

软件体系结构在信息系统建设中的应用初探

郭宁 (首都经济贸易大学 信息学院 100026)

摘要: 本文简要阐述了软件体系结构的技术特点, 分析了采用新技术开发系统不成功的原因与目前组件产品的市场现状, 研究了新技术与信息系统开发的关系, 在此基础上提出开发企业信息系统应采取的对策。

关键词: 软件体系结构 信息系统 多层结构

Primary study to applications
of software system architecture
in construction of
information systems

当前软件厂商推出的各种软件技术产品五花八门, 令人眩目, 学术界提出的概念越来越“虚”, 不深入研究会觉得无所适从, “我们到底应如何开发信息系统?” 要回答这个问题, 首先就要对软件体系结构等问题有一个正确的认识。

1 软件体系结构概述

软件行业工业化的趋势导致了软件体系结构技术的产生。能够像硬件系统那样, 将部分软件组合起来构建软件系统, 一直是软件行业多年来追求的目标。软件体系结构包括系统总体组织和全局控制, 其应用软件主体为相对独立的板块式结构。在规模庞大、功能复杂、需求不断变化的信息系统开发中, 软件体系结构的设计是整个开发过程中的关键一步。选择设计合理的系统体系结构, 才能使开发的软件系统经得起变化, 满足数据一致性、可移植性、可维护性、可重用性、可适应性、易管理性、可伸缩性、便于扩充并容易修改等方面的要求。

一个完整的应用包括用户界面层、商业中间层的商业逻辑组件、数据层数据库系统。界面层提供给用户一个视觉上的界面, 通过界面层, 用户输入数据、获取数据。逻辑层是界面层和数据层的桥梁, 它响应界面层的用户请求, 执行任务并从数据层抓取数据, 然后将必要的数据传送给界面层。数据层完成数据层定义、数据完整性、安全性功能, 它响应逻辑层的请求, 访问数据库。单层结构是指将界面层、逻辑层、数据层合并在一起的系统。双层结构是将界面层和逻辑层合为一层, 数据层是另一层或将逻辑层和数据层合为一层, 界面层是另一层, 通常称为客户/服务器结构。多层结构则将这几层分离处理, 是将应用程序分为: 客户端应用程序、应用程序服务器和远端数据库服务器。其中, 客户端主要负责用户界面的处理; 服务器端主要负责商业逻辑的处理, 为客户提供公共的数据服务, 处理客户端与数据库间的数据流; 远端数据库服务器提供数据库数据的存取与维护。目前流行的主要有两种应用软件体系结构, 各应用体系模型是根据在用户与数据之间所具有的层次来划分的, 每一层次一般都运行在不同的系统或是相同系统的不同进程空间内。两种软件体系结构的特点是:

1.1 两层体系结构

(1) 允许多用户同时存取相同的数据, 来自一个用户的数据更新可以立即被连接到服务器上的所有用户访问, 这为许多小规模商业应用带来简便和灵活性。

(2) 服务器将消耗部分系统资源用于处理与客户端的连接工作, 当客户端的数目增加时, 服务器端的负载会逐渐加大, 直到系统承受不了众多的客户请求而崩溃, 数据存取受到限制, 难以扩展到大企业广域网或国际互联网上; 也难以管理客户端的机群。

(3) 系统的可伸缩性较差和安装维护困难。数据库服务器成为系统可靠性的极大隐患。如果数据库服务器因为某种原因停止工作, 那么整个系统将趋于瘫痪。

(4) 由于商业规则的处理逻辑和用户界面程序交织在一起, 商业规则的任何改动都将是费钱、费时、费力的。客户端应用程序的安装工作与每次对客户端程序的修改和升级工作十分烦琐。

(5) 在存储过程调用中, 即所有处理过程都在数据库层进行, 只是将最终结果返回到客户端。这种结构的业务逻辑需采用专用语言开发, 很难再移植到其他系统上去。

1.2 多层体系结构

多层分布式结构技术的特点商业规则或处理逻辑被从客户端独立出来, 把用户端的业务逻辑独立出来, 使之与数据库服务器中存储过程合并在一起, 构成应用层, 以提高计算能力, 实现灵活性。运行在一个介于用户界面和数据存储的单独的系统之上。应用系统的每一个部分都可以被单独修改而不会影响到另外两个部分。此外, 因为每一层之间是通过接口来相互通信的, 所以只要接口保持不变, 内部程序的变化就不会影响到系统的应用其余部分。这种结构的特点是:

(1) 可以将大型的、复杂的工程项目分解成简单安全的多个子模块, 并分派给不同的开发小组。当需求变化时只需要在服务器上进行修改或配置组件, 并且当应用程序的用户、数据、业务量增加时可以重新部署。

(2) 逻辑流程更为清晰。由于在多层结构中, 业务逻辑与数据操作及用户界面分离, 这使得在着手进行系统设计时, 能够更集中地对业务流程进行分析, 而不必关心业务中具体步骤的细节技术解决。这一点, 使得整个设计过程变得更有效, 针对性更强, 也更符合系统设计的目的。

(3) 功能代码复用性强。由于业务逻辑的分离, 对于业务中一些通用性的功能模块, 通过分层设计, 可以使得模块代码独立出来。尤其是很多业务逻辑的数据访问都是由一组或几组相同的数据操作组合而成, 通过结合面向对象分析方法, 能够使得系统中的代码重用性更强, 开发强度大大降低, 而系统质量大大提高。

(4) 分布式部署。在多层结构系统设计中, 可以将不同的业务分布在系统的不同主机上, 也可以将面向不同客户的同一业务分布在不同主机上。这样, 对于整个系统的硬件体系来说, 利用率得到充分提高, 也使得当业务量增加时, 能够很容易地通过提升局部业务的支持性能来满足变化。

(5) 数据存储方案。由于数据层与业务层的剥离, 客户只会通过业务层与数据层进行操作, 这使得数据层与客户隔离, 减少了数据库遭受错误访问或非法攻击的可能性, 增强了系统的安全性。另外, 系统管理简单, 可支持异种数据库, 有很高的可用性。

(6) 缺点是客户端计算功能的严重浪费。今天的PC机已具有较强的运算能力以及低廉的价格, 真正好的应用模式应该对PC运算能力加以充分利用。另外对服务器端的商业应用

程序要求很高, 往往是微小的失误在多用户访问情况下将很快导致服务器系统的崩溃。能满足要求的服务器价格比较贵。

2 现状分析

从技术角度上讲, 采用多层次结构开发的软件系统应远远好于两层或单层结构的系统, 但在实际应用中我们却发现: 虽然采用了最先进的技术和产品, 在投入了大量人力、物力、资金后, 其效果并不理想, 开发出的信息系统并没有达到预想的结果。存在着效率不高、结构复杂、系统难于维护, 而且开发与管理费用并没有降低, 质量和可靠性并未显著提高等问题。究其原因我们认为主要是:

2.1 用户需求复杂多样

随着企业间竞争的加速, 使得对快速开发具有高适应性应用系统的需求日益迫切。企业越来越多地寄希望建立一个真正开放的信息系统。信息系统不仅仅能够满足集企业内部生产、管理于一体的需要, 而且要建立基于internet/intranet/extranet技术, 关键信息能充分共享, 通过互联网进行营销实现电子商务功能的具有分散和联合处理能力; 其应用软件是完全流程化的、可控和规范的, 而且是大量应用多媒体技术的多功能系统。并能对需求变化能够快速做出反应的系统。

2.2 市场产品缺乏, 标准不统一

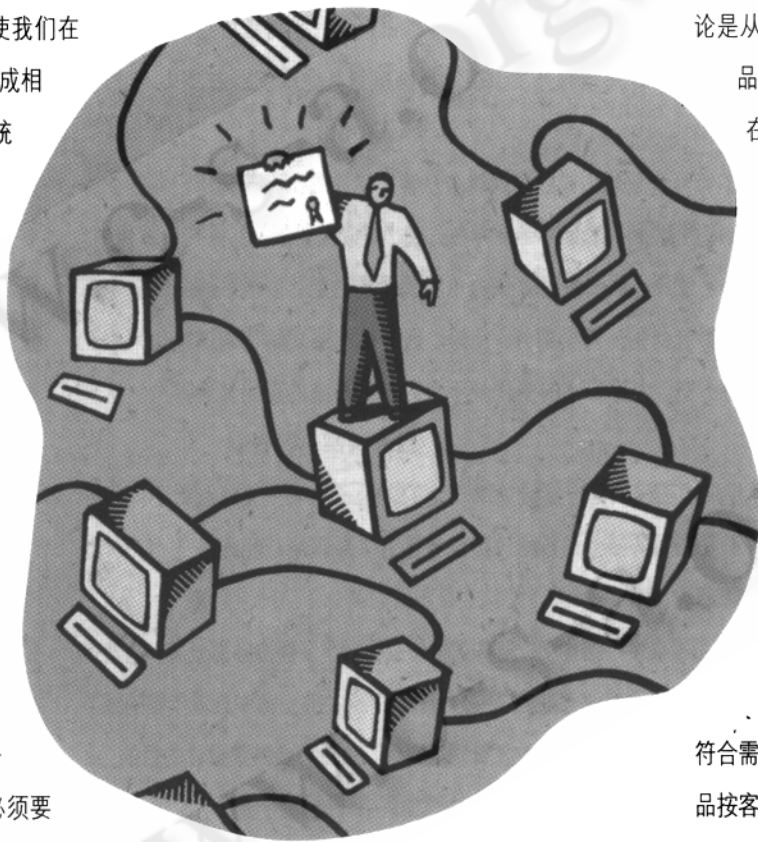
目前采用软件思想开发的

组件产品相继出台，从表面上看组件为开发者提供了一个高层应用环境，将分布式系统中各种不同的计算机硬件与软件系统屏蔽起来，从而使开发的应用具有良好的可扩展性、易管理性、高可用性和可移植性。但采用其思想开发的产品推出的时间还不足五年，不很成熟，离要集成应用软件所需要的构件还有很大距离。常常是不能拿来就用，那么如何提取领域构件，国际上还没有一种可循的办法。使我们在开发应用软件的同时，难以形成相应的领域构件，以便适应系统自身演化或可复用到同领域中的不同系统的开发中。现在业界开发服务器端应用主要依赖于第三方厂商所提供的商用组件，尽管利用它可以将开发周期缩短 50% 到 60%，但可供选择的余地太狭窄，实际应用领域中的可复用领域专用构件缺乏，市场还远未成熟起来。有些软构件，如查询统计构件、报表生成构件，特别是一些与领域有关的专用构件，必须要我们自己开发。

市场上目前主流的组件大致有 5 种：数据存取组件、消息组件、交易组件、事务处理组件对象组件和应用服务器。组件产品往往并不仅仅是一种软件产品，它更多倡导的是一种计算模型与标准，在组件提供的一个良好的网络分布应用开发平台上，开发的适于各种领域应用的软件产生。

3 对软件体系结构形成研究热点，但理论还不成熟

尽管软件体系结构对系统开发的重要性已被业界普遍认同，但在理论上还有许多问题需要研究。人们还缺少将结构化的思想和方法充分体现在系统级的设计实践中的形式化的工具，例如，在表示接口和部件的结合关系等方面无法具体的、显示的进行描述；在软件结构的同一层中缺少对“交互”的定义等，因此系统的复用性、可移植性会受到影响。另



外，在研究部件及其互联的标准化方面存在着 CORBA、DCOM/COM、J2EE 等标准，而这些标准之间还存在着差异。

2.4 人员素质不能满足需要

软件技术变动大，新技术不断出现。针对用户的各种需求和各种系统的不同特点选择设计合理的软件体系结构，需要掌握的知识很多，既要了解各种体系结构的特点与适用范围，又

要熟悉市场上各种产品的功能与特性，并能在较短时间、较低的费用、较好的应用等方面找到结合点，这些都需要技术人员的素质较高，而目前具有相关经验的技术人员比较少。由于涉及许多新技术与新观念，大多数公司对使用该技术开发系统还没有积累丰富的经验，因此该技术远未得到充分利用，其优势还远没有得到体现。虽然人们对软件构件寄予了很大期望，但无论是从理论上、实践经验还是市场产品等方面，离人们的需要都还存在着差异，需要一个过程。

3 应采取的对策

以上这些说明企业信息系统向多层体系结构应用系统跨越已经成为一种必然趋势，但信息技术仅仅是一种工具。为了取得良好的效果，建议采取以下对策：

3.1 建立开发规范和质量保证体系

软件产品是人利用技术做出符合需求的产品，是人的主观活动和产品按客观规律建造的相符过程，符合程度差异决定了产品质量。软件质量除了满足需求的功能、性能以外，还有人类产品的一般要求：安全、寿命、抗干扰、可靠、易用、易维护等。构建一个大型应用软件是一项非常复杂的任务，任何企业的信息系统项目实施都会面临许多问题，许多因素都可能导致建设的结果不理想，即使在信息技术十分发达的美国，也有大量信息系统失败的例子。实践证明，对于大型系统的开发按照软件工程开发过程与规范要求开发，是保证系统质

量的关键。应在开发前就确立使用软构件为前提的开发方法论、目标管理以及维护机制。没有规范会使得系统过程混乱,质量无法得到保证,技术虽然重要但过程管理同样重要。

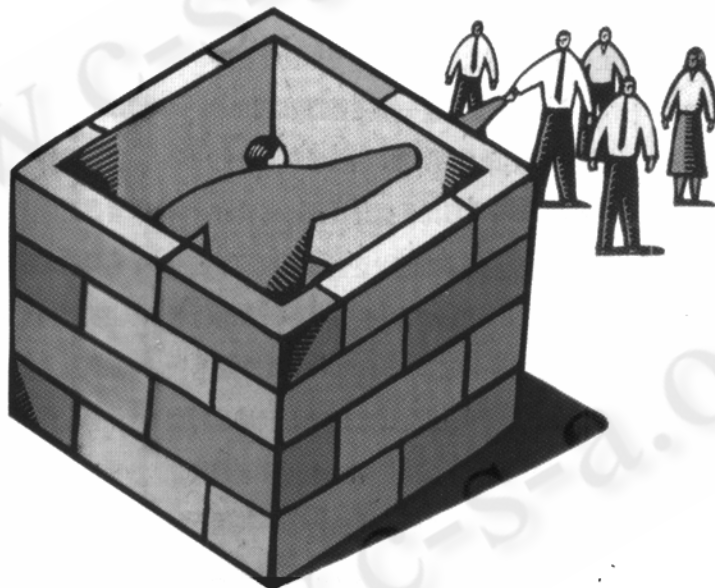
3.2 采取适合的软件体系结构是一个应用系统成功重要因素

多层应用体系结构模型有许多的优点,特别适合于Internet应用程序,但并不是所有的应用程序都应该采取多层模型。开发好多层应用系统关键是清晰、合理地划分好界面层、逻辑层和数据层,并使其独立,可以使系统构成变得简单。一个应用程序采取何种体系结构模型应根据实际情况而定。选择软件技术时,应考虑到解决问题的特点,选择适当的技术,不应一味地为使用新技术而使用,更不能脱离实际地盲目照搬。技术是重要的,但更重要的是对问题的洞察力和经验。如果需求比较稳定、投资规模有限或系统功能并不复杂,就没有必要选择多层结构来达建系统。

3.3 开发平台和组件的选择

系统平台实际上是信息系统的公共支撑环境,它的性能决定着系统的水平,包括互联性、可靠性、可复用性、可移植性、开发性和先进性等。如果系统平台选择不当,尤其是在平台的开放性、先进性等方面选择失误,例如,操作系统水平过低、网络系统开发性不好等,造成的后果将是灾难性的。可能会使系统不能升级,甚至造成整个系统推

倒重来,因此选择一个什么平台是一个非常重要的问题。应用平台本身需要系统平台的支撑,它不能脱离系统平台而独立运行,因此说应用平台是一个承上启下的中间技术层。近几年随着Java技术的发展,越来越多的公司使用Java来开发。由于Java是跨越各应用平台的,而且Java技术本身所具有的特性,它在开发网络环境下的分布式多层结构应用软件时,有着不可比拟的优势。因此,从



发展的角度看,将开发平台定位在Java上是一项明智的选择。对于大型的实用系统来说,所有的软件都自行开发并不是最好的选择。利用现成的、质量好、可靠性强的软构件,按照大规模软件开发的工程规范进行开发,是满足这些大型系统要求的良好途径。一些通用的服务或处理可以由商品化的组件软件完成。

3.4 使用面向对象技术

在开发系统时,使用面向对象的技术十分重要。逻辑层(中间层、应用服

务器)主要封装各类应用的数据请求及处理SQL,是系统设计的关键和难点,划分好客户界面层、中间层和数据层各自所应完成的任务关系到系统的整体性能及伸缩性和维护方面。尽量形成企业对象的方法,减少网络的数据传输。由于实际应用的情况比较复杂,组件产品只能提供一些通用的服务,大量的开发工作还是要由开发人员根据实际需要完成。在使用面向对象技术时,特别要注意使用对象组件技术。现在有许多支持对象组件技术的开发方法,从跨平台和发展的角度看,采用J2EE或CORBA比较好。

4 结束语

软件体系结构的提出是对传统系统开发过程的一次变革,它代表了新一代软件技术的发展方向,极大地满足了多个应用领域的要求。没有无体系结构的软件系统。只有从完备的软件体系结构去开发软件才能经得起改变,而变更、体系结构、过程都是软件系统的固有属性。开发好多层应用系统关键是清晰、合理地划分好界面层、逻辑层和数据层并使其独立,可以使系统构成变的简单,如果把多层和面向对象的系统分析与设计结合起来,进行系统开发,那么开发出来的应用系统就具有很强的适应性。

参考文献

- 1 周之英,《现代软件工程》清华大学出版社,2000。 ■