状态监测系统中实时数据库的研究

林 俊[1],林 敏[2],胡华威[1]
(1.江苏西电南自智能电力设备有限公司,江苏 南京 211102;
2.南京国电南自电网自动化有限公司,江苏 南京 211100)

摘 要:针对智能变电站中状态监测系统的需求,设计适应 IEC 61850 标准应用的实时数据库。数据库采用面向对象的设计,根据 SCL 模型文件中描述的结构,以类似于多叉树的方式在内存中建立实时数据结构,同时将数据库分解成若干独立模块,以完成数据调度、数据压缩、并发控制、查询处理等核心功能。

关键词:状态监测;实时数据库;IEC 61850;面向对象;多叉树
中图分类号:TM769 **文献标志码**:B **文章编号**:1009-0665(2012)03-0031-03

在智能变电站中,状态监测的综合监测单元采用 IEC 61850 标准与站端系统通信,站端监测系统收集各单元汇总的监测信息,将其存入监测数据库中,所有这些数据有些是连续变化的,有些是离散变化的,变化快慢各异,数据库需要适应这些数据存储的要求,并通过人机界面将处理过的数据集中显示,供设备检修人员使用;另一方面需要将数据对远方集中监测主站发布。一般的商用数据库多是磁盘存储,其数据需要按照一定的交换策略被调至内存缓冲区后才能被存取,在执行时受到磁盘 I/O 的束缚,事务的响应时间受等待延迟、查找延迟等限制,因此在系统中需设计实时数据库,数据库常驻内存,消除了事务处理的 I/O 瓶颈问题,提高了系统的性能和吞吐量,满足实时响应的要求[1]。

1 实时数据库的需求

实时数据库是指数据和事务都具有定时特性或确定的定时限制的数据库系统,其正确性不仅依赖于逻辑结果,而且依赖于逻辑结果产生的时间,即系统宁可接受在时间限度内的不准确的数据,也不接受超过时间限制的准确的数据。所以数据库一方面要求很强的时效性,即必须在一定的时间内完成数据的传输和预处理,另一方面要求很强的可靠性和吞吐性能,即使在流量很大时,也不能丢失数据。通常实时数据库应满足下列功能需求[2]:(1)实时数据处理的高速度和高精度,以保证整个系统的稳定运行;(2)历史数据存储和压缩,使系统能够存储足够的信息,以备查询、分析;(3)应用程序开发接口,保证系统的开发和可扩展性;(4)方便、直观的图形化组态工具,便于客户的操作和监视。

2 实时数据库的设计

收稿日期:2011-12-16;修回日期:2012-01-31

为保证实时数据库的运行效率,该系统按地址直接访问的方式建立实时数据库,即通过共享内存数据完成数据的存储,这是最快的可用进程间通信(IPC)形式。在共享内存中构建按地址指针直接访问数据的实时数据库,以满足通信的实时性和可靠性的需求。另外实时数据库需要将数据定时存入历史数据库中以备查询和其他接口使用,历史库利用商业数据库 MySql 通用、灵活等诸多优点定义和保存所需数据[3]。因此实时数据处理进程中建立了两级数据存储机制:(1)内存数据库,保证查询的实时、快速性,但是数据量有限;(2)使用磁盘的历史数据库,将经过压缩的数据存入历史数据库中,保证数据掉电不丢失,且易于查询和综合分析。

2.1 数据库构建过程

由于采用 IEC 61850 标准通信,为适应数据结构,内存数据库采用面向对象的方式构造,根据 IEC 61850 提供的变电站配置描述语言 SCL 形成的 XML 格式的模型文件,为每个 XML 元素创建对应的对象,由于模型文件采用树状结构编排,因此所有解析后的对象也形成树形结构,所有数据都保存在树的节点中,所有的访问均是针对该树状结构[4]。该内存数据库采用多叉树的方式构建,首先,分析模型文件,得到第 1 个元素及其子元素,并为该元素及其子元素分别创建对象 p 和$q_1,\cdots\cdots,q_n$,其中,n 是子元素的个数;其次,构建多叉树,分别将 $q_1,\cdots\cdots,q_n$ 设置为 p 的孩子;最后,对于已将创建对象的元素,记其对象为 p,分析模型文件,得到该元素的子元素,并为其创建对象,不妨仍记为 $q_1,\cdots\cdots,q_n$,重复第二步,直到文档中所有的元素都创建了对应的对象,构造结束[5]。构造出的多叉树结构如图 1 所示。

2.2 数据存储过程

系统初始化时,实时数据处理进程将根据 IEC 61850 的模型文件构建内存数据库,同时将数据结

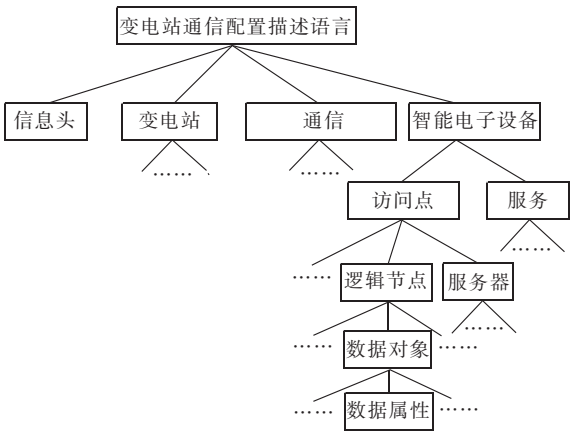

图 1 多叉树结构

构信息通知数据缓存模块，缓存模块为实时数据库模块和历史数据模块,分别开辟数据缓存池,通过通信接口接收各数据点的信息，并将信息同时填入对应的数据缓存结构中。实时数据缓存池根据共享内存数据库提供的操作指针，直接将数据放入内存库中,提供给在线运行模块使用;历史数据缓存池接收数据后首先经压缩模块，产生实时数据送往历史数据处理模块做二次缓存，待缓存数据窗填满或到达自动转存时间后发送至历史数据库。整个实时数据系统的数据存储过程如下图 2 所示。

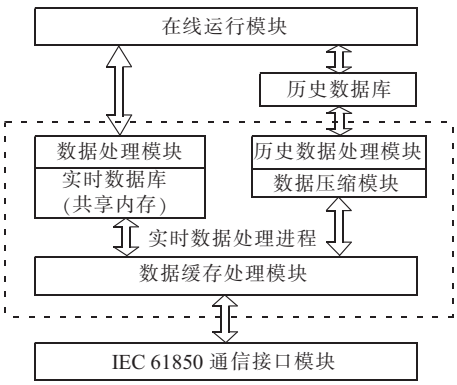

图 2 数据存储过程

在系统运行时，同时要进行数据采集、数据处理、数据读写和查询、事件报警、历史数据存盘等操作,因此需建立并发控制机制,以满足实时性要求[6]。

3 实时数据库的实现

3.1 建立内存数据库

系统定义了多叉树节点的数据结构：

```
struct M_APP
{
  short tag;
  union
  {
 M_Folder   *folder;      // 集合类型
 M_Object   *object;      // 数据对象
 M_Point    *point;       // 数据属性
 M_Var      *var;       // 数据值
 M_Const    *varConst;        // 数据常量
 M_MXLink   *mxLink;       // 数据配置类型
 M_Net      *net;       // 网络配置
 M_Report   *report;      // 报告类型
 M_File     *file;     // 文件类型
 M_Chart    *chart;       // 图表类型
  }g;
  short childNum;
  M_APP **child;
  M_APP *parent;
};
```

每种类型的节点都单独定义了数据结构,如 M_Object 为数据对象节点等,在每个节点实例中都分配了一个 ID 号，作为该类型数据节点数组的下标,ID 号根据哈希算法获得，是每个节点的唯一标示，理想的哈希表采用特定的哈希函数能够在记录的存储位置与其关键值之间建立确定的对应关系，使每个关键字和结构中唯一的存储位置相对应,因此通过这个 ID 可以不需要比较便直接取得查找的数据，使用该 ID 建立起来的索引能够提高数据库的访问速度。

由于内存库采用面向对象的思想构建，每种类型节点都是一张独立的表,表间的关联关系采用父、子指针的方式创建,而非关系型数据库的外键关联。

3.2 定义数据库接口

在实时数据处理进程中提供一组接口函数实现对实时数据库的读写、查询和管理等工作。常用接口如表 1 所示。除了定义常用的读取、查询等操作接口外,数据库提供高效的消息通知机制,以便快速将数据变化发布给系统各进程使用。需要消息通知的进程,如告警信息进程,将感兴趣的数据点在实时数据库中进行注册,并给出通知的触发选项,如“数据变化”、“品质变化”等,被注册过的数据点,当发生上述改变时,即刻调用通知接口,将信息发送给需要的各进程,这些接口均由数据处理模块完成[7]。

3.3 历史数据处理方式

历史数据处理时首先需要经过数据压缩，对于开关类型数据采用变化保存的方式压缩，压缩模块在处理开关量时将其与历史数据处理模块中缓存的最后一个点数据进行比较,若相同则不保存状态,只更新最后一点收到的时间，若不同则产生一条新的缓存数据。对于模拟量数据采用旋转门压缩算法,首先在数据库中为每个数据点设置默认的压缩因子，

表 1 数据库接口

函数类型	函数名	函数说明
void	readAPPMem (M_APP**parentItem, QDataStream & stream, bool deep)	读取父节点下数据结构
bool	readBoolMem (QDataStream & stream)	读取 bool 型数据
int	readIntMem (QDataStream & stream)	读取整型数据
M_APP*	findApp(QString id)	通过 ID 搜索数据
QList<M_APP*>	findApp(int dbType)	通过数据结构搜索数据
bool	setAppValue (QString id,int editType, int index,QVariant v)	设置确定 ID 下,index 索引的值
bool	setAppValue (QString id,int editType, M_APP*editApp)	设置确定 ID 下,所有索引的值

将收到的数据点与缓存数据中前一个被保留的构成压缩偏移覆盖区域,若区域中没有任何点落在区域以外,则无需保存任何数据,若区域中有任何一点未被覆盖,则将新收到的数据点的前一点保存入缓存,之后以该保存点为起点,继续接收数据,并重复该算法。所谓的压缩覆盖区域实际上是平行四边形,起点为前一个被保存的点,终点是新收到的数据点,其垂直边长为压缩因子的 2 倍,并以数据点为中点;使用压缩算法可以大幅减少数据存储量,保证历史数据库能够长期运行[8]。

历史数据写入采用 2 级写入控制,即(1) 历史数据模块中为每种数据类型开辟两个缓存窗口,当一个窗口填满时,自动将窗口数据一次性写入历史库,同时清空该窗口,新收到的数据填入另一缓存窗口中;(2) 历史数据库根据系统设置,定时交替读取 2 个缓存窗口,同时清空被读取的窗口,新收到的数据填入另一缓存窗口中。通过该方法可以减少磁盘的读写次数,加快写入效率。

3.4 数据并发控制

针对数据并发控制,实时数据库提供了数据对象加 / 解锁机制, 具有操作权限的进程在使用数据库时首先执行 LOCK 操作锁定指定的对象,从而独占该资源,并在完成后执行 UNLOCK 操作解锁,释放对应的资源。如果该操作进程退出或出现异常,未能按指定时间解锁资源, 则实时库将自动释放该资源,以避免数据死锁,保证系统的稳定性。

4 结束语

文中分析了状态监测系统中实时数据库的需求, 根据 IEC 61850 标准要求构建了实时内存数据库,并设计了数据库的操作模块,给出了数据库各功能模块的实现方式。通过该方法开发的数据库已用于监测系统软件中,提高了系统的实时性和可靠性,能够适应智能变电站应用的需求。

参考文献:

[1] 傅 蕾,胡敏强. 变电站监控软件系统中内存数据库的研究[J]. 电力自动化设备,2002,22(10):67-70.

[2] 周 钊.基于实时数据库的历史数据处理技术研究[D]. 北京:华北电力大学,2008.

[3] 谈苏伟.电网调度自动化前置子系统实时数据库的设计和实现[J]. 电力自动化设备,2009,29(7):130-133.

[4] 董 越,孙宏斌,吴文传,等. EMS 中公共信息模型导入 / 导出技术[J]. 电力系统自动化, 2002,26(3):10-14.

[5] 黄建才.基于 IEC 61850 的内存数据库的组织[J]. 福建电脑,2007(6):101-102.

[6] 刘 浩,万 昆. 基于动态链接库 DLL 的实时数据库系统研究与开发[J]. 东北电力学院学报, 2005,25(1):36-39.

[7] 金 舒,戴宏斌,贾志敏. 面向对象的高性能实时数据库 ChRDB[J]. 电力自动化设备,2009,29(12):88-93.

[8] 陆会明,周 钊,廖常斌. 基于实时数据库系统的历史数据处理[J]. 电力自动化设备,2009,29(3):127-131.

作者简介:

林 俊(1978),男,江苏南京人,工程师,从事智能变电站相关专业的应用研究工作;

林 敏(1976),女,江苏南京人,助理工程师,从事智能变电站相关专业的应用研究工作;

胡华威 (1983),男,浙江金华人,工程师,从事智能变电站相关应用软件的研发工作。

Research on the Real-time Database in State Monitoring

LIN Jun[1], LIN Min[2], HU Hua-wei[1]

(1. Jiangsu Xidiannanzi Smart Electric Power Equipment Co.Ltd., Nanjing 211102, China;

2. Guodian Nanjing Automation Co. Ltd., Nanjing 211100, China)

Abstract: For the requirements of state monitoring system in smart substation, real-time database adaptive to the IEC 61850 standards should be designed. The database applies object oriented design and establishes real-time data structure in the way similar to multi-way tree according to the structure described by SCL model file. Besides, the database are divided into several independent modules, which can complete functions such as data scheduling, data compression, concurrency control, query processing and so on.

Key words: state monitoring; real-time database; IEC 61850; object oriented; multi-way tree structure