POLITECNICO MILANO 1863

# CODE INSPECTION DOCUMENT

Paolo Antonini 858242
Andrea Corneo 849793

version 1 – 5th January 2016

GlassFish 4.1

# *Contents*

# 1
# Introduction

## 1.1 Purpose

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

## 1.2 Scope

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

## 1.3 Definitions, acronyms, and abbreviations

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem.

Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

## *1.4 References*

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

## *1.5 Overview of the document*

Fusce mauris. Vestibulum luctus nibh at lectus. Sed bibendum, nulla a faucibus semper, leo velit ultricies tellus, ac venenatis arcu wisi vel nisl. Vestibulum diam. Aliquam pellentesque, augue quis sagittis posuere, turpis lacus congue quam, in hendrerit risus eros eget felis. Maecenas eget erat in sapien mattis porttitor. Vestibulum porttitor. Nulla facilisi. Sed a turpis eu lacus commodo facilisis. Morbi fringilla, wisi in dignissim interdum, justo lectus sagittis dui, et vehicula libero dui cursus dui. Mauris tempor ligula sed lacus. Duis cursus enim ut augue. Cras ac magna. Cras nulla. Nulla egestas. Curabitur a leo. Quisque egestas wisi eget nunc. Nam feugiat lacus vel est. Curabitur consectetuer.

# 2
# *Classes and functional analysis*

Classes that were assigned to the group: <state the namespace pattern and name of the classes that were assigned to you>

Functional role of assigned set of classes: <elaborate on the functional role you have identified for the class cluster that was assigned to you, also, elaborate on how you managed to understand this role and provide the necessary evidence, e.g., javadoc, diagrams, etc.>

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem.

Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

# 3
# Problems

List of issues found by applying the checklist: <report the classes/-code fragments that do not fulfill some points in the check list. Explain which point is not fulfilled and why>.

Any other problem you have highlighted: <list here all the parts of code that you think create or may create a bug and explain why>.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel

magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

# A
# Appendix

## A.1 Hours of work

The writing of this document took the following amount of time:

*Paolo Antonini* 38 hours.

*Andrea Corneo* 30 hours.

# B
# Reference - Java checklist

## B.1 Naming Conventions

1. All class names, interface names, method names, class variables, method variables, and constants used should have meaningful names and do what the name suggests. 2. If one-character variables are used, they are used only for temporary âĂIJthrowawayâĂİ variables, such as those used in for loops. 3. Class names are nouns, in mixed case, with the first letter of each word in capitalized. Examples: class Raster; class ImageSprite; 4. Interface names should be capitalized like classes. 5. Method names should be verbs, with the first letter of each addition word capitalized. Examples: getBackground(); computeTemperature(). 6. Class variables, also called attributes, are mixed case, but might begin with an underscore (âĂŸ_âĂŹ) followed by a lowercase first letter. All the remaining words in the variable name have their first letter capitalized. Examples: _windowHeight, timeSeriesData. 7. Constants are declared using all uppercase with words separated by an underscore. Examples: MIN_WIDTH; MAX_HEIGHT;

## B.2 Indention

8. Three or four spaces are used for indentation and done so consistently 9. No tabs are used to indent

## B.3 Braces

10. Consistent bracing style is used, either the preferred âĂIJAllmanâĂİ style (first brace goes underneath the opening block) or the âĂIJKernighan and RitchieâĂİ style (first brace is on the same line of the instruction that opens the new block). 11.All if, while, do-while, try-catch, and for statements that have only one statement to execute are surrounded by curly braces. Example: Avoid this: if ( condition ) doThis(); Instead do this: if ( condition )  doThis();

## *B.4 File Organization*

12.Blank lines and optional comments are used to separate sections (beginning comments, package/import statements, class/interface declarations which include class variable/attributes declarations, constructors, and methods). 13. Where practical, line length does not exceed 80 characters. 14.When line length must exceed 80 characters, it does NOT exceed 120 characters. Wrapping Lines 15. Line break occurs after a comma or an operator. 16. Higher-level breaks are used. 17. A new statement is aligned with the beginning of the expression at the same level as the previous line. Comments 18.Comments are used to adequately explain what the class, interface, methods, and blocks of code are doing. 19. Commented out code contains a reason for being commented out and a date it can be removed from the source file if determined it is no longer needed. Java Source Files 20. Each Java source file contains a single public class or interface. 21. The public class is the first class or interface in the file. 22. Check that the external program interfaces are implemented consistently with what is described in the javadoc. 23. Check that the javadoc is complete (i.e., it covers all classes and files part of the set of classes assigned to you). Package and Import Statements 24.If any package statements are needed, they should be the first non- comment statements. Import statements follow. Class and Interface Declarations 25. The class or interface declarations shall be in the following order: A. class/interface documentation comment B. class or interface statement C. class/interface implementation comment, if necessary D. class (static) variables a. first public class variables b. next protected class variables c. next package level (no access modifier) d. last private class variables E. instance variables a. first public instance variables e. next protected instance variables f. next package level (no access modifier) g. last private instance variables F. constructors G. methods 26.Methods are grouped by functionality rather than by scope or accessibility. 27.Check that the code is free of duplicates, long methods, big classes, breaking encapsulation, as well as if coupling and cohesion are adequate. Initialization and Declarations 28. Check that variables and class members are of the correct type. Check that they have the right visibility (public/private/protected) 29. Check that variables are declared in the proper scope 30. Check that constructors are called when a new object is desired 31. Check that all object references are initialized before use 32. Variables are initialized where they are declared, unless dependent upon a computation 33.Declarations appear at the beginning of blocks (A block is any code surrounded by curly braces âĂIJâĂIJ and âĂIJâĂİ ). The exception is a variable can be declared in a âĂŸforâĂŹ loop. Method Calls 34. Check that parameters are presented in the correct order 35. Check that the correct method is being called, or should it be a different method with a similar name 36. Check that method returned values are used properly Arrays 37. Check that there are

no off-by-one errors in array indexing (that is, all required array elements are correctly accessed through the index) 38.Check that all array (or other collection) indexes have been prevented from going out-of-bounds 39. Check that constructors are called when a new array item is desired Object Comparison 40. Check that all objects (including Strings) are compared with "equals" and not with "==" Output Format 41. Check that displayed output is free of spelling and grammatical errors 42. Check that error messages are comprehensive and provide guidance as to how to correct the problem 43. Check that the output is formatted correctly in terms of line stepping and spacing Computation, Comparisons and Assignments 44.Check that the implementation avoids âĂIJbrutish programming: (see http://users.csc.calpoly.edu/ jdalbey/SWE/CodeSmells/bonehead.html )

45.Check order of computation/evaluation, operator precedence and parenthesizing 46. Check the liberal use of parenthesis is used to avoid operator precedence problems. 47. Check that all denominators of a division are prevented from being zero 48. Check that integer arithmetic, especially division, are used appropriately to avoid causing unexpected truncation/rounding 49. Check that the comparison and Boolean operators are correct 50.Check throw-catch expressions, and check that the error condition is actually legitimate 51. Check that the code is free of any implicit type conversions Exceptions 52. Check that the relevant exceptions are caught 53. Check that the appropriate action are taken for each catch block Flow of Control 54.In a switch statement, check that all cases are addressed by break or return 55. Check that all switch statements have a default branch 56.Check that all loops are correctly formed, with the appropriate initialization, increment and termination expressions Files 57. Check that all files are properly declared and opened 58. Check that all files are closed properly, even in the case of an error 59. Check that EOF conditions are detected and handled correctly 60. Check that all file exceptions are caught and dealt with accordingly

# C
# *Assigned code*

```
   /* Evaluates a client's conformance to a security policies
    * at the client authentication layer.
    *
    * returns true if conformant ; else returns false
    */
    private boolean evaluate_client_conformance_ascontext(
                        SecurityContext ctx,
                        EjbIORConfigurationDescriptor iordesc,
                        String realmName)
    {

        boolean client_authenticated = false;

        // get requirements and supports at the client authentication layer
        AS_ContextSec ascontext = null;
        try {
            ascontext = this.getCtc().createASContextSec(iordesc, realmName);
        } catch (Exception e) {
            _logger.log(Level.SEVERE, "iiop.createcontextsec_exception",e);

            return false;
        }


        /**********************************************************************
         * Conformance Matrix:
         *
         * |------------|--------------------|--------------------|------------|
         * | ClientAuth | targetrequires.ETIC | targetSupports.ETIC | Conformant |
         * |------------|--------------------|--------------------|------------|
         * |     Yes    |          0         |      1
|    Yes     |
         * |     Yes    |          0         |      0
|    No      |
         * |     Yes    |          1         |      X
|    Yes     |
         * |     No     |          0         |      X
|    Yes     |
         * |     No     |          1         |      X
|    No      |
         * |------------|--------------------|--------------------|------------|
         *
         * Abbreviations: ETIC - EstablishTrusInClient
```

```
         *
         ************************************************************************/

        if  ( (ctx != null) && (ctx.authcls != null) &&  (ctx.subject != null))
            client_authenticated = true;
        else
            client_authenticated = false;

        if (client_authenticated) {
            if ( ! ( isSet(ascontext.target_requires, EstablishTrustInClient.value)
                     || isSet(ascontext.target_supports, EstablishTrustInClient.value))){
            return false; // non conforming client
            }
            // match the target_name from client with the target_name in policy

            byte [] client_tgtname = getTargetName(ctx.subject);

            if (ascontext.target_name.length != client_tgtname.length){
                return false; // mechanism did not match.
            }
            for (int i=0; i < ascontext.target_name.length ; i ++)
                if (ascontext.target_name[i] != client_tgtname[i]){
                    return false; // mechanism did not match
                }
        } else {
            if ( isSet(ascontext.target_requires, EstablishTrustInClient.value)){
                return false;  // no mechanism match.
            }
        }
        return true;
    }
```

```
       /* Evaluates a client's conformance to a security policy
    * at the sas context layer. The security policy
    * is derived from the EjbIORConfigurationDescriptor.
    *
    * returns true if conformant ; else returns false
    */
    private boolean evaluate_client_conformance_sascontext(
                        SecurityContext ctx,
                        EjbIORConfigurationDescriptor iordesc)
    {

        boolean caller_propagated = false;

        // get requirements and supports at the sas context layer
        SAS_ContextSec sascontext = null;
        try {
            sascontext = this.getCtc().createSASContextSec(iordesc);
        } catch (Exception e) {
            _logger.log(Level.SEVERE,"iiop.createcontextsec_exception",e);
            return false;
        }


        if  ( (ctx != null) && (ctx.identcls != null) &&  (ctx.subject != null))
            caller_propagated = true;
```

```
        else
            caller_propagated = false;

        if (caller_propagated) {
            if ( ! isSet(sascontext.target_supports, IdentityAssertion.value))
                return false; // target does not support IdentityAssertion

            /* There is no need further checking here since SecServerRequestInterceptor
             * code filters out the following:
             * a. IdentityAssertions of types other than those required by level 0
             *    (for e.g. IdentityExtension)
             * b. unsupported identity types.
             *
             * The checks are done in SecServerRequestInterceptor rather than here
             * to minimize code changes.
             */
            return true;
        }
        return true; //  either caller was not propagated or mechanism matched.
    }
```

```
        /**
     * Evaluates a client's conformance to the security policies configured
     * on the target.
     * Returns true if conformant to the security policies
     * otherwise return false.
     *
     * Conformance checking is done as follows:
     * First, the object_id is mapped to the set of EjbIORConfigurationDescriptor.
     * Each EjbIORConfigurationDescriptor corresponds to a single CompoundSecMechanism
     * of the CSIv2 spec. A client is considered to be conformant if a
CompoundSecMechanism
     * consistent with the client's actions is found i.e. transport_mech,
as_context_mech
     * and sas_context_mech must all be consistent.
     *
     */
    private boolean evaluate_client_conformance(SecurityContext   ctx,
                                                byte[]            object_id,
                                                boolean           ssl_used,
                                                X509Certificate[] certchain)
    {
        // Obtain the IOR configuration descriptors for the Ejb using
        // the object_id within the SecurityContext field.

        // if object_id is null then nothing to evaluate. This is a sanity
        // check - for the object_id should never be null.

        if (object_id == null)
            return true;

        if (protocolMgr == null)
            protocolMgr = orbHelper.getProtocolManager();

        // Check to make sure protocolMgr is not null.
        // This could happen during server initialization or if this call
        // is on a callback object in the client VM.
```

```
        if (protocolMgr == null)
            return true;

        EjbDescriptor ejbDesc = protocolMgr.getEjbDescriptor(object_id);

        Set iorDescSet = null;
        if (ejbDesc != null) {
        iorDescSet = ejbDesc.getIORConfigurationDescriptors();
    }
    else {
        // Probably a non-EJB CORBA object.
        // Create a temporary EjbIORConfigurationDescriptor.
        iorDescSet = getCorbaIORDescSet();
    }

    if(_logger.isLoggable(Level.FINE)) {
        _logger.log(Level.FINE,
            "SecurityMechanismSelector.evaluate_client_conformance: iorDescSet: " + iorDescSet);
    }

        /* if there are no IORConfigurationDescriptors configured, then
         * no security policy is configured. So consider the client
         * to be conformant.
         */
        if (iorDescSet.isEmpty())
            return true;

        // go through each EjbIORConfigurationDescriptor trying to find
        // a find a CompoundSecMechanism that matches client's actions.
        boolean checkSkipped = false;
        for (Iterator itr = iorDescSet.iterator(); itr.hasNext();) {
            EjbIORConfigurationDescriptor iorDesc =
                (EjbIORConfigurationDescriptor) itr.next();
            if(skip_client_conformance(iorDesc)){
        if(_logger.isLoggable(Level.FINE)) {
            _logger.log(Level.FINE,
                "SecurityMechanismSelector.evaluate_client_conformance: skip_client_conformance");
        }
                checkSkipped = true;
                continue;
            }
            if (! evaluate_client_conformance_ssl(iorDesc, ssl_used, certchain)){
        if(_logger.isLoggable(Level.FINE)) {
            _logger.log(Level.FINE,
                "SecurityMechanismSelector.evaluate_client_conformance: evaluate_client_conformance_ssl");
        }
                checkSkipped = false;
                continue;
            }
            String realmName = "default";
            if(ejbDesc != null && ejbDesc.getApplication() != null) {
                realmName = ejbDesc.getApplication().getRealm();
            }
            if(realmName == null) {
                realmName = iorDesc.getRealmName();
            }
            if (realmName == null) {
```

```
                realmName = "default";
            }
            if ( ! evaluate_client_conformance_ascontext(ctx, iorDesc ,realmName)){
        if(_logger.isLoggable(Level.FINE)) {
            _logger.log(Level.FINE,
                "SecurityMechanismSelector.evaluate_client_conformance: evaluate_client_conformance_ascontext");
        }
                checkSkipped = false;
                continue;
            }
            if  ( ! evaluate_client_conformance_sascontext(ctx, iorDesc)){
        if(_logger.isLoggable(Level.FINE)) {
            _logger.log(Level.FINE,
                "SecurityMechanismSelector.evaluate_client_conformance: evaluate_client_conformance_sascontext");
        }
               checkSkipped = false;
               continue;
            }
            return true;  // security policy matched.
        }
        if(checkSkipped)
            return true;
        return false; // No matching security policy found
    }
```