面向合作方协同设计的 PDM 架构①

龚茜茹[1], 蔡丽霞[2]

[1](河南工业职业技术学院 计算机工程系, 南阳 473009)
[2](河南工业职业技术学院 网络管理中心, 南阳 473009)

摘 要: 在企业业务模式由垂直集成向水平集成转换的趋势下, 对企业原有的 PDM 系统进行升级, 设计完成了合作方协同设计管理平台, 构建了基于 Web Services 的系统架构来适应业务模式的改变, 详细叙述了该系统的高层架构设计及服务的搭建.

关键词: 协同设计; Web Services; PDM; 架构设计

Architecture of PDM Based on Partner-Collaborative-Design Oriented

GONG Qian-Ru[1], CAI Li-Xia[2]

[1](Department of Computer Engineering, Henan Polytechnic Institute, Nanyang 473009, China)
[2](Network Management Center, Henan Polytechnic Institute, Nanyang 473009, China)

Abstract: To cater to the trend of enterprise business model converting from vertical integration to horizontal integration, the management platform based on partner-collaborative-design was designed to upgrade the original PDM system, and the Web Services-based system architecture was constructed to adapt for the change of the business model. The high-level architecture and service structure of the system were described in detail.

Key words: collaborative design; Web Services; PDM; architecture design

1 项目背景

本项目来源于与某 “机械设计研究所”的合作项目. 该所历史上在设计管理模式上采用传统的层次化垂直结构. 但是近年来, 随着用户对产品更新换代的要求越来越快、质量要求越来越高, 在竞争日益剧烈、外部压力日益增大的形势下, 该所在管理模型上重新定位, 打破长久以来形成的垂直结构, 形成一种趋向于水平集成的业务模型, 这形成了企业重构的趋势[1], 使企业能更专注于自己的业务特长, 在产品研发时, 能更好地利用国内更先进的技术力量, 以实现合作方异地协同设计[2].

该所已经成功地把 PDM(Product Data Management, 产品数据管理)系统用到本地产品设计管理中, 将产品整个设计生命周期内的所有数据, 按一定模式加以定义、组织和管理, 使产品数据在整个生命周期内保持一致和共享, 为企业设计和生产构筑一个并行产品开发和管理的环境[3].

但是, 现有 PDM 系统是针对当初封闭的管理模式而设计的, 无法应对设计变更比较频繁的异地的合作方协调设计环境. 因此, 该所要求把原来的系统进行扩展, 在原有系统上增加合作方协同设计能力, 搭建基于互联网的合作方协同设计沟通管理平台, 使得部件设计合作方能够在早期就介入产品的研发过程, 及时获取产品信息和变更通知, 并将相关的信息及时反馈到企业, 缩短主要设计部门和合作方的沟通时间[4], 提高合作方在新产品设计中的响应能力, 实现各方共赢.

新的 PDM 系统的开放性, 将为实现产品的异地、异构设计提供强大支持. 通过合理利用 Web Services 技术, 实现业务与分布式数据源整合, 并根据业务发展的需要, 实现业务方法的重新编排, 可以有效管理各合作方协同设计过程. 通过实现工程中设计、制造、测试、维护等职能的综合考虑, 使新产品开发更加有序和有效.

① 收稿时间:2012-06-06;收到修改稿时间:2012-11-19

2 PDM架构设计

2.1 架构设计需要解决的问题

根据对多个方案的比较分析，本项目采用基于Web Services 的面向服务架构的形式，它有如下优点:

① 有利于协调不同的服务领域间的异构数据模型[5]

本 PDM 系统的合作方协同设计是一些特定领域相关的服务集合，在这个服务领域中需要所有服务采用统一的数据模型进行定义. 但是，由于合作方业务的复杂性，数据服务来自不同服务领域，这就使得模型间语义与结构存在巨大差异. 采用 Web Services 技术，将有利于协调不同的服务领域间的异构数据模型.

② 便于实现面向服务的集成(SOI)

通过使用 Web Services 来解决集成与互操作的问题[6]将有利于达到如下目的:

a. 改进现有数据模型，以适应当前集成项目.

b. 根据服务契约对传统系统进行包装，创建当前系统集成所需要的新服务[7].

c. 定义用于 “进行不同数据模型的映射”的数据转换，以便数据能够跨越不同的数据领域边界[8].

d. 为Web服务平台配置执行环境，以支持并实施企业级的服务质量[9].

项目要求与合作协同设计有关的业务，通过瘦GUI Web 客户端或者浏览器实现人机交互. 在设计的初始方案中有四个关键问题需要解决: 第一，服务如何识别与搭建. 第二，如何处理以协同设计为特征的业务模型工作流. 第三，在PDM处理工艺图纸的时候，由于文件体积庞大，需要重点解决文件存放结构与调用方法的问题. 第四，基于 Web Services 的面向服务架构如何实现，实现中需要处理哪些问题.

2.2 顶层架构设计

顶层设计的一个基本策略，就是为每个服务子系统赋予清晰的职责，这些职责是内聚的，相互间的关系松散而简单. 在业务需要服务变化的时候，采取了只增加服务而不是修改服务的策略(开放和封闭的原则). 这样一来，就使系统具有很强的可伸缩、可扩展性，以及对于业务变化的敏捷适应能力.

在项目之初，通过对与项目目标相关的业务进行梳理，依据完整性、自治性、重用性、封装性、松耦合性以及接口的一致性对服务进行了识别. 为了避免点对点服务带来的问题(不能改变信息、难以跟踪服务使用状况、难以验证、安全隐患等)，整个服务使用了企业服务总线(Enterprise Service Bus, ESB)和数据总线(Data Bus)进行集中管理. 根据合作方协同设计的要求，从业务层面来看，本项目共分成三个完全自治的子服务，其架构关系如图 1 所示.

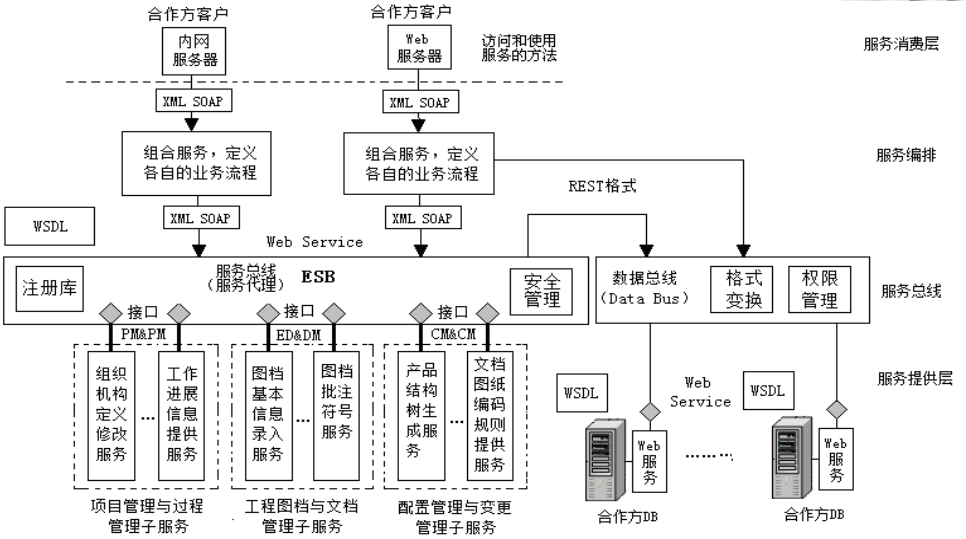

图 1 合作方协同设计顶层架构

下面对主要的服务对象加以说明.

① 项目管理与过程管理子服务(PM&PM)

主要解决在合作方协同设计的情况下，确保有效项目管理的信息服务，主要包括: 组织机构定义和修改服务; 人员角色指派服务; 产品数据操作权限分派服务; 项目开发过程定义服务; 工作流定义服务; 任

务列表生成服务；任务执行状况信息服务；协同开发项目过程监控服务；工作进展信息提供服务等.

② 工程图档与文档管理子服务(ED&DM).

这是PDM系统传统业务的一部分，关注点是工程图档的管理，主要包括：图档基本信息录入服务；图档基本信息编辑服务；图档文件间连接关系服务；图档文件的批量入库和交互入库服务；图档文件释放及传送服务；图档文件显示服务；图档显示模块转换格式服务；图档批注及符号服务等.

③ 配置管理与变更管理子服务(CM&CM).

主要解决工程文件的版本控制以及变更管理的问题，还必须保证在并行情况下图档修改的串行化，主要包括：产品结构树生成服务；图档版本演化可视化服务；产品批次结构信息服务；产品零部件之间的层次关系服务；产品变更管理服务；产品修改串行化服务；文档或图纸编码规则服务等.

这三个子服务相互支持，共同完成合作方协同设计中项目管理、图档管理、版本管理以及图档修改串行化的业务要求. 为了减少子服务之间的耦合并增加子服务的内聚度，项目设计要求各子服务之间不得直接交互，它们只能通过组合服务的上层组件进行交互，从而减少了开发、集成、调试、维护以及后期升级的难度.

2.3 数据资源的可伸缩性设计

① 用数据服务取代远程数据调用

在数据资源的应用上，由于各个合作方数据本身具有敏感性以及数据库和数据结构的多样性，直接通过互联网调用合作方的数据库是不安全，也是不合理的. 因此本设计采用的方法，是使合作方的数据资源以服务的形式提供出来，而且只提供与合作方协同设计相关的数据，而禁止其他不相关的数据被非法使用. 这种数据服务的提供方式采用XML/Web Service技术，就保证了异构数据以统一的格式进行整合.

本系统通过一个数据总线(Data Bus)统一管理数据使用者的权限、格式转换以及其它方面的问题，以确保数据的安全性和使用上的方便性. 由于数据是以服务的形式提供的，数据服务的分布性为大数据量的性能要求提供了支持.

② 数据总线应用的REST风格

为了进一步提高异构数据应用上的方便性，在数据总线的应用上使用了REST风格. 也就是利用URL来表示数据操作，把最基本的四个对数据的原子操作(创建、读取、更新和删除)表示为: GET、POST、PUT和DELETE. 这种针对资源的网络应用设计和开发方式，可以降低开发的复杂性，提高系统的可伸缩性. 也使得资源的调用更加简洁与易于理解. 这样一来，就形成了一种简洁并与HTTP协议的完美结合的表达方式，如图2所示.

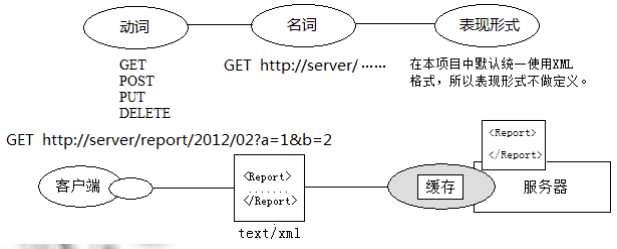

图 2 REST 风格的数据应用表达方式

实践表明，采用 REST 风格的数据应用格式可以获得如下好处:

a. REST 风格天然地具有服务器无状态的特征，在状态迁移的过程中，服务器不需要记录任何 Session，所有的状态都通过 URL 的形式记录在了客户端.

b. REST 风格能够提高系统的可伸缩性，在操作无状态的情况下没有上下文约束. 这样一来，在我们强调要分布式、集群的时候，就不需要考虑上下文的问题.

c. 由于在数据总线中统一采用了 REST 风格，数据应用与具体的数据库 SQL 无关，这样就减轻了大部分程序员的负担，保证了系统后期升级、维护和扩展的可实现性.

在本项目中，只是在数据总线的应用中采用了 REST 风格，在总线的内部设计中还是采用了 XML/Web Service 调用数据的远程服务. 数据总线作为一个中间件，起到了格式转换和统一管理的的作用，如图 3 所示.

2.4 分布式服务解决方案

采用 Web Service 技术尽管能够很好的解决异构系统与数据的整合，但是也带来了性能上的问题[10]. 对于本合作方协同设计系统来说，尽管不是绝对的实时系统，但是对性能也还是有比较高的要求. 在具体设计中，我们采取了服务分层次的分布式布局策略来解决性能问题.

在合作方位置遥远、业务非常复杂而且繁忙时，把所有的服务都集中在中央服务中心是不合适的，这

样会严重影响服务的性能. 而把业务处理全部转移到 GUI 客户端也不合适, 因为业务处理流程必须维护全局状态, 也增加了维护的难度. 本项目解决这个问题的思路是把服务按照不同的级别进行分布式配置, 如图 4 所示.

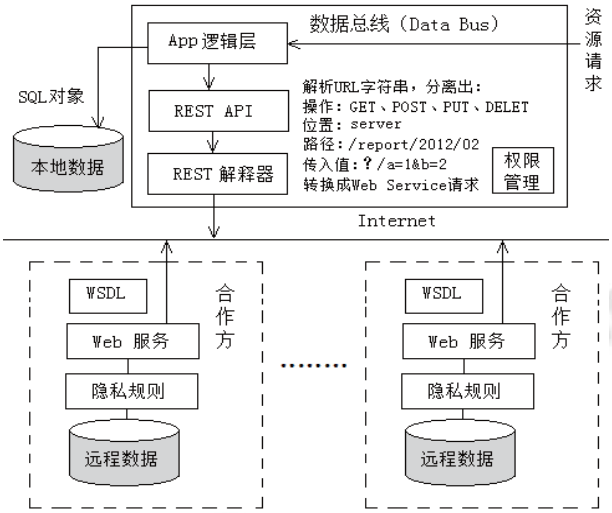

图 3 数据总线的基础架构

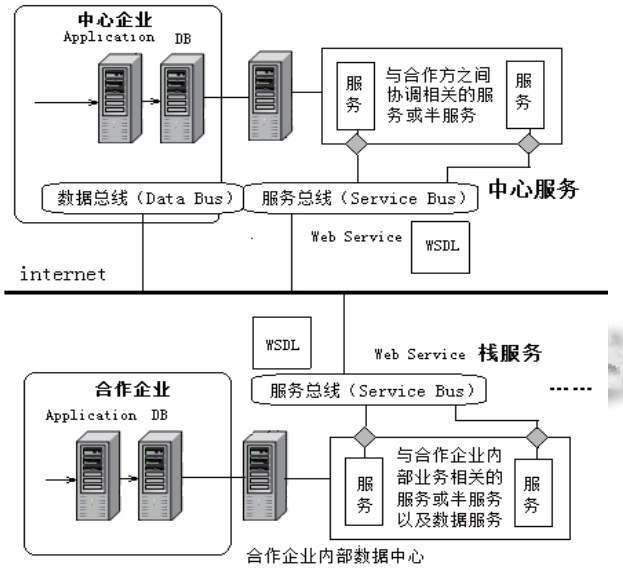

图 4 分布式服务解决方案

在这个方案中, 服务分成两个不同的层次:

a. 中心服务: 这种服务所处的位置在全局数据中心服务器上, 其维护的状态是全局性的, 更多的是关注各个合作方之间的协调的业务. 特别是在响应某个请求的时候需要多个分布式服务站点参与时尤其重要, 中心服务中的服务职责可以居中协调共同完成业务处理. 这种布局的好处在于: 一般来说, 在同一空间(合作方)中会有大量内部相互协调的操作, 而合作方之间的协调在总量上要少得多, 这就可以大幅度减少应用服务的网络流量, 提高系统的整体性能.

b. 栈服务: 这种服务的所谓全局状态是有区域性的. 例如在同一个合作方内, 相当多的业务处理都是在内部进行的, 其全局状态也只是指的这个合作方范围内而言. 我们可以把这类服务(也可能是半服务)在物理上直接置于合作方本身的数据中心, 至于这个合作方数据中心提供服务的方式是采用内网还是外网并不重要(大多数情况下这种处理都是置于内网). 进一步, 某些只与具体的客户相关的业务处理, 也可以置于 GUI 客户端上, 通过这样的处理, 可以极大的提升本地服务的性能, 减轻中央服务器的压力[11].

分布式服务不可避免会带来如何处理会话(Session)的问题. 本设计方案不采用在服务器内存中存放会话的方案, 而是把所有生命周期超过一个任务的的数据都存放在“持久服务”(会话数据库)中. 这种设计尽管在保存和使用会话数据速度方面比内存方式慢了点, 但是由于同一个业务可以在不同物理位置上进行处理, 或者在性能需要的时候可以进行多服务器扩展, 所以这是更合适的方案. 通过在系统中大量使用并行计算的方式, 在总体性能上提升的余地更大.

3 结论

本设计实际应用效果表明: 所设计的合作方协同设计 PDM 系统, 可以很好的适应企业重构趋势下的应用要求, 并且具有比较广阔的未来发展空间. 所采用的分布式服务方案极大地提高了系统的可伸缩性、性能、吞吐量、容错性以及可用性. 可伸缩性的提高是因为服务可以利用服务方已有的分布式硬件, 当合作开始或终止的时候, 只要安装或者关闭这个服务, 也就自然开始或终止了这个业务节点和数据源的使用, 这就使整个系统具有可伸缩型. 性能和吞吐量的提升是因为合理的分布式服务布局, 使网络的使用量被减到最小(绝大部分操作可以在本地内网空间运行而不需要昂贵的互联网络开销). 可用性和容错性的提高是因为某个合作方服务中心发生故障时, 并不会影响其它服务的可用性. 此外, 由于 PDM 体系的相似性, 所有服务都可以由一个通用的设计来支持, 并没有增加设计上的复杂性.

(下转第 9 页)

以上技术措施外, 还应从软件开发过程管理的角度思考. 建立适当的软件开发过程规范以将相关要求和活动整合到组织的软件开发过程中去. 对开发人员实施安全教育以提高安全意识, 实施安全技术培训使之掌握常见的安全风险和规避措施. 同时, 还要确保开发、测试环境的信息安全, 避免敏感测试数据、源代码等信息泄露并给攻击者利用.

4 结语

Web 应用安全存在一些典型的攻击模式和最佳实践, 通过培训使所有开发人员认识风险的本质和预防措施, 有利于开发出安全的应用程序.

在软件开发生命周期的各个阶段系统性地解决问题, 而不是在最后时刻发现和解决问题, 把安全控制纳入日常的软件开发工作, 可以大大降低应用程序出现安全问题的可能性.

本文对 Web 应用安全的现状、常见的安全威胁进行了分析, 给出了在软件开发生命周期各个阶段采取的重点措施和方法, 旨在帮助 Web 应用开发人员了解常见安全漏洞、提高安全意识、掌握方法, 提高自身开发软件的安全性. 在本文研究成果的基础上公司建立了信息系统全生命周期安全管控系列规范, 经过实践证明可以有效提高公司开发 Web 应用程序的安全性, 减少安全漏洞.

参考文献

1 De Keukelaere F, Allan D, Buecker A.利用 IBM Rational AppScan 改进 Web 应用软件开发生命周期的安全性. IBM Redboooks,2010.1−35.
2 Santiago CE, Hondo M. Web 2.0 桌面与移动应用程序安全性设计.IBM developerWorks,2012.1−10.
3 赵静.Web 2.0 应用安全深入解析:企业级 Web 2.0 应用安全解决方案.IBM developerWorks,2009.1−9.
4 OWASP. OWASP Top 1-2010 The Ten Most Critical Web Application Security Risks,2011.1−22.
5 肖国一.Web 应用安全利器:IBM Rational AppScan.IBM developerWorks,2010.1−5.
6 Danahy J.策划安全策略:应询问的三个核心问题.IBM developerWorks,2009.1−3.

(上接第 41 页)

本 PDM 系统应用 Web Services 的重要目的是综合各个合作方的数据, 通过隔离合作方不相关数据, 保证合作方本身内部数据的安全. PDM 系统综合各方数据以后, 将把这些异构数据转换成统一的数据格式(XML)的信息, 便于各方面的应用. 由于采用 Web Services 技术, 各种异构数据和异构平台的整合变得可行, 就为系统下一步的发展, 打下了坚实的基础.

参考文献

1 刘一良.协同设计系统及其关键技术的研究与实现[硕士学位论文].济南:山东师范大学,2007.
2 周亮.基于 PDM 的协同设计系统的研究与开发[硕士学位论文].廊坊:河北工业大学,2005.
3 黄琰.基于 PDM 系统的产品研发过程项目管理.价值工程,2011(11):138−139.
4 张永红,高晓梅.PDM 在机械制造企业技术管理中的应用. 陕西理工学院学报,2011,(4):90−94.
5 王德俊.面向服务的分布式系统动态更新研究[博士学位论文].上海:上海交通大学,2010.
6 裴永胜,杨岩,杨磊.复杂产品集成设计系统构建方法研究. 航天制造技术,2011,(4):38−40.
7 邵建军.网络化协同设计系统的研究与实现[硕士学位论文].长春:东北大学,2006.
8 曾鹏飞,郝永平,邵伟平,等.基于 PDM 协同设计任务建模与调度方法研究.现代制造工程,2007,(9):29−32.
9 崔建伟,徐建军,何鑫,等.基于 PDM 的产品协同设计系统集成.大连交通大学学报,2012(4):93−96.
10 胡学骏,曾凡智.一种基于 XML Web Service 的分布式解决方案.计算机工程,2005,13(7):204−205.
11 陈斌,白晓颖,马博,等.分布式系统可伸缩性研究综述.计算机科学,2011,38(8):17−24.