



OSEHRA Forum Technical Journal Article

**Version 1.0
April 18, 2014**



©2014 Open Source Electronic Health Record Alliance, Inc.

OSEHRA Forum Technical Journal Article by

Frederick D. S. Marshall, Sam Habel, and Linda Yaw

is licensed under a

Creative Commons Attribution-ShareAlike 4.0 International License.

Revision History

Date	Version	Description	Author
4/10/14	0.1	First draft attempt	Yaw
4/15/14	0.2	Pulled a lot of information out into separate documents	Yaw
4/18/14	1.0	Updated Unit Test section, added code walkthrough section Ready for submission	Haniel, Yaw

Table of Contents

1	Introduction	4
1.1	Submission	4
1.2	Primary & Secondary Development	4
1.3	Multiple Software Streams	4
2	OSEHRA Forum Functionality	4
3	Current User Interface	5
4	Executed Unit Tests	5
5	Code Walkthrough.....	7
5.1	Fileman Files	7
5.2	Routine Breakdown	7
6	Conclusion	8

1 Introduction

1.1 Submission

In December 2013, OSEHRA contracted with VISTA Expertise Network to modify the VA Forum package, especially the Patch Module, to meet the needs of the non-VA VistA community. The goal of Phase One was to achieve an Initial Operating Capability that would support not only code released under the Freedom of Information Act by the Department of Veteran's Affairs but would also support a separate patch stream for code that was modified or created by OSEHRA and the VistA community.

We are pleased to submit the results of Phase One to the OSEHRA certification process.

A brief discussion of the changes made to the original code to support the needs of the community follows.

Items included in this submission are:

- ✓ OSEHRA Forum Technical Journal Article—this paper
- ✓ Patch Module 2.4 Code
- ✓ OSEHRA Primary Developer Checklist
- ✓ Initial Design Document—discusses the work done in Phase One and the reasons for that work
- ✓ OSEHRA Forum Configuration Nomenclature—surveys the scheme and nomenclature to be used in assigning version, patch, and sequence numbers to OSEHRA VistA
- ✓ Forum System Installation and Administration—outlines the process of setting up and administering the modified Forum software.

1.2 Primary & Secondary Development

OSEHRA does some primary development, but much of what the organization does is modifying and extending existing VA code. VA's software hub is a Forum system that only supports primary development. **OSEHRA Forum Project's Phase One upgraded the Patch Module software to be able to support both primary and secondary development.**

1.3 Multiple Software Streams

VA's existing Forum system only supports a single outbound stream of patches, but **OSEHRA Forum now supports two outbound software streams, VA FOIA VistA and OSEHRA VistA, and will support any number of streams later on.**

Further, if Indian Health Service (IHS) wishes to take advantage of OSEHRA Forum in the future rather than creating its own IHS Forum, then OSEHRA Forum also will be able to support more than one inbound software stream, since it will also need to be able to consume IHS's stream of patches for distribution without mixing up the VA and IHS patches with each other.

2 OSEHRA Forum Functionality

The OSEHRA Forum system is a shared VistA-community resource that can make possible for the first time a VistA life cycle that integrates the vertical silos of the current VistA community into a truly unified VistA-development community.

The five components of the VistA life cycle that OSEHRA supports are:

- 1) FOIA VistA Patches
- 2) OSEHRA VistA Patches from VA
- 3) OSEHRA VistA patches From Community
- 4) OSEHRA VistA Replacements For FOIA Vista Patches
- 5) New Submissions To OSEHRA From The Community

The first four components are forms of *secondary development*. Secondary development is when you are modifying or extending someone else's VistA package.

The fifth component is different; it supports *primary development* that originates not from VA Forum but from the OSEHRA community itself, when we are not modifying VA's code but creating or modifying our own. File Manager version 22.1 is an example of primary development, since in this release we took responsibility for an entire application rather than just inserting secondary development into VA's application. Other examples would be the initial or subsequent versions of brand new VistA applications developed by the OSEHRA community.

A subset of that fifth component, primary development by VA developers, is the only form of VistA development that VA Forum currently supports. Merely by setting up our own Forum system open to the entire OSEHRA community, we are expanding our support to include all primary development, by VA or by any other primary developer.

3 Current User Interface

Although Gerrit, Git, Jira, and the OSEHRA Technical Journal have modern web user interfaces, VA Forum uses a traditional terminal-based user interface, as will OSEHRA Forum's initial operating capability. Upgrading Forum to a web UI is nontrivial work, too difficult to achieve in the early phases of the project, because the Patch Module's database and business logic are closely intertwined with its UI, to support a rich user dialog.

The early phases of this project, including Phase One, began that disentangling process, especially by opportunistically replacing hard-wired UI output built into the database to instead use more modern UI tools capable of redirecting output to arrays for silent operation, in preparation for sending those arrays to GUIs or web UIs (WUIs?). Even so, for these early phases the Patch Module will remain solidly terminal-based.

4 Executed Unit Tests

The planned unit tests correspond to the non-database tasks planned for Phase One. See the Initial Design Document for more details about what those tasks were. Here is a list of the unit tests we have developed so far:

- * Make the FOIA VISTA and OSEHRA VISTA patch streams.
- * Make a package in file Package (9.4)
- * Make users in file New Person (200)
- * Add a package to Patch Module file DHCP Patch/Problem Package (11007)
- * Setup a package in the Patch Module

- * Setup a new version of a package
- * Delete all e-mail messages in Q-PATCH basket (used to receive patches)
- * Mail a KIDS build to XXX@Q-PATCH.OSEHRA.ORG (simulating a developer)
- * Get Postmaster basket for Q-PATCH in variable QUE
- * Obtain next patch number
- * Set up a new patch
- * Add routine set in file Message
- * Get messages matching a package.
- * Create a patch (status: under development)
- * Complete the patch (status: completed/unreleased)
- * Verify the patch (status: released)
- * Export Patch to File system using new exporter code
- * Create a second patch - complete this one (leave completed)
- * Create a third patch - don't complete or verify (leave under development)
- * Test new Write Identifiers for Patch file
- * Test Report 5^A1AEPH2 (option Summary Report for a Package)
- * Test Report 1^A1AEPH2 (option Completed/unverified Patch Report)
- * Analyze TXT files produced by the VA Patch Module (tested for ALL patches on OSEHRA repository)
- * Import KIDS files directly into the Patch Module (tested for ALL patches on OSEHRA repository)
- * Test Get Stream from Patch Designation
- * Test Import of Single Build KID files
- * Test Import of Multi Build KID files
- * Test that repeated Import does not duplicate patches
- * Test that a Forum message sent to S.A1AE LOAD RELEASED PATCH will add correctly to the current Forum system
- * Test ZWRITE subscript substitution for exporting KIDS components in a versionable format
- * Test export into versionable components for ALL KIDS files in OSEHRA repository
- * Test conversion of designation from Patch Module format to KIDS format
- * Test conversion of designation from KIDS format to Patch Module format

5 Code Walkthrough

The patch module is a classic VISTA Fileman based application. It makes very heavy use of Fileman primitives for virtually all of its functions.

5.1 Fileman Files

The module has 6 files located in the 11000 range:

11005	DHCP PATCHES
11005.1	DHCP PATCH
11005.2	DHCP PATCH STATUS
11005.5	DHCP HFS MESSAGE
11006	DHCP PROBLEMS
11007	DHCP PATCH/PROBLEM
11007.1	DHCP PATCH STREAM

11005 contains the patch message; and 11005.1 contains the KIDS build for the patch. These files are DINUMed to each other. Occasionally, a developer can't send the patch to Forum because it's too large. In that case, when he/she generates a patch to the host file system, a message is sent to S.A1AE HFS MESSAGES containing the checksums of routines in the patch. The message is processed by A1AEHSVR and filed into 11005.5. If a developer says that they don't have a patch in the process of creating the message, the system will check 11005.5 to see if checksums for the routines can be found there.

11005.2 is a newly created reference file and has no relation to the other 11005 files. It was created in version 2.4.

11006 is not used.

11007 contains the package definition for each package in the patch module. Importantly, it contains who is allowed to develop it, verify it, and whether it's a test package. Through a sleight of hand, the patch module entry in this file contains the routing list of who receives any verified patch.

11007.1 is also new. It's created in version 2.4. Its function is to support multiple patch streams.

5.2 Routine Breakdown

Routine A1AEPH1 is the heart of the patch module. It's responsible for adding and editing of patches. Most of the heavy work is done by the Input Template \[A1AE ADD/EDIT PATCHES] on file 11005.

This input template invokes A1AECO* routines, which copy mail messages and routines into 11005 and 11005.1. It also calls A1AEM1 to copy KIDS builds and DIFROM and Packman messages from Mailman.

A1AEM and A1AEM2 are called elsewhere in the module and perform mailman related functions. We will mention them again.

The input transform on field 8 is the center of the patch module's workflow.

It invokes ^A1AEPHS. This routine is responsible for moving the patch through various statuses and eventually mailing it out. The routine A1AEMAL is responsible for building the message.

A1AEM helps with addressing the message for release. A1AECL1 updates the old checksums with the latest patch released.

The rest of the A1AEPH* routines are reporting related.

Routine A1AEPK adds and edits package releases. Package releases are represented as a patch with a patch number of zero. For example, VPR*1*0.

Managing the patch module is done via routine A1AEAU, which calls A1AEKEY for key allocation functions.

A1AEUTL* are shared utility routines. So is A1AEMGR, which is rather misnamed. A1AERD is a primitive reader used prior to the invention of ^DIR.

A1AEVP* are routines to help verifiers.

A1AEPB1 is for adding and editing Problems in 11006. It's not used.

A1AESP, A1AEZCON, A1AEZTST perform one off utility tasks.

A1AERCON compares patches to see if they modify the same routines.

A1AEK2M* are routines newly written in version 2.4 and are invoked through the new A1AE IMPORT menu. They import patches from the file system into the patch module. They are tested by A1AEK2MT. Any operating system functions are performed in A1AEOS if the functionality is not already provided by the Kernel.

A1AEPSVR performs the same function for VA Forum mailed messages received through S.A1AE LOAD RELEASED PATCH.

A1AEK2V* exports patch components to the file system for versioning. It's tested by A1AEK2MT since its invocation is on a cross-reference in 11005.

A1AEM2K exports patches from the Patch Module to the File system as KIDS HFS files.

A1AEUT* provide Unit Tests for the whole of the package.

And that's it. That's all of the routines in the Patch Module.

6 Conclusion

While there are still a lot of improvements that could be made to Forum and the Patch Module, this code provides an Initial Operating Capability for the VistA community to use and a foundation for further development.