# Forecast Massage and Validation Tool

## Currently Supported

|  |  |  |
| --- | --- | --- |
| Scheduler | Script Created | Script Tested |
| ZEKE | Yes | Yes |
| CA7 | No | No |
| CA Scheduler | No | No |
| LSAR | Yes | No |
| TWS | Yes | Yes |

## Requirements

* Perl
* Control-M EM Actively running and client to execute forecastcli.exe
* Source Forecast Files and Days

Script Files

1. ForecastConvert.pl
2. ForecastConvert.cfg

**CONFIGURE BOTH FILES PRIOR TO EXECUTION**

## ForecastConvert.cfg

This file describes the Scheduler, Client and days/source forecast files pairings to be validated against Control-M data. This is used by ForecastConvert.pl to determine which days and to generate and compare forecasts for.

File location is same location as ForecastConvert.pl

### First Line

SCHEDULER|CLIENT FILTER

* SCHEDULER is one of the values from supported table. LSAR is ESP.
* CLIENT FILTER is the client name. In some cases in the source forecasts the client filter indicates a field that needs to be checked and skipped when massaging the forecast file.

#### Example

TWS|SOUTHWEST GAS CORPORATION

### Remaining Lines

ODATE(YYYYMMDD)|<Path to Source Forecast File>

#### Example

20140108|C:\Users\mbobbato\Documents\Work\SWGAS\swg.tws.ltp.rpt.orig.txt

### ForecastConvert.cfg Example

TWS for Southwest Gas Corporation

TWS|SOUTHWEST GAS CORPORATION

20140108|C:\Users\mbobbato\Documents\Work\SWGAS\swg.tws.ltp.rpt.orig.txt

## ForecastConvert.pl

Usage: **perl ForecastConvert.pl**

Assumes ForecastConvert.cfg is in current directory.

### Configuration

Search for **CONTROL-M SETTINGS**

* $user – the EM user that will be passed to forecastcli
* $passwd – the EM users password
* $GUI – the EM GUI Server

Search for **PROJECT SETTINGS**

* $ctm\_name – This is the name of the MF or DS Control-M where client data is stored in Control-M EM. This is critical as we can only filter on forecast data AFTER the forecast is run.
* $ctm\_index is derived and not necessary to set
* $time\_offset – forecastcli automatically uses GMT+00:00. There’s no setting to change this. The time offset is format {+|-}{#}. Examples are -5, +3. After the Forecast is complete all START or time related forecast fields are converted to by the offset specified.

#### @cols and @ext\_cols

@cols – specify the columns from the Forecast that will be used in the comparison. At least 1 field should match from @ext\_cols (usually JOB\_MEM\_NAME).

@ext\_cols – specify column headings used from the source scheduler. They may have different names in source scheduler but we are using these as a comparison. For example it may be APPLICATION in CA7, but this value converts to PARENT\_TABLE in Control-M, so we’d have PARENT\_TABLE in both arrays.

* $ext\_mode – if @ext\_cols contains only JOB\_MEM\_NAME the comparison in 1 to 1 by job name. If more than it will try to compare to PARENT\_TABLE AND JOB\_MEM\_NAME for exact pairings. This value is set automatically.

### #MAIN LOOP

This is the main processing section of the script that does call outs to methods for various tasks

* &CreateAndMassageCTMForecastFile – runs the forecastcli, massages the forecast file according to the $ctm\_name, $time\_offset and @cols fields.
* &MassageExtForecastFile – Reads the source forecast files line by line and calls ExtMassager that will determine how to filter and create each row record. This is most complicated process as each source Forecast is different and unique
* &CreateDataDictionary – this builds internal dictionary’s from the massaged files
* &ForecastValidationExt2Ctm – Check if source scheduler jobs are scheduled in Control-M and create report
* &ForecastValidationCtm2Ext – Check if Control-M jobs are scheduled in source scheduler forecast

ExtMassager is the main method for massaging the External source files. Sometimes the line filters needs to be updated. Here are some examples of compares that are happening. Typically the logic is if a line DOES NOT contain any filters (implying it is not table/job record line and therefore extra junk added by the source forecast) than continue massaging the data.

* =~ and !~ means it is doing a regex comparison meaning contains or not contains.
* $line =~ /${client}/ means if the line contains any occurrence of the value of variable ${client} than it is true
* $line !~ /APPL ID/ means if the line contains string text APPLID it is true
* $line !~ /^\s{2,}/ is using a reg. expression stating if he line starts with at least 2 empty spaces
* Sub string then Trimming strings. Trimming using a regex convention to strip whitespace.
  + my $jobname = substr $line, 0, 8;
  + $jobname =~ s/^\s+|\s+$//g;

## Results

A folder gets created in the scripts working directory based on the Source Scheduler and runtime

3 sub folders get created

* ctm\_forecasts
  + Contains the original forecast file’s as well as the massaged file by ODATE
* <scheduler>\_forecasts
  + Contains original forecast file suffixed .orig, and massaged file
* results – 2 files per forecast
  + ControlMToOrig\_<odate>.csv file created by ForecastValidationCtm2Ext
  + OrigToControlM\_<odate>.csv file created by ForecastValidationExt2Ctm

Results files are in CSV format. Sort by STATUS to identify problematic jobs. Info columns are anything that isn’t job or table columns.