

云计算架构

解决方案设计手册

Cloud Computing Architected Solution Design Handbook

(美) John Rhoton 著
Risto Haukioja
赵龙刚 金振林 等译

宏观上透彻地讲解云计算系统架构与设计的策略、流程和方法

微观上深入阐述云计算的模型、主流平台、安全性、集成技术、数据存储、可伸缩性和可靠性、部署、运行维护，以及营收和推广



机械工业出版社
China Machine Press

云计算技术系列丛书

云计算架构

解决方案设计手册

Cloud Computing Architected
Solution Design Handbook

(美) John Rhoton 著
Risto Haukioja
赵龙刚 金振林 等译



机械工业出版社
China Machine Press

前 言

本书描述了云计算应用的基本组件，介绍了一些可用来创建跨管理域的大规模、分布式应用的架构级选项。

云计算的需求对于软件工程有着深远的影响。在构建云应用时，我们必须保证该应用能够提供灵活且有弹性的服务，而且要针对能够远程使用由各种可靠和不可靠来源所提供的各项功能而进行设计。

在面对这些挑战时，架构师需要在如下几个领域中考虑可伸缩性和多租户的影响：

- 新工具可以使开发人员更有效地开发、测试和运行云服务。
- 基于互联网的交付会促使人们使用浏览器及其他一些精简客户端来作为展示工具。
- 多租户和身份联邦迫使我们使用一些新的身份认证模型。
- 这是一个由众多迥然不同的服务及服务提供商所组成的碎片化的市场，除非将这些服务有效地集成在一起，否则它们无法一起协调地完成工作。
- 与云规模相关的信息量的爆炸式增长要求我们使用一些新的存储机制和数据模型。
- 可用性和伸缩弹性越来越依靠冗余，必须以一种既高效又可靠的方式来对冗余进行编排。
- 面向效用的服务需要使用新的商业模式和盈利策略才能获得收益。
- 平台服务能够给整个软件开发生命周期带来效率，使我们可以更快地完成部署和采纳变更。
- 自动化程度的提高使我们可以采用一种更高效的运作模式，这种运作模式本身需要从以基础设施为导向转变为以服务为导向。

本书所关注的就是上述这些（和其他一些）可能会受到云计算影响的应用软件架构领域。

假设

本书作者的理念基于这样一个大胆的假设：未来的大多数（即使不是全部）应用都应设计为一种云就绪（cloud-ready）的应用。当然，并非所有软件都会从一开始就将自己的目标定位于那些必须具有互联网级的规模和多租户支持的机会。然而，设计一个云就绪解决方案的价值在于：它可以通过不断成长来满足日益增加的需求，并且还能够挖掘到最初可能没有规划的一些潜在收入流。

促使我们设计一个云就绪应用的因素（即使这种需要可能没有立即显现出来）来自以下三方面的考虑：

- 坚实可靠的设计有内在的价值。
- 伸缩弹性是在当今快速变化的商业环境中必不可少的一个条件。
- 灵活的知识产权才最有价值。

坚实可靠的设计有内在的价值。面向服务的架构（SOA）使得服务在其整个生命周期都具有更大的灵活性，从而使得服务可以应对一组处在不断变化中的需求。如果能够从一开始就在软件中置入可靠性、安全性和可伸缩性，那么该软件就会健壮很多，并且在处理突发事件和新的要求时也更容易。

伸缩弹性是在当今快速变化的商业环境中必不可少的一个条件。客户或销售的增长不是你可能需要扩大服务的唯一原因。一个可伸缩的架构可以通过合并和购买的方式来应付需求的瞬时高峰。一个灵活的设计可以使我们能够将应用复用于其他的商业目的和用户群（如合作伙伴和供应商）。

灵活的知识产权才最有价值。有这样一个令人信服的说法：即便在构建一个仅供内部使用的应用时，我们也应该在设计架构时考虑好可伸缩性和多租户的问题。如果一项服务只是重复实现了一些公共的功能，那么你就应该质疑自己：为什么在第一个地方需要这一服务，为什么不能从另一个提供商那里获得相同的服务。反之，如果某项服务具有独特性并且具有自主知识产权，那么你随时都可以选择将服务以授权许可或打包成一

个服务的方式提供给别人，从而创造出一种额外的收入来源。

范围

本书的写作目的既不是对云计算进行高层次的概述，也不是对云计算的商业利益进行详细说明。在市场上有许多讲述这类主题的书籍，比如《Cloud Computing Explained》^①和“参考文献”中所列的其他一些书籍。

作者也没有深入论述任何一种具体平台或服务。因此，寄希望于仅参考本书就能设计出一个基于云计算的解决方案是一个不现实的想法。不过，各服务提供商一般都会为自己的产品提供一份内容丰富的文档，而且本书也提供了另外一些优秀资源的链接，它们可以在技术和技巧两方面提供很多具体的建议。

相反，本书弥补了企业管理层与程序员之间在视角上的差异。本书讲述了各种可能与云计算相关的架构级选项，但本书论述的深度仅足够评估设计是否适当，并不以详细说明实现细节为目标。

读者

正如前面所暗示的那样，本书的主要读者是那些咨询师、架构师、技术专家和策略师，他们可能在大型企业参与 IT 实现的规划工作或者在参与规模须可扩展到数百万用户的、面向消费者的服务的设计工作。

本书的第二类读者是那些仅在外围参与上述设计工作的人员。无论是系统管理员、程序员还是高层管理人员，几乎每个 IT 人员最终都与云计算有着某种联系。每一个希望对云软件架构有透彻理解的人都能从本书关于这些选项和设计权衡的概念性描述中受益。

① 本书中文版《云计算：企业实施手册》已由机械工业出版社引进出版，ISBN：978-7-111-35177-1。

反馈

你可能会从本书中发现一些错误和遗漏之处，也可能会发现一些真的非常有用和令你感兴趣的内容。无论你要反馈什么样的信息，只要您有话要说，我们就很乐意倾听您的意见。请随时与我们联系，邮件请发至：

john.rhoton@gmail.com 或 risto.haukioja@gmail.com

我们不能保证会回复每一封邮件，但我们会竭尽所能，以感谢您的付出！

致谢

本书（英文版）由 Lightning Source (Ingram Content Group) 出版和发行，该公司的生产过程非常高效和敏捷，令我们非常赞赏。此外，我们还得到了 Gill Shaw 的大力帮助，他在校对和编辑方面为我们提供了卓越的帮助。我们还要感谢 Elisabeth Rinaldin，她为本书的封面和版式设计做出了杰出贡献。

我们非常感谢评审者为本书所做出的技术上的宝贵付出，他们是：John Bair、Jamieson Becker、Patrick Joubert、Kevin Laahs、Franz Novak 和 Vikram S。我们要向本书最后所列出的诸多资源喝彩，当我们要深入了解本书所介绍的那些话题的细节时，它们会给我们带来极大的帮助。最后但并非最不重要的是，我们想指出：如果没有那些在 Amazon、Google、Microsoft、Salesforce.com 以及其他云服务提供商处工作的富有远见和创造力的工程师，也不会有本书所要介绍的这些内容。

目 录

译者序 前言

第一部分 模型

第 1 章 云计算定义	2
1.1 云属性	2
1.2 云服务	4
1.3 云交付模型	7
1.3.1 私有云	8
1.3.2 共同的本质	8
1.3.3 云统一体	9
1.3.4 伙伴云	9
1.3.5 社区云	10
1.3.6 混合或多源的交付	11
1.4 云成熟度	12
1.5 对于应用开发的影响	14
第 2 章 参考架构	16
2.1 云组件模型	16
2.2 平台	17
2.3 展示	18
2.4 身份认证	18
2.5 集成	19
2.6 信息	19
2.7 弹性	20
2.8 盈利	20
2.9 部署	21

2.10 运行维护	22
-----------	----

第 3 章 使用场景

3.1 采用 IaaS 存储的托管服务器	24
3.2 专用 IaaS	24
3.3 采用 IaaS 扩展的私有应用	25
3.4 纯 PaaS	26
3.5 采用 IaaS 存储和计算的 PaaS 应用	27
3.6 实用注意事项	29

第二部分 平台

第 4 章 平台定义

4.1 组件	32
4.2 托管	35
4.2.1 内部部署	35
4.2.2 IaaS	36
4.2.3 Web 托管	36
4.2.4 PaaS	37
4.3 选择标准	39
4.3.1 编程语言	39
4.3.2 开发环境	40
4.3.3 开发人员的偏好	40
4.3.4 其他因素	41
4.3.5 混合式组合	41

第 5 章 Google App Engine

5.1 编程语言	44
----------	----

5.2	数据存储	45	7.7	Elastic Beanstalk	68
5.3	沙箱	45	7.8	实用注意事项	69
5.4	开发环境	48	第 8 章 其他可选平台	70	
5.5	实践注意事项	49	8.1	私有云方案	71
第 6 章 Microsoft Windows Azure	51		8.1.1	VMware	71
6.1	开发环境	51	8.1.2	Joyent	72
6.2	Azure 平台	53	8.1.3	Nirvanix	73
6.3	Azure 服务	55	8.1.4	Eucalyptus	73
6.4	实用注意事项	57	8.1.5	Flexiscale	73
第 7 章 Amazon Web Services	59		8.2	伙伴云方案	74
7.1	计算	60	8.3	Rackspace Cloud	74
7.2	存储	61	8.4	GoGrid	75
7.2.1	简单存储服务	61	8.5	Engine Yard	76
7.2.2	弹性块存储	61	8.6	Salesforce.com	78
7.2.3	简单 DB	61	8.6.1	外部编程访问	78
7.2.4	关系数据库服务	61	8.6.2	Apex	79
7.2.5	CloudFront	62	8.6.3	用户界面	81
7.3	集成	62	8.7	Facebook	81
7.3.1	弹性 IP 地址	62	8.7.1	展示	81
7.3.2	简单队列服务	63	8.7.2	应用逻辑	82
7.3.3	简单通知服务	63	8.7.3	数据	82
7.3.4	虚拟私有云	63	8.8	Intuit	83
7.3.5	VM 导入	63	8.9	基于基础设施的平台	84
7.3.6	AWS 导入 / 导出	64	8.10	实用注意事项	84
7.4	可伸缩性	64	第三部分 展示		
7.4.1	高性能计算	64	第 9 章 浏览器界面	87	
7.4.2	弹性负载均衡	65	9.1	浏览器支持	88
7.4.3	自动伸缩	65	9.2	浏览器接口方式的类型	90
7.4.4	弹性 MapReduce	65	9.2.1	静态网站	90
7.5	管理	66	9.2.2	富服务器端接口方式	91
7.5.1	CloudFormation	66	9.2.3	富客户端	92
7.5.2	CloudWatch	66	9.2.4	富客户端 / 服务器接口 方式	94
7.5.3	AWS 生态系统	67	9.3	实践建议	94
7.6	计费	67			
7.6.1	灵活支付服务	67			
7.6.2	DevPay	67			

第 10 章 专用客户端.....95

- 10.1 桌面应用..... 95
- 10.2 移动应用..... 97
- 10.3 常见的数据传输..... 104
- 10.4 实践建议..... 105

第 11 章 展示优化.....106

- 11.1 测量展示层..... 107
 - 11.1.1 Google Analytics..... 107
 - 11.1.2 Pingdom..... 108
 - 11.1.3 Chartbeat.....110
 - 11.1.4 A/B 测试.....112
 - 11.1.5 服务器 / 客户机的性能测量.....112
- 11.2 展示优化技术.....114
 - 11.2.1 减少网络请求次数.....114
 - 11.2.2 最小化下载内容.....114
 - 11.2.3 分发和缓存.....114
 - 11.2.4 优化网页元素顺序.....115
 - 11.2.5 调整应用逻辑.....115
- 11.3 实践建议.....116

第 12 章 实时网络.....117

- 12.1 提供实时信息.....118
 - 12.1.1 Ajax/REST.....118
 - 12.1.2 Comet.....119
 - 12.1.3 WebSocket..... 121
 - 12.1.4 Google Channel API... 122
- 12.2 聚合实时信息..... 122
 - 12.2.1 订阅源..... 122
 - 12.2.2 社交网络..... 124
- 12.3 对外公布信息..... 124
- 12.4 实践建议..... 125

第四部分 身份认证**第 13 章 身份认证.....128**

- 13.1 背景..... 128

- 13.1.1 目录同步..... 128

- 13.1.2 实时查找..... 129

- 13.2 联邦式 Web 身份识别解决方案..... 130

- 13.3 身份架构..... 132

- 13.3.1 消费者应用..... 132

- 13.3.2 企业应用..... 133

- 13.3.3 身份认证实践..... 134

- 13.4 OpenID..... 135

- 13.5 OAuth..... 136

- 13.6 SAML..... 139

- 13.7 微软..... 141

- 13.8 身份服务提供商..... 143

- 13.8.1 例子: myOneLogin..... 144

- 13.8.2 开通..... 145

- 13.9 访问控制..... 145

- 13.10 应用的公布..... 146

- 13.11 实践建议..... 147

第 14 章 个性化.....148

- 14.1 易访问性..... 148

- 14.2 用户体验..... 149

- 14.2.1 效率..... 149

- 14.2.2 清晰..... 151

- 14.2.3 美观.....152

- 14.3 定制内容..... 152

- 14.3.1 位置..... 153

- 14.3.2 社交圈..... 154

- 14.3.3 其他服务..... 155

- 14.4 实践建议..... 155

第五部分 集成**第 15 章 网络集成.....158**

- 15.1 网络的连接方式..... 158

- 15.1.1 所需的连接..... 158

- 15.1.2 连接需求..... 160

- 15.1.3 连接选项..... 161

15.1.4 连接方式配置	163	18.4 NoSQL	197
15.1.5 虚拟私有云	164	18.4.1 横向可伸缩性	197
15.2 内容交付网络	166	18.4.2 NoSQL 的数据结构	198
15.3 实践建议	167	18.5 实践建议	201
第 16 章 应用集成	168	第 19 章 非关系型数据	202
16.1 遗留应用和外部应用	168	19.1 文档数据库	202
16.2 内部集成	171	19.2 图数据库	204
16.3 数据层	172	19.3 搜索	205
16.3.1 数据订阅源	173	19.3.1 商品化企业级搜索	206
16.3.2 信息连接	176	19.3.2 Lucene	206
16.4 应用集成服务	177	19.3.3 Solr	206
16.5 实践建议	178	19.3.4 Katta	207
第六部分 信息		19.3.5 Bobo Browse	207
第 17 章 存储	180	19.4 语义数据的存储	208
17.1 存储需求	181	19.5 分析	209
17.2 架构级选择	181	19.5.1 云基础设施方面 的开源	209
17.3 本地实现选项	183	19.5.2 Hadoop 的分析: Hive、 Pig、Cascading	210
17.3.1 内存	183	19.5.3 按需商务智能	211
17.3.2 文件系统	185	19.6 外部数据需求	211
17.4 外部接口	186	19.6.1 实时 Web 内容	212
17.5 数据的生命周期管理	188	19.6.2 网络搜索	212
17.6 实践建议	189	19.7 实践建议	213
第 18 章 关系型数据	190	第七部分 弹性	
18.1 可伸缩性与完整性	190	第 20 章 可靠性	216
18.2 SQL	192	20.1 可用性	216
18.2.1 SQL 未死	192	20.1.1 组件的可靠性	217
18.2.2 自管理 SQL	193	20.1.2 容错	218
18.2.3 SQL 即服务	194	20.1.3 降低复杂度 / 松耦合	219
18.3 PseudoSQL (近 SQL)	195	20.1.4 快速恢复	220
18.3.1 Amazon SimpleDB	196	20.2 可使用性	221
18.3.2 Google App Engine 数据存储	196	20.3 灾难	222
18.3.3 Microsoft Azure 存储	196	20.4 可恢复性 / 可逆性 / 回滚	224

20.4.1	备份	224
20.4.2	版本化	225
20.4.3	归档	226
20.5	实践建议	226
第 21 章	伸缩弹性	227
21.1	纵向伸缩	227
21.2	分片	229
21.3	高性能计算	231
21.4	MapReduce	232
21.4.1	它是什么	232
21.4.2	它有什么好处	233
21.4.3	为什么使用它	234
21.4.4	它能做什么	235
21.4.5	它的工作原理 是什么	237
21.4.6	如何使用它	238
21.5	数据关联	240
21.6	资源分配 / 容量规划	241
21.6.1	私有云的规划	241
21.6.2	云爆发	242
21.7	实践建议	244

第八部分 盈利

第 22 章	市场营销	247
22.1	广告	247
22.1.1	基于 Web 的横幅 广告	248
22.1.2	按单击次数付费	249
22.1.3	搜索引擎	251
22.1.4	超越搜索	252
22.1.5	二线广告	253
22.1.6	实践注意事项	253
22.2	搜索引擎优化	255
22.2.1	黑帽 SEO	256
22.2.2	白帽 SEO	256

22.3	社交媒体营销	258
22.3.1	研究	258
22.3.2	广告	261
22.3.3	监控	262
22.4	实践建议	263
第 23 章	付款	264
23.1	估价模型	264
23.2	服务分级	265
23.3	直接收费	265
23.3.1	定价	266
23.3.2	计量	267
23.4	结算	268
23.4.1	直接信用卡计费	268
23.4.2	PayPal	269
23.4.3	Amazon Flexible Payments Service	270
23.4.4	Google Checkout	270
23.5	广告	270
23.6	专业服务	272
23.7	应用市场	273
23.8	实践建议	278

第九部分 部署

第 24 章	开发阶段	281
24.1	编码	282
24.2	测试	285
24.2.1	Azure	286
24.2.2	App Engine	286
24.2.3	Amazon Web Services	287
24.3	实践建议	288
第 25 章	测试阶段	289
25.1	持续集成	289
25.2	云集成测试	291
25.3	实践建议	294

第 26 章 部署测试阶段.....295

- 26.1 测试环境..... 295
 - 26.1.1 用户验收测试..... 296
 - 26.1.2 运行验收测试..... 296
- 26.2 部署测试等级..... 297
- 26.3 云部署测试..... 297
- 26.4 本地数据..... 297
- 26.5 负载测试工具..... 298
 - 26.5.1 SOASTA..... 299
 - 26.5.2 LoadStorm..... 300
 - 26.5.3 HP LoadRunner..... 300
 - 26.5.4 IBM Rational Performance Tester..... 300
- 26.6 实践建议..... 300

第 27 章 生产阶段.....302

- 27.1 发布..... 302
- 27.2 数据迁移..... 303
- 27.2 数据迁移..... 303
- 27.3 发布周期..... 303
- 27.4 回滚..... 304
- 27.5 实践建议..... 304

第十部分 运行维护**第 28 章 配置.....308**

- 28.1 CMDB..... 309
- 28.2 DevOps..... 310
 - 28.2.1 Puppet.....311
 - 28.2.2 Chef..... 312
 - 28.2.3 Capistrano..... 313
 - 28.2.4 Fabric..... 313
 - 28.2.5 选择标准..... 313
- 28.3 开通自动化..... 314

28.4 实践建议..... 315**第 29 章 管理.....316**

- 29.1 管理门户..... 316
- 29.2 IT 运营管理..... 317
- 29.3 服务请求处理..... 318
- 29.4 变更管理..... 319
- 29.5 新发布管理..... 320
- 29.6 监测..... 320
- 29.7 容量管理..... 321
- 29.8 可用性管理..... 322
- 29.9 访问管理..... 323
- 29.10 实践建议..... 324

第 30 章 故障排查.....325

- 30.1 事故管理..... 325
- 30.2 问题管理..... 326
- 30.3 事件管理..... 326
- 30.4 技术支持..... 327
 - 30.4.1 文档..... 328
 - 30.4.2 门户网站..... 329
 - 30.4.3 服务台..... 329
 - 30.4.4 支持的分级..... 330
 - 30.4.5 服务知识管理系统..... 332
- 30.5 实践建议..... 333

第 31 章 精益求精.....334

- 31.1 改进过程..... 334
- 31.2 商业环境..... 335
- 31.3 技术演进..... 335
- 31.4 持续服务改进..... 336

附录 A 案例研究：TrendJammer.....337**参考文献.....347**

第一部分 模 型

第一部分将讲述我们对云计算的看法以及它对软件开发人员和架构师的含义。首先，我们将从对一些与基于云的交付和云服务架构相关的主要理论概念进行定义和概述开始。

其次，我们会定义并阐明一个高层次的参考架构，并以此作为本书其余部分的结构基础。

第 1 章 云计算定义

最近几年，关于云计算的各种声音有很多。时至今日，你可能已经很清楚“云计算”一词的含义了，甚至对于它的工作原理你也很清楚了。但是，如果你要负责设计系统或应用，那么仅仅对这一 IT 新趋势进行理论上分析还远远不够。

相反，你需要了解该如何利用这些新的选项来构建一个更大、更好且更便宜的解决方案。本书将探讨的内容包括如何利用这些新工具、如何实现伸缩弹性、如何集成不同的云服务、如何为海量数据建模、如何使运营更有效率，以及如何充分发挥新商业模式的作用。

本书有意识地没有花费过多的篇幅来对云计算进行基础性的概述。如果你对如何从面向业务的视角来看待在企业中实施云计算的优势和挑战感兴趣，那么你可以从阅读《Cloud Computing Explained》一书或本书“参考文献”所列的其他一些介绍性的书籍开始。本书假定你已经很好地掌握了那些基础知识，并已准备好进入一个更高阶的水平了。

尽管如此，我们还是希望能够确保大家在同一个概念基础上探讨云计算。你对云计算的认识可能完全正确，但同时又可能与我们的认识有着显著的差异。为了尽量避免混淆、确保使用同一套框架和术语，我们仍需简单地刻画一下云计算对于我们来说意味着什么，并且描述构成云计算基础设施及解决方案的那些主要服务和交付模型。

1.1 云属性

简单地说，云代表一种网络，更具体地说，就是全球的互联网络。云计算，言下之意就是：使用远程托管并通过互联网进行交付的计算资源。这就是该术语所蕴含的基本理念。对于你的那些不搞技术的朋友和同事来说，这样理解可能就足够了，但对于本书的读者来说，这还远远不够。

如果你曾试图通过寻找一个权威定义来分离出“云计算”的核心意义，那么你很快就会发现这一术语蕴含着许多不同的概念。对于云计算这一技术上的根本性转变有哪些构成要素，专家们也有不同的意见。有些专家能够将自己的观点表达得比别人的观点更优雅，但这并不意味着他们的观点得到了大家普遍接受。

目前人们普遍接受的定义是由 NIST（美国国家标准技术研究所）所阐述的（2009）：

云计算是一种模型，人们可以使用它来方便地按需通过网络访问一个可配置的计算资源（如网络、服务器、存储、应用和服务等）的共享池，只需最小化的管理工作量或服务提供商干预就可以快速地开通和释放资源。

然而，无论是 NIST 的这一公式化定义，还是上述关于云是何意的解释，都没有得到所有人的认可。要理解这一术语的各种常见解释，最有实效的方法是研究一下各种典型云解决方案的属性分类。这并不意味着云计算的每一项云属性都是必不可少的，也不意味着只要将这些属性任意组合在一起，就能形成一个符合云计算思想的方案。对于云计算的概念来说，这些属性本身既不是必要条件，也不是充分条件。然而，这些属性在某个实现上应用得越多，其他人就越有可能将该实现认可为一个云解决方案。

我们非正式地调查了关于此话题的博客、微博以及公开发表的一些文献，从中发现了如下一些关键部件：

外部部署（off-premise）：服务被托管在一个所有权归服务提供商所有的地点，并从那里提供给用户使用。这通常有两层含义：一是该服务要通过公共互联网来提供；二是其处理过程要在公司防火墙之外进行。换句话说，该服务必须同时跨越物理和安全这两道边界。

伸缩弹性（elasticity）：云计算的主要好处之一是服务提供商所固有的、可以提供给最终用户的可伸缩能力。云计算模型在提供弹性开通机制方面进行得越深入，在需要资源时，就可以越快地扩展或者收缩资源。因为资源通常都是按照效用计费的，所以这种伸缩弹性可以等同于节约了直接成本。

灵活的计费方式：将对资源使用情况进行细粒度的计量与按需开通服务相结合，可以很容易地得出多种客户计费的可选方案。可以订阅为基础收取费用，也可根据实际使用（或预留）资源的数量来收取费用。除了制定详细的合同和采用集中计费之外，还可以通过放置广告的方式或依赖简单的信用卡收费来获利。

虚拟化：云服务通常要通过一个抽象化的基础设施来提供。它们会充分利用各种虚拟化机制，通过多租户来实现成本的优化。

服务交付：云功能往往要以某种服务的形式来供用户使用。虽然这些服务

在本质上有很大差异，但它们通常除了具有用户界面之外还提供一些可编程访问的接口。

通用访问：资源民主化（resource democratization）意味着将资源放入一个资源池中来供那些经过授权的用户访问。与此同时，地点无关性和高水平的弹性可以使用户获得一种始终连接的用户体验。

管理简化：自动配置可以满足可伸缩性的要求，用户自助服务可以加快业务流程，通过编程方式可访问的资源便于实现与企业管理框架集成。这些都简化了系统的管理工作。

资源价格合理：资源成本大大降低，原因有二。一是不需要固定采购的资本支出。二是服务提供商的规模经济效应使他们能够利用商品化的硬件和大多数公司所难以匹敌的优化运作流程来优化成本结构。

多租户（multi-tenancy）：云要被多家组织（即租户）所使用，并且也包含一系列的机制来保护每个租户并使其与其他租户隔离。让客户通过一个共享池来访问资源是实现可伸缩性和节约成本的重要因素。

服务水平管理：云服务通常会提供对于服务水平的定义，该定义设定了客户对于该服务的健壮度的预期。有些服务可能只带有一些最低限度的承诺（甚至也可能根本没有承诺）。虽然我们仍可以把它们视作云服务，但与其他那些使用了更加精确的承诺进行监管的服务相比，我们通常不会“信任”这样的服务，不会将它们用于那些关键任务的应用。

1.2 云服务

云计算的一个很突出的倾向就是它对于面向服务的强烈关注。我们经常听到“某某（或一切）即服务”（XaaS）这样一类说法。换句话说，云不是某种单一产品，它通常是一些异构服务的一种零碎合并。

你不能仅仅提供一些可整体安装在台式机和服务器的打包解决方案，或者投资建设一些单一用途的应用，你需要做的是：将用户需要的所有功能分解成为一些可根据需要组装在一起的原语（primitive）。

遗憾的是，除非你对所有可用的服务都了解得非常清楚，否则很难找到一种可以将这些功能聚合在一起的最佳方式。如果你能使用一些结构和一个模型来阐明各种服务之间的相互关系，那这件事就容易做多了。

最常见的分类方法使用的是一种名为 SPI 的模型（NIST，2009），它把服务分成 SaaS、PaaS 和 IaaS 三类，即软件即服务（Software as a Service）、平台即服务（Platform as a Service）和基础设施即服务（Infrastructure as a Service）。Amazon Elastic Compute Cloud（EC2，亚马逊弹性计算云）是 IaaS 的一个经典范例。通常认为 Google App Engine（Google 应用引擎）是一种 PaaS。而 Salesforce.com 则是我们最熟知的 SaaS 的一个例子。

这三种方法向使用者提供了不同程度的共享。基础设施服务共享的是物理硬件。平台服务则还允许租户共享同一个操作系统和应用框架。而软件服务则一般会共享整个软件栈。如图 1-1 所示，这三种方法代表了对优化和灵活性进行平衡时所做出的不同取舍。在优化方面，需要充分利用多租户和大规模的可伸缩性；而在灵活性方面，则需要能够融合各种不同的约束和自定义的功能。

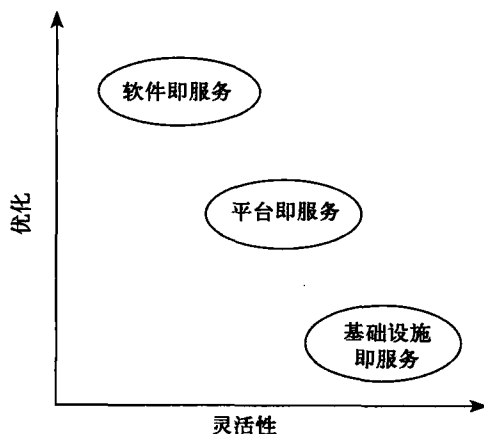


图 1-1 软件、平台和基础设施服务

SPI 模型是一种简单的分类法，它能帮助我们对那些主要的云相关服务建立起一个初步的印象。然而，与其他分类系统经常发生的情况一样，它们实际上的界线并非像图中所示的那样明确。很难将服务准确地归为哪种服务类型。随着时间推移，服务还会在不同服务类型之间漂移。例如 Amazon 公司，该公司一直在增强其 AWS（Amazon