CISQ Weakness Descriptions

This document presents descriptions of the 86 weaknesses contained in the 4 CISQ Quality Characteristic measures. These descriptions have been simplified from their description in the their published OMG specifications that used formalisms from other OMG meta-models to specify the weaknesses for representation in machine-processable XMI notation. The tables below present each weakness with its unique CISQ identifier, a brief descriptive name, and a fuller description of the weakness presented as a recommendation for remediation.

Reliability Weaknesses

a measure of the extent to which software contains weaknesses that cause outages, unexpected behavior, instability, data corruption, long recovery times, or other related problems.

CISQ identifier	Descriptor	Remediation
ASCRM-CWE-120	Buffer overflow	Remove instances where the content of the first buffer is moved into the content of the second buffer while their allocated sizes are incompatible
ASCRM-CWE-252-data	Unchecked return parameter from data handling operations	Remove instances where a function, method, procedure, stored procedure, sub-routine, etc. executes a CRUD SQL statement, yet the return code value of the action is not checked anywhere
ASCRM-CWE-252-resource	Unchecked return parameter from resource handling operations	Remove instances where a function, method, procedure, stored procedure, sub-routine, etc. reads, writes, or manages an external resource, yet the return code value of the action is not checked anywhere
ASCRM-CWE-396	Catch of overly broad exception types	Remove instances where a function, method, procedure, stored procedure, sub-routine, etc. contains a catch of an exception whose type is part of a list of overly broad exception types
ASCRM-CWE-397	Throw of overly broad exception types	Remove instances where a function, method, procedure, stored procedure, sub-routine, etc. throws an exception whose type is part of a list of overly broad exception types
ASCRM-CWE-456	Uninitialized data element	Remove instances where a variable, field, member, etc. is declared, then is evaluated without ever being initialized prior to the evaluation
ASCRM-CWE-674	Recursion	Remove instances in which a control element initiates an execution path that contains itself
ASCRM-CWE-704	Incompatible data type conversion	Remove instances where a variable, field, member, etc. is declared with a data type, and then is updated with a value from a second data type that is incompatible with the first data type

ASCRM-CWE-772	Unreleased resource	Remove instances where a platform resource (messaging, lock, file, stream, directory, etc.) is allocated and assigned a unique resource handler that is used throughout the application, but is never released
ASCRM-CWE-788	Memory access after end of buffer	Remove instances where a value is used as an index in a 'Read' or 'Write' access to a buffer; yet none of the operations performed prior the buffer access check the value with regards to the buffer's maximum size
ASCRM-RLB-1	Empty exception block	Remove instances where an exception handling block (such as catch and finally blocks) of a function, method, procedure, stored procedure, sub-routine, etc. does not contain any instruction
ASCRM-RLB-2	Missing serialization control element	Remove instances where a serializable field, member, etc. has no serialization operation. Notes: * in the case of technologies with classes and interfaces, this means situations where the serializable field is from a class that implements a serializable interface but does not implement a serialization method as part of its list of methods * the serializable nature of an element is technology dependent, for example, serializable capabilities come from sources such as a serializable attribute in .NET and inheritance from the java.io.Serializable interface in Java
ASCRM-RLB-3	Serialized data element containing non-serialized items	Remove instances where a serializable field, member, etc. is composed of a non-serializable data element. Notes: * in case of technologies with classes and interfaces, this means situations where the serializable field is from a class that is serializable but owns the non-serializable field * the serializable nature of an element is technology dependent; for example, serializable capabilities come from sources such as a serializable attribute in .NET and inheritance from the java.io.Serializable interface in Java
ASCRM-RLB-4	Persistent data without proper comparison controls	Remove instances where the persistent variable, field, member, etc. has no dedicated operation handling comparison operations. Note: • In case of technologies with classes, this means situations where a persistent field is from a class that is made persistent while it does not implement methods from the list of required comparison operations (a JAVA example is the list composed of {'hashCode()','equals()'} methods)

ASCRM-RLB-5	Improper runtime resource management	Remove instances where an application is running on an application server, yet uses a low-level resource management API (I/O, sockets, class loaders, etc.) and not the resource management API offered by the application server
ASCRM-RLB-6	Improper copy capabilities for data pointers	Remove instances where a variable, field, member, etc. contains a pointer but no dedicated copy operation or copy constructor
ASCRM-RLB-7	Self-destruction	Remove instances where a class can self-destruct (an example of a self-destruction in C++ is 'delete this')
ASCRM-RLB-8	Variadic parameters	Remove instances where a function, method, procedure, stored procedure, sub-routine, etc. has a variable number of parameters, thanks to the variadic parameter in its signature
ASCRM-RLB-9	Improper equality comparisons of float-type numerical data	Remove instances where the float values of a variable, field, member, etc. are compared for equality using regular comparison operators (an example in JAVA, is the use of '= =' or '!=')
ASCRM-RLB-10	Circumventing data access routines	Remove instances where a function, method, procedure, stored procedure, sub-routine, etc. executes a data access outside of a dedicated data access component, thus circumventing the intended design for data access. Notes: • the dedicated data access component can be either client-side or server-side, which means that data access components can be developed using non-SQL languages • if there is no dedicated data access component, every data access is a violation
ASCRM-RLB-11	Non-final static data in a multi-threaded environment	Remove instances where a function, method, procedure, stored procedure, sub-routine, etc. owns a non-final static variable, field, member, etc. while it operates in a multi-threaded environment
ASCRM-RLB-12	Improper locking of singleton classes	Remove instances where a singleton class is instantiated without any prior locking mechanism being activated
ASCRM-RLB-13	Cyclic dependencies	Remove instances where a module has references that cycle back to itself, e.g., the existence of cycles between packages in JAVA
ASCRM-RLB-14	Parent class referencing child class	Remove instances where a parent class has a reference to one of its child classes, directly or indirectly via its methods and fields.
ASCRM-RLB-15	Class with virtual method missing destructor	Remove instances where a class contains a virtual method, yet the class does not declare any virtual destructor
ASCRM-RLB-16	Parent class missing virtual destructor	Remove instances where, for languages in which custom destructors can be written, the parent has no virtual destructor

ASCRM-RLB-17	Child class missing virtual destructor	Remove instances where, for languages in which custom destructors can be written, the child class does not have its own virtual destructor, while its parent class has a virtual destructor
ASCRM-RLB-18	Hard-coded network resource information	Remove instances where a variable, field, member, etc. is initialized with a hard-coded network resource identification information
ASCRM-RLB-19	Synchronous call missing timeout	Remove instances where a synchronous call is initiated but the time-out argument is not set or is set to infinite time

Security Weaknesses

a measure of the extent to which software contains weaknesses that can be exploited to gain unauthorized access to a system to steal data, cause damage, or other malicious acts.

CISQ identifier	Descriptor	Remediation
ASCSM-CWE-22	Improper path traversal	Remove instances where a user input is ultimately used in a file path creation statement, without any sanitization (based on a list of vetted sanitization functions, methods, procedures, stored procedures, sub-routines, etc.) of the user input value between the user input and the statement.
ASCSM-CWE-78	OS command injection	Remove instances where a user input is ultimately used to execute an OS command, without any sanitization (based on a list of vetted sanitization functions, methods, procedures, stored procedures, sub-routines, etc.) of the user input value between the user input and the statement.
ASCSM-CWE-79	Cross-site scripting	Remove instances where a user input is ultimately displayed to the user, without any sanitization (based on a list of vetted sanitization functions, methods, procedures, stored procedures, sub-routines, etc.) of the user input value between the user input and the statement.
ASCSM-CWE-89	SQL injection	Remove instances where a user input is ultimately used in a SQL statement, without any sanitization (based on a list of vetted sanitization functions, methods, procedures, stored procedures, sub-routines, etc.) of the user input value between the user input and the statement.
ASCSM-CWE-99	Unsanitized user input used to access a named resource	Remove instances where a user input is ultimately used to access a resource by name, without any sanitization (based on a list of vetted sanitization functions, methods, procedures, stored procedures, sub-routines, etc.) of the user input value between the user input and the statement.
ASCSM-CWE-120	Buffer overflow	Remove instances where the content of the first buffer is moved into the content of the second buffer while their allocated sizes are incompatible
ASCSM-CWE-129	Unchecked array index range	Remove instances where a user input is ultimately used in a ‘Read’ or ‘Write’ access to an array, without any range check between the user input and the array access.

ASCSM-CWE-134	Improper format string neutralization	Remove instances where a user input is ultimately used in a formatting statement, without any sanitization (based on a list of vetted sanitization functions, methods, procedures, stored procedures, sub-routines, etc.) of the user input value between the user input and the statement.
ASCSM-CWE-252-resource	Unchecked return parameter from resource handling operations	Remove instances where the function, method, procedure, stored procedure, sub-routine, etc. reads, writes, or manages an external resource, yet the value of the return code is not checked anywhere
ASCSM-CWE-327	Unvetted cryptographic algorithms	Remove instances where the application uses a cryptographic list which is not part of the list of vetted cryptographic libraries.
ASCSM-CWE-396	Catch of overly broad exception types	Remove instances where the function, method, procedure, stored procedure, sub-routine, etc. contains a catch which declares to catch an exception whose type is part of a list of overly broad exception types
ASCSM-CWE-397	Throw of overly broad exception types	Remove instances where the function, method, procedure, stored procedure, sub-routine, etc. throws an exception whose type is part of a list of overly broad exception types
ASCSM-CWE-434	Unsanitized user input in file upload statement	Remove instances where a user input is ultimately used in a file upload statement, without any sanitization (based on a list of vetted sanitization functions, methods, procedures, stored procedures, sub-routines, etc.) of the user input value between the user input and the statement.
ASCSM-CWE-456	Uninitialized data element	Remove instances where a variable, field, member, etc. is declared, then is evaluated without ever being initialized prior to the evaluation.
ASCSM-CWE-606	Unchecked input in loop condition	Remove instances where a user input is ultimately used in the loop condition statement, without any range check between the user input and the loop statement.
ASCSM-CWE-667	Improper locking of shared resources	Remove instances where the shared variable, field, member, etc., is accessed outside a critical section of the application.
ASCSM-CWE-672	Access to released or expired resources	Remove instances where the platform resource (messaging, lock, file, stream, etc.) is deallocated using its unique resource handler which is used later within the application to try and access the resource.
ASCSM-CWE-681	Incompatible numeric type conversion	Remove instances where a variable, field, member, etc. is declared with a numeric type, and then is updated with a value from a second numeric type that is incompatible with the first numeric type

ASCSM-CWE-772	Unreleased resource	Remove instances where a platform resource (CPU, messaging, lock, file, stream, etc.) is allocated and assigned a unique resource handler, and its unique resource handler is used throughout the application along a sequence of operations, but none of which is a release statement
ASCSM-CWE-789	Unchecked range of user input to a buffer	Remove instances where a user input is ultimately used in a 'Read' or 'Write' access to a buffer, without any range check between the user input and the buffer access.
ASCSM-CWE-798	Hard-coded credentials for remote resources	Remove instances where a variable, field, member, etc., is initialized with a hard-coded literal value, and ultimately used to access a remote resource.
ASCSM-CWE-835	Infinite recursion	Remove instances where a recursive function, method, procedure, stored procedure, sub-routine, etc., has no execution path to exit the recursion

Performance Efficiency Weaknesses

a measure of the extent to which software contains weaknesses that can degrade a system's performance or cause excessive use of processor, memory, or other resources.

CISQ identifier	Descriptor	Remediation
ASCPEM-PRF-1	Static block initialization	Remove instances where a variable, field, member, etc. is initialized in a static block of code
ASCPEM-PRF-2	Immutable text data	Remove instances where a function, method, procedure, stored procedure, sub-routine, etc., creates immutable text data via a string concatenation (which could be avoided by using text buffer instead)
ASCPEM-PRF-3	Static data outside of singleton class	Remove instances where a static field or member is declared as static but its parent class is not a singleton class; it does not account for final static fields or members
ASCPEM-PRF-4	Complex read/write access	Remove instances where a very large table, that is, whose number of rows exceeds a threshold value (default is 1,000,000 rows), is accessed by a SQL statement with too many joins (default threshold value for the maximum number of joins is 5), and too many sub-queries (default threshold value for the maximum number of sub-queries is 3).
ASCPEM-PRF-5	Incorrect indices	Remove instances where the syntax of the SQL SELECT statement and the index configuration of the SQL table or SQL view causes the DBMS to run sequential searches
ASCPEM-PRF-6	Excessive number of indices on large tables	Remove instance where a very large table, that is, whose number of rows exceeds a threshold value (default is 1,000,000 rows), has too many indices (default threshold value for the maximum number of indices is 3)
ASCPEM-PRF-7	Excessively large indices on large tables	Remove instances where a very large table, that is, whose number of rows exceeds a threshold value (default is 1,000,000 rows), has an index whose size is too large (default threshold value for the index range is 10)
ASCPEM-PRF-8	Resource-consuming operation in loop	Remove instances where an operation causing consumption of platform resource (messaging, lock, file, stream, directory, etc.) is directly or indirectly called within a loop body or within a loop condition
ASCPEM-PRF-9	Excessive data queries in non-stored procedure	Remove instances where a server-side non-stored procedure contains too many data queries (default value for the maximum number of data queries is 5)
ASCPEM-PRF-10	Excessive data queries in client-side code	Remove instances where a client-side function, method, sub-routine, etc., contains too many data queries (default value for the maximum number of data queries is 2).

ASCPEM-PRF-11	Data access circumventing data manager	Remove instances where a function, method, procedure, stored procedure, sub-routine, etc. accesses data without going a dedicated data manager component (as identified in the vetted data access component list), thus circumventing the authorized data access procedures
ASCPEM-PRF-12	Excessively large data element	Remove instances where a variable, field, member, etc., is an aggregate of too many (non-primitive) data types (default value for the maximum number of aggregated non-primitive types is 5)
ASCPEM-PRF-13	Data access not using connection pool	Remove instances where a function, method, procedure, stored procedure, sub-routine, etc. executes a data resource management action without using a connection pooling capability (the usage of a connection pooling capability is technology dependent; for example, connection pooling is disabled with the addition of 'Pooling=false' to the connection string with ADO.NET or the value of a 'com.sun.jndi.ldap.connect.pool' environment parameter in Java)
ASCPEM-PRF-14	Unreleased memory	Remove instances where a memory resource is explicitly allocated to a variable, field, member, etc. which is used throughout the application, but is never released.
ASCPEM-PRF-15	Unreleased data	Remove instances where a method references an object, without ever de-referencing it

Maintainability Weaknesses

a measure of the extent to which software contains weaknesses that make software hard to understand or change, resulting in excessive maintenance time and cost as well as higher defect injection rates.

CISQ identifier	Descriptor	Remediation
ASCMM-MNT-1	Control transferred outside of switch statement	Remove instances where the control flow is transferred outside a switch statement (e.g., depending on the technology, by using 'go to', 'continue', or 'break' statements)
ASCMM-MNT-2	Excessive inheritance from concrete classes	Remove instances where a class inherits from too many concrete classes (default threshold value for the maximum number of concrete class Inheritances is 1).
ASCMM-MNT-3	Hard-coded literals	Remove instances where a literal value is used to initialize a variable, field, member, etc. (exceptions are simple integers and a static constant variable, field, member, etc.)
ASCMM-MNT-4	Excessive coupling	Remove instances where a function, method, procedure, stored procedure, sub-routine, etc. has a Fan-Out value that is too large, that is, with too many references to other objects within the application. (default threshold value for the maximum number of references to other objects within the application is 5)
ASCMM-MNT-5	Condition value update within loop	Remove instances where a value of a local variable, field, member, etc. used in the condition of a loop is updated within the loop body
ASCMM-MNT-6	Excessive commented-out code	Remove instances where a function, method, procedure, stored procedure, sub-routine, etc. contains too much commented-out code (default threshold value for the maximum percentage of commented-out instructions is 2%).
ASCMM-MNT-7	Circular dependencies among modules	Remove instances where a module has references that cycle back to itself (for example, in JAVA this pattern means cycles between packages)
ASCMM-MNT-8	Excessively large file	Remove instances where a file has too many lines of code (default threshold value for the maximum number of lines of code is 1000)

ASCMM-MNT-9:	Too many/few horizontal layers	Remove instances where a model of the architectural layers of an application contains too may or too few horizontal layers (excluding the vertical utility layers) based on comparison to a threshold value. The default threshold value for the minimal number of horizontal layers is 4, and the default value for maximal number of horizontal layers is 8. Note: • This is a 'control' on the architectural model used in ASCMM-MNT-10 and ASCMM-MNT-12 to avoid defining a model designed to 'pass' these other weaknesses
ASCMM-MNT-10	Layer-spanning component	Remove instances where a function, method, procedure, stored procedure, sub-routine, etc. is part of two architectural layers as defined in a model of the application's architectural layers
ASCMM-MNT-11	Excessive cyclomatic complexity	Remove instances where a function, method, procedure, stored procedure, sub-routine, etc. has a Cyclomatic Complexity that is too large (default threshold value for maximum Cyclomatic Complexity is 20)
ASCMM-MNT-12	Layer-skipping calls	Remove instances where a function, method, procedure, stored procedure, sub-routine, etc. from a higher horizontal layer directly calls a function, method, procedure, stored procedure, sub-routine, etc. in a lower horizontal layer that is not adjacent to the upper layer making the call, as defined in a model of the application's architectural layers (this excludes the vertical utility layers that can be referenced from any horizontal layer)
ASCMM-MNT-13	Excessive parameterization	Remove instances where a function, method, procedure, stored procedure, sub-routine, etc. has too many parameters in its signature (default threshold value for the maximum number of parameters is 7)
ASCMM-MNT-14	Control element with excessive data operations	Remove instances where a function, method, procedure, stored procedure, sub-routine, etc. has too many SQL or file operations (default threshold value for the maximum number of SQL or file operations is 7)
ASCMM-MNT-15	Public data element	Remove instances where a variable, field, member, etc. is declared as public
ASCMM-MNT-16	Cross element data access	Remove instances where a method from a class accesses a field, or member from another class
ASCMM-MNT-17	Excessive inheritance levels	Remove instances where a class inheritance level is too large (default threshold value for maximum Inheritance levels is 7)
ASCMM-MNT-18	Excessive child classes	Remove instances where the class number of children of a class is too large (default threshold value for maximum number of children of a class is 10)

ASCMM-MNT-19	Element redundancy	Remove instances of copy-paste between functions, methods, procedures, stored procedures, sub-routines, etc.
ASCMM-MNT-20	Dead code	Number of instances where a function or method is unreferenced by any other code item in the application (the application determines the scope of the search for code items that could call a function or method element)