IEC 61131 logic for SCADA Production Optimization

Eric Schwantler, Product Director – ClearSCADA, Control Microsystems Inc

This paper describes the versatility and application of IEC 61131 logic as an embedded function of SCADA host software to optimize Production and Asset Management. Most SCADA have some form of scripting capabilities but this generally requires specialist skills, especially for development, reuse and maintainability. IEC 61131 on the other hand is easy to use and if tightly integrated into SCADA can perform powerful functions such as automatic alarm limit adjustment, plunger lift volume projections, nomination management etc. This paper will discuss some of these applications and demonstrate its configuration with SCADA.

Many of you will know that IEC 61131 is an industry standard programming language designed for Programmable Logic Controllers (PLCs) and RTUs. By adopting this standard, engineers and technicians can work on many vendors equipment with minimal retraining. In this day and age when it is difficult to find engineers for SCADA projects this is of increasing importance. So how is this relevant to SCADA host software? Again you will probably be aware that all SCADA have some form of scripting support for advanced calculations and integration with other SCADA components. The most common language is Visual Basic (VB), but now the engineer must have programming skills and his/her functions better be well documented otherwise the next engineer that needs to work on the project may not understand what the function was supposed to do.

By utilizing the IEC 61131 programming standard inside the SCADA host, it offers several advantages over VB script. The technical skill level is lower and possibly the engineer or technician already has the programming skills since he/she has used this standard on controllers and the debugging environment can display the calculation results in real-time. The IEC 61131 standard comprises five programming languages or style in a single software package. The language options are;

i. Ladder Diagram (LD),
ii. Instruction List (IL),
iii. Function Block (FBD),
iv. Structured Text (ST) and
v. Sequential Function Chart (SFC)

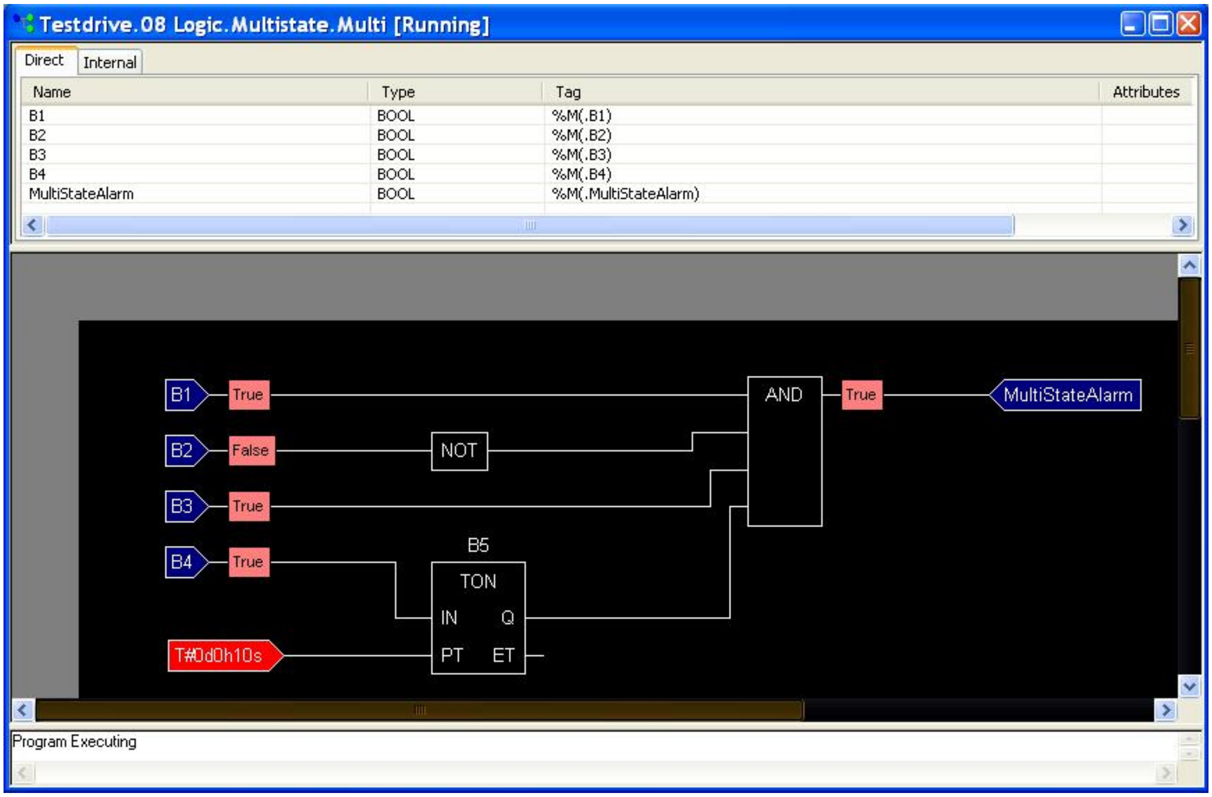
Now you might be thinking, if the SCADA host is configured with logic programs using the IEC 61131 standard does this mean we don't need a PLC or field controller, no, it is not wise to utilize a central control program that is dependent on field communications for control strategy. Consider the example when a high level probe on a tank needs to shutdown the inlet pump. This should always be local control.

So what are some examples where logic programs based on IEC 61131 in the SCADA host make sense? The broad categories that I consider include;

- Advanced or conditional alarming (including prediction)
- KPI calculation for devices or sites
- System wide data aggregation for reporting

In ClearSCADA we have a logic engine with the choice of LD, FBD, ST and SFC objects that the user can configure. All of these programs have full access to the real-time SCADA database, the alarm system, the historian and the configuration database.

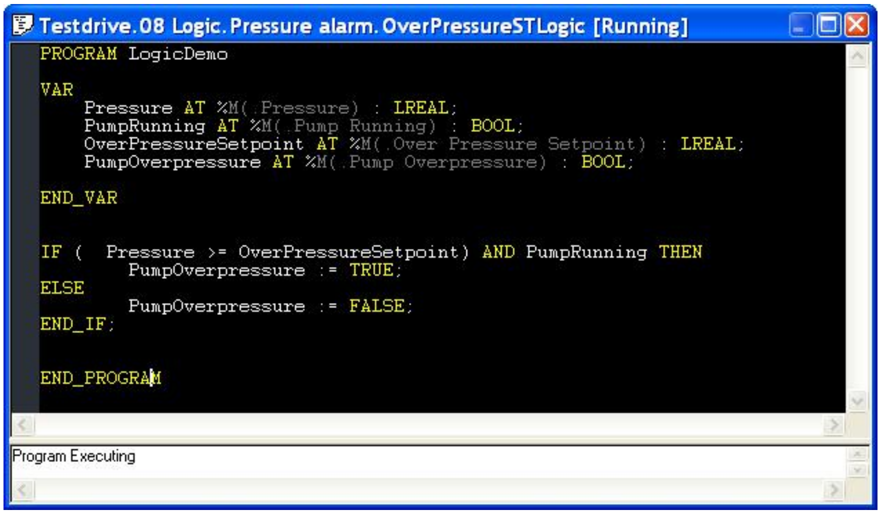
Let us start by describing the Function Block Diagram (FBD) object within ClearSCADA. It has direct and internal variables which are shown in the top section of the diagram below. This is a listing of the tags or points from the SCADA real-time database that are used in the program. You may notice that the tag denotes weather point is input or output and supports relative addressing for program reuse. The centre section has the program blocks with lines drawn to make connections. In addition, the user can show variables in real-time for debugging. The bottom section shows program status or errors if any occur. This is simple example of conditional alarming.



Name	Type	Tag	Attributes
B1	BOOL	%M(.B1)	
B2	BOOL	%M(.B2)	
B3	BOOL	%M(.B3)	
B4	BOOL	%M(.B4)	
MultiStateAlarm	BOOL	%M(.MultiStateAlarm)	

Program Executing

The Structured Text (ST) program type is also popular and better suited for more complex programs especially if involving program loops. The below screen shot is another example of conditional alarming, this time monitoring the line pressure but only generate alarm if above setup and pump is running.



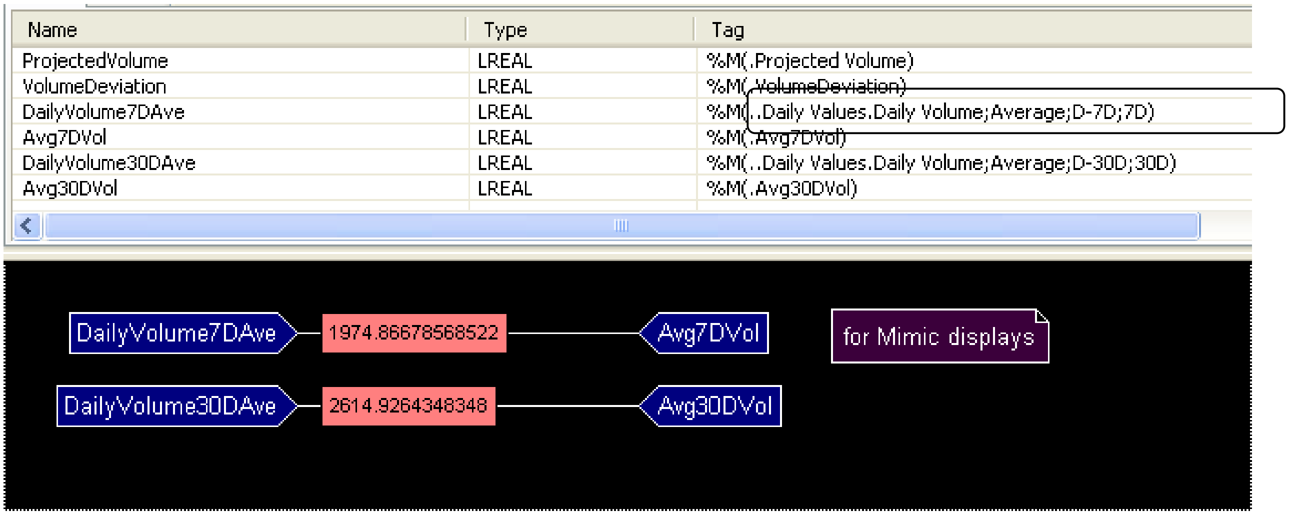
Testdrive.08 Logic.Pressure alarm.OverPressureSTLogic [Running]

```
PROGRAM LogicDemo

VAR
    Pressure AT %M(.Pressure) : LREAL;
    PumpRunning AT %M(.Pump Running) : BOOL;
    OverPressureSetpoint AT %M(.Over Pressure Setpoint) : LREAL;
    PumpOverpressure AT %M(.Pump Overpressure) : BOOL;

END_VAR


IF (  Pressure >= OverPressureSetpoint) AND PumpRunning THEN
        PumpOverpressure := TRUE;
ELSE
        PumpOverpressure := FALSE;
END_IF;


END_PROGRAM
```

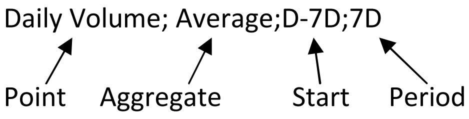
Program Executing

If you look closely at the variable declarations you will notice they are prefixed by ".", this refers to the current level within the ClearSCADA object hierarchy (database) so if this program was defined as part of a device template or copied it will not need to be edited for each device instance. This provides standardization, rapid deployment and simple maintenance resulting in considerable $$ savings for deploying a SCADA project.

Let's now move on to some more practical examples as related to Oil & Gas industry. We all know that gas well production deteriorates over time so by reviewing the past average volume we can set realistic production target and monitor deviation for alarming. In addition we can automatically set the alarm limits so that if volume decreases outside of normal operation, the operators can be alerted. The first thing we need to calculate Is the past average volume, this is very easy in ClearSCADA as the historian can return data aggregate value for any archived point. In this case we will extract the 7 day and 30 day average and put result into an internal point for further calculations.



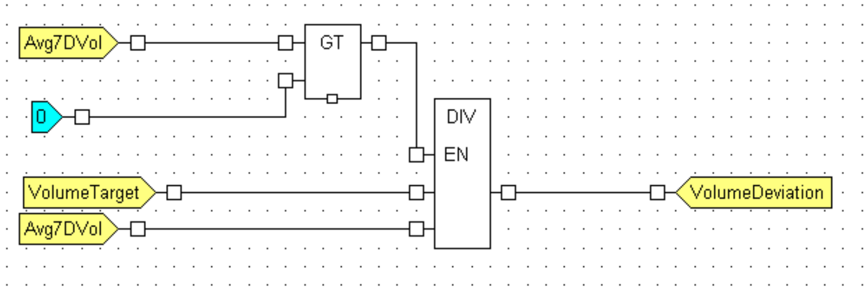
Name	Type	Tag
ProjectedVolume	LREAL	%M(.Projected Volume)
VolumeDeviation	LREAL	%M(.VolumeDeviation)
DailyVolume7DAve	LREAL	%M(..Daily Values.Daily Volume;Average;D-7D;7D)
Avg7DVol	LREAL	%M(.Avg7DVol)
DailyVolume30DAve	LREAL	%M(..Daily Values.Daily Volume;Average;D-30D;30D)
Avg30DVol	LREAL	%M(.Avg30DVol)



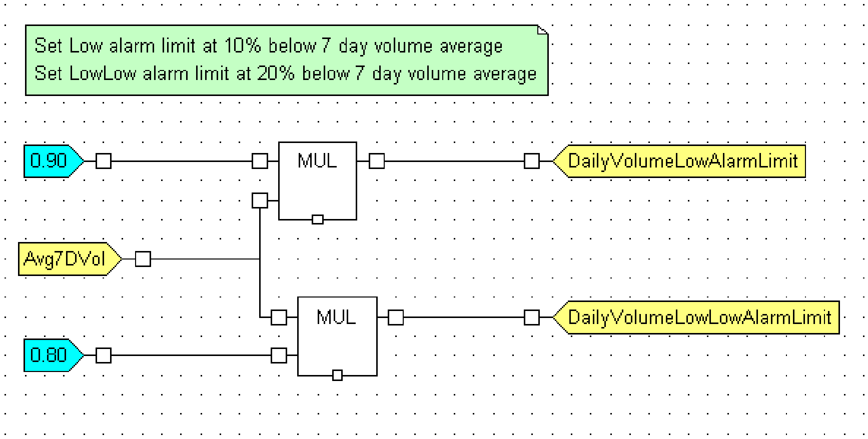
Daily Volume; Average;D-7D;7D

Point Aggregate Start Period

This FBD code segment calculates the volume deviation as a factor from the 7 day average volume vs the daily target volume. The logic blocks available are as defined by the IEC 61131 standard



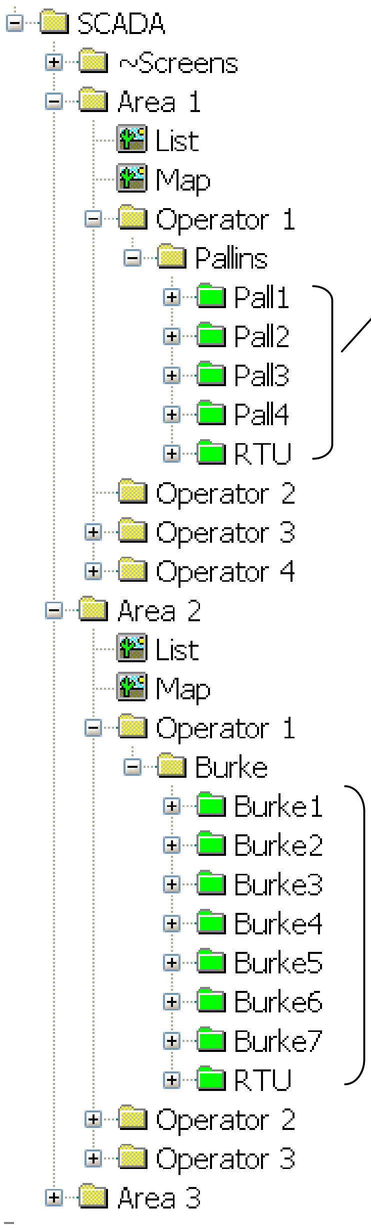
This FBD code segment calculates the alarm set-point limits for the Daily Volume tag. All points and configuration fields in ClearSCADA are exposed as OPC properties.


Set Low alarm limit at 10% below 7 day volume average
Set LowLow alarm limit at 20% below 7 day volume average

These programs would be included in the well template and reference the local tags and history for the well. ClearSCADA is a true object oriented SCADA host so can encapsulate everything associated to a device or site in a template. When a new instance of a well is deployed all of the graphics, points, alarm settings, history configuration, security, reporting and logic is created and this can be done very quickly and without any programming or scripts to auto-generate the configuration.

This is an example of a well template ("Flowrun") with folders to organize the points and configuration. You will notice an assortment of different icons which represent ClearSCADA objects. In the folder group 'Projected Values' you will notice the point for 7 and 30 day average volume, target volume and deviation and the FBD logic programs that do the work.

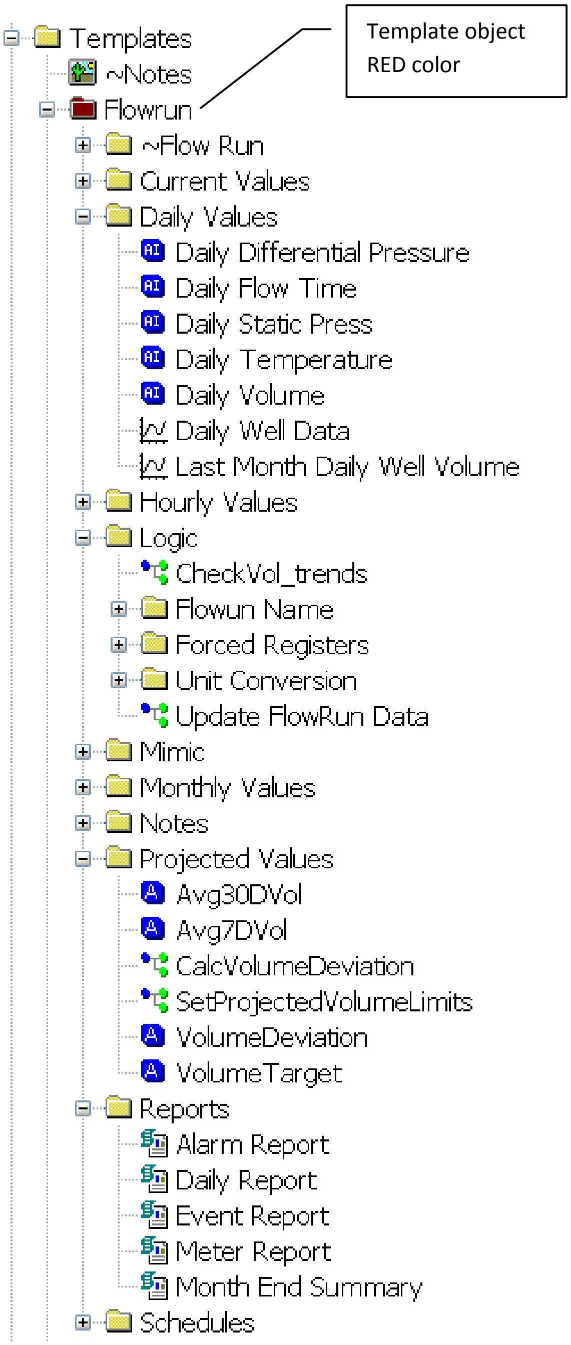
To use the template is simple matter to create an instance and place that instance or well in the system hierarchy of the SCADA database. (See below)



```
⊟ SCADA
    ⊞ ~Screens
    ⊟ Area 1
        List
        Map
        ⊟ Operator 1
            ⊟ Pallins
                ⊞ Pall1
                ⊞ Pall2
                ⊞ Pall3
                ⊞ Pall4
                ⊞ RTU
        Operator 2
        ⊞ Operator 3
        ⊞ Operator 4
    ⊟ Area 2
        List
        Map
        ⊟ Operator 1
            ⊟ Burke
                ⊞ Burke1
                ⊞ Burke2
                ⊞ Burke3
                ⊞ Burke4
                ⊞ Burke5
                ⊞ Burke6
                ⊞ Burke7
                ⊞ RTU
        ⊞ Operator 2
        ⊞ Operator 3
    ⊞ Area 3
```

Instances of template(s) GREEN color

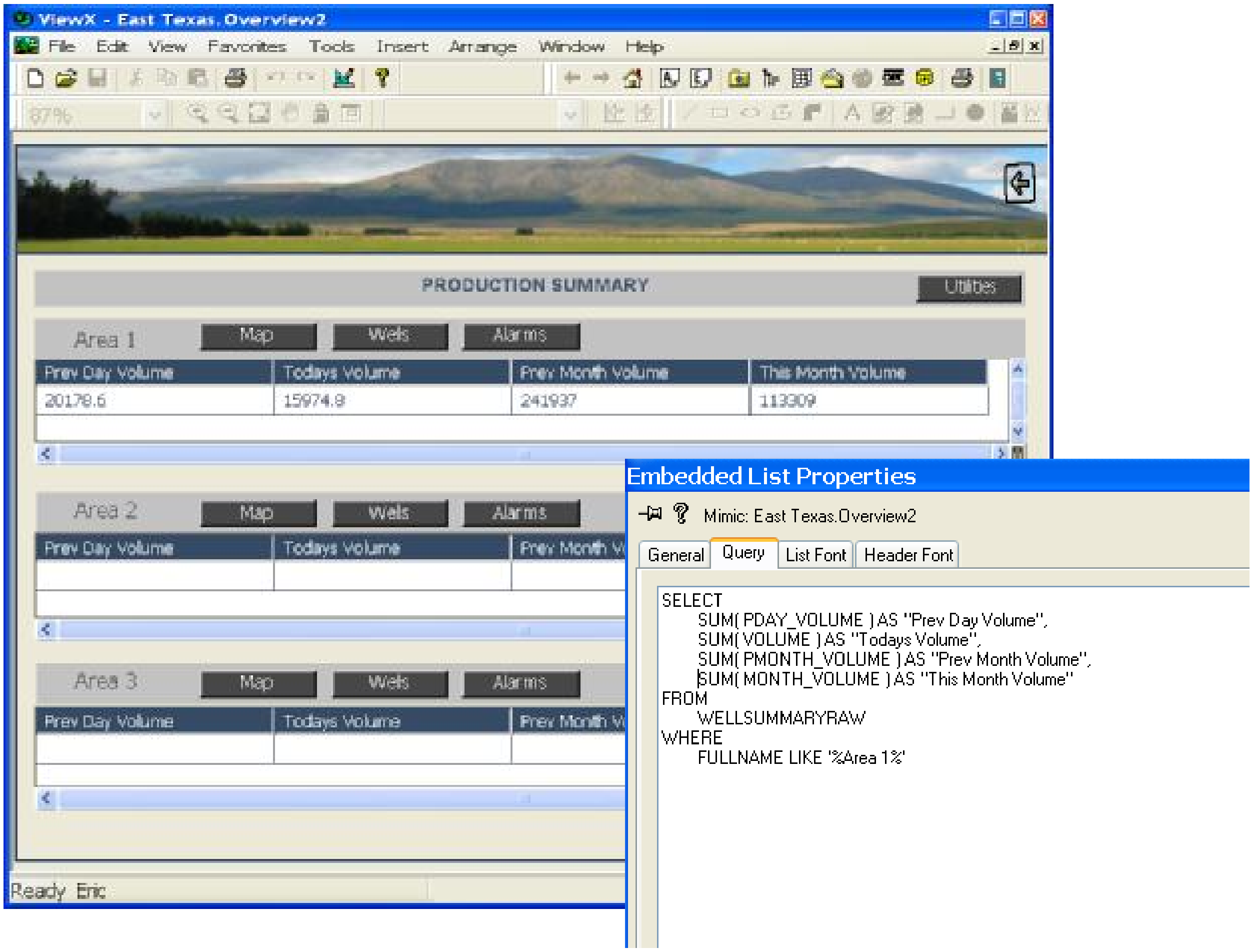
Instances of template(s)



```
⊟ Templates
    ~Notes
    ⊟ Flowrun
        ⊞ ~Flow Run
        ⊞ Current Values
        ⊟ Daily Values
            Daily Differential Pressure
            Daily Flow Time
            Daily Static Press
            Daily Temperature
            Daily Volume
            Daily Well Data
            Last Month Daily Well Volume
        ⊞ Hourly Values
        ⊟ Logic
            CheckVol_trends
            ⊞ Flowun Name
            ⊞ Forced Registers
            ⊞ Unit Conversion
            Update FlowRun Data
        ⊞ Mimic
        ⊞ Monthly Values
        ⊞ Notes
        ⊟ Projected Values
            Avg30DVol
            Avg7DVol
            CalcVolumeDeviation
            SetProjectedVolumeLimits
            VolumeDeviation
            VolumeTarget
        ⊟ Reports
            Alarm Report
            Daily Report
            Event Report
            Meter Report
            Month End Summary
        ⊞ Schedules
```

Template object RED color

Up until now we have been describing the value and ease of configuration to use logic programs at the device or site level, what if we need to perform system wide logic as would be required for nomination management or plunger lift optimization to limit line pressure for compressor capacity. Now the logic program can have any number of input points and these can be hard wired from the well instances but a much smarter way would be to execute an SQL query on the database to sum the production totals from all wells in a field or region. This way if a well site is added or removed, the query does not need to be edited. The logic program can call the SQL query and results displayed in a table which in ClearSCADA is called an embedded list. The screen below shows the production summary for 3 area or regions. The totals in the list are result of an SQL query on the historical database using daily and monthly EFM data.



Prev Day Volume	Todays Volume	Prev Month Volume	This Month Volume
20178.6	15974.9	241937	113309

```
SELECT
      SUM( PDAY_VOLUME ) AS "Prev Day Volume",
      SUM( VOLUME ) AS "Todays Volume",
      SUM( PMONTH_VOLUME ) AS "Prev Month Volume",
      SUM( MONTH_VOLUME ) AS "This Month Volume"
FROM
      WELLSUMMARYRAW
WHERE
      FULLNAME LIKE '%Area 1%'
```

Additional logic program can use the current production totals and calculate nominations for the month. If controlling plunger lift to ensure not exceeding line/compressor capacity, a logic program can be used to queue and enable plunger cycle. This now borders on centralized control even though not safety related so consideration should be given to redundant communications networks and servers.

In conclusion, IEC 61131 can be a valuable tool to enrich the raw data collected by the SCADA host in an easy development environment that is both quicker to deploy and maintainable by lower skill set over traditional SCADA using VB code.