中小MIS快速原型构建与自动代码生成

周 兵,许 俊,吴亚平
(郑州大学 信息工程学院 高校信息网络重点学科开放实验室,河南 郑州 450001)

摘 要:文中介绍了一种中小规模MIS系统快速开发策略,在对自动代码生成所需的数据库操作模型进行了分析的基础上,提出了一种改进的PetShop分层模型,通过采用文档化需求+生成模板的方式实现了代码生成模块,设计了适应框架目标的类和接口结构。结合河南省工业能源利用管理信息系统项目对这种开发模式的实际应用效果进行了验证。对开发过程中的阶段划分以及各个阶段的开发过程进行了分析和定义,对系统代码生成器生成的代码和最终代码中的代码行数情况进行了统计分析。实际工程应用表明,文中的开发策略能够有效缩短开发周期,同时提高系统代码的质量。

关键词:通用管理信息系统框架;代码生成器;开发过程;设计模式;模版
中图分类号:TP31 **文献标识码**:A **文章编号**:1673-629X(2012)05-0028-04

Rapid Prototype Creating for Small or Medium MIS and Auto Code Generation

ZHOU Bing, XU Jun, WU Ya-ping
(Henan Provincial Key Laboratory on Information Networking, School of Information Engineering, Zhengzhou University, Zhengzhou 450001, China)

Abstract: Introduce a rapid small-scale MIS system development strategy. Propose an improved hierarchical model of PetShop based on the analysis of database operations of automatic code generation. Through using documentation requirements and code template achieve a code generation module. Designed a new framework to meet the target of class and interface structure. Defined and analyzed the stage of the development process and various stages of development, with the statistical analysis of the codes lines which the codes generated by code generator and the final codes. Practical engineering application shows that the strategy can shorten the development cycle effectively and improve the quality of the system codes.

Key words: general MIS framework; code generator; development process; design pattern; layout templates

0 引 言

随着计算机技术、网络技术的普及,越来越多的中小企业、单位、个体在经营过程中采用信息化技术来管理生产经营数据,虽然系统的具体业务千差万别,但是其操作都是针对数据库中的表进行数据的录入、查询、修改以及进行数据的统计分析等。因此,开发出一套针对中小企业的通用管理信息系统框架作为开发的起点、采用代码生成作为辅助手段,同时配合一套规范的开发过程,可以大大缩短开发周期。

现有的一些商用的系统如CodeSmith[1,2]虽然功能很强大,但是操作起来很繁琐,程序员必须熟悉它的调用方法,这无疑增加了开发周期。而国内的一些代码生成器如动软架构很简单,只能开发小型的项目。很多类似的代码生成器[3]在与企业现有资源结合方面存在不少的问题,致使系统开发周期长,手工编写代码多[4],正是为了解决这些问题才促使我们需要为通用管理信息系统框架开发符合框架自身特点的代码生成工具。文中就这一问题提出一种快速开发框架和代码生成技术,可以有效地缩短中小型MIS系统的开发周期。

1 改进的PetShop分层架构

通用管理信息系统框架的代码生成器的功能可以定义为:能够生成支持PetShop架构的针对数据库表[5,6]的完整操作。PetShop是目前比较流行的一种开发方式,采用的是分层结构层层调用的方式来完成数据库的访问,所以系统的性能会产生一定的下降,另外会产生层层修改的问题。所以不适合对时间要求严格的工作。为了降低代码的耦合性,提高代码后期的适

收稿日期:2011-10-18;**修回日期**:2012-01-20
基金项目:国家"863"高技术发展计划项目(2008AA01A315)
作者简介:周 兵(1964-),男,教授,CCF会员,研究方向为多媒体技术、p2p流媒体服务;许 俊(1987-),女,硕士研究生,研究方向为Web应用、电子商务。

应性,文中对数据库操作模式进行了详细的定义,其基本思想是对 PetShop 架构进行简化和有针对性的优化。在分层结构上仍然采用经典的三层架构[7,8],分别是数据访问层(DAL)、业务逻辑层(BLL)和数据表示层(UI)。下面主要介绍代码生成器所需要的数据库操作模式和具体实现方法。

1.1 数据库操作模式

通用管理信息系统框架是一个针对中小型企业管理信息系统的开发框架,目标系统的特点决定了我们并不需要过于复杂的分层结构,在这里对 PetShop 分层架构进行修改,大体上仍然分为数据访问层、业务逻辑层和表示层三层,在针对数据表进行操作的数据库操作模式上,为了使系统具有更好的扩展性,采用基于接口的编程思想。由于信息管理框架作为一个通用框架主要关注与业务逻辑相关的数据操作,所以,在生成功能上主要考虑数据存储和数据业务处理。因此,代码生成的重点问题是业务逻辑层和数据访问层代码的生成。经过精心的设计优化,对系统的逻辑结构划分如图1所示:

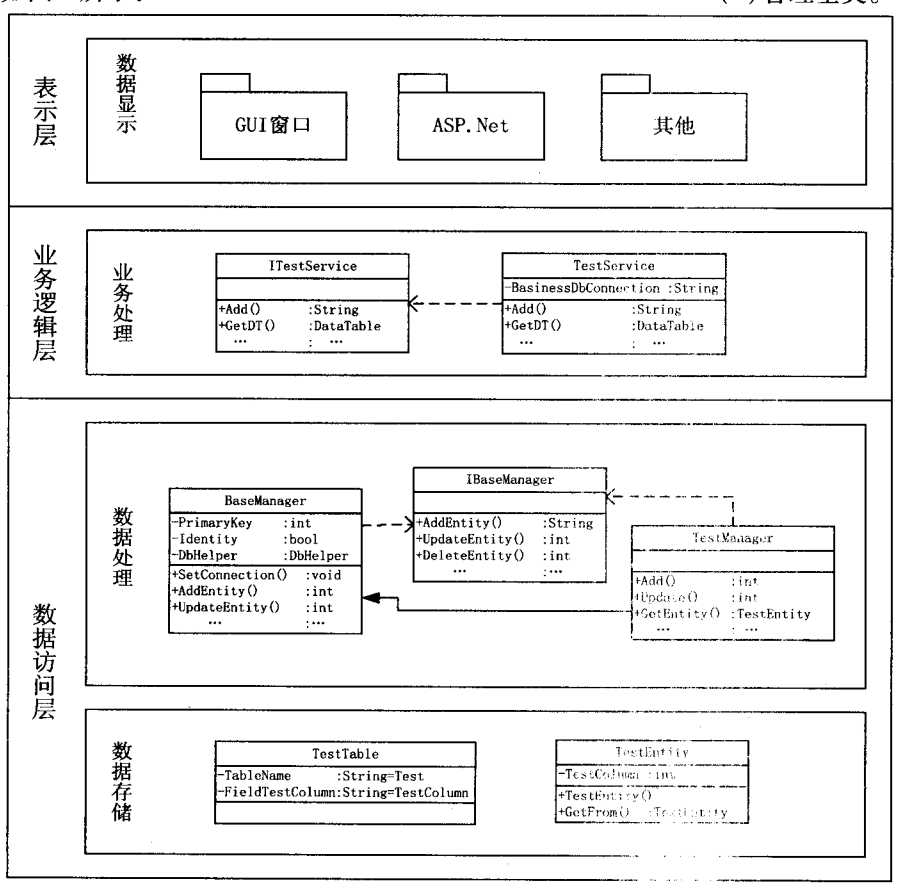

图1　数据库操作系统逻辑结构

1.1.1 数据访问层

为了便于对数据进行处理,在数据访问层上对其在逻辑上分为数据存储和数据处理两层。数据存储层主要是对数据表结构和数据具体内容进行保存,而数据处理层主要是对数据进行相关的加工以及对数据库进行增删改查操作。

1)数据存储层。

数据存储层包括两个类,一个是数据库表结构类,一个是数据实体类。表结构类实现对数据库中原始表结构的定义,包含了表名、字段名、字段类型等相关信息。实体类是程序运行过程中数据表数据的临时存放点,实现对数据的临时存储,其功能包括从各种结构中获得实体类数据的多态方法,用于对实体对象进行初始化。

2)数据处理层。

数据处理包含接口类、管理基类和表管理类三个部分。

(1)接口类。

为了便于系统进行后期扩展,同时为基于反射的编程提供条件,在业务逻辑层的数据处理类上采用了基于接口的编程。IBaseManager 接口定义了数据处理类需要实现的基础方法。这些方法如表1所示。

(2)管理基类。

管理基类作为所有管理类的基类,实现了接口类定义的所有方法,这些方法的实现使得管理类在继承了基类之后,基本上可以不需要进行任何修改就实现对数据库的操作。管理基类方法在实现上采用虚方法来实现,这样结合针对表的管理类就可以完整实现对目标表的操作。

(3)表管理类。

表管理类继承自管理基类,同时也实现了管理接口。在代码生成器生成代码时同时会生成用于补充基类的方法,两者共同协作,完成数据访问层的具体功能。

1.1.2 业务逻辑层

业务逻辑层的主要功能是调用数据访问层来进行具体业务逻辑的开发。由于针对不同的角色或系统会有不一样的业务逻辑,所以首先定义了业务逻辑层的接口。接口定义了需要实现的具体高层功能,如以实体为目标的针对数据库的增删改查操作。同时,为了符合在操作性能上和数据库一致性上的考虑,还需

要实现对数据操作的批量处理。

由于业务逻辑层针对的是各个具体的业务，所以代码生成器生成的代码不能完全满足现实的需要，需要在后期的开发中手工实现对特殊业务的代码编写。也就是说，业务逻辑层中实现的代码有通过代码生成的公共代码和手工编写的特殊业务代码两类。

表 1 数据访问层基础接口类方法

方法	功能
AddEntity、UpdateEntity、DeleteEntity	对数据库实体对象实现增删改
AddBefore、AddAfter、UpdateBefore、UpdateAfter、GetBefore、GetAfter、DeleteBefore、DeleteAfter	实现对事件的支持，在事件执行前、后可触发
Get、GetFrom、GetIds、GetDT、GetProperty、GetId	重载实现各种情况下对数据库的查询
SetProperty	通过参数实现对数据的更新
Exists	判断数据是否存在
Delete	通过条件实现对数据库的删除
BatchSave、BatchDelete	实现数据的批量保存或批量删除

1.2 具体实现方法

针对这个需求，文中通过部分类的方法来实现，也就是说，通过代码生成器生成的代码分为两个，即 Manager. cs 和 Manager. Auto. cs 两个文件对应的是同一个 Manager 类，但是将手工编写的代码放在 Manager. cs 文件中，把代码生成器生成的代码放在 Manager. Auto. cs 中。

这样做的好处是在模板由于业务变化或者存在 Bug 而发生变更时可以自动重新生成所有的公用代码进行覆盖替换，而不会对手工编写的代码产生影响。

2 自动代码生成

代码工具的实现包括代码生成界面、代码生成流程、具体的算法的实现三部分，这里主要讲解代码生成流程和主要的实现算法。

2.1 代码生成流程

代码生成器生成的最终代码是通过对 PowerDesinger[9] 的物理模型进行扫描生成的，其操作步骤如图 2 所示，首先从物理模型中读出数据表列表，然后遍历数据表清单获得每个数据表的字段列表以及字段信息，最后通过代码生成器结合代码生成模板生成对应的类。

2.2 代码生成算法

由于数据库管理系统（DBMS）和开发语言的命名规范有所区别，数据库往往可以支持表名中带空格，而 C#开发语言的命名规范是以字母或下划线开头，字母、数字或下划线组成的字符串。这就导致在代码生成器中需要有一个规范来对表名与类名进行一对一映射。在这里，做法是首先对表名的前后空格去除，然后将表名内部的空格转换为下划线。

数据库表名与类名映射过程描述：

```
GetClassName()
{
string className = tableName ;
if (! string. IsNullOrEmpty(className))
{
className = className. Replace("_"," ");
}
return className;
}
```

代码生成器生成的代码文件在保存时需要根据项目参数对文件名及保存路径进行相应的设置。在此，对文件保存的相关规则定义如表 2：在生成具体代码的过程中，为了精简代码，提高代码生成器的可读性和灵活性，对生成算法中的公共代码进行了进一步的提取封装，由于其生成算法比较简单，对整体生成器操作影响不大，在此不作详述。

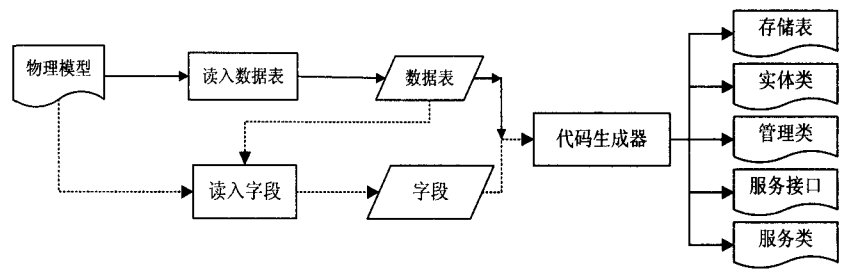

图 2 代码生成流程

2.2.1 表结构类

通过对需求文档的解析，结合生成模板，应该首先生成表结构类。其内容主要包括版权信息、命名空间、注释等相关信息，具体生成过程中引用的相关方法参考如注释的描述：

```
BuilderTable()
{
GenerateCopyright();//生成版权信息，根据配置中的公司信息及项目生成格式版权信息
GenerateUsing();//根据不同类的需求生成对相关命名空间引用
GenerateNamespace("Model", true);//根据 PowerDesinger 中的说明部分生成命名空间
GenerateRemark();//根据 PowerDesinger 中的说明部分生成注释
GenerateClassName();//根据表名生成类名，生成方法：表名+尾缀(Table)
```

GenerateTableNameAndColumnList(tableName);//包含表名和字段名,将数据表中的字段首先设为私有变量,然后再将变量封装为字段(C#中的Field)

}

表2　类保存文件命名规则

类	生成规则
表结构类	工程根路径 + "\Table\" + 表名 + Table.cs
实体类	工程根路径 + "\Entity\" + 表名 + Entity.cs
管理类	工程根路径 + "\DAL\" + 表名 + Manager.Auto.cs
服务接口	工程根路径 + "\IServices\" +"I" + 表名 + Service.cs
服务类	工程根路径 + "\Services\" + 表名 + Service.cs

2.2.2　实体类

实体类的主要内容包括公用信息如版权、注释等。由于实体类需要对数据进行保存,为了实现操作的简便性,同时实现对数据的封装,这里对字段首先定义为私有变量,然后通过C#的Field对其进行引用封装。最后,生成代码中还包括针对不同变量解析到实体类的相关方法,进而在构造函数中对这些解析方法进行调用。

2.2.3　管理器

与上两个类一样,管理器为了保持风格统一,对版权信息、注释也进行了定义。管理器由于事先对实体类的相关操作,所以需要对数据库进行相关的增删改查等基本操作,在此不但需要对实体类进行操作,同时也需要对数据库的主键进行相关处理,例如Sqlserver[10]支持主键的自增,而Oracle的自增则需要通过序列+触发器来实现,由于系统需要实现对多种数据库的支持,故这种情况需在管理器生成时进行处理。管理器生成过程如下:

```
BuilderManager()
{
GenerateCopyright();//生成版权信息
GenerateUsing();//生成对相关命名空间引用
GenerateNamespace("Business");//生成命名空间
GenerateRemark();//根据PowerDesinger中的说明部分生成注释
GenerateClassName();//生成类名
XmlNode xmlNode = GetXmlNode(tableName); //获得表对应的XML节点
GetPrimaryKey(xmlNode);//获得表的主键字段
GenerateClassManager(xmlNode);//生成类实体
SetTableColumns(xmlNode);//创建表格式存放代码
}
```

2.2.4　服务和接口

由于代码生成器生成的服务与服务接口的操作不需要涉及到对表与字段的操作,只需要对数据访问层进行调用,所以,在生成服务与接口时相应的模板设置更加简单,只需要将模板进行简单的关键字替换即可完成。

3　应用实例分析

文中提出的代码生成技术已经应用于河南省工业能源利用信息管理系统,该系统是河南省工业与信息化厅为响应国家节能减排政策而推出的一套企业能源利用情况统计管理系统。其管理对象是河南省能耗在5000吨标准煤以上的工业企业,收集在日常经营过程中对水、电、煤、气、油等能源的使用情况,同时监控能源利用率及循环使用情况[11]。

由于有了通用管理信息系统框架和代码生成工具的支持,在系统开发的各个阶段和过程上,其需要使用的时间与传统开发有一定区别。系统在开发工程中采用了演化原型法进行迭代开发,从开发开始到软件产品化,迭代过程分为规划阶段、开发阶段、稳定化阶段和产品化阶段,每个迭代过程都包括图软件过程中的需求分析、系统设计、程序设计、测试、部署以及系统评价等过程。表3是针对河南省工业能源利用管理信息系统开发过程进行统计得到的一组参考数据。该数据不能说明较复杂系统的使用情况。从开始到产品化大约60天,相对于传统的迭代开发时间上缩短很多[12]。

表3　基于代码生成的开发过程分析

	规划阶段	开发阶段	稳定化阶段	产品化阶段
需求分析	3	2	1	0.2
系统设计	2	1	1	0.2
程序设计	7	14	3	3
测试	2	5	3	2
部署	1	1.1	0.1	0.1
系统评价	4	2	3	1

使用代码统计工具LineCounter对生成代码、UI控制代码和最终代码进行统计,统计结果由表4给出。

表4　代码构成比例对比

	生成代码		UI控制		最终代码	
类别	行数	比例	行数	比例	行数	比例
总行数	38513	100%	10861	100%	54592	100%
代码行数	26413	68.58%	8971	82.60%	33208	60.82%
注释行数	9847	25.57%	1166	10.70%	17519	32.09%
空行数	2280	6.00%	845	7.80%	3865	7.08%

生成代码指的是通过代码生成器自动生成的原始代码,这些代码在最终代码中不存在修改的情况;最终代码是系统发布时包含的所有代码,包括自动生成代码和手工代码;最终代码中的UI控制代码是最终代码

(下转第36页)

从表 3 中可以看出，系统在计算节点数量增加时，计算性能也能随之同步增加，系统整体运行效率处于 88% 到 96% 之间。

3 结束语

由上面的分析可知，运行效果达到了预期的目的，解决了异构 GPU 集群系统的任务调度问题，实现了整个系统的动态负载均衡，提高了整个系统的计算效率，并初步形成了一套支持多优先级任务调度、支持异构 GPU 节点、调度效率高、具有节点容错能力、可扩展能力强的动态调度机制，为将来大规模计算密集型任务的调度提供了解决方案。

参考文献：

[1] Buyya R. 高性能集群计算：结构与系统[M]. 郑纬民，石威，译. 北京：电子工业出版社，2001.

[2] 陈华平，黄刘生，安 虹，等. 并行分布计算中的任务调度及其分类[J]. 计算机科学，2001，28(1)：45-48.

[3] 李丙锋，祝永志，魏榕晖. 异构 Beowulf 系统负载均衡技术的研究与实现[J]. 计算机技术与发展，2008，18(7)：60-65.

[4] 徐 群，祝永志. 集群系统中的负载均衡问题的研究[J]. 计算机技术与发展，2009，19(8)：129-132.

[5] 陈志刚，曾志文. 中间应用服务器动态负载均衡的物理模型[J]. 计算机工程，2001(1)：44-46.

[6] 王 霜，修保新，肖卫东. Web 服务器集群的负载均衡算法研究[J]. 计算机工程与应用，2004(25)：78-80.

[7] Werstein P，Situ H，Huang Zhiyi. Load Balancing in a Cluster Computer[C]//Proceedings of the Seventh International Conference on Parallel and Distributed Computing Applications and Technologies. [s. l.]：[s. n.]，2006.

[8] Chi M，Yu Jung-Lok，Kim Ho-Joong，et al. Improving Performance of a Dynamic Load Balancing System by Using Number of Effective Tasks[C]//IEEE International Conference on Cluster Computing Proceedings 2003. [s. l.]：[s. n.]，2003：436-441.

[9] Tanenbaum A S. Distributed Operating Systems[M]. Englewood Cliffs，New Jersey：Prentice-Hall，1995.

[10] Kunz T. The influence of different workload descriptions on a heuristic load balancing scheme[J]. IEEE Transactions on Software Engineering，1991，17(7)：725-730.

[11] Ferrari D，Zhou S. An empirical investigation of load indices for load balancing applications[M]//Scheduling and Load Balancing in Parallel and Distributed Systems. Los Alamitos，California：IEEE Computer Society Press，1995：487-496.

[12] Devarakonda M V，Iyer R K. Predictability of processs resource usage：a measurement-based study on UNIX[J]. IEEE Transactions on Software Engineering，1989，15(12)：1579-1586.

(上接第 31 页)

中用于界面控制的代码，在这里是指 Aspx 后台代码，这部分代码主要是通过 IDE 工具自动生成的。可以看到，在代码量上，最终代码约为三万八千行，而生成代码两万七千行，生成的代码占大部分，有效减少了手工编写的代码的数量，提高了系统的开发效率。

4 结束语

文中提出一种 MIS 系统快速开发策略，即通用管理信息系统框架+自动代码生成器的架构，适于中小企业信息化管理系统的开发，可以大幅度地减少手工代码编写数量，有效缩短系统的开发周期，降低系统的开发成本。系统架构未来的改进，需要扩充的功能包括：表示层支持、开发过程支持和多表关联支持。

参考文献：

[1] 宋翔宇，曾雅琳. 一种新的代码生成器的设计与实现[J]. 计算机科学，2011，38(7A)：67-69.

[2] 陆 远，胡 莹. .NET 平台下敏捷开发架构及代码生成技术[J]. 微计算机信息，2009，25(11-3)：11-12.

[3] Turner M S V. Microsoft Solutions Framework Essentials Building Successful Technology Solutions[R/OL]. 2009. http://www.docin.com/p-132599666.html.

[4] 张 晨. .NET 环境下基本业务系统生成平台的设计与实现[D]. 西安：西安电子科技大学，2008.

[5] 宋 明，唐 虹. 面向中小型 MIS 的数据库操作池的研究[J]. 计算机工程与设计，2005，26(9)：2519-2521.

[6] 张 静，孔 芳. 一个数据模型驱动的代码生成工具的设计与实现[J]. 计算机应用与软件，2010，27(11)：151-153.

[7] 杨学增，王满敬. 基于 B/S 三层架构的 ASP.NET 系统的代码生成研究[J]. 软件导刊，2009，8(7)：11-13.

[8] 张志杰. 基于分层结构的管理信息系统架构设计[J]. 计算机技术与发展，2010，20(10)：146-153.

[9] 白尚旺. PowerDesiger 软件分析设计技术[M]. 北京：电子工业出版社，2002：79-132.

[10] Yao Leiyue，Chen Yong. An Optimizing Strategy for Massive Data Management System Based on SQLSERVER 2000[C]//Proceedings of 2009 Asia-pacific Conference on Information Processing(Volume 1). [s. l.]：[s. n.]，2009.

[11] 李娟娟，李龙澍. 基于 Web 的 MIS 开发中动态报表的实现[J]. 计算机技术与发展，2008，18(8)：179-181.

[12] Weigert T，Weil F，van den Berg A，et al. Automated Code Generation for Industrial-strength Systems[C]//32nd Annual IEEE International Conference on Computer Software and Applications，COMPSAC '08. [s. l.]：[s. n.]，2008：464-472.