西安交通大學

研究生创新教育系列教材

EA架构与系统分析设计

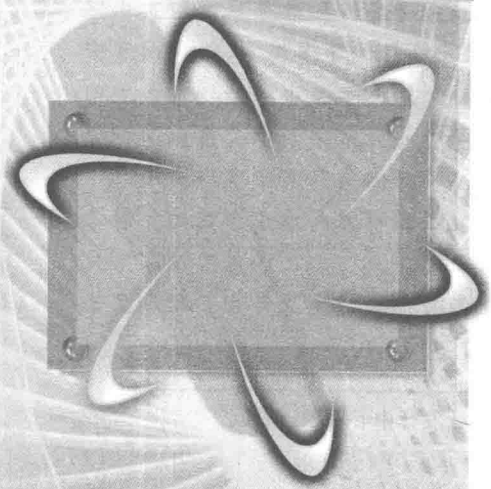
饶 元 吴飞龙 杜小智 赵 亮 编著

西安交通大学出版社
XI'AN JIAOTONG UNIVERSITY PRESS

西安交通大学 研究生创新教育系列教材

EA架构与系统分析设计

饶 元 吴飞龙 杜小智 赵 亮 编著



西安交通大学出版社
XI'AN JIAOTONG UNIVERSITY PRESS

内容简介

随着互联网与移动互联网的兴起，软件系统在不断推陈出新的IT技术推动下，所希望解决和面对的问题域的复杂程度也越来越大，软件系统也正在从低级到高级、从简单到复杂、从封闭孤立到开放协同的方式快速地演化和发展。特别是在目前社会计算、云计算、移动计算以及大数据环境下，社会性软件以及Web2.0领域内的各种复杂的涌现现象及系统内的非线性系统动力学机制，使得对软件系统的分析与设计面临着巨大的全新挑战。面对着复杂而又庞大的应用系统，忽视软件系统设计整体目标的软件开发行为，已无法在复杂环境下，有效地把握外部环境与需求的动态变化对于系统所产生的影响，以及与其他系统之间的交互行为与集成操作模式。本书希望在软件开发从管理工具走向服务平台的演化过程中，通过采用模型驱动的体系结构(MDA)设计理念以及Zachman模型为代表的企业架(EA)软件的系统分析方法，寻找出一个稳定的，且可以在一定的时间内有效的系统设计理论与方法，并对新入行的人员提供有效的帮助和设计指导。

本书可以供软件工程、计算机科学与技术、管理信息系统等相关领域和专业的高年级本科生、研究生进行学习与参考使用。

图书在版编目(CIP)数据

EA架构与系统分析设计/饶元等编著. —西安：西安交通大学出版社，2015.9
ISBN 978-7-5605-7856-9

Ⅰ.①E… Ⅱ.①饶… Ⅲ.①软件工程—系统分析②软件设计 Ⅳ.①TP311.5

中国版本图书馆CIP数据核字(2015)第206121号

书　　名 EA架构与系统分析设计
编　　著 饶　元　吴飞龙　杜小智　赵　亮
责任编辑 任振国　季苏平

出版发行 西安交通大学出版社
(西安市兴庆南路10号　邮政编码710049)
网　　址 http://www.xjtupress.com
电　　话 (029)82668357　82667874(发行中心)
(029)82668315(总编办)
传　　真 (029)82668280
印　　刷 陕西宝石兰印务有限责任公司

开　　本 727mm×960mm　1/16　**印张** 29　**字数** 705千字
版次印次 2015年12月第1版　2015年12月第1次印刷
书　　号 ISBN 978-7-5605-7856-9/TP·695
定　　价 53.00元

读者购书、书店添货，如发现印装质量问题，请与本社发行中心联系、调换。
订购热线：(029)82665248　(029)82665249
投稿热线：(029)82669097　QQ：8377981
读者信箱：lg_book@163.com

《研究生创新教育》总序

创新是一个民族的灵魂，也是高层次人才水平的集中体现。因此，创新能力的培养应贯穿于研究生培养的各个环节，包括课程学习、文献阅读、课题研究等。文献阅读与课题研究无疑是培养研究生创新能力的重要手段，同样，课程学习也是培养研究生创新能力的重要环节。通过课程学习，使研究生在教师指导下，获取知识的同时理解知识创新过程与创新方法，对培养研究生创新能力具有极其重要的意义。

西安交通大学研究生院围绕研究生创新意识与创新能力改革研究生课程体系的同时，开设了一批研究型课程，支持编写了一批研究型课程的教材，目的是为了推动在课程教学环节加强研究生创新意识与创新能力的培养，进一步提高研究生培养质量。

研究型课程是指以激发研究生批判性思维、创新意识为主要目标，由具有高学术水平的教授作为任课教师参与指导，以本学科领域最新研究和前沿知识为内容，以探索式的教学方式为主导，适合于师生互动，使学生有更大的思维空间的课程。研究型教材应使学生在学习过程中可以掌握最新的科学知识，了解最新的前沿动态，激发研究生科学研究的兴趣，掌握基本的科学方法；把教师为中心的教学模式转变为以学生为中心教师为主导的教学模式；把学生被动接受知识转变为在探索研究与自主学习中掌握知识和培养能力。

出版研究型课程系列教材，是一项探索性的工作，也是一项艰苦的工作。虽然已出版的教材凝聚了作者的大量心血，但毕竟是一项在实践中不断完善的工作。我们深信，通过研究型系列教材的出版与完善，必定能够促进研究生创新能力的培养。

西安交通大学研究生院

自　序

随着互联网与移动互联网的兴起，软件系统在不断推陈出新的IT技术推动下，所希望解决和面对的问题域的复杂程度也越来越大，软件系统正在从低级到高级、从简单到复杂、从封闭孤立到开放协同的方向快速地演化和发展。特别是在目前社会计算、云计算、移动计算以及大数据环境下，社会性软件以及Web2.0领域内的各种复杂的涌现现象及系统内的非线性动力学机制，使得对软件系统的分析与设计面临着巨大的全新挑战。面对着复杂而又庞大的应用系统，传统中那种只关注于技术细节，忽视软件系统设计整体目标的软件开发行为，已无法在新的复杂环境下，从整体上把握当外部环境与需求在动态变化时，对系统以及与其他系统进行交互操作和集成的影响与变化。

因此，本书的一个初衷就是希望在软件开发从管理工具走向服务平台的演化过程中，寻找出一个稳定的，且可以在一定时间内有效进行系统设计的理论与方法，来对新入行的一些人员提供帮助和指导，改变他们孤立地看待事物的思维方式，避免造成信息孤岛从而限制了信息与数据资源的有效共享与利用。当前，软件工程领域一直面临着一个巨大的挑战，即系统分析与设计人员一般按照功能性、非功能性的需求来建立需求模型，再根据需求模型获得相应的设计模型，进而实现代码框架的自动化生成。但是在整个软件开发的实践过程中，由于软件开发所涉及的模型具有多层次与多视角的特点，这也造成了软件建模工作的复杂度过高。因此，在软件项目从需求分析到设计的全过程中，如何建立针对不同层次软件模型的统一表达方式，并通过模型的逐步精化从上一层模型可以精确无二义地或者自动化地获得下一层模型，用来指导软件的开发，成为了当前产业与学术界共同关注的焦点。目前，不同领域的研究与应用实践给我们提供了许多有价值的思考方式与分析设计方法，其中最具有代表性的是模型驱动的体系结构(Model Driven Architecture，MDA)所提出的软件建模与实现的理念以及以Zachman模型为代表的企业架构(Enterprise Architecture，EA)系统分析方法。

自对象管理组织(OMG)于2001年7月发布模型驱动体系结构(MDA)以来，人们在软件系统的分析与设计过程中越来越多地考虑到如何利用MDA的基本概

念、思想与工具，将基本的概念进行模型化、抽象化和逐步精化，并在可视化的CASE工具的支撑下，对系统的一部分结构、功能或行为进行形式化规格说明，从而实现软件产品快速且高质量的开发。因此，MDA已使得软件本身从“软件＝数据结构＋算法”大步地演化到“软件＝模型＋模型实现”的一个新阶段。但是，如果希望使模型能够充分地表示软件所涉及的问题域，就必须将相关的数据结构、系统功能以及系统行为等特征进行统一的、形式化的和规格化的严格定义。这一方面要保证所有的定义是唯一且无歧义的，即将模型描述语言与程序语法和语义之间进行了严格地绑定，并实现了模型语言与程序语言之间的映射；另一方面，利用模型语言所具有的可视化与抽象化的能力，帮助人们在软件分析与设计过程中略去无关的细节，并充分地体现出不同层面上系统的整体视图。在软件系统不断精化的过程中，逐步填充与完善相应的实现细节，使模型从抽象到具体，从而快速且高质量地设计并开发一个相关系统。因此，从这一个意义上看，软件的开发与设计过程可以定义为：针对一个实际问题进行建模、转换和精化模型，直至生成可执行代码的过程。

企业架构则是从更高的层面与视角来对一个软件系统的开发目标与所涉及到的不同维度特征和要素进行系统化的组织与建模。作为一个复杂系统的设计方法论，企业架构关注于业务、数据、应用、技术等多个领域，不仅描述了业务架构、数据架构、应用架构和技术架构等模型以及模型之间的相互关系，还定义了具体的设计与开发实施路线图以及利用架构来控制软件项目开发全过程的管理与治理方法。例如Zachman模型围绕一个系统的分析与设计提出了6×6的分析与设计矩阵，其中6列表示了系统的6个不同的维度视角，即在数据、流程、网络、人员、时间和动机等不同的维度下来透视整个企业级软件开发过程中所涉及的核心业务；同时，模型中的6行表示了软件工程中软件分析设计与开发过程的6个不同阶段，在每一个阶段下，针对一个特定的领域来分别建立模型，其包括语义、概念、逻辑、物理、组件和功能等模型，从而通过全局化的视角来审视与软件系统开发相关的业务、信息、技术和应用之间的相互作用关系，以及这种关系对企业业务流程与功能的影响。

在MDA和EA这两种新的视角下，有必要对目前软件系统的分析与设计过程进行重新审视与梳理，建立一种业务分析与IT设计相支撑的均衡化的分析方法与能力，特别针对实际业务高度复杂、高度集成的企业软件开发环境下，如何利用规范化和标准化的统一建模理论方法来帮助我们提高面对实际的企业应用软件系统在建模与开发过程中的分析与设计能力，已成为了目前我们必须解决的一个

关键问题，而这一问题也突出反映了目前在高校的理论教学环节与企业软件开发的实践过程中存在脱节的一个软肋。这不仅是本书所希望进行探索和尝试解决的一个目标，同时也是与其他软件工程领域关于软件系统分析与设计相关教程之间的区别所在。笔者希望将系统分析与设计过程中存在的系统性与全局化的思维，融入到本书的不同环节与章节内容之中，确保一个软件系统是在一个整体的信息化蓝图下进行有效的分析、设计、实施与管控，避免无序的整合和重复投资，并减少软件分析设计与开发过程中的风险。

“学习不在于有没有人教你，而在于你自己有没有觉悟和恒心”，这是被世人称为“昆虫界的荷马”的法国著名昆虫学家法布尔先生的一条名言。通过8年来的教学工作，笔者深刻地认识并感受到了这一点，也就是说如何有效地发挥出学习主体的主观能动性，这才是教育或者说是学习过程中最为核心的工作。正如古人所云“授人以鱼，不如授人以渔”，所指的“鱼”就是我们常常所说到知识与概念，而“渔”则是实现目标的一种方法或能力，可能在初始使用“渔”的过程中，会存在许多的困惑与茫然，但是如果我们抱有觉悟和恒心，就可以利用“渔”不断地获取“鱼”的过程中，同时一定会有更多的能力提升与知识收获。这也是笔者经历了软件企业与高校教学的不同工作环境后，对于现代教学的一些初浅认识与感受，也希望可以通过本书的出版达到抛砖引玉的作用。一方面，能够让更多的学习者在此基础上不断地进行批判式的学习，获得更加有意义的知识，同时也不断地“刺激”和促进笔者自身能力水平的进一步提高，实现教学相长；另一方面，因笔者本身的知识局限性，也借此可以得到更多专业人士的批评与指正，或者也希望存在一些针对系统分析与设计领域思想和观点的共鸣，如果能够达成以上目标，这将是对本书最大的肯定。

亚里士多德在《形而上学》的开篇中写道：“求知是人类的本性，我们乐于使用我们的感觉就是一个说明。”由于软件的系统分析与设计工作对实践性要求比较强，因而本书强调理论与实际的结合，强调系统分析与设计技术的实用化、工具化与文档化，希望在“求知”探索的过程中，不断通过实践强化“我们的感觉”。

本书适合作为高校计算机学院、软件学院、管理信息系统等专业高年级本科生以及研究生的课程参考教材，也可以作为软件开发人员和设计人员的自修教材与工具书。

本书第1～3章由饶元编写，第4～5章由饶元和赵亮编写，第6～8章由饶元和吴飞龙编写，第9～10章由饶元和杜小智编写，第11～12章由饶元编写。在本书的撰写过程中，得到了许多同仁、专家与同事们的热情指导、交流与鼓励。感谢

CUM 的周抒睿博士对本书提出众多宝贵建议。西安交通大学软件学院社会智能与复杂数据处理实验室的多位研究生也参与了本书的修改与审验过程，其中主要包括了王铮、马丹阳、刘雄、唐建中、聂鹏以及范刘兵等，在此一并感谢。

本书得到了国家科技部“火炬计划”项目(2012GH571817)、陕西省科技攻关项目(2012K06－18 和 2013KRZ10)、榆林市科技项目(2012CXY3－2)、中央高校“科研基金”以及西安交通大学研究生院教学改革等项目的支持，在此一并表示衷心的感谢。

饶元于西安交通大学软件学院
社会智能与复杂数据处理实验室
2014 年 5 月

目　录

第 1 章　软件系统分析与设计的环境与目标 …… (1)

1.1　软件系统分析与设计的认识误区 …… (1)

1.2　软件系统要素与信息化 …… (6)

1.3　软件业务场景分析与信息化的成熟度模型 …… (9)

1.4　系统分析与设计需要解决的关键问题 …… (13)

1.5　本章小结 …… (15)

1.6　思考问题 …… (15)

参考文献与扩展阅读 …… (16)

第 2 章　服务设计与需求工程 …… (17)

2.1　服务设计与质量模型 …… (17)

2.2　需求工程 …… (20)

2.3　需求分析与开发 …… (26)

2.4　需求确认与管理 …… (32)

2.5　需求管理工具的设计 …… (34)

2.6　本章小结 …… (36)

2.7　思考问题 …… (37)

参考文献与扩展阅读 …… (37)

第 3 章　企业架构与 Zachman 模型 …… (39)

3.1　企业建模与信息架构方法 …… (39)

3.2　企业架构 …… (45)

3.3　企业架构方法与信息规划 …… (52)

3.4　本章小结 …… (55)

3.5　思考问题 …… (56)

参考文献与扩展阅读 …… (56)

第 4 章　LOGIN 与基于角色的访问控制 …… (58)

4.1 访问控制……（58）
4.2 基于角色的访问控制模型……（63）
4.3 基于 RBAC 的管理模型与设计应用 ……（75）
4.4 本章小结……（80）
4.5 思考问题……（80）
参考文献与扩展阅读 ……（81）
第 5 章 时间管理与资源计划模型 ……（83）
5.1 计划与时间管理……（83）
5.2 物料资源计划——MRP ……（90）
5.3 项目管理与软件资源计划……（98）
5.4 传统软件的设计模型——资源计划 RP 模型 ……（109）
5.5 传统企业资源计划管理 ERP 软件与 Enterprise 2.0 ……（111）
5.6 本章小结……（116）
5.7 思考问题……（116）
参考文献与扩展阅读 ……（117）
第 6 章 业务规则与规则引擎设计 ……（119）
6.1 业务规则与规则引擎……（119）
6.2 专家系统与规则匹配机制……（130）
6.3 基于规则引擎的应用系统的设计与应用……（144）
6.4 本章小结……（150）
6.5 思考问题……（151）
参考文献与扩展阅读 ……（151）
第 7 章 信息工程与数据设计 ……（153）
7.1 信息工程法……（153）
7.2 数据结构与数据库设计方法……（156）
7.3 基于 E－R 模型的数据库建模原则与范式 ……（165）
7.4 数据模型的设计全生命周期过程与数据完整性……（181）
7.5 数据仓库与数据立方……（189）
7.6 本章小结……（212）
7.7 思考问题……（213）
参考文献与扩展阅读 ……（213）

第 8 章　流程分析与建模 …… (215)
8.1　数据流图(DFD)与数据流设计 …… (215)
8.2　BPM 设计与 BPMN 设计方法 …… (227)
8.3　流程设计应用:工作流与工作流引擎设计 …… (239)
8.4　Petri 网与工作流原语之间的映射与验证 …… (252)
8.5　本章小结 …… (258)
8.6　思考问题 …… (259)
参考文献与扩展阅读 …… (260)
第 9 章　面向对象的分析与建模 …… (261)
9.1　面向对象概念与基础 …… (261)
9.2　面向对象方法与 UML 组成 …… (267)
9.3　需求功能分析与用例建模 …… (284)
9.4　动态行为分析与建模 …… (295)
9.5　静态分析与建模 …… (312)
9.6　组件、部署分析与建模 …… (323)
9.7　本章小结 …… (326)
9.8　思考问题 …… (326)
参考文献与扩展阅读 …… (327)
第 10 章　面向对象系统的设计原则与设计模式 …… (328)
10.1　面向对象设计原则 …… (328)
10.2　设计模式 …… (348)
10.3　设计模式的应用 …… (354)
10.4　本章小结 …… (386)
10.5　思考问题 …… (386)
参考文献与扩展阅读 …… (386)
第 11 章　软件体系结构与应用 …… (388)
11.1　软件体系结构的基本概述 …… (388)
11.2　软件体系结构的关键技术 …… (393)
11.3　典型的软件体系结构模式及其应用 …… (403)
11.4　基于体系结构的可重用软件开发方法 …… (416)
11.5　本章小结 …… (428)

11.6 思考问题 …… (429)
参考文献与扩展阅读 …… (429)
第 12 章 MDA 与设计发展方向 …… (431)
12.1 MDA 体系结构 …… (431)
12.2 MDA 的核心技术 …… (436)
12.3 基于 MDA 的软件开发过程 …… (442)
12.4 本章小结 …… (448)
12.5 思考问题 …… (448)
参考文献与扩展阅读 …… (448)
后记 …… (450)

第1章　软件系统分析与设计的环境与目标

信息技术领域的快速发展与知识的快速更新，使得所开发大量的软件系统都在面临着技术升级、业务操作与市场环境快速变化的挑战，如何适应这样的变化，已成为每一个软件的设计者必须面对与考虑的关键问题。特别是由于软件分析与设计方法的快速演变，对于采用什么样的分析方法、设计技术以及 CASE 工具来协助系统设计师在完成软件所需要的功能、保证软件功能的灵活性与可扩展性的前提下，为用户提供在人机交互过程中的友好体验与满足感，也成为了设计师必须要考虑的关键性因素。因此，本章从软件系统开发与设计过程中环境变化所引起的系统分析与设计的认识误区出发，针对软件系统的基本要素以及信息化的核心特征，分析软件设计过程中所涉及到的一些业务场景，并梳理出系统分析与设计过程的目标及要解决的关键性问题。

1.1　软件系统分析与设计的认识误区

1.1.1　程序与软件的基本概念

图灵奖获得者、Pascal 语言之父——尼古拉斯·沃思(Nicklaus Wirth)从程序逻辑结构的角度曾提出了“程序＝算法＋数据结构”的著名公式。在大量的程序设计语言开发和编程实践经验的基础上，沃思于 1971 年 4 月份在 *Communications of ACM* 期刊上发表了著名论文《通过逐步求精方式开发程序》(*Program Development by Step Wise Refinement*)，首次提出了“结构化程序设计”(Structure Programming)的概念，并认为可执行的代码程序不应一步而成，而是应该分若干步自顶向下、逐步求精地实现。其中，第一步实现程序的抽象度最高，第二步实现程序的抽象度则会有所降低……最后一步编出的程序即为可执行的程序。采用该方法编程看似复杂，但实际上使得程序易读、易写、易调试、易维护，且易于验证。这种以“自顶向下”或“逐步求精”为基本思想的结构化程序设计方法，在程序设计领域

引发了一场革命，成为程序开发的一个标准，并在随后发展起来的软件工程中获得了广泛应用。

软件是指一系列按照特定逻辑规则来组织并能够提供用户所需要的功能和性能的计算机指令或计算机程序集合。其不仅应具有程序能够处理、控制与操作的数据结构，还应反映一定的业务逻辑规则。这就需要根据软件自身的功能性需求以及程序执行操作的全过程提供相应的分析、设计、执行、测试以及维护等涵盖整个软件开发生命周期的文档。因此，在我国的国家标准中将软件定义为：与计算机系统操作有关的计算机程序、规程、规则，以及可能有的文件、文档及数据。即：软件应该包含特定领域信息的数据结构，提供相应功能和性能的计算机程序或指令集合以及利用这些程序或指令集合针对数据结构的操作；同时，还包括描述程序功能需求以及程序操作和使用的相应文档。简而言之，软件就是在特定环境下的程序、数据加文档的集合，可用如下公式来表示：

$$\text{软件} = \text{程序} + \text{数据} + \text{文档}$$

目前，软件已成为现代社会中办公、业务管理、金融投资、移动娱乐与休闲生活等各个方面的一项基础性设施。一方面，随着业务需求、数据结构与分类、操作方式以及逻辑复杂度的不断变化，如何有效地分析与设计一个合理的软件系统以适应不同环境和条件的需求，特别是在业务需求不断变化的环境下，如何设计出一个具有可用性、灵活性与可扩展性的软件系统变得十分困难；另一方面，由于自顶向下功能分解的分析方法极大地限制了软件的可重用性，从而导致了对相同软件实体对象在分析设计过程中存在着大量的重复性工作，直接导致了软件生产率与可维护性的降低。因此，如何将已有的软件资产进行有效地封装和重用，则成为软件工程领域中另一个巨大的、现实的挑战。

1.1.2 软件特征与软件系统分析与设计的方法

软件系统(Software Systems)是指由系统软件、应用软件以及中间件组成的计算机软件系统，它是计算机系统中由软件组成的部分。例如，系统软件包括操作系统软件、数据库系统软件与管理软件以及系统性能监测软件等。其中操作系统软件可用于管理计算机的物理资源和控制软件程序的运行，包括了处理器管理、存储资源管理、文件管理、设备管理和作业管理等功能，并实现了进程(任务)调度、同步机制、死锁防止、内存分配、设备分配、并行机制、容错和恢复机制等支撑性的功能。数据库系统软件则是用来支持数据管理和存取操作的软件，它将常驻在计算机系统内的一组数据集合组织起来并按一定的规则形成数据库，在关系代数与关系模式的理论支撑下来定义数据实体之间的关系，并通过数据结构化描述与查询语言(SQL 语言)来描述和定义对数据的操作；数据库管理系统软件则是为用户提

供数据存取、使用和修改等操作的软件。而大量针对特定业务需求的功能化软件均属应用软件。上述的软件系统不仅仅包括可以在计算机(如智能移动终端、网络交换机等)上运行的软件程序,同时也包括与这些程序相关的技术文档。此外,这一系列的软件还包括以下特征:

(1)无形性,即软件没有具体的物理形态,只能通过程序的运行状况来了解软件的功能、特性和质量。

(2)逻辑性,即软件渗透了大量的脑力劳动,特别是逻辑思维、智力活动和技术水平。逻辑的严谨性、算法的合理性与有效性是保障软件产品质量的关键。

(3)目标性,软件的设计与开发往往是为了解决明确的、具体的问题,目标性越强,软件所需要或能够解决的问题就会越明确,且更加具体。

(4)环境依赖性,即软件的开发和运行必须依赖于特定的计算机系统硬件与软件环境,这同时也对软件可移植性提出了新的要求。

(5)可复用性,即开发出来的软件程序代码以及系统功能模块具有较高的可复用能力,这一方面可提高系统的开发效率,另一方面可快速形成多个软件副本。

(6)软件老化,即随着软件需求的不断变更以及技术的不断更新,软件系统在功能与性能等方面也不断地老化。

鉴于软件存在上述特征,如何从一个业务需求出发,通过系统的分析与设计过程,为软件的开发实现与维护运营提供一个有效的保障,已经成为了软件分析师与设计者必须面对的挑战。同时,为了研究、理解和解决客观世界中存在的具体业务问题,在对客观现实中存在的业务过程、文字与图表、符号与逻辑关系式以及实体关系进行抽象后,通过构造一个直观且易于表现和分析的可视化模型,帮助人们分析和研究客观事物中存在的逻辑关系与问题,成为了系统分析与设计的关键。

系统分析(Systems Analysis)是将复杂的业务信息进行合理的整理、加工、归纳和抽象的逻辑化过程,通过简化其内涵并分解成相对简单的要素,找出这些组成要素的根本属性与彼此之间的各种关联来实现对系统的定义与理解。由于这些要素相互之间存在着关联,相互制约、相互影响、相互作用,同时又相互排斥,这就导致了整个系统的复杂性、多变性和不确定性。如何利用模型来对客观事物进行抽象并简化表示,同时协助系统分析师解决具体的需求问题,往往取决于具体问题在不同的分析过程中建立什么样的模型。因此,也衍生出了多种不同的软件分析与设计的方法与理念,其中主要包括结构化方法、面向数据结构的软件开发方法、面向问题的分析法、原型化方法、信息工程方法、可视化开发方法、面向对象的软件开发方法等。这些软件开发方法在过去的软件危机期间曾经发挥了很大的作用,促进了软件业的发展。

沃思则提出了结构化程序开发过程,并对软件开发的环节进行了较为严谨的

描述和定义。他所开展的软件设计主要根据对不同业务的调查分析和论证基础之上来确立的，虽然可以减少系统开发过程中存在的盲目性，但是只适用于开发规模不大、所有业务过程可以事先加以严格说明的系统。而对于那些业务规模较大、存在较多不确定性因素的软件系统而言，进行高效率的开发相对比较困难。因此，在借鉴和模拟人类习惯和思维方式的基础上，Grady Booch，Jim Rumbuahg 和 Ivar Jacobson 三位大师分别从不同的视角提出了基于面向对象的软件工程分析与设计方法，并通过 UML(Uniform Model Language)实现了全局统一的定义，即要求描述业务的问题空间(也称为问题域)与实现空间(也称为求解域)在结构上尽可能一致，使得程序易于理解和维护。

在面向对象的开发过程中采用了迭代的方式，即在分析、设计、实现等阶段采用多次迭代的方式来提高开发的效率，从而使软件开发各阶段的划分变得较为模糊。其中，在面向对象的分析过程中，首先需要抽取和整理用户需求并建立问题域的领域分析模型，并利用面向对象的逻辑来建立求解域模型；其次，利用面向对象的方法进一步针对业务系统进行分析和设计，确定系统的实现策略和目标，并在顶层结构的设计基础上，通过对象设计来确定解空间中的类、关联、接口形式以及实现服务的具体算法；最后，在进行具体代码实现的过程中，将设计结果转换成用程序语言描述的面向对象程序，并通过软件的调试和测试来验证程序是否能够满足相应需求。整个过程通过对象的抽象以及类之间逻辑结构定义来进行逐步精化，最终反映出整个的完整视图。

当前，分布式网络与并行计算环境的发展要求软件能实现跨空间、跨时间、跨设备以及跨用户的资源共享与协同，这进一步导致了软件的规模、复杂度以及功能的增加。由于传统的软件开发方法无法支持异构环境下的业务协同、系统集成以及大粒度的代码重用，为适应这种新需求，基于组件的软件开发技术应运而生，如 COM+，JAVA Bean，Web Service 等。这种开发模式一方面针对分布式环境下的浏览器/服务器结构进行模块化集成，希望将软件的开发变得类似于硬件一样，即通过定义标准的接口来实现不同组件之间的集成；另一方面，利用模块化方法将复杂的系统分解为互相独立且协同工作的部件，并通过反复重用这些预先编译好的、功能明确的软件组件来实现应用系统的扩展和更新，从而利用客户和软件之间的统一接口来实现跨框架与平台的互操作。

1.1.3 目前存在的一些认识误区

目前，在模型驱动体系结构(MDA)的影响下，软件系统分析与设计建模在软件工程领域中的重要性越来越高。在实际的软件开发过程中，程序员或系统分析师们常常会受到编程语言的影响，但是面对越来越复杂软件开发需求，人们发现软

件系统正在与其他学科发生越来来紧密的融合，例如网络游戏软件不仅需要利用软件工程方法来设计与实现游戏的基础功能，同时还需要利用审美学、社会学、心理学以及系统工程学等多个领域的交叉学科知识来提供支持。因此，一个好的软件系统分析与设计人员或系统架构师，应该具备综合的技术实现能力与素养，并培养如图 1.1 所示的知识体系。

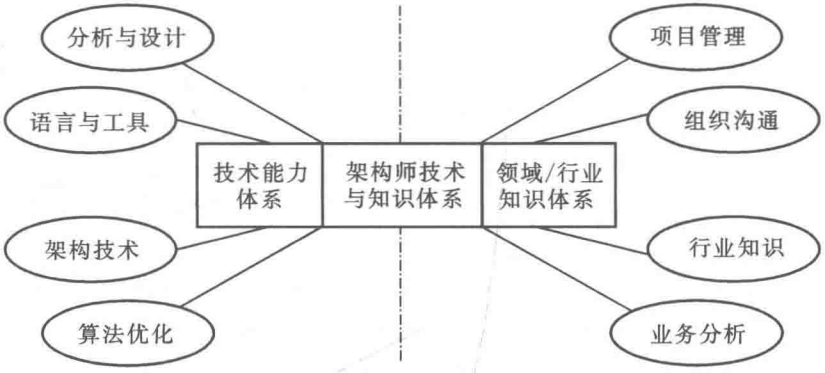


图 1.1　系统架构师技术与知识体系结构示意图

为了更好地适应现代软件系统的设计与优化需求，系统分析与设计人员以及架构师应该突破传统设计过程中存在的一些认识误区与不足。主要体现在以下几个方面：

(1)缺乏整体性思维，尤其是从宏观的整体业务逻辑到微观的程序设计实现的分析能力。由于系统微观实体之间的交互机制是构造整体宏观应用系统的基础，因而微观交互机制和宏观系统之间的关系是软件系统复杂性研究的核心议题。例如，如何利用万维网在宏观层面的演化机制来影响一个具体的 Web 信息系统不同用户实体之间的动态交互行为？可否在微观的 Web 信息系统设计过程中通过定义标准的交互规则来引导万维网的宏观结构向预期的网络拓扑的演化方向发展？而目前针对万维网的整体动态研究主要是基于简化的抽象模型，缺乏万维网对具体的 Web 信息系统的交互行为产生影响的深入研究。

(2)缺乏必要的开放视野，常常限于局部性的思考。在目前适应性软件系统的研究过程中，很少考虑到对其他信息系统的开放支持。往往由于仅局限于针对某一类具体问题进行研发，缺少对不同应用领域的系统之间进行迁移的横向比较，因而提出的系统设计方案在特定的项目背景下只能够适用于特定的应用领域，从而造成所设计的软件系统不仅无法适应业务需求的快速变化，同时也使得不同的业务系统之间的集成能力变弱。