STANDARD OPERATING PROCEDURE

SOP Number: CRI.SOP. SDLC-003	**Title: Software Development**	
Version No.: 1.0	**Effective Date: April 18, 2023**	**Page 1 of 10**
Supersedes Version: N/A **Dated:** N/A	REQUIRED APPROVALS BELOW	
CRI Director:	DocuSigned by: Bill Sanics	**Date:** 5/8/2023
CISIL Approver 1:	DocuSigned by: Becki Gerwitz	**Date:** 5/8/2023
CISIL Approver 2:	DocuSigned by: Stephen Hargrove	**Date:** 5/8/2023

1.0 Purpose

This Standard Operating Procedure (SOP) establishes the process of writing and deploying software.

2.0 Scope

This procedure is the implementation of *CRI.POL-004 Software Development Lifecycle* applies to all software written and deployed by CRI regardless of final form or designation. This SOP is limited to the steps required to define the elements that dictate how the software will be developed, the environments in which it will be developed, and how its life cycle will be tracked. The general Pre/Post award activities (e.g., contracts, terms, etc.) and Project Management (e.g., resource management, customer relations management, etc.) are outside the scope of this SOP.

It is important to understand that not all steps in this SOP must be completed, but all must be addressed. CRI's software projects are quite varied and have different needs and requirements. Each project is categorized by rigor and oversight to ensure CRI's efforts are calibrated to meet the task at hand to be effective and efficient in our product delivery.

3.0 Responsibility

3.1 Technical Director: role bears responsibility for assuring any written software complies with this SOP.

3.2 Senior Manager: performs all functions related to tracking activities, manages resource allocation required for the software development process, and is the Technical Director's primary backup.

3.3 System/Database/Informatics/Software Programmer: responsible for following, writing, and deploying software according to this SOP.

3.4 Project Team: responsible for providing documentation and testing according to this SOP.

4.0 References

4.1 CRI.POL-001 Clinical Research Informatics Quality Management System (QMS)

4.2 CRI.POL-004 Software Development Lifecycle

4.3 UT System: UTS 165 Information Resources Use and Security

4.4 21 CFR Part 11, Electronic Records; Electronic Signatures, March 20, 1997

4.5 General Principles of Software Validation: Final Guidance for Industry and FDA Staff, January 11, 2002, FDA

4.6 Alistair Cockburn (2001) *Agile Software Development* Addison Wesley (ISBN-10 0201699699, ISBN-13 978-0201699692)

5.0 Acronyms and Definitions

Term	Definition
Clinical Research Informatics	Clinical Research Informatics sits at the intersection of information science, information systems, workflow and processes, and leadership and management. It is how data is acquired, structured, stored, processed retrieved, analyzed, presented, and communicated. CRI transforms data into usable actionable information. CRI is also the name of the division within Population Health Sciences performing these SOP activities
CRIS	Clinical Research Informatics Specialist
CRM	Customer Relations Management: Managing the relationship with the customer or sponsor.
Data Transformation	The process of taking data that exists in one format or state and converting it into a different format or state.
DBA	Database Administrator: Someone that works within a database management system using both DML and DDL operations.
DBMS	Database Management System: A Database Management System is a software application used to access, create, and manage databases.
DDL	Data Definition Language: SQL that defines objects within a database management system.
DEF	Data Extraction Form: The template used to identify the type of data retrieved from the clinical data warehouse.
DML	Data Manipulation Language: SQL that interacts with data within a database management system.
ETL	Extract Transform Load: a process that extracts, transforms, and loads data from multiple sources to a data warehouse or other unified data repository.
IDEAS	Informatics Data Exchange and Acquisition System: The CRI-developed research and data management tool/environment.
Project Manager	Someone who organizes, plans, and executes projects while working within restraints like budgets and schedules. Project managers are in charge of leading teams, defining goals, communicating with stakeholders, and seeing a project through to its closure.
QMS	Quality Management System: The QMS consists of the documentation, tools, and processes CRI has put in place to assure quality deliverables and operations.
Software Application	a type of computer program that performs a specific function. Each application is designed to assist end-users in accomplishing a variety of tasks.

SOP	Standard Operating Procedure
SQL	Structured Query Language: a computer language for storing, manipulating and retrieving data stored in a relational database.
SVC or VCS	Software Version Control or Version Control System: CRI code version control provides a logical method to preserve individual contributions of team members without overwriting another developer's work, and then merges and reconciles changes and updates. The system is also a failsafe for finding and supporting issues that are in conflict. The historical record produced can be used to track down problem origins or archive versions for later use.
Sys Admin	Systems Administrator: An IT professional responsible for maintaining the entire IT and technology stack of an organization. They are responsible for maintaining the entire system that supports a business and allows it to operate effectively and efficiently.
UTHSA	University of Texas Health Science Center San Antonio
VM	Virtual Machine: virtual environment that functions as a virtual computer system with its own CPU, memory, network interface, and storage, created on a physical hardware system (located off- or on-premises).
WI	Work Instruction: A Work Instruction contains more detail than a Procedure and is only created if detailed step-by-step instructions are needed. Work Instructions are the “how you address satisfying the SOP” documents.

6.0 Procedure

6.1 Software Project Scope and Definition (Pre-Coding Activities)

The software development process begins during project initiation. A software development project will pass through the necessary phases defined in CRI.POL-004 Software Development Lifecycle. All decisions made during this process shall be documented within the CRI IDEAS Project Tracking System.

6.1.1 ***Definition***: Within the *Definition Phase*, project details dictate the resource structure needed for software development and may require the interaction and coordination of multiple domains of expertise and programming. The Technical Director and Senior Manager shall take the information provided during the pre-award process and determine what type of project this is and what teams are required to be assigned for its completion.

6.1.1.1 **Software Development**: Projects that deliver or enhance software applications.

6.1.1.1.1 Project Team: Lead Developer and other software programmers as required. The lead developer will interact with the Senior Manager to include other programming teams (as needed), schedule milestones, and ensure all project-related documentation is completed and uploaded to the CRI Project Tracking system.

6.1.1.2 **Clinical Research Data Management**: Projects integrating and extracting Clinical/Personal Health Information and related data.

6.1.1.2.1 Project Team: Project Manager (Clinical Research Informatics Specialist -- CRIS) as required, Lead Warehouse Developer, and other warehouse developers as needed. The CRIS will interact with the Senior Manager to determine the programming needs and include other programming teams (as required), schedule milestones, and ensure all project-related documentation is completed and uploaded to the CRI Project Tracking system. The lead warehouse developer will act in this role when there is no defined project manager.

6.1.1.3 **Research Database Support**: Projects that build, manage, or maintain databases and database-related code.

6.1.1.3.1 Project Team: A Database Administrator (DBA) will be assigned to all projects requiring DDL, DML, or other DBMS interactions. When the project is solely focused on research database support, the assigned DBA will collaborate with the Senior Manager to include additional programming teams (as needed), schedule milestones, and ensure all project-related documentation is completed. When database work is required to support other project types, then the DBA will be a member of the project team following the direction of the assigned project lead.

6.1.1.4 **Systems Management**: Projects that pertain to servers (hardware and software), software installation and configuration, networking, security, or infrastructure.

6.1.1.4.1 Project Team: A System Administrator (Sys Admin) will be assigned to all projects requiring changes, additions, or management of any item in their service area/catalog. When the project is solely focused on system support, the assigned Sys Admin will collaborate with the Senior Manager to include other programming teams (as needed), schedule milestones, and ensure all project-related documentation is completed. When system work is required to support different project types, then the Sys Admin will be a member of the project team following the direction of the assigned project lead.

6.1.2 ***Categorization***: This phase categorizes software by three distinct criteria to define (1) confidentiality, (2) programming rigor, and (3) project oversight. The Technical Director and/or Senior Manager shall take the information provided during the pre-award process and determine under what categories the software will be developed.

6.1.2.1 **Data Confidentiality**: Data are classified according to HOP Data Classification Guidelines (HOP 5.8.21)

6.1.2.1.1 The Technical Director/Senior Manager will work with the Project Manager or Team Leader to define any data that will be managed or delivered. The data fall under the following categories.

6.1.2.1.1.1 **Confidential/High Risk**: Protected from unauthorized disclosure or public release.

6.1.2.1.1.2 **Controlled**: Data that is not generally created for or made available for public consumption but may be subject to release to the public through a request via the Texas Public Information Act or similar State or Federal law.

6.1.2.1.1.3 **Published**: Data made available to the public.

6.1.2.2 **Programming Rigor**:

6.1.2.2.1 The Technical Director/Senior Manager will work with the Project Sponsor to determine the rigor required for software or data delivery.

6.1.2.2.1.1 **High**: Projects that warrant the implementation of the whole quality management system based on client requirements, potential harm resulting from data retrieved, collected, or managed, or local, state, or federal guidelines

6.1.2.2.1.2 **Medium**: Projects supporting research or operations that don't necessitate "High" programming rigor but may still support or manage Confidential/High-Risk data.

6.1.2.2.1.3 **Low**: Projects with lower accuracy needs or long-term operational expectations

6.1.2.3 **Project Oversight**: The Technical Director/Senior Manager will work with the Project Sponsor to determine the level of oversight required for software or data delivery.

6.1.2.3.1 **Senior Manager**: Assures projects are delivered within resource constraints and sponsor requirements.

6.1.2.3.2 **Clinical Research Informatics Specialist**: Assures investigator timelines are met, data are accurate, and data delivery meets investigator requirements.

6.1.2.3.3 **Programming Team Lead**: Assures code meets all applicable standards, version control requirements are met, and the correct resources are assigned.

6.1.3 **Requirements**: The requirements, regardless of the method captured or cataloged, will be presented to the Project Manager/Team Lead for review and acceptance. This may be a cyclical process as requirements are edited and refined to meet the software development needs. All requirements documentation shall be uploaded to the appropriate project in the IDEAS Project Tracking System. Specific projects may require data transformation as part of the development or delivery process. Requirements may

come in many forms based on the programming team; however, they will all comply with CRI.POL-004 *Software Development Lifecycle*

6.1.3.1 Requirements Documentation methods

6.1.3.1.1 Data Extraction Form (DEF) – Used for Clinical Research Data Management query extraction projects.

6.1.3.1.2 Requirements Documentation – Project-specific forms/methods of documentation written at a level to suit project needs

6.1.3.2 Data Transformations: Any data transformed from its original format must be defined and approved BEFORE THE TRANSFORMATION PROCESS

6.1.3.3 Testing Requirements: Projects may be subject to several testing methodologies; however, Acceptance Testing shall occur for every delivered product. Acceptance Testing rigor shall be defined in the requirements phase.

6.1.4 **Design**: The design, regardless of the method captured or cataloged, will be presented to the Project Manager/Team Lead for review and acceptance. This may be a cyclical process as designs are edited and refined to meet the system and software development needs. Any documented designs shall be uploaded to the project in the IDEAS Project Tracking System. NOTE: Designs may come in many forms based on the programming team; however, they will all comply with CRI.POL-004 *Software Development Lifecycle*

6.1.4.1 Data Extraction Form (DEF) – Used for Clinical Research Data Management query extraction projects.

6.1.4.2 Design Documentation – Project-specific forms/methods of documentation written at a level to suit project needs

6.2 Software Development/Coding

Software coding may begin at any point the Technical Director or Senior Manager defines. The development methodology (e.g., waterfall, spiral, agile, etc.) may allow coding to begin based on a defined module rather than waiting for a fully defined and specified set of requirements and designs.

6.2.1 **Development Environment**: The Development Environment is the system(s) or system environment(s) used to develop the project's code. The environments are defined logically and not physically, as the groups named during the Definition phase work on different physical systems but share common logical names. For example, the software developers and the database administrators have a "Development" environment; however, they reside on different Virtual Machines configured for their respective needs. The development environment shall follow the listed guidelines.

6.2.1.1 **DEVELOPMENT**: The DEVELOPMENT or DEV environment shall be used for all projects unless otherwise defined by the Technical Director or Senior Manager

6.2.1.2 **TEST**: The TEST environment shall not be used as a DEVELOPMENT environment with the specific exception that the DEVELOPMENT environment is not available

6.2.1.3 **PRODUCTION**: The PRODUCTION or PROD environment may be used by the Clinical Research Data Management or Database Administrator teams to

retrieve required data as DEVELOPMENT environments may not exist or meet the needs to write the data extraction SQL.

6.2.1.4 **LOCAL**: The LOCAL environment is the programmer's computer or laptop. A LOCAL environment may be used if (1) approved by the Technical Director or Senior Manager and (2) the other programming environments are not suited to the task.

6.2.2 **Version Control**: CRI version control shall be used for Software Development and Clinical Research Data Management operations. It is used as specified for Research Data Support and Systems Management Operations.

6.2.2.1 **Software Development/Clinical Research Data Management**: The volume of code produced and the integrated team nature of this development makes it vital that version control procedures are established and closely followed. Due to the nature of the programming and programming environments, specific guidelines for each group are managed within those groups. All processes, locations, servers used, etc., are approved by the Technical Director/Senior Manager and Systems Management. Processes are documented and managed within respective Work Instructions (WI).

6.2.2.2 **Research Data Support:** The Database Administrators' (DBAs) work falls into two areas. The first is the DDL/DML operations written and managed WITHIN the DBMS, and we typically do NOT need to move these into version control; however, it may be used for projects designated as "HIGH" programming rigor. The second area is ETL and management code to move data in and out of the operational environments. This code shall be placed into version control.

6.2.2.3 **System Management**: The system administrators, or Sys Admins, manage software, utilities, operations, disaster recovery, etc., on our department servers. This code is only placed in version control as required based on specific project requirements.

6.2.3 **Software Development/Lifecycle Status**: All software development work is managed through the CRI project management tools (Project Tracking and Task Management). All staff shall utilize the system to track effort and activity on assigned tasks at levels specified within each of the programming groups or to the needs of the Senior Manager. All CRI staff are trained in using Project and Task Management systems.

6.2.4 **Software Development**: The software development (coding) process is managed within the group(s) named during the *Definition Phase.* Each group uses specialized environments, programming languages, servers, standards, etc., to meet their deliverables, and those each come with a unique set of requirements. Programming Team Leaders have the authority and responsibility to ensure their teams follow specific protocols.

6.3 Software Testing

Software testing and code review are critical components of the CRI QMS. Testing assures the code works as expected. The review ensures code meets the respective team's guidelines. One or both

of these activities shall be performed for any software deliverable and may be required for data deliverables produced or deployed by CRI.

While the testing process is the same for all programming groups, the implementation will differ based on the programming environment, programming rigor, sponsor requirements, and negotiations with customers. If the project has a Requirements Traceability Matrix (RTM), all testing will be documented within the RTM. All software shall be tested before delivery. The Technical Director/Senior Manager shall define each project's testing needs.

CRI employs the following testing methods

- **Unit testing**: Validating that each software unit performs as expected. A unit is the smallest testable component of an application.
- **Integration testing**: Ensuring that software components or functions operate together.
- **Functional testing**: Checking functions by emulating business scenarios based on operational requirements. Black-box testing is a common way to verify functions.
- **Performance testing**: Testing how the software performs under different workloads. Load testing, for example, is used to evaluate performance under real-life load conditions.
- **Regression testing**: Checking whether new features break or degrade functionality. Sanity testing can verify menus, functions, and commands at the surface level when there is no time for a full regression test.
- **Stress testing**: Testing how much strain the system can take before it fails. Considered to be a type of non-functional testing.
- **Usability testing**: Validating how well a customer can use a system or web application to complete a task.
- **Acceptance testing**: Verifying whether the whole system works as intended or data delivered meets specifications.

All products designated for delivery (e.g., software applications, data, information, etc.) shall go through acceptance testing at rigor determined by programming rigor and sponsor requirements and shall be documented within the requirements documentation.

6.3.1 **Software Development**

6.3.1.1 Code Review: Periodically done on applications to prevent deviations from the programming guidelines or as directed by the Technical Director/Senior Management.

6.3.1.2 Testing: All testing types may be deployed based on programming rigor and requirements.

6.3.2 **Clinical Research Data Management**

6.3.2.1 Code Review: When retrieving clinical/protected health data, the code review process is a formal process that shall be done on all SQL or Extraction code. Any projects developed under "high rigor" conditions shall undergo a code review and have it documented within the Task Tracking system. Any medium or low projects may undergo code review based on project complexity and specific project requirements.

6.3.2.2 Testing: While this team may employ several testing types, acceptance testing shall be done on all "high rigor" designated projects. Acceptance testing is accomplished by validating data results against requirements and may be done through several mechanisms (e.g., Epic access, SQL checks, etc.)

6.3.3 **Research Database Support**

6.3.3.1 Code Review: Periodically done on applications to prevent deviations from the programming guidelines or as directed by the Technical Director/Senior Management.

6.3.3.2 Testing: All testing types may be deployed based on programming rigor and requirements.

6.3.4 **Systems Management**

6.3.4.1 Code Review: Periodically done on applications to prevent deviations from the programming guidelines or as directed by the Technical Director/Senior Management.

6.3.4.2 Testing: All testing types may be deployed based on programming rigor and requirements; however, performance, stress, and acceptance testing are the most common.

6.4 Delivery

Delivery will take on many forms based on the project type, deliverable, and development team.

6.4.1 **Software Development:** Software applications are delivered by moving all code into the production environment, finalizing operating parameters (e.g., establishing user accounts, roles, etc.), and ensuring a final version of the code is in version control.

6.4.2 **Clinical Research Data Management:**

6.4.2.1 **Confidential Data:** Delivered via a secure pre-defined and agreed on secure delivery method (e.g., REDCap, IDEAS, encrypted email, etc.)

6.4.2.2 **Aggregate Data:** Delivered via pre-defined and agreed-on secure delivery method (e.g., REDCap, IDEAS, encrypted email, etc.)

6.4.2.3 **ETL Code:** Used internally to support infrastructure, so delivery is defined as moving into the production or active environment. All finalized code shall be submitted to version control.

6.4.3 **Research Database Support**

6.4.3.1 **Confidential Data:** Delivered via a secure pre-defined and agreed on secure delivery method (e.g., REDCap, IDEAS, encrypted email, etc.)

6.4.3.2 **Aggregate Data:** Delivered via pre-defined and agreed-on secure delivery method (e.g., REDCap, IDEAS, encrypted email, etc.)

6.4.3.3 **DDL/DML Code:** Used internally to support infrastructure, so delivery is defined as moving into the production or active environment. All finalized code shall be submitted to version control.

6.4.4 **Systems Management**

6.4.4.1 Third-Party Applications: Delivered by moving all code into the production environment and finalizing operating parameters (e.g., establishing user accounts, roles, etc.). Version control is unnecessary, as code can always be retrieved from the source.

6.4.4.2 Locally Developed Packages: Delivered by moving all code into the production environment, finalizing operating parameters (e.g., establishing user accounts, roles, etc.), and ensuring a final code set is in version control.

6.4.4.3 Scripts: Delivered by moving all code into the production environment, finalizing operating parameters (e.g., establishing user accounts, roles, etc.), and, as needed, ensuring a final version of the code is in version control.

7.0 SOP Deviations

Deviations from this and all SOPs are handled according to CRI.POL.001 *Clinical Research Informatics Quality Management System (QMS)*.

8.0 Review & Revisions

Review and revisions of this and all SOPs are handled according to CRI.POL.001 *Clinical Research Informatics Quality Management System (QMS)*.

9.0 Attachments

10.0 Revision History

Version No.	Revision Date	Description of Revision