

文章编号:1001-9081(2005)07-1587-03

织女星地理信息系统 VegaGIS 中的空间数据引擎 CoSDE

何 雄,方金云,唐志敏

(中国科学院 计算技术研究所,北京 100080)

(hex@ict.ac.cn)

摘 要:介绍了 VegaGIS 中的空间数据引擎 CoSDE 的体系结构、设计和实现,通过空间数据引擎 CoSDE 提供的服务实现对多种空间数据库的直接访问。CoSDE 采取了真正的三层 C/S 体系,屏蔽了客户端对数据库的直接访问,它与具体的数据库实现了物理上的分离,同时实现了真正的跨平台。

关键词:织女星地理信息系统;空间数据引擎;CoSDE

中图分类号: TP311.13 **文献标识码:** A

CoSDE: spatial database engine of VegaGIS

HE Xiong, FANG Jin-yun, TANG Zhi-min

(Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100080, China)

Abstract: The architecture, design and implementation of CoSDE in VegaGIS were introduced. It enabled the CoSDE server to access multiple types of spatial database directly. CoSDE has a real three-tiered C/S architecture, which enables the client application to access the database transparently, separates the specific database physically and implements a real cross-platform server application.

Key words: VegaGIS; Spatial Database Engine(SDE); CoSDE

0 引言

空间数据引擎(Spatial Database Engine, SDE)是一种介于应用程序和空间数据库之间的中间件技术,它为应用程序和后台的空间数据库提供了一个访问的关口,并且在用户和空间数据库的数据之间提供了一个用于开发的统一接口。

目前,国内外大型 GIS 的空间数据引擎产品的设计和实现各有千秋。从 SDE 的体系结构角度来看,可以分以下两种:

1) 与客户端直接绑定。SDE 直接是客户端的一部分,它作为数据库的访问接口,客户端通过调用 SDE 接口来访问数据库。如 MapInfo 的 SpatialWare 和 SuperMap 的 XSDE^[2,3]。

2) 采取 C/S 结构。SDE 有自己的服务器端和客户端。客户端通过调用 SDE 接口来访问 SDE 提供的服务。SDE 服务器端通过访问数据库实现 SDE 服务器的接口。如 ESRI 的 ArcSDE,它采用的是混合的两层和三层 C/S 结构^[4],但是 ArcSDE 在物理上必须与具体的数据库绑定在一起,结构不灵活,无法同时管理多个数据库。

从支持的平台来看,国内的 SuperMap 和 MapGIS 的空间数据引擎目前仅能在 Windows 操作系统上运行^[5,6]。

如果采取 C/S 体系结构,同时 SDE 与具体数据库进行物理上的分离,就可使 SDE 同时管理多个数据库。SDE 与数据库的分离意味着它与数据库可以运行在不同的平台上。

织女星地理信息系统 VegaGIS 中的空间数据引擎 CoSDE 采取了真正的三层 C/S 体系结构,屏蔽了客户端对数据库的直接访问,它与具体的数据库实现了物理上的分离,同时实现了真正的跨平台。一方面,可以提供对多种数据库的支持,减少了对特定数据库的依赖,另一方面,为客户端应用提供统一的

空间数据访问接口,实现更大程度的复用。

1 CoSDE 的体系

在 VegaGIS 中, CoSDE 采取的是三层的 C/S 结构。依次为 CoSDE 客户端(SDEClient)、CoSDE 服务器端(SDEServer)和数据库服务器(DBServer)。

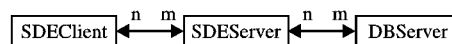


图 1 CoSDE 的体系结构

其中, DBServer 用于存储数据。SDEServer 通过调用 DBServer 的访问接口来实现海量空间数据的存储和管理,它同时充当 DBServer 的客户端。

在 VegaGIS 中,应用服务器 AppServer 充当 CoSDE 的客户端。通过调用 SDEClient 中的 API 来访问 SDEServer 提供的服务,同时,它实现了空间数据分析和基本的空间数据操作和处理。同时 CoCatalog 也作为 CoSDE 的客户端,用于远程管理空间数据。

从用户的角度来看,他看到的空间数据库就是 CoSDE,因而 CoSDE 是一个虚拟的空间数据库。它屏蔽了客户端对数据库的直接访问,实现了对空间数据的一体化管理。

VegaGIS 中的空间数据引擎支持当前主流的大型商用数据库(Oracle, DB2, SQL Server),还支持小型高效的数据库 MySQL 和 Access。基于 Oracle, DB2 以及 MySQL 的 CoSDE 都实现了跨平台。它们都采用了统一的存储格式,对空间数据及其属性数据进行统一存储和管理,通过调用 SDE 的 API 可以直接访问不同类型数据库中的数据,真正实现了多源数据的无缝集成^[5]。

收稿日期:2005-01-07;修订日期:2005-03-08

基金项目:863 计划项目(2002AA114020, 2001AA135210);中国科学院知识创新项目(20036020)

作者简介:何雄(1978-),男,湖北黄冈人,博士研究生,主要研究方向:空间数据库技术、中间件;方金云(1968-),男,山东青岛人,副研究员,博士,主要研究方向:海量空间数据处理关键支撑技术、分布式系统开发方法学、网络 GIS;唐志敏(1966-),男,江苏江阴人,研究员,博士生导师,主要研究方向:计算机系统结构、网络并行处理。

2 空间数据引擎 CoSDE 的功能模型

CoSDE 是应用服务器 AppServer 访问后台数据库的桥梁,它分为客户端和服务端两部分,两者之间通过面向连接的 Socket 进行通信,服务器端采用的是多线程的并发处理模型。CoSDE 提供的主要功能有:建立与空间数据库的连接或断连;空间数据的存取;空间索引的创建和更新;影像金字塔的创建;关系表的定义;关系表的查询;关系表数据的更新和删除;用户权限管理;日志管理等。其中最核心的功能是空间数据的存取。

1) CoSDE 服务器端

服务器端的底层包含一些基本的类库,一个是通信库接口,一个是数据库接口,它还定义了一些空间数据相关类,用于空间数据的网络传输。它定义一系列最基本的原子操作函数来实现上述主要功能。这些原子操作函数形成服务器端最基本的 API。每个 API 对应客户端的一个消息请求。

通信库接口由以下基本类组成, NxCMsg(消息类,用于定义与客户端间通信所用的消息体), NxSocket(套接字类,是一个抽象基类,定义 send() 和 receive() 两个最基本的接口函数), NxServerSocket(从 NxSocket 继承,实现服务器端的通信), NxClientSocket(从 NxSocket 继承,实现客户端的通信), NxStream(消息流,从 NxCMsg 继承,实现类似于 I/O 流的通信操作)。

数据库接口是对关系数据库接口的一种抽象,主要有 HYDB(数据库)和 HYDBProxy(数据库代理)两个抽象的基类,它们定义了关系数据库的最基本操作(连接、数据更新与删除、获取库表结构、获取记录集的字段定义、查询操作、创建表的关联)等。同时数据库接口还包括 HYField(字段定义)、HYFields(字段集合)和 HYRowSet(记录集)。

2) CoSDE 客户端

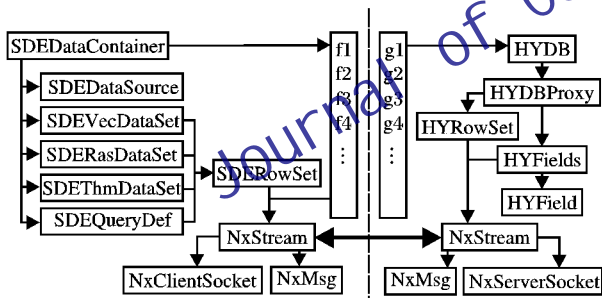


图2 CoSDE 的系统模型

CoSDE 客户端接口由一些最基本的 API 所组成,每一个 API 都是通过发送一个或多个消息到 CoSDE 服务器端,获取最后结果来实现的。底层与服务器端共用相同的通信库。由服务器端提供的最基本的原子操作可以为客户端实现功能丰富的、大量的 API 函数。在实现这些 API 的基础上,客户端通过调用这些 API 函数封装了一套类库,供应用服务器 AppServer 使用。主要包括: SDEDataSource(数据源,定义空间数据的来源)、SDEDataContainer(数据容器,用于管理所有空间数据集,包括专题数据集、矢量图层数据集、栅格图层数据集等)、SDEVecDataSet(矢量数据集)、SDEVecDataDef(栅格数据集)、SDEThmDataSet(专题数据集)、SDERowSet(记录行,用于存储和处理多行关系表的记录)、SDEQueryDef(数据库查询的定义)和 SDEColumn(表中的字段定义)。

图2给出了 CoSDE 的系统模型,虚线左边对应的是客户端的基本结构;右边是服务器端的基本结构。左边的 f1, f2, f3, ... 表示客户端的 API 函数,右边的 g1, g2, g3, ... 表示服务器

端的 API 函数,它们是多对多的关系。

服务器端采用的是多线程并发处理模型,即当服务器端收到来自客户端的一个连接请求时,它会产生一个线程用于处理该连接。多线程并发处理模型同时适用于 Windows 和 Linux 两大类操作系统。右边的虚线框部分对应的就是服务器端一个独立的处理线程。

3 CoSDE 的逻辑模型

3.1 CoSDE 客户端

SDEDataContainer 数据容器类,该类主要实现空间数据和属性数据的存取和查询。主要成员有数据源指针、矢量数据集指针、栅格数据集指针、专题数据集指针和查询定义。其主要操作包括设置数据源,创建图层,创建专题,获取图层,更新图层,图取图元信息,获取专题元信息,查询图层,返回查询的结果,多表查询,创建关系表,执行 SQL 语句,启动事务,终止事务。

SDEDataSource 数据源类,实现设置数据源的功能。主要成员包括:数据源类型、数据源名称、SDE 服务器主机名、SDE 服务器的端口号、数据源的用户名和用于连接数据源的密码。基本操作是设置数据源。

SDEVecDataSet 矢量数据集类,用于存储和管理矢量数据集。主要成员有数据集名称。主要操作包括:查询图层,根据图层 ID 获取图层数据,更新图层数据,删除图层数据,根据空间索引类型创建不同的空间索引,生成图层的缓冲区,添加几何实体,更新几何实体,删除几何实体。

SDERowSet 栅格数据集操作类。主要成员有数据集名称、栅格影像类型、栅格影像块数、影像宽、影像高等。主要操作包括:查询图层,返回符合条件的某一级的块的 ID 列表,创建影像金字塔,添加影像块,删除影像块。

SDEThmDataSet 专题数据集操作类。主要成员有矢量数据集的指针数组,栅格数据集的指针数组。主要操作包括:添加矢量图层,移去矢量图层,添加栅格图层,移去栅格图层。

SDERowSet 记录行集类。主要成员包括记录行数组。主要操作包括:获取记录个数,移动游标到指定的行,移动游标到下一行,移动游标到最后一行,移动游标到第一行,删除某一记录行,更新某一记录行,添加一行记录,判断是否到最后一行。

3.2 通信库接口

CoSDE 中的通信库 Nx 是两端共用的,是一个跨平台的构件库。它主要包括 NxSocket, NxMsg, NxClientSocket, NxServerSocket, NxStream 这几个类。其中 NxSocket 是一个抽象基类,它定义了 send() 和 receive() 两个接口函数。NxClientSocket 和 NxServerSocket 从 NxSocket 类继承,实现了 send 和 receive 接口。

NxCMsg 是消息处理类,主要成员包括 MSGHEAD 结构体类型,定义了通信的消息头和消息体数据;主要操作包括设置消息头。其中,MSGHEAD 的成员包括消息体大小、消息的操作码、消息的序号、结束标志、是否响应标志、校验和、消息保留字。

NxClientSocket 是客户端套接字类,主要成员包括:连接的端口号、套接字描述符和连接的地址。主要操作包括:绑定套接字,建立连接,发送消息,接收消息,启动 w_sock32 库,关闭连接。

NxServerSocket 是服务器端套接字类,主要成员包括:服务的端口号和服务器的套接字描述符。主要操作包括:接受客户端的连接,与一个 IP 地址绑定,发送消息,接收消息,

启动 w_sock32 库,关闭连接。

NxStream 是消息流处理类,它从 NxMsg 类继承,主要成员包括:消息体的最大长度、当前消息流的偏移、消息流空间增长块的大小和指向 NxSocket 接口类的指针。主要操作包括:分配空间(缺省大小为 MAXSTREAMLEN),释放已经分配的内存,将流中的数据清零,发送消息流,接收消息流

3.3 CoSDE 服务器端

主要由类 HYDB, HYDBProxy, ISDEWorker, ISDEServerAPI 和 SDEServer 构成。

HYDB, 是数据库的抽象类。主要成员包括:主机名、用户名、密码、数据库名、数据库端口号、错误号、错误消息和 DBMS 名;主要接口操作包括:创建数据库,连接数据库,关闭数据库连接,检查一个表是否存在,检查表中是否存在某列,创建一个 HYDBProxy 对象,返回其指针,返回数据库中所有的用户表,执行一个 SQL 语句。

HYDBProxy, 是 HYDB 的代理类,是一个抽象基类。主要成员包括:HYDB 型指针,指向它代理的数据库;记录的各个字段定义数组;Proxy 中记录的行数和字段个数。主要接口操作包括:查询关系表,移动到指定的游标位置,释放结果集,返回当前第 i 个字段的值,用字符流表示,返回第 i 个字段的 double 型值,返回第 i 个字段的 bool 型值,返回第 i 个字段对应的 blob 型值,设置第 i 个字段的 blob 值,返回第 i 个字段的定义。

ISDEWorker 是工作者线程类,从跨平台的线程类 Thread 继承,它实现一个单独的处理线程。主要成员包括:日志类对象(用于日志管理)、HYDB 型指针、消息流 NxStream 型指针和 SDEServer 型指针。主要操作包括:工作者初始化操作,实现 Thread 类的 run 接口,通过 pStream 接收消息,得到消息操作码,执行 ISDEServerAPI 的相关操作函数。

ISDEServerAPI 是服务器端 API 类,它是一系列 API 成员函数的集合,函数的原型为: SDEFx(HYDB * pDB, NxStream * pStream, HYLog * log)。它通过调用 pDB 的接口,使用 pStream 将处理结果发送到客户端。

SDEServer 是服务器类,它实现整个服务器的框架,主要成员包括:互斥锁、线程工作者指针数组和 NxServerSocket 型指针。其基本的操作是:服务器的初始化,服务器的运行,服务器终止,处理一个客户端连接,创建对应的工作者线程。

4 CoSDE 的实现

CoSDE 服务器屏蔽了客户端对数据库服务器的直接访问,CoSDE 服务器在充分利用关系数据库接口的基础上,实现空间数据和属性数据的统一管理,是对关系数据库功能的一种扩展。这样使得客户端可以集中处理专题、图层、地理要素等面向对象的空间数据,而不仅仅是普通的关系表记录数据。同时 CoSDE 服务器还屏蔽了具体的数据库类型,以统一的形式实现了对多种数据库的空间数据访问接口。为实现 CoSDE 对多种异构的数据库提供了基础。

它采取的是一种“虚拟”的 C/S 体系,其中空间数据引擎是客户端,而具体的数据库则是服务器端。从 SDE 客户端的角度来看,它无须知道具体的数据库类型,它看到的的就是 SDE 服务器端, SDE 服务器端的框架中调用的是统一抽象出来的数据库接口。通过对 SDE 服务器进行实例化,可以创建不同类型的空间数据引擎,从而使系统达到框架上的复用。不同的数据库对应了不同的 SDE 服务器实例。

而 CoSDE 客户端,通过设置不同的数据源,就可以访问不同的空间数据库。SDE 客户端和服务端 API 是固定不

变的,唯一可变的是服务器端的虚拟数据库接口的具体实现。对每一种不同类型数据库的访问,都可以通过继承虚拟数据库接口来实现。

在 CoSDE 中, VSDB 是一个独立的库,是静态库,也可以是动态库,它抽象了绝大部分关于关系数据库的基本操作接口,这些操作接口定义在抽象类 HYDB 及其代理类 HYDBProxy 当中。在 C++ 当中,这些操作接口表现为一系列纯虚函数。

在 SDEServer 的框架里,只需要在工作者线程类 ISDEWorker 的初始化函数 init() 中对 HYDB 类进行实例化,就可以实现不同类型的空间数据引擎。代码如下:

```
void ISDEWorker::init(int nDBType)
{
    switch(nDBType)
    {
        case DB_MDB: //Access 数据库
            pDB = new HYAdo("JET"); break;
        case DB_MSSQL: //SQL Server 数据库
            pDB = new HYAdo("MSSQL"); break;
        case DB_MySQL: //MySQL 数据库
            pDB = new HYMySQL(); break;
        case DB_ORACLE: //Oracle 数据库
            pDB = new HYOracle(); break;
        case DB_DB2: //DB2 数据库
            pDB = new HYDB2(); break;
    }
}
```

经过初始化以后,服务器端的 API 函数 SDEFx(HYDB * pDB, NxStream * pStream, HYLog * log) 将会调用 HYDB 的子类所实现的接口函数。

在实际实现的过程当中, CoSDE 的服务器端接口大部分都使用原数据库的底层接口实现,如 HYOracle 类是使用 Oracle 的 OCI(Oracle Call Interface, C 语言调用接口)实现的, HYDB2 类是使用 DB2 数据库的 CLI(Call Level Interface, C 语言接口)实现的。HYMySQL 类是使用 MySQL 的 C 接口实现的。由于调用的都是底层接口,因而运行效率较高,同时还实现了跨平台。而 HYAdo 类是使用微软的 ADO(ActiveX Data Object)来实现的,它能保持与 ADO 同等的效率,缺点是只能运行于 Windows 平台上。

5 CoSDE 的测试效果



图3 广州遥感影像图的实时漫游

VegaGIS 使用基于 HYOracle 和 HYAdo(“MSSQL”)的两类空间数据引擎进行了大数据量的测试和测评,对数据量为
(下转第 1619 页)

不断变化的水波,并且车辆颜色和路面颜色很相似。(b)是分割结果,可以看到算法可以准确地从复杂变化的背景中检测和分割出运动的车辆。



图6 视频1分割结果



图7 视频2分割结果

图7(a)取自视频2,背景中的江面波浪使得背景颜色和形状呈无规律的变化。图(b)分割结果显示了算法对复杂背景变化的鲁棒性。

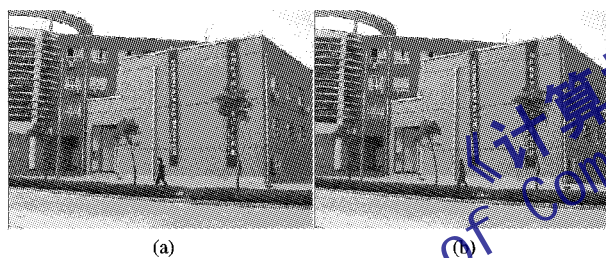


图8 视频3分割结果

图8(a)取自视频3,视频中含有的变化背景有:摇动的树枝和建筑物上的条幅。图(b)的分割结果显示算法对小的前景目标也具有较好的检测和分割性能。

参考文献:

- [1] AGGARWAL JK, CAI Q. Human Motion Analysis: A Review[J]. Computer Vision and Image Understanding, 1999, 73(3): 428 - 440.
- [2] SEKI M, FUJIWARA H, SUMI K. A Robust Background Subtraction Method for Changing Background [A]. Proceedings of IEEE Workshop on Applications of Computer Vision[C], 2000. 207 - 213.
- [3] SEKI M, WADA T, FUJIWARA H, et al. Background Subtraction

Based on Co-occurrence of Image Variations[A]. 2003 IEEE Computer Society Conference on[C], 2003, 2. 65 - 72.

- [4] YAO YS, CHELLAPPA R. Tracking a Dynamic Set of Feature Points[J]. IEEE Transactions on Image Processing, 1995, 4 (10): 1382 - 1395.
- [5] TOYAMA K, KRUMM J, BRUMITT B, et al. Wallflower: Principles and Practice of Background Maintenance[A]. Proceedings of IEEE International Conference on Computer Vision[C], 1999. 255 - 261.
- [6] LI L, HUANG WM, GU IYH, et al. Foreground Object Detection in Changing Background Based on Color Co-occurrence Statistics [A]. Proceedings IEEE Workshop on Application of Computer Vision[C], 2002. 269 - 274.
- [7] MATSUYAMA T, OHYA T, HABE H. Background Subtraction for Non-Stationary Scenes[Z]. Department of Electronics and Communications, Graduate School of Engineering, Kyoto University, 1999.
- [8] OLIVER NM, ROSARIO B, PENTLAND NP. A Bayesian Computer Vision System for Modeling Human Interactions[J]. IEEE Transactions on PAMI, 2000, 22(8): 831 - 843.
- [9] MATSUYAMA T, HABE H, YUMIBA R, et al. Background Subtraction under Varying Illumination[A]. Proceedings of 3rd International Workshop on Cooperative Distributed Vision[C], 1999. 225 - 246.
- [10] NAKAI H. Non-Parameterized Bayes Decision Method for Moving Object Detection[A]. Proceedings of Asian Conference on Computer Vision[C], 1995, 3. 447 - 451.
- [11] UKIDA H, KONISHI K, WADA T, et al. Recovering Shape of Unfolded Book Surface from a Scanner Image using Eigenspace Method[A]. Proceedings of IAPR Workshop on Machine Vision Applications[C], 2000. 463 - 466.
- [12] GRIMSON WEL, STAUFFER C, ROMANO R, et al. Using Adaptive Tracking to Classify and Monitor Activities in a Site[A]. Proceedings of CVPR[C], 1998. 22 - 29.
- [13] HARITAOGU K, HARWOOD D, DAVIS LS. A Fast Background Scene Modeling and Maintenance for Outdoor Surveillance[A]. Proceedings of ICPR[C], 2000, 4. 179 - 183.
- [14] STAUFFER C, GRIMSON W. Learning Patterns of Activity Using Real-time Tracking[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2000, 22(8): 747 - 757.
- [15] SETHI IK, JAIN R. Finding Trajectories of Feature Points in a Monocular Image Sequence[J]. IEEE Transactions on Pattern Analysis Machine Intelligence, 1987, 9 (1) : 56 - 73.
- [16] WANG Y, TAN T, LOE KF. Video Segmentation Based on Graphical Models[A]. Proceedings of IEEE Conference on Computer Vision and Pattern Recognition[C], 2003, 2. 335 - 342.

(上接第1589页)

12GB的800幅广州卫星遥感影像地图创建影像金字塔索引,使用CoSDE的客户端CoCatalog对该地图进行了实时的漫游,其效果如图3。

VegaGIS在数据装载的过程中,采用的是一次性装载,将这800幅影像图幅合成一个大图层。比较接近实际的应用要求。对比ESRI的ArcSDE,因为它没有图幅的数据组织,因此只能按逐个图幅载入,并且每个图幅载入数据库之后都形成一个新的图层,它只能对单个图幅载入以后的数据逐个建立金字塔,难以建立整个大图层的金字塔,因而也难以做到全图中的随意漫游。

参考文献:

- [1] 谭念龙. 空间数据存储技术及其应用[J]. 微电子学与计算机, 2002, (1).
- [2] [http://www.mapinfo.com/common/library/enterprise_whitepaper.pdf/\[EB/OL\]](http://www.mapinfo.com/common/library/enterprise_whitepaper.pdf/[EB/OL]), 2005 - 01.
- [3] [http://www.supermap.com.cn/\[EB/OL\]](http://www.supermap.com.cn/[EB/OL]), 2005 - 01.
- [4] ESRI White Paper, What Is ArcSDE[Z], 2003.
- [5] [http://www.mapgis.com.cn/service/article/index.htm\[EB/OL\]](http://www.mapgis.com.cn/service/article/index.htm[EB/OL]), 2005 - 01.
- [6] 宋克福, 钟尔顺, 程新荣, 等. 多源空间数据无缝集成(SIMS)技术研究[A]. 99'中国GIS年会论文集[C], 1999.