Reverse engineering the RUBICON efficiency report

By Samuel Hutchinson, December 2019

Abstract: This report summarises the processes of how the hydrology group have developed tools to calculate water balances of a pool[1] in the water delivery network. This ground up approach has enabled the ability to audit the branch efficiency report that RUBICON can supply. The branch efficiency report is a water balance of every pool in a branch that calculates the volume of water delivered, the difference between the delivered and the amount that entered the pool (the gap).

Generating topological awareness of pools

To the extent that the hydrology team have investigated, no accurate and concise linkage between assets is available in a format easy to work with. If one wanted to understand the relationship between a meter on a property e.g. M1893/1 and its nearest regulator (in this case Andreattas Reg.), one would have to look up that meter on network visualiser or MI maps and manually explore the relationship through laterals to the nearest upstream regulator, then find the object numbers[2] of each asset to get the required data from a production database, e.g. SC_EVENT_LOG . If one was wanting to find all the meters in an entire pool it can easily be seen that this method does not scale.

A branch summary can be exported as a comma separated file (csv). This file contains information about a branch and the objects in that branch. Unfortunately, it lacks a lot of features. It does not link laterals to a pool, nor provide object numbers, and contains many inconsistencies. By manually working through the csv and comparing with network visualiser, the laterals can be defined and inconsistencies removed. This csv is still not in a format to perform routine large searches on a branch to get data.

The next step is to parse through the csv and generate a more hierarchical link of all the topology in a branch. A program was written to do just this. It moves though the csv and builds up a structure of the network that makes links between any point in a branch and its upstream and downstream neighbours. It finds the object numbers of that asset and makes links in a table so that any relationship required for analysis can be easily extracted later. See example output in Table 1 below. Additional tools have been developed include methods for getting a list of object numbers between any two points.

```
Upstream objects…                                              OBJECT  LINK    LINK TYPE
└── OVERS REGULATOR - Regulator (30840)                        30822   30840   Regulator
    ├── M2615A/1 - D/S Meter (67510)                           30840   67510   D/S Meter
    ├── M2038/2 - D/S Meter (64930)                            30840   64930   D/S Meter
    ├── N665/P - D/S Meter (70693)                             30840   70693   D/S Meter
    ├── M2615/D - D/S Meter (70048)                            30840   70048   D/S Meter
    ├── M2038C/1 - D/S Meter (64933)                           30840   64933   D/S Meter
    ├── M1631H/1 - D/S Meter (67513)                           30840   67513   D/S Meter
    ├── OT L185 - D/S Offtake (30862)                          30862   64936   D/S Meter
    │   ├── M2038B/1 - D/S Meter (64936)                       30862   64939   D/S Meter
    │   ├── M1997/2 - D/S Meter (64939)                        30862   64942   D/S Meter
    │   ├── M1997/3 - D/S Meter (64942)                        30862   64954   D/S Meter
    │   ├── M1985/1 - D/S Meter (64954)                        30862   57141   D/S Scour Valve
    │   └── SV 185-1 - D/S Scour Valve (57141)                 30840   30862   D/S Offtake
    ├── M1631D/1 - D/S Meter (68062)                           30840   68062   D/S Meter
    ├── MC10 TEMPORALIS ESCAPE - D/S Escape (55372)            30840   55372   D/S Escape
    └── QUARY RD REGULATOR - Regulator (30868)                 30840   30868   Regulator
        └── Downstream objects…
```

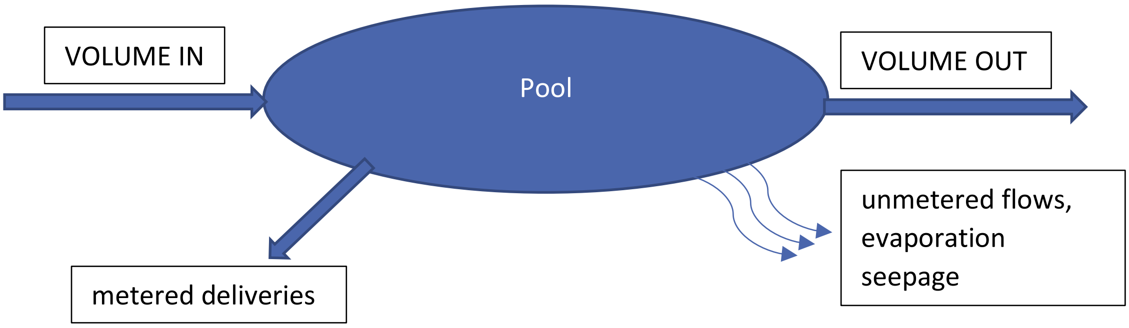
*Table 1: This is an example output of the parsing program. The output has been truncated to only visualise the relationships between Overs and Quarry Rd in LVBC. **NB**: Object numbers in the topology are in brackets and D/S means downstream.*

[1] For the purpose of this study a pool is anything between two main branch regulators.

[2] The unique identifier for each object in the MI databases. The object number ties all the information about that asset; e.g. flow, gate position, battery voltage, etc.

Working with metering data

To generate the efficiency report for a pool, the water balance is calculated. The inputs of the calculation are the flows in, delivered to, and out of the pool. The output is whatever the difference is between the input and the output (minus deliveries). In this study the water balance includes anything that can be measured, and excludes: evaporation, seepage, unmetered flows[3].



This study explored two sources for water volumes in the balance calculation. The first source described is calculating the numerical integral of the flow of water through a certain meter over time. The second uses the total volume accumulated as calculated by the RTU at the gate and periodically sent to the production database.

Volume from numerical integration of flows

To calculate the total volume through a meter between a given time, all the flows of that meter are integrated numerically against time using the trapezoidal integration rule. Since datapoints of the flow rates are relatively close for active sites (see Figure 1) using this rule is fairly acceptable.

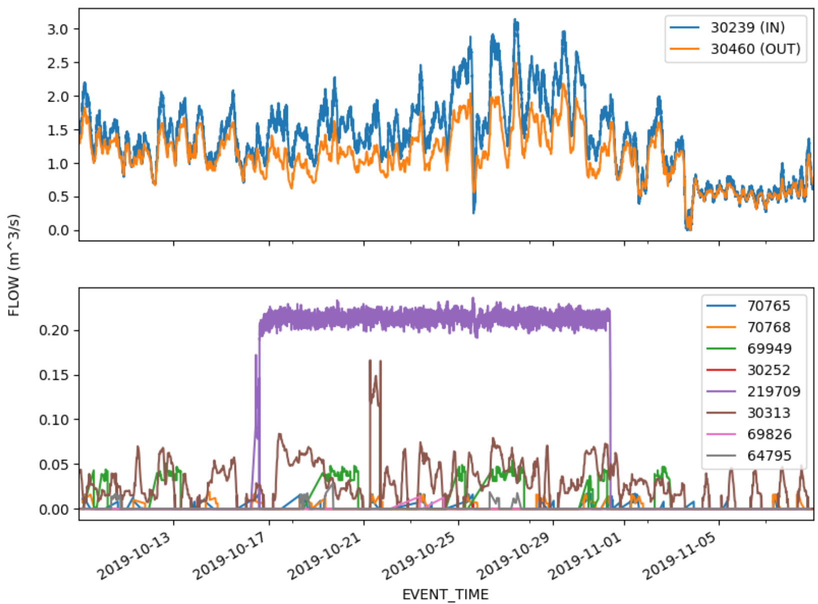


Figure 1: Flow data for all meters within the Andreattas pool over a period of a month. Object numbers are cited. The upper sub-plot is of branch regulators IN and OUT of the pool (Andreattas and Jones respectively). The lower sub-plot is the flow rates of the meters with the pool.

[3] Unmetered flows are any flow that is not described in the events database; meters without telemetry, unauthorised access, leaks.

Volume from RTU totaliser

The RTU keeps track of the volume of water that has passed through the meter over time. It does this by doing its own integration of flow in real time. This occurs every few seconds in the RTU and then around twice a day (depending on the polling configuration of the gate) this totaliser value is sent over SCADA to MI. Calculating the volume of water between two time periods should be as trivial as taking the totaliser value at those two time periods and subtracting the earlier one from the later. However, this is not the case.

There are a few situations that can occur to make this data messy (no longer monotonically increasing) and require some data processing to address. The first and most obvious artefact of the totaliser is that the value must periodically reset back to zero. Generally, this is done at the beginning of the season, but after some investigation it seems it can happen whenever the RTU resets. This situation is easily addressed by just checking when the series goes from a non-zero value to zero and adding the last non-zero value to the later points in the series. See Figure 2.

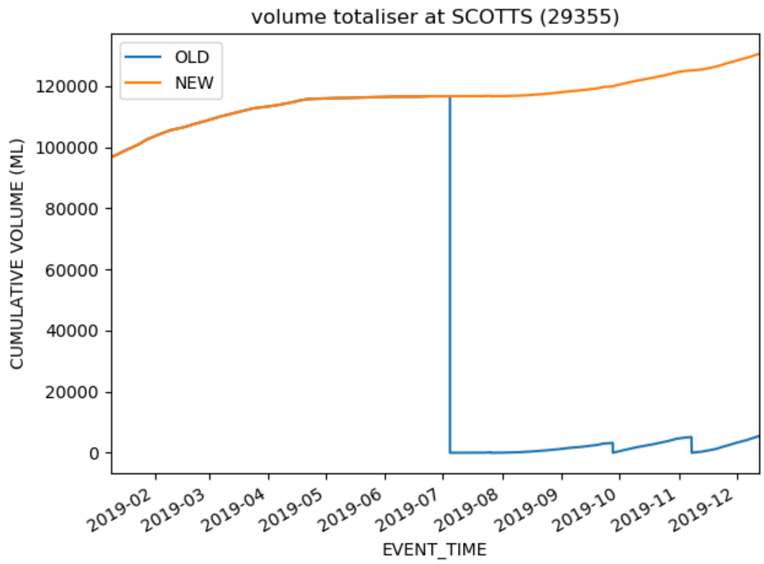


Figure 2: Totaliser volume through Scott's Rd Regulator. The "Old" data series is as taken from the SCADA database, the "NEW" values are a new data series that has had the resets to zero cleaned up.

If the RTU has reset (totaliser has gone to zero) and has been offline, it will continue to calculate the totaliser and transmit the most up to date value when it comes online again. This can cause there to never be a "0.0" value in the database when it has reset, making it difficult to track the reset and handle the data. Additionally, there are occasionally drops to zero in the database that do not coincide with a reset to zero (as the posterior values match the previous totaliser value). Sometimes this effect can be so drastic that the totaliser value ramps down and back up again over several tens of database entries. These artefacts are illustrated in Figure 3 and Figure 4. The causes for this remain a curiosity. To handle all these artefacts the following logic test is performed:

```
For all the values in the period we are interested in:
      if the (value - sensitivity) < all of the next 100 values,
            then consider this a reset and handle accordingly
      otherwise,
            treat as an artefact and do not reset
```

The sensitivity value (1.0 used) is to prevent an artefact caused by the quantisation of the decimal places in the totaliser occasionally causing the totaliser to break the monotonic increasing rule. E.g. 14.84500, 14.87500, 14.87000, 14.89000.

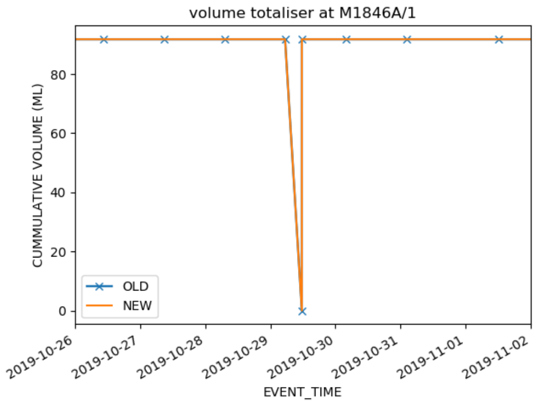


Figure 3: An example of a brief drop to zero in the totaliser. Datapoints are marked with "X"s.

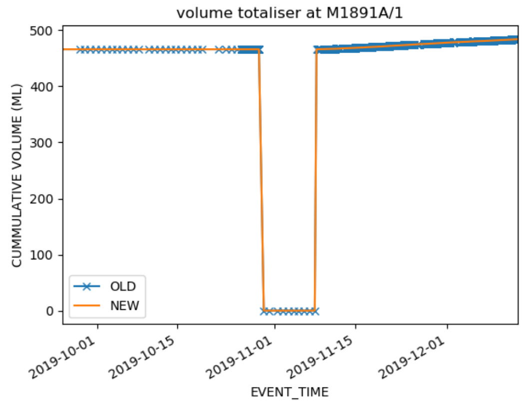


Figure 4: An example of a drop to zero for a period of time and a return. Datapoints are marked with "X"s.

One can note that in dealing with the artefacts illustrated in Figure 3 and Figure 4, there is a possibility that the data might drop to zero at the end of a period we are interested in, giving an incorrect calculation of volume. This situation is handled by taking the maximum value of the cumulative volume for that period. In other words:

```
Volume = Max(Cumulative flow for period) - cumulative flow at beginning
```

This cannot handle the case where the flow is zero at the beginning of a period as it is very difficult to tell whether a jump upwards is due to a zero error or a huge flow demand at the meter. If this situation occurs, the volume may be hugely overestimated. By combining the two methods (manual integration and RTU totaliser) this situation can be flagged and dealt with.

Summary of results

Conclusion and further work

- These tools are very pertinent to honours project (leak detection)
- This is highlighted some major differences about water balance assumptions between RUBICON and MI.
- Need incorporate unmetered data such as sites without telemetry
- Need to estimate/calculate seepage and evaporation which requires the need to include channels in the topology and their dimensions