***Algorithmic Trading***

***System***

***Configuration***

***Management Plan***

Okanagan College

Algorithmic Trading System

2024-04-09

Version 1.6

Authors: Jacob Rawlings,Jake Fischer**,** Alan Abdollahzadeh, Devon Volberg, William Blake, Benjamin Carrier, Vanessa Dubouzet, Dakota Flath, Parker Green, Yuan Hu, Jacob Labelle, Isaac Lengacher-Bergeron , Dominic Presch, Dylan Soares, Jaedan Soukoroff

**Revision Sheet**

| Revision | Date | Brief Summary of Changes |
| --- | --- | --- |
| Version 0.0(draft) | 2023-10-01 | Baseline document draft |
| Version 0.1 | 2023-10-02 | formatting and some notes for future text |
| Version 0.2 | 2023-10-03 | Writing for section 1 |
| Version 0.3 | 2023-10-04 | Writing for section 2, starting 3 |
| Version 1.0 | 2023-10-06 | Main body sections written. “Training” section is left for now as discussion needs to take place. |
| Version 1.1 | 2023-10-12 | General editing, added training section, added diagram to baseline change process |
| Version 1.2 | 2023-10-26 | Revised sections 1 through 6 |
| Version 1.3 | 2023-11-05 | Reviewed all sections |
| Version 1.4 | 2023-12-04 | Modified some of the sections to reflect current setup. Reviewed the document and updated some links. |
| Version 1.5 | 2024-03-14 | Minor revisions, updated Section 5 |
| Version 1.6 | 2024-04-09 | Final Revisions |

[**1. INTRODUCTION 3**](#_heading=)

[**1.1 Purpose 3**](#_heading=)

[**1.2 Scope 3**](#_heading=)

[**1.3 Definitions and Acronyms 3**](#_heading=)

[**1.3.1 Key acronyms 3**](#_heading=)

[**1.3.2 Key terms 3**](#_heading=)

[**1.4 References 5**](#_heading=)

[**2. Software Configuration Management (SCM) 6**](#_heading=)

[**2.1 SCM Organization 6**](#_heading=)

[**2.2 SCM Responsibilities 6**](#_heading=)

[**2.3 Relationship of CM to the software process life cycle 7**](#_heading=)

[**2.4 SCM Resources 7**](#_heading=)

[**3. SCM ACTIVITIES 7**](#_heading=)

[**3.1 Configuration Identification 7**](#_heading=h.3as4poj)

[**3.1.1 Specification Identification 8**](#_heading=)

[**3.1.2 Change Control Form Identification 8**](#_heading=)

[**3.1.3 Project Baselines 8**](#_heading=)

[**3.2 Configuration Control 9**](#_heading=)

[**3.2.1 Procedures for changing baseline 9**](#_heading=)

[**3.2.2 Procedures for processing change requests and approvals 10**](#_heading=)

[**3.2.3 Change Control Boards (CCBs) 10**](#_heading=)

[**3.2.4 Interfaces 10**](#_heading=)

[**3.2.4.1 IONOS Integrations 10**](#_heading=)

[**3.2.5 Level of control 11**](#_heading=)

[**3.2.6 Document revisions 11**](#_heading=)

[**3.3 Configuration Status Accounting 11**](#_heading=)

[**3.3.1 Storage, handling and release of project media 12**](#_heading=)

[**3.3.2 Release process 12**](#_heading=)

[**3.3.3 Document Status Accounting 12**](#_heading=)

[**3.3.4 Change Management Status Accounting 12**](#_heading=)

[**3.4 Configuration Audits and Reviews 12**](#_heading=)

[**4. CM Milestones 13**](#_heading=)

[**4.1 Configuration Management Plan Approval 13**](#_heading=h.46r0co2)

[**4.2 Baseline Establishment 13**](#_heading=h.2lwamvv)

[**4.3 Change Management Implementation 13**](#_heading=h.111kx3o)

[**4.4 Release Management 13**](#_heading=h.206ipza)

[**4.5 Project Closure Review 13**](#_heading=h.2zbgiuw)

[**5. Training 14**](#_heading=)

[**6. Subcontractor/Vendor Support 15**](#_heading=h.34g0dwd)

[**APPENDIX A // Software Change Request Procedures 16**](#_heading=h.43ky6rz)

# **INTRODUCTION**

## **Purpose**

The purpose of this document is to define the procedures relating to version management, system building, change management, and release management that will be performed in the development of the Algorithmic Trading System (ATS) software project. The ATS will collect financial data from websites and API’s, preprocess that information, and store it for future use by Machine learning models.

## **Scope**

Configuration Management (CM) covers activities for all data and software produced during the development of the Algorithmic Trading System (ATS) software. This document applies to all products, services, and data developed and maintained for the ATS software. The CM activities defined here will be applied to all future ATS development cycles. This document conforms to the IEEE standards for software configuration management, and references the ATS Change Request Policy document.

## **Definitions and Acronyms**

### **Key acronyms**

ATS - Algorithmic Trading System

CER - Change/Enhancement Request

CM - Configuration Management

CMP - Configuration Management Plan

CCB - Change Control Board

CRC - Change Request Coordinator

DT&E - Developmental Test & Evaluation

SCM - Software Configuration Manager

SCR - Software Change Request

PO - Product Owner

SM - Scrum Master

VC - Version Control

VDD - Version Description Document

CI - Configuration Item

### **Key terms**

The following key terms are used as defined in the IEEE Guide for the Use of IEEE Standard Dictionary, else they will have one provided or referenced.

**configuration** “A discipline applying technical and administrative direction and

**management** surveillance to: identify and document the functional and physical

characteristics of a configuration item, control changes to those

characteristics of a configuration item, control changes to those

characteristics, record and report change processing

implementation status, and verify compliance with specified requirements.” [IEEE90].

**customer** “The individual or organization that specifies and accepts the project deliverables. The customer may be internal or external to

the parent organization of the project, and may or may not be the

end user of the software product. A financial transaction between

the customer and developer is not necessarily implied.” [IEEE87]

**database** “A collection of data fundamental to a system.” [IEEE91]

**installation** “The period of time in the software life cycle during which a

software product is integrated into its operational environment and

tested in this environment to ensure that it performs as required.”

[IEEE91]

**plan** “A detailed scheme, program, or method worked out beforehand

for the accomplishment of an objective.” [Heritage85]

**process** “A sequence of steps performed for a given purpose.” [IEEE90]

**project** -- unit of work to meet a specific customer requirement.

Includes all tasks, activities, and functions necessary to meet the

requirements.

**project** “The work product(s) to be delivered to the customer. The

**deliverables** quantities, delivery dates, and delivery locations are specified in

the project agreement.” [IEEE87]

**quality** “(1) A planned and systematic pattern of all actions necessary to

**assurance** provide adequate confidence that an item or product conforms to

established technical requirements.” [IEEE90]

“(2) A set of activities designed to evaluate the process by which

products are developed or manufactured.” [IEEE90]

**review** --A process or meeting during which a work product, or set of

work products, is presented to program personnel, managers, users,

customers, or other interested parties for comment or approval.

Types include requirements review, design review, code review,

test readiness review, formal qualification review.

**software** “Computer programs, procedures, and associated documentation

and data pertaining to the operation of a computer system.”

[IEEE90]

**software life** “A project-specific, sequenced mapping of activities.” [IEEE91]

**cycle**

**software quality** --See quality assurance.

**assurance**

**specification** “A document that specifies, in a complete, precise, verifiable

manner, the requirements, design, behavior, or other characteristics

of a system or component, and, often, the procedures for

determining whether these provisions have been satisfied.”

User Manual [IEEE90]

**walk-throughs** “A static analysis technique in which a designer or programmer

leads members of the development team and other interested

parties through a segment of documentation or code and the participants ask questions and make comments about possible

errors, violations of development standards, and other

problems.”[IEEE90]

**product** --Any tangible item that results from a project function, activity, or

task. Examples of work products include customer requirements,

project plan, design documents, source and object code, user’s

manuals.

## **References**

* [STSPF Research Paper](https://arxiv.org/abs/2309.00618) [r1]
* [Software Design Document](https://docs.google.com/document/u/0/d/1kH4S7RFlHHq6SzOlhkYMcuc-0IA7yYNazzqXoAM9lVY/edit) [r2]
* [System Requirements Specification](https://docs.google.com/document/u/0/d/1CbAF2mOfkrObv0FZ-JKMiEawvmQiZl1uLMKGMOtAUPg/edit) [r3]
* [User Manual](https://docs.google.com/document/u/0/d/1MSl4BucQyQG3XGQqiSBJuw8ncuh2JzPF/edit) [r4]
* [Github learning resources](https://docs.github.com/en/get-started/quickstart/git-and-github-learning-resources) [r5]

# **Software Configuration Management (SCM)**

## **SCM Organization**

The Configuration Management organization for this software has responsibilities assigned to each member of the project team. The development team fulfills the role of Change Control Board (CCB). The development team for the SMF software consists of 3 roles:

* Product Owner
* Scrum Master
* Developer

## **SCM Responsibilities**

The roles and their specific responsibilities are as follows:

Product Owner:

* The Product Owner (PO) is responsible for ensuring that the project roadmap is in alignment with Configuration Management goals
* Approves the Configuration Management Plan
* The Product Owner works with the SM and the development team to develop, maintain, and implement effective configuration management.
* Ensure that all Change/Enhancement Requests (CER) are documented properly

Scrum Master:

* The Scrum Master (SM) monitors, reviews, and approves Change Requests
* Ensures that Agile principles are being integrated into the development lifecycle
* Oversees project schedule and deliverables

Developers:

* Follow version control procedures
* Document code changes
* Oversee Version Control (VC) best practices
* Provide feedback or insight on change requests

Change Control Board:

* The Change Control Board (CCB) for ATS consists of the PO, the SM, and the development team.
* Responsible for the review and approval of all Software Change Requests (SCRs).
* It ensures that SCRs align with project goals and requirements. The CCB considers the cost, quality, schedule, and impact of all proposed changes.
* Has the authority to approve a baseline version of the CM process, as well as any changes to the process.
* Has the authority to define the baseline for each software iteration, and to approve the acceptance of completed changes. The CCB also has the authority to approve reversion to a previous version of the product if warranted.

## **Relationship of CM to the software process life cycle**

Configuration Management ensures the effective control of software artifacts at each stage of the development process. During the analysis phase, CM is involved in identifying and modifying initial requirements. During the design/development phase, CM is critical to version control. Developers will work in a separate dev environment where changes to design, software, and documentation will be tracked from. The testing process follows a similar path, changes to unit tests will be monitored. For the deployment and maintenance phase, CM guarantees that the proper versions are delivered. User manuals, bug fixes, and updates are all required to follow the CM process.

## **SCM Resources**

The resources being employed to monitor changes and version control are as follows:

* Github - Version Control for code and and other software files
* Google Drive - Version Control for documentation, and documentation collaboration
* Jira - Team members must all allocate time for CM activities. This is tracked in Jira

# **SCM ACTIVITIES**

## Configuration Identification

Configuration Identification is an integral part of the Software Configuration Management process that involves identifying and defining the items within the software configuration. All ATS Modules and baselines are subject to approval by the CCB. Once a module has been baselined all changes made will be identified in their respective Version Description Documents (VDDs). The ATS is currently in development. The product baseline is to be approved upon delivery. Product baselines are to be maintained by the ATS CM. The Baseline will consist of the following:

* source files
* configuration files
* data files
* VDDs
* User Manual [r4]

### Specification Identification

Specification Identification is a CM activity that consists of identifying, naming, and describing all code, design elements, and data artifacts that will be controlled within the project. Items will include source code, documentation, databases, unit tests, requirements specification, management plans, and environmental elements. The project will define and document these items to ensure they’re managed.

### **Change Control Form Identification**

As a minimum, the information recorded for a proposed change must contain the following:

* The name(s) and version(s) of the CIs where the problem appears
* Originators name and organization
* Date of request
* Priority
* Reason for change request
* Description of the requested change

Additional information, such as priority or classification, may be included to assist in its analysis and evaluation. Other information, such as change request number, status, and disposition, may be recorded for change tracking. An example of the type of information provided in this section follows:

Software Change Request (SCR) procedures for each development team are defined in Appendix A. These procedures are used to add to, change, or remove items from the baselines. The identification and tracking of change requests is accomplished through Change/Enhancement Requests (CER).

### **Project Baselines**

Project baseline defines the grouping of Configuration Items (CI) as a fixed reference point to a specific point in the project lifecycle. The baseline ensures that changes to CIs are controlled effectively and follow prior conventions. This project uses a GitHub repository with various branches to manage this process. The “main” branch acts as a baseline for the project. The “development” branch is the layer before this, and is where updated work from each sprint is uploaded. The change control board reviews the development branch and approves or disapproves a release. If it is not approved, the branch can be reverted to baseline. If the development branch has been approved for release, it is pushed to the main branch and becomes the new baseline. Branches under the main (including but not limited to the development branch) are then updated to the latest baseline.

## **Configuration Control**

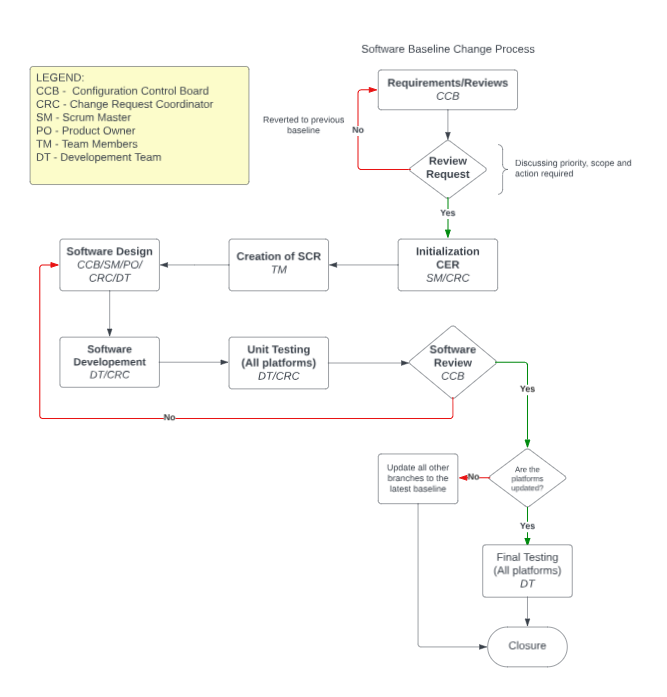


Figure 1: Change Request Process Activity Diagram

### **Procedures for changing baseline**

Changes to each software product baseline are made according to the change request process (3.2.2).

Change Tracking. Changes to each software product baseline are tracked using Github and its built-in version history. Google drive will be used to store documents relating to project changes.

Change Release. The product owner provides reports containing the following information for review at each CCB meeting for:

a) Number of new CERs

b) Type

c) Priority

d) Source (Internal, Field, FOT&E)

e) Action Required

f) Number of CERs in each status

Each development team member is responsible for keeping the information in the Github up to date so the information is available to the PO and SM prior to CCB meetings.

### **Procedures for processing change requests and approvals**

When a Change/Enhancement request is made, the Scrum Master and Product Owner will begin a discussion with the project team in a CCB meeting. During this meeting, the team will discuss the feasibility, priority, scope, and action required to execute the change. The team will then take a vote to decide if the change will be pushed through or not. Once approved, the changes are properly documented.

### **Change Control Boards (CCBs)**

The Change Control Board (CCB) is responsible for evaluating all proposed changes, and assuring that changes maintain or improve the quality of the project. Firstly, the board will assess how the change will impact project schedule, quality, scope, and risks. Changes are then ordered based on their priority. The CCB takes a vote to approve or reject the proposals. If changes are approved, they are documented properly.

### **Interfaces**

### IONOS Integrations

IONOS Integration refers to the interaction between our system and the IONOS cloud hosting service. IONOS serves as our chosen hosting provider for the following aspects of our system:

1. **Web Hosting**: IONOS hosts our web applications, ensuring their availability and performance for end-users.
2. **Database Hosting**: IONOS provides the infrastructure for hosting our system's databases, ensuring data storage and retrieval capabilities.
3. **Backup and Recovery**: Our system uses IONOS services for backup and disaster recovery to maintain data integrity and availability.

Compatibility: Our system is designed to work seamlessly with IONOS' hosting services. It is configured to meet the technical requirements and specifications outlined by IONOS for optimal performance.

Security Considerations: Security measures, including data encryption, firewall rules, and access controls, are implemented to safeguard the data and communication channels between our system and IONOS' infrastructure.

Data Transfer: Data transfer between our system and IONOS' servers occurs using secure protocols and formats compliant with industry best practices.

Dependence: Our system relies on IONOS for its hosting needs, making IONOS a critical external entity. The availability, reliability, and security of IONOS' services directly impact our system's performance and availability*.*

### **Level of control**

Changes to existing baselines must be evaluated by the CCB. Approval and disapproval of changes are strictly reserved for the CCB. Approved changes are pushed to the product baseline by the development team.

### **Document revisions**

If changes to this Configuration Management plan are proposed, they must be submitted to the Google drive in a formal request. These changes must be approved by the Product Owner and reviewed by the project team prior to being implemented.

## **Configuration Status Accounting**

Configuration Status Accounting (CSA) involves the recording and reporting of the status of a Configuration Item. The CM plan outlines the following:

* What will be tracked and reported
  + The initial approved version of the item up for change
  + The current status of request changes
  + The implementation of approved changes
* Types of reports and reporting frequency
  + Baseline reports
  + Change request reports
  + Implementation status reports
* How the information is processed and controlled
  + Data collection methods
  + Data storage methods
  + Data control (consistency reviews)

### **Storage, handling and release of project media**

The initial approved version of the ATS is maintained by the PO until the approval of a subsequent version of the product. Upon approval, the initial version is held in the project CM library. This library is under configuration control.

### **Release proces**s

An acceptance test will be created by the team, and modified as necessary by clients. Prior to a product release, the client will be presented with the acceptance test and sign it if they approve of the current iteration of the project.

### **Document Status Accounting**

All Configuration Management changes to the project will be managed through Github and documented through Google drive.

### **Change Management Status Accounting**

The process for CM will utilize Github and Google Drive as tools to store change requests.

GitHub serves as the primary platform for managing software changes. It describes the status of code changes, pull requests, merge requests, and other version control features. Status reports on Github are updated frequently throughout the project as changes occur. This will guarantee that the development team always has access to the most current status information.

Google Drive will work in tandem with Github by acting as a place for documenting and tracking Change Management beyond just code change. Google Drive will be where detailed reports on the CM process will be uploaded. Reports will be updated on Google Drive at intervals during the development cycles.

## **Configuration Audits and Reviews**

The ATS product owners and scrum masters review the software baseline monthly. This baseline review consists of checking for CER implementation. The CERs are selected based on priority. Discrepancies will be annotated and reported to the PO. Corrective action is taken if required.

The ATS CCB reviews the CM process semi-annually. The most important problem reports are identified and walked through the change process, discrepancies are annotated and reported to the Product Owner and corrective action is taken if required. A software baseline, status reporting, and documentation for each software product are available for quality assurance review at any time.

# **CM Milestones**

## Configuration Management Plan Approval

The ATS Configuration Management Plan (CMP) is developed to be inline with the project’s unique set of requirements. Project stakeholders review the CMP to ensure it is inline with the project's needs. They then approve the CMP following review.

## Baseline Establishment

Baseline configuration items (CIs), including source code, design documents, and relevant specifications, are all identified and documented. The baseline provides a foundation to develop from, and serves as a reference point for all future changes.

## Change Management Implementation

The ATS change control process is fully implemented. This includes outlining the usage of version control software. GitHub and Google Drive are used for tracking, documenting, and managing changes to project-related documents and files.

## Release Management

The final approved project configuration is released for deployment to clients and stakeholders. This process involves the packaging, testing, and releasing of software. Release management ensures that software is delivered on schedule and that stakeholders receive satisfactory artifacts.

## Project Closure Review

Upon the closure of project development, the CM process is completed in order to assess its effectiveness. Any areas of improvement will be discussed and documented.

# **Training**

Development team members who are involved in configuration management activities will receive adequate training on the configuration management procedures utilized in this project. Training covers baseline management, change management, and version control practices. Configuration management training is conducted at the beginning of the project lifecycle, and refreshed as processes evolve.

These resources include the configuration management plan, GitHub documentation [[r5](https://docs.github.com/en/get-started/quickstart/git-and-github-learning-resources)], and the [user manual](https://docs.google.com/document/u/0/d/1MSl4BucQyQG3XGQqiSBJuw8ncuh2JzPF/edit).

Trainees will all be assessed to which level of understanding they have of each technology we require them to become proficient in. Should it be necessary they will be trained in each of these technologies:

* [Jira](https://www.atlassian.com/agile/tutorials)
* [Github](https://docs.github.com/en/get-started/quickstart/git-and-github-learning-resources)
* IONOS
* PhpMyAdmin
* MySQL
* Python
* HTML
* Javascript
* CSS

The effectiveness of training will be assessed by listening to feedback from team members, and closely monitoring Github changelogs.

# Subcontractor/Vendor Support

ATS subcontractor support will be identified by the ATS Product Owner if required. Subcontractor support will be required to follow the process defined in this document. If a waiver is requested, the subcontractor must provide evidence of comparable configuration management procedures. These procedures will follow the same audit and control procedures described in this document.

# 

# APPENDIX A // Software Change Request Procedures

A change to ATS software may be requested by one of the following: Internal Test Report (ITR), Software Problem Report (SPR), or Baseline Change Request (BCR) from external or internal sources or from the product specification/requirements. A change to ATS documentation may be requested by a Document Change Request (DCR). All of the above requested change procedures are referred to as a change request in the text below.

The Change Control Board (CCB) for each development team determines which of the change requests is required for each software release prior to the start of work for that release. Additional change requests are reviewed by the Algorithmic Trading System CCB to determine and assign the proper status to the change requests; these are held for CCB scheduling. Status is one of the following:

**Open** - Change request is to be implemented for current software version.

**Hold** - Change request targeted for another software version.

**Void** - Change request is a duplicate of an existing CER or

does not apply to existing software.

**Working** - Change is currently being implemented.

**Testing** - Implemented change request is under Developmental Test & Evaluation (DT&E)

**Fixed** - Change request is implemented, unit and integration test complete.

The following steps define the procedure for each status. Each procedure starts with the change request being given to the Project Lead or designated Change Request Coordinator (CRC).

**A. Open**

1. INITIALIZATION

a) The CRC for the associated ATS project creates a CER using the CER system.

b) The CRC notifies the Project Lead of the new CER.

2. CREATE SCR

a) The Project Lead reviews the CER, and provides estimates of schedule impact to CCB.

b) CCB prioritizes CER and authorizes implementation.

c) The Project Lead assigns the CER to a Software Engineer (SE) to incorporate.

d) The SE creates a Software Change Request (SCR).

3. SOFTWARE DESIGN

The SE designs changes necessary to implement CER:

a) The SE identifies interface changes and consults with appropriate PLs.

b) Determines changes needed.

c) The SE identifies any changes needed for documentation.

d) The SE estimates work effort required for CER completion.

4. SOFTWARE DEVELOPMENT/MODIFICATION

b) The SE makes needed changes and provides updated documentation for testing.

c) The SE checks out the files from the configuration managed version (not the baselined version) of all the files needed to incorporate the CER.

NOTE: ATS utilizes GitHub for configuration management. GitHub keeps track of the product release versions of each source file.

d) The SE fills in the CM sections of the SCR.

e) The SE puts the changes in the configuration managed files.

5. UNIT TESTING

a) The SE runs the unit tests for the CER software:

1) If the test fails proceed to Step 3, Software Design.

6. SOFTWARE REVIEW

a) The SE prepares a software review package:

1) Hard-copy of SCR.

2) List of documentation changes.

b) Software Review

1) SE assigns software review packages to other team members for review.

2) If SCR is peer approved, proceed to step 7.

3) If SCR is not approved proceed to step 3, Software Design.

7. INCORPORATE APPROVED SCR INTO BASELINE CM

a) The PO changes the SCR status to indicate that it is approved.

b) The SE checks in the files.

c) The SE fills in the CM sections of the SCR.

f) The CRC reviews SCR for completeness.

g) The CRC changes the status on the CER to indicate testing.

h) The designated SCM gives a list of changed files to Product Lead for updating their working directories/executables.

i) The designated SCM installs changes to baseline on all platforms.

8. FINAL TESTING

a) The CRC identifies all CERs with testing status.

b) Final testing by DT&E validation group (If problems are found start over at step 2a)

9. CLOSURE

a) The CRC changes final status on CER to indicate completeness.

**B. Hold**

The CRC creates a CER using the CER system marking status hold and indicates the version associated with the CER. The CRC files the original change request for the next version. The CER will be considered for inclusion in the next version.

**C. Void**

1. Duplicates:

a) The CRC marks the change requests as a duplicate, specifying which CER is a duplicate.

b) The CRC files the change request for the associated version.

2. Other:

a) The CRC marks the reason for cancellation.

b) The CRC files the change request for the associated version.

**D. Testing**

The CRC forwards CER/SCR to Project Lead for review. After review the Project Lead forwards CER to DT&E test for validation. CER status is updated to indicate testing status.

**E. Fixed**

The CRC files the closed change request and updates the database.