Global Flow Measurement Workshop
22nd – 24th October 2024

Technical Paper

Streamlining Daily Production Reporting with IIoT: Case Study of 3R Petroleum Brazil

Anwar Sutan, i-Vigilant Technologies

Danilo Diógenes, 3R Petroleum

Anderson Magalhães, 3R Petroleum

1 INTRODUCTION

In early 2023, 3R Petroleum and i-Vigilant collaborated to develop a solution to meet the regulatory requirements set by the Brazilian government through the Agência Nacional do Petróleo (ANP). Brazilian oil and gas operators are mandated to submit daily production information in XML format via a government portal. However, the manual transfer of data from flow computers to the government website poses significant challenges due to the lack of automation and the potential for human errors, especially for operators managing multiple fields with numerous streams.

To address this issue, an automated solution was developed. Data from flow computers, typically in formats of Modbus [1] or OPC [2], are collected and processed before being transferred to a database in the required format structure. Subsequently, another software generates the XML file, which undergoes automatic validation prior to submission to the government.

The implementation of automated data transfer processes also offers numerous benefits to operators beyond regulatory compliance. These include real-time flow data, alarm and event reporting, master configuration checks for flow computers, calculation verification, and live uncertainty calculations. Moreover, access to critical data from devices such as gas chromatograph (GC), ultrasonic meters (USM), and differential pressure (DP) meter readings is facilitated, improving operational efficiency and decision-making processes. This is further enhanced by using condition based monitoring software which can be used to detect early onset of measurement issues as well as extending calibration frequencies.

Importantly, the adoption of data transfer mechanisms extends beyond offshore operations, as data from the process control network (OT) can be seamlessly integrated into cloud databases (IT), enabling universal access by stakeholders. This centralized data repository allows for tailored processing and structured storage based on stakeholder requirements. This single source of truth can be used by multiple applications, for example data necessary for GC or USM CBM can be consumed by dedicated applications, facilitating collaborative efforts and informed decision-making.

Through a detailed use case presentation, this paper illustrates the implementation process and highlights the tangible benefits that operators can derive from leveraging available technologies. By embracing automation and cloud-based solutions, operators can streamline production reporting processes, enhance data accessibility, and optimize operational efficiency in the dynamic landscape of the energy industry.

2 SYSTEM ARCHITECTURE AND PROJECT AIM

To provide context for the project's aim, the following is a simplified system architecture overview.

Figure 1 illustrates the system architecture for a setup utilizing the S600 flow computer. On these assets, direct data transfer to the cloud is restricted due to firewall limitations. To overcome this, a database synchronization tool has been developed to act as a bridge, facilitating the transfer of data from the site to the cloud database.

Figure 2 depicts the system architecture for a setup using the SMAR HFC302 flow computer. In this configuration, the firewalls are configured to permit automatic data transfer to the cloud database, enabling seamless integration.

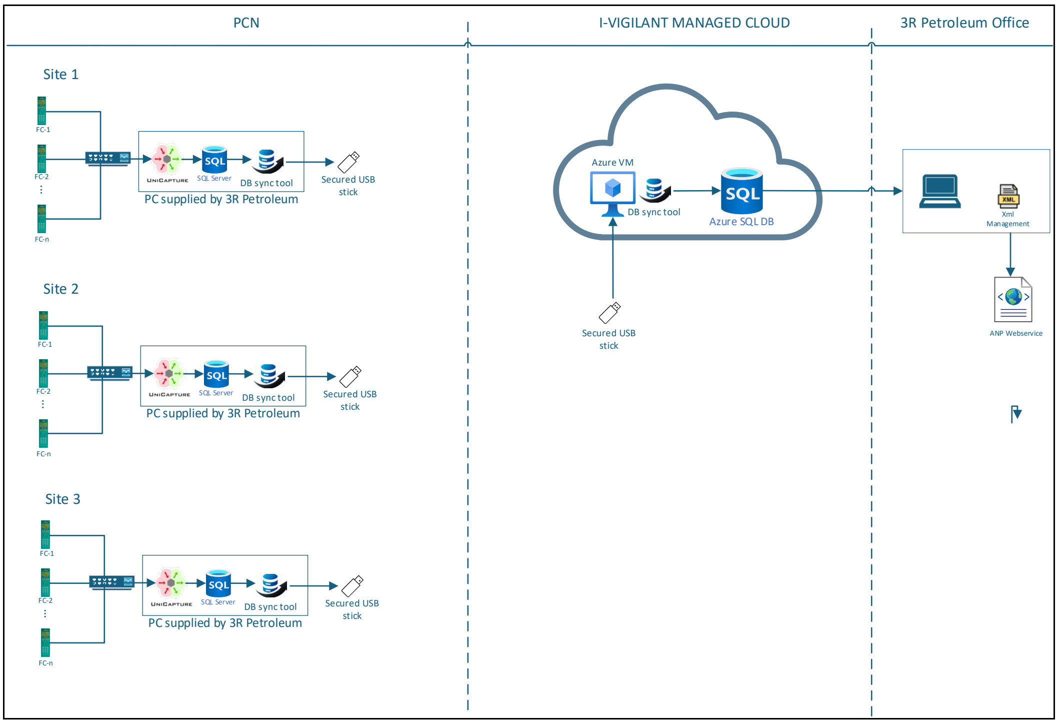


Fig 1. S600 flow computer system architecture

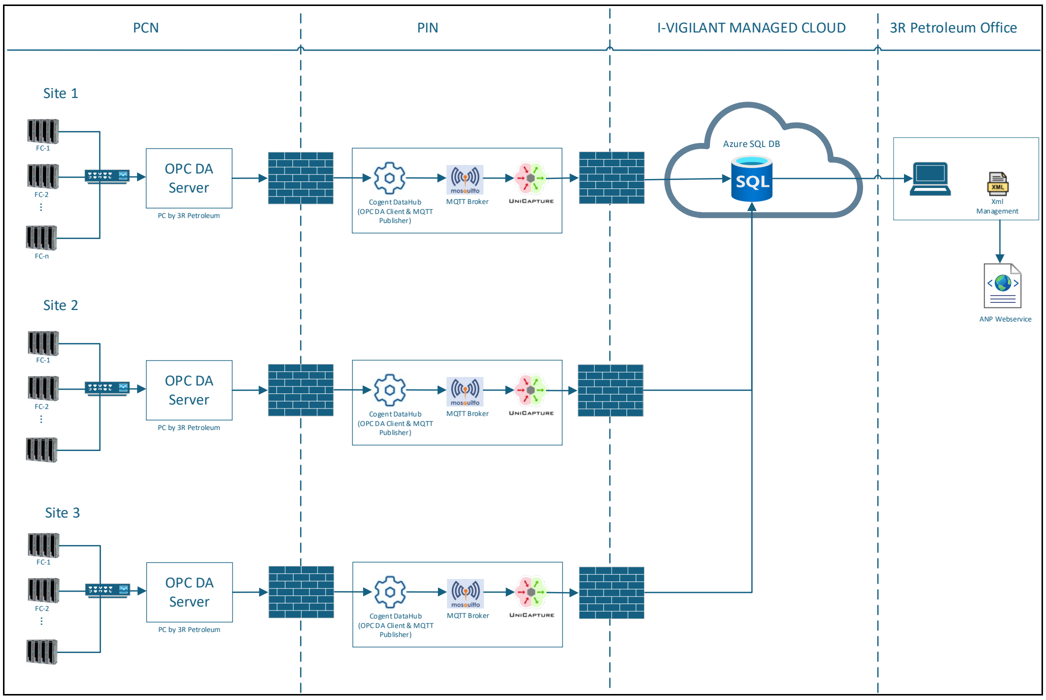


Fig 2. SMAR flow computer system architecture

The main aim of the project is to enable automatic data transfer from offshore flow computers to the cloud. This automation ensures that the generation of XML files, as required by the (ANP), can be accomplished with minimal manual intervention.

3 WORK PERFORMED

3.1 Database design

One of the first tasks undertaken in this project was the design of the database system. This was essential to ensure a unified database that could be used by all flow computers in the system while still distinguishing data from each individual flow computer. Efficient database table design simplifies the data transfer requirements for all flow computers.

Separate database tables were created for each XML requirement, along with a specific table for the flow computer devices to differentiate the source of the data. Although the detailed database design is proprietary and not disclosed in this paper, it is emphasized that a well-structured database design is crucial for efficient data transfer and organization. It also ensures the scalability of the system to accommodate an increasing number of flow computers.

The project is divided into two distinct parts, each focusing on a different type of flow computer with completely different architectures.

1. S600 flow computer (based on system architecture described in Figure 1)
2. SMAR flow computer (based on system architecture described in Figure 2)

There are significant differences between these two flow computers. The primary distinction is that the S600 provides its data via Modbus, while the SMAR flow computer provides its data using OPC DA. Due to these differences, the scope of work for each flow computer type is completely separate.

3.2 Architecture 1 Using S600 Flow Computers

For the system architecture using S600, the work is split into multiple phases as follows:

1. Flow computer update
2. UniCapture configuration
3. DB Sync tool design

The following is a description of work performed on each phase:

3.2.1 Flow computer update

Data gathering from the S600 flow computer is performed using Modbus. To ensure correct data transfer, the following steps were taken:

1. Flow computer communication port is added
   - S600 has limitation of one TCP port to be accessed by maximum 2 masters
   - Two TCP ports are added. One for the data access which consist of hourly and daily data, the other for alarm and events
2. S600 modbus registers are updated to ensure that all required data is available in the S600 for UniCapture to collect.
   - Required data is based on xml requirement
   - Alarm values are based on all available active alarms in S600
   - Events are from constant logs

3.2.2 UniCapture Configuration

UniCapture is a versatile software product designed to acquire data from multiple capture sources and push it to various target storage. On the acquisition side, it supports Modbus (serial and Modbus/TCP), OPC UA, MQTT, and Event Hub. For target storage, it supports common databases such as MySQL, Microsoft SQL Server, Microsoft Azure SQL, SQL Server Compact, and Microsoft Access. Additionally, it can push data to MQTT, IoT Hub, and OPC UA, acting as an OPC Client.

UniCapture not only captures and stores data but also performs calculations. This dual functionality allows it to serve as both an ‘Edge Data Collector’ and an ‘Edge Analytics’ tool, providing a simple yet powerful solution that is easy to configure.

Running as a service, UniCapture can be installed on any Windows-based PC, server, or VM slice, making it easily deployable without the need for additional dedicated hardware such as a PLC, PAC, Industrial Controller, or other Edge devices.

UniCapture is configured to capture all necessary data from the flow computer and push it to the database. Each configuration file is tailored to collect data from a specific flow computer. Below are the elements of UniCapture configuration on each configuration file.

1. Capture sources

   Two capture sources have been configured. One captures data every hour for hourly data collection, while the other monitors data every 15 seconds to track alarms and events occurring in the flow computer.

   These two capture sources are enabled by UniCapture communicating with different TCP ports on the same flow computer.

   Aliases for Modbus registers have been created in UniCapture. Instead of using addresses when pushing data to the database, UniCapture uses these Modbus aliases to simplify configuration for the user. Below is an example of an address alias configuration:

Address	Alias name
4:03153	STR01_BASE_TEMP(Tb)-VALUE
4:04107	Atm_Press (bara/psia)-VALUE
4:03155	STR01_BASE_PRESS(Pb)-VALUE
4:04109	STR01_CHR_SLCT_MOLES-NITROGEN
4:04111	STR01_CHR_SLCT_MOLES-CO2

Fig 3. UniCapture Modbus alias

2. Target storages

   Due to firewall limitations, this system cannot transfer data directly to the cloud. To address this, SQL Server Express has been installed to temporarily store all the required data. In UniCapture, the target storage is defined as the local SQL Server Express database.

3. Capture groups and assignments

   There are multiple types of xml required for different types of streams as follows:

   - DP (for dp meters)
   - Linear gas (for gas ultrasonic meters)
   - Oil (for oil ultrasonic meters)
   - Alarm and events

In the database, each xml type has its own individual tables. To ensure UniCapture pushes the correct data to each table, capture groups are defined. For example, if a meter stream is of a DP type, capture groups are set up to collect data from Modbus and push it to the appropriate database table.

Not all data comes from Modbus; there are various methods to provide data to the database. For each XML table, the following data sources were utilized:

- Constant value → Provides a fixed value to the database. For example, values such as FC_ID use constant values.
- Math operation (multiplication) → Used for example to convert pressure units. For example, if the flow computer uses "bar" as the pressure unit but the XML requires "kPa," a multiplication operation is necessary. The Modbus value needs to be multiplied by 100 before being saved into the database.
- Math operation (addition) → Used for example to Convert from gauge pressure to absolute pressure. This requires the gauge pressure to be added with constant of 101.325.
- DLL Function for More Complex Operations: For example, the XML requires an initial period to be one day before the captured Modbus timestamp, with the time set to 00:00:00. A DLL function was created that is called by UniCapture where UniCapture collects the current date and time, sets the time to 00:00:00, and adjusts the date to one day prior. UniCapture then push The result of this operation into the initial period column in database.

The following is some examples of the above logic:

Table 1. Examples of UniCapture built-in logic

Source	Value written to the database
Constant value: 16	16
Math operation multiply (axbxcx…): a: Atm_Press (bara) = 1.01325 b: Constant = 100	101.325
Math operation addition (a+b+c…): a: STR01_BASE_PRESS(Pb) = 0 b: Constant = 101.325	101.325
Dll function for date time operation ReduceToDayMinus1 2024-09-16 00:01:35	2024-09-15 00:00:00

Table 1 provides examples of data sources, the processes performed on the data, and the resulting data written to the database. Each individual capture group presents its own challenges. In practice, many more functions are available, and much more work is done in the project to ensure all required data is accurately captured in the database. Below are the types of capture groups included on this architecture.

a. Stream hourly and daily

Daily reports are required for XML submissions to the ANP, while hourly reports provide valuable trends. These capture groups are polled every hour to gather hourly information. Since the daily information only changes at midnight, the database is updated once every 24 hours, even though the data is polled hourly.

Data is polled from the flow computer using Modbus. These data are combined with constants and mathematical operations to form data sets suitable for the XML report, which are then transferred to the database.

b. Alarm

The alarm capture group presents its own challenges. For alarms, it is necessary not only to record the time when the alarm becomes active or is cleared, but also to describe whether the alarm is clearing or becoming active, and at the same time to capture the value of the alarm limit. The main issue is that UniCapture needs to compare the current data against the previous data, which is not possible with simple Modbus communication. Below is an example of an alarm recorded in the database:

Table 2. Example of alarm recorded in database

Device_ID	Alarm Date	Description	Value
1	2023-05-1 12:09:49	STR01_CVOL_FR-L - Active	8.33
1	2023-05-1 12:15:13	STR01_CVOL_FR-L - Clear	8.33

To enable UniCapture to push such data, a comprehensive list of all alarms from the flow computer, along with their associated Modbus addresses, has been made available in a CSV file. Below is an example of the CSV list:

Table 3. Example of csv list for alarm

FCID	Alarm address	Value address	Description	Value
1	0:01925	0:01925	STEAM_TASK-MONITOR	0
1	0:01926	0:01926	STEAM_TASK-TSK_ERR	0
1	0:01927	4:03963	45-FT-5400A-1-LL	0
1	0:01928	4:03961	45-FT-5400A-1-L	0

A DLL program is then created to handle the process until the data is pushed to the database. The steps of the DLL program are as follows:

- Polling registers

  UniCapture polls a list of registers as defined in the CSV file. It polls a range of coils with function code 0 based on the alarm address and a range of addresses with function code 4 for the values.

  Note: Modbus function code 0 is for reading coils, and function code 4 is for reading holding registers.

- Comparing values

  It compares the polled address value against the “Value” column in CSV file. If the polled value matches the “Value” column, then no action is taken. If the polled value differs from the “Value” column in CSV file, then the following information is pushed to the database:

  - Device ID = FCID from CSV file
  - Alarm date = current time
  - Description = Description from the CSV file, with the suffix “Active” if the alarm value is 1, and “Clear” if the alarm value is 0
  - Value = the value captured using Modbus as per the “Value address”

- Updating CSV File

  Once the above process is complete, the value in the CSV file is updated with the value captured using Modbus

- Repeating the Procedure

  The above procedure is performed every 15 seconds for alarms listed in the CSV file

c. Event

The event capture group presents another challenge. The goal is to record all events performed on the flow computer, including changes to constants and status updates. This requires capturing and pushing the following information from each flow computer to the database:

- Time of the event
- What was changed
- Original value
- As left value

The following is an example of an event recorded in the database:

Table 4. Example of events recorded in database

Device_ID	Event Date	Description	Original	As left
1	2024-09-09 09:10:25	45-PT-5400B-KEYPAD	668	5.876
1	2024-09-11 20:56:46	20-WT-2320-1-MODE	4	0

Similar to the alarms, a comprehensive list of all events from the flow computer, along with their associated Modbus addresses, has been made available in a CSV file. Below is an example of the CSV list:

Table 5. Example of csv list for events

FCID	Modbus address	Description	Value	Master
1	4:04017	45-PT-5400B-MODE	0	0
1	4:04019	45-PT-5400B-KEYPAD	668	0
1	4:04021	45-PT-5400B-HIHILIM	1000	1000
1	4:04023	45-PT-5400B-HILIM	900	900
1	4:04025	45-PT-5400B-LOLIM	100	100
1	4:04027	45-PT-5400B-LOLOLIM	0	0
1	4:04029	45-PT-5400B-HISCALE	1000	1000
1	4:04031	45-PT-5400B-LOSCALE	0	0

A DLL program is created to handle the process until the data is pushed to the database. The steps of the DLL program are as follows:

- Polling Registers
- UniCapture polls a list of registers as defined in the CSV file.
- Comparing Values
- It compares the polled value against the “Value” column in CSV file. If the polled value matches the “Value” column, then no action is taken. If the polled value differs from the “Value” column, the following information is pushed to the database:
  - Device ID = FCID from csv file
  - Event date = current time
  - Description = description from CSV file
  - Original = the value read from CSV file
  - As left = the value captured using Modbus
- Updating CSV File
- Once the above process is complete, the value in the CSV file is updated with the value captured using Modbus.

- Repeating the Procedure
- The above procedure is performed every 15 seconds for every event listed in the CSV file.

4. Save the Configuration

   Once all configurations are completed, each configuration is saved. One configuration file is created for each flow computer, which can be up to four streams. With 19 different flow computers being monitored, this means 19 concurrent configuration files are running simultaneously, monitoring these flow computers and pushing data to the database accordingly.

3.2.3 DB Sync Tool

Sometimes ideal conditions are not available, necessitating creative programming solutions. For sites using the S600 flow computer, firewalls prevent direct connection to the cloud database, meaning data can only be stored locally. However, for XML generation to work, the data needs to be available in the cloud database.

To address this, an export/import tool was created. This tool allows data for a specific period to be exported to an SQL script, which can then be run using the same tool to import the data into the cloud database. The export tool user interface is as follows:

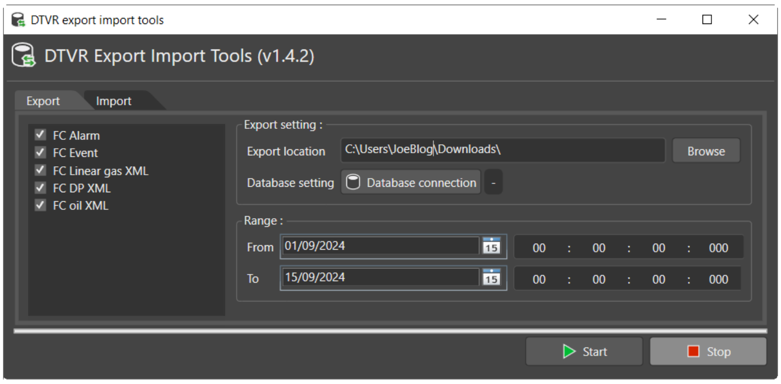


Fig 4. Database export/import tool – export user interface

The export tool exports data from selected tables or all tables into an SQL script. Once the SQL script is generated, the import tool can be used to upload it to the cloud database. The import tool user interface is as follows:

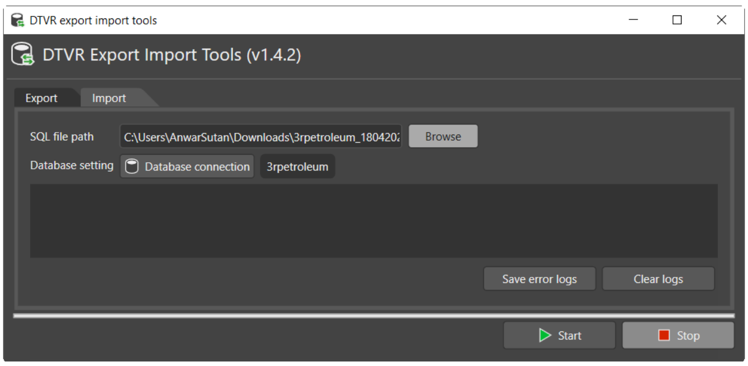


Fig 5. Database export/import tool – import user interface

Although this is not a standard solution and was created specifically for this project, it shows that some creative programming solution can be a viable option for periodically synchronizing offline data.

3.3 Architecture 2 Using SMAR Flow Computers

The work for the system architecture utilizing SMAR flow computers is divided into several phases, as outlined below:

1. Convert OPC DA data to MQTT
2. UniCapture configuration
   - Retrieve data from MQTT Broker
   - Push data to cloud database

The following is a description of the work performed in each phase:

3.3.1 Convert OPC DA data to MQTT

SMAR flow computers provide all their data via OPC DA, an older protocol that is gradually being replaced by OPC UA. Since UniCapture cannot collect OPC DA data, a third-party tool, "Cogent DataHub," is used to convert the necessary OPC DA data into MQTT. To facilitate this, all required software, including Cogent DataHub, Mosquitto MQTT Broker, and UniCapture, are installed on the same PC. The system architecture is illustrated in Figure 2, Section 2. Once the data is available in the MQTT Broker, it becomes accessible to UniCapture.

3.3.2 Data in MQTT Broker

The data in the MQTT Broker mirrors the data from OPC DA, as no conversion occurs between OPC DA and the MQTT Broker. This presents several challenges due to the unique data formats used by SMAR flow computers, which need to be interpreted by UniCapture. The flexibility of UniCapture's edge data processing is crucial in addressing these challenges.

In this section, a few examples of the data conversion will be explained.

a. ASCII Conversion

For example, OPC DA provides information for the transmitter tags as ASCII characters in the form of an array. This information needs to be converted into human-readable text and stored in the database. The following is an example:

MQTT publishes the following JSON string:

{ "Value": [72,101,108,108,111,32,87,111,114,108,100,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32]}

MQTT provides an array of up to 32 characters, with the last characters being spaces (represented by ASCII number 32). For example, the following characters:

72,101,108,108,111,32,87,111,114,108,100 translates to "Hello World"

UniCapture DLL converts the MQTT string into text as follows:

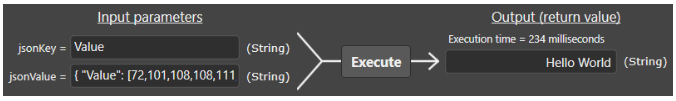


Fig 6. UniCapture dll converts ASCII to string

This DLL file is used to convert all ASCII characters provided by MQTT in array form into a string, which is then pushed to the database.

b. Date / Time conversion

One unique way the SMAR flow computer represents date and time is by presenting it in the form of an array. However, the array structure is quite complex. The following is the structure, where array index 1 is the first data on the left:

Array number 1: Last 2 digit of the year (needs to increased by 2000)

Array number 2: Represents the month

Array number 3: First 3 bits represent the week number Last 5 bits represent the date.

Array number 4: Represents the hour

Array number 5: Represents the minute

Array number 6 and 7: Represents the miliseconds. The value in the array is a decimal representation of a hexadecimal value. For example:

Array 6 value = 48 → DEC2HEX(48) = 30

Array 7 value = 117 → DEC2HEX(117) = 75

Millisecond value → HEX2DEC(7530) = 30000

Therefore the following array is interpreted as per below:

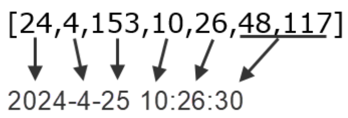


Fig 7. Date time representation from MQTT to database

All date-time representations from OPC are performed in this manner. Therefore, UniCapture needs to collect the data from MQTT, process it as described above, and then send the value to the database. The following is an example of this process performed by UniCapture:

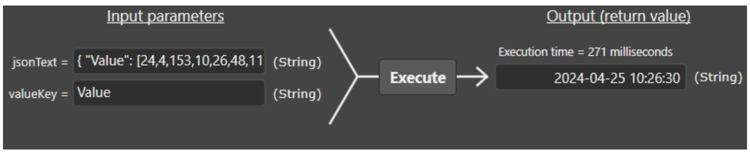


Fig 8. UniCapture MQTT date/time array to universal date time format

c. Same tags are shared by multiple streams

One of the complications in transferring data for this project is that some tags are shared by multiple streams. For example, the totalizer value for stream 1 and stream 2 uses the same tag. UniCapture needs to monitor the run number when the data is updated. If the incoming data is not for the associated stream that the configuration file monitors, all new data will be ignored.

To differentiate between flow computer streams, each stream is represented by its own FC_ID in the database design.

For example:

Table 6. Stream values input - output

Input	Output to database
Expected stream number = 2	
Stream no. = 1 QB = 1000	Ignored
Stream no. = 2 QB = 2000	FC_ID = 21 QB = 2000

Due to the sharing of tags, several special steps are required:

1. Create a CSV file

   This file defines which stream number is associated with which FC_ID. For example, in the above scenario, stream number 1 is associated with FC_ID 20 while stream number 2 is associated with FC_ID 21
2. Create a large JSON

   Collect individual MQTT values and compile them into one large MQTT JSON string containing all data from a single stream.
3. Develop a DLL file

   This file processes the large JSON string and sends the data to the database based on the correct stream number. If the stream number does not match the expected value for the associated stream, the data will not be written to the database. For instance, if the configuration file is specific to FC_ID 21, only data where stream number equals 2 will be processed and pushed to the database. Data for FC_ID 20 will be ignored and not pushed to the database.

d. Different alarm and event tags have different logic

As with the S600 flow computers, all events from SMAR flow computers are also recorded. However, SMAR flow computers use the same OPC tag to represent different types of events, which poses its own challenges. Within the same tags, there are seven different event types, each with its own logic. The appropriate logic to use depends on the data type provided by another OPC tag. Below are two examples that are quite complex:

- 32 Float data represented as array
  - For example, the first 4 data in the array is [109,231,251,61]
  - Converted to hexadecimal, the data becomes [6D,E7,FB,3D]
  - Representation of 32-bit float is read from right to left: 3DFBE76D
  - Converted using IEEE 754 standard [3], this represents value of 0.123

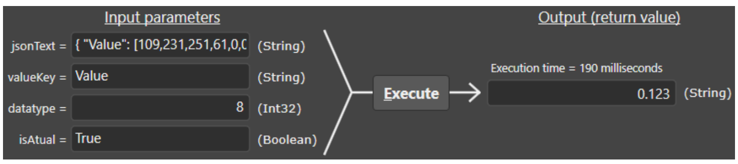


Fig 9. UniCapture converting array into 32 bit floating point data

- Array represents set of values

There is event type within the OPC where each array index represents different values. The following is an example:

Table 7. 11 elements array represents 5 different values

Description	Simulate status as int	Simulate Value as float (41043399 = 8.262597)				Transducer status as int	Transducer value as float (41043399 =8.262597)				Simulate en/disable
Array index	1	2	3	4	5	6	7	8	9	10	11
Values	128	153	51	4	65	128	153	51	4	65	1
Hex	80	99	33	4	41	80	99	33	4	41	1

The above array will provide the following output to be written into database:

Simulate Status – 128

Simulate Value - 8.262597

Transducer Status - 128

Transducer Value - 8.262597

Simulate en/disable - 1

The two examples above are handled by a UniCapture DLL program specifically created to address this scenario. Different projects will have their own challenges regarding data conversion requirements. These examples demonstrate that any data format can be converted into the desired format by performing edge analytics within UniCapture.

4 OTHER POTENTIALS

There are many other potentials for data transfer combined with capabilities of edge analytics. The following are some ideas that can be utilized for Flow Measurement.

4.1 Flow Computer Configuration Checks

It is vitally important that the basis of all flow calculations are understood and are consistent with the requirements of the application. Changing the configuration of a flow computer without proper control can result in reported quantities that are inconsistent with the true flow rates, introducing bias in to reported figures which can create a significant financial impact yet go easily unnoticed.

UniCapture will read the configuration from the flow computer and compare it with a controlled master-configuration. As soon as the there is a deviation in any configuration parameter the discrepancy will be highlighted so that further investigation can be performed.

The figure below shows and example of a configuration check.

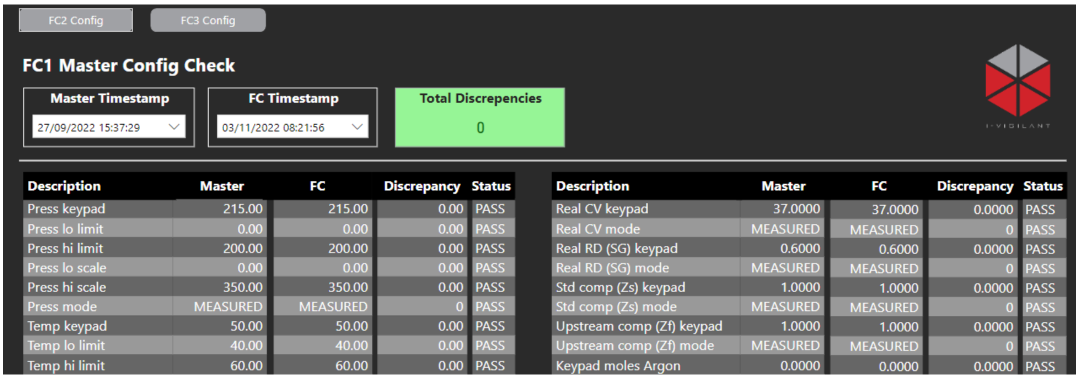


Fig 10. Example of Flow Computer Master Configuration Check

4.2 Flow Computer Calculation Checks

Flow calculation checks are often performed on fiscal and custody transfer measurement systems. These checks typically involve switching a stream into 'maintenance mode' and simulating or manually setting the inputs. This requires taking the stream offline to ensure the test results are not erroneously totalized and reported as true production. It is also common to break down the calculation checks into smaller steps, which introduces the possibility of missing some steps and allowing errors to go unnoticed. Additionally, there is always a risk that a parameter is left in keypad mode or the configuration is altered during the checks.

To overcome these limitations, UniCapture can be configured to perform the calculation checks on live data. The system does not need to be taken offline, put into maintenance mode, or checked against an offline application. All calculations can be checked in real-time, providing ultimate confidence that the system is running as it should be.

The figure below shows and example of a monthly calculation check by performing data averaging.

Monthly Calculation Validation Report

FC Tag: FC2 | Year: 2023 | Month: 3

I-VIGILANT

Description	Value
Frequency (Hz)	1,401.02
KF	60.00
MF	1.0000
T(Cal)	15.56
PCal (barg)	50.00
TCoeff (/degC)	4.120E-5
PCoeff (/barg)	6.000E-6
TBase (degC)	10.00
PBase (barg)	25.00
Temperature (degC)	42.4780
Pressure (barg)	67.5000

Description	Value
Methane	76.2053
Ethane	11.5271
Propane	4.0854
i-Butane	0.2237
n-Butane	0.2368
neo-Pentane	0.0000
i-Pentane	0.0401
n-Pentane	0.0257
Hexanes	0.0106
Heptanes	0.0045
Octanes	0.0000
Nonanes	0.0000
Decanes	0.0000
Nitrogen	2.8823
Carbon Dioxide	4.7543
H2O	0.0039
H2S	0.0001

Description	FC	UniCapture	Difference	Status
GVol obs pls (m³/h)	84,061.3981	84,061.3916	7.7016E-6	Pass
GVol flow rate corrected (m³/h)	84,163.4609	84,163.4592	2.0799E-6	Pass
Mass flow rate (t/h)	5,304.0060	5,304.0047	2.4076E-5	Pass
Std vol flow rate 15 (sm³/h)	6,033,451.0000	6,033,740.2659	-0.0048	Pass
Std vol flow rate 60 (MMscf/d)	213.3678	213.4945	-0.0594	Pass
Energy flow rate (GJ/h)	247,370.3000	247,369.6703	0.0003	Pass

Description	FC	UniCapture	Difference	Status
ctv	0.001109	0.001109	9.055931E-6	Pass
cpv	0.000105	0.000105	0.000000	Pass
vcf	1.001214	1.001214	-5.621719E-6	Pass
MW (g/mol)	20.7228	20.7228	0.0002	Pass
ZL	0.8585	0.8585	0.0024	Pass
ZB	0.9970	0.9970	0.0008	Pass
ZS	0.9970	0.9970	-0.0012	Pass
Line density (kg/sm³)	63.0203	63.0203	3.2503E-5	Pass
Base density (kg/sm³)	3577.9370	3577.7641	0.00	Pass
Ideal relative density	0.7155	0.7155	0.0011	Pass
Real relative density	0.7174	0.7174	0.0023	Pass
CV (MJ/kg)	46.6384	46.6383	0.0002	Pass

Fig 11. Example of monthly calculation check

4.3 Live Uncertainty

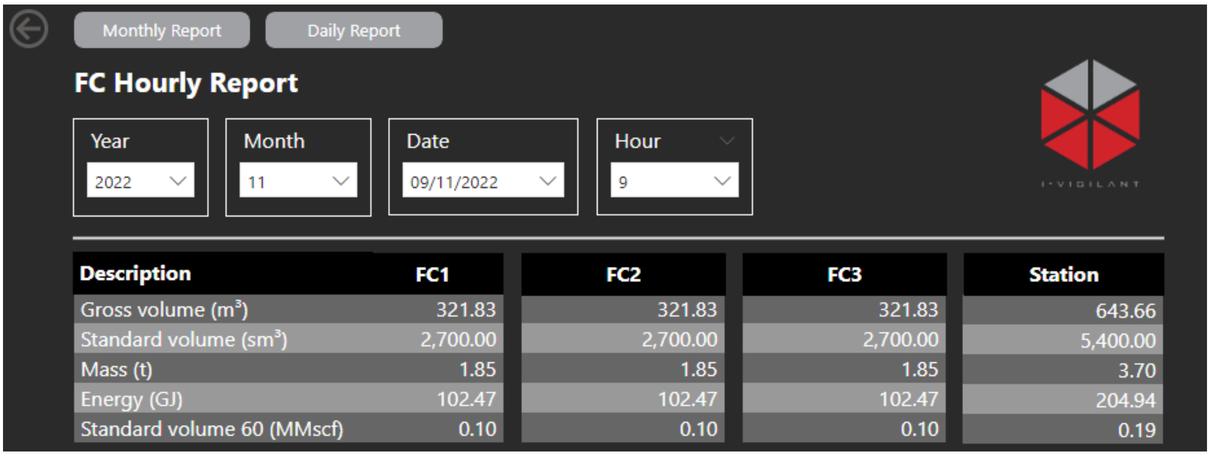
Live uncertainty calculations can be performed using UniCapture under operating conditions. While this uncertainty figure is meaningless unless it is known that the system is installed and operating as designed, when combined with data validation and other condition monitoring techniques to ensure the system is functioning normally, the uncertainty can provide confidence that the measurement is within contractual limits.

4.4 Periodic Data Reporting

Hourly and daily report data can be read directly from the flow computer and displayed, eliminating the need for remote logging to extract the data. This means no more trying to reconstruct flow computer reports from historian databases. The data can be read directly from the flow computer registers and displayed in the reports.

Period reports can be configured to be displayed in a format similar to the flow computer reports and can also be trended to provide reliable and flexible reporting of the true fiscal/custody transfer totals.

The figure below shows an example of an hourly report read from the flow computer.



Description	FC1	FC2	FC3	Station
Gross volume (m^3)	321.83	321.83	321.83	643.66
Standard volume (sm^3)	2,700.00	2,700.00	2,700.00	5,400.00
Mass (t)	1.85	1.85	1.85	3.70
Energy (GJ)	102.47	102.47	102.47	204.94
Standard volume 60 (MMscf)	0.10	0.10	0.10	0.19

Fig 12. Example of hourly report

4.5 Data Trending

All data saved in the database can be trended. This can be using any available tool such as Power BI, excel or any other visualization tool.

4.6 Data Collector for Condition Based Monitoring (CBM) Tool

With the flexibility of UniCapture, it can be utilized to collect the required data for any type of Condition-Based Monitoring (CBM). i-Vigilant uses UniCapture for data capture of its GC Analysis Software (GCAS). It is also used for data capture in i-Vigilant's new condition-based monitoring software called SMA – Smart Metering Analytics, which includes the "Prognosis" module for DP diagnostics and the "Ultrasys" module for ultrasonic meter CBM.

Since UniCapture handles the data transfer to the CBM-required database, the CBM design can focus on its full diagnostic capabilities, while the data gathering module is managed entirely by UniCapture.

4.7 Modbus Server and OPC Server capabilities

UniCapture has the capability to provide data for other systems. For example, UniCapture can collect all Modbus data from multiple flow computers, perform edge analytics, and then act as an OPC server or Modbus server, allowing other systems to collect data from UniCapture. This setup is ideal for providing data to HMI systems while performing edge analytics that add value to both the HMI and flow computer systems.

4.8 If You can Think It You can Do It

The range of applications for UniCapture extends far beyond the measurement industry. With its capability to communicate using multiple protocols, the potential uses of the system are limited only by one's imagination.

As an in-house project, i-Vigilant utilizes UniCapture in combination with Zigbee thermometers and Wi-Fi switches to control heating in the office. By converting Zigbee data into MQTT, UniCapture can monitor the temperature in every room within the i-Vigilant office. By setting temperature setpoints, UniCapture controls the heating system, helping i-Vigilant reduce electricity consumption.

5 CONCLUSION

This paper has discussed the application of a technology to address a problem within 3R Petroleum, specifically in collecting and performing data conversion before sending the data to a database for the purpose of generating XML files. Several examples have been provided to demonstrate the extensive capabilities of the available software in solving complex problems.

Additionally, the potential uses of the software extend beyond the metering industry, showcasing its versatility in various fields. The flexibility and adaptability of this technology highlight its value as a robust solution for diverse data management challenges.

In conclusion, the demonstrated success of this technology in addressing specific issues at 3R Petroleum, combined with its broad applicability, underscores its potential as a transformative tool in both industry-specific and broader contexts. The ability to seamlessly integrate and process data from multiple sources not only enhances operational efficiency but also opens new avenues for innovation and optimization across various sectors.

6 REFERENCES

[1] Modbus Organization, Inc., “Modbus Spec,” [Online]. Available: https://www.modbus.org/specs.php.

[2] OPC Foundation, “OPC Foundation,” [Online]. Available: https://opcfoundation.org/.

[3] IEEE Computer Society, “IEEE Standard for Floating-Point Arithmetic,” *IEEE STD 754,* 2019.