系统架构设计

程序员向架构师转型之路

SYSTEM ARCHITECTURE DESIGN-THE WAY FROM PROGRAMMER TO ARCHITECT

◎ 郑天民 著

专注转型，开发人员向**系统架构师**转型**必备之作**

采用“**思路→方法论→工程实践**”三段式**转型方法**

涵盖**架构设计技术**领域、**系统工程**领域和**软能力**领域**各项技能**的转型模式

中国工信出版集团 人民邮电出版社 POSTS & TELECOM PRESS

系统架构设计

程序员向架构师转型之路

SYSTEM ARCHITECTURE DESIGN-THE WAY FROM PROGRAMMER TO ARCHITECT

◎ 郑天民 著

人民邮电出版社

北京

图书在版编目（CIP）数据

系统架构设计 ：程序员向架构师转型之路 / 郑天民著. -- 北京 ：人民邮电出版社，2017.6
ISBN 978-7-115-45054-8

Ⅰ. ①系… Ⅱ. ①郑… Ⅲ. ①计算机系统 Ⅳ. ①TP30

中国版本图书馆CIP数据核字(2017)第092419号

内 容 提 要

本书主要包含软件开发普通程序员向系统架构师转型的一些思路、方法和工程实践，也包括转型过程中意识形态的转变、技术体系的掌握、系统工程学的拓展及各项软技能的提升等内容。本书深入剖析成为一名合格的架构师所需要的各项软、硬技能，重点对目前业界主流的架构师所需掌握的技术知识领域，以及作为一名技术管理人员所需具备的技术管理能力进行详细介绍，并结合一些典型的场景进行案例分析，从而帮助读者了解并掌握成为架构师所需的各种知识体系和实践技巧。

本书面向立志于转型成为架构师的后端服务开发人员。读者不需要有很深的技术水平，也不限于特定的开发语言，但熟悉 Java EE 常见技术并掌握一定系统设计基本概念将有助于更好地理解书中的内容。同时，本书也可以供具备不同技术体系的架构师同行参考，希望能给日常研发和管理工作带来启发和帮助。

◆ 著　　郑天民
责任编辑　吴　婷
责任印制　杨林杰

◆ 人民邮电出版社出版发行　　北京市丰台区成寿寺路 11 号
邮编　100164　　电子邮件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
大厂聚鑫印刷有限责任公司印刷

◆ 开本：787×1092　1/16
印张：16　　2017 年 6 月第 1 版
字数：418 千字　　2017 年 6 月河北第 1 次印刷

定价：49.80 元

读者服务热线：(010)81055256　印装质量热线：(010)81055316
反盗版热线：(010)81055315
广告经营许可证：京东工商广登字 20170147 号

前言

软件行业技术开发从业人员众多，但具备若干年开发经验的普通开发人员往往面临个人发展的瓶颈，即如何从普通开发人员转型成高层次的系统架构师和技术管理人员。想成为一名架构师，应当具备全面的知识体系，需要进行系统的学习和实践。很多开发人员有往架构师转型的强烈意愿，但苦于找不到好的方法和路径。本书把“程序员向架构师转型”作为切入点，提供架构师所需的各方面技能和相应的学习方法，包含针对转型的一些思路、方法、工程实践及可能会碰到的问题和解决方法。本书从架构师的定位及如何成为一名架构师的角度出发，除了技术和设计之外，还会介绍各项系统工程方法论和软能力，旨在为广大开发人员提供一套系统的、全面的转型指南。

本书从“向架构师转型”的角度出发，结合作者在传统及互联网行业多年的技术与管理工作经历展开论述，结合方法论和工程实践，具有较强的针对性和适用性。架构师是一种综合性强的工种。本书整体上是“技术”结合“过程”的行文思路，具备一定深度的同时也涉及更广的知识领域和体系，满足读者往架构师转型过程中的各种技能需求。同时，本书在介绍技术及过程管理的内容时，采用“思路→方法论→工程实践”的三段式模型，不光告诉读者可以怎么做，更重要的是提供了对问题的分析及解决思路和方法论，并辅以相应的工程实践和案例分析。对架构师而言，具体的技术和工具并不是重点，解决问题的思路和方法论才是本质。本书会在这些方面提供一定的指导并进行总结。

全书共分为 4 个篇幅，共计 9 章内容，分别从不同的领域对架构师转型所需要的各项技能展开讨论。

1. 程序员向架构师转型概述篇：剖析架构师角色，提供架构师的视图和视角及程序员向架构师成功转型的思路。

2. 系统架构设计知识体系篇：介绍软件架构体系结构、领域驱动设计、分布式系统架构设计、构架实现技术体系等架构师所应具备的主要技术体系内容。

3. 软件架构系统工程篇：介绍软件工程学、敏捷方法与实践、软件交付模型等架构师所应具备的系统方法论和相关工程实践。

4. 架构师软能力篇：包括架构师与外部环境、自身团队和转型所需的意识形态。

本书面向立志于转型成为架构师的后端服务开发人员，读者不需要有很深的技术水平，也不限于特定的开发语言，但熟悉 Java EE 常见技术并掌握一定系统设计基本概念有助于更好地理解书中的内容。通过本书的系统学习，读者将在普通开发人员的基础上向前跨出一大步，在思想、方法论、实践能力和综合素质等各个方面往一名合格的架构师方向发展，为后续的工作和学习铺平道路。

在本书的撰写过程中，感谢我的家人特别是我的妻子章兰婷女士在我占用大量晚上和周末陪家人时间进行写作的情况下，能够给予极大的支持和理解。感谢以往及现在公司的同事们，身处在业界领先的公司和团队中，让我得到很多学习和成长的机会。没有平时大家的帮助，不可能有这本书的诞生。最后，要特别感谢北风网的童金浩和罗思捷老师，提供了北风网（http://www.ibeifeng.com）这样优秀的互联网教育平台完成本书配套视频的录制和发布。

由于时间仓促，作者水平和经验有限，书中难免有欠妥和错误之处，恳请读者批评指正。可关注微信公众号“程序员向架构师转型”或扫描以下二维码与本书作者进行联系。

郑天民

2016 年 12 月于杭州钱江世纪城

目 录

第一篇 程序员向架构师转型概述

第 1 章 程序员向架构师转型 …… 2

1.1 架构设计基本概念 …… 2
1.1.1 架构的基本定义 …… 2
1.1.2 架构演进理论 …… 4
1.1.3 架构设计与系统工程 …… 7
1.2 剖析架构师角色 …… 8
1.2.1 架构师角色 …… 8
1.2.2 当程序员遇到架构师 …… 10
1.3 架构师的视图和视角 …… 11
1.3.1 架构师的视图 …… 12
1.3.2 架构师的视角 …… 16
1.3.3 视图视角与系统工程 …… 18
1.4 程序员如何向架构师成功转型 …… 19
1.4.1 转型成功的三段式模型 …… 19
1.4.2 转型思维导图 …… 20
1.4.3 作为架构师开展工作 …… 21
1.5 本章小结 …… 22

第二篇 软件架构设计知识体系

第 2 章 软件架构体系结构 …… 24

2.1 软件体系结构 …… 24
2.2 架构风格 …… 25
2.2.1 分布式 …… 25
2.2.2 事件驱动 …… 28
2.2.3 系统结构 …… 31
2.2.4 消息总线 …… 32
2.2.5 适配与扩展 …… 33
2.3 架构模式 …… 35
2.3.1 数据访问 …… 35
2.3.2 服务定位 …… 36
2.3.3 异步化 …… 38
2.3.4 资源管理 …… 39
2.3.5 依赖管理 …… 41
2.4 架构模型 …… 44
2.5 本章小结 …… 45

第 3 章 领域驱动设计 …… 46

3.1 面向领域思想 …… 46
3.1.1 架构设计与领域驱动 …… 46
3.1.2 领域驱动设计核心概念 …… 47
3.1.3 案例介绍 …… 47
3.2 面向领域的策略设计 …… 48
3.2.1 通用语言 …… 48
3.2.2 领域与上下文 …… 48
3.2.3 领域驱动的架构风格 …… 51
3.2.4 案例策略设计 …… 54
3.3 面向领域的技术设计 …… 56
3.3.1 实体与值对象 …… 56
3.3.2 领域服务 …… 59
3.3.3 领域事件 …… 60
3.3.4 聚合 …… 62
3.3.5 资源库 …… 64
3.3.6 集成界限上下文 …… 65
3.3.7 应用程序 …… 67
3.3.8 案例技术设计 …… 67
3.4 案例实现 …… 69
3.5 本章小结 …… 70

第 4 章　分布式系统架构设计 ……72
4.1　分布式系统 ……73
4.2　RPC 架构 ……74
4.2.1　网络通信 ……75
4.2.2　序列化 ……76
4.2.3　传输协议 ……77
4.2.4　服务调用 ……78
4.3　分布式服务架构 ……81
4.3.1　负载均衡与集群容错 ……81
4.3.2　服务路由 ……83
4.3.3　服务注册中心 ……84
4.3.4　服务发布与调用 ……88
4.3.5　服务监控与治理 ……90
4.4　分布式服务框架 Dubbo 剖析 ……91
4.4.1　Dubbo 核心功能 ……91
4.4.2　Dubbo 原理分析 ……94
4.5　微服务架构 ……102
4.5.1　微服务实现策略 ……103
4.5.2　微服务实现技术 ……104
4.5.3　微服务实现案例 ……108
4.6　本章小结 ……109

第 5 章　架构实现技术体系 ……110
5.1　缓存与性能优化 ……111
5.1.1　性能概述 ……111
5.1.2　Memcached ……112
5.1.3　Redis ……116
5.1.4　Nginx ……120
5.2　消息传递系统 ……122
5.2.1　消息中间件需求 ……122
5.2.2　JMS ……123
5.2.3　AMQP ……126
5.2.4　Kafka ……129
5.3　企业服务总线 ……130
5.3.1　服务总线解决方案 ……130
5.3.2　集成化端点 ……136
5.4　数据分析处理 ……140
5.4.1　轻量级批处理 ……140
5.4.2　Spring Batch ……142
5.5　安全性 ……147
5.5.1　安全性概述 ……147
5.5.2　安全性实现技术 ……148
5.6　本章小结 ……151

第三篇　软件架构设计系统工程

第 6 章　软件工程学 ……154
6.1　软件工程学概述 ……154
6.2　软件实现 ……155
6.2.1　需求工程 ……155
6.2.2　系统建模与案例分析 ……157
6.2.3　软件实现与架构师 ……165
6.3　项目管理 ……166
6.3.1　项目管理体系 ……167
6.3.2　项目研发过程的透明化管理 ……173
6.3.3　项目管理与架构师 ……178
6.4　过程改进 ……179
6.4.1　软件过程模型 ……179
6.4.2　软件过程改进 ……181
6.4.3　过程改进与架构师 ……182
6.5　本章小结 ……183

第 7 章　敏捷方法与实践 ……184
7.1　敏捷方法论概述 ……184
7.2　极限编程与工程实践 ……186
7.2.1　极限编程方法 ……186
7.2.2　极限编程工程实践 ……186
7.3　Scrum 与过程管理 ……191
7.3.1　Scrum 简介 ……191
7.3.2　Scrum 框架 ……192
7.3.3　如何进行敏捷回顾案例分析 ……194
7.4　敏捷方法论与架构师 ……198
7.4.1　敏捷开发中架构师的角色 ……198
7.4.2　识别和消除研发过程浪费 ……199
7.5　本章小结 ……204

第 8 章　软件交付模型 ……205
8.1　软件交付模型概述 ……205

8.2　配置管理……206
8.2.1　配置管理概述……206
8.2.2　配置管理模式与实践……209
8.2.3　SVN/GIT 基本应用与实践……210
8.2.4　系统版本控制策略案例分析……214
8.3　持续集成……217
8.3.1　持续集成理念……217
8.3.2　Jenkins 应用……219
8.4　交付工作流……219
8.5　本章小结……220

第四篇　架构师软技能

第 9 章　架构师必备软技能……222
9.1　架构师与外部环境……222
9.1.1　政治与协商……223
9.1.2　沟通……224
9.1.3　邮件……227
9.2　架构师与自身团队……231
9.2.1　领导力……231
9.2.2　知识管理……232
9.2.3　人员管理……235
9.2.4　绩效管理……237
9.3　架构师与意识形态……240
9.3.1　思维模式……241
9.3.2　引入变化……241
9.4　本章小结……245
参考文献……246

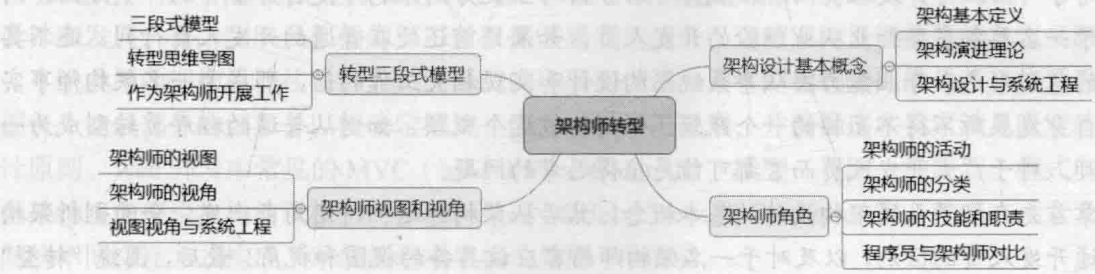
第一篇
程序员向架构师转型概述

本篇内容

本篇从架构设计的基本概念出发，阐述架构设计的理论体系。接着引出架构师角色，从架构师的活动、分类、技能和职责等角度对架构师的角色做了深度剖析，并对普通开发人员和架构师的区别进行了全面比较。成为一名架构师前，需要明确架构师所需掌握的视图和视角。这些视图和视角是架构师手上的武器。最后本章对“程序员如何向架构师成功转型”这个话题进行展开，提出转型成功所需的三段式模型，并提供了转型所需的思维导图。

本篇只有一章，作为开篇总领全书后续章节。

思维导图



第1章 程序员向架构师转型

随着近年来信息化产业的高速发展，一大批由国人自主研发的计算机软件系统，尤其是以电子商务、O2O、移动医疗、在线教育等为代表的互联网和“互联网+”应用已经深刻影响着我们的日常生活模式。面对新的时代潮流，无论对于传统行业还是互联网行业，开发具有功能强大且用户体验好的桌面端和无线移动端应用已经成为众多软件从业人员的目标和要求。然而，分析和设计一个软件系统及管理其研发过程并不是每一个软件行业从业人员都能做的事情，需要具备专业的知识、丰富的实践经验及良好的个人综合能力。我们把具备以上能力的人才称之为软件架构师。

中国目前每年有几十万的软件开发人才缺口，其中具备系统架构设计和实现能力的人才更是紧缺。对于一名软件开发人员而言，成为一名合格乃至优秀的架构师是自身奋斗的一个方向。同时，对于一名具备多年行业从业经验的开发人员，如果目前还处在普通的开发人员行列，还不具备相应的意识形态和专业能力去从事系统架构设计和实现相关工作的话，那成为一名架构师事实上也是自身发展所不得不面临的一个瓶颈。如何打破这个瓶颈，如何从普通的程序员转型成为一名架构师，对于广大开发人员而言都可能是值得思考的问题。

本章首先介绍了系统架构设计的基本概念，然后从架构师这一特定角色出发，全面剖析架构师与普通开发人员的区别，以及对于一名架构师而言应该具备的视图和视角。最后，围绕“转型”问题，提出从程序员到架构师成功转变所应具备的关键因素。

1.1 架构设计基本概念

当下，业务需求层出不穷、技术发展日新月异、团队规模快速扩张，因此，软件系统复杂度及系统共性和特殊性问题在很大程度上决定了软件开发的成败。而软件架构设计（Software Architecture Design）的目的就是对系统进行高度抽象，通过一系列设计原则在最大程度上降低系统复杂度，解决系统中存在的各种共性和特殊性问题。在深入探讨架构设计过程和架构师角色之前，我们先来理解架构的基本含义。

1.1.1 架构的基本定义

我们想要成为一名架构师，有如下两个问题首先需要进行明确。

- 软件架构是什么？
- 软件架构设计是怎么样一种工作内容？

围绕着这两个问题，业界有一些通用的说法，这些说法形成了具有代表性的两大理论体系，分别是架构组成理论和架构决策理论。

1. 架构组成理论

国际标准化组织（International Orgranization for Standardization，ISO）系统和软件工程标准认为，系统的架构是一系列基本概念或者系统在其环境中表现出来的属性，体现在它的元素、关系及设计和发展的原则中。根据 ISO 给出的这一架构定义，系统架构包括系统元素、基本系统属性、设计和发展原则 3 个主要方面。

（1）系统元素

架构的系统元素包括模块、组件、接口、子系统等日常开发中的内容。系统元素之间是有关系的，元素加上它们之间的关系就构成了基本的系统结构。通常，架构师感兴趣的结构类型包括静态结构和动态结构。所谓静态结构体现在设计时，描述系统内部设计时元素及其组合方式；而动态结构则关注运行时的元素及其交互方式。

（2）基本系统属性

架构的基本系统属性一方面包含功能属性，用于说明系统行为属性，回答系统能做什么这一问题，如系统的输入、输出模型。另一方面，系统也关注质量属性，即外部可见非功能性属性，如系统的性能、安全性等属性。

（3）设计和发展原则

架构的设计和发展原则应该能够可度量、可测试和可跟踪，常见的原则如用户操作响应时间在 1s 之内、用户信息需要进行安全性处理、系统可以快速集成第三方服务等。同时，对于架构的设计和发展，每个团队及架构师都可以根据需要制定适合于当前架构发展需要的自定义原则。

组成派的架构设计往往首先关注系统的主要构成部分及它们相互之间的关系，然后进一步挖掘每个构成部分的细节，以便确定模块、组件、接口等元素，并确保这些元素满足既定的架构设计原则。Web 开发中常见的 MVC（Model View Controller，模型—视图—控制器）模式实际上就是架构组成理论在架构风格上的体现，通过 Model、View 和 Controller 等 3 种基本元素及它们之间的不同交互方式构成了系统的基本架构，如图 1-1 所示。

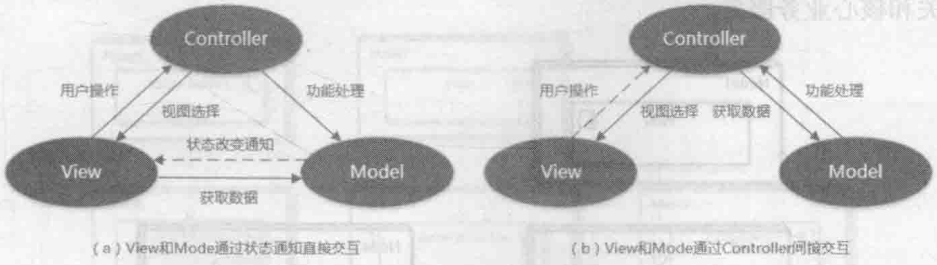


图 1-1　MVC 模式的两种交互方式

2. 架构决策理论

架构决策理论的典型倡导者是 RUP（Rational Unified Process，统一软件过程）。该理论关注架构实践的主体——人，以人的决策为描述对象。架构决策不仅包括软件系统的组织、元素、子系统和架构风格等，还包括众多非功能性需求的决策。当架构设计过程无法顺利进行，如碰到模块如何划分、模块之间交互方式是什么、开发技术如何选型、如何适应可能发生的变化等常见问题时，通过架构师团队根据场景和问题作出相应的决策。不断决策的过程就是问题得到不断解决、架构得到不断发展的过程。通过一系列的决策最终形成完整的系统架构。

决策类的架构设计过程往往可以从系统切分出发，如把系统分成客户端和服务器端两大部分，然后基于服务器端，可以在拆分成服务适配层、业务逻辑层和数据访问层，而业务逻辑层再可以分成多个模块，如图 1-2 所示。每一个切分的过程实际上就是一个决策的过程，通过合理而有效的决策促进系统架构由高层次到较低层次再到实现层次的不断演进。

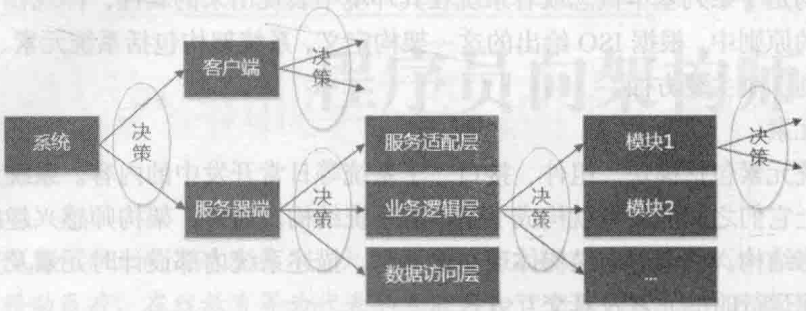


图 1-2　架构决策示例

以上两个派别的理论体系也只是对架构定义问题的一种诠释，业界还有很多相关的说法，很难给出一个标准化的答案。每个架构师基于自身的意识形态和经历也可能会有自己的理解，通常开发人员从日常的开发过程中提炼出对架构设计的理解，被称之为架构演进理论。

1.1.2　架构演进理论

在过去软件开发过程发展的很长一段时间内，软件架构表现为一种集中式的单块（Monolithic）模式，即先对系统进行分层，然后通过单个进程进行部署和维护。典型的分层体系包括界面交互层、业务逻辑层和数据访问层。直至今日，这种单块模式在部分系统构建过程中仍然是最基本的架构模式。

随着业务功能的不断发展及性能、数据存储等系统瓶颈问题的出现，单块模块逐渐不适合系统的维护和扩展。这时，分布式架构应运而生。通过把系统业务进行服务化及完善服务治理功能，系统架构就可以如同搭建积木一样构建成高度可集成、高内聚松耦合的业务系统，图 1-3 所示的系统主体由 Frontend-Service（前端服务）和 Core-Service（核心服务）两层服务化构成，为 Web 层提供网关和核心业务服务。

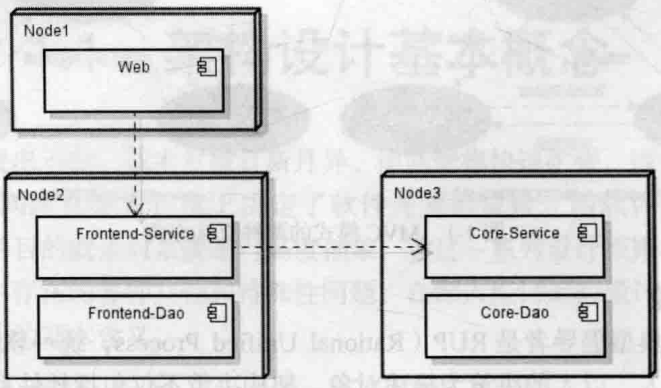


图 1-3　分布式服务模式

服务化架构为系统提供了扩展性和伸缩性，然而随着系统用户体量的增加及分布式系统固有的网络通信机制，性能问题在业务关键链路逐渐成为系统运行的瓶颈。解决性能问题的切入点有很多，一方面可以从硬件设备和软件服务器入手，但对系统架构而言，更多的场合需要我们分析系统实现

方案，并使用以缓存为代表的架构设计手段重构业务关键链路，图 1-4 即为在 Frontend-Service 和 Core-Service 两层服务中分别添加分布式缓存（Distributed Cache）之后所得到的系统部署图。

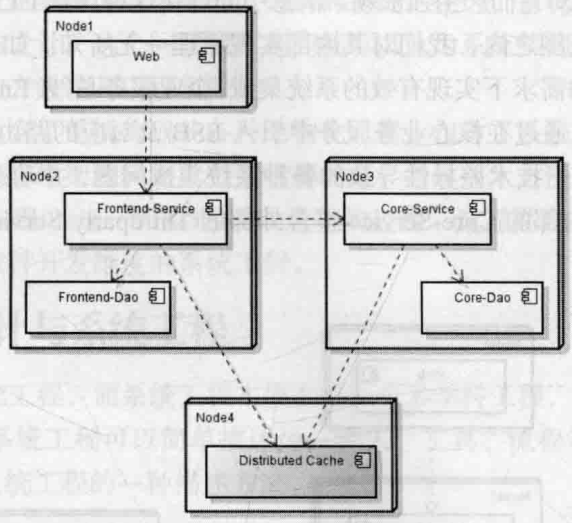


图 1-4　分布式缓存架构

缓存能够提升性能，但不能解耦系统。当系统中分布式服务数量和种类增多，而这些服务又分别属于不同业务层次时，如何合理地管理这些服务之间的调用关系，进一步确保系统的健壮性和扩展性成为系统架构设计的又一大难题。分布式服务的自身特征决定了其在时间、空间和技术上都具有一定程度的系统耦合性。在使用分布式服务时需要谨慎处理服务调用的时序、所使用的服务定义及技术平台的差异性等问题。这些问题为开展快速架构重构和扩展、进行高效分布式团队协作带来了挑战。以各种消息传递组件为代表的中间件系统为降低系统耦合性、屏蔽技术平台差异性带来了新的思路。当不同的服务需要进行交互、但又不需要直接进行服务的定位、调用和管理时，消息中间件（Middleware）能显著降低系统的耦合程度，如图 1-5 所示，在 Frontend-Service 和 Other-Service（其他服务）中添加了消息传递中间件，确保两个服务在并不需要意识到对方存在的前提下进行数据的有效传输。

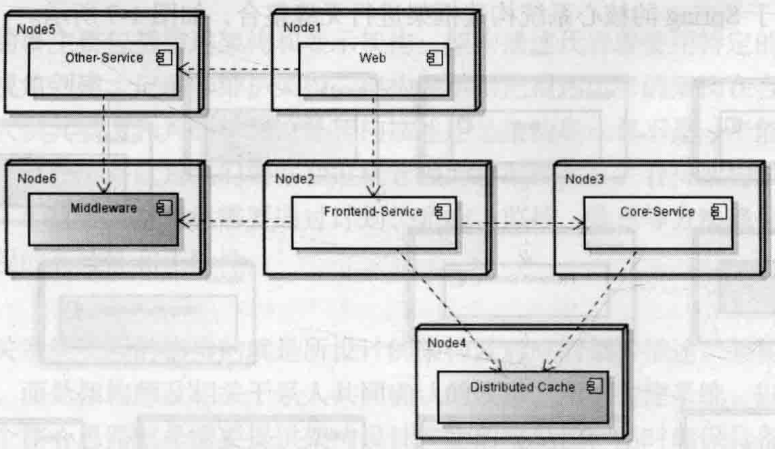


图 1-5　消息中间件架构

试想这样一种场景，我们的系统需要跟外部的多个系统进行集成以形成关键业务链路闭环管理，而这些外部系统分别部署在其他供应商或客户环境，并且每个系统都可能基于完全不同的技

术平台和体系构建，随着业务发展需求，这些外部需求还需要实现动态的注册和注销。对系统架构设计而言，一方面我们需要整合这些外部系统提供的服务进行数据的获取和操作，另一方面，我们又不希望我们的系统对它们产生强依赖。消息中间件在这种场景下已经失去系统解耦的价值，因为外部系统不在控制范围之内，我们对其内部实现原理一无所知。如何在异构系统、分布式服务和基于租户的基本架构需求下实现有效的系统集成，企业服务总线（Enterprise Service Bus，ESB）提供了相应的解决方案。通过在核心业务服务中引入 ESB 及对应的路由、过滤、转换、端点等系统集成模式，即可屏蔽由于技术差异性导致的各种系统集成问题，并动态管理 ESB 上的第三方服务。图 1-6 中的 ESB 为内部的 Core-Service 整合外部的 Thirdparty-Service1 和 Thirdparty-Service2 提供了集成平台。

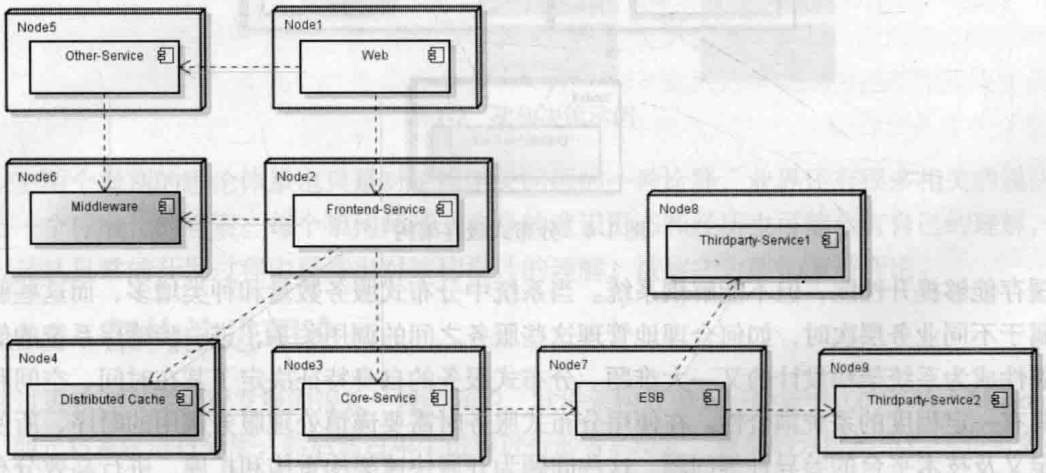


图 1-6　企业服务总线架构

随着大数据时代的到来，许多业务系统也面临着对庞大业务数据进行管理和利用的难题。近年来，以 Hadoop 生态圈为代表的大数据处理平台，以及以 Lucene 为内核的多种垂直化搜索引擎（Search Engine）系统，为业务发展提供了高效的批量数据处理和数据搜索功能。在系统架构设计维度，我们也可以引入如 Spring Batch、Spring Data 等轻量级的批处理（Batch Job）和数据访问框架，以便与基于 Spring 的核心系统构建框架进行无缝整合，如图 1-7 所示。

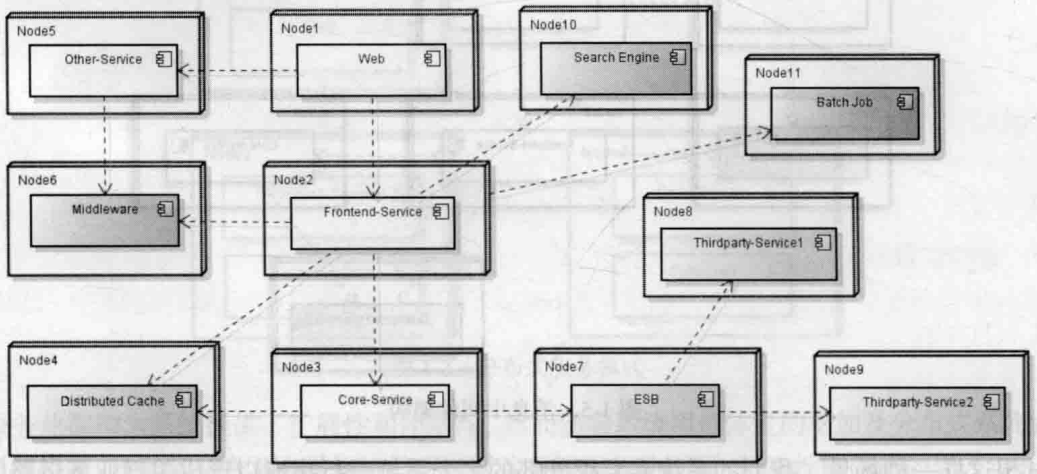


图 1-7　搜索引擎和批量数据处理架构

上述系统架构演进过程在现有的互联网应用中具有一定的代表性，很多 APP（Application，应用程序）后台就是从一个简单的单块模式开始，当面临系统架构设计问题时，通过引入各种技术系统逐步完善架构，直至具备庞大体量的大型集群系统。在这个系统架构演进过程中，我们再来回答“什么是系统架构设计”这个问题时，我们就可以认为系统架构设计是在系统开发演化过程中，解决一系列问题的方法论和工程实践。

方法论和工程实践包含了系统构成的各种结构化元素，包含了系统交互的各种接口和相互协作方式，包含了通用的指导性架构风格。更为重要的是，通过引入方法论和工程实践进行系统架构演进是一个“原型→发现/改进→再发现/再改进”的过程。这显然已经不是一个纯粹的技术问题，而是一项涉及多个软件开发维度的系统工程。

1.1.3　架构设计与系统工程

架构设计是一项系统工程，而系统工程本质上是一个多学科工程，涉及软件开发的技术和管理两方面。一般而言，系统工程可以简单描述为一种人、工具、流程等基本构成元素的组合，图 1-8 即是对架构设计系统工程的一种描述方法。

图 1-8　架构设计系统工程

我们通过上图把架构设计通过系统工程的方法展开，可以清晰地看到以下几点核心思维。

1. 人

架构设计中的人除了架构师本身主要指的是各种干系人（Stakeholder）。所谓干系人，就是架构所需要满足的各种业务方、客户、市场、项目、产品等相关人员。这些干系人代表着功能需求的来源，同时拥有对系统架构是否成功的判断权利。架构设计本质是满足业务需求，业务架构驱动着技术架构，而不是反其道而行。系统工程的第一点核心思路就是我们从事任何架构设计相关的工作都是为了满足干系人的需求。

2. 事

架构设计中的事主要包括描述架构和展示架构。架构描述代表着使用特定的工具、特定的流程和特定的视图视角创建、记录和维护架构。架构展示则把描述出来的架构在合适的时机和场合展示给相关干系人供其参考和判断。无论是架构描述还是架构展示都不是一步能够到位的，所以系统工程的第二个核心思路就是我们要站在过程管理的思维模式上去看待架构设计这件事情：架构设计本身也是一个过程，过程就需要通过计划、实施、监控、管理等方法确保其执行，并通过持续改进实现过程的高效性和正确性。

3. 物

架构设计相关系统工程的物指的就是所设计的架构及对应的架构描述。架构描述并不只是架构师的个人成果，而是架构师及相关干系人共同确认的成果，用于构建系统，以满足业务需求。系统工程的第三个核心思路就是需要提供架构设计产物的可见性，同时确保具备较高设计质量，使其能够接受来自内部和外部的评审和验证。

综上所述，架构设计就是以干系人提出的业务需求为源头、以技术管理和过程改进体系为工作流程、以质量为重心的一项系统工程。在系统工程中，业务需求需要进行分析和抽象、过程需

要进行管理和改进、设计质量需要进行保障，完成包含这些活动在内的整个系统架构设计工作的角色就是架构师。

1.2 剖析架构师角色

架构设计被认为是从问题领域到解决方案的一种桥梁，如图 1-9 所示。从图中我们可以看到架构设计活动与代表问题域的需求分析活动和代表解决域的软件开发活动都有直接交集，连接着两个软件开发的核心领域。

图 1-9 架构设计的桥梁作用

架构师是架构设计的执行者，架构设计的桥梁作用给架构师带来了挑战，意味着架构师需要同时具备处理两个核心领域的能力，即架构师需要能够从问题领域出发推导出满足业务需求的架构体系，同时又能够从实现方法入手设计出能够满足业务架构需求的技术架构体系，最终实现业务架构和技术架构的统一。

1.2.1 架构师角色

1. 架构师的活动与系统工程

架构师是负责设计、记录和领导能够满足所有干系人需求的系统构建过程的人。通常，这个角色需要完成以下 4 项工作。

（1）识别干系人并让他们参与进来

干系人是业务需求的源头，识别正确的干系人能够确保业务需求的正确性，让干系人参与能够确保业务需求的实时性和有效控制需求变更。

（2）理解和记录系统功能和非功能相关的关注点

通过需求分析，架构师梳理并抽象系统的各项功能性和非功能性需求，并对这些需求进行系统建模。

（3）创建并拥有应对这些关注点的架构定义

对功能性和非功能性需求，从扩展性（Extensibility）、性能（Performance）、可用性（Availability）、安全性（Security）、伸缩性（Scalability）等架构设计的基本要素出发定义架构。

（4）在把架构实现为具体系统的过程中起主要作用

推动架构设计活动按照项目和产品计划有序进行，参与需求、设计评审等各种技术评审过程，并管理系统设计和开发团队的日常工作。

就一个完整的系统开发生命周期而言，架构设计活动有其时效性。图 1-10 体现了传统瀑布（Waterfall）模型下的系统开发生命周期与架构师参与情况。从图中可以看出在由需求分析和系统建模所构成的系统初始阶段，以及由服务集成和产品接受所构成的最后交付阶段，架构师会较多地参与到系统建设过程中去。具体参与程度取决于系统本身的特征及生命周期模型。

对于类似 Scrum 的敏捷开发模型，如果把一个个迭代看成是小型的 Water-Scrum-Fall 模型的

话，架构师参与程度实际上也与图 1-10 所示的结果相类似，即重点参与迭代计划阶段和迭代演示回顾阶段。

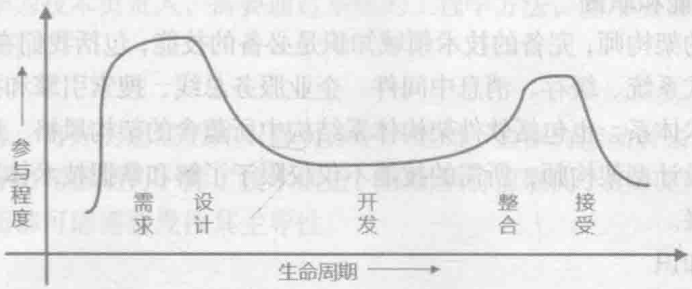


图 1-10　系统开发生命周期与架构师参与情况

从系统工程维度，架构师根据干系人的业务需求捕获系统功能和非功能相关的关注点，并创建架构描述。架构师对架构描述这一架构设计产物具有拥有权，意味着架构师有权力更有责任维护架构描述并管理其实时性和有效性。同时，架构设计表现在系统过程中是一系列架构定义过程。架构过程的正确性决定着架构结果的正确性，架构师需要跟踪并验证架构定义过程的时机、参与者、技术评审和各项阶段性成果。添加了架构师角色及其职责的系统工程如图 1-11 所示。

图 1-11　架构师与系统工程

2. 架构师的分类

基于以上关于架构师的工作内容、参与程度和系统工程的分析，可以看到架构师根据其作用、职责和对系统关注层次的不同，可以分成很多类型。狭义上的架构师往往偏重于技术架构设计。但从广义上讲，业界对架构师的划分有一定的体系，表现在以下 3 个方面。

（1）根据作用

根据所发挥的核心作用，可以把架构师划分成设计型、救火型、布道型、极客型等类型。相较于传统意义上的设计型架构师，救火型、布道型、极客型等类型的架构师更加偏重于执行某一项特定的架构任务，并不一定会完整参与系统开发生命周期，更不一定会从系统工程的角度去看问题。

（2）根据职责

产品型、基础设施型和应用型等架构师是从其所处的业务和职责出发进行分类的结果。产品型架构师偏重于进行业务架构设计，往往在系统开发前期会重点参与；基础设施型架构师偏重于进行技术基础框架设计，一般采用独立于系统开发生命周期的特有开发模式；常见的系统架构师指的是应用型架构师，正如前文所述，负责将问题领域进行建模并转变成解决方案。

（3）根据关注层次

架构师关注的层次有很多，不同的架构师关注的层次会有所不同，包括但不限于功能、非功能、团队组织和管理、产品运营等方面。