IfA/MHPCC
Panoramic Survey Telescope
And Rapid Response System (Pan-STARRS)
Image Processing Pipeline (IPP)
Software Configuration Management Plan (SCMP)
Approval Signatures

Prepared By:

Eric Van Alst, MHPCC
CM and Test Lead

Date

Prepared By:

Michael J. Berning, MHPCC,
Sr. Systems and Software Lead

Date

Concurrence:

Robert DeSonia, MHPCC
Software Architect

Date

Approval:

Bruce Duncan, MHPCC
Program Manager

Date

William Burgett, IfA
Deputy Project Manager, Chief Eng. Engineer

Date

Eugene Magnier, IfA
IPP Lead

Date

Revision History
<table>
<thead>
<tr>
<th>Revision Level</th>
<th>Date</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version 1.0</td>
<td>1/10/2004</td>
<td>Initial Version</td>
</tr>
<tr>
<td>Version 1.1</td>
<td>2/24/2004</td>
<td></td>
</tr>
<tr>
<td>Version 1.2</td>
<td>10/28/2004</td>
<td>Andy Douglas modifications</td>
</tr>
<tr>
<td>Version 1.3</td>
<td>11/03/04</td>
<td>MHPCC modifications</td>
</tr>
</tbody>
</table>
**TABLE OF CONTENTS**

1Introduction........................................................................................................................................1
2Applicable Documents......................................................................................................................2
3Organization and Resources.............................................................................................................2
    3.1CM Lead/Specialist..................................................................................................................2
    3.2Resources...............................................................................................................................3
4Software CM Activities..................................................................................................................3
    4.1Baselines..................................................................................................................................4
    4.2Naming/Version Convention....................................................................................................4
    4.44.2.1 MHPCC SDL..................................................................................................................5
    4.54.2.2 Configuration Management Tools................................................................................8
    4.64.2.3 Non-Deliverable Software (NDS)................................................................................8
    4.74.2.4 Configuration Builds......................................................................................................8
    4.84.2.6 Change Control Forms..................................................................................................8
    4.9................................................................................................................................................11
    4.10 4.2.7 Review Boards..........................................................................................................11
    4.114.3.1 Configuration Audits................................................................................................14
5A. Appendix......................................................................................................................................15
6A.1 Acronyms....................................................................................................................................15
7A.2 Glossary......................................................................................................................................17
8A.3 Software Problem Report Prioritization Categories.................................................................19
9A.4 Software Problem Severity Categories..................................................................................20
10A.5 Software Problem Report Status Categories..........................................................................21
11A.6 Software Problem Report Resolution Categories.................................................................22

**TABLE OF FIGURES, TABLES**

Software Configuration Management Plan
Revision 1.3
11/5/2004

Pan-STARRS Document Control
PSDC-430-003-01
1 Introduction

1.1 Identification

This plan is identified as the Maui High Performance Computing Center (MHPCC) Software Configuration Management Plan (SCMP) and addresses the specific configuration management (CM) requirements for the production of the MHPCC Pan-STARRS IPP software products.

1.2 Scope

The policies and procedures identified in this SCMP apply to the generation and delivery of the MHPCC implementation of IPP software products. The MHPCC Software Work Products (SWP) may include, but are not limited to the following domains that are identified with or required to create the software products.

- Source code
- Run-time executable image(s)
- Test environment (hardware, software, test results, etc.)
- Software Performance Metrics

1.5 Applicable third-party vendor software

- COTS Software Quality Assurance Tools
- Math Libraries
- Collaborative Software Engineering Tools
- Software Development Tools

1.8 Objectives

The SCMP establishes the strategies for creating and maintaining a uniform system of configuration identification, control, status accounting, and audit for the SWPs throughout the software development process.

The objective of the SCMP is to define the policies and procedures to be used by the MHPCC software development personnel in managing the configuration control of the SWPs. Specifically; the SCMP provides guidelines and direction for:

1. Establishing internal software development baseline identification of characteristics and performance criteria (i.e., software code and related documentation).
2. Providing change control and change visibility to SWPs through formal configuration control procedures.
3. Controlling the incorporation of all approved, internally controlled or baseline software, and related documentation, to the Software Development Library (SDL), and the subsequent release of computer software configuration items (CSCI) to MHPCC Integration & Test, Configuration Management (CM), and/or other Customer(s).

4. Providing status accounting of SWPs, which are submitted.

5. Ensuring that only approved changes are incorporated into the baseline SWPs.

6. Maintaining a configuration audit system to ensure that records, which are provided to the SDL, to the Test Team, to the CM, and/or to other customers, are consistent with documentation and SWP identification.

2 Applicable Documents

Listed below are the compliance documents that govern the writing of this SCMP, the reference documents, which add to the understanding of the SCMP, and the process directives, which effect implementation of the software CM function.

2.1 Compliance Documents

TBD

2.2 Reference Documents

TBD/TBS

2.3 Pan-STARRS Software CM Process Directives

TBD/TBS

3 Organization and Resources

3.1 Organizational Structure

The organizational structure of the software development group, including the CM function, is shown in Figure 3.1-1. The software CM Lead reports directly to the Sr. System and Software Engineer Lead.

3.1 CM Lead/Specialist

In support of software product development activities, the software CM Lead responsibilities shall include the following:

1. Establish, implement, and maintain the SDL, and its procedures that control creation of SDL configurations and baselines.
2. Ensure the accountability of all SDL activity.
3. Control all changes made to, and within, the SDL, and maintain the integrity of SWPs, which are placed in the SDL.
4. Support Software Configuration Review Board (SCRB) activities by producing agendas, minutes, summary reports, and maintaining action item records.
5. Update the Software Problem Report (SPR) database to reflect SCRB activities (e.g., disposition, action item assignment(s), etc.) pertaining to change processing.
6. Provide statistical analysis reporting of the SPR trends to support management oversight of the software integration activity.
7. Maintain internal status accounting for SDL configurations, baselines, and SWPs

![Figure 3.1-1 MHPCC Software Development Group](image)

3.2 **Resources**

The software CM establishes performs, and monitors baseline CM activities for software code by a transfer or copy of the code from the software development environment into the software CM controlled Concurrent Versions System (CVS) application environment. It is within the CVS environment that formal builds and deliverable source code products are produced. Software Work Products (SWP) transfers into the CVS environment may occur at regularly scheduled intervals, or may occur in conjunction with software development events or milestones.

4 **Software CM Activities**
This section provides a description of the CM activities associated with configuration identification, change control, audit and status accounting, and shows how these activities apply during the software development life cycle.

4.1 Configuration Identification

Configuration identification of an SP includes all activities required to establish the identification of the product in the form of technical documentation. The initial step in this process is the identification of each CSCI. Any software that is required for logistic support of the software development or the build environment, and that is designated for separate procurement is identified as a CSCI. Configuration audits, and reports of CSCI status are all based upon the identified CSCIs.

4.1 Baselines

4.1.1 Developmental Configuration

The developmental configuration consists of the elements of a SWP description. These product descriptions include designs, source code, integration and test data, and other support data. A SWP remains in the Developmental Configuration state until it is ready for subsystem acceptance testing. Pan-STARRS MHPCC CM is established after completion of the subsystem acceptance testing.

While a product is in the development configuration phase, the SCRB controls discrepancy reporting and change request processing.

4.1.2 Configuration Management Baseline

The CM baseline is submitted to the CM for control during subsystem acceptance testing of the CSCI. Documentation that defines the requirements, design, test of the software products (and their constituent CSCIs), and the engineering drawings that define the hardware environment is also submitted.

4.2 Naming/Version Convention

The software CM Lead in conjunction with the software development leads and the software integration team establishes the convention for assigning name, version, and revision identification to the SU(s). This is in order to maintain historical information and to associate particular unit versions with a specific release or build.

SP baselines are named with the corresponding SU release version identification. This is performed after all formal testing is completed, and IPP Configuration Control Board (CCB) approval is obtained.

Example:
Beginning with “Release 1_0”, the SP release (code) is designated as Release [1-9] .0 (e.g., 1_0, 2_0, etc.). Minor releases are designated by a first level underscore, e.g., 1_1, 1_2, etc.). Approved baseline updates are reflected by a version number rollover.
Developmental configuration items are versioned using an identifier that includes a release level and build number.

The software CM maintains the corresponding master SP code in electronic form within the SDL. The MHPCC CM also maintains the master release media for SP documentation in hardcopy and/or electronic format.

4.2.1 Identifying Media

Physical media labels, whether for production items or backup copies, will include product name, control number, save/recovery format, brief content description, and date. The build or release version is also added by the CM, if applicable.

Electronic media labeling is performed as described in Section 4.1.2 above.

The backup frequency and type is under the auspices of the system network management. For software CM requested backups, the media form will include the naming convention specified in Section 4.1.2, above.

4.2 Configuration Control

The subparagraphs in this section describe the transition of SWPs from informal to formal configuration control within the SDL, and the final transition to software CM (in the case of software code) along with the confirming tests/reviews and inputs/outputs for each transition, plus the approval authorities for each controlling entity.

4.3

4.4 4.2.1 MHPCC SDL

The development and informal design tests of software products are conducted in the software engineering environment of the SDL. This environment supports the management and maintenance of the user work area by providing a convenient repository with user-controlled access to the supporting documents and software. As Software Units (SUs) are successfully tested, they are transferred to user-controlled product directories where they become available for SU Integration testing. The software that supports integration testing, such as software stubs, utility routines, or sets of test data also resides in this portion of the SDL.

After successful SU integration testing, we integrate the SUs into a software build at the “work” level of control at which developers can check them out.

Once the build baseline is established, the software design and code are sequentially promoted to the “Integrate”, “Pre_CM”, and “CM” levels of control for increasingly formal software CM control within the Software Controlled Libraries.

Approval authority for changes to the software CM-controlled Baseline lies with the IPP CCB.
Pre_CM is the software formal internal test-release that is to be used by the MHPCC software test team to evaluate how the software performs against its requirements.

Any discrepancies in the Pre-CM release level are noted in the Bugzilla Bug Tracking System. The CM Lead builds a new Pre_CM releases multiple times per release cycle as SPRs get fixed and included in the software baseline. The number of Pre_CM releases varies with the complexity of an individual build cycle.

CM is the formal tested delivery of the software to the customer for use in their acceptance testing. The customer, in dry running their procedures, uses the developmental CM releases. The customer may load the CM release either in their evaluation environment (s) and/or on the target system. CM releases occur at Pan-STARSS project requested intervals. Figure 4.2-1 illustrates the CM release cycle.

Both the user controlled and the software CM controlled portions of the SDL are subject to audits and reviews by the QA Lead in order to ensure that SDL procedures have been adhered to in the development of the software products.

Prior to subsystem acceptance testing, copies of the current software, test software and test artifacts must be withdrawn from the CM controlled portion of the SDL and transferred to the software engineering test environment. The CM performs updates to the test environment during the formal test activities per SCRB direction. The process for controlling the software test products is the same as for the software products.
Figure 4.2-1 CM Software Release Cycle

While development is incomplete

Initiate New CM Build Cycle

New Developmental Capabilities

Corrective Actions (SPRs)

Change Requests (CCB Items)

Prioritized Build Backlog List

pre-CM Build (1)

pre-CM Build (2)

(...)

pre-CM Build (n)

SU Test

Generate CM Build and Release Document(s)

CM Release of software

CM Release Supporting Documentation

Acceptance Test

Any SPRs?

Yes

No - Start Next CM Release Build

Acceptance Test Cycle

Release CM Build to IfA

SVD

1 to n pre-CM Builds per Formal Release Cycle

Prioritized Build Backlog List

Generate CM Build and Release Document(s)

CM Release of software

CM Release Supporting Documentation

Acceptance Test Cycle

Release CM Build to IfA

SVD
4.5 4.2.2 Configuration Management Tools

CM is performed using the CVS and the Bugzilla Bug Tracking Database. CVS maintains a history of a source tree, in terms of a series of changes. It stamps each change with the time it was made and the name of the performer who made it. As a rule, the performer provides text comments describing why the change was made. CVS is used by the software team to enforce: SP archival, identification, version/change control, status accounting, auditing activities, and all build processes.

The Bugzilla Bug Tracking Database is used for recording, tracking and reporting SPRs. The Software Lead is assigned SPRs by the SCRB. In turn, the Software Lead designates a specific Software Analyst for each SPR. The Analyst then obtains the official description of a specific SPR connected to the Pan-STARRS Bugzilla Database and can query the bug report by its tracking number.

4.6 4.2.3 Non-Deliverable Software (NDS)

A subset of the SDL is the list of non-deliverable, but controlled, items. These are software and reference materials relevant to the generation of the products. Examples of non-deliverable items are tools and utilities used for development and design.

The Software Development Lead authorizes updates to the NDS. For each non-deliverable, support, or COTS CSCI, a clearly designated engineer has sole responsibility for coordinating any changes with all affected users and obtaining SCRB approval prior to making the modifications.

The IfA IPP Lead shall approve all materials classified as NDS. The IfA IPP lead has the authority to elevate designated NDS to a deliverable status.

4.7 4.2.4 Configuration Builds

The CM performs all product release builds. Developmental configuration builds are performed at the local engineering environment level, and are performed at TBD intervals. Any compile/link errors encountered are published informally to the software team for corrective action. Release builds are performed to meet software delivery schedules for build testing, formal testing, and/or customer delivery. A software delivery requires that all component CSCIs be complete, consistent, built within a configuration-controlled environment, and successfully tested with configuration controlled test procedures. All supporting documentation must also be kept current to reflect the software builds.

4.8 4.2.6 Change Control Forms

Change control forms are the media used for initiating, evaluating, approving or rejecting, implementing, and releasing changes. They also are used for reporting problems, requesting modifications, and submitting change proposals.
Change control forms are the key source of information concerning the status of changes during change processing. Software change control forms are currently in electronic form within the Bugzilla Bug Tracking Database Application.
The SPR is used for initiating and processing changes to controlled baselines within the SDL, and in the submittal of proposed software changes requiring IPP CCB approval.

4.8.1 4.2.6.1 Change Classification

Proposed changes are grouped into two classifications, denoted major and minor, according to the impact of the change as defined by the MHPCC CM organization.

Major changes MUST BE APPROVED by the IPP or System CCB in order to proceed; they are changes that impact the required performance, costs, schedule, global characteristics, power consumption, reliability of any hardware component, part or product, or require a change to external interfaces.

Minor changes DO NOT IMPACT ANY other subsystem, part, product, or interface (external or internal), or other required performance, cost, schedule, global characteristics, power consumption, reliability of any associated hardware component, part or product.

Since a major change requires IPP or System CCB approval prior to implementation, a major change package must be submitted to the IPP CCB for processing and it should include the Software Change Request (SCR) as part of the supporting documentation, if applicable. Minor changes are handled by SCRB, within the MHPCC software organization.

For both major and minor changes, the IPP Lead shall be notified electronically of all proposed changes. As stated in the IPP CCB Charter, the IPP Lead shall have the authority to elevate any changes to the IPP CCB

4.8.2 4.2.6.2 SPR/SCR

Proposed SP modifications fall into two categories, SPRs and SCRs. The same form is used to document either category. Within this form, a field is selected to indicate whether the modification is a SPR or a SCR.

The purpose of the SPR is to:

- Report a known or suspected problem, defect, deficiency, or error in software, in hardware, in a database, or in the documentation.
- Report a software or hardware error found during review, informal test, integration test, formal test, or operation.
- Report defects discovered in COTS software or hardware.
- Report defects in software processes.

The purpose of the SCR is to:
• Request a software, hardware, API, documentation, or process modification that is not an error, problem or deficiency as noted in the SPR above.

Enhancement request, or planned developmental task(s).

Approval for SPR/SCR disposition and work off is based on the SPR/SCR Priority and Severity Categories shown in Appendix A.3 and Appendix A.4. Additional severities and/or categories may be added, as necessary. These categories are also represented one-to-one within the Bugzilla Problem Tracking Tool. SPR Status categories are defined in Appendix A.5 and the subsequent resolution categories are defined in Appendix A.6.

4.9

4.10 4.2.7 Review Boards

Depending upon the change classification, either the SCRB or the IPP CCB must authorize proposed modifications to CM controlled software within the SDL. This process determines the total change impact before the change is approved. Figure 4.2-2 shows the SCRB CM Flow and SPR/SCR closure process.

Figure 4.2-2 SCRB CM Flow Chart

4.10.1

4.10.2 4.2.7.1 SCRB

Specific responsibilities of SCRB concerning the software products are to review, approve, prioritize, and track changes to the baselines and to determine when the developmental baseline should be transitioned to the CM baseline. The SCRB constitutes the Sr. Software Lead, the Software Architect, the CM/Test Lead. The SCRB meeting provides visibility of reported problems and proposed modifications by allowing board members to evaluate the impact of a problem under investigation or a change before adoption. The SCRB reviews all newly opened SPRs, all SPRs proposed for closure, and all high-priority open SPRs at each meeting. Changes to baseline SWPs, that do not impact formally baselined software products, may be implemented after SCRB review and approval. All decisions resulting from the SCRB are ultimately the responsibility of the SCRB Chairperson, i.e., the Software Development Lead, or Designee. CM ensures correct implementation of the approved changes in the Software Controlled Libraries and provides closure and status accounting information in the SPR/SCR database.

The SCRB Chairperson may assign Action Items (AIs) that are related to SCRB activities, for example, to request additional information to aid in the disposition of a SPR/SCR, or to divide the tasks associated with change implementation into smaller, more easily tracked sub-tasks. CM is responsible for recording and the status of open and closed SCRB AIs as well as SPRs. AIs and SPRs statuses are reported at each SCRB meeting.
Figure 4.2-3 diagrams the SCRB SPR/SCR evaluation process activities.
Figure 4.2-3. SCRB Process Flow
4.10.3  4.2.7.2  IPP CCB

The IPP CCB is the product-level change approval authority that screens proposed major changes as described in Section 4.2.6.1.

4.3  Configuration Status Accounting

Software Configuration Status Accounting (CSA) consists of recording and reporting information needed to effectively manage the configuration of SWPs. Software Configuration Status Accounting Report (CSAR) provides for: each released CSCI, the status of proposed changes, and the status on the implementation of approved changes.

SDL activity information gathered from the CVS environment and the Bugzilla Bug Tracking SPR Database are used in generating internal status accounting reports to support SWP development and SCR change processing activities. The initial requirement is to generate, on a monthly basis, a general report of open, closed, and in-progress SPR/SCRs, and a metrics report indicating length of time from SPR/SCR submittal to closure, and the current processing time of in-progress SPR/SCRs. Eventually, additional metrics will be collected, such as changes to test cases and requirements.

The primary status accounting document for delivery of the software products is the SVD. The SVD defines the current configuration identification of all SWPs within it and the supporting operational SP Baseline. The SPR/SCR database, the CVS Library, and/or other variations of CSCI status accounting, may be used to produce the metric reports for the Software Product Center and/or the Pan-STARRS project, as required.

4.11  4.3.1  Configuration Audits

The SQE may perform periodic audits of CM activities and/or products. The purpose of the audit is to verify compliance with all documented CM process directives and to ensure the integrity of the SWPs. The CM ensures that only approved and scheduled changes are included in the baseline and product baseline builds.

A formal configuration audit will be performed by the SQE at the completion of the subsystem acceptance testing.
5  A. Appendix

6  A.1 Acronyms

ADD Algorithm Description Document  
AFRL Air Force Research Laboratory  
AI(s) Action Item(s)  
CCB Configuration Control Board  
CM Configuration Management (Manager)  
CM/DM Configuration Management/Data Management  
CMM Capability Maturity Model  
CMS Configuration Management Specialist  
CSA Configuration Status Accounting  
CSAR Configuration Status Accounting Report  
CSCI Computer Software Configuration Item  
CVS Concurrent Versioning System  
FQT Formal Qualification Testing  
I&T Integration & Test  
IfA Institute for Astronomy  
IPP Image Processing Pipeline  
IPP-SW Image Processing Pipeline Software  
IPP-SP Image Processing Pipeline Software Products  
MHPCC Maui High Performance Computing Center  
NDS Non-Deliverable Software  
Pan-STARRS Panoramic Survey Telescope and Rapid Response System  
Pre-CM Pre Configuration Management Release  
QA Quality Assurance  
SCMP Software Configuration Management Plan  
SCR Software Change Request  
SCRB Software Configuration Review Board  
SDD Software Design Description  
SDL Software Development Library  
SDP Software Development Plan  
SFQB Software Formal Qualification Baseline  
SI(s) Software Item(s)  
SPR Software Problem Report  
SP(s) Software Product(s)  
SQA Software Quality Assurance  
SQE Software Quality Engineer  
SSP Standard Software Process  
STP Software Test Plan  
SU(s) Software Unit(s)  
SVD Version Description Document  
SWP(s) Software Work Product(s)  
TBD To Be Determined
TBS  To Be Specified
UH   University of Hawaii
7 A.2 Glossary

* **Software Development Library (SDL)** The SDL is a controlled storage area for SWPs. SDL is composed of program developed source code, object code, and Non-Developed Software (NDS), which are used within the SP. Other SWPs that require CM control are also maintained in the SDL. Examples are: reference and process tools and documentation.

* **Software Item (SI)** An SI is an aggregation of software designated for CM. A SI is treated as a single entity in the CM procedures. Also referred to as a Computer Software Configuration Item (CSCI), a SI is a super set of SUs or work packages, which are smaller than the embodying SI. Examples are: source code, object code, job control code, control data, or a collection of these items.

* **Software Unit (SU)** A SU is a subset of a SI. A software work package is contained within a SI. As is the case with a SI, a SU is treated as a single entity in the CM procedures.

* **Software Product (SP)** The SP is an executable image and/or source code for delivery to an end user. It is the complete set, or any of the individual items of the set, of computer programs, procedures, associated documentation, and data designated for delivery to a customer or end user.

* **Software Work Product (SWP)** The SWP is any SP(s) that is created in the process of developing the product and is either deliverable or used in the development or validation of a deliverable product. Capability Maturity Model (CMM) definition (information only) - Any artifact created as part of defining, maintaining, or using a software process, including process descriptions, plans, procedures, computer programs, and associated documentation, which may or may not be intended for delivery to a customer or end user.

* **Baseline** The baseline is the configuration identification document(s), and/or SI(s) formally designated at a specific software development life cycle.

* **Configuration Management (CM)** The CM Release is a formal release of the SP that is delivered to the customer. This release may refer to either a developmental release or final release of the CSCI. The CM Release is accompanied by a SVD.
* Pre-Configuration Management

The Pre-CM Release is a release intended for use by the software Integration and Test (I&T) performers. This release is an internal MHPCC software release and precedes a CM Release. More than one Pre-CM release may precede a CM Release.

* Software Quality Assurance (SQA)

SQA involves the entire software lifecycle process including monitoring and improving the process, making sure that any agreed-upon standards and procedures are followed, and ensuring that problems are found and dealt with, as early as possible. It is oriented to processes that result in the 'prevention' of problems and is not to be confused to mean software testing.
### 8. A.3 Software Problem Report Prioritization Categories

<table>
<thead>
<tr>
<th>Severity Level</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>P1 - Critical (Emergency)</strong></td>
<td>A software problem is categorized with this severity level if one or more of the following impact statements apply:</td>
</tr>
<tr>
<td></td>
<td>• Prevents the accomplishment of an operational mission-essential capability.</td>
</tr>
<tr>
<td></td>
<td>• Prevents the operator's accomplishment of an operational or mission-essential capability.</td>
</tr>
<tr>
<td></td>
<td>• Jeopardizes safety.</td>
</tr>
<tr>
<td><strong>P2 - High (Serious)</strong></td>
<td>A software problem is categorized with this severity level if one or more of the following impact statements apply:</td>
</tr>
<tr>
<td></td>
<td>• Adversely affects the accomplishment of an operational or mission-essential capability for which no alternative work-around solution is known.</td>
</tr>
<tr>
<td></td>
<td>• Adversely affects the operator's accomplishment of an operational or mission-essential capability for which no alternative work-around solution is known.</td>
</tr>
<tr>
<td></td>
<td>Note: Program restarts/reboots are not acceptable work-around solutions.</td>
</tr>
<tr>
<td><strong>P3 - Medium (System Degraded - Work-Around)</strong></td>
<td>A software problem is categorized with this severity level if one or more of the following impact statements apply:</td>
</tr>
<tr>
<td></td>
<td>• Adversely affects the accomplishment of an operational or mission-essential capability but a work-around solution is known.</td>
</tr>
<tr>
<td></td>
<td>• Adversely affects the operator's accomplishment of an operational or mission-essential capability but a work-around solution is known.</td>
</tr>
<tr>
<td><strong>P4 - Low (System Not Degraded)</strong></td>
<td>Results in user/operator inconvenience or annoyance but does not degrade a required operational or mission-essential capability.</td>
</tr>
<tr>
<td></td>
<td>• Most documentation changes are considered severity level 4.</td>
</tr>
<tr>
<td><strong>P5 - Trivial (Minor Change)</strong></td>
<td>Any other problem is classified as severity level 5.</td>
</tr>
<tr>
<td></td>
<td>• Most coding standard violations are considered severity level 5.</td>
</tr>
</tbody>
</table>
## A.4 Software Problem Severity Categories

<table>
<thead>
<tr>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLOCKER</td>
<td>Blocks development and/or testing work.</td>
</tr>
<tr>
<td>CRITICAL</td>
<td>Crashes, loss of data, severe memory leak.</td>
</tr>
<tr>
<td>MAJOR</td>
<td>Major loss of function.</td>
</tr>
<tr>
<td>NORMAL</td>
<td>This is the run of the mill bug.</td>
</tr>
<tr>
<td>MINOR</td>
<td>Minor loss of function, or other problem where an easy workaround is present.</td>
</tr>
<tr>
<td>TRIVIAL</td>
<td>Cosmetic problem like misspelled words or misaligned text.</td>
</tr>
<tr>
<td>ENHANCEMENT</td>
<td>Request for enhancement.</td>
</tr>
</tbody>
</table>
## A.5 Software Problem Report Status Categories

<table>
<thead>
<tr>
<th>Bug Status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNCONFIRMED</td>
<td>Nobody has validated that this bug needs to be fixed. Users who have the correct permissions may confirm this bug, changing its state to NEW. You can view your permissions here. A bug may be directly resolved and marked RESOLVED but usually a bug will be confirmed by the person to whom it is assigned. Usually, an UNCONFIRMED bug will be left unconfirmed until someone has verified that the bug the reporter submitted actually occurs.</td>
</tr>
<tr>
<td>NEW</td>
<td>This bug has recently been added to the assignee’s list of bugs and must be processed. Bugs in this state may be accepted, and become ASSIGNED, passed on to someone else, and remain NEW, or resolved and marked RESOLVED.</td>
</tr>
<tr>
<td>ASSIGNED</td>
<td>This bug is not yet resolved, but is assigned to someone who thinks they can fix it. From here bugs can be given to another person and become NEW, or resolved and become RESOLVED.</td>
</tr>
<tr>
<td>REOPENED</td>
<td>The bug was once resolved, but the resolution was deemed incorrect. For example, a WORKSFORME bug is REOPENED when more information shows up and the bug is now reproducible. From here bugs are either marked ASSIGNED or RESOLVED.</td>
</tr>
<tr>
<td>RESOLVED</td>
<td>A resolution has been made, and it is awaiting verification by the QA. From here bugs are either re-opened and become REOPENED, are marked VERIFIED, or are closed for good and marked CLOSED.</td>
</tr>
<tr>
<td>VERIFIED</td>
<td>QA has looked at the bug and the resolution and agrees that the appropriate action has been taken.</td>
</tr>
<tr>
<td>CLOSED</td>
<td>The bug is considered dead, the resolution is correct, and the product the bug has been reported against is terminated or shipped. Any zombie bugs that choose to walk the earth again must do so by becoming REOPENED. This state is rarely ever used.</td>
</tr>
</tbody>
</table>
## A.6 Software Problem Report Resolution Categories

<table>
<thead>
<tr>
<th>Resolution</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FIXED</td>
<td>A fix for this bug is checked into the tree and tested.</td>
</tr>
<tr>
<td>INVALID</td>
<td>The problem described is not a bug.</td>
</tr>
<tr>
<td>WONTFIX</td>
<td>The problem described is a bug, which will never be fixed.</td>
</tr>
<tr>
<td>LATER</td>
<td>The problem described is a bug, which will not be fixed in this version of the product.</td>
</tr>
<tr>
<td>REMIND</td>
<td>The problem described is a bug, which will probably not be fixed in this version of the product, but might still be.</td>
</tr>
<tr>
<td>DUPLICATE</td>
<td>The problem is a duplicate of an existing bug. Marking a bug duplicate requires the bug number of the duplicate and that number will be placed in the bug description.</td>
</tr>
<tr>
<td>WORKSFORME</td>
<td>All attempts at reproducing this bug were futile, reading the code produces no clues as to why this behavior would occur. If more information appears later, please re-assign</td>
</tr>
</tbody>
</table>