

Delphi 5

企业级解决方案 及应用剖析

【献给梦想成为或正在成为项目主管的Delphi程序员!】

刘 艺 编著

淡菊资讯工作室 审校



软件开发技术丛书

Delphi 5企业级解决方案 及应用剖析

刘 艺 编著

淡菊资讯工作室 审校

本书附盘可从本馆主页 <http://lib.szu.edu.cn/>
上由“馆藏检索”该书详细信息后下载，
也可到视听部复制



机械工业出版社
China Machine Press

本书是一部面向企业级应用的Delphi 5高级编程开发专著。全书以企业级应用体系结构为框架,以Delphi 5最新最强大的功能为线索,结合软件工程理论和项目管理经验,深入探讨了Delphi 5企业级开发的思想、策略及技术要点,并通过人力资源管理系统/电子商务货物配送系统、自动控制、与监测系统这三个典型的企业级解决方案及应用剖析,全面细致地介绍了如何使用Delphi实现真正意义上的企业级应用开发。

全书内容详实、例程丰富、汇集了作者多年实际开发经验和研究心得,是一本真正面向实战应用的Delphi力作,可为Delphi程序员挑战发展空间提供帮助,亦可供中、高级编程人员及项目主管参考。

版权所有,侵权必究。

图书在版编目(CIP)数据

Delphi 5企业级解决方案及应用剖析/刘艺编著. -北京:机械工业出版社,2000.9
(软件开发技术丛书)

ISBN 7-111-08226-5

I. D… II. 刘… III. Delphi 语言 - 程序设计 IV.TP312

中国版本图书馆CIP数据核字(2000)第44613号

机械工业出版社(北京市西城区百万庄大街22号 邮政编码 100037)

责任编辑:瞿静华

北京忠信诚胶印厂印刷·新华书店北京发行所发行

2000年9月第1版第1次印刷

787mm×1092mm 1/16·26.25印张

印数:0 001-6 000册

定价:58.00元(附光盘)

凡购本书,如有倒页、脱页、缺页,由本社发行部调换

前 言

每一次当Delphi的新版本出来时,我的心情总是非常激动(对于Delphi 5的面世也不例外),然后就是去找最新版本的Delphi书,以期获得最新的资料,掌握最新的功能。对于一个钟情于Delphi的老程序员来说,我已经养成了这个习惯。然而,现在的情况却越来越令人担忧,因为我和我的那些Delphi发烧友们发现,时下Delphi的书越出越快、越出越厚、越出越多,但真正有新内容、有新意、有价值的却越来越少。记得以前Delphi 4刚刚发布的时候,街上还没有Delphi 3的书卖,现在Delphi 5的书几乎和Delphi 5同时发售。知道内情的人告诉我这些书大都是根据Delphi 5的测试版所写或者有的干脆用Delphi 4的内容,起Delphi 5的名字。还有朋友甚至告诉我鉴别和选购方法,就是看看这些书中是否对Delphi 5新增的功能和组件做了详细介绍,比如:新增的Tframe组件、对ADO的支持、MIDAS的最新改进等。看来,读者的眼睛是雪亮的,毕竟他们对书的评价才是最公正的。

有时,我很羡慕那些写从入门到精通一类书的作者,因为他们所面对的读者更多的是初学者(特别是英语不好,看不懂帮助的初学者),而不是那些雄心勃勃、眼光挑剔的程序员。对于后者,我始终感到写作的压力会很大。非常感谢机械工业出版社华章公司,是她让我把这种压力变成了动力,因为我们都觉得应该有这样一本献给Delphi程序员的书,他们梦想成为或正在成为软件开发项目主管。

“All dream will do.”你相信吗?我记得自己10年前初学编程的时候,就认定这样一句座右铭,直到现在。

当我完成这部书时,我明显感到一种如释重负的感觉,同时也感到多年的积累一掏而空。尽管我还不能知道这部书是否成功,但我希望这至少是一种有益的尝试,不妥之处,恳请读者批评指正。

在这本书里,我希望能更多地谈论如何使用Delphi来构建(Architecture)而不仅仅是编程(Program)。因为在企业级的应用中,先进的开发思想可能比单纯的开发技巧更重要。Delphi 5作为企业级的开发工具有着自己独特的技术优势,这种优势往往无法在教科书例题式的孤立程序中得以体现,所以我们结合实际开发项目的解决方案和应用实例进行剖析,让读者真正掌握Delphi的真谛。

本书从结构上分为4篇,第一篇介绍使用Delphi 5构建企业级应用,这部分概要性地介绍了Delphi 5的企业级开发原理、方法、策略及技术要点,是全书的灵魂和纲要。

第二篇是人力资源管理系统解决方案及应用剖析。这部分结合Delphi 5人力资源管理系统的应用实例的剖析,详细介绍了Delphi开发企业管理信息系统的技巧和方法,以便读者全面掌握Delphi的强大功能。本实例侧重点是基于C/S结构的数据库应用开发和Web功能的开发。

第三篇是电子商务物流配送系统解决方案及应用剖析。这部分通过结合电子商务物流配送系统的实例剖析,从软件工程的角度讨论Delphi 5用于开发规模可变的分布式企业应用的技巧和实现方法。本实例侧重点是面向对象的编程和分布式多层结构系统的开发。

第四篇是控制与监测系统解决方案及应用剖析。本实例侧重点是实时控制和串口通信编程

技术，该技术广泛用于需要与硬件打交道的行业，如通信、电力、监控、自动化、科研等诸多领域。

参加本书编写的还有李祥(第四篇)、段立、尹亚兰。在本书的编写过程中，我参考了红千的部分程序，在此表示感谢！另外，我还得到淡菊资讯工作室的支持，这是一家非常优秀的专业IT技术服务机构，与他们合作令人愉快。

最后，还要感谢我的爱妻，她使我能够专心写书和编程而无后顾之忧。

刘 艺

2000年6月26日

E-mail:newdream@jlonline.com

本书约定

术语：有关新术语和缩略语的解释。

注意：解释一些有趣或易忽略的地方，对有关概念和技术进行补充说明。

提示：提示信息，用于帮助读者解决实际应用中的问题，包含了一些有用的经验和窍门。

忠告：提醒读者应该尽量避免的操作和尝试，否则可能导致无法预料的结果或错误。

参见：用于帮助读者在本书中查找相关主题。

第一篇 使用Delphi 5构建企业级应用

- ◆ Delphi 5企业级应用体系结构
 - ◆ 企业级应用开发方略
 - ◆ Delphi 5企业级应用技术要点
-

第1章 Delphi 5企业级应用体系结构

企业级应用是指那些为商业组织、大型企业而创建的解决方案及应用程序。这些大型的企业级应用具有用户数多、数据量大、事务密集等特点，往往需要由专门的开发小组来构建和实现。另外，作为企业级应用，其不但要有强大的功能，还要能够满足未来业务需求的变化，易于升级和维护。

一个好的企业级应用体系结构，通常来自于优秀的解决方案，同时自应用程序设计的开始就要考虑其体系结构的合理性、灵活性、健壮性，从而既可满足企业级应用的复杂需求也能为今后系统的调整和升级留有余地。这样处理实际上是延长整个应用的生命周期，增强了用户在多变的商业社会中的适应性，减少了系统维护的开销和难度，从而给用户带来最大的利益。

在不久以前，一个大型应用程序还需要一台真正的大型计算机。现在，一个大型企业级应用程序将能够在许多相对便宜的基于Windows的计算机上运行。这主要是受益于分布式的企业级应用体系结构，这种体系结构分散了处理负载，并提供了许多其他好处。这些技术中有一些在过去就已经存在，但是只能在昂贵的大型平台上使用，并且花费巨大。现在Delphi 5使我们也能在低费用、价格适中的计算机平台上使用这些技术，实现低成本的企业级应用。

一个理想的企业级应用系统平台应该具有如下特征：

- 获取、部署和维护都很经济。
- 非常可靠的健壮性。
- 自动从失败中恢复，并几乎不丢失信息。
- 能够处理大量的数据。
- 能够同时支持数百个用户。
- 强大的安全特征。
- 数据的高可用性。
- 可以迅速地开发和部署新应用程序。
- 支持现有的应用程序。
- 简化组件重用。
- 直观的编程模型，以便可以迅速而经济地培训开发人员。
- 行业标准协议和编程接口。
- 对于小、中、大型系统都是适当的并且是可承受的。
- 系统费用随系统规模的增长而缓慢增长。
- 能够继续发展以满足无法预见的需求。

虽然少数优秀的软件天才可以开发出一个可以满足这些需求的系统，但对于大多数软件开发人员而言，这仍然是一个极难实现的梦想。而Delphi 5的出现可以极大地降低开发这种系统的复杂性和减少开发的时间。尽管其困难仍然存在，但是Delphi 5所提供的最新技术和实现手段完全能够实现这样一个目标：为基于组件的应用程序提供强健的规模可变的基础框架并简化这种应用程序的开发。

1.1 什么是企业级应用体系结构

企业级应用体系结构是指用于大型企业及商业组织的应用软件的分布式实现的体系结构。它的核心是一个分布式、规模可变的应用系统。首先，作为大型企业及商业组织在地理位置上通常是分布的。这种分布形式，小可以是在企业内部部门之间的近距离分布，大可以是跨国分支机构在全球范围内的远距离分布；其次是业务的分布，大型企业及商业组织的业务繁多，虽然有些业务需要集中处理，但多数业务都是相对独立的，业务部门和分支机构的分布也是造成业务分布的主要原因；最后要指出的是，规模可变的应用程序必须是分布的，成为对象的软件组件必须能够在网络上任何地方的任何计算机上工作，数据必须可分布到多个物理数据库服务器上，而且处理也必须分布到应用程序的各层上。

在网络上，分布式系统对于用户来说应该是透明的，即使是在不同的地理位置，企业内部的各业务部门和分支机构也应该能无障碍地实行分工协作，共同完成目标任务。

企业级应用体系结构通常划分为3层：表现层、业务层和数据层，如图1-1-1所示。表现层用于和用户交互，它提供用户界面及操作导航服务；业务层用于业务处理，提供商业逻辑等各种约束；数据层用于数据的集成存储，这些数据既可以是平面文件，也可以是RDBMS管理的数据。

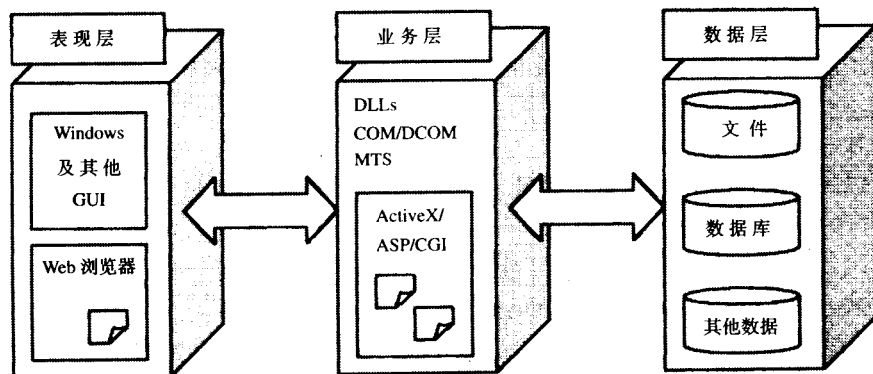


图1-1-1 企业级应用体系结构通常划分为3层

术语 RDBMS是关系型数据库管理系统（Relational Database Management System），是用来存储和管理数据库的引擎。著名的RDBMS有Oracle、SQL Server等。

下面我们对表现层、业务层和数据层及其解决方案进行简单的讨论。

1.1.1 表现层

表现层是应用程序与人进行交互的层次。这一层是必要的，因为人们不能有效地理解硬盘上的数码形式的信息，而在这一层上的数据是我们理解的。现在，人们可以通过键盘、显示器和鼠标来进行人机交互。以后会有更好的设备——语音和手势输入设备来满足更高的交互需求。

现在表现层的焦点是提供对任何时间、任何地点和任何方法的访问的支持。用Internet集成，如浏览器、拨号交互连接和网络技术设备，可以实现这种支持。这一层的工作是挑战性的，因为客户机类型是动态变化的。在企业级解决方案中，客户机不仅仅是基于Windows的，还可以考虑UNIX、Mac和其他操作系统。

要支持不同的平台，表现层的级别必须从胖客户机到瘦客户机。不要认为瘦客户机卸载

的字节就少,因为它也可能是包含有百万字节的图片。实际上,通过不同的参数,可以在客户机端调整其执行的功能。

1. 基于网页的解决方案

基于网页的客户机连接到网络上,并下载它所需要的内容。它不会下载整个应用程序,而是应用程序的网页。

一个瘦客户机提出的是表现层的静态信息。用户可以填满一些文本,设置某些复选框,提交给过程。这种功能类型与主机上的表单批处理相似,唯一的区别是终端可以在Internet上任何地点。一种典型类型是Web浏览器,静态的Web浏览可以保证应用程序有最多的支持,这种应用程序称为以Web浏览器为中心的应用程序。这种情况下,运行环境由客户机决定。

如果需要更多的功能,则需要一个动态的网络应用程序。这时可以使用更高级的网络用户界面,包括动态网页,这将使应用程序的交互性更好。不过,这需要依赖于不同的Web服务器和浏览器的扩展技术支持,这一部分也是不同的技术提供商分歧较大的地方。例如:用户开发的动态网页可能只在Internet Explorer上有效,而在Netscape Navigator中却一筹莫展。这样虽然不能得到更通用的应用程序,但是如果可以知道用户在使用哪一种Web浏览器,就可以利用该Internet技术供应商的技术优势解决问题。

一般情况下,基于网页的解决方案非常适用于基于Web浏览器的企业内部互联网和外部互联网应用程序。它的易于维护、使用方便、跨系统跨平台等诸多优势已经越来越受到人们的追捧,成为一种很有竞争力的解决方案。

2. 基于网络的EXE解决方案

EXE的解决方案需要用到客户机端的一些功能,它依赖于客户机的操作系统。这时应用程序是可执行的,可以安装在客户机上。

依赖于网络的应用程序混合了底层操作系统和网络。由于要处理的任务复杂,这些应用程序需要操作系统提供的功能。这种情况下,Web浏览器提供的内容有限,难以应付复杂的需求。不过,同基于Web浏览器的方案一样,基于网络的EXE解决方案也可以连接在Internet上。例如有一个数据存储应用程序,通过检索数据仓库,一般会生成复杂的数据视图,这些视图数据可以图形化并生成报表以供决策。因为图形化及生成报表的处理十分复杂,Web浏览器不易实现,而利用操作系统的图形和打印支持功能则可以顺利实现。但是又必须连接到网络上,因为无法把数据仓库全部卸载到客户机上再进行处理。通过网络请求到的数据仅仅是在数据仓库服务器上已经处理好的一个结果集,而客户机不需要处理能力非常强大。唯一的限制是客户端应用程序,即EXE文件必须运行在同一个操作系统平台上。

基于网络的EXE解决方案在传统的局域网中比较常用,是一种技术成熟、可靠性好的解决方案。

3. 基于单机的EXE解决方案

最后一种类型是只使用客户机,而且不需知道网络的存在。此应用程序可以是单用户的,不过,这种类型的应用程序越来越少,因为大多数客户希望交换信息。这类客户应用程序包括像WPS 2000那样的商业软件以及专门开发的单机版应用软件。这些应用程序无需网络,但也可以用于合作。

在这种类型的基础上还发展了新的混合型客户应用程序。这种混合型应用程序在需要时

可以联网,可以使用Internet,但用户几乎注意不到这种区别。有些杀毒软件就是一个很好的例子。这些应用程序可以独立地安装在客户机上。当想更新病毒特征代码或更新软件版本时,就必须上网,下载需要的数据。整个过程中,除了连接Internet,用户不必输入URL或网络命令,一切都由应用程序自动进行。也就是说,混合客户型使Internet的使用可以不留痕迹。这种混合型客户应用程序在对网络依赖性不是很强、数据量不是很大、时效性要求不高的情况下,可以说是一种比较经济实用、有生命力的解决方案

提示 以上不同的解决方案使用了不同技术,为了使读者理解不同的客户类型,下面简介这些解决方案涉及到的核心技术:

- **HTML**——Hypertext Mark-Up Language,网络的用户界面语言。一般只说HTML时,指的是3.2版本或4.0版本。它们都支持基于表的处理。这种技术的应用最广。
- **DHTML**——Dynamic HTML,更高级的浏览器用户界面。DHTML与HTML的不同在于支持成熟的对象模型。用户界面的每一部分都可以动态显示、删除或添加,这使用户界面功能相当强大,不过,只有较少的浏览器支持这种性能。
- **脚本**——在DHTML对象模型中,定义了一个脚本界面。似乎不能把脚本当成一种技术。不过,用脚本可以控制任何类型的客户。例如,Microsoft Excel可以用脚本完成某些任务。脚本适合于所有用户,脚本的主要好处在于它提供了应用程序定制化,而不需要复杂的编程技巧。
- **组件**——没有组件,DHTML、脚本和客户机上的大部分任务都不能完成。
- **Win32**——Win32 API提供了一种与操作系统交互的方法,这种技术适合于需要特殊图像或高性能的客户。Win32 API是专门面向Windows的,与HTML和DHTML不是相同的编译模型。

1.1.2 业务层

业务层也叫事务逻辑层或中间层,是应用程序的脉搏。它负责在分布式系统的中间层处理数据,那里有事务处理规则和业务流程约束数据的处理。这一层主要用于大批量处理、事务支持、大型配置、信息传送和网络通信。由此可以看出这一中间层很复杂。

在分布式系统中,可以把复杂业务关系细分为多项功能单一的服务,每项服务都执行一项特殊任务。这些服务可以用相对独立的服务组件来实现其功能。通过分布这些组件,可以平衡数据处理负载、协调业务逻辑关系、调整业务规模和业务规则。

这一切在技术上都是可以实现的,例如:COM+负责基本的COM和事务处理功能。IIS负责Internet服务,如SMTP(Simple Mail Transfer Protocol),FTP(File Transfer Protocol)和HTTP(Hyper-Text Transfer Protocol)。每项服务都可以享用其他服务和创建新服务。

术语 COM+——COM的下一代是COM+。通过添加服务,扩展了COM,使其可以用于企业中。而COM只提供了组件框架。要使COM组件有事务处理能力,还需要附加的服务。这是COM+的目的。COM+现已在微软的Windows 2000中使用。

业务层是很重要的。它包含了目前提供特殊服务的数目最大的组件对象。这种灵活性是大型的企业应用程序所需要的,它可以根据表现层的用户请求,从数据层获取、处理并返回数据,以响应用户需求。

1.1.3 数据层

数据层实际上就是资源管理层。与业务层相比,没有或较少有数据的处理。而是定义了大量数据的管理任务。数据库和资源会变得越来越,因此,这项任务也变得越来越困难。

通常数据层使用大型的RDBMS来管理,如Oracle。使用RDBMS来管理数据的好处是可以协助数据的处理,提高数据的使用效率。

RDBMS处理数据除了完成数据库本身的插入新数据、修改数据、执行系统函数等SQL操作外,还可以通过存储过程和触发器来自动地在数据库上存储应用程序的商务逻辑和函数模块,这样在数据层上就可以对数据做出相关的功能强大的处理,而不是仅仅把请求结果返回给客户机。不过在数据层是否应该有太多的数据处理,一直存在着分歧。主要有以下两种对立的观点:

第一种观点认为,在数据层利用RDBMS处理数据的功能越多越好。他们相信存储过程和触发器应能在数据库的插入与更新中提供一切,以保持数据的时效性。赞成者的观点是,如果业务规则变化,所有要做的仅仅是在数据库服务器上更改存储过程,而不必在客户机上进行修改。即将业务层向数据层靠拢。反对者的观点是,这种解决方案等于把所有的应用程序的业务逻辑放进数据库,会使数据库本身的速度减慢。许多人正在从这种解决方案转向下一种。

第二种观点认为,所有由实际应用程序来处理的功能都应放在数据层之外,并在业务层或表现层进行处理。认为应该放入数据层的东西仅仅是数据本身以及它们的更新、插入和删除等基本操作。任何业务逻辑,比如数据有效性的校验,都应放在业务层或放在实际应用程序的表现层。赞成这种解决方案的是那些对速度要求高,而且其数据库仅仅是为了存储数据的人。反对者的观点是,如果所有的业务逻辑都存储在前端的应用程序中,任何业务逻辑的变化都将导致程序的重写、重编译以及重新分发。另一个主要问题是,当所有的应用程序逻辑都存储在数据库本身时,很难把数据库从一个平台移植到另一个。通常通过使用包含所有业务逻辑的业务层应用程序来解决这个问题。

UDA(Universal Data Format)策略与RDBMS的有所不同。UDA存储并处理数据,涉及的数据类型包括关系型和非关系型数据库、电子邮件与文件系统、Office文档、文本和图片以及客户事务对象。这种方法实际上是创建一个可以处理所有数据类型的存储仓库,数据通过blob存储。

但blob只把数据作为二进制保存,不保证相互关系。例如,若blob正在存储其他文档的参考信息,不会发现非法的引用。在数据层,这种非法的引用是没有意义的。不过,在业务层,这可能会导致一个或多个处理程序的中断。

为了让每条资源处理自己的数据格式,每条资源必须依附于一个普通的对象模型。这样就可以以不确定的格式管理、复制和查询数据。

UDA对象模型称作OLE DB对象模型,这是一个低级格式,其中规定了对处理程序的定义。OLE DB是高速的、低级的。对OLE DB的写操作需要一些数据搜集工作。

OLE DB对象模型的优势在于它是不受限的,必须执行一个特定的子集。可以扩展对象模型,调整对数据的访问。这像表现层一样,要用范围的延伸换取较多的功能。

因为OLE DB的低级性,Delphi、VB或脚本一般不能直接使用它。创建了一种称为

ADO(Active Data Objects)的简化模型。这种对象模型很小,但很有用,允许对大部分OLE DB性能的访问。所以ADO要处理不同的数据,不必太大。

后面会讨论在Delphi 5中使用ADO的方法。

上面讨论的三层框架并不是指任何实际的物理结构;一个三层应用程序可能是由安装在两个计算机上的若干个组件构成的。这种三层体系结构是一种模型框架,是考虑应用程序服务的一种方式。传统的两层客户机/服务器方法通常包括一个数据库组件和一个桌面组件,而业务逻辑被武断地分在两层之间。将所有这三种类型的服务组合成一个单独的组件,就产生出一个整体的应用程序。

这种企业级应用体系结构特别适合于分布式应用程序的开发,它具有以下几个优点:

- 将复杂的业务逻辑封装起来。
- 独立于程序设计语言。
- 可重用组件。
- 减少项目风险。

实际上,在一个组件中实现一个服务使得业务需求改变时的维护得到了简化。我们可以将代码的改变限制在一个单独的组件内。这样只要某个服务的最初接口不改变,就可以添加新功能或改变现在功能的实现而不打乱其他组件。

如果使用的是COM组件,因为COM是一个二进制标准,它和具体语言无关,那么只要该组件实现了提出的服务要求,我们用Delphi开发的组件就可以和别人用Visual Basic、Visual C++或Symantec C++开发的组件共同构建企业级应用。

基于服务的概念而开发出的组件可以提高组件的可重用性,特别是在业务服务处理层,这种优势显而易见。将公共业务逻辑作为企业对象从用户界面和数据层中分离出来,使服务组件更具有一般性并且更适合于重用是企业级应用体系结构思想的精华所在。大量的研究表明当应用程序的大小增加时,程序的复杂性会呈指数形式增长。通过细化企业对象,减小每个组件的复杂性,那么就能获得更有效的质量保证。组件重用也减小了项目风险。

所以企业级应用体系结构的层次关系在表现层、业务层和数据层的三层基础上还可以细化。比如增加防火墙作为安全层。即使在每一层的内部,仍然需要细分,以体现分布式结构的优越性。

分布式结构的优越性还体现在应用程序规模可变上,即在硬件上可以将组件和服务进行分布,以平衡网络及服务器的负载。在软件上可以适应企业业务需求的变化及增长,通过修改和增加相关层的业务组件和服务来适应变化,而不必更新整个应用程序。

这种分布式的、规模可变的企业级应用体系结构是随着计算机技术不断发展,特别是网络应用的发展而逐步成熟起来的。

1.2 企业级应用体系构建思想的发展

计算机刚刚发明的时候,没有人想到它会有如此广泛的应用,甚至有人预言全球计算机的需求不超过5台。无法否认的是企业级的应用在某种程度上推动了计算机的发展。早在二十世纪五六十年代,大型计算机就成为大多数美国商业机构、企业的重要组成部分。个人计算机的出现加速了企业的信息化进程,企业级应用开发的思想也日趋成熟。图1-2-1是对计算机在商业、企业应用领域发展的简单回顾。

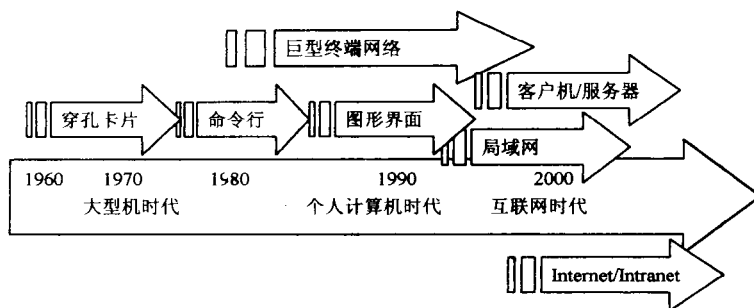


图1-2-1 对计算机在商业、企业应用领域的发展

早期的企业级应用开发依赖于单一的大型机，这意味着对系统资源的不充分使用。执行不相关任务的用户被迫共享相同的系统资源。而且一旦出现故障可能导致所有任务的终止。在达到大型机处理能力极限的时候，通常不得不升级到成本更高和更大型的机器，所以规模可变能力也是一个问题。

现在，客户机/服务器模式已经看作是建立企业级应用程序的一个比较合理的解决方案。客户机/服务器计算的主要优点是它的分布式本质，它对系统资源的充分利用，和与旧的大型机环境相比较便宜的配置成本。客户机/服务器计算为建立企业的基本结构而提供了一个灵活的体系结构。通过将处理要求分解到几个小型服务器机器上运行，就有可能在整体上更好地使用资源和系统的处理能力。另外，客户也不仅仅是不会说话的终端，它们可以处理许多进程和客户请求，例如运行字处理软件，而不需要服务器完成它们的任务。

然而，客户机/服务器计算机有其自身的陷阱。在许多情况下，在企业内部多服务器机器的配置经常导致管理上的恶梦。它们按照单独的独立机器而配置和使用，每一台都可以独立工作，而不是将服务器配置看作是一个单独的逻辑部件。而且，客户机/服务器环境在用户部分更加复杂。寻找某个特定资源的用户不仅仅是要知道这些资源存放在何处，而且需要知道如何连接和使用它。客户机/服务器环境除了它的灵活性和强大功能之外，也是一个难以学习和使用的环境。因此，客户机/服务器模式不是一个十全十美的解决方案，有些应用系统可能更适合于大型机的环境。

对于企业级应用的开发人员，开发思想从大型机到客户机/服务器软件开发的跳跃同样是困难的。在大型机世界中，应用程序在大型机上开发、编译和运行。客户通过终端执行应用程序。这种开发实质上 and 开发一个单机上的应用程序没有两样。受这种思想和模式的影响，有些企业级应用的开发人员虽然也转而开发一些运行在分布式网络上的应用程序，但往往不是真正的，而只是把服务器作为一个存放数据文件的文件服务器而已。典型的例子是在Windows NT服务器上放一个FoxPro数据库，并将存放该数据库的目录共享，于是通过客户机运行这样一个和单机上没有太大差别的桌面数据库应用程序。这样的伪客户机/服务器软件使得系统效率低下，还可能引发网络阻塞。图1-2-2是大型机/终端模式，图1-2-3是伪客户机/服务器模式(实际上是文件服务器模式)，两图表示了这种应用的开发思想。

随着客户机/服务器环境的流行，软件开发过程变得越来越复杂和困难。现在处理过程被分解到客户机和服务器上进行，软件开发人员必须决定将函数模块和功能组件放置在那里。应用程序开发过程已经不再仅仅是编写代码，开发人员不得不理解应用程序分割、用户接口设计和网络协议的概念。决定将功能函数放置在哪里的过程已经和编写实现它的代码同样重

要，决定必须最充分地使用客户和服务器的处理能力。

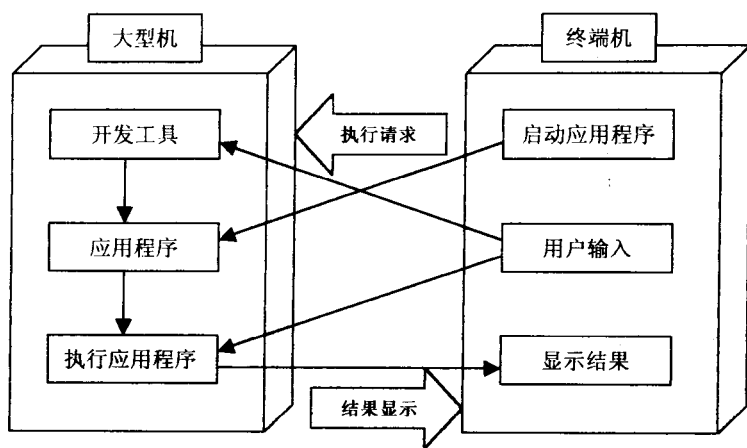


图1-2-2 大型机/终端模式

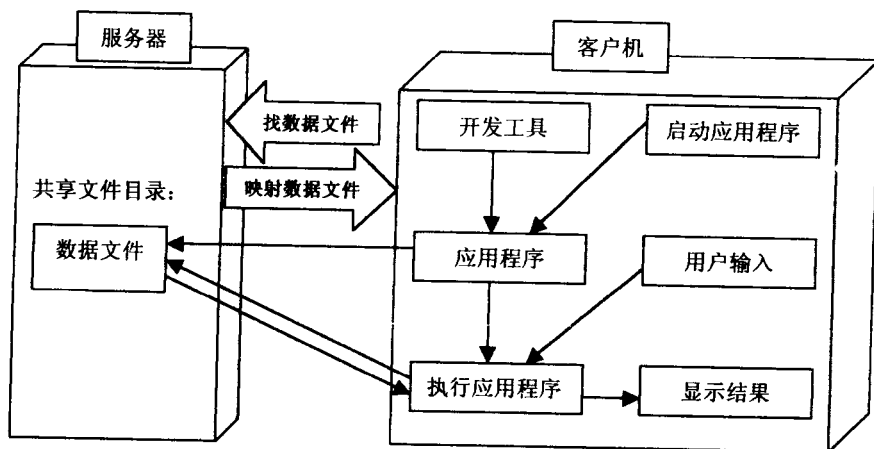


图1-2-3 文件服务器模式

例如，在客户机/服务器环境中，开发人员面对的一个重要决定是将数据库数据处理逻辑放置在哪里。它是应该用存储过程、触发器实现，还是用客户端SQL查询实现。正确的决定通常是在应用程序的需要和设计需求的基础上做出的。

客户机/服务器环境提供的强大灵活性和功能弥补了建立客户机/服务器应用程序的复杂性和困难程度，图1-2-4表示了这种应用开发思想，它与大型机/终端模式有很大不同。

典型的客户机/服务器开发项目包括建立客户用户接口和业务逻辑以及建立和编制数据库服务器的工作。按这样的思想创建分布式数据库应用程序的过程模式是传统的客户机/服务器模式，也叫做两层客户机/服务器模式。

客户机/服务器计算的最新进展已经将业务逻辑从用户接口中分离出来，称为3层客户机/服务器体系结构或多层客户机/服务器体系结构，如图1-2-5所示。业务逻辑部件是一些模块，它们独立于客户和服务端，并运行在它们自己的机器上，提供数据库和客户之间的通信机制，完成与业务规则相关的数据处理。这使客户机不再处理业务逻辑，并且也使客户端开发人员

不再在程序中编写数据库特定的逻辑规则。

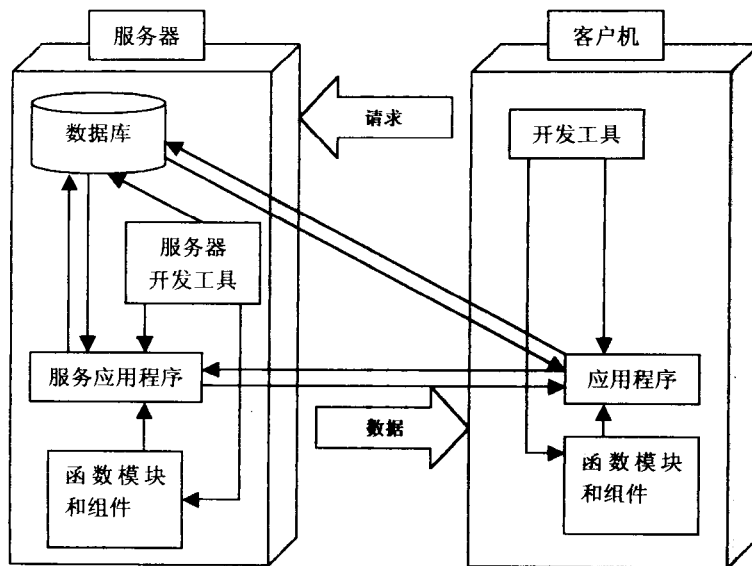


图1-2-4 传统客户机/服务器模式

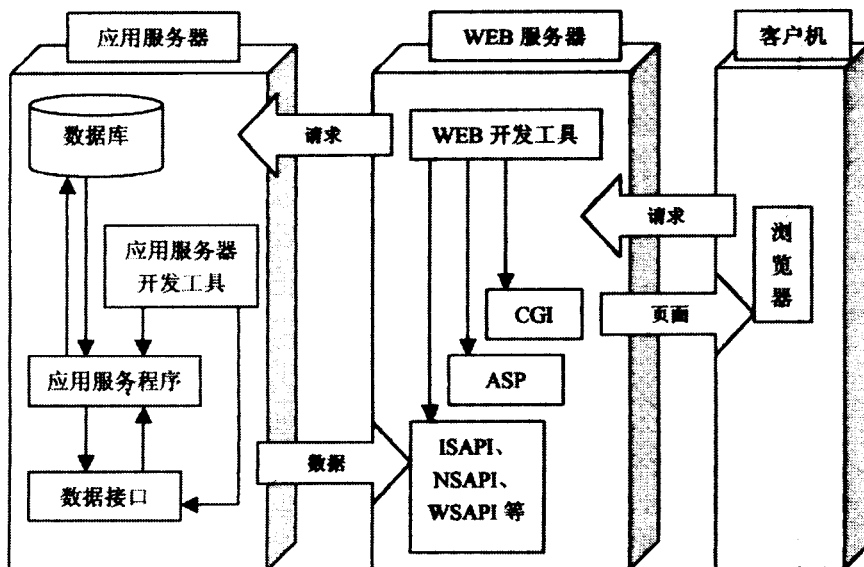


图1-2-5 3层客户机/服务器模式

3层结构的客户机/服务器体系向开发人员提供了更大的开发应用程序的灵活性。客户端开发人员仅仅需要注意和业务逻辑部件的接口，而不需要担心数据库末端的功能函数以及如何查询数据库。他们可以确保自己会得到恰当的数据，以及可以处理数据格式。

业务逻辑部件处理正式提出的适当的数据库查询，以作为对客户请求的响应，并和数据库对话。对系统的每一个单独部分都可以很容易地修改，而不用改变系统中的任何其他部分。业务逻辑部件的主要优点是它们一旦开发和配置，那么就可以用客户机器从整个企业中访

问。

3层体系结构的开发思想是企业级应用体系结构的基础。它很容易扩展为多层的结构体系。最一般的例子是使用基于组件的软件开发机制,以将应用程序的业务逻辑部件细化分解为多个部分,它们可以分布在整个企业中,并提供非常特殊的功能。使用多层结构的结果使得客户端程序可以做得很“瘦”,但它又不是一个大型机时代的终端。

通过将组件分布在企业内部的机器上,多层体系结构可以更好地使用可用的处理能力。另外,对应用程序内的某个组件的升级和增强可以不用重新在每台客户机器上安装整个应用程序。无论组件驻留在网络上的何处,开发人员都可以替换组件的某个实例。那么使用组件的全部客户机器立即可以使用升级了的功能。

随着Internet/Intranet技术的不断发展和电子商务的广泛使用,浏览器/Web服务器体系结构模型作为一种新的开发思想将客户机/服务器概念带到了一个新层次。浏览器/Web服务器仅仅是客户机/服务器计算的另一种形式,其最基本的形式是,使用Web浏览器作为客户,使用Web服务器作为后端。然而,在应用程序开发、配置和易用性方面,浏览器/Web服务器体系提供了比客户机/服务器计算更多的优点。

浏览器/Web服务器的开发思想以及它的网络连接体系结构和传统网络体系结构相比,具有以下主要优点:

1) 较低廉的开放的非专用的标准。

不少企业正在认识到该体系结构的成本效益,而Internet/Intranet的核心技术是公开的、基于标准的和广泛使用的,几个主要环境的成本相对较低。建立Web服务器的服务软件可以从多个供应商那里得到,例如Microsoft和Netscape。客户工作站使用价格低廉和容易使用的Web浏览器软件,以连接到Web服务器上。建立企业内部网即Intranet所需的网络基本结构在大多数企业内已存在。另外,随着Windows中的集成Internet浏览器的出现,这个模型更易于被Windows用户所接受。

2) 可以集中维护和管理资源及应用程序。

浏览器/Web服务器的出现实现了一个交叉体系结构,它将客户机/服务器处理模型以及那些基于主机的处理模型的最好特性聚集到一起。这一体系结构的目标是容易配置、信息的集中管理和简单的资源管理。在这个开发思想中,服务器负责向用户提供信息和要求的数据。另外,通过客户浏览器软件,服务器是呈现给用户的图形化用户界面的关键。客户工作站(典型情况下为台式PC)使用Web浏览器软件来显示由服务器发送的信息,服务器控制信息的布局和内容。这使信息的控制和管理非常可靠,原因是它们是集中管理的。

特别需要指出的是,基于浏览器/Web服务器的应用程序可以集中维护和管理,使得对维护浏览器/Web服务器应用程序的成本比分布式客户机/服务器应用程序的维护和管理成本要低得多。在服务器端对基于Web应用程序的改动可以立即对全部用户产生作用,而不用单独改动每台客户机器,较好地解决了企业级应用维护难的矛盾。

换句话说,这种体系结构是这样—个处理模型,它吸取了客户机/服务器开发思想的精华,并将其和传统的基于主机的由大型机和小型机使用的体系结构的最好特性结合起来。

3) 跨系统、跨平台能力。

最后,浏览器/Web服务器体系的一个重要优点是它可以在一个公共的相互配合的网络中使用各种不同的系统。许多企业花费了大量的金钱和时间,企图将完全不同的不兼容的系统

连接到一个无缝的网络中,结果并不能完全让人满意。浏览器/Web服务器体系可以适应不同的系统,例如PC、Mac和基于UNIX的工作站,使得它们可以共存。

浏览器/Web服务器的开发思想最重要和最显著的优点是创建跨平台应用程序的能力,避免了创建和分发应用程序的多个版本所带来的成本。在客户端使用的Web浏览器,将Web服务器作为浏览器和应用程序服务器,或数据库服务器之间的接口的应用程序,在后端产生HTML输出,它可以在支持Web浏览器的任何平台上由用户查看。

通过以上的分析讨论,我们可以感觉到企业级应用的开发思想正趋向于一种基于Internet /Intranet的跨平台的分布式计算模式。它是在客户机/服务器模式上发展起来的,并且还混合着传统的客户机/服务器模式。实际上,正是这种混合结构的开发思想吸收了不同体系结构的优点,从不同方面满足了企业级应用的需要。图1-2-6所示的就是这样一个混合结构体系。

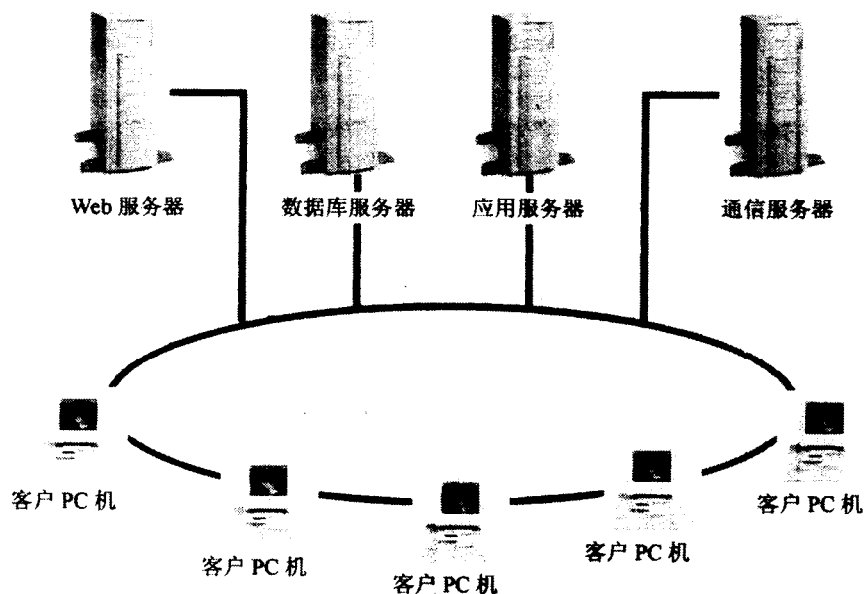


图1-2-6 混合式结构体系

1.3 选择Delphi 5作为企业级开发的利器

在选择企业级应用的开发工具时,会有很多需要考虑的问题。它们涉及的主要方面是:

- 开发工具本身的性能。
- 稳定和可靠。
- 可视化开发。
- 面向对象技术。
- 提供4GL。
- 开放性。
- 团队开发。
- 访问数据库的能力。