

OSEHRA S54 Release Notes

Introduction

The team facing Health Management Platform (HMP-TF) is a platform for delivering software that serves healthcare teams and their needs in caring for our veterans. Our goal is to transition Veterans Health Administration (VHA) from a chart-centric model of care to a team-based, patient-centered model of care through advanced clinical software and a modern software architecture. Our approach is to reestablish close clinician-IT collaborative relationships and to build field-driven, web-based modules that meet VA's functional and technology goals.

This is version S54 (current as of 4/15/2014).

Prerequisites

1. [Caché](#) instance with JSON Data Store (JDS) routines installed (located in \hmp\jds\src\main\mumps\jds.ro)
2. Local VistA account with Virtual Patient Record (VPR) routines installed (located in \hmp\kids\HMP1_S54.KID)
3. [Java 7 JDK](#) (Java Developer Kit) installed
4. [Apache Maven](#) installed
5. Unified Medical Language System (UMLS) account for custom TermDB files (optional)
 - a. Additional documentation in docs\ directory
6. Running [OpenInfoButton](#) server (optional)

Install Overview

1. Create an hmp-dev.properties file and add a unique HMP server identifier:
 - a. (Mac) `echo hmp.server.id=$HOSTNAME > hmp-main\hmp-dev.properties`
 - b. (Windows) `echo hmp.server.id=%COMPUTERNAME% > hmp-main\hmp-dev.properties`
2. Using FileMan in VistA, create a new entry in the VPR SUBSCRIPTION file (#560) and add the hmp.server.id value from step one to the SERVER (.01) field
3. Set up an account in your local VistA system called *VPR User*.
 - a. In VistA, create a service account in the NEW PERSON file (#200) using VistA's VA Fileman or User Management>Add a New User option
 - i. First name should be *USER*
 - ii. Last name should be *VPR*
 - iii. Initials should be *UV*
 - iv. For IRM staff: Determine access and verify codes
 - b. Assign VPR SYNCHRONIZATION CONTEXT as the user's secondary menu option
 - c. Set this user's division as the default
 - d. Set TIMED READ (# OF SECONDS) to 99999
 - e. Set MULTIPLE SIGN-ON to ALLOWED
 - f. In FileMan, set VERIFY CODE NEVER EXPIRES to TRUE
4. Set up your VistA server connection in hmp-main\src\main\hmp-home\vista-accounts.json:
 - a. Change <vistaID>,<division>,<access code>,<verify code> and other values as appropriate to match the VistA user account you just created above
 - b. The <vistaID> value is a four-character hexadecimal, CRC-16 (16-bit Cyclic Redundancy Check) hash of the VistA domain name
 - i. You can obtain it in MUMPS via: "W \$\$\$SYS^VPRUTILS"
5. Compile HMP and run it in Jetty using Maven:
 - a. `cd hmp-main`
 - b. `mvn jetty:run`
6. Wait for Maven to download and resolve all dependencies and compile the code (which could take a little while the first time):
 - a. The application should be running at <http://localhost:8080/> when finished
 - b. You can log in with any VistA access and verify code pair associated with a user who has the VPR UI CONTEXT menu option
 - c. VistA patient and operational data should synchronize automatically on demand
7. Set up Eclipse project files (optional)
 - a. `cd hmp\hmp-main`

Health Management Platform (HMP) Team Facing Document

b. mvn eclipse:eclipse

Project + Directory Structure

\health-time		<p>A supporting library, based on Joda-Time, for handling imprecise dates and other clinically relevant date time concerns (such as FileMan date times, Health Level Seven (HL7) date times, and so forth)</p> <p>Uses Gradle for build and deployment scripts</p>
	health-time-core\	The core Health-Time module for parsing and formatting imprecise dates. The <code>PointInTime</code> class, which the rest of HMP uses extensively, is in this module
	health-time-hibernate\	A plugin module for Hibernate to provide custom data types that handle HL7-formatted date and time
	health-time-jackson\	A plugin module for the Jackson JSON (JavaScript Object Notation) library to support serialization and deserialization of FileMan date times and HL7 date times
	health-time-solr\	A plugin module for Apache Lucene/Solr to provide custom data types for HL7-formatted date and time
\vista-support		<p>A set of supporting libraries for interacting with VistA systems</p> <p>Uses Gradle for build and deployment scripts</p>
	vista-auth\	Spring-security-based support for authenticating web application users against VistA
	vista-rpc-client\	<p>The primary remote procedure call (RPC) client, modeled after Spring Framework templates, that provides a convenient Uniform Resource Locator-ish (URL-ish) syntax for calling VistA RPCs</p> <p>Also contains session-aware connection factories for more efficient interaction with VistA</p>
	vista-rpc-client-metrics\	A plugin module that provides VistA RPC metrics via the Metrics library
\hmp		<p>The main HMP platform project</p> <p>Uses Maven (for now) for build and deployment scripts</p>
	hmp-main\	Primary HMP project, contains all Java middle-tier code as well as JavaScript User Interface (UI)-tier code
	hmp-api\	The start of HMP's modular API, based on OSGi , is contained in this project
	hmp-addon-hello-world\	A "hello world" demonstration HMP module
	hmp-addon-classic-views\	A module that replicates portions of the classic CPRS tabs as HMP web views

Health Management Platform (HMP) Team Facing Document

	jds\	Contains MUMPS routines for the reference implementation of the JDS, a component of the VPR
	kids\	Contains the current (not-yet-released) Vista MUMPS routines VPR buses for efficient clinical data extraction from Vista (in JSON and Extensible Markup Language (XML))
	jrebel-plugin\	A JRebel plugin that Java developers may find useful
	local-addr-servlet-filter\	A localhost Java servlet filter, not currently used
	repo\	A Maven-style artifact repository for non-public build artifacts referenced in pom.xml
	docs\	Reference documentation, including some schema information (likely out of date) and terminology build information