Medicaid Management Information System Replacement (MMISR) Project

MMISR Project Defect Management Plan

HSD Deliverable Owner: Karin Stevenson
Deliverable Owner: EPMO
Configuration Number: V1.1
Date: 4/14/2021

Table of Contents

1.0 Introduction 4
1.1 Defect Management Plan Purpose 4
1.2 Defect Management Plan Scope 4
2.0 Defect Management Approach 5
2.1 Defect Management Roles & Responsibilities 5
3.0 Defect Management Tools and Process 7
3.1 Identification, Defect Severity, and Priority Definitions 8
3.2 Escalation and SLAs 11
3.3 Defect Life Cycle 14
3.3.1 Defect Statuses and Resolutions 14
3.3.2 Defect Workflow 16
3.4 Monitoring and Reporting 20
3.4.1 Enterprise Level 20
4.0 Assumptions / Constraints / Risks 21
4.1 Assumptions 21
4.2 Constraints 21
4.3 Risks 22
4.4 Project Issues 22
5.0 Deliverable Development 22
5.1 Deliverable Review Process and Schedule 22
6.0 Appendices 23
7.1 Appendix A: Deliverable Record of Changes 23
7.2 Appendix B: List of Acronyms 23
7.3 Appendix C: Referenced Documents 24
7.5 Appendix D: Guidance for Defect Creation 24

Table of Tables

Table 1 - Defect Process Roles and Responsibilities 5
Table 2 - Defect Severity Level Definition 8
Table 3 - Jira Field Descriptions 9
Table 4 – SLAs for Severity Levels 12
Table 5 - Defect Statuses for MMISR Project 14
Table 6 - Resolution Descriptions 15
Table 7 - Deliverable Review Process and Schedule 22
Table 8 - Deliverable Record of Changes 23
Table 9 - List of Acronyms 23
Table 10 - Referenced Documents 24

Table of Figures

Figure 1 - Example of Comments on logged Defect in Jira 12
Figure 2 - Defect Management Life Cycle as designed in Jira Workflow 17
Figure 3 - Defect Management Lifecycle Business Process Workflow 18

1.0 Introduction

The Defect Management Plan (DMP) will serve as the framework and guide for the management of defects for the New Mexico (NM) Human Services Department (HSD) Medicaid Management Information System Replacement (MMISR) Project for the Health and Human Services (HHS) 2020 enterprise solution. The process as defined in this document applies to the Design, Development, and Implementation (DDI) phases of the MMISR project. Adjustments required for the Maintenance and Operations phase will be developed and agreed upon as that phase is entered for MMISR modules.

For this DMP, a code defect versus a configuration defect will not be treated differently since the flow for resolution is the same. Based on this, the term "Developer" covers both the person implementing code changes as well as configuration changes.

1.1 Defect Management Plan Purpose

The DMP provides the approach, activities, and roles for handling defects identified as part of the HHS2020/Medicaid MMISR Projects. This Initial DMP includes activities associated with the defect management strategy, reporting, preparing for and conducting the triage meeting, and actions related to updating research results and decision review. The plan's primary goal is to clearly identify the necessary defect management activities, durations, and stakeholder expectations and responsibilities. The Initial DMP will span throughout the DDI phase of the project and will not attempt to address the Maintenance and Operation (M&O) phase. This DMP will be used in conjunction with the Module Contractors' individual Defect Management Plans, should those plans be requested under their statements of work (SOW).

1.2 Defect Management Plan Scope

The scope of the HHS2020/MMISR DMP is to provide an overall defect management approach for the MMISR project.

All HHS 2020 Module Contractors must comply with this DMP for integration into the enterprise. The Enterprise Project Management Office (EPMO) will provide oversight, guidance, and monitoring to Enterprise Stakeholders to ensure compliance with this plan. This DMP will be subject to the annual deliverable review cycle and as additional Module Contractors and other agencies of the HHS2020 enterprise are onboarded, this DMP will be expanded to incorporate their testing efforts.

The following audiences are intended for the DMP: the MMISR Project team, which includes, but is not limited to: testers, test managers, User Acceptance Testing (UAT) testers, UAT Testing Manager, Project Managers (PM), application developers, Infrastructure and operations support, system security managers, NM HSD's Information Technology Division (ITD), NM HSD's Medical Assistance Division (MAD), certification, and staff augmentation teams as well as similar personnel from all of the Module Contractors. Module Contractors may have defect management deliverables included in their SOW and contract. The Module Contractor specific DMPs are necessary to ensure that their solutions and software applications meet expectations. This DMP is to ensure that those Module Contractor solutions can be integrated and functional for the MMISR and HHS2020 enterprise system. Both HSD

and Module Contractors are expected to be collaborative in defect management efforts and support defect management process throughout project lifecycle.

2.0 Defect Management Approach

Defect Management is the process to track and appropriately support the timely resolution of defects identified during the testing phases of the MMISR Project. The processes outlined apply during the DDI phase of the MMISR project up to go-live. Upon go-live, contractual obligations, and M&O defect resolutions processes will apply.

The approach for defect management is intended to support the following goals:

- Minimization of defects in deployed code and configurations
- Functional integrity across the system
- System solution meets business needs and supports operational processes

The defect management process has the following characteristics:

- Uniformity and consistency in methodology, tools, documentation, deliverables, management, and reporting processes employed throughout the defect life cycle
- Documentation of all defect-related activities
- A collaborative, professional and responsive relationship between EPMO, HSD, and Module Contractors' personnel and all stakeholders with the common goal of ensuring the system(s) meet the operational and technical requirements for the State of New Mexico

With these goals and processes in mind, Jira is the tool where the goals and processes are executed. For the reporting through Jira, standardized Jira "Gadgets", as well as custom queries will be employed.

With these goals and characteristics in mind, it is important to document the roles and responsibilities of all project personnel who may play a role in the defect management process.

2.1 Defect Management Roles & Responsibilities

The following roles participate in defect management, and a listing of their responsibilities is noted.

Table 1 - Defect Process Roles and Responsibilities

Defect Role	Responsibility
Any MMISR project stakeholder	
Defect Reporter *(may be any member of any team on MMISR project*)	▪ Identifies the defect and documents the problem in Jira with sufficient detail. Please see Appendix D for details. ▪ Selects routing/assigns the defect to the Defect Assessor in the applicable area of the MMISR project ▪ Reporter or designated tester validates defects marked in an "Ready to Test" status ▪ Reporter or designated representative updates Jira with applicable defect status and comments ▪ Reporter or designated representative communicates additional needed details in defect meetings

Defect Role	Responsibility
	▪ Attends either module level or MMISR Enterprise project level defect management meetings
Defect Assessor* **This refers to the person who is assigned to assess the defect in initial or repeated triage.*	▪ Verifies defect details are complete and understandable ▪ Identifies potential duplicate defects already logged ▪ Calculates level of effort associated with the defect ▪ Determines expected solutions and fixes for defects ▪ Routes the defect to a developer if the assessor is not a developer ▪ Reports progress on defect ▪ Updates Jira with applicable defect status and comments ▪ Communicates status of defect work in defect meetings ▪ Attends either module level of MMISR Enterprise project level defect management meetings
Module Contractor Roles	
Module Contractor Defect Lead	▪ Ensures that the defects are worked and routed to the appropriate party in Jira ▪ Updates Jira with applicable defect status and comments ▪ Determines expected resolution for defects to route to appropriate developer ▪ Oversees the defect process at the module contractor level ▪ Communicates status, issues, or additional needs in defect meetings ▪ Attends the module level defect management meeting ▪ Represents module at the MMISR Enterprise project level defect management meeting ▪ Produces defect resolution reports for module contractor and shares with EPMO Defect Manager
Developer	▪ Verifies defect details are complete and understood ▪ Identifies and corrects the source of the defect ▪ Updates the Jira ticket with resolution details ▪ Performs Unit Testing prior to promotion to an SIT testing environment ▪ Routes the defect for testing by an SIT Tester ▪ Updates Jira with applicable defect status and comments ▪ Communicates status, issues, or additional needs to Module Contractor defect lead ▪ Attends the Module Contractor level defect management meeting, if requested by Module Contractor Defect lead
Tester	▪ Documents test scripts/scenarios for defect verification and regression testing as needed ▪ Performs testing of system functionality and specific assigned defects and documents test results with evidence ▪ Upon successful testing, sets Status of Defect to “Done” with Resolution “Fixed” unless it fails testing and requires to be reopened ▪ Performs smoke testing for defects following promotion from development to higher environments ▪ Updates Jira with applicable defect status and comments ▪ Attends the module level defect management meeting, if necessary ▪ Communicates status, issues, or additional needs to Module Contractor defect lead
HSD & EPMO Roles	

Defect Role	Responsibility
HSD UAT Tester	▪ May identify a defect and documents the problem in Jira with sufficient detail (in this role, acting as Defect Reporter) ▪ Selects routing of the defect to the appropriate Module Contractor defects team lead(s) in the applicable area of the MMISR project, if known. If not known, routes the defect to the EPMO Defect Manager ▪ Performs testing of assigned defects following promotions of new code/configuration and/or as a part of defect validation ▪ Updates Jira with applicable defect status and comments ▪ Attends the module level defect management meeting, if necessary ▪ Communicates status, issues, or additional needs to EPMO Defect Manager ▪ Attends the MMISR Enterprise project level defect management meeting, if necessary
EPMO Defect Manager	▪ Oversees the defect management process at the enterprise level ▪ Facilitates the MMISR Enterprise project level defect management meeting ▪ Serves as escalation point for defects between Module Contractors during integration testing phase ▪ Communicates status, issues, or additional needs to Module Contractor defect leads ▪ Updates Jira with applicable defect status and comments ▪ Facilitates module level defect management meetings ▪ Reports on defects to MMISR project leadership
HSD Workstream PM	▪ Attends the module level defect management meeting for assigned Module Contractor ▪ Attends the MMISR Enterprise project level defect management meeting, as requested by EPMO Defect Manager ▪ Communicates status, issues, or additional needs to Module Contractor defect leads ▪ Updates Jira with applicable defect status and comments as applicable ▪ Reviews and provides input to defects to ensure the functionality of all Module Contractor products/services are working as expected and in accordance with the Module Contractor's solution

3.0 Defect Management Tools and Process

The NM HSD Tools Governance Council (TGC) was established in Summer 2020, and the council's purpose is to provide leadership and oversight of the business software tools that support the resource teams in implementing the MMISR project within the HHS2020 initiative. The use of the tools and procedures as they support the Defect Management process has been defined by the TGC. Module Contractors and HSD staff will follow the guidance and procedures outlined by the TGC. This Council may make recommendations and improvements to the configuration and use of the Jira tool that is currently in use for Defect Management cross all Module Contractors.

Each Module Contractor will have access to the NM HSD instance of Jira. A Jira Project is created for each Module Contractor after they have been onboarded to the MMISR and HHS2020 project and are ready to use the Atlassian tools. Jira Projects for the HHS2020 project follow the naming convention "HHS2020 *module abbreviation"* (e.g., "HHS2020 CCSC", "HHS2020 QA").

3.1 Identification, Defect Severity, and Priority Definitions

All Jira projects that include records for defects utilize the default Jira issue type "Bug" to report and manage defects. The terms “Bug” and “Defect” are used interchangeably.

Any member of the MMISR Project team may identify defects impacting the MMISR Project. Identified defects will be documented in the defect tracking tool, Jira, in the Module Contractor’s specific project. Each Module Contractor, including the System Integrator (SI) will have “projects” defined for them within Jira for logging and documenting defects for their solutions and for the integration of those products and solutions into the MMISR project.

The defect classification “Severity” specifies a subjective rating of a defect on the MMISR project and refers to the degree of impact to system functionality. Table 2 - Defect Severity Level Definition provide definitions that have been defined for defect severity along with qualifying conditions as examples of the type of defect and its severity.

The defect classification "Priority" with levels "Lowest," "Low," "Medium," "High" to "Highest" provides a method to express the importance of a fix in the context of resource availability and impact. This classification allows managers as well as business representatives to specify the urgency of addressing the defect beyond the Severity and can serve as tool to prioritize the workload of the development and testing staff.

Table 2 - Defect Severity Level Definition

Severity Level	Severity Definition
Critical	A critical defect in the MMISR platform impacts system security, causes a work stoppage, leads to a program crash, results in security issues, data loss, impedes certification evidence or creates other issues without any viable workaround. Qualifying condition examples may include: ▪ Inability to adjudicate claims ▪ Failure or inability to process a financial cycle ▪ Failure to provide complete eligibility determinations on greater than 0 number of applications within 45 number of days ▪ Failure to generate or populate a Centers for Medicare and Medicaid Services (CMS) or federal report ▪ Any NM HSD defined mission critical condition
Major	A major defect in the MMISR platform impacts core functionality of the MMISR system and/or the Module Contractor’s solution and creates a serious loss of system functionality that requires significant workarounds, users are partially incapable of completing their normal functions and performance of the application is only available with a workaround. Qualifying condition examples may include: ▪ Incorrect claims adjudication ▪ Limited access to Module Contractor system(s) ▪ Inability to meet established timeframes for production data imports, exports, and loading

Severity Level	Severity Definition
	▪ The issue affects a large group of users with a complicated workaround ▪ Providers are unable to access their enrollment and credentialing applications and/or claim histories ▪ Staff are unable to access claims histories or remittance advice reports ▪ CMS or federal reports are generated with missing or inaccurate data
Minor	A minor defect in the MMISR platform creates a limited loss of functionality, causes undesirable behavior, but the system is still functional, and/or a viable workaround is available. Qualifying condition examples may include: ▪ A report is not available but can be generated manually ▪ The issue affects a small subgroup of users with an uncomplicated workaround ▪ Graphical User Interface (GUI) does not display data in preferred format
Trivial	A trivial defect in the MMISR platform has an inconsequential loss of functionality and the impact on users is minimal. The effect on system functions is negligible. The issue is essentially cosmetic in nature or is an aesthetic improvement such as spelling errors or branding issues. Qualifying condition examples may include: ▪ Report incorrectly named ▪ Minor page layout issue ▪ Help Page missing or incomplete ▪ Mouse hover features not triggering text display ▪ Label says Member and should say Client or spelling typo that does not change meaning

The person who identifies a defect and enters the required information into Jira to log the defect is called the *Defect Reporter* (displayed in Reporter field in Jira)

When entering new defects, the Defect Reporter should provide as much information as possible allowing the defect to be resolved efficiently. For more details on logging a defect in Jira, see Appendix D which contains a "How To" procedure guide for creating a defect in Jira.

Key information related to the defect should include, at a minimum, the following recommended fields. Please note, fields are listed alphabetically and do not necessarily represent the order in which they appear on the screen.

Table 3 - Jira Field Descriptions

Jira Field Name	Description
Affects Versions	Jira System field to specify the build/product version a defect was discovered in. The values of the dropdown field are driven by the Jira Version functionality and is configured for Module Contractor's Jira Project
Assignee	This refers to the person who is assigned to work the defect including performing triage It may be a developer, tester, or another project role depending on where in the process the defect is. The reporter may not know

Jira Field Name	Description
	who to assign the defect when reporting the issue. The reporter may leave the defect Assignee as "Unassigned". If Jira Components are utilized (defined on the Project Settings level) default initial assignees can be configured per component.
Attachments	Screenshots or Excel export files that serve to better illustrate the defect, if available
Component	Module area where the defect is found, if known (e.g., Data, Interface, Integration.) If Jira Components are utilized (defined on the Project Settings level) default initial assignees can be configured per component
Description	A complete description of the defect in the body of the Jira ticket. This should include, at a minimum, information about the expected vs. actual results, how to reproduce the defect, and any prerequisites. See Appendix E for further details
Environment	The Environment a defect is discovered in (None, Dev, SIT, PREPROD, and PROD
Fix Versions	Field to specify the build/product version the defect should be/is fixed in. The values of the dropdown field are driven by the Jira Version functionality
Linked Issues	Links to other Jira records such as other test cases, bugs, stories, etc. The Link Type can specify the type of relationship between the records (e.g., caused by, tested by). Links can be established across projects (Bug in one Jira project, Test in another project)
Priority	Priority levels to provide further granularity in the context of business urgency and workload, e.g., Lowest, Low, Medium, High, Highest
Reference	Field that allows the specification of references to other systems, e.g., (Commercial Off the Shelf) COTS system contractor r defect ID when a bug is classified as COTS Defect
Reporter	Person's name or the Jira ID of the user reporting the issue (automatically filled by Jira)
Severity	Based on criteria as defined in Table 2 - Defect Severity Level Definition, Defect Severity Levels include Critical, Major, Minor, or Trivial
Summary	A one-line title of the defect to allow for quick identification of the defect subject
Affects Versions	Field to specify the build/product version a defect was discovered in. The values of the dropdown field are driven by the Jira Version functionality

Jira provides additional helpful fields that can be added to the Bug issue type (e.g., to specify testing phase when the issue was discovered, Root Cause Analysis, Impact Assessment field, Risk Analysis These fields can be added upon the Module Contractor's request, provided they are approved by the TGC.

3.2 Escalation and SLAs

The cooperation between the Module Contractors and the EMPO Defect Manager not only determines a common understanding of the severity and priority of discussed defects but also serves as the forum to agree upon the origination points of detected and reported defects.

The EPMO Defect Manager also serves as an escalation point between Module Contractors and other stakeholders to support effective prioritization and resolution of defects. The Defect Manager strives to detect that the defects are being addressed at the enterprise level. At the module level, it is the responsibility of the Module Contractor Defect Lead, to ensure that the defects are worked in a timely manner and routed to the appropriate party in Jira and to monitor the progress of the defects to resolution. Each Module Contractor will have a dedicated Jira Project in which to log their defects and work them, generating queries and reports that are relevant. Due to the unified Jira configuration, Bug records can be moved to a different project if needed.

Defect and Issue Review meetings will be held weekly, at a minimum, and defects recorded in Jira will be reviewed in order of severity level from “Critical” to “Trivial.” Escalation of defects outside the meetings will also occur to support project progress. Module Contractor Defect Leads are responsible for ensuring accurate information is available to the project team.

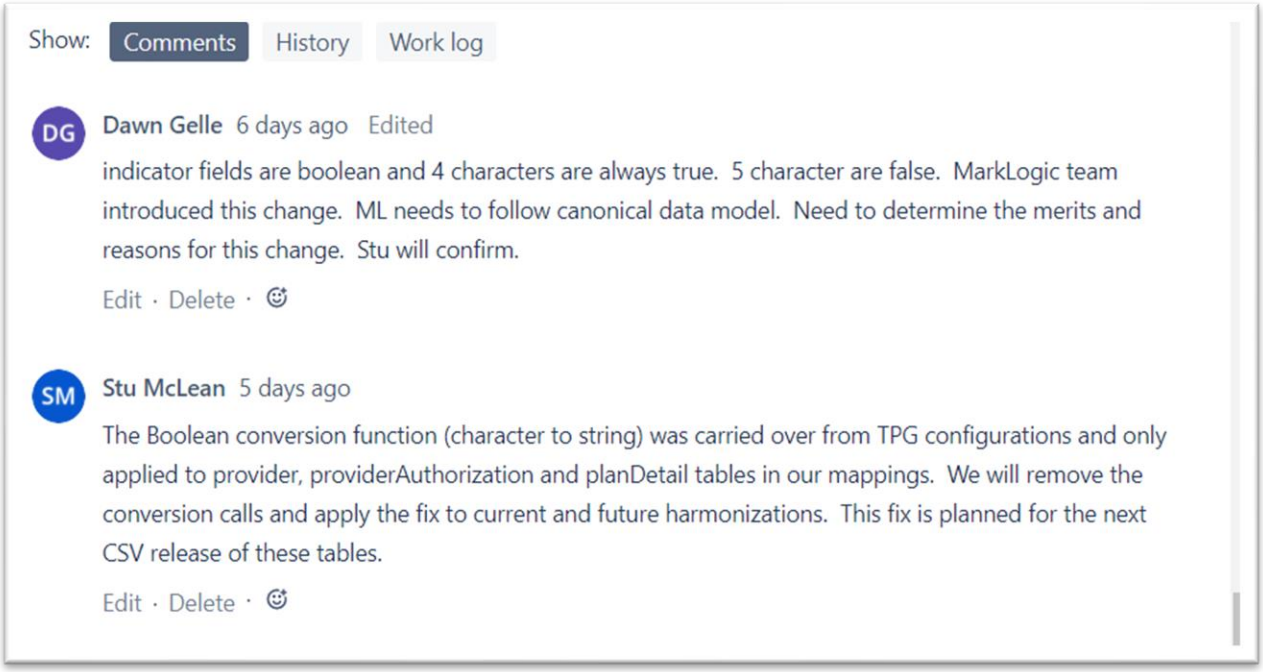
Progress reporting is achieved by the Module Contractor team entering detailed comments in Jira, in advance of the scheduled Defect and Issue Review meeting. The meeting will focus on defects that need further clarification. Verbal updates will be provided during the meeting for the defects that have not been reported timely. The Module Contractor Defect Lead will review the Jira tickets prior to each meeting and provide a status update on the team’s assigned defects. If defects are not resolved or do not have a plan for resolution within the Service Level Agreement (SLA) timeframe, the Defect Lead will escalate to the EPMO Defect Manager for inclusion in the meeting agenda and review at the weekly meeting.

The SLAs specified in Table 4 - SLAs for Severity Levels define the time frames for resolving defects and deploying the solution which does not include the verification by the reporter’s team in case of cross-module defect reports. Verification of solutions and fixes by the reporter or designated tester need to be performed as soon as is feasible. If the verification of a defect is delayed and impacts progress on the enterprise level, it should be escalated to the EPMO Defect Manager for further action.

When a defect is reported, the Assessor responds by submitting details as to what is needed for resolution within the time allotted in the SLA. The detailed steps for resolution need to be documented and may or may not include a Level of Effort (LOE) and expected resolution date, as these items may not be known at the time of initial defect review. The expectation is that the assessor will document their assessment, supply questions that need clarification, and provide a summary status of issues and fixes through Jira Comments.

Progress on resolving the defect must be demonstrated through documented entries on the Jira ticket in the Activity Comments section. An example of issue identified and plan for resolution of the defect is depicted below. Note that the exact date/timestamp is visible when hovering the mouse over the elapsed time since the comment was added.

Figure 1 - Example of Comments on logged Defect in Jira



Defects with the following Severity Levels, the MMISR Project will be establishing SLA timeframes for each severity level detailed in Table 4 – SLAs for Severity Levels. As noted in the introduction, this process is serving the DDI phase. Defect resolution SLAs for the M&O phase will differ from the ones specified in Table 4 - SLAs for Severity Levels.

Table 4 – SLAs for Severity Levels

Severity Level	**Severity Description**	**Priority to Business User**	**SLA for Triage & Analysis:** ***Submit Plan or Update Jira defect status***	**SLA for Resolution:** ***Progress demonstrated before Escalation to EPMO Defect Manager & HSD PM***
Critical	A critical defect in the MMISR platform impacts system security, causes a work stoppage, leads to a system crash, results in data loss, impedes certification evidence or creates other issues without any viable workaround.	Immediate Attention	One (1) business day	Two (2) consecutive Defect Review meetings without progress
Major	A major defect in the MMISR platform impacts core functionality of the MMISR system and/or the Module Contractor's solution and creates a serious loss of system functionality that requires	High Attention	Three (3) business days	Two (2) consecutive Defect Review meetings without progress

Severity Level	Severity Description	Priority to Business User	SLA for Triage & Analysis: *Submit Plan or Update Jira defect status*	SLA for Resolution: *Progress demonstrated before Escalation to EPMO Defect Manager & HSD PM*
	significant workarounds, users are partially incapable of completing their normal functions and performance of the application is only available without a workaround.			
Minor	A minor defect in the MMISR platform creates a limited loss of functionality, causes undesirable behavior, but the system is still functional, and/or a viable workaround is available and does not affect production.	Normal Attention	Five (5) business days	Four (4) consecutive Defect Review meetings without progress. If meetings have not included minor defects, then inclusion of the number of minor defects should be included in a Delivery Iteration report to the State along with a timeline for resolution
Trivial	A trivial defect in the MMISR platform has an inconsequential loss of functionality and the impact on users is minimal. The effect on system functions is negligible. The issue is essentially cosmetic in nature or is an aesthetic improvement such as spelling errors or branding issues.	Low Attention	Thirty (30) business days	Number of Trivial defects will be included in a Delivery Iteration Report to HSD with a timeline for resolution

For example, if no progress is noted in Jira and/or reported for two (2) consecutive Defect Issue and Review meetings for a “Critical” defect, the EPMO Defect Manager will escalate to the HSD workstream lead (e.g., data workstream lead) responsible for that Module Contractor and a determination about next steps will occur. The specific unresolved defect or group of unresolved defects will become a part of the Risk Management process and managed towards resolution through the Risk and Issues process. Contractual remedies and contractual SLAs may apply to the specific Module Contractor and the HSD Workstream Project Manager may choose to exercise those options as well.

The defect classification "Priority" with levels "Lowest," "Low," "Medium," "High" to "Highest" provides a method to express the importance of a fix in the context of resource availability and

impact. This classification allows managers as well as business representatives to specify the urgency of addressing the defect beyond the Severity Level and can serve as tool to prioritize the workload of the development and testing staff.

3.3 Defect Life Cycle

The defect lifecycle captures the transition phases of a defect during its lifetime. It starts when a defect is identified and ends when a defect is closed. Defects will be reported, tracked, classified, communicated, and managed within the Jira tool using the process mapped in the Defect Workflow described in section 3.3.2. Closely related to the Test Management Plan (TMP) (PMO14), this process will be subject to a regular review cycle, synchronized with the PMO14 deliverable annual review cycle, to ensure it accurately captures the defect lifecycle and project changes which may have occurred. Where improvements are needed or suggested, the Defect Workflow process will be updated to coincide with updates made to Jira's Bug workflow detailed in Figure 2 - Defect Management Life Cycle.

3.3.1 Defect Statuses and Resolutions

The defect life cycle is governed by the following status settings in Jira. The statuses and their descriptions listed below in Table 5 - Defect Statuses for MMISR Project are the standard set of defect statuses that will be used for the MMISR project. Individual Module Contractors may request to have additional or modified statuses available to them for the purposes of completing their project development work. To ensure process uniformity across the MMSIR project, such modifications are discouraged and need to be presented to and approved by the Tool Governance Council (TGC) before implementation.

Table 5 - Defect Statuses for MMISR Project

Defect Status	Description
Triage	The defect is newly logged in Jira by the Defect Reporter. This is the initial status of the defect in Jira prior to assignment to the appropriate Module Contractor's or other team members. During this status, the defect's completeness and description is reviewed.
In-Progress	The defect has been triaged and assigned to a developer for assessment and defect work in upcoming development sprints.
Promote to SIT	The defect or solution has been worked to resolution and is ready for deployment for Testing in the SIT environment.
In SIT	The defect fix or solution has been deployed to the SIT environment and assigned for testing to Module Contractor or HSD first level tester. **All defects that require production data for full verification do need to pass through testing in the Pre-Prod environment**
Promote to Pre-Prod	When a defect requires UAT: The defect has passed SIT testing with the available test scenarios and is ready to be promoted to the Preprod environment for HSD UAT testing.
In UAT (used by HSD UAT Testers only)	The defect fix or solution has been deployed to the Preprod environment and has been assigned to an HSD UAT tester for verification.

Defect Status	Description
Done	The defect fix or solution has passed verification (through SIT or SIT and UAT) and can be closed. The Resolution value for Bugs in status Done is "Fixed"
Rejected	The defect has been rejected by Assessor or developer. The reason for rejection is expressed through the Resolution value, e.g. Not a Bug, Duplicate, etc. A full list of available Resolution values with descriptions for the Rejected status can be found in Table 6 - Resolution Descriptions below.
Blocked	The defect is blocked by an internal team member, state user, or external entity and progress cannot occur on resolving defect.

The path for defects follows: Triage -> In-Progress -> Promote to SIT -> In SIT -> Promote to Pre-Prod -> In UAT -> Done. See Figures 2 and 3 below.

If a designated tester that is not the Reporter marks the defect as Done but the Reporter subsequently finds an issue, the Reporter can reopen the bug by setting it back into "Triage" and providing a comment to explain the reason. Please refer to Defect Workflow in Section 3.3.2.

To prepare comprehensive reports of defects that are "Awaiting Verification", both the "In SIT" as well as "In UAT" status values should be included in the Jira query (e.g., "project = "xxx" AND Issue type = Bug AND Status IN ("IN SIT, "IN UAT")).

A code or configuration change suggested by the Triage Assessor or Developer working the defect might require action through the Change Control Management Plan (CCMP) for logging a technical change request with the Technical Change Review Board. Such defects need to be escalated through the EPMO Defect Manager as part of the Defect Review process. For additional details on the CCMP, please refer to Appendix C for a link to the Change Control Management Plan.

A defect that has been marked with a status of "Done" or "Rejected" is further identified with a Resolution Type value. Table 6 - Resolution Descriptions provides a list of Resolution Type values with a description of each. Note, that whenever a defect is reopened by moving it to status "Triage", any Resolution values previously set while closing the defect as Done or Rejected is automatically removed.

Table 6 - Resolution Descriptions

Resolution Type	Description
Duplicate	A duplicate defect should first be linked to its duplicate by selecting Link Type "duplicates" or "duplicated by" and then setting its status to "rejected" as noted in Table 5. The Resolution Type can then be set to "Duplicate." . Duplicates need to be discussed in the Defect and Issues Review meeting.
Cannot Reproduce	The defect identified cannot be reproduced. This resolution can be set when selecting the Jira status "Rejected". Comments need to be added to detail the findings and Reporter approval for this classification.
Not A Bug	The system is functioning as designed and the defect is determined as not a bug. This resolution can be set when selecting the Jira status "Rejected". Comments need to be added to detail the finding and Reporter approval for this classification

Resolution Type	Description
Won't Fix	This resolution type is used when defects are reported that will not be resolved for various reasons. This resolution can be set when selecting the Jira status "Rejected". Comments need to be added to detail the reasons and Reporter approval for this classification.
COTS Defect	When a defect has been determined to be located in a COTS product, the defect should be set to Jira status "Rejected" with Resolution "COTS Defect" after reporting the defect the COTS vendor. Any reference numbers for the report should be entered into the defect field "Reference". Comments need to be added to detail the findings and Reporter approval for this classification.
Fixed	The issue is considered resolved and fixed, the resolution (e.g., code, configuration) is complete, correct, and verified. This Resolution value is only available when setting the Jira Status to "Done".

3.3.2 Defect Workflow

The following two (2) process workflows depict the lifecycle of a defect transitioning through various statuses. The first figure is as the workflow is designed in Jira. The second figure is the workflow (shown using swim lanes) and may best be used by business users. Updates to the Jira Bug workflow for managing defects need to be presented to and approved by the Tools Governance Council and these workflow process diagram will be updated at that time.

In Jira workflows, the functionality is encapsulated in Statuses and Transitions. The color of the Status indicates a rough classification of the status. Gray indicates "To Do", Blue indicates "In Progress", and Green indicates "Done". The connecting arrows between Statuses represent the available transitions from Status to Status that Jira enforces. Transitions labeled as "All" indicate that the Status can be entered without restrictions from any other status in the workflow. Note that Transitions can contain functionality that is not visible in the diagram. For example, a transition can automatically open a screen to ask the user to enter a comment, adding or removing Resolutions. The Jira Bug workflow automatically asks for the entry of comments for all transitions and removes any Resolution values that might be present when a defect is moved to status Triage which implies "unresolved".

Figure 2 - Defect Management Life Cycle as designed in Jira Workflow

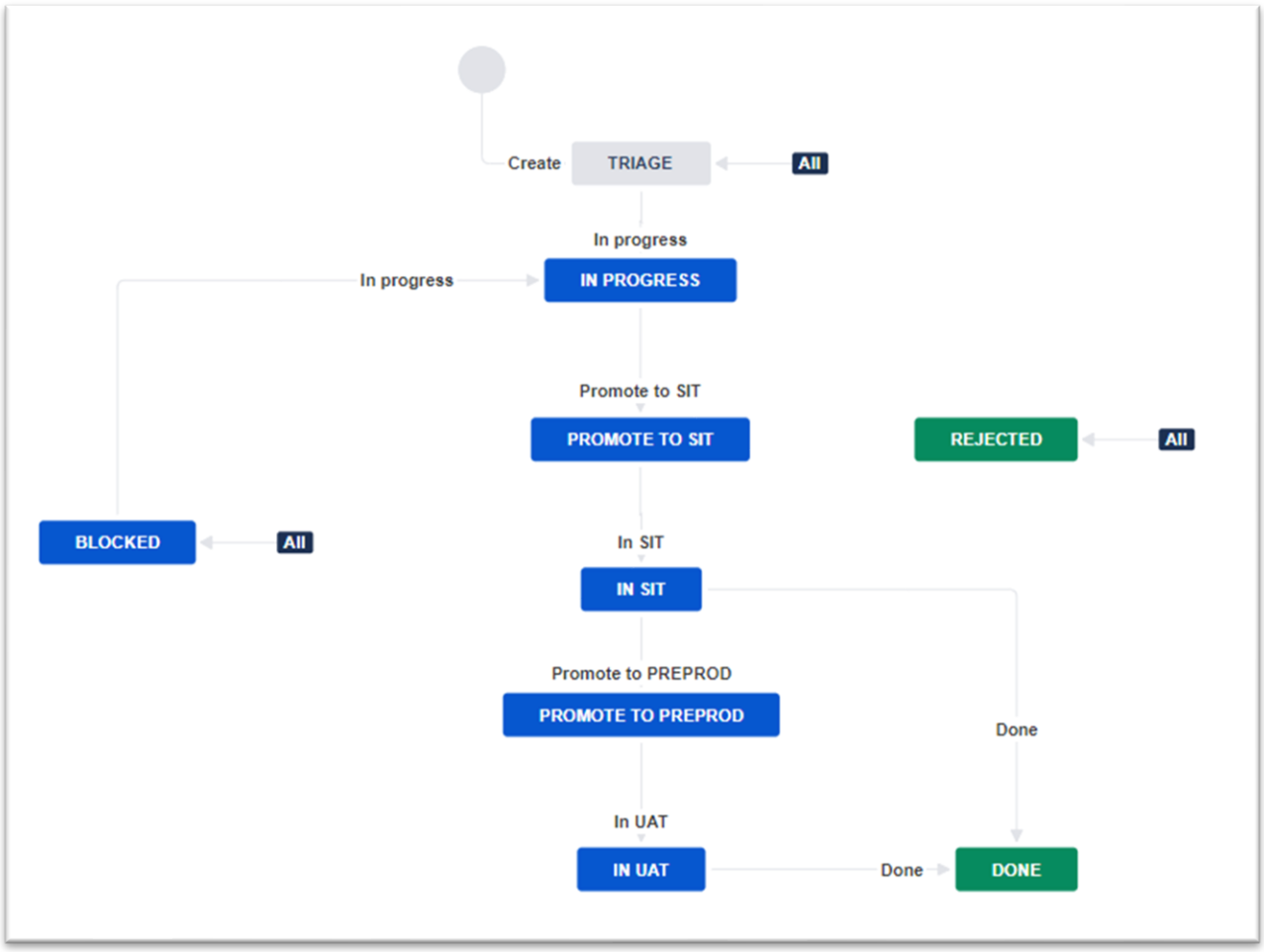
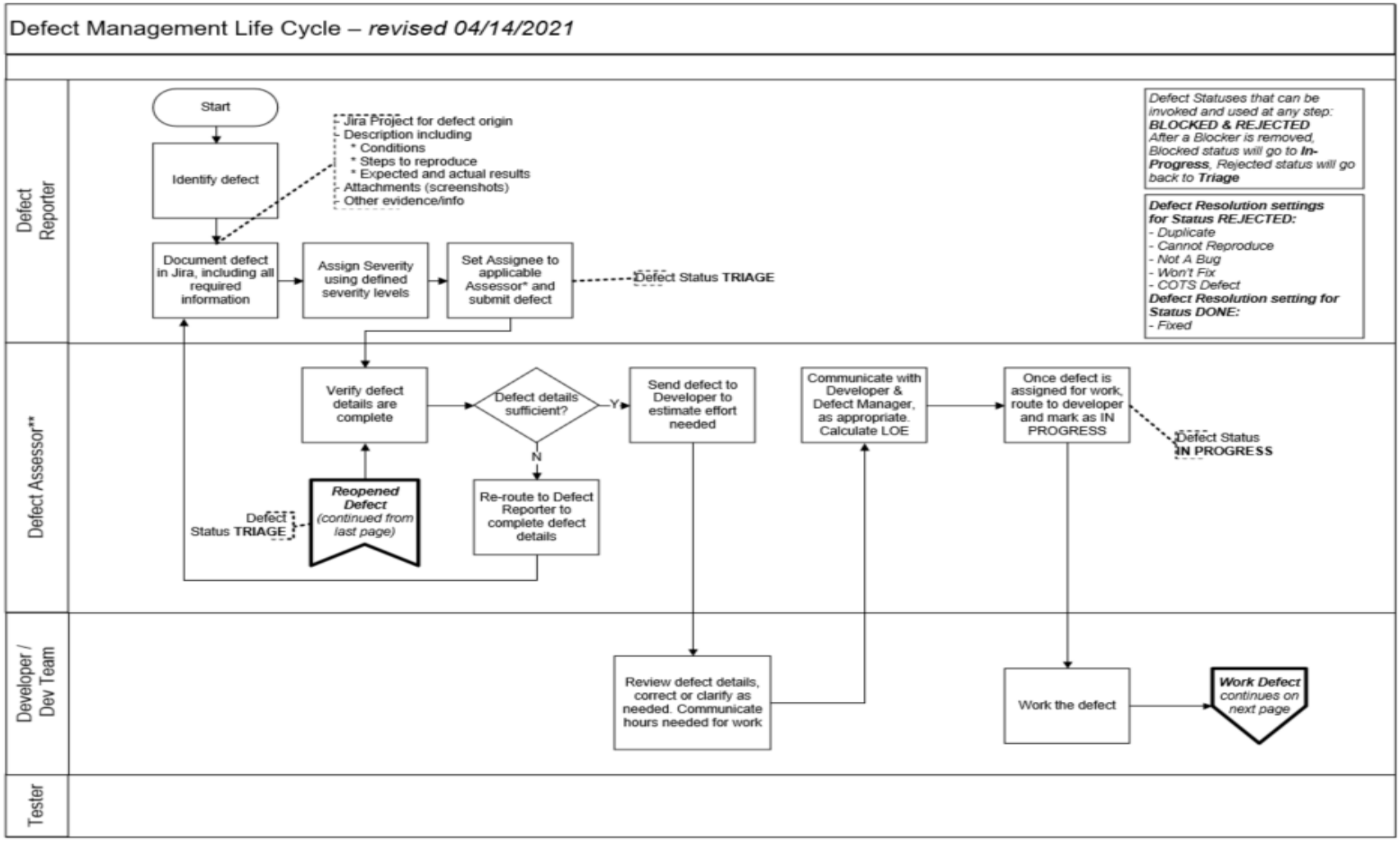
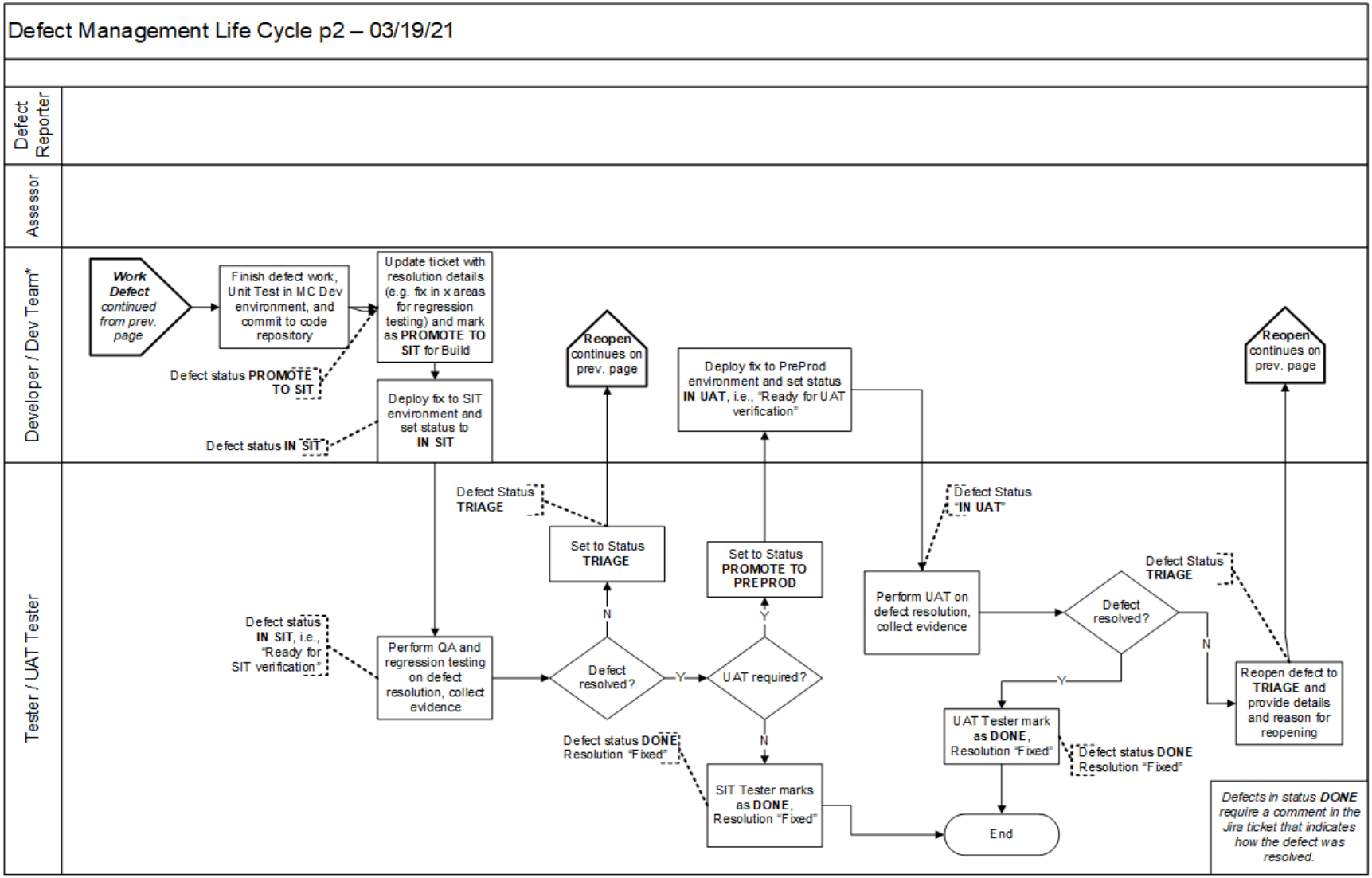


Figure 3 - Defect Management Lifecycle Busines Process Workflow



**Defect *Assessor may be a developer, tester or module contractor defect lead, depending upon status of defect at a given time*



*Dev Team might be Developer(s), Build Master, etc.

3.4 Monitoring and Reporting

Defect Reports will be used to measure compliance with defect SLAs, throughput of defects, and defect resolution quality (in compliance with PMO14. At this time, defects are monitored via Jira with reviews conducted in advance of the regularly scheduled meetings. Formal review is conducted during the Defect and Issue Review meeting.

Time constraints and/or substantial increase in defects may require additional meetings to be scheduled to ensure timely review and updates of the project defects. Issues or concerns regarding defects or the defect management process will be escalated in the MMISR Project Management Office (PMO) Weekly Meeting. Standardized report sets for each Module Contractor and the integration testing work will be developed or configured in Jira. Metrics will be reported as part of the Leadership Meetings and through the EPMO Monthly Status Reports.

At minimum, all module contractors will be expected to produce defect reports from Jira via standard reports or filter-controlled gadgets on Jira Dashboards. Depending upon the circumstances, the following or additional reports will be requested with agreed-upon report criteria:

1) Average Age Reports for defects:
    a. Report that shows age of open defects.
    b. Report showing defect age information within statuses
2) Defects created vs. Defects closed
3) Pie Chart reports for defects (based on status, components, etc.)
4) Recently created defects

Queries and reports will continue to evolve over the project's lifecycle. HSD can foresee the need for standardized filter-based templates and or filter-controlled gadgets that could be used by Module Contractors. The above list of reports should be considered the minimum set of reports.

3.4.1 Enterprise Level

The EPMO Defect Manager for the MMISR Project is responsible for monitoring and tracking defect resolution. Management involves the creation of dashboards and providing real-time visibility into the defects identified within the MMISR project.

The Defect Manager serves as an escalation point between Module Contractors and supports effective prioritization and resolution of defects.

- The first level of risk escalation includes discussions involving the Module Project Manager, the Defect Reporter, and the Assessor. The results of these discussions and next steps are actionable decisions and the creation of plans to support the defect resolution effort which are recorded in the Jira comments for the Defect.
- If defects are not resolved at the first level or do not have a plan for resolution within the SLA timeframe, the Defect Manager will escalate within the EPMO and the defect will become part of the Risk Management process with a new risk (or issue) logged for the Project.
- If a defect has a known impact on certification evidence, the severity level will be set to Major or Critical, depending upon severity, and prioritized for resolution.

Each Module Contractor is expected to have a primary representative attend the weekly Defect and Issue Review meetings and take ownership for any assigned defects that require work by the Module

Contractor. Further, this meeting's purpose will be to conduct a review of Critical and Major defects impacting the Enterprise and integration work and set priority for enterprise-wide defects.

4.0 Assumptions / Constraints / Risks

4.1 Assumptions

The following assumptions have been identified for the MMISR defect management approach and plan:

- NM HSD resources will be available to participate in the discussions, follow up meetings, and email conversations that are needed to support the defect management plan in the identified timeline for the plan.
- NM HSD owns and maintains all the tools used for the implementation of HHS2020 (except for Module Contractor tools provided through their solutions).
- NM HSD TGC may assess, plan, and approve changes to the tools leveraged to support the project.
- Any deviations by any Module Contractors in using this toolset for commonly identified tasks require proper justification and approval from the Tools Governance Council (TGC). All participating Module Contractors will adopt the MMISR testing methodology/phases as appropriate and ensure that their test plans are in alignment with MMISR TMP as described in the EPMO PMO14.
- NM HSD, MMISR Module Contractors, and EPMO will come to a common agreement to follow the defect management plan and processes described in this Plan and will adhere to the roles and responsibilities outlined in this DMP.
- All participating Module Contractors will adhere to the applicable State and Federal standards.
- All participating modules in the HHS 2020 enterprise solution workflow will adhere to the security standards specified in the System Security Plan (SSP).
- All MMISR stakeholders will adopt and use Jira as the NM HSD selected tool for defect management.
- Lack of adherence to the DMP will result in potential quality issues and delays to the project. Defects that are not logged or are missing sufficient detail will not be fixed promptly or may not be fixed at all.
- This DMP relies upon standard Jira terminology and Jira familiarity is assumed which is part of the onboarding process

4.2 Constraints

The following constraints have been identified for the defect management approach and plan:

- Subject Expertise: As the HHS 2020 enterprise solution components are Application Programming Interface (API) driven, SIT testers that are assigned to verify defect fixes need to have at least the minimum subject matter expertise required to verify and validate the business workflows and technical details of fixes. UAT Testers require the expertise to validate the business workflows.
- COTS Products: Unlike custom built components, COTS products come with their own restrictions, some of which may limit interoperability among integrating modules, but which

may not be critical if the COTS products are deployed in support of a Module Contractor's solution. Conversely, COTS products should have an established and production-ready code base that reduces the number of defects discovered for the MMISR platform implementation.

- System Integration: Use of mocking mechanisms is required, as the modules are being developed in different timelines and on separate schedules. This fact will delay the effective testing of the entire enterprise solution until the integration stage. Consequently, integration-related defects may not be discovered until that stage of project lifecycle.

4.3 Risks

The Risk Management Plan (RMP) describes the management of project risks and updates for all areas of the project. The risk register is located on SharePoint.

A link to the Risk Log and the Risk Management Plan can be found in Appendix C of this DMP.

4.4 Project Issues

The RMP also describes the management of project issues and updates for all areas of the project including the TMP. The Issue Log is located on SharePoint.

A link to the Issue Log and the Risk Management Plan can be found in Appendix C of this DMP.

5.0 Deliverable Development

5.1 Deliverable Review Process and Schedule

This Defect Management Process is considered an external addition of the PMO14 as referenced in the PMO14 section 3.1, Defect Management.

Table 7 - Deliverable Review Process and Schedule

Description	Target Completion Date	Participants
EPMO revises the previously approved defect process and creates a new version as a draft, to reflect the EPMO changes and other needed revisions, performs an internal quality assurance check and submits to HSD for review and approval.	2/15/2021	EPMO
HSD reviews and approves the draft deliverable. If content changes are identified, HSD documents changes via in-line comments. The EPMO will conduct a deliverable walkthrough of these sections and any updated sections with HSD.	3/1/2021	NM HSD
EPMO reviews comments, applies edits as appropriate, responds via in-line comments, and resubmits the deliverable to HSD.	3/24/2021	EPMO
HSD reviews final and Final Deliverable Approved	4/72021	NM HSD

6.0 Appendices

7.1 Appendix A: Deliverable Record of Changes

The deliverable will include a record of changes in the following form:

Table 8 - Deliverable Record of Changes

Version Number	Date	Author/Owner	Description of Change
V1.0	1/21/2020	Joan Callahan	Development of the Defect Process as an Appendix
V1.0	2/15/2021	Dawn Gelle	Submittal of Defect Management Plan, inclusive of revisions of Defect process to a Defect Management Plan as a result HSD review of PMO14: Test Management Plan and design of Bug workflow within Jira tool
V1.0	3/5/2021	Karin Stevenson	Review/Rework to align plan with Jira system and latest environment decisions
V1.1	4/14/2021	Karin Stevenson Dawn Gelle	Updates based on HSD feedback

7.2 Appendix B: List of Acronyms

A list of project-specific acronyms will be maintained on the MMISR SharePoint site.

Table 9 - List of Acronyms

Acronym	Definition
API	Application Programming Interface
CCMP	Change Control Management Plan
CMS	Centers for Medicare and Medicaid Services
COTS	Commercial Off the Shelf
DDI	Design, Development, and Implementation
DMP	Defect Management Plan
EPMO	Enterprise Project Management Office
HHS	Health and Human Services
HSD	Human Services Department
ITD	Information Technology Division
LOE	Level of Effort
M&O	Maintenance and Operation
MAD	Medical Assistance Division
MMISR	Medicaid Management Information System Replacement
NM	New Mexico
PMO	Project Management Office
QA	Quality Assurance

Acronym	Definition
RMP	Risk Management Plan
SLA	Service Level Agreement
SOW	Statement of Work
SSP	System Security Plan
TGC	Tools Governance Committee
TMP	Test Management Plan
UAT	User Acceptance Testing

7.3 Appendix C: Referenced Documents

The following is a list of documents references in this plan. Access to the links are based on SharePoint permissions.

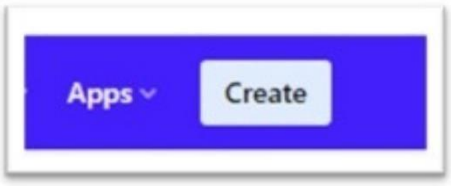
Table 10 - Referenced Documents

Document	Link
Risk Management Plan (PMO7)	Risk Management Plan
Requirements Traceability Matrix (PMO16)	Requirements Traceability Matrix
Change Control Management Plan (PMO10)	Change Control Management Plan
System Security Plan	System Security Plan
Test Management Plan (PMO14)	Test Management Plan
Issue Log	Issue Log
Risk Log	Risk Log

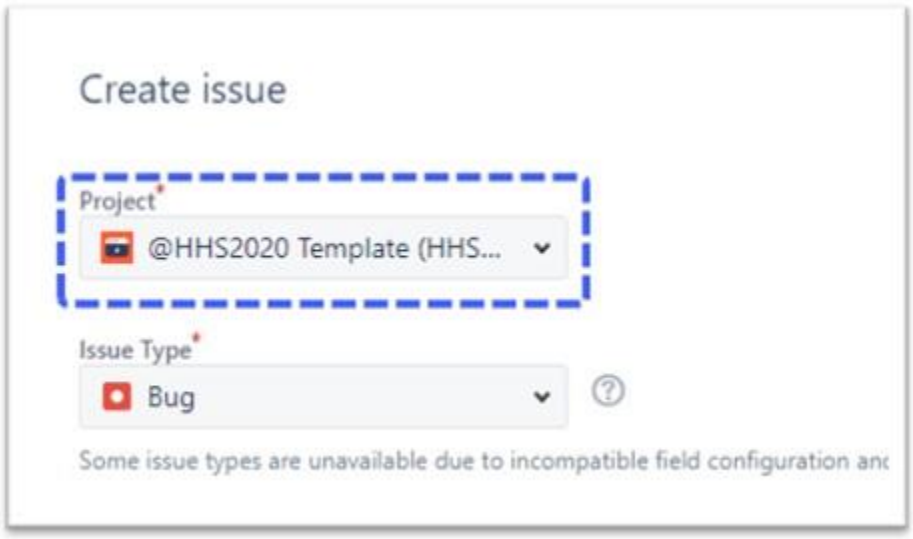
7.5 Appendix D: Guidance for Defect Creation

The steps listed below can serve as a guide for defect reporters learning how to create a defect in Jira.

1. Defects can be created from any project or screen in Jira by clicking on the Create button in the blue main menu bar at the top:



2. Select the project for which the defect should be entered:



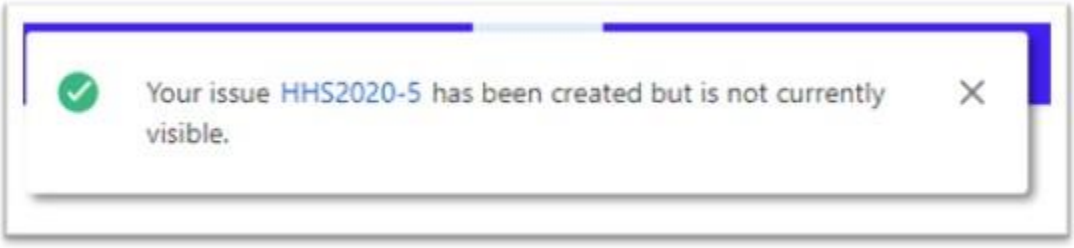
3. Select “Bug” as Issue Type

4. Provide the following information in the fields of the Jira Bug Create screen after the Bug Create screen is displayed. Note that the fields in the screen differ from the fields in other issue types so it is imperative to select the Bug issue type.
    - **Project:** Verify the field is showing the project for which the defect needs to be filed. The Bug can be filed from any other project in Jira when the proper target project is selected.
    - **Issue Type:** Verify the Issue Type field shows “Bug” if not, select it from the dropdown menu.
    - **Summary:** Title of the Bug – make it clear, short, and concise.

- **Affects versions:** Enter the version of the software deployment the defect was found in. The version numbers available for selection are defined in the "Versions" option of the project.
- **Components:** Select one or more of the system components that apply to the defect.
- **Description:** The Description field should contain at least the following information:
  - Detailed Description of the Bug
  - Prerequisites (e.g., version, special permissions, needed to reproduce the defect, browser type, etc.) to reproduce the defect
  - Steps to reproduce the defect
  - Expected Result of functionality (or test)
  - Actual Result of functionality (or test)
- **Severity**: Set the Severity of the defect as it impacts the system functionality. "Major" is set as a default. Defect Severities are listed in Table 2 - Defect Severity Level Definition of this document.
- **Environment:** Select the environment in which the defect was discovered (Dev for development, SIT for Functional QA Testing, Pre-Prod for UAT and other preproduction testing, Production for any Production issues)
- **Attachment:** Add all attachments that clearly support the details of the defect e.g., screenshots.
- **Linked Issues and Issue:** Enter the test case(s) and User Story that are related to the defect, if applicable. The new bug can be linked to issues in other projects, if applicable.
  - Linked Issue for Test Cases = 'Tested By'
  - Linked Issue for User Story = 'Blocks'
- **Assignee:** If known, select the Assessor designated for starting the review and Triage of the defect.
- **Reference:** If known and applicable, enter reference to a related system, e.g., bug number of COTS vendor.

5. Click the 'Create' button.

6. Jira will display a transient notice that the bug was created for easy access:



The screen the user is one will NOT change to display the new bug.

NOTE:

- When displaying the newly created Bug record, additional fields become available that are utilized when working the defect to completion through its life cycle. Among others these are "Fix Version", "Priority", "Epic Link", and any additional custom fields.

- If a new Bug is filed through a Jira Xray Rest Run step, several of the information elements needed for “Description” field are prefilled and formatted. The Bug will also automatically receive the relevant links to the Test and its’ Text Execution record with the proper Link Type.