NOAA NESDIS
CENTER for SATELLITE APPLICATIONS and RESEARCH (STAR)

TASK GUIDELINE

TG-10
CODE TEST AND REFINEMENT (STEP 10) TASK GUIDELINES

Version 3.0

TITLE: TG-10: CODE TEST AND REFINEMENT (STEP 10) TASK GUIDELINE VERSION 3.0

AUTHORS:

Ken Jensen (Raytheon Information Solutions)

VERSION HISTORY SUMMARY

Version	Description	Revised Sections	Date
1.0	No version 1		
2.0	No version 2		
3.0	New Task Guideline adapted from CMMI guidelines by Ken Jensen (Raytheon Information Solutions)	New Document	10/01/2009

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

ATBD	Algorithm Theoretical Basis Document
BB	Baseline Build
CDD	Critical Design Document
CDR	Critical Design Review
CDRR	Critical Design Review Report
CI	Cooperative Institute
CICS	Cooperative Institute for Climate Studies
CIMSS	Cooperative Institute for Meteorological Satellite Studies
CIOSS	Cooperative Institute for Oceanographic Satellite Studies
CIRA	Cooperative Institute for Research in the Atmosphere
CL	Check List
CLI	Check List Item
CoRP	Cooperative Research Program
CM	Configuration Management
CMMI	Capability Maturity Model Integration
CREST	Cooperative Remote Sensing and Technology Center
CTD	Code Test Document
CTR	Code Test Review
CTRR	Code Test Review Report
DDD	Detailed Design Document
DG	Document Guidelines
DM	Data Management
DPP	Development Project Plan
DPR	Development Project Report
EPG	Enterprise Process Group
EPL	Enterprise Product Lifecycle
IPT	Integrated Product Team
NESDIS	National Environmental Satellite, Data, and Information Service
NOAA	National Oceanic and Atmospheric Administration
OCD	Operations Concept Document

PAR	Process Asset Repository
PBR	Project Baseline Report
PDD	Preliminary Design Document
PDR	Preliminary Design Review
PDRR	Preliminary Design Review Report
PG	Process Guidelines
PP	Project Proposal
PRD	Project Requirements Document
PRG	Peer Review Guidelines
PRR	Project Requirements Review
PRRR	Project Requirements Review Report
PSR	Project Status Report
QA	Quality Assurance
R&D	Research & Development
RAD	Requirements Allocation Document
RAS	Requirements Allocation Sheet
RNM	Requirements/Needs Matrix
SG	Stakeholder Guideline
SPSRB	Satellite Products and Services Review Board
SRR	System Readiness Review
STAR	Center for Satellite Applications and Research
STP	System Test Plan
SWA	Software Architecture Document
TD	Training Document
TG	Task Guideline
TRD	Test Readiness Document
TRR	Test Readiness Review
TRRR	Test Readiness Review Report
UTP	Unit Test Plan
UTR	Unit Test Report
VVP	Verification and Validation Plan

1. INTRODUCTION

The NOAA/NESDIS Center for Satellite Applications and Research (STAR) develops a diverse spectrum of complex, often interrelated, environmental algorithms and software systems. These systems are developed through extensive research programs, and transitioned from research to operations when a sufficient level of maturity and end-user acceptance is achieved. Progress is often iterative, with subsequent deliveries providing additional robustness and functionality. Development and deployment is distributed, involving STAR, the Cooperative Institutes (CICS[1], CIMSS[2], CIOSS[3], CIRA[4], CREST[5]) distributed throughout the US, multiple support contractors, and NESDIS Operations.

NESDIS/STAR is implementing an increased level of process maturity to support the development of these software systems from research to operations. This document is a Task Guideline (TG) for users of this process, which has been designated as the STAR Enterprise Product Lifecycle (EPL).

1.1. Objective

The STAR EPL is designed as a sequence of 11 process steps that take a product from initial conception through delivery to operations. These steps are:

- Step 1 - Basic Research (TG-1)
- Step 2 - Focused R & D (TG-2)
- Step 3 - Project Proposal (TG-3)
- Step 4 - Resource Identification (TG-4)
- Step 5 - Development Project Plan (TG-5)
- Step 6 - Project Requirements (TG-6)
- Step 7 - Preliminary Design (TG-7)
- Step 8 - Detailed Design (TG-8)

[1] Cooperative Institute for Climate Studies

[2] Cooperative Institute for Meteorological Satellite Studies

[3] Cooperative Institute for Oceanographic Satellite Studies

[4] Cooperative Institute for Research in the Atmosphere

[5] Cooperative Remote Sensing and Technology Center

- Step 9 - Code & Test Data Development (TG-9)
- **Step 10 - Code Test And Refinement (TG-10)**
- Step 11 - System Integration and Test (TG-11)

The objective of this Task Guideline (TG-10) is to describe how to perform the standard tasks of STAR EPL process step 10, “Code Test and Refinement”.

The intended users of this TG are all participants in the STAR EPL process who are involved in performing the standard tasks of step 10. Participants are referred to as STAR EPL stakeholders.

To determine whether or not they should be involved with this step, the readers of this TG should first determine what stakeholder roles apply to their participation in a STAR research-to-operations development project. Generic stakeholder roles are listed in Section 3 of this TG and discussed in Section 3.2 of the EPL Process Guideline (PG-1)[6]. PG-1 and this TG will direct stakeholders to Stakeholder Guidelines (SG) that are pertinent to their roles.

1.2. Version History

This is the first version of TG-10. It is identified as version 3.0 to align it with the release of the version 3.0 STAR EPL process assets.

1.3. Overview

This TG contains the following sections:

Section 1.0 - Introduction
Section 2.0 - References
Section 3.0 - Stakeholders
Section 4.0 - Code Test Review
Section 5.0 - Project Artifacts
Section 6.0 - Task Descriptions

[6] It is recommended that potential STAR EPL stakeholders either review PG-1 prior to using this TG or use it as a reference while using this TG.

2. REFERENCE DOCUMENTS

All of the reference documents for the STAR EPL process are STAR EPL process assets that are accessible in a Process Asset Repository (PAR) on the STAR website. http://www.star.nesdis.noaa.gov/star/EPL_index.php.

Process assets include:

- Process Guidelines
- Stakeholder Guidelines
- Task Guidelines
- Peer Review Guidelines
- Review Check Lists
- Document Guidelines
- Training Documents

2.1. Process Guidelines

Process Guideline (PG) documents describe STAR's standard set of practices and guidelines for tailoring them to specific projects.

- STAR EPL Process Guidelines (PG-1)
- STAR EPL Process Guidelines Appendix (PG-1.A)
- STAR EPL Tailoring Guidelines (PG-2)

PG-1 and PG-1.A apply generally to each EPL step. Each stakeholder performing tasks during each step can benefit from a familiarity with these documents.

PG-2 is primarily useful for project planners and project plan reviewers during steps 4 and 5. It is also useful during steps 6-11 for project plan revision tasks.

2.2. Stakeholder Guidelines

A Stakeholder Guideline (SG) is a description of how to perform all STAR EPL standard tasks assigned to a given type of stakeholder. It should itemize the actions to be taken. It should contain appropriate standards, conventions, and (where appropriate) examples. It should point to the appropriate references and the required artifacts.

Stakeholder roles are identified in Section 3 of this TG. For each type of stakeholder, the appropriate SG provides that stakeholder with a complete description of the standard tasks for that stakeholder role, along with references to all appropriate process assets and project artifacts (c.f. Section 5 of this TG). This functions as a complement to the TGs (c.f. Section 2.3 of this TG), which provide a completion description of all stakeholder tasks for a specific process step.

Table 2.2.1 lists the Stakeholder Guidelines that are relevant to this step.

TABLE 2.2.1 – Stakeholder Guidelines for Step 10

ID	Stakeholder
SG-4	STAR CM/DM
SG-5	STAR Web Developers
SG-6	STAR Quality Assurance
SG-13	Development Leads
SG-14	Development Scientists
SG-15	Development Testers
SG-16	Development Programmers
SG-17	Technical Review Leads
SG-18	Technical Reviewers

2.3. Task Guidelines

A Task Guideline (TG) is a description of how to perform the tasks of a STAR EPL process step. It should itemize the actions to be taken. It should contain appropriate standards, conventions, and (where appropriate) examples. It should point to the appropriate references and the required artifacts. There is one Task Guideline for each step in the STAR EPL. The relevant TG for this step is TG-10 (this document).

2.4. Peer Review Guidelines

For each review (c.f. Section 4), there is a Peer Review Guideline (PRG) that describes the objectives of the review, the required artifacts, standards for reviewers, requirements for approval, and options other than approval. For step 10, the relevant PRGs include:

- Code Test Review Guidelines (PRG-10)

2.5. Review Check Lists

For each review (c.f. Section 4), there is a Review Check List (CL) that captures all the objectives for a review as a set of check list items. Each item in the check list should have a "Disposition" column that contains "Pass", "Conditional Pass", "Defer", "Waive", or "N/A" (Not Applicable). Each item will also have columns for Risk Assessment and for Actions generated. For step 10, the relevant CLs include:

- Code Test Review Check List (CL-10)

2.6. Document Guidelines

There is a Document Guideline (DG) for each standard STAR EPL document. Each DG includes a description of the purpose for the document, a standard document outline (table of contents), a brief description of each subsection in the outline, and an Appendix containing an example document.

Table 2.6.1 lists the Document Guidelines that are relevant to this step.

TABLE 2.6.1 – Document Guidelines for Step 10

ID	Document
DG-1.2	Software Architecture Document (SWA)
DG-5.1	Development Project Plan (DPP)

DG-5.4	Project Baseline Report (PBR)
DG-5.5	Gate 3 Review Report (G3RR)
DG-6.2	Requirements Allocation Document (RAD)
DG-6.3	Verification and Validation Plan (VVP)
DG-9.1	Unit Test Plan (UTP)
DG-10.1	Unit Test Report (UTR)
DG-10.2	System Test Plan (STP)
DG-10.3	Code Test Document (CTD)
DG-10.3.A	CTD Appendix
DG-10.4	Code Test Review Report (CTRR)

2.7. Training Documents

Training Documents (TD) assist the stakeholders (c.f. Section 3) in performing the process tasks. By using the TDs, the stakeholders should be able to perform the tasks more effectively.

Table 2.7.1 lists the Training Documents that are relevant to this step.

TABLE 2.7.1 – Training Documents for Step 10

ID	**Training Document**
TD-11.1	FORTRAN Coding Standards
TD-11.1.A	Transition from Fortran 77 to Fortran 90
TD-11.2	C Coding Standards

3. STAKEHOLDERS

The STAR Enterprise is comprised of a large number of organizations that participate and cooperate in the development and production of environmental satellite data products and services. Individual project teams are customarily composed of personnel from these organizations, supplemented by contractor personnel. These organizations and project teams are referred to as the STAR Enterprise stakeholders.

An overview of the stakeholder roles is provided in the STAR EPL Process Guidelines (PG-1, c.f. Section 2). A more detailed description can be found in the Stakeholder Guidelines (SGs, c.f. Section 2).

Stakeholders who have a role during step 10 include:

- STAR CM/DM (SG-4)
- STAR Web Developer (SG-5)
- STAR QA (SG-6)
- Development Lead (SG-13)
- Development Scientist (SG-14)
- Development Tester (SG-15)
- Development Programmer (SG-16)
- Technical Review Lead (SG-17)
- Technical Reviewer (SG-18)

STAR CM/DM is the Configuration Management (CM) and Data Management (DM) group for the STAR organization. CM/DM is responsible for establishing and maintaining project baselines for code, test data, documentation, and reports. CM/DM works with each Development Lead to ensure that project artifacts are maintained in accordance with STAR standards. CM/DM works with Operations CM/DM on the transition of the project baseline from pre-operational development to operations.

STAR Web Developer is responsible for maintenance of the STAR web pages. The Web Developer works with STAR CM/DM to ensure that all project baseline items are posted to the appropriate project artifact repository in a timely fashion. The Web Developer works with the STAR EPG and STAR CM/DM to ensure that all STAR EPL process assets are posted to the PAR, and to ensure that all process measures are posted to the STAR Measurement Repository.

STAR QA is the quality assurance (QA) group for the STAR organization. QA is responsible for ensuring that each project's tailored process meets STAR EPL process standards and ensuring that each project meets its process requirements during its pre-operational development phases. QA works with the STAR EPG to ensure effective implementation of the process throughout the organization.

Development Lead is nominally a STAR scientist who leads a project’s development efforts after a Project Proposal (PP) has been approved. The Development Lead is typically identified in the PP and is often the same person who was the Research Lead. The Development Lead works with STAR Management to tailor the STAR EPL process to the project and leads the project's development efforts during the Design and Build phases as the lead of the Integrated Product Team (IPT).

Development Scientist is nominally a STAR scientist who has been assigned by the Development Lead to one or more of the tasks of reviewing the technical content of project proposals, maturing a research algorithm into an operational algorithm, developing project requirements, supporting product design, coding and testing, and providing product validation and science maintenance.

Development Tester is any person located at a research organization who has been assigned by the Development Lead to one or more of the tasks of identifying pre-operational test data, acquiring and integrating the test data into the pre-operational product processing system, creating pre-operational unit and system test plans, executing unit and system tests, and analyzing and reporting test results for review.

Development Programmer is a programmer who has been assigned by the Development Lead to one or more of the tasks of preliminary design and detailed design of pre-operational code, writing pre-operational code, integrating code into a pre-operational system, and supporting Development Testers in testing pre-operational code.

Technical Review Lead is responsible for leading the team of Technical Reviewers for one or more of the six Technical Reviews. The Technical Review Lead works with the Development Lead and the Technical Reviewers to ensure that the review is prepared for, conducted, and closed according to review standards.

Technical Reviewer is responsible for reviewing and approving project artifacts and project status at one or more of the six Technical Reviews. Technical Reviewers work with the Technical Review Lead to ensure that the review is prepared for, conducted, and closed according to review standards.

Stakeholder satisfaction is a critical component of the process. The intention is for the process to be more of a benefit that a burden to stakeholders. If stakeholders are not satisfied that this is the case, the process will require improvement.

Stakeholders are strongly encouraged to provide feedback to the EPG. Comments and suggestions for improvement of the process architecture, assets, artifacts and tools are always welcome. Stakeholders can provide feedback by contacting:

Ken.Jensen@noaa.gov

4. CODE TEST REVIEW

Code Test Review (CTR) is a Build Phase Technical Review. Its purpose is to determine whether the pre-operational software units are ready for integration unto a pre-operational system. Upon successful completion of this review, step 11 (System Integration and Test) commences.

Standard CTR objectives:

- Identify relevant stakeholders and document their involvement according to the project plan.
- Provide all applicable technical data, including:
    - Refined pre-operational code and test data
    - Unit test plan
    - Unit test report
    - System test plan
- Review the unit test plan, focusing on changes since the TRR.
- Review the unit test results
- Review the system test plan
- Identify and evaluate risks. Recommend risk mitigation activities.
- Document the closing of all action items since TRR. Make recommendations for open actions and new actions.

Standard CTR entry criteria:

- Entry # 1 - A Test Readiness Report (TRRR) has been written. The CTR reviewers have access to the current baseline version of the TRRR.
- Entry # 2 -A Development Project Plan (DPP) has been written. The CTR reviewers have access to the current baseline version of the DPP.
- Entry # 3 - A Requirements Allocation Document (RAD) has been written. The CTR reviewers have access to the current baseline version of the RAD.

- Entry # 4 - A Software Architecture Document (SWA) has been written. The CTR reviewers have access to the current baseline version of the SWA.
- Entry # 5 - Detailed Design Documents (DDDs) have been written for each software unit in the software architecture. The CTR reviewers have access to the current baseline version of each DDD.
- Entry # 6 – A Unit Test Plan (UTP) has been written. The CTR reviewers have access to the current baseline version of the UTP.
- Entry # 7 – Pre-operational code units, external interfaces, ancillary data, unit test data and unit test results are in the development test environment. The CTR reviewers have access to this code, test data and test results.
- Entry # 8 – A Unit Test Report (UTR) has been written. The CTR reviewers have access to the current baseline version of the UTR.
- Entry # 9 - A Verification and Validation Plan (VVP) has been written. The CTR reviewers have access to the current baseline version of the VVP.
- Entry # 10 - A System Test Plan (STP) has been written. The CTR reviewers have access to the current baseline version of the STP.
- Entry # 11 - A Project Baseline Report (PBR) has been written. The CTR reviewers have access to the current baseline version of the PBR.
- Entry # 12 - A Code Test Document (CTD) has been written. The CTR reviewers have access to the current baseline version of the CTD.

Standard CTR exit criteria:

- Exit # 1 - TRR "Conditional Pass" items have been satisfactorily disposed of.
- Exit # 2 - TRR "Defer" items have been satisfactorily disposed of.
- Exit # 3 – Changes to the project plan since TRR are approved.
- Exit # 4 - Requirements allocation changes since TRR are approved.
- Exit # 5 - Changes to external interfaces since TRR are approved.
- Exit # 6 - Changes to the software architecture since TRR are approved.
- Exit # 7 - Changes to the detailed design since TRR are approved.
- Exit # 8 - Changes to the verification and validation plan since TRR are approved.
- Exit # 9 – Code units and unit test data are satisfactory

- Exit # 10 – Unit test results and UTR are satisfactory
- Exit # 11 - The system test plan and STP are satisfactory
- Exit # 12 - The project baseline and PBR are satisfactory.
- Exit # 13 - The CTRR documents updated status of project risks and actions.
- Exit # 14 - Project risks and actions are acceptable. The project is ready for system integration and system testing.

Refer to PRG-10 for a more detailed description of the CTR. The standard CTR entry criteria, exit criteria, and check list is documented in the process asset CL-10 (c.f. Section 2).

CTR objectives, entry criteria, exit criteria, and check list may be tailored. Tailoring guidelines are provided in the process asset PG-2 (c.f. Section 2). Refer to the Development Project Plan (DPP) Section 5 to determine whether there has been any project-specific tailoring for the CTR.

5. PROJECT ARTIFACTS

Project Artifacts are a set of items that must be produced by the appropriate stakeholders during the product life cycle to support the reviews. They are established and maintained under Configuration Management (CM) by an Enterprise Process Group (EPG) under the direction of a Steering Committee.

The project artifacts are maintained in a project artifact repository. This is a complete set of configuration-managed artifacts developed by each project in accordance with STAR standards. When a project artifact has been approved at a Technical Review or Gate Review, it is placed in the project artifact repository under CM.

Project artifacts that are recommended for development during step 10 are listed in Table 5.1.

TABLE 5.1 – Step 10 Artifacts

Artifact	Type	Review	Baseline Build
Development Project Plan v3.x	Document	CTR	3.2
Refined Pre-Operational Code	Code	CTR	3.2
Refined Pre-Operational Test Data	Test Data	CTR	3.2
Software Architecture Document v2.3	Document	CTR, SRR	3.2
Detailed Design Document v1.2	Document	CTR, SRR	3.2
Requirements Allocation Document v1.4	Document	CTR, SRR	3.2
Unit Test Plan v1.x	Document	CTR	3.2
Unit Test Report v1.0	Report	CTR	3.2
System Test Plan v1.0	Document	CTR	3.2
Verification and Validation Plan v1.4	Document	CTR, SRR	3.2
Code Test Document	Presentation	CTR	3.2
Project Baseline Report v3.2	Report	CTR	3.2
Code Test Review Report	Report	SRR	3.3
Project Baseline Report v3.3	Report	None	3.3

Development Project Plan v3.x: The Development Project Plan (DPP) documents the plan for the development, testing, review, and transition to operations for the project, including stakeholders, tasks, work breakdown structure (WBS), schedule and resources. DPP v3.0 should have been produced in step 9 for the TRR, and may have been revised (v3.x) as a result of TRR actions. Refer to DG-5.1 for detailed DPP guidelines.

Refined Pre-Operational Code: The Refined Pre-Operational Code (PCOD v2.x) consists of all software components of the detailed design that was approved at the CDR (step 8) and unit tested in step 10. The code is a post-unit test refinement of the Pre-Operational Code, refined to correct bugs and other deficiencies that are revealed by unit testing. It is expected that SPSRB coding standards will be applied to the pre-operational code. Currently, coding standards exist for Fortran, C, and C++ code, and general programming standards exist for all code. These standards are found on the SPSRB web site at http://projects.osd.noaa.gov/spsrb/standards_prog.htm. This requirement may be waived if the circumstances of a specific project provide a compelling reason for a waiver. Waivers should be agreed to as early as possible, included in the project plan, and accepted by operations prior to unit testing.

Refined Pre-Operational Test Data: Refined Pre-Operational Test Data (PTEST v2.x) are the data files used for unit testing of the Pre-Operational Code, including the input data and output data identified in the current baseline versions of the Algorithm Theoretical Basis Document (ATBD) and Software Architecture Document (SWA). These files may be revised and/or upgraded during unit testing, to apply to the refined code.

Software Architecture Document v2.3: The SWA provides the software architecture for the processing code that will implement the algorithm. SWA v2.2, produced for the TRR, should have described the software architecture as it existed prior to unit testing. Often, the SWA must be revised (v2.3) to document revisions to the software architecture that occur as a result of unit testing. Refer to DG-1.2 for detailed SWA guidelines.

Detailed Design Document v1.2: The purpose of the Detailed Design Document (DDD) is to describe the product design at a level of detail that is sufficient for the development programmers to write fully functional pre-operational code. A separate DDD is produced for each software unit that is part of the product processing system. The software units are the Layer-2 elements that are defined in the system layer product software architecture, as described in the SWA. DDD v1.1, produced for the TRR, should have described the detailed design as it existed prior to unit testing. Often, the DDD must be revised (v1.2) to document revisions to the detailed design that occur as a result of unit testing. Refer to DG-8.1 for detailed DDD guidelines.

Requirements Allocation Document v1.4: The Requirements Allocation Document (RAD) contains the basic and derived requirements for the work products and the allocation of the requirements to system components and product components. RAD v1.3, produced for the TRR, should have documented the requirements and requirements allocation prior to unit testing. Often, the RAD must be revised (v1.4) to document revisions to the requirements allocation that are needed to reflect detailed design changes that occur as a result of unit testing. Refer to DG-6.2 for detailed RAD guidelines.

Unit Test Plan v1.x: The Unit Test Plan (UTP) contains the test plan for each software unit in the project's product processing system. The UTP, a complement to the project's Verification and Validation Plan (VVP), focuses on the specifics of the software units and the testing of their functionality and performance. UTP v1.0 should have been produced in step 9 for the Test Readiness Review (TRR). Circumstances may occur during unit testing that result in a decision to revise the plan. In that case, the UTP will be updated to v1.x. Refer to DG-9.1 for detailed UTP guidelines.

Unit Test Report v1.0: The Unit Test Report (UTR) documents the results of testing of each software unit to verify that the requirements allocated to the unit's software components are satisfied. The UTR should describe the results of each unit test in a way that demonstrates the verification of the requirements allocated to components of the software unit, show how the results demonstrate that the requirements allocated to the software units are satisfied, and note any requirements allocations whose verification is incomplete or questionable. Refer to DG-10.1 for detailed UTR guidelines.

System Test Plan v1.0: The System Test Plan (STP) contains the plan for testing to ensure that the requirements specified for the product processing system are satisfied by the completed system (Verification) and that the final developed system will satisfy the users' needs and expectations (Validation). The purpose of the system test is to demonstrate, using verification and validation methods, system readiness for operations. Refer to DG-10.2 for detailed STP guidelines.

Verification and Validation Plan v1.4: The Verification and Validation Plan (VVP) describes the work products to be verified and validated, the requirements for each selected work product and the verification and validation methods for each selected work product. VVP v1.4, produced for the CTR, includes any revisions needed to maintain consistency with the STP and the Requirements Allocation Document (RAD). Refer to DG-6.3 for detailed VVP guidelines.

Code Test Document: The Code Test Document (CTD) consists of the presentation slides for the Code Test Review (CTR). Refer to DG-10.3 and DG-10.3.A for detailed CTD guidelines.

Project Baseline Report v3.2: The Project Baseline Report (PBR v3.2) is the document that describes the status of the configuration items that comprise the project baseline at the CTR. Refer to DG-5.4 for detailed PBR guidelines.

Note that these artifacts are typically included in STAR Baseline Build (BB) 3.2. BB 3.2 provides the artifacts for the CTR. **STAR CM/DM** executes BB 3.2, in consultation with the developers of the BB 3.2 artifacts.

Code Test Review Report: The CTR Report (CTRR) summarizes the CTR Reviewers' assessment of the unit test results, including identified risks and risk mitigation actions. Refer to DG-10.4 for detailed CTRR guidelines.

Project Baseline Report v3.3: When the CTRR is completed, it is added to the baseline for BB 3.3. The PBR is updated to v3.3 to include the addition of the CTRR as well as any CTR artifacts that are revised as the result of CTR actions.

6. TASK DESCRIPTION

6.1 Pre-Operational System Development Process

Pre-operational system development is an iterative process that occurs throughout the Build phase of the product lifecycle. This phase includes three steps that produce an integrated product processing system through an iterative (spiral) development of code, test data and test plans.

- Code & Test Data Development (step 9 of the STAR EPL)
- **Code Test & Refinement (step 10 of the STAR EPL)**
- System Integration & Test (step 11 of the STAR EPL)

Figure 6.1 illustrates the pre-operational system development process, with step 10 highlighted.

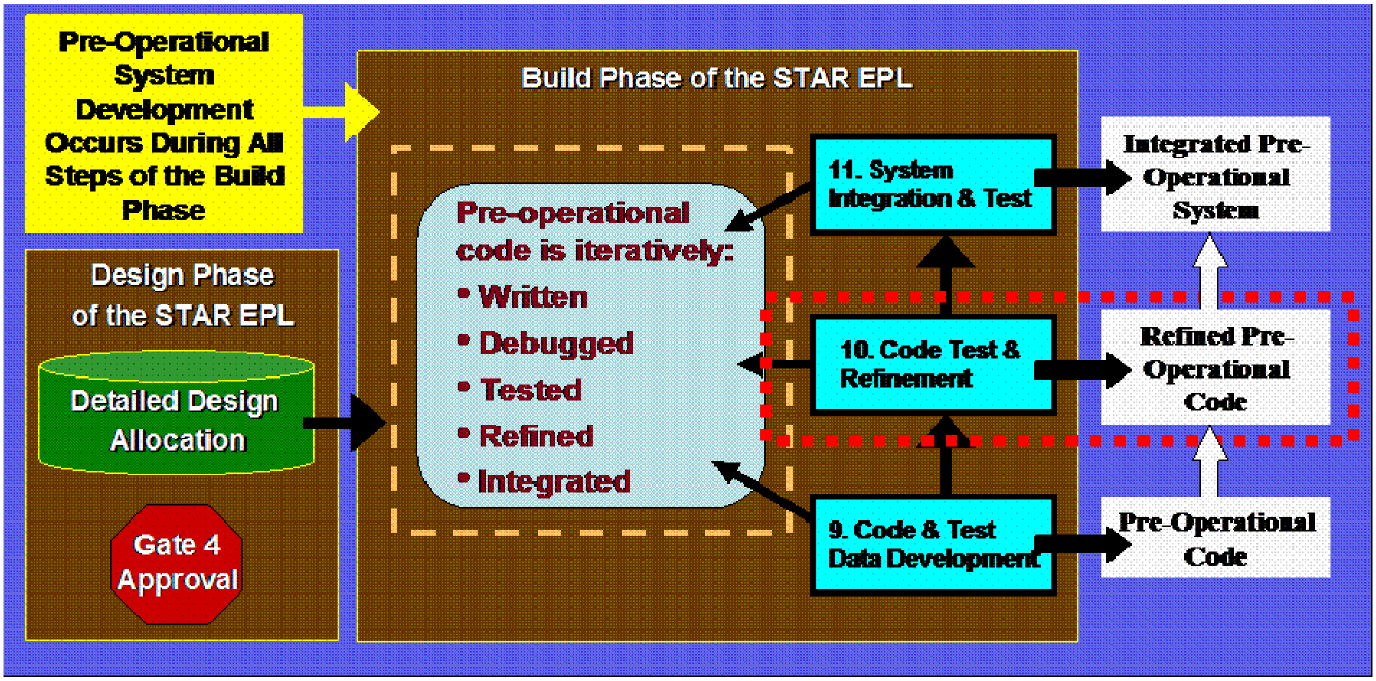
Pre-Operational System Development Occurs During All Steps of the Build Phase
Build Phase of the STAR EPL
Design Phase of the STAR EPL
Detailed Design Allocation
Gate 4 Approval
Pre-operational code is iteratively:
• Written
• Debugged
• Tested
• Refined
• Integrated
11. System Integration & Test
10. Code Test & Refinement
9. Code & Test Data Development
Integrated Pre-Operational System
Refined Pre-Operational Code
Pre-Operational Code

Figure 6.1 – Pre-Operational System Development Process

As Figure 6.1 shows, the objective of step 9 is to produce pre-operational code. Pre-operational code consists of the system components in the detailed design (software units and sub-units). This code will be written and debugged, and ready for unit testing, but not formally tested.

As Figure 6.1 shows, the objective of step 10 is to produce refined pre-operational code. Refined pre-operational code has been unit tested, refined, and debugged until it passes all of the unit tests, and is ready for integration into a pre-operational system.

The process of producing a complete pre-operational product processing system involves writing, debugging, testing, refining, and integrating the code. Because these functions affect each other, the process is inherently iterative. Figure 6.2 illustrates this.

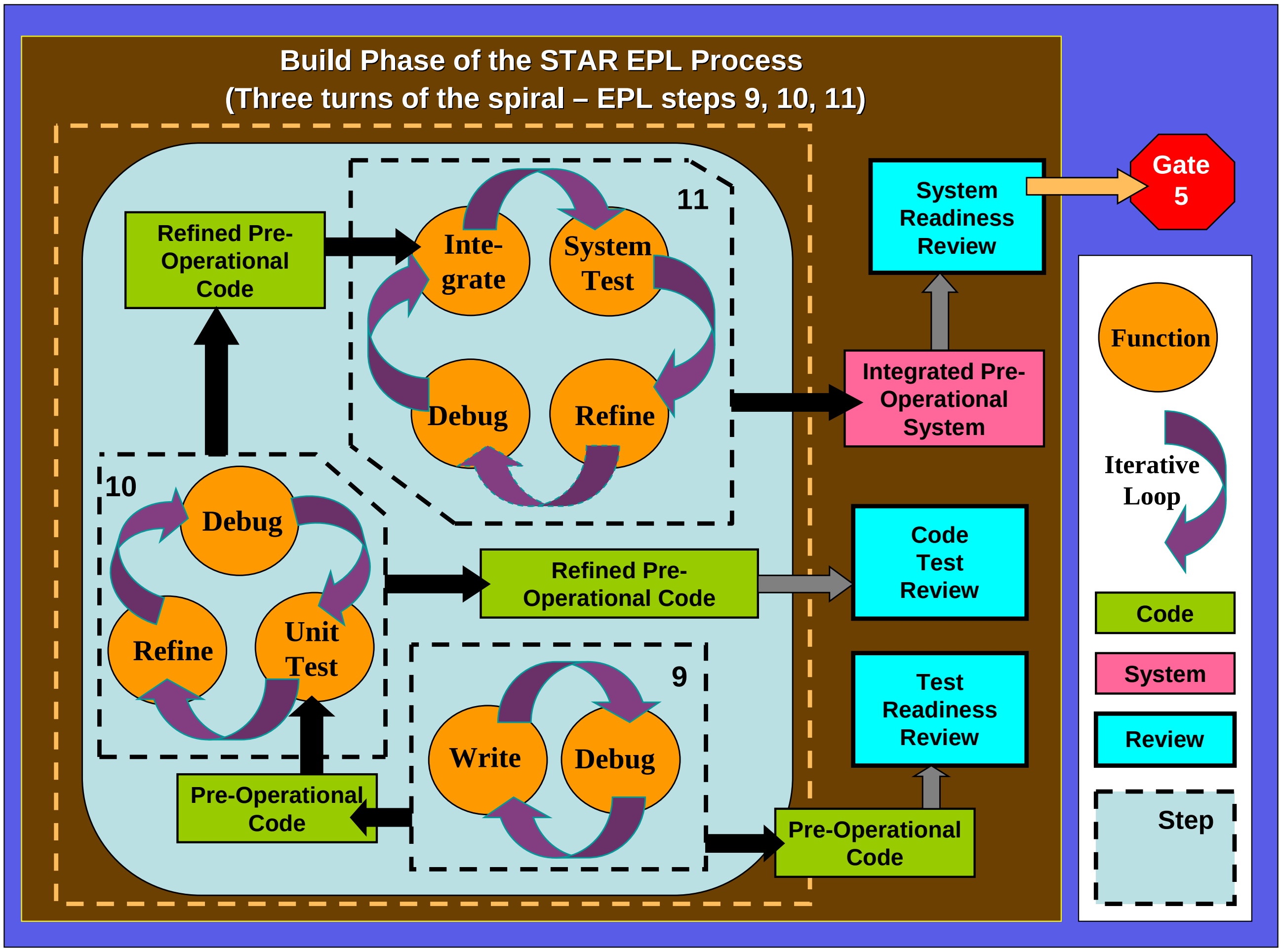

Figure 6.2 – Iterative (Spiral) Development of Pre-Operational System

Step 10 is a continuation of the pre-operational system development that was started in step 9 and that will be completed in step 11. The details of these steps are described in TG-9 and TG-11 respectively. This is illustrated in Figure 6.2 as the output from step 9 is pre-operational code that is input to the Unit Test function of step 10, and the output from debugged until it passes all unit tests.

6.2 Code Test and Refinement Process Flow

Figure 6.3 shows the process flow for step 10.

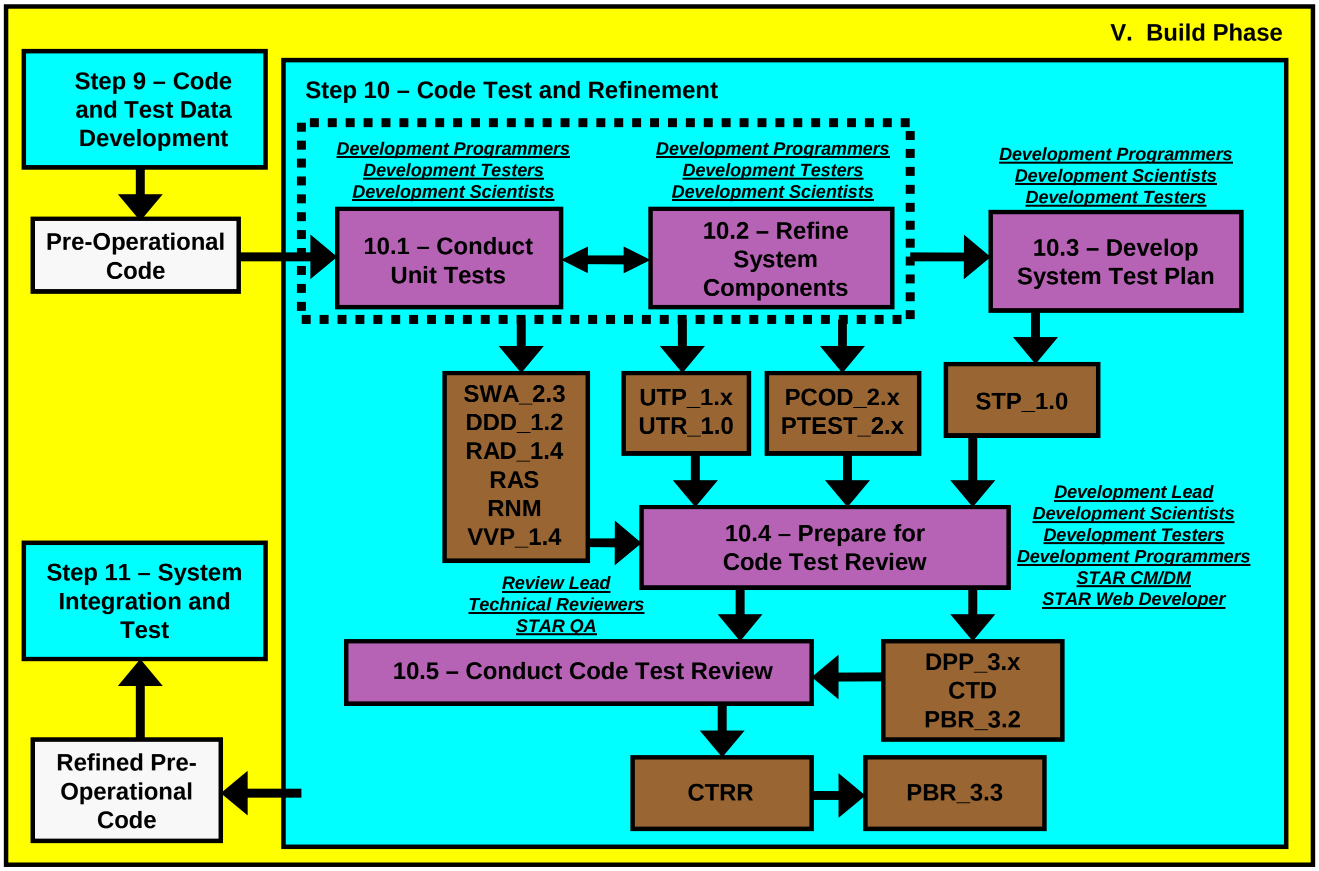

Figure 6.3 – Step 10 Process Flow

Note that processes 10.1 and 10.2 are enclosed by a common dashed border. This is to indicate that the processes are iterative, as explained in Sections 6.1, 6.5.1, and 6.5.2.

6.3 Expected BEGIN State

- REQUIRED: Pre-operational code has been debugged, compiled, and built into executable software units, ready for unit testing in the designated test environment.
- REQUIRED: A plan for unit testing has been developed. The plan ensures that the unit tests will address all required code functionality and code outputs.
- REQUIRED: All data required for implementation of the unit test plan has been acquired or developed, and is available in the designated test environment.
- REQUIRED: A TRR has been conducted
- REQUIRED: TRR reviewers have approved the project to proceed to the Code Test and Refinement step, and have documented this approval in the Test Readiness Review Report (TRRR).
- REQUIRED: Baseline Build (BB) 3.1 has placed the following items in the project artifact repository:
    - Pre-operational code
    - Unit test data
    - DPP, including Appendices
    - RAD, including Appendices
    - VVP
    - ATBD
    - SWA
    - DDD
    - UTP
    - Test Readiness Document (TRD)
    - TRRR
- EXPECTED: BB 3.1 has placed the following items in the project artifact repository:
    - R&D code
    - R&D test data
    - Project Proposal (PP)
    - Gate 2 Review Report (G2RR)
    - Gate 3 Review Report (G3RR)
    - Operations Concept Document (OCD)
    - Project Requirements Document (PRD)

  - Project Requirements Review Report (PRRR)
  - Preliminary Design Document (PDD)
  - Preliminary Design Review Report (PDRR)
  - Critical Design Document (CDD)
  - Critical Design Review Report (CDRR)
  - Gate 4 Document (G4D)
  - Gate 4 Review Report (G4RR)
  - Project Status Report (PSR), including Appendix
- REQUIRED: PBR_3.1 documents the status of the BB 3.1 project baseline

6.3.1 Task Inputs

Task inputs consist of the following BB 3.1 items:

- Pre-operational code (PCOD_1.x)
- Pre-operational test data (PTEST_1.x)
- SWA_2.2
- DDD_1.1
- UTP_1.0
- DPP_3.x
- RAD_1.3
- Requirements/Needs Matrix (RNM)
- Requirements Allocation Sheet (RAS)
- VVP_1.3
- TRD
- TRRR
- PSR_2.x Appendix
- PBR_3.1

6.3.2 Corrective Actions

The TRRR will document any actions that are needed to reduce risk during step 10. Usually, these actions should be closed before the CTR.

Additional corrective actions are typically generated during step 10, to mitigate project risks that are identified during code test and refinement. Project risks and risk mitigation actions should be identified in the PSR Appendix.

The needed corrective actions may require revisions to the project plan, typically by the addition of sub-tasks and revisions to the task schedule. The **Development Lead** should determine whether these revisions are manageable or are so significant that a re-plan is needed. If necessary, the **Development Lead** should consult STAR Management on the advisability of a re-plan. Re-planning is expected to be a rare event, but it may occur if the project requirements have added significant scope or if unexpected technical issues have been discovered.

6.3.2.1 Delta Gate 4 Review

If it is determined that a re-plan is needed, the **Development Lead** should consult STAR Management to determine whether there should be a delta Gate 4 Review. This determination should depend upon whether the re-plan will significantly affect the pre-operational code. If so, a delta Gate 4 Review should be prepared for and conducted in the same manner as the normal Gate 4 Review. Refer to the step 8 Task Guideline (TG-8) and the Gate 4 Peer Review Guideline (PRG-8.2) for guidance. Following approval of the re-plan, the project can return to its step 10 activities under the new plan.

6.4 Desired END State

- A Detailed Design Allocation of the requirements identifies product and system components down to the Sub-Unit-Layer, and traces each component to one or more requirement.
- A plan has been developed for monitoring the status of the requirements and their allocation to ensure that the integrity of the requirements allocation is preserved as the implementation of the detailed design proceeds through the Build phase.
- The functionality of all system components in the detailed design (software units and sub-units) has been implemented in pre-operational code that meets coding standards.

- A plan for unit testing has been developed. The plan ensures that the unit tests will address all required code functionality and code outputs.
- Pre-operational code has been tested in accordance with the unit test plan.
- Pre-operational code has been refined and debugged as necessary until it passes all unit tests.
- Unit test results have been documented in a report.
- A plan for system testing has been developed. The plan ensures that the system test will address all system requirements and product requirements.
- The project plan has been updated as necessary
- The status of project risks and actions has been updated
- A CTR of the project plan, unit test results, and system test plan has been conducted
- A CTRR has been written. The CTRR approves the project to proceed to the next step, System Integration and Test.
- Baseline Build 3.3 has placed the required items in the project artifact repository
- PBR_3.3 documents the status of the BB 3.3 project baseline

6.4.1 Task Outputs

Task outputs consist of the following BB 3.3 items:

- Refined Pre-Operational Code
- Refined Pre-Operational Test Data
- DPP_3.x
- SWA_2.3
- DDD_1.2
- RAD_1.4, including RNM and RAS
- VVP_1.4
- UTP_1.x
- STP_1.0

- CTD
- CTRR
- PBR_3.3

6.5 Code Test and Refinement Activities

Step 10 activities include:

1) Conduct unit tests
2) Refine system components
3) Develop system test plan
4) Prepare for CTR
5) Conduct CTR

6.5.1 Conduct Unit Tests

Development Programmers prepare the unit test environment, in accordance with the unit test plan.

Development Programmers build the unit test configuration, in accordance with the unit test plan. The unit test configuration includes all items that will be used in the unit test, including code modules, test data sets, utilities, libraries, etc. A provisional test configuration will often be built during step 9 to allow a dry run of the unit test. If this has occurred, adopt the provisional test configuration or adapt it as necessary.

Diagnostic code will typically be added to the unit code during step 9 to facilitate the verification of functionality and outputs. If this is done, two versions of the unit code will be created. The first version will contain the diagnostic code. The second version, with all diagnostic code stripped out, is the baseline pre-operational code. The unit test sequence should then include the execution of both code versions. The unit code with diagnostic code should be run first, with all diagnostic messages and outputs captured for verification. This provides a complete verification of code functionality and performance. Then, the baseline pre-operational unit code should be run. Verification of the baseline code is then achieved by confirming that its outputs are identical to the outputs from the unit code with diagnostic code. The test sequence should include this final verification step.

Development Testers run the unit tests, assisted by **Development Programmers**. **Development Scientists** assist in evaluating the unit test results. Examine the unit test output, including runtime messages, diagnostic messages and the content of intermediate and output data files.

Runtime messages are messages written by the operating system to a runtime log file or other designated output source (e.g., a monitor connected to the computer from which the program execution command has been entered). These may occur if the unit code is written to generate such messages as a way to test functionality.

Diagnostic messages are messages written by the unit program to a runtime log file or other designated output source (e.g., a monitor connected to the computer from which the program execution command has been entered). The nominal purpose of a diagnostic message is to report a functional result (e.g., ‘subroutine X called’) or the quantitative value of an input, intermediate, or output variable (e.g., ‘X(50) = 7’).

Data files include the output data sets that are designed to be produced by the unit program. In addition, a diagnostic program may write intermediate data sets to diagnostic files.

If the unit test output does not satisfy all success criteria, iteratively refine and debug the code (c.f. Section 6.5.2), and repeat unit testing until success criteria are satisfied. When success criteria are satisfied, document the results in the UTR, following guidelines in DG-10.1. **Development Testers** lead in the development of the UTR. **Development Programmers** and **Development Scientists** assist with the UTR. The purpose of UTR v1r0 is to document the results of testing of each software unit to verify that the requirements allocated to the unit’s software components are satisfied.

6.5.2 Refine System Components

Development Programmers iteratively refine, debug and re-test the pre-operational code as needed, based on the unit test results. Code refinements should conform to coding standards provided in TD-11.1 (FORTRAN code) and TD-11.2 (C code).

Development Testers refine the test data as necessary until the unit test requirements are satisfied. **Development Scientists** assist in refining the unit test data. The UTP is revised to account for changes to the unit test plan since the TRR. UTP v1r1 updates and refines the test data description, test methods, test sequences and test risks, based on the refinement of the code and test data.

In the process of refining and debugging the pre-operational code, problems with the detailed design may be discovered. In that case, the **Development Scientists** and **Development Programmers** should determine whether the detailed design needs revision to correct the problems. If the revisions are relatively minor and do not affect the Detailed Design Allocation, as is usually the case, document the revisions in updates of the SWA (v2r3) and/or the DDD (v1r2). If the revisions do affect the Detailed Design Allocation, trace the affected system components or product components back to the driving requirements to ensure that these are not compromised by the design revisions.

Update the RAD and its Appendices (RNM and RAS) to v1r4 to capture any changes to the Detailed Design Allocation that have occurred as a result of design revisions during step 10.

Development Scientists and **Development Testers** assist in a revision of the VVP, following the guidelines in DG-6.3. VVP v1r4 adds to v1r3 by updating the listing and description of verification and validation items and plans, based on changes to the Detailed Design Allocation since TRR, as documented in RAD v1r4.

6.5.3 Develop System Test Plan

Development Programmers, Development Scientists and Development Testers collaborate in the development of a system test plan, documented in STP v1r0, following guidelines in DG-10.2. The purpose of STP v1r0 is to present the plan for testing to ensure that the requirements specified for the product processing system (PPS) are satisfied by the completed system (Verification) and that the final developed system will satisfy the users' needs and expectations (Validation). The purpose of the system test is to demonstrate, using verification and validation methods, system readiness for operations. The STP builds on the project's VVP and UTP.

Identify the system requirements and product requirements that will be tested. Trace these to user needs and operator needs. These should be documented in the RAD, RNM, and OCD.

Identify all items that have been selected for testing. These items are system components and product components that should have been identified as verification items in the project's VVP. It is important that the development team confirm that the planned system test items are documented as verification items in the VVP. Discrepancies must be resolved and either the STP or the VVP revised as necessary.

System components are defined as any item that is necessary or useful for building the end-use product, but will not be delivered to customers and/or end users. System components include the elements of the software architecture, as described in the UTP. Typically, they are elements of the process flow of the software units that have been tested in the unit tests, as described in the UTP. Typically, the system test will include the end-to-end execution of the software units.

Product components are defined as any item that will be integrated to form the end-use product, i.e. these are the deliverable items. Typically, the product components are outputs from the end-to-end execution of the software units.

List and describe all data files that will be used as input files for the system test. Files to be listed here include: “Test data” includes sensor data (real, proxy, or simulated), ancillary data, control files, parameter files, and look up tables. Files to be listed here include:

- “Test data”. These data sets include the sensor data (real, proxy, or simulated), ancillary data, control files, parameter files, and look up tables that are needed to run the system test.
- “Truth” data. These are data sets that will be used to assess the quality of the system output. Truth data sets typically contain the values of environmental or weather products that are traceable to performance requirements. Truth data sets may be real, proxy or simulated data. Explain how each real or proxy truth data set has been obtained. Explain how each simulated truth data set has been constructed.

Typically, the system test will use some or all of the same test data and truth data that was used for the unit tests. In that case, information on this data can be obtained from the UTP. If the system test includes new test data in addition to the unit test data, add a description of this data to the STP.

Determine and describe the environment in which the system test will be performed. A project may use the development environment to test the pre-operational system or it may choose to establish a separate test environment. Project constraints will usually determine this choice. For example, operations may request that the test environment be a clone of the operational environment, but cost factors may exclude establishing the development environment as an operational clone. In that case, the best solution may be to use a small operational clone environment as a separate test environment. A project may also choose to perform its pre-operational code unit tests in the development environment and then perform its system integration tests in an operational clone environment. In any case, these choices should be explicitly stated as requirements for the test environment

It is preferable to use the same test environment for the system test as was used for all unit tests. For cases where the test environments differ, the system test should include verification that the same inputs to each software unit results in identical outputs when the unit is run in the system test environment.

Determine and describe the test methods that will be used. Test methods should be closely related to the verification methods that are documented in the VVP. The standard methods include Analysis, Demonstration, Inspection, and Test. Note that unit test methods are not restricted to “Test”. The “Test” verification method refers specifically to a procedure to quantitatively demonstrate compliance with performance specifications. Although test methods will often include “Test” methods for verifying quantitative requirements, they can also include, and usually will include, other methods for verifying other requirements. Note which test items will be verified with each method or combination of methods.

It is permissible to insert relevant material from the VVP into the STP, provided it is referenced appropriately. Alternatively, the STP developers may choose to leave the specific material out of the STP and refer to specific sections of the VVP that pertain to the test methods. The latter choice is recommended if the CtR reviewers are already familiar with the material in the project’s VVP. It is recommended that the STP developers consult with the CTR reviewers before deciding how document test methods in the STP.

Identify all configuration items that will be used in the unit test, including code modules, test data sets, utilities, libraries, etc. The set of configuration items is called the test configuration. Identify the procedure to build the test configuration into a test executable.

Describe the planned sequence of test actions in sufficient detail that a reviewer can confirm that all test items are exercised, all test data are utilized, all planned test methods are used as planned, and the planned output will allow a reviewer to confirm that the requirements will be satisfied. Note which sequence steps exercise which test items, utilize which test data sets, and use which test methods.

Describe the expected output from each sequence step. The expected output should be described in sufficient detail for system test reviewers to be able to confirm that the actual system test output matches the expected output. Output includes runtime messages, diagnostic messages and the content of data files.

State the success criteria. Success criteria should include the following:

- All test sequence steps run as planned in the STP
- Program run time meets requirements
- All runtime message and diagnostic messages are written as expected in the STP
- The contents of all diagnostic files are as expected in the STP
- System test output satisfies all quantitative performance requirements

6.5.4 Prepare for CTR

The CTR review lead (**Technical Review Lead)** and review team (**Technical Reviewers)** should have been selected during steps 5, 6, 7, 8, or 9, and listed in the DPP. If this selection was not completed in step 9, the **STAR Branch Chief**, in consultation with the **Development Lead**, should make this selection as soon as possible during step 10.

The DPP should be updated to address any changes to the project plan since the TRR. The **Development Lead** updates the DPP to version 3.x, with assistance from the **Development Scientists, Development Testers,** and **Development Programmers**.

The **Development Lead** leads the preparation of the CTR presentation. The CTR slide package is the Code Test Document (CTD). The CTD is prepared by the **Development Lead**, **Development Scientists, Development Testers,** and **Development Programmers,** in accordance with CTD guidelines DG-10.3. DG-10.3.A provides CTD slide templates that can be adapted for the project's CTD. The CTD developers should examine the DPP to determine whether the CTR objectives, entry criteria, exit criteria and/or CLI have been tailored. If so, the CTD slide templates must be adapted to accommodate the tailoring. The CTD developers should use the project's TRD as a source for CTD slides, as many TRD slides can be re-used or adapted.

The **Development Lead,** assisted by the **Development Scientists, Development Testers,** and **Development Programmers,** updates the status of the project risks and associated risk mitigation actions for inclusion in the CTD. Risk management guidelines can be found in PG-1.

The **Development Lead** determines which members of the development team will present the CTD sections. These presenters should be noted in Section Title slides. See DG-10.3.A for examples.

STAR CM/DM inserts the standard BB 3.2 items in the baseline, and updates the Project Baseline Report (PBR) to version 3.2, in accordance with PBR guidelines DG-5.4.

The **Development Lead** informs the **STAR Web Developer** that the CTR artifacts are ready for posting on the STAR EPL website. The **STAR Web Developer** works with **STAR CM/DM** to acquire the project baseline items and post them on the website.

Once these are posted the **STAR Web Developer** informs the **Development Lead**, who then informs the **Technical Review Lead** that the CTR artifacts are available for review. The **Technical Review Lead** then informs all review team members that the artifacts are available to them.

The **Technical Review Lead** and **Technical Reviewers** may at their discretion examine the artifacts and communicate issues to the **Development Lead** prior to the review date, so that the artifacts and/or review presentation may be revised to respond to reviewer concerns.

6.5.5 Conduct CTR

The CTR consists of the presentation of the pre-operational code, test data, and test plan by the development team (**Development Lead**, **Development Scientists, Development Testers,** and **Development Programmers)** and the disposition of the review CLI, including entry and exit criteria, by the reviewers (**Technical Review Lead** and **Technical Reviewers**).

The **Technical Review Lead** and the **Technical Reviewers** conduct the CTR to determine whether the refined pre-operational code has satisfied unit test criteria and is ready for integration into a pre-operational product processing system. Reviewers should be familiar with the CTR guidelines (PRG-10) and check list (CL-10).

The CTR reviewers complete a Code Test Review Report (CTRR), following guidelines in DG-10.4. The CTRR will include the reviewers' assessment of the status of the CTR artifacts, the project risks, and associated risk mitigation actions, and an Appendix that consists of the reviewers' disposition of each CTR CLI.

On the basis of its disposition of the CTR CLI, the **Technical Review Lead** and the **Technical Reviewers** determine whether the project is ready to proceed to the next step, "System Integration and Test". If not, the CTRR should direct the **Development Lead** to revise the CTR artifacts through specified actions. These actions may include a new assessment of revised CTR artifacts at a delta review.

If a delta CTR is required, the **Development Lead** and support team upgrade the CTR artifacts as requested by the CTR reviewers and present them at a delta CTR. This is repeated until the **Technical Reviewers** pass the project to the next step.

If a delta review is not required, the revision of the CTR artifacts will be deferred to actions performed during step 11 for review at the System Readiness Review (SRR). All of this should be documented in the final version of the CTRR.

STAR QA verifies that the CTR was conducted in accordance with STAR EPL standards.

STAR CM/DM updates the project baseline via BB 3.3, and updates the Project Baseline Report (PBR) to version 3.3, in accordance with PBR guidelines DG-5.4. BB 3.3 will include all post-CTR revisions to the CTR artifacts, the CTRR, and PBR_3.3.

When the project passes its CTR, the project proceeds to system integration and system testing. The final version of the CTRR should include approval for the project to proceed to step 11, and will indicate all open actions that have been deferred to step 11.

Each stakeholder who performed activities during step 10 is encouraged to document an assessment of the experience in a personal record. This assessment should include: what was good, what was bad, what worked, what did not work, what can be improved, how it can be improved.

The **Development Lead** should remind the stakeholders to do this. At the conclusion of Development (step 11), the **Development Lead** will collect the final edited personal stakeholder records and incorporate them into a Development Project Report (DPR).

END OF DOCUMENT