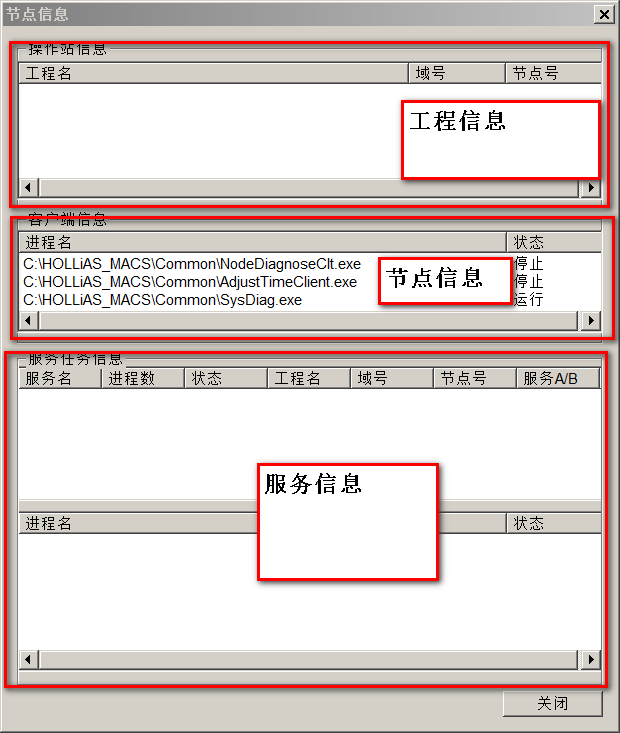
# NodeDaemon功能介绍

## 功能介绍

NodeDaemon是一个进程，也叫节点守护，或者叫“小球”，安装完MACS V6软件之后，电脑重启后，NodeDaemon进程就会自动运行，在电脑屏幕右下角托盘区圆球状的图标就是 NodeDaemon，如图：



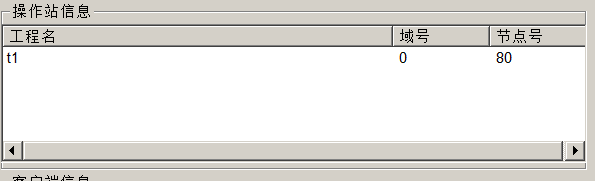
双击该图标弹出主界面



MACSV6系统中的每个节点都会运行NodeDaemon进程，包括工程师站，历史站，OPS节点。这个界面主要有三部分组成：工程信息栏，节点信息栏，服务信息栏。

### 工程信息栏：

工程信息栏主要用来显示当前节点的工程信息，工程名，域号，节点信息。如果该节点被下装为操作员，就会显示出这些信息，否则该栏为空，下图显示将该节点下装为操作员，工程名为“t1”，域号为0，节点号为80。



### 节点信息栏：

节点信息栏主要显示三个进程的运行状态。

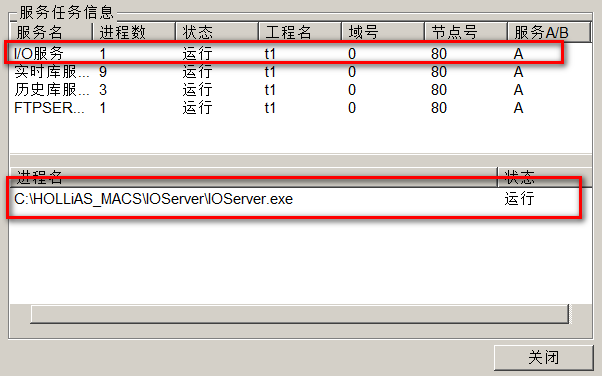
NodeDiagnoseClt（节点诊断客户端）：用于获取本节点的网络状态，网络负荷，CPU负荷，内存使用率，发送给节点诊断服务端。

AdjustTimeClient（校时客户端）：用于校时服务端获取时间，来校正本节点主机时间。

SysDiag：用于检测MACS V6软件的完整性，如果安装路径下某个exe，dll文件被修改或者损坏，则SysDiag会禁止打开工程总控。

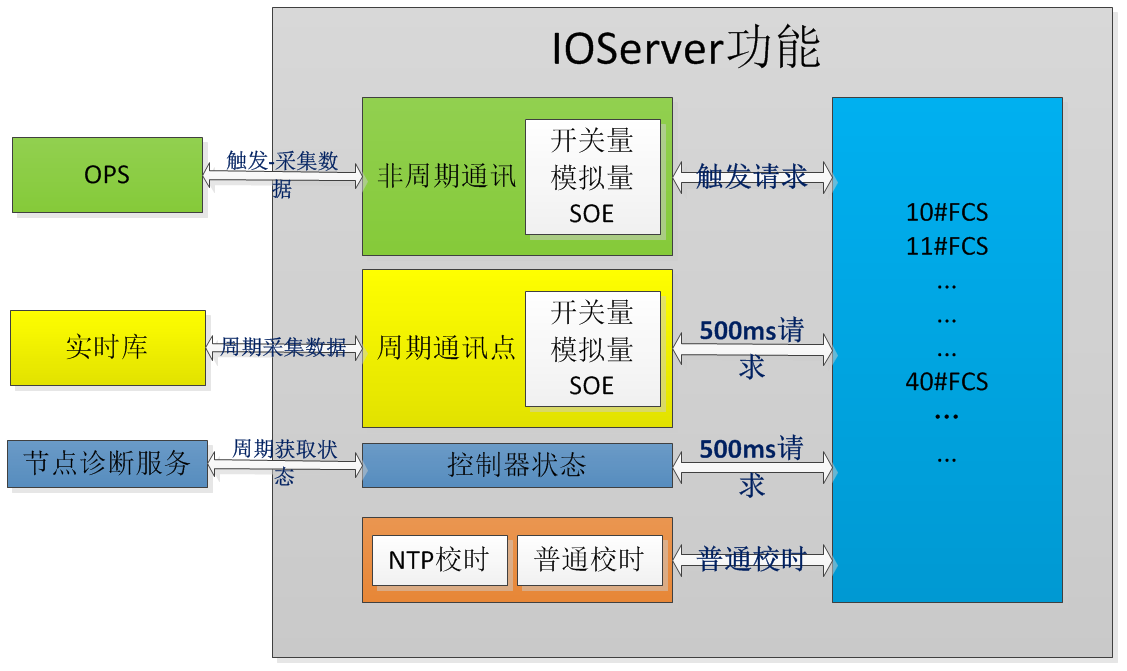
### 服务信息栏：

服务信息栏用来显示服务的运行状态，如果该节点被下装为历史站，服务信息栏里面就会显示服务的运行状态，主要有三组服务：IO服务，实时服务，历史服务，每组服务下面会运行多个进程。如果是历史站，这三组服务是必须运行的。NodeDaemon会监视进程，如果某组服务的某个进程异常退出了，NodeDaemon会重新启动该进程保证系统的正常运行，若这个进程一直没有运行，就说明系统出问题了。



## IO服务

IO服务下面只有一个进程 IOServer进程。IOServer的主要功能如下图：



IOServer涉及的功能主要有非周期点的通讯、周期点的通讯、状态包的获取，校时。

### 非周期通讯点

非周期通讯点是指需要触发一次读操作，IOServer才能从控制器采集数据的点。

非周期通讯点触发一次读操作（一般由OPS触发，如流程图页面，该页面上组态了非周期通讯点，则OPS触发一次读操作），会连续从控制器读10次数据，每次的间隔时间为500ms。如果连续读取了10次数据之后，没有再次触发读操作，那么该点就从非周期点表里面移除了；如果在10次数据之内又触发了一次读操作，则又会重新读取10次数据；如果一直触发读操作，则会一直从控制器读取数据。

间接通讯下，IOServer提供非周期点值给OPS上面组态的点。间接通讯是指OPS上的控制器点项值是从IOServer获取，而直接通讯是指OPS上面的控制器点项值是直接向控制器获取。

### 周期通讯点

周期通讯点是指每隔500ms IOServer都向控制器采集数据的点。周期通讯点不需要触发读操作，IOServer每隔500ms就会主动向控制器发出读请求。

间接通讯下，IOServer提供周期点值给OPS上面组态的点显示。

IOServer提供周期点值给实时库。

### 控制器状态

控制器状态是指IOServer每隔500ms向控制器去请求状态数据，主要包括控制器A机状态，A机A网状态，A机B网状态，B机状态，B机A网状态，B机B网状态等。

IOServer提供控制器状态给节点诊断服务使用。

### 校时

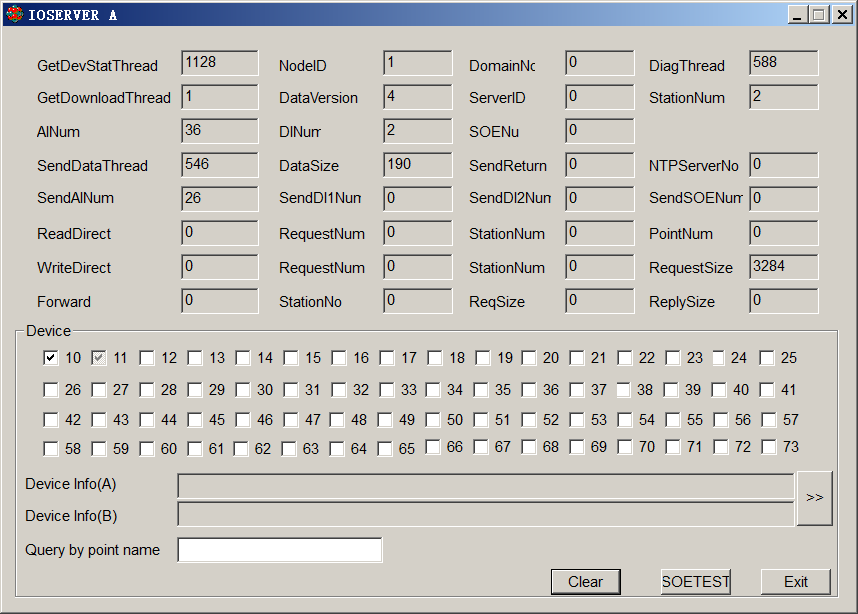
IOServer向控制器校时分两大类。一类为非标准NTP校时，一类为标准NTP校时。

非标准NTP校时。IOServer每隔1分钟将自己电脑的时间传给控制器，控制器用该时间进行校时。

标准NTP校时。IOServer每隔1分钟将NTP时钟源的IP地址传给控制器，控制器使用该IP地址向NTP时钟源拿时间。

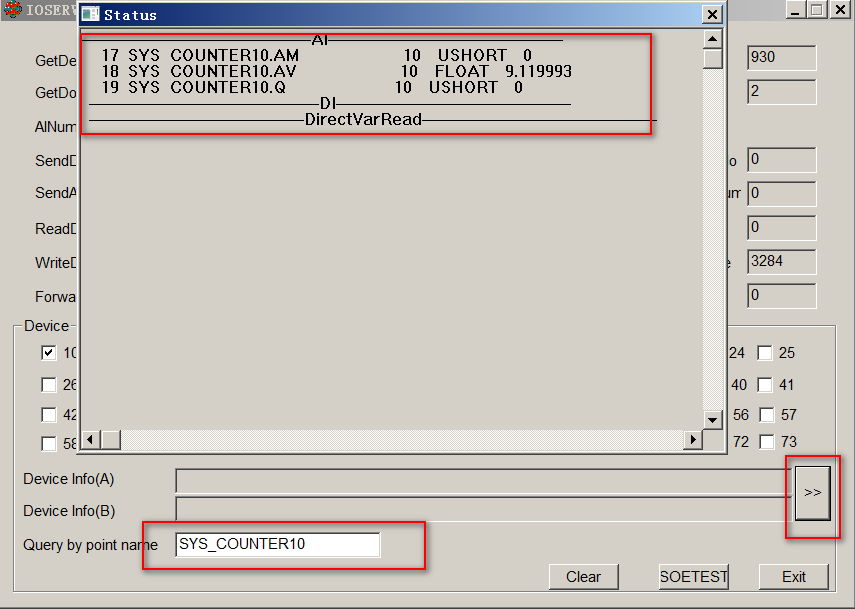
### IOServer工具使用

IOServer工具可以用来查看控制器的状态是否正确，查询控制器中的点值，工具界面如下图。



#### 查看控制器状态

从Device栏可以看出，有10，11号控制器，10号的状态是可选的，就说明10号控制器正确运行，11号的状态是不可选的，就说明11号控制器处在故障状态。



#### 查看点值

在输入框输入想要查询的控制器点名，点击查询按钮，可以查看该点所有控制器点项的值。如果该点值不是期望的的值，那么就需要查看主控中的点项值，如果主控中的点项值是正确的值，那说明IOServer处理的有问题，需要定位IOServer模块。该功能可以定位趋势划线不正确的问题。

## 实时库服务

实时库服务下面主要实时库程序，校时服务程序，诊断服务程序，报警程序等。与实时库程序有关联的模块主要有：报警，诊断，IOServer，历史库，WinRTS，操作员在线，第三方软件，如下图。



### 实时库与操作员在线

操作员在线上面显示的服务器点项值都是从实时库获取的。如

SYS\_NET80A.AV（80号节点128网段的网络负荷）

SYS\_NET80B.AV（81号节点128网段的网络负荷）

SYS\_MEM80.AV（80号节点内存使用率）

SYS\_LOAD80.AV（80号节点CPU负荷）

SYSMCU10A.DV（10号控制器A机运行状态）

SYS10AA.DV（10号控制器A机A网状态）

SYS10AB.DV（10号控制器A机B网状态）

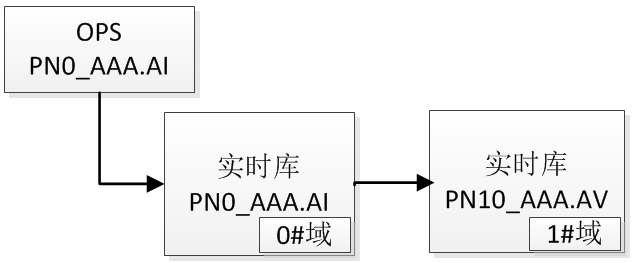
这些点的写值操作也是直接写给实时库。

### 实时库与节点诊断服务

实时库会周期接收节点诊断服务发送的点值信息，主要有控制器的A，B机运行状态点值，控制器的A，B，R网状态点值；控制器CPU负荷，内存使用率点值；MACS节点自身的网络负荷，CPU负荷，内存使用率等。节点诊断服务根据从IOServer接收到的控制器的状态信息以及MACS节点发来的状态，将该信息转化为对应的点值，如SYSMCU10A，SYS10AA，SYS10AB等；节点诊断服务从节点诊断客户端获取MACS节点的相关信息。

### 域间引用

域间引用是指多个域之间可以进行数据通信。实时库支持多域之间的数据通信，如下图，0号域引用1号域的值，0号域实时库读取1号域PN10\_AAA.AV的值，然后将读到的值写到0号域实时库中PN0\_AAA.AI中，OPS显示PN0\_AAA.AI值。



### 实时库与历史库

实时库会周期接收IOServer发来的点值信息，这些点值主要包括模拟量点，开关量点，SOE。开关量和SOE量的变位主动发给历史库，模拟量由历史库主动请求。



历史库收到这些点值主要用来保存历史数据和趋势显示。

### 实时库与WinRTS

实时库与WinRTS只涉及到读写操作。

### 实时库与报警

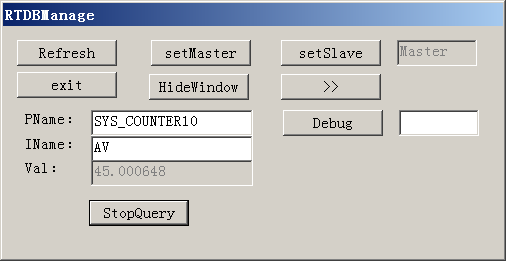
所有的日志，报警都由实时库进行汇总，然后发给报警模块。某模块如果想产生日志或者报警则调用实时库提供的接口SendEventInfo，实时库收到后就会产生一条日志或报警，然后转发给报警模块；实时库中某些周期点值的变化也会产生报警（总控可以配置），如0->1，产生一条报警，1->0产生一条报警恢复，这些报警信息也转发给报警模块。

### 实时库与第三方软件

第三方软件通过实时库提供的接口与MACS系统通信。比如HolliAsComm，通过实时库提供的DBQuery接口可以查询MACS系统的点值；通过实时库提供的DBWrite接口可以写值给MACS系统。

### 实时库工具介绍

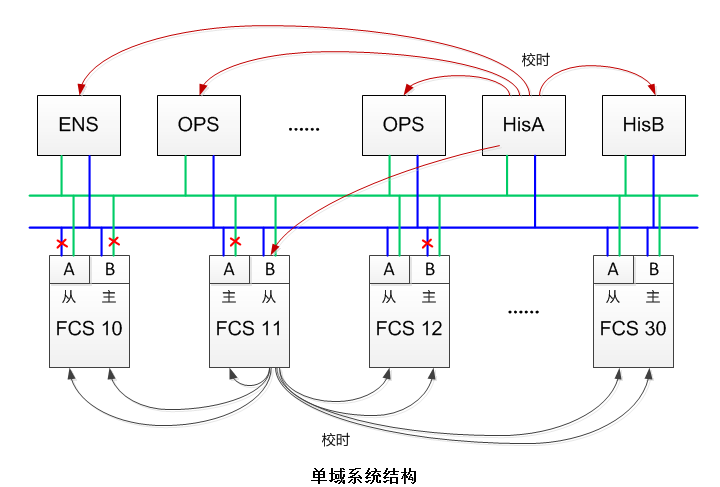
如下图，实时库的主界面可以用来查询点值，切换实时服务的主从状态。



## 校时服务

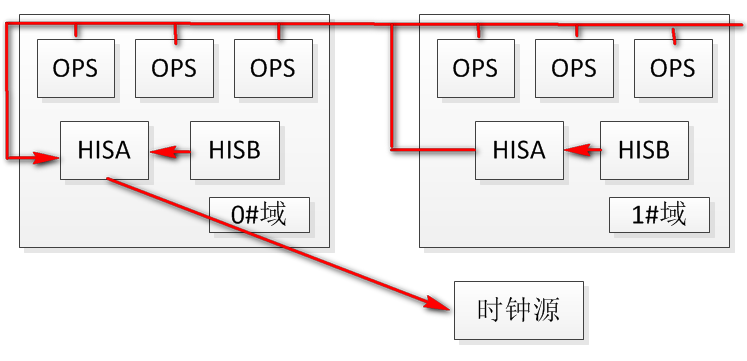
校时服务用来保证MACS节点时间统一，分为单域校时和多域校时。

### 单域校时



单域校时存在于单域工程下，即一个项目下面只存在一个工程。单域校时时，主历史站向时钟源获取时间，从历史站向主历史站获取时间；工程师站，OPS向主历史站获取时间；主历史站向指定的控制器校时，其他控制器向该控制器获取时间。

### 多域校时

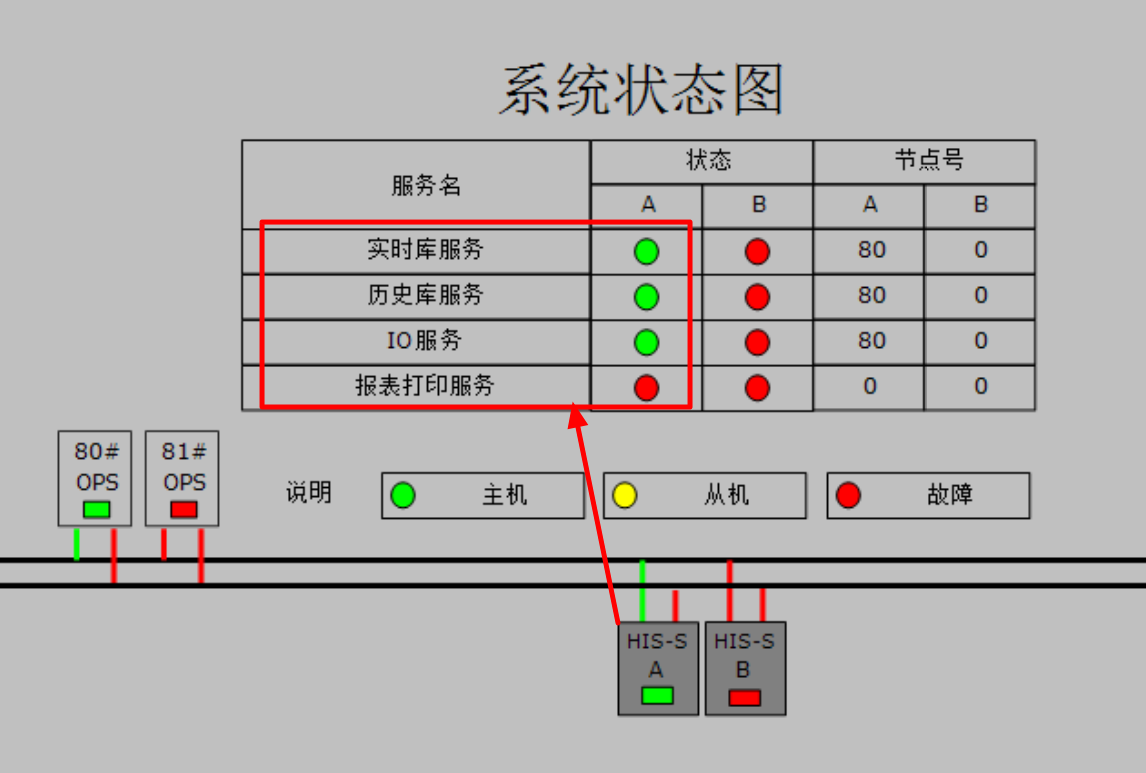


多域校时存在于多域工程下，即一个项目下存在多个工程，每个工程属于一个域。多域校时时，最小号域主历史站向时钟源获取时间，最小号域其他节点包括从历史站，OPS都向主历史站获取时间，它域的主历史站，它域的OPS都向最小号域的主历史站获取时间，它域从历史站向对应的主历史站获取时间。

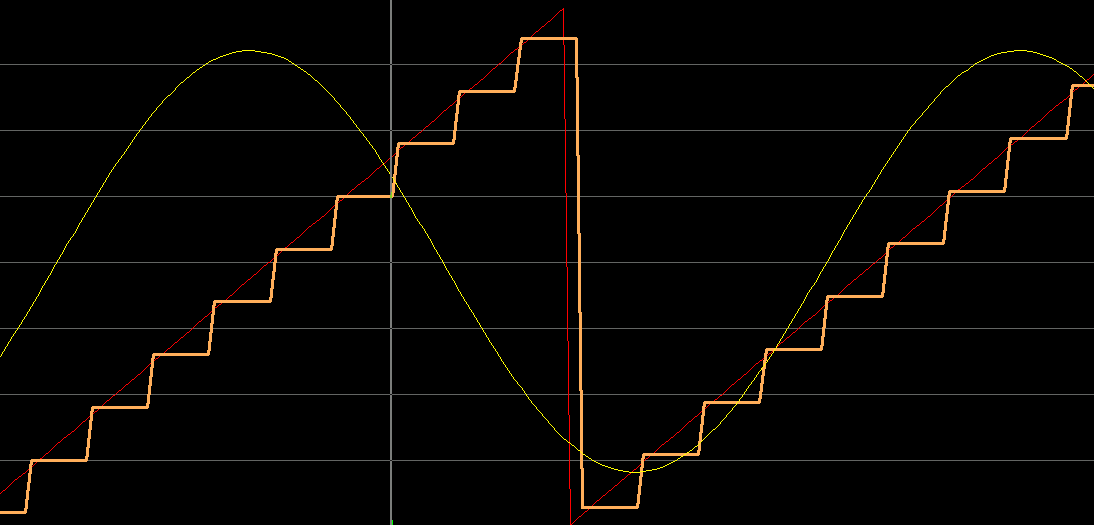
## 历史服务

### 历史站和历史库的区别

一个域一般会运行两台历史站（历史站对应某一台计算机），每台历史站中运行着多个服务，比如实时库服务，历史库服务，报警服务，IO服务等等。如下图所示。图中，A历史站中运行的各个服务的状态都是主(Master)，B历史站此时未开机，处于故障状态。



历史库仅仅是其中一个服务（对应历史站中的某个进程），用来从实时库服务中收集域中的周期数据、开关量变位事件，将这些数据存储到文件，并向操作员以及报表服务（为用户生成报表，报表中的一部分数据来自历史库服务）提供数据查询服务。下图是从操作员界面中看到的趋势信息：

~~~~

因为历史库主要用来向外界提供数据存储和查询服务,本章节主要讲解历史库收集现场数据的过程以及相关的辅助工具。因为历史库针对模拟量和开关量的收集方式不同，所以分别对模拟量和开关量的收集过程以及相关的数据结构进行说明。

### 模拟量

#### 采集周期

历史库对现场模拟量数据的采集过程是周期进行的，但由于以下几个原因，系统可以让用户决定每个点的采集周期:

1.每个点的重要程度并不相同。我们可以将那些重要点的采集周期设定的比较小，将那些次要点的采集周期设定的稍微大一些。

2.每个点的变化程度不同。对变化程度较大、变化速率较快的点，我们应该减小其采集周期，反之增大其采集周期。

3.采集周期对实际产生的文件大小有影响。采集周期越小，单位时间内得到的数据量就越大，文件也越大。若所有点的采集周期都很小，那么占用的磁盘空间就会比较大。

采集周期较大时，得到的实际结果/操作员看到的趋势曲线可能会失真。上图中，两个三角波的数据源是相同的，但一个点(红色曲线)的采集周期是1秒，另一个点(黄色曲线，呈阶梯状)的采集周期是10秒。

每个点的采集周期并不是历史库决定的，而是组态期间由用户设定的(系统提供了默认值)。

#### 文件存储形式

历史库并不使用商用数据库来存储数据，而是采用了私有文件格式。历史库每天生成一个文件夹，用来存放一天内生成的历史数据(如D:\HDBDATAS\20181015文件夹用于存储2018年10月15日生成的历史数据)。

历史库在运行期间，会在内存中保存最近十分钟的数据，并在整十分钟时刻将这些数据保存到文件中。

因为每十分钟产生一个文件，所以历史库每天会生成144个模拟量文件，从ANAHIST0.dat~ANAHIST143.dat。ANAHIST0.dat保存了00:00-00:10期间的数据，ANAHIST.dat保存了23:50-24:00期间的数据。两个文件之间的数据是相互独立的。

dat文件主要包含了两部分内容：

1. 数据区。包含了实际采集到的数据。
2. 对数据区的描述。主要描述了采集的点项名、数据区在文件中的偏移等信息。我们称其索引表。

#### 收集过程

假设历史库一共需要采集四个点。AM1\_1和AM1\_2的采集周期为1秒，AM2\_1和AM2\_2的采集周期为2秒。

因为十分钟包含600秒，所以AM1\_1和AM1\_2在十分钟内需要被采集600次，AM2\_1和AM2\_2在十分钟内需要被采集300次。历史库在开始收集数据时，会分配大小为(2\*600\*6+2\*300\*6)字节的缓冲区来容纳这些数据（历史库使用6个字节来表示一次采集到的数值(4字节，并不一定是浮点类型)和数值的状态(2字节，比如是否可疑)），并在十分钟内将这个缓冲区填满，然后将缓冲区中的数据写入文件。

下图是历史库准备采集数据时数据区的内存分布：



AM1\_1和AM1\_2分别占用了600个位置，AM2\_1和AM2\_2分别占用了300个位置。AM1\_1和AM1\_2这两个点每一秒(0、1、2、3、4…)都会被更新，但AM2\_1和AM2\_2每两秒(0、2、4、6、8…)才会被更新一次。

下图是一秒后(也就是采集完0秒这一时刻后)的内存分布。0秒时，这四个点的索引0处的元素都需要被更新。



1秒时，历史库更新采集周期为1秒的点，但不更新采集周期为2的点：



2秒时，AM2\_1和AM2\_2这两个点进行了第二次更新：



下图是600秒后的内存分布。AM1\_1和AM1\_2这两个点更新了600次。AM2\_1和AM2\_2这两个点更新了300次。整个数据区被填充完毕。

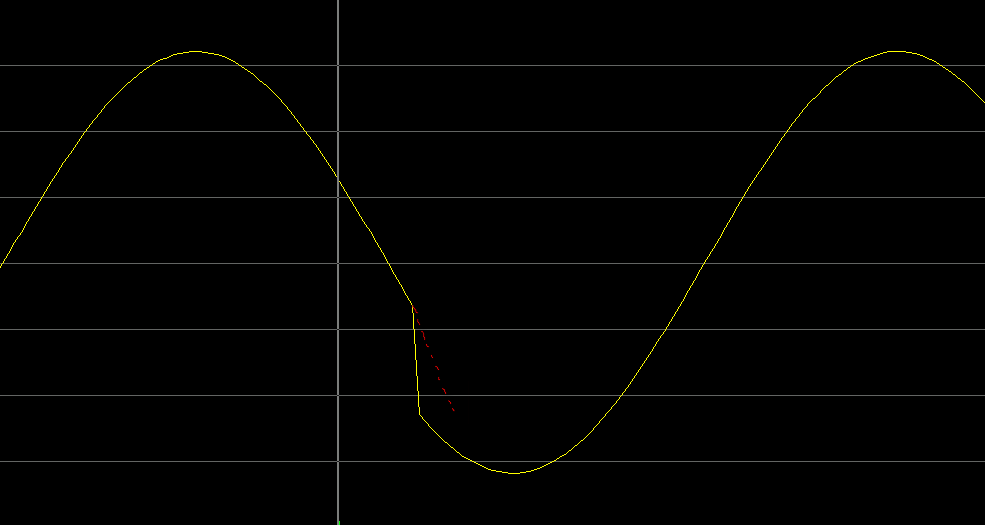


#### 查询

操作员请求查询时向历史库提供待查询的点项名，起始和结束时刻。历史库根据提供的时间定位到某一天中的某个文件，再结合点项信息偏移到文件中的某个位置，读取点值和点值当时的状态。前面提到,历史库会将最近十分钟的数据存储在内存中。所以，如果操作员请求查询的区间在最近十分钟内,历史库就会在内存中检索这部分数据。下文以在文件中检索为例进行说明。

比如：当操作员要查询AM1\_2在00:05~00:10之间的数值时，历史库会根据dat文件中的索引表找到数据区在文件中的偏移，再根据点名顺序(AM1\_2是第二个点)偏移到保存AM1\_2的数据区，然后从600(因为AM1\_2这个点的采集周期是1秒)个数值中获取第6到第11个数值，并返回给操作员。

从存储和查询过程可以看出，历史库存储数据时，并不存储时间戳信息，而是按照约定认为文件中特定偏移处的数据就是点项在某个时间点的值和状态。如果历史站的时间发生变化(比如时间发生了回退)，那么在历史库看来，某个时间段就经历了两次。因为文件只会保留最后一次的数值，之前已经记录的数值就会被覆盖，可能造成的趋势就会是下面这样:



### 开关量

#### 文件存储形式

开关量的文件存储形式与模拟量类似，也包含了索引表和数据区。每天生成的文件名是DIGHIST0.dat~DIGHIST143.dat。但有一点与模拟量不同。对于模拟量，历史库会主动地、周期地从实时库采集所有点的数值，但历史库并不周期地采集开关量的数值。实时库在检测到开关量点的值发生变位(0变成1，或者1变成0)后，向历史库发送一个消息，历史库接到消息后，开始处理，最终将变位信息存储到文件中。当某个十分钟内系统中发生变位的开关量点数越多、每个点变位的次数越多，历史库产生的文件就会越大。

#### 收集过程

为了存储开关量数据，历史库使用了三块内存区：

1. 用来存储实际已经发生的开关量变位信息(结构体)。变位包含的信息有:变位发生的时刻，变位后的值，以及一个指针 (后面会提到)。
2. 一张索引表，用来记录每个开关量点在十分钟内最后一次发生的变位在数据区的索引。
3. 所有点在这十分钟内的初始值。

假设系统中一共有三个点开关量点。分别是DM1、DM2、DM3。

下图是刚开始收集时的内存分布（DM1、DM2、DM3所在的数组，即上文中提到的索引表）。此时，系统中还没有产生任何变位事件。DM1、DM2、DM3都指向空内容。



一段时间后，DM1由0变成1，索引表中的DM1就会指向这个变位。因为DM1这个点当前仅有这一个变位，所以变位结构体中的指针成员指向空：



一段时间后，DM1由1变成0，索引表中的DM1指向第二个变位信息，且第二个变位信息中的指针成员指向前一个(当前十分钟内的第一个)变位信息：



一段时间后，DM2由0变成1，这个变位信息是DM2在数据区中的最后一个变位，因此DM2指向此变位信息。因为DM2当前仅有一个变位，所以变位信息中的指针成员指向空：



一段时间后，DM1由0变成1：



一段时间后，DM2由1变成0：



所以，索引表加数据区可以看作是多个单向链表：点索引表中的每个点相当于每个链表的头。

#### 查询

当操作员请求查询DM2在十分钟内的趋势时，历史库可以通过dat文件中的上述链表信息来实现：

1. 定位到文件中的索引表，得到DM2在十分钟内最后一个变位信息在变位数据区中的索引/偏移。
2. 遍历链表，找出DM2在十分钟内的变位时刻和变位的值。
3. 结合DM2的初始值，将变位信息发送给操作员。

当操作员请求查询DM3在十分钟内的趋势时，历史库通过查询索引表，发现DM3指向空，因此认为在十分钟内没有发生变化。

### 查询优化

#### 模拟量

由上文可知，每个ANAHIST.dat文件存储了十分钟内的模拟量数据。当操作员请求查询十分钟内的趋势时，历史库访问1个文件便可以完成查询。当请求查询一小时内的趋势时，历史库需要访问6个文件才可以完成查询。当操作员请求查询一个月的趋势时，历史库需要访问144\*30=4320个文件才能完成查询。如果要访问完这4320个文件(磁盘的寻道时间和读取时间都会增加很多)，并在每个文件中读取一次数据，那么历史库仅处理一个操作员发来的请求需要花费的时间就无法承受。

另外，操作员虽然可以请求查询一个月的趋势，但趋势画面中最多还是只能显示1800个像素，历史库会从这一个月中仅仅筛选出1800个点值返回给操作员。与其在历史库中对数据进行筛选，为何不直接将数据直接以筛选后的方式进行保存，以便直接为操作员返回筛选后的结果呢？

参考多采集周期的思路，历史库使用了两种均值文件(ANAHIST1\_X.dat、ANAHIST10\_X.dat)。历史库每秒收集一次点值，然后将点值与上次收集到的的值做平均。

每隔一分钟，将运算后的点值写入一分钟均值缓冲区，每隔8小时，将均值缓冲区中的数据写入一分钟均值文件。

每隔十分钟，将运算后的点值写入十分钟均值缓冲区，每隔24小时，将均值缓冲区中的数据写入十分钟均值文件。

当操作员请求查询一个月的趋势时，历史库只需访问30次均值文件即可完成查询。大大减少了访问I/O的次数，进而减少了查询时间。

#### 开关量

模拟量的点值在文件中的位置在绝大多数时候是固定的，因此历史库可以直接在文件的特定位置进行一次I/O便可得到操作员请求的数值。但是开关量与模拟量不同，要查询某个开关量点的变化情况，历史库需要先定位到点的索引处，然后再遍历链表。

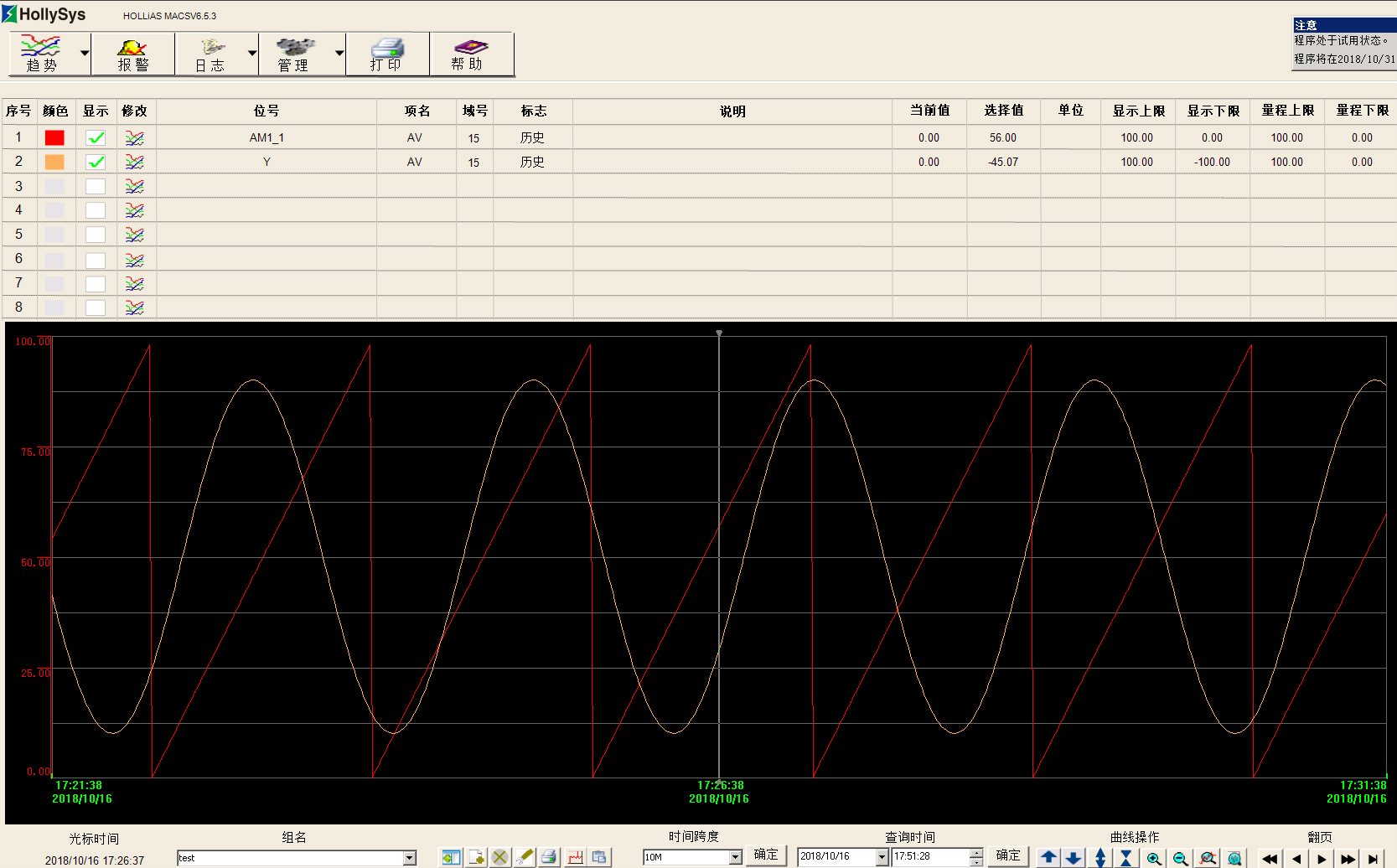
由于现场大多数开关量点的变化程度都很不频繁，因此历史库增加了一个文件(DIGLONG.dat)，用来记录一天内每个点在每个十分钟内发生变位的次数。如果某个十分钟内开关量变位的次数为零，历史库在查询这期间的变位信息时就会直接跳过相应的DIGHIST.dat文件，从而减少了访问I/O的次数，进而减少了查询时间。

## 实用工具

### 离线查询工具

通过离线查询工具，我们可以

1. 打开离线查询工具后，点击…按钮，选择待查询文件夹中的server.ddl文件。
2. 增加趋势组，为趋势组命名。
3. 选择需要查看趋势的点。

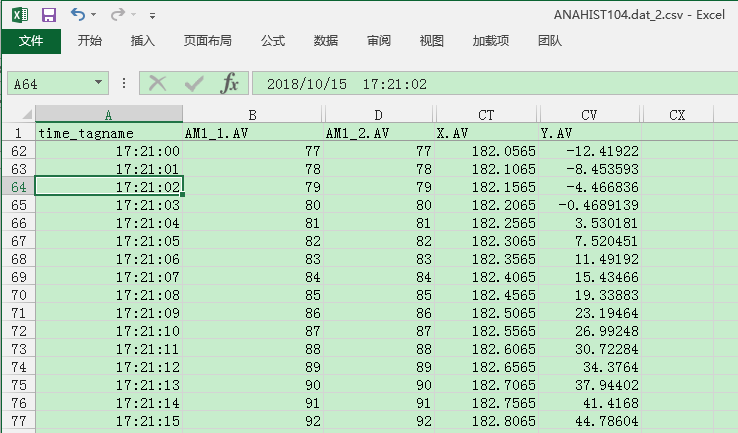


### dat2csv

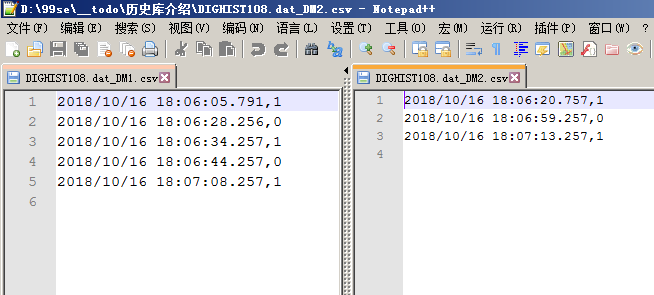
通过操作员在线或者离线查询工具，我们可以通过历史曲线的方式查看历史值。有时为了做数据分析(比如通过excel工具分析历史数据)，我们需要将dat中的数据导出到excel文件中，以便通过其他工具对数据进行深入分析。

dat2csv目前可以解析历史数据文件夹中的模拟量、开关量、均值文件、DIGLONG等文件。

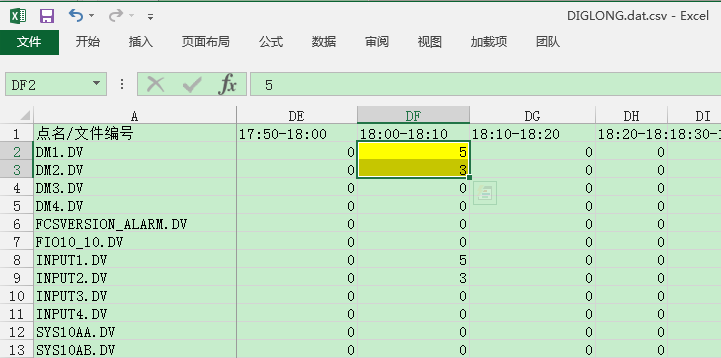
下图是解析得到的模拟量数据：



下图是解析得到的开关量变位信息，可以看出DM1有5次变位，DM2有3次变位。



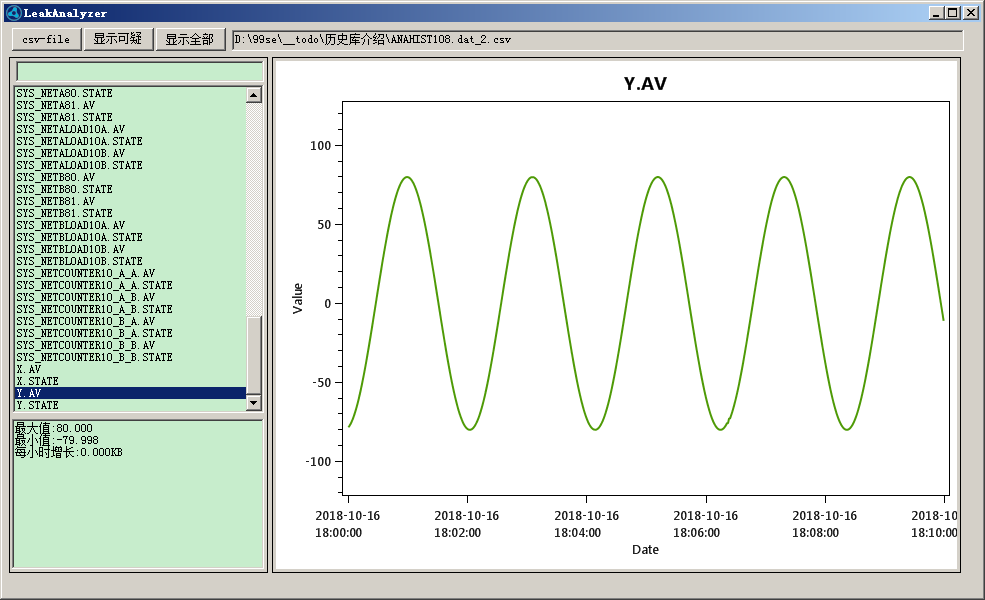
下图是DIGLONG文件中的内容，可以看出DM1在18:00~18:10期间发生了5次变位，在其他时间段内没有发生变位。



### leakanalyzer

在对程序进行性能测试期间，我们通常会借助性能计数器数据对程序的内存和负荷等指标进行观测。通过leakanalyzer工具可以方便地查看性能计数器数据。

因为模拟量数据同性能计数器的格式类似，因为也可以通过leakanalyzer来查看十分钟内模拟量点的趋势。如下图所示：



## 链接

<https://tinyurl.com/how-to-calc-anahist-file-size>

<https://tinyurl.com/dat2csv>

<https://tinyurl.com/leak-analyzer>

<https://tinyurl.com/show-values-using-nodedaemon>