OSEHRA

Open Source Electronic Health Record Agent

OSEHRA
Certification Standards

Delivered in Fulfillment of SLIN 0002AF

Contract Number: VA116-13-C-0008

Version 2.0
December 3, 2013

Revision History

Date	Version	Description	Author
01/14/13	Draft	Initial Draft Submission for discussion	Prior
01/25/13	Draft	Second Draft	Prior
02/08/13	201301	Reformat	Prior
02/09/13	201302	Incorporate comments	Prior
02/12/13	201303	Final modifications	Hewitt, Avila
07/26/13	201307	Updated Procedures	Turner, Ibanez
08/29/13		Incorporate comments	Jensen
10/15/13	Final	Reformat	Badt
11/25/13	Draft	Revise to distinguish certification and submission processes	Henderson, Hewitt, Turner, Edwards
12/3/2013	V2.0	Revision 2	Henderson, Hewitt, Turner, Edwards

Table of Contents

1 Introduction

OSEHRA Open Source Software Quality Certification ("OSEHRA Certification") is a general code evaluation framework designed to support the analysis of any size of code submission, ranging from a bug fix/small code modification to addition of new functionality or the certification of an entire health software system. The certification recognizes four levels of software quality based upon eight independent categories of certification criteria. The outcome of certification scoring will be either a certification failure, or OSEHRA Certification at one of the four levels.

The OSEHRA Certification process is designed to evolve as new code is submitted by the open source community and new interoperability requirements and testing capabilities are provided. OSEHRA is prepared to work with all community members to extend and adapt the certification process to operate on new and revised codebases.

2 OSEHRA Mission Statement

"Build and support an open source community of users, developers, service providers, and researchers engaged in advancing electronic health record software and related health information technology." This mission includes the creation of a vendor-neutral community for the creation, evolution, promotion and support of an open source Electronic Health Record. The OSEHRA community will operate with the transparency and agility that characterize open source software initiatives. This entails not only the development of a community of software experts, clinicians, and implementers, but also a robust ecosystem of complementary products, capabilities and services."

3 Open Source Principles

OSEHRA has adopted the following governing principles as an Open Source entity:

1. Innovation comes from the outside. It must be channeled inside.
2. Software is knowledge transformed into code. It needs an engaged community to maintain it and operate it.
3. Software is never a finished product; it evolves continuously, and that evolution requires an involved community.
4. Attract interested people with shared goals; earn their trust.
5. Transparency; remove any obstacles to free flow of information.
6. Meritocratic governance driven by: Autonomy, Mastery, and Purpose
7. Release Early, Release Often
8. Avoid Private Discussions

9. Establish Credibility; build relationships with Open Source communities.
10. Welcome the unexpected. Listen carefully to the community.

These principles will guide the decisions that will be made when considering different alternatives for the design and implementation of a software quality certification process.

4 Definition of OSEHRA Certification

OSEHRA Certification is initially the attestation and ultimately the verification that an executable artifact submitted for certification by OSEHRA meets the following criteria:

- **Safe -** Individual code units do not cause errors in other components of the system and the code is robust through all code paths and conditions.
- **Compliant -** Code meets agreed-upon interface specifications.
- **Functional -** Code has a defined set of requirements that are met when the code executes.

The term "OSEHRA Certification" is explicitly used by OSEHRA to establish a clear distinction between the OSEHRA Certification process and definitions of certification used by other organizations. In particular, the Department of Veterans Affairs (VA) has developed an extensive set of processes and procedures that are currently used to certify software for internal use. VA maintains information resources on its secure intranet that facilitate testing and certification, and track the progress of the certification process. Much of this certification infrastructure and information, particularly that dealing with internal network addresses and with proprietary code, is not currently shared with the open source development community external to VA and cannot be part of the OSEHRA Certification process.

4.1 Purpose of OSEHRA Certification

The purpose of the OSEHRA Certification process is to raise trust and confidence in the code introduced into its codebase. The OSEHRA Certification process provides the mechanisms through which developers are encouraged to contribute bug fixes, improvements, and new functionality to the codebase, while at the same time ensuring that those submissions are safe, compliant and functional prior to offering them to the OSEHRA community. OSEHRA Certification is thus aligned with its mission to facilitate the development and maintenance of open source health software, and to follow best practices of open source communities.

OSEHRA certification is not intended to replace the user testing, verification and validation, and other pre-production and production testing processes carried out by the VA or any other community member. Rather, OSEHRA is providing a framework of tools, procedures, and educational materials to enable and actively engage a vibrant open source community. The goal is to create a robust, community-based software quality certification process with OSEHRA acting as a unifying body. The OSEHRA Certification process will promote adherence to

community-driven practices and help community members develop code and applications that will be compatible with and easily integrated into the codebase.

5 Software Quality Certification Tools

The OSEHRA Certification environment provides tools for analysis and validation of various types of submissions to the open source codebase. At the core is the concept of automated testing and peer review to verify the safety, compliance and functionality of the code both prior to and after new code submissions. While testing of any legacy codebase is difficult, OSEHRA has started the process by acquiring or generating a number of automated tests. These tests measure compliance (using tests based on established standards and conventions) as well as safety and functionality (using a combination of thread tests, unit tests, and regression tests). This testing suite is just a beginning. OSEHRA is actively seeking additional tests and requiring that any new or revised code submissions be submitted with tests that are evaluated along with the code submission. The degree of test coverage is measured using a code coverage facility that evaluates how much of the codebase is visited and exercised during the testing activities.

Certification and attestations will come in three distinct varieties, each with different emphasis, tools and procedures:

> **Gerrit-based[1] Reviews.** The most frequent method will be the certification of submissions as part of a code review process. These will be Gerrit-based reviews and attestations of incremental changes to the codebase (such as bug fixes made during the review process), prior to merging the changes into the code repository. Review and attestation at this level will be continuous, but local. Automated tests will be run against the codebase with the changes present, but the reviewer will not be expected to go beyond the context of the locally-submitted changes and bug fixes during any manual attestations. The goal here is to keep bad code, i.e., code that has a significant performance penalty, is untested, or does not meet community compliance standards, from being merged into the system without requiring an exhaustive review.
>
> **OSEHRA Technical Journal.** At the next level will be reviews and attestations of larger code submissions, or the submission of new functionality or modules, via the OSEHRA Technical Journal found at http://code.osehra.org/journal/. Such submissions will require a higher degree of testing and review than simple bug fixes.
>
> **Formal Releases.** Finally, there will be periodic formal releases of new codebase versions. Each new release will be initiated by a Gerrit push containing a single modification to increase the software version number, and a JIRA[2] issue initiating the release of the software. Review for formal releases will parallel the normal Gerrit code review process, but will not be local in scope and will be expected to more thoroughly evaluate the testing and attestations over the complete codebase, paying particular

[1] Gerrit is a web based code review system, facilitating online code reviews for projects using the Git version control system.

[2] JIRA is a proprietary issue-tracking product developed by Atlassian, used for bug tracking, issue tracking and project management.

attention to the complete body of changes since the previously accepted formal release. Following a successful release review, the attestation record SHA[3] key will provide an additional level of confidence to users downloading the release.

OSEHRA certification is an organic process incorporating best practices of open source communities. OSEHRA is focused on quality control processes that are closer to developers. This will help raise adherence to the goals of safety, compliance, and functionality, while simultaneously fostering their programming skills by providing constant feedback through the community-wide platforms such as Git, Gerrit, JIRA, and the quality control dashboards. Following open source best practices, certification efforts will focus on areas where OSEHRA can provide value to software development teams and communities, and will avoid redundancy as much as possible. The remainder of this section describes the toolset that is available to perform, evaluate, and capture the various software quality certification activities.

5.1 Code Testing Framework (CMake/CTest)

OSEHRA uses an open source build and testing suite to generate automated tests from test templates and to execute those tests against the current code base. Instructions for obtaining this framework and using it to configure and test the OSEHRA codebase can be found at the following links:

- https://github.com/OSEHRA/VistA/blob/master/Documentation/ObtainingandInstallAuxPrograms.rst
- https://github.com/OSEHRA/VistA/blob/master/Documentation/SetupTestingEnvironment.rst
- https://github.com/OSEHRA/VistA/blob/master/Documentation/RunningandUploadingTests.rst

Using this automated system allows for scheduled, continuous and on demand testing to be executed and results can be reported to a central dashboard. The system is flexible and allows for the execution of any command line driven program as a test. OSEHRA requires that new code contain unit, functional and regression tests along with the source code. Webinar style tutorials will be available on the OSEHRA site to assist in using and extending the testing framework.

5.2 Dashboard

As noted in the previous section, automated testing is integral to OSEHRA Certification. It is essential that results be publicly reviewable, and presented so that viewers can correlate changes to the test results as the system changes. To aid in this, OSEHRA provides a testing dashboard in which the status of the current codebase is displayed. A screenshot from the OSEHRA Software Quality Dashboard ("OSEHRA Dashboard") is shown in Figure 5-1 below.

Gerrit reviews for incremental and formal releases will be reported to the OSEHRA Dashboard and indexed by hash code.

[3] Secure Hash Algorithm, a means of ensuring the integrity of a file.

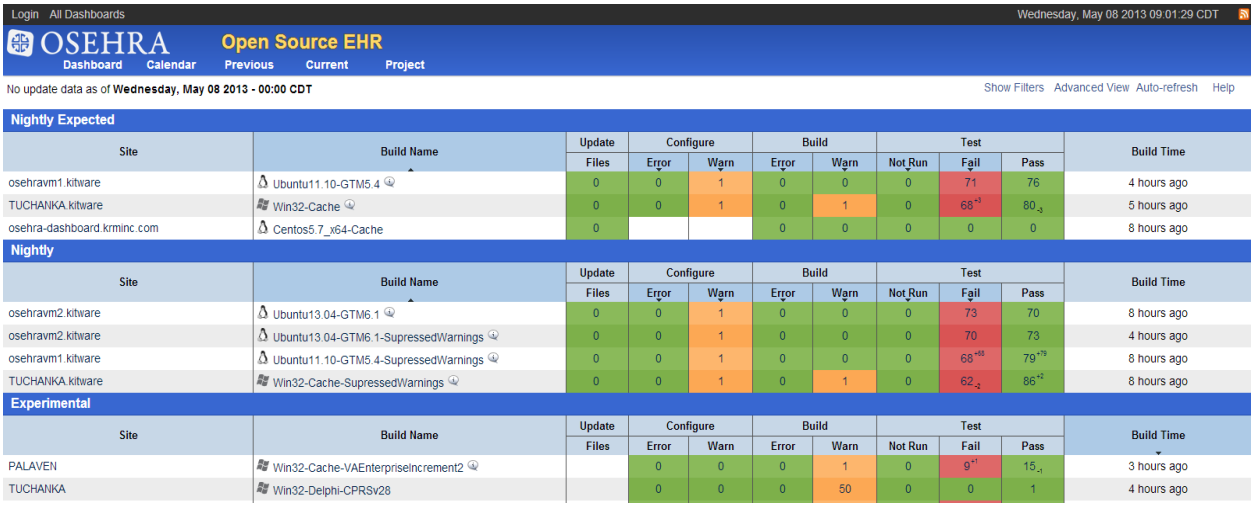

Figure 5-1 - OSEHRA Dashboard

5.3 Issue Tracker

The OSEHRA Certification environment tracks issues and feature requests using an issue tracker based on JIRA, located at http://issues.osehra.org. Both users and developers have access to the issue tracker and may report, confirm, and resolve issues. Separate issue tracker projects are maintained for codebases contributed to or managed by the OSEHRA community. The full list of currently available open source projects managed by the issue tracker can be found at http://issues.osehra.org/secure/BrowseProjects.jspa#all.

5.4 Repositories

OSEHRA uses Git, an open source distributed version control system, to maintain and control the inclusion of code into the various open source projects OSEHRA manages. These open source project repositories contain the public codebase and any tools and utilities that are useful to the community. All repositories are available at http://code.osehra.org/gitweb, but mirrors are maintained on Github at https://github.com/OSEHRA and on Gitorious at https://gitorious.org/osehra. Code from these sites can be freely downloaded and used subject to any licensing placed by the contributors of the code. OSEHRA managed code is distributed under the Apache 2.0 License, which provides liberal access to anything generated by OSEHRA.

The mechanisms by which the repositories are used in the management of the OSEHRA distributions is described in greater detail in the companion document, OSEHRA Certification Process. For an example of the structure, see the VistA Repositories in Appendix A. There are a number of repositories currently maintained by OSEHRA as a courtesy to the open source developers comprising the OSEHRA community. Refer to https://github.com/OSEHRA/VistA, https://github.com/OSEHRA-Sandbox, and http://gitorious.org/osehra for OSEHRA maintained codebases. Users are encouraged to check these locations frequently as the repositories are subject to frequent additions and changes.

5.5 Peer Review Systems

Peer review connects the code repository with OSEHRA Certification. We identify three different mechanisms for peer review, depending on the magnitude and goal of the requested change. These mechanisms provide expedited review when the change is small and localized, typically to address minimal changes or to fix a bug in the codebase. More thorough review is required to add a more substantial capability such as a new module or package, or to attest to the software quality of a formal release. Each of these is discussed below.

5.5.1 Gerrit

For small, evolutionary changes or bug fixes, OSEHRA has implemented a code review system similar to that in use for the NLM-sponsored Insight Segmentation and Registration Toolkit (ITK). This system is based on Gerrit, a code review system developed by Google for the Android project and integrated with the Git environment. Peer review requires that one or more trusted individuals assert that the code meets publically available criteria developed by the community. At a minimum, this includes asserting that the code is safe, compliant, and functional. The Gerrit review tool can be found at http://review.code.osehra.org/ and a representative screen shot showing submissions under review is shown in Figure 5-2 below.

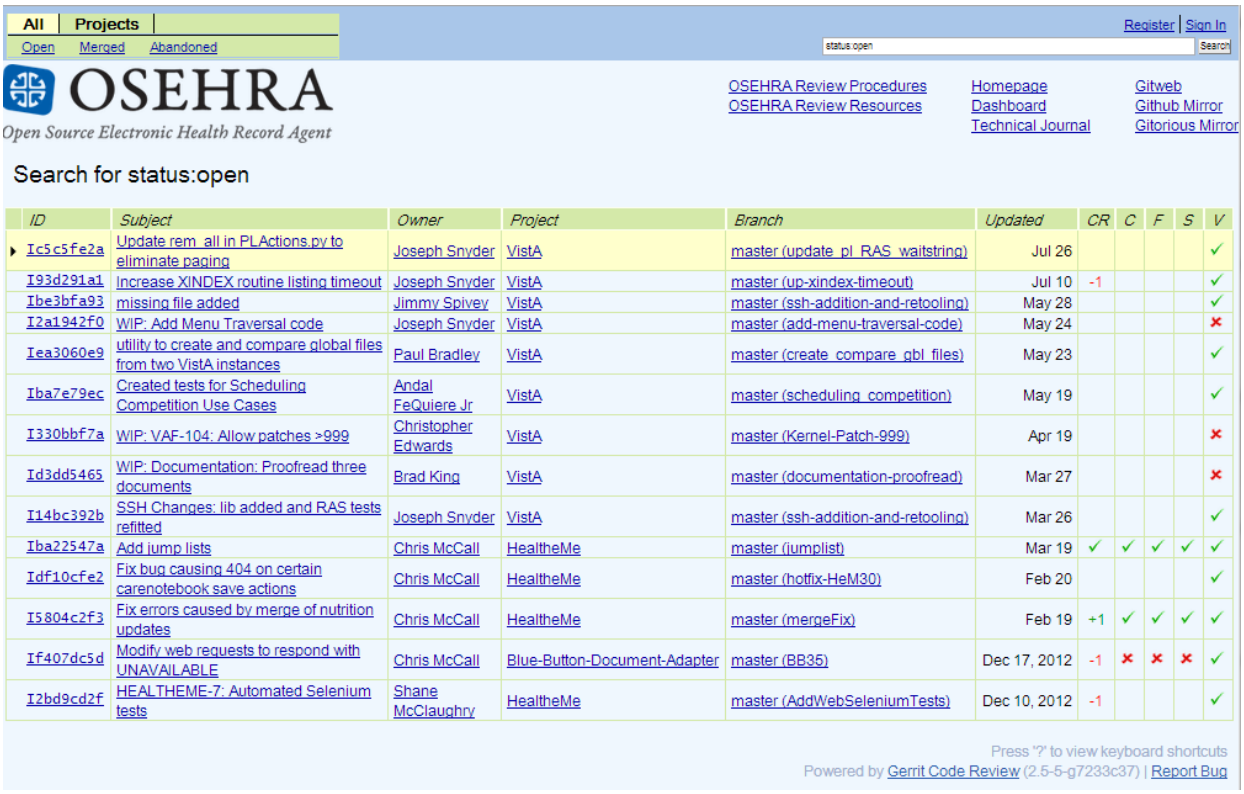

Figure 5-2 - A screenshot of the Gerrit code review tool showing open submissions ready for review

5.5.2 OSEHRA Technical Journal

For larger changes, a review system based on OSEHRA Technical Journal is required. The Journal entries include the code/schema change to be evaluated and a set of materials such as documentation, functional definition, test fixtures, test data and test results. A representative screen shot showing the OSEHRA Technical Journal homepage is shown in Figure 5-3 below.

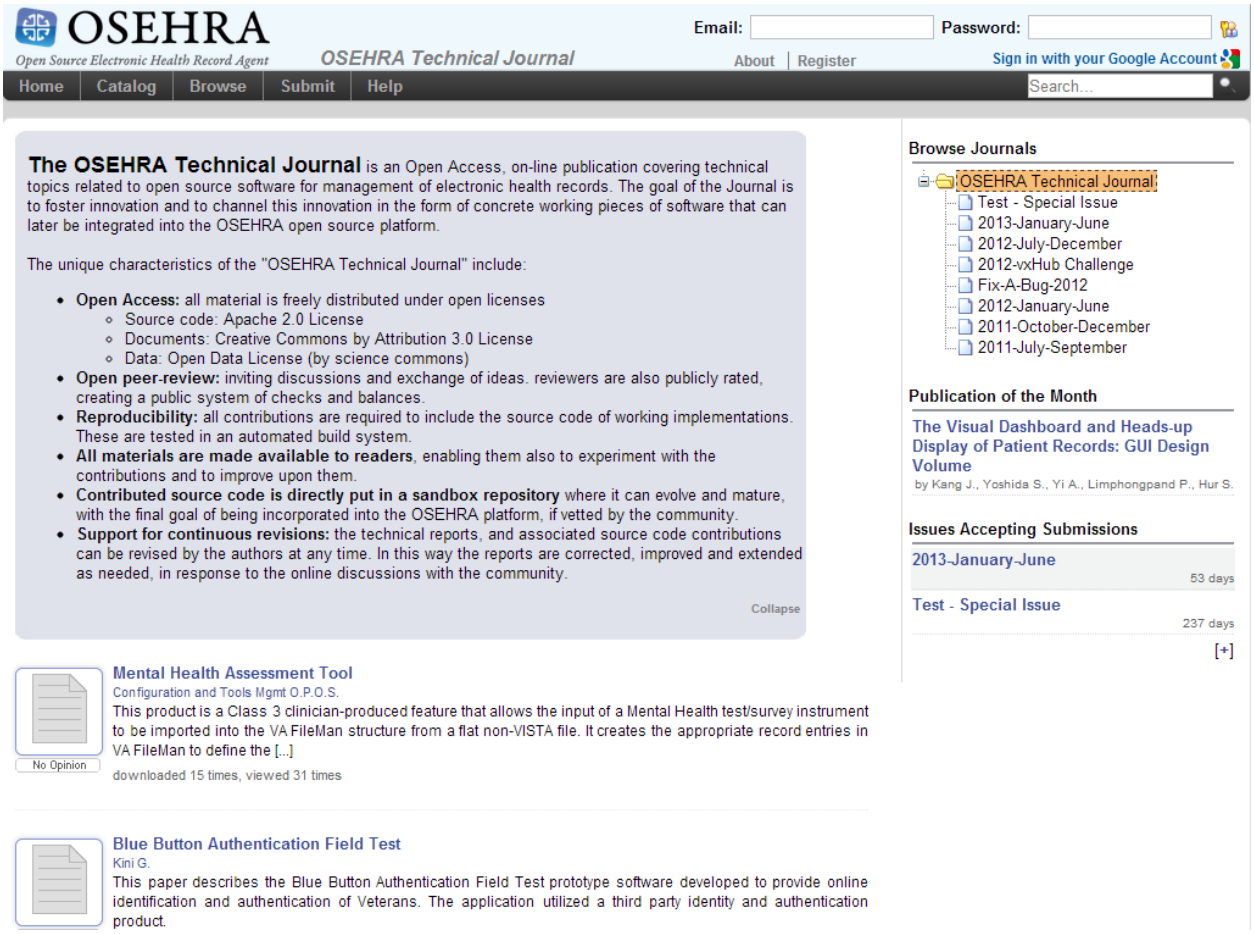

Figure 5-3 - OSEHRA Technical Journal entry

5.5.3 OSEHRA Release Review

Formal releases will use an attestation process initiated by a JIRA ticket and in conjunction with a Gerrit review. On completion, this will result in an explicit tagging of the repository to provide a user verifiable mark. The release process is initiated by the submission of a release task into the appropriate JIRA project. Following a process of testing and remediation and after testers determine that the code is sufficiently safe, compliant, and functional, the attestor will submit a single push consisting of a change to the upper level OSEHRA codebase ATTESTATION file through Gerrit and into the repository. The change should consist of the addition of a single line as the topmost attestation. The line will contain the SHA1 key for the repository to be released, the date, and the name of the attestor. Figure 5-4 shows an example ATTESTATION file after the addition of the first release attestation. All artifacts of the attestation process such as

remediation tickets, code exemptions, etc. are attached to the JIRA ticket prior to attestation. For the short description, the message should replicate the JIRA title “Initiation of release #(SHA1)”. The long description can contain release notes along with the reference to the correct JIRA ticket.

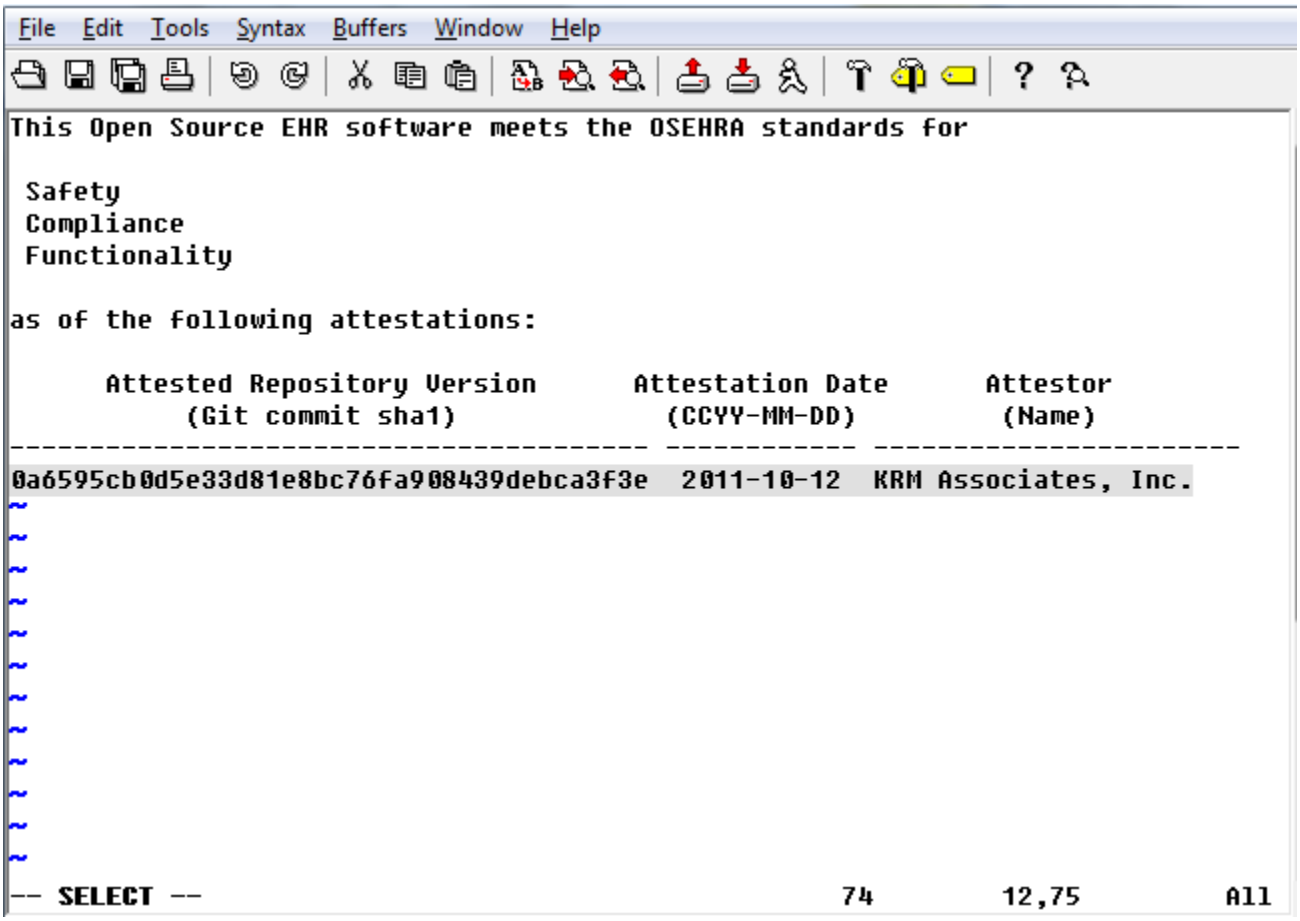

Figure 5-4 - Structure of the ATTESTATION file. The highlighted line shows the structure of the new addition containing the SHA 1 key, date and attester name.

6 OSEHRA Certification Categories

OSEHRA Certification criteria comprise eight categories. Each category is evaluated to produce a specific score. Scores are either pass/fail or graduated within a range of values reflecting the degree to which a submission complies with the criterion. Depending on the score across categories, a submission can be certified at one of four levels. As shown in the table below, Level 1 is a basic certification indicating minimal compliance with the open source principles of OSEHRA, while Level 4, the highest, indicates complete compliance. Strategically, Level 1 is an entry level and will be used exclusively for legacy code.

Minimum criteria necessary to achieve each OSEHRA Certification level are summarized in Table 1 below.

- The first category is designed as a basic check to ensure that the namespace and numberspace allocations for the codebase adhere to community standards developed by the DBAC (DataBase Administration Committee). This is mainly intended for VistA/M code submissions, although namespacing and collision avoidance with existing code modules can also be an issue for other development languages.
- The dependency / SAC category explicitly evaluates dependencies as well as standards and conventions. A third category ensures that OSEHRA-required licensing (Apache 2) is applied to the codebase.
- The documentation category measures the degree to which documentation is provided with the code.
- The code review category allows for the evaluation of the code, preferably by domain experts.
- The test installation category performs an installation of the codebase and catalogs any issues identified.
- Regression testing determines the amount of successfully executing automated regression and unit tests provided with the codebase. Code coverage using an OSEHRA approved code coverage tool will be evaluated to determine progress toward the highest certification levels.
- Finally, functional testing evaluates the level at which specific functional capabilities work as intended and without issues.

	Name/ Number Space	Depen-dency / SAC	Open Source License	Documen-tation	Code Review	Test Installa-tion	Regression Testing	Functional Testing
Level 1	Pass	Pass	Apache 2	None	Large # Non-critical Issues	Large # Non-critical Issues	Existing Tests Pass	Large # Non-critical Issues
Level 2	Pass	Pass	Apache 2	Basic	Small # Non-critical Issues	Small # Non-critical Issues	Existing + Some R. Tests	Small # Non-critical Issues
Level 3	Pass	Pass	Apache 2	Substantial	No Issues	No Issues	Existing + >= 50% Coverage	No Issues
Level 4	Pass	Pass	Apache 2	All Required	No Issues	No Issues	Existing + >= 90% Coverage	No Issues

Table 1 - OSEHRA software quality certification levels and minimum category expectations

7 Certification Standards

OSEHRA has created certification standards by which community members inspect and certify code for compliance with standards of good software engineering practices. The intent of these standards is to accommodate specific VA interoperability needs while also serving the needs of the open source community.

7.1 Detailed Scoring Explanation

The following eight sections provide a detailed explanation of the scoring methods that will be applied for each certification category.

7.1.1 Name / Number Space

It is critical that name space and number space designations be maintained and enforced, particularly in M code. Overlapping name spaces and number spaces can potentially result in unreliable code and unforeseen consequences. Thus, as a component of the technical evaluation, name spaces and number spaces of code being evaluated will be compared to known allocations to ensure appropriate usage. Name and number space overlap evaluation will be scored on a pass/fail basis. All issues associated with name and number space utilization must be resolved in order to achieve any level of OSEHRA certification.

7.1.2 Dependency / SAC

All code submissions will be tested to verify compliance with applicable Standards and Conventions (SAC). This review will then be compared with submitted documentation. Any substantial discrepancy will result in failure of certification, as failure to properly account for the potential impacts of a submission indicates that the submission likely has not been sufficiently tested for integration into the core solution. It is understood that there are occasional valid reasons for violating SAC, but these should be identified and justified in the documentation of the submission. As additional languages and coding styles are brought into the community, the community will need to develop or adapt additional style guides and automated routines to enforce the chosen style. Dependency and SAC conformance will be evaluated on a pass/fail basis. All issues associated with SAC discrepancies must be resolved in order to achieve any level of OSEHRA certification.

7.1.3 Open Source License

OSEHRA requires that all code submissions be made under the Apache 2 license.

7.1.4 Documentation

Documentation is intended to serve the needs of the OSEHRA community by ensuring that submissions fully define the purpose, use, installation, testing, algorithms, and other salient components of the submission. The general criteria by level are as follows:

Level 1 of the documentation category evaluation does not require that documentation be provided with the code. Level 1 certification is restricted to legacy VA code that has generally been running for many years, but for which documentation may not be available.

Level 2 requires that a basic set of documentation be present which must include:

- A description of the intended purpose and requirements of the codebase.
- A description of the installation instructions of the application.
- A description of how to test the code.

Level 3 requires, in addition to the documentation listed in Level 2, that the documentation provide:

- A description of the methods and relevant information needed to understand complex or difficult aspects of the code, such as complicated algorithms, algorithm provenance, or items that require clarification.

Level 4 requires all of the documentation listed in Level 3, plus:

- A description of ongoing or additional development work being performed on the codebase.

Multiple documentation artifacts may be contained in a single document such as an OSEHRA Technical Journal article, or they may be provided separately. In the latter case, the main document can simply be a reference to the rest of the submission. Note that VA defines a substantial number of documentation artifacts as part of their code development process, including the documents in Table 2. These documents are not required, but if present they can be used to address the documentation requirement.

Artifact Name
Business Requirements Document (BRD)
Software Requirements Specification (RSD)
Initial Operating Capability (IOC) Testing
Data Dictionary Modification Request (for Database Administrator)
Data Dictionary Modification Approval (from Database Administrator)
Integration Control Registrations (ICR)s
Clearance on potential impact on FDA Regulated interfaces or devices (from Package maintainers)
Change submission to Architecture Review Group
Application Self-Scoring Evaluation Support System (ASSESS) (Capacity Planning form)
Patch Installation Instructions (Installation Guide)
List of patch dependencies (Patch Release Check)

Artifact Name
National Patch Module Patch Template
Updated Documentation describing new Functionalities
HL7 Messaging Coordinator Approval for related changes
OED Testing Service Report

Table 2 - List of VA documentation that may be submitted and evaluated for OSEHRA certification

7.1.5 Code Review

Code review is one of the most effective methods for identifying quality issues in a codebase. Unfortunately, they are also one of the most labor-intensive components of software quality efforts, and require the participation of developers with specialized skills to deeply review the code and identify issues. As a result, OSEHRA must prioritize the application of code review resources, and spot check code under review for larger submissions.

OSEHRA uses a combination of the Gerrit code review tool and the OSEHRA Technical Journal for reviewing submitted code. These tools facilitate communications and allow reviewer decisions and comments to be fully archived for future investigations. The specifics of the review process differ depending on the tool used; however, both tools allow for community review of the submission, but require final review and approval by a trusted community member. *OSEHRA will always require that critical issues identified during code review be resolved.* This includes any code construct that could result in an incorrect calculation. The criteria by level are as follows:

> **Level 1** of the code review category allows for the identification of large numbers of *non-critical* issues during the code review.
>
> **Level 2** allows a small number of *non-critical* issues during code review.
>
> **Levels 3 and 4** both require that *any* issues identified during code review be resolved.

7.1.6 Test Installation

OSEHRA will install code submissions, using the provided documentation, on an internally hosted testing instance. This instance will act as the testing platform for OSEHRA to conduct its certification functions. *OSEHRA will always require that critical issues identified during test installation be resolved.*

> **Level 1** of the test installation category allows for the identification of large numbers of non-critical issues during the installation evaluation.
>
> **Level 2** requires that only a small number of non-critical issues are found during test installation.
>
> **Levels 3 and 4** both require that any issues identified during test installation be resolved.

7.1.7 Regression Testing

Regression testing seeks to uncover software errors by retesting the program after changes to the program (*e.g.*, bug fixes or new functionality) have been made. Some automated regression test

capability is currently available to OSEHRA. As the OSEHRA community expands, and developers become accustomed to providing regression tests with code submissions, OSEHRA will have an ever-growing number of automated regression tests to apply to the various codebases. Tests from this regression testing infrastructure which are applicable to the received code submission will be run to characterize and document code behavior. In addition, high quality code (that which achieves the higher levels of OSEHRA certification) should be submitted with an increasingly comprehensive set of unit and regression tests. *OSEHRA will always require that critical issues identified during testing be resolved*. The criteria by level are as follows:

Level 1 requires that the codebase under evaluation not cause any failures in the existing set of OSEHRA automated regression tests.

Level 2 requires that Level 1 be achieved and that additional unit and regression tests specifically test the capabilities of the submitted codebase at some level of coverage.

Level 3 extends upon Level 2 and requires that the supplied tests achieve at least 50% code coverage for the code being contributed.

Level 4 extends upon Level 3 and increases the code coverage requirement to 90%.

7.1.8 Functional Testing

Functional testing determines whether specific application capabilities work within a codebase. Although some of these tests can be automated, significant benefits are obtained when a human observer conducts the tests and is able to identify side effects and issues not anticipated by the test designers. Similarly to code review, manual functional testing is highly labor intensive and can be prohibitively expensive when checking large numbers of application functionalities. *OSEHRA will always require that critical issues identified during testing be resolved*. The criteria by level are as follows:

Level 1 of the functional testing category allows for the identification of large numbers of non-critical issues during the testing process.

Level 2 requires that only a small number of non-critical issues be found during functional testing.

Levels 3 and 4 both require that any issues identified during functional testing be resolved.

7.2 Application of the Standards

Not all submissions to OSEHRA's code repository are capable, required, or expected to meet all categories of certification criteria, and the OSEHRA Certification Process will not be fully utilized in all cases. For example, code reviews for the initial VistA FOIA/Enterprise submission will not be expected to completely examine the code, but manual review of a percentage of source code will be performed to increase confidence in the codebase. Subsequent incremental submissions to the core modules will require full review of source as part of the code intake process. Testing of the current OSEHRA VistA, automated or manual, is limited due to the large codebase involved and due to the time and effort required for generating the required procedures and code; however, testing continues to be submitted as part of the existing refactoring efforts

and via community submissions. Any applicable tests submitted to the testing environment will be run against the core modules. As new or refactored components are added to the codebase they will be required to include additional documentation and testing capabilities. Thus, over time the OSEHRA Certification Process will evolve and become more comprehensive. These newer submissions are expected to be certifiable at higher levels, while the legacy code remains at Level 1 certification.

7.3 Gaps in Testing Capabilities

The purpose of a gap analysis is to identify key issues that must be addressed in order to move the open source development community forward. In this section, we address those needs that will enable the community as a whole to move forward with more capable and robust testing processes, which ultimately result in better software and higher confidence in the safety, functionality and compliancy of the codebases.

7.3.1 Support for Non-Mumps and Mobile Environments

The OSEHRA tools, techniques and capabilities of the system are broader than the M code requirements. In fact, the tools and processes being used were developed for testing large, open source, C++ toolkits, even though the testing infrastructure was originally tailored to the requirements of testing the VistA M platform. As more and more non-Mumps code is contributed to OSEHRA and the community, two particular gaps in our testing capabilities have been identified:

- OSEHRA does not currently support non-Mumps or mobile platforms in a consistent and easily adaptable fashion.

- OSEHRA needs to develop standards and examples for testing these different applications.

8 Appendix A - VistA M-Code Specific Considerations

8.1 Introduction

OSEHRA Certification uses a generic testing framework designed to support multiple code repositories, languages and development models, including traditional desktop systems, web enabled applications and mobile devices. However, each of these variations requires specializations for standards and conventions, test organizations, compilers and other components of the development environment. As each of these specializations is developed, they will be described in an appendix specific to the configuration.

This appendix describes the specializations for the VA Veterans Information Systems and Technology Architecture (VistA). VistA is an M-Code based database and application. Tools, standards and requirements specific to VistA are described in the remainder of this document. The OSEHRA certification process is designed to evolve as new packages are added and new interoperability requirements and testing capabilities are provided.

8.2 Software Quality Certification Tools

8.2.1 VistA Issue Tracker

The following links may be used to review activities specific to VistA code:

- VA FOIA release code, http://issues.osehra.org/browse/VAF
- OSEHRA VistA code branch, http://issues.osehra.org/browse/VISTA
- OSEHRA testing code, http://issues.osehra.org/browse/OAT

8.2.2 VistA Repositories

There are two primary repositories used to store the VistA codebase, patch stream, and testing routines:

- VistA.git: OSEHRA VistA and FOIA VistA Development Tools and Patches
- VistA-M.git: OSEHRA VistA and FOIA VistA M Components (raw .m and .zwr)

The VistA repository holds the OSEHRA and FOIA releases as separate branches, incremental patches, and development tools. The VistA-M repository holds an extraction of the M routines and globals that represent the VistA database. The VistA-M repository is automatically maintained based on the contents of the VistA repository, and user commits are not allowed into VistA-M with the exception of the attestation file during OSEHRA Release Review. However, users desiring to stand-up a new VistA instance are encouraged to clone the VistA-M repository as an easy way to rapidly get the full codebase.

Both of these repositories can be found at http://code.osehra.org, and users can get access either by downloading compressed source files:

- OSEHRA VistA.git master branch:
  - .tar.gz (https://github.com/OSEHRA/VistA/tarball/master)
  - .zip (https://github.com/OSEHRA/VistA/zipball/master)
- OSEHRA VistA-M.git master branch:

- .tar.gz (https://github.com/OSEHRA/VistA-M/tarball/master)
- .zip (https://github.com/OSEHRA/VistA-M/zipball/master)

Or by cloning the repositories using the following Git commands:

- git clone git://code.osehra.org/VistA.git
- git clone git://code.osehra.org/VistA-M.git

Additional details on the repositories can be found in the OSEHRA Certification Process document.

8.3 Branches

Differentiation between the official VA VistA branch and the OSEHRA modifications will be maintained using different code branches in the git repositories. Additional details on the branches can be found in the OSEHRA Certification Process document.

8.3.1 Master Branch

The master branch follows the OSEHRA release and represents fixes, enhancements, and other modifications that have been contributed by the community and that have been certified and subsequently selected for inclusion into OSEHRA VistA.

8.3.2 FOIA Branch

The FOIA branch follows the FOIA release from the Department of Veterans Affairs and the process of updating the repository uses the standard VistA M-Code patching procedures developed at the VA. Patches are obtained from the VA as KIDS builds and are applied incrementally.

OSEHRA is helping the VA to incrementally certify the VA VistA as part of the VA Gold Disk standardization effort. The current status of this effort and a list of those packages that comprise the current certified core can be found at http://www.osehra.org/content/va-enterprise-vista-standard.

8.4 VistA Testing

8.4.1 Dependency / SAC

OSEHRA will ensure that any submissions to the VA codebase accurately follow the VA SAC (Standards and Conventions) Guide with the exceptions noted above. The SAC Guide is a document that outlines programmatic requirements of code submissions, and is maintained by VA with input from OSEHRA and the Open Source community.

Compliance testing relies on the XINDEX facility of the VistA code for both legacy and new modules. Because the standards and conventions for writing M-Code have evolved over the years, there are many areas in the legacy code that do not meet the current requirements of XINDEX. These reported errors are being addressed in one of three ways:

1. The failing code is flagged for revision in order to be brought into compliance.

2. The XINDEX test is flagged for update to allow for accepted changes in the SAC.
3. The code is given a waiver accompanied by an error suppression to remove the error from the reporting tools.

Note that option 3 requires that a representative from the VA or other trusted body provide a signed certification that the specific XINDEX error is unlikely to cause degradation in code quality. These waivers are intended to be temporary and the underlying deficiency will need to be fixed if the code is modified or refactored.

XINDEX will be executed on all M-based submissions to accurately detail compliance with SAC. This review will then be compared with submitted documentation and the OSEHRA generated VistA Cross Reference (http://code.osehra.org/dox/index.html). Any substantial discrepancy will result in failure of certification; failure to properly account for the potential impacts of a submission indicates that submission is not sufficiently tested for integration into the core solution. It is understood that there are occasional valid reasons for violating the SAC, but these should be identified and justified in the documentation of the submission.