



STAR JPSS Algorithms Integration Team Configuration Management Plan

Version 1.2

NOAA Center for Weather and Climate Prediction (NCWCP)
NOAA/NESDIS/STAR
5830 University Research Ct
College Park, MD 20740

Revisions

Version	Description of Version	Date Completed
0.1	Initial draft	7/23/2013
0.2	Revised “SCM Framework” section	8/8/2013
0.3	Revised “SCM Process” section	10/25/2013
1.0	Final version	11/06/2013
1.0.1	Updated for grammar correction and inappropriate phrase	12/13/2013
1.1	Added “3.8 JPSS ATBD Configuration and Change Control” section	05/22/2014
1.2	Revised “JPSS ATBD Configuration and Change Control” section	08/28/2014

Table of Content

REVISIONS 2

TABLE OF CONTENT 3

1. INTRODUCTION..... 4

 1.1 BACKGROUND..... 4

 1.2 PURPOSE 4

 1.3 SCOPE 4

 1.4 DEFINITIONS, TERMS, ACRONYMS, AND ABBREVIATIONS..... 5

 1.4.1 Terms..... 5

 1.5 OVERVIEW 7

2. THE CM FRAMEWORK..... 7

 2.1 ORGANIZATION, RESPONSIBILITIES, AND INTERFACES 7

 2.2 TOOLS, ENVIRONMENT, AND INFRASTRUCTURE..... 9

 2.3 ADMINISTRATION AND MAINTENANCE..... 9

3. THE CM PROCESS..... 10

 3.1 CONFIGURATION IDENTIFICATION..... 10

 3.1.1 Identification Methods and Naming Convention 10

 3.2 CONFIGURATION ITEM 12

 3.3 WORKSPACE MANAGEMENT 13

 3.4 PROJECT BASELINES AND BRANCHING STRATEGIES 13

 3.5 PROMOTION MODEL..... 14

 3.6 CONFIGURATION POLICY 15

 3.7 CONFIGURATION AND CHANGE CONTROL..... 15

 3.7.1 Change Request Processing 15

 3.7.2 Change Control Board (CCB)..... 17

 3.7.3 Discrepancy Report, Change Request and Baseline 17

 3.8 JPSS ATBD DOCUMENTATION CONFIGURATION AND CHANGE CONTROL..... 17

 3.8.1 Configuration Identification..... 18

 3.8.2 Configuration Management Procedure 18

APPENDIX: ACRONYMS 20

1. Introduction

This Configuration Management Plan (CMP) provides an overview of the organization, activities, overall tasks, and objectives of Configuration Management for the algorithm integration procedures that the JPSS Algorithm Integration Team (AIT) adopts.

The Configuration Management (CM) discipline is applied to controlled items for which the project organization has development and maintenance responsibilities. The CM organization implements the activities described within this plan to ensure that controlled products and controlled activities are compliant with CM policies and procedures and that changes are appropriately managed.

1.1 Background

The NOAA/NESDIS Center for Satellite Research and Applications (STAR) provides technical support of the Joint Polar Satellite System (JPSS) algorithm development and integration tasks in coordination with the JPSS Ground Project Data Products Engineering and Services (DPES) Team.

The AIT brings technical expertise and support to product algorithms, specifically in testing and validating science algorithms in the Algorithm Development Library (ADL) environment. The AIT assists JPSS science teams in implementing algorithm changes.

1.2 Purpose

The purpose of this document is to identify and describe the overall practices and methods for CM to be used during the lifecycle of JPSS AIT ADL project, and will be updated progressively as the implementation of the components of this plan and the requirements / necessities arise. This CMP provides overall CM guidance and direction to establish unified change management to control the JPSS algorithms updates integrating, testing and validating in ADL within STAR AIT.

1.3 Scope

The *Configuration Management Plan (CMP)* identifies and describes a configuration management (CM) process for JPSS AIT ADL project. This document describes a set of CM terminology, establishes guidelines and standards for system configuration, problem tracking and problem resolution processes. It is expected that both AIT and potential science team developers follow the established guidelines and standards.

This CMP addresses configuration item (CI) identification, change control, and configuration audits. The details regarding CM activities, techniques, and tools are provided in the CM-related procedures.

1.4 Definitions, Terms, Acronyms, and Abbreviations

Configuration Management (CM) is a discipline for ensuring the integrity of a product and to make its evolution more manageable. CM can be defined as the management of security features and assurances through control of changes made to hardware, software, firmware, documentation, test, and test documentation throughout the life cycle of an information system.

CM provides four principal services:

1. Maintaining a project-wide history of the development process.
2. Providing a project-wide "undo" capability.
3. Automating product builds and making them reliable and efficient.
4. Providing for parallel development by baselining "good" builds and providing a defined branching structure for the project.

Algorithm Development Library (ADL) is a programming environment maintained by JPSS Ground Project DPES. ADL emulates the programming interface available within the JPSS operational system. It provides a diagnostic framework that supports diagnostic code blocks within operational algorithms.

1.4.1 Terms

- *Element*: Elements are control objects that include a version tree.
- *Version*: A version is a specific revision of an element.
- *Versioned Object Base (VOB)*: A repository that stores versions of file elements, directory elements, derived objects, and metadata associated with the objects.
- *VOB Database*: The part of a VOB storage directory in which ClearCase metadata and VOB objects are stored.
- *Views*: A view, which is represented as a directory, provides access to a specific version of one or more elements in a VOB.
- *Snapshot View*: A workspace created on your local computer, in which the versions of elements are copied from VOBs. You have to update your view periodically to be able to see the latest versions of elements.
- *Dynamic View*: A dynamic view provides immediate, transparent access to data stored in VOBs, based on configuration specification selection rules. (no copy to local compute, no update needed)

- *Checkout*: A checkout model enables you to get a private and editable copy of a specific element and manage changes to your projects.
- *Check In*: Updating a new version of the elements to the VOB.
- *Branch*: A branch is an object that specifies a sequence of versions of an element. Branching is done in order to provide work areas for development.
- *Version Label*: A version label can be attached to any version of an element to identify that version in an easy to remember way.
- *Configuration Specification*: A configuration specification contains the rules used by a view to select versions of elements. The rules are very flexible, and you can use various specifiers to indicate which versions to select. A view has exactly one configuration specification.
- *Unified Change Management (UCM)* simplifies development by raising the level of abstraction to manage changes in terms of activities, rather than manually tracking individual files and elements. Any change made to any (ClearCase-stored) file or element must, under UCM rules, be made in the context of a task defined in ClearQuest. Technically, ClearQuest displays and manages these tasks as change request tickets, while ClearCase is the tool that enables task-driven change control.
- *Baseline*: A baseline identifies one version of each element in a component that represents the integrated or merged work of team members. It is a snapshot or a position which is recorded. The baseline remains unchanged and available as a reference and comparison against the future changes.
- *Project*: A UCM Project is a logical unit that is mapped to the development structure of an application or system. These projects contain the configuration information (example: components, activities, policies) needed to manage and track the work on a specific application of a development effort. These UCM project configurations are stored within a storage structure called Project Version Object Base (PVOB).
- *Component*: A component is a group of file and directory elements (example: source code, scripts, documentations) that are versioned together and stored within a VOB. A VOB could host one or more components. A project team develops, integrates, and releases a component as a unit.
- *Activity*: An activity is an object that records the set of files or elements (change set) that a developer creates or modifies to complete and deliver a development task, such as a new feature. The activity title usually indicates the cause of the change and is linked to a change request (CR) in ClearQuest.

- *Stream*: A stream is a ClearCase UCM object that maintains a list of activities and baselines and determines which versions of elements appear in your view. In UCM, streams are layered.
- *Rebase*: The Rebase operation reconfigures a stream by adding, dropping, or replacing one or more of the stream's foundation baselines. The file and directory versions selected by those new baselines then become visible in the stream's views.
- *Deliver*: The Deliver operation lets you deliver work from a source stream to a target stream in the same or a different project.

1.5 Overview

This document is divided into three major sections, the Introduction, the Configuration Management (CM) Framework, and the CM Process. The organization of this document is as follows:

- Section 1.0 – Introduction: This section presents preliminary information concerning this document. The introduction provides background information about the AIT work, the scope of this document, the organization of this document, and any references utilized in the assembly of this document.
- Section 2.0 – CM Framework: This section describes the organization and the responsibilities in CM process workflow, the system environment and the software tools to be used in fulfilling the CM functions and the administration & maintenance for the CM implementation.
- Section 3.0 – CM Process: This section describes the work area policies, naming conventions, the branch strategies and the change control in the configuration management process. In section 3.8, the JPSS ATBD Documentation Configuration and Change Control are covered.

2. The CM Framework

2.1 Organization, Responsibilities, and Interfaces

Configuration Management tasks are the responsibility of the Configuration Manager, the System Leads, the Algorithm Leads, the Integration Leads, QA personnel, documentation specialists and the individual developers.

SCM Role	Who	Responsibilities
----------	-----	------------------

System Leads	Walter Wolf Bigyani Das	<ul style="list-style-type: none"> • Define general policies • Define access control • Define integration milestones • Define baseline strategy • Review and approve AIT change requests
Integrator	AIT members	<ul style="list-style-type: none"> • Coordinate the integration of algorithm updates delivered by science team to ADL • Produce and manage internal baseline • Identify build candidates • Produce successful build and promote baseline (INITIAL to BUILT) • Export ADL to STAR users as requested
Algorithm Leads	JPSS Science Team Leads	<ul style="list-style-type: none"> • Review and approve science team code change requests • Deliver code changes to AIT integrators
CM Administrator	Yunhui Zhao	<ul style="list-style-type: none"> • Perform CC/CQ License administration • Backup and restore CC VOBs and CQ databases • Perform network administration • Perform user access control to CC projects • Conduct troubleshooting of CM environment
Configuration Manager	Yunhui Zhao	<ul style="list-style-type: none"> • Maintain CMP • Identify Configuration Items (CI) • Manage the configuration management database • Define CM policies • Create CM repositories • Integrate the official ADL and IDPS MX versions and create corresponding integration streams • Schedule CC/CQ maintenance • Perform CC operation management • Provide CM training to developers • Provide CM technical assistance
Developer	AIT members	<ul style="list-style-type: none"> • Create development streams and views • Work with configuration items • Promote changes
Quality Assurance (QA) Personnel	AIT members	<ul style="list-style-type: none"> • Ensure that the algorithm software complies with established software standards • Verify code changes delivered by science teams • Chain run products to test and validate science algorithms within ADL environment • Promote baseline (BUILT to TESTED) • Ensure the packages delivered to IDPS are functional and complete

Documentation Specialist	Valerie Mikles	<ul style="list-style-type: none"> • Technical writing • Maintain documentation • Track DR difference between official release and the baseline in AIT CM • Maintain JPSS Input Database
STAR JPSS QA	Xingpin Liu	<ul style="list-style-type: none"> • Integrate updated ATBD into ClearCase • Establish baselines for JPSS ATBD updates • Publish updated ATBD online

2.2 Tools, Environment, and Infrastructure

SCM Tool: IBM Rational ClearCase/ClearQuest

ClearCase is a software tool for revision control of source code and other software development assets. ClearCase handles version control, parallel development, workspace management, process configurability, and build management. It also enforces the development process. ClearCase supplies all the basic tools to make CM configuration very configurable and flexible.

ClearQuest provides activity-based change and defect tracking. It can manage all types of change requests, including development, defects, and enhancement changes.

Unified Change Management (UCM) simplifies development by raising the level of abstraction to manage changes in terms of activities, rather than manually tracking individual files and elements. Any changes made to any ClearCase elements must, under UCM rules, be made in the context of a task defined in ClearQuest. Technically, ClearQuest displays and manages these tasks as change requests, but ClearCase actually is the tool that enables task-driven change control.

2.3 Administration and maintenance

A daily backup strategy is adopted for JPSS AIT VOBs and Views. The registry, License and Administration files will be backed up every month. The view check will be performed regularly to eliminate unused views and avoid oversize views.

The Configuration Manager is responsible for maintaining the JPSS AIT CMP. This document is a configuration-controlled item. Any requests for changes to the CMP will be discussed at weekly JPSS AIT meeting.

3. The CM process

3.1 Configuration Identification

Configuration identification is a discipline for identifying an item and documenting its functional and physical characteristics.

3.1.1 Identification Methods and Naming Convention

- **Stream**

Streams (aka. branches) are used to enable parallel development. The major integration stream (**ADL_MAJOR_INT**) should not be used for development work, but reserved for major milestones or officially released versions, so all major development is done on sub-streams.

The integration streams are directly under major integration stream marked with specific ADL version and IDPS MX release info as:

`<ADL version>_<IDPS MX version>_DEV_INT`

For example: **ADL41_MX67_DEV_INT**. Developers are supposed to deliver their updates/changes to integration streams and rebase to get others' updates from integration streams.

The developers create own development streams under desired integration streams. The development stream naming convention is established as:

`<Owner>_<ADL version>_<IDPS MX version>_dev`

For example **yzhao_ADL41_MX67_dev** is the stream on which developer yzhao update delivered algorithm updates to ADL version 4.1 and MX version 6.7. If necessary, stream names can be much longer and more descriptive, but Long stream names can crowd a version tree display.

Last but not the least, test streams can be created directly under integration streams ending with “_test” as:

`<Owner>_<ADL version>_<IDPS MX version>_test`

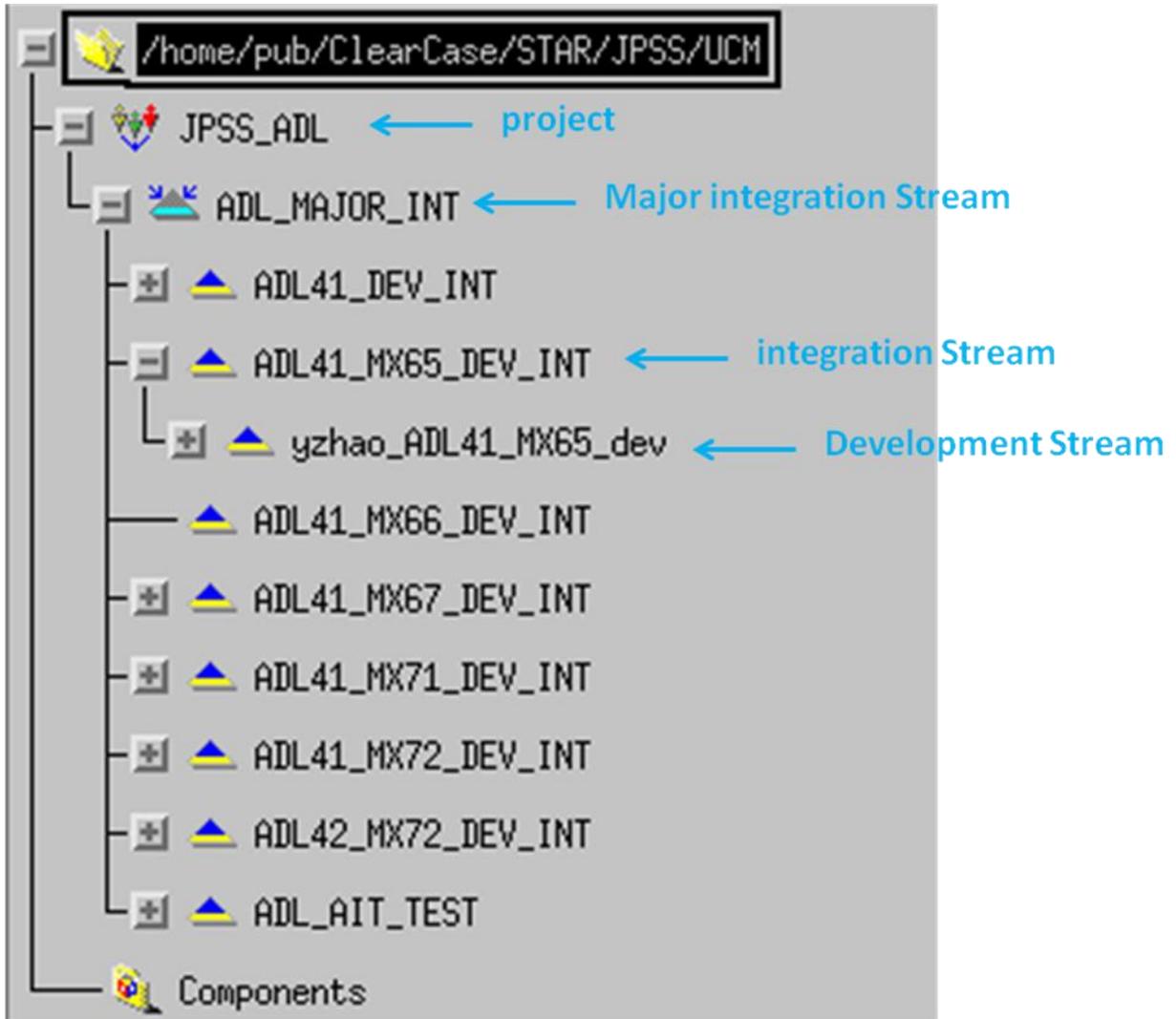


Figure 1 Stream hierarchy in JPSS AIT CM

- **View**

A good naming convention is also needed for views for the same reason as for stream. View name should be associated with the view owner work on project, and the task it is used to.

<Owner>_<ADL version>_<IDPS MX version>_<Purpose>

For example view “**yzhao_ADL41_MX67_int**” is created by yzhao for integration work (deliver operation) on released version “ADL v4.1 + MX v6.7” and view “**yzhao_ADL41_MX71_dev**” is created by yzhao for development based on released version “ADL v4.1 + MX v7.1”.

- **Baseline**

Baseline clearly indicates the particular stage on the project development. It is usually used on the integration branch to indicate milestones, releases, deliveries.

Three types of baseline will be generated based on different stream hierarchies in AIT CM.

- Official release baseline label on major integration stream is established as:

<Project name>_<ADL version>_<IDPS MX version>

For example baseline “**JPSS_ADL4.1_MX7.1**” will be the baseline for ADL version 4.1 and IDPS MX version 7.1 algorithms package which include corresponding CIs listed on the official release.

- Baseline on integration stream branch should include the project name, baseline owner and the time of creation.

<Project name>_<Owner>_<month_day_year_hhmm>

For example baseline “**JPSS_ADL_yzhao_8_8_2013_1430**” was created by “yzhao” on August 8th, 2013 at 14:30.

- Baselines on development stream are not mandated but suggested to follow the naming convention as those on integration stream.

3.2 Configuration Item

A Configuration Item is an object that is treated as a self contained unit for the purpose of identification and change control. Identifying CIs is a fundamental CM responsibility. Items subject to configuration control within JPSS AIT ADL project include:

- Configuration Management Plan (CMP)
- ADL software
- IDPS Algorithm packages
- Delivered Algorithm updates
- Acquired software (e.g., COTS)
- LUTs
- AIT scripts (AIT ADL environment setup, chain run, etc.)
- Tools (e.g., compilers, debuggers, Perl modules)
- AIT documentations
- JPSS ATBD documentations

3.3 Workspace Management

All source code, documentation and test procedures are maintained on ClearCase VOB server and they can be accessed with the proper directories.

- VOB Root: */home/pub/ClearCase/STAR/JPSS/ADL*

Both dynamic view and snapshot view can be used to make changes to the objects in VOBs.

3.4 Project Baselines and Branching Strategies

Baseline provides an official standard on which subsequent work is based where only authorized changes are made. The Configuration Manager / Integration Lead will import the updates from ADL software and IDPS algorithm packages into AIT CM when the new release is available in Raytheon Common CM. A unique label will be given for each file associated with a baseline on major integration stream to identify the original version of corresponding release.

Branches (aka “streams” in ClearCase UCM) are used to enable parallel development. There are four branch types used in JPSS AIT CM: (see *Figure 2 Branching Strategy in JPSS AIT CM*)

- **ADL_MAJOR_INT** branch maintained by Configuration Manager / Integration Lead is the official release code base.
- **_DEV_INT** branch created by Configuration Manager/Integration Lead is for AIT internal algorithm integration release. The initial baseline is always synchronized with the official release from Raytheon Common CM.
- **_dev** branch is created by developers for private development and algorithm integration. **_dev** branches can only be created under **_DEV_INT** branch.
- **_test** branch is used by QA to verify changes. **_test** branch is directly under **_DEV_INT** branch.

The **ADL_MAJOR_INT** branch would remain unchanged between releases and all official release integration and testing would be done on it. After testing determines that the software is stable and ready, a baseline will be labeled on the latest version on **ADL_MAJOR_INT** and released. The configuration manager creates a **_DEV_INT** branch with the new released baseline as the foundation baseline. Developers and QA create their **_dev** and **_test** branch from **_DEV_INT** branch.

When a change request is implemented and tested, developers deliver the updates from their **_dev** branch to **_DEV_INT** branch. Others now can get updates by using the rebase operation from **_DEV_INT** branch to their **_dev** or **_test** branch.

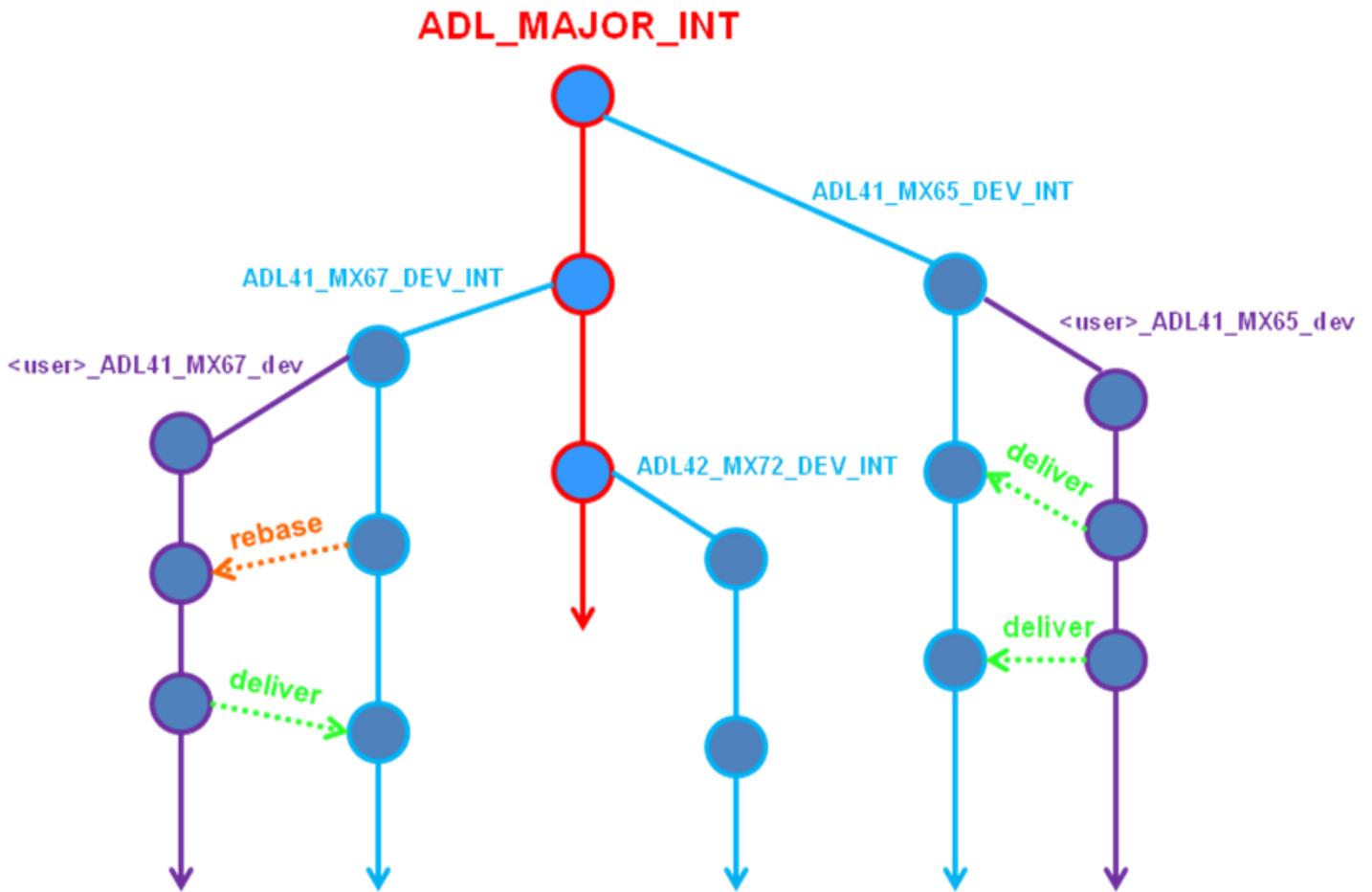


Figure 2 Branching Strategies in JPSS AIT CM

3.5 Promotion Model

There are five promote levels in JPSS AIT CM: INITIAL, BUILT, TESTED, REJECTED, and RELEASED.

The release of ADL software and IDPS algorithm packages from Raytheon Common CM is defined as an official release. The updates from each new release will be integrated into major integration stream **ADL_MAJOR_INT** in JPSS AIT CM. After successful testing, a release baseline is generated and promoted to “RELEASED” level.

All major development will be implemented on stream **_dev**. Before delivery can be made, developers are responsible for the compatibility of the changes they made with others’ updates before this delivery. AIT CM policy requires developers to rebase to the latest recommended baseline to include others’ previous deliveries prior to their own delivery. After the compatibility testing, developers deliver their changes to **_DEV_INT** stream and generate

a baseline. The baseline with recent deliveries will be classified as the default “INITIAL” level. After the integration test and a successful build, the baseline can be promoted to “BUILT”. Furthermore, after QA test and validate the changes, the baseline will be promoted to either “TESTED” or “REJECTED” based upon the test results.

3.6 Configuration Policy

For the security, stability and traceability of JPSS AIT CM system, CM policies are given to whoever uses this system as followed.

- 1) Do not use *rmelem* command to remove element from version control system. *It destroys information irretrievably.*
- 2) No development streams can be created under a major integration stream.
- 3) Adding meaningful comments when checking in and checking out is required.
- 4) All files under version control will have their ownership changed to VOB admin.
- 5) Naming conventions must be strictly followed.
- 6) Only designated personnel are allowed to merge or check out on main integration branch.
- 7) Deliver cannot be conducted with checkouts in the selected activities.
- 8) Rebase development stream to the recommended baseline prior to deliver.
- 9) Rebase cannot be performed while there are checkouts in the current view.
- 10) Empty branches should be removed after checkout is reversed.
- 11) Do not create a branch with the same name as a label.

3.7 Configuration and Change Control

The monitoring of a change request from open to close is performed through the change control process defined in this Configuration Management Plan. During the development lifecycle, changes to the developmental configuration are processed in accordance with the methods and procedures described in this CMP.

3.7.1 Change Request Processing

Algorithm updates delivered from JPSS Science teams will be integrated to baseline configuration items in JPSS AIT CM. The IBM Rational ClearQuest tool is used to initiate tracking, and will also report all AIT change requests. Once the updates are delivered from the science teams, JPSS AIT personnel will analyze the changes and submit the change requests into IBM Rational ClearQuest.

JPSS AIT change requests will be discussed during AIT meetings and implemented by AIT developers. A change will be entered by generating one of the following forms in ClearQuest;

- **Development:** A form used to propose a configuration change. A development change request is generated when a configuration change is required to the system. The change may be an algorithm update, AIT scripts change, AIT documentation change or a change to any other configured item. When changes are completed, developers can close the development change requests by delivering them.
- **Defect:** A form used to record bugs and their resolutions. A defect change request is generated when a defect is discovered in the system and a fix is required. The “Priority” and “Severity” are required to be entered when a defect change request is generated. A defect change request can only be completed when the changes get tested and validated. QA is responsible for changing the resolution state. Only the change requests with “Fixed” resolution state can be delivered.

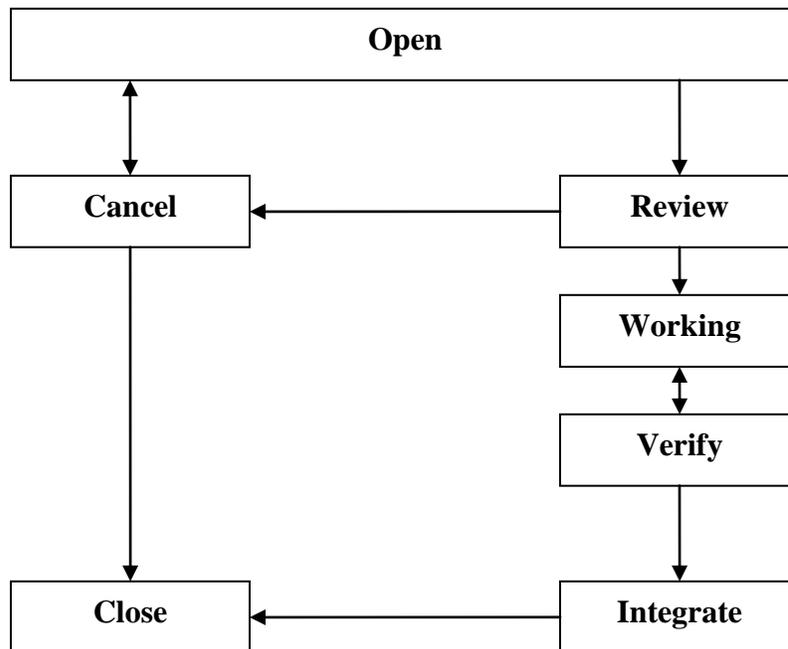


Figure 3 Change Request Process

3.7.2 Change Control Board (CCB)

JPSS AIT Change Control Board consists of the following:

Role	Responsibilities
System Leads	Review/Assign change requests on AIT part
Science Team Leads	Review/Approve changes and deliver changes to AIT
Integration Lead	Integrate/Promote baseline for ADL and IDPS new releases
Developer	Submit change requests / Integrate changes delivered by science teams
Quality Assurance	Verify changes / Test baseline / Close requests

In this CCB, AIT personnel play different roles. They join regular meetings with the science teams for the review of upcoming algorithm updates and submit proper change requests for them. AIT developers also generate change requests for AIT internal development including source code, scripts, documentation and the changes for any other configured items.

All the changes requests will be reviewed by system leads during the weekly JPSS AIT meeting. The assigned change requests will be implemented by developers based upon the priorities of change requests. QA will be responsible for conducting compatibility / integration / function / performance testing.

JPSS AIT CCB meeting could be arranged when it is necessary to convene all participants at a time.

3.7.3 Discrepancy Report, Change Request and Baseline

Algorithm changes are proposed by the Science Teams. The proposal is recorded through a Discrepancy Report (DR) filing system. A change request in AIT CM should be generated with a formal filed DR. The change request will document all the changes involved in this DR. After the change request is verified through various testing, the AIT developer will deliver the changes to the integration stream and a baseline will be generated to recode the version changes.

At some point, the algorithm code in AIT CM might be a bit advanced in relation to the new release from Raytheon Common CM dependent upon the DR deployment to operations. The latest baseline from the most recent integration stream will be used to synchronize the code between AIT CM and the new release from Raytheon Common CM.

3.8 JPSS ATBD Documentation Configuration and Change Control

With the transition of JPSS ATBD to STAR, AIT CM is responsible to perform the configuration and change control through the ATBD documents development lifecycle after the transfer. This CM plan also covers the integration process of JPSS ATBD into AIT CM.

3.8.1 Configuration Identification

All the transferred JPSS ATBDs (both WORD and PDF formats) will be selected as CIs and labeled with appropriate baselines to establish and maintain a definitive basis for control and status accounting. Other than algorithms updates integration which is implemented on project “JPSS_ADL”, ATBD updates will be integrated into ClearCase on an individual project “JPSS_ATBD”. The major integration stream will be “JPSS_ATBD_INT”.

ATBD CI Naming Convention:

<Document ID#>_<Document Type>_<Product Name>_<Version#>

- *Document ID#:* *STAR Document CM #*
- *Document Type:* *ATBD*
- *Product Name:* *Algorithm name*
- *Version #:* *ATBD Version name*

There might be different ATBDs for the same product for different missions (SNPP, J1, etc.). In this scenario, mission name will be identified as a part of Product Name.

Baselines are established when it is necessary to define a formal milestone for control of future changes. The project identifies JPSS ATBD for the applicable updates integration. Configuration baselines will include the ATBD updates identified in the Change Request (CR). A formal baseline is established whenever updates integration is completed. Any changes made to the formal baseline of ATBD are associated with an approved CR.

ATBD baseline Naming Convention:

BL_<Establish Date>_CR-<Unique CR ID#>-<AERB Approve Date>

The initial baseline is established with the current available version on STAR JPSS ATBD web with name “BL_<Establish Date>_STAR-Web”. Starting from next release, the baseline will includes the time when the baseline is created and the change request information and its approved date from Algorithm Engineering Review Board (AERB).

3.8.2 Configuration Management Procedure

- Science teams update ATBD in conjunction with code changes.
- Science teams deliver updated ATBD to STAR AIT as part of algorithm delivery package.

- STAR AIT prepares algorithm delivery package and deliver it to DPES AIT.
- After validated by DPES AIT, the algorithm delivery package is delivered to AERB for review.
- Once get approved from AERB, the updated ATBD will be integrated into ClearCase.
- After the ATBD updates integration is completed, a formal baseline will be established by STAR JPSS QA. The PDF format of the latest ATBD will be synchronized to the corresponding directory on STAR JPSS web server so the up-to-date content will be available immediately online.

Appendix: Acronyms

- ADL Algorithm Development Library
- AERB Algorithm Engineering Review Board
- AIT Algorithm Integration Team
- ATBD Algorithm Theoretical Basis Document
- CC ClearCase
- CM Configuration Management
- CQ ClearQuest
- CR Change Request
- DPE Data Products Engineering
- DR Discrepancy Report
- EDR Environmental Data Record
- IDPS Interface Data Processing Segment
- JPSS Joint Polar Satellite System
- LUT Look Up Table
- QA Quality Assurance
- SCM Software Configuration Management
- SDR Sensor Data Record
- SST Sea Surface Temperature
- VIIRS The Visible Infrared Imaging Radiometer Suite
- VOB Versioned Object Base