Reliability Weakness Descriptions

This document presents descriptions of the 29 weaknesses contained in the CISQ Automated Quality Characteristic Measure for Reliability. These descriptions have been simplified from their description in the published OMG® specification 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