Wavebuoy Processing

# Background

In Autumn 2021, Met Éireann and the Commissioner of Irish Lights will deploy 4 wave sensors on CIL navigational and metocean buoys around the Irish Coast. This trial project will assess the quality of the data for coastal applications, particularly coastal flood forecasting. Alongside the Seaview sensors on CIL buoys, at at least 1 site, a Datawell wave rider buoy will be deployed alongside for comparison.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | South Hunter | Splaugh | Finnis | Ballybunnion |
| Pressure | No | No | No | No |
| Wind Dir | Yes | Yes | Yes | Yes |
| Wind Speed | Yes | Yes | Yes | Yes |
| Temp | Yes | Yes | No | No |
| Dew Point | No | No | No | No |
| Humidity | No | No | No | No |
| Wave Period | Yes | Yes | Yes | Yes |
| Wave Height | Yes | Yes | Yes | Yes |
| Sea Temp | Yes | Yes | No | No |

# Data Processing Pipeline

## 2.1 High-Level Overview

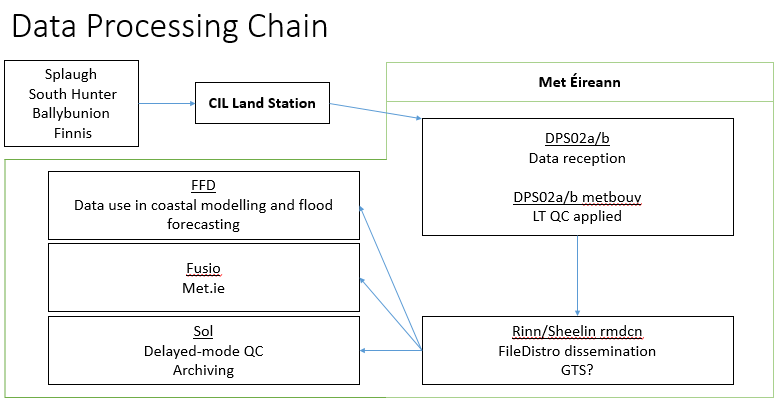


Figure 1: high level schematic of planned CIL metbuoy processing in Met Éireann.

* Data is collected in real-time on 4 buoys with sampling time of 20 minutes for wave data
* Data is transmitted to CIL land station by AIS every 6 minutes
  + Note: sampling time period > data transmission period, thus there will be duplicates
* CIL forward data to Met Éireann by FTP in both raw and QC’d form. Data is FTP’d to the existing CIL account and forwarded to the metbouy account.
  + In the future, CIL will cease FTP transmission. Met Éireann will then use the CIL API to retrieve data.
* The metbouy account:

1. Checks the data is valid:
   * File format (extension, datetime format, station name) is valid
   * Records in file (station and datetime) do not match ones already processed
   1. Applies real-time quality control to the data – see Section 3 for details.
   2. Creates output in form of csv and XML files
   3. Forwards data to Rinn/Sheelin servers

* Rinn/Sheelin host the FileDistro message switching application. This application can forward data by FTP to any user, internal or external. For example:
  + Sol: for data archiving and delayed-mode analysis of trial data
  + FFD: data used in real-time products and forecasting
  + Fusio: met.ie

## 2.2 Metbouy File-Processing

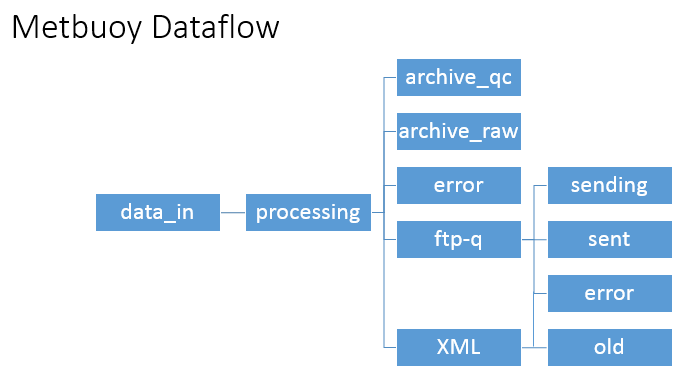


Figure 2: the operational data flow through the metbouy data directory and subdirectories.

### 2.3.1 Process\_cil.py

* The main processing script under the metbouy account is process\_cil.py, which will run as a cronjob. A second cronjob concat\_CIL\_xml.py compiles the latest data from all buoys into 1 XML file for met.ie.
* Incoming data arrives in the data\_in directory, which process\_cil.py checks periodically (currently set to every 20 minutes).
* When files are found, processing\_cil.py moves the file into the processing directory and begins initial pre-QC checks including:
  + Filename is valid: contains “csv” extension, station name and valid timestamp
  + Station is one of the expected stations
  + A file of this name has not been processed before, and if it has, the data in this file is different
  + That data in the row is different from data in the row before (this attempts to remove duplicate records, caused by the transmission frequency being greater than the observation frequency).
* Next, process\_cil.py reads the archive from archive\_qc (or initiates an archive if none exists for the station), and merges the new and archive records into a single dataframe.
  + When doing this, a note is made of what timestamps are “new” and if no new records are found, the processing ends
* The Quality Control process is run on this dataframe, with a list of parameters to QC included as an argument. This process outputs 2 dataframes:

1. FullReport: contains result from all tests applied to all parameters.
2. SummaryReport: subset of the fullReport, containing only the overall parameter result

* These two dataframes are written to csv files under ftp-q/sending.
* The summary report is further used to generate an XML file with the most recent report from the station, this report is saved in the XML directory.
* The initial incoming data file is moved from processing to archive\_raw.
* Finally, process\_cil.py calls the FTP module to FTP the 2 QC reports to FileDistro with TTAAii MCIL01 (agreed with Tech Division for internal use). This module:
  + Appends the TTAAii extension to the filename
  + Opens an FTP connection with either the live or backup FileDistro server
  + FTP’s the file to the placebox directory (where FileDistro looks for new files)
  + Closes the connection and moves the file from ftp-q/sending to ftp-q/sent
  + If any errors are encountered by the FTP module, the file is moved to ftp-q/error, where it can be resent later
* Should an error arise when a file is being processed by process\_cil.py, the file will be moved from the processing directory to the error subdirectory.

Note: for information on FileDistro configuration (ie: what locations are MCIL01 data being transmitted to) contact [itobs@met.ie](mailto:itobs@met.ie)

### 2.3.2 Concat\_cil\_xml.py

* This script simply reads the most recent file in the XML subdirectory for each station and appends these into 1 XML bulletin, cil\_buoy\_latest.xml
* This XML bulletin is forwarded to FileDistro with TTAAii MCIL03. The same FTP process as used in process\_cil.py is used.

**Testing**

The program can be run in a limited test mode by using the “test” argument. For example:

$ python process\_cil.py test

$ pyton concat\_cil\_xml.py test

All normal operations and checks are completed but the summaryReport and fullReport output files are written to the ./data/test\_output/sending directory rather than the operational ./data/ftp-q/sending directory. This ensures that the data is not sent to downstream systems.

However, currently the test function writes data to ./data/archive\_raw and ./data/archive\_qcd files to be updated. If the data used in test is not wanted in these directories: delete the file used as data input from ./data/archive\_raw/ and edit the archive file to remove new records added during test.

## 2.3 Quality Control Checks

Individual station parameter checks are configured using the metadata.csv file. If a buoy is to be QC’d there must be a row in this file with appropriate metadata. For each parameter to be QC’d the following columns are currently required:

* param\_min: minimum climatological value for this parameter at this buoy, ex: hm0\_min is set to 0m for all buoys currently.
* param\_max: maximum climatological value for this parameter at this buoy, ex: hm0\_max is set to 16m for all buoys currently.
* param\_critical: if this parameter fails test 19, then all data from the buoy at this timestamp must be marked as failing. Ex: hm0\_critical is True whilst mdir\_critical is False.
* param\_roc: maximum rate of change for this parameter at this buoy, ex: hm0\_roc is 3m for all buoys.

Note that Metadata.csv must be version controlled, as parameters may be altered over time and it is important that the thresholds imposed on a given record are known for later interpretation.

|  |  |
| --- | --- |
| Test | Notes |
| Missing | Missing data are located by testing for numpy NaN values. This will catch ‘Empty’ slots in csv file. However, sometimes missing data are replaced with an indicative value (9999 etc) –code may need to check for other indicative values in the future. |
| 15  Mean-Stdev | N is set to 4 (data within 4 standard deviations of the mean are considered OK). All data in station archive which has passed QC is used to calculate mean and stdev – up to 24 hours. |
| 16  Flatline | Data is flagged suspect when 3 points in a row are within tolerance of each other and flagged fail when 5 are within tolerance of each other. Tolerance is set to 0.01 by default, however there’s an optional argument to allow it to be adjusted. |
| 19 | Checks if value is outside operator defined limits. The limits are set in the metadata.csv file as outlined above (\_min and \_max). If significant wave height (hm0) fails this test, then mdir and tm02 also fail (even if they passed in their own right). |
| 20 | Checks if the parameter is varying by more than the user defined limit for this parameter at this buoy. This limit is defined in metadata.csv (\_roc). |