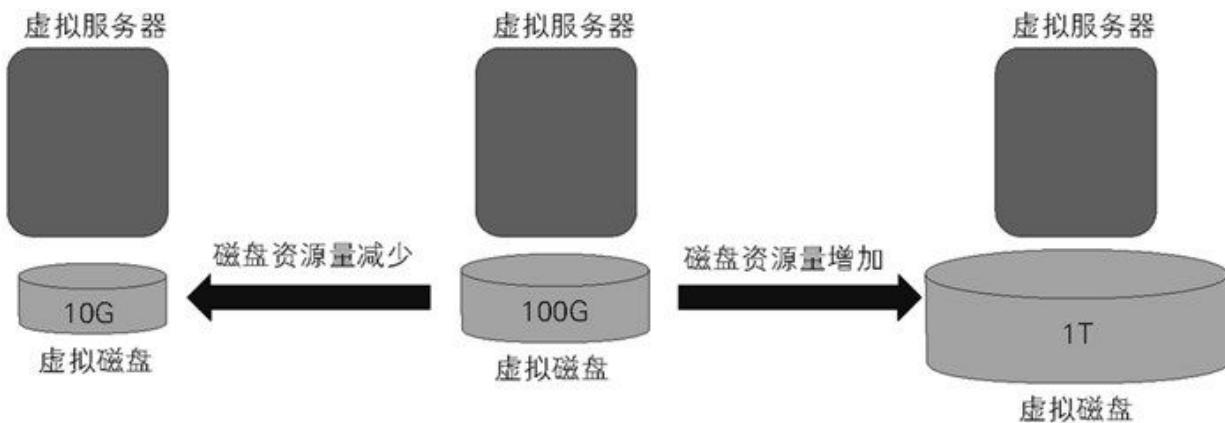


图 7.2 磁盘 资源量增减

许多云服务都允许随时增加或是减少虚拟磁盘，因此企业用户可以随时根据系统的虚拟磁盘使用情况来增减虚拟磁盘。这样，无需为数年后可能产生的数据量增加或是暂时的数据量突增准备额外的磁盘，能够缩减投资成本。

自动伸缩模式

自动地变更（自动伸缩）虚拟服务器的数量（图 7.3）。

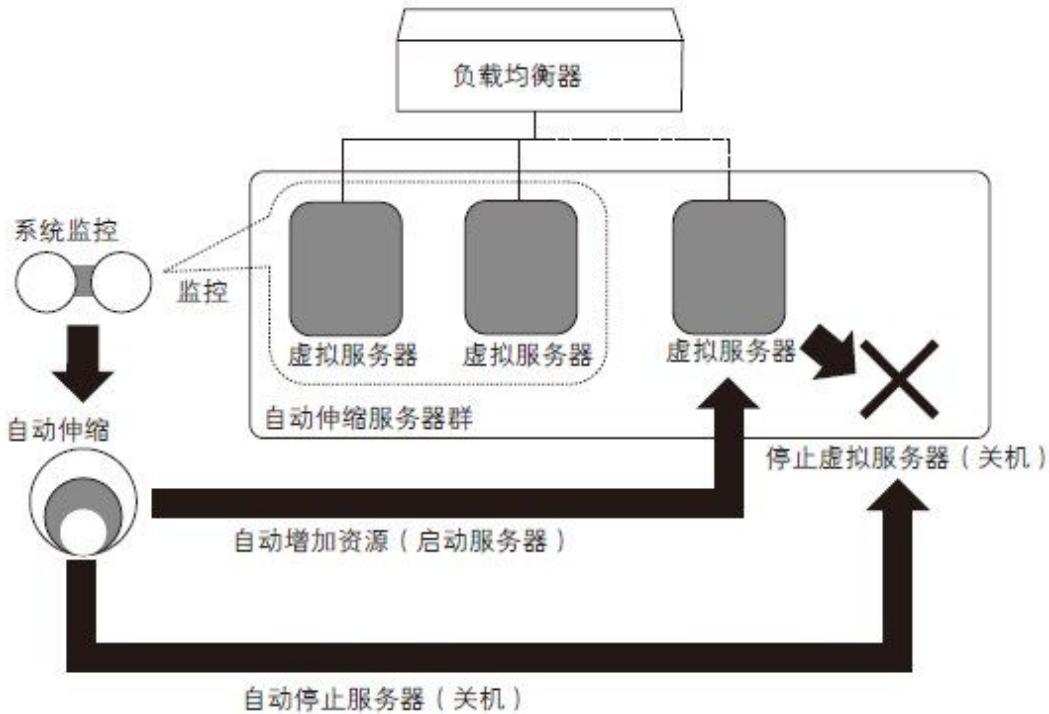


图 7.3 自动伸缩模式

许多云服务都提供自动伸缩功能。该功能可以事先根据资源使用情况和使用时间设置自动伸缩方针，以实现服务器的自动启动和停止。通过结合云服务提供的系统监控功能以及负载均衡功能，在面对突发高负荷状况时，即可自动地增加虚拟服务器并分散负荷。

任务应对横向伸缩模式

根据任务的运行状况自动地变更虚拟服务器的数量（图 7.4）。

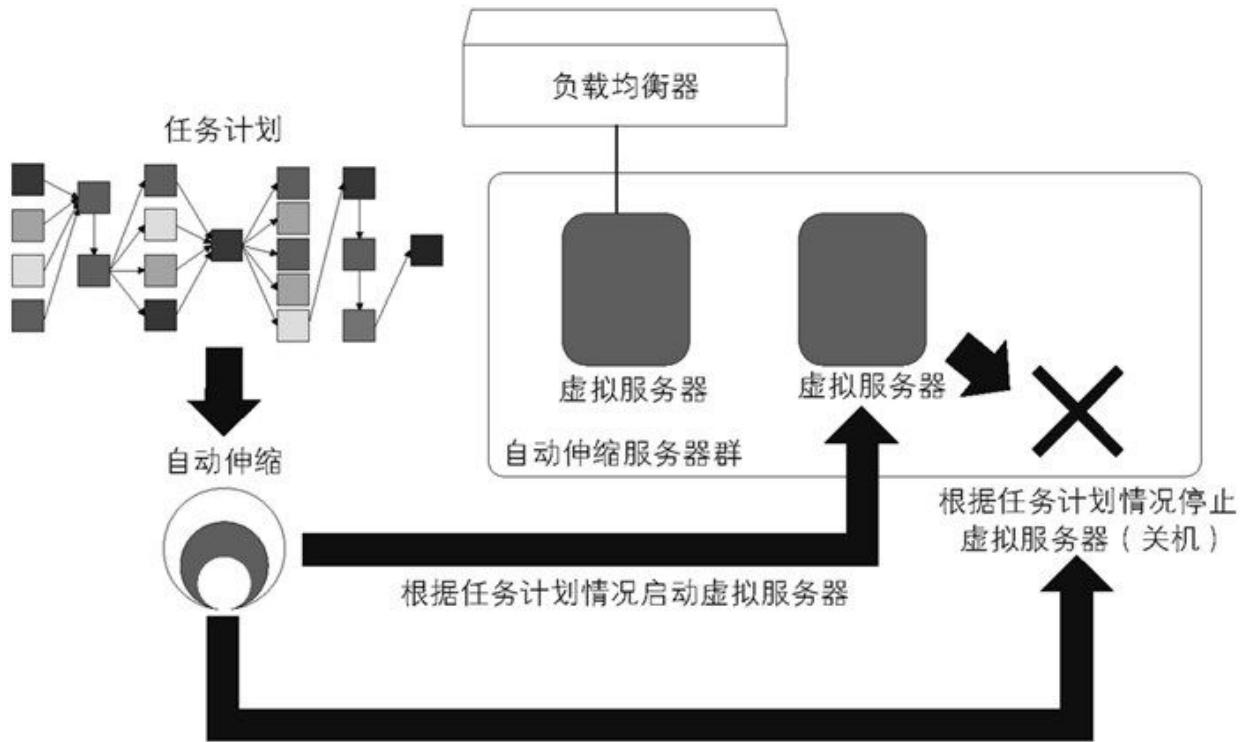


图 7.4 任务应对横向伸缩模式

使用前面讲解的自动伸缩功能，在一系列任务计划运行过程中，当负荷较高的任务运行时启动虚拟服务器，当负荷较低的任务运行时停止虚拟服务器。这样，可以防止批处理任务运行过程中服务器资源的浪费。

计划应对横向伸缩模式

使用前面讲解的自动伸缩功能，在指定的时间自动地变更虚拟服务器的数量（图 7.5）。

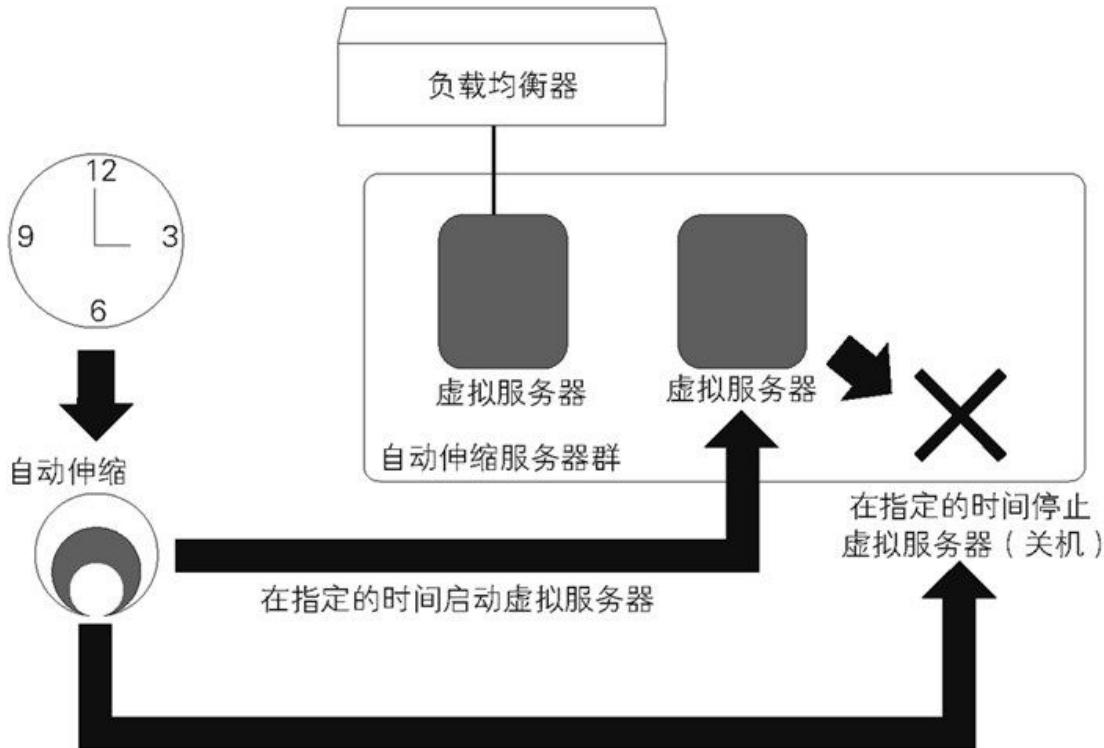


图 7.5 计划应对横向伸缩模式

当检测到系统负荷在数分钟内突增的时候，即使增加了虚拟服务器，如果不能及时启动，还是会造系统性能下降。如果能够事先预测负荷突增的时间，并指定在这个时间进行横向扩展，就可以有效防止性能下降。

手动横向伸缩模式

事先设置增减服务器的系统负荷阈值，由系统管理员监控系统负荷，并在合适的时候手动变更虚拟服务器数量（图 7.6）。

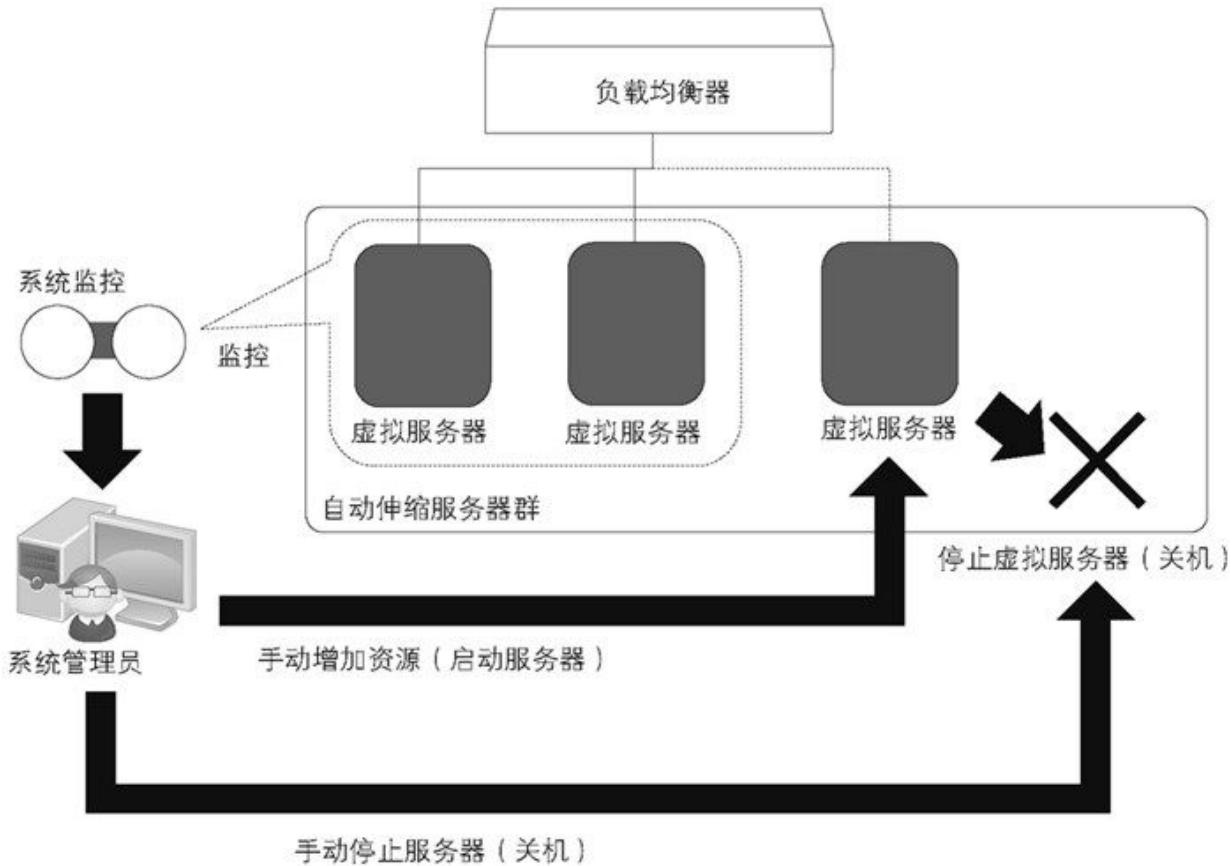


图 7.6 手动横向伸缩模式

当无法使用自动伸缩功能自动变更虚拟服务器数量时可以采用该模式。无法使用自动伸缩功能是指客户端所连接的虚拟服务器固定的情况（推送功能等），以及因为 IP 地址设置了访问限制，需要在扩展后重新进行相关设置的情况等。

各模式的比较结果

我们将各模式的比较结果总结在表 7.1 中。

表 7.1 云服务中性能与可扩展性的设计方式的比较结果

比较	服务器纵向伸缩	磁盘资源量增减	自动伸缩	任务应对横向伸缩	计划应对横向伸缩	手动横向伸缩
----	---------	---------	------	----------	----------	--------

项目					
实现方法	系统管理员进行伸缩作业	通过结合资源使用情况监控功能和可自动增减虚拟服务器的自动伸缩功能实现	通过结合任务计划和可自动增减虚拟服务器的自动伸缩功能实现	通过给可自动增减虚拟服务器的自动伸缩功能的设置运行时间来实现	系统管理员依据业务特性等综合判断是否需要伸缩，如果需要则进行伸缩作业
增减的资源	虚拟CPU 虚拟内存	虚拟磁盘容量 虚拟磁盘数量	虚拟服务器		
手动/自动	手动 系统管理员手动调整系统资源	自动 系统的资源使用情况满足条件时自动调整资源量	自动 根据任务运行状况自动调整资源量	自动 根据事先设置的时间计划自动调整资源量	手动 系统管理员手动调整资源量
系统是否停止	x: 停止 因为需要暂时停止虚拟服务器，所以如果只有一台服务器提供系统服务，那么系统就会停止	o: 不停止 可以在不对运行中的虚拟服务器造成影响的情况下增加或是移除虚拟服务器，不会导致系统停止			

注意点

自动伸缩导致成本增加

根据系统的使用情况进行自动伸缩可能会导致虚拟服务器数量不稳定，在使用某些云服务器时可能会导致产生额外费用。这种情况下可以考虑通过设置“自动扩展，手动收缩”或是“规定更加严格的收缩条件”等措施在确保性能的同时，尽量防止服务器数量出现波动。

虚拟服务器的收费体系

使用虚拟服务器时，有些云服务是按照使用时间收费的，有些情况下短时间使用服务器也按照 1 小时收费。因此，在实现任务应对横向收缩模式或是计划应对横向收缩模式时，需要特别注意服务器启动和停止的时间点。

7.2 云服务中备份的可用性设计方式

系统发生故障时，如果没有获取系统和数据的备份，就会导致系统无法恢复从而使业务恢复时间延长。

云服务中大多提供了可以简单复制系统环境和数据的功能，有些云服务还提供了在多个地区之间进行复制的功能。通过使用这些功能，可以以低廉价格实现高可用性的备份策略。

快照模式

将某个时间点的 OS 和数据作为快照保存下来（图 7.7）。

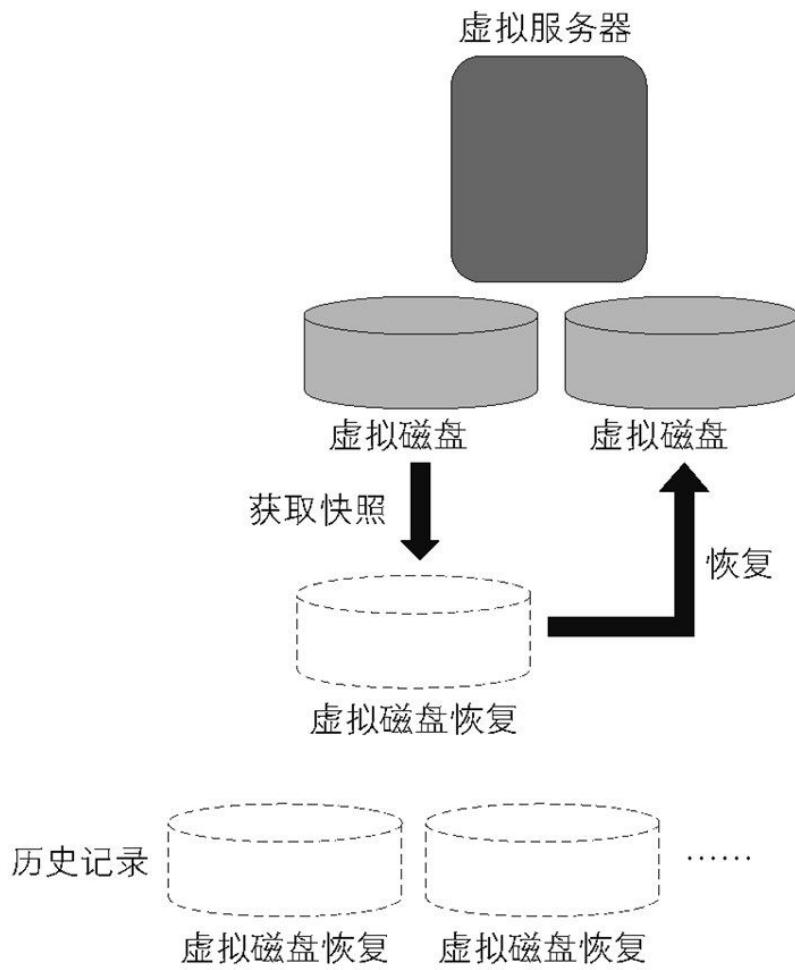


图 7.7 快照模式

许多云服务都提供了快照相关功能，使用快照可以很容易地进行 OS 和数据的备份和恢复。而且，通过编写程序定期获取备份还可以实现备份自动化。另外，该模式也可以获取多份备份，以及保存备份历史记录。

数据中心复制模式

跨数据中心实时进行 DB 复制（图 7.8）。

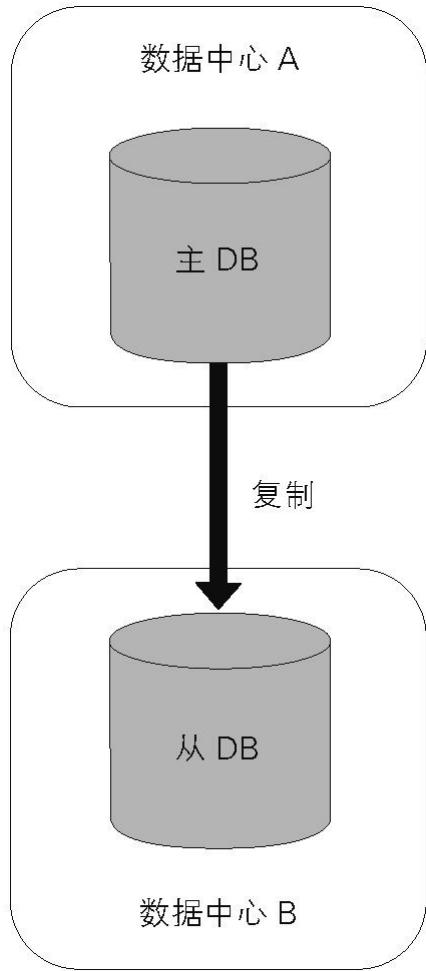


图 7.8 数据中心复制模式

有些云服务在多个地区的数据中心中都提供计算资源，用户使用这些云服务时可以以低廉的价格简单地使用多个地区的计算资源。该模式借助于云服务的这种特点，进行跨数据中心的 DB 数据复制，这样即使其中某个数据中心发生故障，也可以保护重要数据不会丢失。

地区间高速数据传输模式

跨地区进行数据备份（图 7.9）。

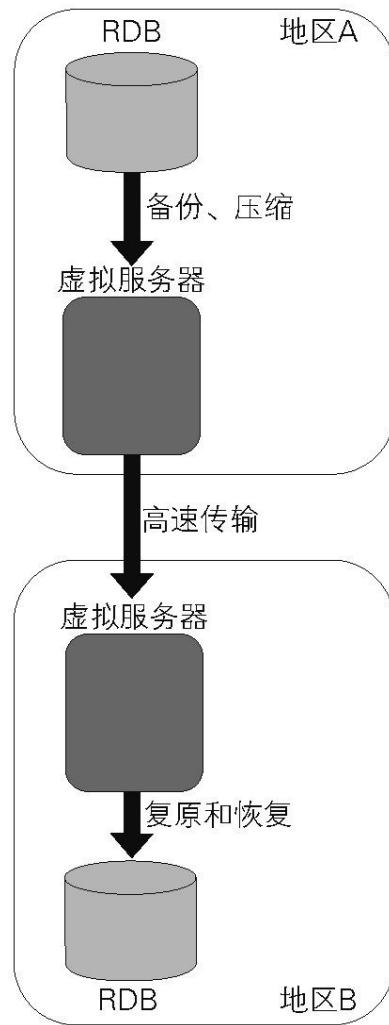


图 7.9 地区间高速数据传输模式

与东京和大阪之间的数据备份相比，东京和美国之间的数据备份所需时间更长。如果备份时数据传输的时间过长，可能就无法在计划时间内完成备份工作。

该模式通过将某个地区的数据先备份、压缩，然后高速传输到其他地区的虚拟服务器上进行复原和恢复，来保证当某个特定地区发生故障时，数据也不会丢失。

跨地区进行高速数据传输时使用 UDP (User Datagram Protocol, 用户数据报协议)¹ 传输协议。

¹ 互联网中主要的通信规范，具有处理速度快的特点，可实现高速通信。

虚拟服务器复制模式

对设置完成的虚拟服务器进行备份（图 7.10）。

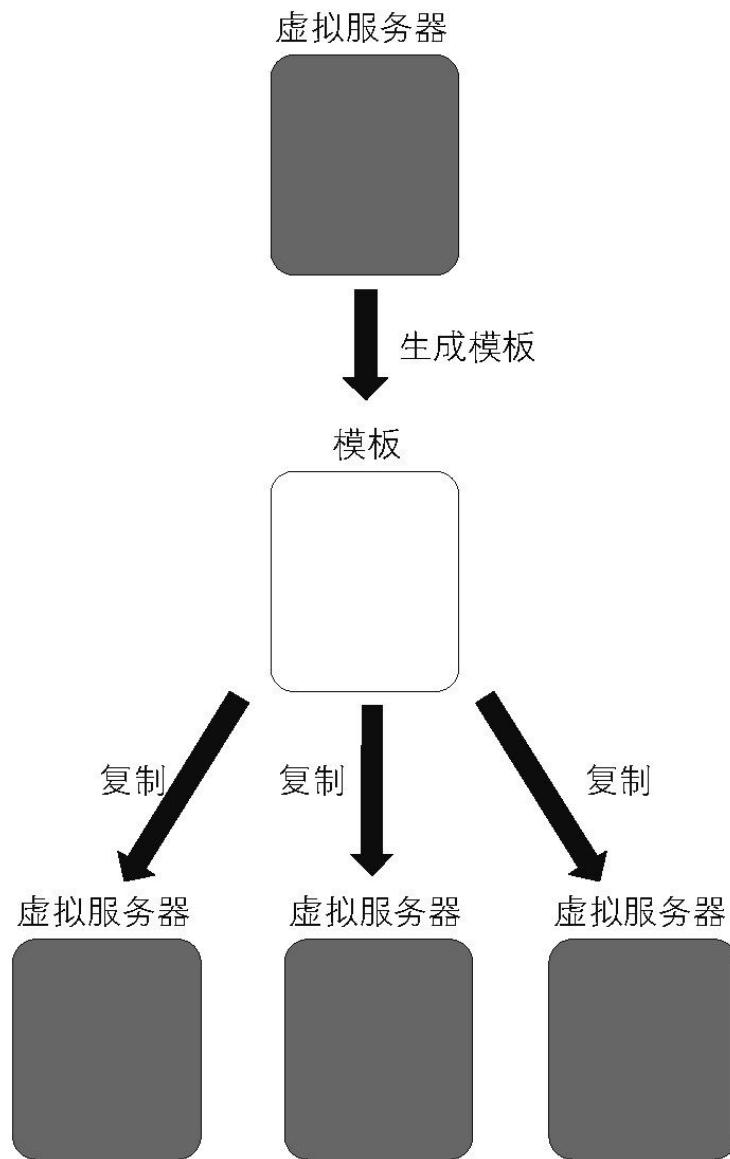


图 7.10 虚拟服务器复制模式

许多云服务都提供模板功能。将 OS 和中间件、AP 服务器等设置完毕的虚拟服务器作为模板，通过使用该模板启动虚拟服务器，即可实现

虚拟服务器的复制（备份）。

由于该模式中复制了设置完全相同的虚拟服务器，而有些设置无法在安装中间件后进行变更。因此如果想要在备份后改变虚拟服务器的设置，就需要在备份前确认安装了中间件后是否还可以改变设置。此外，OS 的补丁和版本升级可能无法反映到模板中，在设计时需要注意这点。

系统复制模式

对整体系统进行备份（图 7.11）。

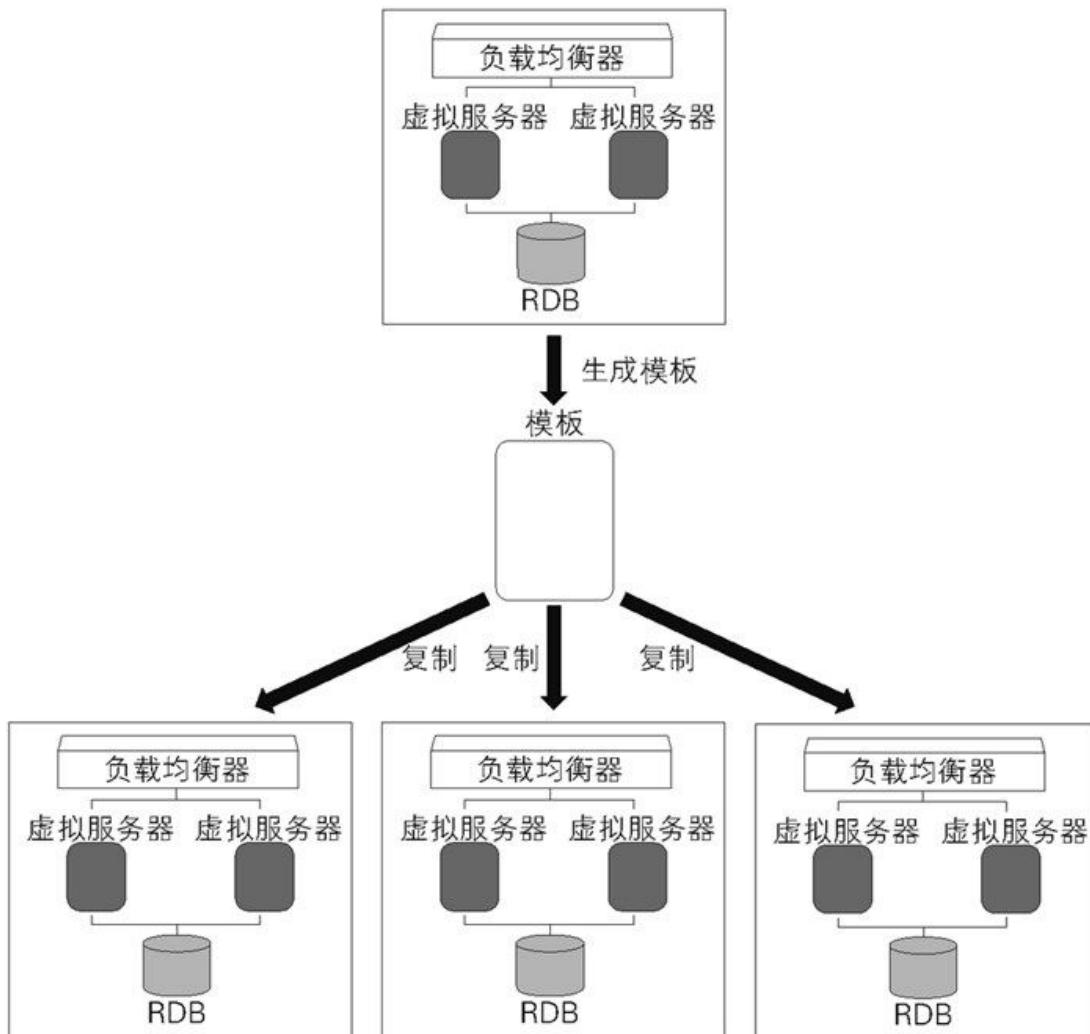


图 7.11 系统复制模式

某些云服务可以将整体系统的组成和构建方法保存为模板。使用这些云服务时，可以很容易地进行整体系统的复制（备份）。

各模式的比较结果

我们将各模式的比较结果总结在表 7.2 中。

表 7.2 云服务中备份的可用性设计方式的比较结果

比较项目	快照	数据中心复制	地区间高速数据传输	虚拟服务器复制	系统复制
实现方式	使用快照功能获取虚拟磁盘的快照	在不同地区分别设置主DB和从DB，将主DB的数据复制到从DB中	在不同地区分别设置具有高速传输功能和数据压缩、重复数据排除功能的虚拟服务器，使它们之间相互传输数据	系统管理员将设置完毕的虚拟服务器作为模板，然后使用模板进行复制	系统管理员将整个系统的组成结构作为模板，然后使用模板进行复制
备份对象	虚拟磁盘（包含业务数据和DB）	DB		虚拟服务器	整个系统结构
备份所需时间	○：短 依赖于保存快照设备的存储性能	◎：极短 主DB的变更异步反映到从DB中		○：短 需要数分钟将虚拟服务器生成为模板	○：短 需要数分钟将系统组成结构生成为模板
RTO	○：短 仅需虚拟磁盘的连接时间	◎：极短 仅需不同地区间失效转移的时间		○：短 仅需虚拟服务器模板的复制时间	△：稍长 系统组成结构的复制需要的时间稍长
RPO	○：数小时~1日前 依赖于备	◎：数秒前 故障时的事务有可能没有反映到从DB中		△：数日~数月前 依赖于备份时间间隔，多数	

	份时间间隔，多数情况下每隔数小时~1日备份一次			情况下每隔数日~数月备份一次
受灾时数据的可靠性	<p><input checked="" type="radio"/> 低 如果数据中心所在地受灾，无法确保备份数据的可靠性</p> <p><input type="radio"/> 普通 数据中心的距离越远，数据的可靠性越高</p> <p><input checked="" type="radio"/> 高 地区间的距离越远，数据的可靠性越高</p>			<p><input checked="" type="radio"/> 低 如果数据中心所在地受灾，无法确保备份数据的可靠性</p>

注意点

没有特别需要注意的地方。

7.3 云服务中虚拟服务器的可用性设计方式

云计算中虚拟服务器也可能会发生故障，因此在云服务中提供了实现了虚拟服务器冗余的功能。通过使用该功能实现虚拟服务器冗余，服务器发生故障时也可以继续提供服务。

虚拟服务器冗余模式

使用多台虚拟服务器，这样当其中1台虚拟服务器发生故障时，系统不会停止，可继续提供服务（图7.12）。

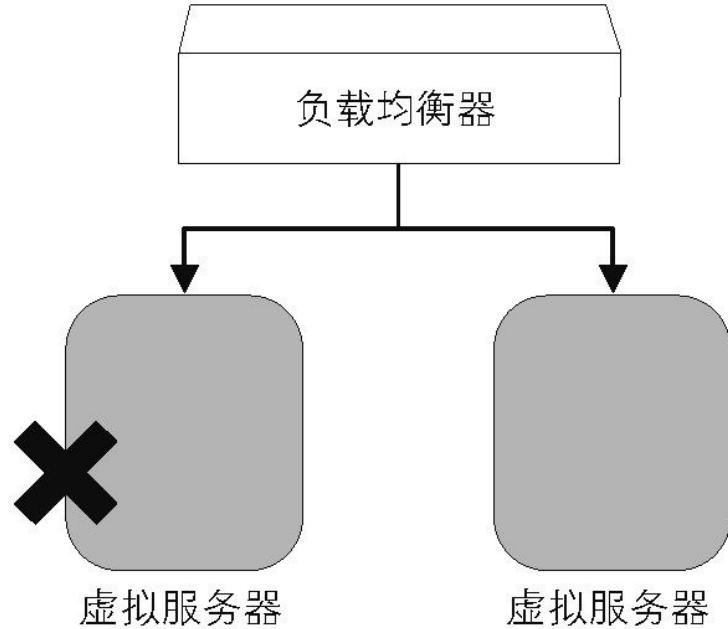


图 7.12 虚拟服务器冗余模式

使用负载均衡器接收发往虚拟服务器的处理请求，然后分派给虚拟服务器处理。当其中 1 台虚拟服务器发生故障时，负载均衡器会检测出故障，并自动地停止向故障服务器发送请求，确保系统可继续提供服务。

许多云服务中都可以根据请求数量来支付使用负载均衡器的费用，因此可以很容易地实现服务器的冗余。

路由变更模式

通过变更路由设置将网络连接从发生故障的虚拟服务器切换到正常的虚拟服务器（图 7.13）。

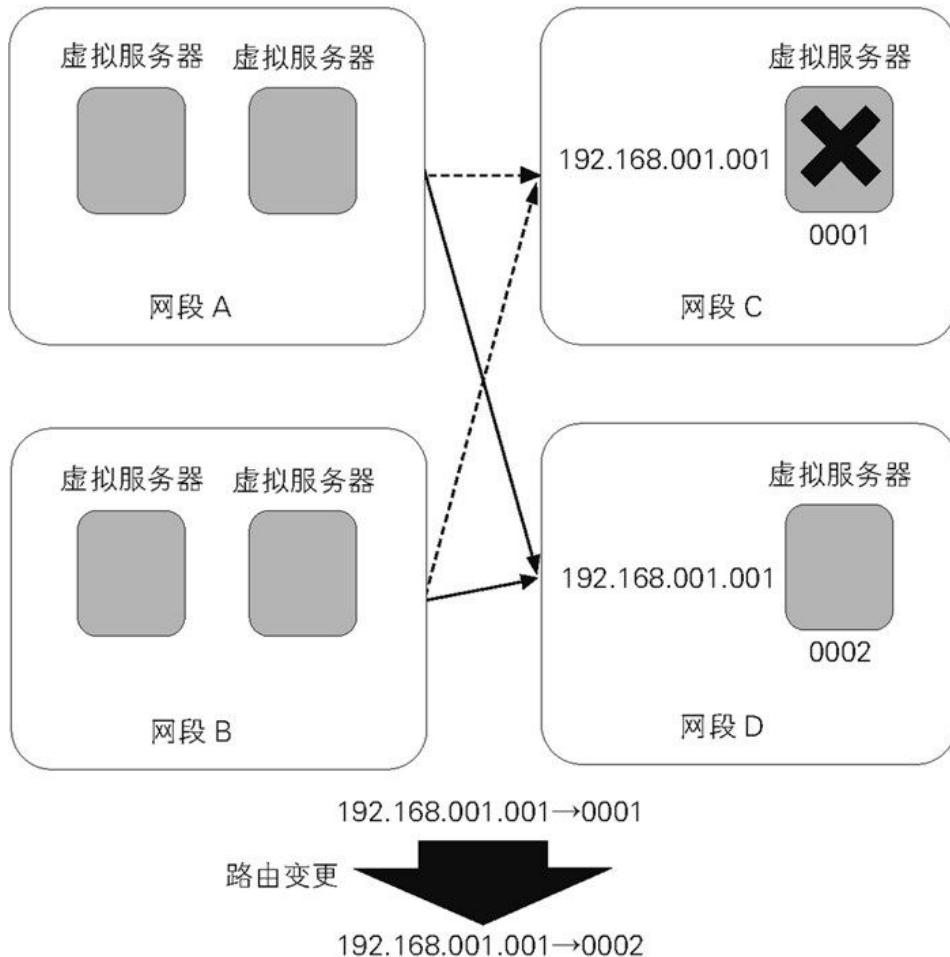


图 7.13 路由变更模式

在实现跨网段或是跨数据中心的失效转移时可使用此模式。首先需要准备多台具备相同功能的虚拟服务器，并为它们分配 IP 地址，然后需要确定与当前工作中的虚拟服务器的路由。这样当 1 台虚拟服务器发生故障时，可以切换路由，使得业务处理不会中断。

备份站点切换模式

在故障发生时自动地切换到备份站点（图 7.14）。

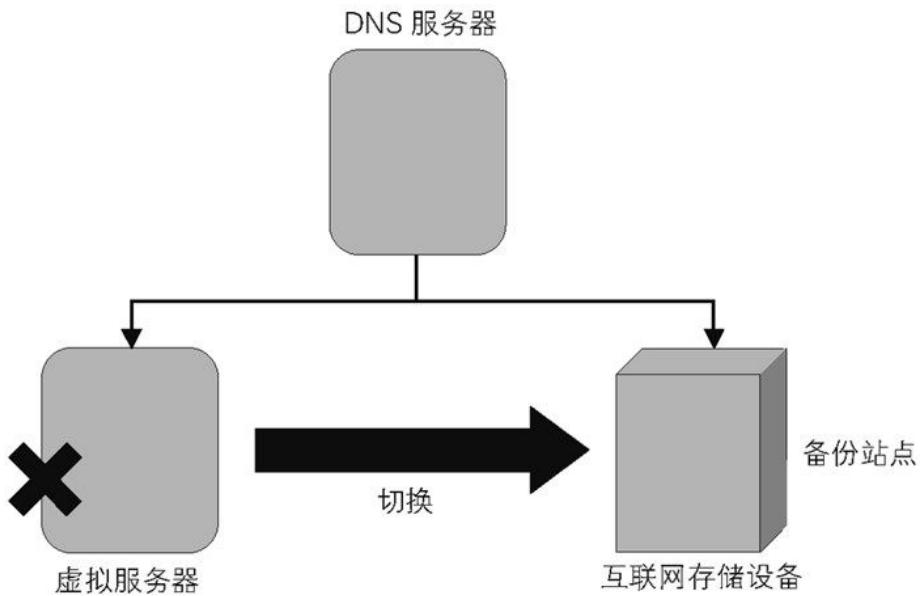


图 7.14 备份站点切换模式

许多云服务都提供互联网存储功能，有一些云服务还允许在存储设备上部署静态站点。此外，许多云服务还提供 DNS 功能，可以进行健康检查²。因此，可以使用互联网存储功能部署备份站点，当主站点发生故障时，DNS 会检测到故障，并自动地切换到备份站点。

² 可以确认服务器的运行情况，并掌握哪些服务发生故障、哪些服务器正常工作的功能。

固定 IP 替换模式

通过替换固定 IP 地址来应对故障（图 7.15）。

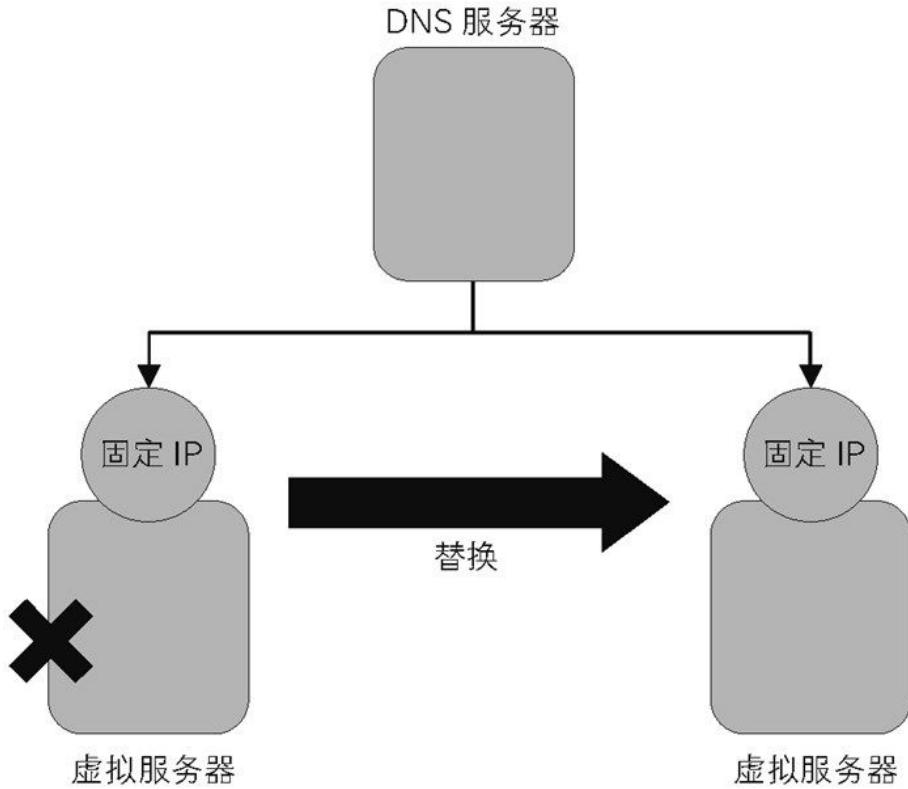


图 7.15 固定 IP 替换模式

使用虚拟服务器复制模式，在复制完成的虚拟服务器上替换固定 IP。当故障发生时，可以在设置完全相同的虚拟服务器上继续处理业务。

需要注意的是固定 IP 的替换会花费一些时间。此外，SSH 连接到该固定 IP 时，由于 IP 相同但服务器不同，可能会弹出警告信息提示账号被盗用，导致无法登录。

NAT 服务器冗余模式

构建和冗余带有 NAT（Network Address Translation，网络地址转换）³ 功能的服务器，切换路由（图 7.16）。

³ 指的是通过网络地址转换技术，将仅在内部网络通用的私有 IP 地址转换为可以连接互联网的全局 IP 地址的技术。

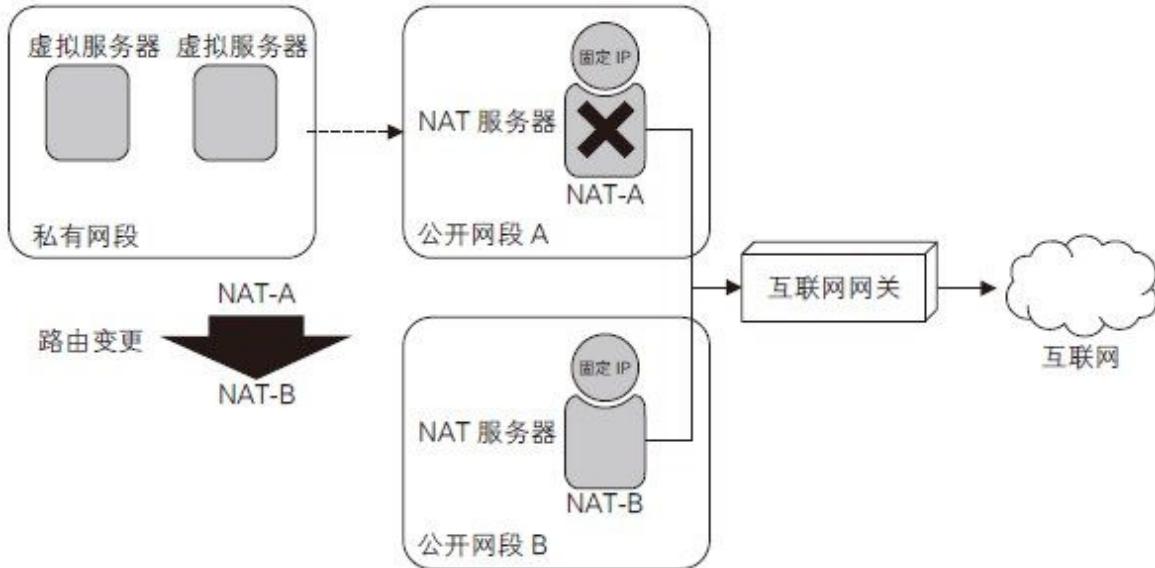


图 7.16 NAT 服务器冗余模式

当内网的服务器通过代理服务器⁴连接互联网时，代理服务器很容易成为 SPOF。使用 NAT 实现代理服务器，并采用 NAT 服务器冗余结构，当生产环境的 NAT 发生故障停止工作时，只需要将路由目标切换至备用 NAT 服务器上，就可以轻松应对 NAT 相关的故障，使业务处理继续进行。

⁴ 指的是在无法直接连接互联网的内部网络中，作为代理连接互联网的服务器。

由于 NAT 服务器很难横向扩展，在设计时需要注意性能问题和可扩展性问题。

虚拟磁盘替换模式

通过替换虚拟磁盘应对故障（图 7.17）。

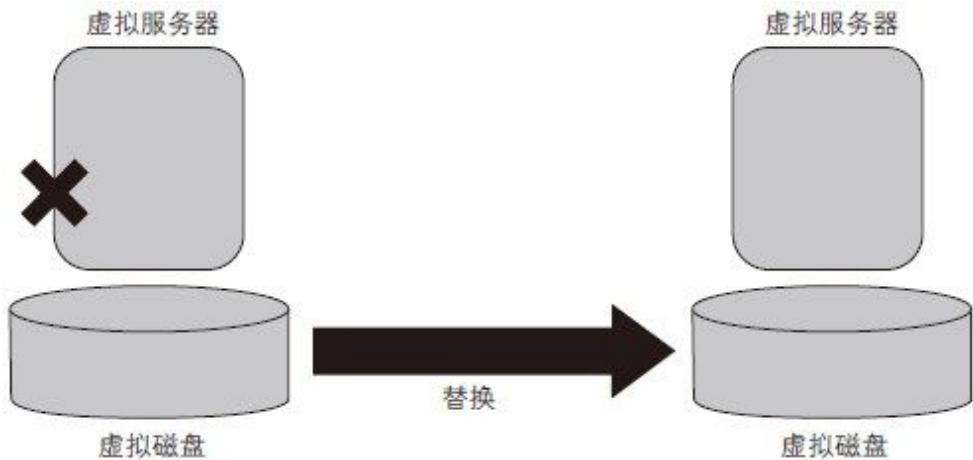


图 7.17 虚拟磁盘替换模式

使用虚拟服务器复制模式，在复制完成的虚拟服务器上替换虚拟磁盘。当故障发生时，可以在设置完全相同的虚拟服务器上继续业务处理。

当虚拟服务器发生故障、但虚拟磁盘自身并没有故障时可以采用该模式。

各模式的比较结果

我们将各模式的比较结果总结在表 7.3 中。

表 7.3 云服务中虚拟服务器的可用性设计方式的比较结果

比较项目	虚拟服务器冗余	路由变更	备份站点切换	固定IP替换	NAT服务器冗余	虚拟磁盘替换
实现方式	通过负载均衡器对横向扩展的虚拟服务器进行健康检查，停止向发生故障的虚拟	系统管理员手动变更路由目标地址 (有些云服务也具	通过DNS服务器对虚拟服务器进行健康检查，当检测到发生故障时自动将DNS目标地址切换到互	系统管理员启动备用虚拟服务器，并替换	系统管理员启动备用NAT服务器，并变更路由目标信息(有些云服务也具备	系统管理员启动新的虚拟服务

	服务器转发处理请求	备自动变更功能)	联网存储设备上的备份站点	固定IP地址	自动变更功能)	器，并替换虚拟磁盘
针对的故障	虚拟服务器宕机	虚拟服务器群宕机	整体系统宕机	虚拟服务器宕机		
从检测出故障到服务恢复的时间	◎：无 由于采用横向扩展，当服务器发生故障时仍可以继续提供服务	△：稍长 手动进行切换作业，需要稍长时间	○：短 DNS服务器检测到故障后立即切换到备份站点	△：稍长 手动进行切换作业，需要稍长时间		
成本(使用云服务成本)	△：稍高 横向扩展会增加负载均衡器和虚拟服务器的使用成本	○：稍低 备用的虚拟服务器处于待机状态，如果不启动备用虚拟服务器，可以减少成本	○：低 不需要备用虚拟服务器，仅使用互联存储设备，降低了成本	◎：低 没有启动备用的虚拟服务器，降低了成本	○：稍低 备用的虚拟服务器处于待机状态，如果不启动备用虚拟服务器，可以减少成本	◎：低 没有启动备用的虚拟服务器，降低了成本
故障时是否进入降级运行模式	△：部分降级运行 处理性能会根据发生故障的虚拟服务器相应下降	○：无降级运行	×：降级运行 故障发生时向用户显示备用站点	○：无降级运行		

注意点

没有特别需要注意的地方。

7.4 总结

本章讲解了使用云服务的设计方式。通过使用云服务，可以以低廉的成本快速地构建系统，来及时且弹性地应对商业速度需求和环境的变化。但是在设计云服务的使用方式时，必须注意要摒弃传统的设计思想，充分活用云服务所提供的功能。

第8章 基于模式的设计实践

本章将通过一个设计实例来介绍基础设施构成的讨论流程，具体会对第1章介绍的基于模式的设计方法进行详细说明，以帮助读者更好地理解各个作业阶段应该如何使用基础设施设计模式。

需要说明的是，设计实例只是为了帮助读者更好地理解基础设施的讨论过程，因此我们并不会详细地介绍设计实例¹。读者在讨论基础设施构成时可将它作为参考，但是还请务必针对实际需求进行充分的分析和讨论。

¹ 实际设计中，一个设计就可能需要很厚的几本文件来说明。

本章中，我们首先介绍基础设施构成的讨论步骤，然后结合实例介绍如何在地理信息系统和综合DB系统中使用基础设施设计模式。

8.1 基础设施构成的讨论步骤

本章将要介绍的基础设施讨论步骤如下所示。

- ① 确认业务需求和涉众需求
- ② 提取功能性需求和非功能性需求

③ 识别风险和注意点，讨论相应回应

④ 设计系统的概念构成

⑤ 设计系统的逻辑构成

以上这些步骤与图 1.3 中展示的基于模式的设计步骤相对应，其中① ~ ③ 对应“确认需求”，④ 对应“选择设计方式”，⑤ 对应“讨论基础设施构成”和“确认基础设施构成”。

① 确认业务需求和涉众需求

首先确认非功能性需求。本节中将介绍系统中典型的需求²。

² 需求是对系统使用方提出的所有要求进行客观地分析以及综合整理而成的。

在 BABOK (Business Analysis Body Of Knowledge, 业务分析知识体系指南)³ 中对业务需求和涉众需求分开进行了讲解。所谓业务需求是指使用该系统想要达到什么目的、满足什么需要。例如，“想在三年后削减成本 10%”等需求属于业务需求。与实现业务需求相关的部门所提出的需求则属于涉众需求。例如，“为了实现 3 年后削减成本 10% 的目标，需要简化系统操作，削减工作量”等需求属于涉众需求。

³ 书中总结了明确开发和引进的解决方案的目的和需求（业务需求分析），并在涉众间搭建沟通的桥梁时所需的技能和任务。

收集完成所有的需求后，需要整理各需求之间的关系。需求中可能既包含类似的需求（相似关系），也包含相互矛盾的需求（对立关系）。例如在构建 EC (Electronic Commerce, 电子商务) 站点时，涉众提出的“想要管理畅销商品（利基产品）以外的商品”需求和“想要减少出库、配送管理工作的工作量”需求这两者是相互矛盾的，因为“想要管理畅销商品（利基产品）以外的商品”就会增加需要管理的商品的数量，导致出库及配送管理工作量的增加。

本书中会将系统涉众的需求之间的关系以示意图展示出来。

② 提取功能性需求和非功能性需求

整理和提取需求（功能性需求和非功能性需求）。

对于具有相似关系的需求可将其合并为同一个需求。

对于具有对立关系的需求，需要讨论解决办法。例如，对于“想要管理畅销商品（利基产品）以外的商品”需求和“想要减少出库、配送管理工作量”需求，可考虑实现库存管理和出库、配送管理自动化等方法。以上属于功能性需求。

非功能需求可以在整理功能需求的过程中从可用性、性能与可扩展性、安全性等方面去提取。例如，刚才提到的 EC 站点具有“可以通过网络查询商品和下订单”的功能性需求，那么从性能与可扩展性的方面看，随之而来的非功能性需求就包括“页面响应速度必须达到让用户满意的水平”等。如果可以从多个功能性需求中提取出相同类型的非功能性需求，则需要从中选择要求最严格的非功能性需求或是以它们的总和作为非功能需求。详细方法可以参考 BABOK 和需求工程，以及使用 UML（Unified Modeling Language，统一建模语言）⁴ 的设计方法等。

⁴ 指的是 OMG（Object Management Group，对象管理组织）开发的面向对象建模方法及其成果。使用方框和线条来抽象表示并简化处理目标和处理程序（对象）之间的关系和结构，以此使最佳解决方案的讨论变得更轻松。

本章中会将各系统的典型功能性需求和非功能性需求整理在一览表中。

③ 识别风险和注意点，讨论对策

通过功能性需求设想可能会存在哪些风险和注意点（问题及需要讨论的事情），并进行整理。

找到可能存在的风险和注意点后，整理它们与需求属于相似关系还是对立关系，讨论对策并落实到需求中。还是以 EC 站点为例，对于提取出的“有可能因系统所收集的个人信息、信用卡号等保密信息泄露导致经济赔偿以及企业信用、形象受损”的风险，可以考虑采取“防止外部非法入侵策略、防止信息泄露策略等安全性策略”，并进而可以提取出“防止第三者获取个人信息和信用卡号等保密信息”“设置合适的访问权限，防止非法访问”等非功能性需求。

本章中会将各系统的典型风险和注意点以及它们的对策整理在一览表中。

④ 设计系统的概念构成

根据功能性需求讨论所构建的系统应当具有的功能。

首先，通过功能性需求设想需要实现的功能。

列举出所有需要实现的功能后，将功能分组。还是以 EC 站点为例，根据功能需求可以知道系统中需要实现下单功能、支付功能、站点的内容管理功能、库存查询和存货预留等库存管理功能以及销售管理功能等，其中下单功能、支付功能、站点的内容管理功能可以组合为“前端系统”，而库存管理功能、销售管理功能可以组合为“管理功能”⁵。

⁵ 实际设计过程中，大部分人会用 UML 中的用例图来明确系统中的功能，然后使用场景（剧本）给功能分组。但本章中为了使读者更容易理解基础设施构成的讨论流程，我们使用设想功能后给功能分组的方式进行讲解。

接着整理各功能之间（数据的传递）的关系。将功能构分解后讨论所必需实现的应用程序（本书中不涉及）。

本章中会将定义好的功能以及它们对应的功能性需求整理在一览表中。同时，我们会用系统概念构成图将各功能之间的关系展示出来。

⑤ 设计系统的逻辑构成

整理出系统所需要的功能后，就需要考虑非功能性需求的设计了。本章中使用基础设施模式来讨论最佳设计，通过查阅各模式的选择标准和各模式之间的比较表，可以选择出合适的模式。例如，对于“防止第三者获取个人信息和信用卡号等保密信息”这条非功能需求，可采用数据加密模式、通信加密模式等。另外，在第1章中已经强调过了，在选择合适的设计方式时，除了查阅各模式的选择标准外，还必须通过示意图和比较表充分理解各种设计模式的特点和注意点。

本章中会将选择非功能性需求和模式时的重点事项以及对应的设计方式整理在一览表中。

接下来需要组合所选择的设计模式，并画出基础设施构成图。

最后，需要与相关人员确认基础设施构成（设计方针）有无问题。

通过本章介绍的作业，可以明确需求与基础设施组成部分之间的关系。由于可以顺利地实施第1章所介绍的“需求变更的应对步骤”，因此本章介绍的基础设施构成讨论流程同样也适用于基础设施的宏观设计（从长远的角度考虑基础结构的构成、迁移计划）等。

8.2 地理信息系统

本节中，我们以物流企业为应对业务需求和顾客需求所导入的地理信息系统（GIS，Geographical Information Systems）的简易模型为例，讲解如何使用基础设施设计模式讨论其基础设施构成。

地理信息系统是通过高度分析地理信息数据，并与地图数据进行映射，以此为企业决策提供支持的系统的总称，广泛应用于道路规划、城市规划、防沙调查、环境和防灾评估等领域。

① 确认业务需求和涉众需求

地理信息系统的业务需求和涉众需求示例参见表8.1和图8.1。

表 8.1 业务需求和涉众需求示例

需求	内容	提出者
业务需求	物流行业竞争激烈，所以想通过提高服务的附加值和优化物流成本来提高企业竞争力	总经理
涉众需求①	委托物流公司收货的当天，由于不知道具体什么时候来取货而无法外出，希望能在指定的时间范围内尽早来取货	顾客
涉众需求②	现在的配送路线效率很低，希望能优化配送计划，以及在业务终端画面上可以确认配送路线	物流管理部经理
涉众需求③	对于当天的收货要求，可以分析出最近的配送员，弹性地变更配送计划	
涉众需求④	关于业务计划，希望即使将来送货量和收货量都增加了，送货和收货也不会发生延误	配送员
涉众需求⑤	为了防止漏发货和漏收货，在配送计划发生变更时，配送员可以随时确认情况	
涉众需求⑥	收货信息和发货信息都是个人保密信息，必须防止泄露	系统部经理
涉众需求⑦	配送业务全年365天都在进行，因此希望系统稳定运行	

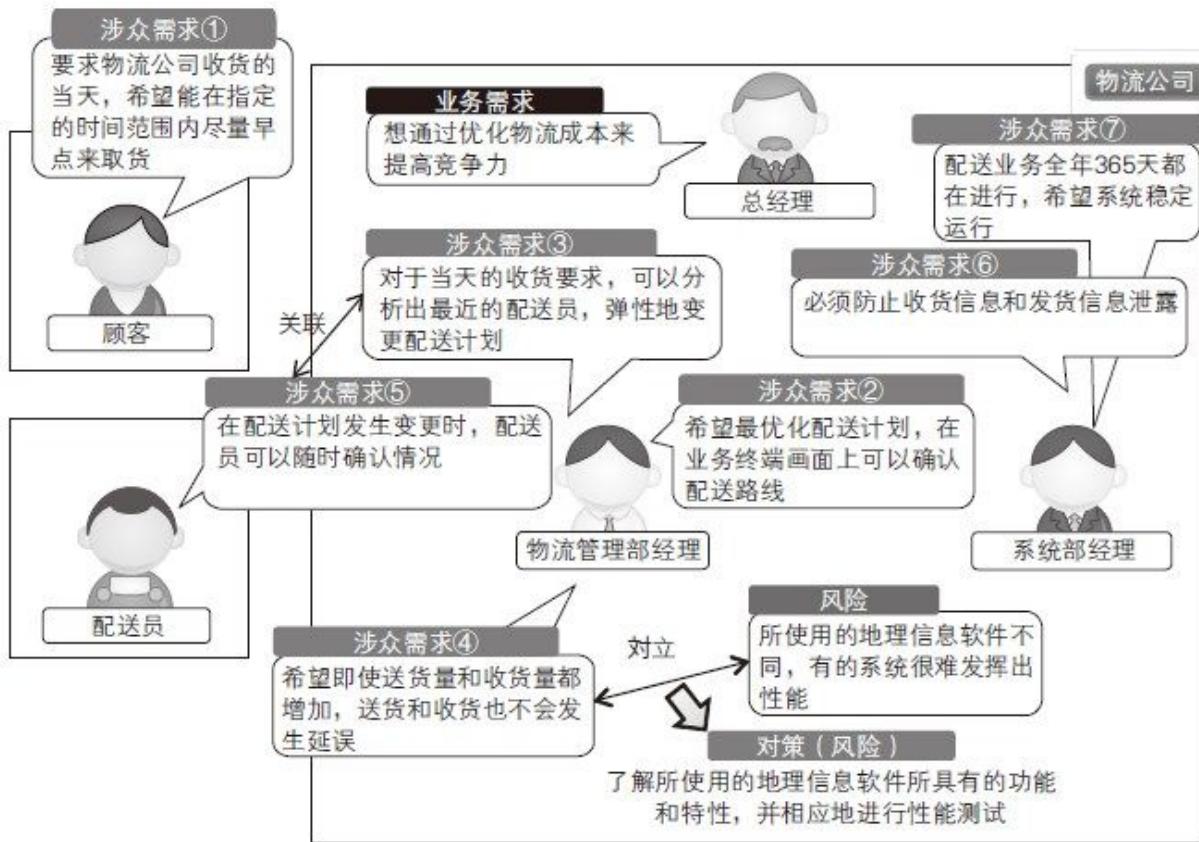


图 8.1 涉众关系

② 提取功能性需求和非功能性需求

提取功能性需求

根据业务需求和涉众需求整理功能性需求（表 8.2）。

表 8.2 功能性需求一览

功能性需求	内容	对应的 需求

功能性需求	内容	对应的需求
功能性需求①	可以按照时间范围分析最佳配送计划，并显示在地图上	涉众需求② 涉众需求③
功能性需求②	可以定位配送员现在所处的位置	涉众需求① 涉众需求③
功能性需求③	可以通过公司内部的PC或是外出时通过智能手机的浏览器访问地理信息系统	涉众需求③ 涉众需求⑤

提取非功能性需求

根据业务需求和涉众需求整理非功能需求（表 8.3）。

表 8.3 非功能需求一览

非功能性需求	内容	对应的需求
可用性	发生单点故障时也可以继续处理	涉众需求⑦
	当系统发生故障时，可以以最小的手工作业量恢复数据，迅速使业务恢复正常	
性能与可扩展性	在根据业务计划设想的业务量范围内，响应速度必须满足性能目标要求	涉众需求⑤
运用与维护性	系统发生故障时可以尽快检测出故障	涉众需求⑦

安全性	采取防止外部入侵系统的策略	涉众需求⑥
	采取防止信息泄露的策略	

③ 识别风险和注意点，讨论对策

整理地理信息系统构建中可能存在的风险和注意点，并讨论对策（表 8.4）。

表 8.4 风险和注意点及其对策

内容	主要对策
风险	
地理信息软件提供的功能和地理信息可能不足，或是性能可能不足，导致无法满足需求	仔细、充分地确认需求中所需的功能和地理信息，谨慎选择地理信息软件。必要时对地理信息软件进行性能测试
注意点	
使用GPS获取配送员现在所处的位置时，需要考虑系统间交互的开发成本和为配送员提供的终端的成本	如有现有系统（POS等），则可通过现有系统分析和推断出配送员所处的位置的大致范围

④ 设计系统的概念构成

根据提取功能性需求时整理出的功能性需求一览（表 8.2），确定系统范围和要实现的功能，以及功能所对应的基础设施的名称，设计系统的概念构成（图 8.2、表 8.5）。

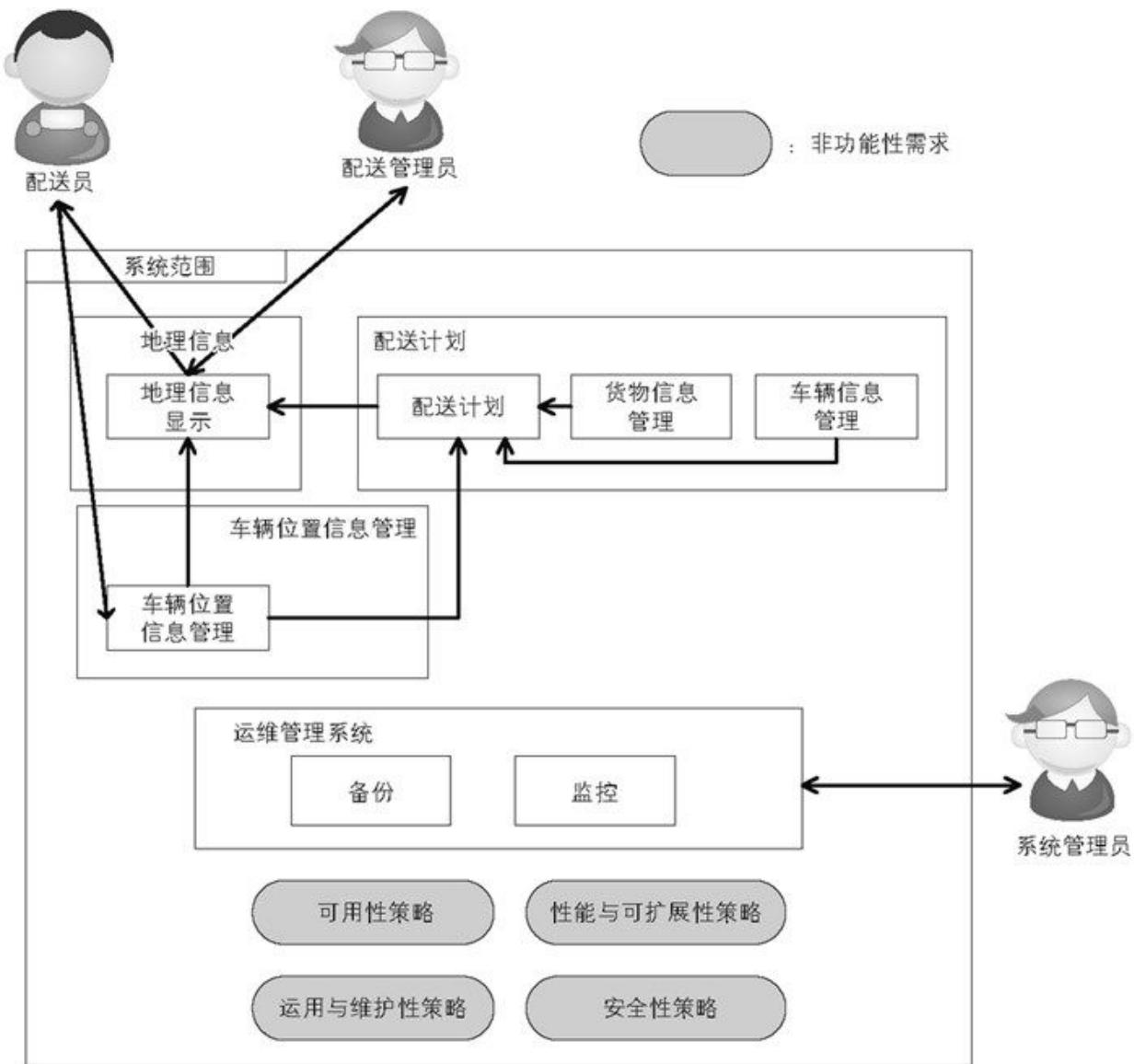


图 8.2 系统概念构成图

表 8.5 要实现的概念功能

实现的功能	功能内容	对应的功能性需求	功能所对应的基础设施名称
地理信息			
地理信息显示功能	通过浏览器在地图上显示配送计划、配送员现在所处的位置	功能需求①	Web服务器 地图AP服务器

		功能需求③	POI ^{*1} 、地图、住址、道路信息DB 配送计划DB
配送计划			
配送计划功能	根据读取的货物信息和车辆信息制定配送计划（配送线路）		Web服务器 配送计划AP服务器 POI ^{*1} 、地图、住址、道路信息DB 配送计划DB
货物信息管理功能	对货物信息（货物重量和容量、货物信息、配送信息等）进行录入、变更、删除	功能需求①	Web服务器 配送计划AP服务器 配送计划DB
车辆信息管理功能	对车辆信息（车辆、载重、有无冷柜、驾驶员信息等）进行录入、变更、删除		Web服务器 配送计划AP服务器 配送计划DB
车辆位置信息管理			
车辆位置信息管理功能	通过车辆上安装的移动终端的GPS功能获取和管理车辆现在所处位置（坐标）的数据	功能需求②	车辆位置信息AP服务器 POI ^{*1} 、地图、住址、道路信息DB
运维管理系统			
备份功能	进行数据备份		备份服务器
监控功能	监控系统状态		监控服务器

*1 POI (Point of Interest) : 在地图画面上显示的位置信息

⑤ 设计系统的逻辑构成

根据提取非功能性需求时整理出的非功能性需求一览（表 8.3），以及之前的系统概念构成图（图 8.2）来选择设计方式（表 8.6），并完成系统逻辑构成图（图 8.3）。

表 8.6 选择设计方式

要点	设计方式	
可用性：发生单点故障时业务可以继续进行		
1 通过集群等冗余方式，可实现在不停止系统的情况下更换硬件	LAN的可用性设计方式	双机热备
	WAN的可用性设计方式	单网双链路
	互联网连接的设计方式	双链路
	Web/AP服务器的可用性设计方式	会话非共享负载均衡
	DB服务器的可用性设计方式	双机互备集群
可用性：当系统发生故障时，可以以最小的手工作业量恢复数据，迅速使业务恢复正常		
2 为了能在故障后10分钟以内恢复系统，对业务数据进行备份	数据备份的可用性设计方式	存储复制
性能与可扩展性：在根据业务计划设想的业务量范围内，响应速度必须满足性能目标要求		
3 对于业务量的增加，通过适当的扩展系统确保系统处理性能不会下降	可扩展性策略的设计方式	集群 共享磁盘
运用与维护性：系统发生故障时可以尽快检测出故障		
4 以合适成本进行必要的、充分的系统监控和系统运维工作	运用与维护体制的设计方式	服务级别提升
安全性：采取防止外部入侵系统的策略		
5 由于系统需要处理住址等个人信息，并且会接入互联网，因此需要采取更严格的防止非法入侵的策略	非法访问应对策略的设计方式	双防火墙 DMZ

6 采取策略防止感染计算机病毒	时钟同步、杀毒 软件更新的设计 方式	前端网络 自动同步
安全性：采取防止信息泄露的策略		
7 为了防止组织内部人员将数据泄露出去，需要加 密数据	信息泄露应对策 略的设计方式	数据加密
网段构成：从整体非功能性需求推演出的网络构成		
8 为了能够容易进行系统监控和容易扩展性能，采 用可以实现多段通信控制的高安全性网络构成	Web系统的网络 构成的设计方式	4网段构 成

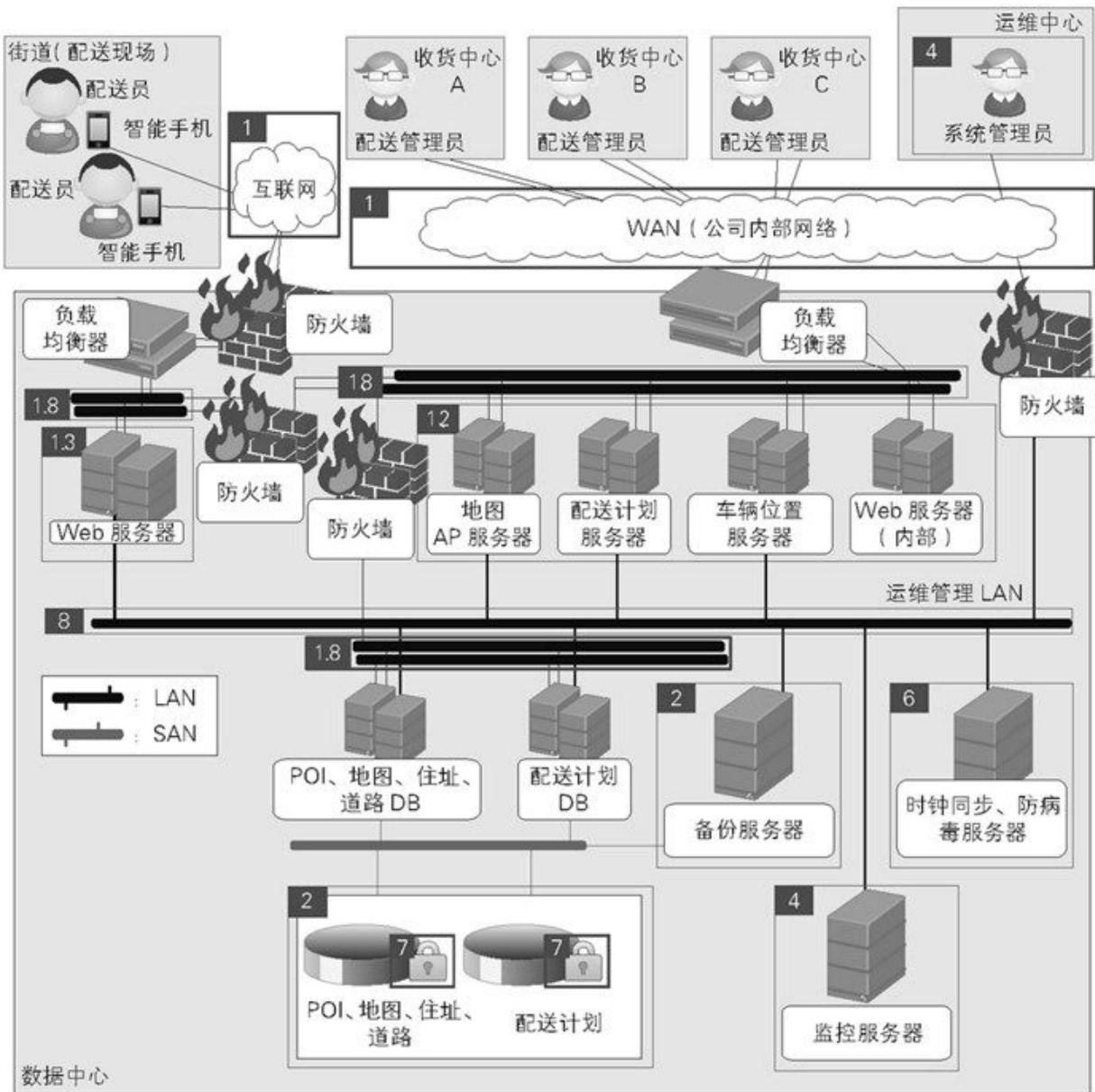


图 8.3 系统逻辑构成图

8.3 综合 DB 系统

本节中，我们以一个与多个主干系统进行间接交互的数据仓库类模型为例，讲解如何使用基础设施设计模式讨论其基础设施构成。

综合 DB 系统是专门构建的系统，旨在防止因构建多个业务系统而导致的主数据⁶管理工作重复，以及各系统之间数据不匹配等问题，从而实现优化系统整体结构。综合 DB 系统既可直接管理多个业务系统的 DB，也可另外设置主数据管理服务器，通过数据通信来间接地管理多个业务系统的 DB。

⁶ 指的是主干系统中多个 DB 共同使用的基础数据。例如“商品主数据”就是将商品名称、编号、生产日期等一般不会发生变化的属性作为主数据。

① 确认业务需求和涉众需求

构建综合 DB 系统时设想的业务需求和涉众需求示例参见表 8.7 和图 8.4。

表 8.7 业务需求和涉众需求示例

需求	内容	提出者
业务需求	系统的运维成本越来越高，导致公司收益减少，想通过削减运维成本来提高公司收益	
涉众需求①	分析业务中产生的大量数据，为企业市场决策和企业内部决策提供支持	总经理
涉众需求②	想要减少因DB过多导致的系统故障，降低系统维护的复杂性	
涉众需求③	当系统规格发生变更时，可以快速地、低成本地应对变更	
涉众需求④	综合DB系统需要与其他各种类型的多个业务系统交互，一旦其发生故障，影响就会很大，所以想要实现综合DB系统稳定运行	系统部经理
涉众需求⑤	由于各种原因，与综合DB系统交互的其他业务系统的业务量随时都在发生变化，希望综合DB系统可以弹性地应对这些变化	

涉众需求⑥	希望能确保与综合DB系统交互的其他业务系统的独立性，当发生故障和计划停机时，相互间产生的影响能降至最低	业务员
涉众需求⑦	综合DB系统保存和管理着个人信息和企业保密信息，必须防止信息泄露	系统部人员

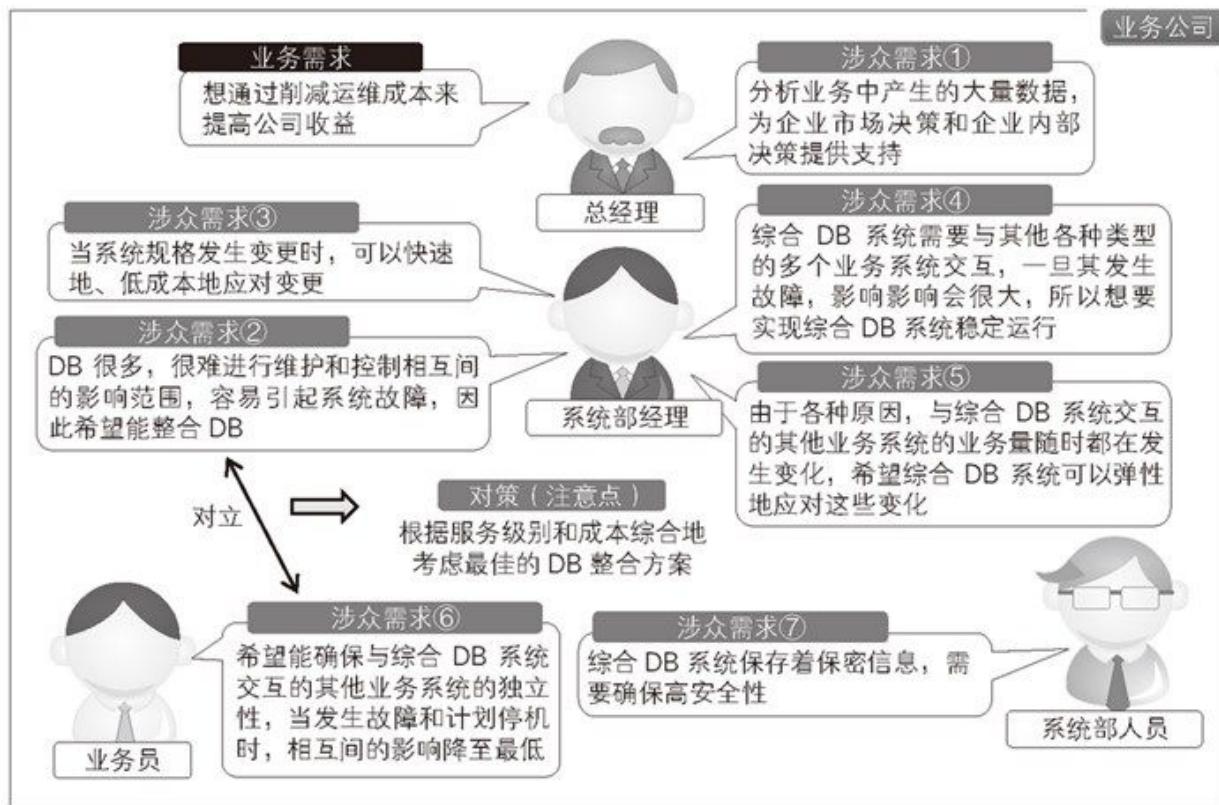


图 8.4 涉众关系

② 提取功能性需求和非功能性需求

提取功能性需求

根据业务需求和涉众需求整理功能性需求（表 8.8）。

表 8.8 功能性需求一览

功能性需求	内容	对应的需求
功能性需求①	接收和保存来自多个业务系统、外部系统的数据	涉众需求②
功能性需求②	确保数据的完整性、正确性和一致性，并进行相应的格式转换，使业务系统和外部系统可以很容易地提取数据	涉众需求③ 涉众需求⑥
功能性需求③	向多个业务系统、外部系统发送数据	涉众需求②
功能性需求④	可以在浏览器上查看、增加、更改、删除主数据	涉众需求③
功能性需求⑤	BI工具可以使用各种数据进行各种分析	涉众需求②
功能性需求⑥	可根据数据种类设置访问权限	涉众需求
功能性需求⑦	可保存和查看用户使用数据的记录	需求⑦

提取非功能性需求

根据业务需求和涉众需求整理非功能性需求（表 8.9）。

表 8.9 非功能性需求一览

非功能性需求	内容	对应的需求
可用性	发生单点故障时也可以继续处理	涉众需求④

性能与可扩展性	预计将来保存数据量和交互系统数量都会增加，必须确保系统的性能与可扩展性	涉众需求⑤
运用与维护性	将数据转移到外部媒介永久保存	涉众需求②
	进行可用监控、错误监控、资源监控、性能监控	涉众需求④
安全性	采取防止信息泄露的策略	涉众需求⑦

③ 识别风险和注意点，讨论对策

整理综合DB系统构建中可能存在的风险和注意点，并讨论对策（表8.10）。

表 8.10 风险和注意点及其对策

内容	主要对策
风险	
如果过于追求数据的标准化和DB的完全整合，可能无法确保处理性能，导致性能比整合前更差	考虑到近年来磁盘资源的成本越来越低，可通过讨论DB的整合范围来实现高性价比的综合DB系统
将信息系统DB整合到业务系统DB中的时候，信息系统不规则检索的性能可能会下降	设想检索模式，讨论是否可以优化
注意点	
如果各系统的重要性、开发和变更频率等差别很大，可能从整体来看，逻辑上整合数据结构体系、物理上不整合DB的方式才是最合适的（即使在同一时间点整合了数据结构体系和基础设施，本质上它们也属于不同的层）	考虑系统的重要性、开发和变更频率综合判断整合范围

④ 设计系统的概念构成

根据在提取功能性需求时整理出的功能性需求一览（表 8.8），确定系统范围和要实现的功能，以及功能所对应的基础设施的名称，设计系统的概念构成（图 8.5、表 8.11）。

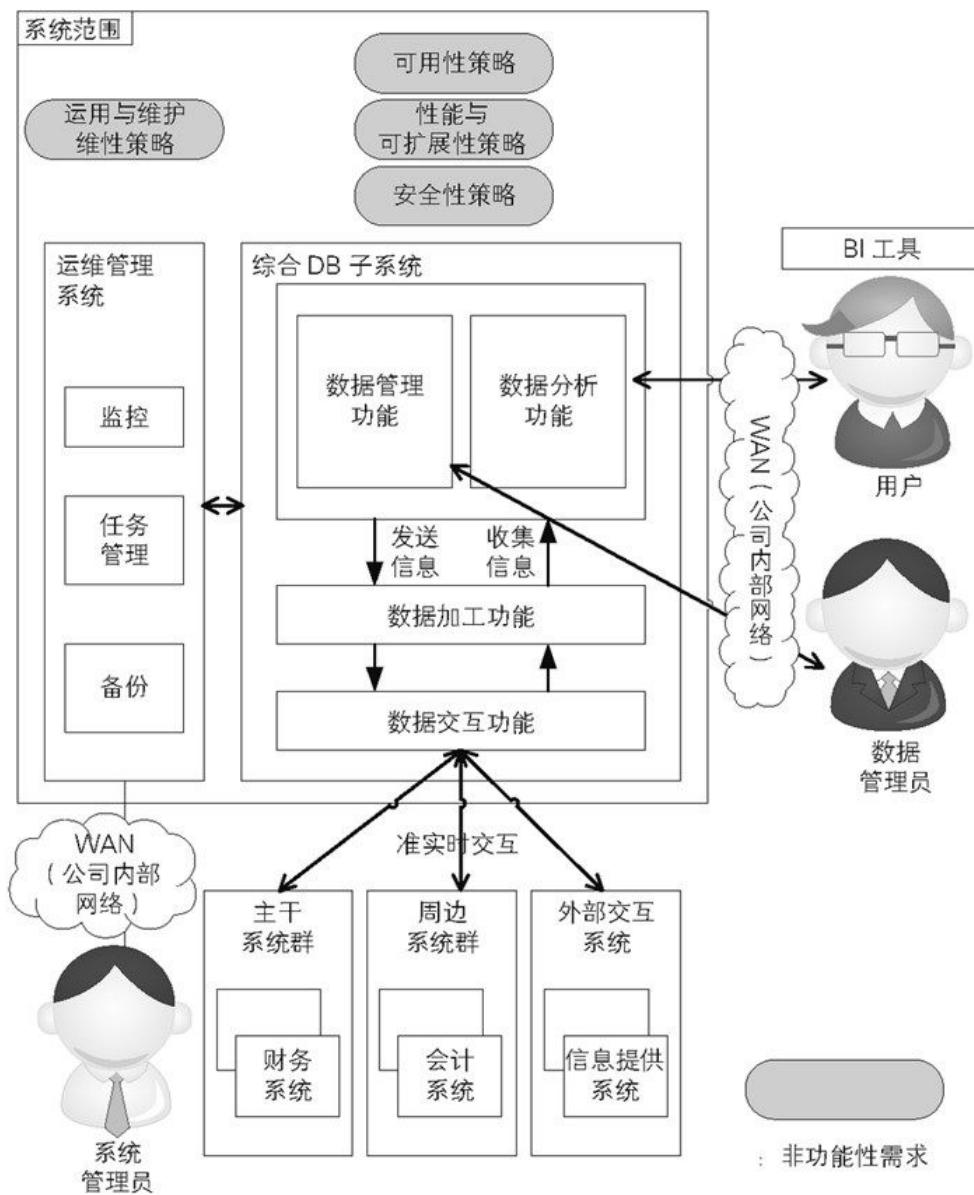


图 8.5 系统概念构成图

表 8.11 要实现的概念功能

实现的功能	功能内容	对应的功能需求	功能对应的基础设施名称
综合DB系统			
数据交互功能	与主干系统、周边系统、外部交互系统进行数据交互	功能需求① 功能需求③	ETL交互服务器 综合DB服务器
数据加工功能	整合各种数据，为了使其他系统更容易使用数据，对数据进行提取、汇集、格式变换	功能需求②	ETL交互服务器 综合DB服务器
数据管理功能	可以在浏览器上查询和更新主数据	功能需求④ 功能需求⑥ 功能需求⑦	认证服务器 Web/AP服务器 综合DB服务器
数据分析功能	使用BI工具对数据进行各种分析	功能需求⑤ 功能需求⑥ 功能需求⑦	认证服务器 Web/AP服务器 综合DB服务器
运维管理系统			
备份功能	进行数据备份以及将数据保存到其他媒介并对其进行管理	—	备份服务器 磁带库
监控功能	对系统进行可用监控、资源监控等		监控服务器
任务管理功能	管理系统上运行的任务		任务管理服务器

⑤ 设计系统的逻辑构成

根据在提取非功能性需求时整理出的非功能性需求一览（表 8.9），以及之前的系统概念构成图（图 8.5）来选择设计方式（表 8.12），并完成系统逻辑构成图（图 8.6）。

表 8.12 选择设计方式

要点	设计方式	
可用性：发生单点故障时业务可继续进行		
1 由于本系统发生故障时影响范围很广，因此为了能确保系统在发生故障时也可以继续提供服务，需要采取服务器冗余和网络设备冗余结构	LAN的可用性设计方式	动态路由
	Web/AP服务器的可用性设计方式	会话非共享负载均衡
	DB服务器的可用性设计方式	N+1集群
性能与可扩展性：预计将来保存数据量和交互系统数量都会增加，必须确保系统的性能与可扩展性		
2 为了能整合系统间的交互，使系统交互具有更高的可扩展性，采用数据中心形式的系统交互方式	基础设施交互结构的设计方式	数据中心
3 大型DB系统中磁盘I/O会成为性能瓶颈，考虑到性能与可扩展性，采用SAN、集线器类型	存储设备结构的设计方式	SAN
可维护性：将数据转移到外部媒介永久保存		
4 大型DB系统种数据备份时间很长，为了缩小业务停止时间，采用快照技术	数据备份的可用性设计方式	SAN快照
可维护性：进行可用监控、错误监控、资源监控、性能监控		
5 为了能够快速检测故障并进行应对，提高服务品质，需要引入运维监控设备和建立运维体制	运用与维护体制的设计方式	服务级别提升
安全性：采取防止信息泄露的策略		
6 为了防止组织内部人员将数据泄露出去，需要加密数	信息泄露策	数据加密

据	略的设计方 式	
其他		
7 向综合DB系统中汇集信息时，不能仅使用ETL进行数据清洗，还要在数据整合基础设施中管理元数据，优化并确保数据结构体系的整体	数据使用和信息分析的设计方式	数据集市+BI构成的数据仓库
网段构成：从整体非功能性需求推演出的网络构成		
8 采用容易进行系统监控和容易扩展性能的网络构成	Web系统的网络构成的设计方式	3网段构成

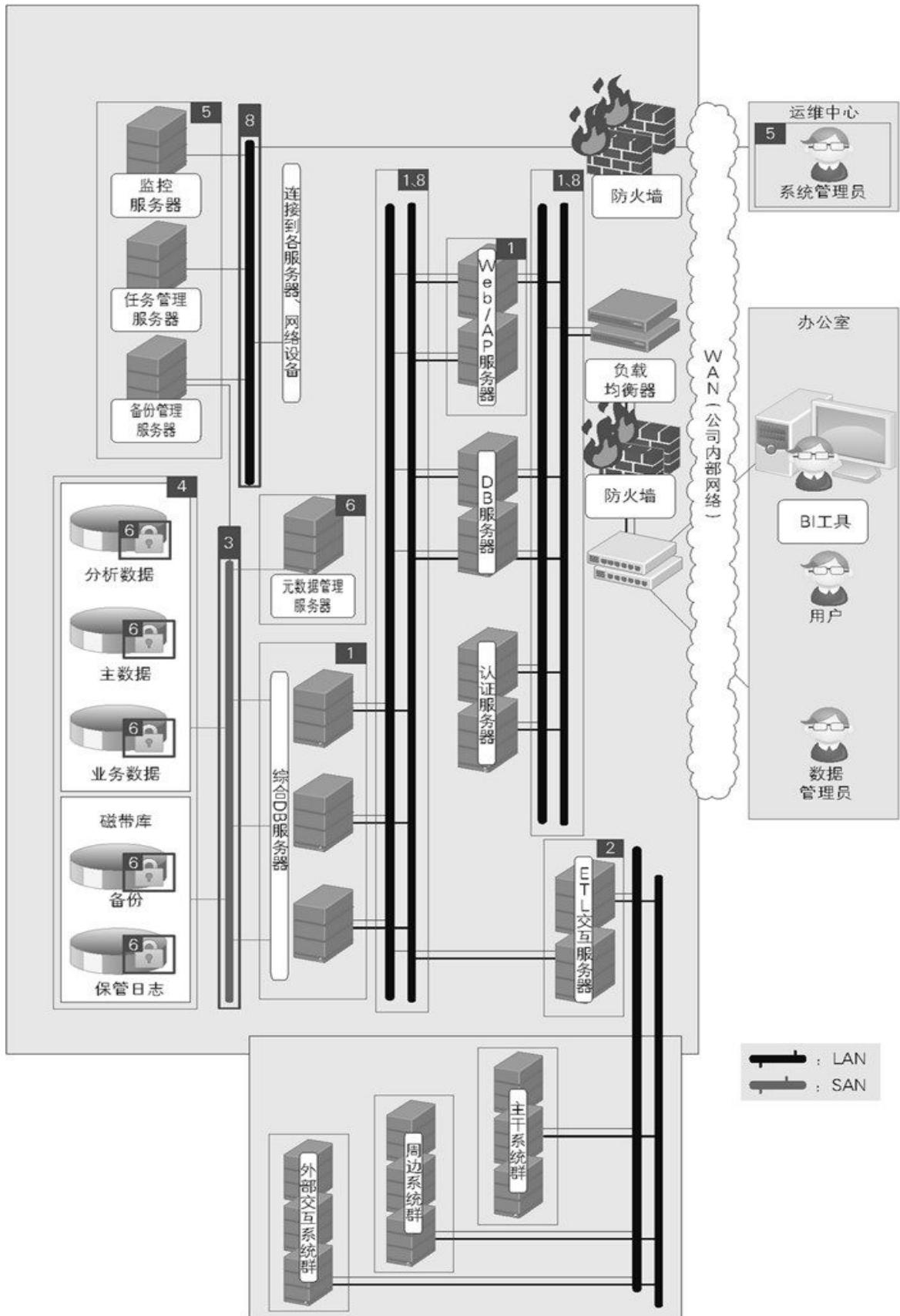


图 8.6 系统逻辑构成图

8.4 总结

本章以地理信息系统以及综合 DB 系统为例讲解了基础设施构成的讨论流程。通过明确需求与设计方式的对应关系，当环境发生变化导致需求和规格发生变更时，也可以迅速地把握影响范围。通过在需求和设计方式中应用非功能性需求等级和基础设施设计模式，可以更加弹性地应对环境变化。

看完了

如果您对本书内容有疑问，可发邮件至contact@turingbook.com，会有编辑或作译者协助答疑。也可访问图灵社区，参与本书讨论。

如果是有关电子书的建议或问题,请联系专用客服邮箱:
ebook@turingbook.com。

在这里可以找到我们：

- 微博 @图灵教育 : 好书、活动每日播报
 - 微博 @图灵社区 : 电子书和好文章的消息
 - 微博 @图灵新知 : 图灵教育的科普小组
 - 微信 图灵访谈 : ituring_interview, 讲述码农精彩人生
 - 微信 图灵教育 : turingbooks

图灵社区会员 张海川 (zhanghaichuan@ptpress.com.cn) 专享 尊重
版权