

中文 SQL Server 2000 数据库 应用开发

李代平 章文 张信一 编著

北 京

冶 金 工 业 出 版 社

2002

内容简介

本书内容包括：系统开发一般原理，项目规划，网络数据库开发理论，数据库的设计，COM 和 DCOM, 开发工具 ODBC、RDO、OLE DB 和 ADO、DB-Library、DAO 和 ODBC Direct 使用技术，同时介绍了 SQL Server 2000 与 Internet、Visual C++ 6、Visual Basic、PowerBuilder、C++Builder 和 Delphi、ASP、XML，以及与其他软件的连接与集成，另外还介绍了 SQL Server 2000 与 Access、Sybase、Oracle 的转换。本书不仅介绍了引导计算机中级应用人员提高数据库开发能力的知识与经验，而且也介绍了一部分只有老练开发应用人员才知道的小信息段。

本书可作为大专院校相关专业的教材和参考书，同时也可供计算机专业的高级人员参考。

图书在版编目 (CIP) 数据

中文 SQL Server 2000 数据库应用开发 / 李代平等编
著. —北京：冶金工业出版社，2002.6
ISBN 7-5024-3020-2

I. 中... II. 李... III. 关系数据库-数据库管理系统,
SQL Server 2000-程序设计 IV. TP311.138

中国版本图书馆 CIP 数据核字 (2002) 第 031648 号

出版人 曹胜利 (北京沙滩嵩祝院北巷 39 号, 邮编 100009)

责任编辑 程志宏

中山市新华印刷有限公司印刷; 冶金工业出版社发行; 各地新华书店经销

2002 年 6 月第 1 版, 2002 年 6 月第 1 次印刷

787mm×1092mm 1/16; 27.75 印张; 642 千字; 433 页; 1-2000 册

45.00 元

冶金工业出版社发行部 电话: (010) 64044283 传真: (010) 64027893

冶金书店 地址: 北京东东西大街 46 号 (100711) 电话: (010) 65289081

(本社图书如有印装质量问题, 本社发行部负责退换)

前 言

一、关于 SQL Server 2000

在信息已成为各行各业重要资源的今天，数据库以其强大的操作和管理功能成为高速信息表达的途经。SQL Server 2000 是 Microsoft 公司最新推出的关系数据库管理系统 SQL Server 的最新版本，也是目前世界上大型的数据库系统之一，其功能的强大是众所周知的。然而，在它强大的功能背后却隐含着涉足者的艰难。要掌握这样一个复杂而庞大的系统，需要阅读很多书籍和文献，并进行大量的实际工作。这在目前信息爆炸的年代是难以让人接受的，许多人都希望在短时间内掌握这门先进的技术，鉴于此我们编写了本书。

二、本书结构

本书共分三个部分介绍了 SQL Server 2000 基本理论和系统开发的方法。其结构如下：第一部分（第 1~5 章）介绍了数据库理论知识。

第 1 章系统开发一般原理，包括：应用程序开发方法、对象的发现和标识、数据分析和数据库设计等。

第 2 章项目规划，包括：需求分析、研制系统的必要性、总体方案、系统平台设计、系统组织与信息分类编码标准及接口等。

第 3 章网络数据库开发理论，包括：三层结构理论和 C/S/S 模式等。

第 4 章数据库的设计，包括：数据库设计概述、数据库设计、物理数据库设计和设计联合数据库服务器等。

第 5 章 COM 和 DCOM，包括：COM 结构、COM 特性、DCOM 结构、DCOM 特性和配置应用程序等。

第二部分（第 6~17 章）介绍了利用开发工具进行系统开发的方法。

第 6 章 ODBC 开发技术，包括：ODBC 概述和配置 ODBC 数据源等。

第 7 章远程数据库对象开发，包括：RDO 体系结构、使用 RDO、在 Visual Basic 中使用 RDO 对象、将 RDO 对象增加到 Visual Basic 中等。

第 8 章 OLE DB 和 ADO 技术，包括：OLE DB 体系结构、ADO 及其体系结构、OLE DB 和 ADO 文件、Visual Basic 引用 ADO 对象、将 ADO 2 参考增加到 Visual Basic 中等。

第 9 章 DB-Library 程序设计，包括：DB-Library 开发环境、数据类型定义、主要数据结构、参数和选项设置、错误和消息处理、数据库访问过程、应用程序的编辑、编译和链接以及 DB-Library 与 ODBC 编程方法比较等。

第 10 章用 DAO 和 ODBC Direct 开发程序，包括：DAO 体系结构、在 Visual Basic 中使用 DAO 对象、将 DAO 3.5 参考增加到 Visual Basic 中、ODBC 常用性能等。

第 11 章 SQL Server 2000 与 Internet，包括：HTML 文档结构、SQL Server Web Assistant、实现 Web 发布的系统存储过程等。

第 12 章 Visual C++ 与 SQL Server 2000 连接，包括：调用级接口、连接到数据库、操作 SQL Server 数据、获取数据、游标、调试 ODBC 代码等。

第 13 章 Visual Basic 与 SQL Server 2000 连接, 包括: 开发高效应用程序、使用开放式数据库连接、数据访问对象和远程数据对象等。

第 14 章 SQL 和 PowerBuilder 开发技术, 包括: SQL 和 PowerBuilder 的关系、将 PowerBuilder 连接到 SQL Server 数据库、PowerBuilder 内的 SQL、事务处理对象、逻辑工作单元、检查 SQL 失败、DECLARE 和 FETCH、动态 SQL、游标绘制等。

第 15 章 C++Builder 和 Delphi 的应用, 包括: 建立数据库访问通道、SQL Builder、存储过程、连接数据库和使用数据控件、Delphi 的数据库模型等。

第 16 章 ASP 技术和数据库, 包括: ASP 对象模型、ASP 基础和使用 ADO 对象的 ASP 等。

第 17 章 XML 技术, 包括: XML 技术应用、XML 文档与数据库表以及数据访问等。

第三部分介绍了数据库的转换技术。

第 18 章数据库集成与转换, 包括: 在 SQL Server 2000 中使用 Excel、Access 与 SQL Server 的比较、将 Access 数据库移植到 SQL Server、使用 DTS 向导转移数据、移植 Access 查询、Visual Basic 程序移植、将 Access 作为 SQL Server 2000 的前端、Oracle 与 SQL Server 2000 的数据转换、Sybase 和 Microsoft 的发展、在 IIS 和 IE 中使用 SQL Server 等。

书中用了近 200 个示例和程序对每一部分内容进行解释和说明。每一个程序都有分析说明。这里提供了高级人员所需要的知识, 讲述了一些老练开发应用人员才知道的小信息段, 高级人员在此可以得到前所没有的收获。

三、本书特点

本书遵循循序渐进的原则, 结构严谨, 内容详实, 涉及面广, 实例丰富。

本书是作者根据近几年对数据库系统的教学、研究与开发, 以及对数据库系统的实际应用, 并结合 SQL Server 2000 系统软件编写而成。在制定提纲前我们作了一些调查: 对 SQL Server 2000 新用户, 介绍了 SQL Server 2000 基础知识和基本操作; 对有经验的高级开发人员, 介绍了在数据库系统开发中各种开发工具的应用与技巧, 及一些只有老练开发应用人员才知道的技术, 还特别介绍了系统开发的项目规划和应用系统实例。

四、适用对象

本书可作为大专院校相关专业师生的教材和参考资料, 也可供计算机专业的高级人员参考。

本书由李代平、章文、张信一编写, 另外, 段生林、袁成武、张智光、谭细明、杨志扬也做了部分工作, 最后由李代平统稿。在编著者中有一部分是网络信息系统课题研究组和网络并行计算课题研究组的开发人员, 他们在研究中有机会深入应用 SQL Server 各类版本及系统开发工具, 也有机会应用其他的数据库管理系统。因此, 他们熟知各类数据库管理系统的内幕, 在多次的系统研究与开发中, 对各种开发工具运用自如。

由于成书时间仓促, 作者水平有限, 书中的错误和疏漏在所难免, 恳请读者批评指正。

编 者

2002 年 5 月

目 录

第1章 系统开发一般原理.....1

- 1.1 应用程序开发方法1
 - 1.1.1 自顶向下的开发法2
 - 1.1.2 原型法3
 - 1.1.3 面向对象的分析和设计法4
- 1.2 对象的发现和标识5
 - 1.2.1 动机5
 - 1.2.2 方法6
 - 1.2.3 三视图模型(3VM)6
 - 1.2.4 基于语言的信息分析系统7
- 1.3 数据分析9
- 1.4 数据库设计10
 - 1.4.1 规范化11
 - 1.4.2 非规范化14
 - 1.4.3 确保数据完整性14
 - 1.4.4 创建触发器、存储过程和索引14
 - 1.4.5 数据库设计的文档化15
- 1.5 小结15

第2章 项目规划.....16

- 2.1 概述16
- 2.2 需求分析16
 - 2.2.1 概况描述16
 - 2.2.2 企业特点17
 - 2.2.3 作业主要流程19
- 2.3 研制系统的必要性22
 - 2.3.1 企业经营目标22
 - 2.3.2 实现目标存在的问题22
 - 2.3.3 实现目标采取的措施23
 - 2.3.4 系统概述24
 - 2.3.5 系统对实现目标的支持24
- 2.4 总体方案27
 - 2.4.1 系统目标28
 - 2.4.2 确定系统方案原则32
 - 2.4.3 系统结构35
 - 2.4.4 分系统界面及技术指标38
- 2.5 系统平台设计43

- 2.5.1 概述43
- 2.5.2 数据库系统设计44
- 2.5.3 网络系统设计50
- 2.6 系统组织与信息分类编码标准52
 - 2.6.1 概述52
 - 2.6.2 编码内容53
- 2.7 接口55
 - 2.7.1 外部接口55
 - 2.7.2 系统结构接口56
 - 2.7.3 子系统间的接口56
 - 2.7.4 人机接口57
- 2.8 小结57

第3章 网络数据库开发理论58

- 3.1 三层结构理论58
- 3.2 应用举例61
 - 3.2.1 设计中间层63
 - 3.2.2 设置DCOM65
 - 3.2.3 设计表示层66
 - 3.2.4 C/S/S模式下的动态查询技术67
- 3.3 小结75

第4章 数据库的设计76

- 4.1 数据库设计概述76
- 4.2 数据库设计76
 - 4.2.1 逻辑数据库设计77
 - 4.2.2 数据完整性77
 - 4.2.3 设计数据库应注意的事项78
- 4.3 物理数据库设计79
 - 4.3.1 RAID79
 - 4.3.2 分区83
 - 4.3.3 使用文件组放置数据85
- 4.4 设计联合数据库服务器87
 - 4.4.1 设计分区88
 - 4.4.2 设计高度可用的联合数据库服务器90
 - 4.4.3 备份和还原联合数据库服务器91
- 4.5 小结91

目 录

第 5 章 COM 和 DCOM	92	第 7 章 远程数据库对象开发	128
5.1 COM 和 DCOM	92	7.1 概述	128
5.2 COM 概述	92	7.2 RDO 体系结构	128
5.3 COM 结构	94	7.3 使用 RDO	129
5.3.1 对象与接口	94	7.4 在 Visual Basic 中使用 RDO 对象	129
5.3.2 客户机/服务器模型	96	7.4.1 初始化 RDO 引擎和环境	130
5.3.3 COM 系统程序库	96	7.4.2 终止 rdoEnvironment	131
5.4 COM 特性	96	7.4.3 使用 RDO 连接	131
5.4.1 语言无关性	97	7.4.4 终止 RDO 连接	132
5.4.2 进程透明特性	97	7.4.5 检索数据	132
5.4.3 可重用性	98	7.4.6 使用结果集和预准备语句	134
5.5 DCOM 概述	98	7.4.7 动态的 SQL 和 RDO Execute	135
5.6 DCOM 结构	99	7.4.8 修改数据	135
5.7 DCOM 特性	100	7.4.9 用 RDO SQL 修改数据	138
5.8 配置应用程序	102	7.4.10 错误处理	141
5.8.1 DCOMCNFG	102	7.5 将 RDO 对象增加到 Visual Basic 中	141
5.8.2 OLEVIEW	105	7.6 应用方法	142
5.9 小结	106	7.6.1 多个结果集的使用	142
第 6 章 ODBC 开发技术	107	7.6.2 异步查询	143
6.1 ODBC 概述	107	7.6.3 服务器端游标	144
6.1.1 使用 ODBC 的原因	107	7.6.4 管理数据并发性	145
6.1.2 统一连接	108	7.7 小结	146
6.1.3 建立技术	108	第 8 章 OLE DB 和 ADO 技术	147
6.1.4 ODBC 应用程序	109	8.1 概述	147
6.1.5 ODBC 驱动管理	109	8.2 OLE DB 体系结构	147
6.1.6 ODBC 驱动程序	109	8.3 ADO 简介	148
6.1.7 数据源	109	8.4 ADO 体系结构	148
6.2 配置 ODBC 数据源	110	8.5 OLE DB 和 ADO 文件	149
6.2.1 ODBC API 的使用基础	114	8.6 Visual Basic 引用 ADO 对象	149
6.2.2 ODBC 初始化函数	114	8.6.1 连接 SQL Server	149
6.2.3 用 ODBC API 检索数据	118	8.6.2 检索数据	152
6.2.4 用 ODBC 修改数据	122	8.6.3 预备 SQL 语句和 Command 对象	158
6.2.5 调用存储过程	124	8.6.4 用 ADO Connection 对象 执行动态的 SQL 语句	159
6.2.6 ODBC 错误处理和 SQLGetDiagRec	126	8.6.5 修改数据	160
6.3 小结	127		

目 录

8.6.6 用 Command 对象调用存储过程	165	10.3.3 用 DAO 检索数据	201
8.6.7 错误处理	167	10.3.4 限制数据	202
8.7 将 ADO 2 参考增加到 Visual Basic 中	168	10.3.5 关闭 Recordset	203
8.8 ADO 的高级功能	168	10.3.6 执行动态的 SQL	204
8.8.1 成批修改	168	10.3.7 修改数据	204
8.8.2 使用多个结果集	169	10.3.8 使用 QueryDef	206
8.8.3 事务操作	170	10.3.9 错误处理	211
8.8.4 存储二进制数据	172	10.4 将 DAO 3.5 参考增加到	
8.9 小结	174	Visual Basic 中	212
第 9 章 DB-Library 程序设计	175	10.5 终止 Jet 引擎	213
9.1 DB-Library 开发环境	175	10.6 常用性能	213
9.2 DB-Library 数据类型定义	176	10.7 ODBC Direct	214
9.3 DB-Library 中的主要数据结构	178	10.8 体系结构	214
9.4 DB-Library 参数和选项设置	179	10.9 创建对象	215
9.4.1 设置登录超时时限	179	10.9.1 CreateWorkspace 使用方法	215
9.4.2 设置命令响应超时时限	180	10.9.2 连接到 SQL Server	216
9.4.3 设置可同时打开的连接数	180	10.10 小结	218
9.4.4 设置和清除选项	180	第 11 章 SQL Server 2000 与 Internet	220
9.5 错误和消息处理	182	11.1 SQL Server 2000 与 Internet 概述	220
9.5.1 DB-Library 错误处理	182	11.2 HTML 文档结构	220
9.5.2 SQL Server 消息处理	184	11.2.1 什么是 HTML 语言	220
9.6 DB-Library 的数据库访问过程	186	11.2.2 标签	221
9.6.1 连接 SQL Server	186	11.2.3 常用标签属性用法举例	221
9.6.2 命令处理	188	11.2.4 HTML 文档结构举例	223
9.6.3 结果处理	192	11.3 SQL Server Web Assistant	224
9.6.4 关闭连接	192	11.3.1 Web Assistant 实现过程	224
9.7 DB-Library 应用程序的		11.3.2 使用 Web Assistant Wizard	
编辑、编译和链接	193	发布数据库	224
9.8 DB-Library 与 ODBC 编程方法比较	195	11.4 实现 Web 发布的系统存储过程	230
9.9 小结	196	11.4.1 创建 Web 发布任务	230
第 10 章 用 DAO 和 ODBC Direct 开发程序	197	11.4.2 执行 Web 发布任务	234
10.1 DAO 概述	197	11.4.3 删除 Web 发布任务	235
10.2 DAO 体系结构	197	11.5 小结	235
10.3 在 Visual Basic 中使用 DAO 对象	198	第 12 章 Visual C++ 与 SQL Server 2000	
10.3.1 创建对象	198	连接	236
10.3.2 连接 SQL Server	199	12.1 编程基础	236

目 录

12.2 调用级接口.....	238	14.1 SQL 和 PowerBuilder 的关系.....	281
12.3 连接到数据库.....	239	14.2 PowerBuilder 与 SQL Server 连接.....	282
12.3.1 ODBC 环境.....	239	14.3 PowerBuilder 内的 SQL.....	283
12.3.2 连接字符串.....	240	14.4 事务处理对象.....	284
12.3.3 建立连接.....	241	14.5 逻辑工作单元.....	285
12.3.4 连接到服务器.....	241	14.6 检查 SQL 失败.....	286
12.4 操作 SQL Server 数据.....	242	14.7 DECLARE 和 FETCH.....	287
12.4.1 语句分析.....	242	14.7.1 DECLARE 语句.....	287
12.4.2 参数描述.....	244	14.7.2 使用 OPEN、EXECUTE 和 FETCH.....	288
12.5 获取数据.....	247	14.8 动态 SQL.....	289
12.6 游标.....	249	14.8.1 类型 1.....	289
12.6.1 滚动游标.....	249	14.8.2 类型 2.....	289
12.6.2 游标敏感性.....	249	14.8.3 类型 3.....	290
12.6.3 块游标和行集.....	249	14.8.4 类型 4.....	291
12.7 调试 ODBC 代码.....	252	14.9 粘贴 SQL 语句.....	293
12.8 大数据项的使用.....	253	14.9.1 粘贴 SELECT.....	293
12.9 小结.....	257	14.9.2 粘贴 INSERT.....	295
第 13 章 Visual Basic 与 SQL Server 2000		14.9.3 粘贴 UPDATE.....	295
连接.....	259	14.9.4 粘贴 DELETE.....	295
13.1 开发高效应用程序.....	259	14.10 游标绘制.....	295
13.1.1 调用级和对象接口.....	259	14.10.1 UPDATE WHERE CURRENT 语句.....	296
13.1.2 三层应用模型.....	260	14.10.2 DELETE WHERE CURRENT 语句.....	296
13.2 使用开放式数据库连接.....	263	14.10.3 DECLARE PROCEDURES 语句.....	296
13.2.1 开放式数据库连接驱动程序.....	263	14.10.4 FETCH FROM Procedure 语句.....	296
13.2.2 连接 ODBC 数据源.....	265	14.11 故障检测.....	297
13.2.3 处理查询.....	266	14.12 PowerBuilder 和 SQL Server.....	299
13.2.4 结果集.....	267	14.12.1 连接数据库的途径.....	299
13.3 数据访问对象.....	270	14.12.2 ODBC 和底层驱动的比较.....	299
13.3.1 数据定义语言操作.....	270	14.12.3 连接数据库.....	300
13.3.2 数据操纵语言操作.....	274	14.12.4 使用 PowerBuilder 管理数据库.....	304
13.4 远程数据对象.....	277	14.12.5 PowerBuilder 数据库对象.....	306
13.4.1 RDO 与 DAO 比较.....	277	14.13 Delphi 和 PowerBuilder 的选用.....	312
13.4.2 RDO 层次和 rdoEngine 对象.....	278		
13.4.3 rdoEnvironment 对象.....	279		
13.5 小结.....	280		
第 14 章 SQL 和 PowerBuilder 开发技术.....	281		

目 录

14.14 PowerBuilder 编程技巧	312	16.1 ASP 技术概述	347
14.14.1 焦点切换	312	16.2 ASP 对象模型	349
14.14.2 数据转储和调入	313	16.3 ASP 基础	350
14.14.3 创建数据报表	313	16.4 使用 ADO 对象的 ASP	352
14.14.4 打印条形码	313	16.4.1 连接到 SQL Server	352
14.14.5 在 PowerBuilder 中数字的 中英文大写转换	316	16.4.2 结束一个连接	353
14.15 小结	319	16.4.3 使用 ADO Recordset 对象 检索数据	353
第 15 章 C++Builder 和 Delphi 的应用	320	16.4.4 关闭记录表	356
15.1 建立数据库访问通道	320	16.4.5 包含连接的 Session	356
15.1.1 访问数据库表	320	16.4.6 使用预准备 SQL 和 Command 对象	358
15.1.2 查询数据库的一般步骤	321	16.4.7 使用 ADO 修改数据	363
15.2 SQL Builder	321	16.4.8 使用 Command 对象执行 存储过程	371
15.3 存储过程	322	16.4.9 错误处理	376
15.3.1 使用 TStoredProc 元件的 一般步骤	323	16.5 小结	378
15.3.2 存储过程的参数	323	第 17 章 XML 技术	379
15.3.3 TStoredProc 的特性、方法 和事件	324	17.1 支持 Web 应用特性	379
15.4 连接数据库和使用数据控件	326	17.2 XML 技术应用	380
15.4.1 TDatabase 元件的常用函数 和事件	326	17.2.1 XML 概述	380
15.4.2 使用数据控件	328	17.2.2 多层体系结构	382
15.5 SQL 与 C++ Builder 应用实例	329	17.2.3 XML 的优点	383
15.5.1 主功能程序源代码	330	17.3 XML 文档与数据库表	383
15.5.2 自动报表程序	336	17.3.1 关系数据产生 XML 文档	383
15.6 Delphi 的数据库模型	340	17.3.2 在表中保存 XML 文档	385
15.7 Delphi 中的数据库编程	341	17.4 数据访问	387
15.7.1 使用数据库访问控件	341	17.5 小结	387
15.7.2 数据控件	343	第 18 章 数据库集成与转换	389
15.7.3 在表单中使用控件	344	18.1 在 SQL Server 2000 中使用 Excel	389
15.7.4 表单向导	344	18.2 Access 与 SQL Server 的比较	390
15.7.5 通过代码来访问数据库	345	18.3 将 Access 数据库移植到 SQL Server	390
15.7.6 通过 Delphi 来管理数据库	345	18.4 移植工具	391
15.8、小结	346	18.4.1 SQL Server Enterprise Manager	391
第 16 章 ASP 技术和数据库	347	18.4.2 数据转换服务	392

目 录

18.4.3 SQL Query Analyzer.....	392	的对比.....	415
18.4.4 SQL Profiler	392	18.12 SQL Server 2000 和 Sybase	
18.5 使用 DTS 向导转移数据.....	393	Adaptive Server 的差别.....	415
18.6 移植 Access 查询.....	397	18.12.1 事务管理模式	416
18.6.1 存储过程	397	18.12.2 兼容模式.....	416
18.6.2 Transact-SQL 脚本	397	18.12.3 隔离等级.....	419
18.6.3 视图	398	18.12.4 保留字.....	419
18.7 Visual Basic 程序移植	401	18.12.5 游标语法.....	420
18.8 将 Access 作为 SQL Server 2000 的		18.12.6 回滚触发器.....	421
前端.....	403	18.12.7 优化程序暗示	423
18.8.1 创建 Access 项目.....	404	18.12.8 优化查询计划	424
18.8.2 使用 Access 项目.....	405	18.12.9 临时表名称.....	426
18.8.3 Data Access Page	407	18.12.10 Raiserror 语句.....	426
18.8.4 SQL Server 的双向数据复制.....	408	18.12.11 数据类型.....	427
18.9 Oracle 与 SQL Server 2000 的		18.12.12 标识列.....	427
数据转换.....	408	18.12.13 Print 语法.....	427
18.9.1 Oracle 简介	408	18.13 与 Microsoft Transaction Server	
18.9.2 Oracle 订阅服务器	410	集成	428
18.9.3 用 Oracle 的 OLE DB		18.13.1 MTS 事务和 IIS	428
提供程序	412	18.13.2 MTS 事务和 SQL Server	431
18.9.4 编辑 DTS 包	413	18.13.3 MTS 事务处理和可视化	
18.9.5 数据变换和转换注意事项.....	413	编程工具.....	432
18.10 Sybase 和 Microsoft 的发展	414	18.14 在 IIS 和 IE 中使用 SQL Server	432
18.11 SQL Server 与 Sybase Adaptive		18.14.1 在服务方访问数据库	432
Server 保留的相似之处	415	18.14.2 使用 IE 进行客户端访问.....	433
18.11.1 系统过程	415	18.15 小结	433
18.11.2 Microsoft T-SQL 与 Sybase T-SQL			

第1章 系统开发一般原理

本章从应用程序的开发方法、对象的发现和标识,以及数据分析和数据库设计,介绍了应用程序开发的原理和方法。

1.1 应用程序开发方法

开发成功的商业应用程序要求用户建立一个在项目的整个生命周期都要用到的完整开发过程。在分布式环境(如两级客户机/服务器应用程序)下开发应用程序的用户利用周期开发来创建、维护和改进自己所提供的解决方案。

应用程序开发周期只是包含项目每一个方面,如质量管理和质量保证等的一个小部分。应用程序开发方法包括三个主要阶段。

1. 重设计过程 (process reengineering)

从某种程度上说是已有问题及其解决方案的发现和定义。重设计的基础就是要捕捉当前过程并确定在何处以及是否可以用基于计算机的解决方案来改进该过程。作为这个过程的一部分,用户收集当前情况下应用程序问题的信息和确定解决这些问题的优化方案,包括将这些方案融入系统内,并通过问题的发现、定义和解决过程不断地管理和提高系统。

2. 结构基础 (architectural foundations)

确定实施在重设计过程中定义的解决方案所使用的工具。它包括了应用程序的软件包的定义以及应用程序、数据源和外部接口之间的相互作用。它还负责监控和优化数据库的性能和应用程序的资源利用。

3. 系统开发 (system development)

系统开发是建立解决方案以便进一步开发和提供优化解决方案的阶段。该阶段包括定义一个用于系统的设计、创建及实现的计划和要求。要实现的系统则通过维护和开发不断地得到监控和提高。

尽管这些阶段相互作用,但每个阶段都有自己的开发生命周期,在其中考虑到了发现、创造和增强。这些阶段实质上是一个循环过程,用户可以在对系统有意义的任一点开始。用户可以从纸上开始然后到计算机上,或者重新设计已有的计算机系统。这个过程如图 1-1 所示。

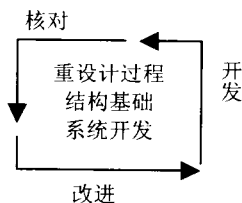


图 1-1 具有单独生命周期的应用程序开发方法

有很多分析和设计的方法。其中一些是经过试验和测试成熟的方法;而另一些则是较

新的不太成熟的方法。一些方法比较适合于特定的场合和设计小组，因此值得加以研究。最常用的一些方法有：自顶向下法、面向对象的分析和设计法、联合应用程序开发法（JAD）、快速应用程序开发法（RAD）、快速循环原型法（RIP）和商业过程设计法。

许多工具如计算机辅助软件工程（CASE）能帮助开发小组实现这些处理过程。这一章简要概述了这些不同方案，然后讨论了用 PowerBuilder 的功能和特点匹配这些分析和设计阶段的过程。还将探讨所有成功的基于数据的应用程序的基石：数据库设计。

1.1.1 自顶向下的开发法

自顶向下的方法通常被认为是传统的或“经典”的软件生命周期。在开发上，它沿袭了结构化分析和结构化设计的方法。自顶向下方法的模型如图 1-2 所示。

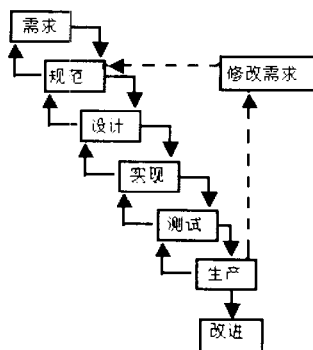


图 1-2 自顶向下的开发方法模型

此模型的第一步是对用户需求的分析和定义，然后在需求规范文档中确定软件的需求，这完全是从系统的外部行为来描述的（注意这不是描述系统内部怎样工作）在该设计阶段中一个重要的因素是系统的行为。详细设计用 Program Design Language（PDL）符号完成，PDL 也称为 Structured English 或伪代码。详细设计用于表达意图和结构而不是实施细节。

自顶向下方法的两个重要内容是：测试和建立文档。测试是在整个软件开发过程中不断地进行的。而文档也是通过每一个阶段不断创建和更新的。因此，自顶向下模型生成了许多文档：规范文档、设计文档、代码文档和许多其他文档，包括数据库、用户和操作手册。这些文档是系统维护和更新必不可少的。

这种开发生命周期方法存在一些固有的问题，最重要的是只有在工作系统建立以后才能对项目需求进行正确的测试。而且只有到那时，才能使需求与功能相匹配。显然这一阶段所提供的解决方案可能无法满足最终用户的期望和要求。在需求规范文档中的问题只有到这时才能发现，因而问题的解决须付出很大代价，因为整个周期不得不重新开始。由于工作模型出现在开发生命周期中较晚的阶段，主要的设计或性能问题直到系统几乎完成后才能遇到。

在当今的商业中，用户正在寻找能够快速和有效地产生的系统。由于自顶向下法在开发周期的后期才产生工作模型，因此开发人员发现这种方法不能满足新的应用程序开发方法要求。解决使用自顶向下法所带来的问题的一个方法就是原型法。

1.1.2 原型法

原型法是一个偏重于设计的方法，在该方法中系统设计人员和一组选定的终端用户在一起非常紧密地工作。他们共同参加交流会，在这种会议中，设计不同的系统方案，然后将其模型化、修改、更新，直到系统设计人员完全明白了系统的要求。

这种方法的明显优点就是用户和系统设计人员紧密地结合在一起，用户看到系统功能并能立即提出反馈意见。用户可提供有关系统切中要害的见解，而这些见解在传统的最终用户评审期间系统设计人员必须花费很大精力才能发现。这种原型有时可进一步细化以生成最终产品，但更常见的情况还是分阶段进行以便将实际的编码框架放置到合适的位置。文档也是这种方法的一部分。利用用户提供的反馈，可以随原型一起创建修正的需求规范文档。

1. 快速循环原型法

快速循环原型（RIP）法充分利用了自顶向下方法模型，再加上原型循环。在原型循环周期终点，最终的需求规范和原型将受到开发小组和用户小组的认可，从而分析阶段也可告一段落。如图 1-3 所示给出了 RIP 方法的思想。

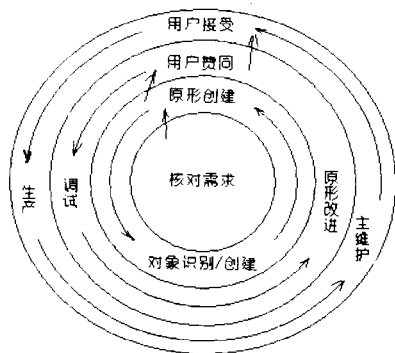


图 1-3 快速循环原型开发方法模型

由于明显地得到了终端用户的高度参与，因此这种方法非常适合于终端用户可以分配足够时间与开发小组一起工作的机构。这种时间投资可能得到的回报之一就是可以大大缩短新系统所要求的培训时间，因为许多用户已经熟悉了应用程序。

2. 联合应用程序开发

联合应用程序开发（JAD）是快速循环原型的一种扩展形式，其中联合开发小组由开发和终端用户小组组成。理想情形是，小组领导应该由两个组以外的人担任，此人对项目的产出没有既得利益，而纯粹是一个协调人和中介者的角色。

联合小组主动接近终端用户小组的需求，并通过开发快速原型提供屏幕显示和报表。这个小组的目标之一是生成需求规范文档。

显然，这种方法要求开发小组和用户小组都必须有大量的投入，但成功的回报应该由双方共享，因此在项目及其成功中将拥有双方既得利益。

3. 快速应用程序开发

快速应用程序开发（RAD）是若干技术的集成，其目标是高质量系统的快速开发。

RAD 的组成部分有：

- 1) 项目规划。
- 2) 需求规划和开发。
- 3) 原型化。
- 4) 可重用代码。

项目规划的概念是使项目使用最少的人员。其思想是项目人员的配备应使用户最大限度地利用开发人员。规划还允许用户根据系统的相互依赖性和开发小组实施规划所投入的时间确定项目的持续时间。相互依赖性可能使项目持续时间比原来预计的要长得多。这些关键步骤可能阻碍进一步的开发直到它们成功地完成为止。

需求规划和开发由终端用户和开发小组共同的成果组成，用于进行结构化分析和应用程序设计。

原型化是 RAD 的中心议题。原型有助于建造最终的系统，因为原型将用作应用程序的基础。

计算机辅助系统工程（CASE）工具主要用在 RAD 中，因为它们能集成现有的可重用模块中的代码。这些工具帮助获取商业规则和功能，根据指定关系将其转化为电子格式，并能够根据听众如终端用户和高级管理人员快速地显示各种格式。

1.1.3 面向对象的分析和设计法

面向对象的分析和设计（OOA 和 OOD）将重点放在对象的开发上而不是对过程的开发上。这个方案是建立在模块化概念的基础上。模块化允许系统设计人员将系统分解成小的更容易管理的单元，并在它们之间定义了完善的接口。

使用模块化的概念，面向对象的方法将信息封装在单独的模块中，以便更容易定义和管理。信息封装有助于实现信息隐藏的概念，这是一个防止使用全局访问信息的技术。这使这些模块（或对象）是自我包含的、可管理、可更新，最重要的是可重用的。

面向对象的方法包含三个概念：

- 1) 对象（Objects）。
- 2) 类（Classes）。
- 3) 继承（Inheritance）。

利用这种方法，对象成为被封装的功能和信息的一般实现。对象就是用户可以对其进行操作和与其交互的程序，它有状态和行为。对象通过消息而不是通过直接访问和调用进行通信。

类是具有共同的行为和共同的功能的对象的集合。用户可以用继承对类进行特殊化处理，这是一种在类之间共享和重用属性的机制。子类通过添加新的或重新定义已存在的特殊和行为来采纳其父类的封装的属性和行为。

这种方法用于收集问题定义和解决的信息。随着更多的面向对象编程和开发环境及工具的出现，看起来对大多数机构的分析和设计来说，这是最有生命力的方案。有关该主题的进一步的探讨，请参看 Grady Booch 编著的《面向对象的分析和设计》一书。

1.2 对象的发现和标识

在过去的数据库系统应用软件的开发应用过程中,我们往往花很大的精力为一个单位开发一个系统。开始使用时无论各方都认为具有一定的水平。但是使用不久发现用面向对象的分析方法失败的迹象显露出来。这倒不是面向对象的分析本身的问题。有经验的系统开发人员会清楚地解释。大多数系统变量的变更都涉及到单位里为了适应形势的发展而制定的策略的变更。事实上,一个系统中根本就不存在策略变更问题的对象。但是策略变更问题是遍布在整个系统之中的。这就是我们的应用系统软件在使用过程中常常出现一旦某个地方的策略变更时,整个系统通常都要进行大量的改变的原因。

按照一般的道理,面向对象技术应能够提高软件的开发的质量和生产率,但是面向对象的这种优势只要在正确地标识了对象集合的基础上才能体现出来。

对于一个给定的问题域,一个合适的对象集合能够保证可复用性、提高可扩充性,并能借助面向对象的开发模式,提高软件产品的质量和生产率。没有一个规范的建立对象的方法,软件开发人员只能承受因为胡乱建立对象的风险。不少书里面都介绍了面向对象的技术。但是,对于发现和标识对象的问题,都仅是普通地讲从系统所涉及的组织、边界、责任等一些原则。作为一个软件开发人员,对于这些原则理解很容易。但是对于一个具体的系统而言,如何发现对象,却缺少一个形式化的东西。

为此,我们在这里介绍一些有助于发现对象的实用技术。

1.2.1 动机

作为软件开发人员,我们所关心的是使事物正常运作。因此必须找到切实实用发现对象的技术。虽然目前所有的关于面向对象技术书,都没有超出目前已有的技术范围,都不具有自己特有的模型结构和表示方法。在分析过程的开始阶段建立初始对象集合时,就会发现这些书上所讲的一般方法是很浅的。而错误地选择对象则会对工程的圆满完成造成巨大的影响。

我们要解决这样的问题时,首先要知道事物的属性和服务(功能)。另外我们知道所有传统的系统分析工具的主要功能是定义数据和说明操作,那么,从某种意义上说,传统的分析工具对于发现对象具有非常重要的作用。

我们可以用传统的三种系统分析工具:数据流图、实体—关系图和状态—变迁图。这些工具涉及到了系统的三个不同的独立方面。我们将这些工具应用于软件的分析过程中,称其为三视图(3VM)。

第二个观点是对象的基本特性。我们知道对象与应用问题域的概念是紧紧结合在一起的。为了能够真正找到合适的对象。我们必须清楚地标明和定义问题域中的概念。对于问题域的理解一直是软件系统分析的难点,这是一个非形式化的主观过程。

在软件的系统分析过程中,我们对概念的处理主要是基于书面或口语的自然语言。到目前为止,将一些自然语言规则应用于系统分析方面已有一些成果。我们把这种将语言规则应用到软件系统分析的处理称为基于语言的信息分析(Linguistic_based Information Analysis,简称LIA)。

1.2.2 方法

我们将面向对象分析中发现对象的方法过程用图 1-4 表示。

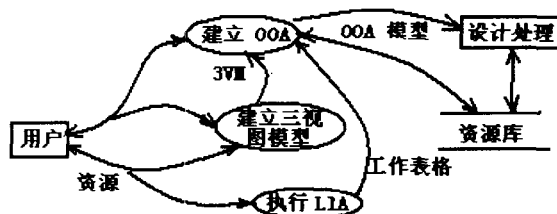


图 1-4 发现对象概要

上图说明了如何用 3VM 和 LIA 应用于发现对象的过程。后面，我们将对这两项技术进行详细介绍。这里我们要指出，3VM 和 LIA 是有别于且独立于面向对象分析的活动。另外，我们从图中可以看出，这些技术的应用是一个不断反复的过程。应用这个方案的目的是在实际应用中降低对象标识的主观性。

1.2.3 三视图模型 (3VM)

关于数据流图、实体—关系图和状态—变迁图在软件分析中的使用已非常普遍。一个系统的不同三视图模型的构造对于发现对象是非常有用的。

1. 实体—关系模型

众所周知，数据流图是 OOA 的一个有力前哨。实体很有可能成为对象，而那些实体的属性则表示成最终要由对象存储的数据。实体间的关系有可能将建立关联对象。而所谓关系的基数和条件性则有可能成为维持这些关系的服务。

尽管实体—关系图是发现对象的有力工具，但是我们也发现在应用时存在一些问题。首先，标识的实体可能与应用的问题域无关，特别是系统分析员企图用第三范式来建立实体时。而且，实体—关系图对于那些不需要存储的数据对象显得非常无力。例如，那些识别事件发生的对象或执行一个控制功能的对象，就是属于这样一类。

2. 数据流模型

数据流模型有两种模型，都是发现对象的有力工具。

首先是上下文图。用它可以确定系统的边界，系统边界从系统分析的角度讲是非常重要的。上下文图所标识的外部实体表示数据流的源头和目的地。因此，外部实体就是候选对象。上下文图的数据流代表了该系统的输入输出。任何一种对象集合都必须说明这些上下文图中的数据流是如何被接受和处理以及生产的。

在系统具有一定规模的情况下，还会产生分层数据流图集合。这个模型表明，将待开发的系统功能分解为一些基本单元。这些基本单元又可以视为是一些详细说明或基本处理说明。而这些基本处理说明最后必须对应于对象的方法和服务。

3. 状态—变迁模型

状态—变迁模型有两种形式，它们都是发现对象的有力工具。

第一种是事件响应模型。它对于发现对象是非常有用的。这个模型标识出了系统必须进行识别所发生的每一件事，以及系统必须作出预期响应的事件。这个模型的事件成分有助于标识一系列的识别事件的对象。而响应成分则有助于标识一系列发生事件的对象。

在一些特别情况下，为系统建立一个或若干个状态—变迁图是很有用的。除了能够标识识别事件的对象和发生事件的对象外。这种状态—变迁图还有助于标识保存状态信息属性。

必须说明，并不是所有的三视图模型对每一个系统都有用。例如一个系统分析员对一个柜员机控制系统建立模型时所用的工具就不同于仓库监控系统。

1.2.4 基于语言的信息分析系统

前面我们介绍的三视图模型尽管是非常有用的。但是，它并不能专门用于指导正确的标识新系统的对象集合。判断、直观感觉和洞察力仍然支配着发现对象的过程。

基于语言的信息分析则在发现对象的方面起着重要的作用。而且，基于语言的信息分析也同样有助于标识对象的成分。因此，它和三视图模型有一定的重叠。

基于语言的信息分析的目的是标识出问题域的概念空间及其这些概念的关系。我们主要介绍短语频率分析（Phrase Frequency Analysis，即 PFA）和矩阵分析（Matrix Analysis 即 MA）。

这两种技术都需要建立资源仓库。资源仓库包括相关的档案、模型、软件、人员以及问题域和系统知识的其他资源。如果问题域具有参考材料，那么这些参考材料必须包含在资源库中。

资源库中可能还包含一些信息：

访问记录；

形式或非形式的规格说明书；

已有的或相关系统的用户手册；

打印格式；

日志。

所有这些资源都包含文本库，而语言的信息分析技术可应用于该文本库。

语言的信息分析技术通常只是应用于资源库的某个子集。这取决于系统分析员希望将什么样的视图用于问题域。通常，与应用有关的资源所产生的结果同与开发系统的规格说明有关的资源所产生的特定结果会有所不同。

短语频率分析搜索选定的资源文本，以标识可以表示问题域概念的术语。用一个二维表列出对某个问题域进行描述的短语频率分析时，得出描述问题的所有短语结果清单。这种清单的建立基本上是一个客观的过程。但是在审查清单时会发现，许多概念是无关的。我们认为短语频率分析的一个很大优点是能够广泛地标识问题域的概念集合。并对其进行评估，得出哪些是有关的哪些是无关的。

表面上看，短语频率分析类似于建立数据模型的技术，实际上，短语频率分析与它们大不相同。因为名词/动词的标识是非常主观的，它们基于什么是名词/动词以及分析员的