NOAA NESDIS
CENTER for SATELLITE APPLICATIONS and RESEARCH

The NOAA JPSS RR Product System
System Maintenance Manual

Version 1.0

TITLE: JPSS RR SYSTEM MAINTENANCE MANUAL VERSION 1.0

AUTHORS:

John Lindeman (IMSG)

APPROVAL SIGNATURES:

Walter Wolf (STAR) June 5, 2015
VIIRS JPSS RR Project Lead

DOCUMENT HISTORY
DOCUMENT REVISION LOG

The Document Revision Log identifies the series of revisions to this document since the baseline release. Please refer to the above page for version number information.

DOCUMENT TITLE: JPSS RR System Maintenance Manual			
DOCUMENT CHANGE HISTORY			
Revision No.	**Date**	**Revision Originator Project Group**	**CCR Approval # and Date**
1.0	03/28/2014	Created by John Lindeman (With PAL's input)	

LIST OF CHANGES

Significant alterations made to this document are annotated in the List of Changes table.

DOCUMENT TITLE: JPSS RR System Maintenance Manual					
LIST OF CHANGE-AFFECTED PAGES/SECTIONS/APPENDICES					
Version Number	**Date**	**Changed By**	**Page**	**Section**	**Description of Change(s)**

Table of Contents

LIST OF FIGURES

Page

LIST OF TABLES

LIST OF ACRONYMS

ABI	Advanced Baseline Imager
AIT	Algorithm Integration Team
AIX	Advanced Interactive eXecutive
ARR	Algorithm Readiness Review
ATBD	Algorithm Theoretical Basis Document

AWG	Algorithm Working Group
CDR	Critical Design Review
CLASS	Comprehensive Large Array-data Stewardship System
CPU	Central Processing Unit
DAP	Delivered Algorithm Package
DDS	Data Distribution Server
DHS	Data Handling System
GB	Gigabyte
GFS	Global Forecast System
GL	Granule List file
GRIB	Gridded Binary format
IDPS	Interface Data Processing Segment
IET	IDPS Epoch Time
INS	Ingest Subsystem
IP	Intermediate Product
JPSS RR	NOAA Joint Polar Satellite System (JPSS) Risk Reduction (RR)
JRR	NOAA Joint Polar Satellite System (JPSS) Risk Reduction (RR)
JRRPS	JRR Product System
NDE	NPOESS Data Exploitation
NCDC	National Climate Data Center
NCEP	National Center for Environmental Prediction
NESDIS	National Environmental Satellite, Data, and Information Service
NetCDF	Network Common Data Format version
NetCDF4	Network Common Data Format version 4
NOAA	National Oceanic and Atmospheric Administration
NPOESS	National Polar-Orbiting Operational Environmental Satellite System
NSOF	NOAA Satellite Operations Facility

NUP	NOAA Unique Product
NWP	Numerical Weather Prediction
OISST	Optimum Interpolation Sea Surface Temperature
OSDPD	Office of Satellite Data Processing and Distribution
PCF	Process Control File
PDA	Product Distribution and Access
PDS	Product Distribution Subsystem
PGM	Product Generation Manager
PGS	Product Generation Subsystem
PS	Polar Stereographic map projection
PSF	Process Status File
RAD	Requirements Allocation Document
RHEL	Redhat Enterprise Linux
SADIE	Science Algorithm Development and Integration Environment
SAN	Storage Area Network
SDR	Sensor Data Record
SOA	Services-Oriented Architecture
SPSRB	Satellite Products and Services Review Board
STAR	Center for Satellite Applications and Research
SWA	Software Architecture Document
VIIRS	Visible Infrared Imager Radiometer Suite
XML	eXtensible Markup Language

EXECUTIVE SUMMARY

The NOAA Joint Polar Satellite System (JPSS) Risk Reduction (RR) Product System generates cloud, aerosol, and cryosphere products for polar orbiting satellites. The system was designed to run within the NPOESS Data Exploitation (NDE) production environment.

JPSS RR is identified as a mission critical and therefore a 24 X 7 service maintenance level is required. Return to service within 2 hours is required.

The requirements for the JPSS RR products are described in the JPSS RR Requirements Allocation Document (RAD) (NESDIS/STAR, 2014) which is available in the JPSS RR project artifact repository.

The JPSS RR product team consists of the following individuals (Table 0-1-1):

Table 0-1-1: Product Team Members

Team Member	Organization	Role	Contact Information
Walter Wolf	STAR	STAR Project Lead	5830 University Research Court Riverdale, MD. 20740 Phone: 301-683-3594 Email: Walter.Wolf@noaa.gov
Shuang Qiu	OSPO	Backup Lead	
Andy Heidinger	STAR	NESDIS Clouds Team Lead	Email: Andrew.heidinger@noaa.gov
Jeff Key	STAR	NESDIS Cryosphere	Email: jkey@ssec.wisc.edu

		Team Lead	
Shobha Kondragunta	STAR	NESDIS Team	
Istvan Laszlo	STAR	NESDIS Team	
Mike Pavolonis	STAR	NESDIS Team	
A.K. Sharma	OSPO	NESDIS Team	
Hanjun Ding	OSPO	NESDIS Team	
Zhaohui Cheng	OSPO	NESDIS Team	
Chris Sisko	OSPO	NESDIS Team	
Donna McNamara	OSPO	NESDIS Team	NSOF 1372 4231 Suitland Rd Suitland MD 20746 Phone: (301)817-3803 Email: Donna.McNamara@noaa.gov
Mitch Goldberg	NOAA JPSS	NESDIS Team	
Eric Gottshall	NOAA JPSS	NESDIS Team	
Sean Helfrich	NIC	NESDIS Team Cryosphere PAL	Email: sean.helfrich@noaa.gov

David Donahue		Clouds PAL	Email: David.R.Donahue@noaa.gov
Liquin Ma		Aerosol PAL	Email: liquin.ma@noaa.gov
Pablo Clemente	NIC	NESDIS Team	
Shanna Sampson	IMSG	NESDIS Team	5825 University Research Court, Suite 1500, Riverdale, MD. 20740 Phone: 240-582-3585 X 23045 Email: Shanna.Sampson@noaa.gov
Peter Romanov	CREST	NESDIS Team	
Veena Jose	IMSG	NESDIS Team	5825 University Research Court, Suite 1500, Riverdale, MD. 20740 Phone: Email: veena.jose@noaa.gov
William Straka III	CIMSS	NESDIS Team	Email: william.straka@ssec.wisc.edu
Ray Garcia	CIMSS	NESDIS Team	
Andi Walther	CIMSS	NESDIS Team	
Pat Heck	CIMSS	NESDIS Team	
Kevin Schrab	NWS	User Team	
Mike Johnson	NWS	User Team	

John Derber	NWS/NCEP/EMC	User Team	
Jeff Ator	NWS/NCEP/NCO	User Team	
Sid Boukabara	JCSDA	User Team	
Carven Scott	NWS	User Team	

The JPSS RR Product System is a NOAA Unique Product (NUP) that is integrated using Delivered Algorithm Package (DAP) in to the S-NPP Data Exploitation (NDE) ground system. The NDE Data Handling System (DHS) consists of Ingest (INS), Product Generation (PGS), Product Distribution (PDS), and Monitoring subsystems; and is physically spread across numerous servers, both IBM and Linux based, and a Storage Area Network (SAN). The IBM Power 7 Series blade servers run the Advanced Interactive eXecutive (AIX) operating system while the Dell blade servers run Redhat Enterprise Linux (RHEL). All servers, i.e. nodes, are 64-bit. The Quantum StorNext SAN file system has a capacity of 60 TB and is used to manage all data storage. An Ethernet network is used as the front-end for all clients to access files on the SAN.

There are six AIX nodes dedicated to product generation, each with 8 quad core CPUs resulting in 32 effective CPUs and 63 GB of memory for each node. The Power 7 series CPUs have a clock frequency of 3 GHz. The Linux nodes have 24 Intel Xeon (x86_64 architecture) CPUs running at 2.8 GHz clock frequency with 37 GB of memory each. The Linux nodes are spread across ingest (2), product generation (2 for the factories and 2 for processing), and product distribution (4 for pull and 4 for push) for a total of 14 nodes. Each processing node (both AIX and Linux) has a RAID subsystem providing local storage of 1.5 TB.

Most NOAA Unique Product (NUP) algorithms currently run on the AIX processing nodes. But JPSS RR algorithm currently runs on the Linux processing node.

The following diagram shows the data flow through the NDE DHS and the physical layout of the hardware.

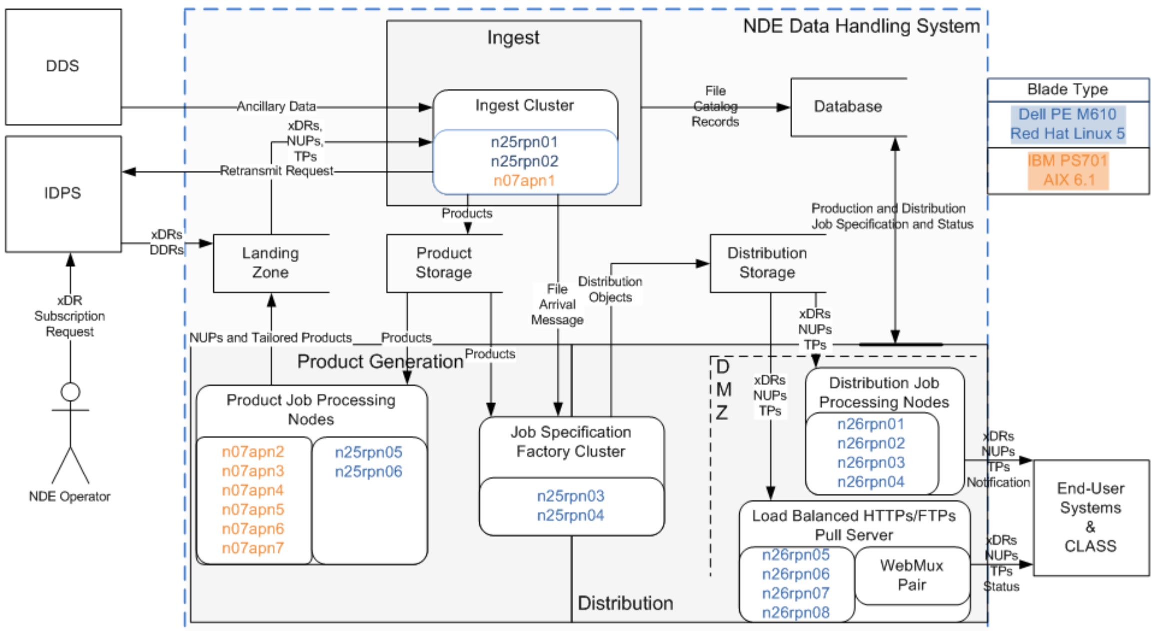


Figure 0-1-1: NDE Data Flow Diagram and Physical Layout

The following diagram shows the network layout for the NDE DHS Production Environment-1 (PE-1).

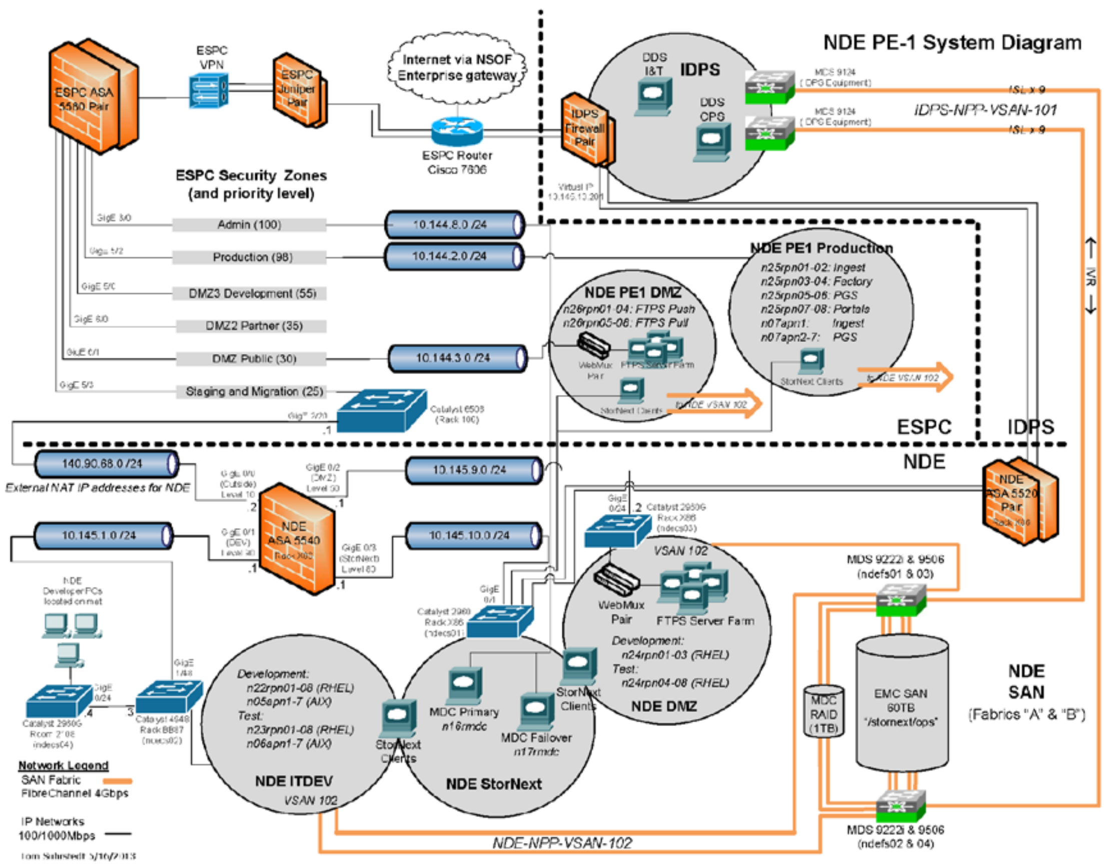


Figure 0-1-2: NDE PE-1 System Diagram

All low-level processing code in JPSS RR is written in compiled languages. These are Fortran 77/90/95, C++, and C. Low-level code performs all data processing, scientific computation, reading/writing, reformatting, and opening/closing of files. All high-level code is written in Perl and Python. High-level tasks include file management, system management, making system calls, and error trapping from low-level processing. The Perl scripts act as driver code for the lower-level processing. The driver scripts manage the JPSS RR software and calls their unit scripts. The system is organized into two sub-systems and each consists of a single driver script. One is Preprocessor, which takes an orbit of VIIRS SDR and IP as input. Another is the Product-Generator, which takes the NetCDF format files as input to generate the JPSS RR products.

The main VIIRS Level 1 input data is received from IDPS and the corresponding ancillary input data like GFS model data is from ESPC DDS. The JPSS RR products are currently distributed to users via NDE distribution server and the products are also archived in CLASS for non real time and research users.

The NOAA JPSS RR products will be used as a risk reduction assessment for a cost effective implementation of common NESDIS algorithms for the JPSS system. The system was designed to run within the NPOESS Data Exploitation (NDE) production environment. The output products are intended for operational and scientific users. Table 0-1-2 provides information on the algorithms and products.

Table 0-1-2: JPSS RR Algorithms and Products

Product Category	Algorithm	Product
Cloud	- Cloud Mask	Cloud Mask
	- Cloud Phase	Cloud Top Phase Cloud Type
	- Cloud Height	Cloud Top Height Cloud Cover Layers Cloud Top Temperature Cloud Top Pressure
	- Daytime Cloud Optical and Microphysical Properties (DCOMP)	Cloud Optical Depth Cloud Liquid Water Path Cloud Ice Water Path

	- Nighttime Cloud Optical and Microphysical Properties (NCOMP)	Cloud Optical Depth Cloud Liquid Water Path Cloud Ice Water Path
Aerosol	- Aerosol Detection - Aerosol Optical Depth - Volcanic Ash	Aerosol Detection Aerosol Optical Depth Aerosol Particle Size Volcanic Ash Mass Loading Volcanic Ash Height
Cryosphere	- Snow Cover - Ice Concentration - Ice Thickness and Age	Snow Cover Snow Fraction Ice Concentration and Cover Ice Surface Temperature Ice Thickness/Age

The details about the algorithm are described in the Theoretical Basis Document (ATBD) for RR products (NESDIS/STAR, 2014).

The NESDIS' Policy on Access and Distribution of Environmental Data and Products is provided at: http://www.ospo.noaa.gov/Organization/About/access.html.

Users need to fill out the Data Access Request Form located on this site and submit to the PAL with a copy to nesdis.data.access@noaa.gov. This address provides the OSPO Data Access Team a copy of the correspondence. The process is defined in the following diagram. Once the request is approved by the OSPO management the data will be delivered by the Data Distribution System

(DDSProd) currently distributing the ESPC data products and later by the Product Distribution and Access (PDA) system. The ESPC Data Distribution Manager, Donna McNamara (donna.mcnamara@noaa.gov) should be contacted for any data accessibility and data distribution problems.

1. INTRODUCTION

1.1. Product Overview

The NOAA JPSS RR System produces a total of 24 products in three different product areas: Clouds, Aerosol, and Cryosphere. The products generated from the Suomi NPP (National Polar Orbiting Partnership) Visible Infrared Imaging Radiometer Suite (VIIRS) Scientific Data Records (SDR) will be used as risk reduction assessment for a cost effective implementation of common NESDIS algorithms for the JPSS program. The output products are intended for operational and scientific users.

1.2. Algorithm Overview

The algorithms in the JPSS products are modified or upgraded versions of GOES-R algorithms adapted to run on S-NPP VIIRS (except for Snow Cover which is GOES heritage). The JPSS RR algorithms run inside a system of supporting software. This is the AIT-Framework system or the GOES-R Algorithm Working Group (AWG) Product Processing System Framework. The Framework has been developed to be a plug-and-play system for GOES-R scientific algorithms enabling the development and testing of the Level 2 GOES-R products within a single system. The system has been created to run products and store them in memory to be used as inputs for other products: i.e. product precedence. Common ancillary data has been used by the algorithms and the ancillary data is also stored in memory and treated as precedence for the products. Within the Framework system, the JPSS RR algorithms have flexible interface design that allows different types of instruments/satellite data sets. Therefore the JPSS RR algorithms are the same for GOES and Polar products. It was originally developed for the future GOES-R ABI instrument.

To generate JPSS RR products, the first step is to obtain level 1b VIIRS data. The JPSS RR products are generated for all areas covered by the polar orbiting satellites. AIT-framework requires the input data to be in NetCDF format.

The details about the algorithm are described in the Theoretical Basis Document (ATBD) for JPSS RR products (NESDIS/STAR, 2014).

1.3. Interfaces Overview

Figure 1-1 shows the JPSS RR interfaces to external systems. See also Figure 2-1 for NDE interfaces.

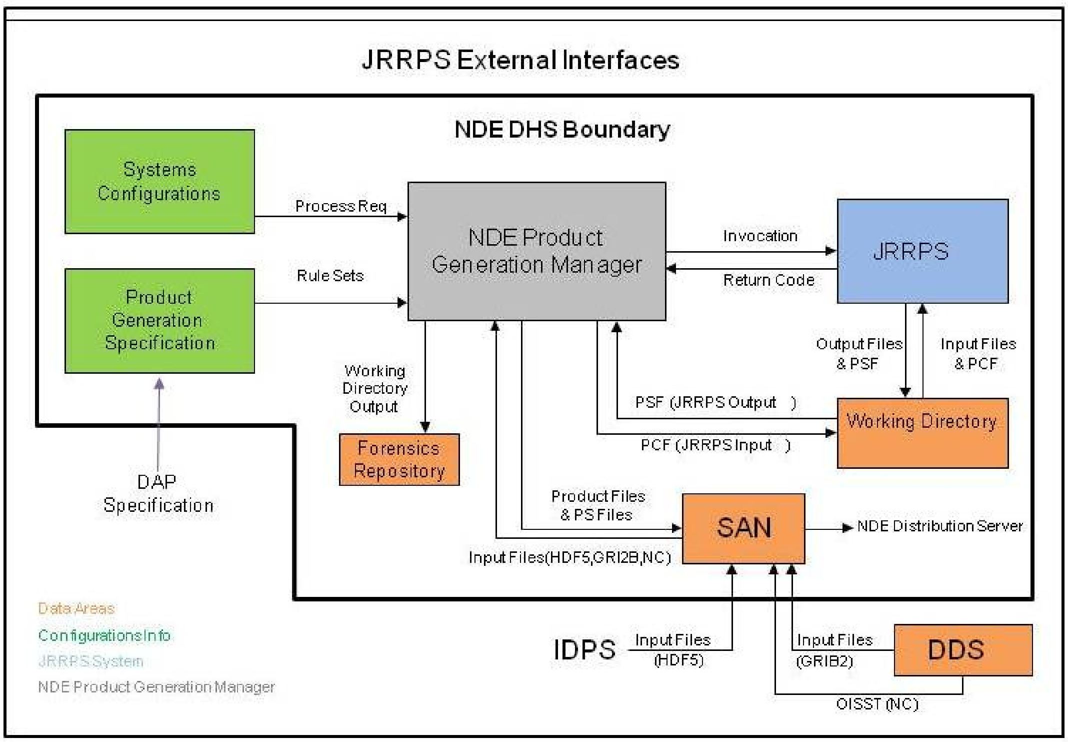


Figure 1-1: JPSS RR External Interfaces

There is only 1 system processing unit for JRR, JRRPS, including 1 preprocessing unit and 5 products units.

2. HARDWARE

2.1. Hardware Description

The hardware to support the NDE Production Environment-1 environment includes computing platforms that support AIX and Red Hat Linux as well as various networking equipment. Some components of the current PE-1 environment are shared resource while other components are PE-1 dedicated resources.

A description of the hardware required for the NDE PE-1 environment requires a basic understanding of the interconnectivity between NDE and the ESPC zones, IDPS, and storage. Figure 2-1 illustrates NDE PE-1 connectivity.

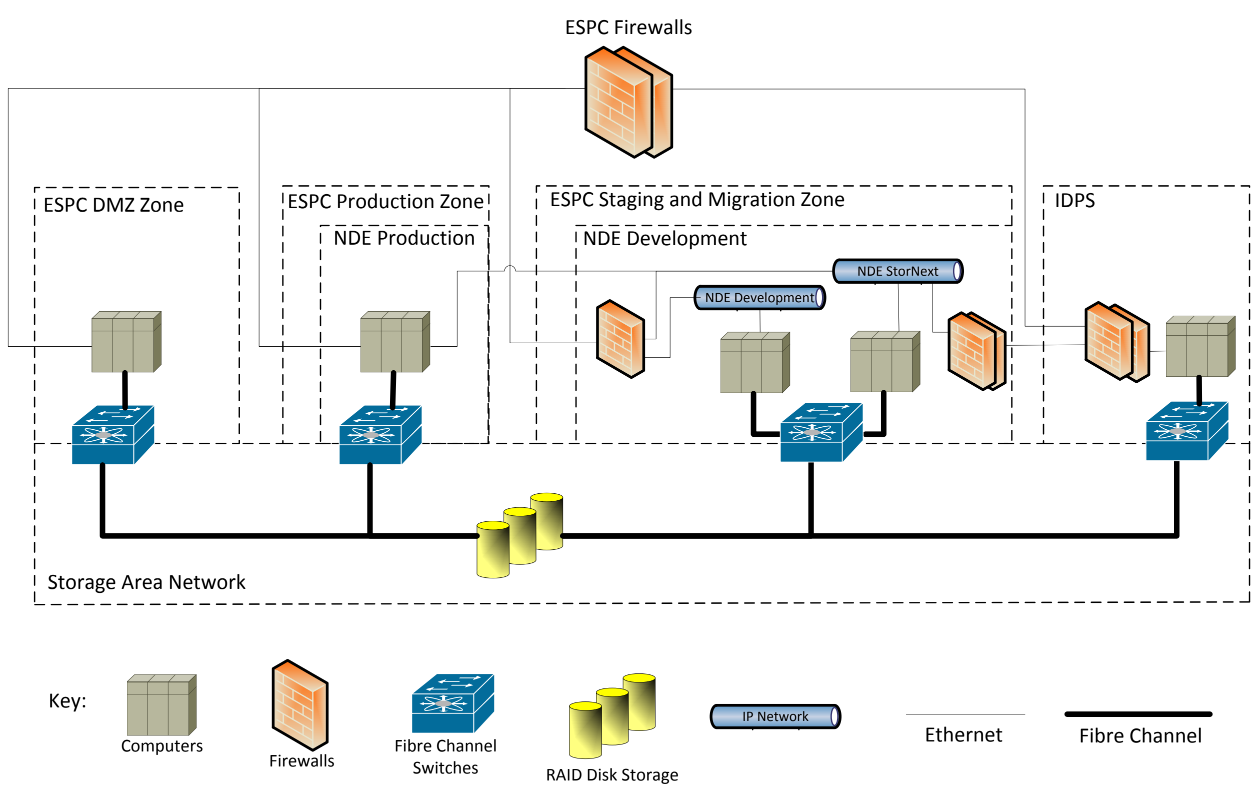


Figure 2-1: A high level logical diagram of the connectivity used to describe NDE PE1 hardware

The NDE Data Handling System (DHS) consists of Ingest (INS), Product Generation (PGS), Product Distribution (PDS), and Monitoring subsystems; and is physically spread across numerous servers, both IBM and Linux based, and a Storage Area Network (SAN). All servers, i.e. nodes, are 64-bit. An Ethernet network is used as the front-end for all nodes to access files on the SAN. The following table lists all pertinent hardware items and their NDE function.

Table 2-1: Summary of PE-1 Hardware

Hardware Item	NDE Function
IBM BladeCenter H Chassis	
(7) IBM PS701 Blade Servers	Product Generation, Science & Tailoring Alg
(2) Cisco 4Gb Fibre Channel Switches	SAN/DAS connectivity for PS701 servers
(2) Cisco Ethernet Switches	Network connectivity for PS701 servers
IBM DS3512 Storage Array	Local storage for PS701 servers
IBM EXP3512 Expansion Array	Expansion chassis for IBM DS3512 Storage
(2) Dell PowerEdge R610 Servers	Oracle Database Servers
(2)EMC AX4-5F Storage Array	Local Storage for R610 servers
Dell M1000E BladeCenter Chassis	
(8) Dell M610 Blade Servers	Ingest, Tailoring,

(2) Dell Fibre Channel Modules	SAN/DAS connectivity for M610 servers
(2) Cisco 3130G Ethernet Switches	Network connectivity for M610 servers
EMC AX4-5F Storage Array	Local storage for M610 servers
(2) CAI Networks 690PG WebMuxes	FTPS/HTTPS server load balancing
EMC Storage Array [ESPC controlled]	Shared storage
(2) Cisco MDS 9506 Fibre Channel Switches	SAN infrastructure component
(2) Cisco MDS 9222 Fibre Channel Switches	SAN infrastructure component
(2) Dell PowerEdge 2950 Servers	StorNext MDC servers [shared filesystem controller]
LSI Storage Array	Local storage for 2950 servers
(2) Cisco ASA 5500 Series Firewalls	Network boundary control [IDPS StorNext interface]
Cisco Catalyst 2960 Ethernet switch	StorNext network switch [shared network switch]

The following table maps the hardware items listed above to their physical interface (see Figure 2-2).

Table 2-2: Mapping of PE-1 Hardware to Physical Interface

Device	Support Group	Physical Interfaces			
		D	P	M	S
IBM BladeCenter H Chassis	ESPC		✓	✓	✓

IBM DS3512 Storage Array	ESPC		✓		✓
IBM EXP3512 Expansion Array	ESPC				
(2) Dell PowerEdge R610 Servers	ESPC		✓		
Dell M1000E BladeCenter Chassis	ESPC	✓	✓	✓	✓
EMC AX4-5F Storage Array	ESPC		✓		✓
(2) CAI Networks 690PG WebMuxes	ESPC	✓			
EMC Storage Array [ESPC controlled]	ESPC				✓
(2) Cisco MDS 9506 Fibre Channel Switches	NDE			✓	✓
(2) Cisco MDS 9222 Fibre Channel Switches	NDE			✓	✓
(2) Dell PowerEdge 2950 Servers	NDE			✓	✓
LSI Storage Array	NDE			✓	✓
(2) Cisco ASA 5500 Series Firewalls	NDE			✓	
Cisco Catalyst 2960 Ethernet switch	NDE				✓

Support Group = responsible administrative group after delivery/turnover of PE-1 to ESPC
Physical Interfaces = environments on which unit has physical interfaces
D=DMZ
P=Production
M=StorNext Ethernet metadata network
S=SAN Fibre Channel network

2.1.1. Storage Area Network

The most widely shared resource is the storage network that is used across the entire environment shown in Figure 2-1**.** The StorNext SAN file system form Quantum is used to manage this common storage across platforms. An Ethernet network carries what is known as the metadata for StorNext file systems, and is used as the front-end for all clients to access files on a StorNext filesystem.

The SAN itself is implemented on a Fibre Channel network. All hosts that require access to SAN storage must have integrated Host Bus Adapters (HBAs) that provide the necessary Fibre Channel connectivity. NDE administers two Cisco MDS Fibre Channel switches on each of two fabrics for this network. EMC Symmetrix disk storage administered by ESPC is used for PE1. Though the StorNext filesystem that is used for PE1 is currently also used for testing within NDE, an additional dedicated storage area is configured on ESPC's EMC Symmetrix for the testing environment [to be PE-2].

The satellite data provided by IDPS is placed on the SAN. NDE production servers use this data to generate products which are also placed on the SAN and then supplied to external customers. The FTPS protocol is used to distribute data from the DMZ. Some customers have data pushed to them and others pull data. External users are provided with accounts on NDE Distribution servers in the DMZ, and data that has been placed on the SAN is available to them when they log in. The FTPS protocol has been selected because it allows for an encrypted FTP control connection (for security) but allows for an unencrypted data connection (for a more efficient transfer of data).

2.1.2. StorNext Metadata Controllers

Currently, there are two Red Hat Enterprise Linux Version 5 (RHEL5) servers on the NDE StorNext network that serve as a failover pair of Metadata Controllers for StorNext. These are rack mounted servers that control the StorNext resources for all environments. A separate RAID chassis provides local storage for these servers.

2.1.3. Dedicated Resources

Much of the equipment for PE1 is dedicated and not shared for any other purpose. A more detailed description of these resources will follow, but a general description of these is provided here.

2.1.4. Dell Blade Servers

The Dell BladeCenter chassis contains eight blade servers. Each blade server is a single board that is placed in one slot of the BladeCenter. These Dell blades use Intel chips and provide the computing resources for one logical host running RHEL. Different Linux servers are used for ingest, tailoring and distribution. The necessary Ethernet interfaces and Fibre Channel HBAs for each server are also integrated on these boards.

Each server requires one Ethernet interface for its main PE-1 network (either production or DMZ), and another for the StorNext network used for metadata control information. The HBAs are required for the Fibre Channel connection to the storage.

There are two Ethernet switches inside the BladeCenter chassis that are used to connect the Blade Servers to the Ethernet switches for the PE1 and StorNext networks. The HBAs connect to pass-through devices in the Blade Center chassis to the NDE Fibre Channel switches. The BladeCenter chassis also has redundant power supplies.

Some blades are located in a BladeCenter chassis in the DMZ zone and some are in a BladeCenter chassis in the PROD zone. All are connected to the SAN Fibre Channel network.

2.1.5. Dell M 1000e PowerEdge Blade Center Enclosure

There are two Dell M100e PowerEdge Blade Center chassis for PE-1. A M1000e chassis can include up to 16 half-height blades (server modules), providing support for each blade with its power supplies, fan modules, Chassis Management Controller (CMC) modules, and I/O modules for external network connectivity.

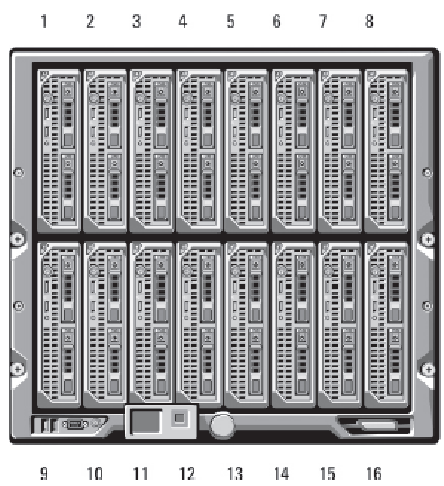


Figure 2-2: As configured for PE-1 each M1000e chassis contains eight PowerEdge M610 half-height blade servers

Each of the M610 blade server cards contains two Xeon 5660 series 6-core 2.80 GHz Intel processors with 36 GB of memory, two 300GB SAS disk and dual Ethernet and Fibre Channel interfaces. Connections to the external networks are provided through the cards shown below:

Table 2-3: External Network Cards

Location	Description	Function
Slot A1	Cisco Catalyst WS-CBS3130G-S Ethernet Switch	Ethernet switch for Production network
Slot A2	Cisco Catalyst WS-CBS3130G-S Ethernet Switch	Ethernet switch for StorNext network
Slot B1	Fibre Channel SAN Pass-Through Module	Fibre Channel Connectivity
Slot B2	Fibre Channel SAN Pass-Through Module	Fibre Channel Connectivity
Primary CMC Bay	Chassis Management Controller (CMC)	Primary CMC
Secondary CMC Bay	CMC	Secondary CMC

2.1.6. Dell M 1000e PowerEdge BladeCenter Local Storage

In addition to the StorNext filesystem available on the SAN, an EMC AX-4 RAID subsystem contains additional storage this is made available through the SAN to each of the eight RHEL5 hosts that are resident in the PROD Dell BladeCenter. The subsystem consists of an AX-4 chassis containing (2) controllers and (12) 600 GB SAS disk, and one chassis with (2) Standby Power Supplies that provide temporary battery backup to controllers in the event of a power failure.

2.1.7. Dell M 1000e PowerEdge Blade Center Enclosure

The Oracle database is hosted on a pair of Dell PowerEdge R610 rack-mounted servers. Each server has a single 3.33 Ghz X5680 Xeon processor. As configured for PE-1, each server also has:

96 GB memory
(2) 149 GB SSD disks
Four integrated 1 Gbps Ethernet NICs
(2) QLogic QLE2562 Fibre Channel HBAs

2.1.8. Dell M 1000e PowerEdge Blade Center Enclosure

Each of the Oracle servers has an EMC AX-4 RAID subsystem dedicated and direct attached to it. The subsystem consists of an AX-4 chassis containing (2) controller and (12) 600 GB SAS disks, and one chassis with (2) Standby Power Supplies that provide temporary battery backup to controllers in the event of a power failure.

2.1.9. IBM Blade Servers

The IBM BladeCenter chassis contains seven blade servers. Each blade server is a single board that is placed in one slot of the BladeCenter. These IBM blades use proprietary Power chips from IBM and provide the computing resources for one logical host running IBM's AIX operating system. The AIX servers are used for product generation and science algorithms. The necessary Ethernet interfaces and Fibre Channel HBAs for each server are also integrated on these boards.

Each server requires one Ethernet interface for the PE-1 network and another for the StorNext network used for metadata control information. The HBAs are required for the Fibre Channel connection to the storage.

There are two Ethernet switches inside the BladeCenter chassis that are used to connect the Blade Servers to the Ethernet switches for the PE1 and StorNext networks. The HBAs connect to Fibre Channel switches that are contained in I/O slots of the Blade Center chassis for connectivity to the NDE Fibre Channel switches. The BladeCenter chassis also has redundant power supplies.

2.1.10. IBM BladeCenter H

The IBM BladeCenter H chassis contains Power supplies, switches and blades in 9U of rack space.

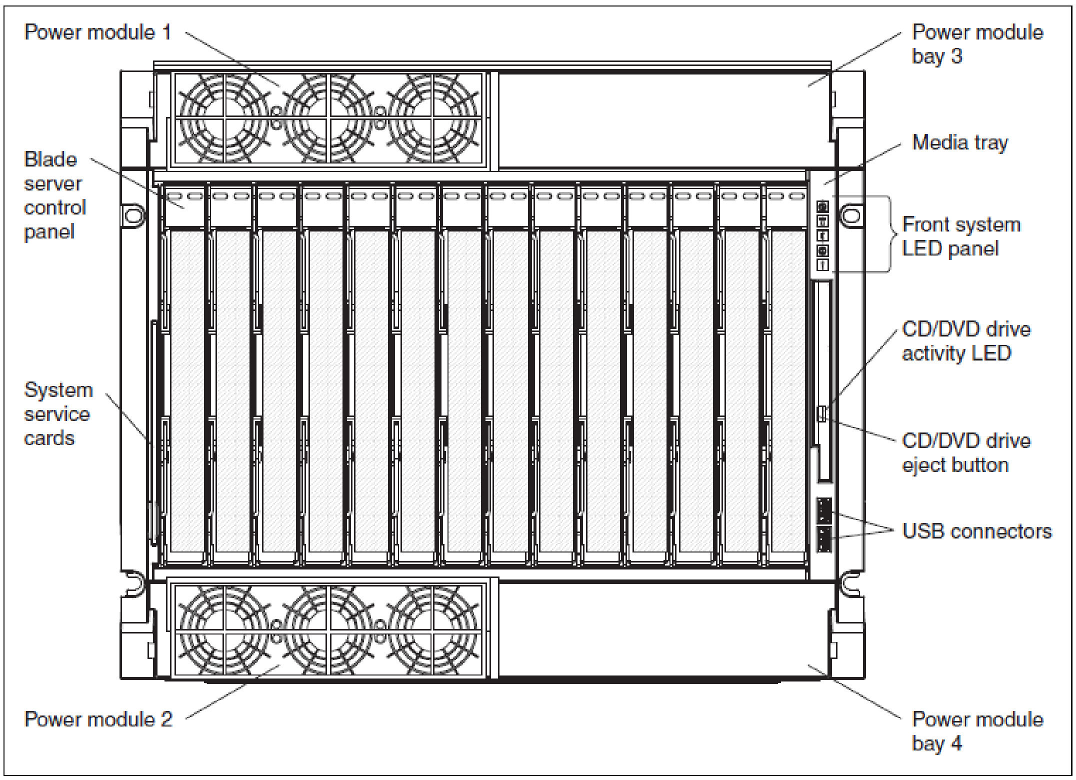


Figure 2-3: Shows the front view of the BladeCenter H chassis, with 14 slots available for blades and shared media tray on the right

The chassis has 14 slots for blades, of which 7 are currently used to hold single-slot PS701 blades (model 8406-71Y), which each of the 7 blades having a single socket 8-core 3.0 GHz POWER7 64-bit processor. As configured for PE-1, each blade also contains:

- 64 GB memory
- 600 GB SAS disk
- Two integrated 1 GB Ethernet NICs
- One Emulex 8 GB Fibre Channel HBA Expansion card (CIOv) with 2 ports

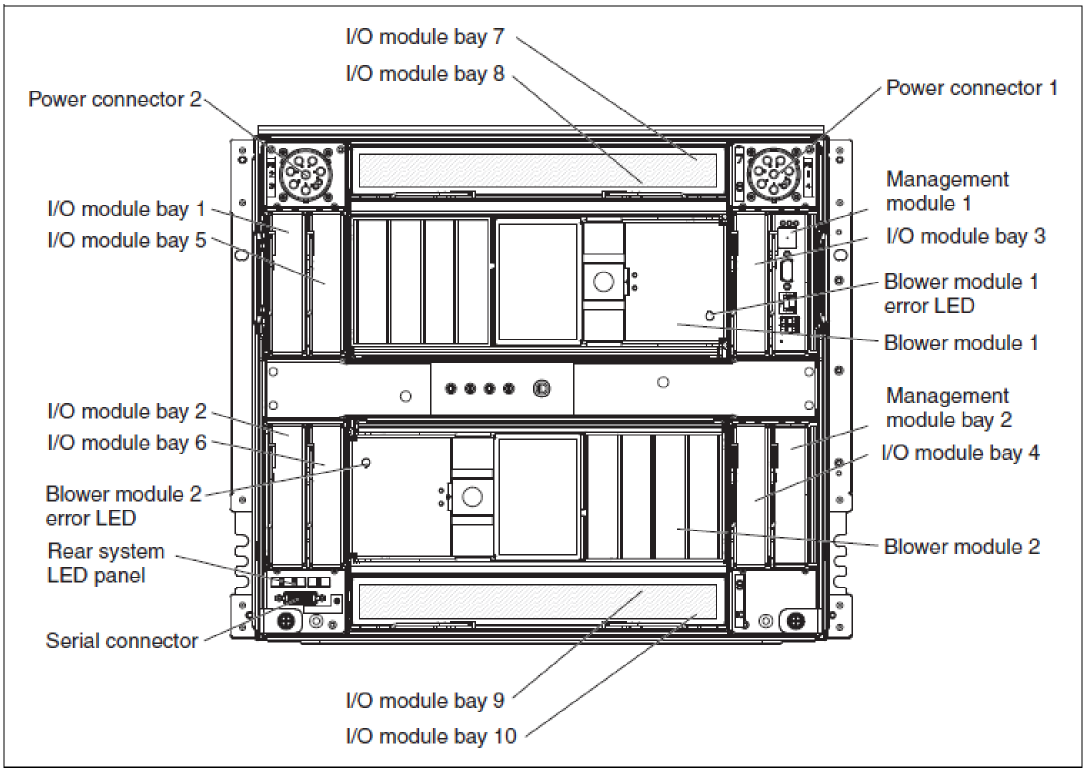


Figure 2-4: This is the rear of the chassis, with the I/O modules that contain the switches and the management modules.

Table 2-4: PE-1 IBM BladeCenter H I/O Bays

Location	Destination	Function
I/O Bay 1	Cisco Catalyst Switch Module 3012	Ethernet switch for production network

I/O Bay 2	Cisco Catalyst Switch Module 3012	Ethernet switch for StorNext network
I/O Bay 3	Cisco 4 GB 10 port FC Switch Module	Fibre Channel Switch
I/O Bay 4	Cisco 4 GB 10 port FC Switch Module	Fibre Channel Switch
Management Module Bay 1	Advanced Management Module (AMM)	Primary AMM
Management Module Bay 1	Advance Management Module (AMM)	Standby AMM

2.1.11. IBM RAID Storage

There are two separate chassis that provide RAID storage for the AIX hosts, a main chassis with dual controllers, disk drives, and an expansion chassis containing just disk drives. Storage created on this RAID subsystem is assigned to each of the AIX servers. The RAID controllers are connected to the NDE Fibre Channel switches, and the AIX servers access the storage through the SAN.

2.1.12. IBM DS3512 Storage Array

In addition to the StorNext filesystem available on the SAN, an IBM DS3512 RAID subsystem contains additional storage that is made available through the SAN to each of the seven AIX hosts that are resident in the IBM BladeCenter. The subsystem consists of one DS3512 chassis containing 2 controllers and 12

600 GB SAS disks, and one EXP 3512 Expansion chassis containing an additional 12 600 GB SAS disks.

Figure 2-5: The DS3512 appears identical from the front.

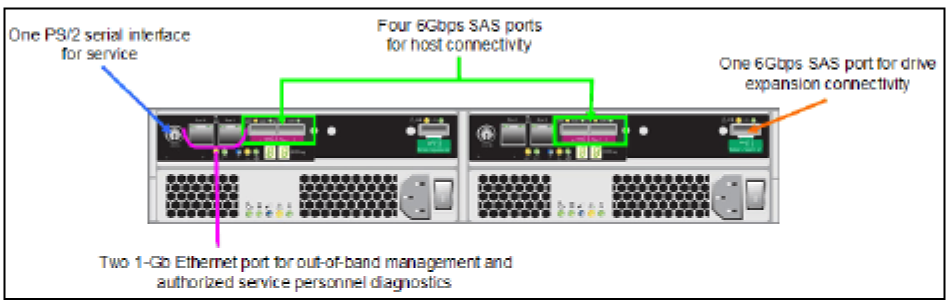


Figure 2-6: The rear of the DS3512 shows the ports available on the two controllers in that unit

2.1.13. Oracle Database Servers

There are rack mounted Dell servers that function as a redundant pair of Oracle database servers. Each runs RHEL5 on Intel processors.

2.1.14. EMC Storage for Dell Servers

There are three rack-mounted RAID storage subsystems from EMC that provide local storage for the Dell servers. One of these subsystems is dedicated as local storage for each of the Oracle servers, and is directly connected to the HBAs of these servers. The other subsystem is connected through the SAN and provides storage for all of the blade servers in the PROD Dell BladeCenter chassis.

2.1.15. WebMux Devices

Two WebMux devices are installed as a failover pair on the DMZ network. External users connect to these devices to establish the FTPS control connection to NDE Distribution servers. Using a round-robin type of allocation algorithm, the WebMux passes the connect request to an FTPS server in the server farm. The FTPS server provides the client with appropriate IP address and port information to establish a passive data connection. The external client will initiate a passive data connection directly to the Distribution server to retrieve data.

There are two WebMux 690PG Load Balancer devices from AVANU, which are a front-end to the NDE distribution servers in the DMZ. These are configured as a redundant pair. The FTPS server farm is configured on round-robin and the HTTPS server farm is configured on persistent round-robin.

2.2. Operating System

The NDE Data Handling System (DHS) is implemented across a host of servers running both Advanced Interactive eXecutive (AIX) on Power Series 7 machines and Redhat Enterprise Linux on Dell Blade servers, both operating systems are based on 64 bit architectures. NOAA Unique Product (NUP) algorithms are compiled for the AIX operating system using the standard set of XL compilers. In the future, all algorithms will be transitioned to Redhat Enterprise Linux with both the GNU and Intel set of compilers. DHS uses the open source JBoss Service-Oriented Architecture (SOA) platform to define and maintain all necessary services. An Oracle Database Management System provides the back-end for all data management. Each processing node (i.e. server) have science libraries installed locally. NDE maintains a set of approved libraries that algorithms can utilize: Hierarchical Data Format 5 (HDF5) and associated libraries (SZIP, ZLIB), Network Common Data Format (NetCDF-4), python, Perl, and Grib/Grib2.

2.3. System Requirements

2.3.1. Storage Requirements

Table 2-5 contains a list of storage requirements for all JPSS RR primary and ancillary input data, intermediate data, and output data for one day. The data size here is an estimated one because the number of granules used to form the remapped files is different case by case (see section 2.3.3). These requirements reflect the current build of JPSS RR. Note that the data product files are compressed as the products are generated.

Table 2-5: JPSS RR Storage Requirements

Storage Item	Size (1 day data storage)
Data Products Files	~300GB
Incoming SDR Data and IP	~206GB
Static Ancillary	~17 GB
GFS Forecasts	~1.1GB
OISST Daily	~8 MB
System Code (executables, include third party of H5augjpss and ms2gt)	~23 MB
System scripts	~208 KB
System Resource Files	96 KB
CRTM version 2.0.2 package	~101 MB

Total	~525 GB

Table 2-5 also contains the storage requirements for all JPSS RR resource files, tools and system code. Tools/libraries such as Linux Intel compilers, ShellB3 (Python), Perl, NetCDF4, HDF5, szip and zlib are installed and managed by IT system administrators, and therefore, are not included in the table. The CRTM version 2.0.2 is required by JPSS RR, and is part of the delivery since it is not an NDE managed and shared resource. The storage for this is included in the table.

(For 1 granule, there are 23 SDR files in netCDF of 578M, CldMask of 139M, CldHeight of 155M, CldPhase of 40M, DCOMP of 87M, NCOMP of 112M, ADP of 61M, AOD of 308M, VolAsh of 219M, IceAge of 47M, IceConc of 52M, SnowCover of 132M, and SnowFraction of 113M. Night time granule: NCOMP; Daytime granule: DCOMP, aerosols, snow cover/fraction)

2.3.2. Computer Resource Requirements

This section describes the computer hardware and software resources needed for the operations environment. This includes storage capacity and timeliness requirements.

2.3.2.1. Processing Requirements

The processing requirement for JPSS RR products is for them to be made available to users (e.g. on the distribution server) within 3 hours (maximum 6 hours) of observation, or 30 minutes of data receipt from IDPS. This requirement is identified in the JPSS RR RAD and CDR (2014). The JPSS RR software was developed and tested to run on the 64-bit Linux (16 3.20GHz CPUs, 2GB memory and 12288 KB cache size per CPU) hardware platform running on Intel 13.0.1 (STAR) and Intel 14.0.1 (NDE). Observation of run times shows that for nominal operational processing, the system requires an average of 2 CPUs at any given time to maintain latency. More CPUs will be required to keep up with the processing if the system has been shut down and is required to catch up.

Table 2-6 lists the time requirements for an instance run of each sub-system. These specifications on these processes assume nominal data flows. The maximum memory for the processing is also listed in Table 2-6. The maximum memory occurs during the execution of the AIT-framework code. The maximum memory requirement for one copy of AIT-framework run is 3.4 GB. The maximum number of the copies of AIT-framework run is 1 in JRR_PRODUCT_CLOUD_MASK.pl, JRR_PRODUCT_CLOUD_CLOUDS.pl, JRR_PRODUCT_AEROSOL_AODADP.pl, JRR_PRODUCT_AEROSOL_VOLASH.pl, JRR_PRODUCT_CRYOSPHERE_ICE.pl, and JRR_PRODUCT_CRYOSPHERE_SNOW.pl. The maximum memory is 3.4 GB for processing of JRR_PRODUCT_CLOUD_MASK.pl, JRR_PRODUCT_CLOUD_CLOUDS.pl, JRR_PRODUCT_AEROSOL_AODADP.pl, JRR_PRODUCT_AEROSOL_VOLASH.pl, JRR_PRODUCT_CRYOSPHERE_ICE.pl, and JRR_PRODUCT_CRYOSPHERE_SNOW.pl. The maximum memory is defined as the largest amount of memory that a given process uses at any time during its execution. The time shown is the amount of time required for the entire process to run, not the amount of time of peak memory consumption. The memory values are approximate since most data are dynamically allocated. Times are also approximate since they may vary due to fluctuations in bandwidth availability to external servers.

Table 2-6: JPSS RR System Processing Time Requirements

Sub-System	**Maximum Memory**	**CPU Time**	**Process Time**
JRR_PREPROCESS_SDR.pl	*6.8 GB*	~ 2 minutes	~2 minutes
JRR_PRODUCT_CLOUD_MASK.pl	*3.4 GB*	~2 minutes	~2 minutes
JRR_PRODUCT_CLOUD_CLOUDS.pl		~5 minutes	~5 minutes
JRR_PRODUCT_AEROSOL_AODADP.pl		~4 minutes	~4 minutes
JRR_PRODUCT_AEROSOL_VOLASH.pl		~2 minutes	~2 minutes
JRR_PRODUCT_CRYOSPHERE_ICE.pl		~1	~1

		minute	minute
JRR_PRODUCT_CRYOSPHERE_SNOW.pl		~2 minutes	~2 minutes

2.3.2.2. Libraries and Utilities

JPSS RR requires the following libraries and utilities with compiler Intel 13.0.1:

- NetCDF4 Library version 4.1.3 (including ncdump, available from Unidata website)
  - Compiled as 64 bit
- HDF5 Library version 1.8.10 (including h5dump, available from HDF Group website)
  - Compiled as 64 bit
  - Requires szip 2.1 – http://www.hdfgroup.org/ftp/lib-external/szip/2.1/src/szip-2.1.tar.gz, and zlib 1.2.5 – http://www.hdfgroup.org/ftp/lib-external/zlib/zlib-1.2.5.tar.gz , or zlib 1.2.7 (for Intel 14.0.1)
- ShellB3/python version 2.7.3, a package that includes source code packages and a shell script to extract and build them is: ftp://ftp.ssec.wisc.edu/pub/shellb3/ShellB3-Linux-x86_64-20120621-r713-core.tar.gz
- libxml2 (required by h5augjpss, see below)
- wgrib2 version 0.1.8.3 (available from NCEP CPC website -- http://www.cpc.ncep.noaa.gov/products/wesley/wgrib2)
  - Compiled as 64 bit

The above libraries and utilities will be maintained by NDE or OSPO as they are a common resource shared by all the NDE product teams. JPSS RR anticipates regularly updating to the newest working and stable versions of these tools throughout the project lifecycle. If the Intel compiler version is other than 13.0.1, a different version of NetCDF4, HDF5, szip and zlib might be used to match the compiler.

The following software/utility are also required by JPSS RR:

- CRTM Library version 2.0.2 (available from Joint Center for Satellite Data Assimilation(JCSDA)
  - Compiled as 64 bit
  - Source code supplied within the JPSS RR DAP -- REL-2.0.2.JCSDA_CRTM_gm.tar
- JPSS HDF5 to NetCDF4 conversion utility ***(h5augjpss,** v1.1.0 --* http://www.hdfgroup.org/projects/jpss/h5augjpss_index.html). It will require Libxml2 to compile (see README.txt contained in the package).
  - Command line tool that modifies JPSS RR HDF5 files by adding associated data or metadata or by hiding HDF5 elements in order to make the file accessible to NetCDF4 based applications and tools.

2.3.2.3. Linux Intel Compilers

The software requires be compiled and run on 64bit Linux machine. Only the test machine requires the presence of compilers. The production machine does not need a compiler because only the tested compiled code is necessary on the production machine. After the system has been successfully tested on the test machine, it is moved (copied) over to the production machine. However, only the compiled code is moved over. This also ensures that nothing is changed after the test and that the identical code is used on the test machine in the event of a failover from production.

To compile on the test machine the Intel version 13.0.1Fortran and C/C++ compiler must be present.

2.3.2.4. Perl

The system was developed using Perl version 5.10.1, but should function with any newer version. The following Perl modules are required by the system:

FileHandle
POSIX
Sys

Switch
File
Date
Cwd
Time

These are all standard Perl library modules and should not vary significantly among versions of Perl.

2.3.3. Communication Needs

Table 2-7 shows the one day data volumes to be transferred to NDE. Table 2-8 shows the one day data volumes to be transferred out of NDE's data distribution server to the outside world. These tables should be used as a guide for determining the disk space and bandwidth for the DHS.

Table 2-7: JPSS RR Input Data Size

Product	Number of Files/day	Size/day	Provider
VIIRS GMTCO HDF5	~1000	~32 GB	IDPS
VIIRS GITCO HDF5	~1000	~113GB	IDPS
VIIRS SVM01 HDF5	~1000	~4.9 GB	IDPS
VIIRS SVM02 HDF5	~1000	~4.9 GB	IDPS
VIIRS SVM03 HDF5	~1000	~8.5GB	IDPS
VIIRS SVM04 HDF5	~1000	~8.8GB	IDPS
VIIRS SVM05 HDF5	~1000	~9.1GB	IDPS
VIIRS SVM06 HDF5	~1000	~4.9 GB	IDPS
VIIRS SVM07 HDF5	~1000	~9.1GB	IDPS
VIIRS SVM08 HDF5	~1000	~6.5GB	IDPS
VIIRS SVM09 HDF5	~1000	~2.4GB	IDPS
VIIRS SVM10 HDF5	~1000	~6.6GB	IDPS
VIIRS SVM11 HDF5	~1000	~6.2Gb	IDPS
VIIRS SVM12 HDF5	~1000	~6.1GB	IDPS

VIIRS SVM13 HDF5	~1000	~11GB	IDPS
VIIRS SVM14 HDF5	~1000	~6GB	IDPS
VIIRS SVM15 HDF5	~1000	~6GB	IDPS
VIIRS SVM16 HDF5	~1000	~6GB	IDPS
VIIRS SVI01 HDF5	~1000	~24GB	IDPS
VIIRS SVI02 HDF5	~1000	~25GB	IDPS
VIIRS SVI03 HDF5	~1000	~26GB	IDPS
VIIRS SVI04 HDF5	~1000	~25GB	IDPS
VIIRS SVI05 HDF5	~1000	~24GB	IDPS
GFS GRIB2	16	~1.1GB	NCEP
OISST Daily	1	~8 MB	NCDC
Total	~23013	~382 GB	IDPS, NCEP and NCDC

Table 2-8: JPSS RR Output Products Size (Compressed)

Product	**Number of Files/Day**	**Size/Day**	**User**
Aerosol Detection	~500	~8.5GB	CLASS/NCEP
Aerosol Optical Depth	~500	~72GB	CLASS/NCEP
Volcanic Ash	~1000	~50GB	CLASS/NCEP
Cloud Mask	~1000	~20GB	CLASS/NCEP
Cloud Height	~1000	~41GB	CLASS/NCEP
Cloud Phase	~1000	~9GB	CLASS/NCEP
Daytime Cloud Optical and Microphysical Properties (DCOMP)	~500	~8GB	CLASS/NCEP
Nighttime Cloud Optical and Microphysical Properties (NCOMP)	~500	~15GB	CLASS/NCEP
Ice Thickness and Age	~1000	~8GB	CLASS/NCEP
Ice Concentration	~1000	~9GB	CLASS/NCEP
Snow Cover	~500	~13GB	CLASS/NCEP
Total	~8500	~255GB	CLASS/NCEP

3. SOFTWARE

This section describes the system-level software elements that are invoked by the NDE production system. For more details about the entire JPSS RR design down to the sub-system level and data flows at each level, see section 7 Appendix 1. Next, this section describes the source code and system files delivered to NDE. These files are organized into subdirectories. The contents of each subdirectory are identified and their purpose is explained.

3.1. Software Description

There are seven processing sub-systems in JPSS RR: the sub-system Preprocessor and the six sub-system Product-Generators. Each sub-system is operated by its own Perl driver script. Two processing units are involved in the sub-system Preprocessor: (1) the *SDR-Data-Preparation unit*, (2) the *Bayes Cloud-Mask-Generation unit*, (3) the *Cloud-Products-Generation unit*, (4) the *AODADP-Products-Generation unit*, (5) the *Volcanic-Ash-Products-Generation unit*, (6) the *Cryosphere-Ice-Products-Generation unit*, and (7) the *Cryosphere-Snow-Products-Generation unit*. The association between the sub-systems and units with the scripts is shown in Figure 3-1 and Figure 3-2.

The Preprocessor takes an orbit of VIIRS ADR and IP as input to create the NetCDF format file. The Preprocessor reads-in its Process Control File (PCF), then it will: 1) make necessary symbolic links for files and directories; 2) create subworking directory for each unit; 3) create unit level process control file for each unit; 4) invoke each unit one by one to generate the file; then 5) write the file to the PSF.

The *SDR-Data-Preparation unit* converts the HDF5 format VIIRS data to NetCDF4 format. It performs the following functions:

- It handles the VIIRS SDR SVM01-16 and SVI01-05 data gap filling if the GAPFILL key is ON; otherwise it skips the gap filling.
  It converts VIIRS SDR SVM01-16, SVI01-05, GMTCO, and GITCO from HDF5 to NetCDF4.

The *Product-Generation units* reside downstream of the *SDR-Data-Preparation Unit*. They are called to generate the Framework output of Cloud Products, Aerosol Products, and Cryosphere Products by calling the Framework program.

The sub-system Product-Generator units take the NetCDF format file as input to generate the JPSS RR products. The Product-Generator units read-in their PCF files, then it will: 1) rename all SDR data for Framework use by extracting the scan time information from the GMTCO file and attaching it to the netCDF file names; 2) set up running environment for Framework by making necessary symbolic links for files and directories; 3) create Framework process control file using a template ; 4) invoke the JPSS RR-Product-Generation unit to generate the JPSS RR products; then 5) write the JPSS RR products to the sub-system PSF.

3.2. Directory Description

This section describes the directory structure of the source code, the processing scripts and the system files that are delivered to NDE in the Delivered Algorithm Package (DAP). The content and purpose of each subdirectory is briefly described. Note: this section does not describe the directory structure of the JPSS RR processing system. The directory structure for all JPSS processing is determined by the NDE system developers (see section 4.1 the system PCF files).

Paths are always shown in italics. Because the system could be installed at any location on a given machine, the highest level directory path of installation for the system is always referred to as "*$BASE*". So, for example, $BASE might be "/data/data092/hxie/dlv/npp_dap/JRR_DAP_201512". Therefore, the location of all files and directories is described with respect to this high level directory. Since the NDE DHS will handle the staging of all JPSS RR files and directories, we only provide the source code and the processing system directories description.

The NOAA VIIRS JPSS RR Processing System processing script and system files are located in the following directory:
$BASE/JRRPS-OPS

*A*nd below are its sub-directories*:*

- *Binary subdirectories* – A place to put the system's executable files. All the successfully generated executables need to be moved to this location. Utilities including python and h5augjpss are also required to be in (or link to) this directory. This bin directory can be set anywhere as long as it matches the setting of the variable $OPS_BIN in the system's PCF files. Below are files that must appear in the $OPS_BIN directory:

  $BASE/JRRPS-OPS/bin

  get_sdr_scan_time.exe
  pcf_framework.exe
  h5augjpss

  There is also a link to the 'python' executable in this directory.

  Note, the above executables and utilities are not included in the DAP.

- *Template subdirectories* – Contains template AIT-framework level configuration files (CFG) and SDR product control files (pcf). All files in this directory should not be modified. Files contained in the directory are:

  $BASE/JRRPS-OPS/templates

  NPP_VIIRS_AEROSOL_AODADP_Template.cfg
  NPP_VIIRS_AEROSOL_VOLASH_Template.cfg
  NPP_VIIRS_CLOUD_CLOUDS_Template.cfg
  NPP_VIIRS_CLOUD_MASK_Template.cfg
  NPP_VIIRS_CRYOSPHERE_ICE_Template.cfg
  NPP_VIIRS_CRYOSPHERE_SNOW_Template.cfg
  NPP_VIIRS_SDR_Template.pcf

- *PCF_Overwrites subdirectories* – Contains AIT-Framework level process control files (Framework-PCF). Some of them are used as templates. All files in this directory should not be modified, unless the write-out option is switched on each of the related intermediate data files (see below). Files contained in the directory are:

  $BASE/JRRPS-OPS/PCF_Overwrites

  AWG_AER_AOD.pcf
  AWG_Cloud_Micro_Day.pcf

AWG_Ice_Age_VIIRS_BAYES.pcf
AWG_Ice_Conc_VIIRS_BAYES.pcf
Climatic_LST.pcf
Coast_Mask.pcf
CRTM.pcf
Desert_Mask.pcf
EPS_ADP.pcf
EPS_Cld_Base.pcf
Land_Mask.pcf
NPP_BAYES_Cloud_Mask_NW.pcf
NPP_BAYES_Cloud_Mask.pcf
NPP_VIIRS_AERADP.pcf
NPP_VIIRS_Cld_Height_NW.pcf
NPP_VIIRS_Cld_Height.pcf
NPP_VIIRS_Cld_NCOMP.pcf
NPP_VIIRS_Cld_Phase.pcf
NPP_VIIRS_SNOW_COVER.pcf
NPP_VOLCANIC_ASH_NW.pcf
NPP_VOLCANIC_ASH.pcf
NWP_Data_grib2_0.5deg.pcf
OISST_Daily_QtrDeg.pcf
Pseudo_Emissivity.pcf
Snow_Mask.pcf
Surface_Albedo.pcf
Surface_Elevation.pcf
Surface_Emissivity.pcf
Surface_Type.pcf
Volcano_Mask.pcf

Note, switch the “OUTPUT” to “Y” from “N” the associated framework-PCF to write out each related intermediate data files, or vice versa.

- Script subdirectories – Contains the processing systems processing scripts, each will be descript in section 3.3.

The source code is located within the four subdirectories of the JPSS RR products: $BASE/SOURCE. All the source code is located within SOURCE, and at least one of the following subdirectories is also located:

- *h5augjpss-1.0.0* – Contains third party software H5AUGJPSS
- *extract_scan_time* – Contains SDR scan-time-extract code and build
- *AIT-framework – Contains* AIT-framework code.

  Under the directory *AIT-framework/src*, there are sub-directories:

  Algorithm_Factory – Framework specific code
  Algorithm_Factory/Algorithms – Algorithm code
  Data_Handling_Structures – Framework specific code
  Data_Process – Framework specific code
  Data_Processing_Manager – Framework specific code
  Data_Structure_Code – Algorithm input and output data structures
  Includes – Global Parameter Code
  Main – Framework driver code
  NWP_Utils – Utility functions for the NWP GFS data
  RTM – Radiative transfer model functions
  Satellite_Readers – Functions for reading in the satellite data
  Scientific_Library_Code – Scientific library code
  Temporal_Readers – Readers for temporal data
  Utils – Framework utility functions

  The JPSS RR algorithm code and interface code are located within the following directories:

  AIT-framework/src/Algorithm_Factory/Algorithms/

3.3. Source Code Description

The JPSS RR system contains over twelve hundred programs, subprograms (functions and subroutines) and scripts written in Fortran 77/90/95, C/C++, Perl and Python. In total there are nearly four hundred thousand lines of code. The third-party software code h5augjpss is included in the DAP but not included in the number of programs and lines accumulating. It is simply not possible to identify each program and describe its purpose in this section. Therefore, this section identifies the script and source code of each main program, notes its relative location in the source tree, and describes its function.

Source Program

The source program (Fortran 77/90/95, C/C++) is located within the SOURCE subdirectory:

comp.sh -- A script to build all the executables from the source program (shared library for AIT-framework). Modify the script to point to the correct CRTM, NetCDF4, HDF5, ZLIB, SZIP and libxml2 libraries. Then, type:

```
sh   comp.sh   nde          – for NDE environment with shared library
```

Or

```
sh  comp.sh   nde  static  – for NDE environment with static library
```

The successfully generated executables will be located at: JRRPS-OPS/bin/

SOURCE/h5augjpss-1.0.0 *– contains unpacked H5AUGJPSS package* download from http://www.hdfgroup.org/projects/jpss/h5augjpss_index.html . Refer to the document contained in the package for the details about this software. To compile it, read the README.txt in the directory.

SOURCE/extract_scan_time – Contains SDR scan-time-extract code and build. Two files:

get_sdr_scan_time.f90 – The source code to extract the SDR scan time
build_get_sdr_scan_time – Build file for *get_sdr_scan_time.f90*

Modify the build file to point to the correct NetCDF4, HDF5 libraries. To compile, type:

```
. build_get_sdr_scan_time
```

After successful compilation the executable get_sdr_scan_time.exe will be generated in the same directory.

Command Line:
get_scan_time.exe <Name_Of_GMTCO_NetCDF_File>

The output is a text file:
time.txt – contains scan time information

SOURCE/AIT-framework – *Contains* AIT-framework code

Under the directory *SOURCE/AIT-framework/src*, there are the following sub-directories:

Algorithm_Factory – Framework specific code
Algorithm_Factory/Algorithms – Algorithm code
Data_Handling_Structures – Framework specific code
Data_Process – Framework specific code
Data_Processing_Manager – Framework specific code
Data_Structure_Code – Algorithm input and output data structures
Includes – Global Parameter Code
Main – Framework driver code
NWP_Utils – NWP utility functions
RTM – RTM code
Satellite_Readers – Satellite reader code
Scientific_Library_Code – Scientific library code
Temporal_Readers – Temporal reader code
Utils – Framework utility functions

The main program of the AIT-framework code is a C++ program:

SOURCE/AIT-framework/src/Main/framework.cpp

To build the Framework executable, go to the directory:
SOURCE/AIT-framework/src

There is a Makefile, update the Makefile to point to the correct NetCDF4, HDF5,

ZLIB, SZIP and CRTM2.0.2 libraries. To compile, type:
make clean
make

After successful compilation, the executable will be generated:
*SOURCE/AIT-framework/*pcf_framework.exe

Command Line:
pcf_framework.exe <framework-cfg-file>

Where,
<framework-cfg-file> is a configuration file that establishes the system setting and gives the Framework a list of algorithms to run, and products to make. The configuration file has four main sections. These are: Satellite information, Data dimensions, PCF Framework settings, Run settings. See section 7.3.2 for the examples. Couple things need to be aware about this file:

- There can only be one dot character in the full path of file <framework-cfg-file>. For the examples below, the first AIT-framework CFG file is fine, but not the second one:

  /data/data046/working/OUT/viirs_results_clouds/AWG_NPP_VIIRS_Clouds_2013122_0438_52_d201305020438.cfg

  /data/data046/**working.2**/OUT/viirs_results_clouds/AWG_NPP_VIIRS_Clouds_2013122_0438_52_d201305020438.cfg

» The pcf_framework.exe expects the following directories and script exist locally -- located in its current working directory:

Directories:
Default_PCF
CustomProductLists
PCF_Overwrites
algorithm_ancillary
framework_ancillary

perl script:
run_wgrib.pl

Scripts

The system processing scripts (Perl and Python) are located in the following directory with the individual scripts:

$BASE/JRRPS-OPS/scripts

jrr_preprocess_cvth52nc.pl – Subscript of the preprocessing SDR data, converting all SDR data from h5 to netCDF,

jrr_preprocess_gapfill_M.pl – Subscript of preprocessing SDR data, doing gap filling for the the M-Band SVM01-16 and SVI01-05 data

jrr_preprocess_gapfill_I.pl – Subscript of preprocessing SDR data, doing gap filling for the the I-Band SVM01-16 and SVI01-05 data

JRR_PREPROCESS_SDR.pl – Driver script for preprocessing SDR data

jrr_product_aer_aodadp.pl – Subscript of aerosol optical depth (AOD) and aerosol detection product (ADP) product driver script, obtaining AOD and ADP

JRR_PRODUCT_AEROSOL_AODADP.pl – Top level driver script for AOD and ADP

JRR_PRODUCT_AEROSOL_VOLASH.pl – Top level driver script for the aerosol volcanic ash

jrr_product_aer_volash.pl – Subscript of aerosol volcanic ash driver script, obtaining the aerosol volcanic ash

jrr_product_cld_clouds.pl – Subscript of clouds driver script, obtaining the cloud height, phase, NCOMP (for night time granule only), and DCOMP (for day time granule) products

jrr_product_cld_mask.pl – Subscript of cloud mask driver script, obtaining the Bayes cloud mask

JRR_PRODUCT_CLOUD_CLOUDS.pl – Top level driver script for the cloud height, phase, NCOMP(for night time granule only), and DCOMP (for day time granule)

JRR_PRODUCT_CLOUD_MASK.pl – Top level driver script for the Bayes cloud mask

jrr_product_cryos_ice.pl – Subscript of cryosphere ice driver script, obtaining the cryosphere ice concentration and ice age products

JRR_PRODUCT_CRYOSPHERE_ICE.pl – Top level driver script for the ice concentration and ice age

JRR_PRODUCT_CRYOSPHERE_SNOW.pl – Top level driver script for snow cover and snow fraction

jrr_product_cryos_snow.pl – Subscript of cryosphere snow driver script, obtaining the snow cover and snow fraction products

jrr_product_env4frmwk.pl – Subscript of product driver script, setting up the environment for running the Framework under NDE

jrr_product_sdr4frmwk.pl – Subscript of product driver script, renaming GMTCO/SVM??/GITCO/SVI?? data in netCDF for Framework use

mender.py – A Python script used to do the gap-filling on SDR HD5 data

run_wgrib.pl – A perl script used to read NWP data

viirsmend.py - Handles basic bowtie replacement on VIIRS granules

viirsmend.pyc - Handles basic bowtie replacement on VIIRS granules

Figure 3-1 and Figure 3-2 show the tree of script/executable/utility that been invoked in each driver script and the associated sub-system, unit and sub-unit names for the pre-processor and product generator, respectively:

JRR_PREPROCESS_SDR.pl

- jrr_preprocess_gapfill.pl
  - mender.py
    - viirsmend.py
- jrr_preprocess_cvth52nc.pl
  - h5augjpss

Figure 3-1: Sub-System Preprocessor Execution Tree

JRR_PRODUCT_CLOUD_MASK.pl (sub-system Cloud Mask Product-Generator)
- jrr_product_sdr4frmwk.pl
    - get_sdr_scan_time.exe
- jrr_product_env4frmwk.pl
- jrr_product_cld_mask.pl (unit JRR Cloud Mask Product-Generation)
    - pcf_framework.exe (sub-unit Framework Cloud-Mask-Generation)
        - run_wgrib.pl

JRR_PRODUCT_CLOUD_CLOUDS.pl (sub-system Clouds Product-Generator)
- jrr_product_sdr4frmwk.pl
    - get_sdr_scan_time.exe
- jrr_product_env4frmwk.pl
- jrr_product_cld_clouds.pl (unit JRR Cloud Clouds Product-Generation)
    - pcf_framework.exe (sub-unit Framework Cloud-Clouds-Generation)
        - run_wgrib.pl

JRR_PRODUCT_AEROSOL_AODADP.pl (sub-system Aerosol Product-Generator)
- jrr_product_sdr4frmwk.pl
    - get_sdr_scan_time.exe
- jrr_product_env4frmwk.pl
- jrr_product_aer_aodadp.pl (unit JRR Aerosol Product-Generation)
    - pcf_framework.exe (sub-unit Aerosol Framework-Generation)
        - run_wgrib.pl

JRR_PRODUCT_AEROSOL_VOLASH.pl (sub-system Volcanic Ash Product-Generator)
- jrr_product_sdr4frmwk.pl
    - get_sdr_scan_time.exe
- jrr_product_env4frmwk.pl
- jrr_product_aer_volash.pl (unit JRR Volcanic Ash Product-Generation)
    - pcf_framework.exe (sub-unit Volcanic Ash Framework-Generation)
        - run_wgrib.pl

JRR_PRODUCT_CRYOSPHERE_SNOW.pl (sub-system Cryosphere Snow Product-Generator)
- jrr_product_sdr4frmwk.pl
    - get_sdr_scan_time.exe
- jrr_product_env4frmwk.pl
- jrr_product_cryos_snow.pl (unit JRR Cryosphere Product-Generation)
    - pcf_framework.exe (sub-unit Cryosphere Snow Framework-Generation)
        - run_wgrib.pl

JRR_PRODUCT_CRYOSPHERE_ICE.pl (sub-system Cryosphere Ice Product-Generator)
- jrr_product_sdr4frmwk.pl
    - get_sdr_scan_time.exe
- jrr_product_env4frmwk.pl
- jrr_product_cryos_ice.pl (unit JRR Cryosphere Ice Product-Generation)
    - pcf_framework.exe (sub-unit Cryosphere Ice Framework-Generation)
        - run_wgrib.pl

Figure 3-2: Sub-System Product-Generator Execution Tree

4. NORMAL OPERATIONS

4.1. System Control

4.1.1. System Control Files

This section describes the input controls required to operate JPSS RR. Before any of the two JPSS RR driver script is invoked by the NDE system, the NDE system creates a working directory and a Process Control File (PCF) specific to that driver script and to that given run. The PCF is placed in the working directory and then an instance is run. The driver script, in turn, assumes the PCF is located in the current working directory. NDE will make a single call to the driver script with the working directory path as an argument. When the driver script finishes, in addition to its output products, it produces a log file and a Processing Status File (PSF), and returns a value to the calling NDE process. The NDE system then uses this to obtain information to determine whether the run was successful and which files are available for distribution or transfer to the next stage of processing.

4.1.2. Processing Controls

There are seven PCF and seven PSF files for/from the JPSS RR product types. These files are associated with the operation of each JPSS sub-system represented by the driver scripts JRR_PREPROCESS_SDR.pl.PCF,
JRR_PRODUCT_CLOUD_CLOUDS.pl.PCF,
JRR_PRODUCT_CLOUD_MASK.pl.PCF,
JRR_PRODUCT_AEROSOL_AODADP.pl.PCF,
JRR_PRODUCT_AEROSOL_VOLASH.pl.PCF,
JRR_PRODUCT_CRYOSPHERE_ICE.pl.PCF,
and JRR_PRODUCT_CRYOSPHERE_SNOW.pl.PCF. Note that there are two CLOUDS PCF files, one that includes DCOMP and the other with NCOMP. Each PCF file will contain the system setting and the names of the files to be processed along with their full path (location in the file system) if not located at the working directory. The PSF just contains the full path of the names of the output files generated successfully. A description of the various fields of a

preprocessing and product PCF and a preprocessing PSF are provided below followed by an example.

JRR_PREPROCESS_SDR.pl.PCF

working_directory – This is the directory at which the job processing occurs
job_coverage_start -- This is the pole-pass time of the first remapped file in YYYYMMDDHHMMSSS format (It is not used by the driver script but for NDE in searching the remapped file)
job_coverage_end -- This is the pole-pass time of the third remapped file in YYYYMMDDHHMMSSS format (It is not used by the driver script but for NDE in searching the remapped file)
PERL_LOC -- The location of the Perl interpreter
OPS_SCRIPT -- The location of the system script
OPS_BIN -- The location of the system executables
TIME_TIMEOUT – The limit of processor time for I-Band and M-Band that gap filling may occur on. If exceeded, gap-filling will be killed.
gitco – name of the VIIRS GITCO file
gmtco – name of the VIIRS GTMCO file
svi01 – name of the VIIRS SVI file channel 1
svi02 – name of the VIIRS SVI file channel 2
svi03 – name of the VIIRS SVI file channel 3
svi04 – name of the VIIRS SVI file channel 4
svi05 – name of the VIIRS SVI file channel 5
svm01 – name of the VIIRS SVM file channel 1
svm02 – name of the VIIRS SVM file channel 2
svm03 – name of the VIIRS SVM file channel 3
svm04 – name of the VIIRS SVM file channel 4
svm05 – name of the VIIRS SVM file channel 5
svm06 – name of the VIIRS SVM file channel 6
svm07 – name of the VIIRS SVM file channel 7
svm08 – name of the VIIRS SVM file channel 8
svm09 – name of the VIIRS SVM file channel 9
svm10 – name of the VIIRS SVM file channel 10
svm11 – name of the VIIRS SVM file channel 11
svm12 – name of the VIIRS SVM file channel 12
svm13 – name of the VIIRS SVM file channel 13
svm14 – name of the VIIRS SVM file channel 14
svm15 – name of the VIIRS SVM file channel 15
svm16 – name of the VIIRS SVM file channel 16

Below is an example of a
JRR_PREPROCESS_SDR.pl.PCF
file:

```
# working directory:
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b1
7764
# ProdRuleName: VIIRS granule (CloudMask)

working_directory=/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t105
4237_e1055479_b17764
job_coverage_start=201504021054237
job_coverage_end=201504021055479

GAPFILL=ON

PERL_LOC=/usr/bin/perl
OPS_SCRIPT=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-OPS/scripts
OPS_BIN=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-OPS/bin
TIME_TIMEOUT=300

gitco=GITCO_npp_d20150402_t1054237_e1055479_b17764_c2015040217225906705
9_noaa_ops.h5
gmtco=GMTCO_npp_d20150402_t1054237_e1055479_b17764_c20150402171617371
175_noaa_ops.h5
svi01=SVI01_npp_d20150402_t1054237_e1055479_b17764_c20150402172912367353
_noaa_ops.h5
svi02=SVI02_npp_d20150402_t1054237_e1055479_b17764_c20150402172855756881
_noaa_ops.h5
svi03=SVI03_npp_d20150402_t1054237_e1055479_b17764_c20150402174048712352
_noaa_ops.h5
svi04=SVI04_npp_d20150402_t1054237_e1055479_b17764_c20150402172950556360
_noaa_ops.h5
svi05=SVI05_npp_d20150402_t1054237_e1055479_b17764_c20150402172953919670
_noaa_ops.h5
svm01=SVM01_npp_d20150402_t1054237_e1055479_b17764_c20150402172853771
802_noaa_ops.h5
svm02=SVM02_npp_d20150402_t1054237_e1055479_b17764_c20150402172837504
244_noaa_ops.h5
svm03=SVM03_npp_d20150402_t1054237_e1055479_b17764_c20150402172859491
592_noaa_ops.h5
```

svm04=SVM04_npp_d20150402_t1054237_e1055479_b17764_c20150402172907467365_noaa_ops.h5
svm05=SVM05_npp_d20150402_t1054237_e1055479_b17764_c20150402172932197325_noaa_ops.h5
svm06=SVM06_npp_d20150402_t1054237_e1055479_b17764_c20150402172928866557_noaa_ops.h5
svm07=SVM07_npp_d20150402_t1054237_e1055479_b17764_c20150402172933775655_noaa_ops.h5
svm08=SVM08_npp_d20150402_t1054237_e1055479_b17764_c20150402172930996633_noaa_ops.h5
svm09=SVM09_npp_d20150402_t1054237_e1055479_b17764_c20150402172912259531_noaa_ops.h5
svm10=SVM10_npp_d20150402_t1054237_e1055479_b17764_c20150402172939082298_noaa_ops.h5
svm11=SVM11_npp_d20150402_t1054237_e1055479_b17764_c20150402172938840475_noaa_ops.h5
svm12=SVM12_npp_d20150402_t1054237_e1055479_b17764_c20150402172941620436_noaa_ops.h5
svm13=SVM13_npp_d20150402_t1054237_e1055479_b17764_c20150402172946217239_noaa_ops.h5
svm14=SVM14_npp_d20150402_t1054237_e1055479_b17764_c20150402172912285171_noaa_ops.h5
svm15=SVM15_npp_d20150402_t1054237_e1055479_b17764_c20150402172946982983_noaa_ops.h5
svm16=SVM16_npp_d20150402_t1054237_e1055479_b17764_c20150402172944040007_noaa_ops.h5

Example of **JRR_PREPROCESS_SDR.pl.PSF**:

/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b17764/GMTCO_npp_d20150402_t1054237_e1055479_b17764_c20150402171617371175_noaa_ops.nc
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b17764/SVM01_npp_d20150402_t1054237_e1055479_b17764_c20150402172853771802_noaa_ops.nc
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b17764/SVM02_npp_d20150402_t1054237_e1055479_b17764_c20150402172837504244_noaa_ops.nc
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b17764/SVM03_npp_d20150402_t1054237_e1055479_b17764_c20150402172859491592_noaa_ops.nc
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b17764/SVM04_npp_d20150402_t1054237_e1055479_b17764_c20150402172907467365_noaa_ops.nc
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b17764/SVM05_npp_d20150402_t1054237_e1055479_b17764_c20150402172932197325_noaa_ops.nc
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b17764/SVM06_npp_d20150402_t1054237_e1055479_b17764_c20150402172928866557_noaa_ops.nc

/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b17764/SVM07_npp_d20150402_t1054237_e1055479_b17764_c20150402172933775655_noaa_ops.nc
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b17764/SVM08_npp_d20150402_t1054237_e1055479_b17764_c20150402172930996633_noaa_ops.nc
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b17764/SVM09_npp_d20150402_t1054237_e1055479_b17764_c20150402172912259531_noaa_ops.nc
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b17764/SVM10_npp_d20150402_t1054237_e1055479_b17764_c20150402172939082298_noaa_ops.nc
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b17764/SVM11_npp_d20150402_t1054237_e1055479_b17764_c20150402172938840475_noaa_ops.nc
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b17764/SVM12_npp_d20150402_t1054237_e1055479_b17764_c20150402172941620436_noaa_ops.nc
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b17764/SVM13_npp_d20150402_t1054237_e1055479_b17764_c20150402172946217239_noaa_ops.nc
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b17764/SVM14_npp_d20150402_t1054237_e1055479_b17764_c20150402172912285171_noaa_ops.nc
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b17764/SVM15_npp_d20150402_t1054237_e1055479_b17764_c20150402172946982983_noaa_ops.nc
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b17764/SVM16_npp_d20150402_t1054237_e1055479_b17764_c20150402172944040007_noaa_ops.nc
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b17764/GITCO_npp_d20150402_t1054237_e1055479_b17764_c20150402172259067059_noaa_ops.nc
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b17764/SVI01_npp_d20150402_t1054237_e1055479_b17764_c20150402172912367353_noaa_ops.nc
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b17764/SVI02_npp_d20150402_t1054237_e1055479_b17764_c20150402172855756881_noaa_ops.nc
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b17764/SVI03_npp_d20150402_t1054237_e1055479_b17764_c20150402174048712352_noaa_ops.nc
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b17764/SVI04_npp_d20150402_t1054237_e1055479_b17764_c20150402172950556360_noaa_ops.nc
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b17764/SVI05_npp_d20150402_t1054237_e1055479_b17764_c20150402172953919670_noaa_ops.nc

#END-of-PSF

JRR_PRODUCT_CLOUD_MASK.pl.PCF

The JRR_PRODUCT_CLOUD_MASK.pl.PCF file contains the name of JRR product that is generated in the current instance run, including the full path.

working_directory – This is the directory at which the job processing occurs
job_coverage_start -- This is the pole-pass time of the first remapped file in YYYYMMDDHHMMSSS format (It is not used by the driver script but for NDE in searching the remapped file)

job_coverage_end -- This is the pole-pass time of the third remapped file in YYYYMMDDHHMMSSS format (It is not used by the driver script but for NDE in searching the remapped file)
VERSION – JRR version
SCRIPT_FOR_GFS -- a script to read GFS grib/grib2 data (run_wgrib.pl), used by AIT-framework program
PERL_LOC -- The location of the Perl interpreter
OPS_SCRIPT -- The location of the system script
OPS_BIN -- The location of the system executables
PCF_TEMPLATE -- Full path of directory where the templates of AIT-Framework level PCFs are located
CFG_TEMPLATE -- Full path of directory where the templates of AIT-Framework level CFGs are located
DIR_DEFAULT -- Full path of directory where the AIT-Framework level default PCFs are located
DIR_PCF_OVERWRITES -- Full path of directory where the AIT-Framework level overwritten PCFs are located
DIR_CUSTOMPRODUCTLISTS -- Full path of directory where the AIT-Framework level to the custom product lists
CRTM_VIIRS_M_SpcCoeff -- Full path of viirs-m_npp.SpcCoeff.bin
CRTM_VIIRS_M_TauCoeff -- Full path of viirs-m_npp.TauCoeff.bin
CRTM_CloudCoeff -- Full path of CloudCoeff.bin
CRTM_AerosolCoeff -- Full path of AerosolCoeff.bin
CRTM_EmisCoeff -- Full path of EmisCoeff.bin
NPPBAYES_CLDMASK_COEF- Full path of the Bayes Cloud Mask coefficients
NPP_VIIRS_SDR_ANCIL_M -- Full path of npp_viirs_ancil.Mbands.nc
NPP_VIIRS_SDR_ANCIL_I -- Full path of npp_viirs_ancil.Ibands.nc
COAST_MASK_NASA_1KM -- Full path of coast mask coast_mask_1km.nc
SFC_TYPE_AVHRR_1KM -- Full path of surface type gl-latlong-1km-landcover.nc
LAND_MASK_NASA_1KM -- Full path of land mask lw_geo_2001001_v03m.nc
SFC_ELEV_GLOBE_1KM -- Full path of surface elevation GLOBE_1km_digelev.nc
seebor_emiss_01 -- Full path of month 1 seebor_emiss data file: global_emiss_intABI_2005001.nc
seebor_emiss_02 -- Full path of month 2 seebor_emiss data file: global_emiss_intABI_2005032.nc
seebor_emiss_03 -- Full path of month 3 seebor_emiss data file: global_emiss_intABI_2005060.nc
*seebor_emiss_0*4 -- Full path of month 4 seebor_emiss data file: global_emiss_intABI_2005091.nc
seebor_emiss_05 -- Full path of month 5 seebor_emiss data file: global_emiss_intABI_2005121.nc

seebor_emiss_06 -- Full path of month 6 seebor_emiss data file: global_emiss_intABI_2005152.nc
seebor_emiss_07 -- Full path of month 7 seebor_emiss data file: global_emiss_intABI_2005182.nc
*seebor_emiss_0*8 -- Full path of month 8 seebor_emiss data file: global_emiss_intABI_2005213.nc
seebor_emiss_09 -- Full path of month 9 seebor_emiss data file: global_emiss_intABI_2005244.nc
seebor_emiss_10 -- Full path of month 10 seebor_emiss data file: global_emiss_intABI_2005274.nc
seebor_emiss_11 -- Full path of month 11 seebor_emiss data file: global_emiss_intABI_2005305.nc
seebor_emiss_12 -- Full path of month 12 seebor_emiss data file: global_emiss_intABI_2005335.nc
sfc_albedo_01 – Full path of the surface albedo for file 1
sfc_albedo_02 – Full path of the surface albedo for file 2
sfc_albedo_03 – Full path of the surface albedo for file 3
sfc_albedo_04 – Full path of the surface albedo for file 4
sfc_albedo_05 – Full path of the surface albedo for file 5
sfc_albedo_06 – Full path of the surface albedo for file 6
sfc_albedo_07 – Full path of the surface albedo for file 7
sfc_albedo_08 – Full path of the surface albedo for file 8
sfc_albedo_09 – Full path of the surface albedo for file 9
sfc_albedo_10 – Full path of the surface albedo for file 10
sfc_albedo_11 – Full path of the surface albedo for file 11
sfc_albedo_12 – Full path of the surface albedo for file 12
sfc_albedo_13 – Full path of the surface albedo for file 13
sfc_albedo_14 – Full path of the surface albedo for file 14
sfc_albedo_15 – Full path of the surface albedo for file 15
sfc_albedo_16 – Full path of the surface albedo for file 16
sfc_albedo_17 – Full path of the surface albedo for file 17
sfc_albedo_18 – Full path of the surface albedo for file 18
sfc_albedo_19 – Full path of the surface albedo for file 19
sfc_albedo_20 – Full path of the surface albedo for file 20
sfc_albedo_21 – Full path of the surface albedo for file 21
sfc_albedo_22 – Full path of the surface albedo for file 22
sfc_albedo_23 – Full path of the surface albedo for file 23
sfc_albedo_24 – Full path of the surface albedo for file 24
sfc_albedo_25 – Full path of the surface albedo for file 25
sfc_albedo_26 – Full path of the surface albedo for file 26
sfc_albedo_27 – Full path of the surface albedo for file 27
sfc_albedo_28 – Full path of the surface albedo for file 28

sfc_albedo_29 – Full path of the surface albedo for file 29
sfc_albedo_30 – Full path of the surface albedo for file 30
sfc_albedo_31 – Full path of the surface albedo for file 31
sfc_albedo_32 – Full path of the surface albedo for file 32
sfc_albedo_33 – Full path of the surface albedo for file 33
sfc_albedo_34 – Full path of the surface albedo for file 34
sfc_albedo_35 – Full path of the surface albedo for file 35
sfc_albedo_36 – Full path of the surface albedo for file 36
sfc_albedo_37 – Full path of the surface albedo for file 37
sfc_albedo_38 – Full path of the surface albedo for file 38
sfc_albedo_39 – Full path of the surface albedo for file 39
sfc_albedo_40 – Full path of the surface albedo for file 40
sfc_albedo_41 – Full path of the surface albedo for file 41
sfc_albedo_42 – Full path of the surface albedo for file 42
sfc_albedo_43 – Full path of the surface albedo for file 43
sfc_albedo_44 – Full path of the surface albedo for file 44
sfc_albedo_45 – Full path of the surface albedo for file 45
sfc_albedo_46 – Full path of the surface albedo for file 46
sfc_albedo_47 – Full path of the surface albedo for file 47
sfc_albedo_48 – Full path of the surface albedo for file 48
sfc_albedo_49 – Full path of the surface albedo for file 49
sfc_albedo_50 – Full path of the surface albedo for file 50
sfc_albedo_51 – Full path of the surface albedo for file 51
sfc_albedo_52 – Full path of the surface albedo for file 52
sfc_albedo_53 – Full path of the surface albedo for file 53
sfc_albedo_54 – Full path of the surface albedo for file 54
sfc_albedo_55 – Full path of the surface albedo for file 55
sfc_albedo_56 – Full path of the surface albedo for file 56
sfc_albedo_57 – Full path of the surface albedo for file 57
sfc_albedo_58 – Full path of the surface albedo for file 58
sfc_albedo_59 – Full path of the surface albedo for file 59
sfc_albedo_60 – Full path of the surface albedo for file 60
sfc_albedo_61 – Full path of the surface albedo for file 61
sfc_albedo_62 – Full path of the surface albedo for file 62
sfc_albedo_63 – Full path of the surface albedo for file 63
sfc_albedo_64 – Full path of the surface albedo for file 64
sfc_albedo_65 – Full path of the surface albedo for file 65
sfc_albedo_66 – Full path of the surface albedo for file 66
sfc_albedo_67 – Full path of the surface albedo for file 67
sfc_albedo_68 – Full path of the surface albedo for file 68
sfc_albedo_69 – Full path of the surface albedo for file 69
gfs_file – name of the NWP GFS file

oisst_file – name of the SST file
gitco –name of the VIIRS GITCO file
gmtco – name of the VIIRS GMTCO file
svi01 – name of the VIIRS SVI file channel 1
svi04 – name of the VIIRS SVI file channel 4
svi05 – name of the VIIRS SVI file channel 5
svm03 – name of the VIIRS SVM file channel 3
svm05 – name of the VIIRS SVM file channel 5
svm07 – name of the VIIRS SVM file channel 7
svm09 – name of the VIIRS SVM file channel 9
svm10 – name of the VIIRS SVM file channel 10
svm11 – name of the VIIRS SVM file channel 11
svm12 – name of the VIIRS SVM file channel 12
svm14 – name of the VIIRS SVM file channel 14
svm15 – name of the VIIRS SVM file channel 15
svm16 – name of the VIIRS SVM file channel 16

Below is an example of **JRR_PRODUCT_CLOUD_MASK.pl.PCF**:

```
#
# name:
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b17764/JR
R_PRODUCT_CLOUD_MASK.pl.PCF
#
# working directory:
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b17764
# ProdRuleName: VIIRS granule (CloudMask)

working_directory=/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e10
55479_b17764
job_coverage_start=201504021054237
job_coverage_end=201504021055479

VERSION=v1r1

SCRIPT_FOR_GFS=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-OPS/scripts/run_wgrib.pl
PERL_LOC=/usr/bin/perl
OPS_SCRIPT=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-OPS/scripts
OPS_BIN=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-OPS/bin
PCF_TEMPLATE=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-
OPS/templates/NPP_VIIRS_SDR_Template.pcf
CFG_TEMPLATE=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-
OPS/templates/NPP_VIIRS_CLOUD_MASK_Template.cfg
DIR_DEFAULT_PCF=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-OPS/Default_PCF
```

DIR_PCF_OVERWRITES=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-OPS/PCF_Overwrites
DIR_CUSTOMPRODUCTLISTS=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-OPS/CustomProductLists

CRTM_VIIRS_M_SpcCoeff=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/crtm_2.0.2/CRTM_Coefficients/Big_Endian/viirs-m_npp.SpcCoeff.bin
CRTM_VIIRS_M_TauCoeff=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/crtm_2.0.2/CRTM_Coefficients/Big_Endian/viirs-m_npp.TauCoeff.bin
CRTM_CloudCoeff=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/crtm_2.0.2/CRTM_Coefficients/Big_Endian/CloudCoeff.bin
CRTM_AerosolCoeff=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/crtm_2.0.2/CRTM_Coefficients/Big_Endian/AerosolCoeff.bin
CRTM_EmisCoeff=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/crtm_2.0.2/CRTM_Coefficients/Big_Endian/EmisCoeff.bin

NPPBAYES_CLDMASK_COEF=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_BAYES_CldMask/viirs_default_nb_cloud_mask_lut.nc

NPP_VIIRS_SDR_ANCIL_M=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/ancil_file/npp_viirs_ancil.Mbands.nc
NPP_VIIRS_SDR_ANCIL_I=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/ancil_file/npp_viirs_ancil.Ibands.nc

COAST_MASK_NASA_1KM=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/masks/coast_mask_1km.nc
SFC_TYPE_AVHRR_1KM=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/masks/gl-latlong-1km-landcover.nc
LAND_MASK_NASA_1KM=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/masks/lw_geo_2001001_v03m.nc
SFC_ELEV_GLOBE_1KM=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_elevation/GLOBE_1km_digelev.nc

seebor_emiss_01=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005001.nc
seebor_emiss_02=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005032.nc
seebor_emiss_03=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005060.nc
seebor_emiss_04=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005091.nc
seebor_emiss_05=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005121.nc
seebor_emiss_06=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005152.nc

seebor_emiss_07=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005182.nc
seebor_emiss_08=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005213.nc
seebor_emiss_09=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005244.nc
seebor_emiss_10=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005274.nc
seebor_emiss_11=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005305.nc
seebor_emiss_12=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005335.nc

sfc_albedo_01=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.001.0.659_x4.nc
sfc_albedo_02=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.001.1.64_x4.nc
sfc_albedo_03=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.001.2.13_x4.nc
sfc_albedo_04=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.017.0.659_x4.nc
sfc_albedo_05=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.017.1.64_x4.nc
sfc_albedo_06=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.017.2.13_x4.nc
sfc_albedo_07=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.033.0.659_x4.nc
sfc_albedo_08=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.033.1.64_x4.nc
sfc_albedo_09=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.033.2.13_x4.nc
sfc_albedo_10=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.049.0.659_x4.nc
sfc_albedo_11=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.049.1.64_x4.nc
sfc_albedo_12=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.049.2.13_x4.nc
sfc_albedo_13=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.065.0.659_x4.nc
sfc_albedo_14=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.065.1.64_x4.nc
sfc_albedo_15=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.065.2.13_x4.nc
sfc_albedo_16=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.081.0.659_x4.nc
sfc_albedo_17=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary

/sfc_albedo/AlbMap.WS.c004.v2.0.2004.081.1.64_x4.nc
sfc_albedo_18=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.081.2.13_x4.nc
sfc_albedo_19=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.097.0.659_x4.nc
sfc_albedo_20=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.097.1.64_x4.nc
sfc_albedo_21=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.097.2.13_x4.nc
sfc_albedo_22=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.113.0.659_x4.nc
sfc_albedo_23=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.113.1.64_x4.nc
sfc_albedo_24=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.113.2.13_x4.nc
sfc_albedo_25=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.129.0.659_x4.nc
sfc_albedo_26=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.129.1.64_x4.nc
sfc_albedo_27=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.129.2.13_x4.nc
sfc_albedo_28=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.145.0.659_x4.nc
sfc_albedo_29=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.145.1.64_x4.nc
sfc_albedo_30=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.145.2.13_x4.nc
sfc_albedo_31=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.161.0.659_x4.nc
sfc_albedo_32=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.161.1.64_x4.nc
sfc_albedo_33=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.161.2.13_x4.nc
sfc_albedo_34=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.177.0.659_x4.nc
sfc_albedo_35=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.177.1.64_x4.nc
sfc_albedo_36=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.177.2.13_x4.nc
sfc_albedo_37=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.193.0.659_x4.nc
sfc_albedo_38=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.193.1.64_x4.nc
sfc_albedo_39=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.193.2.13_x4.nc
sfc_albedo_40=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary

/sfc_albedo/AlbMap.WS.c004.v2.0.2004.209.0.659_x4.nc
sfc_albedo_41=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.209.1.64_x4.nc
sfc_albedo_42=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.209.2.13_x4.nc
sfc_albedo_43=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.225.0.659_x4.nc
sfc_albedo_44=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.225.1.64_x4.nc
sfc_albedo_45=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.225.2.13_x4.nc
sfc_albedo_46=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.241.0.659_x4.nc
sfc_albedo_47=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.241.1.64_x4.nc
sfc_albedo_48=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.241.2.13_x4.nc
sfc_albedo_49=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.257.0.659_x4.nc
sfc_albedo_50=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.257.1.64_x4.nc
sfc_albedo_51=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.257.2.13_x4.nc
sfc_albedo_52=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.273.0.659_x4.nc
sfc_albedo_53=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.273.1.64_x4.nc
sfc_albedo_54=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.273.2.13_x4.nc
sfc_albedo_55=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.289.0.659_x4.nc
sfc_albedo_56=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.289.1.64_x4.nc
sfc_albedo_57=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.289.2.13_x4.nc
sfc_albedo_58=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.305.0.659_x4.nc
sfc_albedo_59=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.305.1.64_x4.nc
sfc_albedo_60=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.305.2.13_x4.nc
sfc_albedo_61=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.321.0.659_x4.nc
sfc_albedo_62=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.321.1.64_x4.nc
sfc_albedo_63=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary

/sfc_albedo/AlbMap.WS.c004.v2.0.2004.321.2.13_x4.nc
sfc_albedo_64=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.337.0.659_x4.nc
sfc_albedo_65=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.337.1.64_x4.nc
sfc_albedo_66=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.337.2.13_x4.nc
sfc_albedo_67=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.353.0.659_x4.nc
sfc_albedo_68=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.353.1.64_x4.nc
sfc_albedo_69=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.353.2.13_x4.nc

gfs_file=gfs.t06z.pgrb2.0p50.f003.20150402
gfs_file=gfs.t06z.pgrb2.0p50.f006.20150402
oisst_file=avhrr-only-v2.20150401_preliminary.nc

gmtco=GMTCO_npp_d20150402_t1054237_e1055479_b17764_c20150402171617371175_noa
a_ops.nc
svm03=SVM03_npp_d20150402_t1054237_e1055479_b17764_c20150402172859491592_noaa
_ops.nc
svm05=SVM05_npp_d20150402_t1054237_e1055479_b17764_c20150402172932197325_noaa
_ops.nc
svm07=SVM07_npp_d20150402_t1054237_e1055479_b17764_c20150402172933775655_noaa
_ops.nc
svm09=SVM09_npp_d20150402_t1054237_e1055479_b17764_c20150402172912259531_noaa
_ops.nc
svm10=SVM10_npp_d20150402_t1054237_e1055479_b17764_c20150402172939082298_noaa
_ops.nc
svm11=SVM11_npp_d20150402_t1054237_e1055479_b17764_c20150402172938840475_noaa
_ops.nc
svm12=SVM12_npp_d20150402_t1054237_e1055479_b17764_c20150402172941620436_noaa
_ops.nc
svm14=SVM14_npp_d20150402_t1054237_e1055479_b17764_c20150402172912285171_noaa
_ops.nc
svm15=SVM15_npp_d20150402_t1054237_e1055479_b17764_c20150402172946982983_noaa
_ops.nc
svm16=SVM16_npp_d20150402_t1054237_e1055479_b17764_c20150402172944040007_noaa
_ops.nc

#END-of-PCF

Example of **JRR_PRODUCT_CLOUD_MASK.pl.PSF**:

/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b17764/JRR-CloudMask_v1r1_npp_s201504021054237_e201504021055479_c201606291430530.nc

#END-of-PSF

JRR_PRODUCT_CLOUD_CLOUDS.pl.PCF (with DCOMP)

working_directory – This is the directory at which the job processing occurs
job_coverage_start -- This is the pole-pass time of the first remapped file in YYYYMMDDHHMMSSS format (It is not used by the driver script but for NDE in searching the remapped file)
job_coverage_end -- This is the pole-pass time of the third remapped file in YYYYMMDDHHMMSSS format (It is not used by the driver script but for NDE in searching the remapped file)
VERSION – JRR version
SCRIPT_FOR_GFS -- a script to read GFS grib/grib2 data (run_wgrib.pl), used by AIT-framework program
PERL_LOC -- The location of the Perl interpreter
OPS_SCRIPT -- The location of the system script
OPS_BIN -- The location of the system executables
PCF_TEMPLATE -- Full path of directory where the templates of AIT-Framework level PCFs are located
CFG_TEMPLATE -- Full path of directory where the templates of AIT-Framework level CFGs are located
DIR_DEFAULT -- Full path of directory where the AIT-Framework level default PCFs are located
DIR_PCF_OVERWRITES -- Full path of directory where the AIT-Framework level overwritten PCFs are located
DIR_CUSTOMPRODUCTLISTS -- Full path of directory where the AIT-Framework level to the custom product lists
CRTM_VIIRS_M_SpcCoeff -- Full path of viirs-m_npp.SpcCoeff.bin
CRTM_VIIRS_M_TauCoeff -- Full path of viirs-m_npp.TauCoeff.bin
CRTM_CloudCoeff -- Full path of CloudCoeff.bin
CRTM_AerosolCoeff -- Full path of AerosolCoeff.bin
CRTM_EmisCoeff -- Full path of EmisCoeff.bin
NPP_VIIRS_CLD_NCOMP_COEF – Full directory path of the VIIRS NCOMP coefficients
NPP_VIIRS_CLD_DCOMP_COEF_01 – Full directory path of the ref lookup table of ice/cloud of VIIRS DCOMP coefficients for channel 5
NPP_VIIRS_CLD_DCOMP_COEF_02 – Full directory path of the ref lookup table of water/cloud of VIIRS DCOMP coefficients for channel 5

NPP_VIIRS_CLD_DCOMP_COEF_03 – Full directory path of the ref lookup table of ice/cloud of VIIRS DCOMP coefficients for channel 10
NPP_VIIRS_CLD_DCOMP_COEF_04 – Full directory path of the ref lookup table of water/cloud of VIIRS DCOMP coefficients for channel 10
NPP_VIIRS_CLD_DCOMP_COEF_05 – Full directory path of the ref lookup table of ice/cloud of VIIRS DCOMP coefficients for channel 11
NPP_VIIRS_CLD_DCOMP_COEF_06 – Full directory path of the ref lookup table of water/cloud of VIIRS DCOMP coefficients for channel 11
NPP_VIIRS_CLD_DCOMP_COEF_07 – Full directory path of the ems lookup table of ice/cloud of VIIRS DCOMP coefficients for channel 12
NPP_VIIRS_CLD_DCOMP_COEF_08 – Full directory path of the ems lookup table of water/cloud of VIIRS DCOMP coefficients for channel 12
NPP_VIIRS_CLD_DCOMP_COEF_09 – Full directory path of the ref lookup table of ice/cloud of VIIRS DCOMP coefficients for channel 12
NPP_VIIRS_CLD_DCOMP_COEF_10 – Full directory path of the ref lookup table of water/cloud of VIIRS DCOMP coefficients for channel 12
NPP_VIIRS_SDR_ANCIL_M -- Full path of npp_viirs_ancil.Mbands.nc
NPP_VIIRS_SDR_ANCIL_I -- Full path of npp_viirs_ancil.Ibands.nc
COAST_MASK_NASA_1KM -- Full path of coast mask coast_mask_1km.nc
SFC_TYPE_AVHRR_1KM -- Full path of surface type gl-latlong-1km-landcover.nc
LAND_MASK_NASA_1KM -- Full path of land mask lw_geo_2001001_v03m.nc
SFC_ELEV_GLOBE_1KM -- Full path of surface elevation GLOBE_1km_digelev.nc
seebor_emiss_01 -- Full path of month 1 seebor_emiss data file: global_emiss_intABI_2005001.nc
seebor_emiss_02 -- Full path of month 2 seebor_emiss data file: global_emiss_intABI_2005032.nc
seebor_emiss_03 -- Full path of month 3 seebor_emiss data file: global_emiss_intABI_2005060.nc
*seebor_emiss_0*4 -- Full path of month 4 seebor_emiss data file: global_emiss_intABI_2005091.nc
seebor_emiss_05 -- Full path of month 5 seebor_emiss data file: global_emiss_intABI_2005121.nc
seebor_emiss_06 -- Full path of month 6 seebor_emiss data file: global_emiss_intABI_2005152.nc
seebor_emiss_07 -- Full path of month 7 seebor_emiss data file: global_emiss_intABI_2005182.nc
*seebor_emiss_0*8 -- Full path of month 8 seebor_emiss data file: global_emiss_intABI_2005213.nc
seebor_emiss_09 -- Full path of month 9 seebor_emiss data file: global_emiss_intABI_2005244.nc
seebor_emiss_10 -- Full path of month 10 seebor_emiss data file: global_emiss_intABI_2005274.nc

seebor_emiss_11 -- Full path of month 11 seebor_emiss data file: global_emiss_intABI_2005305.nc
seebor_emiss_12 -- Full path of month 12 seebor_emiss data file: global_emiss_intABI_2005335.nc
sfc_albedo_01 – Full path of the surface albedo for file 1
sfc_albedo_02 – Full path of the surface albedo for file 2
sfc_albedo_03 – Full path of the surface albedo for file 3
sfc_albedo_04 – Full path of the surface albedo for file 4
sfc_albedo_05 – Full path of the surface albedo for file 5
sfc_albedo_06 – Full path of the surface albedo for file 6
sfc_albedo_07 – Full path of the surface albedo for file 7
sfc_albedo_08 – Full path of the surface albedo for file 8
sfc_albedo_09 – Full path of the surface albedo for file 9
sfc_albedo_10 – Full path of the surface albedo for file 10
sfc_albedo_11 – Full path of the surface albedo for file 11
sfc_albedo_12 – Full path of the surface albedo for file 12
sfc_albedo_13 – Full path of the surface albedo for file 13
sfc_albedo_14 – Full path of the surface albedo for file 14
sfc_albedo_15 – Full path of the surface albedo for file 15
sfc_albedo_16 – Full path of the surface albedo for file 16
sfc_albedo_17 – Full path of the surface albedo for file 17
sfc_albedo_18 – Full path of the surface albedo for file 18
sfc_albedo_19 – Full path of the surface albedo for file 19
sfc_albedo_20 – Full path of the surface albedo for file 20
sfc_albedo_21 – Full path of the surface albedo for file 21
sfc_albedo_22 – Full path of the surface albedo for file 22
sfc_albedo_23 – Full path of the surface albedo for file 23
sfc_albedo_24 – Full path of the surface albedo for file 24
sfc_albedo_25 – Full path of the surface albedo for file 25
sfc_albedo_26 – Full path of the surface albedo for file 26
sfc_albedo_27 – Full path of the surface albedo for file 27
sfc_albedo_28 – Full path of the surface albedo for file 28
sfc_albedo_29 – Full path of the surface albedo for file 29
sfc_albedo_30 – Full path of the surface albedo for file 30
sfc_albedo_31 – Full path of the surface albedo for file 31
sfc_albedo_32 – Full path of the surface albedo for file 32
sfc_albedo_33 – Full path of the surface albedo for file 33
sfc_albedo_34 – Full path of the surface albedo for file 34
sfc_albedo_35 – Full path of the surface albedo for file 35
sfc_albedo_36 – Full path of the surface albedo for file 36
sfc_albedo_37 – Full path of the surface albedo for file 37
sfc_albedo_38 – Full path of the surface albedo for file 38

sfc_albedo_39 – Full path of the surface albedo for file 39
sfc_albedo_40 – Full path of the surface albedo for file 40
sfc_albedo_41 – Full path of the surface albedo for file 41
sfc_albedo_42 – Full path of the surface albedo for file 42
sfc_albedo_43 – Full path of the surface albedo for file 43
sfc_albedo_44 – Full path of the surface albedo for file 44
sfc_albedo_45 – Full path of the surface albedo for file 45
sfc_albedo_46 – Full path of the surface albedo for file 46
sfc_albedo_47 – Full path of the surface albedo for file 47
sfc_albedo_48 – Full path of the surface albedo for file 48
sfc_albedo_49 – Full path of the surface albedo for file 49
sfc_albedo_50 – Full path of the surface albedo for file 50
sfc_albedo_51 – Full path of the surface albedo for file 51
sfc_albedo_52 – Full path of the surface albedo for file 52
sfc_albedo_53 – Full path of the surface albedo for file 53
sfc_albedo_54 – Full path of the surface albedo for file 54
sfc_albedo_55 – Full path of the surface albedo for file 55
sfc_albedo_56 – Full path of the surface albedo for file 56
sfc_albedo_57 – Full path of the surface albedo for file 57
sfc_albedo_58 – Full path of the surface albedo for file 58
sfc_albedo_59 – Full path of the surface albedo for file 59
sfc_albedo_60 – Full path of the surface albedo for file 60
sfc_albedo_61 – Full path of the surface albedo for file 61
sfc_albedo_62 – Full path of the surface albedo for file 62
sfc_albedo_63 – Full path of the surface albedo for file 63
sfc_albedo_64 – Full path of the surface albedo for file 64
sfc_albedo_65 – Full path of the surface albedo for file 65
sfc_albedo_66 – Full path of the surface albedo for file 66
sfc_albedo_67 – Full path of the surface albedo for file 67
sfc_albedo_68 – Full path of the surface albedo for file 68
sfc_albedo_69 – Full path of the surface albedo for file 69
cldmask – name of the cloud mask file
gfs_file – name of the NWP GFS file
oisst_file – name of the SST file
gmtco – name of the VIIRS GTMCO file
svm05 – name of the VIIRS SVM file channel 5
svm11 – name of the VIIRS SVM file channel 11
svm12 – name of the VIIRS SVM file channel 12
svm13 – name of the VIIRS SVM file channel 13
svm14 – name of the VIIRS SVM file channel 14
svm15 – name of the VIIRS SVM file channel 15
svm16 – name of the VIIRS SVM file channel 16

Below is an example of a **JRR_PRODUCT_CLOUD_CLOUDS.pl.PCF** file (with DCOMP):

```
#
# name:
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b17764/JR
R_PRODUCT_CLOUD_CLOUDS.pl.PCF
#
# working directory:
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b17764
# ProdRuleName: VIIRS-CloudsProuducts granule (CldPhase/CldHeight/CldNCOMP(night
granule)/CldDCOMP(daytime granule))

working_directory=/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e10
55479_b17764

job_coverage_start=201504021054237
job_coverage_end=201504021055479

VERSION=v1r1

SCRIPT_FOR_GFS=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-OPS/scripts/run_wgrib.pl
PERL_LOC=/usr/bin/perl
OPS_SCRIPT=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-OPS/scripts
OPS_BIN=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-OPS/bin
PCF_TEMPLATE=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-
OPS/templates/NPP_VIIRS_SDR_Template.pcf
CFG_TEMPLATE=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-
OPS/templates/NPP_VIIRS_CLOUD_CLOUDS_Template.cfg
DIR_DEFAULT_PCF=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-OPS/Default_PCF
DIR_PCF_OVERWRITES=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-
OPS/PCF_Overwrites
DIR_CUSTOMPRODUCTLISTS=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-
OPS/CustomProductLists

CRTM_VIIRS_M_SpcCoeff=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorith
m_ancillary/crtm_2.0.2/CRTM_Coefficients/Big_Endian/viirs-m_npp.SpcCoeff.bin
CRTM_VIIRS_M_TauCoeff=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorith
m_ancillary/crtm_2.0.2/CRTM_Coefficients/Big_Endian/viirs-m_npp.TauCoeff.bin
CRTM_CloudCoeff=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancill
ary/crtm_2.0.2/CRTM_Coefficients/Big_Endian/CloudCoeff.bin
CRTM_AerosolCoeff=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_anc
illary/crtm_2.0.2/CRTM_Coefficients/Big_Endian/AerosolCoeff.bin
CRTM_EmisCoeff=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancill
```

ary/crtm_2.0.2/CRTM_Coefficients/Big_Endian/EmisCoeff.bin

NPP_VIIRS_CLD_NCOMP_COEF=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/AWG_Cloud_Micro_Night/VIIRS_coefs.nc

NPP_VIIRS_CLD_DCOMP_COEF_01=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/AWG_Cloud_Micro_Day/VIIRS_ch5_ref_lut_ice_cld.nc
NPP_VIIRS_CLD_DCOMP_COEF_02=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/AWG_Cloud_Micro_Day/VIIRS_ch5_ref_lut_wat_cld.nc
NPP_VIIRS_CLD_DCOMP_COEF_03=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/AWG_Cloud_Micro_Day/VIIRS_ch10_ref_lut_ice_cld.nc
NPP_VIIRS_CLD_DCOMP_COEF_04=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/AWG_Cloud_Micro_Day/VIIRS_ch10_ref_lut_wat_cld.nc
NPP_VIIRS_CLD_DCOMP_COEF_05=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/AWG_Cloud_Micro_Day/VIIRS_ch11_ref_lut_ice_cld.nc
NPP_VIIRS_CLD_DCOMP_COEF_06=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/AWG_Cloud_Micro_Day/VIIRS_ch11_ref_lut_wat_cld.nc
NPP_VIIRS_CLD_DCOMP_COEF_07=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/AWG_Cloud_Micro_Day/VIIRS_ch12_ems_lut_ice_cld.nc
NPP_VIIRS_CLD_DCOMP_COEF_08=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/AWG_Cloud_Micro_Day/VIIRS_ch12_ems_lut_wat_cld.nc
NPP_VIIRS_CLD_DCOMP_COEF_09=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/AWG_Cloud_Micro_Day/VIIRS_ch12_ref_lut_ice_cld.nc
NPP_VIIRS_CLD_DCOMP_COEF_10=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/AWG_Cloud_Micro_Day/VIIRS_ch12_ref_lut_wat_cld.nc

NPP_VIIRS_SDR_ANCIL_M=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/ancil_file/npp_viirs_ancil.Mbands.nc
NPP_VIIRS_SDR_ANCIL_I=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/ancil_file/npp_viirs_ancil.Ibands.nc

COAST_MASK_NASA_1KM=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/masks/coast_mask_1km.nc
SFC_TYPE_AVHRR_1KM=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/masks/gl-latlong-1km-landcover.nc
LAND_MASK_NASA_1KM=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/masks/lw_geo_2001001_v03m.nc
SFC_ELEV_GLOBE_1KM=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_elevation/GLOBE_1km_digelev.nc

seebor_emiss_01=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005001.nc
seebor_emiss_02=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005032.nc
seebor_emiss_03=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005060.nc

seebor_emiss_04=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005091.nc
seebor_emiss_05=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005121.nc
seebor_emiss_06=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005152.nc
seebor_emiss_07=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005182.nc
seebor_emiss_08=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005213.nc
seebor_emiss_09=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005244.nc
seebor_emiss_10=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005274.nc
seebor_emiss_11=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005305.nc
seebor_emiss_12=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005335.nc

sfc_albedo_01=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.001.0.659_x4.nc
sfc_albedo_02=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.001.1.64_x4.nc
sfc_albedo_03=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.001.2.13_x4.nc
sfc_albedo_04=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.017.0.659_x4.nc
sfc_albedo_05=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.017.1.64_x4.nc
sfc_albedo_06=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.017.2.13_x4.nc
sfc_albedo_07=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.033.0.659_x4.nc
sfc_albedo_08=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.033.1.64_x4.nc
sfc_albedo_09=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.033.2.13_x4.nc
sfc_albedo_10=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.049.0.659_x4.nc
sfc_albedo_11=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.049.1.64_x4.nc
sfc_albedo_12=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.049.2.13_x4.nc
sfc_albedo_13=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.065.0.659_x4.nc
sfc_albedo_14=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary

/sfc_albedo/AlbMap.WS.c004.v2.0.2004.065.1.64_x4.nc
sfc_albedo_15=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.065.2.13_x4.nc
sfc_albedo_16=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.081.0.659_x4.nc
sfc_albedo_17=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.081.1.64_x4.nc
sfc_albedo_18=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.081.2.13_x4.nc
sfc_albedo_19=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.097.0.659_x4.nc
sfc_albedo_20=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.097.1.64_x4.nc
sfc_albedo_21=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.097.2.13_x4.nc
sfc_albedo_22=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.113.0.659_x4.nc
sfc_albedo_23=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.113.1.64_x4.nc
sfc_albedo_24=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.113.2.13_x4.nc
sfc_albedo_25=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.129.0.659_x4.nc
sfc_albedo_26=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.129.1.64_x4.nc
sfc_albedo_27=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.129.2.13_x4.nc
sfc_albedo_28=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.145.0.659_x4.nc
sfc_albedo_29=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.145.1.64_x4.nc
sfc_albedo_30=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.145.2.13_x4.nc
sfc_albedo_31=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.161.0.659_x4.nc
sfc_albedo_32=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.161.1.64_x4.nc
sfc_albedo_33=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.161.2.13_x4.nc
sfc_albedo_34=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.177.0.659_x4.nc
sfc_albedo_35=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.177.1.64_x4.nc
sfc_albedo_36=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.177.2.13_x4.nc
sfc_albedo_37=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary

/sfc_albedo/AlbMap.WS.c004.v2.0.2004.193.0.659_x4.nc
sfc_albedo_38=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.193.1.64_x4.nc
sfc_albedo_39=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.193.2.13_x4.nc
sfc_albedo_40=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.209.0.659_x4.nc
sfc_albedo_41=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.209.1.64_x4.nc
sfc_albedo_42=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.209.2.13_x4.nc
sfc_albedo_43=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.225.0.659_x4.nc
sfc_albedo_44=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.225.1.64_x4.nc
sfc_albedo_45=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.225.2.13_x4.nc
sfc_albedo_46=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.241.0.659_x4.nc
sfc_albedo_47=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.241.1.64_x4.nc
sfc_albedo_48=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.241.2.13_x4.nc
sfc_albedo_49=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.257.0.659_x4.nc
sfc_albedo_50=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.257.1.64_x4.nc
sfc_albedo_51=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.257.2.13_x4.nc
sfc_albedo_52=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.273.0.659_x4.nc
sfc_albedo_53=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.273.1.64_x4.nc
sfc_albedo_54=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.273.2.13_x4.nc
sfc_albedo_55=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.289.0.659_x4.nc
sfc_albedo_56=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.289.1.64_x4.nc
sfc_albedo_57=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.289.2.13_x4.nc
sfc_albedo_58=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.305.0.659_x4.nc
sfc_albedo_59=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.305.1.64_x4.nc
sfc_albedo_60=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary

/sfc_albedo/AlbMap.WS.c004.v2.0.2004.305.2.13_x4.nc
sfc_albedo_61=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.321.0.659_x4.nc
sfc_albedo_62=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.321.1.64_x4.nc
sfc_albedo_63=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.321.2.13_x4.nc
sfc_albedo_64=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.337.0.659_x4.nc
sfc_albedo_65=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.337.1.64_x4.nc
sfc_albedo_66=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.337.2.13_x4.nc
sfc_albedo_67=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.353.0.659_x4.nc
sfc_albedo_68=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.353.1.64_x4.nc
sfc_albedo_69=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary
/sfc_albedo/AlbMap.WS.c004.v2.0.2004.353.2.13_x4.nc

cldmask=JRR-
CloudMask_v1r1_npp_s201504021054237_e201504021055479_c201606291430530.nc

gfs_file=gfs.t06z.pgrb2.0p50.f003.20150402
gfs_file=gfs.t06z.pgrb2.0p50.f006.20150402
oisst_file=avhrr-only-v2.20150401_preliminary.nc

gmtco=GMTCO_npp_d20150402_t1054237_e1055479_b17764_c20150402171617371175_noa
a_ops.nc
svm05=SVM05_npp_d20150402_t1054237_e1055479_b17764_c20150402172932197325_noaa
_ops.nc
svm11=SVM11_npp_d20150402_t1054237_e1055479_b17764_c20150402172938840475_noaa
_ops.nc
svm14=SVM14_npp_d20150402_t1054237_e1055479_b17764_c20150402172912285171_noaa
_ops.nc
svm15=SVM15_npp_d20150402_t1054237_e1055479_b17764_c20150402172946982983_noaa
_ops.nc
svm16=SVM16_npp_d20150402_t1054237_e1055479_b17764_c20150402172944040007_noaa
_ops.nc

#END-of-PCF

Example of **JRR_PRODUCT_CLOUD_CLOUDS.pl.PSF** (with DCOMP):

/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b17764/JR

R-CloudPhase_v1r1_npp_s201504021054237_e201504021055479_c201606291440560.nc
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b17764/JR
R-CloudHeight_v1r1_npp_s201504021054237_e201504021055479_c201606291440570.nc
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b17764/JR
R-CloudDCOMP_v1r1_npp_s201504021054237_e201504021055479_c201606291441330.nc

#END-of-PSF

JRR_PRODUCT_CLOUD_CLOUDS.pl.PCF (with NCOMP)

working_directory – This is the directory at which the job processing occurs
job_coverage_start -- This is the pole-pass time of the first remapped file in YYYYMMDDHHMMSSS format (It is not used by the driver script but for NDE in searching the remapped file)
job_coverage_end -- This is the pole-pass time of the third remapped file in YYYYMMDDHHMMSSS format (It is not used by the driver script but for NDE in searching the remapped file)
VERSION – JRR version
SCRIPT_FOR_GFS -- a script to read GFS grib/grib2 data (run_wgrib.pl), used by AIT-framework program
PERL_LOC -- The location of the Perl interpreter
OPS_SCRIPT -- The location of the system script
OPS_BIN -- The location of the system executables
PCF_TEMPLATE -- Full path of directory where the templates of AIT-Framework level PCFs are located
CFG_TEMPLATE -- Full path of directory where the templates of AIT-Framework level CFGs are located
DIR_DEFAULT -- Full path of directory where the AIT-Framework level default PCFs are located
DIR_PCF_OVERWRITES -- Full path of directory where the AIT-Framework level overwritten PCFs are located
DIR_CUSTOMPRODUCTLISTS -- Full path of directory where the AIT-Framework level to the custom product lists
CRTM_VIIRS_M_SpcCoeff -- Full path of viirs-m_npp.SpcCoeff.bin
CRTM_VIIRS_M_TauCoeff -- Full path of viirs-m_npp.TauCoeff.bin
CRTM_CloudCoeff -- Full path of CloudCoeff.bin
CRTM_AerosolCoeff -- Full path of AerosolCoeff.bin
CRTM_EmisCoeff -- Full path of EmisCoeff.bin
NPP_VIIRS_CLD_NCOMP_COEF – Full directory path of the VIIRS NCOMP coefficients
NPP_VIIRS_CLD_DCOMP_COEF_01 – Full directory path of the ref lookup table of ice/cloud of VIIRS DCOMP coefficients for channel 5

NPP_VIIRS_CLD_DCOMP_COEF_02 – Full directory path of the ref lookup table of water/cloud of VIIRS DCOMP coefficients for channel 5
NPP_VIIRS_CLD_DCOMP_COEF_03 – Full directory path of the ref lookup table of ice/cloud of VIIRS DCOMP coefficients for channel 10
NPP_VIIRS_CLD_DCOMP_COEF_04 – Full directory path of the ref lookup table of water/cloud of VIIRS DCOMP coefficients for channel 10
NPP_VIIRS_CLD_DCOMP_COEF_05 – Full directory path of the ref lookup table of ice/cloud of VIIRS DCOMP coefficients for channel 11
NPP_VIIRS_CLD_DCOMP_COEF_06 – Full directory path of the ref lookup table of water/cloud of VIIRS DCOMP coefficients for channel 11
NPP_VIIRS_CLD_DCOMP_COEF_07 – Full directory path of the ems lookup table of ice/cloud of VIIRS DCOMP coefficients for channel 12
NPP_VIIRS_CLD_DCOMP_COEF_08 – Full directory path of the ems lookup table of water/cloud of VIIRS DCOMP coefficients for channel 12
NPP_VIIRS_CLD_DCOMP_COEF_09 – Full directory path of the ref lookup table of ice/cloud of VIIRS DCOMP coefficients for channel 12
NPP_VIIRS_CLD_DCOMP_COEF_10 – Full directory path of the ref lookup table of water/cloud of VIIRS DCOMP coefficients for channel 12
NPP_VIIRS_SDR_ANCIL_M -- Full path of npp_viirs_ancil.Mbands.nc
NPP_VIIRS_SDR_ANCIL_I -- Full path of npp_viirs_ancil.Ibands.nc
COAST_MASK_NASA_1KM -- Full path of coast mask coast_mask_1km.nc
SFC_TYPE_AVHRR_1KM -- Full path of surface type gl-latlong-1km-landcover.nc
LAND_MASK_NASA_1KM -- Full path of land mask lw_geo_2001001_v03m.nc
SFC_ELEV_GLOBE_1KM -- Full path of surface elevation GLOBE_1km_digelev.nc
seebor_emiss_01 -- Full path of month 1 seebor_emiss data file: global_emiss_intABI_2005001.nc
seebor_emiss_02 -- Full path of month 2 seebor_emiss data file: global_emiss_intABI_2005032.nc
seebor_emiss_03 -- Full path of month 3 seebor_emiss data file: global_emiss_intABI_2005060.nc
*seebor_emiss_0*4 -- Full path of month 4 seebor_emiss data file: global_emiss_intABI_2005091.nc
seebor_emiss_05 -- Full path of month 5 seebor_emiss data file: global_emiss_intABI_2005121.nc
seebor_emiss_06 -- Full path of month 6 seebor_emiss data file: global_emiss_intABI_2005152.nc
seebor_emiss_07 -- Full path of month 7 seebor_emiss data file: global_emiss_intABI_2005182.nc
*seebor_emiss_0*8 -- Full path of month 8 seebor_emiss data file: global_emiss_intABI_2005213.nc
seebor_emiss_09 -- Full path of month 9 seebor_emiss data file: global_emiss_intABI_2005244.nc

seebor_emiss_10 -- Full path of month 10 seebor_emiss data file: global_emiss_intABI_2005274.nc
seebor_emiss_11 -- Full path of month 11 seebor_emiss data file: global_emiss_intABI_2005305.nc
seebor_emiss_12 -- Full path of month 12 seebor_emiss data file: global_emiss_intABI_2005335.nc
sfc_albedo_01 – Full path of the surface albedo for file 1
sfc_albedo_02 – Full path of the surface albedo for file 2
sfc_albedo_03 – Full path of the surface albedo for file 3
sfc_albedo_04 – Full path of the surface albedo for file 4
sfc_albedo_05 – Full path of the surface albedo for file 5
sfc_albedo_06 – Full path of the surface albedo for file 6
sfc_albedo_07 – Full path of the surface albedo for file 7
sfc_albedo_08 – Full path of the surface albedo for file 8
sfc_albedo_09 – Full path of the surface albedo for file 9
sfc_albedo_10 – Full path of the surface albedo for file 10
sfc_albedo_11 – Full path of the surface albedo for file 11
sfc_albedo_12 – Full path of the surface albedo for file 12
sfc_albedo_13 – Full path of the surface albedo for file 13
sfc_albedo_14 – Full path of the surface albedo for file 14
sfc_albedo_15 – Full path of the surface albedo for file 15
sfc_albedo_16 – Full path of the surface albedo for file 16
sfc_albedo_17 – Full path of the surface albedo for file 17
sfc_albedo_18 – Full path of the surface albedo for file 18
sfc_albedo_19 – Full path of the surface albedo for file 19
sfc_albedo_20 – Full path of the surface albedo for file 20
sfc_albedo_21 – Full path of the surface albedo for file 21
sfc_albedo_22 – Full path of the surface albedo for file 22
sfc_albedo_23 – Full path of the surface albedo for file 23
sfc_albedo_24 – Full path of the surface albedo for file 24
sfc_albedo_25 – Full path of the surface albedo for file 25
sfc_albedo_26 – Full path of the surface albedo for file 26
sfc_albedo_27 – Full path of the surface albedo for file 27
sfc_albedo_28 – Full path of the surface albedo for file 28
sfc_albedo_29 – Full path of the surface albedo for file 29
sfc_albedo_30 – Full path of the surface albedo for file 30
sfc_albedo_31 – Full path of the surface albedo for file 31
sfc_albedo_32 – Full path of the surface albedo for file 32
sfc_albedo_33 – Full path of the surface albedo for file 33
sfc_albedo_34 – Full path of the surface albedo for file 34
sfc_albedo_35 – Full path of the surface albedo for file 35
sfc_albedo_36 – Full path of the surface albedo for file 36

sfc_albedo_37 – Full path of the surface albedo for file 37
sfc_albedo_38 – Full path of the surface albedo for file 38
sfc_albedo_39 – Full path of the surface albedo for file 39
sfc_albedo_40 – Full path of the surface albedo for file 40
sfc_albedo_41 – Full path of the surface albedo for file 41
sfc_albedo_42 – Full path of the surface albedo for file 42
sfc_albedo_43 – Full path of the surface albedo for file 43
sfc_albedo_44 – Full path of the surface albedo for file 44
sfc_albedo_45 – Full path of the surface albedo for file 45
sfc_albedo_46 – Full path of the surface albedo for file 46
sfc_albedo_47 – Full path of the surface albedo for file 47
sfc_albedo_48 – Full path of the surface albedo for file 48
sfc_albedo_49 – Full path of the surface albedo for file 49
sfc_albedo_50 – Full path of the surface albedo for file 50
sfc_albedo_51 – Full path of the surface albedo for file 51
sfc_albedo_52 – Full path of the surface albedo for file 52
sfc_albedo_53 – Full path of the surface albedo for file 53
sfc_albedo_54 – Full path of the surface albedo for file 54
sfc_albedo_55 – Full path of the surface albedo for file 55
sfc_albedo_56 – Full path of the surface albedo for file 56
sfc_albedo_57 – Full path of the surface albedo for file 57
sfc_albedo_58 – Full path of the surface albedo for file 58
sfc_albedo_59 – Full path of the surface albedo for file 59
sfc_albedo_60 – Full path of the surface albedo for file 60
sfc_albedo_61 – Full path of the surface albedo for file 61
sfc_albedo_62 – Full path of the surface albedo for file 62
sfc_albedo_63 – Full path of the surface albedo for file 63
sfc_albedo_64 – Full path of the surface albedo for file 64
sfc_albedo_65 – Full path of the surface albedo for file 65
sfc_albedo_66 – Full path of the surface albedo for file 66
sfc_albedo_67 – Full path of the surface albedo for file 67
sfc_albedo_68 – Full path of the surface albedo for file 68
sfc_albedo_69 – Full path of the surface albedo for file 69
cldmask – name of the cloud mask file
gfs_file – name of the NWP GFS file
oisst_file – name of the SST file
gmtco – name of the VIIRS GTMCO file
svm12 – name of the VIIRS SVM file channel 12
svm14 – name of the VIIRS SVM file channel 14
svm15 – name of the VIIRS SVM file channel 15
svm16 – name of the VIIRS SVM file channel 16

Below is an example of a **JRR_PRODUCT_CLOUD_CLOUDS.pl.PCF** file (with NCOMP):

```
#
# name:
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150511_t0919230_e0920
472_b18316/JRR_PRODUCT_CLOUD_CLOUDS.pl.PCF
##
# working directory:
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150511_t0919230_e0920
472_b18316
# ProdRuleName: VIIRS-CloudsProuducts granule
(CldPhase/CldHeight/CldNCOMP(night granule)/CldDCOMP(daytime granule))

working_directory=/data/data241/hxie/dlv/JRR_DAP_201606/working_d2015051
1_t0919230_e0920472_b18316
job_coverage_start=201505110919230
job_coverage_end=201505110920472

VERSION=v1r1

SCRIPT_FOR_GFS=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-
OPS/scripts/run_wgrib.pl
PERL_LOC=/usr/bin/perl
OPS_SCRIPT=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-OPS/scripts
OPS_BIN=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-OPS/bin
PCF_TEMPLATE=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-
OPS/templates/NPP_VIIRS_SDR_Template.pcf
CFG_TEMPLATE=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-
OPS/templates/NPP_VIIRS_CLOUD_CLOUDS_Template.cfg
DIR_DEFAULT_PCF=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-
OPS/Default_PCF
DIR_PCF_OVERWRITES=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-
OPS/PCF_Overwrites
DIR_CUSTOMPRODUCTLISTS=/data/data241/hxie/dlv/JRR_DAP_201606/JRR
PS-OPS/CustomProductLists
```

CRTM_VIIRS_M_SpcCoeff=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/crtm_2.0.2/CRTM_Coefficients/Big_Endian/viirs-m_npp.SpcCoeff.bin
CRTM_VIIRS_M_TauCoeff=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/crtm_2.0.2/CRTM_Coefficients/Big_Endian/viirs-m_npp.TauCoeff.bin
CRTM_CloudCoeff=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/crtm_2.0.2/CRTM_Coefficients/Big_Endian/CloudCoeff.bin
CRTM_AerosolCoeff=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/crtm_2.0.2/CRTM_Coefficients/Big_Endian/AerosolCoeff.bin
CRTM_EmisCoeff=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/crtm_2.0.2/CRTM_Coefficients/Big_Endian/EmisCoeff.bin

NPP_VIIRS_CLD_NCOMP_COEF=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/AWG_Cloud_Micro_Night/VIIRS_coefs.nc

NPP_VIIRS_CLD_DCOMP_COEF_01=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/AWG_Cloud_Micro_Day/VIIRS_ch5_ref_lut_ice_cld.nc
NPP_VIIRS_CLD_DCOMP_COEF_02=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/AWG_Cloud_Micro_Day/VIIRS_ch5_ref_lut_wat_cld.nc
NPP_VIIRS_CLD_DCOMP_COEF_03=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/AWG_Cloud_Micro_Day/VIIRS_ch10_ref_lut_ice_cld.nc
NPP_VIIRS_CLD_DCOMP_COEF_04=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/AWG_Cloud_Micro_Day/VIIRS_ch10_ref_lut_wat_cld.nc
NPP_VIIRS_CLD_DCOMP_COEF_05=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/AWG_Cloud_Micro_Day/VIIRS_ch11_ref_lut_ice_cld.nc
NPP_VIIRS_CLD_DCOMP_COEF_06=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/AWG_Cloud_Micro_Day/VIIRS_ch11_ref_lut_wat_cld.nc
NPP_VIIRS_CLD_DCOMP_COEF_07=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/AWG_Cloud_Micro_Day/VIIRS_ch12_ems_lut_ice_cld.nc

NPP_VIIRS_CLD_DCOMP_COEF_08=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/AWG_Cloud_Micro_Day/VIIRS_ch12_ems_lut_wat_cld.nc
NPP_VIIRS_CLD_DCOMP_COEF_09=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/AWG_Cloud_Micro_Day/VIIRS_ch12_ref_lut_ice_cld.nc
NPP_VIIRS_CLD_DCOMP_COEF_10=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/AWG_Cloud_Micro_Day/VIIRS_ch12_ref_lut_wat_cld.nc

NPP_VIIRS_SDR_ANCIL_M=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/ancil_file/npp_viirs_ancil.Mbands.nc
NPP_VIIRS_SDR_ANCIL_I=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/ancil_file/npp_viirs_ancil.Ibands.nc

COAST_MASK_NASA_1KM=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/masks/coast_mask_1km.nc
SFC_TYPE_AVHRR_1KM=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/masks/gl-latlong-1km-landcover.nc
LAND_MASK_NASA_1KM=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/masks/lw_geo_2001001_v03m.nc
SFC_ELEV_GLOBE_1KM=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_elevation/GLOBE_1km_digelev.nc

seebor_emiss_01=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005001.nc
seebor_emiss_02=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005032.nc
seebor_emiss_03=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005060.nc
seebor_emiss_04=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005091.nc
seebor_emiss_05=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005121.nc
seebor_emiss_06=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005152.nc
seebor_emiss_07=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005182.nc

seebor_emiss_08=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005213.nc
seebor_emiss_09=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005244.nc
seebor_emiss_10=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005274.nc
seebor_emiss_11=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005305.nc
seebor_emiss_12=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005335.nc

sfc_albedo_01=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.001.0.659_x4.nc
sfc_albedo_02=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.001.1.64_x4.nc
sfc_albedo_03=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.001.2.13_x4.nc
sfc_albedo_04=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.017.0.659_x4.nc
sfc_albedo_05=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.017.1.64_x4.nc
sfc_albedo_06=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.017.2.13_x4.nc
sfc_albedo_07=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.033.0.659_x4.nc
sfc_albedo_08=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.033.1.64_x4.nc
sfc_albedo_09=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.033.2.13_x4.nc
sfc_albedo_10=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.049.0.659_x4.nc
sfc_albedo_11=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.049.1.64_x4.nc
sfc_albedo_12=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.049.2.13_x4.nc
sfc_albedo_13=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.065.0.659_x4.nc
sfc_albedo_14=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/fra

mework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.065.1.64_x4.nc
sfc_albedo_15=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/fra
mework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.065.2.13_x4.nc
sfc_albedo_16=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/fra
mework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.081.0.659_x4.nc
sfc_albedo_17=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/fra
mework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.081.1.64_x4.nc
sfc_albedo_18=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/fra
mework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.081.2.13_x4.nc
sfc_albedo_19=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/fra
mework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.097.0.659_x4.nc
sfc_albedo_20=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/fra
mework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.097.1.64_x4.nc
sfc_albedo_21=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/fra
mework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.097.2.13_x4.nc
sfc_albedo_22=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/fra
mework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.113.0.659_x4.nc
sfc_albedo_23=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/fra
mework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.113.1.64_x4.nc
sfc_albedo_24=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/fra
mework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.113.2.13_x4.nc
sfc_albedo_25=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/fra
mework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.129.0.659_x4.nc
sfc_albedo_26=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/fra
mework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.129.1.64_x4.nc
sfc_albedo_27=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/fra
mework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.129.2.13_x4.nc
sfc_albedo_28=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/fra
mework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.145.0.659_x4.nc
sfc_albedo_29=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/fra
mework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.145.1.64_x4.nc
sfc_albedo_30=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/fra
mework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.145.2.13_x4.nc
sfc_albedo_31=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/fra
mework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.161.0.659_x4.nc
sfc_albedo_32=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/fra
mework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.161.1.64_x4.nc
sfc_albedo_33=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/fra

mework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.161.2.13_x4.nc
sfc_albedo_34=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/fra
mework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.177.0.659_x4.nc
sfc_albedo_35=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/fra
mework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.177.1.64_x4.nc
sfc_albedo_36=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/fra
mework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.177.2.13_x4.nc
sfc_albedo_37=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/fra
mework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.193.0.659_x4.nc
sfc_albedo_38=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/fra
mework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.193.1.64_x4.nc
sfc_albedo_39=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/fra
mework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.193.2.13_x4.nc
sfc_albedo_40=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/fra
mework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.209.0.659_x4.nc
sfc_albedo_41=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/fra
mework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.209.1.64_x4.nc
sfc_albedo_42=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/fra
mework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.209.2.13_x4.nc
sfc_albedo_43=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/fra
mework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.225.0.659_x4.nc
sfc_albedo_44=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/fra
mework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.225.1.64_x4.nc
sfc_albedo_45=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/fra
mework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.225.2.13_x4.nc
sfc_albedo_46=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/fra
mework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.241.0.659_x4.nc
sfc_albedo_47=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/fra
mework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.241.1.64_x4.nc
sfc_albedo_48=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/fra
mework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.241.2.13_x4.nc
sfc_albedo_49=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/fra
mework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.257.0.659_x4.nc
sfc_albedo_50=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/fra
mework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.257.1.64_x4.nc
sfc_albedo_51=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/fra
mework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.257.2.13_x4.nc
sfc_albedo_52=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/fra

mework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.273.0.659_x4.nc
sfc_albedo_53=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.273.1.64_x4.nc
sfc_albedo_54=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.273.2.13_x4.nc
sfc_albedo_55=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.289.0.659_x4.nc
sfc_albedo_56=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.289.1.64_x4.nc
sfc_albedo_57=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.289.2.13_x4.nc
sfc_albedo_58=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.305.0.659_x4.nc
sfc_albedo_59=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.305.1.64_x4.nc
sfc_albedo_60=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.305.2.13_x4.nc
sfc_albedo_61=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.321.0.659_x4.nc
sfc_albedo_62=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.321.1.64_x4.nc
sfc_albedo_63=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.321.2.13_x4.nc
sfc_albedo_64=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.337.0.659_x4.nc
sfc_albedo_65=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.337.1.64_x4.nc
sfc_albedo_66=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.337.2.13_x4.nc
sfc_albedo_67=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.353.0.659_x4.nc
sfc_albedo_68=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.353.1.64_x4.nc
sfc_albedo_69=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_albedo/AlbMap.WS.c004.v2.0.2004.353.2.13_x4.nc

cldmask=JRR-CloudMask_v1r1_npp_s201505110919230_e201505110920472_c20160629151

1510.nc

gfs_file=gfs.t06z.pgrb2.0p50.f003.20150511
gfs_file=gfs.t06z.pgrb2.0p50.f006.20150511
oisst_file=avhrr-only-v2.20150510_preliminary.nc

gmtco=GMTCO_npp_d20150511_t0919230_e0920472_b18316_c20150511153534992851_noaa_ops.nc
svm12=SVM12_npp_d20150511_t0919230_e0920472_b18316_c20150511155105440174_noaa_ops.nc
svm14=SVM14_npp_d20150511_t0919230_e0920472_b18316_c20150511154036022067_noaa_ops.nc
svm15=SVM15_npp_d20150511_t0919230_e0920472_b18316_c20150511154036848547_noaa_ops.nc
svm16=SVM16_npp_d20150511_t0919230_e0920472_b18316_c20150511155108100104_noaa_ops.nc

#END-of-PCF

Example of **JRR_PRODUCT_CLOUD_CLOUDS.pl.PSF** (with NCOMP):

/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150511_t0919230_e0920472_b18316/JRR-CloudPhase_v1r1_npp_s201505110919230_e201505110920472_c201606291524020.nc
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150511_t0919230_e0920472_b18316/JRR-CloudHeight_v1r1_npp_s201505110919230_e201505110920472_c201606291524080.nc
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150511_t0919230_e0920472_b18316/JRR-CloudNCOMP_v1r1_npp_s201505110919230_e201505110920472_c201606291524290.nc

#END-of-PSF

JRR_PRODUCT_AEROSOL_AODADP.pl.PCF

working_directory – This is the directory at which the job processing occurs
job_coverage_start -- This is the pole-pass time of the first remapped file in YYYYMMDDHHMMSSS format (It is not used by the driver script but for NDE in searching the remapped file)
job_coverage_end -- This is the pole-pass time of the third remapped file in YYYYMMDDHHMMSSS format (It is not used by the driver script but for NDE in searching the remapped file)
VERSION – JRR version
SCRIPT_FOR_GFS -- a script to read GFS grib/grib2 data (run_wgrib.pl), used by AIT-framework program
PERL_LOC -- The location of the Perl interpreter
OPS_SCRIPT -- The location of the system script
OPS_BIN -- The location of the system executables
PCF_TEMPLATE -- Full path of directory where the templates of AIT-Framework level PCFs are located
CFG_TEMPLATE -- Full path of directory where the templates of AIT-Framework level CFGs are located
DIR_DEFAULT -- Full path of directory where the AIT-Framework level default PCFs are located
DIR_PCF_OVERWRITES -- Full path of directory where the AIT-Framework level overwritten PCFs are located
DIR_CUSTOMPRODUCTLISTS -- Full path of directory where the AIT-Framework level to the custom product lists
AWG_AER_AOD_AEROSOL_COEF – Full path of directory for the AOD aerosol coefficients
AWG_AER_AOD_SUNGLINT_COEF - Full path of directory for the AOD sunglint coefficients
AWG_AER_AOD_COEF - Full path of directory for the AOD coefficients
AWG_AER_AOD_BRTLNDREF_COEF - Full path of directory for the AOD bright surface reflectance coefficients
NPP_VIIRS_SDR_ANCIL_M -- Full path of npp_viirs_ancil.Mbands.nc
NPP_VIIRS_SDR_ANCIL_I -- Full path of npp_viirs_ancil.Ibands.nc
LAND_MASK_NASA_1KM -- Full path of land mask lw_geo_2001001_v03m.nc
SFC_ELEV_GLOBE_1KM -- Full path of surface elevation GLOBE_1km_digelev.nc
cldmask –Name of the JRR cloud mask (dynamic)
cldheight –Name of the JRR cloud height (dynamic)
gfs_file – name of the NWP GFS file
oisst_file – name of the SST file
gmtco – name of the VIIRS GTMCO file
svm01 – name of the VIIRS SVM file channel 1
svm02 – name of the VIIRS SVM file channel 2
svm03 – name of the VIIRS SVM file channel 3

svm04 – name of the VIIRS SVM file channel 4
svm05 – name of the VIIRS SVM file channel 5
svm06 – name of the VIIRS SVM file channel 6
svm07 – name of the VIIRS SVM file channel 7
svm08 – name of the VIIRS SVM file channel 8
svm09 – name of the VIIRS SVM file channel 9
svm10 – name of the VIIRS SVM file channel 10
svm11 – name of the VIIRS SVM file channel 11
svm12 – name of the VIIRS SVM file channel 12
svm13 – name of the VIIRS SVM file channel 13
svm14 – name of the VIIRS SVM file channel 14
svm15 – name of the VIIRS SVM file channel 15
svm16 – name of the VIIRS SVM file channel 16

An example of **JRR_PRODUCT_AEROSOL_AODADP.pl.PCF** is:

```
#
# name:
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b17764/JR
R_PRODUCT_AEROSOL_AODADP.pl.PCF
#
# working directory:
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b17764
# ProdRuleName: VIIRS-AerosolsProuducts granule (AOD/ADP)

working_directory=/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e10
55479_b17764

job_coverage_start=201504021054237
job_coverage_end=201504021055479

VERSION=v1r1

SCRIPT_FOR_GFS=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-OPS/scripts/run_wgrib.pl
PERL_LOC=/usr/bin/perl
OPS_SCRIPT=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-OPS/scripts
OPS_BIN=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-OPS/bin
PCF_TEMPLATE=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-
OPS/templates/NPP_VIIRS_SDR_Template.pcf
CFG_TEMPLATE=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-
OPS/templates/NPP_VIIRS_AEROSOL_AODADP_Template.cfg
DIR_DEFAULT_PCF=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-OPS/Default_PCF
DIR_PCF_OVERWRITES=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-
OPS/PCF_Overwrites
```

DIR_CUSTOMPRODUCTLISTS=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-OPS/CustomProductLists

AWG_AER_AOD_AEROSOL_COEF=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/AERAOD/VIIRS_Aerosol_Lut.nc
AWG_AER_AOD_SUNGLINT_COEF=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/AERAOD/VIIRS_Sunglint_Lut.nc
AWG_AER_AOD_COEF=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/AERAOD/VIIRS_Coeff.nc
AWG_AER_AOD_BRTLNDREF_COEF=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/AERAOD/VIIRS_BrightSfc_Refl_Coef.nc

NPP_VIIRS_SDR_ANCIL_M=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/ancil_file/npp_viirs_ancil.Mbands.nc
NPP_VIIRS_SDR_ANCIL_I=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/ancil_file/npp_viirs_ancil.Ibands.nc

LAND_MASK_NASA_1KM=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/masks/lw_geo_2001001_v03m.nc
SFC_ELEV_GLOBE_1KM=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_elevation/GLOBE_1km_digelev.nc
SFC_TYPE_AVHRR_1KM=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/masks/gl-latlong-1km-landcover.nc

cldmask=JRR-CloudMask_v1r1_npp_s201504021054237_e201504021055479_c201606291430530.nc
cldheight=JRR-CloudHeight_v1r1_npp_s201504021054237_e201504021055479_c201606291440570.nc
volash=JRR-VolcanicAsh_v1r1_npp_s201504021054237_e201504021055479_c201606291436270.nc

gfs_file=gfs.t06z.pgrb2.0p50.f003.20150402
gfs_file=gfs.t06z.pgrb2.0p50.f006.20150402
oisst_file=avhrr-only-v2.20150401_preliminary.nc

gmtco=GMTCO_npp_d20150402_t1054237_e1055479_b17764_c20150402171617371175_noaa_ops.nc
svm01=SVM01_npp_d20150402_t1054237_e1055479_b17764_c20150402172853771802_noaa_ops.nc
svm02=SVM02_npp_d20150402_t1054237_e1055479_b17764_c20150402172837504244_noaa_ops.nc
svm03=SVM03_npp_d20150402_t1054237_e1055479_b17764_c20150402172859491592_noaa_ops.nc
svm04=SVM04_npp_d20150402_t1054237_e1055479_b17764_c20150402172907467365_noaa_ops.nc
svm05=SVM05_npp_d20150402_t1054237_e1055479_b17764_c20150402172932197325_noaa

_ops.nc
svm06=SVM06_npp_d20150402_t1054237_e1055479_b17764_c20150402172928866557_noaa_ops.nc
svm07=SVM07_npp_d20150402_t1054237_e1055479_b17764_c20150402172933775655_noaa_ops.nc
svm08=SVM08_npp_d20150402_t1054237_e1055479_b17764_c20150402172930996633_noaa_ops.nc
svm09=SVM09_npp_d20150402_t1054237_e1055479_b17764_c20150402172912259531_noaa_ops.nc
svm10=SVM10_npp_d20150402_t1054237_e1055479_b17764_c20150402172939082298_noaa_ops.nc
svm11=SVM11_npp_d20150402_t1054237_e1055479_b17764_c20150402172938840475_noaa_ops.nc
svm12=SVM12_npp_d20150402_t1054237_e1055479_b17764_c20150402172941620436_noaa_ops.nc
svm13=SVM13_npp_d20150402_t1054237_e1055479_b17764_c20150402172946217239_noaa_ops.nc
svm15=SVM15_npp_d20150402_t1054237_e1055479_b17764_c20150402172946982983_noaa_ops.nc
svm16=SVM16_npp_d20150402_t1054237_e1055479_b17764_c20150402172944040007_noaa_ops.nc

#END-of-PCF

Example of **JRR_PRODUCT_AEROSOL_AODADP.pl.PSF**:

/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b17764/JRR-AOD_v1r1_npp_s201504021054237_e201504021055479_c201606291453320.nc
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b17764/JRR-ADP_v1r1_npp_s201504021054237_e201504021055479_c201606291453340.nc

#END-of-PSF

JRR_PRODUCT_AEROSOL_VOLASH.pl.PCF

working_directory – This is the directory at which the job processing occurs
job_coverage_start -- This is the pole-pass time of the first remapped file in YYYYMMDDHHMMSSS format (It is not used by the driver script but for NDE in searching the remapped file)
job_coverage_end -- This is the pole-pass time of the third remapped file in YYYYMMDDHHMMSSS format (It is not used by the driver script but for NDE in searching the remapped file)
VERSION – JRR version

SCRIPT_FOR_GFS -- a script to read GFS grib/grib2 data (run_wgrib.pl), used by AIT-framework program
PERL_LOC -- The location of the Perl interpreter
OPS_SCRIPT -- The location of the system script
OPS_BIN -- The location of the system executables
PCF_TEMPLATE -- Full path of directory where the templates of AIT-Framework level PCFs are located
CFG_TEMPLATE -- Full path of directory where the templates of AIT-Framework level CFGs are located
DIR_DEFAULT -- Full path of directory where the AIT-Framework level default PCFs are located
DIR_PCF_OVERWRITES -- Full path of directory where the AIT-Framework level overwritten PCFs are located
DIR_CUSTOMPRODUCTLISTS -- Full path of directory where the AIT-Framework level to the custom product lists
CRTM_VIIRS_M_SpcCoeff -- Full path of viirs-m_npp.SpcCoeff.bin
CRTM_VIIRS_M_TauCoeff -- Full path of viirs-m_npp.TauCoeff.bin
CRTM_CloudCoeff -- Full path of CloudCoeff.bin
CRTM_AerosolCoeff -- Full path of AerosolCoeff.bin
CRTM_EmisCoeff -- Full path of EmisCoeff.bin
NPP_VIIRS_SDR_ANCIL_M -- Full directory path of npp_viirs_ancil.Mbands.nc
NPP_VIIRS_SDR_ANCIL_I -- Full directory path of npp_viirs_ancil.Ibands.nc
LAND_MASK_NASA_1KM -- Full directory path of land mask lw_geo_2001001_v03m.nc
SFC_ELEV_GLOBE_1KM -- Full directory path of surface elevation GLOBE_1km_digelev.nc
SFC_TYPE_AVHRR_1KM – Full directory path of surface type gl-latlong-1km-landcover.nc
VOLCANO_SMITH_1KM - Full directory path of volcano data volcano_mask_1km.nc
gfs_file – name of the NWP GFS file
oisst_file – name of the SST file
seebor_emiss_01 – full directory path of global_emiss_intABI_2005001.nc
seebor_emiss_02 – full directory path of global_emiss_intABI_2005032.nc
seebor_emiss_03 – full directory path of global_emiss_intABI_2005060.nc
seebor_emiss_04 – full directory path of global_emiss_intABI_2005091.nc
seebor_emiss_05 – full directory path of global_emiss_intABI_2005121.nc
seebor_emiss_06 – full directory path of global_emiss_intABI_2005152.nc
seebor_emiss_07 – full directory path of global_emiss_intABI_2005182.nc
seebor_emiss_08 – full directory path of global_emiss_intABI_2005213.nc
seebor_emiss_09 – full directory path of global_emiss_intABI_2005244.nc
seebor_emiss_10 – full directory path of global_emiss_intABI_2005274.nc
seebor_emiss_11 – full directory path of global_emiss_intABI_2005305.nc

seebor_emiss_12 – full directory path of global_emiss_intABI_2005335.nc
gmtco – name of the VIIRS GTMCO file
svm14 – name of the VIIRS SVM file channel 14
svm15 – name of the VIIRS SVM file channel 15
svm16 – name of the VIIRS SVM file channel 16

An example of **JRR_PRODUCT_AEROSOL_VOLASH.pl.PCF** is:

```
#
# name:
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b17764/JR
R_PRODUCT_AEROSOL_VOLASH.pl.PCF
#
# working directory:
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b17764
# ProdRuleName: VIIRS-AerosolsProuducts granule (volcanic ash)

working_directory=/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e10
55479_b17764
job_coverage_start=201504021054237
job_coverage_end=201504021055479

VERSION=v1r1

SCRIPT_FOR_GFS=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-OPS/scripts/run_wgrib.pl
PERL_LOC=/usr/bin/perl
OPS_SCRIPT=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-OPS/scripts
OPS_BIN=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-OPS/bin
PCF_TEMPLATE=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-
OPS/templates/NPP_VIIRS_SDR_Template.pcf
CFG_TEMPLATE=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-
OPS/templates/NPP_VIIRS_AEROSOL_VOLASH_Template.cfg
DIR_DEFAULT_PCF=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-OPS/Default_PCF
DIR_PCF_OVERWRITES=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-
OPS/PCF_Overwrites
DIR_CUSTOMPRODUCTLISTS=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-
OPS/CustomProductLists

CRTM_VIIRS_M_SpcCoeff=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorith
m_ancillary/crtm_2.0.2/CRTM_Coefficients/Big_Endian/viirs-m_npp.SpcCoeff.bin
CRTM_VIIRS_M_TauCoeff=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorith
m_ancillary/crtm_2.0.2/CRTM_Coefficients/Big_Endian/viirs-m_npp.TauCoeff.bin
CRTM_CloudCoeff=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancill
ary/crtm_2.0.2/CRTM_Coefficients/Big_Endian/CloudCoeff.bin
CRTM_AerosolCoeff=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_anc
```

illary/crtm_2.0.2/CRTM_Coefficients/Big_Endian/AerosolCoeff.bin
CRTM_EmisCoeff=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/crtm_2.0.2/CRTM_Coefficients/Big_Endian/EmisCoeff.bin
NPP_VOLCANIC_ASH_COEF=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VOLCANIC_ASH/ir_retrieval_coeffs_Apr6_2016_154943.nc

NPP_VIIRS_SDR_ANCIL_M=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/ancil_file/npp_viirs_ancil.Mbands.nc
NPP_VIIRS_SDR_ANCIL_I=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/ancil_file/npp_viirs_ancil.Ibands.nc

LAND_MASK_NASA_1KM=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/masks/lw_geo_2001001_v03m.nc
SFC_ELEV_GLOBE_1KM=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_elevation/GLOBE_1km_digelev.nc
SFC_TYPE_AVHRR_1KM=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/masks/gl-latlong-1km-landcover.nc
VOLCANO_SMITH_1KM=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/masks/volcano_mask_1km.nc

seebor_emiss_01=/data/data092/hxie/dlv/npp_dap/nde_apr2015/VCM_DAP_apr2015/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005001.nc
seebor_emiss_02=/data/data092/hxie/dlv/npp_dap/nde_apr2015/VCM_DAP_apr2015/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005032.nc
seebor_emiss_03=/data/data092/hxie/dlv/npp_dap/nde_apr2015/VCM_DAP_apr2015/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005060.nc
seebor_emiss_04=/data/data092/hxie/dlv/npp_dap/nde_apr2015/VCM_DAP_apr2015/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005091.nc
seebor_emiss_05=/data/data092/hxie/dlv/npp_dap/nde_apr2015/VCM_DAP_apr2015/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005121.nc
seebor_emiss_06=/data/data092/hxie/dlv/npp_dap/nde_apr2015/VCM_DAP_apr2015/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005152.nc
seebor_emiss_07=/data/data092/hxie/dlv/npp_dap/nde_apr2015/VCM_DAP_apr2015/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005182.nc
seebor_emiss_08=/data/data092/hxie/dlv/npp_dap/nde_apr2015/VCM_DAP_apr2015/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005213.nc
seebor_emiss_09=/data/data092/hxie/dlv/npp_dap/nde_apr2015/VCM_DAP_apr2015/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005244.nc
seebor_emiss_10=/data/data092/hxie/dlv/npp_dap/nde_apr2015/VCM_DAP_apr2015/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005274.nc
seebor_emiss_11=/data/data092/hxie/dlv/npp_dap/nde_apr2015/VCM_DAP_apr2015/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005305.nc
seebor_emiss_12=/data/data092/hxie/dlv/npp_dap/nde_apr2015/VCM_DAP_apr2015/DATA/static_input/framework_ancillary/seebor_emiss/global_emiss_intABI_2005335.nc

gfs_file=gfs.t06z.pgrb2.0p50.f003.20150402

gfs_file=gfs.t06z.pgrb2.0p50.f006.20150402
oisst_file=avhrr-only-v2.20150401_preliminary.nc
gmtco=GMTCO_npp_d20150402_t1054237_e1055479_b17764_c20150402171617371175_noaa_ops.nc
svm14=SVM14_npp_d20150402_t1054237_e1055479_b17764_c20150402172912285171_noaa_ops.nc
svm15=SVM15_npp_d20150402_t1054237_e1055479_b17764_c20150402172946982983_noaa_ops.nc
svm16=SVM16_npp_d20150402_t1054237_e1055479_b17764_c20150402172944040007_noaa_ops.nc

#END-of-PCF

Example of **JRR_PRODUCT_AEROSOL_VOLASH.pl.PSF**:

/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b17764/JRR-VolcanicAsh_v1r1_npp_s201504021054237_e201504021055479_c201606291436270.nc

#END-of-PSF

JRR_PRODUCT_CRYOSPHERE_SNOW.pl.PCF

working_directory – This is the directory at which the job processing occurs
job_coverage_start -- This is the pole-pass time of the first remapped file in YYYYMMDDHHMMSSS format (It is not used by the driver script but for NDE in searching the remapped file)
job_coverage_end -- This is the pole-pass time of the third remapped file in YYYYMMDDHHMMSSS format (It is not used by the driver script but for NDE in searching the remapped file)
VERSION – JRR version
SCRIPT_FOR_GFS -- a script to read GFS grib/grib2 data (run_wgrib.pl), used by AIT-framework program
PERL_LOC -- The location of the Perl interpreter
OPS_SCRIPT -- The location of the system script
OPS_BIN -- The location of the system executables
PCF_TEMPLATE -- Full path of directory where the templates of AIT-Framework level PCFs are located
CFG_TEMPLATE -- Full path of directory where the templates of AIT-Framework level CFGs are located
DIR_DEFAULT -- Full path of directory where the AIT-Framework level default PCFs are located
DIR_PCF_OVERWRITES -- Full path of directory where the AIT-Framework level overwritten PCFs are located

DIR_CUSTOMPRODUCTLISTS -- Full path of directory where the AIT-Framework level to the custom product lists
snow_occurrence_latlon_week_01 – Full directory path of the snow probability data for week 1
snow_occurrence_latlon_week_02 – Full directory path of the snow probability data for week 2
snow_occurrence_latlon_week_03 – Full directory path of the snow probability data for week 3
snow_occurrence_latlon_week_04 – Full directory path of the snow probability data for week 4
snow_occurrence_latlon_week_05 – Full directory path of the snow probability data for week 5
snow_occurrence_latlon_week_06 – Full directory path of the snow probability data for week 6
snow_occurrence_latlon_week_07 – Full directory path of the snow probability data for week 7
snow_occurrence_latlon_week_08 – Full directory path of the snow probability data for week 8
snow_occurrence_latlon_week_09 – Full directory path of the snow probability data for week 9
snow_occurrence_latlon_week_10 – Full directory path of the snow probability data for week 10
snow_occurrence_latlon_week_11 – Full directory path of the snow probability data for week 11
snow_occurrence_latlon_week_12 – Full directory path of the snow probability data for week 12
snow_occurrence_latlon_week_13 – Full directory path of the snow probability data for week 13
snow_occurrence_latlon_week_14 – Full directory path of the snow probability data for week 14
snow_occurrence_latlon_week_15 – Full directory path of the snow probability data for week 15
snow_occurrence_latlon_week_16 – Full directory path of the snow probability data for week1 6
snow_occurrence_latlon_week_17 – Full directory path of the snow probability data for week 17
snow_occurrence_latlon_week_18 – Full directory path of the snow probability data for week 18
snow_occurrence_latlon_week_19 – Full directory path of the snow probability data for week 19
snow_occurrence_latlon_week_20 – Full directory path of the snow probability data for week 20

snow_occurrence_latlon_week_21 – Full directory path of the snow probability data for week 21
snow_occurrence_latlon_week_22 – Full directory path of the snow probability data for week 22
snow_occurrence_latlon_week_23 – Full directory path of the snow probability data for week 23
snow_occurrence_latlon_week_24 – Full directory path of the snow probability data for week 24
snow_occurrence_latlon_week_25 – Full directory path of the snow probability data for week 25
snow_occurrence_latlon_week_26 – Full directory path of the snow probability data for week 26
snow_occurrence_latlon_week_27 – Full directory path of the snow probability data for week 27
snow_occurrence_latlon_week_28 – Full directory path of the snow probability data for week 28
snow_occurrence_latlon_week_29 – Full directory path of the snow probability data for week 29
snow_occurrence_latlon_week_30 – Full directory path of the snow probability data for week 30
snow_occurrence_latlon_week_31 – Full directory path of the snow probability data for week 31
snow_occurrence_latlon_week_32 – Full directory path of the snow probability data for week 32
snow_occurrence_latlon_week_33 – Full directory path of the snow probability data for week 33
snow_occurrence_latlon_week_34 – Full directory path of the snow probability data for week 34
snow_occurrence_latlon_week_35 – Full directory path of the snow probability data for week 35
snow_occurrence_latlon_week_36 – Full directory path of the snow probability data for week 36
snow_occurrence_latlon_week_37 – Full directory path of the snow probability data for week 37
snow_occurrence_latlon_week_38 – Full directory path of the snow probability data for week 38
snow_occurrence_latlon_week_39 – Full directory path of the snow probability data for week 39
snow_occurrence_latlon_week_40 – Full directory path of the snow probability data for week 40
snow_occurrence_latlon_week_41 – Full directory path of the snow probability data for week 41

snow_occurrence_latlon_week_42 – Full directory path of the snow probability data for week 42
snow_occurrence_latlon_week_43 – Full directory path of the snow probability data for week 43
snow_occurrence_latlon_week_44 – Full directory path of the snow probability data for week 44
snow_occurrence_latlon_week_45 – Full directory path of the snow probability data for week 45
snow_occurrence_latlon_week_46 – Full directory path of the snow probability data for week 46
snow_occurrence_latlon_week_47 – Full directory path of the snow probability data for week 47
snow_occurrence_latlon_week_48 – Full directory path of the snow probability data for week48
snow_occurrence_latlon_week_49 – Full directory path of the snow probability data for week 49
snow_occurrence_latlon_week_50 – Full directory path of the snow probability data for week 50
snow_occurrence_latlon_week_51 – Full directory path of the snow probability data for week 51
snow_occurrence_latlon_week_52 – Full directory path of the snow probability data for week 52
NPPBAYES_CLDMASK_COEF – Full directory path of the Bayes cloud mask coefficients
NPP_VIIRS_SDR_ANCIL_M -- Full directory path of npp_viirs_ancil.Mbands.nc
NPP_VIIRS_SDR_ANCIL_I -- Full directory path of npp_viirs_ancil.Ibands.nc
COAST_MASK_NASA_1KM – Full directory path of coast_mask_1km.nc
LAND_MASK_NASA_1KM -- Full directory path of land mask lw_geo_2001001_v03m.nc
SFC_ELEV_GLOBE_1KM -- Full directory path of surface elevation GLOBE_1km_digelev.nc
SFC_TYPE_AVHRR_1KM – Full directory path of surface type gl-latlong-1km-landcover.nc
climatic_lst_month_01 – Full directory path of climatic land surface temperature month 1
climatic_lst_month_02 – Full directory path of climatic land surface temperature month 2
climatic_lst_month_03 – Full directory path of climatic land surface temperature month 3
climatic_lst_month_04 – Full directory path of climatic land surface temperature month 4
climatic_lst_month_05 – Full directory path of climatic land surface temperature month 5
climatic_lst_month_06 – Full directory path of climatic land surface temperature month 6
climatic_lst_month_07 – Full directory path of climatic land surface temperature month 7
climatic_lst_month_08 – Full directory path of climatic land surface temperature month 8
climatic_lst_month_09 – Full directory path of climatic land surface temperature month 9

climatic_lst_month_10 – Full directory path of climatic land surface temperature month 10
climatic_lst_month_11 – Full directory path of climatic land surface temperature month 11
climatic_lst_month_12 – Full directory path of climatic land surface temperature month 12
cldmask – JPSS RR cloud mask file
gfs_file – name of the NWP GFS file
oisst_file – name of the SST file
gmtco – name of the VIIRS GTMCO file
gitco – name of the VIIRS GITCO file
svi01 – name of the VIIRS SVI file channel 1
svi02 – name of the VIIRS SVI file channel 2
svi03 – name of the VIIRS SVI file channel 3
svi04 – name of the VIIRS SVI file channel 4
svi05 – name of the VIIRS SVI file channel 5

Example of **JRR_PRODUCT_CRYOSPHERE_SNOW.pl.PCF**:

```
#
# name:
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150511_t1225439_e1227081_b18318/JRR_PRODUCT_CRYOSPHERE_SNOW.pl.PCF
#
# working directory:
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150511_t1225439_e1227081_b18318
# ProdRuleName: VIIRS granule (SnowCover/SnowFraction)

working_directory=/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150511_t1225439_e1227081_b18318
job_coverage_start=201505111225439
job_coverage_end=201505111227081

VERSION=v1r1

SCRIPT_FOR_GFS=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-OPS/scripts/run_wgrib.pl
PERL_LOC=/usr/bin/perl
OPS_SCRIPT=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-OPS/scripts
OPS_BIN=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-OPS/bin
PCF_TEMPLATE=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-OPS/templates/NPP_VIIRS_SDR_Template.pcf
CFG_TEMPLATE=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-OPS/templates/NPP_VIIRS_CRYOSPHERE_SNOW_Template.cfg
DIR_DEFAULT_PCF=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-OPS/Default_PCF
```

DIR_PCF_OVERWRITES=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-OPS/PCF_Overwrites
DIR_CUSTOMPRODUCTLISTS=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-OPS/CustomProductLists

snow_occurrence_latlon_week_01=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_01.nc
snow_occurrence_latlon_week_02=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_02.nc
snow_occurrence_latlon_week_03=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_03.nc
snow_occurrence_latlon_week_04=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_04.nc
snow_occurrence_latlon_week_05=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_05.nc
snow_occurrence_latlon_week_06=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_06.nc
snow_occurrence_latlon_week_07=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_07.nc
snow_occurrence_latlon_week_08=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_08.nc
snow_occurrence_latlon_week_09=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_09.nc
snow_occurrence_latlon_week_10=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_10.nc
snow_occurrence_latlon_week_11=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_11.nc
snow_occurrence_latlon_week_12=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_12.nc
snow_occurrence_latlon_week_13=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_13.nc
snow_occurrence_latlon_week_14=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_

week_14.nc
snow_occurrence_latlon_week_15=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_15.nc
snow_occurrence_latlon_week_16=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_16.nc
snow_occurrence_latlon_week_17=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_17.nc
snow_occurrence_latlon_week_18=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_18.nc
snow_occurrence_latlon_week_19=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_19.nc
snow_occurrence_latlon_week_20=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_20.nc
snow_occurrence_latlon_week_21=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_21.nc
snow_occurrence_latlon_week_22=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_22.nc
snow_occurrence_latlon_week_23=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_23.nc
snow_occurrence_latlon_week_24=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_24.nc
snow_occurrence_latlon_week_25=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_25.nc
snow_occurrence_latlon_week_26=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_26.nc
snow_occurrence_latlon_week_27=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_27.nc
snow_occurrence_latlon_week_28=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_28.nc
snow_occurrence_latlon_week_29=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_29.nc

snow_occurrence_latlon_week_30=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_30.nc
snow_occurrence_latlon_week_31=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_31.nc
snow_occurrence_latlon_week_32=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_32.nc
snow_occurrence_latlon_week_33=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_33.nc
snow_occurrence_latlon_week_34=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_34.nc
snow_occurrence_latlon_week_35=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_35.nc
snow_occurrence_latlon_week_36=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_36.nc
snow_occurrence_latlon_week_37=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_37.nc
snow_occurrence_latlon_week_38=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_38.nc
snow_occurrence_latlon_week_39=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_39.nc
snow_occurrence_latlon_week_40=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_40.nc
snow_occurrence_latlon_week_41=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_41.nc
snow_occurrence_latlon_week_42=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_42.nc
snow_occurrence_latlon_week_43=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_43.nc
snow_occurrence_latlon_week_44=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_44.nc
snow_occurrence_latlon_week_45=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/

algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_45.nc
snow_occurrence_latlon_week_46=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_46.nc
snow_occurrence_latlon_week_47=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_47.nc
snow_occurrence_latlon_week_48=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_48.nc
snow_occurrence_latlon_week_49=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_49.nc
snow_occurrence_latlon_week_50=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_50.nc
snow_occurrence_latlon_week_51=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_51.nc
snow_occurrence_latlon_week_52=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_VIIRS_SNOW_COVER/climatic_snow_cover/snow_occurrence_latlon_week_52.nc

NPPBAYES_CLDMASK_COEF=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_ancillary/NPP_BAYES_CldMask/viirs_default_nb_cloud_mask_lut.nc

NPP_VIIRS_SDR_ANCIL_M=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/ancil_file/npp_viirs_ancil.Mbands.nc
NPP_VIIRS_SDR_ANCIL_I=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/ancil_file/npp_viirs_ancil.Ibands.nc

COAST_MASK_NASA_1KM=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/masks/coast_mask_1km.nc
SFC_TYPE_AVHRR_1KM=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/masks/gl-latlong-1km-landcover.nc
LAND_MASK_NASA_1KM=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/masks/lw_geo_2001001_v03m.nc
SFC_ELEV_GLOBE_1KM=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/sfc_elevation/GLOBE_1km_digelev.nc

climatic_lst_month_01=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/climatic_lst/climatic_lst_month_01.nc

climatic_lst_month_02=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/climatic_lst/climatic_lst_month_02.nc
climatic_lst_month_03=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/climatic_lst/climatic_lst_month_03.nc
climatic_lst_month_04=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/climatic_lst/climatic_lst_month_04.nc
climatic_lst_month_05=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/climatic_lst/climatic_lst_month_05.nc
climatic_lst_month_06=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/climatic_lst/climatic_lst_month_06.nc
climatic_lst_month_07=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/climatic_lst/climatic_lst_month_07.nc
climatic_lst_month_08=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/climatic_lst/climatic_lst_month_08.nc
climatic_lst_month_09=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/climatic_lst/climatic_lst_month_09.nc
climatic_lst_month_10=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/climatic_lst/climatic_lst_month_10.nc
climatic_lst_month_11=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/climatic_lst/climatic_lst_month_11.nc
climatic_lst_month_12=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/climatic_lst/climatic_lst_month_12.nc

cldmask=JRR-CloudMask_v1r1_npp_s201505111225439_e201505111227081_c201606291550210.nc

gfs_file=gfs.t06z.pgrb2.0p50.f006.20150511
gfs_file=gfs.t06z.pgrb2.0p50.f009.20150511
oisst_file=avhrr-only-v2.20150510_preliminary.nc

gitco=GITCO_npp_d20150511_t1225439_e1227081_b18318_c20150511185022326281_noaa_ops.nc
gmtco=GMTCO_npp_d20150511_t1225439_e1227081_b18318_c20150511185016595446_noaa_ops.nc
svi01=SVI01_npp_d20150511_t1225439_e1227081_b18318_c20150511185719505736_noaa_ops.nc
svi02=SVI02_npp_d20150511_t1225439_e1227081_b18318_c20150511185628384461_noaa_ops.nc
svi03=SVI03_npp_d20150511_t1225439_e1227081_b18318_c20150511185735602082_noaa_ops.nc
svi04=SVI04_npp_d20150511_t1225439_e1227081_b18318_c20150511185733825498_noaa_ops.nc
svi05=SVI05_npp_d20150511_t1225439_e1227081_b18318_c20150511190707493399_noaa_ops.nc

#END-of-PCF

Example of **JRR_PRODUCT_CRYOSPHERE_SNOW.pl.PSF**:

/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150511_t1225439_e1227081_b18318/JRR-SnowCover_v1r1_npp_s201505111225439_e201505111227081_c201606291557090.nc

#END-of-PSF

JRR_PRODUCT_CRYOSPHERE_ICE.pl.PCF

working_directory – This is the directory at which the job processing occurs
job_coverage_start -- This is the pole-pass time of the first remapped file in YYYYMMDDHHMMSSS format (It is not used by the driver script but for NDE in searching the remapped file)
job_coverage_end -- This is the pole-pass time of the third remapped file in YYYYMMDDHHMMSSS format (It is not used by the driver script but for NDE in searching the remapped file)
VERSION – JRR version
SCRIPT_FOR_GFS -- a script to read GFS grib/grib2 data (run_wgrib.pl), used by AIT-framework program
PERL_LOC -- The location of the Perl interpreter
OPS_SCRIPT -- The location of the system script
OPS_BIN -- The location of the system executables
PCF_TEMPLATE -- Full path of directory where the templates of AIT-Framework level PCFs are located
CFG_TEMPLATE -- Full path of directory where the templates of AIT-Framework level CFGs are located
DIR_DEFAULT -- Full path of directory where the AIT-Framework level default PCFs are located
DIR_PCF_OVERWRITES -- Full path of directory where the AIT-Framework level overwritten PCFs are located
DIR_CUSTOMPRODUCTLISTS -- Full path of directory where the AIT-Framework level to the custom product lists
NPP_ICE_CONC_COEF – Full directory path of the ice concentration coefficients
NPP_ICE_AGE_COEF – Full directory path of the ice age coefficients
NPP_VIIRS_SDR_ANCIL_M -- Full directory path of npp_viirs_ancil.Mbands.nc
NPP_VIIRS_SDR_ANCIL_I -- Full directory path of npp_viirs_ancil.Ibands.nc
LAND_MASK_NASA_1KM -- Full directory path of land mask lw_geo_2001001_v03m.nc
SFC_ELEV_GLOBE_1KM -- Full directory path of surface elevation GLOBE_1km_digelev.nc
cldmask – JPSS RR cloud mask netcdf file

cldheight – JPSS RR cloud height netcdf file
gmtco – name of the VIIRS GTMCO file
svm03 – name of the VIIRS SVM file channel 3
svm04 – name of the VIIRS SVM file channel 4
svm05 – name of the VIIRS SVM file channel 5
svm07 – name of the VIIRS SVM file channel 7
svm10 – name of the VIIRS SVM file channel 10
svm15 – name of the VIIRS SVM file channel 15
svm16 – name of the VIIRS SVM file channel 16

Example of **JRR_PRODUCT_CRYOSPHERE_ICE.pl.PCF**:

```
#
# name:
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150511_t2240157_e2241381_b18324/JR
R_PRODUCT_CRYOSPHERE_ICE.pl.PCF
#
# working directory:
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150511_t2240157_e2241381_b18324
# ProdRuleName: VIIRS-CryosphereProuducts granule (IceConcentration/IceAge)

working_directory=/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150511_t2240157_e22
41381_b18324
job_coverage_start=201505112240157
job_coverage_end=201505112241381

VERSION=v1r1

SCRIPT_FOR_GFS=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-OPS/scripts/run_wgrib.pl
PERL_LOC=/usr/bin/perl
OPS_SCRIPT=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-OPS/scripts
OPS_BIN=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-OPS/bin
PCF_TEMPLATE=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-
OPS/templates/NPP_VIIRS_SDR_Template.pcf
CFG_TEMPLATE=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-
OPS/templates/NPP_VIIRS_CRYOSPHERE_ICE_Template.cfg
DIR_DEFAULT_PCF=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-OPS/Default_PCF
DIR_PCF_OVERWRITES=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-
OPS/PCF_Overwrites
DIR_CUSTOMPRODUCTLISTS=/data/data241/hxie/dlv/JRR_DAP_201606/JRRPS-
OPS/CustomProductLists

NPP_ICE_CONC_COEF=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm
_ancillary/AWG_ICE_CONC/IceSrfTempCoeff_NPP_VIIRS.nc
NPP_ICE_AGE_COEF=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/algorithm_a
```

ncillary/AWG_ICE_AGE/AITA_INPUT_Coefficients_ResiFlux.nc

NPP_VIIRS_SDR_ANCIL_M=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/ancil_file/npp_viirs_ancil.Mbands.nc
NPP_VIIRS_SDR_ANCIL_I=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/ancil_file/npp_viirs_ancil.Ibands.nc

LAND_MASK_NASA_1KM=/data/data241/hxie/dlv/JRR_DAP_201606/DATA/static_input/framework_ancillary/masks/lw_geo_2001001_v03m.nc

cldmask=JRR-CloudMask_v1r1_npp_s201505112240157_e201505112241381_c201606291603120.nc
cldheight=JRR-CloudHeight_v1r1_npp_s201505112240157_e201505112241381_c201606291606150.nc

gmtco=GMTCO_npp_d20150511_t2240157_e2241381_b18324_c20150512050818652746_noaa_ops.nc
svm03=SVM03_npp_d20150511_t2240157_e2241381_b18324_c20150512051509071605_noaa_ops.nc
svm04=SVM04_npp_d20150511_t2240157_e2241381_b18324_c20150512051536984227_noaa_ops.nc
svm05=SVM05_npp_d20150511_t2240157_e2241381_b18324_c20150512051544908335_noaa_ops.nc
svm07=SVM07_npp_d20150511_t2240157_e2241381_b18324_c20150512051554476685_noaa_ops.nc
svm10=SVM10_npp_d20150511_t2240157_e2241381_b18324_c20150512051553047968_noaa_ops.nc
svm15=SVM15_npp_d20150511_t2240157_e2241381_b18324_c20150512051619162409_noaa_ops.nc
svm16=SVM16_npp_d20150511_t2240157_e2241381_b18324_c20150512051540527131_noaa_ops.nc

#END-of-PCF

Example of **JRR_PRODUCT_CRYOSPHERE_ICE.pl.PSF**:

/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150511_t2240157_e2241381_b18324/JRR-IceConcentration_v1r1_npp_s201505112240157_e201505112241381_c201606291611560.nc
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150511_t2240157_e2241381_b18324/JRR-IceAge_v1r1_npp_s201505112240157_e201505112241381_c201606291611560.nc

#END-of-PSF

4.2. Installation

Setup of JPSS RR occurs in four stages: (1) unpacking the DAP, (2) compiling of the code, (3) installation of the compiled code and system files into the NDE production system, (4) and configuration of the NDE system using the production rules supplied in the JPSS documentation. This section identifies the items to be installed

4.2.1. Installation Items

This section lists all of the items that are delivered as part of the JPSS RR DAP that are to be used for generating the JPSS RR products. The JPSS RR DAP produces three subdirectories after it is unpacked. Two of these are the JRRPS-*OPS* and *SOURCE* subdirectories. Details are described in section 3.

All the source code is located at $BASE/SOURCE and below are its sub-directories:

$BASE/SOURCE

SOURCE/extract_scan_time – Contains SDR scan-time-extract code and build

SOURCE/AIT-framework – Contains AIT-framework code.

Under the directory *SOURCE/AIT-framework/src*, there are sub-directories:

Algorithm_Factory – Framework specific code
Algorithm_Factory/Algorithms – Algorithm code
Data_Handling_Structures – Framework specific code
Data_Process -- Framework specific code
Data_Processing_Manager -- Framework specific code
Data_Structure_Code – Algorithm input and output data structures
Includes – Global Parameter Code
Main – Framework driver code
NWP_Utils – NWP utility functions
RTM – RTM code
Satellite_Readers – Satellite reader code

Scientific_Library_Code – Scientific library code
Temporal_Readers – Temporal reader code
Utils – Framework utility functions

The JPSS RR algorithm code and interface code is located within various subdirectories:

SOURCE/AIT-framework/src/Algorithm_Factory/Algorithms/

All of the JPSS system files are in the JRRPS-OPS directories:

$BASE/JRRPS-OPS

and below are its sub-directories*:*

-OPS/bin – A place to put the system's executable files. All of the successfully generated executables need to be moved to this location. Utility Python (version 2.7.3) are also required to be in this directory (can be linked). This directory can be set anywhere as long as it matches the setting of the variable $OPS_BIN in the system's PCF files.

templates – Contains template AIT-framework level configuration files (CFG). All files in this directory should not be modified. Files contained:

NPP_VIIRS_AEROSOL_AODADP_Template.cfg
NPP_VIIRS_AEROSOL_VOLASH_Template.cfg
NPP_VIIRS_CLOUD_CLOUDS_Template.cfg
NPP_VIIRS_CLOUD_MASK_Template.cfg
NPP_VIIRS_CRYOSPHERE_ICE_Template.cfg
NPP_VIIRS_CRYOSPHERE_SNOW_Template.cfg
NPP_VIIRS_SDR_Template.pcf

-OPS/PCF_Overwrites – Contains AIT-framework level process control files (PCF). All files in this directory should not be modified, unless switching the write-out option on each of the related intermediate data files (see below).

Note, switch the "OUTPUT" to "Y" from "N" to write out each related intermediate data files, or vice versa.

-OPS/scripts – contains the processing systems processing scripts.

4.2.2. Compilation Procedures

All NOAA Unique Product (NUP) algorithms are compiled on the AIX operating system statically linked to the standard set of XL compilers. In the near future, algorithms will be migrated to the Dell Blade servers running Redhat Enterprise Linux (RHEL). In the Linux environment, algorithms can statically link to the standard set of GNU Compiler Collection (GCC) compilers or the Intel compilers for Linux.

NUP science algorithms are compiled on dedicated build machines under Configuration Management (CM) control. There is an AIX Power 6 Series build server and a Dell Blade build server running RHEL. When compiling algorithms on the Power 6 build machine, care must be taken to set XL compiler option for “qarch” to “pwr6” which enables the statically built binaries to run on both Power 6 and Power 7 Series. The NDE Test and Dev strings have Power 6 Series machines rather than Power 7 as in Production.

NDE Algorithm Integrators generate a set of build instructions (or in some cases scripts) for CM control and builds. These instructions are specific to each Delivered Algorithm Package (DAP) and are created using the Dev environment for unit testing. The CM Lead builds the algorithms on the appropriate build machine following the controlled instructions (and/or script). The algorithm binaries and all necessary driver scripts, internal static data, tables, etc. are deployed across all processing nodes in the Test Environment. All source code is maintained by the Subversion CM tool and is not deployed to the Test or Production Environments. After verification of the algorithm in the Test Environment, it is promoted to the Production Environment. The same binaries and associated scripts and data are deployed to the /opt/data/nde/NDE_OP1/algorithms/<ALGORITHM NAME> folder on each processing node.

The NOAA-Unique JPSS RR source code is expected to be compiled on a Linux 64 bit machine with Intel compiler version 13.0.1 up and associated NetCDF, HDF5, ZLIB libraries, libxml2. The code must be compiled with "qarch" flag set to "pwr6" for them to work on P6 and P7 machines. Below is the build Instructions for the JPSS RR:

1. Copy and unpack DAP, and build instructions to LINUX build machine
   JRR_DAP_201512.tar.gz
   tar xvfz JRR_DAP_201512*.tar.gz*
2. Set up environment variables for compiling with intel compilers
   source /opt/apps/ots/intel/bin/compilervars.sh intel64
   export PATH=$PATH:/opt/apps/ots/intel/bin
3. CTRM build: Run the following commands
   cd JRR_DAP_<date>/
   tar xvf REL-2.0.2.JCSDA_CRTM_gm.tar
   cd REL-2.0.2.JCSDA_CRTM
   vi configure/ifort.setup

   Add the following to F_COMMON_FLAGS
   F_COMMON_FLAGS="-static-intel"

   . configure/ifort.setup
   make
   make test
   make install
4. Compile the source code:

   Go to the directory *$BASE/SOURCE, modify the build script comp.sh* to point to the correct CRTM, NetCDF4, HDF5, SZIP, ZLIB and libxml2. Then type

   sh comp.sh nde -- for NDE environment with shared library

   Or

   sh comp.sh nde static -- for NDE environment with static library

   After successful compilation, three executables will be generated and located at:

$BASE/SOURCE/bin

The four executables are:
get_sdr_scan_time.exe
pcf_framework.exe
h5augjpss

To compile the executables separately, see section 3.3.

4.2.3. Installation Procedures

The NDE Data Handling System (DHS) “registers” all necessary information for executing instances of NOAA Unique Product (NUP) algorithms in the Oracle Database. Algorithms are executed when all necessary input data has arrived and met all defined criteria for execution. Production jobs are created in the database with all necessary information for that particular instance of execution and are placed in a processing queue. A processing node picks up the queued job and a Process Control File (PCF) is created with all the necessary information for the production job in the database. At this time a driver script for executing the algorithm is called on the appropriate processing node. Therefore, installation of an algorithm into NDE is identical for all algorithms. The first step is to build the executables on the NDE Configuration Management (CM) build machines followed by deployment of those executables (and all other necessary data, e.g. scripts) to the processing nodes. After this, it is simply a matter of registering (to the database) various XMLs describing the algorithm and how to execute it. The following is the minimum set of XMLs required:

- Product Definition XML(s): Each product has its own product definition file which describes all identifying information for a product such as product short name, file name pattern, product quality summary attributes, size, data retention period, etc. This enables the NDE system to ingest products and make them available for product generation or distribution. All ingested input and output products for an algorithm must have a Product Definition XML.
- Algorithm Definition XML: Each algorithm has a single algorithm definition file describing the general characteristics of an algorithm such as the algorithm name and version, the location of the executables, the name of

the driver script, names of all input/output products (only those that require ingest - no static ancillary data needs to be identified), and any algorithm parameter names.

- Production Rule Definition XML: Each algorithm may have one or more production rules associated with it. The production rule file describes how to execute an instance of an algorithm such as specific input data and its characteristics, output data, algorithm parameter values (e.g. flags, location of static data), execution characteristics (temporal refresh, gazetteer, timeout thresholds waiting for input, etc.)

After all XMLs associated with an algorithm and it's production rules have been written, these are then registered to the database using registration scripts:

- registerProduct.pl
- registerAlgorithm.pl
- registerProductionRule.pl
- registerNodeAlgorithm.pl - associates an algorithm with a particular processing node or nodes which allows the node to pick the produciton job up off the queue
- Activate the production rule

Depending on the production rule, a Gazetteer XML may need to be registered. This just describes a region of the Earth that a production rule should be executed in. If this is the case, then registerGazetteer.pl should be run before the registerProductionRule.pl.

The following procedure describes how to register an algorithm and associated production rule to the database.

1. Register the Product Definition XML file
    a. $dm/pds/registerProduct.pl –m NDE_OP1 –f $dm/xmlDefinitionsOps/products/<ALGONAME>/<productName_Definition.xml>
    b. NOTE: if a product is provided to NDE from an external host other than IDPS (e.g. ESPC DDS), then the external host must be registered prior to registering the product (e.g. GFS): $dm/common/registerExternalDataHost.pl –m NDE_OP1 –f $dm/xmlDefinitionsOps/externalDatahosts/<hostname_edh.xml>

Register Algorithm Definition XML file located in the xmlDefinitionsOps folder in $dm=/opt/data/nde/NDE_OP1

. $dm/pgs/registerAlgorithm.pl –m NDE_OP1 –f $dm/xmlDefinitionsOps/algorithms/<ALGONAME>/< ALGONAME_algo.xml>

Register Gazetteer XML file (if applicable – for production rules with a gzFeatureName attribute defined)

. $dm/common/registerGazetteer.pl –m NDE_OP1 –f $dm/xmlDefinitionsOps/gazetteer/<GZNAME_gz.xml>

Register Production Rule Definition XML file located in the xmlDefinitionsOps folder in $dm=/opt/data/nde/NDE_OP1

. $dm/pgs/registerProductionRule.pl –m NDE_OP1 –f $dm/xmlDefinitionsOps/algorithms/<ALGONAME>/< PRNAME_algo.xml>

Register algorithm to processing node(s). May use the interactive script, $dm/pgs/registerNodeAlgorithm.pl, or the drag/drop GUI page for "Algorithm to Node".

Activate the production rule. May use the interactive script, $dm/pgs/managePRFlags.pl, or the GUI page for "Production Rule Description" and click on the "Active Flag" to "ON" and then click the "Save" button (see the GUI interface below).

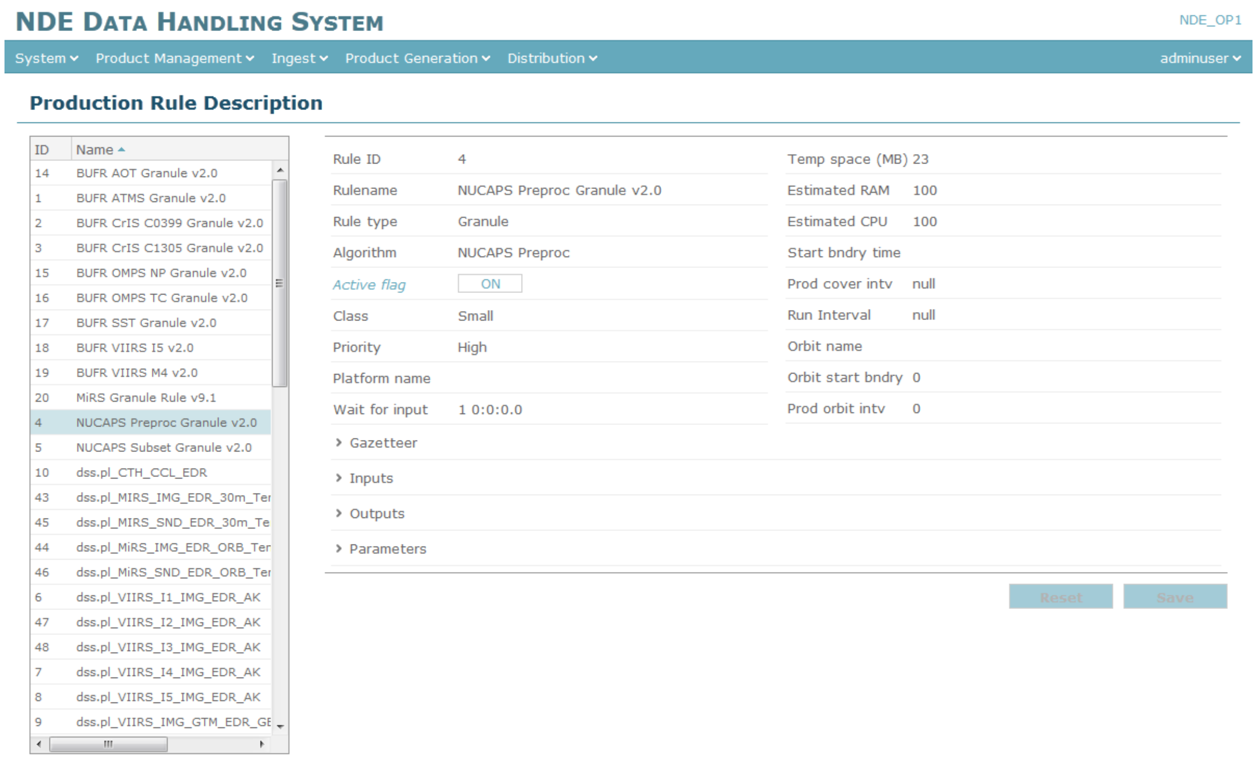


Figure 4-1: The NDE Data Handling System

4.3. Configuration Procedures

4.3.1. Production Rules

Production rules are written as an eXtensible Markup Language (XML) file by the NDE Algorithm Integrator. These XMLs are then registered into the database using the registerProductionRule.pl script provided with the NDE tools. This is the only step required to enter the production rule into the database. No other configuration is required.

The rest of this section describes the production rules required for NDE to properly execute the JPSS RR processing driver scripts. These rules cover three key issues with production: (1) they describe the required inputs and the logic needed to decide what those inputs should be, (2) they describe how to invoke the driver scripts, and (3) they explain what to do with the output.

Input

The driver scripts for the JPSS RR are SDR preprocessing driver script JRR_PREPROCESS_SDR.pl and product driver scripts including JRR_PRODUCT_CLOUD_MASK.pl, JRR_PRODUCT_CLOUD_CLOUDS.pl, JRR_PRODUCT_AEROSOL_AODADP.pl, JRR_PRODUCT_AEROSOL_VOLASH.pl, JRR_PRODUCT_CRYOSPHERE_ICE.pl, and JRR_PRODUCT_CRYOSPHERE_SNOW.pl.
The SDR preprocessing driver scipt will take a VIIRS granule raw data GMTCO, GITCO, SVM01-16, SVI01-05 in h5 format as inputs to do gap filling (if the switch of GAPFILL is on in its PCF) and format conversion from h5 to netCDF. The gap filling is completed in two perl scripts, one for M-band and the other for I-Band resolutions. The gap filling will be killed if its processing time is over a specific period (currently set to 300 seconds in JRRR_PREPROCESS_SDR.pl.PCF).
The product driver script will take required SDR data in netCDF as inputs to generate the JRR products. The following conditions must be met for the driver scripts to be executed.

All of the following input files must be available for JRR_PREPROCESS_SDR..pl:
gmtco – This is the SDR GMTCO HDF5 granule file. gitco -- Same as gmtco, but for image bands. It will have exactly the same time stamp as the gmtco (except the create time)
svm01 -- Same as gmtco, but for SVM01 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm02 -- Same as gmtco, but for SVM02 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm03 -- Same as gmtco, but for SVM03 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm04 -- Same as gmtco, but for SVM04 data. It will have exactly the same time stamp as the gmtco (except the create time)

svm05 -- Same as gmtco, but for SVM05 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm06 -- Same as gmtco, but for SVM06 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm07 -- Same as gmtco, but for SVM07 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm08 -- Same as gmtco, but for SVM08 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm09 -- Same as gmtco, but for SVM09 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm10 -- Same as gmtco, but for SVM10 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm11 -- Same as gmtco, but for SVM11 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm12 -- Same as gmtco, but for SVM12 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm13 -- Same as gmtco, but for SVM13 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm14 -- Same as gmtco, but for SVM14 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm15 -- Same as gmtco, but for SVM15 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm16 -- Same as gmtco, but for SVM16 data. It will have exactly the same time stamp as the gmtco (except the create time)
svi01 -- Same as gmtco, but for SVI01 data. It will have exactly the same time stamp as the gitco (except the create time)
svi02 -- Same as gmtco, but for SVI02 data. It will have exactly the same time stamp as the gitco (except the create time)
svi03 -- Same as gmtco, but for SVI03 data. It will have exactly the same time stamp as the gitco (except the create time)
svi04 -- Same as gmtco, but for SVI04 data. It will have exactly the same time stamp as the gitco (except the create time)
svi05 -- Same as gmtco, but for SVI05 data. It will have exactly the same time stamp as the gitco (except the create time)

All of the following input files must be available for JRR_ PRODUCT_CLOUD_MASK.pl:

gmtco – This is the SDR GMTCO netCDF granule file.
gitco -- Same as gmtco, but for image bands. It will have exactly the same time stamp as the gmtco (except the create time)
svm03 -- Same as gmtco, but for SVM03 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm05 -- Same as gmtco, but for SVM05 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm07 -- Same as gmtco, but for SVM07 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm09 -- Same as gmtco, but for SVM09 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm10 -- Same as gmtco, but for SVM10 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm11 -- Same as gmtco, but for SVM11 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm12 -- Same as gmtco, but for SVM12 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm14 -- Same as gmtco, but for SVM14 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm15 -- Same as gmtco, but for SVM15 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm16 -- Same as gmtco, but for SVM16 data. It will have exactly the same time stamp as the gmtco (except the create time)
svi01 -- Same as gmtco, but for SVI01 data. It will have exactly the same time stamp as the gitco (except the create time)
svi04 -- Same as gmtco, but for SVI04 data. It will have exactly the same time stamp as the gitco (except the create time)
svi05 -- Same as gmtco, but for SVI05 data. It will have exactly the same time stamp as the gitco (except the create time)

**gfs_file* -- This is the 0.5 degree resolution GFS GRIB2 data file whose time most closely matches that of the gmtco target granules. Normally, two GFS data files are expected to each corresponding satellite data. If the expected ones are missing, use the earlier analysis times and forecast times in increments of 6 hours. Its name convention is like:
gfs.tXXz.pgrb2.0p50.fXXX.YYYYMMDD
oisst_file -- This is the 0.25 degree resolution daily OISST data file whose time

most closely matches that of the gmtco target granule. Normally, the current day's data (corresponding to the gmtco granule time) is expected. In the case that the current day's data is not available, use the previous day's data (can back up to seven days to find the nearest available OISST data set). Its name convention is like: avhrr-only-v2.YYYYMMDD_preliminary.nc

All of the following input files must be available for JRR_PRODUCT_CLOUD_CLOUDS.pl:
gmtco – This is the SDR GMTCO netCDF granule file.
svm05 -- Same as gmtco, but for SVM05 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm11 -- Same as gmtco, but for SVM11 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm12 -- Same as gmtco, but for SVM12 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm14 -- Same as gmtco, but for SVM14 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm15 -- Same as gmtco, but for SVM15 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm16 -- Same as gmtco, but for SVM16 data. It will have exactly the same time stamp as the gmtco (except the create time)

cldmask—This is from previous run of Cloud Mask. Its name conversion is like: JRR-CloudMask_v1r1_npp_dYYYYMMDD_tHHMMSSS_eHHMMSSS_b?????_sYYYYMMDDHHMMSSS_eYYYYMMDDHHMMSSS_cYYYYMMDDHHMMSSS.nc. Successful run of Cloud Mask is a prerequisite of running Cloud Height, Cloud Phase, Cloud NCOMP (for a night time granule only), and Cloud DCOMP (for a day time granule only) at a given time.

**gfs_file* -- This is the 0.5 degree resolution GFS GRIB2 data file whose time most closely matches that of the gmtco target granules. Normally, two GFS data files are expected to each corresponding satellite data. If the expected ones are missing, use the earlier analysis times and forecast times in increments of 6 hours. Its name convention is like:
gfs.tXXz.pgrb2.0p50.fXXX.YYYYMMDD
oisst_file -- This is the 0.25 degree resolution daily OISST data file whose time

most closely matches that of the gmtco target granule. Normally, the current day’s data (corresponding to the gmtco granule time) is expected. In the case that the current day’s data is not available, use the previous day’s data (can back up to seven days to find the nearest available OISST data set). Its name convention is like: avhrr-only-v2.YYYYMMDD_preliminary.nc

All of the following input files must be available for JRR_PRODUCT_AEROSOL_AODADP.pl:
gmtco – This is the SDR GMTCO netCDF granule file.
svm01 -- Same as gmtco, but for SVM01 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm02 -- Same as gmtco, but for SVM02 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm03 -- Same as gmtco, but for SVM03 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm04 -- Same as gmtco, but for SVM04 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm05 -- Same as gmtco, but for SVM05 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm06 -- Same as gmtco, but for SVM06 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm07 -- Same as gmtco, but for SVM07 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm08 -- Same as gmtco, but for SVM08 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm09 -- Same as gmtco, but for SVM09 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm10 -- Same as gmtco, but for SVM10 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm11 -- Same as gmtco, but for SVM11 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm12 -- Same as gmtco, but for SVM12 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm13 -- Same as gmtco, but for SVM13 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm15 -- Same as gmtco, but for SVM15 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm16 -- Same as gmtco, but for SVM16 data. It will have exactly the same time

stamp as the gmtco (except the create time)

cldmask—This is Cloud Mask from previous run of Cloud Mask. Its name conversion is like:
JRR-CloudMask_v1r1_npp_dYYYYMMDD_tHHMMSSS_eHHMMSSS_b?????_sYYYYMMDDHHMMSSS_eYYYYMMDDHHMMSSS_cYYYYMMDDHHMMSSS.nc.
cldheight —This is Cloud Height from previous run of Clouds. Its name conversion is like:
JRR-CloudHeight_v1r1_npp_dYYYYMMDD_tHHMMSSS_eHHMMSSS_b?????_sYYYYMMDDHHMMSSS_eYYYYMMDDHHMMSSS_cYYYYMMDDHHMMSSS.nc.
cldheight

Successful run of Cloud Mask and Cloud Height is a prerequisite of running AOD and ADP at a given time.

gfs_file -- This is the 0.5 degree resolution GFS GRIB2 data file whose time most closely matches that of the gmtco target granules. Normally, two GFS data files are expected to each corresponding satellite data. If the expected ones are missing, use the earlier analysis times and forecast times in increments of 6 hours. Its name convention is like:
gfs.tXXz.pgrb2.0p50.fXXX.YYYYMMDD
oisst_file -- This is the 0.25 degree resolution daily OISST data file whose time most closely matches that of the gmtco target granule. Normally, the current day's data (corresponding to the gmtco granule time) is expected. In the case that the current day's data is not available, use the previous day's data (can back up to seven days to find the nearest available OISST data set). Its name convention is like: avhrr-only-v2.YYYYMMDD_preliminary.nc

All of the following input files must be available for JRR_PRODUCT_AEROSOL_VOLASHpl:
gmtco – This is the SDR GMTCO netCDF granule file.
svm14 -- Same as gmtco, but for SVM14data. It will have exactly the same time stamp as the gmtco (except the create time)
svm15 -- Same as gmtco, but for SVM15 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm16 -- Same as gmtco, but for SVM16 data. It will have exactly the same time

stamp as the gmtco (except the create time)

**gfs_file* -- This is the 0.5 degree resolution GFS GRIB2 data file whose time most closely matches that of the gmtco target granules. Normally, two GFS data files are expected to each corresponding satellite data. If the expected ones are missing, use the earlier analysis times and forecast times in increments of 6 hours. Its name convention is like:
gfs.tXXz.pgrb2.0p50.fXXX.YYYYMMDD
oisst_file -- This is the 0.25 degree resolution daily OISST data file whose time most closely matches that of the gmtco target granule. Normally, the current day's data (corresponding to the gmtco granule time) is expected. In the case that the current day's data is not available, use the previous day's data (can back up to seven days to find the nearest available OISST data set). Its name convention is like: avhrr-only-v2.YYYYMMDD_preliminary.nc

All of the following input files must be available for JRR_PRODUCT_CRYOSPHERE_ICE.pl:
gmtco – This is the SDR GMTCO netCDF granule file.
svm03 -- Same as gmtco, but for SVM03 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm04 -- Same as gmtco, but for SVM04 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm05 -- Same as gmtco, but for SVM05 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm07 -- Same as gmtco, but for SVM07 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm10 -- Same as gmtco, but for SVM10 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm15 -- Same as gmtco, but for SVM15 data. It will have exactly the same time stamp as the gmtco (except the create time)
svm16 -- Same as gmtco, but for SVM16 data. It will have exactly the same time stamp as the gmtco (except the create time)

cldmask—This is Cloud Mask from previous run of Cloud Mask. Its name conversion is like:
JRR-CloudMask_v1r1_npp_dYYYYMMDD_tHHMMSSS_eHHMMSSS_b?????_sYYYYMMDDHHMMSSS_eYYYYMMDDHHMMSSS_cYYYYMMDDHHMMSSS.nc.

cldheight —This is Cloud Height from previous run of Clouds. Its name conversion is like:
JRR-CloudHeight_v1r1_npp_dYYYYMMDD_tHHMMSSS_eHHMMSSS_b?????_sYYYYMMDDHHMMSSS_eYYYYMMDDHHMMSSS_cYYYYMMDDHHMMSSS.nc.
cldheight

Successful run of Cloud Mask and Cloud Height is a prerequisite of running Ice Age and Ice Concentration at a given time.

All of the following input files must be available for JRR_PRODUCT_CRYOSPHERE_SNOW.pl:
gmtco – This is the SDR GMTCO netCDF granule file.
gitco -- Same as gmtco, but for GITCO data. It will have exactly the same time stamp as the gmtco (except the create time)
svi01 -- Same as gmtco, but for SVI01 data. It will have exactly the same time stamp as the gmtco (except the create time)
svi02 -- Same as gmtco, but for SVI02 data. It will have exactly the same time stamp as the gmtco (except the create time)
svi03 -- Same as gmtco, but for SVI03 data. It will have exactly the same time stamp as the gmtco (except the create time)
svi04 -- Same as gmtco, but for SVI04 data. It will have exactly the same time stamp as the gmtco (except the create time)
svi05 -- Same as gmtco, but for SVI05 data. It will have exactly the same time stamp as the gmtco (except the create time)

cldmask—This is Cloud Mask from previous run of Cloud Mask. Its name conversion is like:
JRR-CloudMask_v1r1_npp_dYYYYMMDD_tHHMMSSS_eHHMMSSS_b?????_sYYYYMMDDHHMMSSS_eYYYYMMDDHHMMSSS_cYYYYMMDDHHMMSSS.nc.
_cYYYYMMDDHHMMSSS.nc.
cldheight

Successful run of Cloud Mask is a prerequisite of running snow cover and snow fraction at a given time.

**gfs_file* -- This is the 0.5 degree resolution GFS GRIB2 data file whose time

most closely matches that of the gmtco target granules. Normally, two GFS data files are expected to each corresponding satellite data. If the expected ones are missing, use the earlier analysis times and forecast times in increments of 6 hours. Its name convention is like:
gfs.tXXz.pgrb2.0p50.fXXX.YYYYMMDD
oisst_file -- This is the 0.25 degree resolution daily OISST data file whose time most closely matches that of the gmtco target granule. Normally, the current day's data (corresponding to the gmtco granule time) is expected. In the case that the current day's data is not available, use the previous day's data (can back up to seven days to find the nearest available OISST data set). Its name convention is like: avhrr-only-v2.YYYYMMDD_preliminary.nc

Once these input files are available, the NDE DHS must produce a PCF file and put it into a working directory. Then execute the JPSS RR driver script with the working directory as an argument – the driver script assumes that there is an instance of the PCF inside the working directory. The PCF file contains static input files which will be delivered as part of the DAP, and they should be stored and maintained within NDE's database. They should be made available to the JPSS RR via the PCF prior to execution. The PCF will also contain directory settings (see section 4.1 for details).

Execution

Each JPSS RR driver script assumes that a given run will occur in a single working directory unique to the instance of that run. The driver script requires any static input file residing inside or outside of the working directory to be specified with the full path in the PCF file. The script will make the necessary links to the input data. The script reads and writes everything to the working directory. PCF entries cannot be added or removed because the driver script must know to look for them in advance.

Prior to execution, NDE must create a working directory and a PCF file (e.g. JRR_PREPROCESS_SDR.pl.PCF, JRR_PRODUCT_CLOUD_MASK.pl.PCF etc.), located at the working directory for the driver script. See the JPSS RR SMM section 7 for the required contents.

The execution should be as follows:

$PERL_LOC -w $OPS_SCRIPT/$JRR_ PREPROCESS_SDR.pl $WORKING_DIR

$PERL_LOC -w $OPS_SCRIPT/$JRR_PRODUCT $WORKING_DIR

Where:
$PERL_LOC = the Perl command (which may include the full path) with the –w option to show warnings.
$OPS_SCRIPT = the path to the driver script.
$WORKING_DIR = the working directory where the instance of run will happen there
$JRR_PRODUCT refers to the siz product scripts (JRR_PRODUCT_CLOUD_MASK.pl, JRR_PRODUCT_CLOUD_CLOUDS.pl, JRR_PRODUCT_AEROSOL_AODADP.pl, JRR_PRODUCT_AEROSOL_VOLASH.pl, JRR_PRODUCT_CRYOSPHERE_ICE.pl , and JRR_PRODUCT_CRYOSPHERE_SNOW.pl)

After execution, the driver script will return a value. A zero return value indicates success. Any non-zero indicates an error. Details of the error can be found in the log file.

Output

The outputs of the instance run of the SDR preprocessing scripts are GMTCO/GITCO/SVM09-16/SVI01-05 files in netCD. If the GAPFILL switch in the PCF is “ON”, SVM01-16/SVI01-05 data are gapfilled; otherwise data are non-gapfilled. The output of the instance run of the product scripts is the JRR products.

Here is an example of the JPSS RR Product naming convention:

JRR-CloudMask_v1r1_npp_s201504021054237_e201504021055479_c201507171644170.nc

Where,

JRR- JPSS Risk Reduction
CloudMask - Product
v1r1 – Version 1, Release 1
npp – satellite platform
s201504021054237 – Beginning time (4digit year + 2digit month + 2digit day + 2digit hour + 2digit minute + 3digit second)
e201504021055479 -- End time (same as above)
c201507171644170 – product create time (same as above)

4.4. Operations Procedures

4.4.1. Normal Operations

The NDE Data Handling System (DHS) consists of Ingest (INS), Product Generation (PGS), Product Distribution (PDS), and Monitoring subsystems, and is physically spread across numerous servers, both IBM and Linux based, and utilizes a Storage Area Network (SAN) for all data retention.

NDE DHS uses the open source JBoss Service-Oriented Architecture (SOA) platform Java-based middleware to provide the infrastructure (messaging, clustering, failover, load balancing, security, etc) for the INS, PGS, PDS, and Monitoring subsystems. These subsystems are further decomposed into applications with each application built and managed independently. Applications are built using the Enterprise Service Bus (ESB) framework provided by JBoss. NDE DHS consists of five ESB applications which perform the core tasks for ingest, product generation and product distribution; they are Ingest, PGSFactory, DISTFactory, PGSProcessor, and DISTProcessor and are summarized below. Each application (e.g. ingest.esb) is deployed on JBoss servers on different physical servers (e.g. ingest.esb is deployed to JBoss servers on n25rpn01 and n25rpn02). Each JBoss server logs all messages, warnings, and errors in persistent log files that are rolled-over to a new one every 24 hours.

The DHS utilizes the Oracle Spatial 11g Database Management System for all back end data storage and geospatial support. All persistent data, whether it is the catalog of products, the definition and registration of software in the NDE framework, or the messages passed between NDE services, are managed by the Oracle database. DHS access to the backend Oracle database is accomplished with the use of stored procedures, version controlled software (queries) precompiled for efficiency.

In the following paragraphs, each subsystem is described as data flows through the system. Figure 4-2 illustrates the data flow through each subsystem.

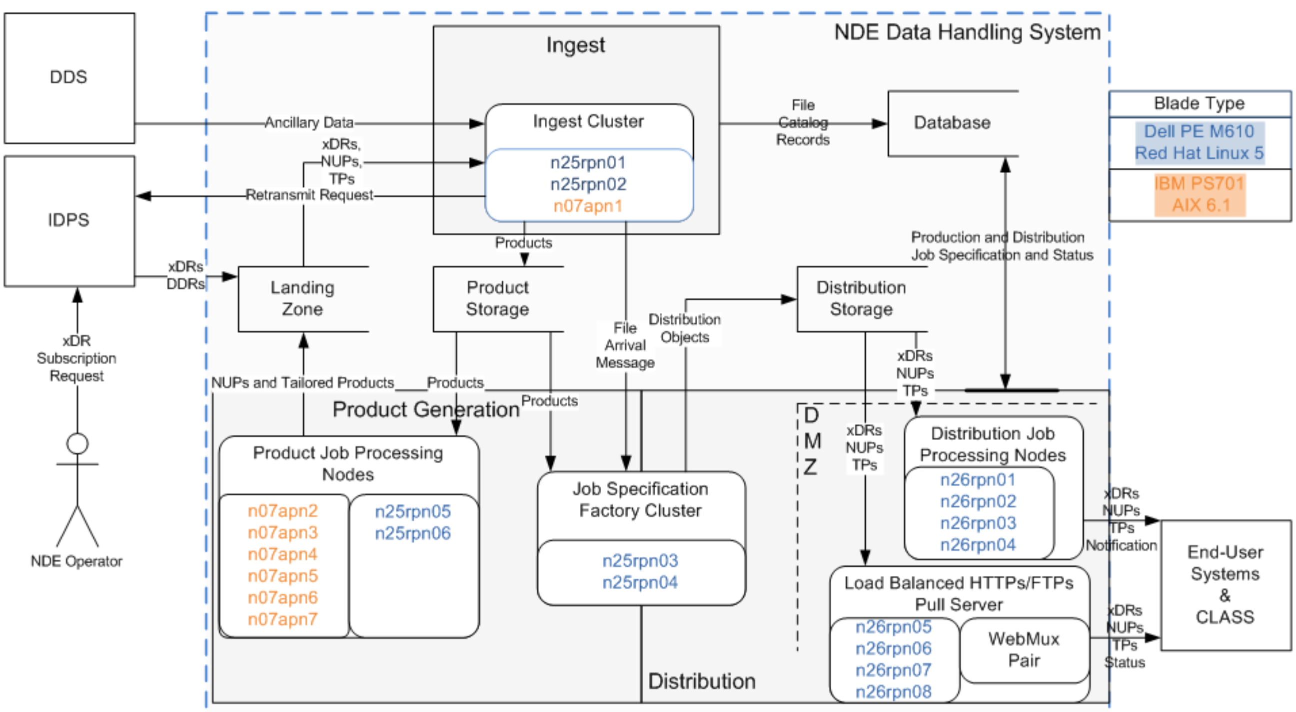


Figure 4-2: NDE DHS Data Flow Diagram

4.4.1.1. Ingest Subsystem (INS)

Ingest (INS) is responsible for the receipt and catalog of all NDE related files. After successful validation, incoming product files (Raw/Sensor/Environmental Data Records xDRs, NOAA Unique Products or NUPs, intermediate products, and ancillary data) and their identifying information (i.e. file metadata) are registered to the database as they arrive in the landing zone of the SAN.

Files are processed by the INS in a Last-In First-Out (LIFO) fashion allowing the most current data to be ingested first if a backlog occurs. This is accomplished using the ingest throttle (a Perl script) which sorts valid files by observation time and makes them available to the Ingest ESB application.

Each arriving file has an associated product alert file (PAF) which initiates the ingest throttle. The PAFs for xDRs distributed by the Joint Polar Satellite System (JPSS) Integrated Data Processing Segment (IDPS) system are Cyclic Redundancy Check (CRC-32) files generated by IDPS in order to monitor file integrity. All other incoming product files (NUPs, intermediate products, and ancillary data) have NDE generated companion (COMP) files which are simply text files containing the associated filename.

The INS servers also handle the acquisition of all ancillary data required by NDE science algorithms from the Environmental Satellite Processing Center (ESPC) Data Distribution Server (DDS) via a File Transfer Protocol (FTP pull) service. The FTP downloader service monitors pull directories for new files (files that have not already been downloaded) via a loop. When new files are made available, the downloader ftp pulls them onto the SAN landing zone and caches them to prevent duplicate pulls.

When a new product file and associated PAF arrives in the SAN landing zone (incoming_input directory), the ingest throttle performs its LIFO function and inserts a record into a temporary buffer table in the database. The Ingest ESB application then has a SQLAdapter monitoring this table for new records; the following services then occur within the Ingest ESB application (the software is implemented as ESB services, with messaging, in a pipeline order):

1. Validate incoming file against the registered products in the database.
    a. Check PAF file name against registered products and match exactly one filename patterns in the database.

   b. If a the pattern matching fails then the file is moved to the ingest_failed directory on the SAN, an error is logged in the INS server log file, and continued messaging through the services ceases.
2. Execute integrity checks on incoming product files.
   a. Retrieves the Ingest Processing Steps defined in the database for the matched product. These steps include a metadata extractor routine (e.g. h5dump, ncdump, or custom extractor scripts) and may include a crc checksum (e.g. with IDPS xDRs). At a minimum, the metadata extractor must be able to retrieve the file observation date, and start/end time. Additional information is retrieved from IDPS xDRs and NUPs such as geolocation of a granule, day/night flag, quality summaries, etc.
   b. Executes a checksum (CRC32) and matches it with the checksum in the PAF (for IDPS xDRs). If checksums fail, the file is moved to the ingest_failed directory on the SAN, an error is logged in the INS server log file, and continued messaging through the services ceases.
3. Persist the product file metadata in the database.
   a. Execute the defined metadata extractor routine.
   b. Insert extracted metadata into the database. If a file has already been ingested (i.e. file metadata has been inserted in the database), it is rejected as a duplicate, moved to the ingest_failed directory on the SAN, an error is logged in the INS server log file, and continued messaging through the services ceases.
   c. Move file to the products directory on the SAN.
   d. Move PAF file to the incoming_processed directory on the SAN.
   e. Send message to PGS Factory and DIST Factory to notify of the newly arrived file.

The Ingest ESB application is deployed to two Linux nodes running JBoss servers. The JBoss ingest servers are clustered together, allowing for workload sharing. If either INS node is unavailable, the other node will take over all of the processing.

4.4.1.2. Product Generation Subsystem (PGS)

NDE PGS handles all data management and processing for NUP science algorithms and product file tailoring for xDRs and NUPs. File tailoring includes aggregation of granules, reformatting, subsampling, and file thinning (selection of specific arrays/variables). PGS implements two overarching applications: the factory (pgsFactory.esb) which manages all potential production jobs and the processor (pgsProcessor.esb) which executes production jobs.

NUP science algorithms and product tailoring production is defined by a set of definition (XML) files which describe the algorithm and its specific execution criteria. There is one algorithm definition file for each algorithm. This XML file describes the general algorithm characteristics such as the algorithm name, driver script name, version, input/output data (product short names), and algorithm parameter names (e.g. flags, directory names). Each algorithm may have one or more XML production rule definition files. The production rule definition file describes the specific execution criteria for running an algorithm such as temporal and geospatial characteristics (e.g. refresh rate and geographical regions for execution), input data characteristics (e.g. imagery solar bands for day granules only), algorithm parameter values (e.g. flag values, paths to static ancillary directories), and output product names. The XML definition files are controlled and are used to register the information into the database via Perl scripts. The figure below illustrates the interface between algorithms and NDE DHS.

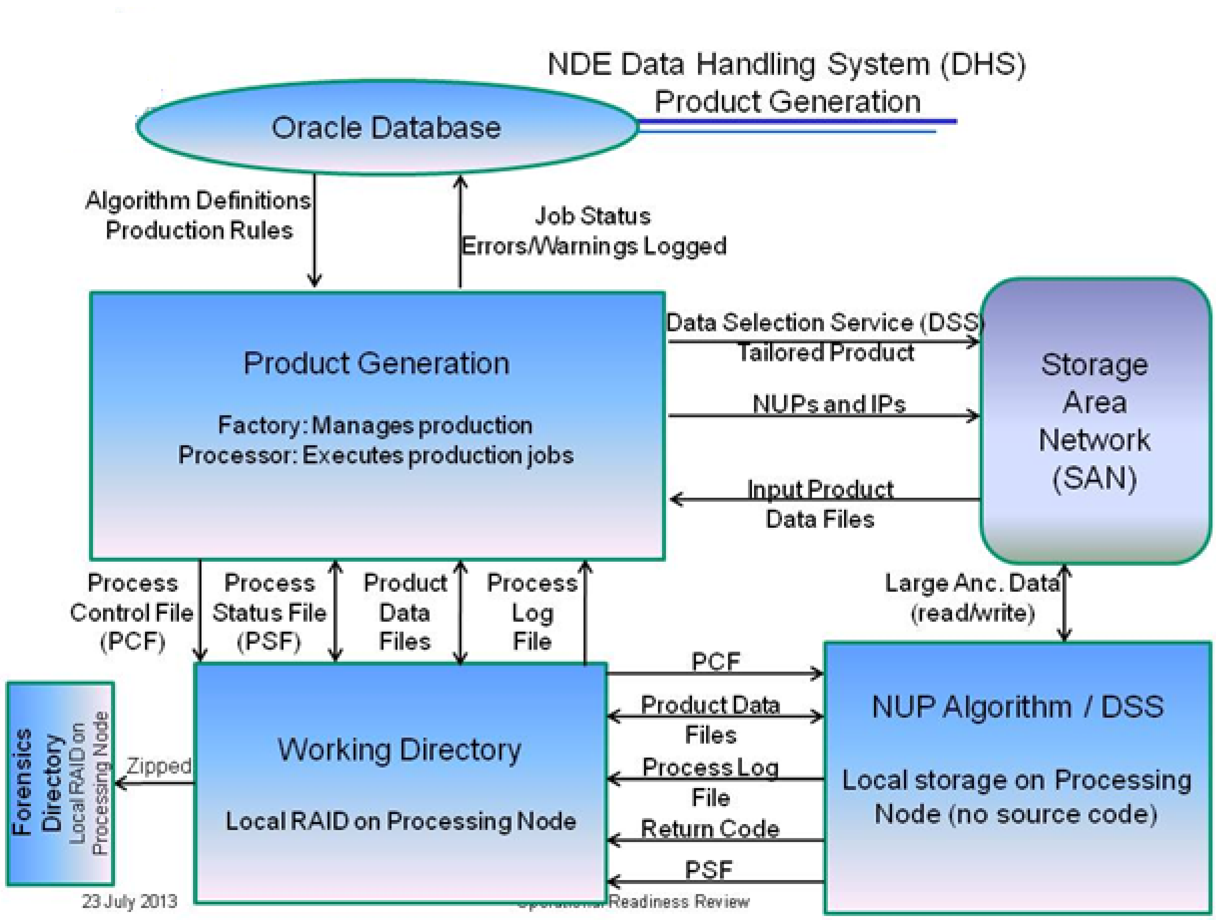


Figure 4-3: Product Generation Data Flows

4.4.1.2.1. PGS Factory

The PGSFactory ESB application is deployed to two Linux nodes running JBoss servers. Both JBoss PGSFactory servers are clustered, allowing for workload sharing between the factory application nodes. If either factory node is unavailable, the other node will take over. It should be noted that the factories are not resource intensive so these JBoss servers support both the production and distribution factories.

All production rules identify a “TRIGGER” input product. When a trigger product file is ingested, a message is sent from INS to the PGSFactory ESB application initiating the PGS process which begins with the creation of a production job spec. The following occurs and is managed by the PGSFactory ESB application (the software is implemented as ESB services, with messaging, in a pipeline order):

1. Create a production job spec.
   a. Factory queries the database for all production rules that have defined the ingested product as a trigger file.
   b. Creates a production job specification for all production rules with the defined trigger product.
2. Monitor production job specs for completion via a loop. A production job spec is not complete until all the required input data has been ingested and covers the observation temporal period while meeting all production rule criteria. A production job spec can time out if required input data does not arrive in a predetermined period of time (defined in the production rule). The clock for timing out begins when the production job spec is created.
   a. Retrieve all input data and associated file metadata for the completed (or timed-out) production job spec. Product files that have been defined as required input for the production rule and that meet the temporal criteria will be retrieved.
   b. Filter out any input data that does not pass specific tests (geo-spatial, quality metadata, etc). For example, a production rule may define an input product as valid only if the observation was during the day (contained in a day/night flag metadata). If a product file does not pass a test, then the product file is not considered valid for the production job spec.
   c. Evaluate input file accumulation threshold. A minimum threshold for temporal coverage can be defined for an input product. If the retrieved input data that has passed the filter tests does not cover enough of the production job spec observation time period, then the production job spec is completed but has a status of “COMPLETE-NOINPUT” and no production job is created.

3. Complete the production job spec after all input data has been ingested, passed the filter tests, and met the minimum file accumulation threshold.
4. Create a production job and queue it for the PGSProcessor.

4.4.1.2.2. PGS Processor

Algorithms (including NDE tailoring) are assigned to dedicated processing nodes which make up the bulk of NDE DHS servers. There are AIX and Linux-based servers (i.e. nodes) that algorithms can be assigned to (mapped to in the database) depending on what platform they are compiled on. Processing nodes are independent (not clustered like Ingest or Factories) allowing for the simple addition of new servers if capacity is reached. Workload sharing is achieved through the use of a database table as a queue.

Each processing node runs a JBoss server for the PGSProcessor ESB application which handles the execution of queued production jobs. Each node is limited with the number of production jobs that can be run simultaneously by a database setting known as production job boxes. This feature allows for load balancing between the nodes and is configurable and can be production rule specific.

After a processing node has picked a production job off the database queue (via looping), the following occurs and is managed by the PGSProcessor ESB application (the software is implemented as ESB services, with messaging, in a pipeline order):

1. Copy input files to working directory.
    a. Creates working directory on processing node local RAID storage.
    b. Copies all required input files for the production job from the SAN product directory to the working directory.
    c. Create the Process Status File (PSF) in the working directory. This file is empty until the algorithm returns from execution and writes all output file names to it.
    d. Create the Process Control File (PCF). A Process Control File (PCF) is created in the working directory which lists all necessary information (name=value) for the production job (e.g. input data file

names, production rule parameters, working directory location, job id). This file has all the information an algorithm requires for executing a particular production job.

2. Run the production job. The algorithm driver script is called by the PGSProcessor application and is given the working directory path as an argument. This allows the algorithm driver script to read the PCF and call all other scripts and binaries for normal execution.
    a. Algorithm returns with a code of zero (success) and writes the output filenames to the PSF.
    b. Algorithm returns with a non-zero code (failure).
3. Perform error handling.
    a. After an algorithm returns and regardless of return code, PGSProcessor scans the algorithm log file for any error/warning messages and enters them in the database.
    b. A non-zero return code from the algorithm causes the PGSProcessor to compress the entire working directory into a zip file and moves it to the forensics directory on the local RAID of the processing node.
4. Get output filenames written to the PSF. Validate file name patterns. All output product files must be registered to the database.
5. Copy output files to the SAN landing zone. The PGSProcessor reads the PSF file written by the algorithm for output filenames. These files are copied from the local working directory to the SAN landing zone for ingestion. PAF files are generated for each output files. PGSProcessor then removes the local working directory.
6. Release the job box for another queued job to be picked up by the processing node.

4.4.1.3. Product Distribution Subsystem (PDS)

NDE Distribution manages all product distribution to customers via subscriptions. All customers are registered in the database and have subscriptions for each desired product. NDE implements File Transfer Protocol Secure (FTPS) push/pull for distribution. Similar to PGS, distribution implements two applications: the factory (distFactory.esb) which creates and manages all distribution jobs and the processor (distProcessor.esb) which executes all push distribution jobs.

There are four dedicated Linux-based servers for both FTPS push and pull jobs for a total of eight distribution platforms. The pull servers interface with a WebMux which load balances across the four pull servers. The WebMux is where pull customers interface with NDE DHS. The pull servers have a monitor to check pull logs on each one and update the database when a customer has pulled a particular file. The four push servers run the JBoss server with the DISProcessor application deployed. The eight servers are located in a different security zone (DMZ) than the rest of the NDE DHS since they must interface with external customer sites.

4.4.1.3.1. DIS Factory

The DISTFactory ESB application is deployed to two Linux nodes running JBoss servers. Both JBoss DISTFactory servers are clustered, allowing for workload sharing between the factory application nodes. If either factory node is unavailable, the other node will take over. It should be noted that the factories are not resource intensive so these JBoss servers support both the production and distribution factories.

After a product file has been ingested, the INS servers send a message to the DISFactory that a subscribed product has arrived. The following occurs and is managed by the DISFactory ESB application (the software is implemented as ESB services, with messaging, in a pipeline order):

1. Qualify the product file for distribution.
    a. Create a distribution prepare request (DPR) for the product file and distribution requests (DRs) for all subscriptions associated with the product. A DPR is similar to the PGS production job spec; it contains all the information required for a distribution job. A single DPR can service several customer subscriptions if they are subscribed to the same product with the same

checksum/compression options. In this scenario, there will be one DPR linked to more than one DR.

   b. Check the data file against any predefined tests set in the subscription (geo-spatial, quality metadata, etc). A customer may specify in the subscription that a product data file must have particular file metadata met such as day/night flags, orbit number, file size, ascending/descending indicator, etc. A customer may also filter data based on quality using the global product quality summaries for IDPS and NUP products. If a file fails any filter tests, the DR for that subscription is removed from the database. If all DRs for a DPR are removed then the DPR is also removed.

2. Retrieve DPRs for completion in a LIFO fashion.
3. Compress product files if the subscription specifies this. Outgoing product files can be compressed using gzip or zip compression. Compressed files are created on the SAN in the dist folder.
4. Generate a checksum if the subscription specifies this. Outgoing product files can have a checksum file created using CRC32 or MD5. Checksums are created on the SAN in the dist folder.
5. Create pull links to product files. The DISFactory creates a link to the product file in the dist directory on the SAN if no compression is specified. Otherwise, the compressed product file is already in the dist directory.
6. Create and queue a distribution job and complete the distribution prepare request and associated distribution requests.
7. Create user links for pull subscriptions. Customers with pull subscriptions access files in their user folder on the SAN. These files are actually links which points either the compressed file or to the link created in the dist folder (which, in turn, points to the product file in the product directory). At this point, pull distribution jobs are ready for the customer and no further action is taken by PDS until the file has been pulled.
8. Create notification jobs if the subscription specifies this. Notifications can be in the form of an email message, a SOAP message, or an FTPS push of a Data Delivery Notification file. The notification job is queued.

When a user pulls a file, the action is logged. A separate standalone Java process (pull log scanner) monitors this log and updates the database to indicate that a user has pulled the file.

4.4.1.3.2. DIS Processor

The NDE DHS DIS Processor handles the execution of all push distribution jobs and notifications. Similar to the PGSProcessor, the DISProcessor ESB application is deployed to JBoss servers that run individually (i.e. not clustered) on push distribution nodes. There are distribution job boxes assigned to each node for load balancing push distributions. After a distribution node has picked a push distribution job off the database queue (via looping), the following occurs and is managed by the DISProcessor (the software is implemented as ESB services, with messaging, in a pipeline order):

1. Retrieve push distribution jobs and execute FTPS push of the product file to the customer.
2. Retrieve notification jobs and either execute an FTPS push notification or send an email or SOAP message.
3. Release the distribution job box for another queued job to be picked up by the distribution node.

4.4.2. Data Preparation

The primary source of input data for NDE is the Joint Polar Satellite System (JPSS) Interface Data Processing Segment (IDPS). The IDPS maintains a connection to the NDE SAN and writes all Suomi-National Polar Partnership (NPP) data files or xDRs (Sensor Data Records, Environmental Data Records, and some Raw Data Records) to the NDE incoming_input directory. In addition, IDPS generates and writes accompanying checksum files to NDE ingest. The checksums are generated using CRC-32 polynomial division. The CRC files are used by the NDE Ingest subsystem to check the integrity of the IDPS xDR file (after NDE generates it's own checksum for comparison). If a file fails the checksum comparison, it is not ingested into NDE and is placed in an ingest_failed directory and noted in the ingest log. During ingest, NDE reads particular metadata fields from the xDR files, which are in Hierarchical Data

Format version 5 (HDF5), using the HDF5 library h5dump executable. If this fails, the product is moved to the ingest_failed directory and is noted in the ingest log.

The other source for external data in the NDE system is the Environmental Satellite Processing and Distribution (ESPDS) Data Distribution System (DDS) which provides all necessary algorithm ancillary data. The NDE Ingest subsystem File Transfer Protocol (FTP) pulls from the DDS servers. There is no checksum capability available for DDS to NDE data. All FTP pull traffic is monitored by the FTP pull log scanner which logs any errors or warnings. NDE extracts the necessary file metadata (file observation start/end times) from the file name using internally developed scripts. If this extraction fails, the file is moved to the ingest_failed directory and is noted in the ingest log.

This is the extent of NDE interaction with an ingested file (besides copying files for execution). All other I/O with ingested files is performed by the algorithm itself. If an algorithm is not able to read a file (e.g. file is corrupted), the algorithm is expected to exit with an appropriate error message in the algorithm log and return a non-zero code to NDE. This will cause the NDE processing node to compress the working directory into a forensics file which is moved to a forensics folder for later analysis.

4.5. Distribution

4.5.1. Data Transfer / Communications

NDE Data Handling System (DHS) distribution is implemented with either a push or pull transaction via FTP Secure (FTPS) protocol. FTPS allows for authentication encryption but no encryption of the data itself. This is more efficient than sFTP protocol which encrypts both. There are 8 distribution servers, 4 for push customers and 4 for pull customers allowing for high availability of distribution. The NDE Distribution subsystem offers notification options for customers. They include email, Simple Object Access Protocol (SOAP) message, or a file-based Data Availability Notification (DAN).

4.5.2. Distribution Restrictions

There is no restriction regarding the release of data products to users, however, only real time users will be served from OSPO distribution system, the non real-time users may order the data from the CLASS.

4.5.3. Product Retention Requirements

The data will be retained for 96 hours on NDE SAN. The NDE Product Quality Monitoring system will retain the level-2 metadata in the database in order to generate the statistics. The statistic result of the JPSS RR pproducts will be kept on database for longer time analysis.

4.5.4. External Product Tools

No external product tools are supplied. The JPSS RR output files are NetCDF4 files. External users can choose their own tools to display and analyze these output files.

5. MONITORING AND MAINTENANCE

5.1. Job Monitoring

Monitoring of the status of the job will be performed by the OSPO operators on a 24 X 7 basis. The monitoring procedures for product generation will be provided by the NDE system developers. The products will be monitored every 30 minutes to ensure their uninterrupted production. The product monitoring also includes the ancillary data as inputs to the products generation.

During the day time the product monitoring function and data quality assurance are also performed by the OSPO contractor and the winds PAL using the NDE product monitor tools which are developed by another project.

5.2. Data Signal Monitoring

The quality of the main VIIRS Level 1 input data stream will be monitored by IDPS. The JPSS RR algorithms will also have some functions to check the quality of the input data.

5.3. Product Monitoring

5.3.1. Unit Test Plans

The test plans and test results are document in the Unit Test Review part in the JPSS RR Readiness Review Document (NESDIS/STAR, 2014).

5.3.2. Internal Product Tools

No internal product tools are supplied. The JPSS RR output files are NetCDF4 files. External users can choose their own tools to display and analyze these output files.

5.3.3. Performance Statistics

NDE product quality monitoring system is used to generate the near real time statistic from the metadata of the JPSS RR products and monitor the quality of the JPSS RR products. The variables for the JPSS RR monitoring, daily JPSS RR monitoring and monthly JPSS RR monitoring include total number of target identified, total number of good values for the products, minimum, maximum, mean, and standard deviation of the products, and percent of targets associated with each QA flag value. These performance statistics are kept in the database of NDE product quality monitoring system.

5.3.4. Product Monitoring

The quality of JPSS RR products is monitored by near real time NDE product quality monitoring system and the comparison with the radiosonde data. The performance statistics are generated by product quality monitoring system. Based on these statistics, the system provides capabilities for automatic monitoring of JPSS RR product quality in near real time and send out notification of the anomalies.

In addition, STAR routinely compares the JPSS RR products with the radiosonde data to monitor the quality of products.

5.3.5. Product Criticality

JPSS RR is identified as a mission critical and therefore a 24 X 7 service maintenance level is required. Return to service within 2 hours is required.

5.4. Maintenance

5.4.1. Monitoring

The NDE Data Handling System (DHS) provides a Graphical User Interface (GUI) for monitoring all of the subsystem functionality (ingest, product generation, and distribution), resource utilization (CPUs, memory, storage), and system performance. The GUI is provided through the interactive NDE DHS Portal and consists of a Dashboard monitoring overall system health and a series of pages and links for increasingly detailed looks at the various subsystems. For further details on monitoring, see the NDE Operations Handbook and the NDE Software User's Manual. The following summarizes simple ways to monitor ingest, product generation, and product distribution.

Monitoring begins with the NDE DHS dashboard where the high level summary of the state of the system is described. The window shows the state of all subsystems and their associated nodes, i.e. servers, (green = nominal, red = down). All windows show the backlog and throughput for each subsystem. A user can configure the stats period ("Last 24 hrs" in the figure below) to show information going back for different periods of time. The ingest window shows the number of failed ingests over the user defined period. The production window summarizes the number of failed production jobs over the defined stats period, and the excessive run time jobs – these are jobs that have been running far too long, i.e., stuck in a processing state. Finally, the distribution window shows the number of failed and expired (did not meet customer defined latency) distribution jobs. All of the summaries are clickable and will take the user to a more detailed page.

In addition to the NDE Portal, every JBoss server within the DHS (ingest/factory/product generation/distribution) maintain active logs. These logs record all activity on the server and capture any warnings and/or errors from NDE applications. All server logs can be found locally in the $logdir (/opt/apps/ots/Jboss/Jboss-soa-p-5/Jboss-as/server/nde_op1/log) directory. They are stored indefinitely with new files created daily at 0Z. The server.log is the current log file. Log files should be used to check that Jboss servers are up and running and for any error messages.

All NOAA Unique Products (NUP) Delivered Algorithm Packages (DAPs) are required to generate a log for each instance of execution of a production job. This log is scanned by the NDE DHS for any warnings or errors which, if found, are recorded in the database (PRODUCTIONJOBLOGMESSAGES table). If an algorithm fails, the working directory which includes the log file are compressed and stored in a forensics folder for later analysis. The forensics folder is located on the locally on the processing server in /opt/data/nde/NDE_OP1/pgs/forensics_data. The compressed forensics files are named "forensicsDump_<DATE>_<OBS_START_TIME>_<production_job_id>.zip".

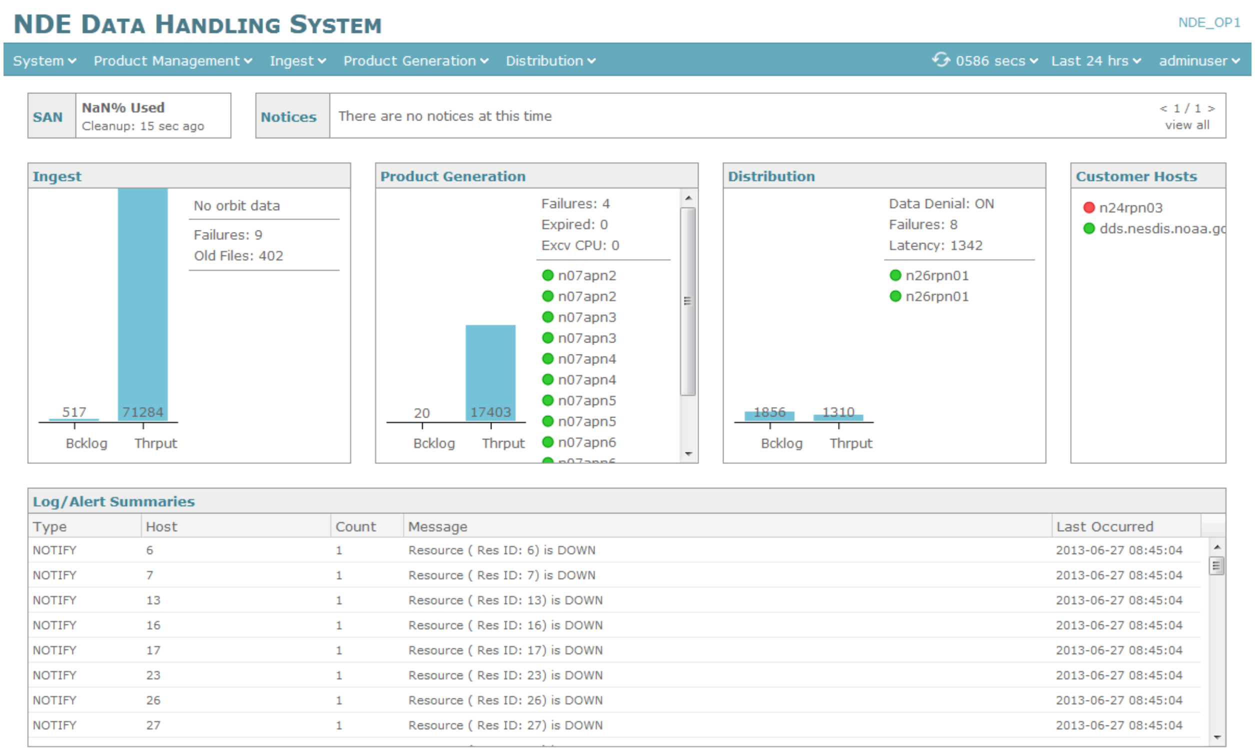
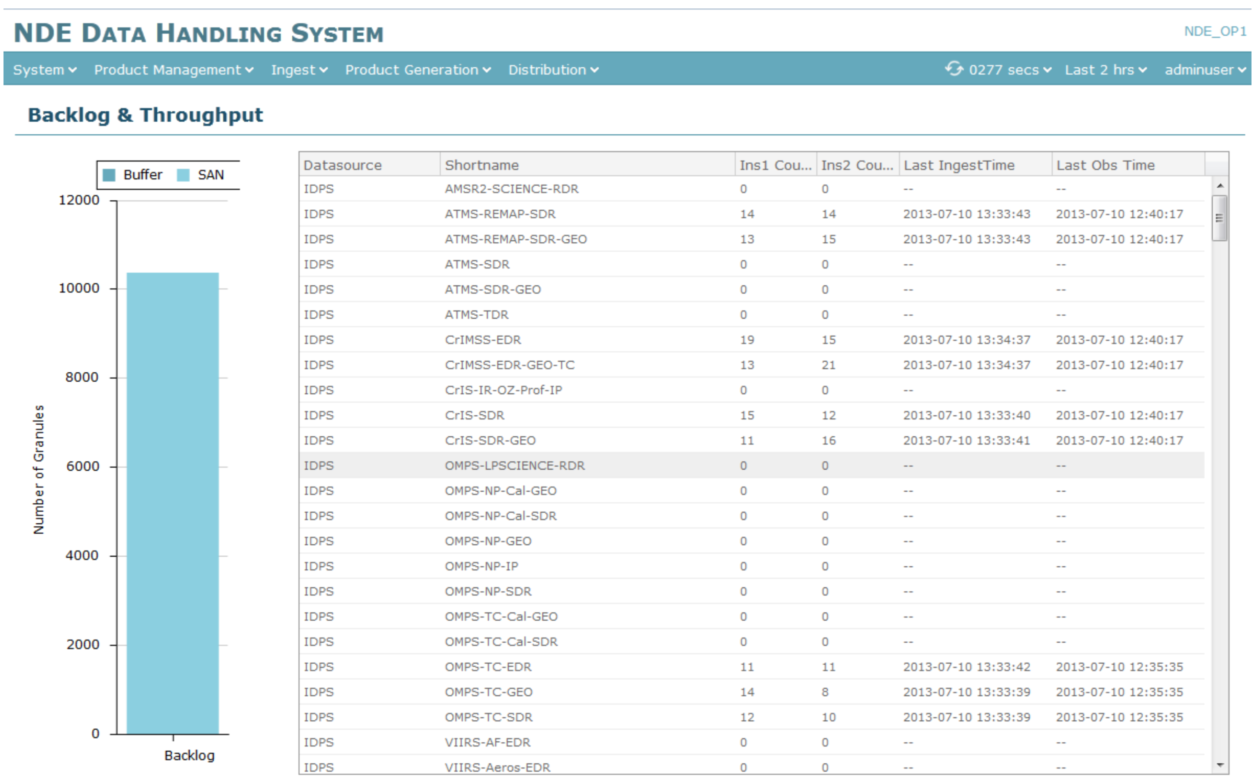


Figure 5-1: NDE DHS Dashboard GUI page

5.4.1.1. Ingest Monitoring

Monitoring of ingest can be accomplished using the Ingest Backlog and Throughput page. This page shows the backlog in the landing zone (incoming_input directory on the SAN) and any files in the ingest buffer table (post LIFO sorting by the ingest throttle but pre ingest by the ingest JBoss server) - this is also shown on the dashboard. A list of every registered product by product shortname and the time of the last ingested file for that product is shown on this page. It also shows the observation time of the last ingested file along with the number of files ingested by each server.

If the backlog is large (>10K) or is growing, then there is a problem with ingest. The ingest servers may be slow or servers may be down. Tailing the ingest log (“tail -f $logdir/server.log”) will show whether the server is ingesting products or not. The log will also capture any errors that may have occurred. The ingest throttle maintains a log file in the /opt/data/nde/NDE_OP1/logs/ins folder.



Datasource	Shortname	Ins1 Cou...	Ins2 Cou...	Last IngestTime	Last Obs Time
IDPS	AMSR2-SCIENCE-RDR	0	0	--	--
IDPS	ATMS-REMAP-SDR	14	14	2013-07-10 13:33:43	2013-07-10 12:40:17
IDPS	ATMS-REMAP-SDR-GEO	13	15	2013-07-10 13:33:43	2013-07-10 12:40:17
IDPS	ATMS-SDR	0	0	--	--
IDPS	ATMS-SDR-GEO	0	0	--	--
IDPS	ATMS-TDR	0	0	--	--
IDPS	CrIMSS-EDR	19	15	2013-07-10 13:34:37	2013-07-10 12:40:17
IDPS	CrIMSS-EDR-GEO-TC	13	21	2013-07-10 13:34:37	2013-07-10 12:40:17
IDPS	CrIS-IR-OZ-Prof-IP	0	0	--	--
IDPS	CrIS-SDR	15	12	2013-07-10 13:33:40	2013-07-10 12:40:17
IDPS	CrIS-SDR-GEO	11	16	2013-07-10 13:33:41	2013-07-10 12:40:17
IDPS	OMPS-LPSCIENCE-RDR	0	0	--	--
IDPS	OMPS-NP-Cal-GEO	0	0	--	--
IDPS	OMPS-NP-Cal-SDR	0	0	--	--
IDPS	OMPS-NP-GEO	0	0	--	--
IDPS	OMPS-NP-IP	0	0	--	--
IDPS	OMPS-NP-SDR	0	0	--	--
IDPS	OMPS-TC-Cal-GEO	0	0	--	--
IDPS	OMPS-TC-Cal-SDR	0	0	--	--
IDPS	OMPS-TC-EDR	11	11	2013-07-10 13:33:42	2013-07-10 12:35:35
IDPS	OMPS-TC-GEO	14	8	2013-07-10 13:33:39	2013-07-10 12:35:35
IDPS	OMPS-TC-SDR	12	10	2013-07-10 13:33:39	2013-07-10 12:35:35
IDPS	VIIRS-AF-EDR	0	0	--	--
IDPS	VIIRS-Aeros-EDR	0	0	--	--

Figure 5-2: Backlog and Throughput

The ingest panel of the dashboard shows the number of failed ingests. In some cases, failing ingest is part of normal operations (e.g. duplicate files arrive on the landing zone). This number should never be rising rapidly. There is a failed granules page in the ingest dropdown menu that lists all failed granules. In terms of product counts per orbit, nominally there should be ~72 granules of VIIRS files (for each channel), ~191 ATMS and CrIS granules each, and ~160 OMPS granules. An orbit is ~101 minutes so VIIRS granules are ~86 seconds, CrIS and ATMS are ~32 seconds, and OMPS is ~38 seconds.

5.4.1.2. Production Job Monitoring

Product generation is defined by a single or set of production rules for each algorithm (science and NDE Data Selection Services). The NDE DHS is data driven, therefore every production rule defines a single "trigger" product that, upon ingest, will cause the product generation factories to instantiate a production job spec for that particular production rule. The production job spec has an initial state of "INCOMPLETE" and represents a potential production job. The actual production job is not created and made available to the processors until the production job spec is put into a state of "COMPLETE". This doesn't happen until all required input data has been ingested into NDE and all production rule criteria have been met (e.g. observation data is over a particular region/gazetteer). If, however, not all required data that meets the production rule criteria arrives in a predefined period of time, then the production job spec will enter a permanent state of "COMPLETE-NOINPUT" and all activities associated with that spec will end.

After a production job spec has completed, a production job will be created by the product generation factory and initialized to a state of "QUEUED". The assigned processing nodes for the particular algorithm will be looking for production jobs that are queued. After finding a queued job and if the processing node has an available job box, the production job state will change to "COPYINPUT" while the processor copies all input data to the working directory. After all data has been successfully staged in the working directory, the production job changes to a state of "RUNNING" while the algorithm executes that particular job. The production job will remain in this state until the algorithm returns with a code or errors out. If the algorithm returns with a non-zero code or simply errors out (e.g. core dump), then the production job state will change to "FAILED". A failed state will cause the processor to compress the working directory and all of its contents into a zip file that is copied to the forensics directory for offline analysis. If the algorithm returns with a zero code, then the state will change to "COPYOUTPUT" while the processor copies the output algorithm files to the SAN for ingest. After successful completion, the state changes to "COMPLETE". The following tables list the various states a production job spec and production job can be in.

Table 5-1: Production Job Spec States

State	Initialization	Transition
INCOMPLETE	PG Factory initiates a Production Job Spec when a trigger product file is ingested (trigger is defined in the production rule).	PG Factory checks the database for all required input data for the Production Job Spec (as defined in the production rule) that meets all defined criteria (geospatial, tests, accumulation threshold). Can transition to COMPLETING or COMPLETE-NOINPUT.
COMPLETING	PG Factory has determined	PG Factory populates all of

	that all required input data that satisfies the production rule criteria has been ingested in time.	the JOBSPEC tables and creates the Production Job. Can transition only to COMPLETE.
COMPLETE	JOBSPEC tables and Production Job have been created.	Production Job Spec is done – no transition from this state. Production Job is created in a state of QUEUED.
COMPLETE-NOINPUT	PG Factory has determined that required input data has not been ingested within a predefined period of time (as defined in the production rule).	Production Job Spec is done – no transition from this state.

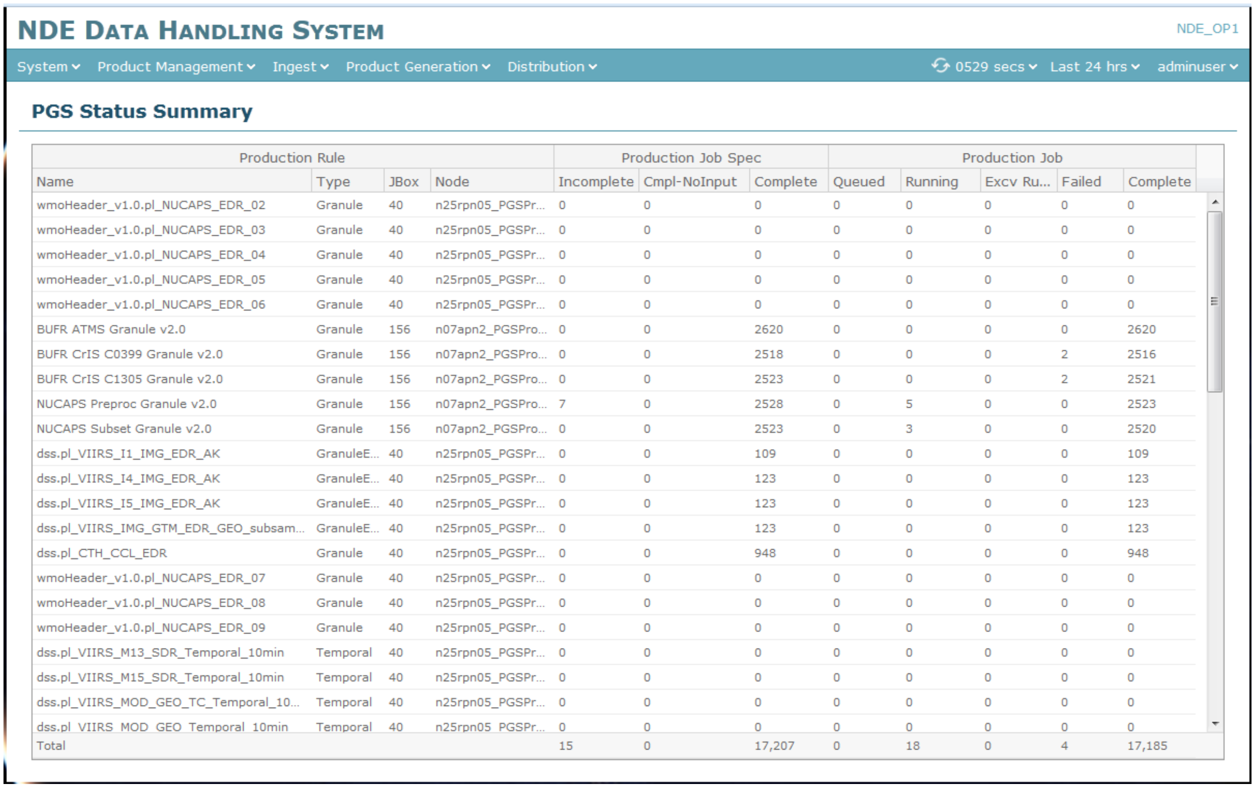
Table 5-2: Production Job Status

State	**Initialization**	**Transition**
QUEUED	PG Factory has completed a production job spec and created the production job.	Processing nodes that have been assigned to the algorithm and that have available job boxes select a queued job. Can transition only to ASSIGNED.

ASSIGNED	Processing node has picked the production job off of the QUEUE.	Transition to COPYINPUT.
COPYINPUT	Processor has picked up queued job and is copying all input data from the SAN to the working directory.	Processing node successfully completes copying input data from the SAN to the working directory. Can transition to RUNNING or FAILED.
RUNNING	All input data has been copied to the working directory and the processor has executed the algorithm driver script.	Algorithm returns with a code. Can transition to METADATA or FAILED
METADATA	Algorithm has returned with a zero code and written output file names to the Process Status File (PSF). NDE DHS reads the PSF file names to be ingested.	Processing node successfully reads the output file names from the PSF and validates the file name pattern with a registered product. Can transition to FAILED or COPYOUTPUT
COPYOUTPUT	File names have been read from the PSF file and validated..	Processing node successfully completes copying output data from the working directory to the

		SAN. Can transition to COMPLETE or FAILED.
COMPLETE	All output data has been successfully copied from the working directory to the SAN and the working directory is removed.	Production Job is done – no transition from this state.
FAILED	An error has occurred somewhere along the processing chain (e.g. algorithm returns non-zero code, copy fails).	Processor compresses the working directory into a zip file and copies it to the forensics directory for offline analysis. Production Job is done – no transition from this state. However, job can be manually re-queued if desired.

The Product Generation drop down menu has a link to the PGS Status Summary which represents a tally of all the current states a particular production rule is in. The figure below shows the status page. All states are clickable and will bring up the specific production job specs or production jobs for that status. The Production Job Spec and Production Job Status page provides more detailed information.



Production Rule				Production Job Spec			Production Job				
Name	Type	JBox	Node	Incomplete	Cmpl-NoInput	Complete	Queued	Running	Excv Ru...	Failed	Complete
wmoHeader_v1.0.pl_NUCAPS_EDR_02	Granule	40	n25rpn05_PGSPr...	0	0	0	0	0	0	0	0
wmoHeader_v1.0.pl_NUCAPS_EDR_03	Granule	40	n25rpn05_PGSPr...	0	0	0	0	0	0	0	0
wmoHeader_v1.0.pl_NUCAPS_EDR_04	Granule	40	n25rpn05_PGSPr...	0	0	0	0	0	0	0	0
wmoHeader_v1.0.pl_NUCAPS_EDR_05	Granule	40	n25rpn05_PGSPr...	0	0	0	0	0	0	0	0
wmoHeader_v1.0.pl_NUCAPS_EDR_06	Granule	40	n25rpn05_PGSPr...	0	0	0	0	0	0	0	0
BUFR ATMS Granule v2.0	Granule	156	n07apn2_PGSPro...	0	0	2620	0	0	0	0	2620
BUFR CrIS C0399 Granule v2.0	Granule	156	n07apn2_PGSPro...	0	0	2518	0	0	0	2	2516
BUFR CrIS C1305 Granule v2.0	Granule	156	n07apn2_PGSPro...	0	0	2523	0	0	0	2	2521
NUCAPS Preproc Granule v2.0	Granule	156	n07apn2_PGSPro...	7	0	2528	0	5	0	0	2523
NUCAPS Subset Granule v2.0	Granule	156	n07apn2_PGSPro...	0	0	2523	0	3	0	0	2520
dss.pl_VIIRS_I1_IMG_EDR_AK	GranuleE...	40	n25rpn05_PGSPr...	0	0	109	0	0	0	0	109
dss.pl_VIIRS_I4_IMG_EDR_AK	GranuleE...	40	n25rpn05_PGSPr...	0	0	123	0	0	0	0	123
dss.pl_VIIRS_I5_IMG_EDR_AK	GranuleE...	40	n25rpn05_PGSPr...	0	0	123	0	0	0	0	123
dss.pl_VIIRS_IMG_GTM_EDR_GEO_subsam...	GranuleE...	40	n25rpn05_PGSPr...	0	0	123	0	0	0	0	123
dss.pl_CTH_CCL_EDR	Granule	40	n25rpn05_PGSPr...	0	0	948	0	0	0	0	948
wmoHeader_v1.0.pl_NUCAPS_EDR_07	Granule	40	n25rpn05_PGSPr...	0	0	0	0	0	0	0	0
wmoHeader_v1.0.pl_NUCAPS_EDR_08	Granule	40	n25rpn05_PGSPr...	0	0	0	0	0	0	0	0
wmoHeader_v1.0.pl_NUCAPS_EDR_09	Granule	40	n25rpn05_PGSPr...	0	0	0	0	0	0	0	0
dss.pl_VIIRS_M13_SDR_Temporal_10min	Temporal	40	n25rpn05_PGSPr...	0	0	0	0	0	0	0	0
dss.pl_VIIRS_M15_SDR_Temporal_10min	Temporal	40	n25rpn05_PGSPr...	0	0	0	0	0	0	0	0
dss.pl_VIIRS_MOD_GEO_TC_Temporal_10...	Temporal	40	n25rpn05_PGSPr...	0	0	0	0	0	0	0	0
dss.pl_VIIRS_MOD_GEO_Temporal_10min	Temporal	40	n25rpn05_PGSPr...	0	0	0	0	0	0	0	0
Total				15	0	17,207	0	18	0	4	17,185

Figure 5-3: Product Generation Subsystem (PGS) Status Summary GUI page

The PGS Job Spec & Job Status page includes detailed information such as id numbers, observation times, start/stop times for all specs and jobs, and the statuses. Clicking on an individual job will bring up all current information on a particular job spec and/or job.

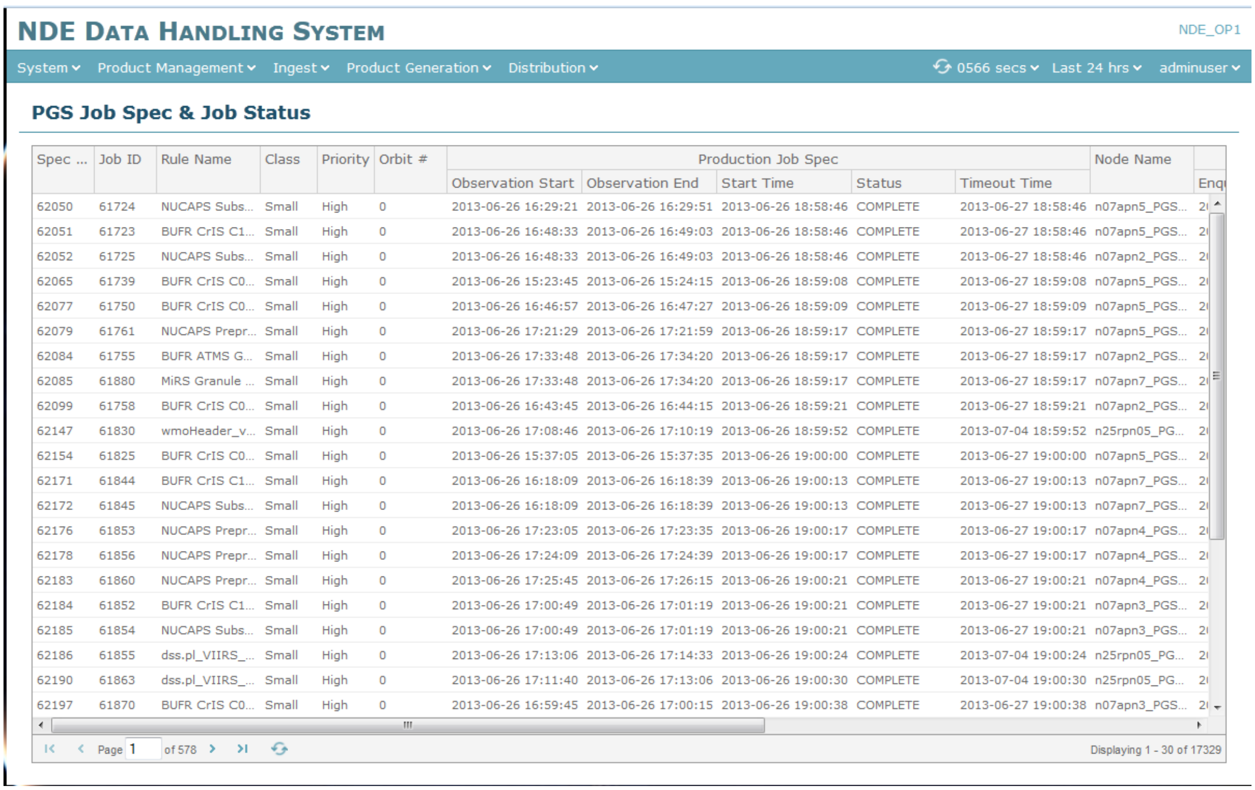


Figure 5-4: Product Generation Subsystem (PGS) Job Spec and Production Job Status GUI page

The detailed window shows all input data for a particular job (figure below). In this screenshot there are two input products listed when there should only be one, given the production rule description. This has happened because the two files have overlapping observation times. This is a known issue with this particular algorithm. This serves as an example of clicking through the GUI pages starting at the dashboard to look at a particular problem and finding the issue.

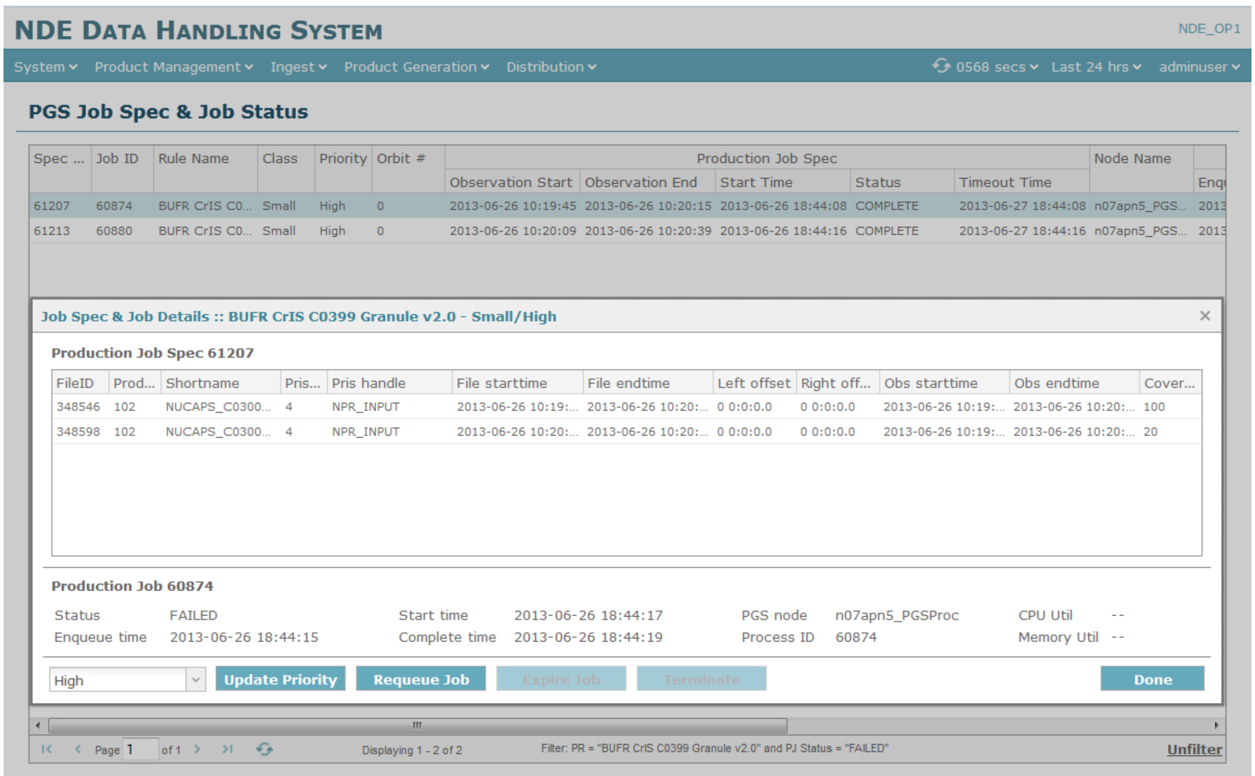


Figure 5-5: Production Job Spec & Production Job details GUI page

5.4.1.3. Product Distribution Monitoring

Product distribution begins when a user subscribed product file is ingested and the ingest server sends a message to the distribution factory. The distribution factory creates a distribution prepare request which contains all the information about a particular product file and simultaneously creates distribution requests for all users subscribed to the product. A single distribution prepare request may have one or many distribution requests associated with it. After a distribution prepare request completes, the factory creates a distribution job. The following table lists the various states for distribution prepare requests and distribution jobs.

Table 5-3: Distribution Prepare Request States

State	Initialization	Transition
INITIAL	Created by the distribution factory when subscribed product is ingested and message is received from ingest server.	Distribution prepare request has been created. Transitions to PREPARING.
PREPARING	Factory created a distribution prepare request.	Factory sorts DPRs by priority and LIFO. Transitions to PROCESSING.
PROCESSING	Factory has sorted DPRs.	Factory creates links to subscribed product file in the dist directory on the SAN if it is a pull job. If defined in the subscription, the factory compresses the product file and/or creates a checksum in the dist directory. Can transition to COMPLETE or EXPIRED.

COMPLETE	Factory has successfully created pull links to product file in the dist directory if it is a pull job and, if applicable, compressed file and created checksum.	Distribution job is created by the factory.
EXPIRED	Subscription time latency threshold has been met.	No transition from this state.

Table 5-4: Pull Distribution Job States

State	**Initialization**	**Transition**
FINALIZING	Distribution prepare request is complete.	Stored procedure creates distribution job.
QUEUED	Distribution prepare request is complete and distribution job has been created.	Transitions to RUNNING.
RUNNING	Immediate transition from queued state for pull jobs.	Factory successfully creates user links to product link in dist directory. Can transition to LINKREADY or LINKFAILED.
LINKREADY	Factory successfully	User must FTPS pull the

	created user links.	product to transition to PULLCOMPLETE. If pull fails or user does not pull, stays in this state.
LINKFAILED	Factory did not create the user link.	No transition from this state without user intervention.
PULLCOMPLETE	User pulls the product.	Updated by cron job script that scans the FTP log for successful pulls.

Table 5-5: Push Distribution States

State	**Initialization**	**Transition**
FINALIZING	Distribution prepare request is complete.	Stored procedure creates distribution job.
QUEUED	Distribution prepare request is complete.	Can transition to RUNNING or EXPIRED.
RUNNING	Processor has picked up queued job and is attempting to push the data file.	Can transition to DELIVERED or FAILED.
DELIVERED	The data file has been successfully pushed to its	No transition from this state.

	destination.	
FAILED	The data file has failed to be pushed to its destination.	No transition from this state without user intervention.
EXPIRED	Subscription time latency threshold has been met.	No transition from this state.

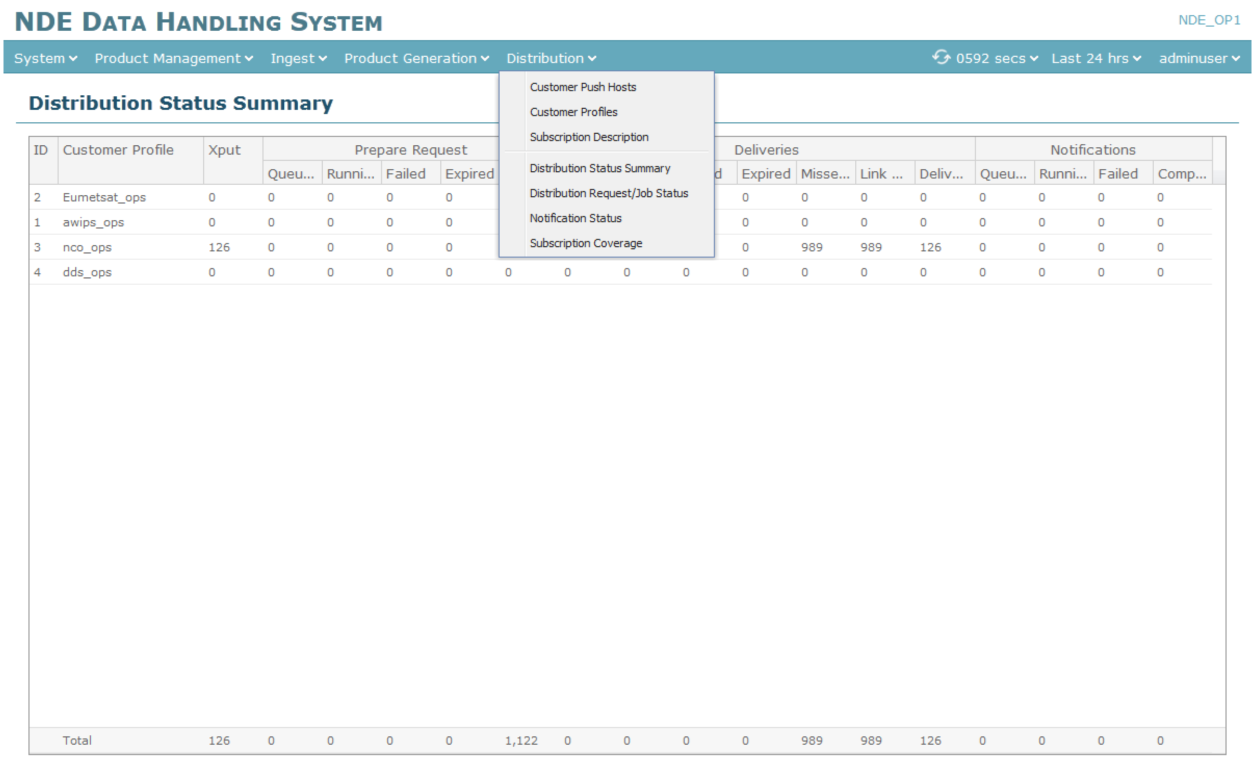
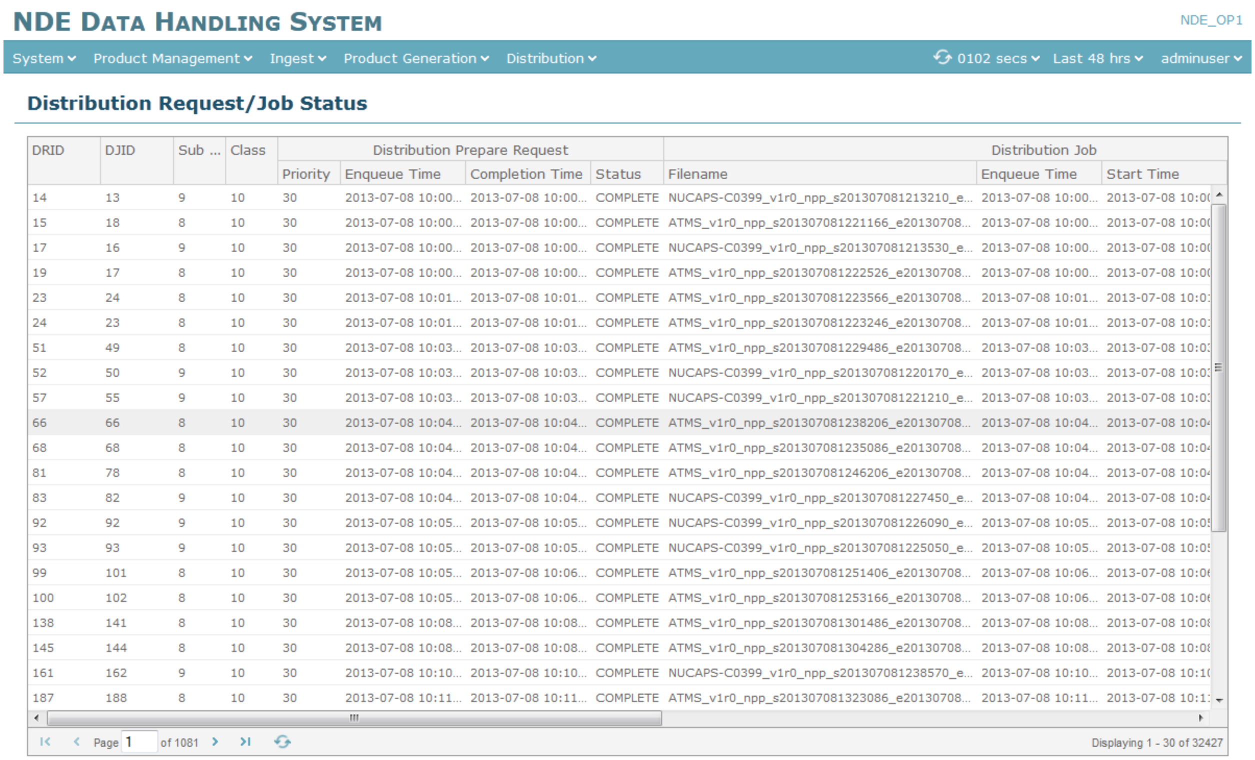


Figure 5-6: NDE Distribution Summary Status showing distribution prepares request and distribution job status summaries for each customer



DRID	DJID	Sub ...	Class	Distribution Prepare Request				Distribution Job		
				Priority	Enqueue Time	Completion Time	Status	Filename	Enqueue Time	Start Time
14	13	9	10	30	2013-07-08 10:00...	2013-07-08 10:00...	COMPLETE	NUCAPS-C0399_v1r0_npp_s201307081213210_e...	2013-07-08 10:00...	2013-07-08 10:0
15	18	8	10	30	2013-07-08 10:00...	2013-07-08 10:00...	COMPLETE	ATMS_v1r0_npp_s201307081221166_e20130708...	2013-07-08 10:00...	2013-07-08 10:0
17	16	9	10	30	2013-07-08 10:00...	2013-07-08 10:00...	COMPLETE	NUCAPS-C0399_v1r0_npp_s201307081213530_e...	2013-07-08 10:00...	2013-07-08 10:0
19	17	8	10	30	2013-07-08 10:00...	2013-07-08 10:00...	COMPLETE	ATMS_v1r0_npp_s201307081222526_e20130708...	2013-07-08 10:00...	2013-07-08 10:0
23	24	8	10	30	2013-07-08 10:01...	2013-07-08 10:01...	COMPLETE	ATMS_v1r0_npp_s201307081223566_e20130708...	2013-07-08 10:01...	2013-07-08 10:0
24	23	8	10	30	2013-07-08 10:01...	2013-07-08 10:01...	COMPLETE	ATMS_v1r0_npp_s201307081223246_e20130708...	2013-07-08 10:01...	2013-07-08 10:0
51	49	8	10	30	2013-07-08 10:03...	2013-07-08 10:03...	COMPLETE	ATMS_v1r0_npp_s201307081229486_e20130708...	2013-07-08 10:03...	2013-07-08 10:0
52	50	9	10	30	2013-07-08 10:03...	2013-07-08 10:03...	COMPLETE	NUCAPS-C0399_v1r0_npp_s201307081220170_e...	2013-07-08 10:03...	2013-07-08 10:0
57	55	9	10	30	2013-07-08 10:03...	2013-07-08 10:03...	COMPLETE	NUCAPS-C0399_v1r0_npp_s201307081221210_e...	2013-07-08 10:03...	2013-07-08 10:0
66	66	8	10	30	2013-07-08 10:04...	2013-07-08 10:04...	COMPLETE	ATMS_v1r0_npp_s201307081238206_e20130708...	2013-07-08 10:04...	2013-07-08 10:0
68	68	8	10	30	2013-07-08 10:04...	2013-07-08 10:04...	COMPLETE	ATMS_v1r0_npp_s201307081235086_e20130708...	2013-07-08 10:04...	2013-07-08 10:0
81	78	8	10	30	2013-07-08 10:04...	2013-07-08 10:04...	COMPLETE	ATMS_v1r0_npp_s201307081246206_e20130708...	2013-07-08 10:04...	2013-07-08 10:0
83	82	9	10	30	2013-07-08 10:04...	2013-07-08 10:04...	COMPLETE	NUCAPS-C0399_v1r0_npp_s201307081227450_e...	2013-07-08 10:04...	2013-07-08 10:0
92	92	9	10	30	2013-07-08 10:05...	2013-07-08 10:05...	COMPLETE	NUCAPS-C0399_v1r0_npp_s201307081226090_e...	2013-07-08 10:05...	2013-07-08 10:0
93	93	9	10	30	2013-07-08 10:05...	2013-07-08 10:05...	COMPLETE	NUCAPS-C0399_v1r0_npp_s201307081225050_e...	2013-07-08 10:05...	2013-07-08 10:0
99	101	8	10	30	2013-07-08 10:05...	2013-07-08 10:06...	COMPLETE	ATMS_v1r0_npp_s201307081251406_e20130708...	2013-07-08 10:06...	2013-07-08 10:0
100	102	8	10	30	2013-07-08 10:05...	2013-07-08 10:06...	COMPLETE	ATMS_v1r0_npp_s201307081253166_e20130708...	2013-07-08 10:06...	2013-07-08 10:0
138	141	8	10	30	2013-07-08 10:08...	2013-07-08 10:08...	COMPLETE	ATMS_v1r0_npp_s201307081301486_e20130708...	2013-07-08 10:08...	2013-07-08 10:0
145	144	8	10	30	2013-07-08 10:08...	2013-07-08 10:08...	COMPLETE	ATMS_v1r0_npp_s201307081304286_e20130708...	2013-07-08 10:08...	2013-07-08 10:0
161	162	9	10	30	2013-07-08 10:10...	2013-07-08 10:10...	COMPLETE	NUCAPS-C0399_v1r0_npp_s201307081238570_e...	2013-07-08 10:10...	2013-07-08 10:1
187	188	8	10	30	2013-07-08 10:11...	2013-07-08 10:11...	COMPLETE	ATMS_v1r0_npp_s201307081323086_e20130708...	2013-07-08 10:11...	2013-07-08 10:1

Page 1 of 1081 Displaying 1 - 30 of 32427

Figure 5-7: NDE Distribution Prepare Requests and Distribution Job Statuses for every job

5.4.2. Science Maintenance

Product quality monitoring is performed by the OSPO product quality monitoring system and the STAR developers. STAR and OSPO personnel should communicate regularly to discuss potential data quality issues and to formulate and schedule updates to JPSS RR science code.

5.4.3. Library Maintenance

5.4.3.1. Java upgrades

Java upgrades are done on a regular schedule on NDE DHS Linux and AIX machines. As of now (…), the following Java version is installed on DHS Servers:
On Linux

- java version "1.6.0_24"
- OpenJDK Runtime Environment (IcedTea6 1.11.11) (rhel-1.40.1.11.11.el5_9-x86_64)
- OpenJDK 64-Bit Server VM (build 20.0-b12, mixed mode)

On Aix:

- java version "1.6.0"
- Java(TM) SE Runtime Environment (build pap6460sr9-20101125_01(SR9))
- IBM J9 VM (build 2.4, JRE 1.6.0 IBM J9 2.4 AIX ppc64-64 jvmap6460sr9-20101124_69295 (JIT enabled, AOT enabled)
- J9VM - 20101124_069295
- JIT - r9_20101028_17488ifx2
- GC - 20101027_AA)
- JCL - 20101119_01

The Java software vendors for Java installed on Linux and AIX servers are RedHat and IBM respectively. The release numbers for Java software from these vendors do not necessarily correlate to the numbers from Oracle. On PE1 (Operations), only the Java 64-bit Runtime is required and not the full JDK.
On Linux servers, Java updates are managed by ESPC Satellite Server which maintains all the update packages. It connects and downloads the packages available on RedHat Network (RHN) on a daily basis. All NDE DHS Linux servers are registered with ESPC Satellite Server and are told by the Server to update specified packages. Each Linux server then pulls down the specified packages

from the Satellite Server and installs it.

On AIX server, Java updates are managed manually by AIX System Administrator. The versions released by IBM are available at http://www.ibm.com/developerworks/java/jdk/aix/service.html.
Java updates are performed just like any other software updates on AIX. Note that a Tech Level AIX upgrade doesn't perform an update of Java being used by DHS software. All Java updates to be performed on DHS servers must be tested and verified on Dev/Test servers before pushed onto the Operational servers.

5.4.3.2. JBoss SOA Platform Upgrade

JBoss SOA Platform upgrades include patching the existing release or installing a completely new release/version. All patches and Releases are available on http://jboss.com Customer Portal. You need to login to the site with proper credentials and then go to the Downloads page to view all the releases and the patches to any specific release/version.
The current version (as of July 2013) of JBoss SOA Platform installed on DHS servers is:
JBoss SOA 5.1.0.GA
Build Date: 20110211
Patch: SOA 5.1 ESB/jBPM/Messaging/CXF Cumulative Patch

5.4.3.3. Patching

Patching an existing JBoss server involves downloading the patch from the link mentioned above and then following the instructions from the readme file in the patch. Most of the times, the instructions include manually renaming certain existing jar files on servers and copying the same from the Patch into the server. At times, there are instructions on how to edit certain XML files in the server as part of the patching. Overall, it is a manual process and JBoss Administrator can develop a script accordingly based on the instructions in the readme file and run it on each server being patched.

5.4.3.4. Installing New Release/Version

Installing a new release or version of JBoss software is a significant task. The new Releases/Versions are available for download at the jboss.com portal mentioned above. The portal also has all the relevenat documentation that can be downloaded or accessed online. A thorough reading of all the relevant documents is preferable before proceeding with the installation of JBoss software.

Not all versions of JBoss/Java and JBoss/Oracle are compatible to each other. See the link below for a table that lists the JBoss compatibility with various other Software packages.

https://access.redhat.com/site/articles/111863

After the installation of the software, a whole lot of customization is required in order to have JBoss server host and run DHS applications successfully. This customization includes integrating with Oracle backend, clustering, port bindings, security, and data sources, among other things. In addition, appropriate Oracle accounts must be created for the server where it persist a variety of data for its smooth functioning.

The DHS applications need to be tested with the new JBoss server after all the necessary customization is done. If any changes are required in the apps, specific to new Release of the JBoss software installed, they must be done and tested thoroughly.

Once all the JBoss server customization is done and tested in Dev/Test environment, the changes must be captured into CM area. A script must be developed and tested to install/apply these changes onto a vanilla JBoss Server.

When the new JBoss release is ready to be installed into Operations, it should be done in these steps:

1. Since multiple releases of JBoss Server can co-exist on a server, install the new JBoss software in a separate directory (different from the directory in which is current/older JBoss release is installed) on the DHS server
2. Create appropriate Oracle accounts for its backend
3. Change the environment to reflect new JBOSS_HOME
4. Run the script to apply all the changes (customization)
5. Install the appropriate DHS applications.
6. Attempt to start the server.
7. When it is determined that the new JBoss server is running fine and the DHS apps are deployed and are running as expected, then older version of the JBoss software can removed from the server.

It is recommended to upgrade the DHS servers in steps. Since all the PGS, DIST Processors, and Web Portal servers are standalone servers (they are not part of any cluster), it would be easy to upgrade them first. Later, the Ingest and Factory servers can be upgraded at the same time.

5.4.3.5. Science Library Maintenance

NDE maintains specific software libraries for science data manipulation. The primary libraries are the Hierarchical Data Format (HDF5) and network Common Data Format (netCDF4 – Climate Forecast Conventions). These are the primary formats for data ingested and produced by NDE. The HDF5 library requires the compression libraries, SZIP and ZLIB, while netCDF4 requires HDF5 as it is a subset of the format. In addition to the primary libraries, NDE maintains the GRIB (GRIdded Binary or General Regularly-distributed Information in Binary form) library for handling GRIB2 files (e.g. Global Forecast Simulation model files). NDE upgrades the standard set of libraries on an as needed basis which is determined by the system administrator and integration teams. Multiple versions of a library can be maintained but is not preferable. When a new version of a science library is required, the integration team first builds and tests the library in the development environment. The library source code along with build scripts are CM controlled and deployed to the /opt/apps/ots directory on all servers. Algorithms that require non-standard libraries (e.g. HDF4) must deliver those libraries with the Delivered Algorithm Package (DAP).

NDE also maintains scripting languages such as Perl which is an integral part of the NDE DHS and Python which some algorithms require. Maintenance and upgrades of these tools are at the sole discretion of the NDE system administrators and software lead.

5.4.4. Special Procedures

Describe all non-routine maintenance procedures, including how to decide when to implement them. These may include procedures for new satellite implementation, adding a new job, data recovery and modification of the system hardware and/or software. *(Document Object 72,* Integration Programmers*)*

It is described in section 5.4.1 and 5.4.3.

5.4.5. Maintenance Utilities

Provide a listing and description of any programs that are available to a system analyst or operator for looking at data. The use can be for observational, analytical, or troubleshooting purposes. *(Document Object 84,* Integration Programmers*)*

NDE does not monitor scientific data quality. The NDE GUI provides real-time monitoring of the operational environment such as throughput/backlog for ingest, product generation, and distribution, memory and cpu usage, production job failure/success, latency, distribution. Also, the Interactive Data Language (IDL) is installed on the Linux-based processing nodes. IDL is very useful in data visualization, however, no specific tools have been developed by NDE.

5.5. Program Backup

Describe project specific procedures for backing up programs outside the norm of the data center. These procedures should include information on the frequency of backups. *(Document Object 88,* PAL*)*

CBU will be providing the backup for the JPSS RR production system (PE-1) including the data required for processing. Alternate back up for JPSS RR operational system can be the JPSS RR test system that is operating at the NDE. For a mission critical data one would require a hot backup in order to have a fault protected products generation. Requirements for backing up the software and data shall be on a daily basis.

6. TROUBLESHOOTING

6.1. Problem Diagnosis and Recovery

6.1.1. Quality Control Output

The JPSS RR output files have a number of failure codes. The following tables (Table 6-1 to Table 6-12) list the error codes for the various products.

Table 6-1: Cloud Mask Failure Codes.

Cloud Mask Quality Control Codes	
QC_Flag	**Definition**
0	Good
1	Invalid pixel due to space view
2	Invalid pixel due to being outside of sensor zenith range
3	Invalid earth pixel due to bad data (bad or missing 11mm BT or bad/missing clear sky 11 mm BT)
4	Reduced quality Cloud mask (bad 3.9mm pixel)
5	Reduced quality 0.64mm tests
6	Reduced quality due to other bad channels (excluding 0.64, 3.9, or 11 mm)

Table 6-2: Cloud Phase/Type Quality Flags.

Cloud Phase/Type Quality Control Codes		
Bit	**Definition**	**Bit Interpretation**
1	**Overall cloud phase product quality flag** – the overall quality will be set to “low quality” if any of the more specific quality flags listed below are set to “low quality”	0 = high quality 1 = low quality

2	**L1b quality flag** – this will be set to "low quality" if any of the spectral data used in the algorithms is of low quality, based on L1b calibration flags	0 = high quality spectral data 1 = low quality spectral data
3	**Beta quality flag** – this will be set to "low quality" if β_{stropo}(12/11μm), $\beta_{sopaque}$(12/11μm), β_{stropo}(8.5/11μm), **or** $\beta_{sopaque}$(8.5/11μm) fall outside of the 0.1 – 10.0 range	0 = high quality beta calculation 1 = low quality beta calculation
4	**Ice cloud quality flag** – this will be set to "low quality" if the cloud phase was determined to be ice and the ε_{stropo}(11μm) < 0.05	0 = ice cloud determination based on strong radiative signal 1 = ice cloud determination based on weak radiative signal (low quality)
5	**Surface emissivity quality flag** – this will be set to "low quality" if the result of the Low Surface Emissivity (LSE) Test is TRUE and the result of the Overall Opaque Cloud (OOC) Test is FALSE	0 = surface emissivity does NOT significantly impact product quality 1 = surface emissivity significantly impacts product quality (low quality)
6	**Satellite zenith angle quality flag** – this will be set to "low quality" if the cosine of the satellite zenith angle is less than 0.15 (~82 degrees)	0 = satellite zenith angle does NOT significantly impact product quality 1 = satellite zenith angle significantly impacts product quality (low quality)

Table 6-3: Cloud Height Failure Codes.

Cloud Height Quality Control Codes	
QC_Flag	**Definition**
0	Good
1	Invalid pixel due to space view
2	Invalid pixel due to being outside of sensor zenith range
3	Invalid earth pixel due to bad data (bad or missing 11mm BT or bad/missing clear sky 11 mm BT)

4	Invalid due to cloud mask being clear or probably clear
5	Invalid due to missing cloud type
6	Failed retrieval

Table 6-4: DCOMP Failure Codes.

DCOMP Quality Control Codes	
QC_Flag	**Definition**
0	Valid, Quality may be degraded due to snow or sea-ice
1	Valid, Quality may be degraded due to twilight conditions
2	Valid, but degraded quality due to twilight conditions (solar zenith between 65 and 82 degree)
3	Invalid due to cloud-free condition
4	Invalid pixel due to being outside of observation range
5	Invalid pixel due to missing input data
6	Invalid pixel, DCOMP attempted but failed retrieval

Table 6-5: Cloud Phase/Type Quality Flags.

NCOMP Control Codes		
Bit	**Quality Flag Name**	**Cause and effect**
Angle restriction flags		
1	QC_CYCLE_VZA	Viewing Zenith Angle >= 72.0
2	QC_CYCLE	Solar Zenith Angle < 82.0
Ancillary Data Flags		
3	QC_CYCLE_NOCLOUD	Cloud Type indicates it is not a cloud
4	QC_CYCLE_CLOUDTYPE	Cloud Type has an unknown value
5	QC_CYCLE_TCLOUD	Cloud Temperature is < 0.0
No Retrieval Flags		
6	QC_MINERR_WATER_0	No retrieval: Minimum error model for water = 0
7	QC_MINERR_ICE_0	No retrieval: Minimum error model for ice = 0

Valid Retrieval Flags		
8	QC__TWILIGHT_	82.0 <= Solar Zenith Angle < 90.0
9	QC_CTWATER_NCOMPICE	Cloud Type = water, NCOMP preferred phase = ice
10	QC_CTICE_NCOMPWATER	Cloud Type = ice, NCOMP preferred phase = water
11	QC_CTMIX_NCOMPWATER	Cloud Type = mixed, NCOMP preferred phase = water
12	QC_CTMIX_NCOMPICE	Cloud Type = mixed, NCOMP preferred phase = ice
13	QC__NCOMPWATER	Cloud Type = supercooled, NCOMP preferred phase = water
14	QC__NCOMPICE	Cloud Type = supercooled, NCOMP preferred phase = ice

Table 6-6: Aerosol Detection Failure Codes.

Aerosol Detection Quality Control Codes				
Byte/Bit	**Quality flag name**	**Meaning**		
		1bit: 0 (default)	**1**	
		2bit: 00 (default)	**01**	**11**
0	QC_SMOKE_DETECTION	Determined (good)	not Determined(bad)	
1	QC_DUST_DETECTION	Determined(good)	not Determined(bad)	
2-3	QC_SMOKE_CONFIDENCE	Low	Medium	High
4-5	QC_DUST_CONFIDENCE	Low	Medium	High
6	SPARE			
7	SPARE			

Table 6-7: Aerosol Optical Depth Failure Codes.

Aerosol Optical Depth Quality Control Codes			
Byte	**Bits**	**Quality Flag Name**	**Meaning**
1: Input Geometry Quality Flag	0	QC_INPUT_LON	0: valid longitude (-180 - 180°) 1: out-of-range longitude
	1	QC_INPUT_LAT	0: valid latitude (-90 - 90°) 1: out-of-range latitude
	2	QC_INPUT_ELEV	0: valid elevation (-2 – 10 km) 1: out-of-range elevation
	3	QC_INPUT_SOLZEN	0: valid solar zenith (0 - 90°) 1: out-of-range solar zenith
	4	QC_INPUT_SATZEN	0: valid satellite zenith (0 - 90°) 1: out-of-range satellite zenith
	5	QC_INPUT_SOLAZI	0: valid solar azimuth (0 - 180°) 1: out-of-range solar azimuth
	6	QC_INPUT_SATAZI	0: valid satellite azimuth (0 - 180°) 1: out-of-range satellite azimuth
2: Input Ancillary Data Flag	0 1	QC_INPUT_TPW	00: constant TPW data (2.0 cm) 01: valid TPW data from ABI retrieval (0-20 cm) 10: valid TPW data from model (0-20 cm)
	2 3	QC_INPUT_OZONE	00: constant ozone data (0.35 atm-cm) 01: valid ozone data from ABI retrieval (0.0 – 0.7 atm-cm) 10: valid ozone data from model (0.0 – 0.7 atm-cm)
	4	QC_INPUT__PRES	0: valid model surface pressure (500 – 1500 mb) 1: constant surface pressure (1013 mb)
	5	QC_INPUT_HGT	0: valid model surface height (-2 – 10 km) 1: constant surface height (0 km)
	6	QC_INPUT_WSP	0: valid model surface wind speed (0 – 100 m/s) 1: constant surface wind speed (6 m/s)
	7	QC_INPUT_WDR	0: model surface wind direction (0° - 360°) 1: fixed surface wind direction (90°)
3: Input Reflectance Data Flag	0	QC_INPUT_REFL_CH1	0: valid ABI reflectance in band 1 (0 – 1) 1: out-of-range ABI reflectance in band 1
	1	QC_INPUT_REFL_CH2	0: valid ABI reflectance in band 2 (0 – 1) 1: out-of-range ABI reflectance in band 2
	2	QC_INPUT_REFL_CH3	0: valid ABI reflectance in band 3 (0 – 1) 1: out-of-range ABI reflectance in band 3
	3	QC_INPUT_REFL_CH5	0: valid ABI reflectance in band 5 (0 – 1) 1: out-of-range ABI reflectance in band 5
	4	QC_INPUT_REFL_CH6	0: valid ABI reflectance in band 6 (0 – 1)

			1: out-of-range ABI reflectance in band 6
4: Critical Path Flag	0	QC_CLOUD_MASK	0: clear sky 1: cloudy sky
	1	QC_RET_SCENE	0: over-land algorithm is used 1: over-water algorithm is used
	2	QC_LAND_TYPE	0: vegetation 1: soil
	3	QC_LAND_BRISFC	0: dark surface 1: bright surface
	4	QC_LAND_SNOW	0: no snow contamination 1: with snow contamination
	5	QC__WATER_GLINT	0: no sunglint contamination 1: with sunglint contamination
5: AOD Product Quality Flag	0	QC_RET	0: AOD is retrieved 1: AOD is not retrieved
	1	QC_RET_EXTRP	0: interpolation within LUT AOD range 1: extrapolation of AOD used
	2	QC_OUT_SPEC	0: within F&PS specification range 1: out of F&PS specification range
	3	QC_LOWSUN	0: solar zenith angle not larger than 80° 1: solar zenith angle larger than 80°
	4	QC_LOWSAT	0: local zenith angle not larger than 60º 1: local zenith angle larger than 60º

Table 6-8: Volcanic Ash Detection Quality Flags.

Volcanic Ash Detection Quality Control Codes			
Byte	**Bit**	**Name**	**Values**
1	**1**	Overall QF	0 – High Quality **1 – Low Quality**
1	**2**	Invalid Data QF	0 – High Quality **1 – Low Quality**
1	**3**	Local Zenith Angle QF	**0 – High Quality** 1 – Low Quality
1	**4-6**	Ash Single Layer Confidence QF	0 – High 1 – Moderate 2 – Low 3 – Very Low **4 – Not-Ash**
1	**7-8**	Spare	n/a
2	**1-3**	Ash Multi Layer Confidence QF	0 – High 1 – Moderate

			2 – Low 3 – Very Low **4 – Not-Ash**

Table 6-9: Volcanic Ash Retrieval Quality Flags.

Volcanic Ash Retrieval Quality Control Codes			
Byte	**Bit**	**Name**	**Values**
1	**1-2**	Retrieval Status	0 - Successful 1 - Failed 2 - Not Attempted
1	**3-4**	T_{cld} QF	0 – High Quality 1 – Medium Quality 2 – Low Quality
1	**5-6**	ε_{cld} QF	0 – High Quality 1 – Medium Quality 2 – Low Quality
1	**7-8**	β(12/11μm) QF	0 – High Quality 1 – Medium Quality 2 – Low Quality
2	**1-4**	Ash Particle Size	0 – < 2 µm 1 – ≥2 – < 3 µm 2 – ≥3 – < 4 µm 3 – ≥4 – < 5 µm 4 – ≥5 – < 6 µm 5 – ≥6 – < 7 µm 6 – ≥7 – < 8 µm 7 – ≥8 – < 9 µm 8 – ≥9 – < 10 µm 9 – ≥ 10 µm 10 - invalid

Table 6-10: Snow Cover Retrieval Quality Flags.

Snow Cover Retrieval Quality Control Codes	
Bit	**Name** **Values**
0	no-data value in band data
1	missing data in band data

2	modeled cloudy
3	water
4	solar zenith angle less than 0 or greater than MAX_SOLAR_ZENITH_ANGLE
5	sensor zenith angle less than 0.0 or greater than MAX_SENSOR_ZENITH_ANGLE
6	bad metadata or ancillary data
7	N/A

Table 6-11: Ice Concentration Retrieval Quality Flags.

Ice Concentration Retrieval Quality Control Codes		
Quality Flag Name	**Variable Type**	**Definition**
QC_Flags	LONG	Quality Control Flags
Tot_QACat01	LONG	Total number of pixels with QA category 1 (Normal or optimal)
Tot_QACat02	LONG	Total number of pixels with QA category 2 (Uncertain or suboptimal)
Tot_QACat03	LONG	Total number of pixels with QA category 3 (Non-retrievable)
Tot_QACat04	LONG	Total number of pixels with QA category 4 (Bad data)

Table 6-12: Ice Thickness And Age Retrieval Quality Flags.

Ice Thickness And Age Retrieval Quality Control Codes		
Quality Flag Name	**Variable Type**	**Definition**
QC_Flags	LONG	Quality Control Flags
Tot_QACat01	LONG	Total number of pixels with QA category 1 (Normal or optimal)
Tot_QACat02	LONG	Total number of pixels with QA category 2 (Uncertain or suboptimal)
Tot_QACat03	LONG	Total number of pixels with QA category 3

		(bad or missing)
Tot_QACat04	LONG	Total number of pixels with QA category 4 (Non-retrievable)

6.1.2. Error Correction

All NDE Data Handling System (DHS) applications are Java-based and are built using the Enterprise Service Bus (ESB) framework provided by JBoss. NDE DHS consists of five ESB applications which perform the core tasks for ingest, product generation and product distribution. Each Jboss server logs all messages, warnings, and errors in persistent log files that are refreshed every 24 hours and stored indefinitely on each Jboss server. The applications handle all exceptions with Java catch/try handlers and write appropriate messages to the log files. In addition, JBoss has built in exception handling for the ESB code. The table below lists NDE DHS application messages, possible reasons, and possible resolutions.

In the event of an algorithm error, the entire working directory for the algorithm is compressed into a forensics file and moved into the forensics directory on the local processing node for later analysis. In addition, NDE DHS scans algorithm log files for errors and warnings and logs them in the database (PRODUCTIONJOBLOGMESSAGES table).

Table 6-13: NDE Exception Handling

Message	Subsystem Application	Reason	What to do?
executeChecks : Exception thrown	Ingest	An error occurred during the validation of a product file	Check to make sure h5dump (or

		being ingested. Error could be during metadata extraction or when checksums don't match.	other relevant MD extractors) is installed on the server. Make sure (manually) that the file is not corrupt. If it is determined to be corrupt, report it to the Data Provider.
CRC checksums did not match	Ingest	Checksum received in PAF did not match with the one generated by the App for the file being ingested	Notify Data Provider of the problem.
validateFilenamePattern : Unable to retrieve PAF data from message	Ingest	PAF file (*.crc, etc.) was corrupt or empty	Same as above.
executeChecks : Filename is NULL	Ingest	The PAF file (.crc, etc) is incomplete and doesn't have a File name in it	Same as above.
HeartbeatUpdateAction:: Error while initializing variables	Ingest	LoopName attribute may not have been defined in the Action definition (in	Check to make sure LoopName attribute in

		jboss-esb.xml)	properly defined in the ESB configuration file (jboss-esb.xml) of the Ingest app.
HeartbeatUpdateAction::updateHeartBeat:: Exception caught while updating heartbeat	Ingest	Ingest app is unable to update the DB with the Heartbeat reading due to some SQL/DB error.	Check to make sure DB is functioning normally.
persistMetadata : Exception thrown, moving file from landing zone to failed dir	Ingest	Ingest app failed to extract metadata and/or persist it in database	Same as in row 1 above.
homeFile : Exception thrown routing message to queue	Ingest	JMS bridge is not working. This bridge connects Ingest and Factory apps and Ingest uses it to send messages over it to Factory app.	Check to make sure there are no errors during Ingest App deployment in JBoss server log. Check to make sure at least one Factory server is running.

validateFilenamePattern : Unable initialize IR/Resources objects	Ingest	Ingest app unable to retrieve rows from IngestRequestLog table for a given IR_ID. May be a DB error or the IRB and IRL tables must be out of sync.	Make sure the DB is functioning normally. If it is, a TAL must determine the cause of the inconsistencies between IRL and IRB table.
validateFilenamePattern : Exception occured during validation of the product file	Ingest		
validateProduct : Product File Unknown	Ingest	The product file being ingested is of unknown product. There DHS database doesn't have a Product registered with the same pattern as the product file being ingested.	Make sure DHS database is properly registered with all the products. If the products registration fine, then we may have received a file that is not associated with any known registered products.

Rejected as duplicate	Ingest	The File being ingested has already been ingested.	Make sure no duplicate files are sent to Ingest (into incoming_input).
execute :Unable to extract metadata	Ingest	H5dump or other metadata extractor has failed to extract metadata from the file. File may be corrupt or in a format that is incompatible with known format(s).	
homeFile : Exception caught (moving file);	Ingest	Ingest app unable to move the file to Products 'home' directory after successful MD extraction and persistence.	Make sure SAN (Product Home folder) is OK.
validateDataFilePattern : Product Unknown	Ingest	The product file being ingested is of unknown product. There DHS database doesn't have a Product registered with the same pattern as the product file being ingested	Make sure DHS database is properly registered with all the products. If the products registration fine, then we may have received a file that is not

			associated with any known registered products.
validateDataFilePattern : Multiple product dfs found	Ingest	Ingest app found there are multiple products registered in DB that has same filename pattern as the file being ingested.	Correct the Products registration in DB. Make sure there the filename pattern in unique in ProductsDescriotion table of DHS DB.
Bridge Failed to set up connections	Ingest	JMS bridge between Ingest and Factory is not correctly deployed.	Make sure the specified Factory servers in the bridge configuration are up and running.
createJobs: Exception detected	Dist Factory		
Error while initializing variables	Dist Factory, PGS Factory, PGS Processor,	LoopName attribute may not have been defined in the Action definition (in jboss-esb.xml)	Check to make sure LoopName attribute in properly defined

	DIST Processor		in the ESB configuration file (jboss-esb.xml) of the app.
Error while checking active flag	Dist Factory, PGS Factory, PGS Processor, DIST Processor	Apps unable to find the relevant loop 'Active' flags in ConfigurationRegistry table.	Make sure there are 'Active' flags defined in the ConfigurationRegistry table for all the loops in various apps.
getCompletedJobSpecs : Exception occurred	PGS Factory	App unable to execute stored proc (SP_GET_COMPLETED_JOB_SPECS) to retrieve the completed ProductionJobSpecs. May be a DB error.	Make sure the Stored Proc is properly registered in the DB and DB is functioning normally.
getJismoContents - Exception occurred	PGS Factory	App unable to execute stored proc (SP_GET_JISMO) to retrieve Job Spec Inputs.	Same as above.
getFileAccumulationContents - Exception occurred	PGS Factory	App unable to execute stored proc (SP_GET_FILE_ACCUMULATION) to retrieve the File	Same as above.

		accumulation thresholds for a ProductionJobSpec.	
createJobSpecInput : Exception detected	PGS Factory	App unable to create (insert into DB) a Job Spec Input for a ProductionJobSpec.	Make sure DB is OK. If it is, then follow the Exceptin stack trace to see which specific SQL error caused this failure.
updateJobStatus : Exception while updating job status	DIST,PGS Factory	App unable to update the Job status of a ProductionJob	Same as above.
generateChecksum : Exception caught	DIST ,PGS Factory	Apps unable to compute a checksum for a specified file. Or the app may be unable to create a checksum file.	Make sure the file in question (for which checksum is being computed) is valid and exists. Also, make sure relevant jar files (used for checksums) exists in Jboss environment.

			Also, make sure the folder in which the checksum file is created exists.
generateMD5Checksum : exception encountered	DIST Factory	Same as above for MD5 checksum.	Same as above.
generateCRC32Checksum : exception encountered	DIST Factory	Same as above for CRC checksum.	Same as above.
CompressionBean::compressFile : Exception caught	DIST Factory	App is unable to compress the file(s).	Make sure file(s) being compressed exists and are valid. If they are, follow the exception's stack trace in the log file and determine the exact cause of the Exception.
createNotificationRequest : Exception caught creating NR	DIST Factory	App unable to create (insert into DB) a NotificationRequest for a	Make sure the DB is functioning normally. If it is,

		given Distribution Job after it is executed.	follow the Exception's stack trace to determine the root cause of the Exception.
NO Notification Request created for job id. The associated subscription has NO notification type.	DIST Factory	The relevant subscription (for which a job has been completed) has no Notif type defined.	Make sure a Notif type is defined for subscription in question. See appropriate documentation (User Manual) for different types of valid Notifs.
DistJobStatus : Exception caught	DIST Factory	The App is unable to update the status of a specified Distribution job. Usually, these updates happen as Job is moved from one state to another.	Make sure the DB is functioning normally. If it is, follow the Exception's stack trace to determine the root cause of the problem.
UpdateJobStatusActio	DIST Factory	Same as above but for a	Same as above.

n::updateNotifJobStatus : Exception occurred		NotificationJob.	
CopyFilesAction::copyInputFiles : Exception occurred	PGS Processor	The app is unable to copy the files (may be due to problems with source and/or target folders) to PGS Job staging area. This area is first populated with the relevant input files required for executing the algorithm.	Make sure the files in question exist in Product 'home' directory (from which files are being copied from). Make sure the target directory can be created OK. If everything looks good, then follow the Stack trace of the exception logged to determine the root cause of the problem.
Exception scnning PSF file for output files	PGS Processor	The app is unable to scan the PSF file and/or create the list of output files from PSF file. PSF (Process Status File) file is generated	Make sure the PSF file is properly generated by Science

		by the Science algorithm. It lists all the Product files that have been generated by Science algorithm and are required to be ingested.	Algorithm. If it is, then follow the Exception Stack trace to determine the root cause of error/exception.
JobBoxDataProviderAction::process() exception occurred	PGS Processor	This is an internal app error message logged when it is unable to manage the Job Box data in the memory. Job Box data in memory holds information on the current load (running jobs) on the processor server host. The error occurs when it is unable to update this data.	Follow the Exceptions stack trace to determine the root cause of the error. In most cases, it may require a server restart to resolve it.
PerformErrorHandlingAction::performErrorHandling : Exception scanning <logfilename>	PGS Processor	This error occurs when the app is unable to scan (and persists them in DB) the errors/messages from the error/log file generated by the Science Algorithm.	Make sure the Science Algorithms are generating the error/log files properly and DB is functioning normally.
Exception in	PGS Processor	The app is unable to	Make sure the

runProdJob Action		execute an algorithm for whatever reason.	algorithm is properly installed on the processor. If it is, follow the exception's stack trace to determine the root cause of the problem.
StartProcessingNode Action::runLoop:: Exception retrieving jobs list	PGS Processor	This occurs when the app is unable to create a list of various Job class/size type and their number from JobBoxDataProviderSvc service. This info is used to retrieve a specific number of jobs of each class/size type.	Make sure the Job Box data is properly defined for the processor node. If it is, then an internal error might have caused this and may require a server restart to resolve it.
Exception occured in service RunJobSvc: UpdateJobStatus	PGS Processor	The app is unable to update the status of a Production job. The status is updated many times as it changes from one state to another.	Make sure the DB is functioning normally. If it is then follow the exception's stack trace to

			determine the root cause.
Exception retrieving pid for algorithm command	PGS Processor	The app is unable to retrieve the PID (Process ID) of the algorithm process launched.	Determine the root cause by following the Exception's stack trace.
Exception invoking stored proc LIB_SP_REQUESTJOB	PGS Processor	The app is unable to execute the stored proc (SP_REQUESTJOB) to retrieve Productionjobs of a specific class/size type.	Make sure stored proc is registered in DB and DB is functioning normally.
notifySubscriber : FTPS Push Return not OK	DIST Processor		
runLoop:: Exception retrieving jobs list	DIST Processor	The app is unable to execute stored proc (SP_RETRIEVE_JOBS) to retrieve distribution jobs from DB.	Make sure stored proc is registered in DB and DB is functioning normally.
Exception invoking FTPS Service	DIST Processor	This error occurs when messages are not able to be delivered to service that	Make sure the Dist processor app is deployed

		actually does the FTPs pushes.	without any errors. If needed, redeploy the app.
runLoop : Exception invoking Notification Service	DIST Processor	This error occurs when messages are not able to be delivered to the service that notifies subscribers of the data availability.	Same as above.
updateJobBoxConfigAfterRun : Exception occurred	DIST Processor	This error occurs when the app is unable to update its internal Job Box Data after a job is executed.	
Exception while updating Job Status to Fail	DIST ,PGS Processor	The app is unable to update the Job Status to Fail in DB.	Make sure database is functioning normally. If it is, then follow the stack trace in the log file to determine the root cause of exception.
getDistNodeId() : Exception occurred	DIST Processor	The app is unable to get the node id of the Distribution	Make sure Distribution node

		processor. Usually occurs when host on which the app is deployed and running is not configured as a distribution node.	is properly defined in the DB.
transferFiles() : Connection refused	DIST Processor	This occurs when a FTPs connection is refused by the subscriber's external host during the ftps transfers of the files.	Make sure external host data in DB is correct and that the host is able to receive ftps connections with the specified authentication. If everything is registered correctly in DHS database, then contact the concerned person on the subscriber's side (and NDE side) and report the issue.
FtpsPushBean::transferFiles : Could not	DIST Processor	Same as above	Same as above

connect to server			
transferFiles : Login was unsuccessful	DIST Processor	The app is unable to login successfully into subscriber's ftps server.	Same as above
transferFiles : Ftps Transfer failed	DIST Processor	FTPS transfer of files onto subscriber's ftps server has failed.	Same as above.
transferFiles : Ftps Rename failed	DIST Processor	The rename (post-transfer) of files failed on subscriber's ftps server	Same as above.
transferFiles : FTPConnectionClosed Exception	DIST Processor	The ftps connection to subscriber's server (where files are being pushed to) has been closed unexpectedly.	Same as above.
transferFiles : CopyStreamException	DIST Processor	The error is logged when the data files are unable to be steamed (transferred) onto the subscribers ftps server. Usually occurs when data started to flow over ftps and then stopped for some reason before entire file is uploaded.	Same as above.
transferFiles :	DIST	This error is logged when	Same as above

IOException	Processor	IO problems occur during ftps transfer on the subscriber's server where files are being pushed to.	
transferFiles : Unknown exception	DIST Processor	The error is reported when any unknown error caused ftps transfers to fail.	Same as above
notifyEmail : Exception occurred	DIST Processor	The app is unable to notify subscribers via email that the data is available.	Make sure Email (to send emails out) is properly configured in Jboss. Make sure the email address is valid.
notifySoap : Exception occurred	DIST Processor	Same as above but for notifying via SOAP protocol.	Make sure the URL where SOAP message is being addressed to is correct in DB. Make sure there are no networking issues.

6.1.3. Problem Diagnosis and Recovery Procedures

This section discusses error messages and procedures for only those errors produced by the JPSS RR system software itself. The error output from the JPSS RR system is not coded to numerical values. This makes it easier for developers to understand the meaning of the error in the program being developed without having to search for a numerical code in a table. Also, error messages can be more easily tailored to a given program.

Errors not described in section 6.1.3.1 and 6.1.3.2 may occur in the event of hardware failure. The manifestation of hardware problems in the system operation can be very unpredictable. Operators and the technical operation staff should consider hardware problems as a possibility if large numbers of granules/files fail and produce messages unlike those discussed below or make reference to the UNIX operating system files. If a hardware failure is suspected, the technical operations staff should contact both OSDPD system administrators and the JPSS RR development team. The system administrators can check the machine logs to compare with the failure times presented by the operations staff. Operations should switch to the backup machine. Meanwhile, the development team can provide assistance with system recovery.

6.1.3.1. High-Level Errors

Table 6-14 contains all the high-level errors that operators may encounter while checking the log file of the driver script. Each error message lists the script that produced the error as $SCRIPT (such as JRR_PREPROCESS_SDR.pl JRR_PRODUCT_CLOUD_MASK.pl , ..), the number of arguments as $NUMBER, the argument as $ARG associated with that error, the number of orbit as $NUMOBT, the SDR file name time string as $TIME, the directory as $DIR, or file/program names as $FILE, $TARGET, $PROGRAM.

The context string for these error messages is “ERROR in”. The NDE DHS will detect JPSS RR error messages using this context string and handle the errors accordingly.

Table 6-14: JPSS RR Error Messages (High-Level)

Error Message	Explanation	Action
ERROR in $SCRIPT: $NUMBER argument is/are required to run this script -- $ARG	The number of argument is not correct when this error happens. $ARG is the description of the arguments.	If this error occurs, this error message may be forwarded to the JPSS RR development lead.
ERROR in $SCRIPT: file $FILE does not exist.	The file or program $FILE does not exist in the current working directory or in the expected path. It may have a wrong path in the PCF file, have never been created, or was deleted prematurely.	If this error occurs, this error message may be forwarded to the JPSS RR development lead.
ERROR in $SCRIPT: directory $DIR does not exist.	The directory $DIR may not exist or have a wrong path in the PCF file. Or there was an unknown error in the OS.	If this error occurs, this error message may be forwarded to the JPSS RR development lead.
ERROR in $SCRIPT: failed to open file $FILE	The script failed to open the file or create a new file $FILE. It may have a wrong path in the PCF file, have never been created, or was deleted prematurely. Or there was an unknown error in the OS.	If this error occurs, this error message may be forwarded to the JPSS RR development lead.
ERROR in $SCRIPT: running $PROGRAM Failed	There was an error running the program or script $PROGRAM. A log file will be appended providing additional details. If the error is from the running	If this error occurs, this error message may be forwarded to the JPSS RR development lead. If the error is from the low-level error, then

	of a program, then check the low-level error	check the associated action should be taken.
ERROR in $SCRIPT: Could NOT make the link.	Failed to make a soft-link to the file. The file may not exist or have a wrong path. Or there was an unknown error in the OS.	If this error occurs, this error message may be forwarded to the JPSS RR development lead.
ERROR in $SCRIPT: ncdump does NOT work correctly	The utility ncdump does not work correctly. The utility may not installed or use a wrong version. Or there was an unknown error in the OS.	

For all the possible error messages listed in Table 6-14, there should be no need for the operator to rerun processing. Log files contain the standard output of the executable that was run inside the calling scripts. This output will contain low-level error messages and output that may only be interpretable to the JPSS RR development team.

6.1.3.2. Low-Level Errors

Section 6.1.3.1 outlined high-level error messages returned from scripts. The origins of many of these errors, however, are from low-level sources. When this is the case, log files are added to these high-level messages containing the output of the low-level errors. These errors and their explanation are shown in Table 6-14. Each error message lists the program that produced the error as $PROGRAM. The low-level error message is described in section 8 Appendix 2. Most of what needs to be done in response to these errors depends on the type of response suggested by the associated high-level errors originating from the calling script.

6.1.4. Data Recovery Procedures

NDE provides a product generation subsystem with redundant and highly available processing nodes. If a single node fails, there are multiple nodes backing up algorithm execution. Since nodes are picking production jobs off of the queue, any failures will have zero impact to operations (processing nodes are not clustered). NDE DHS also has the capability to requeue any jobs that may have failed. This can be done through the GUI.

The table below lists various NDE data recovery scenarios.

Table 6-15: NDE Data Recovery Procedures

Bad/Lost Data Scenario	Reason	Recovery Procedures
PGS Jobs failed and unable to generate products for Distribution.	The jobs can fail for genuine reason because of bad input data. However, if they are failed for other reasons such as any technical issues in DHS (temporary algo errors, IO/resource errors on server, etc.), we need to be able to rerun those jobs to generate products for Distribution.	Check the Jboss server.log file for the reasons the jobs have failed. If jobs failed because of any technical issues in DHS (and there are no data issues), the jobs can be requeued so that they are run again and produce products for Distribution. To requeue jobs: 1. Log onto

		DHSPortal and go to the Production Job Summary page. 2. Select the jobs to requeue and change their status to 'QUEUE' and save them. 3. The jobs should be run again and produce the data for Distribution
DIST Jobs failed and products are not pushed to subscribers.	The Jobs can fail because of any temporary network issues or due to any DHS technical issues or genuinely due to problems on subscriber's server. Once the problem is resolved, we might want to push the data again to subscribers.	Requeue the DIST jobs as follows so jobs can be rerun and files pushed again. 1. Log on to DHSPortal and go to Distribution Summary page 2. Select the Jobs to requeue and change their status to 'QUEUE' and save them. 3. The jobs should be run again and files should be pushed to subscribers.
Orphaned data files in Landing Zone.	The data files that arrive into Landing Zone are accompanied by a companion files (PAF file). These companion	To regenerate a companion file, follow these steps. 1. TBD. Refer the script to produce the crc

	files trigger the process of Ingest. In cases when throttle (which picks up, reads, and renames the file) is terminated abruptly for any technical issues while it is processing PAF files, the data files (associated with those PAF files) can be left in LZ until a companion is found.	and other PAF files manually.

Algorithms are assigned across multiple nodes. If a node fails there is enough capacity so that an alternate node can continue the processing without operator intervention. If the entire system fails then job specifications can be requeued and restarted. If necessary the database will failover to the backup database.

6.1.5. Program Recovery Procedures

DHS Recovery includes Systems recovery and Data recovery. Systems Recovery includes recovering Servers such as Jboss, Throttle, or other servers when they stop completely or functioning abnormally for whatever reason. Data Recovery mainly involves changing the status of DHS data in database or on SAN/RAID, etc. so data gets reprocessed and recovered. Data recovery also includes generating certain files manually (via scripts) on SAN for reprocessing.

The table below lists different ways servers can stop functioning and the steps to recover from those conditions. This is based in the experiences from Dev/Test runs over the years.

Table 6-16: NDE Server Recovery

Server Behavior	Reason	Recovery Procedures
Jboss sever stopped abruptly and is not responding to any commands. Log file show no apparent errors or exception messages.	Network issues and/or I/O issues on SAN/Disk can cause this JBoss behavior among other things. Sometimes, it recovers by itself and returns to normal functioning. Any internal deadlocks within the Jboss server can also cause this.	If no auto-recovery, follow these steps. 1. Take the thread dump of the Jboss server (via Jboss Admin Console, see separate doc for procedures on how to do it) so it can be examined by JBoss Admn and/or Support to see if any internal deadlocks 2. Save the server.log, boot.log files from Jboss log folder to be sent to Support for debugging. 3. Consult the System Administrator to see if there are any network and/or IO issues occurred at the time of server stoppage (the timestamp of the last message logged in the server.log file shows when the last activity occurred). 4. Attempt to issue the shutdown command (manageJbossServer.pl) to

		shutdown the server. If it is not responding to the shutdown command, use the OS ‘kill’ command and restart the server.
A Jboss server in a cluster of Jboss servers went down and the surviving JBoss server is running very slowly or pausing frequently	Certain Messaging and/or Clustering issues can cause this behavior occasionally. Don’t have any particular reason why this is happening but Jboss Support suggested to upgrade it to new Version of the software.	Follow these steps to recover from this: 1. If the server in the cluster that went down is available, restart it. Most of the time, this should resolve the problem. 2. If server that went down is unavailable, restart the surviving server that is having problems. Most of the times, this should resolve the problem. 3. If the above steps did not help, need help from Jboss Admin who can perform some cleanup in JBoss backend DB and restart the server.
Jboss server looses Oracle/DB connections and doesn’t recover from it and fails to do any processing	When there are DB connection issues, Jboss server sometimes can’t recover from it. Some of the Connection instances in the pools it maintains have become stale and	Restart the Jboss server so connection instances are recreated in all the pools. A software fix to rectify this behavior is purposefully

	throws errors when they are used.	not implemented as this may affect the server performance. We may (after enough testing) implement this in a future release.
Jboss Server stops functioning under load and throws OutOFMemory Exceptions in log file	All Jboss servers currently have been allocated enough heap size limits such that a backlog of 2-3 days can be handled without any resource issues. However, if backlog is significantly large and if there is shortage of memory and/or other resources on the server host, we may see this behavior.	Increase the heap size limits for Jboss server and restart it. Follow Jboss Documentation on how to do this. Also, make sure that there are no leaks in server (in the DHS apps deployed in it) and Garbage collection settings are set optimally so as to avoid such issues. See relevant Java documentation on GC settings.
Ingest Throttle is unable to 'pump' any files into DHS database for processing. As a result, Ingest stops cataloging.	Throttle will stop pumping new files into IngestRequestBuffer (a DB table where Jboss/Ingest App picks'em up and validates/cataloges) when there are already certain number of data rows present in that table. Under normal conditions,	Remove the rows from IngestRequestBufer table manually. This will create room for Throttle to put new files into that table for validation and cataloging of new files.

	these rows are deleted by Ingest app after it catalogs the files referenced by them. If there are any DB issues, those rows might not get deleted from the table thus stopping the throttle from pumping new rows into it.	
FTP server (supporting the Pull subscribers who pull files) sometimes stops logging because of IO issues. This can cause DIST job status updates not happen eventhough it is pulled by the subscribers, which can mess up the latency measurements.	IO or Space issues can cause FTP server to stop logging into the /var/log/xferlog.	System Administrator should resolve the IO/space issues and restart the FTP server

6.2. Application Shutdown and Restart

6.2.1. Application Shutdown Procedures

The major components that comprise the DHS system include JBoss servers, Monitoring Utilities, Ingest Throttle, Cleanup, and Logs Scanners. In order to functionally shutdown DHS System, the components that need to be shutdown are Jboss Servers and Ingest Throttle process. Shutting down of these components effectively shuts down the whole DHS system. However, if a shutdown is required for an installation of any new release of DHS software, all the components mentioned above must be shutdown before installing a new release.

The first application to shutdown is the Ingest Throttle. To do so, use the manageConfigurationRegistry.pl script in /opt/app/nde/NDE_OP1/common and update the [host]_IngestThrottle_ActiveFlag to 0 (zero).

Next, shutdown the JBoss servers; This involves stopping the loops and servers themselves in a systematic fashion. For each server the loops are stopped first and then the server itself.

To stop the loops, use the manageConfigurationRegistry.pl script in /opt/app/nde/NDE_OP1/common and update all xxxxxxLoop_ActiveFlag parameters to 0 (zero). The loop flags are listed below:

[host]_FTPDownloaderLoop_ActiveFlag
[host]_ProcessIRLoop_ActiveFlag
[host]_ProcessPJSLoop_ActiveFlag
[host]_ProcessDPRLoop_ActiveFlag
[host]_ProcessSubLoop_ActiveFlag
[host]_ProcessNRLoop_ActiveFlag
[host]_RetrievePJLoop_ActiveFlag
[host]_RetrieveDJLoop_ActiveFlag
[host]_RetrieveNJLoop_ActiveFlag

To stop the JBoss servers, open PuTTY sessions to the following machines :

n25rpn[01-04] - Ingest, PGS/DIS Factory
n25rpn[05-06] - Linux-based Processor Nodes (PGS)
n07apn[2-7] - AIX-based Processor Nodes (PGS)
n26rpn[01-04] - Distribution Processor Nodes (DIS)
n26rpn[05-08] - Data Consumer Portal
n25rpn[07-08] - Data Handling System Portal (internal)

Execute the following commands on each machine, in the following machine order: Portals, Factory, Ingest, PGS Processor, DIS Processor

[host](JBoss)> cd $JBOSS_HOME/bin

[host](JBoss)> ./manageJBossServer.pl stop nde_op1

Stopping production rules from running is a simple matter of turning the active flag to "off". This can be done by running a script: /opt/apps/nde/NDE_OP1/pgs/managePRFlags.pl: interactive script that allows user to turn a production rule on/off. The other option is to use the GUI page for "Product Rule Description". There is a button that can be clicked to turn Active Flag from "ON to "OFF". This will prevent any new production job specs from being created.

Figure 6-1: Production Rule Description Deactivation

6.2.2. Application Restart Procedures

To restart the system, first perform a shutdown as detailed in the Application Shutdown Procedures. Starting the system involves starting the JBoss servers, the loops on the servers (if any), and the Ingest Throttle. To start the JBoss servers, open PuTTY sessions to the following machines :

- n25rpn[01-04] - Ingest, PGS/DIS Factory
- n25rpn[05-06] - Linux-based Processor Nodes (PGS)
- n07apn[2-7] - AIX-based Processor Nodes (PGS)
- n26rpn[01-04] - Distribution Processor Nodes (DIS)
- n26rpn[05-08] - Data Consumer Portal

n25rpn[07-08] - Data Handling System Portal (internal)

Execute the following commands on each machine, in the following machine order: Factory, Ingest, PGS Processor, DIS Processor, Portals

[host](JBoss)> cd $JBOSS_HOME/bin
[host](JBoss)> ./manageJBossServer.pl start nde_op1 [*type*]

Note: enter ***ingest*, *factory*, *processor*, *dmz*** for the [*type*] depending on which host the server is on. Use *processor* for PGS Processor Nodes and *dmz* for DIS Processor Nodes and Portals.

When is server starts there will be a message on the screen indicating how long it took the server to start. *JBoss (Microcontainer) [5.1.0.GA_SOA ….] Started in 48s:834ms.*

To start the loops, use the manageConfigurationRegistry.pl script in /opt/app/nde/NDE_OP1/common and update all xxxxxxLoop_ActiveFlag parameters to 1. The loop flags are listed below:

- [host]_FTPDownloaderLoop_ActiveFlag

- [host]_ProcessIRLoop_ActiveFlag
- [host]_ProcessPJSLoop_ActiveFlag
- [host]_ProcessDPRLoop_ActiveFlag
- [host]_ProcessSubLoop_ActiveFlag
- [host]_ProcessNRLoop_ActiveFlag
- [host]_RetrievePJLoop_ActiveFlag
- [host]_RetrieveDJLoop_ActiveFlag
- [host]_RetrieveNJLoop_ActiveFlag

NOTE: Once a loop is started it will update the display at regular intervals.

Finally, start the Ingest Throttle. To do so, use the manageConfigurationRegistry.pl script in /opt/app/nde/NDE_OP1/common and update the [host]_IngestThrottle_ActiveFlag to 1.

6.3. System Shutdown and Restart

6.3.1. System Shutdown Procedures

The first application to shutdown is the Ingest Throttle. To do so, use the manageConfigurationRegistry.pl script in /opt/app/nde/NDE_OP1/common and update the [host]_IngestThrottle_ActiveFlag to 0 (zero).

Although not required, it is recommended that application loops be shutdown prior to stopping the JBoss servers. To stop the loops, use the manageConfigurationRegistry.pl script in /opt/app/nde/NDE_OP1/common and update the following parameters to 0 (zero):

- [host]_FTPDownloaderLoop_ActiveFlag
- [host]_ProcessIRLoop_ActiveFlag
- [host]_ProcessPJSLoop_ActiveFlag
- [host]_ProcessDPRLoop_ActiveFlag

- [host]_ProcessSubLoop_ActiveFlag
- [host]_ProcessNRLoop_ActiveFlag
- [host]_RetrievePJLoop_ActiveFlag
- [host]_RetrieveDJLoop_ActiveFlag
- [host]_RetrieveNJLoop_ActiveFlag

JBoss servers run on each NDE DHS server host. As of Build 4/5 Release 1 (July 2013), the following hosts run the JBoss Servers with various applications deployed on them. The name of the JBoss server instance on the hosts is 'nde_op1'. The server instance name can vary depending upon the mode; for PE1, it is 'nde_op1'.

- n25rpn[01-02] – Ingest
- n25rpn[03-04] – PGS/DIS Factory
- n25rpn[05-06] - Linux-based Processor Nodes (PGS)
- n07apn[2-7] - AIX-based Processor Nodes (PGS)
- n26rpn[01-04] - Distribution Processor Nodes (DIS)
- n26rpn[05-08] - Data Consumer Portal Web Server
- n25rpn[07-08] - Data Handling System Portal Web Server

To shutdown a JBoss server:

1. Open a putty session and login (as 'jboss') to the hosts mentioned above. It is recommended that JBoss servers be shutdown on hosts in the following order: Portals, Ingest, Factory, DIS Processor, PGS Processor
2. Go to $JBOSS_HOME/bin directory:
    a. [host](JBoss)> cd $JBOSS_HOME/bin
3. Execute the manageJBossServer.pl script:
    a. [host](JBoss)> ./manageJBossServer.pl stop nde_op
4. To verify that JBoss server shutdown is complete, go the $JBSOS_HOME/server/nde_op1/log and make sure 'Shutdown complete' message appears at the end.

Shutting Down Other DHS Component/Utilities

To shutdown monitoring utilities, use the manageConfigurationRegistry.pl script in /opt/app/nde/NDE_OP1/common and update the *ActiveFlag parameters to 0 (zero):

<hostname>_ ResourceMonitors_ActiveFlag

To shutdown the cleanup, JBossLogScanner, and FtpLogScanner utilities:

1. Comment out the relevant crontab command that starts these utilities
2. Kill their corresponding OS process on each of the servers.

In the next release, these would be shutdown using the same mechanism as throttle and Monitoring utilities via the database flags.

To shutdown a machine [rhel5 or aix]:

Login to [host] as admin user (root):

sudo su -

shutdown now

6.3.2. System Restart Procedures

Startup involves starting all the components mentioned above in Shutdown Procedures. These are the JBoss servers, ingest throttle, cleanup, monitoring utilities, and logs scanners.

Starting JBoss Servers

Assuming all server hosts and DHS database are up and running normally, open a putty session to each of the hosts and start the JBoss servers, following the steps below:

n25rpn[01-02] – Ingest

n25rpn[03-04] – PGS/DIS Factory

n25rpn[05-06] - Linux-based Processor Nodes (PGS)

n07apn[2-7] - AIX-based Processor Nodes (PGS)

n26rpn[01-04] - Distribution Processor Nodes (DIS)

n26rpn[05-08] - Data Consumer Portal Web Server n25rpn[07-08] - Data Handling System Portal Web Server

a. Open a putty session and login (as 'jboss')
b. Go to $JBOSS_HOME/bin directory:
    a. [host](JBoss)> cd $JBOSS_HOME/bin
c. Execute the manageJBossServer.pl script. The subsystem argument can be *ingest*, *factory*, *processor*, or *dmz* depending upon which host the server is being started on. Use *dmz* for DIS processors and Web servers, *processor* is for PGS processors, *factory* is for factory servers and *ingest* is for ingest servers.
    a. /manageJBossServer.pl start nde_op1 <subsystem>
d. To make sure JBoss server has started fine, follow these steps:
    a. Change to the JBoss server log directory:
    b. [host](JBoss)> cd $JBOSS_HOME/server/nde_op1/log
    c. Open the file server.log and look for any 'ERROR', 'Exception' messages. If there are, then the server hasn't started normally or the app had deployment errors. These should be corrected before using the server and may require restart of the server. Contact a JBoss Administrator for support.
    d. If server started correctly, there should be no 'ERROR' and/or 'Exception' messages and should have a similar message to the following: *JBoss (Microcontainer) [5.1.0.GA_SOA ….] Started in 48s:834ms*

Starting Application Loops

After all JBoss servers are started successfully (and assuming apps are deployed prior to server startup), make sure the application loops are turned on in the database. When loops (these are like engines) are not active, most of the applications' functionality remains inactive even though they are deployed in the server. To start the loops, use the manageConfigurationRegistry.pl script in /opt/app/nde/NDE_OP1/common and update the *Loop_ActiveFlag parameters to 1:

<hostname>_ FTPDownloaderLoop_ActiveFlag

<hostname>_ IngestThrottle_ActiveFlag

<hostname>_ ProcessIRLoop_ActiveFlag

<hostname>_ ResourceMonitors_ActiveFlag

<hostname>_ ProcessPJSLoop_ActiveFlag
<hostname>_ ProcessDPRLoop_ActiveFlag
<hostname>_ ProcessSubLoop_ActiveFlag
<hostname>_ ProcessNRLoop_ActiveFlag
<hostname> _RetrieveNJLoop_ActiveFlag
NOTE: Once a loop is started it will update the display at regular intervals.

Starting Ingest Throttle

To start the ingest throttle, ensure that the [host]_IngestThrottle_ActiveFlag is set to 1 (ON). To do so, use the manageConfigurationRegistry.pl script in opt/app/nde/NDE_OP1/common to update the value. Next, start the ingest throttle by uncommenting the crontab entry for the ingest throttle (use crontab –e to edit the entries). Verify the throttle has started by using the ps command (ps –ef | grep throttle) and/or by looking at the ingest throttle log in the /opt/data/nde/NDE_OP1/logs/ins directory.

Starting Other DHS Component/Utilities

The other components/utilities include the JBoss log scanners and FTP pull log scanner, resource monitors, and cleanup. Follow these steps to start them and verify their status.

a. Log on to each DHS server host as 'jboss'
b. Uncomment the corresponding crontab entry in the cron jobs list (use crontab –e to edit the entries). These crontab entries are normally created during the DHS System Initialization process.
c. Wait for few minutes for jobs to start. Verify with ps command that the processes have started.
d. Go to /opt/data/nde/NDE_OP1/logs/ folder and open the corresponding log file and look for any errors. If there are any, they need to be rectified (or make sure they are benign and do not cause any functional problems).

6.3.3. System Reboot Procedures

To restart a machine [rhel5 or aix]:

Login to [host] as admin user (root):

```
# sudo su -
# reboot
```

7. APPENDIX 1 – DATA DESCRIPTION

7.1. Data Flow

The various levels of the JPSS RR data flow are presented here. Section 7.1.1 presents the external interfaces to the JPSS RR system. These interfaces represent the connectivity of the JPSS RR system to the NDE DHS. Section 7.1.2 shows the system level design and data flow. The section identifies the main system sub-systems, describes their function, and reveals their interrelationships through data flow. Section 7.1.3 describes each sub-system detailing its components, their function, and the data flows.

7.1.1. External Interfaces

The JPSS RR system code will be run inside the NDE DHS. The relevant external interfaces to the JPSS RR system are shown in Figure 7-1.

The JPSS RR is composed of two sub-systems Preprocessor and Product-Generator. The light blue box is the *JPSS RR* system – it can be any one of the two sub-systems because the way that any sub-system will function within the NDE DHS is the same. The part of the NDE DHS that drives the generation of products is the Product Generation Manager (PGM).

The PGM will invoke an instance of a processing sub-system when all the required inputs are available on the Storage Area Network (SAN). During this invocation the DHS PGM will (1) set up a working directory, (2) produce a Process Control File (PCF) in the working directory for the driver script to use at run time, (3) run the driver script as a system call with the working directory as an argument, (4) obtain the return status when execution of the driver script has completed, and then (5) get the PSF produced by the run from the working directory.

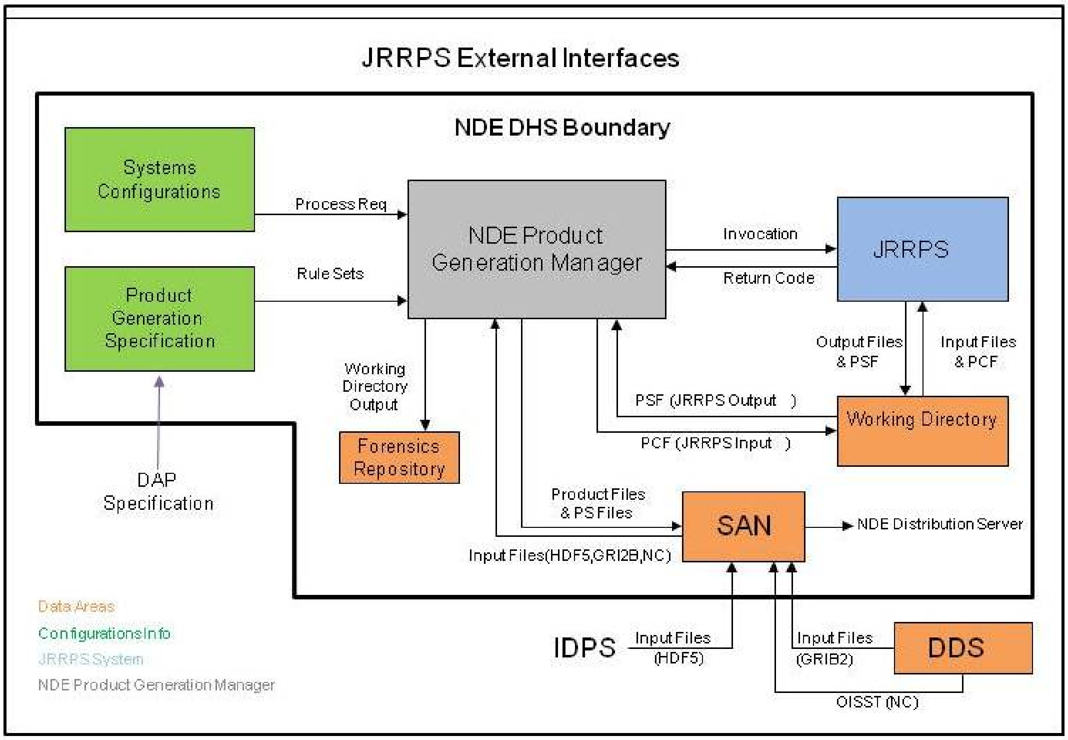


Figure 7-1: JPSS RR External Interfaces

Because the blue box in Figure 7-1 can represent any one of the processing sub-systems, the actual inputs for each sub-system will vary. For this reason, the actual types, names and formats of input and output files associated with each sub-system are shown later in Table 7-1 and Table 7-2 in Section 7.1.2 entitled System Level Data Flow. Instead, in this section only general high-level information about inputs and outputs is provided. Below each component of Figure 7-1 is described.

The behavior, dependencies, and resources associated with a run of each sub-system instance are defined by the green boxes called *Product Generation*

Specifications and *System Configuration*. This is static information provided to the NDE DHS in the Delivered Algorithm Package (DAP) about the necessary inputs required to trigger a run of a given instance and what hardware resources (if any) need to be allocated. The description of the system control and configuration also refer to section 4.1.

The PCF files are supplied by the PGM to the JPSS RR sub-system. They will contain all the required information necessary for a particular run. The working directory is the actual location on the disk where the run will take place. The JPSS RR system's PCF, PSF, and log files will be produced there.

The input data files will ultimately arrive in the NDE DHS from the Interface Data Processing Segment (IDPS), Environmental Satellite Processing Center (ESPC). The IDPS will supply the Visible Infrared Imager Radiometer Suite (VIIRS) data files in HDF5 format. ESPC will supply the Global Forecast System (GFS) forecast files in GRIB2 format. ESPC will also supply the optimum interpolation (OI) sea surface temperature (SST) analysis daily data files (OISST daily) in NetCDF format. The DHS will ingest these files and store them on the SAN. The VIIRS files will arrive in HDF5 format. The OISST daily will arrive as NetCDF format with name convention of avhrr-only-v2.YYYYMMDD_preliminary.nc and will be renamed as avhrr-only-v2.YYYYMMDD.nc through the symbolic links by JPSS RR system. Same to the GFS data, it will arrive with the name gfs.tXXz.pgrb2.0p50.fXXX.YYYYMMDD and will be renamed as gfs.tXXz.pgrbfXX and be placed in the corresponding sub-directory of YYYY/MM/DD through the symbolic links by the JPSS RR system (YYYY is the 4-digit year, MM is the 2-digit month, and DD is the 2-digit day of month). Details see section 7.2.

The execution of a JPSS RR instance run will produce a single return code indicating status (e.g. success or failure). This will be checked by the PGM since the PGM will be performing the execution. The remapped file or JPSS RR product (in NetCDF 4 format) will be placed in the working directory. At the same time, a log file and a PSF file from the driver script (e.g. JRR_PRODUCT_CLOUD_MASK.pl.log and JRR_PRODUCT_CLOUD_MASK.pl.PSF) will also be generated and located right under the working directory. The log file will contain the run status information and the PSF file will contain the full path of the remapped file or JPSS RR product file name if the current instance run is successful.

Following an instance run, the PGM will decide about what to do with a given run's output according to the run status and the contents of the PSF. During normal operations, the remapped file or JPSS RR product from a successful run will be directed to the SAN from the working directory by the DHS where the DHS will later redirect them for tailoring or distribution to NDE customers on a distribution server. The run logs will be sent to a *Forensics Repository* for temporary storage. These logs will be used if problems are encountered and troubleshooting is necessary.

Figure 7-2 shows the context level data flows into and out of the cryosphere, aerosol, and clouds products. The collective input and output of the system are generalized as static input, dynamic input and output relative to the blue JPSS RR box in the center of the figure.

The arrows represent only the high-level file-based interfaces to the system, as opposed to other types of interfaces such as the unit return values, program arguments, or shell variables. Some of the file interfaces actually represent collections of files such as the *VIIRS SDR SVM01-16, VIIRS GMTCO, Framework CFG templates, Framework PCF templates and GFS*. These files are too numerous to show in a single context level diagram. More details about these collective blocks of files, and the units to which they are relevant, are provided later in section 7.1.2.

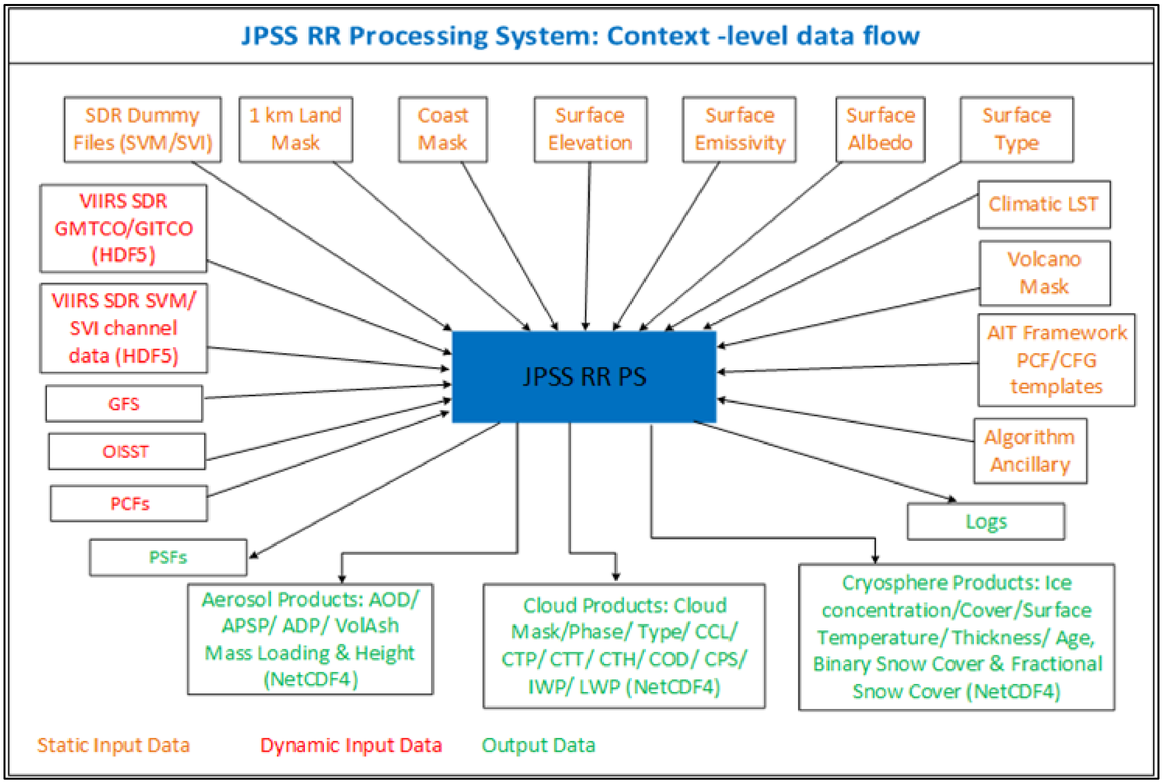


Figure 7-2: JPSS RR Context Level Data Flow for Aerosol, Clouds, and Cryosphere Products

7.1.2. System Level Design and Data Flow

Before the description of system level design and data flow, it is necessary to introduce the general background for the JPSS RR:

- The JPSS RR algorithm runs inside the AIT-Framework
- The images should contain variables from SDR, cloud, aerosol, and cryosphere products.
- AIT-framework requires the input data to be in NetCDF.

There are two sub-systems in JPSS RR: Preprocessor and Product-Generator. The Preprocessor is to create the NetCDF format file and the Product-Generator is to generate the JPSS RR clouds, aerosol, and cryosphere products.

The JPSS RR system level data flow is show in Figure 7-3 for the cryosphere, aerosol, and clouds products. The Preprocessor sub-system is identified by the pink box and Product-Generator as the grey shaded box, and they are driven by the driver script JRR_PREPROCESS_SDR.pl and the six product driver scripts JRR_PRODUCT_CLOUD_MASK.pl, JRR_PRODUCT_CLOUD_CLOUDS.pl, JRR_PRODUCT_AEROSOL_AODADP.pl, JRR_PRODUCT_AEROSOL_VOLASH.pl, JRR_PRODUCT_CRYOSPHERE_ICE.pl, and JRR_PRODUCT_CRYOSPHERE_SNOW.pl. The white space around each sub-system is the domain of the NDE DHS. The white boxes are the files generated by the sub-system and will be directed to the SAN from the working directory by the DHS. Any input static data and system files are not shown in this figure. Table 7-1 and Table 7-2 show the complete input and output files from the corresponding sub-systems. All output files from each sub-system are deposited into a local working directory by default. All input and output files from each sub-system are managed solely by the DHS.

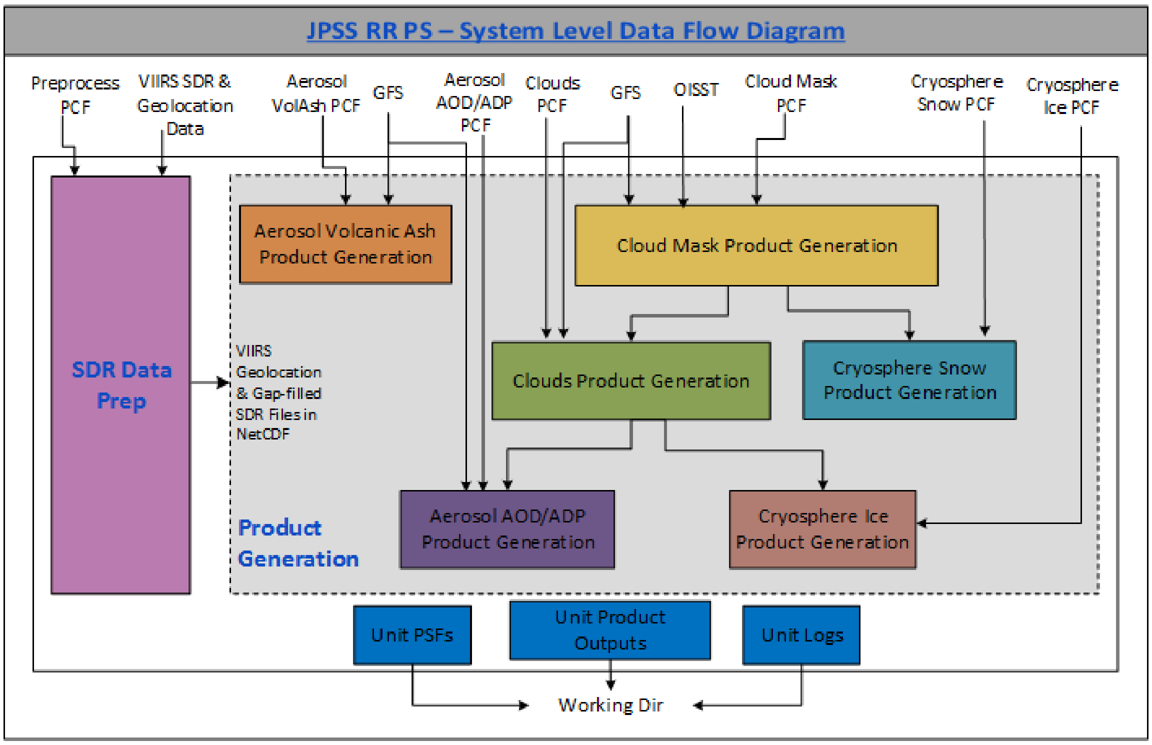


Figure 7-3: JPSS RR System Level Data Flow for Cryosphere, Cloud, and Aerosol Products

(The blue box stands for the sub-system. The blue boxes are the files generated by the sub-system and will be directed to the SAN from the working directory by the DHS. The white space around each unit within the pink and grey shaded boxes is the domain of NDE DHS. Any input static data and system files are not shown in this figure. See Table 7-1 and Table 7-2 for the complete input files.)

The Preprocessor is to create the NetCDF format file which will be the input to the sub-system Product-Generator. The Product-Generator is to generate the JPSS RR products. The two sub-systems are independent with respect to program execution scope. For example, one sub-system is not calling another. Each of the two sub-systems is called by the DHS PGM when any of the given run requirements are met. Therefore, the two sub-systems are only related to each other through the dependency of their external data flows.

The data flows for the cryosphere, aerosol, and clouds products in Figure 7-3 are simplified and greater details about the input and output of each sub-system are provided in Table 7-1 and Table 7-2. The two sub-systems described in this section produce intermediate data files and internal log files. They are not described in this section because the information in these files does not need to travel directly into or out of the sub-systems. These files are described in section 7.1.3 and listed in Table 7-3 and Table 7-4.

The Tables contain the file name, the file's input/output designation, the files source, a description of the file (or purpose), and the *state* of the file. The file's input/output designation is listed as *input*, *intermediate*, or *output*. To make this a little easier to visualize, the table cells are highlighted in light blue for *input*, light gray for *intermediate*, and light yellow for *output*. The file source is defined as the agent that generated the file, not the agent providing the file to the unit during execution. The agent providing all files to the JPSS RR sub-systems is the NDE DHS. The file *state* is either static or dynamic. Dynamic files are defined as those files that regularly change such as instrument data files, gfs data files, for example. Static files are those files that do not change. Static files may change only when there is a new delivery of the algorithm. CRTM coefficient files are a good example of static files.

Table 7-1: JPSS RR Sub-System Preprocessor Data Flow

File	Input/ Output	Source	Description	State
JRR_PREPROCESS_SDR.pl.PCF	Input	NDE	The JPSS RR sub-system Preprocessor PCF supplied by the NDE PGM.	Dynamic
GMTCO_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.h5	Input	NDE	VIIRS SDR Moderate Resolution Terrain Corrected Geolocation granule file in HDF5 format.	Dynamic
GITCO_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.h5	Input	NDE	VIIRS SDR Moderate Resolution Terrain Corrected Geolocation granule file in HDF5 format.	Dynamic

SVM01_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.h5	Input	NDE	VIIRS SDR Moderate Resolution Band 01 granule file in HDF5 format. There are multiple of such files.	Dynamic
SVM02_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.h5	Input	NDE	VIIRS SDR Moderate Resolution Band 02 granule file in HDF5 format. There are multiple of such files.	Dynamic
SVM03_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.h5	Input	NDE	VIIRS SDR Moderate Resolution Band 03 granule file in HDF5 format. There are multiple of such files.	Dynamic
SVM04_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.h5	Input	NDE	VIIRS SDR Moderate Resolution Band 04 granule file in HDF5 format. There are multiple of such files.	Dynamic
SVM05_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.h5	Input	NDE	VIIRS SDR Moderate Resolution Band 05 granule file in HDF5 format. There are multiple of such files.	Dynamic
SVM06_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.h5	Input	NDE	VIIRS SDR Moderate Resolution Band 06 granule file in HDF5 format. There are multiple of such files.	Dynamic
SVM07_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.h5	Input	NDE	VIIRS SDR Moderate Resolution Band 07 granule file in HDF5 format. There are multiple of such files.	Dynamic
SVM08_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.h5	Input	NDE	VIIRS SDR Moderate Resolution Band 08 in HDF5 format. There are multiple of such files.	Dynamic
SVM09_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.h5	Input	NDE	VIIRS SDR granule files containing Moderate Resolution Band 09 granule file in HDF5 format. There are multiple of such files.	Dynamic
SVM10_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.h5	Input	NDE	VIIRS SDR Moderate Resolution Band 10 granule file in HDF5 format. There are multiple of such files.	Dynamic
SVM11_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.h5	Input	NDE	VIIRS SDR Moderate Resolution Band 11 granule file in HDF5 format. There are multiple of such files.	Dynamic

SVM12_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.h5	Input	NDE	VIIRS SDR Moderate Resolution Band 12 granule file in HDF5 format. There are multiple of such files.	Dynamic
SVM13_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.h5	Input	NDE	VIIRS SDR Moderate Resolution Band 13 granule file in HDF5 format. There are multiple of such files.	Dynamic
SVM14_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.h5	Input	NDE	VIIRS SDR Moderate Resolution Band 14 granule file in HDF5 format. There are multiple of such files.	Dynamic
SVM15_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.h5	Input	NDE	VIIRS SDR Moderate Resolution Band 15 granule file in HDF5 format. There are multiple of such files.	Dynamic
SVM16_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.h5	Input	NDE	VIIRS SDR Moderate Resolution Band 16 granule file in HDF5 format. There are multiple of such files.	Dynamic
SVI01_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.h5	Input	NDE	VIIRS SDR Moderate Resolution Band 01 granule file in HDF5 format. There are multiple of such files.	Dynamic
SVI02_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.h5	Input	NDE	VIIRS SDR Moderate Resolution Band 02 granule file in HDF5 format. There are multiple of such files.	Dynamic
SVI03_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.h5	Input	NDE	VIIRS SDR Moderate Resolution Band 03 granule file in HDF5 format. There are multiple of such files.	Dynamic
SVI04_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.h5	Input	NDE	VIIRS SDR Moderate Resolution Band 04 granule file in HDF5 format. There are multiple of such files.	Dynamic
SVI05_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.h5	Input	NDE	VIIRS SDR Moderate Resolution Band 05 granule file in HDF5 format. There are multiple of such files.	Dynamic
GMTCO_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.nc	Output	NDE	VIIRS SDR Moderate Resolution Terrain Corrected Geolocation granule file in NetCDF format.	Dynamic
GITCO_npp_d????????	Output	NDE	VIIRS SDR Moderate	Dynamic

_t???????_e???????_b?????_c?????????????????????_noaa_ops.nc			Resolution Terrain Corrected Geolocation granule file in NetCDF format. There are multiple of such files.	
SVM01_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.nc	Output	NDE	VIIRS SDR Moderate Resolution Band 01 granule file in NetCDF format. There are multiple of such files.	Dynamic
SVM02_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.nc	Output	NDE	VIIRS SDR Moderate Resolution Band 02 granule file in NetCDF format. There are multiple of such files.	Dynamic
SVM03_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.nc	Output	NDE	VIIRS SDR Moderate Resolution Band 03 granule file in NetCDF format. There are multiple of such files.	Dynamic
SVM04_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.nc	Output	NDE	VIIRS SDR Moderate Resolution Band 04 granule file in NetCDF format. There are multiple of such files.	Dynamic
SVM05_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.nc	Output	NDE	VIIRS SDR Moderate Resolution Band 05 granule file in NetCDF format. There are multiple of such files.	Dynamic
SVM06_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.nc	Output	NDE	VIIRS SDR Moderate Resolution Band 06 granule file in NetCDF format. There are multiple of such files.	Dynamic
SVM07_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.nc	Output	NDE	VIIRS SDR Moderate Resolution Band 07 granule file in NetCDF format. There are multiple of such files.	Dynamic
SVM08_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.nc	Output	NDE	VIIRS SDR Moderate Resolution Band 08 in NetCDF format. There are multiple of such files.	Dynamic
SVM09_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.nc	Output	NDE	VIIRS SDR granule files containing Moderate Resolution Band 09 granule file in NetCDF format. There are multiple of such files.	Dynamic
SVM10_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.nc	Output	NDE	VIIRS SDR Moderate Resolution Band 10 granule file in NetCDF format. There are multiple of such files.	Dynamic

SVM11_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.nc	Output	NDE	VIIRS SDR Moderate Resolution Band 11 granule file in NetCDF format. There are multiple of such files.	Dynamic
SVM12_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.nc	Output	NDE	VIIRS SDR Moderate Resolution Band 12 granule file in NetCDF format. There are multiple of such files.	Dynamic
SVM13_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.nc	Output	NDE	VIIRS SDR Moderate Resolution Band 13 granule file in NetCDF format. There are multiple of such files.	Dynamic
SVM14_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.nc	Output	NDE	VIIRS SDR Moderate Resolution Band 14 granule file in NetCDF format. There are multiple of such files.	Dynamic
SVM15_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.nc	Output	NDE	VIIRS SDR Moderate Resolution Band 15 granule file in NetCDF format. There are multiple of such files.	Dynamic
SVM16_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.nc	Output	NDE	VIIRS SDR Moderate Resolution Band 16 granule file in NetCDF format. There are multiple of such files.	Dynamic
SVI01_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.nc	Output	NDE	VIIRS SDR Moderate Resolution Band 01 granule file in NetCDF format. There are multiple of such files.	Dynamic
SVI02_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.nc	Output	NDE	VIIRS SDR Moderate Resolution Band 02 granule file in NetCDF format. There are multiple of such files.	Dynamic
SVI03_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.nc	Output	NDE	VIIRS SDR Moderate Resolution Band 03 granule file in NetCDF format. There are multiple of such files.	Dynamic
SVI04_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.nc	Output	NDE	VIIRS SDR Moderate Resolution Band 04 granule file in NetCDF format. There are multiple of such files.	Dynamic
SVI05_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.nc	Output	NDE	VIIRS SDR Moderate Resolution Band 05 granule file in NetCDF format. There are multiple of such files.	Dynamic

Table 7-2: JPSS RR Sub-System Product-Generator Flow for JPSS RR products

File	Input/ Output	Source	Description	State
JRR_PRODUCT_CLOUD_MASK.pl.PCF	Input	NDE	The JRR sub-system Product-Generator PCF supplied by the NDE PGM.	Dynamic
JRR_PRODUCT_CLOUD_CLOUDS.pl.PCF	Input	NDE	The JRR sub-system Product-Generator PCF supplied by the NDE PGM.	Dynamic
JRR_PRODUCT_AEROSOL_VOLASH.pl.PCF	Input	NDE	The JRR sub-system Product-Generator PCF supplied by the NDE PGM.	Dynamic
JRR_PRODUCT_AEROSOL_AODADP.pl.PCF	Input	NDE	The JRR sub-system Product-Generator PCF supplied by the NDE PGM.	Dynamic
JRR_PRODUCT_CRYO	Input	NDE	The JRR sub-system Product-	Dynamic

SPHERE_SNOW.pl.PCF			Generator PCF supplied by the NDE PGM.	
JRR_PRODUCT_CRYOSPHERE_ICE.pl.PCF	Input	NDE	The JRR sub-system Product-Generator PCF supplied by the NDE PGM.	Dynamic
GITCO_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.nc	Input	NDE	VIIRS SDR Moderate Resolution Terrain Corrected Geolocation granule file in NetCDF format. There are multiple of such files.	Dynamic
GMTCO_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.nc	Input	NDE	VIIRS SDR Moderate Resolution Terrain Corrected Geolocation granule file in NetCDF format. There are multiple of such files.	Dynamic
SVIO1_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.nc	Input	NDE	VIIRS SDR Moderate Resolution Band 01 granule file in NetCDF format. There are multiple of such files.	Dynamic
SVIO2_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.nc	Input	NDE	VIIRS SDR Moderate Resolution Band 02 granule file in NetCDF format. There are multiple of such files.	Dynamic
SVIO3_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.nc	Input	NDE	VIIRS SDR Moderate Resolution Band 03 granule file in NetCDF format. There are multiple of such files.	Dynamic
SVIO4_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.nc	Input	NDE	VIIRS SDR Moderate Resolution Band 04 granule file in NetCDF format. There are multiple of such files.	Dynamic
SVIO5_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.nc	Input	NDE	VIIRS SDR Moderate Resolution Band 05 granule file in NetCDF format. There are multiple of such files.	Dynamic
SVM01_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.nc	Input	NDE	VIIRS SDR Moderate Resolution Band 01 granule file in NetCDF format. There are multiple of such files.	Dynamic
SVM02_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.nc	Input	NDE	VIIRS SDR Moderate Resolution Band 02 granule file in NetCDF format. There are multiple of such files.	Dynamic
SVM03_npp_d????????_t???????_e???????_b	Input	NDE	VIIRS SDR Moderate Resolution Band 03 granule file	Dynamic

?????_c????????????? ???????_noaa_ops.nc			in NetCDF format. There are multiple of such files.	
SVM04_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.nc	Input	NDE	VIIRS SDR Moderate Resolution Band 04 granule file in NetCDF format. There are multiple of such files.	Dynamic
SVM05_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.nc	Input	NDE	VIIRS SDR Moderate Resolution Band 05 granule file in NetCDF format. There are multiple of such files.	Dynamic
SVM06_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.nc	Input	NDE	VIIRS SDR Moderate Resolution Band 06 granule file in NetCDF format. There are multiple of such files.	Dynamic
SVM07_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.nc	Input	NDE	VIIRS SDR Moderate Resolution Band 07 granule file in NetCDF format. There are multiple of such files.	Dynamic
SVM08_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.nc	Input	NDE	VIIRS SDR Moderate Resolution Band 08 granule file in NetCDF format. There are multiple of such files.	Dynamic
SVM09_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.nc	Input	NDE	VIIRS SDR Moderate Resolution Band 09 granule file in NetCDF format. There are multiple of such files.	Dynamic
SVM10_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.nc	Input	NDE	VIIRS SDR Moderate Resolution Band 10 granule file in NetCDF format. There are multiple of such files.	Dynamic
SVM11_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.nc	Input	NDE	VIIRS SDR Moderate Resolution Band 11 granule file in NetCDF format. There are multiple of such files.	Dynamic
SVM12_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.nc	Input	NDE	VIIRS SDR Moderate Resolution Band 12 granule file in NetCDF format. There are multiple of such files.	Dynamic
SVM13_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.nc	Input	NDE	VIIRS SDR Moderate Resolution Band 13 granule file in NetCDF format. There are multiple of such files.	Dynamic
SVM14_npp_d????????_t???????_e???????_b?????_c?????????????	Input	NDE	VIIRS SDR Moderate Resolution Band 14 granule file in NetCDF format. There are	Dynamic

???????_noaa_ops.nc			multiple of such files.	
SVM15_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.nc	Input	NDE	VIIRS SDR Moderate Resolution Band 15 granule file in NetCDF format. There are multiple of such files.	Dynamic
SVM16_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.nc	Input	NDE	VIIRS SDR Moderate Resolution Band 16 granule file in NetCDF format. There are multiple of such files.	Dynamic
gfs.t??z.pgrb2f??.?????????	Input	NCEP	The GFS 6-hour forecast file in GRIB2 format, 0.5 degree resolution. There are multiple of such files.	Dynamic
run_wgrib.pl	Input	STAR	Perl script used to obtain the NWP GFS data.	Static
avhrr-only-v2.????????_preliminary.nc	Input	STAR	Daily OISST data at 0.25 degree resolution.	Dynamic
Land_Mask.pcf	Input	STAR	AIT-framework PCF file to process Land Mask data	Static
OISST_Daily_QtrDeg.pcf	Input	STAR	AIT-framework PCF file to process OISST data	Static
NWP_Data_grib2_0.5deg.pcf	Input	STAR	AIT-framework PCF file to process NWP Grib2 data	Static
Surface_Elevation.pcf	Input	STAR	AIT-framework PCF file to process surface elevation data	Static
Surface_Emissivity.pcf	Input	STAR	AIT-framework PCF file to process surface emissivity data	Static
CRTM.pcf	Input	STAR	AIT-framework PCF file to process CRTM data	Static
Coast_Mask.pcf	Input	STAR	AIT-framework PCF file to process coast mask data	Static
Desert_Mask.pcf	Input	STAR	AIT-framework PCF file to process desert mask data	Static
Snow_Mask.pcf	Input	STAR	AIT-framework PCF file to process snow mask data	Static
Climatic_LST.pcf	Input	STAR	AIT-framework PCF file to process climatic LST data	Static
Pseudo_Emissivity.pcf	Input	STAR	AIT-framework PCF file to process pseudo emissivity data	Static
Surface_Type.pcf	Input	STAR	AIT-framework PCF file to process surface type data	Static
Surface_Albedo.pcf	Input	STAR	AIT-framework PCF file to process surface albedo data	Static
NPP_BAYES_Cloud_Ma	Input	STAR	A template AIT-framework PCF	Static

sk.pcf			file to generate Bayes cloud mask PCF file	
NPP_BAYES_Cloud_Mask_NW.pcf	Input	STAR	A template AIT-framework PCF file to generate Bayes cloud mask PCF file	Static
NPP_BAYES_Cld_Cld_Phase.pcf	Input	STAR	A template AIT-framework PCF file to generate cloud phase da PCF file ta	Static
NPP_BAYES_Cld_Height.pcf	Input	STAR	A template AIT-framework PCF file to generate cloud height PCF file	Static
NPP_BAYES_Cld_NCOMP.pcf	Input	STAR	A template AIT-framework PCF file to generate NCOMP PCF file	Static
NPP_BAYES_Cloud_Micro_Day.pcf	Input	STAR	A template AIT-framework PCF file to process DCOMP PCF file	Static
NPP_BAYES_Cld_Height_NW.pcf	Input	STAR	A template AIT-framework PCF file to generate cloud height PCF file	Static
AWG_AER_AOD.pcf	Input	STAR	A template AIT-framework PCF file to generate aerosol optical depth PCF file	Static
NPP_VIIRS_AERADP.pcf	Input	STAR	A template AIT-framework PCF file to generate aerosol optical depth PCF file	Static
Volcano_Mask.pcf	Input	STAR	A template AIT-framework PCF file to generate volcano mask PCF file	Static
NPP_VOLCANIC_ASH.pcf	Input	STAR	A template AIT-framework PCF file to generate volcanic ash PCF file	Static
AWG_Ice_Conc_VIIRS_BAYES.pcf	Input	STAR	A template AIT-framework PCF file to generate ice concentration PCF file	Static
AWG_Ice_Age_VIIRS_BAYES.pcf	Input	STAR	A template AIT-framework PCF file to generate ice age PCF file	Static
NPP_VIIRS_SNOW_COVER.pcf	Input	STAR	A template AIT-framework PCF file to generate ice concentration PCF file	Static
npp_viirs_ancil.Mbands.nc	Input	STAR	Ancillary data required by the VIIRS SDR reader of AIT framework	Static
npp_viirs_ancil.Ibands.nc	Input	STAR	Ancillary data required by the VIIRS SDR reader of AIT framework	Static

lw_geo_2001001_v03m.nc	Input	STAR	Global 1km land/water used for MODIS collection 5	Static
GLOBE_1km_digelev.nc	Input	STAR	Digital surface elevation at 1km resolution	Static
global_emiss_intABI_???????.nc	Input	STAR	Surface emissivity at 5km resolution. There are multiple of such files.	Static
AerosolCoeff.bin	Input	STAR	CTRM Coeff	Static
CloudCoeff.bin	Input	STAR	CTRM Coeff	Static
EmisCoeff.bin	Input	STAR	CTRM Coeff	Static
viirs-i_npp.SpcCoeff.bin	Input	STAR	CTRM Coeff	Static
viirs-i_npp.TauCoeff.bin	Input	STAR	CTRM Coeff	Static
viirs-m_npp.SpcCoeff.bin	Input	STAR	CTRM Coeff	Static
viirs-m_npp.TauCoeff.bin	Input	STAR	CTRM Coeff	Static
viirs-m_npp.SpcCoeff.bin	Input	STAR	CTRM Coeff	Static
viirs-m_npp.TauCoeff.bin	Input	STAR	CTRM Coeff	Static
coast_mask_1km.nc	Input	STAR	Global 1km land/water used for MODIS collection 5	Static
gl-latlong-1km-landcover.nc	Input	STAR	Surface type mask based on AVHRR at 1km resolution	Static
global_emiss_intABI_2005???.nc	Input	STAR	Seebor global emissions from 2005. There are multiple of such files.	Static
AlbMap.WS.c004.v2.0.2004.???.0.659_x4.nc	Input	STAR	White sky reflectance at 0.659 um.. There are multiple of such files.	Static
AlbMap.WS.c004.v2.0.2004.???.1.64_x4.nc	Input	STAR	White sky reflectance at 1.64 um. There are multiple of such files.	Static
AlbMap.WS.c004.v2.0.2004.???.2.13_x4.nc	Input	STAR	White sky reflectance at 2.13 um. There are multiple of such files.	Static
viirs_default_nb_cloud_mask_lut.nc	Input	STAR	Ancillary data required by the AIT-framework	Static
VIIRS_coefs.nc	Input	STAR	Coefficients for NCOMP	Static
volcano_mask_1km.nc	Input	STAR	Volcano mask at 1km resolution	Static
VIIRS_Aerosol_Lut.nc	Input	STAR	VIIRS aerosol look up table data	Static
VIIRS_Sunglint_Lut.nc	Input	STAR	VIIRS sun glint look up table data	Static
VIIRS_Coeff.nc	Input	STAR	VIIRS aerosol coefficients	Static

VIIRS_BrightSfc_Refl_Coef.nc	Input	STAR	VIIRS surface reflectance coefficients	Static
snow_occurrence_latlon_week_??.nc	Input	STAR	Weekly snow occurrence ancillary file. There are multiple of such files.	Static
climatic_lst_month_??.nc	Input	STAR	Monthly climatic LST ancillary file. There are multiple of such files.	Static
IceSrfTempCoeff_NPP_VIIRS.nc	Input	STAR	Ice surface temperature coefficients.	Static
AITA_INPUT_Coefficients_ResiFlux.nc	Input	STAR	Ice age coefficients.	Static
VIIRS_ch?_ref_lut_ice_cld.nc	Input	STAR	VIIRS DCOMP lookup table for ice	Static
VIIRS_ch?_ref_lut_wat_cld.nc	Input	STAR	ViIRS DCOMP lookup table for water	Static
JRR-CloudMask_v1r1_npp_d????????_t???????_e???????_b?????_s???????????????_e???????????????_c???????????????.nc	Output	JRR	The Bayes cloud mask	Dynamic
JRR-CloudPhase_v1r1_npp_d????????_t???????_e???????_b?????_s???????????????_e???????????????_c???????????????.nc	Output	JRR	Cloud phase	Dynamic
JRR-CloudHeight_v1r1_npp_d????????_t???????_e???????_b?????_s???????????????_e???????????????_c???????????????.nc	Output	JRR	Cloud height	Dynamic
JRR-CloudDCOMP_v1r1_npp_d????????_t???????_e???????_b?????_s???????????????_e???????????????_c???????????????.nc	Output	JRR	Cloud Daytime COMP	Dynamic
JRR-	Output	JRR	Cloud Nighttime COMP	Dynamic

CloudNCOMP_v1r1_npp_d????????_t???????_e???????_b?????_s???????????????_e???????????????_c???????????????.nc				
JRR-AOD_v1r1_npp_d????????_t???????_e???????_b?????_s???????????????_e???????????????_c???????????????.nc	Output	JRR	Aerosol Optical Depth	Dynamic
JRR-ADP_v1r1_npp_d????????_t???????_e???????_b?????_s???????????????_e???????????????_c???????????????.nc	Output	JRR	Aerosol Path	Dynamic
JRR-SnowCover_v1r1_npp_d????????_t???????_e???????_b?????_s???????????????_e???????????????_c???????????????.nc	Output	JRR	Snow Cover	Dynamic
JRR-IceConcentration_v1r1_npp_ d????????_t???????_e???????_b?????_s???????????????_e???????????????_c???????????????.nc	Output	JRR	Ice Concentration	Dynamic
JRR-IceAge_v1r1_npp_d????????_t???????_e???????_b?????_s???????????????_e???????????????_c???????????????.nc	Output	JRR	Ice Age	Dynamic
JRR_PRODUCT_CLOUD_MASK.pl.PSF	Output	JRR	This is the JPSS RR sub-system Product-Generator PSF containing the instance run generated JRR product file,	Dynamic

			including the full path.	
JRR_PRODUCT_CLOUD_CLOUDS.pl.PSF	Output	JRR	This is the JPSS RR sub-system Product-Generator PSF containing the instance run generated JRR product file, including the full path.	Dynamic
JRR_PRODUCT_AEROSOL_VOLASH.pl.PSF	Output	JRR	This is the JPSS RR sub-system Product-Generator PSF containing the instance run generated JRR product file, including the full path.	Dynamic
JRR_PRODUCT_AEROSOL_AODADP.pl.PSF	Output	JRR	This is the JPSS RR sub-system Product-Generator PSF containing the instance run generated JRR product file, including the full path.	Dynamic
JRR_PRODUCT_CRYOSPHERE_SNOW.pl.PSF	Output	JRR	This is the JPSS RR sub-system Product-Generator PSF containing the instance run generated JRR product file, including the full path.	Dynamic
JRR_PRODUCT_CRYOSPHERE_ICE.pl.PSF	Output	JRR	This is the JPSS RR sub-system Product-Generator PSF containing the instance run generated JRR product file, including the full path.	Dynamic
JRR_PRODUCT_CLOUD_MASK.pl.log	Output	JRR	This is the run log containing the standard output and return status of the JPSS RR sub-system Product-Generator instance run. When an error happens while running a unit script, the log information from that unit will be appended to it.	Dynamic
JRR_PRODUCT_CLOUD_CLOUDS.pl.log	Output	JRR	This is the run log containing the standard output and return status of the JPSS RR sub-system Product-Generator instance run. When an error happens while running a unit script, the log information from that unit will be appended to it.	Dynamic
JRR_PRODUCT_AEROSOL_VOLASH.pl.log	Output	JRR	This is the run log containing the standard output and return status of the JPSS RR sub-	Dynamic

			system Product-Generator instance run. When an error happens while running a unit script, the log information from that unit will be appended to it.	
JRR_PRODUCT_AEROSOL_AODADP.pl.log	Output	JRR	This is the run log containing the standard output and return status of the JPSS RR sub-system Product-Generator instance run. When an error happens while running a unit script, the log information from that unit will be appended to it.	Dynamic
JRR_PRODUCT_CRYOSPHERE_SNOW.pl.log	Output	JRR	This is the run log containing the standard output and return status of the JPSS RR sub-system Product-Generator instance run. When an error happens while running a unit script, the log information from that unit will be appended to it.	Dynamic
JRR_PRODUCT_CRYOSPHERE_ICE.pl.log	Output	JRR	This is the run log containing the standard output and return status of the JPSS RR sub-system Product-Generator instance run. When an error happens while running a unit script, the log information from that unit will be appended to it.	Dynamic

7.1.3. Sub-System Level Design and Data Flow

JPSS RR is composed of the two sub-systems Preprocessor and Product-Generator. This section describes the general purpose, design, and data flows within each sub-system. Each sub-system is driven by a single Perl script that wraps around a number of smaller programs such as compiled data handling and science code (written in Fortran 77/90/95 and C/C++), Perl scripts, netCDF4 library utilities, and UNIX system calls. The driver script for sub-systems Preprocessor is JRR_PREPROCESS_SDR.pl and the driver scripts for Product-Generator are JRR_PRODUCT_CLOUD_MASK.pl, JRR_PRODUCT_CLOUD_CLOUDS.pl, JRR_PRODUCT_AEROSOL_AODADP.pl, JRR_PRODUCT_AEROSOL_VOLASH.pl, JRR_PRODUCT_CRYOSPHERE_ICE.pl, and JRR_PRODUCT_CRYOSPHERE_SNOW.pl. Each of the two sub-systems is independent with respect to program execution scope. For example, one sub-system is not calling another. Each of the two sub-systems is called by the DHS PGM when any of the given unit's run requirements are met. Therefore, the sub-systems are only related to each other through the dependency of their external data flows.

The sub-system Preprocessor calls two units: the data gapfilling unit and the conversion to netcdf format. The Preprocessor takes a granule of VIIRS SDR and IP as input to create the NetCDF format file. The Preprocessor reads-in its Process Control File (PCF), then it will: 1) make necessary symbolic links for files and directories; 2) handles the VIIRS SDR SVM01-16 and SVI01-05 data gap filling if the GAPFILL key is ON; otherwise it skips the gap filling; 3) converts VIIRS SDR SVM01-16, SVI01-05, GMTCO, and GITCO from HDF5 to netCDF4.

The Product-Generator takes the NetCDF format files as input to generate the JPSS RR products. The Product-Generator reads-in its PCF, then it will: rename all SDR data for Framework use by extracting the scan time information from the GMTCO file and attaching it to the netCDF file names; 2) set up running enviromnet for Framework by making necessary symbolic links for files and directories; 3) create Framework process control file using a template; 4) invoke the JPSS RR-Product-Generation unit to generate the JPSS RR product; then 5) write the JPSS RR product to the sub-system PSF.

The Product-Generator sub-system's data flow is shown in Figure 7-4, Figure 7-9, and Figure 7-10 for the cryosphere, aerosols, and clouds products, respectively. To be simple, only the interfaces to each unit are shown in the Figures. Also only the dynamic data files are shown in the Figures. The static data files are assumed to be part of the system files and are not shown in the Figures. The sub-system access the static data files through its PCF file that is generated by PGM. In Figure 7-4, Figure 7-5, and Figure 7-6, the boundary of the sub-system domain is indicated by the box with the red bolded lines (red box). The light gray space around each unit within the red box is the sub-system's functional domain. The light gray space outside the red box is the domain of NDE DHS. The units are identified as blue boxes. There are three units in the sub-system Product-Generator, the renaming of the SDR for Framework use, setting up the environment for the Framework, and running the Framework to obtain the relevant products. The files generated by the sub-system are identified with the orange boxes. The files generated by each unit are identified with the yellow boxes. Files with no boxes are the dynamic data files that NDE put into the working directory. All input and output files from each sub-system are managed solely by the DHS. The data flows in these figures are simplified.

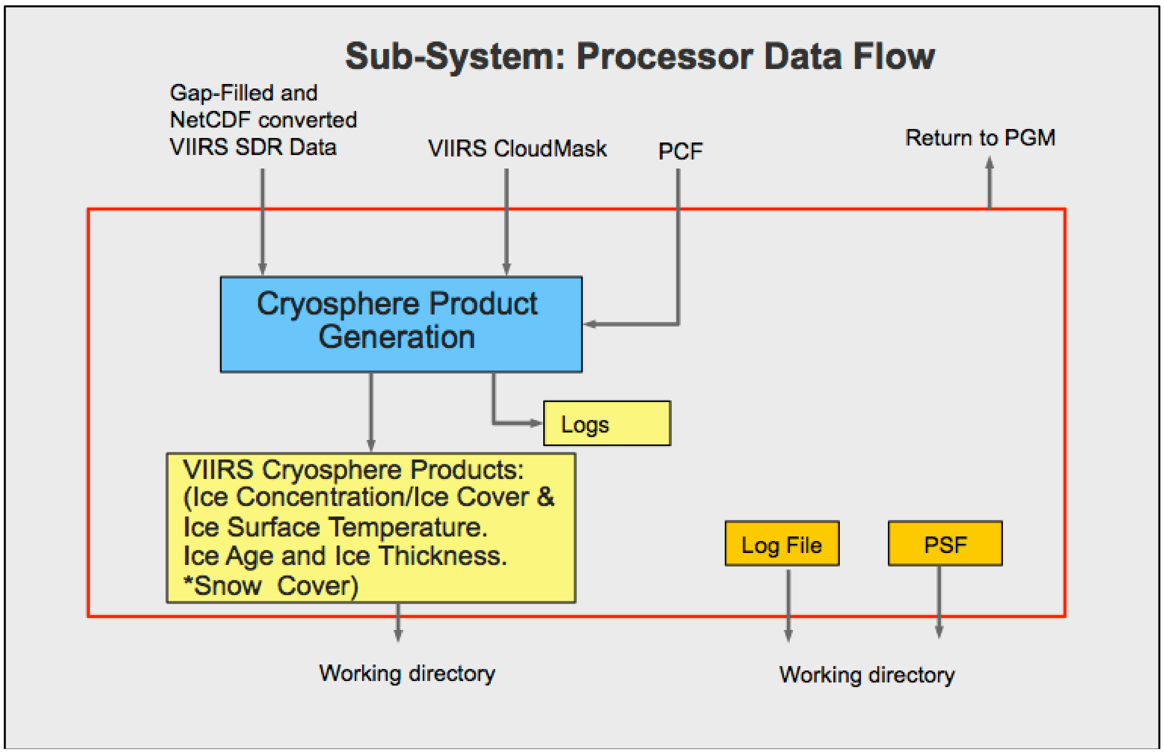


Figure 7-4: Sub-System Processor Data Flow for Cryosphere Products

(The boundary of the sub-system processor domain is indicated by the red box. The light gray space around each unit within the red box is the driver script's functional domain. The light gray space outside the red box is the domain of NDE DHS. The units are identified as blue boxes. Files generated by the driver script are identified as the orange boxes. Files generated by each unit are identified as the yellow boxes. All the files with no boxes are the dynamic data files that NDE put to the working directory. Any input static data and system files are not shown in this figure.)

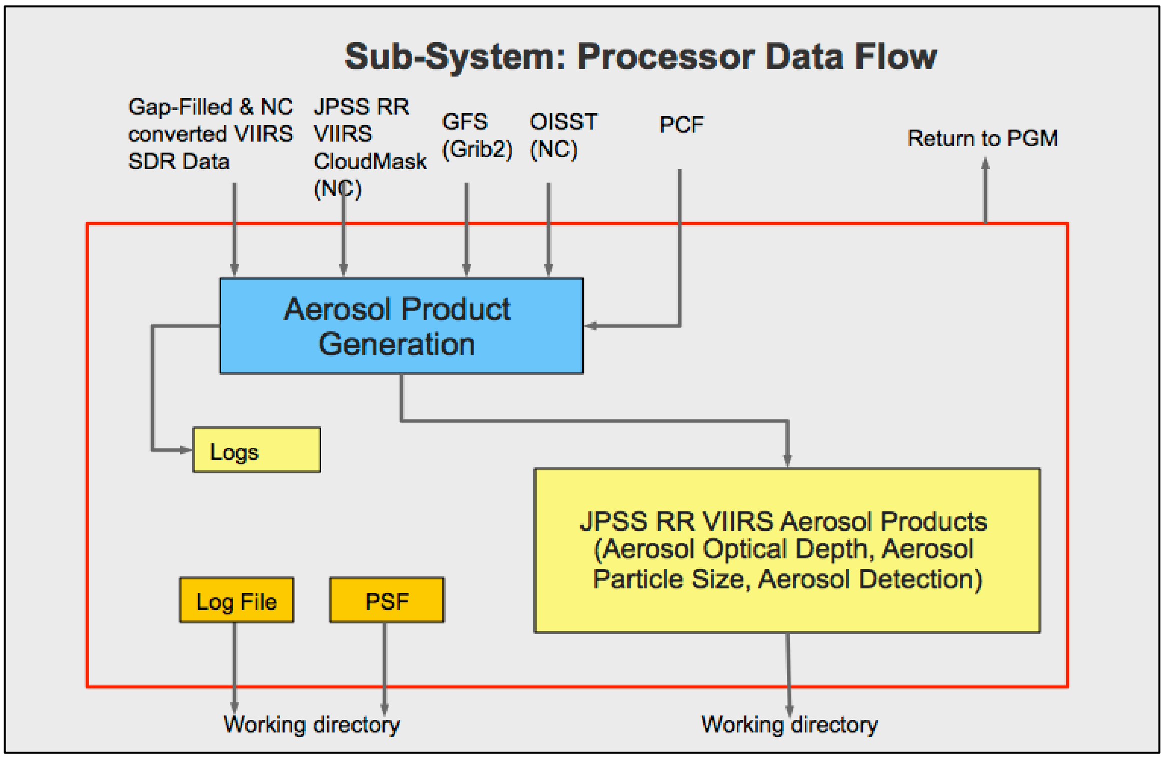


Figure 7-5: Sub-System Processor Data Flow for Aerosol Products

(The boundary of the sub-system processor domain is indicated by the red box. The light gray space around each unit within the red box is the driver script's functional domain. The light gray space outside the red box is the domain of NDE DHS. The units are identified as blue boxes. Files generated by the driver script are identified as the orange boxes. Files generated by each unit are identified as the yellow boxes. All the files with no boxes are the dynamic data files that NDE put to the working directory. Any input static data and system files are not shown in this figure.)

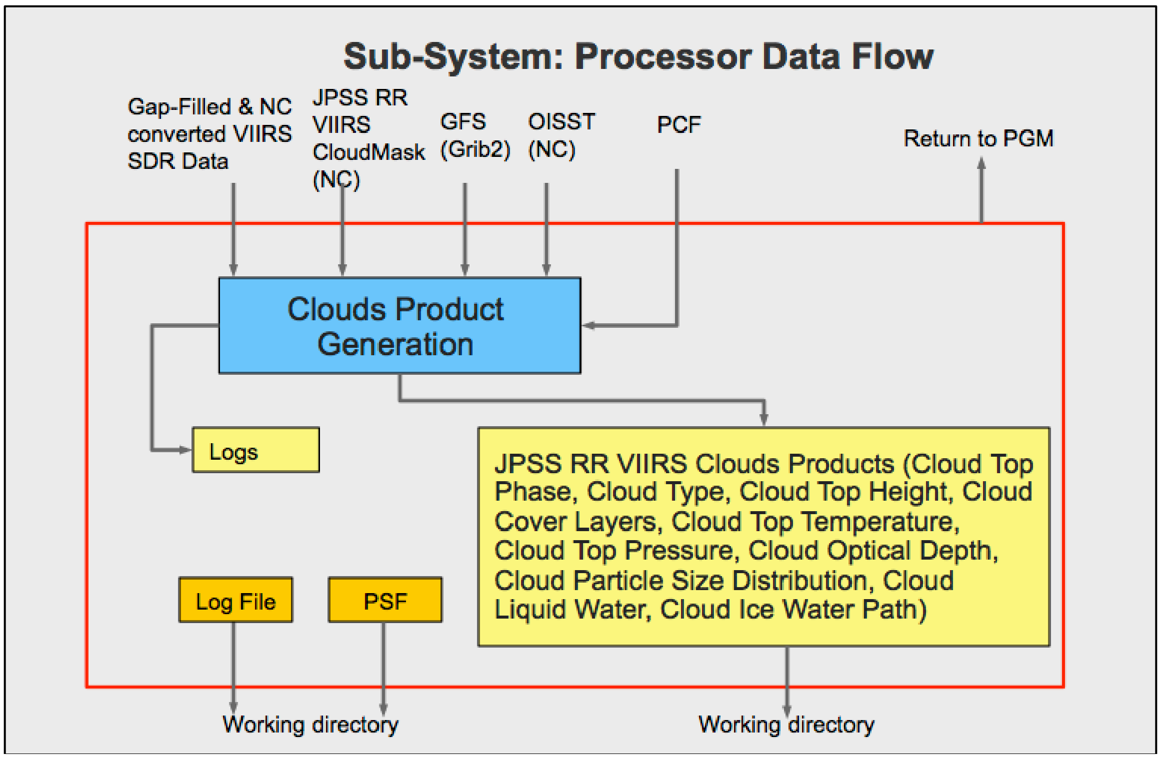


Figure 7-6: Sub-System Processor Data Flow for Clouds Products

(The boundary of the sub-system processor domain is indicated by the red box. The light gray space around each unit within the red box is the driver script's functional domain. The light gray space outside the red box is the domain of NDE DHS. The units are identified as blue boxes. Files generated by the driver script are identified as the orange boxes. Files generated by each unit are identified as the yellow boxes. All the files with no boxes are the dynamic data files that NDE put to the working directory. Any input static data and system files are not shown in this figure.)

7.1.3.1. Sub-System Preprocessor

This section describes the general purpose, design, and data flows within each unit of the sub-system Preprocessor. The association between the sub-system Preprocessor, units, subunits and the scripts is shown in Figure 3-1. Reference will be made to the files listed in Table 7-4 since they are input, intermediate, and output of these internal data flows. Each unit is wrapped by a perl script. Each of the two units is independent with respect to program execution scope. For example, one unit is not calling another. Each of the two units is called by the sub-system driver script when any of the given unit's run requirements are met. Therefore, the units are only related to each other through the dependency of their "external" data flows within the driver script's functional domain.

Each unit has its own work space under the local working directory, to put the unit level pcf, psf and log files from the run of each unit:

SDR-Data-Preparation: $working/sdr

SDR-Data-Preparation Unit

The inputs, intermediates and outputs for the SDR-Data-Preparation unit are internal to the sub-system Preprocessor and are described in Table 7-3. The main function of this unit is to convert incoming VIIRS data from HD5 to NetCDF4 and to provide the input to the downstream unit Product-Generation. The VIIRS data to be used are the SDR Moderate Resolution Band 01-16 data SVM01-16, the SDR Moderate Resolution Terrain Corrected Geolocation data GMTCO (section 7.2.1), Bands 01 – 05 of the SDR SVI01-05 data, and GITCO. Two sub-units are involved in this unit: One is the Gap-filling (wrapped by a Python script mender.py) and another is the Converter (wrapped by a Perl script jrr_preprocess_cvth52nc.pl).

Before the converter sub-unit, the SVM01-16/SVI01-05 data will be first processed with gap-filling and the gap-filled SVM/SVI data will be located at the same directory as the original HD5 data Then the gap-filled SVM01-16 (section 7.3.4.1), GMTCO, SVI01-05, and GITCO will be converted to NetCDF4 by the sub-unit Converter – the converting will be repeated within the sub-unit Converter for each single files of each granule.

Table 7-3: SDR-Data-Preparation Unit Data Flow

File	Input/ Output	Source	Description	State
JRR_PREPROCESS_SDR.pl.PCF	Input	JRR	The unit process control file created by the driver script.	Dynamic
GMTCO_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.h5	Input	NDE	VIIRS SDR Moderate Resolution Terrain Corrected Geolocation granule file in HDF5 format.	Dynamic
SVM01_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.h5	Input	NDE	VIIRS SDR Moderate Resolution Band 01 granule file in HDF5 format.	Dynamic
SVM02_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.h5	Input	NDE	VIIRS SDR Moderate Resolution Band 02 granule file in HDF5 format.	Dynamic
SVM03_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.h5	Input	NDE	VIIRS SDR Moderate Resolution Band 03 granule file in HDF5 format..	Dynamic
SVM04_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.h5	Input	NDE	VIIRS SDR Moderate Resolution Band 04 granule file in HDF5 format..	Dynamic
SVM05_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.h5	Input	NDE	VIIRS SDR Moderate Resolution Band 05 granule file in HDF5 format.	Dynamic
SVM06_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.h5	Input	NDE	VIIRS SDR Moderate Resolution Band 06 granule file in HDF5 format.	Dynamic
SVM07_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.h5	Input	NDE	VIIRS SDR Moderate Resolution Band 07 granule file in HDF5 format.	Dynamic
SVM08_npp_d????????_t???????_e???????_b?????_c?????????????	Input	NDE	VIIRS SDR Moderate Resolution Band 08 in HDF5 format.	Dynamic

???????_noaa_ops.h5				
SVM09_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.h5	Input	NDE	VIIRS SDR granule files containing Moderate Resolution Band 09 granule file in HDF5 format.	Dynamic
SVM10_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.h5	Input	NDE	VIIRS SDR Moderate Resolution Band 10 granule file in HDF5 format.	Dynamic
SVM11_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.h5	Input	NDE	VIIRS SDR Moderate Resolution Band 11 granule file in HDF5 format.	Dynamic
SVM12_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.h5	Input	NDE	VIIRS SDR Moderate Resolution Band 12 granule file in HDF5 format.	Dynamic
SVM13_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.h5	Input	NDE	VIIRS SDR Moderate Resolution Band 13 granule file in HDF5 format.	Dynamic
SVM14_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.h5	Input	NDE	VIIRS SDR Moderate Resolution Band 14 granule file in HDF5 format.	Dynamic
SVM15_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.h5	Input	NDE	VIIRS SDR Moderate Resolution Band 15 granule file in HDF5 format.	Dynamic
SVM16_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.h5	Input	NDE	VIIRS SDR Moderate Resolution Band 16 granule file in HDF5 format..	Dynamic
GITCO_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.h5	Input	NDE	VIIRS SDR Moderate Resolution Terrain Corrected Geolocation granule file in HDF5 format..	Dynamic
SVI01_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.h5	Input	NDE	VIIRS SDR Moderate Resolution Band 01 granule file in HDF5 format..	Dynamic
SVI02_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.h5	Input	NDE	VIIRS SDR Moderate Resolution Band 02 granule file in HDF5 format.	Dynamic

SVI03_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.h5	Input	NDE	VIIRS SDR Moderate Resolution Band 03 granule file in HDF5 format.	Dynamic
SVI04_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.h5	Input	NDE	VIIRS SDR Moderate Resolution Band 04 granule file in HDF5 format..	Dynamic
SVI05_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.h5	Input	NDE	VIIRS SDR Moderate Resolution Band 05 granule file in HDF5 format.	Dynamic
SVM01_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.mended.h5	Intermediate	JPSS RR	Gap-filled VIIRS SDR Moderate Resolution Band 01 granule file in HDF5 format.	Dynamic
SVM02_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.mended.h5	Intermediate	JPSS RR	Gap-filled VIIRS SDR Moderate Resolution Band 02 granule file in HDF5 format.	Dynamic
SVM03_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.mended.h5	Intermediate	JPSS RR	Gap-filled VIIRS SDR Moderate Resolution Band 03 granule file in HDF5 format.	Dynamic
SVM04_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.mended.h5	Intermediate	JPSS RR	Gap-filled VIIRS SDR Moderate Resolution Band 04 granule file in HDF5 format.	Dynamic
SVM05_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.mended.h5	Intermediate	JPSS RR	Gap-filled VIIRS SDR Moderate Resolution Band 05 granule file in HDF5 format.	Dynamic
SVM06_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.mended.h5	Intermediate	JPSS RR	Gap-filled VIIRS SDR Moderate Resolution Band 06 granule file in HDF5 format.	Dynamic
SVM07_npp_d????????_t???????_e???????_b?????_c?????????????	Intermediate	JPSS RR	Gap-filled VIIRS SDR Moderate Resolution Band 07 granule file in HDF5 format.	Dynamic

???????_noaa_ops.mended.h5				
SVM08_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.mended.h5	Intermediate	JPSS RR	Gap-filled VIIRS SDR Moderate Resolution Band 08 in HDF5 format.	Dynamic
SVM09_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.mended.h5	Intermediate	JPSS RR	Gap-filled VIIRS SDR granule files containing Moderate Resolution Band 09 granule file in HDF5 format.	Dynamic
SVM10_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.mended.h5	Intermediate	JPSS RR	Gap-filled VIIRS SDR Moderate Resolution Band 10 granule file in HDF5 format.	Dynamic
SVM11_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.mended.h5	Intermediate	JPSS RR	Gap-filled VIIRS SDR Moderate Resolution Band 11 granule file in HDF5 format.	Dynamic
SVM12_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.mended.h5	Intermediate	JPSS RR	Gap-filled VIIRS SDR Moderate Resolution Band 12 granule file in HDF5 format.	Dynamic
SVM13_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.mended.h5	Intermediate	JPSS RR	Gap-filled VIIRS SDR Moderate Resolution Band 13 granule file in HDF5 format..	Dynamic
SVM14_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.mended.h5	Intermediate	JPSS RR	Gap-filled VIIRS SDR Moderate Resolution Band 14 granule file in HDF5 format.	Dynamic
SVM15_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.mended.h5	Intermediate	JPSS RR	Gap-filled VIIRS SDR Moderate Resolution Band 15 granule file in HDF5 format.	Dynamic
SVM16_npp_d????????_t???????_e???????_b?????_c??????????????	Intermediate	JPSS RR	Gap-filled VIIRS SDR Moderate Resolution Band 16 granule file in HDF5 format.	Dynamic

???????_noaa_ops.mended.h5				
SVI01_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.mended.h5	Intermediate	JPSS RR	Gap-filled VIIRS SDR Moderate Resolution Band 01 granule file in HDF5 format..	Dynamic
SVI02_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.mended.h5	Intermediate	JPSS RR	Gap-filled VIIRS SDR Moderate Resolution Band 02 granule file in HDF5 format.	Dynamic
SVI03_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.mended.h5	Intermediate	JPSS RR	Gap-filled VIIRS SDR Moderate Resolution Band 03 granule file in HDF5 format	Dynamic
SVI04_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.mended.h5	Intermediate	JPSS RR	Gap-filled VIIRS SDR Moderate Resolution Band 04 granule file in HDF5 format.	Dynamic
SVI05_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.mended.h5	Intermediate	JPSS RR	Gap-filled VIIRS SDR Moderate Resolution Band 05 granule file in HDF5 format.	Dynamic
GMTCO_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.nc	Output	JPSS RR	VIIRS SDR Moderate Resolution Terrain Corrected Geolocation granule file in NetCDF4 format.	Dynamic
SVM01_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.nc	Output	JPSS RR	VIIRS SDR Moderate Resolution Band 01 granule file in NetCDF4 format.	Dynamic
SVM02_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.nc	Output	JPSS RR	VIIRS SDR Moderate Resolution Band 02 granule file in NetCDF4 format.	Dynamic
SVM03_npp_d????????_t???????_e???????_b?????_c?????????????	Output	JPSS RR	VIIRS SDR Moderate Resolution Band 03 granule file in NetCDF4 format.	Dynamic

???????_noaa_ops.nc				
SVM04_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.nc	Output	JPSS RR	VIIRS SDR Moderate Resolution Band 04 granule file in NetCDF4 format.	Dynamic
SVM05_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.nc	Output	JPSS RR	VIIRS SDR Moderate Resolution Band 05 granule file in NetCDF4 format. .	Dynamic
SVM06_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.nc	Output	JPSS RR	VIIRS SDR Moderate Resolution Band 06 granule file in NetCDF format..	Dynamic
SVM07_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.nc	Output	JPSS RR	VIIRS SDR Moderate Resolution Band 07 granule file in NetCDF4 format..	Dynamic
SVM08_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.nc	Output	JPSS RR	VIIRS SDR Moderate Resolution Band 08 granule file in NetCDF4 format.	Dynamic
SVM09_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.nc	Output	JPSS RR	VIIRS SDR Moderate Resolution Band 09 granule file in NetCDF4 format..	Dynamic
SVM10_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.nc	Output	JPSS RR	VIIRS SDR Moderate Resolution Band 10 granule file in NetCDF4 format.	Dynamic
SVM11_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.nc	Output	JPSS RR	VIIRS SDR Moderate Resolution Band 11 granule file in NetCDF4 format.	Dynamic
SVM12_npp_d????????_t???????_e???????_b?????_c??????????????	Output	JPSS RR	VIIRS SDR granule files containing Moderate Resolution Band 12 in NetCDF4 format	Dynamic

???????_noaa_ops.nc				
SVM13_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.nc	Output	JPSS RR	VIIRS SDR Moderate Resolution Band 13 granule file in NetCDF4 format.	Dynamic
SVM14_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.nc	Output	JPSS RR	VIIRS SDR Moderate Resolution Band 14 granule file in NetCDF4 format.	Dynamic
SVM15_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.nc	Output	JPSS RR	VIIRS SDR Moderate Resolution Band 15 granule file in NetCDF4 format..	Dynamic
SVM16_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.nc	Output	JPSS RR	VIIRS SDR Moderate Resolution Band 16 granule file in NetCDF4 format.	Dynamic
GITCO_npp_d????????_t???????_e???????_b?????_c?????????????????????_noaa_ops.nc	Output	JPSS RR	VIIRS SDR Moderate Resolution Terrain Corrected Geolocation granule file in NetCDF4 format.	Dynamic
SVI01_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.nc	Output	JPSS RR	VIIRS SDR Moderate Resolution Band 01 granule file in NetCDF4 format.	Dynamic
SVI02_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.nc	Output	JPSS RR	VIIRS SDR Moderate Resolution Band 02 granule file in NetCDF4 format..	Dynamic
SVI03_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.nc	Output	JPSS RR	VIIRS SDR Moderate Resolution Band 03 granule file in NetCDF4 format.	Dynamic
SVI04_npp_d????????_t???????_e???????_b?????_c??????????????	Output	JPSS RR	VIIRS SDR Moderate Resolution Band 04 granule file in NetCDF4 format.	Dynamic

??????_noaa_ops.nc				
SVI05_npp_d????????_t???????_e???????_b?????_c????????????????????_noaa_ops.nc	Output	JPSS RR	VIIRS SDR Moderate Resolution Band 05 granule file in NetCDF4 format..	Dynamic
JRR_PREPROCESS_SDR.pl.log	Output	JPSS RR	This is the run log containing the standard output and return status of the unit processes. When an error happens while running a sub-unit script, the log information from that sub-unit will be appended to it.	Dynamic
convert_viirs_hd5_2_nc4.pl.log	Output	JPSS RR	This is the run log containing the standard output and return status of the running of convert_viirs_hd5_2_nc4.pl	Dynamic
h5augjpss.log	Output	JPSS RR	This is the run log containing the standard output of utility h5augjpss – updated every granule	Dynamic
mender.log	Output	JPSS RR	This is the run log containing the standard output of script mender.py – updated every granule	Dynamic

The unit only starts to convert the granule data if all the 23 VIIRS SDR, GMTCO, SVI)1-05, and GITCO data files exist. If there are any missing files, the unit returns an error message.

The SDR-Data-Preparation unit level PCF (unit 1 PCF: JRR_PREPROCESS_SDR.pl.PCF) file only provides the following fields: the name of the working directory, created by NDE job_coverage_start/end, supplied by NDE Gapfilling option flag, “ON”, upon current requirements the location of scripts/bin, supplied by NDE the dynamic VIIRS SDR input data (23), listed in Table 7-3.

1) *OPS_CONFIG* -- It is the sub-system’s PCF file, supplied by NDE,

passed in by the driver script

When the SDR-Data-Preparation unit completes, it returns a status to the driver script indicating success or failure. The output consists of those dynamic files identified in Table 7-3. These output files are the unit and sub-unit log files, the VIIRS SDR netCDF4 data files, and the unit level PSF (JRR_PREPROCESS_SDR.pl.PSF) for the given run of the unit. The VIIRS SDR in NetCDF4 data files with full path are written into the unit PSF by the unit. These NetCDF4 output files will be used as dynamic input for the downstream unit clouds, aerosol, and cryosphere generation.

7.1.3.2. Sub-System Product-Generator

This section describes the general purpose, design, and data flows within the unit of the sub-system Product-Generator. The association between the sub-system Product-Generator, unit, subunits and the scripts is shown in Figure 3-2. Reference will be made to the files listed in Table 7-4 since they are input, intermediate and output of these internal data flows. The unit is called by the sub-system's driver script. The unit is wrapped by a perl script. The unit is called by the sub-system driver script when the unit's run requirements are met.

Figure 7-7, Figure 7-8, Figure 7-9, and Figure 7-10 show the unit's data flow for the cryosphere snow, cryospheree ice, aerosol, and clouds products, respectively. The boundary of the unit's functional domain is indicated by a box with the dark bolded lines. The blue boxes represent the sub-units.   The white boxes are files handled within the unit and are input output of the sub-unit. The orange boxes are files generated by the unit wrapper scrip. Note that the unit receives a unit level PCF (produced by the driver script) and produces a unit level PSF which are represented by the orange boxes.

The unit has its own work space under the local working directory, to put the unit level pcf, psf and log files from the run of each unit:
JRR-Product-Generation: $working/pw

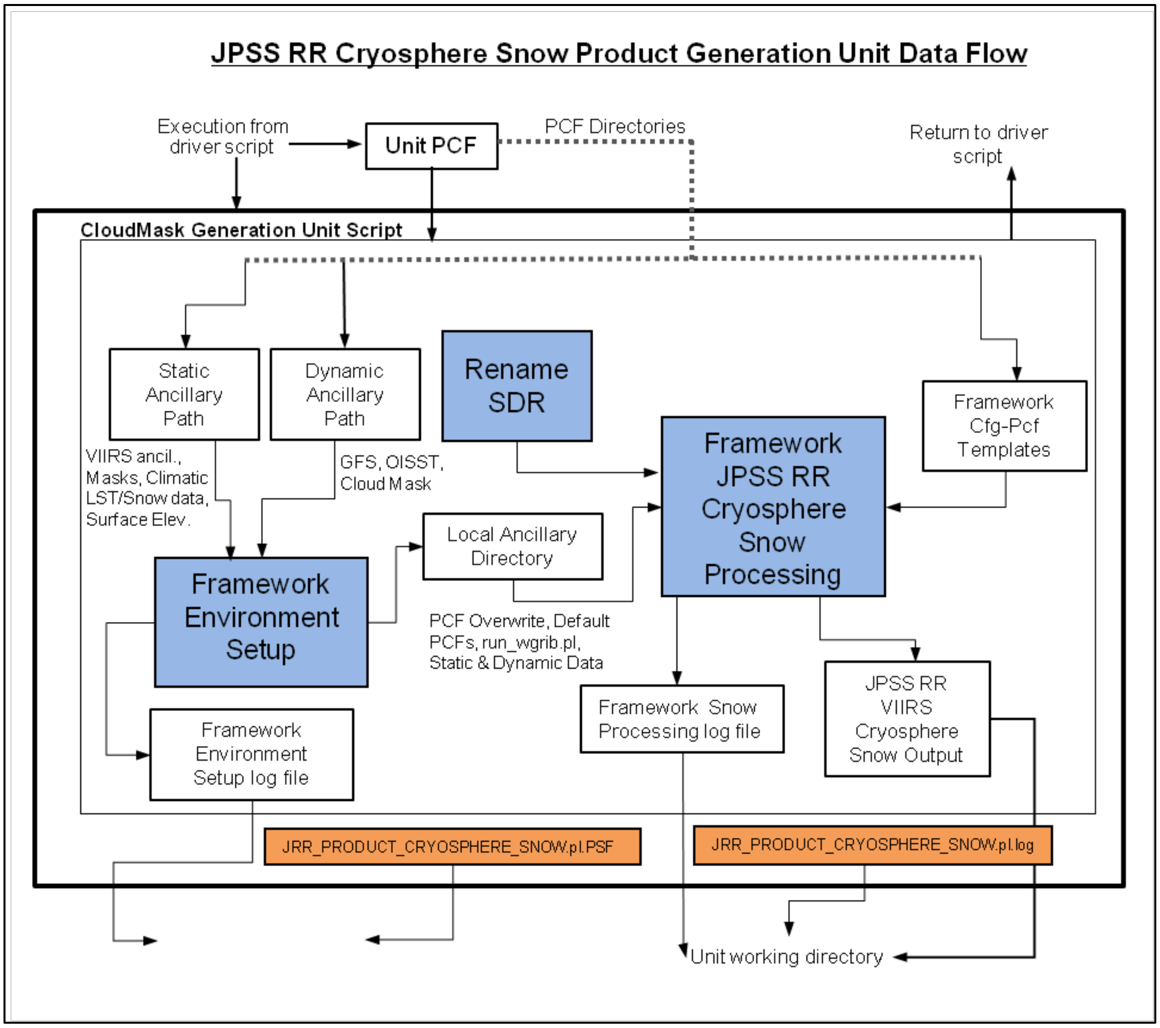


Figure 7-7: Framework Cryosphere Snow Product Generation Unit Data Flow

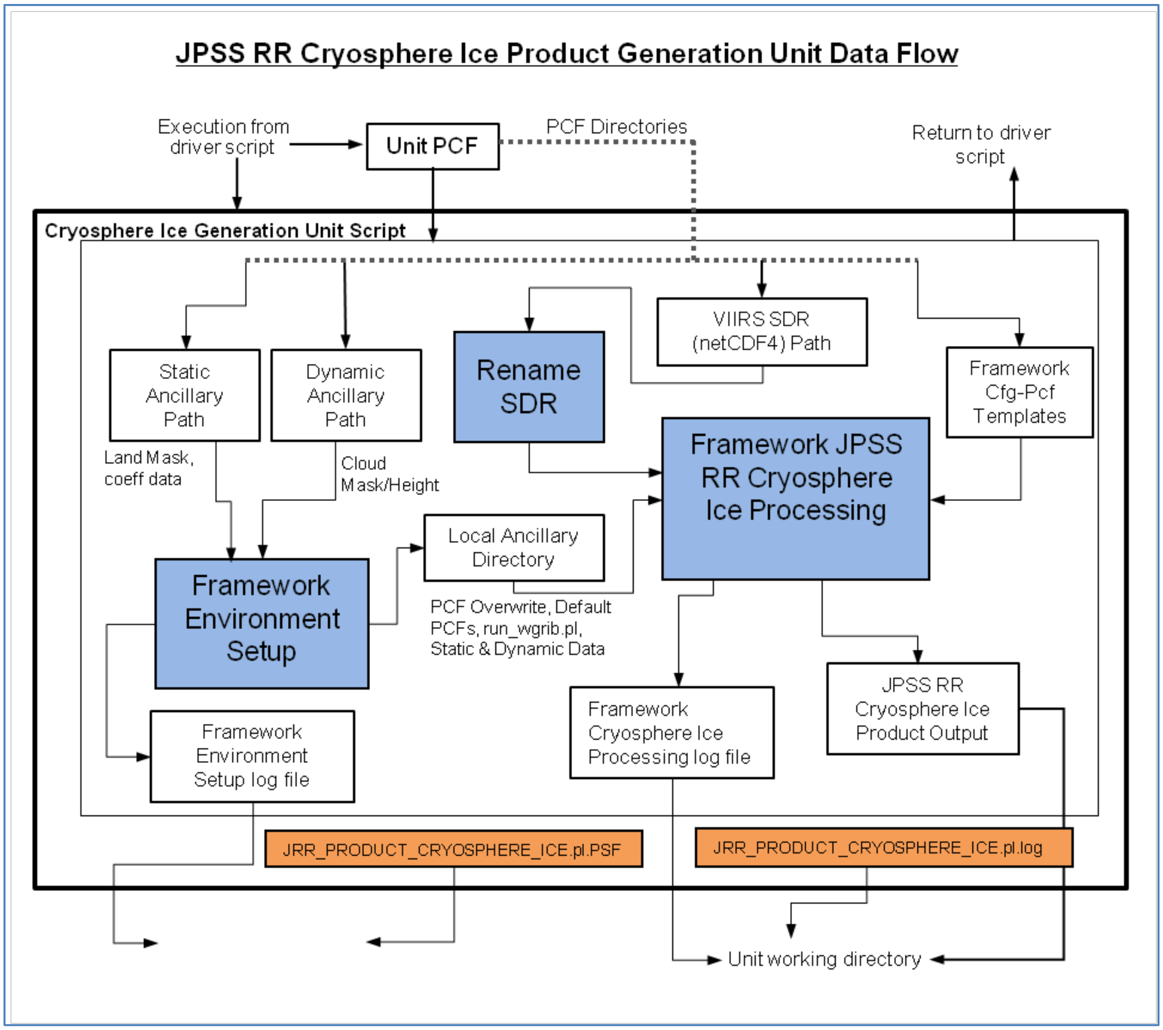


Figure 7-8: Framework Cryosphere Ice Product Generation Unit Data Flow

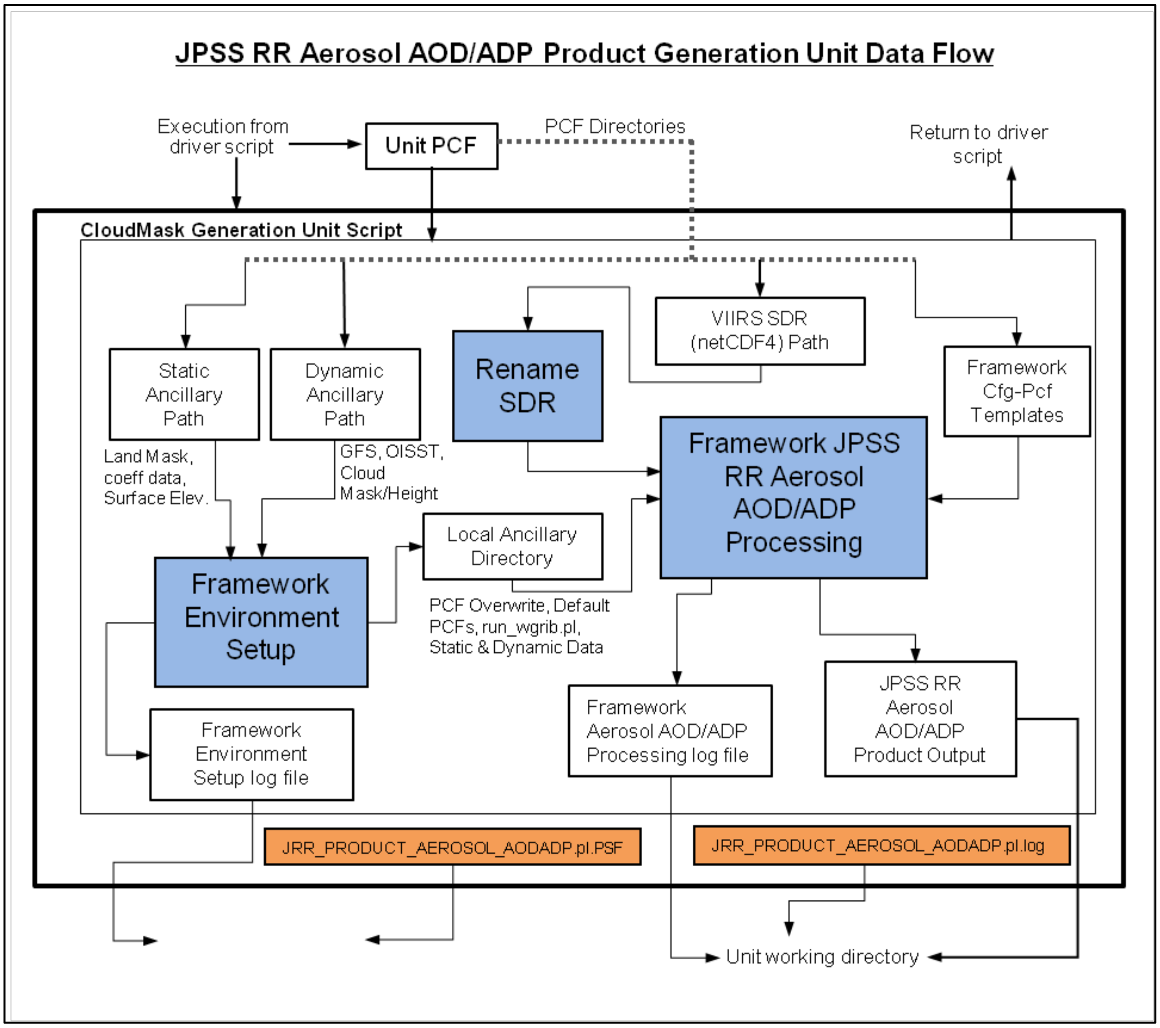


Figure 7-9: Framework Aerosol Product Generation Unit Data Flow

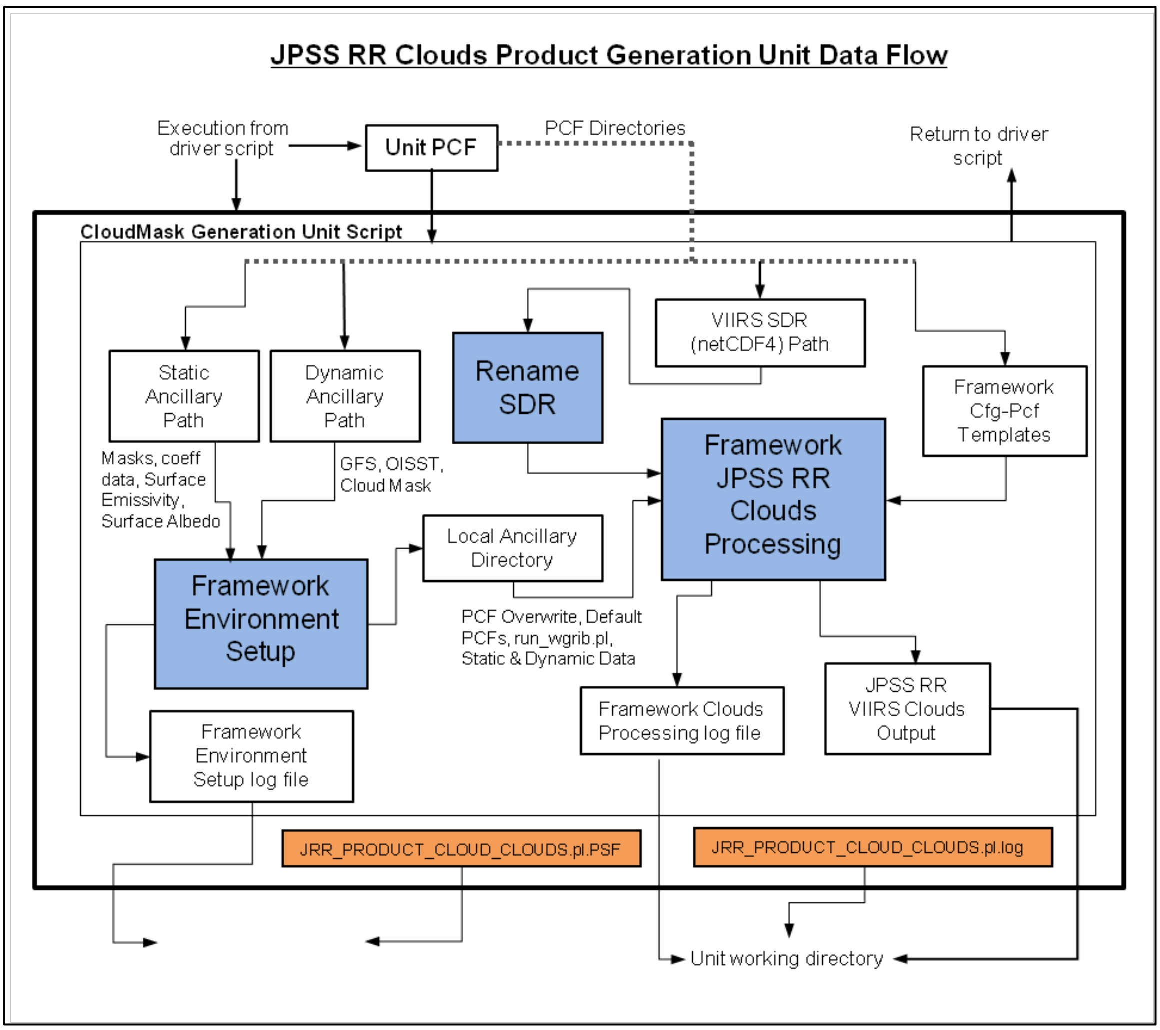


Figure 7-10: Framework Clouds Product Generation Unit Data Flow

JRR-Product-Generation unit

The JRR-Product-Generation unit will be invoked only when all three NetCDF4 files are available. This unit generates the VIIRS JPSS RR product. The details of the inputs, intermediates and outputs for the JPSS-Product-Generation unit are described in Table 7-4. In addition to NetCDF4 files (section 7.4.1), the other dynamic data required by the JRR-Product-Generation unit is the GFS 6-hour forecast GRIB2 files in 0.5 degree resolution (section 7.2.2.14) and 0.25 degree OISST data. A number of static data files are also required by the unit. These static data files include the land mask (section 7.2.2.1). The unit also requires template AIT-framework level configuration file and process control files (template framework-CFG and framework- PCFs, section 7.2.3) to dynamically generate framework-CFG and framework-PCFs (section 7.2.3.1) corresponding to the remapped NetCDF files. The unit requires common framework-PCF (section 7.2.3.2) as well.

Table 7-4: JPSS RR-Product-Generation Unit Data Flow

File	Input/ Output	Source	Description	State
JRR_PRODUCT_CLOUD_MASK.pl.PCF	Input	NDE	One of the JPSS RR sub-system Product-Generator PCF supplied by the NDE PGM.	Dynamic
JRR_PRODUCT_CLOUD_CLOUDS.pl.PCF	Input	NDE	One of the JPSS RR sub-system Product-Generator PCF supplied by the NDE PGM.	Dynamic
JRR_PRODUCT_AEROSOL_VOLASH.pl.PCF	Input	NDE	One of the JPSS RR sub-system Product-Generator PCF supplied by the NDE PGM.	Dynamic
JRR_PRODUCT_AEROSOL_AODADP.pl.PCF	Input	NDE	One of the JPSS RR sub-system Product-Generator PCF supplied by the NDE PGM.	Dynamic
JRR_PRODUCT_CRYOSPHERE_SNOW.pl.PCF	Input	NDE	One of the JPSS RR sub-system Product-Generator PCF supplied by the NDE PGM.	Dynamic
JRR_PRODUCT_CRYOSPHERE_ICE.pl.PCF	Input	NDE	One of the JPSS RR sub-system Product-Generator PCF supplied by the NDE PGM.	Dynamic
JRR_PREPROCESS_SDR.pl.PCF	Input	JRR	The unit process control file created by the driver script.	Dynamic

gfs.t??z.pgrb2f.0p50.f???.????????	Input	NCEP	The GFS 6-hour forecast file in GRIB2 format, 0.5 degree resolution. There are multiple such files.	Dynamic
d20150402_t1054237_e1055479_b17764_cm.cfg	Input	STAR	A template configure file to generate AIT-framework configure file to produce the cloud mask products	Dynamic
d20150402_t1054237_e1055479_b17764_clouds.cfg	Input	STAR	A template configure file to generate AIT-framework configure file to produce the other cloud products	Dynamic
d20150402_t1054237_e1055479_b17764_aodadp.cfg	Input	STAR	A template configure file to generate AIT-framework configure file to produce the other cloud products	Dynamic
d20150402_t1054237_e1055479_b17764_volash.cfg	Input	STAR	A template configure file to generate AIT-framework configure file to produce the other cloud products	Dynamic
Land_Mask.pcf	Input	STAR	AIT-framework PCF file to process Land Mask data	Static
OISST_Daily_QtrDeg.pcf	Input	STAR	AIT-framework PCF file to process OISST data	Static
NWP_Data_grib2_0.5deg.pcf	Input	STAR	AIT-framework PCF file to process NWP Grib2 data	Static
Surface_Elevation.pcf	Input	STAR	AIT-framework PCF file to process surface elevation data	Static
Surface_Emissivity.pcf	Input	STAR	AIT-framework PCF file to process surface emissivity data	Static
CRTM.pcf	Input	STAR	AIT-framework PCF file to process CRTM data	Static
Coast_Mask.pcf	Input	STAR	AIT-framework PCF file to process coast mask data	Static
Desert_Mask.pcf	Input	STAR	AIT-framework PCF file to process desert mask data	Static
Snow_Mask.pcf	Input	STAR	AIT-framework PCF file to process snow mask data	Static
Climatic_LST.pcf	Input	STAR	AIT-framework PCF file to process climatic LST data	Static
Pseudo_Emissivity.pcf	Input	STAR	AIT-framework PCF file to process pseudo emissivity data	Static

Surface_Type.pcf	Input	STAR	AIT-framework PCF file to process surface type data	Static
Surface_Albedo.pcf	Input	STAR	AIT-framework PCF file to process surface albedo data	Static
NPP_BAYES_Cloud_Mask.pcf	Input	STAR	A template AIT-framework PCF file to generate Bayes cloud mask PCF file	Static
NPP_BAYES_Cloud_Mask_NW.pcf	Input	STAR	A template AIT-framework PCF file to generate Bayes cloud mask PCF file	Static
NPP_BAYES_Cld_Cld_Phase.pcf	Input	STAR	A template AIT-framework PCF file to generate cloud phase da PCF file ta	Static
NPP_BAYES_Cld_Height.pcf	Input	STAR	A template AIT-framework PCF file to generate cloud height PCF file	Static
NPP_BAYES_Cld_NCOMP.pcf	Input	STAR	A template AIT-framework PCF file to generate NCOMP PCF file	Static
NPP_BAYES_Cloud_Micro_Day.pcf	Input	STAR	A template AIT-framework PCF file to process DCOMP PCF file	Static
NPP_BAYES_Cld_Height_NW.pcf	Input	STAR	A template AIT-framework PCF file to generate cloud height PCF file	Static
AWG_AER_AOD.pcf	Input	STAR	A template AIT-framework PCF file to generate aerosol optical depth PCF file	Static
NPP_VIIRS_AERADP.pcf	Input	STAR	A template AIT-framework PCF file to generate aerosol optical depth PCF file	Static
Volcano_Mask.pcf	Input	STAR	A template AIT-framework PCF file to generate volcano mask PCF file	Static
NPP_VOLCANIC_ASH.pcf	Input	STAR	A template AIT-framework PCF file to generate volcanic ash PCF file	Static
AWG_Ice_Conc_VIIRS_BAYES.pcf	Input	STAR	A template AIT-framework PCF file to generate ice concentration PCF file	Static
AWG_Ice_Age_VIIRS_BAYES.pcf	Input	STAR	A template AIT-framework PCF file to generate ice age PCF file	Static
NPP_VIIRS_SNOW_COVER.pcf	Input	STAR	A template AIT-framework PCF file to generate ice	Static

			concentration PCF file	
npp_viirs_ancil.Mbands.nc	Input	STAR	Ancillary data required by the VIIRS SDR reader of AIT framework	Static
npp_viirs_ancil.Ibands.nc	Input	STAR	Ancillary data required by the VIIRS SDR reader of AIT framework	Static
lw_geo_2001001_v03m.nc	Input	STAR	Global 1km land/water used for MODIS collection 5	Static
GLOBE_1km_digelev.nc	Input	STAR	Digital surface elevation at 1km resolution	Static
global_emiss_intABI_???????.nc	Input	STAR	Surface emissivity at 5km resolution. There are multiple of such files.	Static
AerosolCoeff.bin	Input	STAR	CTRM Coeff	Static
CloudCoeff.bin	Input	STAR	CTRM Coeff	Static
EmisCoeff.bin	Input	STAR	CTRM Coeff	Static
viirs-i_npp.SpcCoeff.bin	Input	STAR	CTRM Coeff	Static
viirs-i_npp.TauCoeff.bin	Input	STAR	CTRM Coeff	Static
viirs-m_npp.SpcCoeff.bin	Input	STAR	CTRM Coeff	Static
viirs-m_npp.TauCoeff.bin	Input	STAR	CTRM Coeff	Static
viirs-m_npp.SpcCoeff.bin	Input	STAR	CTRM Coeff	Static
viirs-m_npp.TauCoeff.bin	Input	STAR	CTRM Coeff	Static
coast_mask_1km.nc	Input	STAR	Global 1km land/water used for MODIS collection 5	Static
gl-latlong-1km-landcover.nc	Input	STAR	Surface type mask based on AVHRR at 1km resolution	Static
global_emiss_intABI_2005???.nc	Input	STAR	Seebor global emissions from 2005. There are multiple of such files.	Static
AlbMap.WS.c004.v2.0.2004.???.0.659_x4.nc	Input	STAR	White sky reflectance at 0.659 um.. There are multiple of such files.	Static
AlbMap.WS.c004.v2.0.2004.???.1.64_x4.nc	Input	STAR	White sky reflectance at 1.64 um. There are multiple of such files.	Static
AlbMap.WS.c004.v2.0.2004.???.2.13_x4.nc	Input	STAR	White sky reflectance at 2.13 um. There are multiple of such files.	Static
viirs_default_nb_cloud_mask_lut.nc	Input	STAR	Ancillary data required by the AIT-framework	Static
VIIRS_coefs.nc	Input	STAR	Coefficients for NCOMP	Static
volcano_mask_1km.nc	Input	STAR	Volcano mask at 1km resolution	Static
VIIRS_Aerosol_Lut.nc	Input	STAR	VIIRS aerosol look up table	Static

			data	
VIIRS_Sunglint_Lut.nc	Input	STAR	VIIRS sun glint look up table data	Static
VIIRS_Coeff.nc	Input	STAR	VIIRS aerosol coefficients	Static
VIIRS_BrightSfc_Refl_Coef.nc	Input	STAR	VIIRS surface reflectance coefficients	Static
snow_occurrence_latlon_week_??.nc	Input	STAR	Weekly snow occurrence ancillary file. There are multiple of such files.	Static
climatic_lst_month_??.nc	Input	STAR	Monthly climatic LST ancillary file. There are multiple of such files.	Static
IceSrfTempCoeff_NPP_VIIRS.nc	Input	STAR	Ice surface temperature coefficients.	Static
AITA_INPUT_Coefficients_ResiFlux.nc	Input	STAR	Ice age coefficients.	Static
VIIRS_ch?_ref_lut_ice_cld.nc	Input	STAR	VIIRS DCOMP lookup table for ice	Static
VIIRS_ch?_ref_lut_wat_cld.nc	Input	STAR	ViIRS DCOMP lookup table for water	Static
npp_viirs_sdr_d????????_t???????_e???????_b?????_cm.pcf	Intermediate	STAR	An AIT-framework level PCF file to generate cloud mask PCF file	Dynamic
npp_viirs_sdr_d????????_t???????_e???????_b?????_clouds.pcf	Intermediate	STAR	An AIT-framework level PCF file to generate clouds products PCF file	Dynamic
npp_viirs_sdr_d????????_t???????_e???????_b?????_volash.pcf	Intermediate	STAR	An AIT-framework level PCF file to generate volcanic ash PCF file	Dynamic
D????????_t???????_e???????_b?????_aodadp.cfg	Intermediate	STAR	An AIT-framework level PCF file to generate aerosol products PCF file	Dynamic
npp_viirs_sdr_d????????_t???????_e???????_b?????_snow.pcf	Intermediate	STAR	An AIT-framework level PCF file to generate volcanic ash PCF file	Dynamic
npp_viirs_sdr_d????????_t???????_e???????_b?????_ice.pcf	Intermediate	STAR	An AIT-framework level PCF file to generate aerosol products PCF file	Dynamic
JRR_PREPROCESS_SDR.pl.log	Output	STAR	The run log generated by the running of the JRR preprocessor	Dynamic
jrr_preprocess_cvth52nc.pl.log	Output	STAR	The run log generated by the running of the hdf to netCDF converter	Dynamic

jrr_preprocess_gapfill_I.pl.log	Output	STAR	The run log generated by the running of the gap filling algorithm for I-Band	Dynamic
jrr_preprocess_gapfill_M.pl.log	Output	STAR	The run log generated by the running of the gap filling algorithm for M-Band	Dynamic
h5augjpss.log	Output	STAR	The run log generated by the running of h5augjpss	Dynamic
jrr_product_env4frmwk.pl.log	Output	STAR	The run log generated by the running of setting the environment variables	Dynamic
jrr_product_sdr4frmwk.pl.log	Output	STAR	The run log generated by the running of sdr data for the framework	Dynamic
mender.log	Output	STAR	The run log generated by the running of mender	Dynamic
jrr_product_cld_mask.pl.log	Output	JRR	The run log generated by the running of the AIT-framework program	Dynamic
JRR_PRODUCT_CLOUD_MASK.pl.log	Output	JRR	This is the run log containing the standard output and return status of the unit processes. When error happens on running a sub-unit script, the log information from that sub-unit will be appended to it.	Dynamic
jrr_product_cld_mask.pl.log	Output	JRR	The run log generated by the running of the AIT-framework program	Dynamic
viirs_cm.log	Output	JRR	The run log generated by the running of the AIT-framework program	Dynamic
JRR_PRODUCT_CLOUD_CLOUDS.pl.log	Output	JRR	This is the run log containing the standard output and return status of the unit processes. When error happens on running a sub-unit script, the log information from that sub-unit will be appended to it.	Dynamic
jrr_product_cld_clouds.pl.log	Output	JRR	The run log generated by the running of the AIT-framework program	Dynamic
viirs_clouds.log	Output	JRR	The run log generated by the running of the AIT-framework program	Dynamic

JRR_PRODUCT_AEROSOL_AODADP.pl.log	Output	JRR	This is the run log containing the standard output and return status of the unit processes. When error happens on running a sub-unit script, the log information from that sub-unit will be appended to it.	Dynamic
jrr_product_aer_aodadp.pl.log	Output	JRR	The run log generated by the running of the AIT-framework program	Dynamic
viirs_aodadp.log	Output	JRR	The run log generated by the running of the AIT-framework program	Dynamic
JRR_PRODUCT_AEROSOL_VOLASH.pl.log	Output	JRR	This is the run log containing the standard output and return status of the unit processes. When error happens on running a sub-unit script, the log information from that sub-unit will be appended to it.	Dynamic
jrr_product_aer_volash.pl.log	Output	JRR	The run log generated by the running of the AIT-framework program	Dynamic
viirs_volash.log	Output	JRR	The run log generated by the running of the AIT-framework program	Dynamic
JRR_PRODUCT_CRYOSPHERE_SNOW.pl.log	Output	JRR	This is the run log containing the standard output and return status of the unit processes. When error happens on running a sub-unit script, the log information from that sub-unit will be appended to it.	Dynamic
jrr_product_cryos_snow.pl.log	Output	JRR	The run log generated by the running of the AIT-framework program	Dynamic
viirs_snow.log	Output	JRR	The run log generated by the running of the AIT-framework program	Dynamic
JRR_PRODUCT_CRYOSPHERE_ICE.pl.log	Output	JRR	This is the run log containing the standard output and return status of the unit processes. When error happens on running a sub-unit script, the log information from that sub-	Dynamic

			unit will be appended to it.	
jrr_product_cryos_ice.pl.log	Output	JRR	The run log generated by the running of the AIT-framework program	Dynamic
viirs_ice.log	Output	JRR	The run log generated by the running of the AIT-framework program	Dynamic
JRR_PREPROCESS_SDR.pl.PSF	Output	JRR	This is the Product Status File containing the preprocessor file names, including the full file path	Dynamic
JRR_PRODUCT_CLOUD_MASK.pl.PSF	Output	JRR	This is the Product Status File containing the cloud mask file names, including the full file path	Dynamic
JRR_PRODUCT_CLOUD_CLOUDS.pl.PSF	Output	JRR	This is the Product Status File containing the cloud file names, including the full file path	Dynamic
JRR_PRODUCT_AEROSOL_AODADP.pl.PSF	Output	JRR	This is the Product Status File containing the aerosol file names, including the full file path	Dynamic
JRR_PRODUCT_AEROSOL_VOLASH.pl.PSF	Output	JRR	This is the Product Status File containing the volcanic ash file names, including the full file path	Dynamic
JRR_PRODUCT_CRYOSPHERE_SNOW.pl.PSF	Output	JRR	This is the Product Status File containing the snow file names, including the full file path	Dynamic
JRR_PRODUCT_CRYOSPHERE_ICE.pl.PSF	Output	JRR	This is the Product Status File containing the ice file names, including the full file path	Dynamic

When the NetCDF4 format data files are available to the driver script, the driver script executes the JPSS RR-Product-Generation unit. The unit level PCF for the JPSS RR-Product-Generation units (JRR_PRODUCT_CLOUD_MASK.pl.PCF, JRR_PRODUCT_CLOUD_CLOUDS.pl.PCF, JRR_PRODUCT_AEROSOL_VOLASH.pl.PCF, JRR_PRODUCT_AEROSOL_AODADP.pl.PCF, JRR_PRODUCT_CRYOSPHERE_SNOW.pl.PCF, JRR_PRODUCT_CRYOSPHERE_ICE.pl.PCF) are generated in its working directory by the driver script. The unit assumes that its unit level PCF will be available in its working directory when it runs. The unit PCF contains:

1) *OPS_CONFIG* -- It is the system's PCF file, supplied by NDE, passed in by the driver script
2) *PSF*A text file contains NetCDF file name, full path

The unit generates the JPSS RR products by running AIT-framework program: It reads templates of AIT-framework level CFG and PCF files. Then the dynamic AIT-framework level configuration file and processing control files (framework-CFG and framework-PCFs) for JPSS RR are created by the unit (although there is a sub-unit Framework Cfg-pcf-generator, there is no wrapped script for this, but done by the unit script). The input data files are specified in these framework-CFG and framework-PCFs. The framework-CFG file contains the location of where to put the framework output and framework-PCFs to be used. Then the sub-unit Framework-JPSS RR-Generation is invoked (by running Framework program with that specific framework-CFG file), JPSS RR products are generated. The AIT-framework output of JPSS RR products are in NetCDF4 format. The file name of the product with full path will be recorded in the unit PSF files (JRR_PRODUCT_CLOUD_MASK.pl.PSF, JRR_PRODUCT_CLOUD_CLOUDS.pl.PSF, JRR_PRODUCT_AEROSOL_VOLASH.pl.PSF, JRR_PRODUCT_AEROSOL_AODADP.pl.PSF, JRR_PRODUCT_CRYOSPHERE_SNOW.pl.PSF, JRR_PRODUCT_CRYOSPHERE_ICE.pl.PSF) if the run is successful. The JRR-Product-Generation unit then exits. The return status is obtained by the driver script.

7.2. Input Data Files

This section describes the input data required by the JPSS RR system. These data includes the satellite data; the ancillary data required by the AIT-framework to generate JPSS RR products in the JRR-Product-Generation units. All of these files are defined in the system's PCF files (JRR_PRODUCT_CLOUD_MASK.pl.PCF, JRR_PRODUCT_CLOUD_CLOUDS.pl.PCF,

JRR_PRODUCT_AEROSOL_VOLASH.pl.PCF,
JRR_PRODUCT_AEROSOL_AODADP.pl.PCF,
JRR_PRODUCT_CRYOSPHERE_SNOW.pl.PCF,
JRR_PRODUCT_CRYOSPHERE_ICE.pl.PCF) through the File-Handle_Name (the left hand side of the equal sign in the PCF file).

The JPSS RR system also requires input text files in order to run the AIT-framework program. These are the AIT-framework configuration files and process control files. There are three types of these files: the template Framework-CFG, the template Framework-PCF, and the common Framework-PCF. The system's PCF file only defines the location where the three types of files are located, but not the file names:
CFG_TEMPL_DIR – location to put the template Framework-CFG
PCF_TEMPL_DIR – location t put the template Framework-PCF
PCF_OVERWRITES – location to put common Framework-PCF

An AIT-framework CFG file is required to run the AIT-framework program and in the CFG file, a number of AIT-framework PCF files are specified. AIT-framework program is called in the JRR-Product-Generation units. The data is passed to the AIT-framework program through the Framework-CFG and Framework-PCF files by specifying the data files in these Framework-CFG and PCF.

7.2.1. VIIRS Data Files

23 input IDPS output data files are required to run JPSS RR system. They are the VIIRS Science Data Records (SDR) Moderate Resolution band 01 -16 (SVM01-16), Terrain Corrected Geolocation data (GMTCO), SVI01-05, and GITCO. These data are the satellite input to the JPSS RR system. They are also the input to the SDR-Data-Preparation unit. All of these files are in HDF5 format and are handled by NDE DHS. The Table 7-5 is the File-Handle-Name in the system's PCF (section 4.1.1) and the corresponding satellite data files.

The details of these input data files are presented in the NPOESS Common Data Format Control Book (2009).

Table 7-5: JPSS RR System Input Satellite Data

File Handle Name in PCF	**File Name**	**Description**	**Size (MB)**
gmtco	*GMTCO_npp_d${Year}${Month}${Day}_t${Granule_Start}_e${Granule_End}_b${Dump}_c${ Data_Creation }_noaa_ops.h5*	VIIRS SDR granule files containing Moderate Bands SDR Terrain Corrected Geolocation in HDF5 format	~78
svm01	*SVM01_npp_d${Year}${Month}${Day}_t${Granule_Start}_e${Granule_End}_b${Dump}_c${ Data_Creation }_noaa_ops.h5*	VIIRS SDR granule files containing Moderate Resolution Band 01 in HDF5 format	~12
svm02	*SVM02_npp_d${Year}${Month}${Day}_t${Granule_Start}_e${Granule_End}_b${Dump}_c${ Data_Creation }_noaa_ops.h5*	VIIRS SDR granule files containing Moderate Resolution Band 02 in HDF5 format	~12
svm03	*SVM03_npp_d${Year}${Month}${Day}_t${Granule_Start}_e${Granule_End}_b${Dump}_c${ Data_Creation }_noaa_ops.h5*	VIIRS SDR granule files containing Moderate Resolution Band 03 in HDF5 format	*~17*
svm04	*SVM04_npp_d${Year}${Month}${Day}_t${Granule_Start}_e${Granule_End}_b${Dump}_c${ Data_Creation }_noaa_ops.h5*	VIIRS SDR granule files containing Moderate Resolution Band 04 in HDF5 format	*~17*
svm05	*SVM05_npp_d${Year}${Month}${Day}_t${Granule_Start}_e${Granule_End}_b${Dump}_c${ Data_Creation }_noaa_ops.h5*	VIIRS SDR granule files containing Moderate Resolution Band 05 in HDF5 format	*~17*
svm06	*SVM06_npp_d${Year}${Month}${Day}_t${Granule_Start}_e${Granule_End}_b${Dump}_c${ Data_Creation }_noaa_ops.h5*	VIIRS SDR granule files containing Moderate Resolution Band 06 in HDF5 format	*~12*
svm07	*SVM07_npp_d${Year}${Month}${Day}_t${Granule_Start}_e${Granule_End}_b${*	VIIRS SDR granule files containing Moderate	*~17*

	Dump}_c${ Data_Creation }_noaa_ops.h5	Resolution Band 07 in HDF5 format	
svm08	*SVM08_npp_d${Year}${Month}${Day}_t ${Granule_Start}_e${Granule_End}_b${ Dump}_c${ Data_Creation }_noaa_ops.h5*	VIIRS SDR granule files containing Moderate Resolution Band 08 in HDF5 format	*~12*
svm09	*SVM09_npp_d${Year}${Month}${Day}_t ${Granule_Start}_e${Granule_End}_b${ Dump}_c${ Data_Creation }_noaa_ops.h5*	VIIRS SDR granule files containing Moderate Resolution Band 09 in HDF5 format	*~12*
svm10	*SVM10_npp_d${Year}${Month}${Day}_t ${Granule_Start}_e${Granule_End}_b${ Dump}_c${ Data_Creation }_noaa_ops.h5*	VIIRS SDR granule files containing Moderate Resolution Band 10 in HDF5 format	*~12*
svm11	*SVM11_npp_d${Year}${Month}${Day}_t ${Granule_Start}_e${Granule_End}_b${ Dump}_c${ Data_Creation }_noaa_ops.h5*	VIIRS SDR granule files containing Moderate Resolution Band 11 in HDF5 format	*~12*
svm12	*SVM12_npp_d${Year}${Month}${Day}_t ${Granule_Start}_e${Granule_End}_b${ Dump}_c${ Data_Creation }_noaa_ops.h5*	VIIRS SDR granule files containing Moderate Resolution Band 12 in HDF5 format	*~12*
svm13	*SVM13_npp_d${Year}${Month}${Day}_t ${Granule_Start}_e${Granule_End}_b${ Dump}_c${ Data_Creation }_noaa_ops.h5*	VIIRS SDR granule files containing Moderate Resolution Band 13 in HDF5 format	*~22*
svm14	*SVM14_npp_d${Year}${Month}${Day}_t ${Granule_Start}_e${Granule_End}_b${ Dump}_c${ Data_Creation }_noaa_ops.h5*	VIIRS SDR granule files containing Moderate Resolution Band 14 in HDF5 format	*~12*
svm15	*SVM15_npp_d${Year}${Month}${Day}_t ${Granule_Start}_e${Granule_End}_b${ Dump}_c${ Data_Creation }_noaa_ops.h5*	VIIRS SDR granule files containing Moderate Resolution Band 15 in HDF5 format	*~12*
svm16	*SVM16_npp_d${Year}${Month}${Day}_t ${Granule_Start}_e${Granule_End}_b${*	VIIRS SDR granule files containing Moderate	*~12*

	Dump}_c${ Data_Creation }_noaa_ops.h5	Resolution Band 16 in HDF5 format	
gitco	*GITCO_npp_d${Year}${Month}${Day}_t${Granule_Start}_e${Granule_End}_b${Dump}_c${ Data_Creation }_noaa_ops.h5*	VIIRS SDR granule files containing Moderate Bands SDR Terrain Corrected Geolocation in HDF5 format	~131
svi01	*SVI01_npp_d${Year}${Month}${Day}_t${Granule_Start}_e${Granule_End}_b${Dump}_c${ Data_Creation }_noaa_ops.h5*	VIIRS SDR granule files containing Moderate Resolution Band 01 in HDF5 format	~24
svi02	*SVI02_npp_d${Year}${Month}${Day}_t${Granule_Start}_e${Granule_End}_b${Dump}_c${ Data_Creation }_noaa_ops.h5*	VIIRS SDR granule files containing Moderate Resolution Band 02 in HDF5 format	~25
svi03	*SVI03_npp_d${Year}${Month}${Day}_t${Granule_Start}_e${Granule_End}_b${Dump}_c${ Data_Creation }_noaa_ops.h5*	VIIRS SDR granule files containing Moderate Resolution Band 03 in HDF5 format	*~26*
svi04	*SVI04_npp_d${Year}${Month}${Day}_t${Granule_Start}_e${Granule_End}_b${Dump}_c${ Data_Creation }_noaa_ops.h5*	VIIRS SDR granule files containing Moderate Resolution Band 04 in HDF5 format	*~25*
svi05	*SVI05_npp_d${Year}${Month}${Day}_t${Granule_Start}_e${Granule_End}_b${Dump}_c${ Data_Creation }_noaa_ops.h5*	VIIRS SDR granule files containing Moderate Resolution Band 05 in HDF5 format	*~24*

Where:
${Year} = The 4-digit year
${Month} = The 2-digit month
${Day} = The 2-digit day of month
${HH} = The 2-digit hour
${MM} = The 2-digit minute
${Granule_Start} = The 7-digit granule start time HHMMSSS (hour, minute,

second, and tenth of a second)
${Granule_End} = The 7-digit granule end time HHMMSSS (hour, minute, second, and tenth of a second)
${Dump} = The 5-digit dump number
${Data_Creation} = The 14-digit product file creation time YYYYMMDDHHMMSS (4-digit year, 2-digit month, day, hour, minute, second)

7.2.2. Ancillary Data Files Required by AIT-framework

The ancillary data files listed in this section are required by the JRR-Product-Generation units when they call the Framework program. These ancillary data files are defined in the system's PCF file through the File-Handle-Name (left hand side of the equal sign in the PCF file) and can be located anywhere. The driver script and the sub-script Setup_links_for_static_ancillary.pl will make symbolic links of these files to each of the directories described below – the $working is the working directory created by NDE PGM. By doing the links, the template Framework-CFG, template and common Framework-PCF (see section 7.2.3) does not need to be updated in the case that the location of these ancillary data files changes. The reason is because the relative rather than full path are used in the AIT-framework processing control files when specifying the ancillary data file(s). The directories of ${ALGORITHM_ANCILLARY_DIR} and ${FRAMEWORK_ANCILLARY_DIR} described in the following sections are also defined in the system's PCF, should be always defined as:

ALGORITHM_ANCILLARY_DIR=algorithm_ancillary
FRAMEWORK_ANCILLARY_DIR=framework_ancillary

All of these ancillary files are in NetCDF4 format, except the CRTM coefficient files, which are in binary format

7.2.2.1. Land Mask

It is the global 1km land mask data file. It is required by the JRR Product Generators.
File Handle Name: LAND_MASK_NASA_1KM
File Name: lw_geo_2001001_v03m.nc
Size: 890 MB

Links to: $working/${FRAMEWORK_ANCILLARY_DIR}/masks
Framework-PCF: Land_Mask.pcf

7.2.2.2. Coast Mask

It is the global 1km land/water mask data file. It is required by unit JRR-Product-Generation.

File Handle Name: COAST_MASK_NASA_1KM
File Name: coast_mask_1km.nc
Size: 890 MB
Links to: $working/${FRAMEWORK_ANCILLARY_DIR}/masks
Framework-PCF: Coast_Mask.pcf

7.2.2.3. NWP Snow Mask

The NWP Snow Mask ancillary algorithm generates the Snow Mask from the following ancillary products: GFS NWP Data (section 7.2.2.14), Land Mask (section 7.2.2.1), and OISST Daily Data (section 7.2.2.15).

7.2.2.4. Calculated Desert Mask

The Calculated Desert Mask uses two ancillary products to generate the desert mask: Land Mask (section 7.2.2.1), and Surface Type (section 7.2.2.11). A value of 0 means no desert, 1 is wooden grass, closed shrubs, open shrubs, grasses, or cropland, and 2 is bare surface.

7.2.2.5. Climatic LST

The Climatic Land Surface Temperature (LST) product is monthly-averaged mean surface temperatures over the globe at 2.5 degree resolution. The data is from the International Satellite Cloud Climatology Project (ISCCP). Data is interpolated between two consecutive months to arrive at an average of the date of the satellite data. The temperature is in degrees K, and the twelve input netcdf datasets (Table 7-6) are in the following format: climatic_lst_month_XX.nc, where XX is the two digit number of the month (01 to 12).

Table 7-6: Climate LST Data

File Handle Name	File Name	Description	Size (KB)
climatic_lst_month_01	climatic_lst_month_01.nc	Climate LST data for January	48
climatic_lst_month_02	climatic_lst_month_02.nc	Climate LST data for February	48
climatic_lst_month_03	climatic_lst_month_03.nc	Climate LST data for March	48
climatic_lst_month_04	climatic_lst_month_04.nc	Climate LST data for April	48
climatic_lst_month_05	climatic_lst_month_05.nc	Climate LST data for May	48
climatic_lst_month_06	climatic_lst_month_06.nc	Climate LST data for June	48
climatic_lst_month_07	climatic_lst_month_07.nc	Climate LST data for July	48
climatic_lst_month_08	climatic_lst_month_08.nc	Climate LST data for August	48
climatic_lst_month_09	climatic_lst_month_09.nc	Climate LST data for September	48
climatic_lst_month_10	climatic_lst_month_10.nc	Climate LST data for October	48
climatic_lst_month_11	climatic_lst_month_11.nc	Climate LST data for November	48
climatic_lst_month_12	climatic_lst_month_12.nc	Climate LST data for December	48

7.2.2.6. Surface Elevation Mask

It is the digital surface elevation data file at 1km resolution. It is required by the JRR Product Generators.

File Handle Name: SFC_ELEV_GLOBE_1KM
File Name: GLOBE_1km_digelev.nc
Size: 1843.2 MB
Links to: $working/${FRAMEWORK_ANCILLARY_DIR}/sfc_elevation
Framework-PCF: Surface_Elevation.pcf

7.2.2.7. Desert Mask

The desert mask uses the NASA 1km land mask and 1km surface type ancillary algorithms to calculate the desert mask. A value of '0' denotes no desert, '1' refers to wooden grass, closed shrubs, open shrubs, grasses, or croplands, and '2' is desert. There are no external files associated specifically with this algorithm.

7.2.2.8. Pseudo Emissivity

The pseudo emissivity uses the Plank function to calculate the channel 7 emissivity. There are no external files associated with this algorithm.

7.2.2.9. Surface Albedo

The surface albedo provides a global estimate of the cloud-clear white sky reflectance from 2004 MODIS data. The albedo is an averaged value over a seventeen day period. There are three wavelengths, 0.659 um, 1.64 um, and 2.13 um.

Filenames:
AlbMap.WS.c004.v2.0.2004.DDD.0.659_x4.nc
AlbMap.WS.c004.v2.0.2004.DDD.1.64_x4.nc
AlbMap.WS.c004.v2.0.2004.DDD.2.13_x4.nc

Where DDD is the Julian day of the year, which ranges from 001 to 353 in increments of 17. There are a total of 66 files. Each file is 28 MB in size.

7.2.2.10. Surface Emissivity Seebore

These are the surface emissivity data files at 5km resolution. They are climatologically monthly data. They are required by jRR Product generator units (Table 7-7).

Table 7-7: Surface Emissivity Data

File Handle Name	File Name	Description	Size (MB)
F1_SEEBOR_EMISS	global_emiss_intABI_2005001.nc	SEEBOR data for January	693

F2_SEEBOR_EMISS	global_emiss_intABI_2005032.nc	SEEBOR data for February	693
F3_SEEBOR_EMISS	global_emiss_intABI_2005060.nc	SEEBOR data for March	693
F4_SEEBOR_EMISS	global_emiss_intABI_2005091.nc	SEEBOR data for April	693
F5_SEEBOR_EMISS	global_emiss_intABI_2005121.nc	Seebore data for May	693
F6_SEEBOR_EMISS	global_emiss_intABI_2005152.nc	Seebore data for June	693
F7_SEEBOR_EMISS	global_emiss_intABI_2005182.nc	Seebore data for July	693
F8_SEEBOR_EMISS	global_emiss_intABI_2005213.nc	Seebore data for August	693
F9_SEEBOR_EMISS	global_emiss_intABI_2005244.nc	Seebore data for September	693
F10_SEEBOR_EMISS	global_emiss_intABI_2005274.nc	Seebore data for October	693
F11_SEEBOR_EMISS	global_emiss_intABI_2005305.nc	Seebore data for November	693
F12_SEEBOR_EMISS	global_emiss_intABI_2005335.nc	Seebore data for December	693

Links to: $working/${FRAMEWORK_ANCILLARY_DIR}/seebor_emiss
Framework-PCF: Surface_Emissivity.pcf

7.2.2.11. Surface Type Mask

It is the surface type mask based on AVHRR at 1km resolution. It is required by the JRR Product Generator units.

File Handle Name: SFC_TYPE_AVHRR_1KM
File Name: gl-latlong-1km-landcover.nc
Size: 890 MB
Links to: $working/${FRAMEWORK_ANCILLARY_DIR}/masks
Framework-PCF: Surface_Type.pcf

7.2.2.12. CRTM Coefficient

These are the CRTM coefficient files for VIIRS data. They are required by the JRR Product Generator units, see Table 7-8.

Table 7-8: CRTM Coefficient Data

File Handle Name	File Name	Description	Size (KB)
CRTM_AerosolCoeff	AerosolCoeff.bin	Aerosol Coeff data for CRTM	5766260
CRTM_CloudCoeff	CloudCoeff.bin	Cloud Coeff data for CRTM	1654180
CRTM_EmisCoeff	EmisCoeff.bin	Emissivity Coeff data for CRTM	1888256
CRTM_VIIRS_M_SpcCoeff	viirs-m_npp.SpcCoeff.bin	Space Coeff data for NPP VIIRS-M	472
CRTM_VIIRS_M_TauCoeff	viirs-m_npp.TauCoeff.bin	Tau Coeff data for NPP VIIRS-M	105704

Links to:
$working/${ALGORITHM_ANCILLARY_DIR}/crtm_2.0.2/CRTM_Coefficients/Big_Endian
Framework-PCF: CRTM.pcf

7.2.2.13. Ancillary Data for VIIRS SDR Reader

It is the ancillary data for VIIRS SDR Reader. It is required by the JRR Product Generator units.

File Handle Name: NPP_VIIRS_SDR_ANCIL
File Name: npp_viirs_ancil.Ibands.nc

Size: 2240 KB
File Name: npp_viirs_ancil.Mbands.nc
Size: 2258 KB
Links to: $working/${FRAMEWORK_ANCILLARY_DIR}/ancil_file
Framework-PCF: NPP_VIIRS_SDR_Data.pcf

7.2.2.14. GFS GRIB2 Forecast Files

These are forecast files in 0.5 degree resolution GRIB2 generated by NCEP and pulled to the ESPC/DDS. These files are required by the units of JRR-Product-Generation when running the Framework.

File Handle Name: gfs_file
File Name: *gfs.t${Hour}z.pgrb2f${Forecast}.YYYYMMDD*
Size: 51~52 MB
Links to: $working/${FRAMEWORK_ANCILLARY_DIR}/
gfs_grib2_0.5deg/YYYY/MM/DD/ *gfs.t${Hour}z.pgrbf${Forecast}*
Framework-PCF: NWP_Data_grib2_0.5deg.pcf
where:
${Hour} = the time for which the forecast is run (00Z, 06Z, 12Z, and 18Z)
${Forecast} = the forecast projection time (in hours = 00, 03, 06, 09, and 12, ..)
YYYY = 4-digit year
MM = 2-digit month
DD = 2-digit the day of month

7.2.2.15. OISST Daily Data

It is the Reynolds OISST daily analysis on 0.25 degree resolution, generated by NCDC and pulled to DDS. These files are required by the units of JRR Product Generation when running the Framework.

File Handle Name: oisst_file
File Name: avhrr-only-v2.YYYYMMDD_preliminary.nc
Size: 8.0 MB
Links to: $working/${FRAMEWORK_ANCILLARY_DIR}/oisst_daily/ avhrr-only-v2.YYYYMMDD.nc

Framework-PCF: OISST_Daily_QtrDeg.pcf
Where,
YYYYMMDD – 4 digit year plus 2 digit month plus 2 digit day

7.2.3. AIT-framework CFG and PCF Files

An AIT-framework CFG file is required to run the AIT-framework program (as an argument) and in the CFG file, a number of AIT-framework PCF files are specified. The AIT-framework program is called in the JRR-Product-Generation units. In the rest of this section, all the CFG and PCF refer to the AIT-framework level CFG and PCF files.

There are two types of these files. One is taken as a template (including CFG and some PCF file), from which a particular CFG or PCF can be generated corresponding to a specific input data. Another can be taken as a common PCF files that can be used directly by the AIT-framework CFG file – no specific to a special input, all of the input share the same PCF.

Note that all directory variables in the following sub-section are defined in the system's processing control files JRR_PRODUCT_CLOUD_MASK.pl.PCF, JRR_PRODUCT_CLOUD_CLOUDS.pl.PCF, JRR_PRODUCT_AEROSOL_VOLASH.pl.PCF, JRR_PRODUCT_AEROSOL_AODADP.pl.PCF, JRR_PRODUCT_CRYOSPHERE_SNOW.pl.PCF, JRR_PRODUCT_CRYOSPHERE_ICE.pl.PCF (see section 4.1). The $working is the working directory created by NDE PGM

7.2.3.1. AIT-framework CFG and PCF Template Files

The system uses a number of template files. These are all static. They are never modified by the scripts and programs that use them. Scripts will only read-in these files to create a new one corresponding to the system configure and/or the input data. The script generated CFG and PCF files will be described in section 7.2.3.1.

There are six CFG template files. They are NPP_VIIRS_CLOUD_MASK_Template.cfg, NPP_VIIRS_CLOUD_CLOUDS_Template.cfg, NPP_VIIRS_AEROSOL_AODADP_Template.cfg, NPP_VIIRS_AEROSOL_VOLASH_Template.cfg, NPP_VIIRS_CRYOSPHERE_ICE_Template.cfg, and NPP_VIIRS_CRYOSPHERE_SNOW_Template.cfg.
All cfg files are used by JRR-Product-Generation units. There is also one PCF template file. In each unit, based on the input data, a specific CFG and a number of PCF files will be generated from the template CFG and PCF files. The specific CFG file will then be used to run the AIT-framework program to generate products.

The AIT-framework CFG template files located at ${CFG_TEMPL_DIR}
The AIT-framework PCF template files located at ${PCF_TEMPL_DIR}

The AIT-framework level CFG and PCF template files for the system are shown in Table 7-9:

Table 7-9: AIT-framework CFG and PCF Template Files

Template File Name	Description	Size (KB)	Unit Used
NPP_VIIRS_SDR_Template.pcf	A template PCF file to generate AIT-framework SDR PCF file	1.2	JRR-Product-Generation
NPP_VIIRS_CLOUD_MASK_Template.cfg	A template AIT-framework configuration file to generate the Bayes Cloud mask	3.0	JRR-Product-Generation
NPP_VIIRS_CLOUD_CLOUDS_Template.cfg	A template AIT-framework configuration file to generate the clouds products	3.2	JRR-Product-Generation
NPP_VIIRS_AEROSOL_VOLASH_Template.cfg	A template AIT-framework configuration file to generate the volcanic ash products	2.9	JRR-Product-Generation
NPP_VIIRS_AEROSOL_AODADP_Template.cfg	A template AIT-framework configuration file to generate the aerosol products	2.8	JRR-Product-Generation

NPP_VIIRS_CRYOSPHERE_ICE_Template.cfg	A template AIT-framework configuration file to generate the ice products	2.5	JRR-Product-Generation
NPP_VIIRS_CRYOSPHERE_SNOW_Template.cfg	A template AIT-framework configuration file to generate the snow products	2.7	JRR-Product-Generation

7.2.3.2. AIT-framework Common PCF Files

There are AIT-framework PCF files that never need to be updated as long as the system configuration does not change. They are named as Common PCF. The Common PCF files are static text files. They are generated when the system is setup. They are never modified by the scripts and programs that use them. “Common” means that to each input data, they are the same and never need to be modified. Table 7-10 listed the Common PCF files for the AIT-framework in units JRR-Product-Generation. The common PCF files are located at ${PCF_OVERWRITES}

Table 7-10: AIT-framework Common PCF Files

File Name	Description	Size (KB)	Unit Used
Land_Mask.pcf	AIT-framework PCF file to process Land Mask data	0.6	JRR-Product-Generation
OISST_Daily_QtrDeg.pcf	AIT-framework PCF file to process OISST daily 0.25 degree data	1.7	JRR-Product-Generation
NWP_Data_grib2_0.5deg.pcf	AIT-framework PCF file to process the GFS 0.5 degree GRIB2 data	1.3	JRR-Product-Generation

Surface_Elevation.pcf	AIT-framework PCF file to process surface elevation data	0.3	JRR-Product-Generation
Surface_Emissivity.pcf	AIT-framework PCF file to process surface emissivity data	0.3	JRR-Product-Generation
CRTM.pcf	AIT-framework PCF file to process CRTM	0.4	JRR-Product-Generation
Coast_Mask.pcf	AIT-framework PCF file to process coast mask data	0.25	JRR-Product-Generation
Desert_Mask.pcf	AIT-framework PCF file to process desert mask	0.2	JRR-Product-Generation
Snow_Mask.pcf	AIT-framework PCF file to process snow mask	0.2	JRR-Product-Generation
Pseudo_Emissivity.pcf	AIT-framework PCF file to process Pseudo emissivity	0.2	JRR-Product-Generation
Surface_Type.pcf	AIT-framework PCF file to process surface type	0.3	JRR-Product-Generation
NPP_VIIRS_Cld_Phase.pcf	AIT-framework PCF file to generate the VIIRS cloud phase product	1.4	JRR-Product-Generation
NPP_VIIRS_Cld_Height.pcf	AIT-framework PCF file to generate the VIIRS cloud Height product	1.4	JRR-Product-Generation
NPP_VIIRS_AERADP.pcf	AIT-framework PCF file to generate the Aerosol Optical Depth	1.8	JRR-Product-Generation
AWG_Cloud_Micro_Day.pcf	AIT-framework PCF file to generate the the DCOMP products	1	JRR-Product-Generation
AWG_Ice_Age_VIIRS_BAYES.pcf	AIT-framework PCF file to generate the Ice Age	0.6	JRR-Product-Generation
AWG_Ice_Conc_VIIRS_BAYES.pcf	AIT-framework PCF file to generate the Ice Concentration	0.5	JRR-Product-Generation
Climatic_LST.pcf	AIT-framework PCF file to generate the climatic LST	0.2	JRR-Product-Generation
NPP_BAYES_Cloud_Mask_NW.pcf	AIT-framework PCF file to generate the Bayes cloud mask	1.5	JRR-Product-Generation
NPP_BAYES_Cloud_Mask.pcf	AIT-framework PCF file to generate the Bayes cloud mask	1.5	JRR-Product-Generation
NPP_VIIRS_Cld_Height_NW.pcf	AIT-framework PCF file to generate the VIIRS cloud Height product	1.4	JRR-Product-Generation
NPP_VIIRS_Cld_NCOMP.pcf	AIT-framework PCF file to generate the NCOMP products	0.6	JRR-Product-Generation
NPP_VIIRS_SNOW_C	AIT-framework PCF file to generate	5.1	JRR-Product-

OVER.pcf	the snow cover		Generation
NPP_VOLCANIC_ASH.pcf	AIT-framework PCF file to generate the volcanic ash	0.9	JRR-Product-Generation
Surface_Albedo.pcf	AIT-framework PCF file to generate the surface albedo	0.2	JRR-Product-Generation
Volcano_Mask.pcf	AIT-framework PCF file to generate the volcano mask	0.3	JRR-Product-Generation

7.3. Intermediate Files

This section describes the data sets that are internal to the JPSS RR system.

7.3.1. Driver Script Created Intermediate Files

Table 7-11 shows the intermediate files produced by the product generator script. For the intermediate files from each unit, they will be described in the following sub-sections.

Table 7-11: Intermediate Files Produced by Driver Scripts

File Name	Description
JRR_PREPROCESS_SDR.pl.PCF	The SDR-Data-Preparation unit process control file, created by the driver script `JRR_PREPROCESS_SDR.pl`
JRR_PRODUCT_CLOUD_MASK.pl.PCF	The Cloud Mask unit process control file, created by the driver script JRR_PRODUCT_CLOUD_MASK.pl
JRR_PRODUCT_CLOUD_CLOUDS.pl.PCF	The Cloud Products unit process control file, created by the driver script JRR_PRODUCT_CLOUD_CLOUDS.pl
JRR_PRODUCT_AEROSOL_AODADP.pl.PCF	The Aerosol unit process control file, created by the driver script JRR_PRODUCT_AEROSOL_AODADP.pl
JRR_PRODUCT_AEROSOL_VOLASH.pl.PCF	The Volcanic Ash unit process control file, created by the driver script JRR_PRODUCT_AEROSOL_VOLASH.pl
JRR_PRODUCT_CRYOSPHERE_SNOW.pl.PCF	The Snow unit process control file, created by the driver script JRR_PRODUCT_CRYOSPHERE_SNOW.pl
JRR_PRODUCT_CRYOSPHERE_ICE.pl.PCF	The Ice unit process control file, created by the driver script JRR_PRODUCT_CRYOSPHERE_ICE.pl

7.3.2. AIT-framework CFG and PCF Files

These are the dynamic AIT-framework CFG and PCF files. They are generated by the scripts based on the specific input data files. The details about the files are listed in Table 7-12 for the ones from the JRR-Product-Generation unit.

In which,
${YYYY} = The 4-digit year
${JJJ} = The 3-digit day of year
${HH} = The 2-digit of hour
${MM} = The 2-digit of minute
${SS} = The 2-digit of second
${Granule_Start} = The 12-digit of granule start time (4-digit year, 2-digit month, day, hour, minute)

Files listed in Table 7-12 will be located at:
$working/configs

Table 7-12: AIT-framework CFG and PCF Files for JPSS RR Generation

File Name	**Description**	**Size (KB)**
d*${YYYY}${MMDD}_t${???????}_e${???????}_$b{?????}_cm.*cfg	An AIT-framework level configuration file to produce VIIRS Cloud Mask. For example: d20150402_t1054237_e1055479_b17764_cm.cfg	3.0
d*${YYYY}${MMDD}_t${???????}_e${???????}_$b{?????}_clouds.cfg*	An AIT-framework level configuration file to produce VIIRS Clouds products. For example: d20150402_t1054237_e1055479_b17764_clouds.cfg	3.3
d*${YYYY}${MMDD}_t${???????}_e${???????}_$b{?????}_aodadp.cfg*	An AIT-framework level configuration file to produce VIIRS aerosol products. For example: d20150402_t1054237_e1055479_b17764_aodadp.cfg	3.0
d*${YYYY}${MMDD}_t${???????}_e${???????}_$b{?????}_volash.cfg*	An AIT-framework level configuration file to produce VIIRS volcanic ash products. For example: d20150402_t1054237_e1055479_b17764_volash.cfg	2.8
d*${YYYY}${MMDD}_t${???????}_e${???????}_$b{?????}_snow.cfg*	An AIT-framework level configuration file to produce VIIRS snow products. For example: d20150511_t1225439_e1227081_b18318_snow.cfg	2.7

d*${YYYY}${MMDD}_t${???????}_e${???????}_${?????}_ice.cfg*	An AIT-framework level configuration file to produce VIIRS ice products. For example:	2.7
npp_viirs_sdr_d*${YYYY}${MMDD}_t${???????}_e${???????}_$b{?????}_cm.*pcf	An AIT-framework level PCF file to generate Bayes cloud mask. For example: npp_viirs_sdr_d20150402_t1054237_e1055479_b17764_aodadp.pcf	1.2
npp_viirs_sdr_d*${YYYY}${MMDD}_t${???????}_e${???????}_$b{?????}_clouds.pcf*	An AIT-framework level PCF file to generate cloud products. For example: npp_viirs_sdr_d20150402_t1054237_e1055479_b17764_clouds.pcf	1.2
npp_viirs_sdr_d*${YYYY}${MMDD}_t${???????}_e${???????}_$b{?????}_aodadp.pcf*	An AIT-framework level PCF file to generate aerosol products. For example: npp_viirs_sdr_d20150402_t1054237_e1055479_b17764_cm.pcf	1.2
npp_viirs_sdr_d*${YYYY}${MMDD}_t${???????}_e${???????}_$b{?????}_volash.pcf*	An AIT-framework level PCF file to generate volcanic ash. For example: npp_viirs_sdr_d20150402_t1054237_e1055479_b17764_volash.pcf	1.2
npp_viirs_sdr_d*${YYYY}${MMDD}_t${???????}_e${???????}_$b{?????}_snow.pcf*	An AIT-framework level PCF file to generate snow products. For example: npp_viirs_sdr_d20150511_t1225439_e1227081_b18318_snow.pcf	1.2
npp_viirs_sdr_d*${YYYY}${MMDD}_t${???????}_e${???????}_$b{?????}_ice.pcf*	An AIT-framework level PCF file to generate ice products. For example: npp_viirs_sdr_d20150511_t2240157_e2241381_b18324_ice.pcf	1.2

Example of the CFG and PCF files in Table 7-12 is below -- The one with green color should be updated each time corresponding to the associated input data:

d20150402_t1054237_e1055479_b17764_cm.cfg

```
Satellite info
SATELLITE: NPP
INSTRUMENT: VIIRS

SEGMENT_MODE: SCANLINE

#
# Data dimensions
#
ROWS: 768
COLUMNS: 3200
CHANNELS: 16
```

```
#
# PCF Framework Settings
#

#
# Segment details
#
ROW_START: 1
ROW_END: 768
COLUMN_START: 1
COLUMN_END: 3200
SEGMENT_SIZE: 120

#---Total number of resolutions
RESOLUTIONS: 1
DEFAULT_KEY: VIIRS_750M
ADDITIONAL_KEYS:

#---Key: Total_Rows,Total_Columns,  Row_Start,Row_End,  Column_Start,Column_End,  Size_Factor,
Total_Channels,Total_Channels_Vis,Total_Channels_IR

VIIRS_750M :  768, 3200, 1,  768, 1, 3200, 1, 16, 12, 5

#Choose the Channels
AVAILABLE_CHANNELS, VIIRS_750M: 3, 5, 7, 9, 10, 11, 12, 14, 15, 16

#
# RUN_MODE - USER_DEFINED_PCF or ALLOW_DEFAULT_PCF
# USER_DEFINED forces user to specify each product in this file
# instead of relying on dependencies to determine if that product should run
# and using the default pcf for that product
#
RUN_MODE: ALLOW_DEFAULT_PCF

#
# DEPENDENCY_INFO_LOCATION - ALLOW_IN_MAIN or PCF_ONLY
# ALLOW_IN_MAIN allows specific dependencies in pcf files to be
# blank, but a type must be declared in the default.
# If they specific dependencies are missing the framework will search
# the main list to see if any of that type has been declared and
# add them to the list as a dependency
# PCF_ONLY forces user to specify all PCF dependency information
# in that PCFs
#
#
DEPENDENCY_INFO_LOCATION: ALLOW_IN_MAIN

#
# Output Directory
#
OUTPUT_DIRECTORY:

#
# Output Option
# NETCDF_SEGMENT - creates a netCDF file for each segment
# NETCDF_ALL - creates a single file for the full run
# Important to note, each product (team) has its own output file
```

```
# Specifying the ALL part will not create a single file with everything
# in it.
#
OUTPUT_FILE_OPTION: NETCDF_ALL
FILE_VERSION: NETCDF_V4

#
#COMPRESSION
#
#DEFLATE_LEVEL of 0: turn off  compression.
#DEFLATE_LEVEL of 2: deflation level recommended by ASSISTT.

DEFLATE_LEVEL: 2

#
# Specify here what you want to run
#
RUN:

#
# For satellite data, specify actual data file instead of pcf
#
SATELLITE_DATA:
VIIRS_SDR_MULTIRES =
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b17764/SVM03_npp_2015092_105458_d20150402_t1054237_e1055479_b17764.nc + npp_viirs_sdr_d20150402_t1054237_e1055479_b17764_cm.pcf

LAND_MASK:
LAND_MASK_NASA_1KM = PCF_Overwrites/Land_Mask.pcf

SST:
OISST_DAILY_QTRDEG = PCF_Overwrites/OISST_Daily_QtrDeg.pcf

NWP_DATA:
NWP_GFS = PCF_Overwrites/NWP_Data_grib2_0.5deg.pcf

SURFACE_ELEVATION:
SFC_ELEV_GLOBE_1KM = PCF_Overwrites/Surface_Elevation.pcf

SURFACE_EMISSIVITY:
SFC_EMISS_SEEBOR = PCF_Overwrites/Surface_Emissivity.pcf

RTM:
CRTM = PCF_Overwrites/CRTM.pcf

COAST_MASK:
COAST_MASK_NASA_1KM = PCF_Overwrites/Coast_Mask.pcf

DESERT_MASK:
DESERT_MASK_CALCLTED = PCF_Overwrites/Desert_Mask.pcf

SNOW_MASK:
SNOW_MASK_NWP =  PCF_Overwrites/Snow_Mask.pcf

PSEUDO_EMISSIVITY:
GOESR_ABI_CHN7_EMISS = PCF_Overwrites/Pseudo_Emissivity.pcf
```

SURFACE_TYPE:
SFC_TYPE_AVHRR_1KM = PCF_Overwrites/Surface_Type.pcf

SURFACE_ALBEDO:
SFC_ALBEDO = PCF_Overwrites/Surface_Albedo.pcf

CLOUD_MASK:
NPP_BAYES_CLOUD_MASK = PCF_Overwrites/NPP_BAYES_Cloud_Mask.pcf

npp_viirs_sdr_d20150402_t1054237_e1055479_b17764_cm.pcf

VIIRS_SDR_MULTIRES
ANCILLARY

#
Set flag for output
#
OUTPUT_TO_FILE: N

For buffering
WINDOW_SIZE: 3

#Required only for Multiple Resolution Type
WRITER_TYPE: MULTI_RES

OTHER:
Sat_Name: NPP
Instrument_Name: VIIRS
Sat_ID: 300

16 M-bands (DNB is not included)
Total_Channels: 16

Size of the VIIRS granule for base resolution
Total_Rows: 768
Total_Columns: 3200

#--- Reflectance bands (11 Vis +1 NIR)
ReflChannels: 12

#--- IR bands (5 IR bands)
IR_Channels: 5

#--- ABI channel mapping
ChnMapTotalChns: 16

#Ancillary file contains satellite and sensor info
AncilFilename: framework_ancillary/ancil_file/npp_viirs_ancil.Mbands.nc

MaxSatZenAngle: 120

#Data coverage
Coverage_Type:

Interpolation box size for indexs from 750m(M Bands) to 375m(I bands)

```
# Must be an odd number(3,5,7,9,11)
Box_Size: 3

#
# For Planck
#
PlanckMaxTemp: 340.0
PlanckMinTemp: 179.0
PlanckDeltaTemp: 1.0

#
# Need to figure out what version we have, if applicable.
#
# Version: 1.0


#
# File containing lats/lons/etc
#
StaticFile:
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b17764//GMTCO_npp_2015092_105458_d20150402_t1054237_e1055479_b17764.nc

END VIIRS_SDR_MULTIRES
```

d20150402_t1054237_e1055479_b17764_clouds.cfg (with DCOMP)

```
# Satellite info
SATELLITE: NPP
INSTRUMENT: VIIRS

SEGMENT_MODE: SCANLINE

#
# Data dimensions
#
ROWS: 768
COLUMNS: 3200
CHANNELS: 16

#
# PCF Framework Settings
#

#
# Segment details
#
ROW_START: 1
ROW_END: 768
COLUMN_START: 1
COLUMN_END: 3200
SEGMENT_SIZE: 120

#---Total number of resolutions
RESOLUTIONS: 1
DEFAULT_KEY: VIIRS_750M
```

```
ADDITIONAL_KEYS :

#---Key: Total_Rows,Total_Columns,  Row_Start,Row_End,  Column_Start,Column_End,  Size_Factor,
Total_Channels,Total_Channels_Vis,Total_Channels_IR

VIIRS_750M :  768, 3200, 1,  768, 1, 3200, 1, 16, 12, 5

#Choose the Channels
#day: 5, 11, 14, 15, 16
#night: 12, 14, 15, 16
AVAILABLE_CHANNELS, VIIRS_750M: 5, 11, 14, 15, 16

#
# RUN_MODE - USER_DEFINED_PCF or ALLOW_DEFAULT_PCF
# USER_DEFINED forces user to specify each product in this file
# instead of relying on dependencies to determine if that product should run
# and using the default pcf for that product
#
RUN_MODE: ALLOW_DEFAULT_PCF

#
# DEPENDENCY_INFO_LOCATION - ALLOW_IN_MAIN or PCF_ONLY
# ALLOW_IN_MAIN allows specific dependencies in pcf files to be
# blank, but a type must be declared in the default.
# If they specific dependencies are missing the framework will search
# the main list to see if any of that type has been declared and
# add them to the list as a dependency
# PCF_ONLY forces user to specify all PCF dependency information
# in that PCFs
#
#
DEPENDENCY_INFO_LOCATION: ALLOW_IN_MAIN


#
# Output Directory
#
OUTPUT_DIRECTORY:

#
# Output Option
# NETCDF_SEGMENT - creates a netCDF file for each segment
# NETCDF_ALL - creates a single file for the full run
# Important to note, each product (team) has its own output file
# Specifying the ALL part will not create a single file with everything
# in it.
#
OUTPUT_FILE_OPTION: NETCDF_ALL
FILE_VERSION: NETCDF_V4

#
#COMPRESSION
#
#DEFLATE_LEVEL of 0: turn off  compression.
#DEFLATE_LEVEL of 2: deflation level recommended by ASSISTT.

DEFLATE_LEVEL: 2
```

```
#
# Specify here what you want to run
#
RUN:

#
# For satellite data, specify actual data file instead of pcf
#
SATELLITE_DATA:
VIIRS_SDR_MULTIRES =
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150511_t0919230_e0920472_b18316/SVM01_npp_2015131_091958_d20150511_t0919230_e0920472_b18316.nc + npp_viirs_sdr_d20150511_t0919230_e0920472_b18316_clouds.pcf

LAND_MASK:
LAND_MASK_NASA_1KM = PCF_Overwrites/Land_Mask.pcf

SST:
OISST_DAILY_QTRDEG = PCF_Overwrites/OISST_Daily_QtrDeg.pcf

NWP_DATA:
NWP_GFS = PCF_Overwrites/NWP_Data_grib2_0.5deg.pcf

SURFACE_ELEVATION:
SFC_ELEV_GLOBE_1KM = PCF_Overwrites/Surface_Elevation.pcf

SURFACE_EMISSIVITY:
SFC_EMISS_SEEBOR = PCF_Overwrites/Surface_Emissivity.pcf

RTM:
CRTM = PCF_Overwrites/CRTM.pcf

COAST_MASK:
COAST_MASK_NASA_1KM = PCF_Overwrites/Coast_Mask.pcf

DESERT_MASK:
DESERT_MASK_CALCLTED = PCF_Overwrites/Desert_Mask.pcf

SNOW_MASK:
SNOW_MASK_NWP =  PCF_Overwrites/Snow_Mask.pcf

PSEUDO_EMISSIVITY:
GOESR_ABI_CHN7_EMISS = PCF_Overwrites/Pseudo_Emissivity.pcf

SURFACE_TYPE:
SFC_TYPE_AVHRR_1KM = PCF_Overwrites/Surface_Type.pcf

SURFACE_ALBEDO:
SFC_ALBEDO = PCF_Overwrites/Surface_Albedo.pcf

CLOUD_MASK:
NPP_BAYES_CLOUD_MASK = JRR-CloudMask_v1r1_npp_s201505110919230_e201505110920472_c201603231707530.nc + PCF_Overwrites/NPP_BAYES_Cloud_Mask_NW.pcf

CLOUD_PHASE:
NPP_VIIRS_CLD_PHASE = PCF_Overwrites/NPP_VIIRS_Cld_Phase.pcf
```

CLOUD_HEIGHT:
NPP_VIIRS_CLD_HEIGHT = PCF_Overwrites/NPP_VIIRS_Cld_Height.pcf

CLOUD_MICRO_DAY:
AWG_CLOUD_MICRO_DAY = PCF_Overwrites/AWG_Cloud_Micro_Day.pcf

npp_viirs_sdr_d20150402_t1054237_e1055479_b17764_clouds.pcf (with DCOMP)

```
VIIRS_SDR_MULTIRES
ANCILLARY

#
# Set flag for output
#
OUTPUT_TO_FILE: N

# For buffering
WINDOW_SIZE: 3

#Required only for Multiple Resolution Type
WRITER_TYPE: MULTI_RES

OTHER:
Sat_Name: NPP
Instrument_Name: VIIRS
Sat_ID: 300

# 16 M-bands ( DNB is not included)
Total_Channels: 16

# Size of the VIIRS granule for base resolution
Total_Rows: 768
Total_Columns: 3200

#--- Reflectance bands ( 11 Vis +1 NIR)
ReflChannels: 12

#--- IR bands ( 5 IR bands)
IR_Channels: 5

#--- ABI channel mapping
ChnMapTotalChns: 16

#Ancillary file contains satellite and sensor info
AncilFilename: framework_ancillary/ancil_file/npp_viirs_ancil.Mbands.nc

MaxSatZenAngle: 120

#Data coverage
Coverage_Type:

# Interpolation box size for indexs from 750m(M Bands) to 375m(I bands)
# Must be an odd number(3,5,7,9,11)
Box_Size: 3
```

```
#
# For Planck
#
PlanckMaxTemp: 340.0
PlanckMinTemp: 179.0
PlanckDeltaTemp: 1.0

#
# Need to figure out what version we have, if applicable.
#
# Version: 1.0

#
# File containing lats/lons/etc
#
StaticFile:
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b17764//GMTCO_npp_2015092_105458_d20150402_t1054237_e1055479_b17764.nc

END VIIRS_SDR_MULTIRES
```

d20150511_t0919230_e0920472_b18316_clouds.cfg (with NCOMP)

```
# Satellite info
SATELLITE: NPP
INSTRUMENT: VIIRS

SEGMENT_MODE: SCANLINE

#
# Data dimensions
#
ROWS: 768
COLUMNS: 3200
CHANNELS: 16

#
# PCF Framework Settings
#

#
# Segment details
#
ROW_START: 1
ROW_END: 768
COLUMN_START: 1
COLUMN_END: 3200
SEGMENT_SIZE: 120

#---Total number of resolutions
RESOLUTIONS: 1
DEFAULT_KEY: VIIRS_750M
ADDITIONAL_KEYS :

#---Key: Total_Rows,Total_Columns,  Row_Start,Row_End,  Column_Start,Column_End,  Size_Factor,
Total_Channels,Total_Channels_Vis,Total_Channels_IR

VIIRS_750M :  768, 3200, 1,  768, 1, 3200, 1, 16, 12, 5
```

```
#Choose the Channels
#day: 5, 11, 14, 15, 16
#night: 12, 14, 15, 16
AVAILABLE_CHANNELS, VIIRS_750M: 12, 14, 15, 16

#
# RUN_MODE - USER_DEFINED_PCF or ALLOW_DEFAULT_PCF
# USER_DEFINED forces user to specify each product in this file
# instead of relying on dependencies to determine if that product should run
# and using the default pcf for that product
#
RUN_MODE: ALLOW_DEFAULT_PCF

#
# DEPENDENCY_INFO_LOCATION - ALLOW_IN_MAIN or PCF_ONLY
# ALLOW_IN_MAIN allows specific dependencies in pcf files to be
# blank, but a type must be declared in the default.
# If they specific dependencies are missing the framework will search
# the main list to see if any of that type has been declared and
# add them to the list as a dependency
# PCF_ONLY forces user to specify all PCF dependency information
# in that PCFs
#
#
DEPENDENCY_INFO_LOCATION: ALLOW_IN_MAIN

#
# Output Directory
#
OUTPUT_DIRECTORY:

#
# Output Option
# NETCDF_SEGMENT - creates a netCDF file for each segment
# NETCDF_ALL - creates a single file for the full run
# Important to note, each product (team) has its own output file
# Specifying the ALL part will not create a single file with everything
# in it.
#
OUTPUT_FILE_OPTION: NETCDF_ALL
FILE_VERSION: NETCDF_V4

#
#COMPRESSION
#
#DEFLATE_LEVEL of 0: turn off  compression.
#DEFLATE_LEVEL of 2: deflation level recommended by ASSISTT.

DEFLATE_LEVEL: 2


#
# Specify here what you want to run
#
RUN:
```

```
#
# For satellite data, specify actual data file instead of pcf
#
SATELLITE_DATA:
VIIRS_SDR_MULTIRES =
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150511_t0919230_e0920472_b18316/SVM14_npp_2015131_091958_d20150511_t0919230_e0920472_b18316.nc + npp_viirs_sdr_d20150511_t0919230_e0920472_b18316_clouds.pcf

LAND_MASK:
LAND_MASK_NASA_1KM = PCF_Overwrites/Land_Mask.pcf

SST:
OISST_DAILY_QTRDEG = PCF_Overwrites/OISST_Daily_QtrDeg.pcf

NWP_DATA:
NWP_GFS = PCF_Overwrites/NWP_Data_grib2_0.5deg.pcf

SURFACE_ELEVATION:
SFC_ELEV_GLOBE_1KM = PCF_Overwrites/Surface_Elevation.pcf

SURFACE_EMISSIVITY:
SFC_EMISS_SEEBOR = PCF_Overwrites/Surface_Emissivity.pcf

RTM:
CRTM = PCF_Overwrites/CRTM.pcf

COAST_MASK:
COAST_MASK_NASA_1KM = PCF_Overwrites/Coast_Mask.pcf

DESERT_MASK:
DESERT_MASK_CALCLTED = PCF_Overwrites/Desert_Mask.pcf

SNOW_MASK:
SNOW_MASK_NWP =  PCF_Overwrites/Snow_Mask.pcf

PSEUDO_EMISSIVITY:
GOESR_ABI_CHN7_EMISS = PCF_Overwrites/Pseudo_Emissivity.pcf

SURFACE_TYPE:
SFC_TYPE_AVHRR_1KM = PCF_Overwrites/Surface_Type.pcf

SURFACE_ALBEDO:
SFC_ALBEDO = PCF_Overwrites/Surface_Albedo.pcf

CLOUD_MASK:
NPP_BAYES_CLOUD_MASK = JRR-CloudMask_v1r1_npp_s201505110919230_e201505110920472_c201606291511510.nc + PCF_Overwrites/NPP_BAYES_Cloud_Mask_NW.pcf

CLOUD_PHASE:
NPP_VIIRS_CLD_PHASE = PCF_Overwrites/NPP_VIIRS_Cld_Phase.pcf

CLOUD_HEIGHT:
NPP_VIIRS_CLD_HEIGHT = PCF_Overwrites/NPP_VIIRS_Cld_Height.pcf

CLOUD_MICRO_NIT:
NPP_VIIRS_CLD_NCOMP = PCF_Overwrites/NPP_VIIRS_Cld_NCOMP.pcf
```

npp_viirs_sdr_ d20150511_t0919230_e0920472_b18316_clouds.pcf (with NCOMP)

```
VIIRS_SDR_MULTIRES
ANCILLARY

#
# Set flag for output
#
OUTPUT_TO_FILE: N

# For buffering
WINDOW_SIZE: 3

#Required only for Multiple Resolution Type
WRITER_TYPE: MULTI_RES

OTHER:
Sat_Name: NPP
Instrument_Name: VIIRS
Sat_ID: 300

# 16 M-bands ( DNB is not included)
Total_Channels: 16

# Size of the VIIRS granule for base resolution
Total_Rows: 768
Total_Columns: 3200

#--- Reflectance bands ( 11 Vis +1 NIR)
ReflChannels: 12

#--- IR bands ( 5 IR bands)
IR_Channels: 5

#--- ABI channel mapping
ChnMapTotalChns: 16

#Ancillary file contains satellite and sensor info
AncilFilename: framework_ancillary/ancil_file/npp_viirs_ancil.Mbands.nc

MaxSatZenAngle: 120

#Data coverage
Coverage_Type:

# Interpolation box size for indexs from 750m(M Bands) to 375m(I bands)
# Must be an odd number(3,5,7,9,11)
Box_Size: 3

#
# For Planck
#
PlanckMaxTemp: 340.0
PlanckMinTemp: 179.0
PlanckDeltaTemp: 1.0
```

```
#
# Need to figure out what version we have, if applicable.
#
# Version: 1.0


#
# File containing lats/lons/etc
#
StaticFile:
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150511_t0919230_e0920472_b18316/GMTCO_npp_2015131_091958_d20150511_t0919230_e0920472_b18316.nc

END VIIRS_SDR_MULTIRES
```

d20150402_t1054237_e1055479_b17764_aodadp.cfg

```
# Satellite info
SATELLITE: NPP
INSTRUMENT: VIIRS

SEGMENT_MODE: SCANLINE

#
# Data dimensions
#
ROWS: 768
COLUMNS: 3200
CHANNELS: 16

#
# PCF Framework Settings
#

#
# Segment details
#
ROW_START: 1
ROW_END: 768
COLUMN_START: 1
COLUMN_END: 3200
SEGMENT_SIZE: 120

#---Total number of resolutions
RESOLUTIONS: 1
DEFAULT_KEY: VIIRS_750M
ADDITIONAL_KEYS :

#---Key: Total_Rows,Total_Columns,  Row_Start,Row_End,  Column_Start,Column_End,  Size_Factor,
Total_Channels,Total_Channels_Vis,Total_Channels_IR

VIIRS_750M :  768, 3200, 1,  768, 1, 3200, 1, 16, 12, 5

#
# RUN_MODE - USER_DEFINED_PCF or ALLOW_DEFAULT_PCF
# USER_DEFINED forces user to specify each product in this file
```

```
# instead of relying on dependencies to determine if that product should run
# and using the default pcf for that product
#
RUN_MODE: ALLOW_DEFAULT_PCF

#
# DEPENDENCY_INFO_LOCATION - ALLOW_IN_MAIN or PCF_ONLY
# ALLOW_IN_MAIN allows specific dependencies in pcf files to be
# blank, but a type must be declared in the default.
# If they specific dependencies are missing the framework will search
# the main list to see if any of that type has been declared and
# add them to the list as a dependency
# PCF_ONLY forces user to specify all PCF dependency information
# in that PCFs
#
#
DEPENDENCY_INFO_LOCATION: ALLOW_IN_MAIN


#
# Output Directory
#
OUTPUT_DIRECTORY:

#
# Output Option
# NETCDF_SEGMENT - creates a netCDF file for each segment
# NETCDF_ALL - creates a single file for the full run
# Important to note, each product (team) has its own output file
# Specifying the ALL part will not create a single file with everything
# in it.
#
OUTPUT_FILE_OPTION: NETCDF_ALL
FILE_VERSION: NETCDF_V4

#
#COMPRESSION
#
#DEFLATE_LEVEL of 0: turn off  compression.
#DEFLATE_LEVEL of 2: deflation level recommended by ASSISTT.

DEFLATE_LEVEL: 2

#
# Specify here what you want to run
#
RUN:

#
# For satellite data, specify actual data file instead of pcf
#
SATELLITE_DATA:
VIIRS_SDR_MULTIRES =
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b17764//SVM01_npp_2015092_10
5458_d20150402_t1054237_e1055479_b17764.nc +
npp_viirs_sdr_d20150402_t1054237_e1055479_b17764_aodadp.pcf
```

LAND_MASK:
LAND_MASK_NASA_1KM = PCF_Overwrites/Land_Mask.pcf

NWP_DATA:
NWP_GFS = PCF_Overwrites/NWP_Data_grib2_0.5deg.pcf

SURFACE_ELEVATION:
SFC_ELEV_GLOBE_1KM = PCF_Overwrites/Surface_Elevation.pcf

SURFACE_TYPE:
SFC_TYPE_AVHRR_1KM = PCF_Overwrites/Surface_Type.pcf

DESERT_MASK:
DESERT_MASK_CALCLTED = PCF_Overwrites/Desert_Mask.pcf

SST:
OISST_DAILY_QTRDEG = PCF_Overwrites/OISST_Daily_QtrDeg.pcf

SNOW_MASK:
SNOW_MASK_NWP = PCF_Overwrites/Snow_Mask.pcf

CLOUD_MASK:
NPP_BAYES_CLOUD_MASK = JRR-CloudMask_v1r1_npp_s201504021054237_e201504021055479_c201507171644170.nc + PCF_Overwrites/NPP_BAYES_Cloud_Mask_NW.pcf

CLOUD_HEIGHT:
NPP_VIIRS_CLD_HEIGHT = JRR-CloudHeight_v1r1_npp_s201504021054237_e201504021055479_c201507171652020.nc + PCF_Overwrites/NPP_VIIRS_Cld_Height_NW.pcf

VOLCANIC_ASH:
NPP_VOLCANIC_ASH = JRR-VolcanicAsh_v1r1_npp_s201504021054237_e201504021055479_c201606291436270.nc + PCF_Overwrites/NPP_VOLCANIC_ASH_NW.pcf

AEROSOL_AOD:
AWG_AER_AOD = PCF_Overwrites/AWG_AER_AOD.pcf

AEROSOL_ADP:
NPP_VIIRS_AERADP = PCF_Overwrites/NPP_VIIRS_AERADP.pcf

npp_viirs_sdr_d20150402_t1054237_e1055479_b17764_aodadp.pcf

VIIRS_SDR_MULTIRES
ANCILLARY

#
Set flag for output
#
OUTPUT_TO_FILE: N

For buffering
WINDOW_SIZE: 3

#Required only for Multiple Resolution Type
WRITER_TYPE: MULTI_RES

```
OTHER:
Sat_Name: NPP
Instrument_Name: VIIRS
Sat_ID: 300

# 16 M-bands ( DNB is not included)
Total_Channels: 16

# Size of the VIIRS granule for base resolution
Total_Rows: 768
Total_Columns: 3200

#--- Reflectance bands ( 11 Vis +1 NIR)
ReflChannels: 12

#--- IR bands ( 5 IR bands)
IR_Channels: 5

#--- ABI channel mapping
ChnMapTotalChns: 16

#Ancillary file contains satellite and sensor info
AncilFilename: framework_ancillary/ancil_file/npp_viirs_ancil.Mbands.nc

MaxSatZenAngle: 120

#Data coverage
Coverage_Type:

# Interpolation box size for indexs from 750m(M Bands) to 375m(I bands)
# Must be an odd number(3,5,7,9,11)
Box_Size: 3

#
# For Planck
#
PlanckMaxTemp: 340.0
PlanckMinTemp: 179.0
PlanckDeltaTemp: 1.0

#
# Need to figure out what version we have, if applicable.
#
# Version: 1.0


#
# File containing lats/lons/etc
#
StaticFile:
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b17764//GMTCO_npp_2015092_1
05458_d20150402_t1054237_e1055479_b17764.nc

END VIIRS_SDR_MULTIRES
```

d20150402_t1054237_e1055479_b17764_volash.cfg

```
# Satellite info
SATELLITE: NPP
INSTRUMENT: VIIRS

SEGMENT_MODE: SCANLINE

#
# Data dimensions
#
ROWS: 768
COLUMNS: 3200
CHANNELS: 16

#
# PCF Framework Settings
#

#
# Segment details
#
ROW_START: 1
ROW_END: 768
COLUMN_START: 1
COLUMN_END: 3200
SEGMENT_SIZE: 120

#---Total number of resolutions
RESOLUTIONS: 1
DEFAULT_KEY: VIIRS_750M
ADDITIONAL_KEYS :

#---Key: Total_Rows,Total_Columns,  Row_Start,Row_End,  Column_Start,Column_End,  Size_Factor,
Total_Channels,Total_Channels_Vis,Total_Channels_IR

VIIRS_750M :  768, 3200, 1,  768, 1, 3200, 1, 16, 12, 5
VIIRS_375M : 1536, 6400, 1, 1536, 1, 6400, 2, 5, 3, 2

#
# RUN_MODE - USER_DEFINED_PCF or ALLOW_DEFAULT_PCF
# USER_DEFINED forces user to specify each product in this file
# instead of relying on dependencies to determine if that product should run
# and using the default pcf for that product
#
RUN_MODE: ALLOW_DEFAULT_PCF

#
# DEPENDENCY_INFO_LOCATION - ALLOW_IN_MAIN or PCF_ONLY
# ALLOW_IN_MAIN allows specific dependencies in pcf files to be
# blank, but a type must be declared in the default.
# If they specific dependencies are missing the framework will search
# the main list to see if any of that type has been declared and
# add them to the list as a dependency
# PCF_ONLY forces user to specify all PCF dependency information
# in that PCFs
```

```
#
#
DEPENDENCY_INFO_LOCATION: ALLOW_IN_MAIN

#
# Output Directory
#
OUTPUT_DIRECTORY:

#
# Output Option
# NETCDF_SEGMENT - creates a netCDF file for each segment
# NETCDF_ALL - creates a single file for the full run
# Important to note, each product (team) has its own output file
# Specifying the ALL part will not create a single file with everything
# in it.
#
OUTPUT_FILE_OPTION: NETCDF_ALL
FILE_VERSION: NETCDF_V4

#
#COMPRESSION
#
#DEFLATE_LEVEL of 0: turn off  compression.
#DEFLATE_LEVEL of 2: deflation level recommended by ASSISTT.

DEFLATE_LEVEL: 2

#
# Specify here what you want to run
#
RUN:

#
# For satellite data, specify actual data file instead of pcf
#
SATELLITE_DATA:
VIIRS_SDR_MULTIRES =
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b17764/SVM01_npp_2015092_105458_d20150402_t1054237_e1055479_b17764.nc + npp_viirs_sdr_d20150402_t1054237_e1055479_b17764_volash.pcf

LAND_MASK:
LAND_MASK_NASA_1KM = PCF_Overwrites/Land_Mask.pcf

SST:
OISST_DAILY_QTRDEG = PCF_Overwrites/OISST_Daily_QtrDeg.pcf

NWP_DATA:
NWP_GFS = PCF_Overwrites/NWP_Data_grib2_0.5deg.pcf

SURFACE_ELEVATION:
SFC_ELEV_GLOBE_1KM = PCF_Overwrites/Surface_Elevation.pcf

SURFACE_EMISSIVITY:
SFC_EMISS_SEEBOR = PCF_Overwrites/Surface_Emissivity.pcf

RTM:
```

CRTM = PCF_Overwrites/CRTM.pcf

COAST_MASK:
COAST_MASK_NASA_1KM = PCF_Overwrites/Coast_Mask.pcf

SNOW_MASK:
SNOW_MASK_NWP = PCF_Overwrites/Snow_Mask.pcf

SURFACE_TYPE:
SFC_TYPE_AVHRR_1KM = PCF_Overwrites/Surface_Type.pcf

VOLCANO_MASK:
VOLCANO_SMITH_1KM = PCF_Overwrites/Volcano_Mask.pcf

VOLCANIC_ASH:
NPP_VOLCANIC_ASH = PCF_Overwrites/NPP_VOLCANIC_ASH.pcf

npp_viirs_sdr_d20150402_t1054237_e1055479_b17764_volash.pcf

VIIRS_SDR_MULTIRES
ANCILLARY

#
Set flag for output
#
OUTPUT_TO_FILE: N

For buffering
WINDOW_SIZE: 3

#Required only for Multiple Resolution Type
WRITER_TYPE: MULTI_RES

OTHER:
Sat_Name: NPP
Instrument_Name: VIIRS
Sat_ID: 300

16 M-bands (DNB is not included)
Total_Channels: 16

Size of the VIIRS granule for base resolution
Total_Rows: 768
Total_Columns: 3200

#--- Reflectance bands (11 Vis +1 NIR)
ReflChannels: 12

#--- IR bands (5 IR bands)
IR_Channels: 5

#--- ABI channel mapping
ChnMapTotalChns: 16

#Ancillary file contains satellite and sensor info
AncilFilename: framework_ancillary/ancil_file/npp_viirs_ancil.Mbands.nc

MaxSatZenAngle: 120

#Data coverage
Coverage_Type:

Interpolation box size for indexs from 750m(M Bands) to 375m(I bands)
Must be an odd number(3,5,7,9,11)
Box_Size: 3

#
For Planck
#
PlanckMaxTemp: 340.0
PlanckMinTemp: 179.0
PlanckDeltaTemp: 1.0

#
Need to figure out what version we have, if applicable.
#
Version: 1.0

#
File containing lats/lons/etc
#
StaticFile:
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150402_t1054237_e1055479_b17764//GMTCO_npp_2015092_105458_d20150402_t1054237_e1055479_b17764.nc

END VIIRS_SDR_MULTIRES

d20150511_t1225439_e1227081_b18318_snow.cfg

Satellite info
SATELLITE: NPP
INSTRUMENT: VIIRS

SEGMENT_MODE: SCANLINE

#
Data dimensions
#
ROWS: 768
COLUMNS: 3200
CHANNELS: 16

#
PCF Framework Settings
#

#
Segment details
#
ROW_START: 1
ROW_END: 768
COLUMN_START: 1
COLUMN_END: 3200

```
SEGMENT_SIZE: 120

#---Total number of resolutions
RESOLUTIONS: 2
DEFAULT_KEY: VIIRS_750M
ADDITIONAL_KEYS: VIIRS_375M

#---Key: Total_Rows,Total_Columns,  Row_Start,Row_End,  Column_Start,Column_End,  Size_Factor,
Total_Channels,Total_Channels_Vis,Total_Channels_IR

VIIRS_750M :  768, 3200, 1,  768, 1, 3200, 1, 16, 12, 5
VIIRS_375M : 1536, 6400, 1, 1536, 1, 6400, 2, 5, 3, 2

#
# RUN_MODE - USER_DEFINED_PCF or ALLOW_DEFAULT_PCF
# USER_DEFINED forces user to specify each product in this file
# instead of relying on dependencies to determine if that product should run
# and using the default pcf for that product
#
RUN_MODE: ALLOW_DEFAULT_PCF

#
# DEPENDENCY_INFO_LOCATION - ALLOW_IN_MAIN or PCF_ONLY
# ALLOW_IN_MAIN allows specific dependencies in pcf files to be
# blank, but a type must be declared in the default.
# If they specific dependencies are missing the framework will search
# the main list to see if any of that type has been declared and
# add them to the list as a dependency
# PCF_ONLY forces user to specify all PCF dependency information
# in that PCFs
#
#
DEPENDENCY_INFO_LOCATION: ALLOW_IN_MAIN

#
# Output Directory
#
OUTPUT_DIRECTORY:

#
# Output Option
# NETCDF_SEGMENT - creates a netCDF file for each segment
# NETCDF_ALL - creates a single file for the full run
# Important to note, each product (team) has its own output file
# Specifying the ALL part will not create a single file with everything
# in it.
#
OUTPUT_FILE_OPTION: NETCDF_ALL
FILE_VERSION: NETCDF_V4

#
#COMPRESSION
#
#DEFLATE_LEVEL of 0: turn off  compression.
#DEFLATE_LEVEL of 2: deflation level recommended by ASSISTT.

DEFLATE_LEVEL: 2
```

```
#
# Specify here what you want to run
#
RUN:

#
# For satellite data, specify actual data file instead of pcf
#
SATELLITE_DATA:
VIIRS_SDR_MULTIRES =
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150511_t1225439_e1227081_b18318/SVM01_npp_2015131_122
618_d20150511_t1225439_e1227081_b18318.nc + npp_viirs_sdr_d20150511_t1225439_e1227081_b18318_snow.pcf

LAND_MASK:
LAND_MASK_NASA_1KM = PCF_Overwrites/Land_Mask.pcf

NWP_DATA:
NWP_GFS = PCF_Overwrites/NWP_Data_grib2_0.5deg.pcf

SURFACE_ELEVATION:
SFC_ELEV_GLOBE_1KM = PCF_Overwrites/Surface_Elevation.pcf

LST_CLIMATIC:
CLIMATIC_LST_ISCCP = PCF_Overwrites/Climatic_LST.pcf

CLOUD_MASK:
NPP_BAYES_CLOUD_MASK = JRR-
CloudMask_v1r1_npp_s201505111225439_e201505111227081_c201603231732560.nc +
PCF_Overwrites/NPP_BAYES_Cloud_Mask_NW.pcf

CRYOS_SNOW_COVER:
NPP_VIIRS_SNOW_COVER = PCF_Overwrites/NPP_VIIRS_SNOW_COVER.pcf
```

npp_viirs_sdr_d20150511_t1225439_e1227081_b18318_snow.pcf

```
VIIRS_SDR_MULTIRES
ANCILLARY

#
# Set flag for output
#
OUTPUT_TO_FILE: N

# For buffering
WINDOW_SIZE: 3

#Required only for Multiple Resolution Type
WRITER_TYPE: MULTI_RES

OTHER:
Sat_Name: NPP
Instrument_Name: VIIRS
Sat_ID: 300
```

16 M-bands (DNB is not included)
Total_Channels: 16

Size of the VIIRS granule for base resolution
Total_Rows: 768
Total_Columns: 3200

#--- Reflectance bands (11 Vis +1 NIR)
ReflChannels: 12

#--- IR bands (5 IR bands)
IR_Channels: 5

#--- ABI channel mapping
ChnMapTotalChns: 16

#Ancillary file contains satellite and sensor info
AncilFilename: framework_ancillary/ancil_file/npp_viirs_ancil.Mbands.nc

MaxSatZenAngle: 120

#Data coverage
Coverage_Type:

Interpolation box size for indexs from 750m(M Bands) to 375m(I bands)
Must be an odd number(3,5,7,9,11)
Box_Size: 3

#
For Planck
#
PlanckMaxTemp: 340.0
PlanckMinTemp: 179.0
PlanckDeltaTemp: 1.0

#
Need to figure out what version we have, if applicable.
#
Version: 1.0

#
File containing lats/lons/etc
#
StaticFile:
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150511_t1225439_e1227081_b18318/GMTCO_npp_2015131_122618_d20150511_t1225439_e1227081_b18318.nc

END VIIRS_SDR_MULTIRES

d20150511_t2240157_e2241381_b18324_ice.cfg
Satellite info
SATELLITE: NPP
INSTRUMENT: VIIRS

SEGMENT_MODE: SCANLINE

```
#
# Data dimensions
#
ROWS: 768
COLUMNS: 3200
CHANNELS: 16

#
# PCF Framework Settings
#

#
# Segment details
#
ROW_START: 1
ROW_END: 768
COLUMN_START: 1
COLUMN_END: 3200
SEGMENT_SIZE: 120

#---Total number of resolutions
RESOLUTIONS: 1
DEFAULT_KEY: VIIRS_750M
ADDITIONAL_KEYS :

#---Key: Total_Rows,Total_Columns,  Row_Start,Row_End,  Column_Start,Column_End,  Size_Factor,
Total_Channels,Total_Channels_Vis,Total_Channels_IR

VIIRS_750M :  768, 3200, 1,  768, 1, 3200, 1, 16, 12, 5

#
# RUN_MODE - USER_DEFINED_PCF or ALLOW_DEFAULT_PCF
# USER_DEFINED forces user to specify each product in this file
# instead of relying on dependencies to determine if that product should run
# and using the default pcf for that product
#
RUN_MODE: ALLOW_DEFAULT_PCF

#
# DEPENDENCY_INFO_LOCATION - ALLOW_IN_MAIN or PCF_ONLY
# ALLOW_IN_MAIN allows specific dependencies in pcf files to be
# blank, but a type must be declared in the default.
# If they specific dependencies are missing the framework will search
# the main list to see if any of that type has been declared and
# add them to the list as a dependency
# PCF_ONLY forces user to specify all PCF dependency information
# in that PCFs
#
#
DEPENDENCY_INFO_LOCATION: ALLOW_IN_MAIN

#
# Output Directory
#
OUTPUT_DIRECTORY:
```

```
#
# Output Option
# NETCDF_SEGMENT - creates a netCDF file for each segment
# NETCDF_ALL - creates a single file for the full run
# Important to note, each product (team) has its own output file
# Specifying the ALL part will not create a single file with everything
# in it.
#
OUTPUT_FILE_OPTION: NETCDF_ALL
FILE_VERSION: NETCDF_V4

#
#COMPRESSION
#
#DEFLATE_LEVEL of 0: turn off  compression.
#DEFLATE_LEVEL of 2: deflation level recommended by ASSISTT.

DEFLATE_LEVEL: 2

#
# Specify here what you want to run
#
RUN:

#
# For satellite data, specify actual data file instead of pcf
#
SATELLITE_DATA:
VIIRS_SDR_MULTIRES =
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150511_t2240157_e2241381_b18324/SVM01_npp_2015131_224
050_d20150511_t2240157_e2241381_b18324.nc + npp_viirs_sdr_d20150511_t2240157_e2241381_b18324_ice.pcf

LAND_MASK:
LAND_MASK_NASA_1KM = PCF_Overwrites/Land_Mask.pcf

CLOUD_MASK:
NPP_BAYES_CLOUD_MASK = JRR-
CloudMask_v1r1_npp_s201505112240157_e201505112241381_c201603231752040.nc +
PCF_Overwrites/NPP_BAYES_Cloud_Mask_NW.pcf

CLOUD_HEIGHT:
NPP_VIIRS_CLD_HEIGHT = JRR-
CloudHeight_v1r1_npp_s201505112240157_e201505112241381_c201603231756300.nc +
PCF_Overwrites/NPP_VIIRS_Cld_Height_NW.pcf

CRYOS_ICE_CONC:
AWG_ICE_CONC = PCF_Overwrites/AWG_Ice_Conc_VIIRS_BAYES.pcf

CRYOS_ICE_AGE:
AWG_CRYOS_ICE_AGE = PCF_Overwrites/AWG_Ice_Age_VIIRS_BAYES.pcf
```

npp_viirs_sdr_d20150511_t2240157_e2241381_b18324_ice.pcf

```
VIIRS_SDR_MULTIRES
ANCILLARY
```

```
#
# Set flag for output
#
OUTPUT_TO_FILE: N

# For buffering
WINDOW_SIZE: 3

#Required only for Multiple Resolution Type
WRITER_TYPE: MULTI_RES

OTHER:
Sat_Name: NPP
Instrument_Name: VIIRS
Sat_ID: 300

# 16 M-bands ( DNB is not included)
Total_Channels: 16

# Size of the VIIRS granule for base resolution
Total_Rows: 768
Total_Columns: 3200

#--- Reflectance bands ( 11 Vis +1 NIR)
ReflChannels: 12

#--- IR bands ( 5 IR bands)
IR_Channels: 5

#--- ABI channel mapping
ChnMapTotalChns: 16

#Ancillary file contains satellite and sensor info
AncilFilename: framework_ancillary/ancil_file/npp_viirs_ancil.Mbands.nc

MaxSatZenAngle: 120

#Data coverage
Coverage_Type:

# Interpolation box size for indexs from 750m(M Bands) to 375m(I bands)
# Must be an odd number(3,5,7,9,11)
Box_Size: 3

#
# For Planck
#
PlanckMaxTemp: 340.0
PlanckMinTemp: 179.0
PlanckDeltaTemp: 1.0

#
# Need to figure out what version we have, if applicable.
#
# Version: 1.0
```

```
#
# File containing lats/lons/etc
#
StaticFile:
/data/data241/hxie/dlv/JRR_DAP_201606/working_d20150511_t2240157_e2241381_b18324/GMTCO_npp_2015131_224050_d20150511_t2240157_e2241381_b18324.nc

END VIIRS_SDR_MULTIRES
```

7.3.3. Other Text Files

Other intermediate text files are generated by scripts/programs in unit SDR-Data-Preparation. The intermediate text file from each unit will be located at the unit's working directory.

The two text files generated in SDR-Data-Preparation and updated for each granule:
time.txt -- This is the text file containing the scan time string of a granule (leap second is included) in format of *YYYY_MM_DD_HH_mm_SS.S*. It is updated for every granule. It is generated by get_sdr_scan_time.exe. The SDR-Data-Preparation unit uses this string in generating the NetCDF4 SDR file names. Its content looks like:
2015_04_02_10_54_58.781909

7.3.4. Intermediate Data Files

7.3.4.1. Gap-filled SDR in HDF5

Before the VIIRS satellite data is converted to the NetCDF4 format from HDF5, the SVM01 to SVM16 data needs to be processed with gap-filling. The gap-filled files are listed in Table 7-13. These files are in HDF5 format with the same size as the original ones.

Table 7-13: Gap-filled HDF5 SDR Data

File Name	Description	Size (MB)

SVM01_npp_d${Year}${Month}${Day}_t${Granule_Start}_e${Granule_End}_b${Dump}_c${Data_Creation }_noaa_ops.mended.h5	VIIRS SDR granule files containing Moderate Resolution Band 01 in HDF5, gap-filled.	~12
SVM02_npp_d${Year}${Month}${Day}_t${Granule_Start}_e${Granule_End}_b${Dump}_c${Data_Creation }_noaa_ops.mended.h5	VIIRS SDR granule files containing Moderate Resolution Band 02 in HDF5, gap-filled.	~12
SVM03_npp_d${Year}${Month}${Day}_t${Granule_Start}_e${Granule_End}_b${Dump}_c${Data_Creation }_noaa_ops.mended.h5	VIIRS SDR granule files containing Moderate Resolution Band 03 in HDF5, gap-filled.	*~17*
SVM04_npp_d${Year}${Month}${Day}_t${Granule_Start}_e${Granule_End}_b${Dump}_c${Data_Creation }_noaa_ops.mended.h5	VIIRS SDR granule files containing Moderate Resolution Band 04 in HDF5, gap-filled.	*~17*
SVM05_npp_d${Year}${Month}${Day}_t${Granule_Start}_e${Granule_End}_b${Dump}_c${Data_Creation }_noaa_ops.mended.h5	VIIRS SDR granule files containing Moderate Resolution Band 05 in HDF5, gap-filled.	*~17*
SVM06_npp_d${Year}${Month}${Day}_t${Granule_Start}_e${Granule_End}_b${Dump}_c${Data_Creation }_noaa_ops.mended.h5	VIIRS SDR granule files containing Moderate Resolution Band 06 in HDF5, gap-filled.	*~12*
SVM07_npp_d${Year}${Month}${Day}_t${Granule_Start}_e${Granule_End}_b${Dump}_c${Data_Creation }_noaa_ops.mended.h5	VIIRS SDR granule files containing Moderate Resolution Band 07 in HDF5, gap-filled.	*~17*
SVM08_npp_d${Year}${Month}${Day}_t${Granule_Start}_e${Granule_End}_b${Dump}_c${Data_Creation }_noaa_ops.mended.h5	VIIRS SDR granule files containing Moderate Resolution Band 08 in HDF5, gap-filled.	*~12*
SVM09_npp_d${Year}${Month}${Day}_t${Granule_Start}_e${Granule_End}_b${Dump}_c${Data_Creation }_noaa_ops.mended.h5	VIIRS SDR granule files containing Moderate Resolution Band 09 in HDF5, gap-filled.	*~12*
SVM10_npp_d${Year}${Month}${Day}_t${Granule_Start}_e${Granule_End}_b${Dump}_c${Data_Creation }_noaa_ops.mended.h5	VIIRS SDR granule files containing Moderate Resolution Band 10 in HDF5, gap-filled.	*~12*
SVM11_npp_d${Year}${Month}${Day}_t${Granule_Start}_e${Granule_End}_b${Dump}_c${Data_Creation }_noaa_ops.mended.h5	VIIRS SDR granule files containing Moderate Resolution Band 11 in HDF5, gap-filled.	*~12*
SVM12_npp_d${Year}${Month}${Day}_t${Granu	VIIRS SDR granule files	*~12*

le_Start}_e${Granule_End}_b${Dump}_c${ Data_Creation }_noaa_ops.mended.h5	containing Moderate Resolution Band 12 in HDF5, gap-filled.	
SVM13_npp_d${Year}${Month}${Day}_t${Granule_Start}_e${Granule_End}_b${Dump}_c${ Data_Creation }_noaa_ops.mended.h5	VIIRS SDR granule files containing Moderate Resolution Band 13 in HDF5, gap-filled.	*~22*
SVM14_npp_d${Year}${Month}${Day}_t${Granule_Start}_e${Granule_End}_b${Dump}_c${ Data_Creation }_noaa_ops.mended.h5	VIIRS SDR granule files containing Moderate Resolution Band 14 in HDF5, gap-filled.	*~12*
SVM15_npp_d${Year}${Month}${Day}_t${Granule_Start}_e${Granule_End}_b${Dump}_c${ Data_Creation }_noaa_ops.mended.h5	VIIRS SDR granule files containing Moderate Resolution Band 15 in HDF5, gap-filled.	*~12*
SVM16_npp_d${Year}${Month}${Day}_t${Granule_Start}_e${Granule_End}_b${Dump}_c${ Data_Creation }_noaa_ops.mended.h5	VIIRS SDR granule files containing Moderate Resolution Band 16 in HDF5, gap-filled.	*~12*
SVI01_npp_d${Year}${Month}${Day}_t${Granule_Start}_e${Granule_End}_b${Dump}_c${ Data_Creation }_noaa_ops.mended.h5	VIIRS SDR granule files containing Moderate Resolution Band 01 in HDF5, gap-filled.	~25
SVI02_npp_d${Year}${Month}${Day}_t${Granule_Start}_e${Granule_End}_b${Dump}_c${ Data_Creation }_noaa_ops.mended.h5	VIIRS SDR granule files containing Moderate Resolution Band 02 in HDF5, gap-filled.	~25
SVI03_npp_d${Year}${Month}${Day}_t${Granule_Start}_e${Granule_End}_b${Dump}_c${ Data_Creation }_noaa_ops.mended.h5	VIIRS SDR granule files containing Moderate Resolution Band 03 in HDF5, gap-filled.	*~25*
SVI04_npp_d${Year}${Month}${Day}_t${Granule_Start}_e${Granule_End}_b${Dump}_c${ Data_Creation }_noaa_ops.mended.h5	VIIRS SDR granule files containing Moderate Resolution Band 04 in HDF5, gap-filled.	*~25*
SVI05_npp_d${Year}${Month}${Day}_t${Granule_Start}_e${Granule_End}_b${Dump}_c${ Data_Creation }_noaa_ops.mended.h5	VIIRS SDR granule files containing Moderate Resolution Band 05 in HDF5, gap-filled.	*~25*

Where:
${Year} = The 4-digit year
${Month} = The 2-digit month

${Day} = The 2-digit day of month
${HH} = The 2-digit hour
${MM} = The 2-digit minute
${Granule_Start} = The 7-digit granule start time HHMMSSS (hour, minute, second, and tenth of a second)
${Granule_End} = The 7-digit granule end time HHMMSSS (hour, minute, second, and tenth of a second)
${Dump} = The 5-digit dump number
${Data_Creation} = The 14-digit product file creation time YYYYMMDDHHMMSS (4-digit year, 2-digit month, day, hour, minute, second)

Located at: $working/${HDF_DATA} or ${HDF_DATA}

7.3.4.2. NetCDF Format VIIRS SDR

The NetCDF format VIIRS SDR is input to the AIT-framework. They are also the output files from the unit SDR-Data-Preparation. These files are the same as the input files described in section 7.2, but gap-filled (to SVM01-16) and converted into the NetCDF4 format. These data files are listed in Table 7-14.

In which:

${ScanYearMonthDay} = The 8 digit of scan day (4 digit year + 2 digit month + 2 digit day) from the data file
${ScanTime} = The 6 digit of scan time (2 digit hour + 2 digit minute + 2 digit second) from the data file
${Year} = The 4-digit year
${Month} = The 2-digit month
${Day} = The 2-digit day of month
${HH} = The 2-digit hour
${MM} = The 2-digit minute
${Granule_Start} = The 7-digit granule start time HHMMSSS (hour, minute, second, and tenth of a second)

Located at: $working/${SDR_DATA} or ${SDR_DATA}

Table 7-14: NetCDF Format VIIRS SDR and Cloud Mask

File Name	Description	Size (MB)
GMTCO_npp_${ScanYearMonthDay}_$(ScanTime}_d${Year}${Month}${Day}_t${Granule_Start}.nc	VIIRS SDR granule files containing Moderate Bands SDR Terrain Corrected Geolocation in NetCDF4 format. The scan time information is attached to the file name	~78
SVM01_npp_${ScanYearMonthDay}_$(ScanTime}_d${Year}${Month}${Day}_t${Granule_Start}.nc	VIIRS SDR granule files containing Moderate Resolution Band 01 in NetCDF4 format. The scan time information is attached to the file name	~12
SVM02_npp_${ScanYearMonthDay}_$(ScanTime}_d${Year}${Month}${Day}_t${Granule_Start}.nc	VIIRS SDR granule files containing Moderate Resolution Band 02 in NetCDF4 format. The scan time information is attached to the file name	~12
SVM03_npp_${ScanYearMonthDay}_$(ScanTime}_d${Year}${Month}${Day}_t${Granule_Start}.nc	VIIRS SDR granule files containing Moderate Resolution Band 03 in NetCDF4 format. The scan time information is attached to the file name	~17
SVM04_npp_${ScanYearMonthDay}_$(ScanTime}_d${Year}${Month}${Day}_t${Granule_Start}.nc	VIIRS SDR granule files containing Moderate Resolution Band 04 in NetCDF4 format. The scan time information is attached to the file name	~17
SVM05_npp_${ScanYearMonthDay}_$(ScanTime}_d${Year}${Month}${Day}_t${Granule_Start}.nc	VIIRS SDR granule files containing Moderate Resolution Band 05 in NetCDF4 format. The scan time information is attached to the file name	~17
SVM06_npp_${ScanYearMonthDay}_$(ScanTime}_d${Year}${Month}${Day}_t${Granule_Start}.nc	VIIRS SDR granule files containing Moderate Resolution Band 06 in NetCDF4 format. The scan time information is attached to the file name	~12
SVM07_npp_${ScanYearMonthDay}_$(ScanTime}_d${Year}${Month}${Day}_t${Granule_Start}.nc	VIIRS SDR granule files containing Moderate Resolution Band 07 in NetCDF4 format. The scan time information is attached to the file name	~17
SVM08_npp_${ScanYearMonthDay}_$(ScanTime}_d${Year}${Month}${Day}_t${Granule_Start}.nc	VIIRS SDR granule files containing Moderate Resolution Band 08 in NetCDF4 format. The scan time information is attached to the file name	~12
SVM09_npp_${ScanYearMonthDay}_$(ScanTime}_d${Year}${Month}${Day}_t${Gran	VIIRS SDR granule files containing Moderate Resolution Band 09 in	~12

ule_Start}.nc	NetCDF4 format. The scan time information is attached to the file name	
SVM10_npp_${ScanYearMonthDay}_$(ScanTime}_d${Year}${Month}${Day}_t${Granule_Start}.nc	VIIRS SDR granule files containing Moderate Resolution Band 10 in NetCDF4 format. The scan time information is attached to the file name	~12
SVM11_npp_${ScanYearMonthDay}_$(ScanTime}_d${Year}${Month}${Day}_t${Granule_Start}.nc	VIIRS SDR granule files containing Moderate Resolution Band 11 in NetCDF4 format. The scan time information is attached to the file name	~12
SVM12_npp_${ScanYearMonthDay}_$(ScanTime}_d${Year}${Month}${Day}_t${Granule_Start}.nc	VIIRS SDR granule files containing Moderate Resolution Band 12 in NetCDF4 format . The scan time information is attached to the file name	~12
SVM13_npp_${ScanYearMonthDay}_$(ScanTime}_d${Year}${Month}${Day}_t${Granule_Start}.nc	VIIRS SDR granule files containing Moderate Resolution Band 13 in NetCDF4 format. The scan time information is attached to the file name	~22
SVM14_npp_${ScanYearMonthDay}_$(ScanTime}_d${Year}${Month}${Day}_t${Granule_Start}.nc	VIIRS SDR granule files containing Moderate Resolution Band 14 in NetCDF4 format. The scan time information is attached to the file name	~12
SVM15_npp_${ScanYearMonthDay}_$(ScanTime}_d${Year}${Month}${Day}_t${Granule_Start}.nc	VIIRS SDR granule files containing Moderate Resolution Band 15 in NetCDF4 format. The scan time information is attached to the file name	~12
SVM16_npp_${ScanYearMonthDay}_$(ScanTime}_d${Year}${Month}${Day}_t${Granule_Start}.nc	VIIRS SDR granule files containing Moderate Resolution Band 16 in NetCDF4 format. The scan time information is attached to the file name	~12
GITCO_npp_${ScanYearMonthDay}_$(ScanTime}_d${Year}${Month}${Day}_t${Granule_Start}.nc	VIIRS SDR granule files containing Moderate Bands SDR Terrain Corrected Geolocation in NetCDF4 format. The scan time information is attached to the file name	~113
SVI01_npp_${ScanYearMonthDay}_$(ScanTime}_d${Year}${Month}${Day}_t${Granule_Start}.nc	VIIRS SDR granule files containing Moderate Resolution Band 01 in NetCDF4 format. The scan time information is attached to the file name	~25
SVI02_npp_${ScanYearMonthDay}_$(ScanTime}_d${Year}${Month}${Day}_t${Granule_Start}.nc	VIIRS SDR granule files containing Moderate Resolution Band 02 in NetCDF4 format. The scan time information is attached to the file name	~25
SVI03_npp_${ScanYearMonthDay}_$(ScanTime}_d${Year}${Month}${Day}_t${Granu	VIIRS SDR granule files containing Moderate Resolution Band 03 in	~25

le_Start}.nc	NetCDF4 format. The scan time information is attached to the file name	
SVI04_npp_${ScanYearMonthDay}_$(ScanTime}_d${Year}${Month}${Day}_t${Granule_Start}.nc	VIIRS SDR granule files containing Moderate Resolution Band 04 in NetCDF4 format. The scan time information is attached to the file name	~25
SVI05_npp_${ScanYearMonthDay}_$(ScanTime}_d${Year}${Month}${Day}_t${Granule_Start}.nc	VIIRS SDR granule files containing Moderate Resolution Band 05 in NetCDF4 format. The scan time information is attached to the file name	~25

7.4. Output Data Files

The output data files are the NetCDF format JPSS RR Product files.

7.4.1. JPSS RR Product Data Files

The JPSS RR products are output from the sub-system Product-Generator with naming convention:

JRR-CloudMask_v1r1_npp_s${YYYYMMDDhhmmsss}_e${YYYYMMDDhhmmsss}_c${YYYYMMDDhhmmsss}.nc

JRR-CloudHeight_v1r1_npp_s${YYYYMMDDhhmmsss}_e${YYYYMMDDhhmmsss}_c${YYYYMMDDhhmmsss}.nc

JRR-CloudPhase_v1r1_npp_s${YYYYMMDDhhmmsss}_e${YYYYMMDDhhmmsss}_c${YYYYMMDDhhmmsss}.nc

JRR-CloudDCOMP_v1r1_npp_s${YYYYMMDDhhmmsss}_e${YYYYMMDDhhmmsss}_c${YYYYMMDDhhmmsss}.nc

JRR-

CloudNCOMP_v1r1_npp_s${YYYYMMDDhhmmsss}_e${YYYYMMDDhhmmsss}_c${YYYYMMDDhhmmsss}.nc

JRR-AOD_v1r1_npp_s${YYYYMMDDhhmmsss}_e${YYYYMMDDhhmmsss}_c${YYYYMMDDhhmmsss}.nc

JRR-ADP_v1r1_npp_s${YYYYMMDDhhmmsss}_e${YYYYMMDDhhmmsss}_c${YYYYMMDDhhmmsss}.nc

JRR-VolcanicAsh_v1r1_npp_s${YYYYMMDDhhmmsss}_e${YYYYMMDDhhmmsss}_c${YYYYMMDDhhmmsss}.nc

JRR-SnowCover_v1r1_npp_s${YYYYMMDDhhmmsss}_e${YYYYMMDDhhmmsss}_c${YYYYMMDDhhmmsss}.nc

JRR-IceAge_v1r1_npp_s${YYYYMMDDhhmmsss}_e${YYYYMMDDhhmmsss}_c${YYYYMMDDhhmmsss}.nc

JRR-IceConcentration_v1r1_npp_s${YYYYMMDDhhmmsss}_e${YYYYMMDDhhmmsss}_c${YYYYMMDDhhmmsss}.nc

In which:
v1r1 = Version 1, Release 1
npp = Satellite name is NPP
s = Indicates observation start date and time string, it is the first image pass-over the pole time (scan time of the pole-pass-over granule from the first remapped image).
e = Indicates observation end date and time string, it is the third image pass-over the pole time (scan time of the pole-pass-over granule from the third remapped image).
c = Indicates the file creation (in the production environment) date and time string

YYYYMMDDhhmmsss = Date and time string in UTC where,
YYYY = 4-digit year
MM = 2-digit month of year
DD = 2-digit day of month
hh = 2-digit hour
mm = 2-digit minute
sss = Decimal seconds to the 1/10 precision (e.g. "103" = 10.3 seconds)

Note that the *.nc product data files undergo level 2 compression.

Located at: $working/${JRR_OUTPUT}/results or ${JRR_OUTPUT}/results

Size: Around 10 ~ 100 MB

7.5. Archive Data Files

The JPSS RR products will not be archived to the CLASS / NCDC archives.

8. APPENDIX 2 – LOW-LEVEL ERROR MESSAGE

Section 6.1.3.1 outlined high-level error messages returned from the scripts. The origins of many of these errors, however, are from low-level sources. When this is the case, log files are added to these high-level messages containing the output of the low-level errors. These errors and their explanation are show in Table 6-14. Each error message lists the program that produced the error as $PROGRAM in Table 6-14.

The low-level error messages are described here. Most of what needs to be done in response to these errors depends on the type of response suggested by the associated high-level errors originating from the calling script.

Table 8-1 to Table 8-5 shows the messages from the execution of Framework. They are FATAL ERROR, WARNING, UNKNOWN, NOTICE, and MISC (the one that not belong to any first four's), respectively. However, for the VIIRS Polar Winds system, the below message can be ignored:

NOTICE: (read_emissivity) Unable to determine geocat channel map for 13

Table 8-1: JPSS RR Error Messages (Low-Level) – Framework FATAL ERROR

Fatal Message	**Explanation**	**Action**
Data conversion		
FATAL ERROR: (Data_Processing_Manager:: Load_Main_PCF) Unable to convert value for $VariableName	There is an error in converting value for variable $VariableName.	Contact AIT framework team.
Data Reading		

FATAL ERROR: ($RoutineName) Coeff SDS not found in Coeff file.	There is an error in finding Coeff SDS in Coeff file.	Check the Coeff SDS in Coeff file.
FATAL ERROR: ($RoutineName) $ErrorStatus	There is an error in calling NetCDF function.	Check the NetCDF file or contact AIT framework team.
FATAL ERROR: ($RoutineName) Error in Read_netCDF while attempting to read $VariableName	There is an error in reading $VariableName.	Check the $VariableName in NetCDF file.
FATAL ERROR: ($RoutineName) error reading histogram file, iostat = $Status	There is an error in reading histogram file.	Check the variables in histogram file.
FATAL ERROR: ($RoutineName) get_var $ErrorStatus	There is an error in calling nf90_get_var.	Check the variables in NetCDF file or contact AIT framework team.
FATAL ERROR: ($RoutineName) Invalid or missing Channel flag.	There is an error in opening the Channel flag file.	Check the channel flag in the file.
FATAL ERROR: ($RoutineName) NF90_CLOSE: $ErrorStatus	There is an error in calling NF90_CLOSE.	Contact AIT framework team.
FATAL ERROR: ($RoutineName) nf90_get_att1 $ErrorStatus	There is an error in calling nf90_get_att1.	Check the attributes of the variables in NetCDF file or contact AIT framework team.
FATAL ERROR: ($RoutineName) nf90_get_att2 $ErrorStatus	There is an error in calling nf90_get_att2.	Check the attributes of the variables in NetCDF file or contact AIT framework team.
FATAL ERROR:	There is an error in	Contact AIT framework

($RoutineName) NF90_GET_VAR: $ErrorStatus	calling NF90_GET_VAR.	team.
FATAL ERROR: ($RoutineName) NF90_GET_VAR: error	There is an error in getting data in routine Read_Land_Sfc_Net	Check the variables in NetCDF file.
FATAL ERROR: ($RoutineName) NF90_INQ_DIMENSION error.	There is an error in getting variable dimension ID: Sds_ID.	Check the variable in NetCDF file.
FATAL ERROR: ($RoutineName) nf90_inq_varid $ErrorStatus	There is an error in calling nf90_inq_varid.	Check the variables in NetCDF file or contact AIT framework team.
FATAL ERROR: ($RoutineName) NF90_INQ_VARID $ErrorStatus	There is an error in calling NF90_INQ_VARID.	Contact AIT framework team.
FATAL ERROR: ($RoutineName) NF90_INQ_VARID error.	There is an error in getting variable ID: Sds_ID.	Check the variable in NetCDF file.
FATAL ERROR: ($RoutineName) nf90_inquire_variable $ErrorStatus	There is an error in calling nf90_inquire_variable.	Check the variables in NetCDF file or contact AIT framework team.
FATAL ERROR: ($RoutineName) Problems encountered with reading $FileType	There is a problem in reading $FileType.	Check the file variables.
FATAL ERROR: ($RoutineName) Read failed for ch. 14	There is an error in reading data from Temporal_File.	Check the data in Temporal_File.
File management		

FATAL ERROR: ($RoutineName) Can’t open file: $Filename	There is a problem in opening the file $Filename.	Check the file location and name.
FATAL ERROR: ($RoutineName) error closing histogram file ($ReadOrWrite), iostat = $Status	There is an error in closing histogram file.	Contact AIT framework team.
FATAL ERROR: ($RoutineName) error opening $Type file (read), iostat = $Status	There is an error in opening $Type file.	Check the file location, file name and corresponding setup in PCF.
FATAL ERROR: ($RoutineName) $ErrorStatus: $Filename	There is an error in opening $Filename.	Check the file location, file name and corresponding setup in PCF.
FATAL ERROR: ($RoutineName) error writing histogram file, iostat = $Status	There is an error in writing histogram file.	Check the file location and file name.
FATAL ERROR: ($RoutineName) $FileType does not exist: $Filename	The file $Filename does not exist.	Check the location of file $Filename, check the file name and the corresponding setup in PCF.
FATAL ERROR: ($RoutineName) Incomplete channel flag file.	There is an error in processing the Channel flag file.	Check the channel flag file.
FATAL ERROR: ($RoutineName) nc_close failed to close the output file.	There is an error in opening file in routine $RoutineName.	Check the file location.
FATAL ERROR: ($RoutineName)	There is an error in opening Coeff file.	The the Coeff file location.

Open Coeff file.		
FATAL ERROR: ($RoutineName) Unable to close file $Filename	There is an error in closing the file $Filename.	Contact the AIT Framework team.
FATAL ERROR: ($RoutineName) Unable to open file.	There is an error in opening file in routine $RoutineName.	Check the file location.
FATAL ERROR: ($RoutineName) Unable to open file $Filename	There is an error in opening the file $Filename.	Check the location of file $Filename.
FATAL ERROR: ($RoutineName) Unable to open $FileType.	There is an error in opening $FileType	Check the file location, file name and corresponding setup in PCF.
FATAL ERROR:Wrong input file, $Filename	The input file is wrong.	Check the file location, file name and corresponding setup in PCF.
Function call failure		
FATAL ERROR: ($RoutineName) Error in Call of $CallRoutineName	There is an error in the caller routine $CallRoutineName.	Check the AIT framework team.
FATAL ERROR: ($RoutineName) Failed at call to $CallRoutine	There is an problem in calling $CallRoutine	Contact AIT framework team.
FATAL ERROR: ($RoutineName) Failed at first call to Read_AVN_File.	There is a problem in reading the AVN forecast data from the grib file.	Contact AIT framework team.
FATAL ERROR: ($RoutineName) Failed at second call to Read_AVN_File.	There is a problem in reading the AVN forecast data from the grib file.	Contact AIT framework team.

Incorrect value		
FATAL ERROR: (Data_Processing_Manager:: Process_Algorithms) Counter > Segment_Variables. Segment_Total.	There is an error in processing segments.	Contact AIT Framework team.
FATAL ERROR: (Data_Processing_Manager:: Determine_Time_Segments) Invalid End Time in Segment_Variables.	End Time is wrong.	Check the PCF file or contact AIT Framework team.
FATAL ERROR: (Data_Processing_Manager:: Determine_Time_Segments) Invalid Start Time in Segment_Variables.	Start Time is wrong.	Contact AIT Framework team.
FATAL ERROR: (Data_Processing_Manager:: Determine_Time_Segments) Segment_Variables. Time_Increment <= 0.	The Time_Increment of Segment_Variables should be greater than 0.	Check the PCF file or contact AIT Framework team.
FATAL ERROR: (Data_Processing_Manager:: Determine_Time_Segments) Segment_Variables. TotalTime <= 0.	The TotalTime of Segment_Variables should be greater than 0.	Check the PCF file or contact AIT Framework team.
FATAL ERROR:	The month input is	Contact AIT framework

($RoutineName) Invalid month in days	incorrect.	team.
FATAL ERROR: ($RoutineName) Must specify valid time interval to continue.	Time step should be one of the followings: 5 min, 7 min, 10 min, 13 min, 15 min, 15.5 min, 17 min, 30 min, 99 min.	Check the PCF setup.
FATAL ERROR: ($RoutineName) Nested tracking should be off when processing clear-sky wv winds	Nested tracking is on.	Check the PCF setup.
FATAL ERROR: ($RoutineName) Number of vector greater than MaxVecOutNum	The vector count is greater than MaxVecOutNum.	Contact AIT framework team.
FATAL ERROR: ($RoutineName) numElem or numLine not correct	numElem and numLine should be in the range between 0 and Box_Size.	Check PCF setup.
FATAL ERROR: ($RoutineName) Search box must be larger than target scene	Search box size should be greater than the target size.	Check PCF setup.
FATAL ERROR: ($RoutineName) $VariableName MUST BE BETWEEN $Number1 AND $Number2	Variable $VariableName is not in the correct range, and should be between $Number1 and $Number2.	Contact AIT framework team.
Memory allocation		
FATAL ERROR: (Data_Processing_M	There is an error in allocating resolution	Contact AIT framework team.

anager:: Load_Main_PCF) Unable to allocate resolution arrays in SegmentInfo Structure.	arrays.	
FATAL ERROR: (Data_Processing_Manager:: Set_Process_Generator) Unable to allocate Process_Generator.	There is an error in allocating Process_Generator.	Contact AIT Framework team.
FATAL ERROR: (Data_Processing_Manager:: Set_Process_Generator) Unable to create Process_Generator for $SatelliteName $InstrumentName.	There is an error in creating Process_Generator.	Contact AIT Framework team.
FATAL ERROR: ($RoutineName) Allocation error.	There is an error in allocating memory.	Contact AIT framework team.
FATAL ERROR: ($RoutineName) Allocation error for $DataName.	There is an error in allocating memory for $DataName.	Contact AIT framework team.
FATAL ERROR: ($RoutineName) Allocation error, stat = $Status	There is an error in allocating space.	Contact AIT framework team.
FATAL ERROR: ($RoutineName) Allocation failed ch14	There is an error in allocating space.	Check the memory or contact AIT framework team.
FATAL ERROR: ($RoutineName)	There is an error in allocating variable	Check the memory or contact AIT framework team.

Error allocating variable $VariableName	$VariableName.	
FATAL ERROR: ($RoutineName) Error allocating Data array	There are some problems in allocating memory for data in routine $RoutineName.	Check the inputs for Allocate_Rows and Allocate_Columns in PCF file or contact the AIT framework team.
FATAL ERROR: ($RoutineName) Failed to allocate spaces	There is an error in allocating space.	Contact AIT framework team.
FATAL ERROR: ($RoutineName) Failed to allocate space for variables	There is an error in allocating data.	Contact AIT framework team.
FATAL ERROR: ($RoutineName) Failed to allocate structure.	There is an error in allocating structure in routine $RoutineName.	Contact the AIT Framework team.
FATAL ERROR: ($RoutineName) Failed to allocate $VariableName.	There is an error in allocating $VariableName.	Check the memory or contact AIT framework team.
FATAL ERROR: ($RoutineName) Failed to set up $VariableName.	There is an error in setting up variable $VariableName.	Contact AIT framework team.
FATAL ERROR: ($RoutineName) memory allocation failed	There is an error in memory allocation.	Check the memory or contact AIT framework team.
FATAL ERROR: ($RoutineName) Not enough memory to allocate 2D buffer.	There is an error in allocating array in routine Read_Land_Sfc_Net	Check the memory or contact AIT framework team.
FATAL ERROR: ($RoutineName) Not	There is an error in allocating	Check the memory or contact AIT framework team.

enough memory to allocate $VariableName	$VariableName.	
FATAL ERROR: (Check_Status) Not enough memory to allocate variable $ArrayName.	There is an error in allocating array $ArrayName in routine Check_Status	Check the memory or contact AIT framework team.
WARNING: ($RoutineName) Unable to allocate $VariableName	There is an error in allocating $VariableName.	Check the memory or contact AIT framework team.
Memory deallocation:		
FATAL ERROR: (Data_Processing_Manager:: ~Data_Processing_Manager) Unable to delete Process_Generator.	There is an error in deleting Process_Generator.	Contact AIT framework team.
FATAL ERROR: ($RoutineName) Deallocation error	There is an error in deallocating data.	Contact AIT framework team.
FATAL ERROR: ($RoutineName) Dealocation error for $DataName.	There is an error in deallocating memory for $DataName.	Contact AIT framework team.
FATAL ERROR: ($RoutineName) De-allocation error, stat = $Status	There is an error in allocating space.	Contact AIT framework team.
FATAL ERROR: ($RoutineName) Error deallocating $VariableName	There is an error in deallocating data in routine $RoutineName	Contact AIT framework team.
FATAL ERROR:	There is an error in	Contact AIT framework

($RoutineName) Error deallocating variable $VariableName	deallocating variable.	team.
FATAL ERROR: ($RoutineName) Failed to deallocate data	There are an error in deallocating data.	Contact AIT framework team.
FATAL ERROR: ($RoutineName) Failed to deallocate space	There is an error in deallocating data.	Contact AIT framework team.
FATAL ERROR: ($RoutineName) Failed to deallocate space for variables	There is an error in deallocating space for variables.	Contact AIT framework team.
FATAL ERROR: ($RoutineName) Failed to deallocate structure.	There is an error in deallocating structure in routine $RoutineName.	Contact the AIT Framework team.
FATAL ERROR: ($RoutineName) Failed to deallocate $Variable.	There is an error in deallocating $Variable.	Contact the AIT framework team.
FATAL ERROR: ($RoutineName) Problem deallocating $VariableName	There is an error in deallocating $VariableName.	Contact AIT framework team.
PCF and Config file setup		
FATAL ERROR: (Data_Processing_Manager:: Determine_Segments) Invalid segment mode option.	Segment mode should be SCANLINE_SEGMENT_OPTION, TIME_SEGMENT_OPTION, SCANLINE_EOF_SEGM	Check the Config file.

	ENT_OPTION or TIME_EOF_SEGMENT_ OPTION.	
FATAL ERROR: (Data_Processing_Manager:: Initialize_system) $ProcessName either does not have a user defined pcf associated with it or unable to read user defined PCF with RUN_MODE set to USER_DEFINED_PCF.	The process $ProcessName may not have a user defined pcf, or unable to read user defined PCF with RUN_MODE set to USER_DEFIND_PCF.	Check the file location of user defined PCF.
FATAL ERROR: (Data_Processing_Manager::Load_Main_PCF) Invalid option for FILE_VERSION: $Value	$Value specified in FILE_VERSION should be NETCDF_V4 or NETCDF_V3	Check the config file.
FATAL ERROR: (Data_Processing_Manager::Load_Main_PCF) Invalid option for OUTPUT_FILE_OPTION: $Value	$Value specified in OUTPUT_FILE_OPTION should be NETCDF_SEGMENT or NETCDF_ALL	Check the config file.
FATAL ERROR: (Data_Processing_Manager:: Load_Main_PCF) Invalid option for SEGMENT_MODE: $Value.	$Value in SEGMENT_MODE should be SCANLINE, TIME SCANLINE_EOF OR TIME_EOF.	Check the config file.
FATAL ERROR: (Data_Processing_M	There is no ADDITIONAL_KEYS in	Add ADDITIONAL_KEYS into config file.

anager:: Load_Main_PCF) Unable to find ADDITIONAL_KEYS info.	config file.	
FATAL ERROR: ($RoutineName) Cannot find sensor, using sensor index = 1	There is a problem in finding the sensor index being 1.	Contact the AIT framework team.
FATAL ERROR: ($RoutineName) Image subsection is all in space or corrupted.	Water vapor channel is not available.	Check the water vapor channel setup in PCF.
FATAL ERROR: ($RoutineName) Invalid QC_Ret_Option	QC_Ret_Option should be 1, 2 or 3.	Check the PCF setup.
FATAL ERROR: ($RoutineName) Line Segment size is too small, can not be smaller than Box_Size.	The lower bound of the line segment size is Box_Size. The segment size specified by PCF is too small.	Check the PCF setup.
FATAL ERROR: ($RoutineName) Neither Water Vapor channel available	Water vapor channel is not available.	Check the water vapor channel setup in PCF.
FATAL ERROR: ($RoutineName) No satellite image pixels in the required domain.	The error message is issued because there are no satellite image pixels.	Check the latitude and longitude in geolocation file.

FATAL ERROR: ($RoutineName) Segment size must be a multiple of Total Rows	The warning message is issued because segment size is not a multiple of total rows.	Check the config file and the PCF.
FATAL ERROR: ($RoutineName) Unable to find the channel from CRTM	There is an error in finding the channel	Check the channel setup in PCF.
FATAL ERROR: ($RoutineName) Unknow SAT_ID	There are some problems in finding SAT_ID	Check your SAT_ID in PCF File.
Others		
FATAL ERROR: (Data_Processing_Manager:: Load_Main_PCF) Unable to read Default Key.	There is an error in reading default key.	Contact AIT framework team.
FATAL ERROR: (Data_Processing_Manager:: Load_Main_PCF) Unable to read Resolution Key.	There is an error in reading resolution key.	Contact AIT framework team.
FATAL ERROR: (Data_Processing_Manager:: Load_Main_PCF) Unable to read Resolution Key List.	There is an error in reading resolution key list.	Contact AIT framework team.
FATAL ERROR: ($RoutineName) Problem with wgrib Command	There is an error in calling wgrib command.	Check the wgrib command and script.

FATAL ERROR: ($RoutineName) FP SETUP Error for CTRL_Block	There is an error in FP setup.	Check the AIT framework team.

Table 8-2: JPSS RR Error Messages (Low-Level) – Framework WARNING

Warning Message	Explanation	Action
Data conversion		
WARNING: ($RoutineName) Error converting platform number.	There is an error in converting platform number.	Contact AIT framework team.
WARNING: ($RoutineName) Error converting $StringName to integer.	There is an error in converting $StringName to integer.	Contact AIT framework team.
WARNING: ($RoutineName) Unable to convert Timestep to string.	There is an error in converting Timestep to string.	Contact AIT framework team.
WARNING: ($RoutineName) Unable to convert value for $VariableName	There is an error in converting value.	Contact AIT framework team.
WARNING: ($RoutineName) Unable to convert $VariableName.	There is an error in converting $VariableName.	Contact AIT framework team.
Data reading and writing		

WARNING: ($RoutineName) Attribute reading error for attribute $AttributeName.	There is an error in getting attribute $AttributeName.	Check the attribute $AttributeName in NetCDF file.
WARNING: ($RoutineName) Dimensions in file $Filename for $VariableName do not match those specified by pcf.	The $VariableName in $ Filename do not match that specified by pcf.	Check the $VariableName in $Filename and your PCF to see if they match.
WARNING: (read_netcdf) Error inquiring about NetCDF file: nf90_inq_varid $VariableName	There is an error of getting the NetCDF variable ID of variable $VariableName by calling nf90_inq_varid.	Check the NetCDF file to see if the variable $VariableName is correct.
WARNING: ($RoutineName) Error determining valid data type for variable $VariableName	There is an error in determining data type.	Contact AIT Framework team.
WARNING: ($RoutineName) Error extract attribute: $AttributeName from $FileName.	There is an error in extracting attribute $AttributeName from file $FileName.	Check the attribute $AttributeName in file $FileName.
WARNING: ($RoutineName) Error in finding GMTCO string in SDR nav filename	There is an error in finding GMTCO string in SDR navigation filename.	Check the navigation filenames.
WARNING: ($RoutineName) Error in	There are some problems in reading NetCDF $VariableName	Check NetCDF file to see if the variable $VariableName is correct.

Read_NetCDF reading $VariableName	in routine $RoutineName	
WARNING: ($RoutineName) Error inquiring about NetCDF file: nf90_get_att	There is an error in getting the NetCDF attribute.	Check the NetCDF file.
WARNING: ($RoutineName) Error inquiring about NetCDF file: nf90_get_varid	There is an error in getting the NetCDF variable.	Check the NetCDF file.
WARNING: ($RoutineName) error inquiring on histogram file, iostat = $Status	There is an error in inquiring on histogram file.	Check the histogram file location, file name and corresponding setup in PCF.
WARNING: ($RoutineName) Error inquiring variable $Filename	There is an error in inquiring variable from file $Filename.	Check the variable in the file.
WARNING: ($RoutineName) Error in reading netCDF file: nf90_get_att	There is an error in reading the NetCDF attribute.	Check the NetCDF file.
WARNING: ($RoutineName) Error in Read_netCDF while attempting to read $VariableName	There is an error in reading $VariableName.	Check the $VariableName in NetCDF file.
WARNING: ($RoutineName) Error in Read_netCDF_Dimensions while	There is an error in reading dimensions.	Check the dimensions in NetCDF file.

attempting to read dimensions		
WARNING: ($RoutineName) Error in Read_NetCDF_Dimensions while attempting to read dimensions for $VariableName	There are some problems in reading NetCDF dimensions in routine $RoutineName.	Check the file to see if the NetCDF dimension is correct.
WARNING: (RoutineName) Error reading $DataType field: nf90_get_var.	There is an error in reading data values by calling nf90_get_var.	Check the NetCDF file to see if there is something wrong in data field.
WARNING: ($RoutineName) Error reading instrument Name.	There is an error in reading instrument name.	Check the variable Sensor in NetCDF file.
WARNING: ($RoutineName) Error reading $VariableName: nf90_get_att.	There is an error in reading $VariableName.	Check $VariableName in NetCDF file.
WARNING: ($RoutineName) Error reading variable $Filename	There is an error in reading variable from file $Filename.	Check the variable in the file.
WARNING: (RoutineName) Error retrieving information: nf90_inquire_dimension	There is an error of obtaining data dimensions by calling nf90_inquire_dimension.	Check the dimension in NetCDF file.
WARNING: ($RoutineName) Error retrieving information: nf90_inquire_variabl	There is an error of obtaining data type by calling nf90_inquire_variable.	Check the data type in NetCDF file.

e		
WARNING: ($RoutineName) Error with nf90_get_att: $VariableName $AttributeName	There is an error in getting attribute $AttributeName of variable $VariableName.	Check the attribute and variable in NetCDF file.
WARNNG: ($RoutineName) Invalid dimension sizes for $VariableName in $StructName.	There is an error in dimension size.	Check the variables dimensions in NetCDF file.
WARNING: ($RoutineName) NetCDF file $FileName has no global attribute: $AttributeName	There is an error in getting the attribute $AttributeName in file $FileName.	Check the attribute $AttributeName in NetCDF file.
WARNNG: ($RoutineName) nf90_get_var: $ErrorStatus	There is an error in calling nf90_get_var.	Check the variables in NetCDF file or contact AIT framework team.
WARNNG: ($RoutineName) nf90_inq_varid $ErrorStatus	There is an error in calling nf90_inq_varid.	Check the variables in NetCDF file or contact AIT framework team.
WARNNG: ($RoutineName) nf90_inquire_variabl e $ErrorStatus	There is an error in calling nf90_inquire_variable.	Check the variables in NetCDF file or contact AIT framework team.
WARNING: ($RoutineName) Platform name not recognized $SatName	Platform name $SatName is not one of the key words below: Terra, terra, TERRA, Aqua, aqua, AQUA.	Check the platform name in the PCF file.
WARNING: ($RoutineName)	There is an error in creating variable	Check the output NetCDF location and the variable

Unable to create variable: $VariableName	$VariableName while writing into output NetCDF file.	$VariableName.
WARNING: ($RoutineName) Unable to define $AttributeName attribute for variable: $VariableName.	There is an error in defining coordinates attribute for variable $VariableName in routine $RoutineName.	Check variable $VariableName in NetCDF File.
WARNING: ($RoutineName) Unable to define attribute $AttributeName in NetCDF file: $FileName	There is an error in defining attribute $AttributeName in file $FileName.	Check the variable in output NetCDF file.
WARNING: ($RoutineName) Unable to define variable $VariableName in NetCDF file: $FileName	There is an error in defining variable $VariableName in file $FileName.	Check the variable in output NetCDF file.
WARNNG: ($RoutineName) Unable read ice density.	There is an error in reading ice density.	Check ice density in NetCDF file.
WARNNG: ($RoutineName) Unable read normal error variance.	There is an error in reading normal error variance.	Check normal error variance in NetCDF file.
WARNNG: ($RoutineName) Unable read sst.	There is an error in reading sst.	Check sst in NetCDF file.
WARNING: ($RoutineName) Unable to read dim $VariableName from	There is an error in reading the dimension for variable $VariableName in	Check the variable $VariableName in file.

file.	routine $RoutineName.	
WARNING: ($RoutineName) Unable to read dim $VariableName ID from file.	There is an error in reading the dimension ID for variable $VariableName in routine $RoutineName.	Check the variable $VariableName in file.
WARNING: ($RoutineName)Unable to read $FileType	There is an error in reading variables in $FileType.	Check the variables in $FileType.
WARNING: ($RoutineName) Unable to read frequencies.	There is an error in reading frequencies.	Check the frequencies in frequency file.
WARNING: ($RoutineName) Unable to read $VariableName	There are problems in reading $VariableName in $RoutineName.	Check variable $VariableName in file $RoutineName.
WARNING: ($RoutineName) Unable to read var $VariableName from file.	There is an error in reading the variable $VariableName in routine $RoutineName.	Check the variable $VariableName in file.
WARNING: ($RoutineName) Unable to write var $VariableName to file.	There is an error in writing variable $VariableName in routine $RoutineName.	Check variable $VariableName in NetCDF File.
File management		
WARNING: ($RoutineName) Date verification failed. Can't open file: $Filename	There is a problem in opening the file $Filename.	Check the file location and name.
WARNING: (read_netcdf) Error closing input file:	There is an error in closing NetCDF file by calling nf90_close.	Contact AIT Framework team.

nf90_close		
WARNING: ($RoutineName) Error closing NetCDF file: nf90_close $FileName.	There is an error in closing file $FileName.	Contact AIT framework team.
WARNING: ($RoutineName) Error opening file: $Filename	There is an error in opening file $Filename.	Check the file location.
WARNING: ($RoutineName) Error opening NetCDF file: nf90_open $FileName	There is an error opening NetCDF file $FileName.	Check the NetCDF file $FileName to see if the file location is correct.
WARNING: ($RoutineName) New filename : $Filename does not match previously set filename $PrevFilename for PCF_ID: $PCF_ID.	Filename do not match for $PCF_ID.	Check the PCF and config files.
WARNING: ($RoutineName) No Bias file is found.	There is an error in finding bias file.	Check the bias file location, file name and corresponding setup in PCF.
WARNNG: ($RoutineName) OISST file does not exist: $Filename	The file $Filename does not exist.	Check the file location and name.
WARNING: ($RoutineName) Unable to create NetCDF File: $Filename.	There is an error in creating NetCDF file $Filename.	Check the file location.
WARNNG:	There is an error in	Contact AIT framework

($RoutineName) Unable to close OISST file: $Filename.	closing OISST file.	team.
WARNNG: ($RoutineName) Unable to find OISST file for satellite date.	There is an error in finding OISST file.	Check the file location, file name and the corresponding setup in PCF.
WARNING: ($RoutineName) Unable to get Filename for $AlgorithmName $TimeStepType $PCF_ID.	There is an error in getting Filename.	Check the temporal setup in PCF.
WARNING: ($RoutineName) Unable to get filename for $PCF_ID	There is an error in getting the filename.	Check the file location, filename and corresponding setup in PCF.
WARNING: ($RoutineName) Unable to open Default PCF for $ProcessName.	There is an error in opening Default PCF for process $ProcessName.	Check the default PCF location or contact AIT framework team.
WARNING: ($RoutineName) Unable to open $Filename.	There are problems in open file $Filename in $RoutineName.	Check if file $ Filename exists, check the file location, file name and corresponding setup in PCF.
WARNING: ($RoutineName) Unable to open frequency file: $FileName.	There is an error in opening file $FileName in routine $RoutineName.	Check the file location.
WARNING: ($RoutineName) Unable to open Main	There is an error in opening the Main PCF.	Check the file location.

PCF: $FileName		
WARNNG: ($RoutineName) Unable to open OISST file: $Filename	There is an error in opening OISST file.	Check the file location, file name and corresponding setup in PCF and config file.
Function call failure		
WARNING: ($RoutineName) Bad value returned by latlondistance!	There is an error in calling latlondistance.	Contact AIT framework team.
WARNING: ($RoutineName) Error IN CRTM Forward Model	There is an error in calling CRTM forward model.	Contact AIT framework team.
WARNING: ($RoutineName) Error IN CRTM Forward Model call for SST	There is an error in calling CRTM forward model for SST.	Contact AIT framework team.
WARNING: ($RoutineName) Error in $FunctionName	There is an error in processing routine $ProcessedRoutineName.	Contact AIT framework team.
WARNING: ($RoutineName) Error initializing CRTM	There is an error in CRTM initialization.	Contact AIT framework team.
WARNING: ($RoutineName) Problem in $FunctionName	There is an error in calling $FunctionName.	Contact AIT framework team.
WARNING: ($RoutineName) Problem with $FunctionName	There is an error in calling $FunctionName.	Contact AIT framework team.

WARNING: ($RoutineName) Unable to init Temporal. Ancillary Data for index $Time_String.	There is an error in initiating Temporal	Check the file location, filename and corresponding setup in PCF.
WARNING: ($RoutineName) Unable to nullify Temporal. Ancilillary Data for index: $Time_String	There is an error in nullifying Temporal.	Contact AIT framework team.
Incorrect value		
WARNING: ($RoutineName) Bad precip value!	The value of precip is not correct	Check PCF setup.
WARNING: ($RoutineName) BT_coefficients has more than 3 sets of offset and scale	The dimension size of BT_coefficients is greater than 3.	Check the dimension size in NetCDF file.
WARNING: ($RoutineName) DependencyCounter -1 < 0	DependencyCounter should be greater than or equal to 1.	Contact AIT framework team.
WARNING: ($RoutineName) Dimensions in file $Filename do not match those specified by pcf.	The dimensions in file $Filename do not match those specified in PCF.	Check the dimensions Total_Columns, Total_Rows and Total_Channels in $Filename and your PCF to see if they match.
WARNING: ($RoutineName) Invalid channel number.	There is an error in locating channel number	Check the channel set up in PCF file.
WARNNG: ($RoutineName)	The data version is incorrect.	Check PCF and data version in NetCDF file.

Invalid data version: $Version.		
WARNING: ($RoutineName) Invalid row range specified.	There is an error in row range calculation.	Contact AIT framework team.
WARNING: (Get_Nav_Data) Invalid satellite sub-point geolocation.	The satellite sub-point longitude is outside the range (-180, 180) or the latitude is outside the range (-90, 90).	Check the geolocation file.
WARNNG: ($RoutineName) MaxWeeks < 0	The value of variable MaxWeeks is invalid.	Check the PCF setup.
WARNING: ($RoutineName) Number of RH values < 0.0 : $BadCount	Number of RH values should be less than 0.	Check the PCF file and corresponding data file.
WARNING: ($RoutineName) NWP date does not match requested date	There is a problem in matching the file date.	Check the dates for all the related files.
WARNING: ($RoutineName) Same forecasts at 15 and 180 min: bad file(s)	Warning message is issued because the rain rate forecasts are identical at 15 and 180 mins.	Check PCF setup.
WARNING: ($RoutineName) SizeFactor <= 0	The SizeFactory should be greater than 0.	Check the size factor in PCF.
WARNING: ($RoutineName) Time not found in Time_Index_Map: $Time_String	Time is not found in the index map.	Check the time setup in PCF.
WARNING:	Total_Segments should	Check the config file and

($RoutineName) Total_Segments <= 0 for $Name.	be greater than 0.	PCF, or contact AIT framework team.
WARNING: ($RoutineName) Total_Times <= 0.	Total_Times should be greater than 0.	Contact AIT framework team.
WARNING: ($RoutineName) Unknow Dependency_Name	Dependency_Name should be AWG_FOG.	Contact AIT framework team.
WARNING: ($RoutineName) Unrecognized Temporal PCF_ID: $PCF_ID.	The algorithm or product which has TEMPORAL_DATA_FIXED property in your PCF is not recognized.	Check the Temporal setup in PCF.
WARNING: ($RoutineName) Used size < SegmentNumber for type $Type for $Name.	The used size is at least as big as segment number.	Contact AIT framework team.
WARNING: ($RoutineName) $VariableName has more than one set of offset and scale	The dimension size of $VariableName is greater than 2.	Check the dimension size in NetCDF file.
Memory allocation		
WARNING: ($RoutineName) Either Input/Output/Temporal pointer is Null.	The Framework_Input_Structure_Ptr is not allocated.	Contact AIT Framework team.
WARNING: ($RoutineName) Error allocating Data array	There are some problems in allocating memory for data in routine $RoutineName.	Check the inputs for Allocate_Rows and Allocate_Columns in PCF file or contact the AIT framework team.

WARNING: ($RoutineName) Error allocating $Name	There is a problem in allocating $Name.	Contact the AIT framework team.
WARNING: ($RoutineName) Error allocating Offset and Scale Factor arrays for $BandName	There is an error in allocating arrays for $BandName.	Contact the AIT Framework team.
WARNING: ($RoutineName) Error allocating the array called $ArrayName: Read_netcdf	There is an error in allocating data array for $ArrayName.	Contact AIT Framework team.
WARNING: ($RoutineName) Error allocating ReflOffset arrays for 1km vis	There is an error in deallocating arrays	Contact the AIT Framework team.
WARNING: ($RoutineName) Error allocating space for $Name	There is an error in allocating memory for $Name.	Contact AIT framework team.
WARNING: ($RoutineName) Error allocating $VariableName structure array	There is an error in allocating $VariableName.	Contact AIT framework team.
WARNING: ($RoutineName) Failed to allocate structure	There is an error in allocating structure.	Check the memory or contact AIT framework team.
WARNING: ($RoutineName) Memory allocation problem	There is an error in allocating memory.	Contact AIT framework team.

WARNNG: ($RoutineName) Memory allocation problem in $RoutineName.	There is an error in allocating memory.	Contact AIT framework team.
WARNING: ($RoutineName) Problem allocating Framework_Input_Structure.	There is an error in allocating space for Framework_Input_Structure.	Contact AIT framework team.
WARNING: ($RoutineName) Segment_Variables is NULL.	Segment_Variables is not allocated, being NULL.	Contact AIT framework team.
WARNING: ($RoutineName) Unable to allocate algorithm for $PCF_ID.	There is an error in allocating algorithm for $PCF_ID.	Contact AIT framework team.
WARNING: ($RoutineName) Unable to allocate Processed for $Name.	There is an error allocating memory for variable $Processed for $Name.	Contact AIT framework team.
WARNNG: ($RoutineName) Unable to allocate $VariableName.	There is an error in allocating $VariableName.	Check the memory and contact AIT framework team.
WARNING: ($RoutineName) Unable to allocate Dependencies for Type: $Dependency_Type for Process: $Name	There is an error in allocating memory for dependency $Dependency_Type.	Contact AIT framework team.
WARNING: ($RoutineName) Unable to create	There is an error in creating algorithm $AlgorithmName.	Contact AIT framework team.

algorithms $AlgorithmName.		
WARNING: ($RoutineName) Unable to Point $PointerName to $VariableName	The pointer $PointerName is not allocated.	Contact AIT framework team.
Memory deallocation		
WARNING: ($RoutineName) Error deallocating $ArrayName	There are problems in deallocating $ArrayName.	Contact AIT Framework team.
WARNING: ($RoutineName) Error deallocating Data array after 1km data read.	There is an error in deallocating data array	Contact AIT framework team.
WARNING: ($RoutineName) Error deallocating $Name	There is a problem in deallocating $Name.	Contact the AIT framework team.
WARNING: ($RoutineName) Error deallocating Data array after 2km data read	There is an error in deallocating data array.	Check the AIT framework team.
WARNING: ($RoutineName) Error deallocating the array called $DataName: Read_netcdf	There is an error in deallocating data array for shortdata.	Contact AIT Framework team.
WARNING: ($RoutineName) Error deallocating ReflOffset arrays for	There is an error in deallocating arrays for $BandName.	Contact the AIT Framework team.

$BandName		
WARNING: ($RoutineName) Error deleting existing Process.	There is an error in deleting existing process.	Contact AIT framework team.
WARNING: ($RoutineName) Error destroying CRTM	There is an error in destroying CRTM.	Contact AIT framework team.
WARNING: ($RoutineName) Problem deallocating $Name.	There is an error in deallocating space for $Name.	Contact AIT framework team.
WARNING: ($RoutineName) Unable to deallocate $Name	There is an error in deallocating memory for $Name.	Contact AIT framework team.
WARNING: ($RoutineName) Unable to delete item from Precedence_Info.	There is an error deleting item for Precedence_Info.	Contact AIT framework team.
WARNING: ($RoutineName) Unable to delete $Name.	There is an error deleting $Name.	Contact AIT framework team.
Option setup		
WARNING: ($RoutineName) Invalid read option.	Read option should be OUTPUT_FILE_NETCDF_ALL or OUTPUT_FILE_NETCDF_SEGMENT.	Check the config file.
WARNING: ($RoutineName) Invalid write option.	Write option should be OUTPUT_FILE_NETCDF_ALL or OUTPUT_FILE_NETCDF_SEGMENT.	Check the config file.

Others		
WARNING: ($RoutineName) Channel not found in available channels.	The channels of some resolutions are not found in the available channels.	Check the resolution channels and the total channels.
WARNING: ($RoutineName) Dependency $DependencyName not found in table.	There is an error in finding the dependency $DependencyName in the table.	Contact AIT framework team.
WARNING: ($RoutineName) Missing dependency of type $DependencyType for $ProcessName. Either none specified or failed to run.	There is an error in finding the dependency type $DependencyType.	Contact AIT framework team.
WARNING: ($RoutineName) No dependencies listed for type $DependencyType.	There is an error in finding the dependencies of the dependency type.	Contact AIT framework team.
WARNING: ($RoutineName) No types exist for $ProcessName.	There is an error in finding the type for $ProcessName.	Contact AIT framework team.
WARNING: ($RoutineName) Pointers not associated	There is an error in associating the pointers.	Contact AIT framework team.
WARNING: ($RoutineName) $ProcessName not found in Precedence_Table.	There is an error in finding process $ProcessName in the Precedence_Table.	Contact AIT framework team.
WARNING: ($RoutineName)	There is an error in locating $ProcessName	Check the PCF and Config files; or contact AIT

$ProcessName not found in precedence table.	in precedence table.	framework team.
WARNING: ($RoutineName) $Process_Name not found in table.	Process $Process_Name is not found in precedence table.	Contact AIT framework team.
WARNING: ($RoutineName) $Process_Type does not have any dependencies for $Process_Name.	There is an error in finding the dependencies for process $Process_Name.	Contact AIT framework team.
WARNING: ($RoutineName) $ProcessType type dependency found in process $ProcessName does not have any dependencies.	Process $ProcessName has a type but has no dependency.	Contact AIT framework team.
WARNING: ($RoutineName) $RequiredProcessName required for $ProcessName not found in Precedence_Table.	There is an error in finding the required process $RequiredProcessName for $ProcessName.	Contact AIT framework team.
WARNING: ($RoutineName) StartIndex > EndIndex for PCFID $PCFID and variable $VariableName.	The start index should not greater than end index.	Check the channel index in the PCF file.
WARNING: ($RoutineName) Temporal does not exist. Unable to set	The Temporal of the process is NULL during setting Actual_Time.	Contact AIT framework team.

Actual_Time.		
WARNING: ($RoutineName) Temporal does not exist for $ProcessName.	The Temporal of the process $ProcessName is NULL.	Contact AIT framework team.
WARNING: ($RoutineName) Type $Type already exists in dependency list for $ProcessName.	In dependency list, there are more than one dependency type.	Contact AIT framework team.
WARNING: ($RoutineName) Unable to add dependency $Dependency_Name. Type not found: $Dependency_Type.	There is an error in adding dependency $Dependency_Name. Its type $Dependency_Type is not found.	Contact AIT framework team.
WARNING: ($RoutineName) Unable to convert channel index: $ChannelIndex for temporal pcf_id: $PCFID.	There is an error in converting channel index.	Contact AIT framework team.
WARNING: ($RoutineName) Unable to find PCF_ID: $PCF_ID.	There is an error in finding $PCF_ID.	Contact AIT framework team.
WARNING: ($RoutineName) Unable to find PCF_ID $PCF_ID for Timestep Type: $Timestep_ID timestep: $Time_String	There is an error in finding PCF_ID.	Contact AIT framework team.

WARNING: ($RoutineName) Unable to find $ProcessName in precedence table.	There is an error in finding the process $ProcessName in precedence table.	Contact AIT framework team.
WARNING: ($RoutineName) Unable to find Timestep: $Time_String	There is an error in finding Timestep for $Time_String.	Contact AIT framework team.
WARNING: ($RoutineName) Unable to get Actual Time for $AlgorithmName.	There is an error in getting Actual Time for $AlgorithmName.	Check the temporal setup in PCF.
WARNING: ($RoutineName) Unable to get PCF_IDs for $AlgorithmName.	There is an error in getting PCF_IDs for $AlgorithmName.	Check the temporal setup in PCF.
WARNING: ($RoutineName) Unable to get Temporal_List.	There is an error in getting Temporal_List	Contact AIT framework team.
WARNING: ($RoutineName) Unable to get Temporal Timestep for $AlgorithmName.	There is an error in getting Temporal Timestep for $AlgorithmName.	Check the temporal setup in PCF.
WARNING: ($RoutineName) Unable to get Temporal Timestep IDs for $AlgorithmName.	There is an error in getting Temporal Timestep IDs for $AlgorithmName.	Check the temporal setup in PCF.
WARNING: ($RoutineName) Unknown satellite	There is an error in determining the right satellite name.	Check the satellite name in PCF file.

name		
WARNING: ($RoutineName) $VariableName is NULL.	During processing $RoutineName, Nav structure is not allocated.	Contact AIT framework team.
PCF and Config file setup		
WARNING: ($RoutineName) Conflicting data in cfg file: RESOLUTIONS and ADDITIONAL_KEYS	Total resolutions in config file do not match the size of resolution list.	Check the config file.
WARNING: ($RoutineName) Error matching the Data and Time for Cloud mask and L1B Data	The warning message is issued because the Date and Time do not match.	Check the date and time of the satellite data file.
WARNING: ($RoutineName) Incorrect default Size_Factor value; resetting to 1	Size factor of resolutions is incorrect.	Check the config file.
WARNING: ($RoutineName) Incorrect default Total_Channels; resetting to Satellite Total Channels	Resolution total channels do not match the satellite total channels.	Check the config file.
WARNING: ($RoutineName) Insufficient data for $ResolutionKey	There are too many resolutions.	Check the config file.
WARNING: ($RoutineName) Invalid	$Value specified in DEPENDENCY_INFO_LOCATION should be	Check the config file.

DEPENDENCY_INFO_LOCATION: $Value	ALLOW_IN_MAIN or PCF_ONLY.	
WARNING: ($RoutineName) Invalid Keys: $Key	There is an error in finding PCF key.	Check the PCF.
WARNING: ($RoutineName) Invalid RUN_MODE: $RunModeValue	Run mode $RunModeValue should be ALLOW_DEFAULT_PCF or USER_DEFINED_PCF	Check the config file.
WARNING: ($RoutineName) Invalid Satellite PCF ID	Satellite PCF ID is wrong.	Check the satellite PCF ID in the config file and PCF file.
WARNING: ($RoutineName) $ProcessName does not have a default PCF.	The process $ProcessName does not have a default PCF.	Contact AIT framework team to add default PCF to this process.
WARNING: ($RoutineName) Problem creating products for $Main_PCF_Filename	There are some problems processing the file with name $Main_PCF_Filename in routine Process_PCF_File	Check your $Main_PCF_Filename to see if there is some entry in the file is wrong.
WARNING: ($RoutineName) SATELLITE_DATA_Src1->Planck.PlanckSize <= 0	The planck size should be greater than 0.	Check the PCF file and corresponding NetCDF file.
WARNNG: ($RoutineName) SDS_Dims(1) does not match size of DataArray 1	The dimension size does not match.	Check the dimension of NetCDF file.

WARNNG: ($RoutineName) SDS_Dims(2) does not match size of DataArray 2	The dimension size does not match.	Check the dimension of NetCDF file.
WARNING: ($RoutineName) Unable to determine geocat channel map	There are some errors in determining geocat channel map.	Check the channel setup in PCF file.
WARNING: ($RoutineName) Unknown $NameID	The $NameID cannot be found.	Check the config file and PCF.
WARNING: ($RoutineName) Unknown SATELLITE_DATA_ID	The SATALLITE_DATA_ID should be GOESR_ABI_CI or MSG_SEVIRI_GEOCAT.	Check the config file and PCF.
WARNING: ($RoutineName) Unknown Satellite ID	There is an error in finding satellite ID.	Check PCF and config file.
WARNING: ($RoutineName) User defined PCF found in main PCF & $ProcessName process PCF are not the same.	PCF file name are not the same.	Contact AIT framework team.

Table 8-3: JPSS RR Error Messages (Low-Level) – Framework UNKNOWN

Unknown Message	Explanation	Action

Data reading		
UNKNOWN: ($RoutineName) Error reading fog probability tables	There is an error in reading fog probability tables	Contact AIT framework team.
UNKNOWN: ($RoutineName) Error reading SO2 LUTs	There is an error in reading SO2 LUTs.	Check SO2 LUTs in NetCDF file.
UNKNOWN: ($RoutineName) Failed to get att $AttributeName	There is an error in getting attribute $AttributeName.	Check the attribute in NetCDF file.
UNKNOWN: ($RoutineName) Failed to get var $VariableName	There is an error in getting variable $VariableName.	Check the variable in NetCDF file.
UNKNOWN: ($RoutineName) Failed to inq $VariableName	There is an error in inquiring variable $VariableName.	Check the variable in NetCDF file.
UNKNOWN: ($RoutineName) Failed to read in $Name LUTs - stopping	There is an error in reading in $Name LUTs.	Check the file location in the PCF.
File management		
UNKNOWN: ($RoutineName) Failed to open $filename.	There is an error in opening $filename	Check file location in PCF.
Incorrect value		
UNKNOWN: ($RoutineName) Invalid water vapor channel.	The water vapor channel should be CHN_ABI8 or CHN_ABI9.	Check the PCF setup.

Memory allocation		
UNKNOWN: ($RoutineName) Cannot dellocate $Name LUTs - stopping	There is an error in deallocating $Name LUTs	Contact AIT framework team.
UNKNOWN: ($RoutineName) Error Allocating Arrays	There is an error in allocating arrays.	Check the memory and contact AIT framework team.
Memory deallocation		
UNKNOWN: ($RoutineName) Error deallocating Fog Luts	There is an error in deallocating Fog Luts	Contact AIT framework team.
Others		
UNKNOWN: ($RoutineName) ABI LUT's not implemented	The message is issued because framework currently does not support LUT for ABI.	Contact AIT framework team.
UNKNOWN: ($RoutineName) Error with lookup table for AOD or COT	There is an error in looking up table.	Contact AIT framework team.

Table 8-4: JPSS RR Error Messages (Low-Level) – Framework NOTICE

Notice Message	Explanation	Action

Data reading		
NOTICE: (read_netcdf) Error getting fill value ($DataType) from netcdf file: ‘ nf90_get_att. Using default missing value for $VariableName.	This NOTICE message says that there is an error in getting fill value from netcdf file.	Check the fill value of variable $VariableName.
NOTICE: ($RoutineName) Reading in SO2 LUT $Filename	The informational message says that the framework is reading in SO2 LUT from file $Filename.	Check the File location.
File management		
NOTICE: ($RoutineName) Finished reading $Name LUTs	This message says that the framework successfully finishes reading $Name LUTs.	No action is needed.
NOTICE: ($RoutineName) Reading in fog LUT $Filename.	This message says that the framework is reading in fog LUT file.	No action is needed.
Memory management		
NOTICE: ($RoutineName) Deallocating $Name LUT’s	This message says that the framework starts deallocating $Name LUT’s.	No action is needed.
NOTICE: ($RoutineName) Unable to Point to $VariableName	There are some errors in the pointer of variable $VariableName.	Check the channel setup in PCF file.
Others		

NOTICE: ($RoutineName) No training for the input satellite images day: $Day	The message says that the framework is going to get training for the input satellite images day	No action is needed.
PCF and Config file		
NOTICE: ($RoutineName) Unable to determine geocat channel map	There are some errors in determining geocat channel map.	Check the channel setup in PCF file.

Table 8-5: JPSS RR Error Messages (Low-Level) – Framework MISC

Misc Message	**Explanation**	**Action**
Incorrect value		
Warning: Month is out of range (1-12)	The input month is not in the range between 1 and 12.	Contact AIT framework team.
Memory allocation		
Error allocating memory for $VariableName: $Status	There is an error in allocating memory.	Contact the AIT framework team.
Error destroying allocate channel $ChannelNumber rtm profile.	There is an error in deallocating.	Contact AIT framework team.
FAIL ALLOCATION ARRAY:	There is an error in allocating memory for	Contact AIT framework team.

$ArrayName	$ArrayName.	
Failed to allocate $ArrayName	There is an error in allocating $ArrayName.	Contact AIT framework team.
$VariableName failed to allocate, most likely out of memory	There is an error in allocating memory.	Contact the AIT framework team.
Memory deallocation		
Deallocate error in BLINT, stat = $Status	There is an error in deallocating space.	Contact AIT framework team.
Deallocation error in $VariableName – Error: $Status	There is an error in deallocating memory.	Contact the AIT framework team.
Error deallocating memory for $VariableName: $Status	There is an error in deallocating memory.	Contact the AIT framework team.
Error deallocating $VariableName dependent arrays. STAT = $Status	There is an error in deallocating $VariableName arrays.	Contact AIT framework team.
Failed to deallocate $ArrayName	There is an error in deallocating $ArrayName.	Contact AIT framework team.
Other		
ERROR: Insufficient space for bit packing	The space exceeds the space allowed for the packed bytes.	Contact AIT framework team.

Error interpolating Ozone profile in the vertical - aborting	There is an error in interpolating ozone profile.	Contact AIT framework team.
Error interpolating profile in the vertical - aborting	There is an error in interpolating the current temperature and water vapor profiles.	Contact AIT framework team.

Table 8-6 shows the messages from the execution of other standalone program.

For error messages from H5AUGJPSS, please refer to the corresponding document contained in their associated software package.

Table 8-6: JPSS RR Error Messages (Low-Level) - Other Standalone Program

Message	Executable	Explanation	Action
Error allocating	get_sdr_scan_time.exe	There is an error in allocating memory.	Contact the AIT framework team.

ERROR: Could not open file: $Filename	get_data_nc.exe	There is a problem in opening the file $Filename.	Contact AIT framework team.
ERROR: Could not open data: $Variablename	get_data_nc.exe	There is an error in reading 1-dimensional $VariableName.	Contact AIT framework team.
ERROR: dataset $Variablename requires a valid channel index value	get_data_nc.exe	There is an error in reading 3-dimensional $VariableName	Contact the AIT framework team.

9. LIST OF REFERENCES

NESDIS/STAR (2014), JPSS Risk Reduction: Uniform Multi-Sensor Cryosphere Algorithms for Consistent Products Unit Test Readiness Review

NESDIS/STAR (2014), JPSS Risk Reduction: Uniform Multi-Sensor Aerosol Algorithms for Consistent Products Unit Test Readiness Review

NESDIS/STAR (2014), JPSS Risk Reduction: Uniform Multi-Sensor Cloud Algorithms for Consistent Products Unit Test Readiness Review

NESDIS/STAR (2012), JPSS Risk Reduction: Requirements Allocation Document

NESDIS/STAR (2014), JPSS Risk Reduction: Uniform Multi-Sensor Cryosphere Fractional Snow Cover Algorithm for Consistent Products Critical Design Review

NESDIS/STAR (2013), JPSS Risk Reduction: Uniform Multi-Sensor Cloud Algorithm for Consistent Products Critical Design Review

NESDIS/STAR (2013), JPSS Risk Reduction: Uniform Multi-Sensor Cryosphere Algorithm for Consistent Products Critical Design Review

NESDIS/STAR (2013), JPSS Risk Reduction: Uniform Multi-Sensor Aerosol, Volcanic Ash, and Daytime Cloud Optical and Microphysical Properties Algorithm for Consistent Products Critical Design Review

NDE (2013), Standards for Algorithm Delivery and Integration Using Delivered Algorithm Packages (DAPs), Version 1.4

NESDIS/STAR (2013), The NUCAPS System Maintenance Manual, Version 1.0.

NPOESS(2009), NPOESS Common Data Format Control Book, Version D.

NESDIS/STAR (2016), JPSS Aerosol Detection (Suspended Matter) Product Algorithm Theoretical Basis Document

NESDIS/STAR (2015), Ice Surface Temperature, Ice Concentration, and Ice Cover Algorithm Theoretical Basis Document

Satellite Products and Services Review Board (2015), AWG Cloud Height Algorithm (ACHA) Algorithm Theoretical Basis Document

Satellite Products and Services Review Board (2015), VIIRS Binary Snow Map Product Algorithm Theoretical Basis Document

Satellite Products and Services Review Board (2015), VIIRS Fractional Snow Cover Map Product Algorithm Theoretical Basis Document

NESDIS/STAR (2015), Ice Surface Temperature. Ice Concentration, and Ice Cover Algorithm Theoretical Basis Document

NESDIS/STAR (2015), ABI and VIIRS Ice Thickness and Age Algorithm Theoretical Basis Document

NESDIS/STAR (2015), AWG Cloud Height Algorithm (ACHA) Algorithm Theoretical Basis Document

NESDIS/STAR (2016), A Naïve Baynesian Cloud Mask Delivered to NOAA Enterprise Algorithm Theoretical Basis Document

NESDIS/STAR (2016), Daytime Cloud Optical and Microphysical Properties (DCOMP) Algorithm Theoretical Basis Document

NESDIS/STAR (2015), Enterprise Algorithm Theoretical Document for Volcanic Ash (Detection and Height)

NESDIS/STAR (2010), Enterprise Algorithm Theoretical Basis Document For Cloud Type and Cloud Phase

NESDIS/STAR (2016), EPS Aerosol Optical Depth (AOD) Algorithm Theoretical Basis Document

NESDIS/STAR (2015), EPS Aerosol Optical Depth (AOD) Algorithm Theoretical Basis Documentation

NESDIS/STAR (2015), JPSS Visible Infrared Radiometer Suite (VIRRS) Algorithm Theoretical Basis Document For Nighttime Could Optical Depth, Cloud Particle Size, Cloud Ice Water Path, and Cloud Liquid Water Path