Standard Operating Procedure (SOP)
BRICS Software Development Process

Document Information

Document Owners: Matthew McAuliffe, Yang Fann, and Dominic Nathan

Organizations: NINDS DIR ITBP, CIT OIR ISL BIRSS, CNRM

Approval / Distribution Process

The SOP approval/distribution process is as follows:

1. The SOP author sends the SOP SharePoint link to their peers/subject matter experts (SMEs) for review.
2. After editing, the SOP author decides whether the SOP is ready for approval. If the SOP is ready, the author adds the SOP to the ITBP Manager meeting agenda.
3. At the ITBP Managers meeting or via email, NINDS Management formally approves/disapproves the SOP.
4. The SOP Author sends the SOP link to all people on the Distribution List.

NINDS Approval

This Standard Operating Procedure (SOP) is approved for distribution and implementation as of the Director ITBP approval date listed below. NINDS ITBP management is authorized to conduct periodic audits to ensure compliance with this procedure. Requests for corrections or changes to any part of this procedure must be submitted to the Document Owner to review. Exceptions to any procedure must be approved by the ITBP Management and documented.

Approved By:

Name	Title	Organization	Approval Date
Yang Fann	IT Director	NINDS DIR ITBP	03/28/19
Matthew McAuliffe	BIRSS Chief	CIT OIR ISL BIRSS	03/28/19

Name	Title	Organization	Approval Date
Dominic Nathan	Informatics Core Director	CNRM	03/28/19
Mark Edwards	IT Manager	NINDS DIR ITBP	03/28/19
Willy Calderon	ISSO	NINDS DIR ITBP	03/28/19

Peer Reviewers

This Standard Operating Procedure was reviewed by the peers (i.e., subject matter experts) listed below. The procedure will be reviewed by the peer reviewers at least annually.

Reviewed By:

Name	Title	Organization	Date
Tsega Gabremichael	Team Lead	CIT OIR ISL BIRSS	03/27/19
Leonie Misquitta	Sr Scientific Advisor	CIT OIR ISL BIRSS	03/27/19
Dominic Nathan	Informatics Core Director	CNRM	03/27/19

Distribution List

This Standard Operating Procedure impacts the individuals on this Distribution List. The SOP author should notify everyone on this list about changes to this SOP *within one week* of NINDS approval.

Distributed To:

Name / Department / Group / Team
Yang Fann
Matthew McAuliffe
Dominic Nathan
Willy Calderon

1. Introduction

1.1 Overview

This document presents the Standard Operating Procedure for Software Development Process at NINDS. The intended audience for this procedure includes the groups/individuals listed below:

- NINDS DIR Clinical Informatics Development Team
- CIT OIR ISL BIRSS Development Team

NINDS also developed the DIR Application Development Best Practices SOP to address the NIST 800-53 security control requirement(s) stated below:

- SI-10 (1) and (2) Input Validation
- SI-11 (a) and (b) Error Messages
- DM-3 Minimization of PII Used in Testing, Training, and Research.
- DM-3 (1) Risk Minimization Techniques.

1.2 Purpose

The purpose of this Standard Operating Procedure is to document the approved set of written instructions for Agile software development projects, from requirements development through to the delivery and maintenance of the project results at NINDS to address specific business needs and security controls.

1.3 Scope

This Standard Operating Procedure is applicable to the collection of components that comprise custom software development of the BRICS and its associated systems such as CiSTAR, CASA, and ProFoRMS at NINDS and CIT.

1.4 Roles and Responsibilities

The following personnel are primarily responsible for performing the procedures:

Name	Title	Responsibility
Clinical Trial Unit	NINDS DIR CTU	NINDS Governance committee for approvals
Steering Committee	Informatics Core	CNRM Governance committee for approvals

Name	Title	Responsibility
Yang Fann	BRICS Co-Director NINDS IT Director	Authorizing Official to operate Approve requirements
Matthew McAuliffe	BRICS Co-Director CIT BIRSS Chief	Approve requirements
Dominic Nathan	Informatics Core Director	Manage the project
Leonie Misquitta	Sr Scientific Advisor	Provide scientific consulting
Tsega Gebremichael	Sr Software Engineer	Provide technical guidance
Change Control Board	Subject Matter Experts	Manage and approve change requests and system enhancements
Business, Product owner, Instance Program Manager	Key Stakeholders	Review and validate requirements and work products
NINDS/CIT Clinical Informatics Development team	Software Engineer	Responsible for understanding and following the scrum development processes outlined in this document.

1.5 System Operational Status

BRICS/CiSTAR/CASA Clinical Trial Software are currently in the operational (production) phase in accordance with the system development life cycle (SDLC) as defined in the NIST SP 800-18.

1.6 Definitions

The following definitions may assist in understanding this SOP.

- Software Development – Activities that production or enhance software and its documentation, including the testing and distribution of software that is developed at NIH.
- Requirements - A list of expected results for the project, stated as the minimum functionality and performance that is acceptable.
- Deliverable – The project results to be delivered to the customer.

- Task – A software development activity, intended to satisfy a portion of the requirements, which is assigned to a developer.
- Jira – The Web-based issue tracking system used by NINDS and CIT.
- Confluence – The collaboration software program where team members can create, organize and discuss work with the team.

1.7 Key Words

The following key terms are used in this SOP.

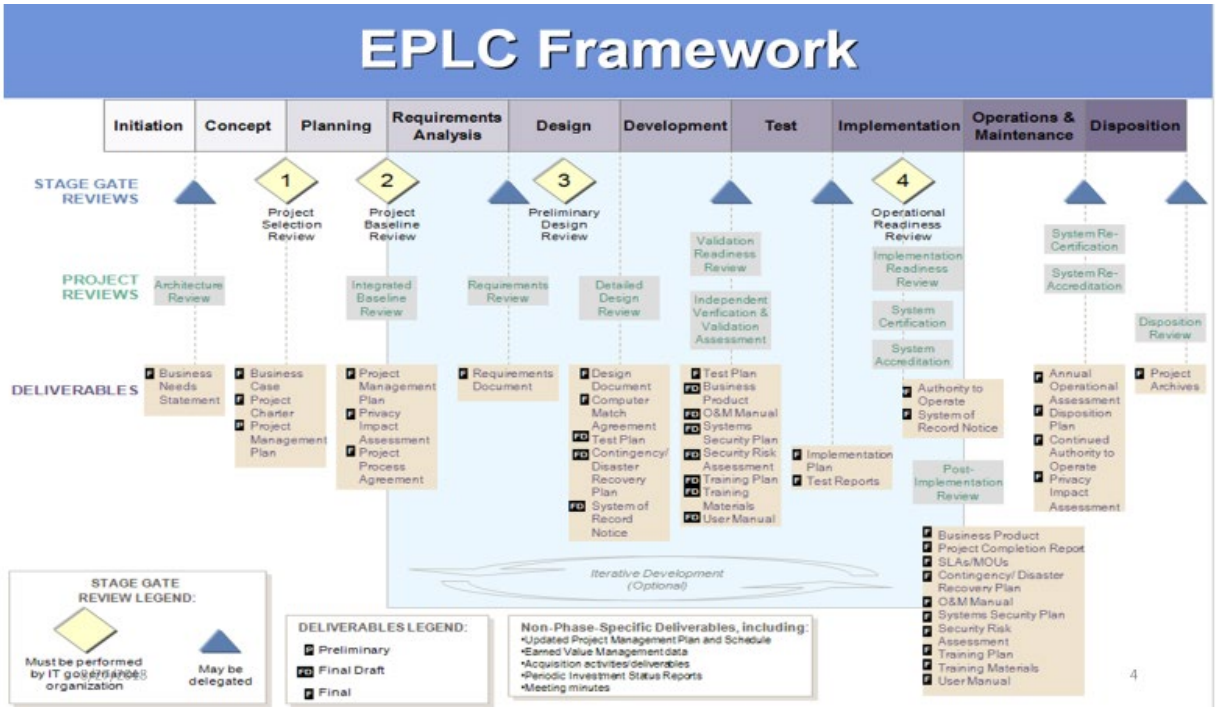
- BRICS – Biomedical Research Informatics Computing System
- CiSTAR – Clinical Informatics System for Trials and Research
- CASA – Collection Access Sharing Analytics Platform
- CCB – Change Control Board

2. Procedures

The Standard Operating Procedure for Agile development methodology and procedures are as follows. Scrum is a framework for developing and sustaining complex products.

2.1 Overview of the Agile SDLC

The EPLC (Enterprise Performance Life Cycle) is a standardized project management methodology that guides HHS IT investments and ensures that mission objectives are being met throughout the lifetime of a system. Agile software development life cycle is an iterative development process between Requirement analysis and Implementation phases in the overview diagram of the EPLC framework below:



2.2 Development and Management Tools

To follow the Agile/Scrum development methodology more efficiently, NINDS DIR ITBP and CIT OIR ISL BIRSS chose to use the following development and management tools listed in the Table 1 below:

Table 1: Lifecycle Management Tools

Project Activity	Management Tool
Help Desk Queue	JIRA
Design Comps	MS Visio, InVision, Balsamiq mockups, MS PowerPoint, Adobe Photoshop,
Project Planning / Scheduling	MS Excel, MS Visio
Defect Tracking and Baseline / Change Requests	JIRA
Wireframes	MS Visio, InVision, Balsamiq mockups, MS PowerPoint, Adobe Photoshop,
User Stories	JIRA
Product Backlog	JIRA
User Roles Matrix	Confluence
Design Comps	MS Visio, InVision, Balsamiq mockups, MS PowerPoint, Adobe Photoshop,
Logical Data Model	Confluence
Physical Data Model	Confluence, Bitmap
System Testing	JIRA, UFT, Selenium Web Driver
User Acceptance Testing	JIRA

Project Activity	Management Tool
Defect Tracking / Change Requests	JIRA
Version Control	Subversion, BitBucket

2.3 Sprint Zero Activities

During Sprint zero, a series of facilitated sessions to orient team to the project's business value, the process, and one another, just enough planning and design prior to Sprint Development cycles:

- **Scope Definition:** Scrum team reviews an initial set of features with Product Owner, SMEs, key stakeholders to prioritize them into two categories: one for the first release and the rest of features will be put in the product backlog. The first category will be organized into a notional sprint plan to determine when each feature will be implemented. This defined scope will drive the requirements elicitation process to ensure the most needed features are validated with detailed and complete requirements for development actions.
- **Release Planning:** Project planning session for a software release is to review the set of features in the first category and to set production release date. The release planning results are such as refined the product vision, updated estimates, priorities, release plan and roadmap. There are two primary work streams for the BRICS project: Roadmap development and Enhancement development work. These two development teams operate in parallel during a major relase:
    - Critical development team (CRIT) – is responsible for delivering on major functionality that is part of system roadmap and requires substantial development effort.
    - Production Support (PS) – is responsible for smaller features (enhancements) that are derived from CCB and scoped in throughout a release.
- **Wireframe Development** – Annotated Wireframes (WFs) represent the layout of a web or mobile screen. They define fields, interactions, validation rules and other features needed by the business. Wireframes are created in low-fidelity format to focus on information content and function. Corresponding high-fidelity Design Compositions (DCs) will conform to actual style guide design standards (e.g. typography, color palette, graphic elements). Wireframes and Design Compositions will be created based on Use Cases before each Sprint.
- **User Story Development** – User Stories represent what a user needs to do as part of his or her job function. User Stories are based on associated Wireframes and conform to the INVEST model. They are: Independent, Negotiable, Valuable,

Estimable, Small, and Testable. The standard user story format is as follows: “As a <user type> I want to be able to <action> so that <benefit>. Included in the user story are addition details to further define the requirement including: business need(s)/goal(s), pre-conditions and acceptance criteria. The user story is complete when the acceptance criteria is fully satisfied. Stories will be tracked in the Product Backlog and will include estimates (derived from the summation of all decomposed technical sub-tasks), Epic association, and detailed acceptance criteria.

- **LOE Estimation** – A combination of traditional estimation techniques are applied to the project including: Bottom-up estimation, analogous estimation, parametric estimation, and relative estimation using story points.

Upon completion of LOE Estimation, the development cycle will begin for the current Sprint. In parallel, the requirements and design team will begin creating all necessary requirements for future sprints.

2.4 Sprint Planning

At Sprint planning, the agile development team generally refines story point estimates, task hour estimates, determine what to do and how to do it. Initial estimates will be revisited to confirm accuracy. User story estimates in comparison with total team capacity (hours) will determine how many stories can be included in a Sprint. The BRICS calculates the total effort of a Sprint by scoping in User Story estimates that equal Current Capacity for a defined iteration (3 weeks). For example: A 3-week team capacity is equal to 200 available resource hours; the scoped in user story total will be equal to 200 hours. Stories will be added to the Sprint based on Priority until Current Capacity is reached. As a result of sprint planning, estimated and prioritized user stories are committed by the team.

2.5 Sprint “N” Activities

The scrum team can address complex adaptive problems within sprint N, while productively and creatively delvering products of the highest possible value. The following activities will occur within Sprint cycles during the Development:

- **Development**: Technical design activities, development of code, system testing and updates to project/system artifacts will occur within the Development portion of the Sprint cycle.

- **Daily Standup:** The daily standup or “Scrum” will occur every day at the same time and place and will include the entire development team as well as any Business SMEs or Critical Partners who wish to attend. The BRICS development agile board will be reviewed to identify how many user stories and/or technical sub-tasks remain for the current Sprint and each team member will answer three

questions: 1) “What did you do yesterday?” 2) “What will you do today?” and 3) “What obstacles are in your way?”

- **Sprint Review:** The development teams will provide a demonstration of working software in set intervals during the Sprint cycle. Demonstrations performed during a Sprint will be performed during a weekly Capability Review Session. Newly developed features will be reviewed with the Business Owner, Business SMEs, and Critical Partners who provide feedback. During the Sprint Review and Capability Review Session, the Business Owner will accept or not accept each story. They may also determine if the software is ready for release.

- **Sprint Retrospective**: The development teams will review what worked and what didn’t for the Sprint so that immediate improvements can be made to the Sprint process.

Upon completion of the Sprint Retrospective, the Sprint cycle will begin again with Sprint Planning, unless the Product Owner has requested to deploy software to production.

During the Sprint, Agile metrics will be tracked including:

- **Story Points**: Relative estimates (not actual hours) used by the development team to estimate User Stories determined through a relative estimation technique known as “Planning Poker” which uses the Fibonacci Series (1, 2, 3, 5, 7, 13).

- **Velocity**: Number of Story Points that the development team can accomplish in a given Sprint, expected to increase over time as teams become more productive.

- **Capacity**: Amount of time in hours that the development team can use in a given Sprint. Capacity changes for each Sprint based on the length of the Sprint, actual workdays available, number of developers and estimated holidays or vacation days.

- **Defect Rate:** Number of defects found in a Sprint, expected to trend lower over time.

These metrics will be tracked over time so that improvements can be monitored between Sprints and across Releases.

3. Schedule Management

The scrum development team, using iterative and incremental practices, can adjust to rapidly-changing requirements, and produce a product that meets evolving business goals. Agile software development methodologies promote a disciplined project management process that encourages frequent inspection and adaptation, a leadership that encourages teamwork, self-organization and accountability, a set of best practices intended to allow for rapid delivery of quality software, and a business approach that aligns development with customer needs and organizational goals.

3.1 Sprint Planning Session

Planning sessions will occur during the week prior to the start of the next Sprint. Prior to the start of the planning session, development Track Leads are expected to have scoped in user stories properly decomposed and technical sub-tasks estimated and assigned.

3.2 Sprint Review Session

Product Owner identifies what has been done versus planned during the Sprint Review at the end of every Sprint. Team demonstrates completed functionality in the sprint, feedback is recorded, and participants discuss the functionality they have seen and projections to decide what to do next. Therefore new features are adeed and product backlog is reprioritized.

3.3 Daily Standup

Daily meeting is to inspect progress against sprint goal and adjust plans. It focuses on making trade-offs, coordinating efforts and risk management. Scrum Master starts with context and aligns with release and sprint goals, days left in the sprint, and scheduling (who is in/out, who is offsite). Each team member will answer three questions and focus on coordination. The entire team plans and collaborates to get to “done” stage and makes tradeoffs. The taskboard, action items, blockers, risks, and burndown will be updated. Informal follow up meetings arranged when necessary. The 15-minute Daily Standups for all subtasks are as follows:

- Critical Development: will occur every weekday at approximately 10:00AM EST
- Production Support: will occur every weekday at approximately 10:15AM EST
- Infrastructure Support: will occur every weekday at approximately 12:00PM EST

4. Change Management

Change control management will be handled using the appropriate process for managing changes through the Change Control Board (CCB).

The CCB is responsible for prioritizing and selecting business issues to be resolved during each release. This CCB is responsible for managing system enhancements and change requests. The core roles included in the CCB are the Business Owner, Production Support (PS) Technical Lead, Requirements Analyst, Enhancement/Change Request Owners and/or representatives, and Quality Assurance representatives.

Change Control Board (CCB) meeting is scheduled monthly or on an as-needed basis. In this meeting, we discuss improvements to the system with the Business Owner in order to prioritize them based off business value. After an enhancement or change request is approved, the following defined process is followed so that requirements, design, story creation, estimation, and re-prioritization will be performed.

The change control process manages the acquisition, verification, approval, and execution of externally identified system enhancements and issues identified during application development and or production operations.

The BRICS CCB process and workflow are diagramed with step-by-step descriptions in Appendix B.

5. Risk and Issue Management

Risk and Issue Management begins at the estimating stage of the project, prior to task inception, and continues throughout the software development lifecycle. Any assumptions made in the development of a plan or schedule will be considered for documentation as a risk. Any problems that need to be resolved will be documented as an issue.

BRICS/CiSTAR/CASA project defines, maintains, and tracks the risks and issues on two separate documents. Please refer to its risk registry and issue log for more details. An **Issue** is defined as one of the following:

- Defect Report
- Enhancement Request
- Change Request

The issue management process for this project is restricted to those issues identified by outside parties (reviewers, testers, end-users, etc.) as opposed to those identified by members of the development team. The Development team's issues will be managed internally by the development team.

The roles associated with change control include:

a) Submitters, who are generally end-users or testers.
b) The project leads (quality assurance lead and project management lead) acting in concert to perform verification and validation of new issues.
c) The project CCB which controls the assignments of issues to future build releases
d) The Project Support team, which performs issue analysis and issue correction

Each issue flows through the following four phases:

5.1 Discovery Phase

Issues may be discovered by end users during the production operations or by members of the functional team during software development.

5.2 Analysis Phase

Quality assurance lead will validate whether this issue is valid. Validated issues are also assigned priority level based on the severity of the issue. New issues that are deemed not valid are presented to the CCB. Invalid issues are placed in an archive of invalid issues.

Valid new issues are examined by the appropriate members of the team to identify the root cause of the issue, as well as the configuration items that have to be changed in order to close the issue. The Production Support team members estimate the level of effort required to make the defined changes. If the issue is an enhancement the functional team will begin developing requirements if needed.

5.3 Authorization Phase

The CCB evaluates the open issues to determine the production release that the issue will be corrected in, according to the level of effort submitted by the developer or quality assurance lead. The quality assurance lead will assign the issue to be corrected in the appropriate development branch.

5.4 Sprint Phase

Each sprint of the product release is focused on the accomplishment of a specific set of goals, for either the application or specific component. This is the phase where the developer corrects the issue that was identified by the end-user. The CCB authorizes the initiation of future product releases if needed.

6. Requirements Management

Requirements Development (RD) is the process of producing, and analyzing customer, product, and product-component requirements. These requirements address the needs of relevant stakeholders, including those pertinent to various product life-cycle phases (e.g., acceptance testing criteria) and product attributes (e.g., safety, reliability, and maintainability). Requirements also address constraints caused by the selection of design solutions (e.g., integration of COTS products). Requirements development planning must start as soon as the scope (goals and objectives) of the system is determined.

Projects may derive requirements from many sources, including the Statement of Work (SOW), a client-provided requirement specification, Joint Application Design (JAD) sessions and interviews with the Business Sponsor or end-users. The requirements that specify the functional aspects of the software to be developed must be sufficient to allow project team members to perform a formalized analysis for completeness and correctness. During requirements development, it is critical to assess resources, risks, costs, and overall feasibility of the project.

Requirements are defined and captured as Wireframes, User Stories, User Roles Matrices and Data Specifications. Requirements are normally validated with Product Owner and Instance Program Managers through Wireframes, Mock-ups, and/or Prototypes. Once this initial validation is complete, the Wireframes are refined and finalized.

User Stories, which detail the Business Needs/Goals, Pre-conditions, and Acceptance Criteria, are crafted next. These are also validated by Product Owner and Instance Program Managers along with User Permissions and Data Specifications. Product Owner and Instance Program Managers will review all requirements and ensure that they comprehensively address the business needs.

Once validated, test cases and design compositions are created. Test cases ensure the requirements are fully tested for quality assurance throughout the development process. Design compositions ensure that the front-end design of the system matches the visual style and integrity that Product Owner and Instance Program Managers have requested.

During each sprint, the Product Owner confirms the specific set of stories that will be developed. At the end of each sprint, the key stakeholders reviews the status of each story, using the Acceptance Criteria, and accepts or rejects each story. A rejected story will be re-evaluated for new requirements and put back into the backlog to be enhanced.

7. Scope Management

The development scope of the project includes the planning, design, development, testing, training, and going live. Project completion will occur when the software has been successfully installed and implemented for production. All project work will be performed internally at NIH and no portion of this project will be outsourced. The scope of this project does not include external dependencies such as server-side changes in standard operating systems or virtual matchines to run the software, security updates or patches.

Scope Management is the process for ensuring that the project activities include everything necessary to meet the goals of the project and only those activities are required. It provides a clear process for:

- Collecting Requirements – defining and documenting stakeholder needs to meet project goals.
- Defining Scope – developing a detailed description of the project and determining the final product will or will not be included in each release.
- Creating the Work Breakdown Structure (WBS) – dividing the deliverables and project work into manageable activities that can be assigned to the project team.
- Verifying Scope – formalizing the acceptance of project deliverables.
- Controlling Scope – monitoring the status of the project and managing any changes to the defined scope (i.e. the scope baseline).

7.1 Collecting Requirements

BRICS/CiSTAR/CASA project gathers the requirements in a number of ways, including but not limited to:

- One-on-One interviews – These sessions will be held whenever the stakeholder's schedule is available and effort needs to be made to accommodate to their needs. They may also be conducted remotely through screen sharing or email exchanges.

- Focused Working Group Sessions – These sessions will be held with a group of stakeholders and subject matter experts who have key insight into the business problem. The purpose of these sessions is to understand expectations and derive core requirements and critical success factors.

7.2 Defining Scope

This is the process for collecting business needs that help define what the final product will be. It includes not only system functionality but all major deliverables for the project (documentation, training, etc.).

Product Owner and key stakeholders will define scope with the following activities:

- Outline and review high-level features.
- Determine which features are absolutely necessary, and if some features can be simplified for the first release.
- Share all features that are wanted by the business, even if those features are not feasible for the initial release. The goal is to create a full backlog of everything the business wants even beyond the currently planned release.
- Prioritize the feature backlog by scope.

Upon completion of these activities, enough information will be available to group the prioritized features into planned releases.

7.3 Verifying Scope

The Veriying Scope process covers the formal acceptance of deliverables. While approval of scope will rest in the hands of the Business Owner and Instance Program Managers, effort will be made to ensure that deliverables meet the expectations of all stakeholders for the final product. Since this project will be implemented using a Scrum methodology within the EPLC framework, there will be regular opportunities of audits and interim reviews (i.e., Sprint Reviews, User Acceptance Testing, Demonstrations, etc.) by stakeholders and critical partners before the final product is delivered/deployed.

7.4 Controlling Scope

The Controlling Scope step is focused on monitoring the status of the project and all features identified during Defining Scope. As development progresses, Epics, stories, and technical sub-tasks will be tracked through JIRA and updated based on the defined development workflow. Items completed will have a workflow status equal to “Closed” showing progress towards the release goal.

As part of the development process, Weekly Status Meetings, Change Control Board (CCB), Defect Reviews, and other checkpoints with the business will result in new requirements and modifications to already-implemented features.

While additional requirements and enhancements being captured, it will be critical that stakeholders prioritize functionality with the goal of deploying a product that best addresses the original requirements and acknowledging that new enhancements should be incorporated into subsequent releases.

Scope will be controlled through a regular assessment of changes to the scope baseline. Before scope changes are validated, an assessment will be made to understand which project schedule elements could be impacted by the change and the degree of the impact. Regular audits will also occur to assess the variance of the scope being implemented against the scope baseline to identify nominal shifts that result from day-to-day implementation decisions and assess whether or not corrective or preventative actions is required.

8. Quality Management

All members of the project team will play a role in quality management. It is imperative that the team ensures that work is completed at an adequate level of quality from individual work packages to the final project deliverable. The following are the quality roles and responsibilities for this Project:

The Project Sponsor or Product Owner will ensure compliance and sign off on the final acceptance of the project deliverable.

The Project Manager is responsible for quality management throughout the duration of the project. The Project Manager will work with the project's quality specialist to establish acceptable quality standards.

The Quality Specialist is responsible for working with the Project Manager to develop tools and methodologies for tracking quality and standards to establish acceptable quality levels. The Quality Specialist will create Master Test Plan and maintain Quality Control and Assurance Logs throughout the project.

The remaining members of the project team, as well as the stakeholders will be responsible for assisting the Project Manager and Quality Specialist in the establishment of acceptable quality standards. They will also work to ensure that all quality standards are met and communicate any concerns regarding quality to the Project Manager.

9. Records Management

All data and/or records generated during this procedure are stored in the NINDS SharePoint-based Document Library.

10. Review/Revision History

Date	Author	Description of Change
02/27/2019	Gladys Wang	Document Creation
04/08/2019	Leonie Misquitta	Added Appendix A
04/10/2019	Tsega Gebremichael	Added Appendix B

Appendix A. RTM

The Requirements Traceability Matrix (RTM) in the table below was prepared before a software release.

#	Key	Summary	Linked Test Cases/Comments		
1	PS-4420	Remove/hide the 'Biorepository Subject ID' field and user inputs in ProFoRMS	PS-4477		
2	PS-4378	"See also" - need to increase the size of the field in the database and the text box that displayed in DD	PS-4389		
3	PS-4331	Add to GUID tool Country of Birth	PS-4487	PS-4488	PS-4489
4	PS-4268	As user, system should let me refresh my session in Data Dictionary	PS-4336		
5	PS-4267	As an Operation User, data error report email should be get updated with no of request for archived, deleted data set and requested study.	PS-4327		
6	PS-4266	As a user, system should allow me to change subject label GUID to Subject ID or vice versa in Manage Protocol Section of ProForms	PS-4335		
7	PS-4179	SVN to GIT Migration	Not a Functional Test Item hence NO Test Cases		
8	PS-4135	As a admin user, I need my previous admin noted migrated into the new admin note functionality box.	PS-4308		
9	PS-4031	As cdRNS user, I should able to see cdNRS specific Form Structures	PS-4479		
10	PS-4028	As FITBIR Public site data dictionary - default DE view to Awaiting Publication and Published DEs	PS-4307		

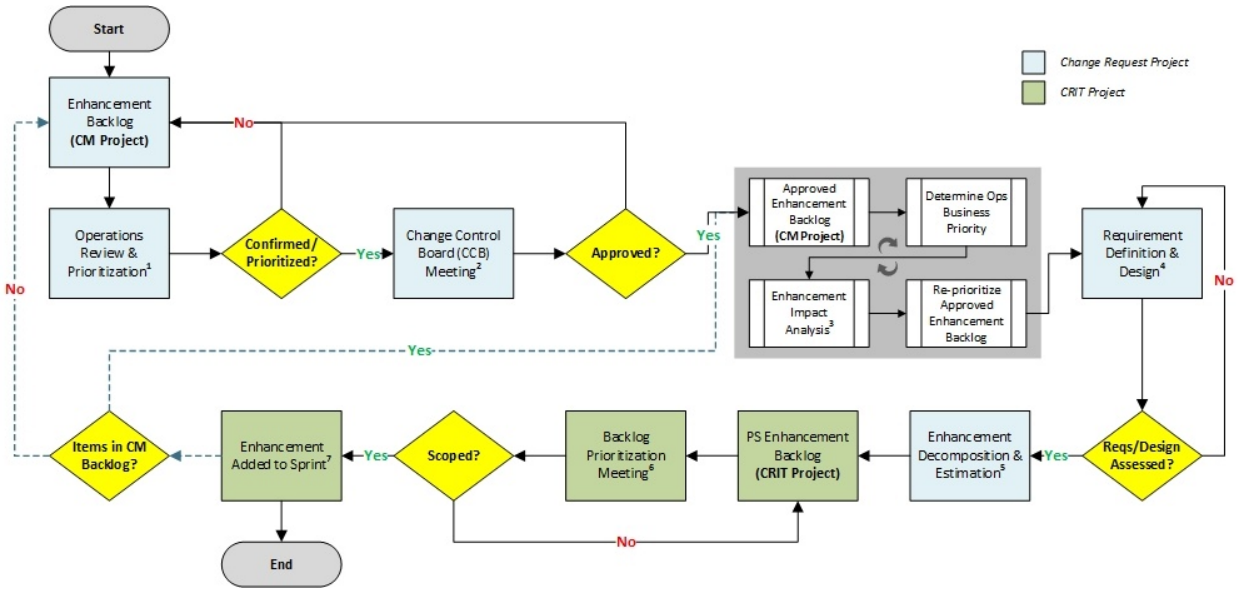
11	PS-4027	As a user, I should know which environment my email came from	PS-4338		
12	PS-4009	As a user, I should know whether my email is for BioSample or BioFind	Duplicate of PS-4027 so was closed		
13	PS-3830	Data Repository - Research Management Table Sort Recommendation	PS-4326	PS-4325	
14	PS-3829	Need to update the primary BRICS public site	Duplicate of CRIT-9102 so was closed		
15	PS-3828	As Account Admin, I should able to approve or rejected privileges individually.	PS-4013		
16	PS-3827	As a Account Admin/Reviewer, I should able to identify user, whose privileges been expired.	PS-4328		
17	PS-3821	As a user, I should see consistent in data set statuses across table	PS-4340	PS-4341	
18	PS-3659	Supporting Documentation | The "File" field information appears under the Column Name.	PS-4314	PS-4315	
19	PS-3441	As a user, I want files I'm attempting to upload to be maintained when there is a validation error in related fields when trying to upload the file (PF, Act, DD))	PS-4334		
20	PS-3309	As a user, I should have the Ability to make an optional/recommended question required on the eform level	PS-4343		
21	PS-3226	As an admin or end user, I want the data dictionary search to support special characters *,? and "" in order to increase the likelihood of returning a result the user is searching for.	PS-4483	PS-4484	
22	PS-3214	Implementation of converting byte array to hex is flawed	Not a Functional Test Item hence NO Test Cases		
23	CRIT-10029	Update GUID Manual file	CRIT-10043		

24	CRIT-9903	Data Validation Calculation Rules - an error message should occur if a total score or a subscore is given while one of the needed data inputs is missing	Spreadsheets with datasets attached to the User Story		
25	CRIT-9874	Data validation calculation rules - DHI	Spreadsheets with datasets attached to the User Story		
26	CRIT-9844	Data validation calculation rules - Headache Impact Tool (HIT-6)	Spreadsheets with datasets attached to the User Story		
27	CRIT-9774	Data Validation Calculation Rules - FIM Instrument	Spreadsheets with datasets attached to the User Story		
28	CRIT-9581	As a developer, I should be able to migrate GUID information from one MongoDB instance to another	Not a Functional Test Item hence NO Test Cases		
29	CRIT-9549	As a user, I should be able to generate GUIDs in a batch process	CRIT-9729	CRIT-9730	
30	CRIT-9535	As a user, my PII should be sent to the server as a Hashcode	CRIT-10048		
31	CRIT-9534	As a user, I should be able to interface with the GUID server	CRIT-9767		
32	CRIT-9533	As a user, I should be able to see the interface of the GUID client	CRIT-9721		
33	CRIT-9532	As a user, I should not be able to enter invalid PII	CRIT-8983	CRIT-9787	CRIT-9788
34	CRIT-9531	As a user, I should be able to validate that a GUID exists	CRIT-9785	CRIT-9786	
35	CRIT-9530	As a user, I should be able to convert a pseudoGUID to a GUID	CRIT-9770	CRIT-9771	
36	CRIT-9520	For large result sets in QT, adding to download queue crashes the server	CRIT-9796		
37	CRIT-9511	3.6 Code Merge for NINDS/NIA/GRDR instances + add mongoDB conn info	Not a Functional Test Item hence NO Test Cases		

38	CRIT-9356	NINDS/NIA/GRDR Code Merge	Not a Functional Test Item hence NO Test Cases		
39	CRIT-9259	Data validation-calculation rules - SCAT-3	Spreadsheets with datasets attached to the User Story		
40	CRIT-9223	GUID Javascript Client Design	Not a Functional Test Item hence NO Test Cases		
41	CRIT-8571	Design Basic GUID JS Framework	Not a Functional Test Item hence NO Test Cases		
42	CRIT-7779	As a user, I should be able to access the GUID user guide	CRIT-8990		
43	CRIT-7655	As a user, I should be able to generate a GUID through the new JS Client	CRIT-9768	CRIT-9769	
44	CRIT-7640	As an Admin, I should see all legacy & new GUIDs that have been created by my Entity	CRIT-8993	CRIT-8995	
45	CRIT-7613	As a NINDS user, I should be able to access legacy NINDS GUIDs	CRIT-8993		
46	CRIT-7612	As an NIA user, I should be able to access legacy NIA GUIDs	CRIT-8993		

Appendix B. CCB Process

The Change Control Board (CCB) process and workflow are described as follows:



1 - Operations Review & Prioritization: On an as-needed basis, A member of the PMO/Requirements Track will export a list of current enhancement tickets from JIRA and send the export to the appropriate Operations team (based on 'Reporter'). The submitting Operations team(s) validates that the Enhancement(s) is 1) still required and 2) has an up-to-date description. Based on the validation and number of submitted enhancement requests, the Operations team will provide a prioritization ranking (e.g. 1, 2, 3...) for each submitted enhancement. The output of this stage is a list of 10-12 prioritized enhancements.

2 - Change Control Board (CCB) Meeting: CCB is scheduled monthly or on an as-needed basis. During this meeting, the final list of enhancements derived from step #1, is presented to the client/product owner. Members of the Operations team(s) along with representatives from Development & QA will be present. The intent of CCB is for Operations to present their business case/value-add statement as to why a particular enhancement should be approved. This meeting is not intended for solutioning nor should decisions be influenced based on technical constraints or system limitations. If approved, the PMO/Requirements Track will meet with the appropriate stakeholders to capture and define functional requirements. If rejected, the enhancement will remain in the Enhancement Backlog for future consideration/logging purposes. The output of this stage is a list of client approved enhancements ready for additional analysis.

3 - Enhancement Impact Analysis: As provided by the Operations team, the initial functional requirements for each enhancement will be reviewed internally by Requirements, Development, and QA. During this meeting, each enhancement ticket is reviewed for: 1) high-level technical feasibility; 2) design (UI/UX); 3) potential duplication/redundancies; and 4) analysis for transition/addition to a Roadmap item (escalation from PS-enhancements to CRIT track). The output of this stage is development team input, feasibility, risks/constraints, and insight into new scope.

4 - Requirement Definition & Design: A member of the PMO/Requirements track will work directly with the Operations team that submitted the enhancement request to gather requirements and present design concepts. This stage also involves presenting the final solution (requirements & design) to representatives from all of the Operations teams to ensure visibility, awareness, and acceptance. Once final requirements and design concepts are approved by Ops, the PMO/Requirements track will generate user stories and acceptance criteria for review with the Development team. The output of this stage is an approved requirements package (per enhancement), which is reviewed and confirmed by end-users and Operations, that includes: 1) Business need/goal, 2) Pre-conditions, and 3) detailed Acceptance Criteria.

5 - Enhancement Decomposition & Estimation: Once requirements have been finalized and approved, the PMO/Requirements and the Development teams will decompose the Acceptance Criteria into Technical sub-tasks and estimate the enhancement(s). The output of this stage is a defined and estimated list of enhancements ready for client/product owner review and prioritization.

6 - Backlog Prioritization Meeting: Fully defined and estimated enhancements will be presented to the client/product owner. During this meeting, the client/product owner will be presented the fully defined enhancement solution and level of effort (LOE)/estimate involved to implement the enhancement. Enhancements not approved will remain in the PS Backlog for future consideration. The output of this meeting is an approved list of PS Backlog enhancements that can be scoped into an upcoming sprint.

7 - Enhancement Added to Sprint: Approved enhancements will be reviewed by the PMO/Requirements and the Development team during an internal sprint planning session. A resource capacity plan will be developed to align resource availability with enhancement estimations. At this point, the enhancement is scoped into a sprint iteration and is scheduled for development.

***Once a PS sprint is fully scoped, the PMO/Requirements track performs a "grooming" process of the remaining CCB approved enhancements. If no CCB approved enhancements are in the backlog, the process is restarted and the full CCB process is initiated.*