01/22/2020

NCDIT-T GIS Unit Code Standards

VERSION 1.1

Contents

Document Control

Author	Version	Date	Description
GIS Unit	1.0	12/11/2019	Original Document
GIS Unit	1.1	01/22/2020	Updated to reflect the Corrective Action Plan process in the Business Process Diagram.

Overview

The State of North Carolina Department of Information Technology - Transportation (NCDIT-T) reserves the right to conduct **Software Code Quality Checking** (**SCQC**) and **Database Quality Checking (DQC)** as needed throughout the project life cycle. SCQC and DQC are used to determine the *usability*, *efficiency performance*, and *maintainability* of new software solutions and databases. These may include both standalone deliverables and those included as part of software solutions, and the quality checks can employ a combination of automated scans and/or manual audits.

NCDIT-T's expectation is that vendors will review this standard and related standards referenced within this document throughout the software development process, and that the final product will comply with the standards that are in force when the final product is delivered.

Information gathered from SCQC reviews is used by NCDIT-T to determine the viability of continuing the Vendor's work upon each scheduled delivery (i.e., Sprint, Milestone, or other scheduled delivery).

SCQC and DQC may be performed via any combination of manual and automated check tools available to NCDIT-T, which include static code analysis, static security analysis, dynamic code analysis, dynamic security analysis and architecture analysis.

In addition to SCQC and DQC, adherence to the NC Statewide Information Security Manual is expected. The vendor shall address any specific security measures pertaining to components that are identified in NCDIT-T vulnerability scans. Common program error conditions and exceptions shall be handled and logged, including validation of *type*, *value*, and *length* of data entry fields, in such a way that an error is not generated without enough information to correct the issue. All software and databases will be deployed in NCDIT-T's server environment, which follows the current NCDOT GIS Architectural Standards. The vendor is expected to consider the specifics of this environment when designing a software solution as these standards impact how solutions are designed and delivered to NCDIT-T.

It is recommended that prior to meeting with the agency business customer, the vendor should meet with the GIS Unit staff to discuss the vision for the new solution to ensure products are started on a viable path.

Appendix A provides an overview of IT industry software quality and technology standards with which the vendor deliverables should comply:

- Consortium for IT Software Quality (CISQ)
- Object Management Group (OMG)
- International Organization for Standardization 25010 Standard (ISO)

Acceptance Process for Software and Database Deliverables

All components necessary to compile the deliverable into an executable system must be received by the NCDIT-T Project Manager prior to the submittal being considered complete. Typical components may include files such as source code files, databases, database scripts, configurations, or other source-level components necessary to compile the deliverable in NCDIT-T's environment. A Business Process Diagram is included in Appendix B to clarify how this process might flow.

Abiding by the approved project schedule for the vendor's specific contract, once the Vendor presents deliverable(s) for acceptance, the NCDIT-T Project Manager will either:

1.1. Accept the Deliverable(s) in writing
1.2. Reject the Deliverable(s) in writing, including why the Deliverable(s) is not accepted

If a deliverable is rejected or returned, the NCDIT-T Project Manager will identify representative instances of non-compliance issues or areas to be corrected in writing to enable Vendor to accomplish corrections. Once notified of the issues with the deliverable, the vendor is expected to remedy all instances of those issues throughout the entire deliverable and resubmit the deliverable to NCDIT-T for acceptance at no additional charge.

All deliverables are by default deemed not accepted unless written acceptance is provided to the vendor by the NCDIT-T GISU Project Manager.

If NCDIT-T rejects a deliverable, the vendor is expected to provide a resolution or propose an acceptable solution within 14 business days. If the vendor fails to respond within this timeframe, NCDIT-T may enforce at its discretion the Termination for Cause portion of the contract with the vendor.

Acceptance Criteria for Software and Database Deliverables

NCDIT-T GISU will evaluate the quality of Software Deliverables using a combination of CAST Application Intelligence Platform (AIP), other CISQ-conformant technology, and any necessary manual inspections using the following measures:

Application Quality Measure	Measurement Criteria (Appendix C)
Reliability	Delivered codebase will incorporate error handling as per the approved design and adhere to the rules in the CISQ Reliability Specifications.
Security	Delivered codebase will not contain any security vulnerability issues as specified by the rules in the CISQ Security specification.
Performance Efficiency	Delivered codebase will not contain any performance efficiency issues, as specified by the rules in the CISQ Performance Efficiency specification.
Maintainability	Delivered codebase will have maintainability criteria (*i.e.*, flexibility to make changes easily) implemented as per the rules in the CISQ Maintainability specification.

Appendix A - Standards and Standards Bodies

Object Management Group (OMG):

The Object Management Group® (OMG®) is an international, open membership, not-for-profit **technology standards** consortium. Founded in 1989, OMG standards are driven by vendors, end-users, academic institutions and government agencies. OMG Task Forces develop enterprise integration standards for a wide range of technologies and an even wider range of industries. OMG's modeling standards, including the Unified Modeling Language (UML) and Model Driven Architecture (MDA), enable powerful visual design, execution and maintenance of software and other processes. OMG also hosts organizations such as the user-driven information-sharing Cloud Standards Customer Council (CSCC) and the IT industry software quality standardization group, the Consortium for IT Software Quality (CISQ).

Consortium for IT Software Quality (CISQ):

The Consortium for IT Software Quality (CISQ) is an IT industry leadership group comprised of IT executives from the Global 2000, system integrators, outsourced service providers, and software technology vendors committed to introducing a computable metrics standard for measuring software quality & size. CISQ is a neutral, open forum in which customers and suppliers of IT application software can develop an industry-wide agenda of actions for improving IT application quality to reduce cost and risk.

The CISQ Quality Characteristic rules are consistent with ISO/IEC 25010 definition. They are designed to be automated on source code to identify critical vulnerabilities in the software that are severe enough that they need to be fixed. Combined with a sizing measure, a density metric is produced for each quality characteristic. Thresholds can be set for each characteristic.

The CISQ Quality Characteristic Measures cover eighty-six well-established software engineering rules to ensure secure, reliable, efficient and easy to maintain software. The table at http://it-cisq.org/standards/ shows a snapshot of software engineering rules contained in the measurement of each quality characteristic at the unit level and system level. It is considered best practice to identify a critical set of these rules based on the specific environment and goals of the client – either at the application level or enterprise wide.

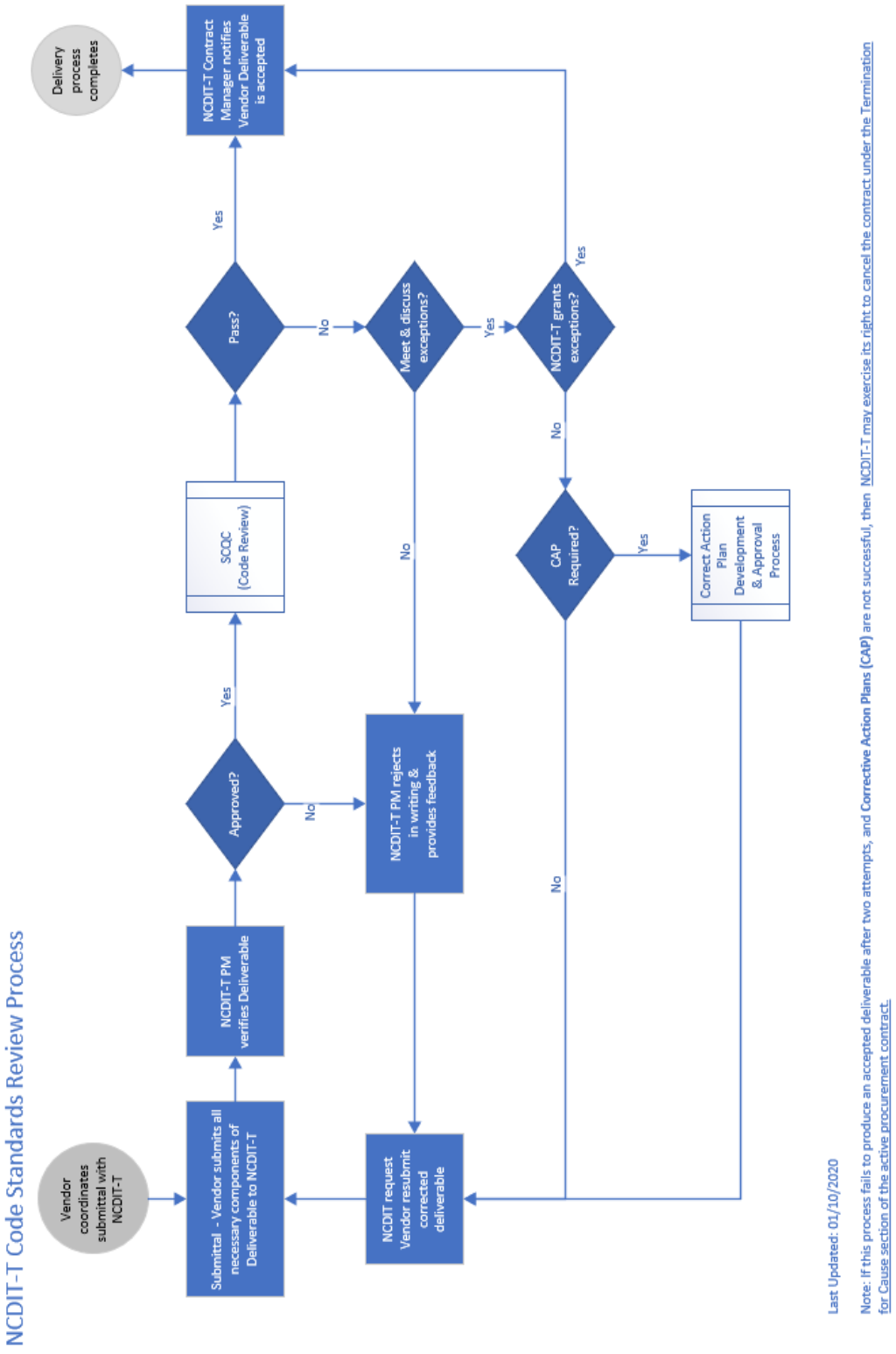
International Organization for Standardization (ISO) 25010 Standard:

1. A quality in use model composed of five characteristics (some of which are further subdivided into sub-characteristics) that relate to the outcome of interactions when a product is used in a specific context. This system model is applicable to the complete human-computer system, including both computer systems in use and software products in use.

2. A product quality model composed of eight characteristics (which are further subdivided into sub-characteristics) that relate to static properties of software and dynamic properties of the computer system. The model is applicable to both computer systems and software products.

The characteristics defined by both models are relevant to all software products and computer systems. The characteristics and sub-characteristics provide consistent terminology for specifying, measuring and evaluating system and software product quality. They also provide a set of quality characteristics against which stated quality requirements can be compared for completeness.

Appendix B – Code Acceptance Process Diagram



Appendix C – Critical Rules

This list contains the known Critical Rules that may potentially cause GISU to fail a deliverable, but it is not an exhaustive list of rules that may cause a deliverable to be failed.

Reliability – Error and Exception Handling

Rule	Technologies
Avoid empty catch blocks	All
Avoid empty finally blocks	All
Avoid return statement in finally block	All
Avoid catching an exception of type Exception, RuntimeException, or Throwable	C#, VB.NET, JEE
The exception Exception should never be thrown. Always Subclass Exception and throw the subclassed Classes	C#, VB.NET, JEE
Avoid catch-all except blocks with empty handlers	Python

Secure Coding – Input Validation and Encapsulation

Rule	Technologies
Avoid disabling Strict Contextual Escaping (SCE) when created	All
Avoid disabling withCredentials option for the httpProvider	All
Avoid using unsanitized AngularJS application	All
Avoid cross-site scripting DOM vulnerabilities (CWE-79)	C#, VB.NET, JEE
Avoid file path manipulation vulnerabilities (CWE-73)	C#, VB.NET, JEE
Avoid LDAP injection vulnerabilities (CWE-90)	C#, VB.NET, JEE
Avoid Log forging vulnerabilities (CWE-117)	C#, VB.NET, JEE
Avoid OS command injection vulnerabilities (CWE-78)	C#, VB.NET, JEE
Avoid SQL injection vulnerabilities (CWE-89)	C#, VB.NET, JEE
Avoid XPath injection vulnerabilities (CWE-91)	C#, VB.NET, JEE
Avoid having iframe inside a tag	HTML5
Avoid hosting HTML code in iframe srcdoc	HTML5
Avoid using autofocus and onblur in submitted markup	HTML5
Avoid using autofocus and onfocus in submitted markup	HTML5
Avoid using console.log()	HTML5
Avoid using eval()	HTML5
Avoid using import with external URI	HTML5
Avoid using JavaScript or expression in the CSS file	HTML5
Avoid using oninput in body containing input autofocus	HTML5
Avoid using onscroll event with autofocus input	HTML5
Avoid using setData in ondragstart with attribute draggable set to true	HTML5
Avoid using setInterval()	HTML5
Avoid using source tag in video/audio with event handler	HTML5
Avoid using submit markup related to "form" with id attribute	HTML5
Avoid using submitted markup containing "form" and "formaction" attributes	HTML5

Avoid using video poster attributes in combination with JavaScript	HTML5
Avoid white listing the "dirname" attribute in user generated content	HTML5
Enable Content Security Policy when creating an AngularJS application	HTML5
Avoid Python string interpolations to prevent SQL injections	Python
Avoid disabling certificate check when requesting secured urls	Python
Avoid using eval()	Python
Avoid using exec	Python
Protect sensitive data in config files from disclosure	Python
Avoid hardcoded passwords	Python
Avoid using MD5 hashes to hash passwords or to encrypt data	Python
Avoid using unsecured cookie	Python

Performance Efficiency

Rule	Technologies
Avoid "SELECT *" queries	ALL
Avoid the use of the default JavaScript implementation [].forEach in AngularJS web app	ALL
Avoid using SQL queries inside a loop	ALL
Avoid instantiations inside loops	C#, VB.NET
Close SQL connection ASAP	C#, VB.NET
Use dedicated stored procedures when multiple data accesses are needed (ASCPEM-PRF-10)	C#, VB.NET, JEE
Avoid calling a function in a termination loop	HTML5
Avoid using querySelectorAll	HTML5
Avoid leaving open file resources	Python
Avoid using a web service with Python urllib.request inside a loop	Python
Avoid using a web service with Python urllib2 inside a loop	Python

Maintainability

Rule	Technologies
Avoid cyclical calls and inheritance between namespaces content	C#, VB.NET
Avoid declaring public Fields	C#, VB.NET
Avoid using untyped DataSet	C#, VB.NET
Avoid using Web SQL databases	HTML5

Appendix D – Data Development and Documentation Standards

The Database Standards are owned and maintained by NCDIT-T Information Technology, and the GISU is required to abide by these standards. Therefore, all externally sourced products must be delivered with these standards in mind. It is the vendor's responsibility to get the most recent Database Standards documentation from the GISU Spatial Data Management Group at the onset of each new project. Below is an example based on current standards, but these are subject to change at any time and should not be relied upon without verification for any new development.

In addition, the Vendor should be aware that Database Standards generally follow ESRI recommendations. The following are standards specifically related to database deliverables.

Dataset Naming Management

The following standards and guidelines are provided for the purpose of aiding NCDOT/IT software designers, analysts, and programmers in constructing and managing IBM DataSet files. The use of these guidelines will support the use of standard dataset naming conventions and assist in establishing improved dataset management techniques and processes throughout the system.

DATASET NAMING

DataSets established on either tape or disk are to follow the below naming standards -

Application DataSets:

DOT.BBB.CCCCCC.XX------.XX------.XX

DOTT.BBB.CCCCCC.XX------.XX------.XX

1st Qualifier Format: DOT	Three letter departmental code for PROD (Production)
1st Qualifier Format: DOTT	Four letter departmental code for TEST (Testing)
2nd Qualifier Format: BBB	Three letter application bill code
3rd Qualifier Format: CCCCCC	Six positions First four positions are either TEST or PROD Fifth position is either B or O B = datasets used in batch only O = datasets used by online or online and batch Sixth position is as follows: L = load libraries P = partitioned datasets S = sequential files V = VSAM files

4th Qualifier Format: XX	Optional qualifier, up to eight positions. If unique to a given subsystem, include the subsystem (ex: unit, sytm,cltr, prod)
5th Qualifier Format: XX	Optional qualifier, up to eight positions
6th Qualifier Format: XX	Optional qualifier, up to eight positions

Note: Name cannot exceed 44 positions <u>including the periods</u>.

Personal TSO DataSets:

DOTTSO.TEAMXX.TSXXYNN(member)

XX = Team identifier (e.g. 68, 69, 70)

Y = Employee type (e.g. S = state employee, C = contractor)

NN = Unique identifier (e.g. AA, A1, etc.)

Member – up to eight-character member name

Personal Non-TSO DataSets:

TSXXYNN.ZZZZZZZZ

XX = team identifier (e.g. 68,69,70)

Y = employee type (e.g. S = state employee, C = contractor)

Z = unique identifier

1st Qualifier is equal to the individual's RACF-ID.
2nd Qualifier is to be determined by individual and should be descriptive of the data in the data set.

DATASET MANAGEMENT

Application DataSets:

Any application-related DataSets (test or production) should be periodically reviewed by the Project Team Leaders to determine if there are candidates for elimination. All DataSets that are obsolete (no longer needed for any reason) should be deleted from the system. This is especially true for DataSets that were established for testing purposes during application development.

Personal DataSets:

All Personal DataSets (TSO or Non-TSO) should be periodically reviewed by the DataSet owners and deleted from the system if no longer needed. All Personal DataSets should be deleted by the owner at the time of employment termination.

CLASS WORD	DEFINITION (comment)
Address	Identifies a place (building, post office box, mail stop, etc.) where something is or could be located or placed (i.e. place of business or location). Examples: Employee Home Address, Employee Street Address.
Age	Time reflected in terms of years. Possibly a derived count Examples: Employee Retirement Age, Policy Maturity Age.
Amount	A numeric monetary value. Examples: Unit Cost Amount, Insurance Monthly Premium Amount. Note: Recommended default standard definitions are (9,2) for line items, and (11,2) for total accumulations. Exceptions to these definitions will be considered on an individual case basis.
Average	A numeric value representing an arithmetic mean. Example: Employee Test Score Average.
Code	An abbreviated representation of a value that is a member of a set of two or more values. Examples: Vehicle Paint Code, Employee Pay Code.
Count	A numeric non-monetary value derived from a counting process. Not a pre-determined or fixed value. Examples: DOT Employee Count, Hours Worked Count.
Cycle	A numeric time interval in which a regularly repeated event or sequence of events occurs. Non-data element entities. Examples: Payroll Processing Cycle, Employee Evaluation Cycle.
Date	A calendar date identifying a specific chronological instance. Example: Employee Birth Date.
Description	Alphanumeric data of variable content and structure describing persons, places, events, things, times, actions, etc.

Height	A numeric value representing the height (feet, inches, etc.) of an object or person. Examples: Customer Inches Height, Bridge Feet Height.
Identifier (or Id)	A unique alphanumeric value which identifies and is characteristic of a person, place, or thing; and which remains constant. Example: Employee Fingerprint Identifier.
Indicator (or Ind)	A single position code of Y (Yes), N (No), or blank. Examples: Employee Active Status Indicator, Employee Retirement Eligibility Indicator.
LargeObject	Large and unstructured data such as text, image, video, and spatial data up to 4 gigabytes in size.
Length	A numeric value (miles, feet, inches, meters, centimeters, etc.) representing the length of an object. Example: Bridge Span Length, Vehicle Body Length
Name	Alphanumeric data that represents a person, place, thing, organization, or concept. Examples: Employee Last Name, Department Name, Street Name.
Number	Alphanumeric data used to uniquely identify, categorize, or classify a person, place, thing, organization, or concept. Examples: Employee Social Security Number, Manufacture's Part Number, Building Number, Document Number.
Percent	A numeric value representing parts of 100. Examples: Employee Days Absent Percent, Employee FICA Withholding Percent.
Quantity	A pre-determined, non-monetary, non-accumulating numeric value. Examples: Vehicle Maximum Passenger Quantity, Authorized Leave Day Quantity.
Rate	A measured numeric value that is a proportional share or factor in relation to another quantity or amount. Examples: Employee Hourly Pay Rate, Vehicle Miles Per Hour Rate.

Record	An occurrence of logically associated data elements. Examples: Payroll Timestamp Record, Employee Traffic Violations Record.
Score	Numeric representation of the achievement of a task, based on a specific scale. Example: Driver's License Test Score.
Text	Unformatted alphanumeric informative data used to clarify or summarize characteristics of a person, place or thing. Examples: Employee Evaluation Text, Project Review Text.
Time	Time (hours, minutes, and seconds) identifies a duration, or a specific instant, within a 24-hour day. Examples: Payroll Elapsed Run Time, Employee Work Start Time.
Timestamp	A special record of an instant in time. Generally composed of hour, minute, second data.
Title	A distinctive heading, or a descriptive or distinguishing name. Examples: Employee Job Title, Report Title, Book Title.
Weight	A numeric value (tons, pounds, ounces) representing the heaviness of an object. Example: Gross Vehicle Weight.
Width	A numeric value (miles, feet, inches, meters, centimeters, etc.) representing the width of an object. Example: Bridge Width, Vehicle Body Width.

SQL Server Standard Data Naming Conventions v1.0 2011-12-09

The following guidelines are provided for the purpose of aiding NCDOT/IT software designers, analysts and programmers in constructing data names for SQL Server database tables and data elements. The use of these guidelines will lead to the improved application of standard naming conventions and consistency throughout IT.

Data Name Definitions

A “data name” is defined and developed for the purpose of uniquely defining data for use in computer software/program development and for the purpose of clarifying and describing the data to the user or software developer. SQL Server data names should be developed using only logical form. The purpose of the logical form is to spell out the data name using full descriptive words for clarity and recognition.

Logical Data Name Construction

Logical data names should be made up of unabbreviated words and phrases. A logical data name will always end in a class word preceded by as few modifier words as necessary to make the name unique and descriptive (see the MS SQL Server Standard Class Words document). The general guidelines for logical data names for SQL Server are as follows:

- Use unabbreviated words and phrases
- Append a class word to a data name (such as ID/Identifier, Number, Date, Address, Code)
- Use Pascal casing, including the usage of upper and lower case for data names
- Do not use special characters in data names
- Do not use spaces as word separators in data names
- Limit to the maximum number of characters for SQL Server data names

Examples of valid data names are as follows:

- ApplicationFormID
- CommercialDriverName
- BridgeStatusCode

Table Name Construction

Table names (or Entities) may be constructed using full words. Full words (rather than abbreviations) should be used in singular form (i.e., Customer, not Customers; BridgeSpan, not BridgeSpans). Table name should not include (or end with) the word "TABLE."

Additional Information

All new schemas or modifications to existing schemas must be reviewed with the DBA's.

The DBA review of the new schemas and modifications to existing schemas will be an iterative process as outlined below:

1. **Share database plans with DBA's for new initiatives and projects**
   Communicate with the DBA's that a new schema or modification to an existing schema is forthcoming via email, telephone or meeting.

2. **Send documented database requests to DBA's via email in free-style format and/or attached database request form outlining what the DBA's need to do.**
   For small-to-medium sized initiatives or projects, the requested database work should be submitted to the DBA's via email in freestyle format and/or as an attached database request form.

For small-to-medium sized projects, the database request form is an option for the applications team to request database work.

For large initiatives and projects, the designated database request form is required to request database work.

3. **Keep DBA's informed of any changes to the schemas during the development stage of the initiative or project**
   The applications teams should inform the DBA's of any revisions for the schema(s) during the development stage. On a regular basis, the DBA's should be informed of any revisions to the new or existing schemas; the associated scripts for these modifications should be sent to the DBA's, if applicable.

4. **Include DBA representative as member of initiative or project team**
   A DBA representative should be on the project team for all initiatives and projects requiring anticipated DBA work to ensure that the DBA's are apprised of the work requiring DBA involvement.

The vendor should meet early with the Spatial Data Management Group to get the most recent database requirements documents and establish any additional requirements or documentation necessary for contracted deliverables.

Appendix E – Example Results from CAST

Summary

CISQ	Total Vulnerabilities	Added Vulnerabilities	Removed Vulnerabilities
CISQ-Maintainability	743	37	1,353
CISQ-Performance-Efficiency	399	0	421
CISQ-Reliability	305	4	1
CISQ-Security	47	9	3

Figure 1 CAST Example Summary from the report provided to the vendor.

CISQ-Maintainability	Total Violations	Added Violations	Removed Violations
ASCMM-MNT-1 Control Flow Transfer Control Element outside Switch Block	**44**	**0**	**0**
Avoid using GOTO statement	0	0	0
Avoid missing default in switch statements	44	0	0
ASCMM-MNT-4 Callable and Method Control Element Number of Outward Calls	**308**	**0**	**2**
Avoid Artifacts with High Fan-Out	308	0	2
ASCMM-MNT-5 Loop Value Update within the Loop	**24**	**0**	**0**
Avoid String concatenation in loops (.NET)	24	0	0
Avoid having transaction with the	0	0	0

Figure 2 CAST Example Summary by Type

Violations

Objects in violation for rule Close SQL connection ASAP

Violations: 1

Rationale:

To avoid SQL connection leakage, it is highly recommended to close the SQL connection as soon as you are done using it, preferably within the Method that opened it.

Description:

SQL connection should be closed within the Method / Function / Sub that opened it

Remediation:

Close the SQL connection within the same Method

Violation #1 Close SQL connection ASAP

Object Name: [redacted]

Object Type: C# Method

Status: unchanged

Associated Value: 1

File path: [redacted]

```
420 :     }
421 :
422 :     // Creates and opens an internal SQL connection
423 :     private void OpenConnection()
424 :     {
425 :         _connection = CreateSqlConnection();
426 :         _connection.Open();
```

Figure 3 CAST Violation Details Example

Appendix F – Glossary of terms specific to this standard

Regarding this standard, State of North Carolina Department of Information Technology - Transportation (NCDIT-T) defines the following terms:

Term	Definition
Applications Development Group (ASG)	The group within the NCDIT-T GIS Unit responsible for maintaining enterprise applications.
Architectural Standards Review	A review of a new solution's infrastructure design and including a review of compliance with NC IT Security policies and best practices, which may be supported by automated tools, but are usually conducted by manual walk-through of documentation and visual inspection of the code.
Critical Rules	A subset of CISQ, OMG, ISO, or other standard rules that, if violated, will render a software product unacceptable to NCDIT-T and prevent NCDIT-T from taking final delivery. A strict interpretation of the Critical Rules is desired, and compliance with Critical Rules can only be waived via an Exception by expressed written permission by NCDIT-T. Violation of Critical Rules is cause for automatic rejection of deliverables.
Database Quality Checking (DQC)	Reviews of the database organization, fields, feature classes, look-up tables, domain tables and other database components and related artifacts (such as metadata), with the goal of obtaining information necessary to assess the quality of the database currently under development.
Dynamic Program Analysis	The analysis of computer software and related documentation that is performed by executing programs built from that software on a real or virtual processor.
Dynamic Security Analysis	The analysis of computer software that is performed by executing programs to detect and report weaknesses that can lead to security vulnerabilities.
Exception	An Exception is written permission to not comply with Critical Rules and can only be granted by NCDIT-T. If Exceptions are not granted, the vendor is expected to ensure the deliverable complies with all Critical Rules.

GIS Unit (GISU)	The NCDIT-T GIS Unit serves as the enterprise GIS support for DOT customers.
Maintainability	Ability to understand and modify software quickly
NCDIT-T	Transportation Division of NCDIT (formerly DOT IT), responsible for serving the NC DOT.
NCDIT-T GISU Project Manager	The project manager responsible for execution of projects, which in the context of this standard also manages the vendor contracts for code and database deliverables.
North Carolina Department of Information Technology (NCDIT)	Leading provider of IT services to state agencies, local governments, and educational institutions across North Carolina. Services include hosting, network, telecommunications, desktop computing and unified communications, including email and calendaring.
Performance Efficiency	Ability to avoid response degradation, resource overuse
Reliability	Ability to avoid outages and to recover operations quickly
Software Code Quality Checking (SCQC)	Reviews of the source code, executables, and related artifacts, with the goal of obtaining information necessary to assess the quality of the new software solution currently under development.
Spatial Data Management Group (SDMG)	The group within the NCDIT-T GIS Unit responsible for maintaining enterprise databases.
Static Code Analysis	The analysis of computer software and related documentation that is performed without executing programs built from the software.
Static Security Analysis	The analysis of computer software that is performed without executing programs to detect and report weaknesses that can lead to security vulnerabilities.
Usability	The extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use.

Appendix G – Related Links

Software

- Cast Software (https://www.castsoftware.com/)
- This page – https://technologies.castsoftware.com/rules - which is public facing, lists all of the Standards that CAST AIP supports. Users can drill down to information on each Standard, including what Rules CAST AIP validates against and how the rule is administered.
- Sample CAST Reports (CWE-Top-25 Security Report) https://doc.castsoftware.com/display/DOCCOM/CAST+Report+Generator+-+Sample+Reports

Code Standard

CISQ Standard Links:

- CWE List https://www.it-cisq.org/pdf/cisq-weaknesses-in-ascqm.pdf
- Technical Debt https://www.omg.org/spec/ATDM/1.0/PDF

Architectural Standards

- NCDOT/NCDIT https://connect.ncdot.gov/resources/gis/Pages/GIS-Standards.aspx
- State IT Standards https://it.nc.gov/services/service-directory/it-architecture

Appendix H - Roles and Responsibilities

The RACI chart below outlines the roles and responsibilities for completing a successful software delivery to NCDIT-T GISU.

- **Responsible**: This group/person does the work to complete the task.
- **Accountable**: This group/person signs off on task and may perform final reviews.
- **Consulted**: This group/person provides input based on how the task will be completed.
- **Informed**: This group/person needs to be included in communications for each task.

Task	Vendor Team	Vendor PM	NCDIT-T GISU Team	NCDIT-T PM
Collect/Verify all requirements for software development	R	A	C	I
Review and comply with all North Carolina IT Standards (includes periodic reviews for changes throughout project)	R	A	I	C
Review and comply with all NCDIT-T GISU GIS Standards (includes periodic reviews for changes throughout project)	R	A	C	I
Delivery of software	R	A	I	I
Confirms software deliverable is complete			R	A
Notifies vendor if it is not complete	I	I	I	R/A
Performs SCQC & DQC processes			R	A
Communicates to vendor results of SCQC & DQC processes	I	I	I	A
Addresses defects in deliverables	R	A	C	I
Approves any deviations from current GISU standards necessary for successful delivery			R	A