LIFT IWS Challenge 2023

Optimizing System Flushing With a Smart Phone and a Real Time Model

1 The Team

The Newport News Waterworks Team consists of the four members listed below. There were no changes to the team during this project.

- Chris Basford, PE — Team Lead, Engineer
- Sherry Williams — Laboratory Manager
- Mike Lawson — Operations Coordinator
- Jon Hudson — Programmer

All members are employed by Newport News Waterworks except for Jon Hudson who is employed by Marathon Consulting. Jon has worked on Waterworks IT projects since 2018.

2 Problem Statement

The Waterworks distribution system consists of 1700 miles of pipe and 135,000 customers, the primary disinfectant being chloramine. During summer months, the system is consistently flushed both manually with fire hydrants and with automatic devices. Flushing site locations are selected based on laboratory readings, customer complaints, and historical knowledge.

The primary goals of this project are to:

1. Maintain total chlorine above 1 ppm to all customers,
2. Minimize time spent flushing,
3. Minimize volume flushed and
4. Track total volume of water flushed.

This project will seek to link field flushing activity to the real time distribution model via a smart phone application. A separate thread of the real time model will be created to run daily and calculate where and how much flushing is required to bring quality to desired levels. A dashboard will be used to provide the daily list of prioritized flushing sites.

3 The Solution

3.1 Recap and Modifications

This project connects field personnel with the hydraulic model to enable sending and receiving information. The clean interfaces used on the data entry and dashboard screens lead to quick acceptance by operations personnel and makes operational information readily available to office personnel.

3.1.1 Mobile Device Application

The original intent was to write a specific application for the mobile device. This was completed and tested on an Android phone. However, when we tried to load the iPhones in use by the utility, the available hardware was too old to load the required software versions. As a work around, a Google Gmail survey form was used for data collection, and an application was written to access the survey data via the Google API.

3.1.2 Additional Data Collection

When working on the new flushing code, it became apparent that valve position and automatic flushing were not being adequately captured and this would impact the results. Two additional forms were created for the capture of this information and the real time model was modified to read this data and make changes as indicated. Adding these two data elements required very little additional work as it fit within the framework already established for the project, and the existing load routine used by the real time model was used for the data import.

3.1.3 Fire Flow

The utility typically conducts around 50 flow tests each year. Automated tools are in place to simulate each flow test with the hydraulic model; however, this still required the engineer to receive an email from operations and enter the flow test results. When the simulation indicated a potential system problem, the engineer would then call operations and discuss additional work. Operations personnel would revisit the site and check valve positions, then rerun the flow test if appropriate. This process added days or weeks to the overall time for completion and added additional driving time. With the framework for submitting data from the field in place, the only additional work was to write a module to validate the test data, find any valves that may be closed, and publish the results to a dashboard.

3.1.4 Real Time Model Code Modification

In the previous version of the real time model, automatic flushing hours were treated as an integer where the end time must occur after the begin time. This code had worked well since timed automated flushing always began after midnight and changes only occurred on the hour. After implementing this project, we soon received an entry where flushing began at 11:30 PM and ended at 4:00 AM. Code changes were implemented to allow flushing across a date change, and the integer values for hour were changed to represent hours and minutes, so 11:30 PM is now represented as 2330. Making this change proved to be quite simple, requiring only a minor update to the validation code receiving the field data and to two database views. Field crews traditionally work with time in this format, and we now have a more robust solution that does not impose unneeded restrictions on flushing times. A flow chart showing modifications to the real time model is found in Appendices 4.4 and 4.6.

3.1.5 Flushing Calculation

The original concept was to calculate required flushing daily in a separate thread of the model. This approach offered the advantage of using the model to calculate flush duration but had drawbacks on additional complexity and timeliness.

The initial flushing solution was written using the most recent simulation results loaded to the database. This was done simply because it allowed focus on the solution approach without the added complexity of running a new instance of the model. After reviewing results, it made more sense to continue this approach and add it as part of the real time model, solving once per hour. During the actual field work, flushing at a site may impact nearby sites as well. A daily solution would not consider this impact, nor would it consider other changes throughout the day such as valve position or pumping. When considering flush time and driving time, it makes little sense to execute the solution more frequently than hourly, so the code is set to run once per hour from 7:00 AM – 3:00 PM on weekdays. The frequency of code execution is easily changed should the need arise.

3.2 Description of Implementation

This project utilizes mobile devices to provide field updates to the hydraulic model. The model then returns specific information on flushing and potentially closed valves back to the mobile devices.

The mobile application uses Google Forms to generate a survey form for data input. The four forms created are:

- Set valve position,
- Change settings on automatic flushing device,
- Perform manual flushing and
- Perform fire flow test.

Images of these forms are found in Appendix 4.1.

A custom application running on Windows scheduler employs the Google Sheets API to retrieve new field data, then validates entries and loads them to a SQL Server Database. Changes in valve position or flushing are incorporated in the next simulation of the real time model. Each hour, a flushing module is called by the real time model to recalculate flushing priority and publish results.

Fire flow testing was added to this project because the workflow and data fit well within this framework. When a new flow test is entered from the field, the data is validated in a stand-alone hydraulic model. Within minutes, results are available on the user's mobile device indicating valid test, model error, or potential closed valves. In the case of potential closed valves, a map displays the potentially closed valves along with a percentage indicating the likelihood of the valve being closed.

All data resides in a SQL Server database and dashboards present up to date field activity to the entire organization.

3.3 Value Added by Solution

With this solution in place,

- Field personnel know where to flush,
- Management and Engineering know how much is being flushed,
- All pipes are evaluated for needed flushing,
- Flow tests are validated while on site.

Prior to this solution, flushing was managed with spreadsheets and a mainframe workorder system. Flushing sites were selected by identification from the laboratory during routine sampling, and flush volume was determined by trial and error. There was no accessible dashboard that provided up to date information on location and volume of flushing.

Because of these data problems, the real time model was often not accurately representing field conditions and was therefore generating incorrect water quality results. As these results are relied upon for various reporting, we want them to be as correct as possible.

For several years, the utility has automatically simulated each flow test. As described, sometimes there are system problems that can be easily corrected while still on site, and this project enables that. Other times there may be a system problem unrelated to valves, or a problem with the model. In these cases, this project eliminates valve position as being suspect, leading to faster resolution. This flow data is regularly provided to external customers for design purposes. Combining the field tests with the model provides confidence that we are providing correct information.

3.4 Next Steps

The initial roll out of this project has been successful and we plan to continue development of this project. The first step we will take is to improve security on the data forms used in the field. After our new work order system is in place, we will consider using its mobile component to replace the Google survey forms and revisit the phone application developed during this project. After generating more data, we will better optimize the location and flow rates of our automatic flushing. Initially this will be a manual solution, and later automated as appropriate. We also plan to use this solution seasonally to indicate when automated flushing should begin and end.

3.5 Financial Support

As the utility has already developed their own real time model and possesses the coding expertise, no purchases were budgeted. We assigned three existing PowerBI licenses to personnel who may not otherwise have received them. The individual license costs are approximately $100 each.

The hours spent on the solution break down as follows.

Mobile phone application	16 hours
Google Forms and API	40 hours (most of this was learning to use the API)
Flushing Code	24 hours
Real Time Model Changes	8 hours
Fire flow Code	8 hours
Dashboards	16 hours
Concept Discussions	8 hours
Final Report	24 hours
Total Time	**144 hours**

3.6 Replication by Other Utilities

This project can be easily replicated by any utility for little to no cost other than time. The only requirement for this project is to have a hydraulic model. These tools were used to create this application.

- Google Gmail account
- Google developer account
- EPANET
- EPANET Developers Toolkit
- SQL Server Database
- SQL Server Management Studio
- Visual Studio
- PowerBI
- Virtual Server

All these products can be obtained free of charge for individual development purposes. Additionally, each product can be changed out. For example, a different database could be used, or code could be written in a different development environment. Replacing PowerBI could be accomplished with the Google API by generating email to send the results to users. While many utilities may use a proprietary hydraulic model, if it allows for custom code, it can be used. Otherwise, many models have functionality to export EPANET model files, or the adept user could generate a suitable EPANET model from their GIS.

While this project capitalizes on the real time model, this is not a requirement, and a stand-alone simulation could be substituted.

To support running simultaneous EPANET simulations using the C# language, Newport News previously wrote specialized code to call the EPANET library. For anyone new to this type of work, it would be easier to use the legacy version of the EPANET Toolkit if programming in C#, or alternatively to write code in C++.

3.7 Difficulties and Mitigation

The team was presented with two challenges during this project. test

1. The timeframe coincided with the anticipated go live date for a new work order system. Team members top priority was to the work order project, which caused significant delay on when the work could be started. We were aware of this at the start of the project but given our experience with model automation, we believed we could complete all necessary work.
2. This was our first attempt at an iPhone application, and we did not anticipate the difficulty of loading the application with the equipment owned by the utility. When this problem became apparent, we were well into July and did not have time to solve the application load problem. This was mitigated by moving to the Google platform, which added an extra week to the timeline.

3.8 How Results are Used and Communicated

All results are saved in the production database and available in preconfigured dashboards.

3.8.1 Immediate Use by Field Personnel

The PowerBI dashboards created always display the current valve and flushing condition, as well as the ranked locations that need flushing.

Within minutes of inputting a fire flow test, the dashboard displays whether the validation passed, there is a model error, or there is a valve that is likely closed. With this information, the crew can immediately verify valve positions and rerun the flow test if needed.

3.8.2 Use by Utility

Everyone in the utility has the same access to data as the field crews. When questions arise, dashboards are available to provide current information. For example, during regular meetings between Water Production and Distribution Operations the question of flushing will arise. With this project, everyone can quickly see that the utility is currently flushing over 400,000 gpd, where the flushing is, and which areas are still in need of flushing.

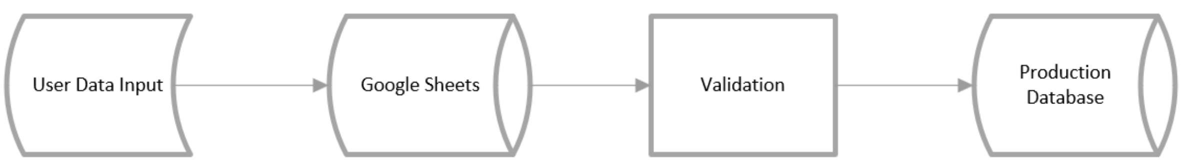
3.8.3 Use By Engineering

Model calibration is fundamental to much of the engineering work. Often calibration is performed only occasionally and based on a short time frame. With these processes, calibration becomes a continuous process. With the fire flow testing verification, the engineer is presented with only those tests where a model error is indicated. Additionally, the engineer can easily determine areas where the model has not been validated by a flow test, or where error tends to be higher. Likewise, a simple GIS view displays the water quality calibration of the model over any selected time frame.

Ultimately, the goal is to reduce the total volume of flushing and the hours spent on the program while providing water meeting quality requirements to all customers.

3.9 Data Streams QA QC

Data is input at the user's mobile device into one of four forms, and automatically transferred to the appropriate Google spreadsheet. On 5-minute time intervals, a custom application reads the next new entry in each spreadsheet, validates the data, and loads validated data to the utility's production database as seen in the following diagram.



A detailed flow chart of this process is found in Appendix 4.3.

Data validation ensures numbers are within ranges and input strings are trimmed to required length. When an entry fails validation, a note is posted to the database. If a bot were to overload the spreadsheets with entries, the code will still retrieve only one entry from each spreadsheet each 5 minutes. If bad data were to get loaded to the model, such as an extreme flush rate causing negative pressure, the model will return an error code. The model provides information only and all decisions are made by employees, so any failure of the model will not affect operations.

While the original mobile application was developed with user verification, there is currently no user verification process in place. One of the immediate action items is to allow only verified users to add data.

Though not part of this project, the solution is dependent on the water quality calibration of the model. There are dashboards in place that correlate laboratory sample results with the real time model predictions. This new data supplied from the field includes chlorine readings and will be incorporated into the water quality validation process. With all this sample data stored in a database and correlated with the model, the utility is positioned to pursue automatic quality calibration.

3.10 Analysis & Interpretation

The fire flow test component added to this solution is simply automating a process that was already in place. On the first day of field testing, the dashboard reflected a list of potentially closed valves. The first valve on the list was found buried with grass growing over the lid. Two people were needed to open the valve, so it had been closed for a long time. Upon rerunning the flow test, model validation passed with an error of less than one psi. With the valve opening, discolored water was created, and the pipe was flushed until clear. This flushing was logged in the application along with a note of why it occurred. As the utility was already validating all flow tests with modeling, the benefit of this automation is mainly saving time.

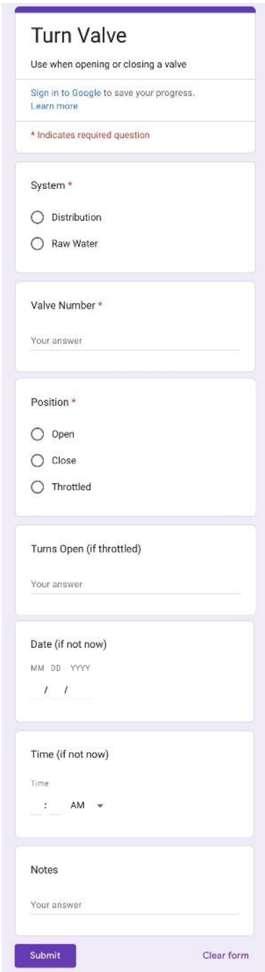
The flushing component is less straightforward because the water quality analysis is time dependent. The analyses results display the ranked flush locations along with a flush duration based on flowing 300 gpm. Prior to any manual flushing, the employee will take a chlorine reading to confirm flushing is needed. Once flushing is begun, it is continued until the chlorine reached acceptable levels. We will most likely need to increase the solution frequency of the real time model and plan to work with five- or ten-minute intervals. An example of one of the areas selected by the flushing module is found in Appendix 4.5

With the dashboard in place, everyone can see the daily automated flush volume. The annual retail cost of this volume exceeds $250,000 and there is incentive to better optimize this process.

4 APPENDICES

4.1 USER INPUT FORMS

These are the four forms used by field personnel to input data. The clean design leads to quick user acceptance.



Turn Valve

Use when opening or closing a valve

Sign in to Google to save your progress. Learn more

* Indicates required question

System *

- Distribution
- Raw Water

Valve Number *

Your answer

Position *

- Open
- Close
- Throttled

Turns Open (if throttled)

Your answer

Date (if not now)

MM DD YYYY

/ /

Time (if not now)

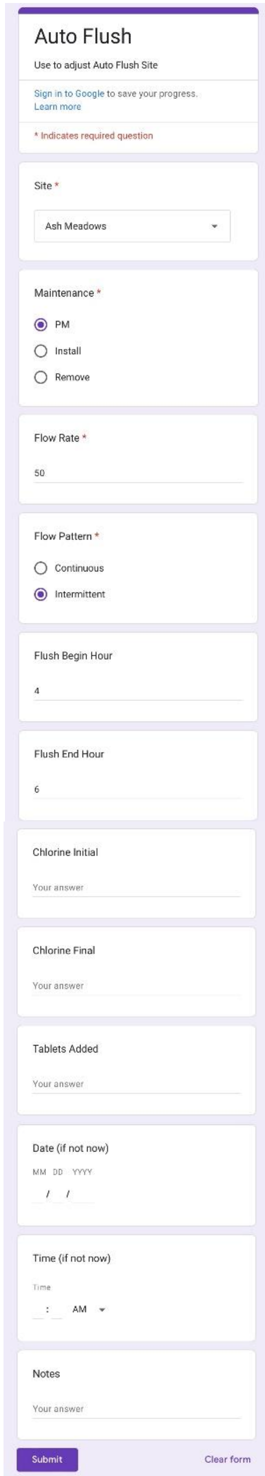
Time

: AM

Notes

Your answer

Submit Clear form



Auto Flush

Use to adjust Auto Flush Site

Sign in to Google to save your progress. Learn more

* Indicates required question

Site *

Ash Meadows

Maintenance *

- PM
- Install
- Remove

Flow Rate *

50

Flow Pattern *

- Continuous
- Intermittent

Flush Begin Hour

4

Flush End Hour

6

Chlorine Initial

Your answer

Chlorine Final

Your answer

Tablets Added

Your answer

Date (if not now)

MM DD YYYY

/ /

Time (if not now)

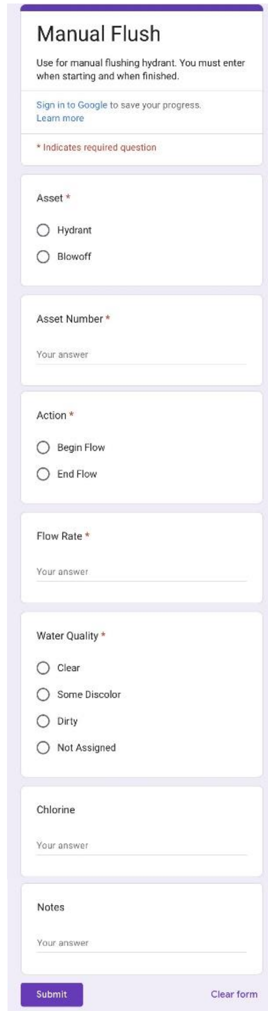
Time

: AM

Notes

Your answer

Submit Clear form



Manual Flush

Use for manual flushing hydrant. You must enter when starting and when finished.

Sign in to Google to save your progress. Learn more

* Indicates required question

Asset *

- Hydrant
- Blowoff

Asset Number *

Your answer

Action *

- Begin Flow
- End Flow

Flow Rate *

Your answer

Water Quality *

- Clear
- Some Discolor
- Dirty
- Not Assigned

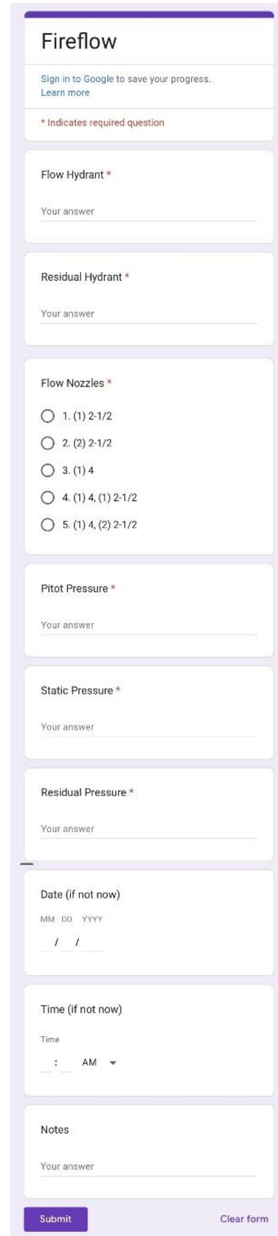
Chlorine

Your answer

Notes

Your answer

Submit Clear form



Fireflow

Sign in to Google to save your progress. Learn more

* Indicates required question

Flow Hydrant *

Your answer

Residual Hydrant *

Your answer

Flow Nozzles *

- 1. (1) 2-1/2
- 2. (2) 2-1/2
- 3. (1) 4
- 4. (1) 4, (1) 2-1/2
- 5. (1) 4, (2) 2-1/2

Pitot Pressure *

Your answer

Static Pressure *

Your answer

Residual Pressure *

Your answer

Date (if not now)

MM DD YYYY

/ /

Time (if not now)

Time

: AM

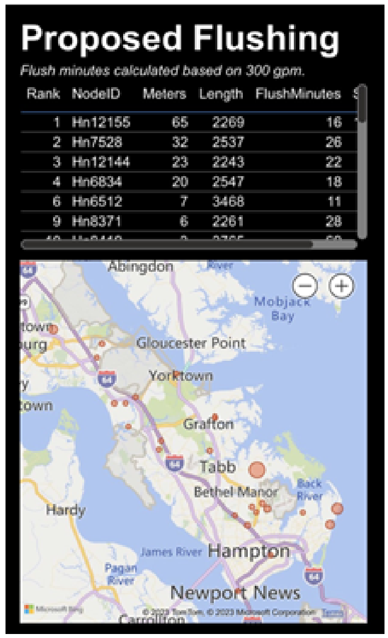
Notes

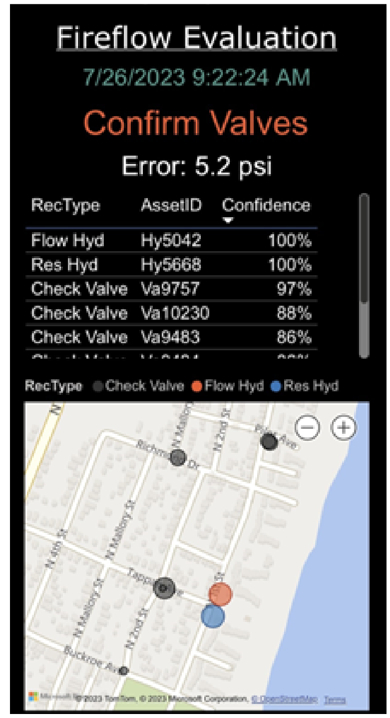
Your answer

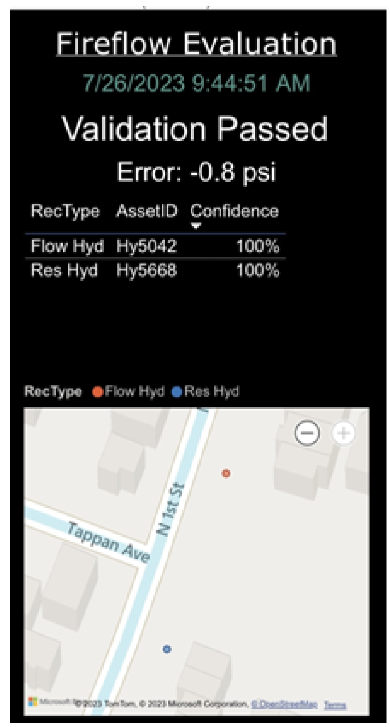
Submit Clear form

4.2 PowerBI Dashboards

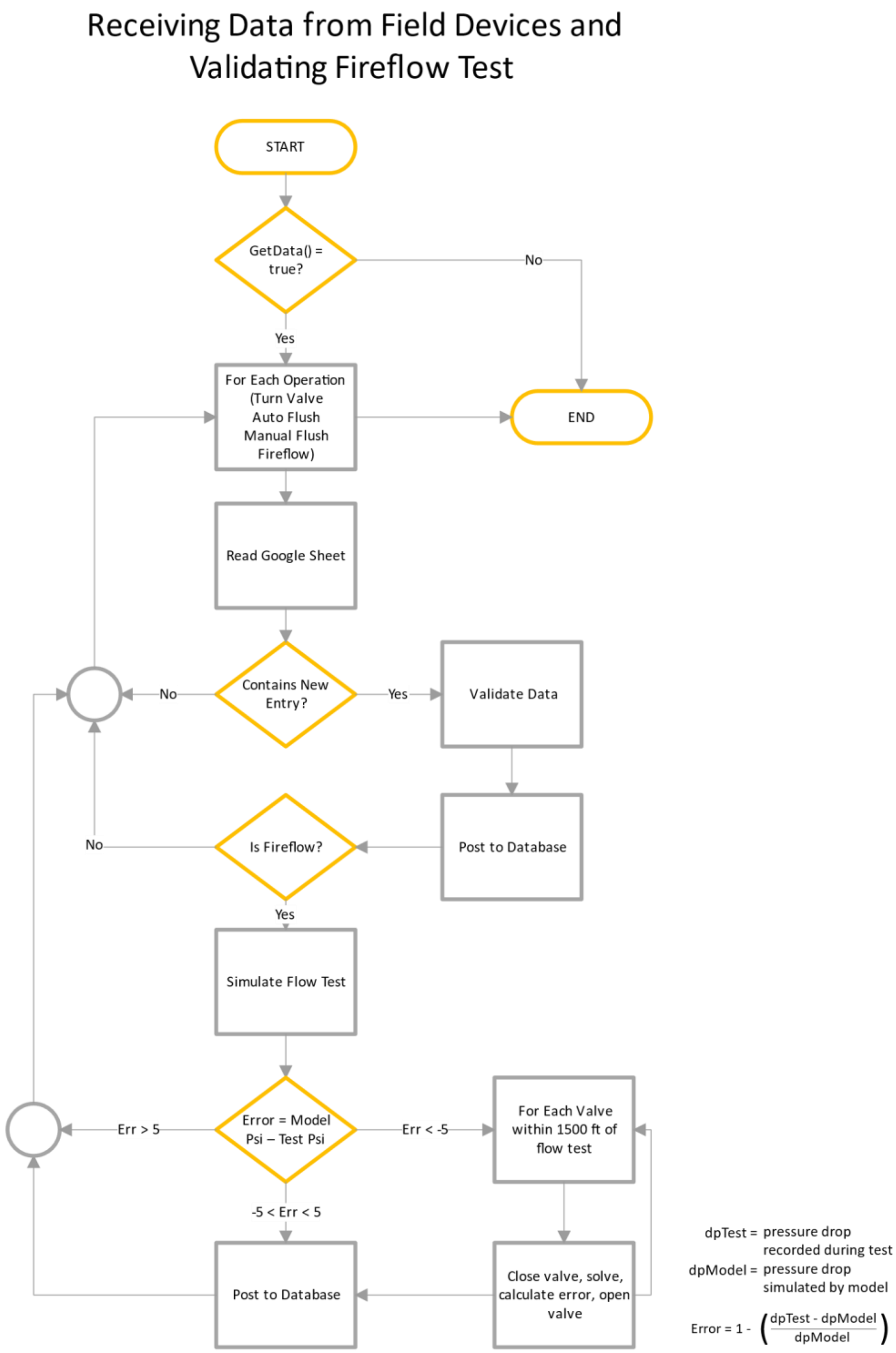
These are the three dashboards created for field use. On the right is the flow test where a valve was found closed. Notice the elapsed time between the two tests is 22 minutes. Normally this would be days or weeks.





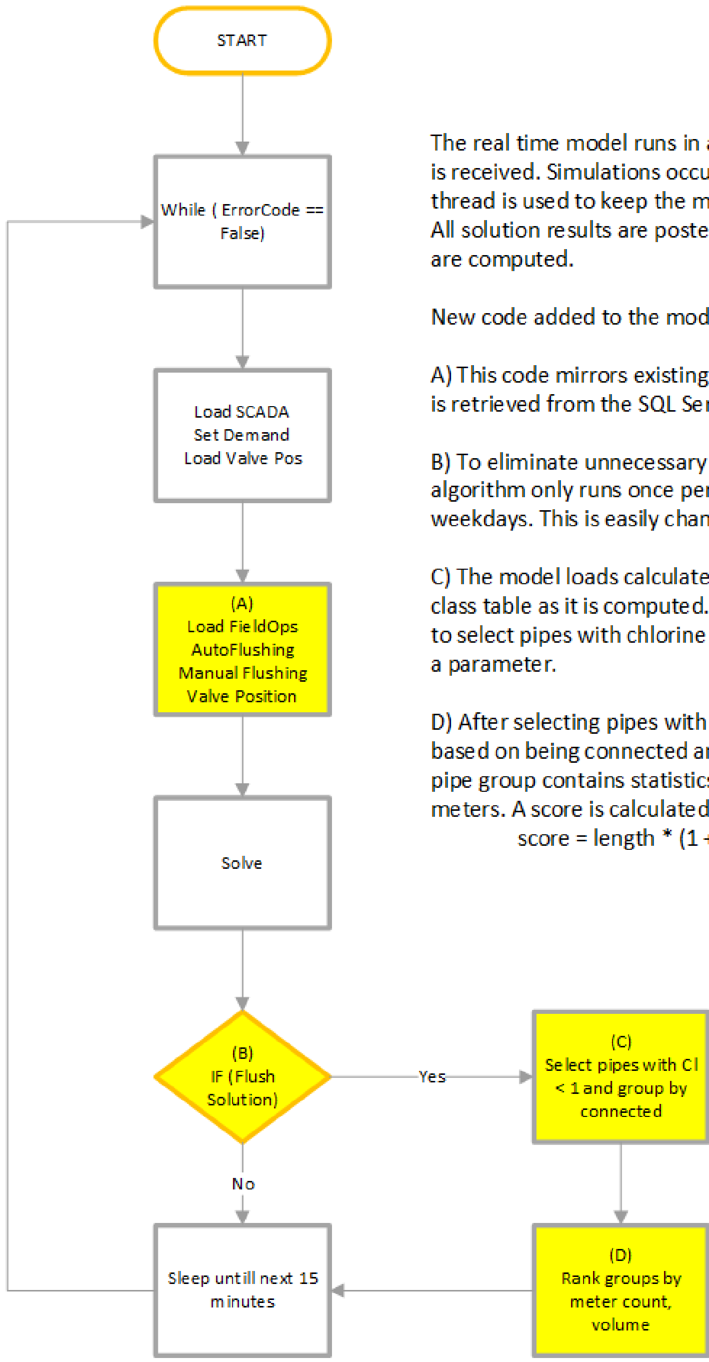


4.3 Flow Chart for Data Load Application



4.4 Flow Chart for Real Time Model

Real Time Model Modifications



The real time model runs in a continuous loop until an error code is received. Simulations occur each 15 minutes and a sleep thread is used to keep the model synchronized with clock time. All solution results are posted to a SQL Server Database as they are computed.

New code added to the model is highlighted in yellow.

A) This code mirrors existing code from the prior step where data is retrieved from the SQL Server and loaded to the model.

B) To eliminate unnecessary processing time, the flushing algorithm only runs once per hour, from 7:00 AM – 3:00 PM on weekdays. This is easily changed with a parameter.

C) The model loads calculated chlorine levels to a GIS feature class table as it is computed. This step is a simple database query to select pipes with chlorine < 1. The chlorine threshold is set as a parameter.

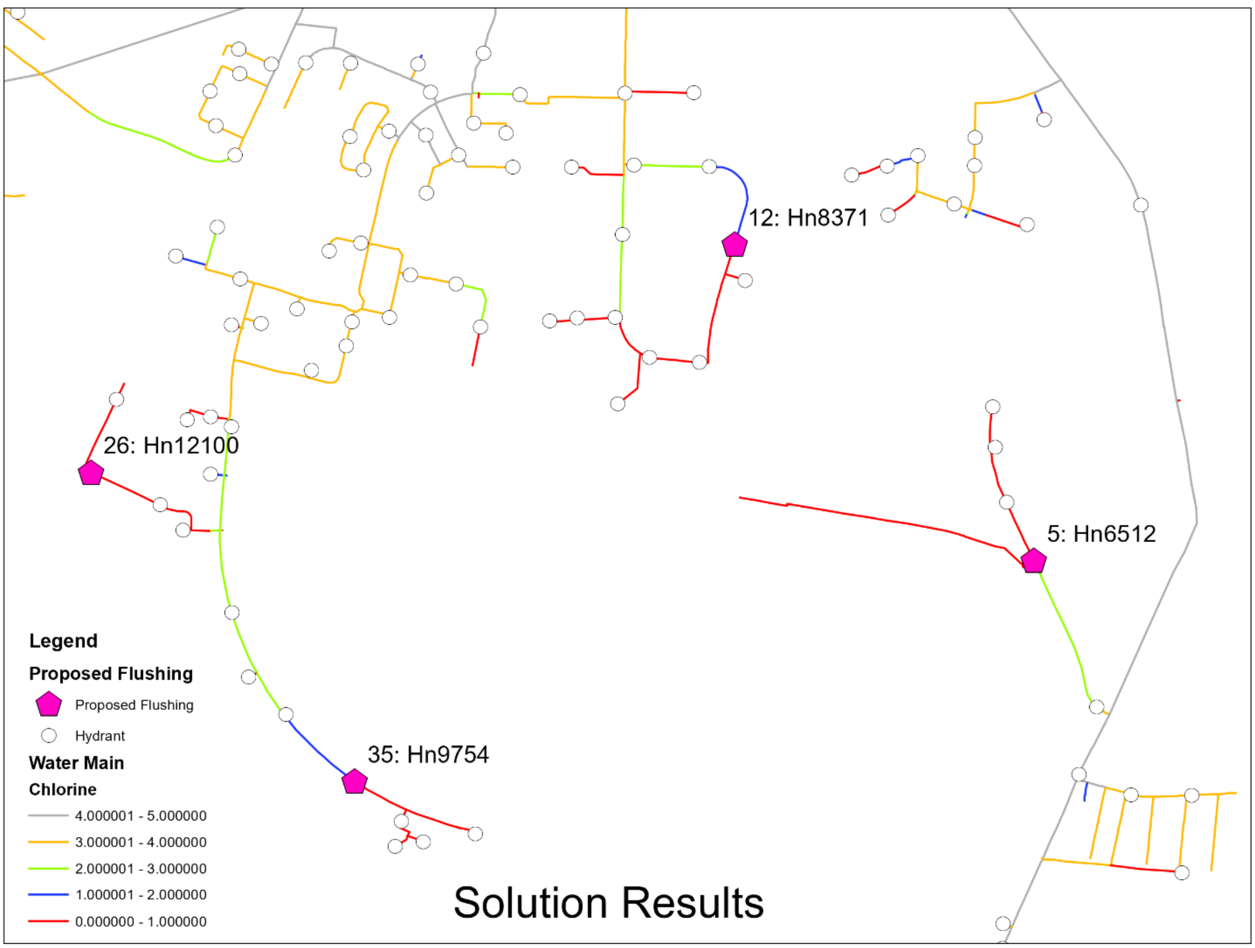
D) After selecting pipes with low chlorine, they are grouped based on being connected and assigned a group number. Each pipe group contains statistics on total length, volume, connected meters. A score is calculated as

score = length * (1 + meter count)

4.5 Sample Solution Results

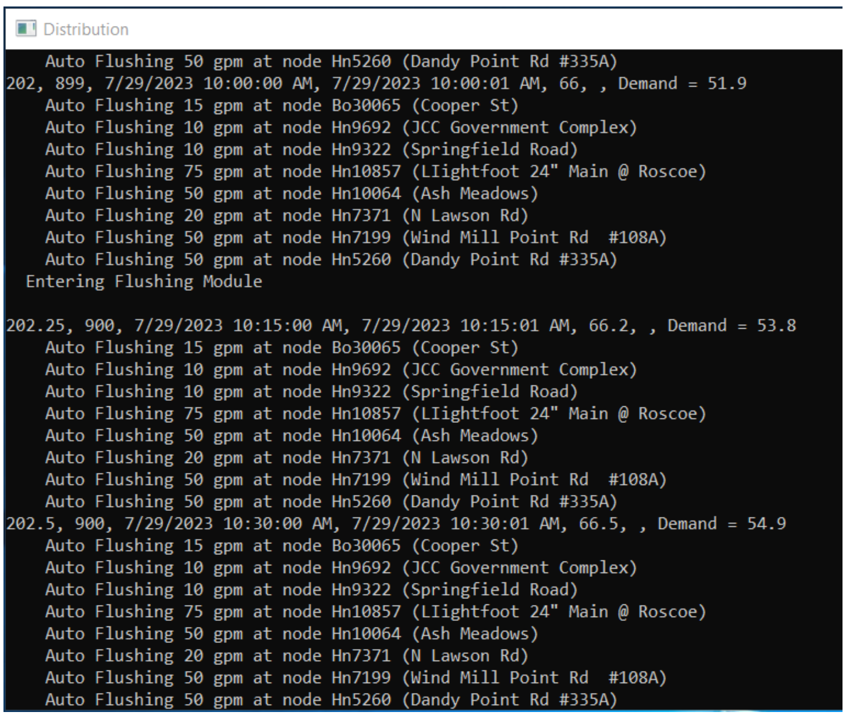
Referring to the map below, there are several pipes colored in red that have predicted chlorine less than 1 mgl. The algorithm selected four sites based on the volume and number of meters.

- Hn6512. The meters (not displayed) are all located on the bottom pipe yet there are no hydrants to flush along this pipe. The top pipe has no demand, so the same water age was calculated for each node, and the solution arbitrarily chose hydrant 6512. To prevent this situation, small demands were applied at each dead end, but it appears these demands need to be increased. The field personnel know to flush at dead ends when possible.
- Hn8371. Flushing could occur at any of the dead ends in this area.
- Hn9754. This is a well-known problem area and automated flushing was removed because of property owner complaints. Flushing this location will impact required flushing at Hn12100 and depending on customer demand, flushing at Hn12100 may not be recommended by the next solution.



4.6 Real Time Model Console Output

Newport News Waterworks has maintained a real time model for the past 10 years. This model uses the EPANET Developers Toolkit and runs in three threads solving hydraulics, water age, chlorine concentration, and water source. Solutions occur on 15-minute increments and receive input from SCADA and from Dispatch[1]. This model is written in the C# programming language and runs in a continuous loop on a virtual server. When logging into the server, a console (**Error! Reference source not found.**) displays information relating to the solution. The console output was modified to print flushing received from the field data and to indicate when the code has entered the flushing module. The console is typically only viewed when the model has stopped running. For viewing data, results are continuously updated in the SQL Server Database.



```
Distribution
    Auto Flushing 50 gpm at node Hn5260 (Dandy Point Rd #335A)
202, 899, 7/29/2023 10:00:00 AM, 7/29/2023 10:00:01 AM, 66, , Demand = 51.9
    Auto Flushing 15 gpm at node Bo30065 (Cooper St)
    Auto Flushing 10 gpm at node Hn9692 (JCC Government Complex)
    Auto Flushing 10 gpm at node Hn9322 (Springfield Road)
    Auto Flushing 75 gpm at node Hn10857 (LIightfoot 24" Main @ Roscoe)
    Auto Flushing 50 gpm at node Hn10064 (Ash Meadows)
    Auto Flushing 20 gpm at node Hn7371 (N Lawson Rd)
    Auto Flushing 50 gpm at node Hn7199 (Wind Mill Point Rd  #108A)
    Auto Flushing 50 gpm at node Hn5260 (Dandy Point Rd #335A)
  Entering Flushing Module

202.25, 900, 7/29/2023 10:15:00 AM, 7/29/2023 10:15:01 AM, 66.2, , Demand = 53.8
    Auto Flushing 15 gpm at node Bo30065 (Cooper St)
    Auto Flushing 10 gpm at node Hn9692 (JCC Government Complex)
    Auto Flushing 10 gpm at node Hn9322 (Springfield Road)
    Auto Flushing 75 gpm at node Hn10857 (LIightfoot 24" Main @ Roscoe)
    Auto Flushing 50 gpm at node Hn10064 (Ash Meadows)
    Auto Flushing 20 gpm at node Hn7371 (N Lawson Rd)
    Auto Flushing 50 gpm at node Hn7199 (Wind Mill Point Rd  #108A)
    Auto Flushing 50 gpm at node Hn5260 (Dandy Point Rd #335A)
202.5, 900, 7/29/2023 10:30:00 AM, 7/29/2023 10:30:01 AM, 66.5, , Demand = 54.9
    Auto Flushing 15 gpm at node Bo30065 (Cooper St)
    Auto Flushing 10 gpm at node Hn9692 (JCC Government Complex)
    Auto Flushing 10 gpm at node Hn9322 (Springfield Road)
    Auto Flushing 75 gpm at node Hn10857 (LIightfoot 24" Main @ Roscoe)
    Auto Flushing 50 gpm at node Hn10064 (Ash Meadows)
    Auto Flushing 20 gpm at node Hn7371 (N Lawson Rd)
    Auto Flushing 50 gpm at node Hn7199 (Wind Mill Point Rd  #108A)
    Auto Flushing 50 gpm at node Hn5260 (Dandy Point Rd #335A)
```

[1] When field crews shut off water, they notify dispatch who enter the information to a database. Custom code using the EPANET model then calculates the outage list. The real time model reads current outage information from this database.