

基于 Sync Services for ADO.NET 的智能客户端应用研究

高艳宏, 马光思

(西安建筑科技大学 信息与控制工程学院, 陕西 西安 710055)

摘要:针对在智能客户端中数据同步存在的缺陷,依据具体的实例,分析了基于 .NET 平台下 Sync Services for ADO.NET 框架体系结构,描述了不同层次的功能,示意了数据下载的过程,阐述了数据同步和冲突处理的详细步骤,说明了遇到问题的解决方法。采用对比的方法,验证了基于 DataSet 与 SqlCeResultSet 内存消耗,及利用 LINQ 技术访问 SqlCeResultSet 数据源细节。同时,给出了基于角色的访问控制的安全设计。与传统的以数据为中心的同步方案相比,该方案具有松耦合、内存使用率高、数据访问智能化、安全可靠的特点。为智能客户端架构应用提供了技术参考且具有实用价值。

关键词:智能客户端;数据同步;冲突处理;语言集成查询

中图分类号:TP393

文献标识码:A

文章编号:1673-629X(2010)10-0224-04

Application Technology Research of Smart Client Based on Sync Services for ADO.NET

GAO Yan-hong, MA Guang-si

(School of Info. and Control Eng., Xi'an Univ. of Architecture & Tech., Xi'an 710055, China)

Abstract: To improve the defects of smart client in data synchronization, relying on a specific application case, analyzed architecture of Sync Services for ADO.NET under studying .NET platform, described function of different level, demonstrated process of data download, given process of data synchronization and conflicting data processing, illustrated solution of problem. By the method of antitheses, proposed an experimental analysis to memory consumption based on DataSet and SqlCeResultSet, in addition, the details of accessing SqlCeResultSet by LINQ technology was given. At the same time, gave safety design based on RBAC. Compared with traditional data-centric synchronization, it has a loosely coupled, high memory usage, intelligent data access, security. To smart client application, it has technical reference and actual value.

Key words: smart clients; data synchronization; conflicts processing; LINQ

0 引言

智能客户端(Smart Client)概念和技术的提出与迅速推广应用,得益于两个方面的动力:一是理论研究发现,可以综合运用 C/S 和 B/S 模式,发挥双方优势,以求互相取长补短^[1];二是结合实际,动态加载应用与更新数据,实现系统的在线及离线使用。智能客户端的体系结构由若干独立的应用块组成。这些应用块各司其职、互相协作,以完成系统的缓存、更新、离线、数据、安全、记录和异常处理等各项功能^[2,3]。实际应用日

益增加的数据需求,促使相关技术领域的研究不断深入,硬件的进步和 Sync Services for ADO.NET 框架(文中简称为 SSA)的推出,使得构建以服务为核心的移动智能客户端成为可能。SSA 支持 Web Service 技术。Web 服务是最适合实现 SOA 的技术^[4]。SOA 是服务的集合,服务是核心的抽象手段^[5]。文中采用 Visual Studio 2008 平台的仿真工具对有关技术进行研究与实现。

1 SSA 的数据同步处理策略

基于 SSA 多层架构的移动设备应用设计, Microsoft. Synchronization. Data. Server. dll 在中间层^[6]。实现客户端与中间层的通信,还需为客户端同步代理 RemoteProvider 属性,指定服务器同步提供程序代理实例,在实例中指定中间层服务的统一资源定位符。

收稿日期:2010-02-06;修回日期:2010-05-15

基金项目:陕西省教育科研项目(07JK306)

作者简介:高艳宏(1980-),男,硕士,研究方向为软件框架技术及应用、面向对象分析与设计;马光思,教授,研究方向为软件理论、软件工程、网络安全。

具体实现参见 2.1 节的 Part4。中间层的服务公布服务器同步提供程序方法,执行服务器同步提供程序的方法,结果通过中间层返回客户端。图 1 是对下载过程的简单示意。上传数据与下载数据相似,差别是同步代理调用服务器端同步提供程序检索增量数据,并将其应用到服务器端数据库。

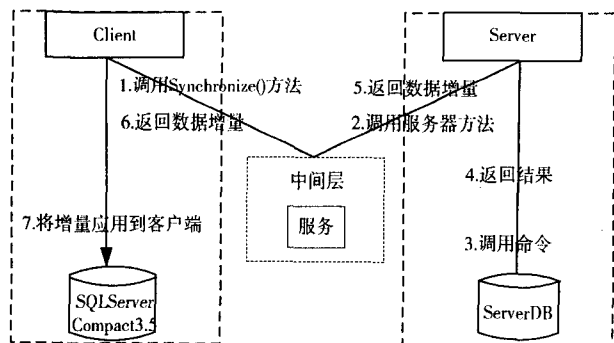


图 1 SSA 的数据同步示意

下载数据过程:

Step1: 客户端调用同步代理的同步方法 Synchronize()。

Step2: 中间层调用服务器同步提供程序的 GetServerInfo() 等方法。

Step3: 服务器调用同步适配器增加、删除、查找命令。

Step4: 将数据增量返回服务器端。

Step5: 将数据增量返回中间层。

Step6: 将数据增量返回客户端。

Step7: 同步代理调用客户端同步提供程序检索更改并应用到客户端数据库。

2 实现步骤、案例及遇到的问题

2.1 实现步骤

实现客户端与服务器之间的同步通信,需要完成以下 4 部分的工作。

Part1 同步适配器类 SyncAdapter: SyncAdapter 同步适配器为服务器同步提供程序提供了与服务器数据库交互所需的特定命令。

定义适配器基类 SyncAdapter 的 6 个子类,继承关系见图 2。每个类均指定增、删、改命令,指定选择增量的增、删、改命令并定义同步适配器表名。

下面的代码片段是产品同步表删除命令的例子,其余命令类似。

```
DeleteCommand.CommandText = "DELETE FROM
dbo.Product WHERE ([ProductID] = @ProductID)
AND (@sync_force_write = 1 OR ([LastEditDate] <
= @sync_last_received_anchor)) SET @sync_row_
```

```
count = @@rowcount";
```

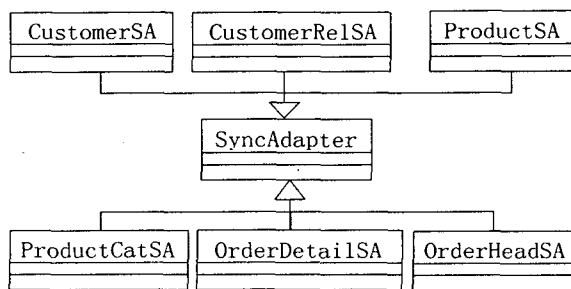


图 2 SyncAdapter 与子类继承关系

Part2 ServerSyncProvider 类: 检索更改,将增量更改应用与服务器数据库。

定义 MyDBServerSyncProvider 类继承于 DbServerSyncProvider。包括私有同步适配器属性,通过访问器获取或设置属性值。定义初始化同步适配器方法,添加同步适配器类到服务同步提供程序的同步适配器集合。在构造函数中,读取配置文件连接字符串值赋给服务器同步提供程序连接属性,调用初始化同步适配器方法,然后读取从服务器数据库返回新定位点值的查询,作为上传或下载增量数据依据。

Part3 客户端同步提供程序类 ClientSyncProvider: 为客户端数据库指定连接字符串。

定义 MyDBClientSyncProvider 继承客户端同步提供程序。客户端数据库连接字符串为 ConnectionString = "Data Source = " + (Path.GetDirectoryName(Assembly.GetExecutingAssembly().GetName().CodeBase) + @"\SaleDB.sdf;"

Part4 同步代理类 SyncAgent。

定义 MyDBSyncAgent 继承同步代理。定义 SyncTable 表,设置表名、同步方向、创建选项,并将其添加到同步代理同步表集合中,语句为 this.Configuration.SyncTables.Add(SyncTable)。分别指定 SyncAgent 的 LocalProvider 和 RemoteProvider 属性:其语句为 service = new SmartClient.localhost.Service(); service.Url = service.Url.Replace("localhost", "Machine name"); syncAgent.RemoteProvider = new ServerSyncProviderProxy(service); 和 this.LocalProvider = new MyDBClientSyncProvider();

2.2 遇到的问题及解决方法

(1) 调用 Synchronize() 时出现的参数异常 (argument exception)。

解决方法: 打开 Web 引用,注释 local host 中的类和枚举: SyncAnchor、SyncGroupMetadata、SyncTableMetadata、SyncTableInfo、SyncSession、SyncServerInfo、SyncConflit、ConflictType、SyncStage、SyncTablePro-

gress、SyncGroupProgress、SyncContext、SyncParameter、SyncDirection。

(2)找不到 Pinvoke DLL“sqlceme35.dll”。

解决方法:1.将 D:\Program Files\Microsoft Visual Studio 9\SmartDevices\SDK\SQLServer\Mobile\v3.0\wce500\armv4i 目录下的 NETCFv35.wm.armv4i.cab、sqlce.dev.CHS.wce5.armv4i.CAB、sqlce.repl.wce5.armv4i.CAB、sqlce.wce5.armv4i.CAB 拷贝到设备并安装。2.将 System.Data.SqlServerCe 是否复制到本地设备 false。

3 访问离线数据

智能客户端系统最大的优势就是可以离线使用^[7]。智能客户端通常需要在本地缓存数据。无法访问网络资源时,通过缓存数据,提供应用脱机工作所需数据,保证应用正常运行^[8]。

3.1 数据源:SqlCeResultSet

通常使用 DataSet 作为数据源访问数据库。DataSet 将数据存储在不同表中,在表之间建立关联关系,操作类似于关系型数据库。

SqlCeResultSet 不存储数据,可以直接读取数据,也可以直接更新数据,是一个轻量型的数据访问对象。利用 .NET Compact Framework Power Toys 3.5 软件中的 .NETCF CLR Profiler 组件,对 DataSet 和 SqlCeResultSet 消耗内存所做定量分析结果参见表 1 和表 2。通过实际的数据可以得到 SqlCeResultSet 比 DataSet 消耗的内存少。在内存有限的移动设备中,访问本地数据,DataSet 内存开销大,而 SqlCeResultSet 能节省内存资源。

表 1 DataSet 内存消耗情况

Allocated bytes:	546 964
Relocated bytes:	0
Final Heap bytes:	546 692
Objects finalized:	0
Critical objects finalized:	0
Total collections:	0

表 2 SqlCeResultSet 内存消耗情况

Allocated bytes:	4 740 196
Relocated bytes:	2 283 476
Final Heap bytes:	2 836 128
Objects finalized:	42
Critical objects finalized:	0
Total collections:	4

3.2 查询技术

LINQ 是 Visual Studio 2008 和 .NET Framework 3.5 中一项突破性的创新,在对象和数据领域之间架起了一座桥梁^[9]。统一了各种数据源查询方式,查询语法类似 SQL;在编译时进行语法和类型检查,支持 IntelliSense。使用 LINQ 可以大量减少查询及操作数据库或数据源中数据使用的代码,并可在一定程度上避免 SQL 语句注入,从而提高应用程序安全性。

3.3 实现过程及案例

下面的代码段是结合 SqlCeResultSet 和 LINQ 查询商品目录名的例子:

```
SqlCeResultSet categoryResult;
categoryResult = com.ExecuteResultSet(ResultSetOptions.Sensitive
| ResultSetOptions.Scrollable);
var r = from SqlCeUpdatableRecord categoryRow in categoryRe-
sultSelect categoryRow["CategoryName"];
this.cbbGoodsCat.DataSource = r.ToList();
```

4 数据冲突处理策略

数据同步关键技术也包括数据的一致性^[10]。在任何同步方案中,只要在多个节点上变更数据,便会发生数据冲突。显然,在双向同步的情况下会产生冲突,注意,下载或上载时,也有可能产生冲突。

数据冲突处理过程如下:

Step1:当客户端或服务端应用行数为 0 时执行;

Step1.1:服务器端根据不同冲突类型在 ApplyChangeFailed 事件中做相应处理;

Step1.2 若服务器端覆盖客户端数据,Actio 设置为 Continue; Action 设置为 RetryWithForceWrite。

4.1 避免冲突

通过筛选,可以避免冲突的发生。商品配送员只需要下载与其相关的客户信息,其语句如下:

```
string customerFilterClause = "SalesPerson = @SalesPerson";
customerBuilder.FilterClause = customerFilterClause;
SqlParameter filterPar = new SqlParameter ("@ SalesPerson",
SqlDbType.NVarChar);
customerBuilder.FilterParameters.Add(filterPar);
string namePerson = name;
Configuration.SyncParameters.Add(new SyncParameter ("@ Sales-
Person", namePerson));
```

4.2 解决冲突

避免冲突可以减少冲突,但冲突产生时,需要根据冲突类型,决定忽略冲突并继续执行同步还是重新尝试逻辑以强制应用变更,使客户端与服务器数据保持一致。冲突类型为客户端删除同时服务端更新、客户

端与服务端同时插入、客户端与服务端同时更新, Action 属性设置为 Continue, 忽略冲突并继续执行同步; 类型为客户端更新的同时服务端删除, Action 属性设置为 RetryWithForceWrite, 重新尝试逻辑以强制应用变更。

5 权限设计实现

安全模块是智能客户端应用的重要部件。基于角色的访问控制 (Role - Based Access Control, RBAC) 逐渐取代了其他两种访问控制^[11,12]。把基于角色的访问控制应用到安全模块的设计, 对加强安全控制管理有一定的积极作用。这里的角色包括: 系统管理员、派送员及一般用户。系统管理员依据角色为用户分配权限, 以确保不同用户严格按照自己拥有的权限执行相应操作, 维护系统安全^[13]。例如, 对移动购物应用可设计如图 3 所示的权限管理机制。

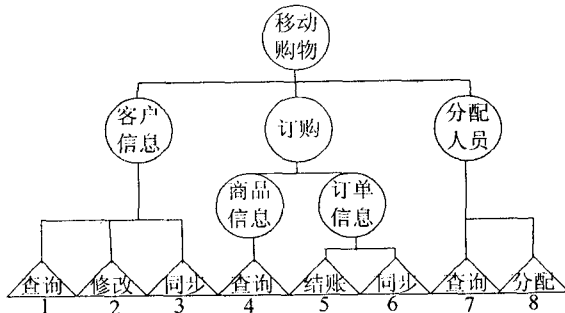


图 3 权限管理

其中, 派送员操作 1、2 和 3; 一般用户只能操作 4、5、6; 系统管理员有权操作全部, 包括更新用户角色和权限。User 表 Right 字段可以标识操作权限。普通用户、派送员、系统管理员 Right: “00111000”、“00000111”、“11111111”。应用依据 0 和 1 值设置按钮 Enabled 属性, 控制按钮是否可用。

6 结束语

课题研究基于 Visual Studio 2008 平台, 应用部署

于 CHS Windows Mobile 6 Classic Emulator 模拟器。通过设置同步方向、数据表、数据项、命令等, 加强了对应用的控制。进一步的研究和实践, 是对各功能的改进、利用并移植到真实设备。

参考文献:

- [1] 徐 毅. 基于智能客户端的工作流模型设计[J]. 计算机仿真, 2008(12): 281 - 284.
- [2] 王焕彬, 夏靖波. 智能客户端技术在军用装备管理中的应用研究[J]. 微计算机信息, 2008(8): 8 - 11.
- [3] 朱 涛, 张水平, 李云云, 等. 基于智能客户端架构的自助服务系统的设计与实现[J]. 计算机工程, 2007(16): 205 - 208.
- [4] Kuayk R. Web services: Standardizing EAI[J]. eAiJournal, 2002(4): 23 - 25.
- [5] Anagnostopoulos C, Vergados D, Kayafas E, et al. A computer vision approach for textile quality control[J]. The Journal of Visualization and Computer Animation, 2001, 12(1): 31 - 44.
- [6] Microsoft 公司. MSDN 技术资源 [DB/OL]. 2008. <http://msdn.microsoft.com/en-us/library/bb384572.aspx>.
- [7] 索红光, 王雷全. 智能客户端系统中数据同步策略的研究与实现[J]. 计算机工程与设计, 2007(2): 351 - 354.
- [8] 彭玉卓, 杨开英, 马 俊. Smart Client 技术实现 MIS 系统中的离线应用[J]. 计算机技术与发展, 2007, 17(3): 200 - 203.
- [9] 侯利军. LINQ 数据访问技术—基于 C# [M]. 北京: 人民邮电出版社, 2008.
- [10] Barbara D, Imielinski T. Sleepers and workaholics: Caching strategies in mobile environments[J]. Journal of VLDB, 1995, 4(4): 567 - 602.
- [11] Sandhu R, Coyne E, Feinstein H, et al. Role - Based Access Control Model[J]. IEEE Computer, 1996, 29(2): 38 - 47.
- [12] 陈启泓, 邹 杜, 艾 飞, 等. 基于 RBAC 的限制约束在权限控制中的实现[J]. 微计算机信息, 2009(7): 3 - 6.
- [13] 蒋 伟. Ajax 在整合框架中的研究与应用[D]. 西安: 西安建筑科技大学, 2007.

(上接第 223 页)

- [J]. 计算机工程, 2009, 35(9): 40 - 42.
- [29] Simonas S, Christian S, Scott J, et al. Indexing the Positions of Continuously Moving Objects [C]//In: Proceedings of the 2000 SIGMOD International Conf. on Management of Data. Dallas, TX, USA: [s. n.], 2000: 331 - 342.
- [30] Mario A N, Jefferson R, Silva O. TOWARD HISTORICAL R - TREES [C]//In: The ACM Symposium on Applied States. [s. l.]: [s. n.], 1998: 235 - 240.
- [31] Yufei T, Dimitris P. The MV3R - Tree: A Spatio - Temporal

Access Method for Timestamp and Interval Queries [C]//In: Proceedings of the 27th VLDB Conference. Roma, Italy: [s. n.], 2001: 431 - 440.

- [32] Guo Jing, Guo Wei, Zhou Dongru. Indexing of Constrained Moving Objects for Current and Near Future Positions in GIS [C]//In: Proceedings of the 1st International Multi - Symposiums on Computer and Computational Sciences (IMSCCS' 06). [s. l.]: [s. n.], 2006: 504 - 509.
- [33] 张明波, 陆 锋, 申排伟, 等. R 树家族的演变和发展[J]. 计算机学报, 2005, 28(3): 289 - 300.