第35卷 第10期 计 算 机 工 程 2009年5月
Vol.35 No.10 Computer Engineering May 2009
·软件技术与数据库· 文章编号：1000—3428(2009)10—0054—03 文献标识码：A 中图分类号：TP311.13

Oracle 监控处理系统的设计与实现

刘凤龙
(湖南人文科技学院信息中心，娄底 417000)

摘 要：针对 Oracle 数据库的有效监控问题，提出一个基于 CORBA 的分布式 Oracle 监控处理系统。系统采用“代理+Oracle 视图”技术，通过灵活的监控策略配置、多层次多粒度内容呈现等方法，实现对物理上分布的多个 Oracle 数据库进行不间断的集中式监控、分析处理和故障分析预警等功能，采用预测模型方法有效降低监控引起的网络流量。实验结果和型号项目中的实际应用验证了系统的有效性。

关键词：Oracle 数据库；监控；服务器资源；CDOMS 系统；故障

Design and Implementation of Oracle Monitoring and Processing System

LIU Feng-long
(Information Center, Hunan Institute of Humanities Science and Technology, Loudi 417000)

【Abstract】In order to guarantee the continuous and efficient performance of Oracle databases, a CORBA-based Distributed Oracle Monitoring System(CDOMS) is designed and implemented. “Agent plus oracle view surface” is used in CDOMS, and flexible configuration, two-level based content presentation and prediction techniques are adopted in CDOMS adopts to achieve cost-efficient monitoring. It is proved experimentally and practically that CDOMS can continuously monitor distributed Oracle database system and server’s resource with different granularities in real-time, alarm for potential faults in advance, analyze the database system comprehensively and manage the remote nodes effectively. Experimental result and the actual application of nomenclature project prove effectivity of the system.

【Key words】Oracle; monitoring; server resources; CDOMS system; malfunction

1 概述

随着信息技术的不断发展，各种信息资源在物理上日趋分布，因此，分布式数据库系统得到广泛应用[1]。在这些系统中存在多个物理上分布的数据库，相互配合完成一系列复杂的业务流程和各种操作。要保障这些数据库的持续高效运行，须对其进行不间断的监控。

本文针对物理上分布的多台 Oracle 数据库服务器，设计实现了一种基于 CORBA 的分布式数据库性能监控系统(CORBA-based Distributed Oracle Monitoring System, CDOMS)，旨在通过对数据库 7×24 h 不间断的监控，为分布式系统的持续高效运行提供支持。CDOMS 系统采用了基于层次的内容呈现和预测模型相结合的方法，在提供持续监控、分析预警及远程处理等功能的基础上，尽量降低 CPU 及内存利用率，减少网络通信开销，以减少对在线系统的影响。

2 研究现状

当前有不少商用工具对 Oracle 数据库服务器提供监控和管理功能，比较有影响的有 DBA Studio, OEM[2], Quest Central[3]等。通过比较可以看出，现有的监控工具在集成度、集中式多点监控、实时监测、提前预警、灵活性、二次开发和安全性等方面都存在一些不足。前期关于 Oracle 数据库系统的管理维护，主要集中在对部分参数的性能优化、故障诊断等方面，如对磁盘 I/O 的优化[4]等。近几年来，研究者普遍关注对 Oracle 系统的全面监控和优化，并提出了相关的系统设计实现方案，如基于 J2EE 的数据库性能监控和分析系统的总体设计及部分实现[5]。这些研究与本文的工作类似，但是本文面向的是多个 Oracle 数据库的全面监控，通过灵活可配置的监控策略、基于层次的内容呈现等方面进行了进一步的优化设计，因此，在可监控的内容、监控灵活性、监控策略、监控开销等方面具有更好的功能、性能和可扩展性。

3 CDOMS 系统设计与实现

3.1 系统结构

影响数据库可用性的因素主要包括 2 个方面：数据库所在服务器的资源使用情况和数据库自身参数的设置。对于前者，可以通过操作系统提供的 API 函数获得相关信息，而 Oracle 自身提供的大量系统视图为及时获知 Oracle 内部参数、各种资源争用和瓶颈判断等提供了依据。因此，CDOMS 采用“代理+访问系统视图”的总体框架，如图 1 所示。

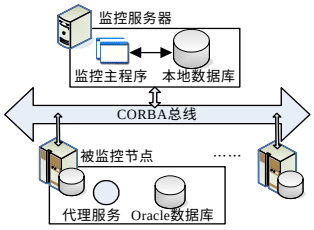


图 1 CDOMS 系统结构

基金项目：湖南省普通高校教学改革研究基金资助项目(湘教通[2006]171 号 No.165)
作者简介：刘凤龙(1971－)，男，讲师、硕士，主研方向：数据库技术，信息系统开发，信息安全
收稿日期：2008-09-26 **E-mail：**liu_feng_long@126.com

CDOMS 采用基于代理的 C/S 方式。监控软件主程序依据本地数据库的配置信息，通过调用被监控节点上的 NodeManager 代理服务和直接访问 Oracle 数据库系统视图，分别获得被监控节点操作系统和数据库内部实时状态信息，并根据存储在本地数据库中的规则库和知识库进行判断，对可能会发生问题的节点及时进行预警，并提供一定的辅助管理功能。代理服务和监控主程序通过 CORBA 协议进行通信。

3.2 监控系统模块划分

CDOMS 的系统功能模块划分如图 2 所示。

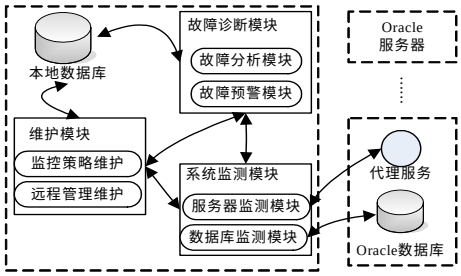


图 2 CDOMS 系统功能模块

各功能模块的具体分析如下：

(1)系统监测模块

高效、准确地形成系统的全局状态监测视图不仅可以使系统管理员全面直观地了解到系统当前的运行情况，而且是实施系统自主诊断和自主维护能力的重要前提。系统监测模块具体包括 2 部分功能：

1)数据库监测模块。在不同粒度和不同层次上监控数据库系统状态和行为，包括数据库的表空间使用情况、数据库链路的占用情况、数据库操作的响应时间、数据库性能情况、数据库日志查询等。

2)服务器监测模块。对 Oracle 数据库所在服务器的系统资源进行全面监控，包括 CPU/内存利用率、磁盘空间、系统进程数等信息。

(2)故障诊断模块

系统对监测模块获得的实时状态信息进行判断，通过预存在本地数据库中的知识和规则进行推理，对可能影响系统正常运行、可能造成系统故障或导致性能下降的场景提前给出警告提示。故障诊断模块就是在故障将要发生以前发现该故障，并及时进行处理，避免故障的发生，具体包括 2 部分功能：

1)故障分析模块。故障按其影响的直接与否可分为显性和隐性 2 类。显性故障是指可以直接导致系统失效的一类问题，比如线程死锁、访问悬空指针所致的中间件服务器失效等。隐性故障指须经过积累才会引发系统失效的一类问题，比如内存泄漏等。故障分析模块可以结合日志分析技术与本地知识规则库，通过主动可用性探测来发现显性故障，通过累计性能差分(differentiation)技术来发现隐性异常。故障分析模块的另一个重要功能就是故障定位，目标是将故障定位到资源一级(如果是硬件故障，资源指发生故障的硬件设备；如果是软件故障，资源指导致故障的进程)。

2)故障预警模块。发现并定位故障后，故障预警模块负责采用鲜明形式主动报警。故障预警模块与本地数据库中的知识库结合，提供关于故障或预警的详细描述，如警告来源、警告类型、警告描述、可能的原因以及建议采取的动作等。

(3)维护模块

维护模块除了具有本地参数配置的功能，还可以在故障发生并被定位后，对发生故障的资源实施远程故障修复。维护模块具体包括 2 部分功能：

1)监控策略维护模块。提供灵活的参数设置功能，方便用户根据实际需要进行设置。这些参数包括为不同被监控节点设置的不同监控策略、系统定时监测的时间周期、是否主动弹出报警、是否生成系统日志等。

2)远程管理维护模块。提供一定的 GUI 远程处理和管理功能，既可以作为预警发生后解决问题的手段，又可用于日常的数据库维护和优化工作。这些管理功能包括磁盘清理功能、远程表空间配置管理、远程回退段配置管理、杀进程以及自由空间碎片索引整理等。

3.3 监控信息获取

3.3.1 通过系统视图获得 Oracle 状态的实现

Oracle 提供了大量的系统 v$视图和统计表，从不同角度实时记录了系统的当前状态。例如 v$session 表中记录的是当前数据库链接相关信息、v$sess_io 统计的是物理逻辑 I/O 信息等。对这些视图进行关联和统计查询，可以从不同粒度上获得 Oracle 系统的当前性能数据。通过对大量 Oracle 系统视图的分析查询，CDOMS 实现了对 Oracle 数据库内部状态信息的获取功能。

3.3.2 通过系统 API 获得服务器信息的实现

CDOMS 面向的是分布式系统，因此，服务器代理必须具有分布、跨平台、高效等特性。目前主流的分布对象技术主要有 COM/DCOM, J2EE 和 CORBA。

与其他技术相比，CORBA 在性能、跨平台、跨语言等方面具有诸多明显的优点，因此，本文实现的 CDOMS 基于 CORBA 规范，在具体实现中选择了遵循 CORBA 2.3 规范的 VisiBroker。通过与服务器操作系统相关 API 的交互，代理实现了服务器资源相关信息的获取。服务代理的 IDL 接口定义如下：

```
interface ServerAgent
{ void getSumInfo(out SumInfo summaryInfo); //系统概要信息
  void getMemInfo(out MemInfo memoryInfo); //内存使用信息
  boolean getValidDrives(out Drives drvs); //系统磁盘分区列表
  boolean getDiskInfo(in string filesystem, out DiskInfo
diskinformation, inout unsigned long flag);//获得指定盘符的硬盘信息
  ProcessSequence getProcessInfo();//获得系统进程列表
  boolean getDiskUsage(out DiskUsage dusage);
   //一次性获得所有磁盘分区的空间信息
  void getNetworkInformation(out NTISequence nsequence);
  //获得网络流量统计信息
  ProcessPortSequence getPortInformation();   //进程端口信息
  };
```

可见，代理能够获得服务器资源的概要和详细状态信息，主要包括 CPU 利用率、内存利用率、系统进程数、磁盘分区数目以及可用空间情况、网络 I/O 利用率以及对应的详细列表信息等。

3.4 系统实现类图

系统实现类图如图 3 所示，主要类介绍如下：

(1)OracleMonitor：监控软件主控制类，它依赖于其他功能类，完成系统的监控、故障分析预警以及远程处理功能。

(2)HostMonitor：监控主机信息的功能类。它通过本地数据库中的配置信息获得被监控节点上 NodeManager 服务的对象引用，并通过 NodeManager 获得系统信息。

(3)NodeManager：实体功能类，拥有能够获得系统资源

信息(如 CPU 利用率、硬盘空间等)的方法。

(4)DBMonitor：监控数据库信息的功能类。它通过与 Oracle 数据库直接通信，获取其系统视图内的数据库相关信息。

(5)SysInfo：系统信息类。DBinfo 和 HostInfo 均继承自 SysInfo 类。DBInfo 是数据库信息类，包括数据库的表空间使用情况、数据库链路的占用情况、数据库操作的响应时间、数据库性能情况、数据库日志查询等；HostInfo 是主机信息类，包括 CPU、内存利用率、磁盘空间、系统进程数等信息。

(6)TjcThread：系统监测线程类。该线程通过调用其他功能类，定时获得要监测的信息。

(7)TjcReportThread：该线程用于对某个被监测节点做全面分析，并生成 html 格式的分析报告。

(8)WarnAnalysis：故障分析类。它根据监控类获得的信息分析定位故障。

(9)WarnGenerator：该类根据故障分析的结果，准确进行预警提示。

(10)Processor 类和 SysConf 类：远程处理和系统配置。

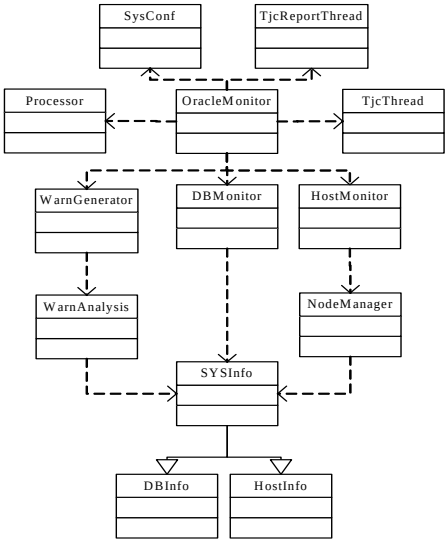


图 3　CDOMS 系统类图

4　CDOMS 系统关键技术

CDOMS 的目的是同时监控多个物理上分布的 Oracle 数据库。为了减小对原有业务造成的影响，系统采用了一系列关键技术，降低监控引起的计算、通信等方面的额外开销。

4.1　基于层次的内容呈现

CDOMS 采用基于层次的内容呈现方法，定时刷新必须判断的可能产生报警的各种概要内容，只在需要时才去获取详细信息。

突出“层次”后，会给系统带来如下优点：

(1)方便用户使用：用户可以迅速了解数据库运行的概况，对感兴趣的内容可以查看详细信息；

(2)减轻对在线系统的影响：只定时获得必须的监控信息，减少了被监控服务器的 CPU 和内存开销；

(3)减轻网络负担：只传输必须的监控信息，有效减少网络流量；

(4)提高系统运行效率：监控系统须处理的内容变少，运行更加高效。

4.2　采用预测模型降低网络通信开销的方法

CDOMS 针对不同的监测内容采用不同的预测模型，以进一步降低网络通信开销。采用预测模型降低网络通信开销的基本思想是：在中心节点和分节点分别增加预测模型，根据历史数据对当前值进行预测，只有在精度误差超出预定的容忍范围时才进行必要通信以更新相关参数，否则就用中心节点的预测值代替实际值。这样可以在保证精度的基础上，有效地降低中心节点和分节点之间的通信量。

系统采用 3 种预测模型[6]：静态模型(Static Model-SM)，线性模型(Linear Model, LM)，加速模型(Acceleration Model, AM)。预测模型描述如下：

SM：$f_{i,j}^{p}(t)=f_{i,j}(t_{i,j}^{prev})$

LM：$f_{i,j}^{p}(t)=f_{i,j}(t_{i,j}^{prev})+v_{i,j}\Delta t_{i,j}$

AM：$f_{i,j}^{p}(t)=f_{i,j}(t_{i,j}^{prev})+v_{i,j}\Delta t_{i,j}+(\Delta t_{i,j})^2 a_{i,j}$

4.3　可配置的监控策略

CDOMS 系统提供灵活的参数设置功能，方便用户根据被监控节点重要性的不同，定制不同监控策略。可定制参数包括系统定时监控判断的时间周期、各种报警阈值、是否主动弹出报警、是否忽略某特定预警信息、具体监控判断的内容等。添加新的监控对象时，采用默认的监控策略，用户可以根据实际需要对该策略进行修改，生成自己的监控策略。

5　CDOMS 系统实现与测试

为了验证本文设计实现方法的有效性，对 CDOMS 系统进行功能性能测试。测试环境为通过集线器连接的 3 台 PC 机组成的局域网，其中 1 台机器作为集中式监控节点，另外 2 台安装 Oracle 10g，作为被监控的数据库服务器。PC 机基本配置为 CPU 2.0 GHz，1 GB 内存，120 GB 硬盘。测试方法是在集中式监控节点上配置多个本地网络服务名，分别指向 2 台数据库服务器中的某个 Oracle 实例，监控主程序通过网络服务名访问数据库，模拟多个被监控节点，收集在不同条件下 CDOMS 系统的效率信息。

5.1　对被监控节点的影响

将某个被监控节点的监控周期设置为 15 s，在取消监控和进行监控 2 种情况下，记录被监控节点的 CPU 和内存使用情况，对比得到监控引起的系统额外开销情况如下：

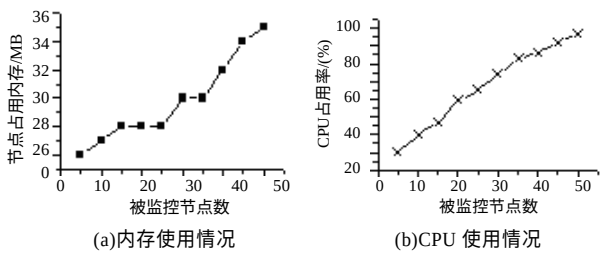
(1)服务器代理常驻内存，占用内存不足 10 MB；

(2)每次监控周期来临时，代理仅需调用系统 API 函数获取一定的资源使用信息，CPU 占用率峰值约为 1%；

(3)监控程序访问 Oracle 系统视图带来的额外 CPU 峰值约为 4%。

5.2　系统监控规模

逐渐增多被监控节点数目，考查集中式监控节点上的 CPU 和内存利用情况，获得系统最大监控规模。测试结果如图 4 所示。



(a)内存使用情况　　(b)CPU 使用情况

图 4　集中式节点资源利用峰值随被监控节点数变化情况

可以看出，随着被监控节点数的不断增加，集中式监控节点内存占用峰值和 CPU 占用率峰值均呈上升趋势。在最坏

(下转第 59 页)