# FC Service detailed functional description

## Parent Service

### Parent service logging

The parent service will produce 3 log files: 2 short-term (kept only during the execution of a single instance), and one long-term.

These log files are:

1. Startup log (short-term)
2. Incoming data files log (short-term)
3. Processed files log (long-term)

### Using startup log: Starting parent service instance

The parent service is responsible for identifying incoming data files and sending them for processing to FC child services.

By the nature of the parent service design, each environment can have only a single instance running at a time. Considering that the parent service will be started by at intervals a cron process we need to prevent cron from starting a second instance of the parent service if the previous instance has not yet completed. To do this we suggest to use a simple startup log file with the following logic:

The startup log file will be used by parent service to record execution events (start and end time) of each instance.

This startup log file will contain 4 types of records: “start”, “exec”, “end” and “delay”. Each record will have a corresponding timestamp recorded and the unique identifier of a service instance, preferably Linux pid.

When parent service starts it will read a startup log file to check if there is a matching “end” record for the last “start” record. If it exists it will clear out the content of the startup log file and record a new “start” record and continue execution. During the execution it will add a “exec” record at each iteration of the internal cycle. This “exec” record will serve as a marker that this instance is actively working and not crushed or stalled.

If it does not have a matching “end” record, it will add a “delay” record at the end of the startup log file and calculate the total execution time for the running instance counting from the last “start” record. If the total execution time is higher than a “max\_parent\_execution\_time” parameter it will send an email notification to the system operators, and terminate.

**Important:** In this scenario the normal execution of the parent service will have to be restarted by a system admin after investigation on the cause of the stalled execution. Alternatively, the parent service can restart automatically, but only it can confirm by looking at the “exec” log records that the previous instance is not active anymore.

This functionality is presented in the following diagram:



### Use of incoming data files log: Incoming data files handling

The main function of the parent service is to analyze incoming files and send them for further processing by the child service. This function must conform to the following:

1. Requirement: Avoid processing files that have not been fully transferred yet.
2. Requirement: Avoid duplicate processing of the same data file in case of the parent service abnormal termination
3. Assumption: Airlines send their data files in batches, there is no continuous flow of data files by an airline.

To satisfy requirement 1) we suggest that the parent service will keep short-time logs of the files received from each airline. This log will be used to set a short delay in data files processing by comparing the current list of incoming files to a corresponding log, and if the current list of files matches the log content (meaning no new files were received and no previously received files changed their size) then the files will be sent for processing, the contents of the incoming log will be added to the long-term processed files log, after that the incoming file log is cleared.

As far as the requirement 2) is concerned, if the execution of a parent service is abnormally terminated at any point, the next instance will just complete the job that was started by the previous instance of the parent service.

This functionality is displayed on the following diagram:

