信息科学与工程学院

物联网平台与标准

实验报告

姓　　名：_____

学　　号：_____

专业班级：物联网工程

指导教师：**

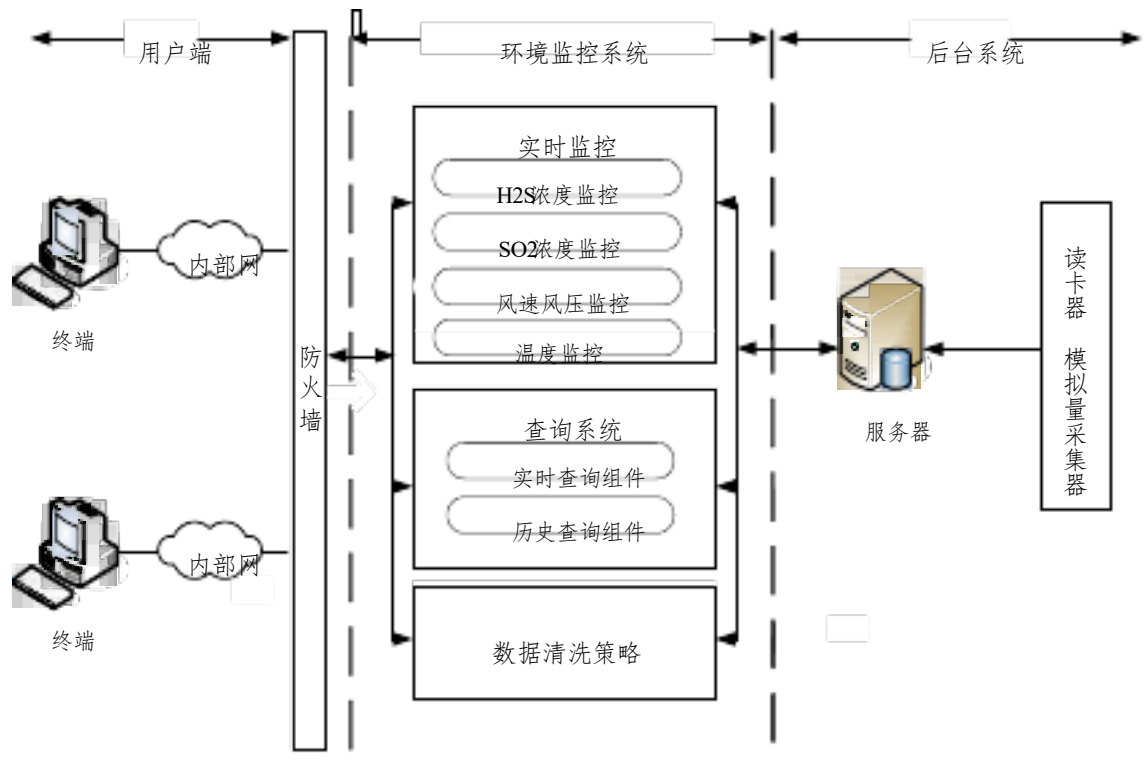
完成时间：_____

目录

实验一 物联网中间件体系结构分析

1.1实验背景

环境监控系统利用传感器实时采集环境信息（如温度、风速、风压等），再通过监控软件处理传感器采集到的数据，可以实现对环境的监控和报警。其体系结构如下图所示：



本次实验是为后续的实验室环境监控系统提供数据库支持。数据库设计了两个核心表：T_TAG_DATA 和 T_TAG_INDEX，其中 T_TAG_DATA 用于存储传感器实时采集到的数据，T_TAG_INDEX 用于记录传感器相关信息。

表 T_TAG_INDEX 字段定义如下：

序号	字段名	字段类型	说明	备注
1	TAG_ ID	char（20）	TAG 标识	主键
2	TAG_NAME	varchar（40）	TAG 名称	
3	TYPE_ID	char（4）	数据类型标识	
4	UNIT	char（10）	单位	
5	MIN	decimal（15,4）	最小值	
6	MAX	decimal（15,4）	最大值	
7	SAVE_RATE	int	存储频率	
8	DEFAULT_V AL	decimal（15,4）	缺省值	
9	REMARK	varchar（100）	TAG 描述	
10	IS_ALARM	bit	是否报警	

表 T_TAG_DATA 字段定义如下：

序号	字段名	字段类型	说明	备注
1	TAG_ ID	char（20）	TAG 标识	主键、外键
2	GET_DATE	datetime	取值时间	主键
3	TAG_VALUE	decimal（15,4）	TAG 取值	
4	QUALITY	char（1）	质量码	

1.2实验目的

1、了解物联网中间件体系结构；

2、掌握使用 PowerDesigner 建立数据库模型；

3、掌握使用 SQLServer2008 建立数据库。

1.3 实验环境

开发工具：

PowerDesigner 12.5

MS SQLServer 2008

1.4 实验内容

1、阅读物联网中间件体系结构设计说明；

2、使用 PowerDesigner 建立核心数据模型；

3、在 MS SQLServer 2008 中基于数据模型建立数据库。

1.5 实验步骤

使用 PowerDesigner 建立核心数据库模型：

1、启动 PowerDesigner

在安装了 Powerdesigner 的计算机上，点击 PowerDesigner 图标，即可启动 PowerDesigner 开发工具。

2、建立数据库物理模型

（1）在打开的 PowerDesigner 中，点击 File——>new，选择建立物理模型。选择 Table，在页面上添加两个表。

（2）按照实验背景中描述的数据库物理模型，对每一个表进行设计。将鼠标放在每一个表上，单击右键属性，选择 Columns 栏，输出表内的信息。

（3）再选择 Keys 栏，给表 T_TAG_DATA 添加主键，同样的方法给 T_TAG_INDEX 添加主键。

（4）给表格 T_TAG_DATA 添加外键。

3、建立数据库

打开 SQLServer 2008，在数据库中创建名为 IOTDataBase 的数据库。

4、添加数据源

（1）控制面板——>管理工具——>数据源（ODBC），添加新的数据源，如下图所示：

（2）选择 SQL Server，如下图所示：

（3）选择服务器，添加数据源名称。

（4）更改默认数据库。

（5）测试数据库连接。

（6）测试连接成功后，获得的结果。

5、生成数据库表

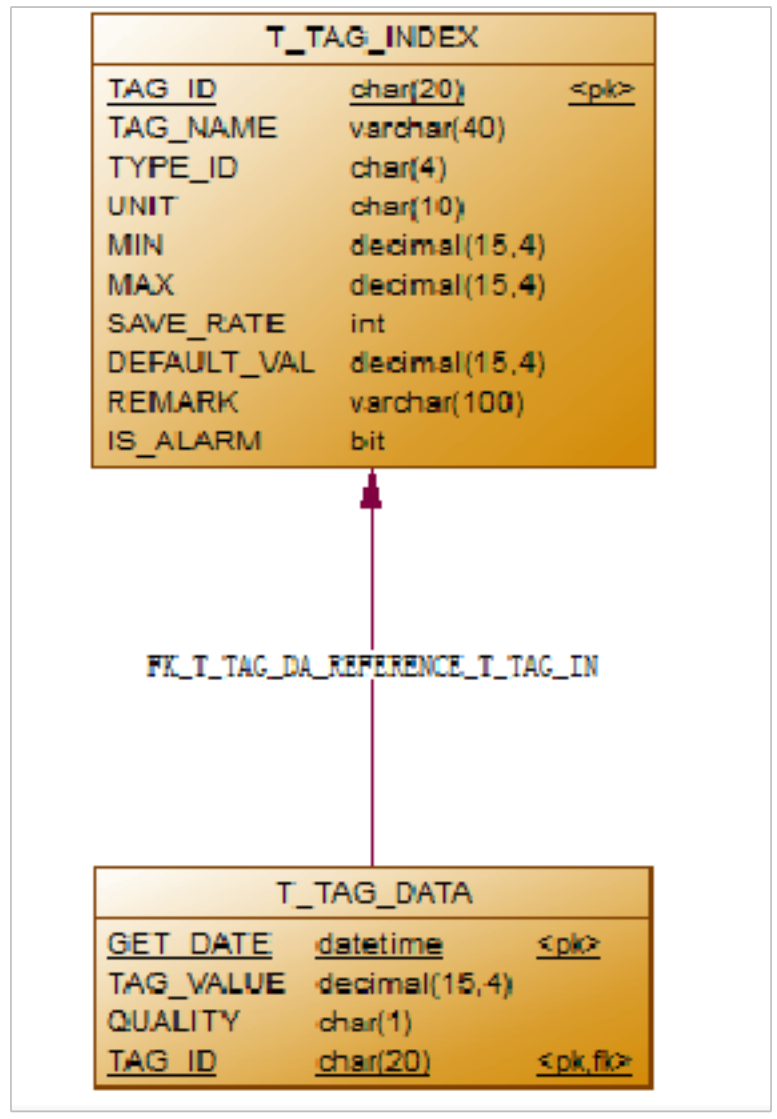
（1）在 PowerDesigner 工具栏中点击 DataBase——>Generation DataBase，选择数据源（IOTDataBase），输入数据库用户名密码。

（2）预览 SQL 脚本。

（3）点击“Run”，在 SQL Server2008 中即可查看产生的表。

1.6 实验结果

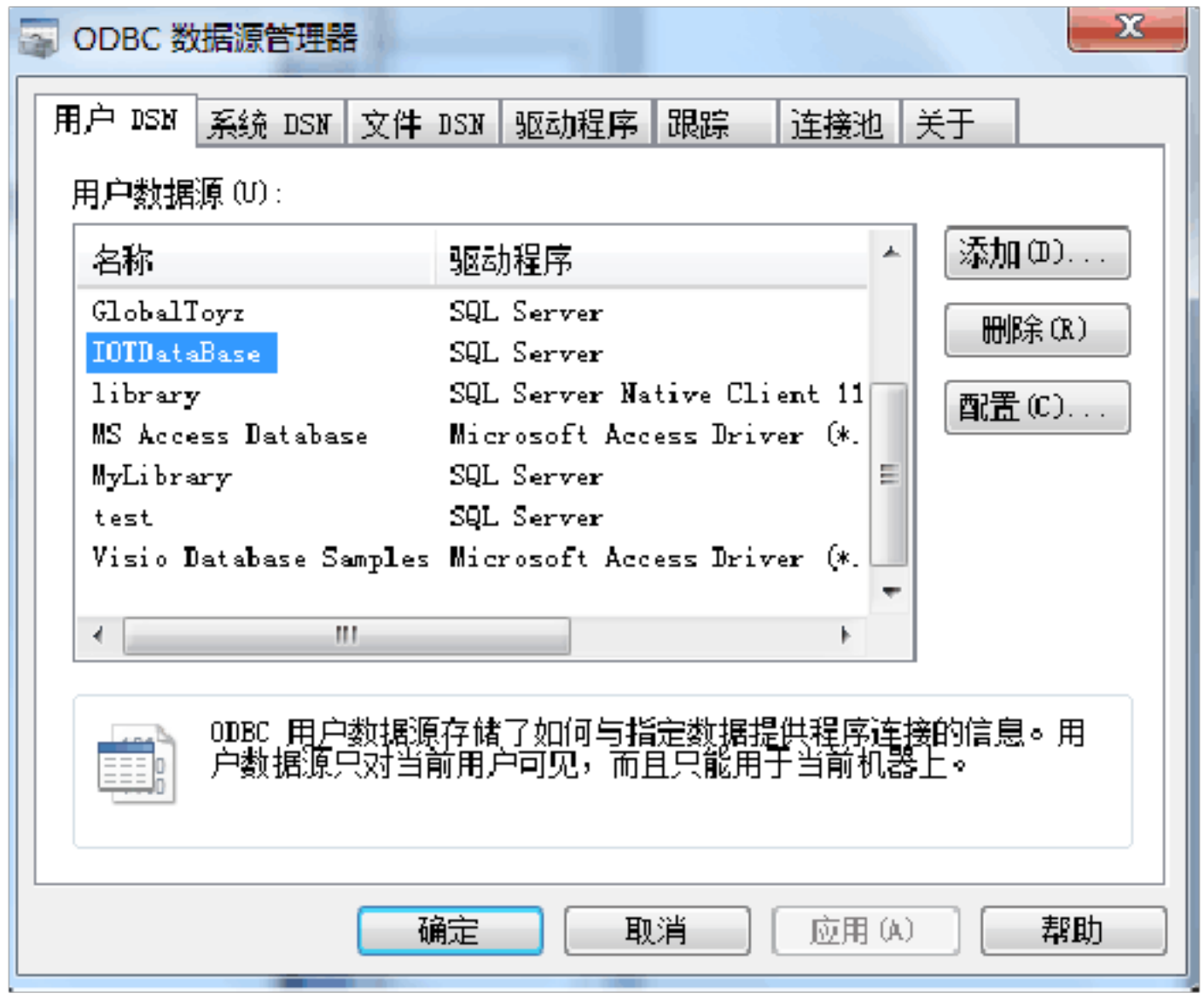
1、使用 PowerDesigner 建立物理模型如下图所示：



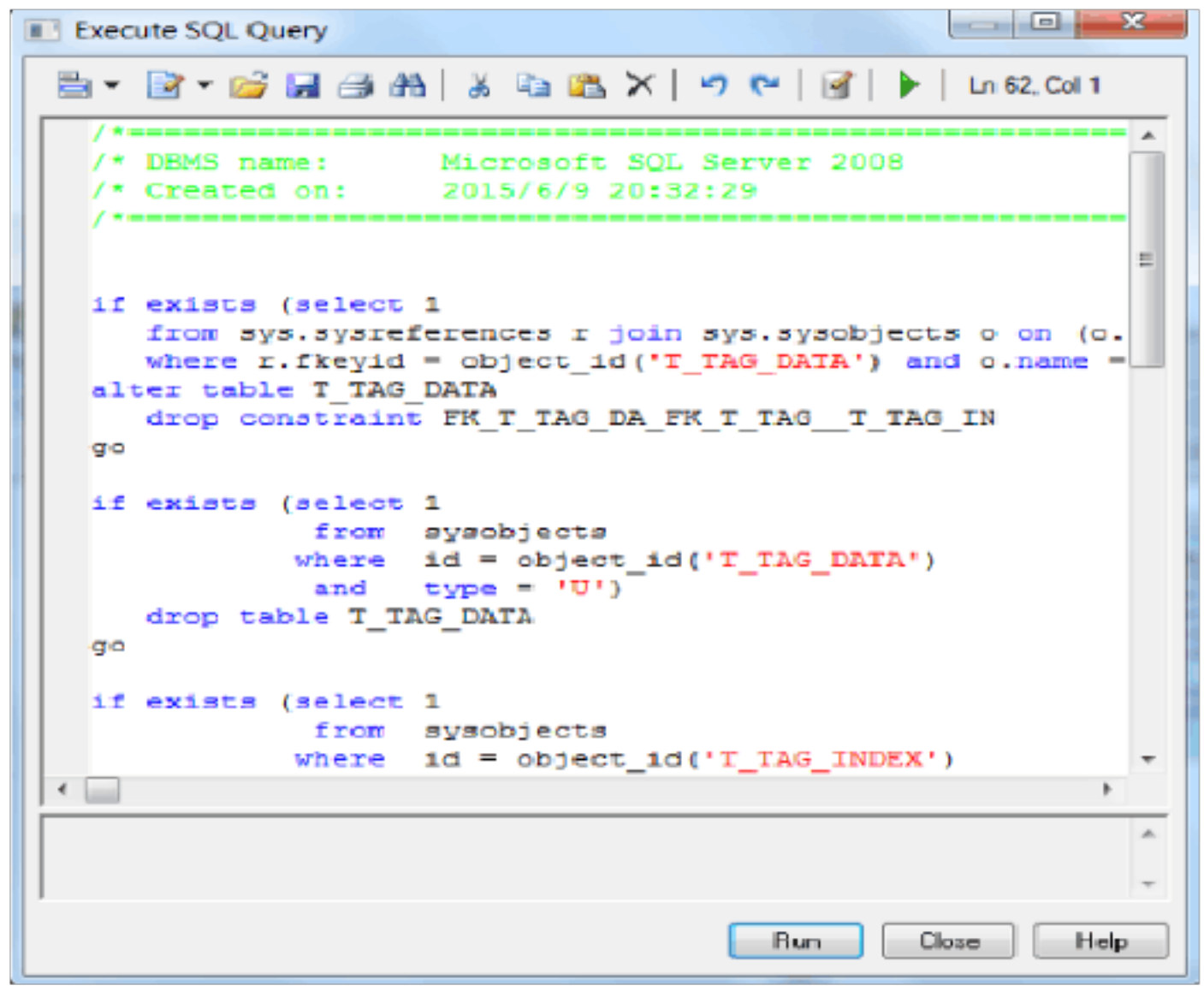
2、添加数据源，连接成功时截图如下：


ODBC Microsoft SQL Server 安装
将按下列配置创建新的 ODBC 数据源:
Microsoft SQL Server ODBC 驱动程序版本 06.01.7601
数据源名称: IOTDataBase
数据源描述:
服务器: LOUYW7Y3MUF8WV3
数据库: IOTDataBase
语言: (Default)
翻译字符数据: Yes
日志长运行查询: No
日志驱动程序统计: No
使用区域设置: No
预定义的语句选项: 在断开时删除临时存储过程
使用故障转移服务器: No
使用 ANSI 引用的标识符: Yes
使用 ANSI 的空值，填充和警告: Yes
数据加密: No
测试数据源(T)...
确定
取消

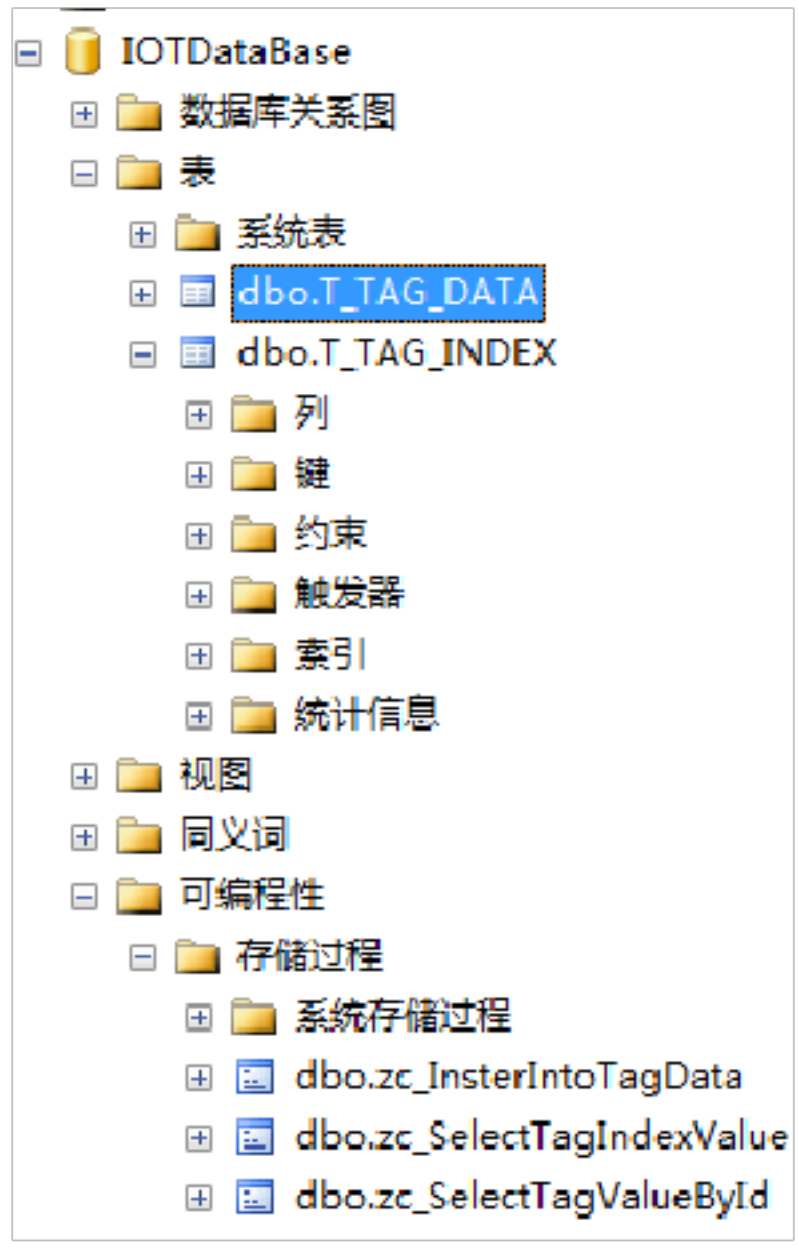

SQL Server ODBC 数据源测试
测试结果
Microsoft SQL Server ODBC 驱动程序版本
06.01.7600
正在运行连接测试...
试图连接
连接被建立
验证选项设置
断开与服务器的连接
测试成功!
确定



3、查看生成的数据库表，如下图：



Category	Check	Object	Location
Table	Existence of index	Table 'T_TAG_...	<Model>
Table	Existence of index	Table 'T_TAG_I...	<Model>



实验二 物联网中间件的创建及数据清洗

2.1 实验背景

近年来，随着物联网概念提出以及 RFID 射频识别技术在超市供应链管理上的成功运用，使得以往仅用于军事领域的 RFID 技术，凭借其成本低、体积小、无须接触等特性越来越多地运用在生产自动化、门禁、公路收费、货物跟踪等民用领域和矿山的人员定位系统中。然而，由于受到各种环境的影响和射频技术本身的一些特点，RFID 阅读器识别标签时会出现各种错误识别现象，例如积极读、消极读、标签冗余、阅读器冗余等，这些因素都制约着 RFID 技术应用进一步发展。在传感器中同样可能产生很多脏数据，例如错读，漏读现象。

对上述几种脏数据，可以分别采用相应的策略加以清洗。由于原始数据流具有无限性，因此在清洗算法中使用了滑动窗口技术。其中，平均值策略用于消除数据冗余，即将一分钟的数据进行平均，将平均值来取代这一分钟内产生的多个数据，这样使得存入数据库的数据量大大减少，当用户做历史数据查询时，也能体现出数据变化的大致趋势。 移动平均值策略对数据错读现象进行平滑处理，将该数据和该数据的前后几次数据求平均值，在很多情况下这个平均值能更好地

体现出这个时间点在时间段中的数据。滑动窗口策略则可以消除积极读的现象，消除积极读实质上是去除原始数据流中的噪音，算法通过统计滑动窗口中的标签个数来判断标签是否为噪音。

2.2 实验目的

1、了解物联网中间件数据清洗的意义；
2、了解物联网中间件的开发；
3、掌握物联网中间件处理数据的流程；
4、掌握物联网中间件数据清洗的几种基本策略；
5、掌握物联网中间件实时数据预警的方法。

2.3 实验环境

开发工具： Visual Studio 2010 + SQLServer 2008
开发语言： C#

2.4 实验内容

2.4.1 具体内容

1、物联网中间件的开发；
2、编写物联网中间件数据清洗策略；
3、为物联网中间件添加实时数据预警功能。

2.4.2 内容详解

1、本实验设计的物联网中间件包括以下 5 个功能：
（1）初始化工作，包括初始化一级缓存和 8 小时缓存；
（2）数据采集工作，采集由 OPCserver 产生的数据；
（3）数据处理工作，采用适当的清洗策略对数据进行清洗处理；
（4）数据转存工作，将处理过的数据存入到数据库和 8 小时缓存中。
（5）对外接口，供客户端应用软件调用。
2、使用的数据清洗策略包括：
（1）平均值策略：用一分钟产生数据的平均值取代这一分钟产生的所有数据，这样可以在不影响数据的大致走势下减少数据冗余、减轻数据库的压力。
（2）滑动窗口策略：定义一个固定大小的时间窗口，窗口每次滑动一个读取

如果窗口内的标签 Tagi 次数大于预先设定的最小阈值 Min，即认为标签 Tagi 为非噪声标签，则该窗口就不会输出标签 Tagi 。滑动窗口策略可以给数据去噪。

（3）移动平均值策略：用该数据的前几次数据和后几次数据加上该数据本身的平均值去取代该数据的值，从而消除数据的随机性。

2.5 实验步骤

2.　　连接数据库

1、创建解决方案

打开 Visual Studio 2010，选择 C#语言，新建一个类库程序，修改工程名。

2、数据库相关步骤

因为中间件要对采集的数据进行初始化工作,并且要把处理之后的数据转存到数据库中，所以必须对数据库进行操作。下面新建一个项目来对数据库进行操作。

（1）新建项目，右击解决方案，添加新建项目，命名为：GetDataAccess。

（2）添加 SQLHelper 的项，这个类的作用是通过存储过程来对数据库进行操作，这里就不对这个类进行详解。首先将源码里面的 SQLHelper.cs 复制到刚刚新建的 GetDataAccess 文件夹下面，然后在 VS 解决方案中添加现有项，找到 SQLHeleper 并添加。

（3）新建一个名为 DBOperation 的类，加入下列代码，其中连接字符串中的 Server 一项的 IP 为数据库所在 IP 地址，password 一项为数据库的密码，这两项应该依实际情况做相应修改。

（4）在数据库中添加相应的存储过程。

2.5.2 中间件的创建及其清洗策略

1、新建 TagObject 类来封装中间件获取的数据信息。首先右击 MiddleWare 项目，找到添加中的新建项，将类名修改为 TagObject。

2、编写 TagObject，首先写入字段属性，对应传感器或者读卡器产生数据。 给字段属性添加 Set 和 Get 方法，下面只给出示例，其余属性相同处理。

3、按上述方法依次新建 RTDataRemoteObject 类（中间件类，负责数据采集、清洗及转存），TagCacheL1 类（一级缓存，保存所有传感器读卡器的实时值），CacheTagList 类（8 小时缓存，保存服务器运行 8 小时的值），DateObject 类（保 存上次采集时间和采集频率的类，用途是控制采集频率），

GetConfigFromXml XML 文件的操作，这里用来读取配置文件）。RTDataRemoteObject 类是最重要的类，负责数据的采集、清洗与转存，还提供对外的数据接口。下面分功能对其进行说明。

A ：导入包及定义变量

B ：启动函数，依次调用对数据进行处理的函数，实现中间件的功能，注意这里不能使用类名的初始化函数，因为不能要求建立中间件对象的时候，就要让它工作；

C ：准备工作，初始化一级缓存，作用是为采集、处理数据做初始化工作；

D：准备工作，初始化 8 小时缓存；

E：数据采集工作，采用线程每隔一秒钟执行一次采集函数，采集一组数据到缓存中；

F：移动平均值策略 #region 计算移动平均值；

G：平均值策略和滑动窗口策略，平均值策略即求 1 分钟的平均值，减少了 存入数据库中数据的冗余，又不影响数据的走势。而滑动窗口策略是为了给数据 去噪，去掉那些极可能是干扰的数据；

H：对外接口，供应用软件调用。

至此，中间件的编写就初步完成了。但是此时的中间件是不能独立运行的，因为还需要与 OPC Client 连接起来去 OPC Server 中取数据。

2.5.3 OPC Server 的安装和配置

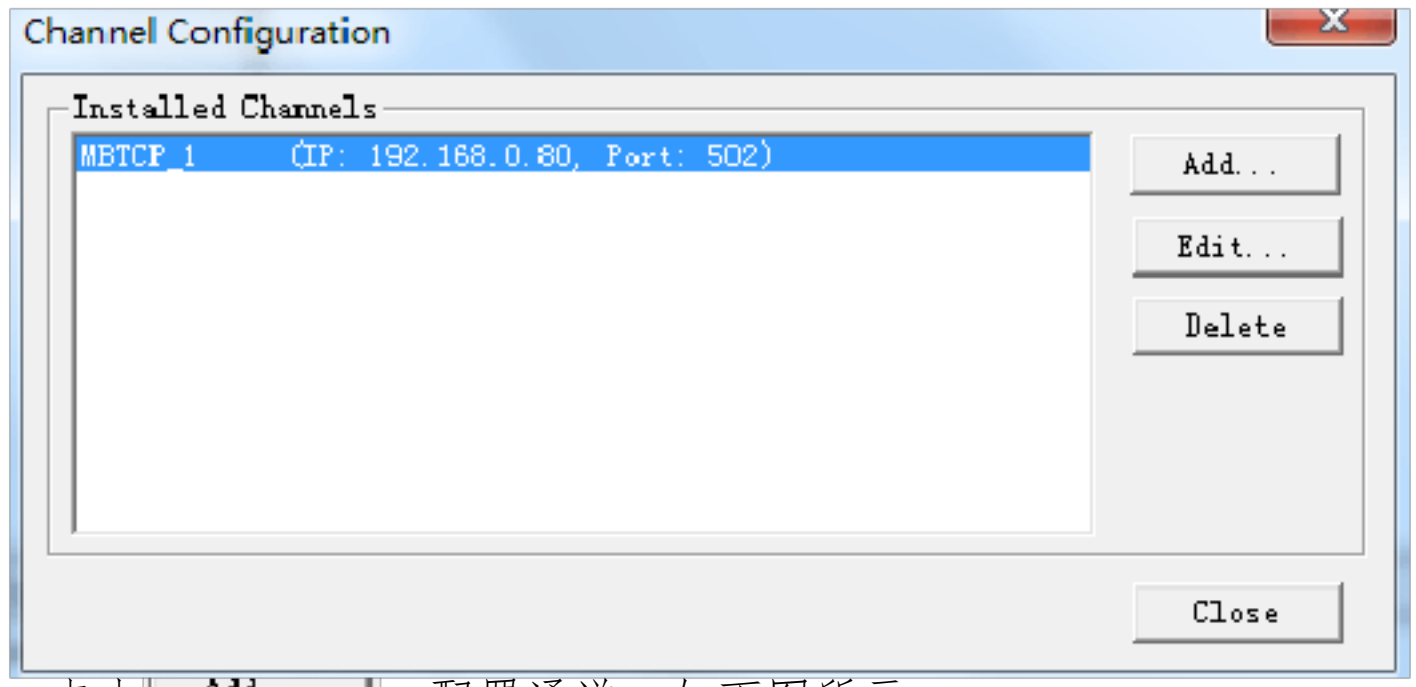
安装：

点击 OPC Server安装文件，安装即可。

配置：

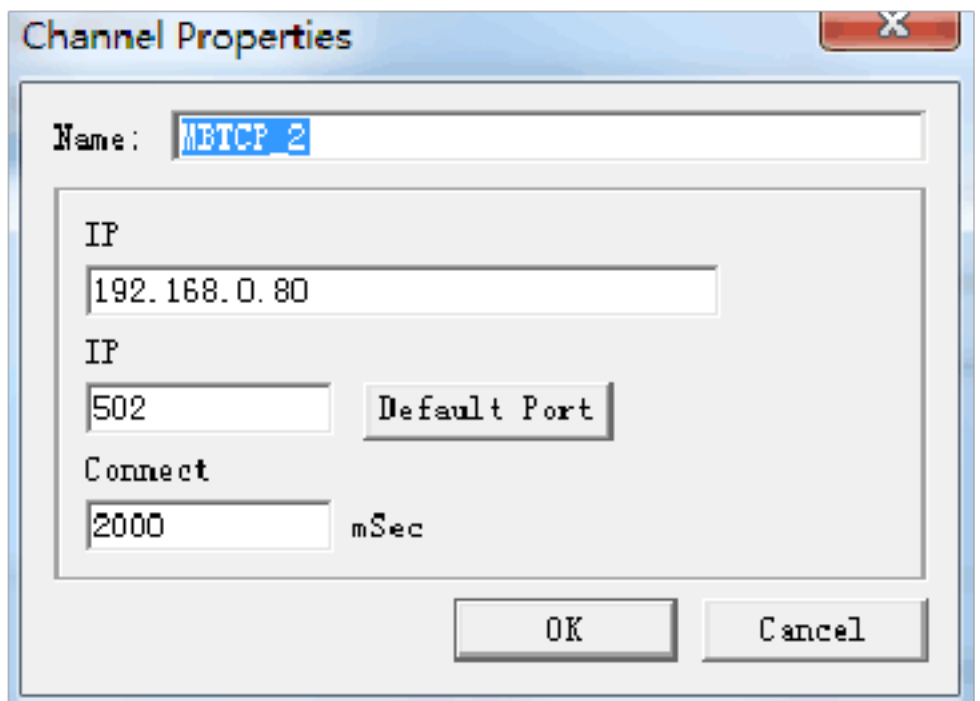
点击 

（Channel）添加一个新的通道，如下图所示：



点击

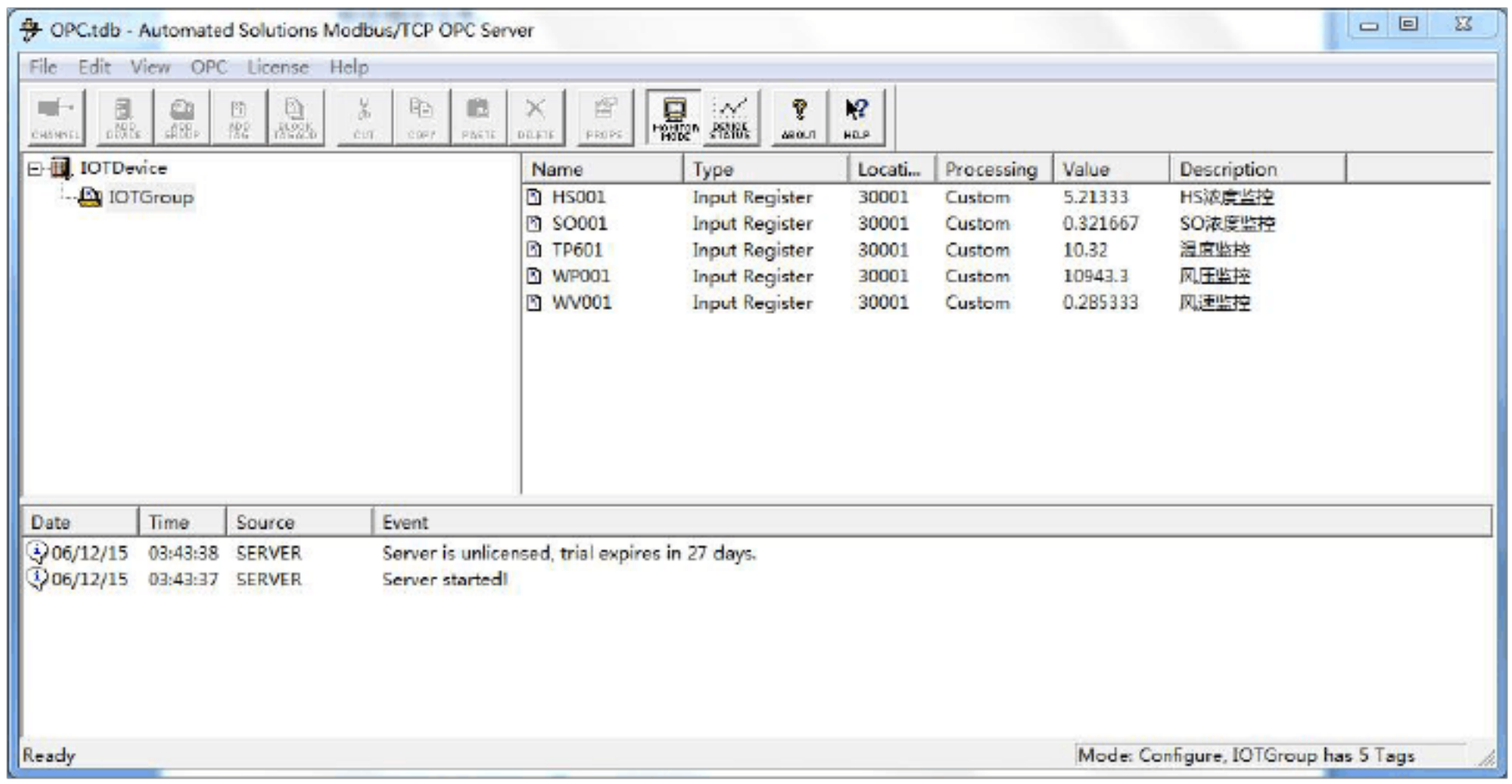
，配置通道，如下图所示：



，修改设备名称。因为是模拟产生监控数据，所以选择 Simulate I/O；添加组，点击，修改组名；添加标签，点击，配置标签相关信息；

重复添加若干标签，分别为：HS浓度监控、SO浓度监控、温度监控、风压监控、风速监控。

点击，使用监控模式，即可看到模拟数据的产生，如下图所示：



2.5.4 OPC Client 的开发

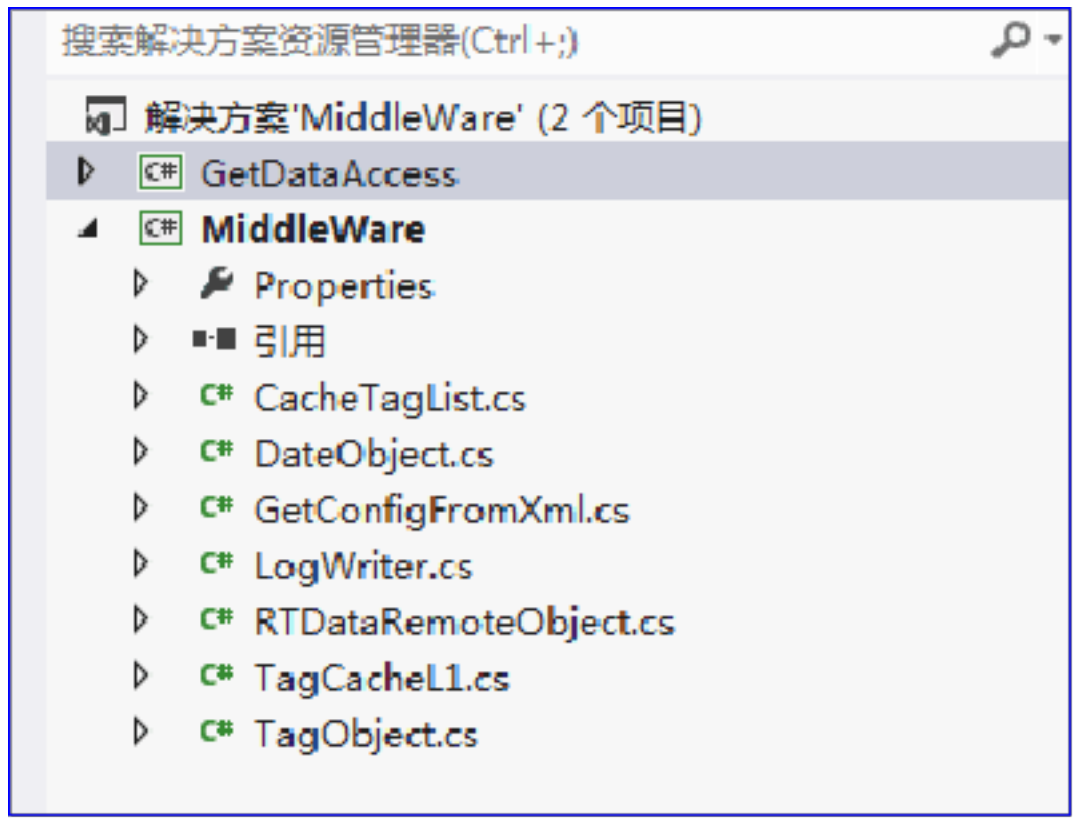
创建工程：

打开 Visual Studio 2010，选择 C#语言，新建一个 Windows 窗体应用程序，修改工程名和保存位置，在主窗口上添加若干控件；

控件名称	控件命名	说明
Label	Label1	显示提示信息“OPCServer: ”
	Label2	显示提示信息“OPCServerInformation: ”
TextBox	txtServer	显示 OPCServer ID
	txtServerinfo	显示 OPCServer信息
TreeView	treeOpcItems	显示 OPC Sever中各个项的信息
ListView	listOpcView	显示 Item 的基本信息
	itemDetailList	显示 Item 的详细信息
Button	viewHashTable	查看 Hashtable 中的信息，即传送给展现层的数据
StatusStrip	IOTServerStatus	OPC客户端状态栏，显示 OPC Server 其他的信息
Timer	timer	定时器，定时获取数据

2.6 实验结果

1、 MiddleWare 项目截图如下：



2、GetDataAccess 项目文件截图如下：