Testing JPSS ATMS and OMPS SDRs in Cloud in ADL Block 2.1 MX8

Bigyani Das[1], Aiwu Li[1], Yunhui Zhao[2], Weizhong Chen[2], Tom King[3] and Walter Wolf[3]
[1]IMSG, Rockville, MD 20852, USA
[2]GAMA-1 Technologies LLC, Greenbelt, MD 20770, USA
[3]NOAA/NESDIS/STAR, College Park, MD 20740, USA

Introduction

STAR has initiated a project to a test Cloud-based environment in AWS to support science algorithm creation, R2O, and management. Several algorithms are included to be evaluated in this project. Among them ATMS and VIIRS algorithms are included. These algorithms are currently tested by STAR ASSISTT in the Algorithm Development Library (ADL) framework in the Linux system and then packaged with the inputs, outputs and documents to DPES for regression testing. DPES sends to Raytheon for final integration. The goal is to make these deliveries be made through the Cloud in future.

We have transitioned, run, and integrated ATMS algorithm successfully in Cloud. Even though OMPS algorithms were not included in the first phase of this pilot project, we have tested them to gain experience and understand if there are any problems.

STAR JPSS ASSISTT

ASSISTT: The STAR Algorithm Scientific Software Integration and System Transition Team.

STAR JPSS ASSISTT provides expertise on integration of JPSS algorithms into operational systems. Three major tasks of ASSISTT are:

- Code Testing in Algorithm Development Library (ADL) for IDPS and Code Testing for implementation in NDE, Troubleshooting, and Integration.
- Communication with Science Teams and Data Product and Engineering Services (DPES), OSPO, and Other Developers.
- Change Request Submission/DAP submission

Method

- Get account in the Cloud
- Evaluate documents to understand how the Cloud system works.
- Request help of the system staff to install Cloud workspace.
- Perform simple testing of small programs.
- Import the COTS required by ADL to the Cloud.
- Install and Build ADL in the Cloud.
- Create a test case for ATMS
- Select a Golden Day.
- Collect all the needed inputs.
- Collect RDRs from storage system.
- After testing compare with the results that is obtained in the on premise Linux system for the same input data.
- Other group members test and verify the results.

ATMS RDR to SDR Processing

ATMS RDR to SDR processing in ADL has the following requirements:

Each ATMS RDR to SDR granule processing requires 2 additional granules (1 before and 1 after) for the ATMS granule to be processed. Three consecutive granules are loaded to the ADL system for processing of the middle granule during each SDR processing.

Dynamic data required for each ATMS RDR to SDR processing:
One current weekly Polar Wander (PW) data file.

Required static ancillary data include:
1. a set of ATMS SDR LUTs and PCT for ADL MX8
2. a set of global Terrain-Eco Tiles files.

ATMS RDR to SDR Processing: Dynamic Data

ATMS RDR H5 data files
If any granule is missing from the 3 consecutive RDR granules, the SDR processing for the middle RDR granule will not be invoked, thus will be skipped. Work around of this is to use 5 granules instead of 3.

When duplicated granule RDR files (same granule ID but with different creating time) are found in the RDR H5 data directory, the latest one will be selected during the chain processing; the older ones are skipped.
(Data source: SCDR)

Dynamic ancillary data
USNO Polar Wander data
The USNO Polar Wander data files are weekly products. Use the most recent weekly PW file available in the PW data directory. Currently, the number of days to look back is set to 7 (one week); it can also go back to 2-3 weeks

Near Real-Time (NRT) Processing Data Flow

When a new ATMS RDR file is available in the RDR data directory, it will be used as a job trigger to invoke the single granule RDR to SDR chain processing for that granule. The single granule processing driver script will call the utility file searching scripts to find 3 consecutive RDR granule files and the latest PW file.

If all files are found, copy all 4 H5 files into an intermediate input directory created for the granule followed by calling the Docker container wrapper script to launch a container to run the SDR algorithm. All required input data is made available to the container run during the SDR processing through the mounted directories to the container. After the ATMS SDR algorithm run in ADL completes successfully, the output SDR file will be moved out of the container to an output directory. Processing log for each RDR to SDR container run is saved in a log directory.

Note: Recently our testing failed because of Spacecraft RDR issues and hence we changed it to 5 consecutive granules from 3 consecutive granules. Adding TLE Aux file did not help.

Git Repository and CI Pipeline

1. The ADL system source code has been put in the Git repository on the Cloud. The Continuous Integration (CI) pipeline has been established to enable an automated build and automated tests for algorithm code changes in ADL.

2. A Docker image containing the ADL system and the required environment has been created and made available on AWS ECR. The image is used to set up the ADL working environment inside a runtime container for ATMS SDR test runs on AWS EC2 machines.

Testing OMPS SDR in Cloud

OMPS includes 3 instruments.
OMPS Nadir Mapper (OMPS NM)
OMPS Limb Profiler (OMPS LP)
OMPS Nadir Profiler (OMPS NP)

Currently ASSISTT supports OMPS NM and NP SDR and EDR integration.

OMPS SDRS were tested in the Cloud. They produced identical results as obtained in the Linux system

Steps for Testing ATMS Algorithm

Organize the working directory properly to help the integration and testing process.

Step1
Put RDR files into the database
Go to RDR directory. Use the command:
$ADL_HOME/tools/bin/ADL_Unpacker.exe *.h5

Step2
Put LUTs in the database
Go to LUT directory
$ADL_HOME/script/DMS/ImportIntoDms.plx -path $LUT_DIR
(Use full path name)

Step3
Put Polar files in the database

Go to POLAR directory

a) Unpack the HDF5 file
$ADL_HOME/tools/bin/ADL_Unpacker.exe *.h5

.msd file will be created

b) Copy that .msd file to /ADL/data/MSD_LZ/ANC directory
cp *.msd ${ADL_MSD_LZ_PATH}/ANC

c) Put that in DMS
$ADL_HOME/tools/bin/IngMsd.exe -lw
$ADL_HOME/cfg/dynamic/IngMsdLwFile.xml

Step4
Put Tile files in the database

Go to TILE directory and use the command
$ADL_HOME/script/DMS/ImportIntoDms.plx -path $TILE_DIR

Note: Use full pathname of $TILE_DIR

Step5
Put Eph file in the database

Go to EPH directory
$ADL_HOME/script/DMS/ImportIntoDms.plx -path $EPH_DIR

Note: Use full pathname of $EPH_DIR

Step6
Run the test cases

Step 7
Collect the output HDF5 files which would be generated at
mv $ADL_HOME/data/hdfOutputs/ .*

ADL Framework

- ADL is the Test System - Developed by Raytheon
- ADL mimics the Operational Interface Data Processing Segment (IDPS)
- ADL provides a Diagnostic Framework

Current Version mimics IDPS Operation: ADL BLK 2.1– MX 8.0

Acronyms

- IDPS – Interface Data Processing Segment
- ATMS - Advanced Technology Microwave Sounder
- VIIRS – Visible Infrared Imaging Radiometer Suite
- DPES – Data Products Engineering Services
- DAP – Delivered Algorithm Package
- ADL – Algorithm Development Library
- COTS – Commercial Off the Shelf
- RDR – Raw Data Record
- SDR - Sensor Data Record
- IDPS - Interface Data Processing Segment
- NDE - NPP Data Exploration
- AWS – Amazon Web Services
- ECR - Elastic Container Registry
- EC2 - Elastic Compute Cloud

Corresponding Author: bigyani.das@noaa.gov