IICS Control-M Integration Playbook

Summary

[Section 1: Introduction 1](#_Toc905334347)

[Section 2: Configuring RunAJobCli utility 2](#_Toc358524510)

[Section 3: Using RunAJobCli to run a job in IICS: 3](#_Toc56058244)

[Section 4: Testing Cases 4](#_Toc638917457)

[Section 5: Control-M Deployment Steps: 16](#_Toc1754262800)

# Section 1: Introduction

One can have business requirements to run IICS jobs from an external application or check the status of a running task from a Third-Party scheduler. To achieve this requirement the Informatica REST API can directly be invoked. However, one needs to have an understanding of the REST framework, form the web requests for each action, execute it and parse the response. This could be complex and time-consuming. For that Informatica provides a free utility called RunAJobCli which simplifies this operation by performing these individual activities and encapsulating the internal requests and responses. The end user is required to provide only necessary information - for e.g. IICS Account details, Task details, etc.

RunAJobCli is a command line tool used to trigger IICS Tasks. This utility internally uses the REST API service in order to execute IICS tasks.

#### **Summary of Preparatory Tasks**

#### **RunAJobCli package must be enabled in your Informatica Intelligent Cloud Services organization to use the RunAJob utility.**

#### To see if your organization is licensed to use the utility, log in to your organization and clickAdministrator Licenses, and then look for the RunAJobCli package toward the bottom of the page. If you do not see the package, contact Informatica Global Customer Support to enable it.

#### When the package is enabled, the utility can be found in the following location:

#### <Secure Agent installation directory>\apps\runAJobCli​

#### **Note**: Java 1.8 is required to use this utility. To check the java version in the machine, run the following command: ***java -version***

#### If it is not 1.8, please install java 1.8 in the machine/server from where you will be running this command. ​

#### **What happens if there is no java 1.8 installed**

#### Exception in thread "main" java.lang.ClassFormatError: com.informatica.saas.utilities.plugins.RunAJobPlugin (unrecognized class file version)​

#### **The Utility is recommended to be used from "{AgentDir}\apps\runAJobCli\"**. However, since the utility is a standalone java program, one can use the binaries from {AgentDir}\Downloads\package-runAJobCli.**xx**​" and trigger jobs from another server (like Ctrl+m server etc.).

#### **Have on hand the following Information:**

* + IICS Base URL: https://dm1-us.informaticacloud.com/ma
  + IICS Login: This user will the IICS Credential
  + Encrypted password (run the utility {AgentDir}\Downloads\package-runAJobCli.xx\cli.sh encryptText -t <password>

# Section 2: Configuring RunAJobCli utility

#### Following are the steps to run the job using the batch script:

#### Install Java1.8 or just use the Java shipped in Secure Agent

#### Define $JAVA\_HOME and add it to the $PATH including $JAVA\_HOME/bin

1. Include the $INFA\_AGENT\_HOME/apps/runAJobCli in the $PATH system variable

#### Go to {SECURE\_AGENT\_DIR}/downloads/package-runAJobCli.xx/package directory

#### Run the command below in order to encrypt the password of the user that will connect to IICS to run the job: (very important because it is going to generate the keystore.txt file)

#### sh -x cli.sh encryptText -t PASSWORD

#### Make a copy from **restenv\_default.properties** to **restenv.properties**

#### Edit restenv.properties file:

**baseUrl=**https://dm1-us.informaticacloud.com/ma

**username=**USERNAME\_WITH\_PRIVILEGIES\_TO\_RUN\_TASKS (this is the POC user used to connect to IICS and run jobs the final user must have Service Consumer Role)

**password=**:1xmODMaxVEbShcmlRY2Nb8DFhPoWBK8/yE1G45MQ== (this is the encrypted password used in the POC - see step 5)

**ACTIVITYMONITORWAIT=**30000

**TOTALWAIT=**60000

**PROXYHOST=**

**PROXYPORT=**

**RETRYCOUNT=**30

**use.encryption=**true

**Explanation for wait and retry parameters**: **ACTIVITYMONITORWAIT** and **TOTALWAIT** are the key parameters in restenv.properties file to be specified in milliseconds.

#### **ACTIVITYMONITORWAIT** is used to handle the delay if any for the task to appear in Activity Monitor. 30 seconds is good for this.

#### **TOTALWAIT** will set the frequency at which the utility polls for job status. Usually set to 30sec/1min/2min/3min i.e, 30000/60000/120000

#### **RETRYCOUNT** is the number of times it polls for the job status. The recommendation is to know the longest running job and set the retry count accordingly

#### Make a copy from **log4j\_default.properties** file to **log4j.properties** file

#### Make a Copy for the following files from SECURE\_AGENT\_DIR}/downloads/package-runAJobCli.xx/package directory to {SECURE\_AGENT\_DIR}/apps/runAJobCli directory:

#### Restenv.properties

#### Log4j.properties

#### Key-store.txt

#### Go to the RunAJobCli app directory {SECURE\_AGENT\_DIR}/apps/runAJobCli (this is the final directory where Control-M needs access to run the jobs)

Now RunAJobCli is ready to run.

# Section 3: Using RunAJobCli to run a job in IICS:

Run the invoke command below:

sh -x cli.sh runAJobCli -n taskflow\_RunAJobCliInvok -t workflow -fp "POC/DW CDI ControlM" -w True

#### 

#### **Return Codes**

**Error Code is as follows:**

EXCEPTION("Exception", -1)

SUCCESS("Success", 0)

WARNING("Warning", 1)

NOWAIT("NoWait", 2)

FAILURE("Failure", 3)

TIMEOUT("Timeout", 4)

ERROR("Error", 5)

RUNNING("Running", 6)

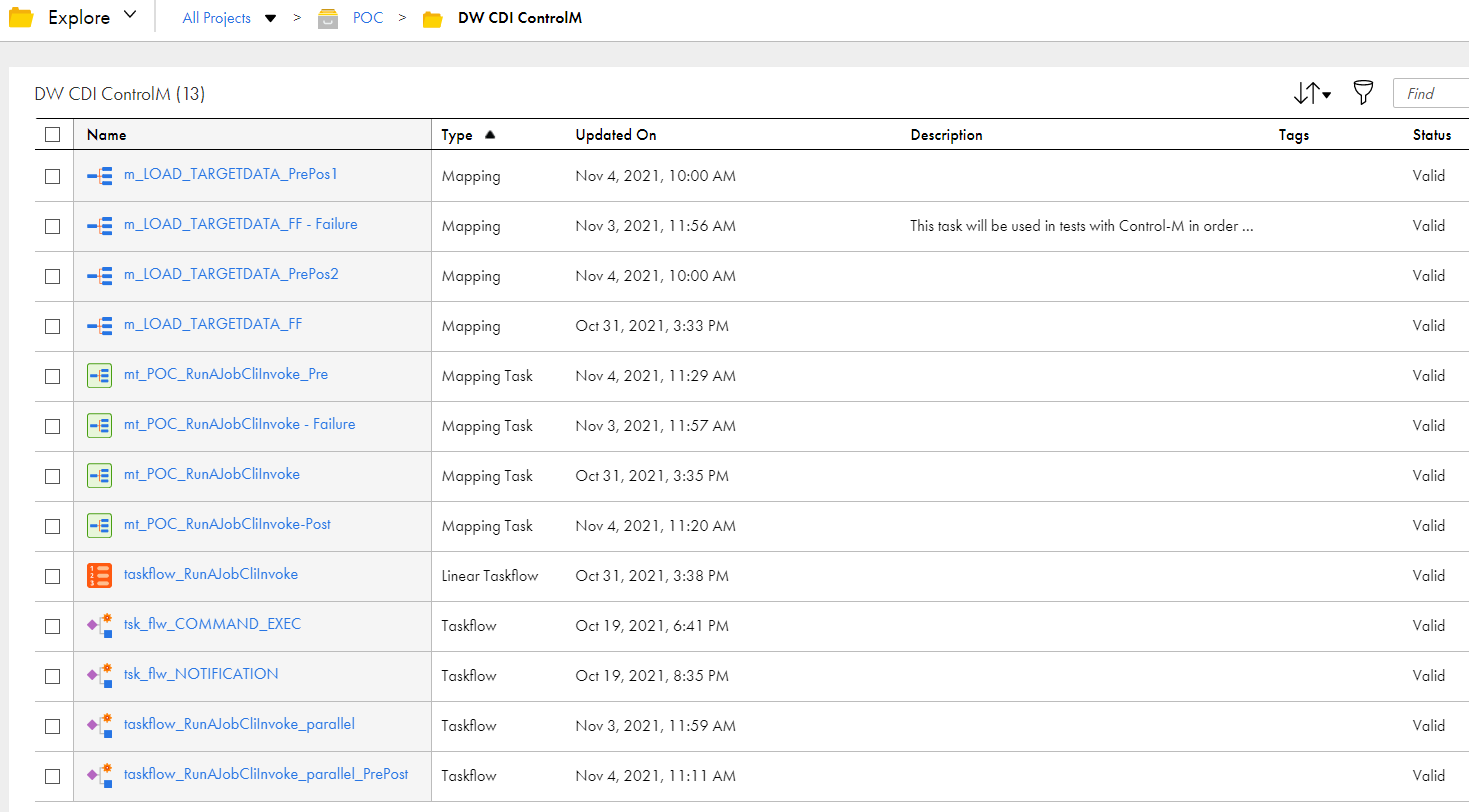
START("Failure", 7)

#### 

# Section 4: Testing Cases

In order to test the integration between IICS jobs and external schedulers, there are 3 test cases in order to ensure Control-M will run successfully and all the IICS jobs can be invoked passing their statuses to Control-M.

All testing assets have been created on the following IICS location:



**Teste Case 1 Description:**

Test a Linear Taskflow. If the job fails it must pass its status to Control-M invoking job. There will be two executions, one for a normal execution without failure and another for a situation where the IICS Job is going to fail.

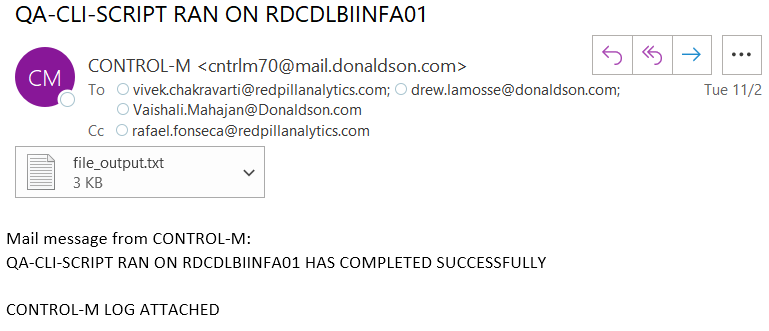
**IICS Asset Name:** taskflow\_RunAJobCliInvoke

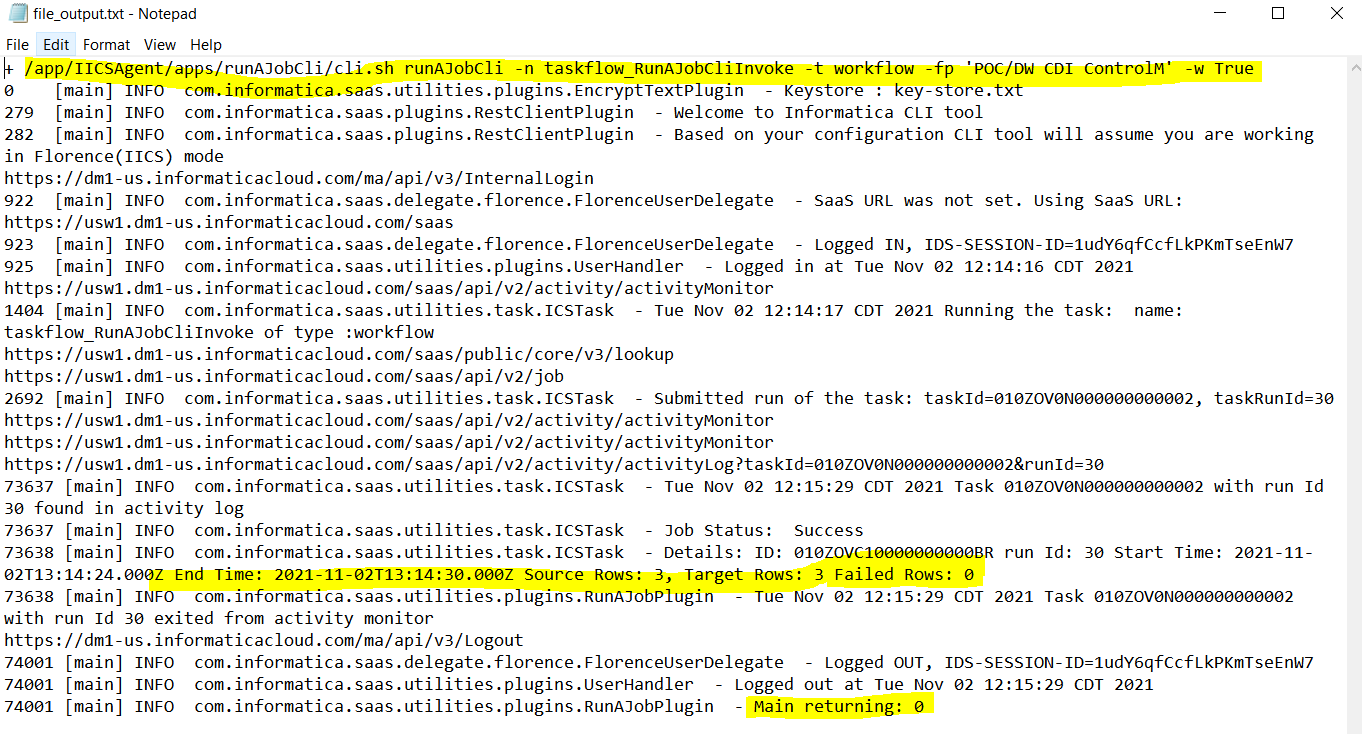
Objective: Read a flatfile source data and write it into another flatfile target.

**Command line for the Control-m job:** /app/IICSAgent/apps/runAJobCli/cli.sh runAJobCli -n taskflow\_RunAJobCliInvoke -t workflow -fp 'POC/DW CDI ControlM' -w True

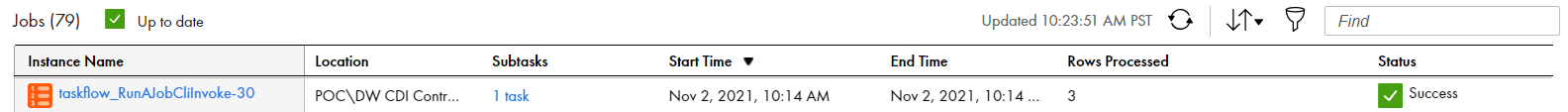
**Execution 1:** The IICS job will run successfully and will pass its status to the Control-M invoking job as a successful execution.

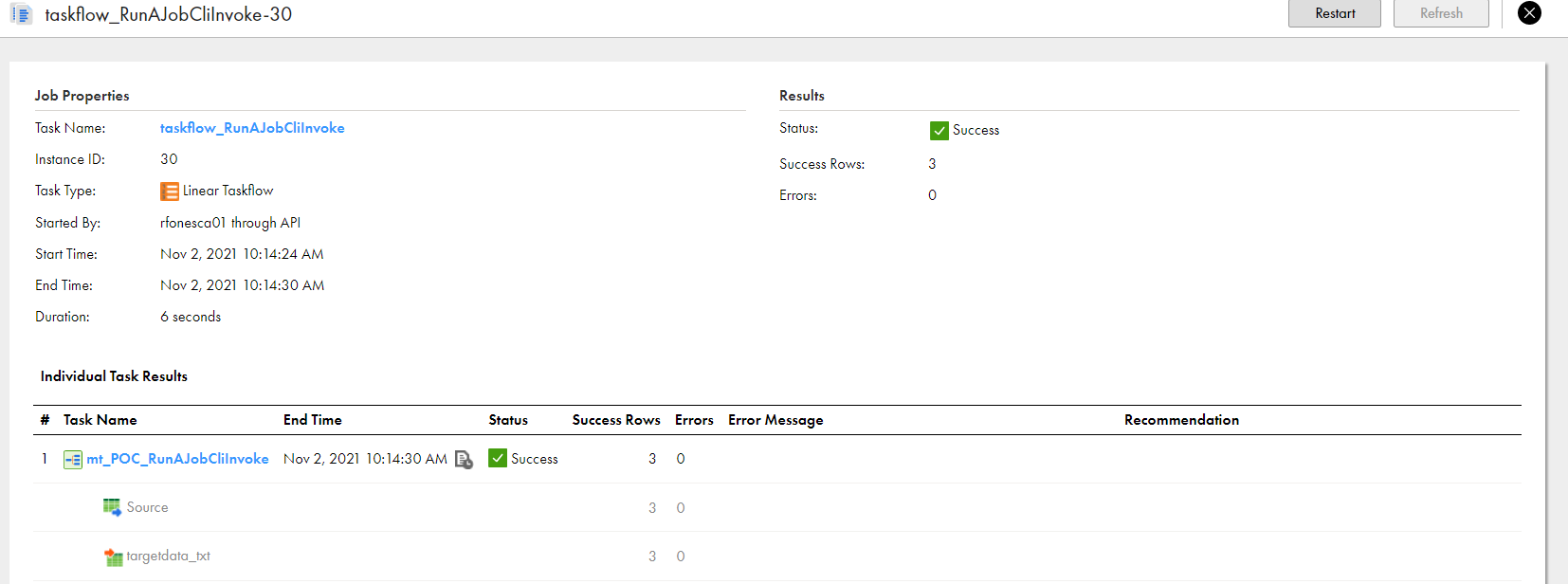
**Email from Control-M:**





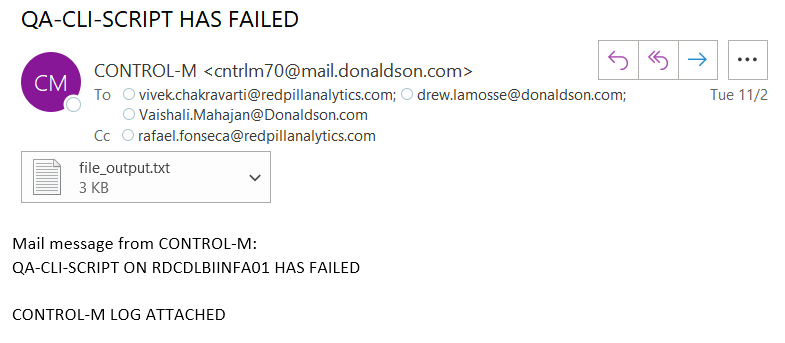
**Status in IICS Monitor:**





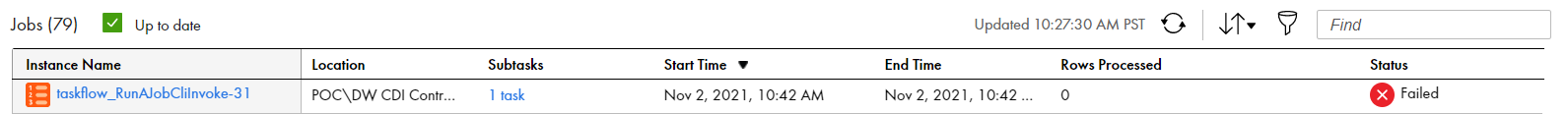
**Execution 2:** The IICS job will fail due missing source data so the Control-M invoking job must fail as well.

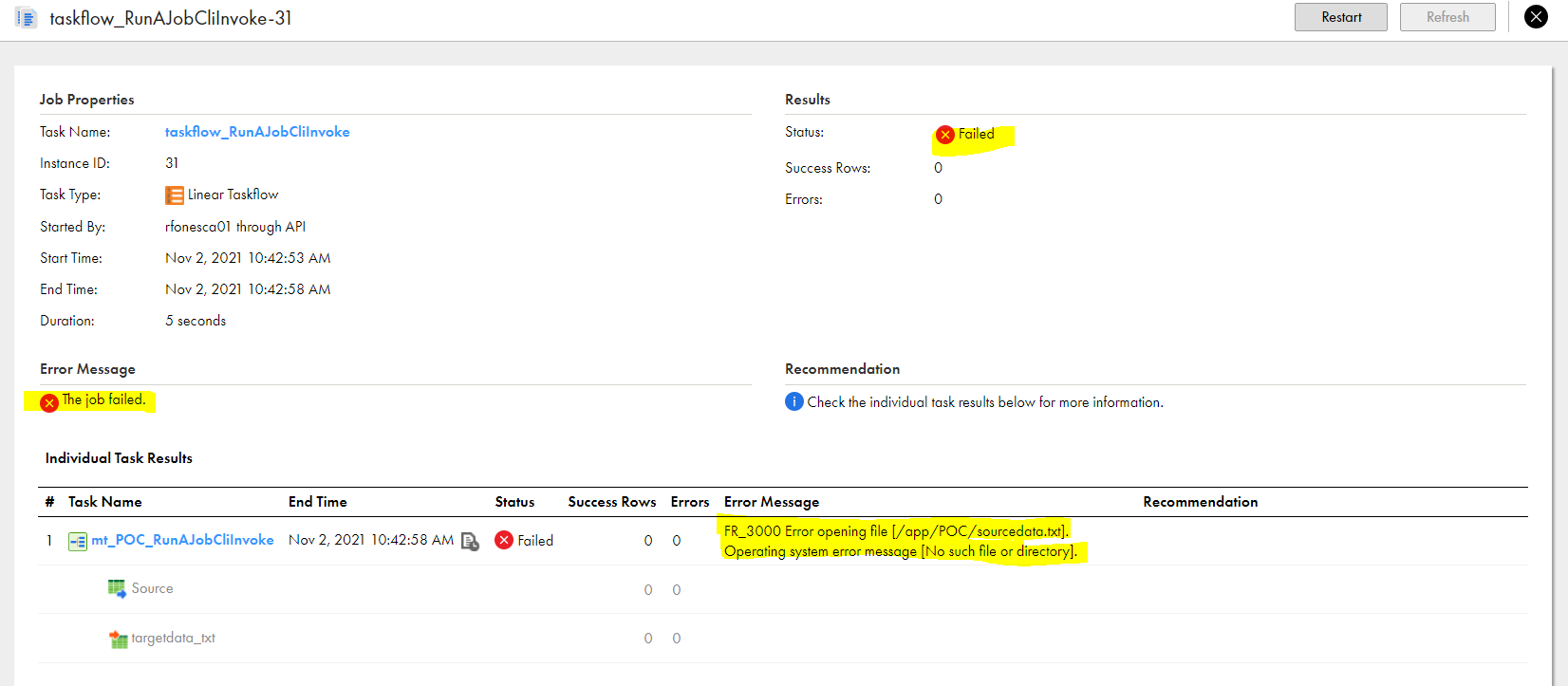
**Email from Control-M:**





**Status in IICS Monitor:**





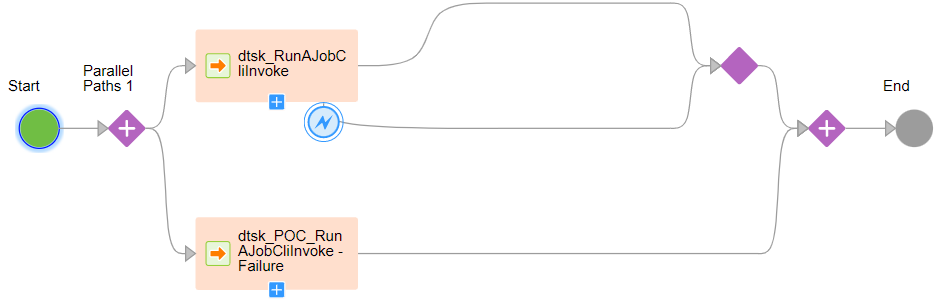
**Test Case 1 Final Status:** All the tests have successfully passed.

**Test Case 2 Description:**

Running a Parallel Taskflow. If the job fails it must pass its status to Control-M invoking job. There will be two executions, one for a normal execution without failure and another for a situation where the IICS Job is going to fail.

**IICS Asset Name:** taskflow\_RunAJobCliInvoke\_parallel

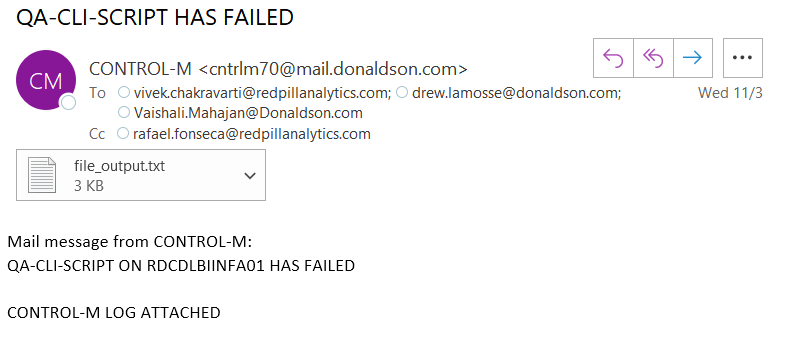
Objective: Run two Data Tasks running in parallel that reads data from a flatfile source data and write it into another flatfile target.

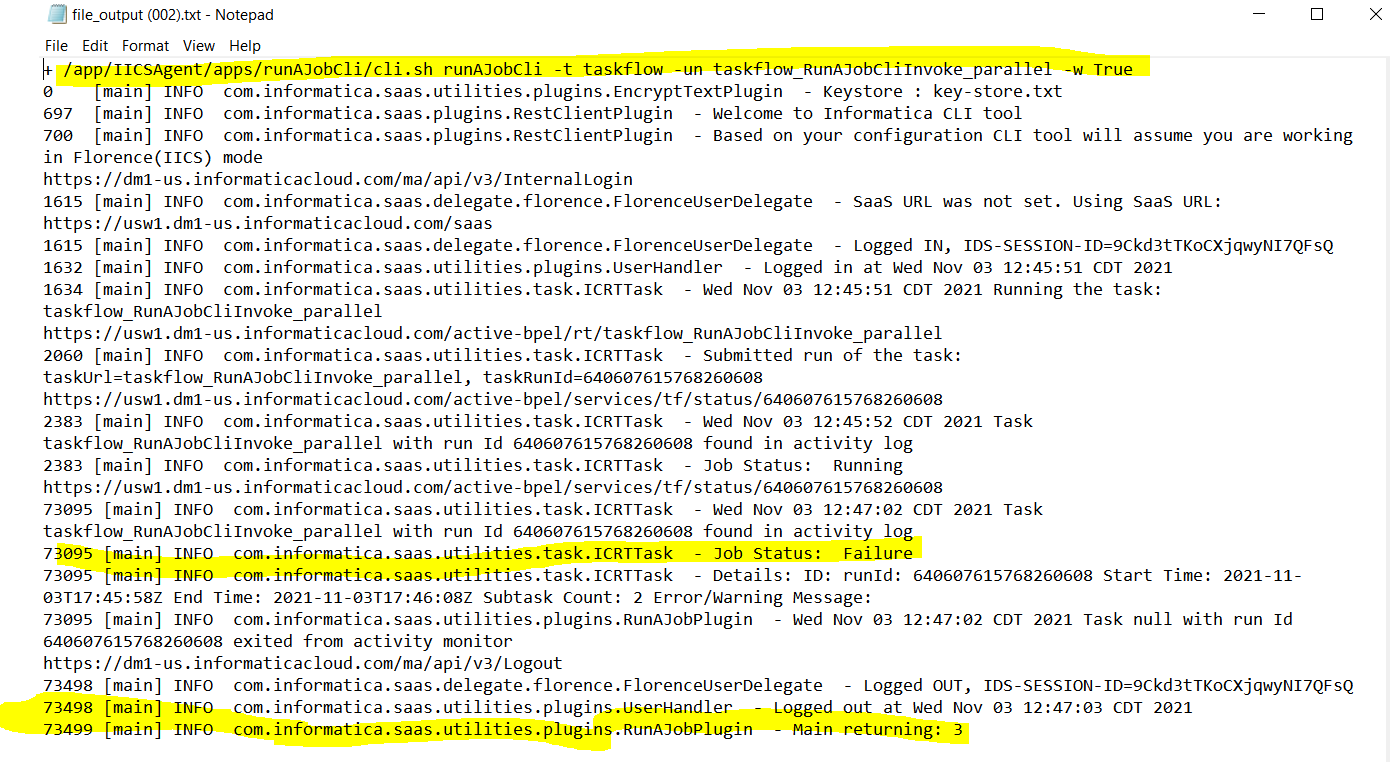


**Command line for the Control-m job:** /app/IICSAgent/apps/runAJobCli/cli.sh runAJobCli -t taskflow -un taskflow\_RunAJobCliInvoke\_parallel -w True

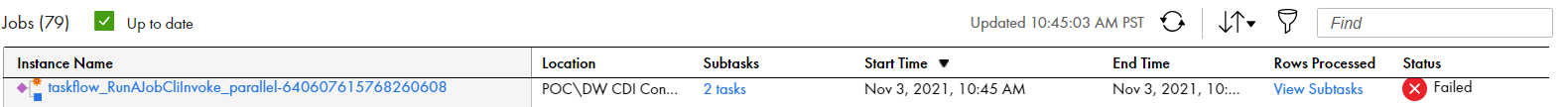
**Execution 1:** The IICS job will fail due missing source data so the Control-M invoking job must fail as well.

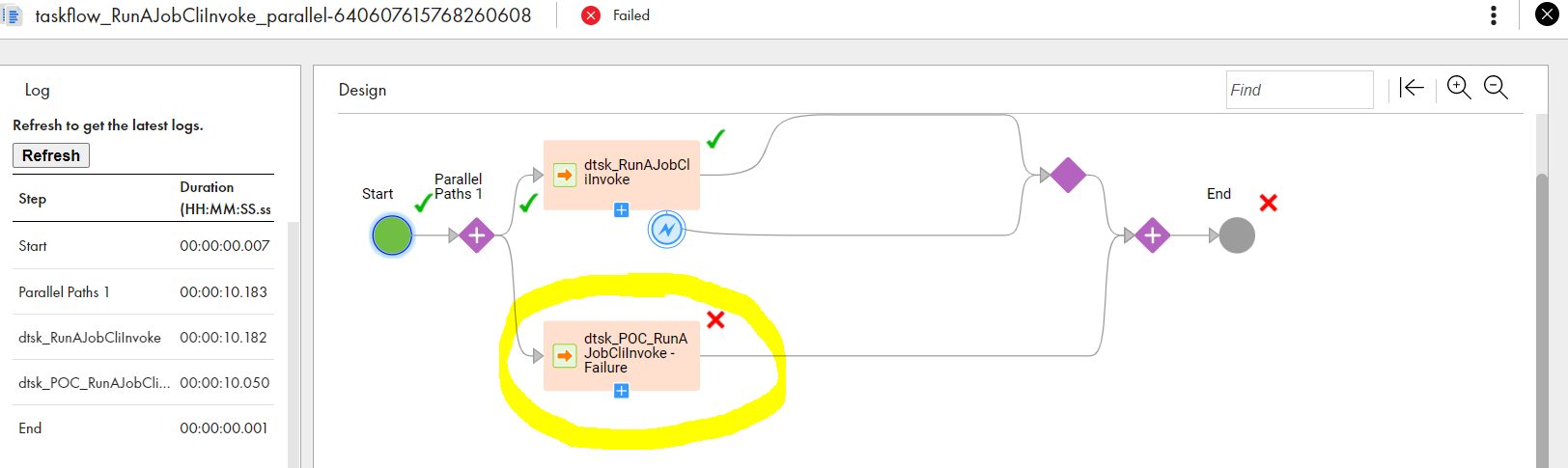
**Email from Control-M:**





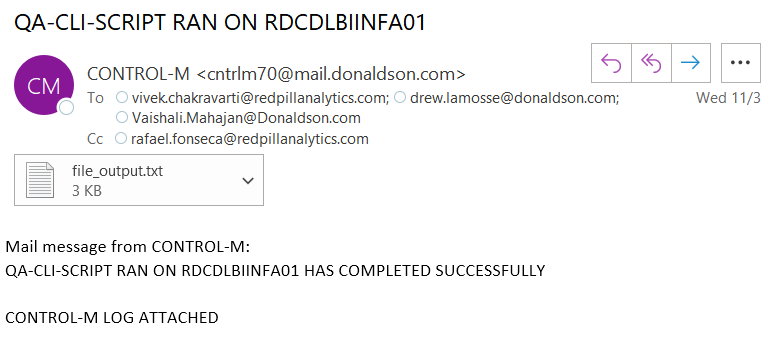
**Status in IICS Monitor:**

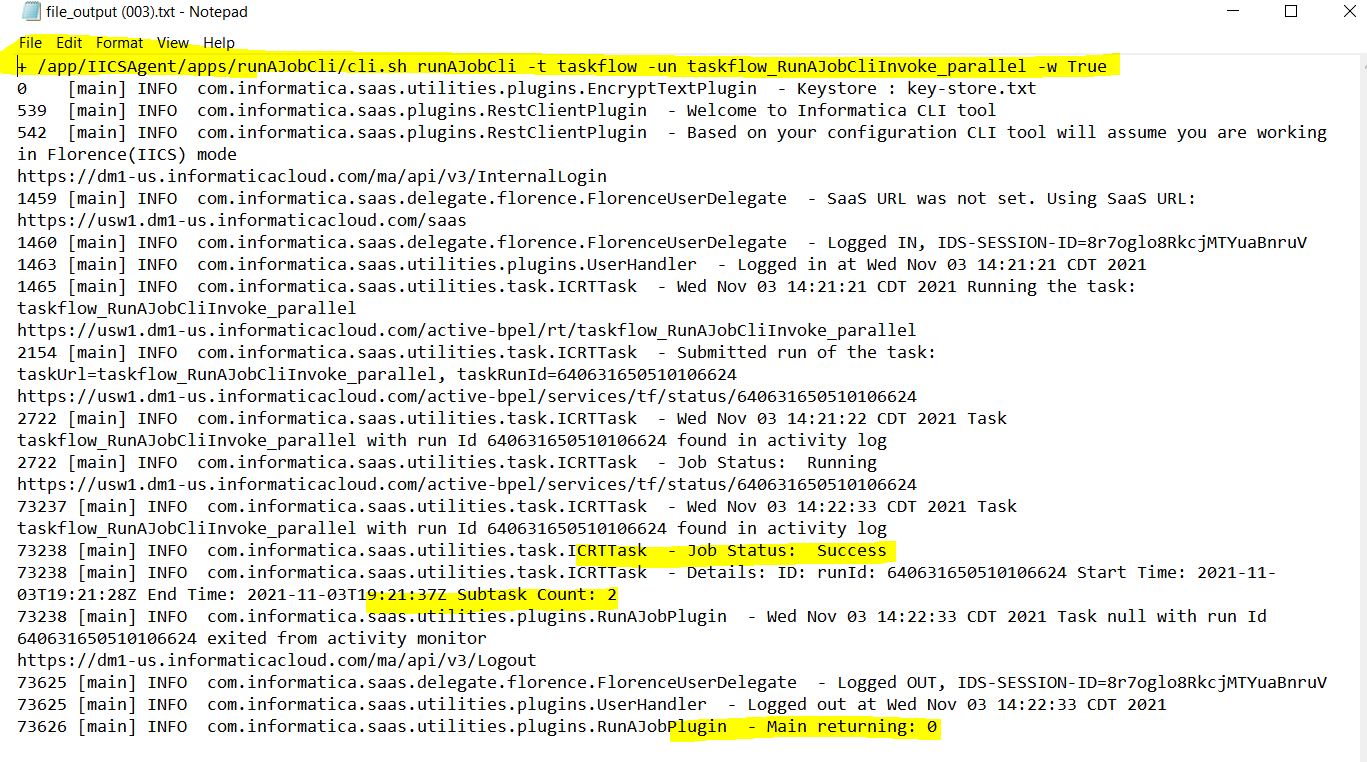




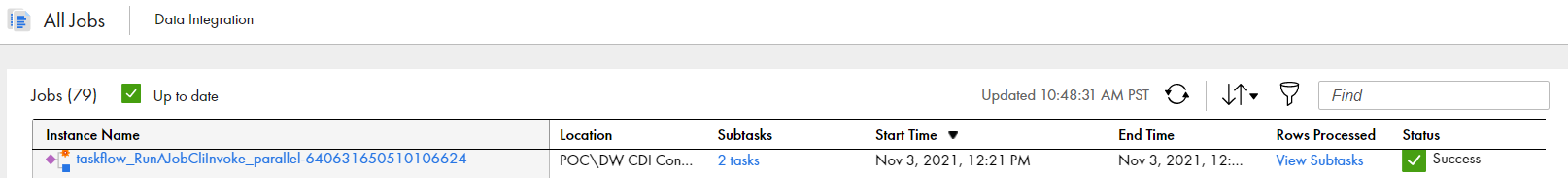
**Execution 2:** The IICS job will run successfully and will pass its status to the Control-M invoking job as a successful execution.

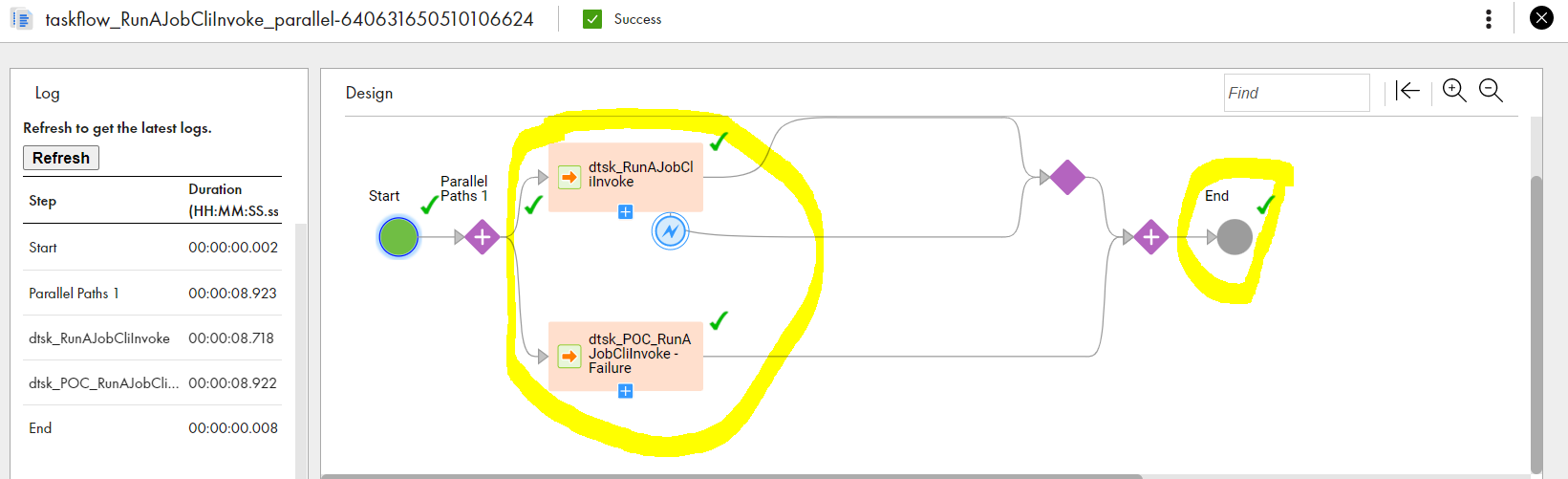
**Email from Control-M:**





**Status in IICS Monitor:**





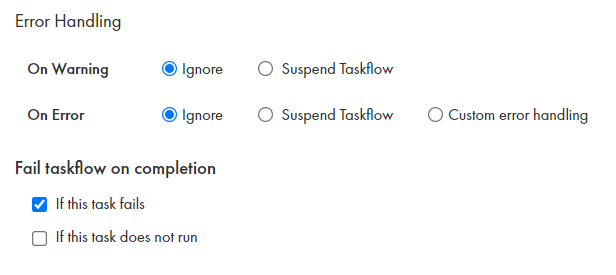
**Teste Case 2 Final Status:** All the tests have successfully passed.

**Teste Case 3 Description:**

Running a Parallel Taskflow using Pre and Post Commands. If the job or any Pre/Post command fail it must pass its status to Control-M invoking job. There will be two executions, one for a normal execution without failure and another for a situation where the IICS Job is going to fail.

**IICS Asset Name:** taskflow\_RunAJobCliInvoke\_parallel\_PrePost

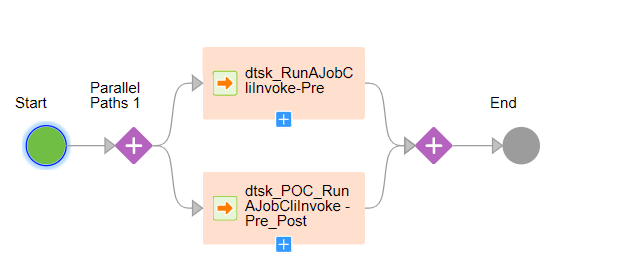
Each Data Task in the taskflow have the Error Handling set as the following:



The Taskflow contains two Mappings/Data Tasks:

1. mt\_POC\_RunAJobCliInvoke\_Pre (has a Pre Command: cp /app/POC/abc.txt /app/POC/abc\_latest\_poc.txt)
2. mt\_POC\_RunAJobCliInvoke-Post (has a Post Command: cp /app/POC/sourcedata.txt /app/POC/sourcedata\_poc\_last\_test.txt)

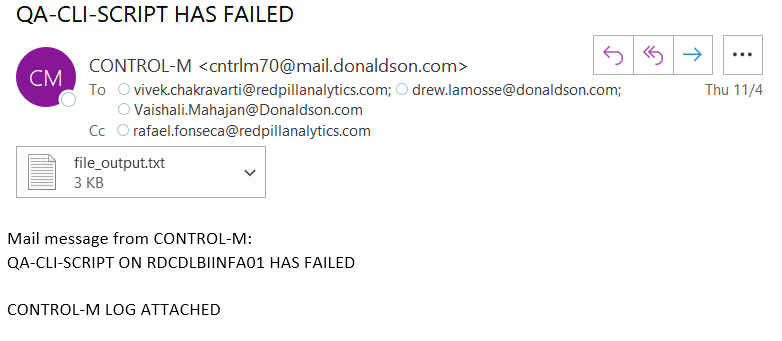
Objective: Run two Data Tasks running in parallel that reads data from a flatfile source data and write it into another flatfile target. Before Loading the source data the mapping task mt\_POC\_RunAJobCliInvoke\_Pre will execute a Pre Command before reading the source data. The mapping task mt\_POC\_RunAJobCliInvoke-Post will execute a Post Command after loading data into the flat file target.

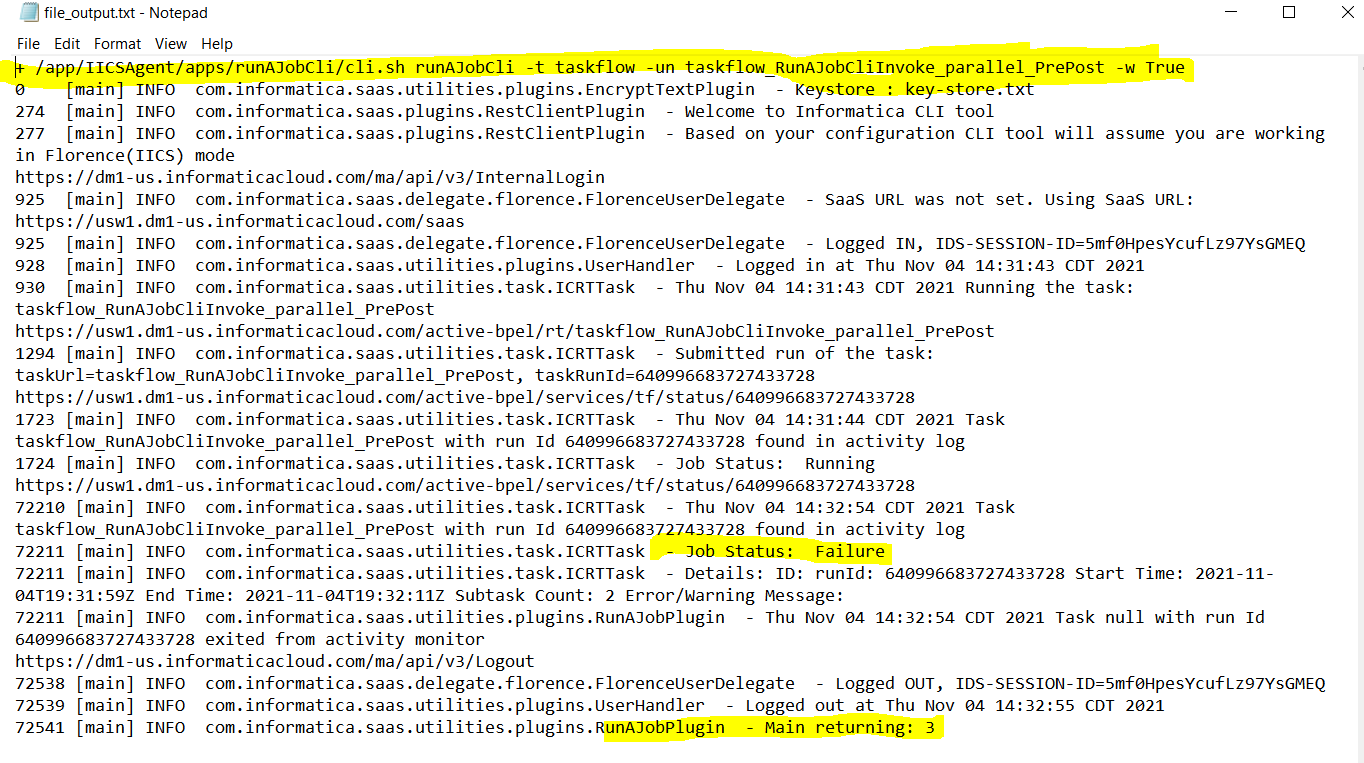


**Command line for the Control-m job:** /app/IICSAgent/apps/runAJobCli/cli.sh runAJobCli -t taskflow -un taskflow\_RunAJobCliInvoke\_parallel\_PrePost -w True

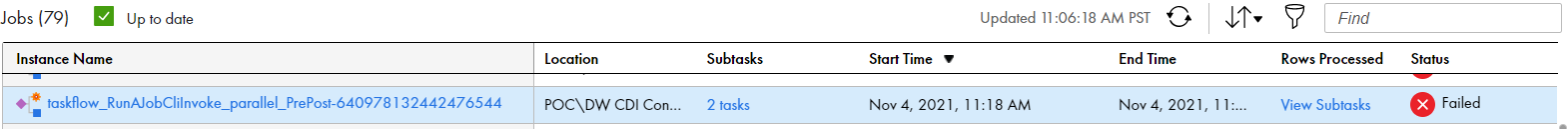
**Execution 1:** The IICS job will fail due the Pre Command failure so the Control-M invoking job must fail as well.

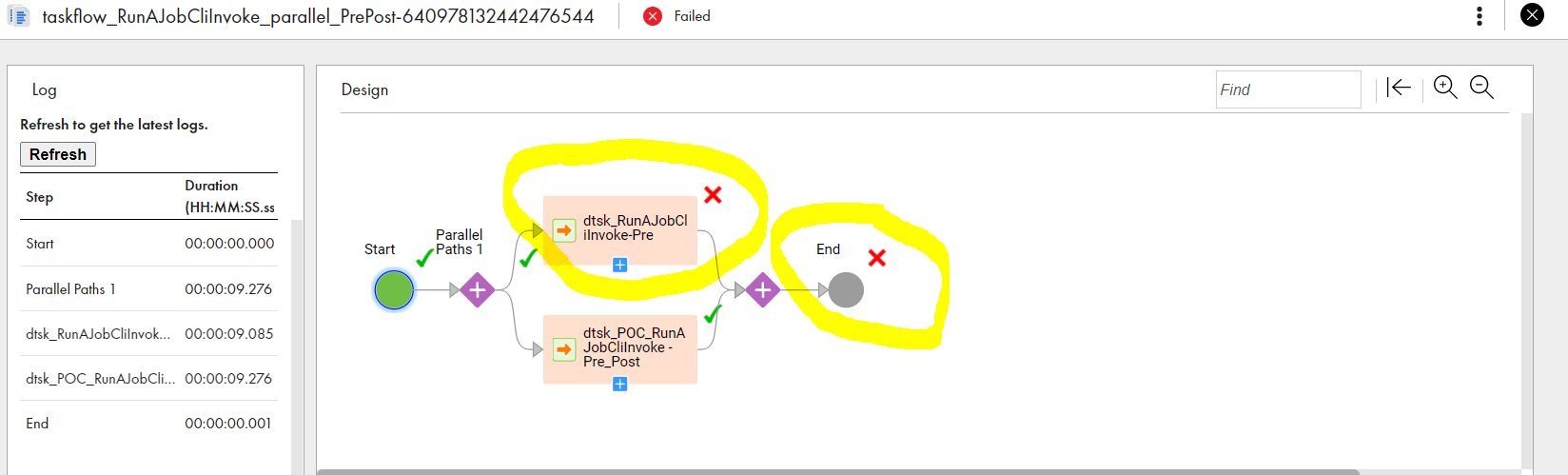
**Email from Control-M:**





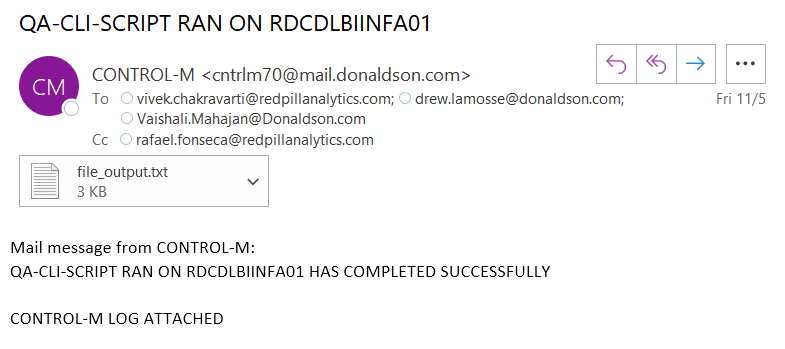
**Status in IICS Monitor:**

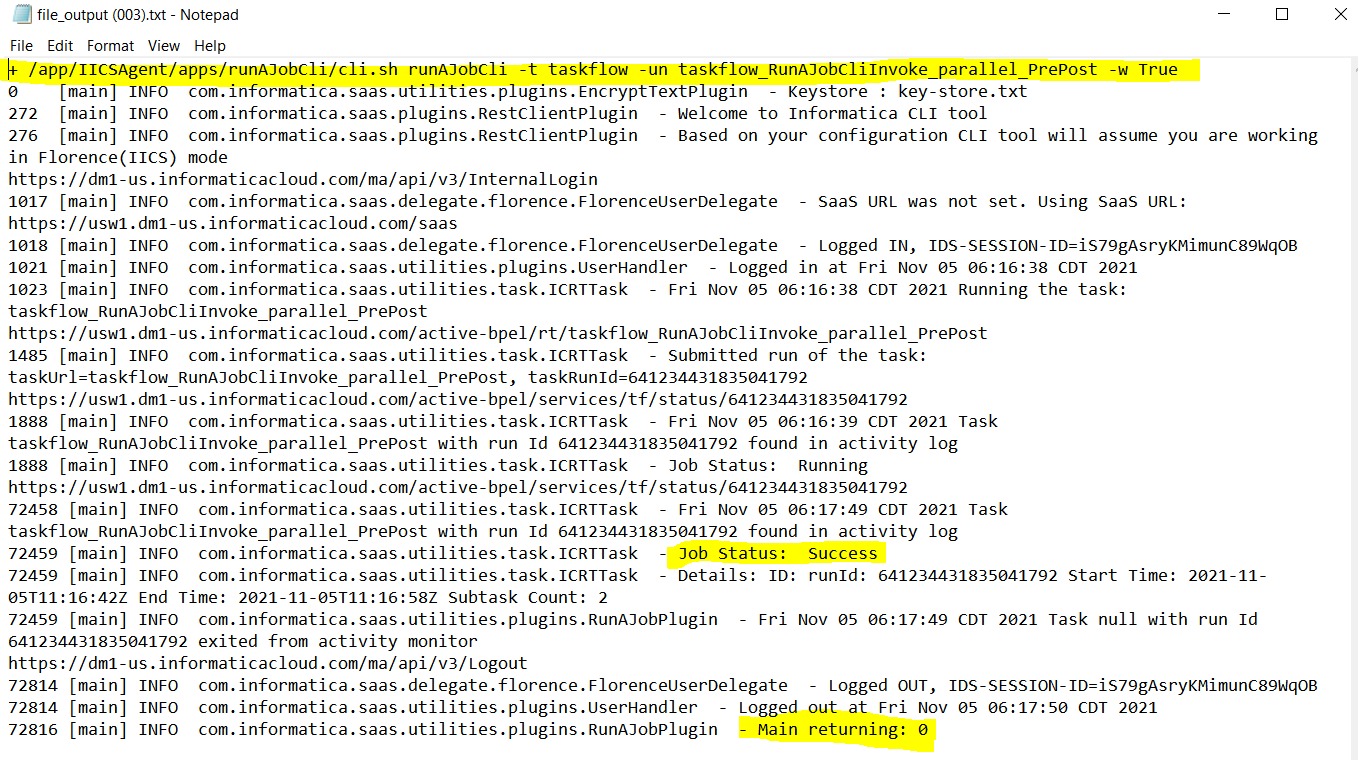




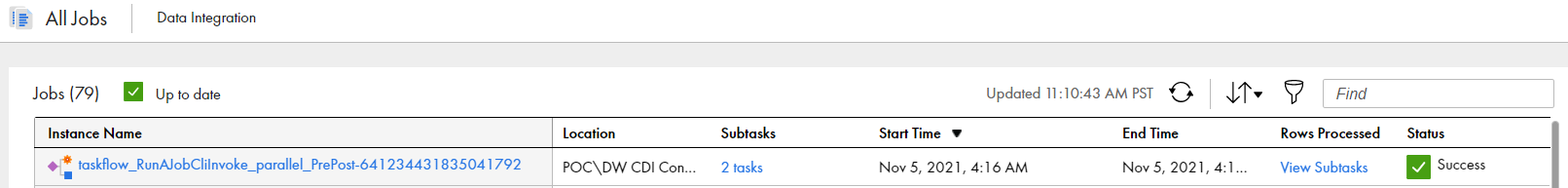
**Execution 2:** The IICS job will run successfully and will pass its status to the Control-M invoking job as a successful execution.

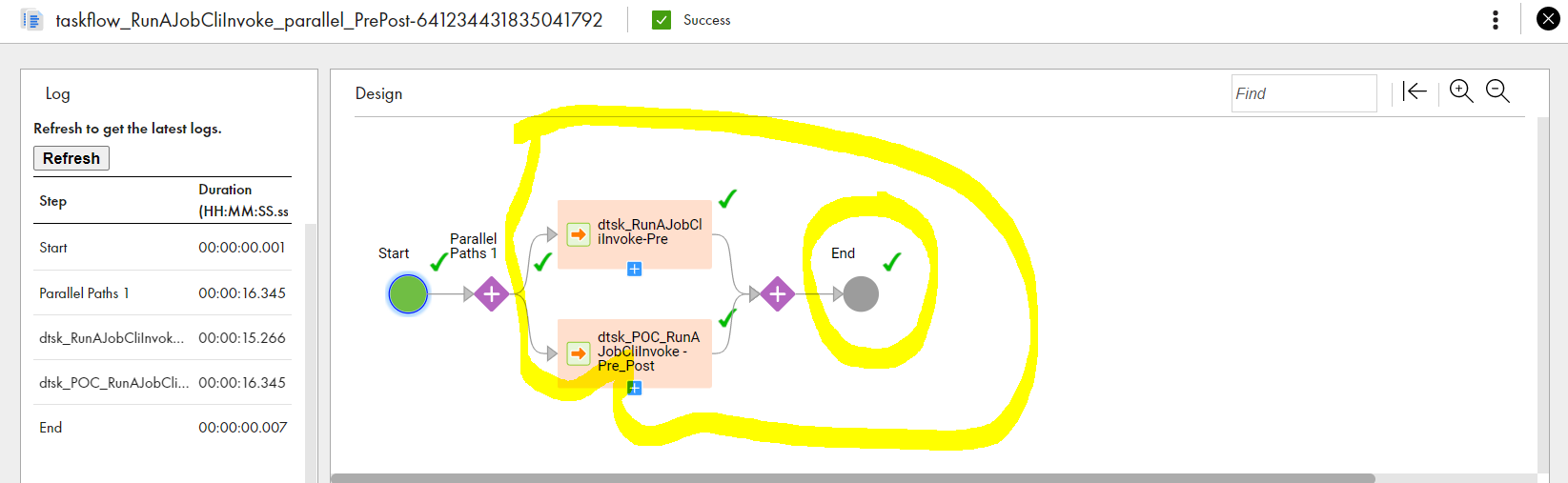
**Email from Control-M:**





**Status in IICS Monitor:**





**Teste Case 3 Final Status:** All the tests have successfully passed.

# Section 5: Control-M Deployment Steps:

**Steps to set up Control-M setup (Control-M team to do):**

1. Control M agent installed on secure agent machine.
2. Set on control-M the command lines for each taskflow (commands that will kick off taskflows on IICS, sent by IICS team). The job (taskflow) will run as INFA user.
3. Setup notifications in Control-M to get start, failure and success emails.

ITGRPOBIEE and Redpill team needs to get these notifications

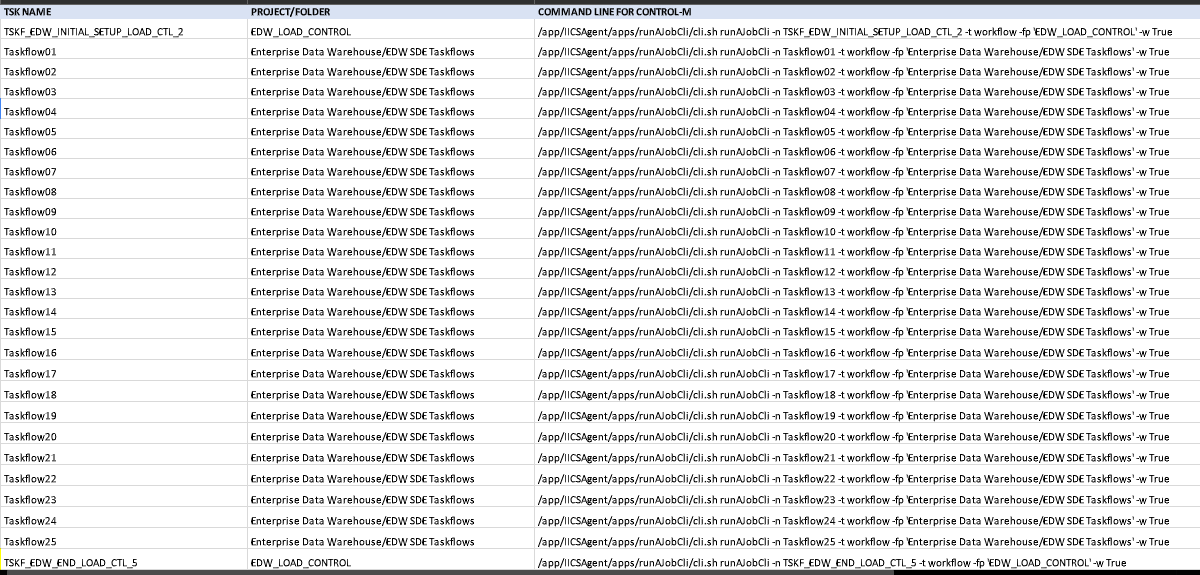
1. Setup access for DCI and Redpill team in Control-M.

**Steps Done in the IICS environment where deployment will be executed:**

**1) Make the command line to run taskflows on control-M (ie, /app/IICSAgent/apps/runAJobCli/cli.sh runAJobCli -un TSKF\_EDW\_INITIAL\_SETUP\_LOAD\_CTL\_2 -t taskflow -fp 'EDW\_LOAD\_CONTROL' -w True). This command line will be used at control-M to start a taskflow on IICS. In this example, the taskflow control-M will be starting on iics is TSKF\_EDW\_INITIAL\_SETUP\_LOAD\_CTL\_2, at folder EDW\_LOAD\_CONTROL.**

**Link for command line document on DEV and QA org:**

<https://donaldson.sharepoint.com/:x:/r/sites/IT-Project-Mgmt-Office-Enterprise-Data-Platform/_layouts/15/Doc.aspx?sourcedoc=%7B2549AFE6-09DA-4F4A-9D7A-F367BDE8F1C1%7D&file=IICS%20-%20Control-M%20Taskflow%20commands.xlsx&action=default&mobileredirect=true>



Line example: **Taskflow01 Enterprise Data Warehouse/EDW SDE Taskflows /app/IICSAgent/apps/runAJobCli/cli.sh runAJobCli -n Taskflow01 -t workflow -fp 'Enterprise Data Warehouse/EDW SDE Taskflows' -w True**

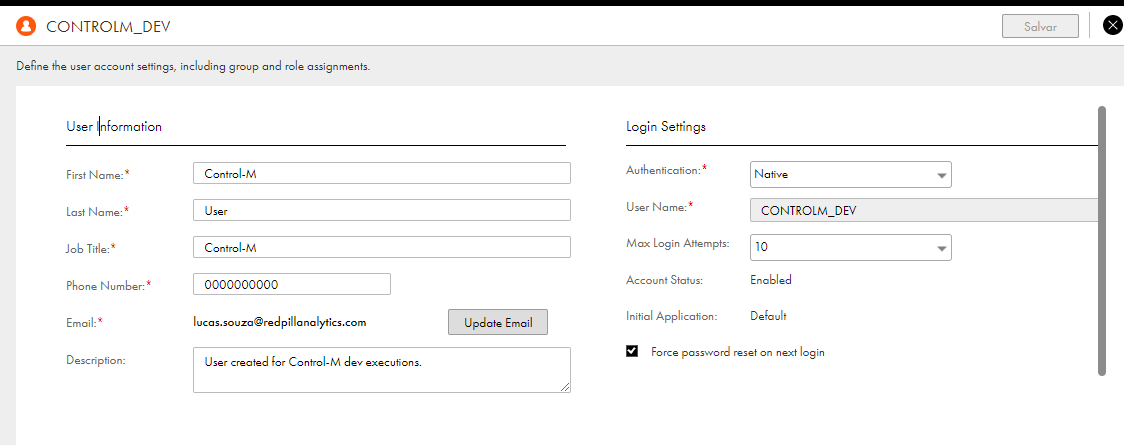
**2) Create an IICS user in the ORG that Control-M will Run (ex: CONTROLM\_QA) having the following roles:**

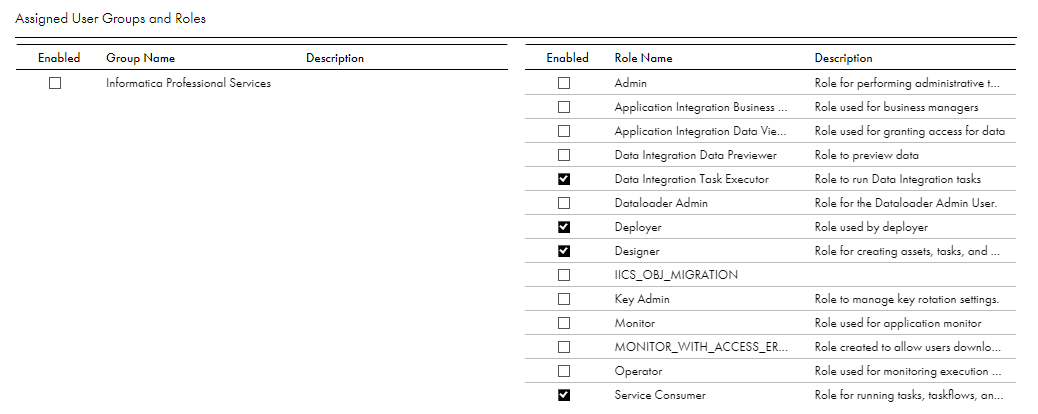
a) Service Consumer Role

b) Data Integration Task Executor

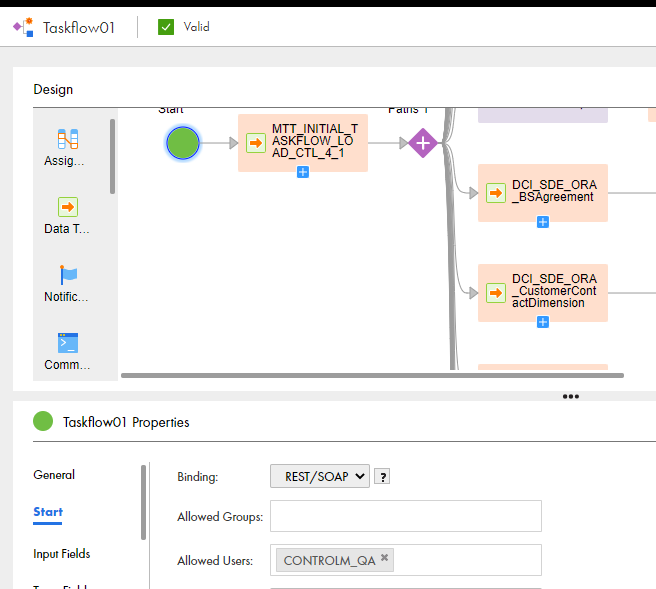
c) Deployer (optional)

d) Designer (optional)





**3) For each Taskflow that needs to run (Load Control and EDW Taskflows) it is necessary to allow the IICS Control-M User (ie, CONTROLM\_QA). Open taskflow, go to start step, click on ‘start’, allow the controlM user on ‘Allowed Users’ (must insert the exatly username that was created for controlM user).**



**4) Update the CONTROLM IICS user and encrypted password in the RunAJobCli utility in the iics agent environment at linux, so Control-M will start the taskflows using RunAJobCli:**

a) Encrypt the IICS password that user Control-M will use to run the taskflows (ie, CONTROLM\_QA)

Command to encrypt:

sh -x /app/IICSAgent/apps/runAJobCli/cli.sh encryptText -t INCLUDE\_THE\_IICS\_PASSWORD\_HERE

b) If the file /app/IICSAgent/apps/runAJobCli/restenv.properties does not exist, then make a copý from the file /app/IICSAgent/apps/runAJobCli/restenv\_default.properties

Command to copy the file to a new one:

(I.e, cp /app/IICSAgent/apps/runAJobCli/ restenv\_default.properties /app/IICSAgent/apps/runAJobCli/ restenv.properties)

c) Edit the /app/IICSAgent/apps/runAJobCli/restenv.properties (using filezilla) to update the USER and PASSWORD (encrypted password done in step A)

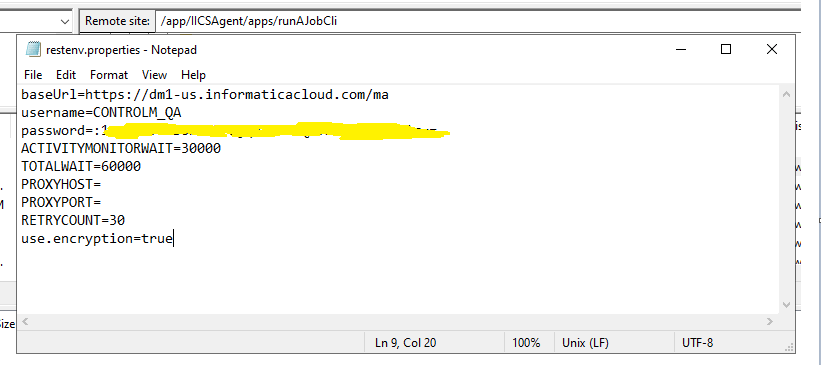
Include the IICS User (ie, CONTROLM\_QA)

Below is an example:

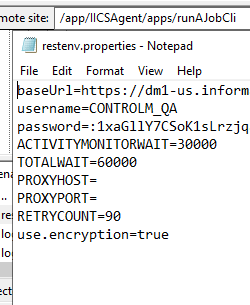
username=CONTROLM\_QA

password=\*\*\*\*\*\*\*\*\*

Once the password gets encrypted, the option use.encryption must turn to ‘true’, as image below:



D)\*\*\*\*\*IMPORTANT\*\*\*\*\* Once you open restenv.properties file, change the RETRYCOUNT from 30 to 90 and save, this parameter controls how many minutes the job will be monitoring the taskflow, and if its 30, after 30 minutes if the taskflow did not finished yet, it will fail. So a good tip is to change to 90 (if a taskflow on iics takes longer than 90 minutes to finish, then it must be changed to a larger number).



**5) Test the RunAJobCli execution from the Linux Server to make sure it is working:**

Command for running an **Advanced** Taskflow on Putty:

/app/IICSAgent/apps/runAJobCli/cli.sh runAJobCli -un TSKF\_EDW\_INITIAL\_SETUP\_LOAD\_CTL\_2 -t taskflow -fp 'EDW\_LOAD\_CONTROL' -w True

Command for running a **Linear** Taskflow on Putty:

app/IICSAgent/apps/runAJobCli/cli.sh runAJobCli -n TSKF\_EDW\_END\_LOAD\_CTL\_5 -t workflow -fp 'EDW\_LOAD\_CONTROL' -w True