Enterprise Health Management Platform (eHMP)
System Design Document



Department of Veterans Affairs

October 2014

Version 2.3.1

Revision History

Date	Version	Description	Author
10/17/2014	2.3.1	Updated Patient Information Section 8.4.3, added Section 6.2.11 and associated appendixes.	ASM Research
10/10/2104	2.3	Updates for AERB Review for eHMP Increment 2 MS 1 Brief.	ASM Research
9/26/14	2.2.2	Submitted as contract deliverables for CLINs: 0003AS, 1001AS, 1002AS, 1003AS, 2004AS. Draft version posted to Final Acceptance.	[redacted]
9/22/2014	2.2.1	Draft version posted for final edit.	[redacted]
9/8/14	2.2	Review and updates for eHMP 1.0. Submitted as Draft	[redacted]
5/23/2014	2.1.1	Updated Figure 3.5	[redacted]
5/15/2014	2.1	Revisions requested from 2.0	[redacted] Waltman
5/5/2014	2.0	Updates for MS 1 Increment 1, eHMP Project	[redacted]
4/22/2014	1.6	Updated terminology/graphics, MS 1 Increment 2, UX Project	[redacted]
3/3//2014	1.4	Updates to incorporate D. Waltman comments	[redacted]
3/3/2014	1.5	Updated terminology regarding project name.	[redacted]
2/26/2014	1.3	Updates to enterprise terminology	[redacted]
2/12/2014	1.2	Update after review	[redacted]
1/14/2014	1.1	PSI 1 Draft Submission	[redacted]
11/7/2013	1.0	Submission for PMAS Milestone (MS) 1, UX Project	[redacted]

Table of Contents

1. Introduction 8
   1.1. Purpose of the SDD 8
   1.2. Identification 9
   1.3. Scope 9
      1.3.1. Increment 2 Capability 9
      1.3.2. Future Incremental Capabilities 9
   1.4. Relationship to Other Plans 10
      1.4.1. VistA 4 Product Roadmap 10
   1.5. Methodology, Tools and Techniques 11
   1.6. Policies, Directives, and Procedures 11
   1.7. Constraints 12
   1.8. Design Trade-Offs 12
   1.9. User Characteristics 12
   1.10. User Problem Statement 13
2. Background 15
   2.1. Overview of the System 15
   2.2. Overview of the Business Process 15
   2.3. Assumptions 16
   2.4. Legacy System Retirement 16
3. Conceptual Design 17
   3.1. Conceptual Application Design 17
      3.1.1. Application Context 17
      3.1.2. High-Level Application Design 18
      3.1.3. Application Design Objects 20
      3.1.4. Internal Data Stores 21
      3.1.5. Application Locations 22
      3.1.6. Application Users 22
   3.2. Conceptual Data Design 22
      3.2.1. Project Conceptual Data Model 22
      3.2.2. Database Information 22
   3.3. Conceptual Infrastructure Design 23
      3.3.1. System Criticality and High Availability 23
      3.3.2. Special Technology 23
      3.3.3. Technology Locations 23
      3.3.4. Conceptual Infrastructure Diagram 24
         3.3.4.1. Load Balance 25
         3.3.4.2. Web Service 26

3.3.4.3. Cache 26
3.3.4.4. Synchronization 26
4. System Architecture 27
4.1. Hardware Architecture 27
4.2. Software Architecture 27
4.3. Communications Architecture 27
4.4. Third Party Software Components 27
5. Data Design 29
5.1. Data Model 29
5.2. Database Management System (DBMS) Files 30
5.2.1. Security 31
5.2.2. Scalability 32
5.2.3. JDS API Operations 33
5.2.4. JDS Features 37
5.2.4.1. Templates 37
5.2.5. Testing 38
5.2.6. Indexes 38
5.2.7. Required Fields 39
5.2.8. Adding Domains, Indexes, Etc. 40
5.3. Non-DBMS Files 40
6. Detailed Design 41
6.1. Hardware Detailed Design 41
6.2. Software Detailed Design 45
6.2.1. eHMP Software Development Kit (SDK) 45
6.2.1.1. Goal of the Web Application Architecture 45
6.2.1.2. SDK to Help to Achieve Goals 47
6.2.1.3. ADK Features/Anatomy 48
6.2.1.4. Roles 49
6.2.1.5. Technical Information 49
6.2.1.6. ADK Responsibilities for Applet Developers 49
6.2.1.7. ADK Responsibilities for Screen Designers: 50
6.2.1.8. ADK Responsibilities for Application Designers: 50
6.2.1.9. RDK Features/Anatomy 50
6.2.1.10. Roles 50
6.2.1.11. Technical Information 50
6.2.1.12. RDK Responsibilities for Resource Developers: 50
6.2.1.13. RDK Responsibilities for Deployment Engineer 51
6.2.1.14. Other responsibilities 51
6.2.1.15. ADK Details - SPA 51
6.2.1.16. Development Lifecycle 52
6.2.1.17. Applet Development 53
6.2.1.18. Accessing Patient Record 56
6.2.1.19. RDK Details 57

6.2.1.19.1. Resource Server 57
6.2.1.20. RDK 57
6.2.1.21. Resource Developer Responsibility 58
6.2.2. Web Server 59
6.2.3. HTTP Proxy 59
6.2.4. Web Service API 59
6.2.4.1. Implementation of the Web Service 60
6.2.4.2. Authentication 60
6.2.4.3. Access Control 63
6.2.4.4. Audit 65
6.2.4.5. Patient Search and Selection 65
6.2.5. VistA/MUMPS Development 66
6.2.6. Patient Sync Subsystem 66
6.2.7. Patient Correlation Service 67
6.2.8. Cache 67
6.2.9. VistA Access Approach 67
6.2.10. Patient Generated Data 68
6.2.11. eHMP VistA Components 68
6.2.11.1. Data Dictionaries (DD) 68
6.2.11.2. Options 68
6.2.11.3. Security Keys 68
6.2.11.4. Protocols 69
6.2.11.5. Remote Procedure Calls (RPC) 69
6.2.11.6. Health Level 7 (HL7) Application Parameters 69
7. External Interface Design 70
8. Human-Machine Interface 71
8.1. Summary 71
8.2. JLV Functionality 71
8.3. Patient Search 71
8.4. Patient Information 71
8.4.1. Crisis Notes, Warnings, Allergies, and Directives (CWAD) Postings 72
8.5. HMP Functionality 72
8.6. Header 73
8.7. Footer 73
9. System Integrity Controls 78
9.1. Transmission Encryption 78
9.2. System Administrator Access 79
9.3. Flaw Remediation 79
9.4. Configuration Management (CM) 80
Attachment A – Approval Signatures 81
A. Appendix 1 – Acronym List i

B. Appendix 2 – FHIR Mappings v
C. Appendix 3 - VistA Routines (Entry Points) vii
D. Appendix 4 - Data Dictionaries (DD) lxxxviii
E. Appendix 5 - Options xciii
F. Appendix 6 - Security Keys ciii
G. Appendix 7 - Protocols cv
H. Appendix 8 - Remote Procedure Calls (RPC) cxiv
I. Appendix 9 - Health Level 7 (HL7) Application Parameter cxxiii

Table of Figures

Figure 1-1 Layered Architecture with Planned Components 9
Figure 2-1 Current Means for Accessing VA Patient Data 16
Figure 3-1 eHMP Business Logical View for ONC Certification 17
Figure 3-2 VistA Exchange Infrastructure Phase 1 19
Figure 3-3 VistA Exchange Infrastructure Phase 2 20
Figure 3-4 eHMP Regional Architecture 24
Figure 3-5 AITC Topology 25
Figure 5-1 ECP Architecture 32
Figure 6-1 eHMP System Architecture 45
Figure 6-2 VistACore SDK 48
Figure 6-3 Example of a 3x3 Screen Layout 49
Figure 6-4 Applet Development Lifecycle 52
Figure 6-5 Application Development Lifecycle 53
Figure 6-6 Authentication Workflow 61
Figure 6-7 Interim Solution Access/Verify Code 62
Figure 6-8 September 2015 Enterprise Solution 62
Figure 6-9 CPRS Tab Access 63
Figure 6-10 Patient Sensitivity Command Response 64
Figure 8-1 Patient Cover Sheet 74
Figure 8-2 Patient Selection Screen 75
Figure 8-3 Patient Selection (Clinic) 76
Figure 8-4 Global Patient Selection 77

Table of Tables

Table 3-1 Application Design Objects 20
Table 3-2 Internal Data Stores 21
Table 4-1 Third Party Components 27
Table 5-1 FHIR Pros and Cons 29
Table 5-2 OPD and OPR Mechanisms 30
Table 5-3 Configuration 30
Table 5-4 Clearing JDS 31
Table 5-5 JDS Implements 31
Table 5-6 Patient Data VPR 33
Table 5-7 Non Patient Data (ODS) 34
Table 5-8 Searching for Patients 34
Table 5-9 Examples 35
Table 5-10 Indexing 37
Table 5-11 Running Unit/Integration Tests 38
Table 5-12 Simple Integration Test 38
Table 5-13 Template Integration Test 38
Table 5-14 Destroying Test Patients 38
Table 5-15 jds.ro Excerpt 39
Table 6-1 Production Detail VM requirements 41
Table 6-2 Pre-Production Detail VM Requirements 43
Table 6-3 Shared Servers Detail VM Requirements 44
Table 6-4 Extra Servers Detail VM Requirements 44
Table 6-5 Applet Example Code 54
Table 6-6 Applet Template Example Code 55
Table 6-7 Example of Row Template Code 56
Table 6-8 Example of Simplified Applet Code 56
Table 6-9 Resource Config 58
Table 6-10 Installed Software 59
Table 6-11 Resource Server Technology Stack 60
Table A-1 Acronym List i

1. Introduction

This System Design Document (SDD) has been developed, and will be regularly updated, as part of the Enterprise Health Management Platform (eHMP) project. The eHMP project, within the Veterans Health Information Systems and Technology (VistA) Evolution program, will introduce expanded capabilities and modernize existing features of the Department of Veterans Affairs' (VA) VistA Electronic Health Record (EHR) system. The eHMP project is a multi-year effort to evolve a modern, service-oriented Enterprise Health Application Platform. This platform includes VistA Exchange, the eHMP Clinical Practice Environment (CPE), and certain clinical knowledge enrichment and decision support services. VistA Exchange is the clinical data services engine, federating clinical data from a variety of VA and Department of Defense (DoD) sources into an Enterprise Virtual Patient Record (eVPR).The eHMP CPE framework will incorporate capabilities currently provided by the Joint Legacy Viewer (JLV) and the hi2 HMP. eHMP CPE will eventually replace the Computerized Patient Record System (CPRS) as VA's primary point of care application. eHMP Services components will span all application layers, including presentation, business and core services, as well as data access. The eHMP project will produce epics (capabilities) supportive of the VistA 4 Product that relate to the core clinical user experience [see the VistA 4 Product Roadmap]. For an in-depth introduction to the project, refer to the Contractor Project Management Plan (CPMP).

1.1. Purpose of the SDD

The SDD translates the requirement specifications for each increment of the eHMP project into a design from which developers can create the system. It identifies the top-level system architecture and identifies hardware, software, communication, and interface components. More specifically, this document details how the eHMP project integrates architecture spanning the presentation, service, and data layers, and supports the vision of a robust core clinical user experience application as laid out in the Vista 4 Product Roadmap. The key mandates supported by the eHMP project include: Department of Defense (DoD) Interoperability, Commercial third party interoperability, Meaningful Use (MU) Certification, and full Generation 3 Electronic Health Record (EHR) features. Figure 1-1 below provides a summary of these layers and their planned components.

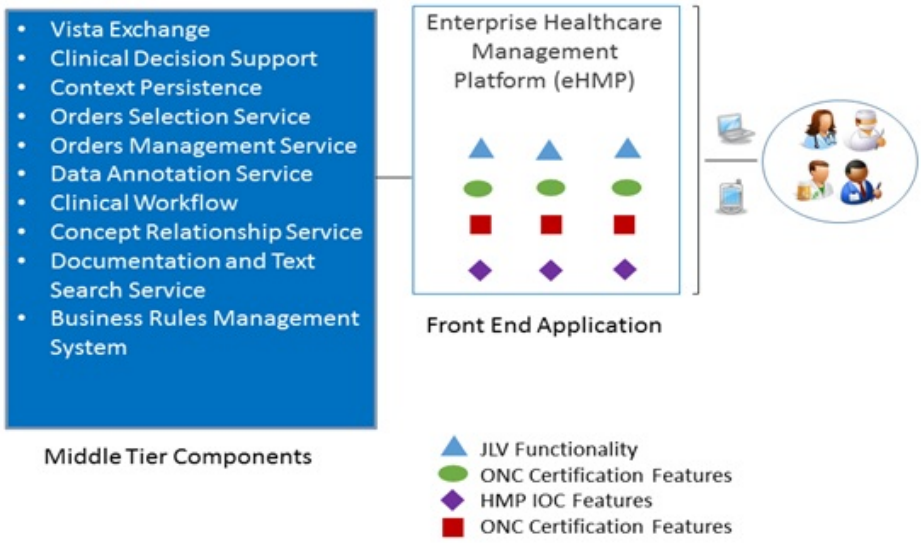


Figure 1-1 Layered Architecture with Planned Components

1.2. Identification

The SDD describes the ongoing evolution of the eHMP systems initiated under the VistA UX project, and that continues under the second increment of the eHMP project. Version numbers will be assigned to the resulting system once development of an increment has been completed and delivered to the VA. This is a living document that will be updated to reflect the rapid and iterative evolution of the design, functional, and architectural capabilities defined for each increment of the eHMP project.

1.3. Scope

1.3.1. Increment 2 Capability

Increment 2 will have three main focus areas. First will be Computerized Provider Order Entry (CPOE) for Medications, Laboratory and Radiology/imaging orders. Functionality surrounding Drug-Drug/Drug-Allergy interaction checking will also be developed. The last main focus area surrounds the management of Consolidated Clinical Document Architecture (C-CDA) documents. ONC requires a system to be able to receive, display and incorporate data coming in from external parties in that format as well as produce C-CDA documents to send to external parties. Beyond these main focus areas we will also address managing Immunizations and capturing Smoking Status.

1.3.2. Future Incremental Capabilities

Based on priorities established by the eHMP governance structure, future incremental capabilities are expected to include:

- Enhanced DoD Interoperability: Architectural improvements will expand performance and reduce maintenance for components supporting DoD interoperability.
- Advanced indexing and clinical data enrichment: VistA Exchange will leverage the eVPR cache to create a universal index for search, as well as allow the addition of

external clinical enrichment features, e.g., natural language processing, tagging, and annotations.

- Web-Based Computerized Patient Record System (CPRS): One or more increments will be devoted to designing a web-based alternative to CPRS. The UI will be built leveraging designs, functionality and capabilities that exist in the hi2 HMP UI.
- Open standards: eHMP will enable VA to decouple ancillaries from core clinical functionality through open standards. One or more increments will be devoted to creating open, standards-based services for orders management and order selection. This will allow orders to be managed as enterprise activities, and will decouple the generation of the clinical record from ancillary fulfillment, thus facilitating the selection and integration of Commercial Off-the-Shelf (COTS) ancillary products, such as Laboratory and Pharmacy (the ancillary product acquisitions are outside the scope of eHMP project).
- Other Evolutionary Generation 3 EHR Capabilities: eHMP will enable other future evolutionary EHR capabilities, including services for Clinical Workflow and Activity Management, Decision Support, and Concept Relationship.

1.4. Relationship to Other Plans

This document discusses general system designs for the VistA Exchange and eHMP web application components within the eHMP project.

1.4.1. VistA 4 Product Roadmap

The architecture and system components described in this document align with the VistA 4 Product Roadmap. Particular areas in which this SDD and the VistA 4 Product Roadmap align include:

- Data Exchange Approach: Fast Healthcare Interoperable Resource (FHIR), JavaScript Object Notation (JSON), and Representational State Transfer (REST), among other standards, will be used to support data exchanges with local VistA instances and outside data sources, such as the DoD.
- User Experience and the UI: Based on a web services approach, the VistA Exchange architecture will support both web browser hosted and locally installed, service-aware applications on various end user devices. The multi-layered approach (presentation, services, and data) described in this document is consistent with the qualities described for an effective user experience.
- Use of Service-Oriented Architecture (SOA): VistA Exchange will be built on web service architecture compliant with the principles detailed in the VistA 4 Product Roadmap.
- VistA Infrastructure: The VistA Exchange and eVPR services may be deployed within a virtualized VA cloud environment.
- Data Model: The eVPR cache will be implemented using a canonical model pattern, in which clinical data will be cached following a standard information model that allows for translation into various interoperability standard formats. The internal canonical models for the eVPR cache will be the VPR JSON and FHIR-based canonical models.

- Decision Support: The eVPR will provide the aggregated patient records necessary to enable an efficient and effective clinical decision support system.
- Meaningful Use: This effort will enable VA to achieve the Meaningful Use goals outlined in the Office of Management and Budget (OMB) memorandum via EHR Meaningful Use.
- Database Management System: InterSystems Caché will be used to support the caching function of the eVPR service within VistA Exchange. VistA Exchange is not a new data source, but rather an aggregation/federation engine for data from existing Systems of Record.
- Support for Future Ancillary Systems Integration: Open standards will be used to facilitate the interfaces between the orders management service and future ancillary COTS.
- Contractor Project Management Plan (CPMP): Summarizes the project and describes how the project will be managed.
- Solution Architecture: Brief summary of the architecture and design of the system.
- Open Source Software Systems Engineering Plan: Describes the engineering management approach and methodologies employed on this project.

1.5. Methodology, Tools and Techniques

The SDD is a living document. This version reflects the ongoing system design as proposed and developed for increments 1 and 2. Additional updates are planned as the design evolves within each increment, and for each subsequent increment of the eHMP project. The Scaled Agile Methodology (SAM), summarized in the CPMP is a framework for a governance structure that aligns executive level priorities to an agile development methodology. Each increment of development will be managed through six, two-week sprints grouped into Potentially Shippable Increment (PSI) cycles. Thus, the system design will be evolutionary. Documented updates will occur with each PSI. In addition to PSI planning and development activities, other updates to the system design will be driven by the VA Accreditation with Assessment and Authorization (A&A) Process and other certification events (e.g., user experience and 508).

1.6. Policies, Directives, and Procedures

The system design will comply with VA policies, directives and procedures including:

- VA-One Technical Reference Model (TRM)
- VA ProPath and PMAS standards
- VA A&A and related security policies
- VA Directive 6500 Information Security Handbook

If the planned system design is found to conflict with a VA policy, directive, or procedure, this conflict will be discussed with the VA Project Manager, who will help to determine whether a waiver or exception will be requested from the applicable VA governing board.

1.7. Constraints

The system is constrained so as not to be a system of record for any patient data for this current incremental release. VistA Exchange is an aggregation system, but any data aggregated will remain stored in source systems. VistA Exchange's use of a data store is for caching and/or performance optimization only.

1.8. Design Trade-Offs

The system's responsibility is to provide a feature-rich and responsive UI. VistA Exchange provides consolidated data services to drive the UI, including longitudinal patient data from various sources. It is expected that the number of users will be relatively low initially (during pilot stages), but will need to grow to the same number of users that utilize CPRS enterprise-wide. VistA Exchange could potentially support other consumers, such as DoD or veteran-centric mobile applications, staff-centric mobile applications, and external applications; thus, VistA Exchange is architected to scale by ensuring that performance does not degrade as the number of users increases with linear addition of hardware resources. This architectural focus can be demonstrated in the following design trade-offs:

- Favor caching data within VistA Exchange over retrieving all data from sources at every query. The cache allows for data to be aggregated from multiple sources, indexed based upon various usage, and to normalize terms against terminologies without having the user wait for these events to occur.
- Maximize the responsiveness of the UI by utilizing client-side controllers and a Single Page Application to make network calls to retrieve data.
- Favor an asynchronous API interface to maximize scalability over performance. When data is not already cached in VistA Exchange for a given patient, the API returns a deferred response using standard HTTP constructs, instructing the client to check again in a few seconds. During this time, the data is synchronized from its sources. This approach does minimally increase the time to get data in a situation where the data has not been cached prior to request, because the client will make a second request providing a delta between when the data is available in the cache to the time that the second request is made. However, this per-transaction performance impact is expected to be overshadowed by the performance and scalability gains, versus a synchronous interface. In the synchronous approach, HTTP connections would need to be maintained (left open) for several seconds to minutes while data is retrieved from source systems, requiring more memory to the web API. This approach is further explained in section 6.2.4 (Web Service API).
- Favor the use of microservices, rather than traditional web server clusters, to maximize discrete scalability options, and to ensure that services remain loosely coupled.

1.9. User Characteristics

The first PMAS increment of this project will introduce the eHMP CPE for authorized users throughout a VA medical facility. Users will benefit from accessing a comprehensive patient record that includes documented care within the VA and DoD medical systems. Clinicians will use the system to help make health assessments, diagnoses, and determine treatment based on a more complete longitudinal patient record. Clinicians will continue to use CPRS to document the

delivery of care, including order entry, until those CPRS capabilities are replaced by later increments of this system. It is also expected that in future increments, select Benefits Claims Examiners of the Veterans Benefits Administration (VBA), will use the system to help adjudicate benefits claims that require review of both DoD and VA medical records information. Initially, and through at least 2018, VBA examiners will use the JLV functionality within eHMP.

1.10. User Problem Statement

The eHMP system represents the portion of the VistA 4 Product that implements end-user facing features, e.g., the UX for the clinical users (team members who are responsible for direct patient care) with the VistA Exchange web services architecture. As such, it includes the UI as well as underlying supporting services and access to data. The eHMP system is an extension and evolution of VistA, and does not represent a system wholly separate from VistA. Specifically, the eHMP system is not meant to produce a separate System of Records (SOR) from the existing VistA SOR.

The following excerpt from the Vista 4 Product Roadmap defines the user problem statement that the eHMP project addresses:

> Today, VA providers depend on a point-of-care GUI called Computerized Patient Record System (CPRS) for accessing and recording most of the patient information needed for clinical documentation and decisions. CPRS provides very limited capability for accessing patient information that may have been recorded in many different locations both during and after a Veteran's military service. While CPRS has been an effective tool for over 17 years, its GUI is based on a paper-chart metaphor and segregates data that need to be considered together to improve understanding of the patient's state of health. For example, medications often need to be considered with laboratory tests to understand how medications might affect these results in negative or positive ways. In CPRS, users must frequently switch between windows to see both medication and other data.
>
> The VistA 4 GUI will facilitate display of interoperable data and help drive data standardization. The VistA 4 GUI, when integrated with a shared VA/DoD interoperability platform, will provide access to a Veteran's complete longitudinal health record, comprised of data collected from every military and VA treatment facility in which the Veteran has received care. For many Veterans, health data from other providers will be available through eHealth connections with participating institutions. The rich, multi-sourced data comprising a Veteran's longitudinal health record must be understood by all systems accessing and adding to it, including the VistA 4 GUI. The VistA 4 user experience will drive advances in the application of national standards for coding, sharing and interpreting health information both in VA and the nation. For example, in order to reliably graph vital signs or laboratory results, the GUI must be served homogeneous data such as a particular chemical measured by blood and not urine tests. The units of measurement must also be the same to allow meaningful comparison of all results.

The eHMP project will address these problems in the context of three major VA mandates:

- DoD Integration as defined by the 2014 National Defense Authorization Act (NDAA) legislation and 2014 Consolidated Appropriation acts;
- Meaningful Use EHR certification using the 2014 ONC criteria (stage 1 and 2);
- Generation 3 EHR capabilities.

2. Background

2.1. Overview of the System

The eHMP project is a multi-faceted development effort to modernize VistA. At its completion, this effort will produce:

- Enterprise Virtual Patient Record (eVPR): A longitudinal patient record for a given patient that includes data from VA sources (such as VistA), as well as external sources such as the DoD and Nationwide Health Information Network (NwHIN)/eHealth Exchange.
- VistA Exchange: Middleware that provides the eVPR, allowing for delivery of a single, common health record to be utilized throughout the continuum of care. This implementation includes a Web Service API to access a given patient's data; retrieval, indexing, and aggregation of clinical data from current VistA instances (and eventually other data sources); caching of the patient record to optimize performance; document enrichment to add additional data to a given patient record event; and normalizes the data from various sources using standard terminology terms.
- eHMP Web Application: Includes the integration of selected HMP UI features, and selected JLV UI features. JLV functionality and the eHMP Web Application will support ONC 2014 EHR certification.
- Clinical Decision Support Service
- Clinical Workflow and Activity Management Service
- Orders Management and Orders Selection Service
- Concept Relationship Service
- Universal indexing and searching
- Data Annotation
- Persistence of context for a clinical user across devices
- Integration with VA Enterprise Identity and Access Management

This document covers the portion of the eHMP project that is currently being developed, specifically the introduction of the eHMP CPE enabling viewing of aggregated VA and DoD patient data with initial supporting functions (e.g., patient search, free text search).

2.2. Overview of the Business Process

Presently, using CPRS, most VA clinicians have access only to patient data that resides at a single VistA location. Though standalone local demonstration projects exist through HMP and a single means to access enterprise-wide patient data only exist with the view-only JLV application, no enterprise wide application with both a data view and write back exists. Figure 2-1 identifies how patient record data is currently extracted from VistA. In addition, though CPRS has been an effective UI to local VistA data, it requires substantial efforts to update the thick client server configuration that is essentially bound to desktop computers.

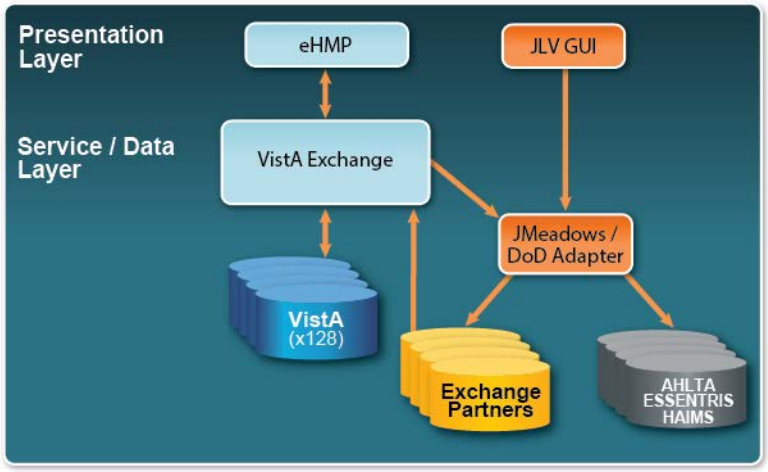


Figure 2-1 Current Means for Accessing VA Patient Data

Through the eHMP project, the presentation, service, and data layers will be modernized to enable the creation of a comprehensive patient record extracted from each VistA instance and outside sources including the DoD and outside partners. Figure 2-1 depicts the means for accessing patient data with eHMP. The resulting architecture will provide a single, adaptable, and flexible web-based UI allowing to the use of web-based services. To support the access and display of DoD and other externally sourced patient data, a JLV functionality (the same as DoD's JLV functionality) will be available in the eHMP CPE. Beneficiaries of this single UI, with multiple views, will include clinicians and Benefits claims examiners who also require access to a comprehensive EHR.

2.3. Assumptions

The following assumptions were made in preparation for this project:

- The proposed architecture and system design are accepted by VA Office of Information and Technology (OI&T) and Veterans Health Administration (VHA) decision makers.
- The VA Project Manager, under guidance from the Configuration Control Board (CCB), will determine when increments developed under this project will be deployed into a production environment; one or more increments could be bundled for deployment and submitted for PMAS Milestone 2 review.
- This SDD will be regularly updated and identified with version numbers to describe the expanded system design for each PSI and PMAS increment.
- Unless otherwise stated, software will be open source and compliant with VA's Technical Reference Manual (TRM).

2.4. Legacy System Retirement

The eHMP Web Application provides support for clinicians at point of care. This includes allowing a clinician to view the patient's data from VA, DOD, and other sources; update a patient record by updating data in VistA; perform a patient encounter; and place orders.

This functionality, at full deployment, potentially allows for deprecation of VistaWeb, BHIE Viewer, JLV and CPRS.

3. Conceptual Design

3.1. Conceptual Application Design

3.1.1. Application Context

Figure 3-1 provides context for eHMP project, in relation to other VA and DoD systems. It shows the context in which VistA Exchange and the eHMP CPE will operate towards meeting ONC 2014 Certification during the August 2015 period.

The system under development for eHMP Increments 1 and 2 includes VistA Exchange and the eHMP CPE which will be introduced to the system. Further, VistA Exchange will be enhanced to support other eHMP services, offering patient data optimized for eHMP functionality and supplying operational data (such as lookup lists).

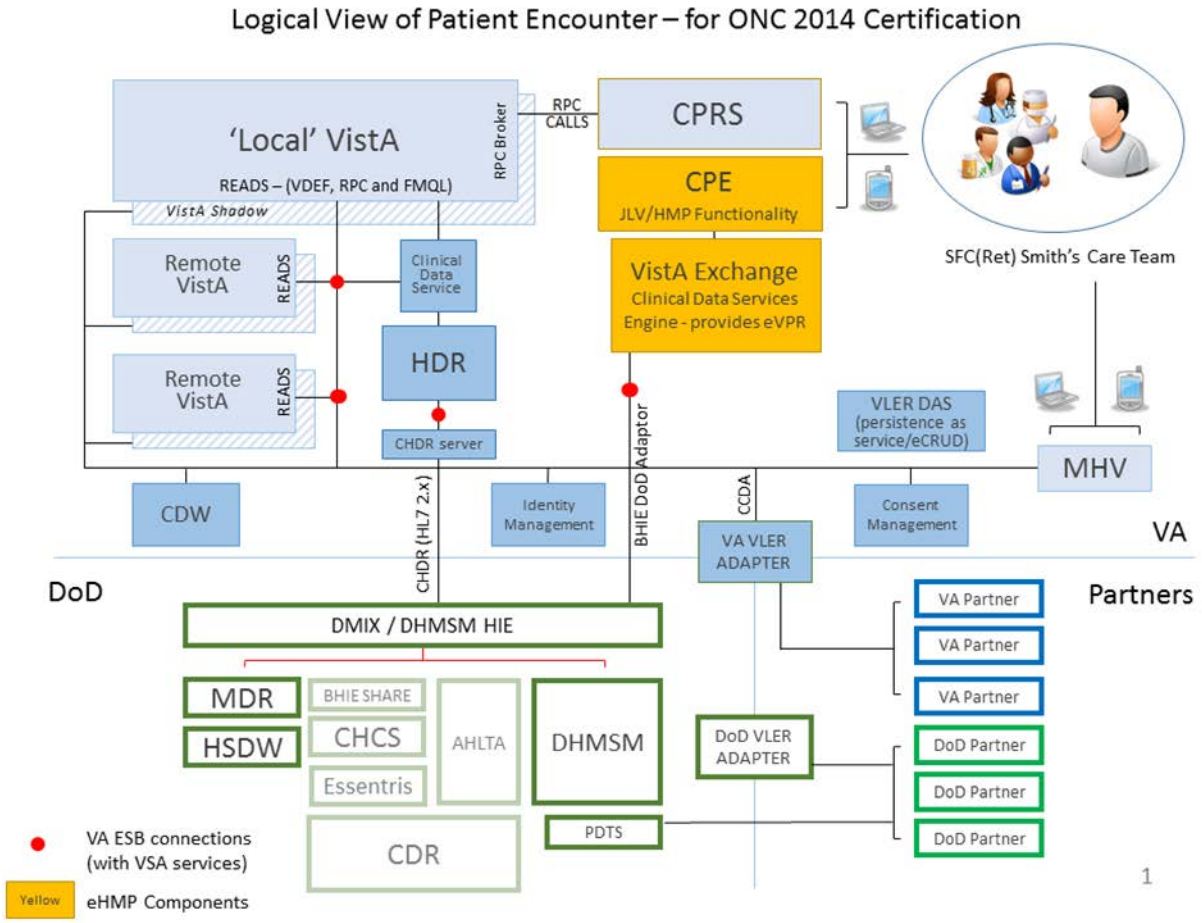


Figure 3-1 eHMP Business Logical View for ONC Certification

Based upon current VistA Exchange requirements, the focus is on providing the patient record from VistA, with future requirements planned on access from other systems.

3.1.2. High-Level Application Design

The system under development for eHMP Increments 1 and 2 include VistA Exchange and the eHMP CPE. The high-level application design for each application will be described below.

VistA Exchange: VistA Exchange is responsible for providing an enterprise patient record, including data from the VA and DoD. Based upon business triggers (such as an appointment, admission, or patient search), VistA Exchange begins to aggregate data from source systems for a selected patient. The data is normalized using standard terminologies to include consistent terms; that data is stored in an indexed method to ensure rapid retrieval based upon the patient identifier or through free text search.

VistA Exchange will leverage the HMP Virtual Patient Record (VPR) logic. During early deployment of VistA Exchange, the eVPR will be used essentially as is. VistA Exchange will then be evolved to achieve scalability and performance goals as features and users are added. This evolution is depicted in the diagrams below where Figure 3-2 depicts Phase 1 and Figure 3-3 depicts Phase 2. Tables 3-1 and 3-2 describe each entity depicted in the Phase 1 and 2 figures. This evolution may iteratively happen during Increments 1 and 2, or in future increments. The primary influencer to drive these changes will be in the ability to acquire direct connections to VistA and DoD Adapter (or the ability for the Enterprise Service Bus (ESB) to mediate these connections) and the ability to deploy the HMP VPR Publish/Subscribe (pub/sub) remote procedure calls (RPCs) across the enterprise.

The VistA Exchange web API allows a user to retrieve a patient's record by Integration Control Number (ICN). If that patient has not been synchronized (does not currently have a valid record in the cache), a synchronization request is initiated. This establishes a subscription against a local site using a set of HMP VPR RPCs. Within VistA, the existing data for the patient is gathered and made available to the cache. All future changes to the patient's record will be made available to the cache. This process is explained in further detail in Section 6.2.

In addition to getting data from the local site, Vista Exchange retrieves data from remote VA and DoD sites via Clinical Data Service (pull) and the DoD Adaptor. Eventually, VistA Exchange data will be retrieved from VistA via pub/sub (push/pull).

The data is normalized using a set of terminology tables. This process ensures that the data has a standards-based term for certain concepts within the patient record, such as using an LOINC-based term for a lab test.

Data is cached in two ways. First, the data is stored within a custom document store JSON Data Store (JDS), developed as part of HMP, and hosted by an InterSystems Caché database. This is used for primary retrieval of data by patient. Second, the data is stored within Searching On Lucene w/Replication (SOLR), optimized for free text searches across a patient's record.

eHMP Web Application: The eHMP web application provides functionality to support a clinician at the point of care with a patient. It is the application that will ultimately perform many of the functions of CPRS, and utilizes VistA Exchange as its source of patient data. The web application is developed as a Single Page Application, with client-side model view controller (MVC). This approach will limit the impact on the user experience associated with network latency as all behavior logic is contained within the client, not the server.

The application provides a development time Software Development Kit (SDK) and a runtime container in which small units of functionality can be developed and executed. The SDK is described in section 6.2.1.

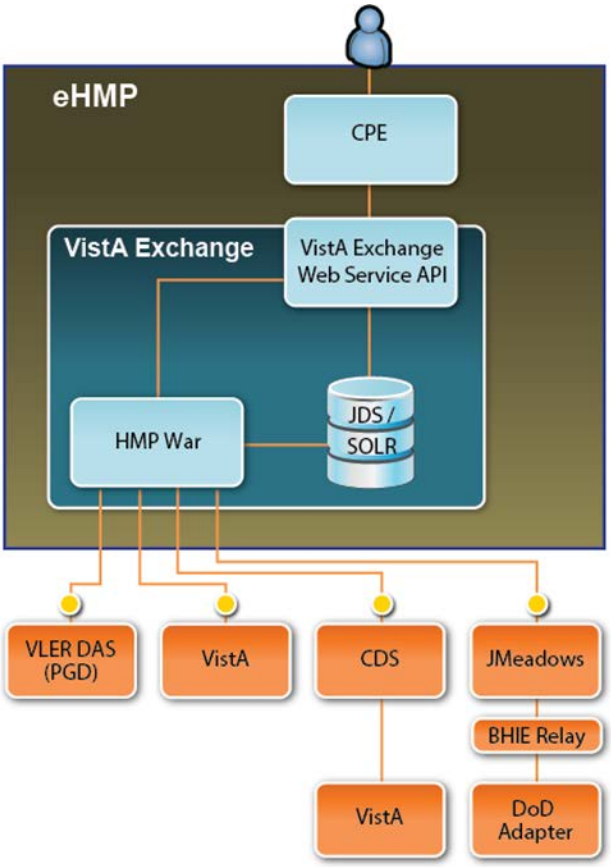


Figure 3-2 VistA Exchange Infrastructure Phase 1

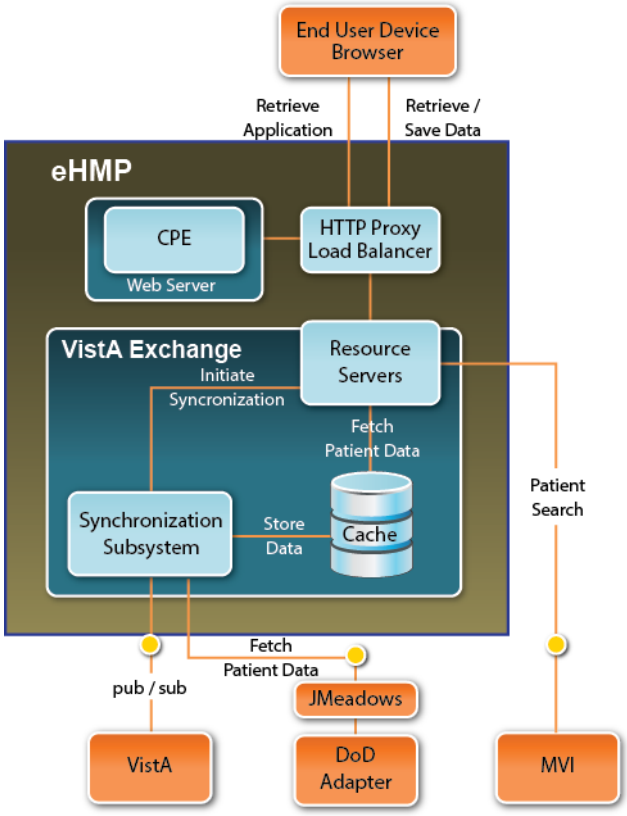


Figure 3-3 VistA Exchange Infrastructure Phase 2

3.1.3. Application Design Objects

Table 3-1 describes the primary objects of the high-level application design for Phases 1 and 2.

Table 3-1 Application Design Objects

ID	Name	Description	Service or Legacy Code
1	eHMP CPE	The web application to support provide point-of-care healthcare activities	None
2	VistA Exchange	The middleware that aggregates, normalizes, and indexes the enterprise patient record	HMP Synchronization Process
3	VistA Exchange Web Service API	Exposes an API that allows for consumers (such as the eHMP CPE) to use to fetch a given's patient's record.	None
4	JDS	Custom JSON document store, originally part of the HMP platform. This stores patient data in the form of JSON documents	HMP JDS Code

ID	Name	Description	Service or Legacy Code
5	SOLR	SOLR indexes patient data, both structured and unstructured, to allow for full text searching	HMP SOLR Code
6	HMP WAR	This web application archive (WAR) is the deployable artifact from HMP. For VistA Exchange, this is primarily used to initiate synchronization with VistA	Modified from original HMP code
7	JMeadows	JMeadows is a middleware component part of the JLV stack used from for retrieving data from the DoD and VA	Service
8	BHIE Relay	BHIE Relay acts as an adapter from the JMeadows SOAP service to the DoD Adapter REST interface	Service
9	DoD Adapter	DoD Adapter is a set of REST web services to retrieve data from various DoD systems	Service
10	MVI	Master Veteran Index provides the ability to search for patients and to translate veteran patient identifiers to other patient identifiers	Service
11	CDS	Clinical Data Services is used to retrieve data from remote VistA sources	Service
12	ESB	Enterprise Service Bus (denoted by yellow circles at connection points in the figures) – if available at the time of deployment, the ESB will mediate web service connections with the VA network.	Service

3.1.4. Internal Data Stores

Table 3-2 describes the internal data stores of the high-level application design for Phases 1 and 2.

Table 3-2 Internal Data Stores

ID	Name	Data Stored	Steward	Access
1	JDS	Cached instance of the patient health record data	VistA Exchange	This is read-only via the patient record Web Service API
2	SOLR	Cached instance of the patient health record data, optimized for full text search	VistA Exchange	This is read-only via the patient record Web Service API

3.1.5. Application Locations

The application integration environments will be located within the VA Enterprise Development Environment (EDE) cloud based out of the Austin Information Technology Center (AITC). The Pre-Production and Production environments will be in the VA Enterprise Operations (EO) cloud environment, also based out of AITC, and eventually to other regional data centers (as described in section 3.3.4). The EDE and EO backup cloud environments are located within the Philadelphia Information Technology Center (PITC).

3.1.6. Application Users

Application users of the eHMP system will be clinicians at VA medical facilities. Future users are expected to include VBA claims examiners, using the JLV application view, who require access to comprehensive VA and DoD patient records.

3.2. Conceptual Data Design

3.2.1. Project Conceptual Data Model

The canonical data model for eHMP is VPR. The VPR model is that which has been established by the Health Management Platform (HMP). This is the native format that is exposed by the HMP RPC's for its request/response and publication/subscription pub/sub routines. This is a custom format.

In addition to the RPC's, this format is realized in the JDS (the VistA Exchange cache), the VistA Exchange API (the set of web services exposing the clinical data), and the transport model for the web application.

In addition to the VPR web services, there is a set of FHIR-based Web Service API's for VistA Exchange. For these services, the data is transformed just in time from VPR to FHIR.

The choice to support these two models is based on the following assertions:

- the existing HMP RPC's natively support VPR
- the web application needs a lightweight view model
- the VPR model is better optimized (lightweight) to be a view model than FHIR
- VistA Exchange needs a standards-based transport model to support interoperability
- FHIR is standards based

The data model is described further in section 5.1.

3.2.2. Database Information

The clinical data is cached in a subsystem called JDS, which is an M-based document data store developed originally as part of the HMP program. JDS provides an HTTP interface that allows access through a set of HTTP GET and HTTP PUT commands. JDS is hosted within InterSystems Caché.

The JDS system has no direct communication with the individual VistA instances, but rather connects to the eHMP infrastructure. The eHMP connects to each VistA instance through a series of broker calls and protocols. After an initial subscription an HMP protocol will trigger

when patient data or operational data is changed. This will trigger a VistA file to be created. When the eHMP polls the VistA instance an HMP Remote Procedure Call (RPC) is called that returns all the data elements that require re-synching. eHMP then begins the process to refresh this data.

3.3. Conceptual Infrastructure Design

The deployment of the eHMP system (herein, the system) will be done using a multi-instance approach. Each instance is an autonomous deployment of the full stack of software, including the eHMP CPE web application and VistA Exchange. It is expected that at full deployment, there will be four instances, each to support 1/4 of the VistA sites, and a fifth instance to support external consumers. This configuration is referred to as the 4+1 model. Figure 3-4 diagrams the 4+1 model.

Each instance is autonomous from the other instances. An instance provides for a front-end proxy to direct HTTP traffic from the end user browsers or other consumers of the system; a web server to provide the static HTML/JavaScript/CSS of the web application; a cluster of middleware servers; a Caché instance to host the cached patient record; and synchronization processes to maintain data from VistA and retrieve data from other sources.

An instance has affinity towards a given set of VistA sites. That is, for a given site, when users access the eHMP CPE web application, they will be directed to a specific instance of the system. That instance, however, will have the ability to collect data from all VistA instances to ensure that the patient data is an enterprise perspective, including all VistA instances and external sources such as DoD. Although not required, it is expected to deploy each instance of the system to a different geographic data center. These can be organized to be optimized for the lowest network related latency for end users.

3.3.1. System Criticality and High Availability

The application integration environments will be located within the VA EDE cloud environment based out of AITC. The Pre-Production and Production environments will be in the VA EO cloud environment, also based out of AITC. The EO environment is a Federal Information Security Management Act (FISMA) high cloud environment and, as such, has disaster recovery and tape backup already in place. The EDE and EO cloud environments are also located within PITC.

3.3.2. Special Technology

No special technology is needed.

3.3.3. Technology Locations

The application integration environments will be located within the VA EDE cloud environment based out of AITC. This is the environment in which integration testing will be performed. The Pre-Production and Production environments will be in the VA EO cloud environment. The initial instance will be in AITC, rolling out to other data centers in future increments. Section 3.3.4 describes the future deployment model, which expands to regional data centers.

3.3.4. Conceptual Infrastructure Diagram

The full deployment of the system includes four autonomous instances, along with a fifth for supporting external consumers (4+1 model). The VA has four regional Datacenters that will have Enterprise Operations (EO) cloud environments. The EO cloud environments will be deployed in each of the four regional data centers where the VistA systems are also located. The +1 deployment can happen at any of the 4 regional datacenters and will be used for the external calls to systems like the DoD. Each regional datacenter will have the same deployment model/structure. Figure 3-4 diagrams the 4+1 model.

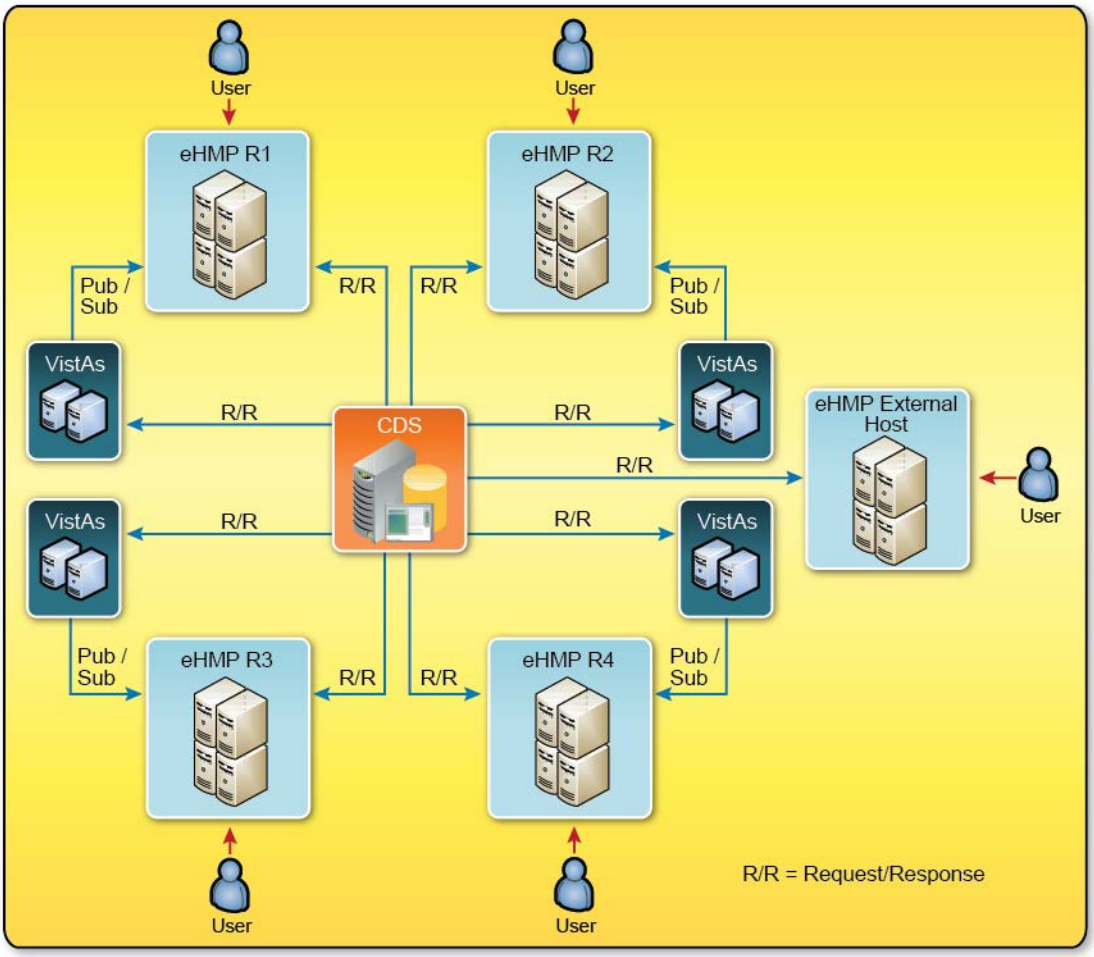


Figure 3-4 eHMP Regional Architecture

Each eHMP instance of the system in the Regional datacenters will be primarily tied to the VistA systems that are collocated at the datacenter.

Figure 3-5 diagrams the initial instance infrastructure. The initial instance will be deployed at the AITC EO cloud environment. The environment will sit behind the AITC firewall(s) and will be hosted on EO VMWare ESXi hosts. Each of the ESXi hosts will be connected to a Storage Area Network (SAN) that will hold all data and virtual images. EO will manage the SAN and ESXi servers and AITC will manage the firewall(s).

The virtual server will consist of a front end load balancer (unless an F5 or some other Load Balancing appliance can be provided) that will sit in front of the UI and API servers. The front

end load balancer (hardware or virtual machine) will allow for easy expansion of the UI/API servers to meet capacity/load needs as they arise.

The Storage and Synchronization servers will be placed behind the UI/API servers. The storage server will be a single instance that will host JDS and SOLR, with additional servers to be added as needed. The Synchronization server will be configured as a two-box cluster and expanded as needed.

Location within AITC will enable full connectivity to required remote systems, including VistA instances, MVI, Corporate Data Warehouse (CDW), Health Data Repository (HDR)/Clinical Data Service (CDS), and Veteran Lifetime Electronic Record (VLER) Data Access Service (DAS). Figure 3-5 further describes the above characteristics of the AITC environment.

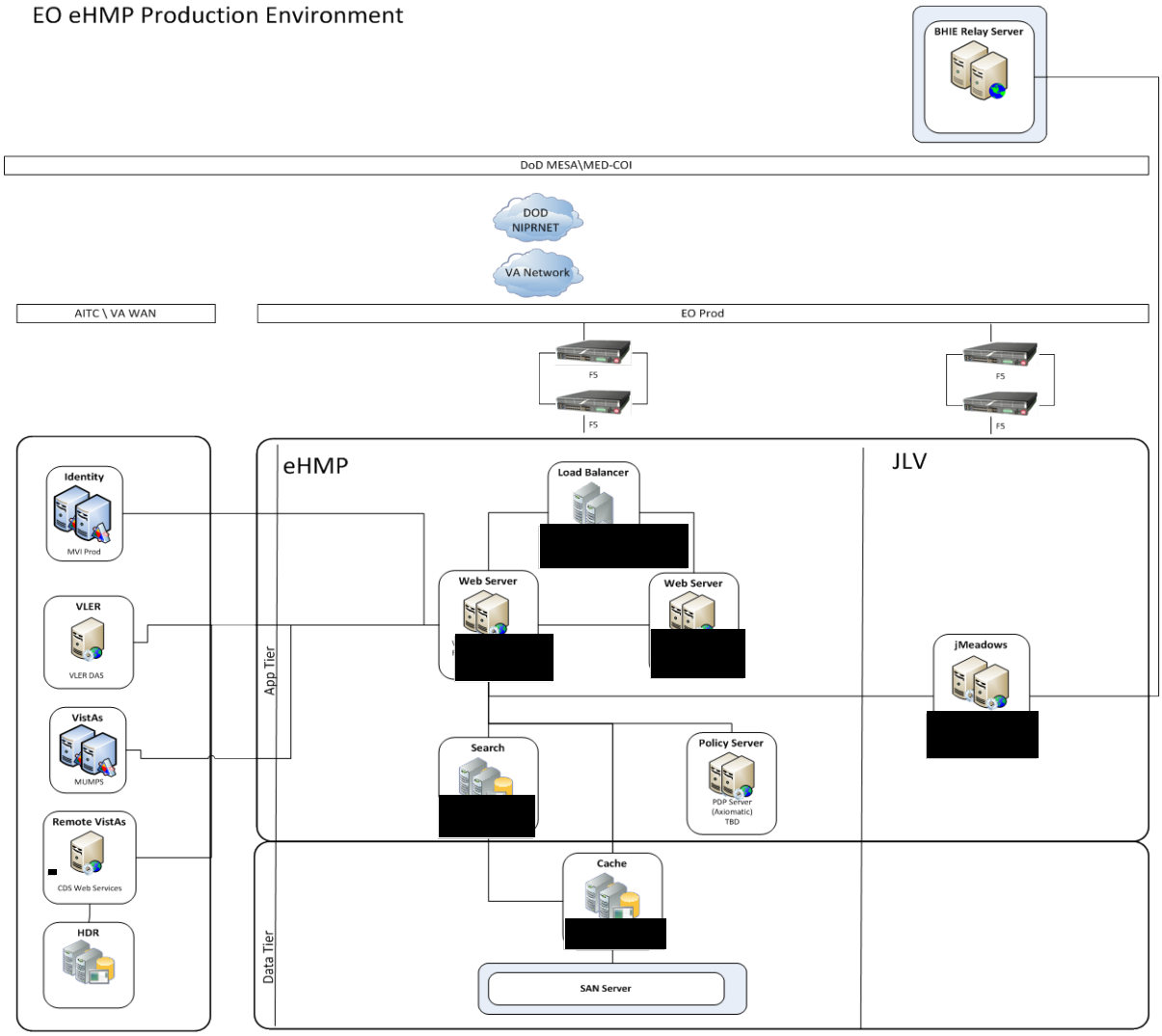


Figure 3-5 AITC Topology

3.3.4.1. Load Balance

The load balancer will consist of an Apache server running Red Hat Enterprise Linux (RHEL). Additional details on the load balancer will be provided with updates to this document.

3.3.4.2. Web Service

The web server FHIR API cluster will be supported by Jetty running RHEL 6.5. Additional details on the web server will be provided with updates to this document.

3.3.4.3. Cache

The cache server will consist of SOLR and JDS running on RHEL 6.5. Additional details on the cache server will be provided with updates to this document.

3.3.4.4. Synchronization

The synchronization cluster will consist of Jetty and AMQ running on RHEL 6.5. Additional details on the synchronization virtual machine will be provided in future updates to this document.

4. System Architecture

The development process and architecture are characteristic of the following key components:

- First, the infrastructure follows a principle of automation. This automation includes compile/build/packaging, infrastructure deployments onto local developer workstations, and infrastructure deployments in test and production.
- Second, this system architecture favors small, lightweight infrastructure (termed micro-services), rather than large, unmanageable infrastructure. This manifests itself in the use of small, autonomous, executable Java Archive (JAR) files used for web services and message handlers, rather than a traditional application server.

The system developed under the eHMP project will be designed to run in a cloud-based environment.

4.1. Hardware Architecture

The program will be set up within the VA's EO cloud environment. For details on the hardware architecture, please refer to section 6.1, Hardware Detailed Design.

4.2. Software Architecture

The software architecture is explained in section 6.2 Software Detailed Design.

4.3. Communications Architecture

All calls coming into the system will be over 443/HTTPS as either a web call from a browser to the UI server, or a REST API call to the API server. Communication from the UI/API servers will be over a secure channel to either the synchronization server or to the JDS server. Communication to remote systems, VistA/CDS/DoD Adapter, will be over HTTPS on various ports, mostly 443.

4.4. Third Party Software Components

There are a number of third party software components planned for this increment. Table 4-1 describes each component, its version, and its role.

Table 4-1 Third Party Components

Component	Version	Role
Apache Tomcat	7.0.47	Host for HMP UI.
Caché	2011	InterSystems Caché database to store the cached patient record.
Express.js	4.2.1	Node.js module to handle routing of HTTP calls
Java Virtual Machine (JVM)	1.7.0_25	The JVM is used to host the VistA Exchange API
Node.js	0.10.29	Runtime engine for JavaScript
Red Hat Enterprise Linux (RHEL)	6.5	Primary hosting software.

Component	Version	Role
SOLR	4.7.0	Creates full text indexes, allowing for text search across the entire patient record.

5. Data Design

5.1. Data Model

Based on the project team's review of the given options, it was recommended that HL7 FHIR and VPR JSON be selected as the canonical model family to be used both for storage as well as the external service-facing interface. HL7 FHIR has support of HL7 and, based on the analysis performed, will be able to support all of the required data types. It can also be extended to support any VA-specific data fields within the data types. In addition, it provides support for handling the various terminology code sets that will be used within the VA. The freely-available tools will give a jump-start to the development of eVPR, and will easily support the required technologies. eVPR JSON is the native JSON document model used by HMP, and will provide a performance-tuned eHMP CPE model. Together these models will enable the eHMP system to meet both the interoperability and currently defined UI requirements. Table 5-1 describes some of the trade-offs in selecting FHIR.

Table 5-1 FHIR Pros and Cons

FHIR Pros	FHIR Cons
• Published standard (by HL7) • Fast and easy to implement • Libraries freely available (reference implementation) • Standard is free for use • Support for needed technologies: JSON, XML, REST, etc. • Many resources already defined, including one for each required data type • Each resource can be easily extended with additional data fields • Extensions can be published to the broader community	• Currently the specification is in Draft Status for Trial Use • Full specification likely in 2015

5.2. Database Management System (DBMS) Files

JDS, also known as Caché Document Data Store, VPR Cache, eVPR or simply VPR, is a Caché MUMPS implementation of a NoSQL database with a RESTful API. In the HMP environment, it is installed on a Caché runtime which is cohosted on the HMP Java Middleware server. In the eHMP environment, the JDS server is often hosted separately from the HMP server. JDS is composed of a patient data store (eVPR/VPR or Enterprise Virtual Patient Record) and non-patient data store (ODS or Operational Data Store).

The mechanisms used for OPD and VPR collections differ in many key areas, as depicted in Table 5-2:

Table 5-2 OPD and OPR Mechanisms

	VPR	OPD
Scope	Patient-specific	Site-specific
Root endpoint	/vpr/<pid>/	/data/
UID format	urn:va:<collection>:<site>:<dfn>:<ien> urn:va:allergy:0CD5:16:106 or urn:va:patient:9E7A:229:229	urn:va:<collection>:<id> urn:va:syncstatus:10104V248233 or urn:va:user:9E7A:1
VPR Extract Routine	VPRDJ	VPREF

Table 5-3 Configuration

Access JDS menu in Caché session
; Start Caché session: ; sudo csession cache ; Change to JDS namespace ZNSPACE "JSONVPR" ; Start JDS Menu DO MENU^VPRJ

Table 5-4 Clearing JDS

Access JDS menu in Caché session
on ehmp VM sudo /usr/local/tomcat/default/bin/shutdown.sh # verify tomcat is stopped # on jds VM sudo csession cache ZNSPACE "JSONVPR" DO FULLRSET^VPRJ # on ehmp VM sudo /usr/local/tomcat/default/bin/startup.sh # wait until ODC sync is complete (~3 minutes) # There are several ways to check operational data status: # 1. In JDS: http:/[redacted]/data/find/syncstatus?filter=eq(forOperational,true) # 2. In HMP UI: refresh login page, and wait for loading bars to complete

Alternative way to execute FULLRSET on a local VM:

ssh vagrant@10.2.2.110 "echo -e \"d FULLRSET^VPRJ\" | sudo csession cache -U \"JSONVPR\""

5.2.1. Security

JDS implements check on TCP requests to ensure that only local applications can retrieve data as shown in Table 5-5.

Table 5-5 JDS Implements

LOOP^VPRJREQ
I '$$LCLHOST^VPRJRUT() W *-2 G LOOP ; reject & close port if not localhost

Where LCLHOST^VPRJRUT checks if the remote IP address of the TCP connection matches IPv4 and IPv6 versions of the loopback IP.

In the eHMP environment, this check is disabled when JDS is deployed to a separate VM.

5.2.2. Scalability

JDS supports horizontal scalability using Intersystems' Enterprise Cache Protocol (ECP), pictured in Figure 5-1. ECP is an application layer protocol on top of TCP, and functions by caching M Globals and user connections across multiple physical hosts. The database itself is still hosted on a single, vertically scaled server. ECP uses a Multiversion concurrency consistency[1] model. ECP supports up to 254 Application servers per Data Server.

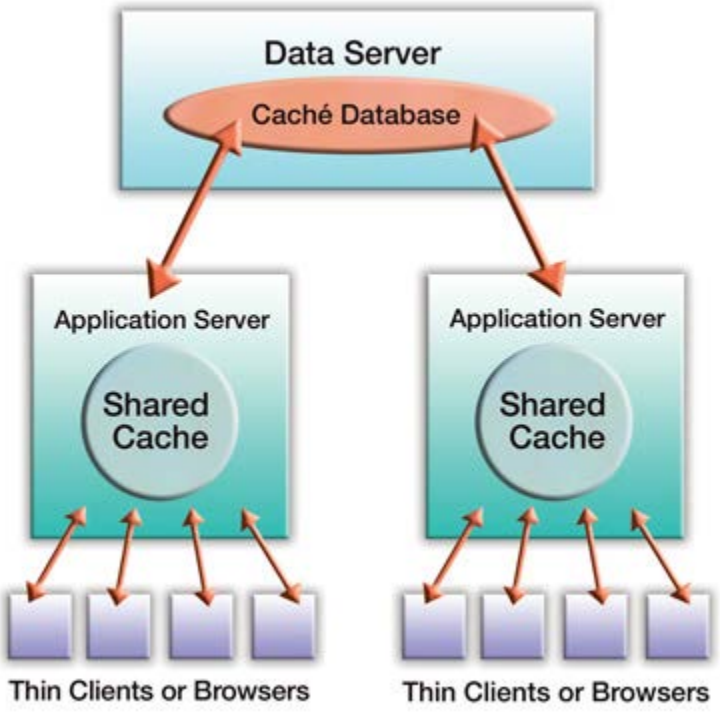


Figure 5-1 ECP Architecture

[1] http://docs.intersystems.com/cache20131/csp/docbook/DocBook.UI.Page.cls?KEY=GCDI_datacheck

5.2.3. JDS API Operations

JDS API Operations are detailed in Tables 5-6 through 5-9. Note: PUT and POST operations are synonymous in the JDS REST API.

Table 5-6 Patient Data VPR

URL Pattern	Description
PUT vpr	Add object
DELETE vpr	Delete the entire patient data store
GET vpr/all/count/{countName}	Get count of objects across patients
GET vpr/all/index/pid/pid	Get all PIDs
GET vpr/all/index/{indexName} GET vpr/all/index/{indexName}/{template}	Get objects by index
GET vpr/all/find/{collection} GET vpr/all/find/{collection}/{template}	Find objects using filters
DELETE vpr/all/collection/{collectionName}	Delete an entire collection
GET vpr/uid/{uid} GET vpr/uid/{uid}/{template}	Get an object by UID
DELETE vpr/uid/{uid}	Delete an object by UID
GET vpr/pid/{icndfn}	Get demographics by ICN
PUT vpr/{pid}	Add patient data
GET vpr/{pid}/index/{indexName} GET vpr/{pid}/index/{indexName}/{template}	Get objects by index
GET vpr/{pid}/last/{indexName} GET vpr/{pid}/last/{indexName}/{template}	Get most recent objects by index See GET vpr/{pid}/index/{indexName}
GET vpr/{pid}/find/{collection} GET vpr/{pid}/find/{collection}/{template}	Find collections using filters
GET vpr/{pid}/{uid} GET vpr/{pid}/{uid}/{template}	Get patient data by uid
GET vpr/{pid}/count/{countName}	Get count of object for patient.
GET vpr/{pid}	Get demographics.
GET vpr/{pid}/checksum/{system}	Get CRC checksum for all patient objects
DELETE vpr/{pid}/{uid} DELETE vpr/{pid}	Delete patient data
DELETE vpr/{pid}/collection/{collectionName}	Delete collection

Table 5-7 Non Patient Data (ODS)

URL Pattern	Description
PUT data PUT data/{collectionName}	Create data
GET data/{uid} GET data/{uid}/{template}	Get data by uid.
GET data/index/{indexName} GET data/index/{indexName}/{template}	Get objects by index
GET data/last/{indexName}	Get most recent objects by index
GET data/count/{countName}	Get count of objects
GET data/find/{collection} GET data/find/{collection}/{template}	Find collections by filter
GET data/all/count/{countName}	Get count of all objects
DELETE data/{uid}	Delete data by uid
DELETE data/collection/{collectionName}	Delete collection
DELETE data	Delete the entire non-patient data store
GET ping	Return operational status
GET version	Return build version
GET jds/logger/this	Get current log level
PUT jds/logger/this	Set log level

Table 5-8 Searching for Patients

URL Pattern	Notes
GET /data/index/pt-select-pid?range="{pid}"	
GET /data/index/pt-select-icn?range="{icn}"	
GET /data/index/pt-select-name?range="{name}"	
GET /data/index/pt-select-ssn?range="{ssn}"	
GET /data/index/pt-select-last4?range="{last4}"	last 4 SSN digits
GET /data/index/pt-select-last5?range="{last5}"	first letter of last name, last 4 SSN digits

Table 5-9 Examples

Operation	Description	Example Data
GET /ping	Verify that JDS is operational	{ "status": "running" }
GET /version	Return build information	{ "version": "0.7-S51-SNAPSHOT", "build": "20140116175604" }
GET /vpr/all/index/pid/pid	Return all PIDs	
GET /vpr/{PID}	Get patient	
PUT /vpr request body expand source	Add new patient	201 CREATED
PUT /vpr/{PID} request body expand source	Add object	201 CREATED
GET /vpr/{PID}/urn:va:med:93EF:-7:18068	Get object for patient	
DEL /vpr/{PID}/urn:va:med:93EF:-7:18068	Delete object for patient	200 OK
GET /vpr/all/find/med/uid?filter=like("products[].ingredientName","ASPIRIN%25")	Find medication by JSON-Path expression across all patients	
GET /vpr/{PID}/find/med?filter=like("products[].ingredientName","ASPIRIN%25")	Find medication detail by JSON Path expression from one patient	

Operation	Description	Example Data
GET /vpr/{PID}/index/every GET /vpr/{PID}/index/every?start=3&limit=3	Get all objects for a patient Get all objects for a patient (with paging)	
GET /vpr/{PID}/find/med	Get all meds for a patient	
GET /vpr/{PID}/last/med-ingredient-name?range=Metformin, Aspirin Tab GET /vpr/{PID}/last/med-ingredient-name?range=*M*	Query for last instances of items that have a drug ingredient matching either "Metformin" or "Aspirin Tab" Query for the last instances of items that have a drug ingredient with the letter M	
GET /vpr/{PID}/index/medication?order=qualifiedName asc	Get medications for patient ordered by name	
GET /vpr/all/find/med?filter=eq(facilityCode,DOD)	Get all DOD medications	
GET /vpr/all/find/med/summary?filter=eq(facilityCode,DOD)	Get all DOD medications summaries	
GET /vpr/all/count/collection	Get count of all collections	

5.2.4. JDS Features

Indexing as defined in VPRJPMX is displayed in Table 5-10 below.

Table 5-10 Indexing

Name		Description	VPR	ODS
list type indexes	index:attr (list)	These indexes are defined for the clinically relevant time in each domain collection.	✓	
tally type indexes	index:tally	These indexes keep track of totals (object per domain, lab results by name and vitals by name)	✓	✓
time type indexes	index:time	These indexes are defined for datetime fields of each domain.	✓	
cross-patient indexes	index:xattr	These indexes are defined for fields which are queried across patient (e.g., fullName, ssn/s).	✓	
attribute type indexes	index:attr	These indexes are defined for specific attributes which might be queried (e.g., med.orders[].provider.name/s, lab.lnccodes[]/s)	✓	✓
match type indexes	index:match	These indexes are currently disabled.		

5.2.4.1. Templates

Templates are defined in TLT^VPRJPMT. They allow VPR to build data fields based on computable data. For example, the summary field can be generated by concatenating other fields in the patient data. Currently, only summary and medication dosage are generated with templates.

5.2.5. Testing

Tests are annotated in the JDS source code with: ;; @TEST. Test examples are provided in Tables 5-11 through 5-14.

Table 5-11 Running Unit/Integration Tests

Run all unit tests
D ALL^VPRJT

EACH^VPRJT contains a list of available unit and integration tests.

Table 5-12 Simple Integration Test

Create test patient for simple integration tests
N I,TAGS F I=1:1:5 S TAGS(I)="MED"_I_"^VPRJTP02" D BLDPT^VPRJTX(.TAGS) W "PID="_VPRJTPID

Table 5-13 Template Integration Test

Create test patient for template integration tests
N I,DATALST F I=1:1:5 S DATALST(I)="DATA"_I_"^VPRJTP03" D BLDPT^VPRJTX(.DATALST) W "PID="_VPRJTPID

Table 5-14 Destroying Test Patients

Destroy test patients
; Destroy test patient D CLRPT^VPRJTX

5.2.6. Indexes

Indexes are a defined as a union of one or more collections, filtered by required fields and sorted. In the VX API, type indexes are synonymous with domain.

5.2.7. Required Fields

The variable X is the name of the domain, and caret-delimited list Y is the list of attributes that JDS requires in the JSON object as shown in Table 5-15.

Table 5-15 jds.ro Excerpt

jds.ro excerpt

```
ATTR(X) ; -- return list of attributes needed for collection X
 N Y S Y=""
 I X="vital"        S Y="^observed^typeName^result^"
 I X="problem"      S Y="^onset^problemText^statusName^"
 I X="allergy"      S Y="^entered^summary^"
 I X="order"        S Y="^start^name^statusName^"
 I X="treatment"    S Y="^start^name^statusName^"
 I X="med"          S Y="^overallstart^name^vaStatus^"
 I X="consult"      S Y="^startDate^typeName^statusName^"
 I X="procedure"    S Y="^dateTime^name^statusName^"
 I X="obs"          S Y="^observed^typeName^result^"
 I X="lab"          S Y="^observed^typeName^"
 I X="image"        S Y="^dateTime^name^statusName^"
 I X="surgery"      S Y="^dateTime^typeName^statusName^"
 I X="document"     S Y="^referenceDateTime^localTitle^statusName^"
 I X="mh"           S Y="^administeredDateTime^name^"
 I X="immunization" S Y="^administeredDateTime^name^"
 I X="pov"          S Y="^entered^name^"
 I X="skin"         S Y="^entered^name^result^"
 I X="exam"         S Y="^entered^name^result^"
 I X="cpt"          S Y="^entered^name^"
 I X="education"    S Y="^entered^name^result^"
 I X="factor"       S Y="^entered^name^"
 I X="appointment"  S Y="^dateTime^typeName^appointmentStatus^"
 I X="visit"        S Y="^dateTime^typeName^"
 I X="ptf"          S Y="^arrivalDateTime^icdCode^"
 Q Y
 ;
```

5.2.8. Adding Domains, Indexes, Etc.

To add a new domain, it must be added to the VPR collation in VPRJPMX, which contains the associations for indexing (tally, time, matching, cross-patient) and templates.

URLMAP^VPRJRSP contains the mapping of JDS REST URLs to M linetags.

5.3. Non-DBMS Files

There are no non-database system files.

6. Detailed Design

6.1. Hardware Detailed Design

The current design consists of multiple physical servers running ESXi that will host all Virtual Machines (VMs) for the system. The Load Balancer, Web UI and API servers will have 2 VPUs, 4 GBs RAM and 40 GBs of hard drive space total. The JDS Server will have 4 VPUs, 16 GBs RAM and 200 GBs of hard drive space. The Synchronization Server will have 2 VPUs, 4 GBs RAM and 60 GBs of hard drive space. All of these servers will be virtual and their resource allocations could change based on what is found with continued load testing.

Table 6-1 outlines the Detail VM Requirements in the Production environment:

Table 6-1 Production Detail VM requirements

VM	RAM (GB)	C:\ for OS	Space * (GB)	CPUs	VM Description/Use/DNS Required	Comments	Replicated Storage
1	16	65	0	4	RHEL 6.5	Apache frontend load balancer for web API servers cluster	32
2	16	65	0	4	RHEL 6.5	Apache frontend load balancer for web API servers cluster	32
3	16	65	0	4	RHEL 6.5	eHMP CPE UI Server	32
4	16	65	0	4	RHEL 6.5	eHMP CPE UI Server	32
5	16	65	0	4	RHEL 6.5	VistA Exchange (HMP) service, VistA Exchange Web Service API	32
6	16	65	0	4	RHEL 6.5	VistA Exchange (HMP) service, VistA Exchange Web Service API	32
7	16	65	0	4	RHEL 6.5	Search (SOLR) Server	32
8	16	65	0	4	RHEL 6.5	Search (SOLR) Server	32
9	16	65	0	4	RHEL 6.5	Policy (PDP) Server	64
10	16	65	0	4	RHEL 6.5	Policy (PDP) Server	64

VM	RAM (GB)	C:\ for OS	Space * (GB)	CPUs	VM Description/Use/DNS Required	Comments	Replicated Storage
11	32	65	5935	8	RHEL 6.5	JDS (Caché) Server	6000
12	32	65	5935	8	RHEL 6.5	JDS (Caché) Server	6000
13	16	65	67	4	RHEL 6.5	Centralized logging	132
14	16	65	67	4	RHEL 6.5	Monitoring, performance metrics, and status checks	132
Total	**256**	**910**	**12004**	**64**			**12648**

Table 6-2 describes the Detail VM Requirements in the Pre-Production environment:

Table 6-2 Pre-Production Detail VM Requirements

VM	RAM (GB)	C:\ for OS	Space * (GB)	CPUs	VM Description/Use/DNS Required	Comments
1	16	65	0	4	RHEL 6.5	Apache frontend load balancer for web API servers cluster
2	16	65	0	4	RHEL 6.5	Apache frontend load balancer for web API servers cluster
3	16	65	0	4	RHEL 6.5	eHMP CPE UI Server
4	16	65	0	4	RHEL 6.5	eHMP CPE UI Server
5	16	65	0	4	RHEL 6.5	VistA Exchange (HMP) service, VistA Exchange Web Service API
6	16	65	0	4	RHEL 6.5	VistA Exchange (HMP) service, VistA Exchange Web Service API
7	16	65	0	4	RHEL 6.5	Search (SOLR) Server
8	16	65	0	4	RHEL 6.5	Search (SOLR) Server
9	16	65	0	4	RHEL 6.5	Policy (PDP) Server
10	16	65	0	4	RHEL 6.5	Policy (PDP) Server
11	32	65	5935	8	RHEL 6.5	JDS (Caché) Server
12	32	65	5935	8	RHEL 6.5	JDS (Caché) Server
13	16	65	67	4	RHEL 6.5	Centralized logging
14	16	65	67	4	RHEL 6.5	Monitoring, performance metrics, and status checks
Total	256	910	12004	64		

Table 6-3 depicts the Detail VM Requirements for Shared Servers:

Table 6-3 Shared Servers Detail VM Requirements

VM	RAM (GB)	C:\ for OS	Space * (GB)	CPUs	VM Description/Use/DNS Required	Comments
1	8	65	67	4	RHEL 6.3	Encrypted CM repository for binaries and configuration files
2	8	60	0	4	RHEL 6.3	Server from which deployments are executed
Total	16	125	67	8		

Table 6-4 lays out the Detail VM Requirements for Extra Servers:

Table 6-4 Extra Servers Detail VM Requirements

VM	RAM (GB)	C:\ for OS	Space * (GB)	CPUs	VM Description/Use/DNS Required	Comments
1	8	65	67	2		
2	8	65	67	2		
Total	16	130	134	4		

6.2. Software Detailed Design

Figure 6-1 depicts the various components of the eHMP system architecture.

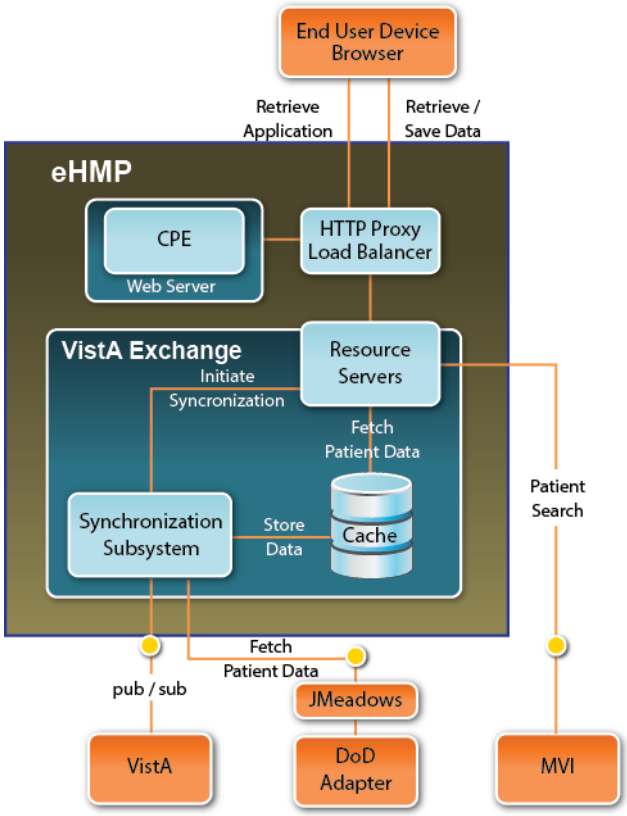


Figure 6-1 eHMP System Architecture

6.2.1. eHMP Software Development Kit (SDK)

With release 1.1, eHMP introduces a web application based upon a custom software development kit. The SDK is explained in this section.

6.2.1.1. Goal of the Web Application Architecture

The VistA Core/eHMP program's functional goal is to create a web application for point-of-care clinical use that contains, at a minimum, the following functionality:

- Meet criteria for meaningful use ONC 2014 EHR certification (PSI 5-PSI 8);
- JLV functionality of patient record, showing several domains at once (PSI 4);
- Advanced text search across the patient record (PSI 4-5);
- Medication Review (PSI 4-5);
- Ability to document a clinical encounter (PSI 5-8);
- Ability to edit a patient record (add allergies …) (PSI 5-PSI 8);
- Ability to place orders (PSI 5-PSI 8);

- Ability to recommend interventions based upon a patient record derived from clinical practices (PSI 5-PSI 8).

In addition, there are several non-functional criteria driving the architecture and development approach:

- In order to achieve meaningful use criteria by September 2015, functionality across several teams in parallel must be created. Collaboration will involve teams with members who are both collocated and remote, but are part of the same Release Train [http://scaledagileframework.com/agile-release-train/]. In addition, teams with infrequent communication who are working from different projects, programs, and/or contracts will collaborate. In order to build a single application rather than a patchwork quilt of an application, a set of guidelines are needed that will address: 1) adding extremely decoupled functionalities simultaneously, and 2) staying consistent with functional and architectural approaches. Further, the artifacts generated from different development teams must be safe from conflicting action which can be caused by global variables and global namespaces associated with immature JavaScript development. These artifacts are referred to as 'applets' and support listed principles, such as the ability to develop applets in isolation without applet-to-applet dependencies or by modifying the underlying framework.
- Incremental functionality must be testable in a manner to provide fast, consistent feedback to the development team. Testing of applets must be automated and should be able to be done in isolation.
- The certification process for VA applications sets a high bar. It includes ensuring 508 compliance (standard look and feel toward usability), and technical standardization. In order to ensure high quality releases through the certification process, it will be important to set restrictions and guidelines toward adding additional functionality; common UI controls will be selected and will be used throughout development.
- It is expected that a group of teams will develop a set of functionality, then move on to another area. Teams must be fluid enough to move to different functional areas of the application. To ensure that teams can work fluidly in different areas of the application, it will be important to standardize the technology and development approach; it is not feasible for each team to make their own technical choices when adding incremental functionality to the web application. A set of technologies and patterns will be selected and identified as standard.
- To add non-trivial, incremental functionality, modifications will need to be made to both the front-end (web application) and the back-end (resource server). Therefore, a team must possess the skills to develop holistically, rather than being organized around specific skill sets. The use of node.js will assure the use of the same technology on both the client and the server.
- The application is expected to be a vital part of the VA's ecosystem for a long time. Therefore, it must provide a mechanism to add incremental functionality and have an easy deploy strategy. The use of applets will allow for teams to develop new functionality in isolation and have their own release cycle.
- Although many teams will develop applications, as the program moves toward a sustainment model, maintenance will move to a much smaller team. It will be important

to standardized the technology and development approach in order to ensure that a small number of developers are able to maintain the application. As such, it is not feasible for each team to make their own technical choices in how to add incremental functionality to the web application. Therefore, a set of technologies and patterns will be selected and identified as the standard in order to reduce the number of technologies required to support the application.

- The application will be utilized by many end users as the primary UI for many hours a day. A highly responsive application is vital to garner end user satisfaction. The architecture takes into account minimization of perceived latency. Putting the behavior logic in the client rather than the server will provide a faster UI.
- The application has a complex relationship with other parts of the VA. It will be important to provide capability to fine-tune deployment for maximum scalability. These realities favor applications and resources to be decoupled, and favors microservices rather than single WAR file deployment.

6.2.1.2. SDK to Help to Achieve Goals

The eHMP team will utilize an SDK to make development more efficient. The SDK is comprised of:

- An Application Development Kit (ADK) to drive development of the client-side web application; and,
- A Resource Development Kit (RDK) to drive the development of service-side resources (web services) to support the web application.

Figure 6-2 below depicts the VistACore SDK.

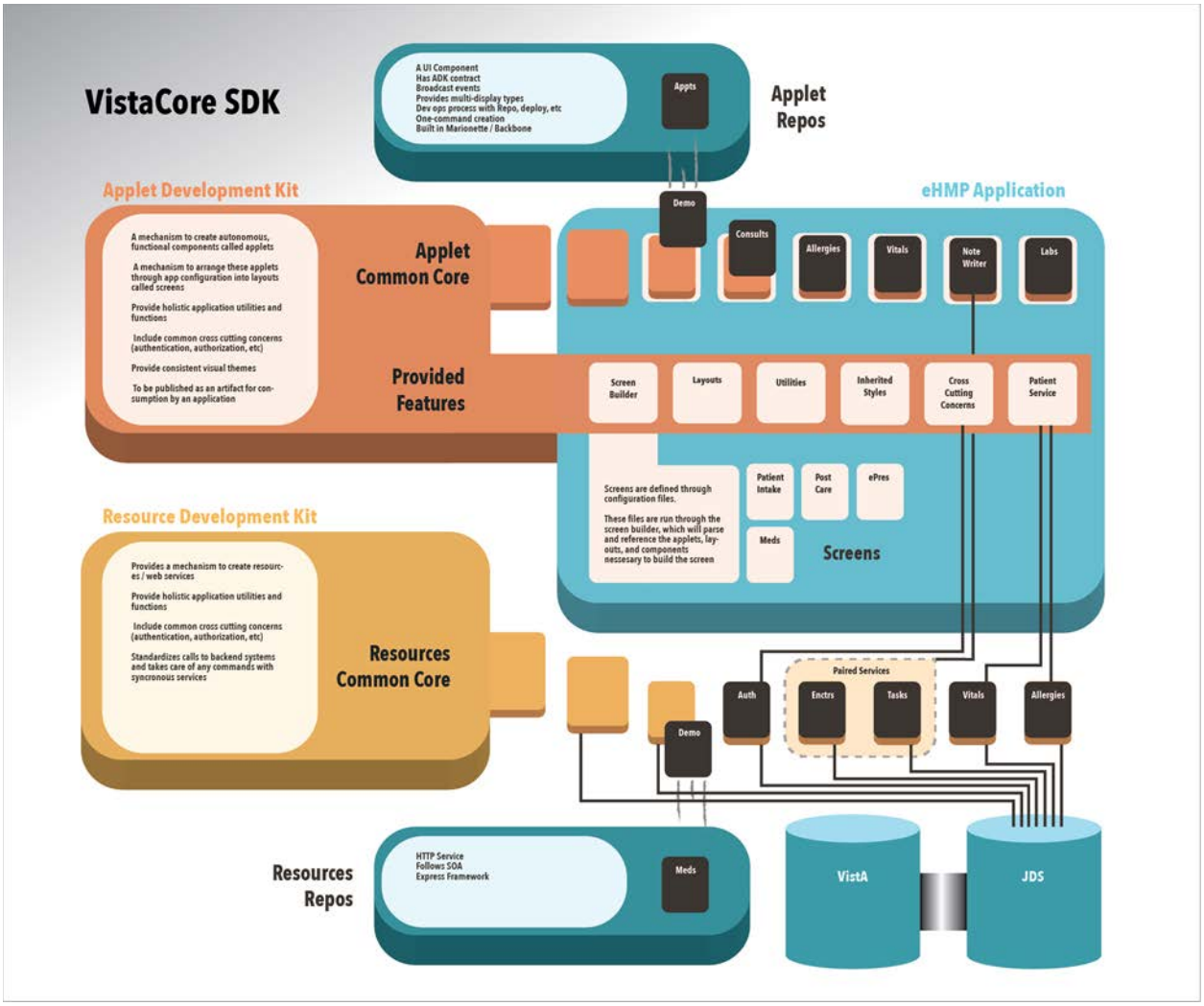


Figure 6-2 VistACore SDK

6.2.1.3. ADK Features/Anatomy

The breakdown of the ADK anatomy is as follows:

- An **applet** is a single unit of visual functionality. Examples of applets include a list of allergies, the ability to enter a new allergy, and the ability to place a new order.
- A **screen** is a defined collection of applets and UI components comprising the application chrome.
- A **layout** is an abstract arrangement of applets. A screen typically chooses from a set of ADK pre-defined layouts.
- An **application** is a collection of screens.

Figure 6-3 provides an example of a screen utilizing a 3x3 layout to display nine applets:



Figure 6-3 Example of a 3x3 Screen Layout

6.2.1.4. Roles

- Applets are created by developers, who can often create an applet without consideration to the screen on which the applet will run. Applet developers utilize JavaScript and Marionette.
- Screen designers choose how to arrange a set of applets, and use configuration files.

6.2.1.5. Technical Information

- The SDK is built using JavaScript, backbone.js, marionette, and bootstrap.
- The resulting application is an SPA, static HTML from web server (not dynamic web pages such as JSP, PHP, etc. No J2EE web server.)

6.2.1.6. ADK Responsibilities for Applet Developers

- Provide a mechanism for applet developers to discover the current patient;
- Provide a mechanism for applet developers to fetch patient data from VistA exchange. They can use a canonical model (based on VPR), or provide a custom view model;
- Provide a mechanism for applet developers to bind data to backbone views, and provide templating;
- Provide a mechanism for applet developers to choose pre-selected display paradigms (tableview, etc.) and UI controls. UI style will be set by the application;
- Applet developers will create three different size views to support both responsive and adaptive design;
- CI pipeline/DevOps provides for compiling the applets, packing, running tests, and publishing to the artifact repository.

The applet developer is responsible for producing a marionette view. This will often fall into the following patterns:

- An HTML template for a row using handlebars;
- The creation of a backbone view-model;
- Registering these objects with ADK;
- Showing examples for allergy templates, both the short and complex versions.

6.2.1.7. ADK Responsibilities for Screen Designers:

- Provide a mechanism for screen designers to create screens, choose from predefined layouts, and assign applets to regions.

6.2.1.8. ADK Responsibilities for Application Designers:

- SDK provides the runtime UI shell for the application. It displays the current patient, the current user, and provides navigation.
- Provide a mechanism for the application designer to choose the screens that are part of the application;
- CI pipeline/DevOps will pull applets from the artifact repository, compile, pack, test, and publish them to the artifact repository.

6.2.1.9. RDK Features/Anatomy

The RDK anatomy breaks down to the following:

- A **resource** is implemented as a single web service (e.g., allergies, or save allergies). In JAX-RS, a single method that gets the @GET annotation;
- A **resource family** is a collection of resources that are generally deployed together. In JAX-RS, a class;
- A **resource server** is a deployable unit, including a set of resource families and specific configurations. In JAX-RS, a WAR file.

6.2.1.10. Roles

- A resource developer develops the resources and resource families in JavaScript.
- A deployment engineer chooses how to deploy a set of resource families as a resource server, and develops in JavaScript.

6.2.1.11. Technical Information

- The RDK is developed using JavaScript and express.js;
- The RDK is deployed to node.js, and relies on npm for package management.

6.2.1.12. RDK Responsibilities for Resource Developers:

- Provide the configuration of the resource family;
- Specify the relative mount path and method using standard express signature function (request, response, next) for each resource;

- Specify resource characteristics (sensitive, audit information);
- A request provides resource information about the request: URL, path parameters, query parameters, header, current user, access to RDK utility methods, access to current resource server, configuration information, logger;
- A response provides resource ability to send response: status code, JSON or text body, media information;
- The RDK provides resources with handles to common external systems, including JDS, VistA(s), and SOLR;
- CI pipeline/DevOps of resources, including compiling, packing, testing, and publishing to artifact repository.

6.2.1.13. RDK Responsibilities for Deployment Engineer

When creating a resource server the deployment engineer is responsible for:

- Grouping the resources into a logical deployment unit (resource server).
- Specifying how many processes use cluster/fork;
- Deploying the Resource Server behind a reverse proxy for load balancing
- Enabling authentication, authorization/PEP, audit;
- Specifying logging rules;
- Specifying configuration;
- Specifying resource caching rules;
- RDK routes calls to resources based upon URL and media type (content-negotiation)
- Providing a health check (both binary and discrete information) based upon each resource registered, and specifying additional health check rules;
- CI pipeline/DevOps pulls resources from the artifact repository, compiles, packs, tests, and publishes them to the artifact repository.

6.2.1.14. Other responsibilities

The RDK also provides:

- Centralized dependency management; and,
- Automatic creation of the resource directory.

6.2.1.15. ADK Details - SPA

The implementation of the web application created through the SDK is a Single-Page Application (SPA), that utilizes a set of server-side resources (RESTful web services).

The HTML/JavaScript/CSS for the application is returned as static HTML from a web server, specifically Apache httpd[2]. The client-side application utilizes the resource server (web services)

[2] Apache Web Server: http://httpd.apache.org

to execute various data-driven activities, such as reading the patient record, updating the patient record, documenting a patient encounter, and/or ordering medications.

Note that this pattern is similar to a traditional client-server or native mobile application:

- The user downloads the application upon first use;
- The application itself contains behavior but not the data;
- The application reaches back out to a server for read/write activities.

The term single-page application can be misleading, in that it may suggest that there is a single view for the entire application. The term page refers to the handling of the lifecycle of the page within the browser. Rather than recycling the entire page, elements on the Document Object Model (DOM) are modified as needed.

Although there are differing approaches for the implementation of a SPA, ours will be to have the static assets (HTML/JavaScript/CSS) decoupled from the data-centric resource services, upon which they rely. The static assets will be available from a web server (like Apache), independent from the resource server (running within JVM and/or node.js).

6.2.1.16. Development Lifecycle

Figures 6-4 and 6-5 address the applet and application development lifecycles.

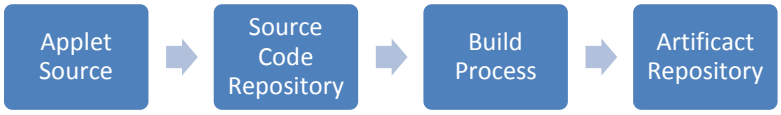


Figure 6-4 Applet Development Lifecycle

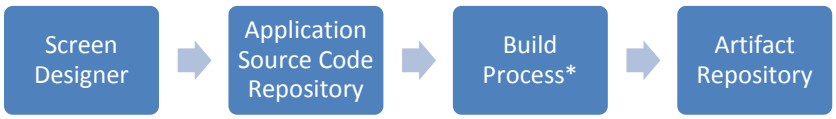


Figure 6-5 Application Development Lifecycle

*NOTE: The application build process pulls from the applet artifact repository.

- Deployment process deploys local source code (applets, screens, application) to private cloud (local VMs);
- Deploys application from artifact repository to test/integration environment;
- Deploys specified official application to preproduction/production environment.

6.2.1.17. Applet Development

The majority of web developers working within the creation of the eHMP UI will spend their time developing applets. An applet (the incremental user functionality), is a set of HTML/CSS/JavaScript, and runs within the ADK.

Applet development begins with using a DevOps service to initialize the applet. Currently, all applet development will be done in a single eHMP UI application repository. The result of this service includes: 1) a new Git repository for the applet, seeded based upon an applet template; and, 2) CI (Jenkins) jobs responsible for reacting to check-in, compiling the HTML/CSS/JavaScript, running unit and acceptance tests, and publishing the resulting artifact to an artifact repository.

The resulting artifact of an applet will likely include two versions - a development version optimized for debugging (not minified), and a version optimized for production (minified).

The goal of the ADK is to provide the guidelines, constraints, and technology to ensure that teams can efficiently create incremental functionality without requiring unsalable close collaboration between development teams. Yes, collaboration is a good thing. However, many teams modifying the same code or having to coordinate routine changes is not scalable to an ecosystem of many parallel functional development areas.

The ADK includes:

- Provide mechanism for applet developers to discover the current patient;
- Provide mechanism for applet developers to fetch patient data from VistA exchange, can use canonical model (based on VPR) or provide custom view model;

- Provide mechanism for applet developers to bind data to backbone views, provide templating;
- Provide mechanism for applet developers to choose from preselected display paradigms (tableview, etc.) and UI controls. UI style set by application;
- Applet developers create three different size views to support responsive/adaptive design;
- CI pipeline/DevOps for compiling applet, packing, running tests, publishing to artifact repository;
- Applet Developer responsible for producing a marionette view. The view returned by the applet can be any type of marionette view, CollectionView, CompositeView, ItemView, or LayoutView. A CompositeView and LayoutView can include additional regions that applet developers have control of to load other sub views. This often will fall into patterns as depicted in Tables 6-2 through 6-4;
- HTML template for a row using handlebars;
- Creation of backbone view-model;
- Registering these with ADK.

Example code to create allergy applet:

Table 6-5 Applet Example Code

```
var dependencies = [
  backbone,
  marionette,
  underscore,
  ADKApp,
  hbs!main/applets/allergies/table,
  hbs!main/applets/allergies/row,
  api/AllergyFactory
];
define(dependencies, exportAMD);
function exportAMD(Backbone, Marionette, _, ADKApp, AllergyViewTemplate,
AllergyItemTemplate, AllergyFactory) {
  var AllergyItemView = Backbone.Marionette.ItemView.extend({
    tagName: tr,
    template: AllergyItemTemplate
  });
  var AllergyCompositeView = Backbone.Marionette.CompositeView.extend({
    childView: AllergyItemView,
    childViewContainer: tbody,
    template: AllergyViewTemplate,
    className: panel panel-info,
    collectionEvent: {
      sync: render
    }
  });
  var AppletLayoutView = Backbone.Marionette.LayoutView.extend({
    template: _.template('<div id=applet-main></div>'),
    regions: {
```

```
        appletMain: #applet-main
      }
    });
    var rootView;

    var applet = {
      id: allergies,
      getRootView: function() {
        rootView = new AppletLayoutView();
        return rootView;
      },
      onDisplay: function() {
        var viewModel = {
          parse: function(response) {
            if (response.reactions) {
              response.reaction = response.reactions[0].name;
            }

            return response;
          }
        };
        var allergies = AllergyFactory.initCollection(patient, viewModel);
        var allergyCompositeView = new AllergyCompositeView({
          collection: allergies
        });
        rootView.appletMain.show(allergyCompositeView);
      }
    };
    return applet;
  }
```

Table 6-6 Applet Template Example Code

```
 <div class=panel-heading>
  <h3 class=panel-title>Allergies</h3>
</div>
<div class=panel-body id=appletContent>
  <table class=table table-condensed table-striped table-hover>
    <thread>
      <tr>
        <th>Summary</th>
        <th>Reaction</th>
        <th>Facility</th>
      </tr>
    </thread>
    <tbody>
    </tbody>
  </table>
</div>
```

Table 6-7 Example of Row Template Code

```
<td>{{summary}}</td><td>{{reaction}}</td><td>{{facilityName}}</td>
```

Much of the boilerplate of the applet has been reduced utilizing an ADK convenience method as shown in Table 6-5:

Table 6-8 Example of Simplified Applet Code

```
var dependencies = [
  main/ADK,
  hbs!main/applets/allergies/table,
  hbs!main/applets/allergies/row
];
define(dependencies, exportAMD);
function exportAMD(ADK, allergyTableTemplate, allergyRowTemplate) {
  var viewModel = {
    parse: function(response) {
      // this is handling of canonical model to view model
      if (response.reactions) {
        response.reaction = response.reactions[0].name;
      }
      return response;
    }
  };
  var appletDefinition = {
    appletId: allergies,
    resource: allergies,
    viewModel: viewModel,
    tableTemplate: allergyTableTemplate,
    rowTemplate: allergyRowTemplate
  };
  return ADK.createSimpleApplet(appletDefinition);
}
```

6.2.1.18. Accessing Patient Record

Applet developers can access the patient record using the ADK.PatientRecordService. The fetchCollection method returns a Backbone Collection object.

```
var fetchOptions = {};
fetchOptions.patient = adkApp.session.patient; //optional, defaults to
current patient context
fetchOptions.resourceTitle = 'patient-record-med'; //resource title from
resource directory
fetchOptions.viewModel = viewModel; //optional override of default viewModel
fetchOptions.criteria = criteria; //optional criteria object gets converted
to query string parameters on resource call
medicationCollection =
ADK.PatientRecordService.fetchCollection(fetchOptions);
```

6.2.1.19. RDK Details

6.2.1.19.1. Resource Server

A resource server represents the information and processing that occurs in the server tier of the web application. The resources include the patients clinical record, an order, a user, the current patient, and a list of allergens known to the VA. The resource server includes web services that allow for the retrieval of a patient's clinical record, the edit of a patient's record, and the ability to place an order, etc.

The existing patient record resource server is written as a microservice. The implementation is in Java, using a lightweight embedded HTTP listener. The approach of the microservice allows for discrete instances to be independently deployed, optimized, and load balanced. This is contrary to an approach of deploying several logical services as a single WAR file and deployed to a clustered app server. With the single WAR file deployed to cluster, individual optimization becomes more challenging.

Additional resource web services will be written in JavaScript, using node.js. The concept of a microservice will be used. Each additional web service is to be written as a Node Packet Manager (npm) module. The modules (implementation of the web service) are then packaged in a logically decomposed microservice, resulting in a deployable, runnable artifact.

As a team developing an applet recognizes that there is a need for an additional resource web service (e.g., the need to discoverer the list of clinics within a VA facility) the team will then create an additional resource module.

Like the applet, the resource module development begins with the usage of the DevOps service to initialize the module. The result of this service includes a new Git repository for the module, seeded and based upon a module template; and CI (Jenkins) jobs responsible for reacting to check-in, running unit and acceptance tests, and publishing the resulting artifact to an artifact repository.

All new resources services are to be written in node.js. As such, there should be an expectation to migrate the existing patient record web service to node.js.

6.2.1.20. RDK

The cross-cutting features provided by the RDK include:

- Authentication
- Authorization
- Web service/resource server decomposition, registration
- Logging
- Audit
- Health check
- Cluster/fork
- Caching
- Content negotiation
- Data formatters

- Handle to JDS
- Handle to VistA RPCs

6.2.1.21. Resource Developer Responsibility

The responsibilities of the Resource Developer include:

1. Create a method to represent a single endpoint. This method supports the express format of function(request, response, next).

Common items to access on request include:

- param: includes any path parameters, query parameters, http header parameters. See param documentation.
- app.config: application configuration.
- logger: access to RDK logger.
- audit: if this resource needs to provide any custom audit information, resource developer can specify by adding to req.audit.

The resource must send a response using response.send, typically response.send(myObject). See response documentation.

2. Create a resource configuration. This is invoked by the resource server during startup. This is done by specifying an exported method of getResourceConfig. The resource config is an array of objects, with each object containing the following, as outlined in Table 6-9:

Table 6-9 Resource Config

```
exports.getResourceConfig = function() {
  return [{
    // optional: name of resource
    name: '',
    // optional: relative path for this resource
    path: '',
    // specify http action (get, put, post, delete, head), followed by
resource method with pattern function(req, res, next)
    get: performPatientSearch,
    // optional: provide method for health check, which returns true if
healthy or false if not healthy
    healthcheck: function() {
      return true;
    },
    // provide a list of common middleware/interceptors
    // note that method for specifying these likely to change in next
internal release
    interceptors: ['audit', 'authentication', 'pep', 'metrics',
'operationalDataCheck']
  }];
};
```

6.2.2. Web Server

The role of the web server is to provide the static files that represent the web application, and present the application to the user. These static files contain the HTML, JavaScript, and CSS to drive the behavior of the application. No data is provided from this web server.

The web server is implemented with Apache Web Server.[3]

6.2.3. HTTP Proxy

The roles of the HTTP Proxy are:

- Act as a network buffer between the public client and the internal Web Service API instance;
- Perform Uniform Resource Locator (URL) routing to Web Service API instances;
- Load balance calls to appropriate Web Service API instances.

The HTTP proxy is implemented with Apache Web Server.

When an HTTP request is performed, the HTTP Proxy interrogates the request, and based upon the URL, routes it to the appropriate server. These requests include:

- Calls from a browser to access the web application static assets;
- Calls from browser-based web application to access data resources;
- Calls from any other client to access data resources.

Under Release 1.1, the proxy server has the additional responsibility of validating the client certificate, presented from the client application. This will be used to ensure that the user has a Personal Identity and Verification (PIV) card.

6.2.4. Web Service API

The VistA Exchange Web Service API allows for a consumer to retrieve a record for a given patient. The Web Service API provides a VPR and FHIR web service that allows for the retrieval of patient records using the patient's ICN. Upon request for retrieval, a request is made to all VistA instances to initiate a patient subscription, if the patient data has not previously been synchronized. This subscription is initiated by invoking the HMP VPR pub/sub RPCs. Within a few seconds, the data is available within the JDS cache. Once it is available, the Web Service API transforms the data into the FHIR, and returns it to the user. Table 6-5 depicts the technology stack for the Web Service API.

Table 6-10 Installed Software

Technology	Version	Role
Apache Tomcat	7.0.47	Host for HMP UI
Java Runtime Environment	1.7.0_25	JVM

[3] Apache Web Server: http://httpd.apache.org

6.2.4.1. Implementation of the Web Service

Web services are developed using a set of Java classes. These web services are compiled into an executable JAR file that contains an embedded web server (Jetty). This allows for the JAR to be executed rather than deployed to a traditional app server. To scale the web service, an additional microprocess is executed. The microprocesses are controlled via a process watcher to ensure that they are started, and restarted should they shut down.

Under release 1.1, the Java web services will be deprecated in favor of the SDK implementation of a resource server. These SDK-based resource servers expose the data to be used by the web application, ranging from patient data, lookup lists of allergens, and interventions recommended by a clinical decision support system.

If the scaling characteristics between the Patient Resource (which has a patient correlation/MVI dependency) and the Patient Record Resource (which has a cache and Patient Sync Subsystem dependency) require different scaling concerns, these can be deployed to separate virtual machines. This will ensure that one resource does not negatively affect the other from a performance and scalability perspective.

The SDK resources utilize node.js and are written in JavaScript.

Table 6-11 Resource Server Technology Stack

Technology	Version	Role
Node.js	0.10.29	JavaScript runtime engine
Express.js	4.1.2	Node.js module used to handle routing of HTTP calls

6.2.4.2. Authentication

User authentication includes several items.

First, the user can utilize their access and verify code with the selection of a site (VistA instance) in order to log on. For Release 1.0, the web services accept an access/verify code plus site as part of the HTTP header, consistent with HTTP Basic Authentication. The web service validates this against the VistA instance using an RPC call. The HTTP header is passed on all calls.

For release 1.1, the introduction of a token-based scheme works as depicted in Figure 6-6:

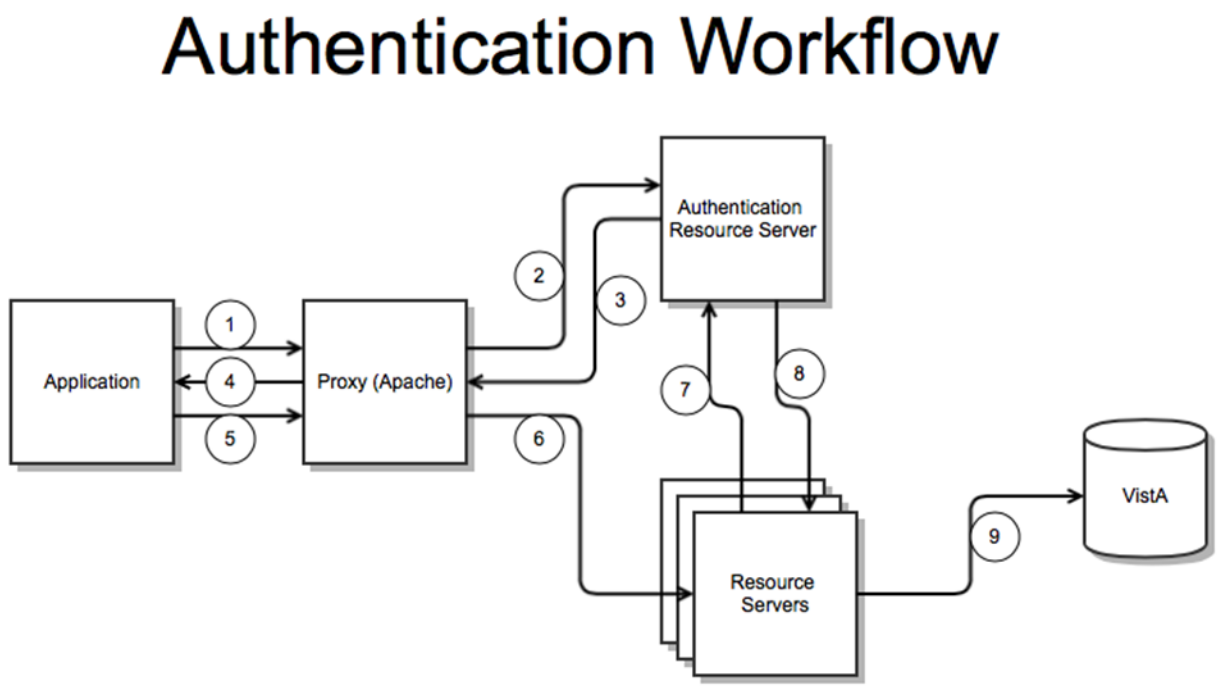


Figure 6-6 Authentication Workflow

1. Application presents access/verify code (via HTTP Basic authentication header) to authentication resource.
2. Proxy (Apache) routes to authentication resource server. Authentication resource validates the access/verify code against VistA. If valid, creates storage (assume memory) for this session keyed by token, with values for any user information that needed, including the access/verify code, the user's name, and anything else needed for authorization downstream.
3. The Authentication Resource Server returns status 200 with token in a header value.
4. The proxy returns to the application, and the ADK holds onto the token.
5. The application attempts to access patient data by performing a call to the resource server. The call is sent to proxy. SDK attaches a token to the header.
6. The proxy routes call to appropriate resource server.
7. Resource Server authentication interceptor validates the token by making a call to the resource server endpoint (passing token).
8. Authentication server returns an indicator to state that token is or is not valid. If valid, it also returns information known about user, such as the access/verify code
9. Resource Server establishes connection to VistA using the access/verify code.

An alternate view of this workflow is depicted in Figure 6-7:

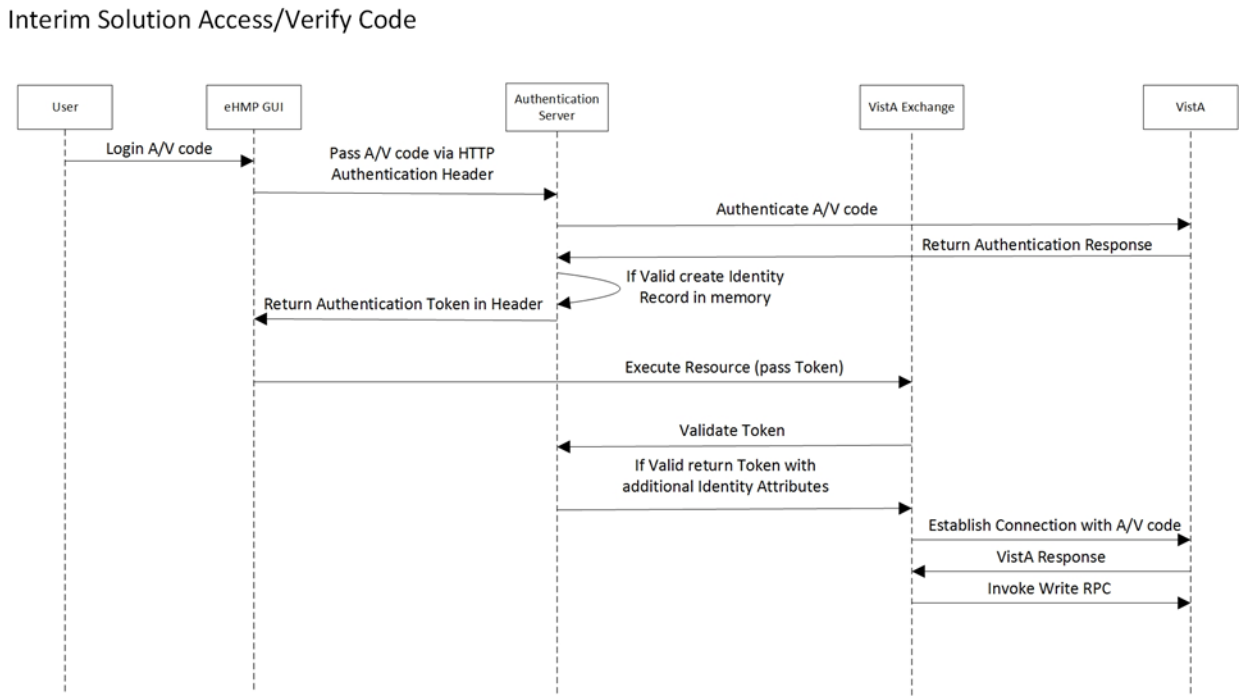


Figure 6-7 Interim Solution Access/Verify Code

A longer-term approach relies upon IAM SSOi to provide mapping from the user's active directory identity to their VistA identity. See Figure 6-8 below:

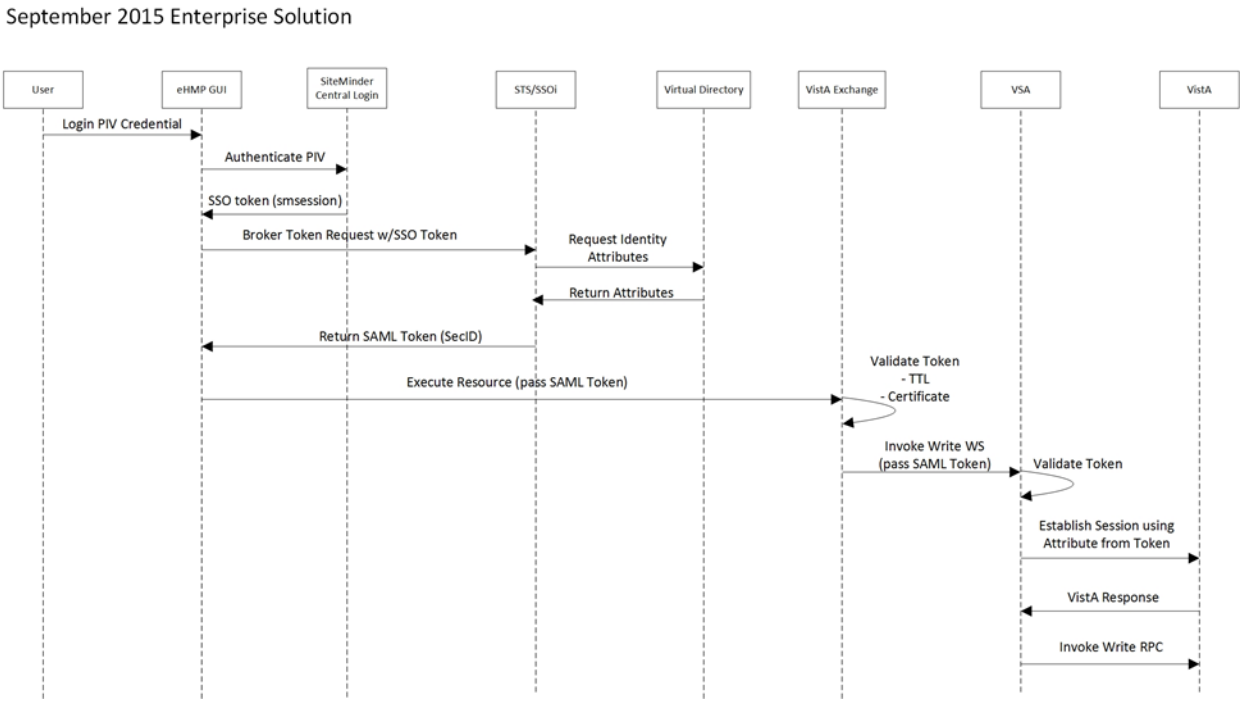


Figure 6-8 September 2015 Enterprise Solution

In addition, multi-factor user authentication will be employed as PIV cards are implemented throughout the VA staff/user population. It is anticipated that a single-factor user ID and password may be required at the initial launch of the eHMP system until the entire VA staff/user

population are issued PIV cards. Authentication of the Web Service API will be driven by the user's access/verify code. When the user navigates to the web application, the web application static content (HTML/JavaScript/CSS) is accessed from the web server. The access to this content is not restricted to an authenticated user because, at this point, there is no data.

6.2.4.3. Access Control

Once a user is identified and authenticated, his or her access to system resources and application functionality will be predicated upon role-based access controls. When the web application attempts to access a protected resource such as patient data, the Web Service API's Policy Enforcement Point (PEP) gathers the information about the request (the user, the subject, the resource, and the action) and provides this information to the Policy Decision Point (PDP) via an Extensible Access Control Markup Language (XACML) request. Note that the PDP is implemented using Axiomatics.

The PDP then checks the request against registered rules and provides a permit or deny response, potentially with a set of obligations that the PEP must perform to allow access. The rules that will be implemented as part of this increment are to ensure that the user has a valid role such as a CPRS User or CAPRI User (flag within VistA). A CPRS User will have full read/write capabilities, while a CAPRI User will have read only capabilities. A check is also made to determine if the patient is identified as a sensitive patient at any of the VistA sites. If the patient is flagged as sensitive, then the obligation to break the glass is returned to the web application, prompting the user that they will be audited if they access the data. If the patient is flagged as sensitive because a user is attempting to access their own record, access to the patient record is denied without the option to break the glass.

A CPRS User is defined as a VistA user that has at least one CPRS TAB ACCESS setting. VistA user management includes two options for CPRS TAB ACCESS—CPRS GUI Core tabs (COR) and Reports tab (RPT). Access will be denied to any resource requiring CPRS User access if a user does not have either of these settings enabled. The Vista RPC used by CPRS to validate this setting is ORWU USERINFO. The RPC response includes the COR and RPT tab settings in fields 22 and 23 of the response, as illustrated in Figure 6-9 below.

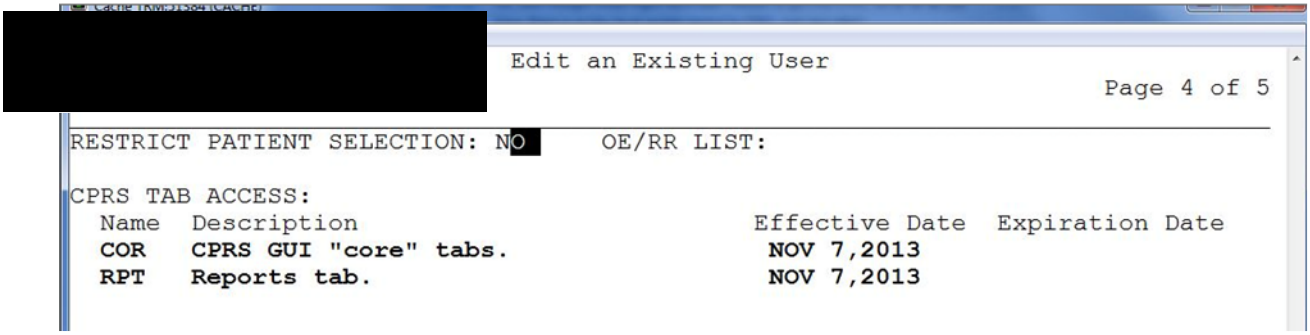


Figure 6-9 CPRS Tab Access

Patient sensitivity rules are based on a number of attributes that will continue to grow as new patient access control policies are defined and implemented. Many of these attributes are not centralized, and exist in each VistA host. In order to fully leverage the benefit of an externalized access control solution and provide centralized access control policies for centralized patient

records, all user and resource (e.g., patient) attributes required for defining access control rules and policies will need to be centralized.

The current HMP solution and the initial implementation of Axiomatics with the FHIR API will be based on access control attributes retrieved from a single VistA host, using the existing patient sensitivity RPC interface. The VistA RPC used for patient sensitivity checks is VPRCRPC RPC with a command parameter of isPatientSensitive. Figure 6-8 depicts the command response, which includes attributes of sensitive, logAccess, mayAccess, and text as defined below and shown in the following figure:

- Sensitive: When exists, indicates that the patient is sensitive
- logAccess: When true, indicates that patient access auditing should occur
- mayAccess: When true, indicates that the user may access the sensitive patient with obligation
- Text: Includes the obligation prompt text for the consuming application

Patient Sensitivity RPC Response

{
isPatientSensitive:{
sensitive:{
dfn:20,
logAccess:true,
mayAccess:true,
text:\r\n***RESTRICTED RECORD***\r\n* * * *

- Patient is an EMPLOYEE (primary or secondary eligibility code) and the user is not a DG SECURITY OFFICER Key holder.

6.2.4.4. Audit

The design of the PEP/PDP allows for future rules to be rapidly added to the ecosystem. As steps are taken to centralize access control attributes and remove the dependency on Vista, the access control solution will be modified to include the rule logic currently embedded in decentralized VistA RPCs.

6.2.4.5. Patient Search and Selection

Patient Search and Selection will be accomplished by locating a given patient within the JDS cache, or, if not found in the cache, by querying the primary VistA site. Subsequent releases will also perform searches against the VA MVI. There are three scenarios that will be supported.

First, the eHMP UI will perform a search against the JDS cache when an end user performs a patient search. This can be done by entering patient traits or identifiers. The search options that will be supported for patient traits are Last Name, First Name, and Social Security Number, consistent with VHA Business Requirements. The UI controls will require a minimum of three characters to be entered for performing trait-based searches. Selection of a patient using an identifier will support either the Patient ICN or the Patient DFN and Site. If a complete patient record is found in the JDS cache, no additional search or synchronization calls will be made to the primary VistA site. If the patient is not found in the cache (or if only a minimal demographic record is found), then the UI will retrieve the full demographic and clinical record using the Patient Sync Subsystem described in section 6.2.6.

Second, the FHIR API is called to perform a patient search against the JDS cache when an external application or process wants to perform a patient search. This can be done by passing in patient traits or identifiers. The search options that will be supported for patient traits are Last Name, First Name, Social Security Number, Date of Birth, and Gender. Partial field entries will not be supported for performing trait-based searches. Selection of a patient using an identifier will support either the Patient ICN or the Patient DFN and Site. If a complete patient record is found in the JDS cache, no additional search or synchronization calls will be made to the primary VistA site. If the patient is not found in the cache (or if only a minimal demographic record is found), then the system will retrieve the full demographic and clinical record using the Patient Sync Subsystem described in section 6.2.6.

Third, the FHIR API is called to search for clinical data (for a given domain, such as allergies) by patient using the ICN or FHIR Patient Subject Identifier. In this case, the calling application or process would pass in the exact ICN or FHIR Patient Subject Identifier. If the patient is found in the JDS cache along with the clinical data, then the FHIR API would retrieve that data directly from the JDS cache. If the patient is not found in the cache, then the FHIR API would initiate a Patient Sync (using the ICN), and wait for the JDS cache to be populated with the clinical data before returning that to the calling application.

In addition to the approaches above for a patient search, release 1.1 will add support for doing an enterprise search utilizing the VA's Master Veteran Index (MVI). This will establish a SOAP connection from VistA Exchange, passing the patients traits. Unlike the approaches above, this is

intended to be used only for very narrow search results. The implementation will be worked out during the next increment.

6.2.5. VistA/MUMPS Development

Initially, eHMP utilizes VistA Code developed by the Health Informatics Initiative (Hi2) Health Management Platform (HMP). This code is organized within the VPR namespace. The main purpose of this code is to extract the patient data from VistA into an M queue used to populate the JDS cache—this mechanism is described in more detail in section 6.2.6 (Patient Sync Subsystem) below. The extraction occurs initially at the time a patient is "subscribed" into the cache. Code is also invoked via a trigger mechanism at the time subsequent data is generated in VistA for a subscribed patient, in order to keep the cache up to date. In addition to this "per patient" subscription mechanism, HMP code is also used to populate JDS with basic demographic information about all the patients at a connected VistA site—this is used for patient selection. In addition, the RPC used to determine whether patient is sensitive was described in section 6.2.4.3 (Access Control).

Since the VPR namespace code is under configuration control of the HMP project, eHMP will fork this code into its own assigned "HMP" namespace. Future changes/maintenance of this code will be performed inside the HMP namespace to avoid collisions with VPR namespace.

In addition to the VistA code, there is additional M development as part of the JDS implementation, hosted in a separate Caché instance. This baseline code is also reused from HMP. Minor changes have been made to this M code, including the increase in the number of listeners to increase scalability. Since the instantiation of the JDS for eHMP is separate than the JDS used in HMP, there is no need to fork the code or move to separate namespace.

6.2.6. Patient Sync Subsystem

The Patient Sync Subsystem is responsible for orchestrating patients being synchronized. The Patient Sync Subsystem provides a Java API to request a sync and to check if there is a sync being performed for the patient.

When a patient sync is requested, the Patient Sync Subsystem invokes a VistA RPC to initiate a subscription for the given patient. This retrieves all data for the patient within VistA and places it into an M-based, outbound queue. The VistA Exchange Patient Sync Subsystem periodically checks for data available from the outbound VistA queue using a polling RPC check. As data becomes available, the Patient Sync Subsystem retrieves the data. The patient sync process then checks the data against the known terminology lookup tables, and applies any of the standards-based terms to the data.

The enriched data record is then stored into the JDS cache using an HTTP PUT command.

The supported terminology data includes:

- Allergies (UMLS CUI)
- Immunizations (CVX)
- Lab Results (LOINC)
- Medications (RxNorm)
- Notes (LOINC)

- Problems (SNOMED CT)
- Vitals (LOINC)

6.2.7. Patient Correlation Service

The Patient Correlation Service is responsible for performing a patient search and for lookup of patient identifiers, including looking up the patient DFN/site code for a given enterprise identifier. The Patient Correlation Service is embedded within the Web Service API and the Patient Sync Subsystem.

Pending assessment from Identity Access Management (IAM) via an existing service request, the Patient Correlation Service will perform real time request/response calls to the MVI to perform the search and get corresponding IDs.

6.2.8. Cache

The clinical data is cached in the JDS subsystem, which is an M-based document data store developed originally as part of the Health Management Platform (HMP) program. JDS is hosted within InterSystems Caché, and provides an HTTP-based API to allow for VistA Exchange to read and update the data. The cached clinical data also gets stored within an instance of SOLR. This is optimized to support full text search.

When the Patient Synch Subsystem receives additional data for a patient, it adds/updates that data to the JDS and SOLR, independent of any end user access to that data. This near real time synchronization architecture keeps the cache updated for the set of patients that have been synchronized with one or more VistA host locations.

6.2.9. VistA Access Approach

There are several technology approaches used to integrate with VistA.

First, when the synchronization process initiates a subscription or checks for published events from VistA, it will connect to VistA using a direct connection to the RPC broker. These calls are done within the context of a system proxy account, and are responsible for initiating a subscription from a VistA Exchange instance to a given VistA host for a specific patient, polling to receive published events representing availability of data subscribed patients, and the pull of operational data, such as lookup lists. This data is stored to the VistA Exchange cache (JDS).

Second, for the retrieval of patient data from remote VistA hosts, the subscription process performs a request by invoking the CDS web service. The CDS web service, in turn, invokes VistA instances to retrieve data for that patient. This data is stored to the VistA Exchange cache (JDS).

Third, under Release 1.1, VistA Exchange will utilize a direct RPC connection for performing writes. These writes will cover the domains of allergies, vitals, and problems. All of these writes will be to VistA. This connection to VistA will be performed using the actual user's access/verify code; no proxy user account will be used in this scenario.

Forth, under a future release when VistA Service Assmebler (VSA) is available, it is expected to migrate the writes from direct RPC broker connection to utilize VSA. This requires an ability to

present a token to VSA as provided from Identity Access Management (IAM) Single Sign-On Internal service SSOi to identify the user.

6.2.10. Patient Generated Data

VistA Exchange has the ability to pull data in from multiple sources. One of the planned sources of data is Patient Generated Data (PGD). This would allow for PGD to be presented alongside data from clinicians with indicators of the patient data source.

To demonstrate this capability, the Release 1.0 reaches out to a mock implementation of web services that represent the expected to-be state of the Virtual Lifetime Electronic Record (VLER) Data Access Service (DAS) Patient Generated Data (PGD) instance. The synchronization process translates the patient data to the Electronic Data Interchange Personal Identifier (EDIPI), and performs a request/response fetch of data from the mock data. This data is then stored to the VistA Exchange cache.

6.2.11. eHMP VistA Components

The definition for the eHMP package is HMP and the assigned number space is 800000-800010. The HMP name and number space consist of 114 routines, 33 RPCs, 14 menu options, 15 protocols, 9 parameters, 9 files, and 1 HL7 application. Initially the addition of new triggers/protocols will allow for system awareness of when it is necessary to refresh patient data within the JDS. Eventually a complete awareness of all patient data and operational data changes will trigger appropriate updates to eHMP. It is important to note that though the VistA components represented here are new in the HMP namespace they were initially developed within the VPR namespace. Initially additional components will be added to enhance existing functionality of either VPR or the HMP code. However the eHMP project has a much more comprehensive need and it is envisioned that there will be both significant modifications to the following components as well as additions of new elements as this project addresses the requirements in an agile fashion.

Please see Appendix C for a complete list of VistA Routines (Entry Points).

6.2.11.1. Data Dictionaries (DD)

Please see Appendix D for Data Dictionaries.

6.2.11.2. Options

Please see Appendix E for Options.

6.2.11.3. Security Keys

Please see Appendix F for Security Keys.

6.2.11.4. Protocols

Please see Appendix G for Protocols.

6.2.11.5. Remote Procedure Calls (RPC)

Please see Appendix H for Remote Procedure Calls

6.2.11.6. Health Level 7 (HL7) Application Parameters

Please see Appendix I for HL7 Application Parameters.

7. External Interface Design

The Web Service API is the public API for accessing data within VistA Exchange. This Web Service API follows the FHIR specification[4]. The detailed interfaced architecture and design will be defined during the development of this increment.

The interfaces that eHMP consumes are Vista, CDS, MVI, and JMEADOWS.

[4] FHIR Specification: http://www hl7.org/implement/standards/fhir/

8. Human-Machine Interface

8.1. Summary

As of the date of this document version, human-machine interface details are still being defined. There will be an eHMP web service delivered at the end of the eHMP Increment 2. The detailed interface design rules, inputs, outputs, and navigation hierarchy are subject to change as additional customer elicitation is held.

In addition, the application will have the following engineering standards and principles:

- 8.1.1 – Responsive Design
- 8.1.2 – Adaptive Design
- 8.1.3 – 508 Accessibility
- 8.1.4 – User-Centered Design iterations with participation from the VA Human Factors Engineering team

8.2. JLV Functionality

The JLV functionality as included in eHMP will be known as Cover Sheet. Cover Sheet is an instance of a screen as defined in section 6.2.1.3. Cover Sheet is meant to be a role-based, preset of applets designed for initial patient EHR summary functions. It will blend VA and DoD data to fully represent the patient record. Additionally, there will be screens/views that are customizable, in order to accommodate user preference.

8.3. Patient Search

Patient Search has been decoupled from the Patient Record in accordance with VA Patient Safety Office requirements.

- Four search types are expected:
  - Global Search: Search is issued against MVI to return patients from across the DoD and VA enterprise. User can search using Name, DOB, and SSN criteria.
  - My Site (All): Allows for Name, SSN search against current VistA (as defined by the facility identification required at login)
  - My Site (Clinic): Returns all patients for a selected clinic location with ability to add data range and name filter qualifiers
  - My Site (Ward): Returns all patients for a selected ward location with ability to add name filter qualifiers
- Patient selection will include a confirmation, giving enough patient information to further qualify the selection.
- Patient selection will elicit sensitivity flags and break the glass scenarios. These flags will be presented as part of the patient selection and will require a confirmation.

8.4. Patient Information

Patient information will be presented in a vertical and will persist across all patient-centric actions, so as to always keep the patient in context.

8.4.1. Crisis Notes, Warnings, Allergies, and Directives (CWAD) Postings

The CWAD postings will include:

- 8.4.1.1 – Allergies
- 8.4.1.2 – Crisis Notes
- 8.4.1.3 – Directives
- 8.4.1.4 - Flags

Patient Record Fields will include:

- 8.4.2 - Last Name, First Name
- 8.4.3 - SSN (555-00-5555)
- 8.4.4 - DOB (01/01/1900)
- 8.4.5 - Age (80y)
- 8.4.6 - Gender (M)
- 8.4.7 - Provider
- 8.4.8 - Ward (if applicable)
- 8.4.9 - Bed (if applicable)
- 8.4.10 - Primary Care Team
- 8.4.11 - Attending Provider (if applicable)
- 8.4.12 - Inpatient Provider (if applicable)

8.5. HMP Functionality

The following functional areas from HMP will be retained:

- Cover Sheet (as described Figure 8.1)
- Medications Review: An aggregate of patient medications both inpatient and outpatient with date range qualifiers to limit the data set.
- News Feed: A historical view of past events in the medical record to include problem list, active medications, and current patient tasks.
- Documents: A listing of all document types with date range qualifiers to limit the data set. This set includes, but is not limited to:
  - Progress Notes
  - Radiology Exams
  - C-CDAs
  - Discharge/Essentris Notes
  - Surgery Reports
  - Consults
- Text Search: To be known as Search Record. This function gives deep exposure to the EHR through a free-text search, as well as predefined lookups into domains, labs, and LOINC codes.

    - Unlike HMP, Text search is not the standard method for data retrieval, and will act a secondary path to discover patient data.
- Tasks: (PSI 6-8) Tasks will be displayed, and given that they are in-progress, can be initialized and completed using the orders and note-writer applets (PSI 6-8).
- Add Task: (PSI 6-8) The ability to create a new note or order against the patient record will be available.

8.6. Header

The application header will contain the following information:

- Patient Selection: Will allow access to the patient selection applet.
- Last 20 Patient Selection: (PSI 6) Will give list of the last 20 patients viewed, and will allow for hot-swapping of patient context (along with confirmation and sensitivity flags).
- Display of User Information: Displays the current logged in user and facility.
- Access to help features for any given screen (PSI 7).

8.7. Footer

The application footer will contain the following information:

- Application Name (eHMP).
- Application version (v##.##.##).

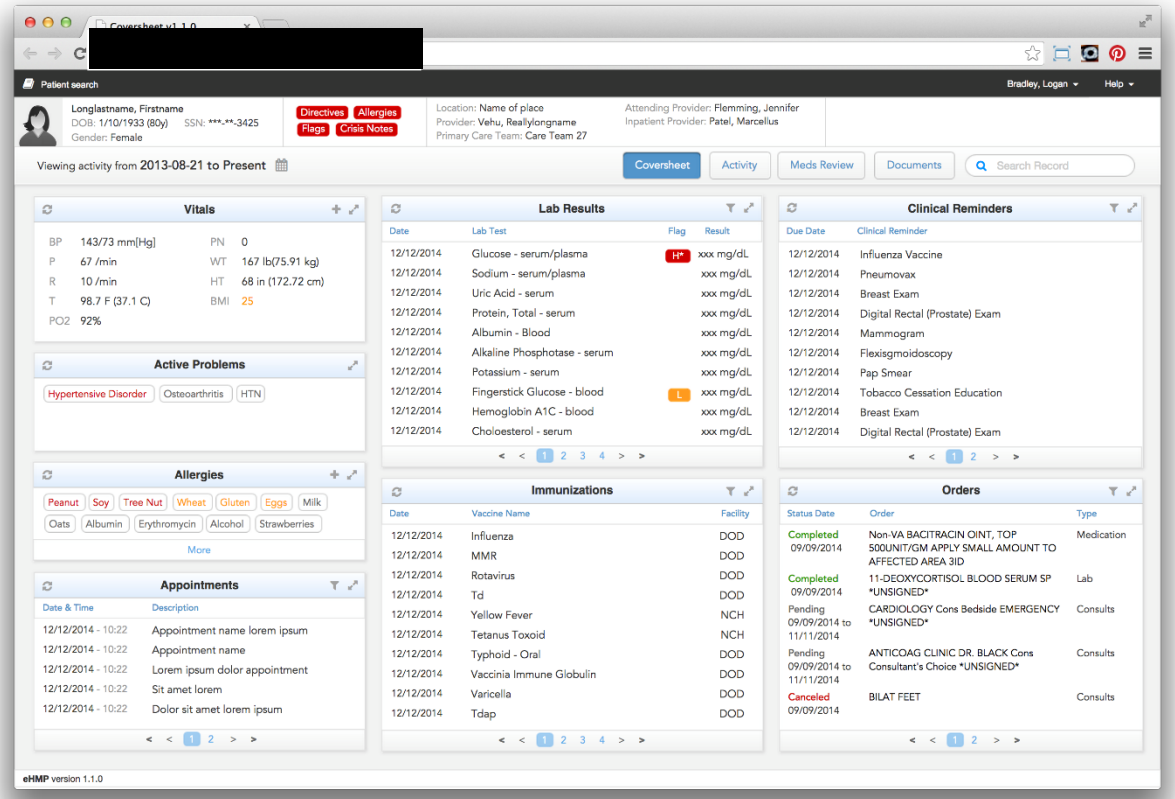


Figure 8-1 Patient Cover Sheet

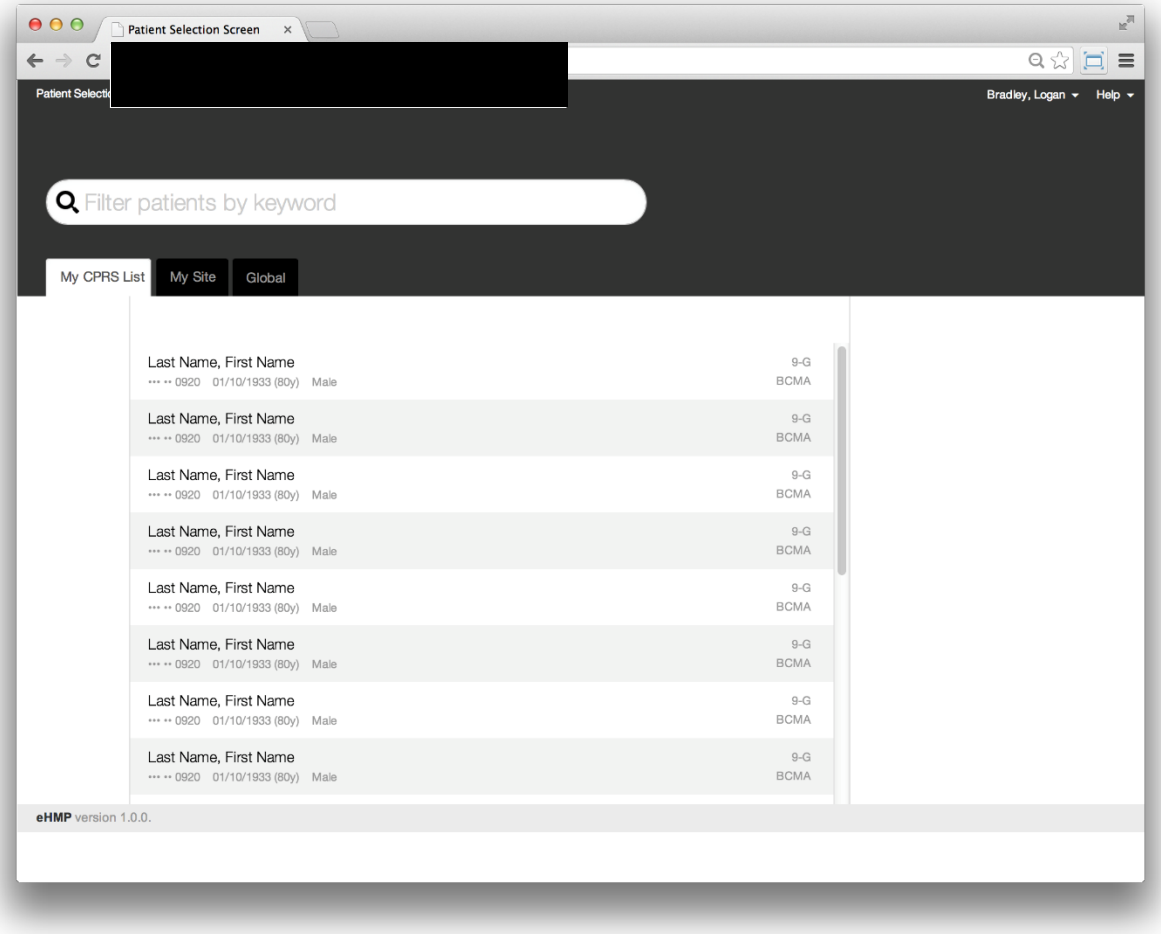


Figure 8-2 Patient Selection Screen

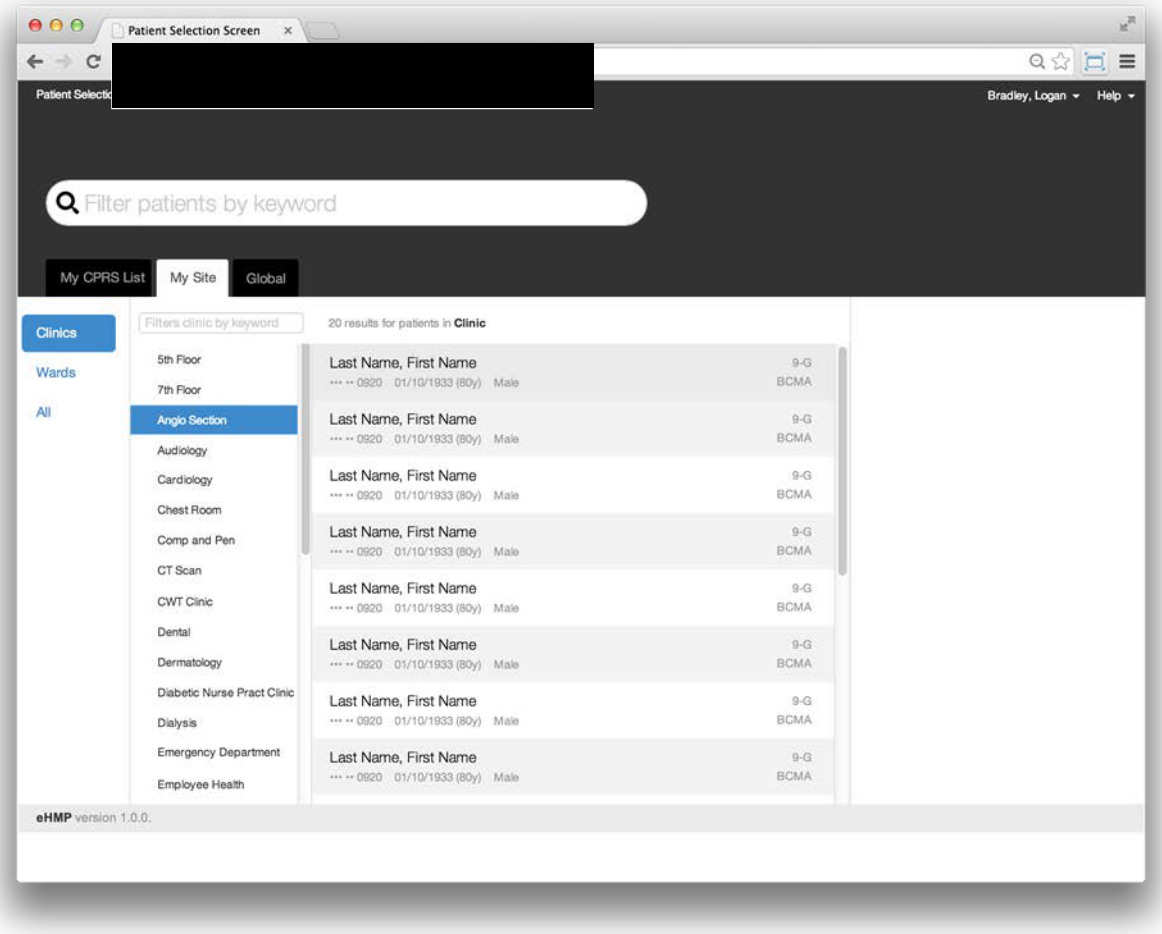


Figure 8-3 Patient Selection (Clinic)

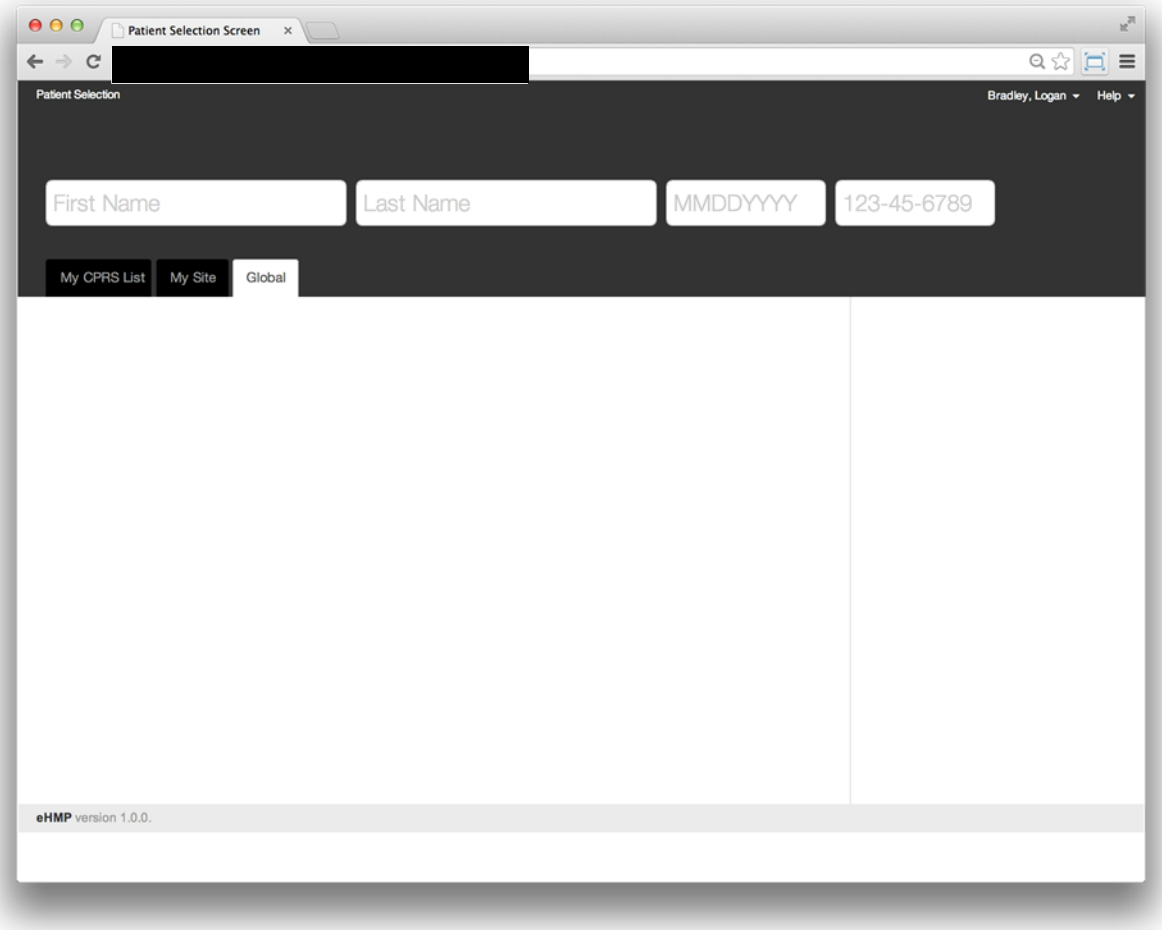


Figure 8-4 Global Patient Selection

9. System Integrity Controls

The eHMP system design will comply with the NIST 800-53 Rev4[5] Security Controls as described in the eHMP System Security Plan (SSP). This plan will be available within the VA's Governance Risk and Compliance (GRC) tool.

Encryption of data at rest will be protected using Federal Information Professing Standard (FIPS) 140-2 compliant encryption algorithms and products to protect patient data. A Caché database will be used to store data in multiple files (or UNIX raw disk partitions) called databases. All database reads and writes occur in 8192-byte blocks. This includes application data as well as metadata such as indices, bitmaps, pointers, allocation maps, and incremental backup maps. All blocks (with the exception of a single initial label block) are always encrypted on disk.

The actual encryption and decryption is done at a low-level in the Caché system code, and uses the United States Government Advanced Encryption Standard (AES) [FIPS 197] in Cipher Block Chaining (CBC) mode [NIST 800-38A], with a 128-, 192-, or 256-bit database encryption key. The initialization vector for each block is generated by encrypting a big-endian representation of the logical block number using AES in Electronic Codebook (ECB) mode [NIST 800-38A & Addendum], using the database encryption key.

Data at rest stored in other areas of the system, such as log files that could hold sensitive data, will be encrypted using Red Hat Encrypted File System (EFS), which utilizes FIPS 140-2 compliant encryption algorithms.

9.1. Transmission Encryption

The eHMP system design will employ the secure Transport Layer Security (TLS) protocol to protect the integrity of data in transit. TLS will establish FIPS-compliant, encrypted communications sessions on connections outside of the eHMP environment. These connections will be established based upon VA-issued Digital Encryption Certificates installed on each of the eHMP servers, and will protect the confidentiality and integrity of the data being transmitted.

Multifactor user authentication will be employed as PIV cards are implemented throughout the VA staff user population. It is anticipated that single-factor user ID and password may be required at the initial launch of the eHMP system, until the entire VA staff user population is issued PIV cards. The multifactor authentication will consist of something a user has (PIV card), and something they know (PIN). These factors will enhance the integrity controls by making it exponentially more difficult for a user account to be compromised. Once a user is identified and authenticated, his or her access to system resources and application functionality will be predicated upon role-based access controls.

[5] NIST 800-53 Rev4: http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-53r4.pdf

9.2. System Administrator Access

The eHMP production environment will be hosted within the VA AITC EO cloud computing environment. The system administrator access necessary to manage eHMP resources will employ the following technical resources and methods to ensure the integrity of the production environment:

- Administrator access will require a user account that is authorized and granted administrator privileges for the eHMP system resources.
- Administrators will be limited to accessing only those resources necessary to accomplish their assigned tasks.
- Administrator access will be subject to auditing of all privileged actions to include logon, logoff, user account management, and administrative access to system servers and sensitive resources such as database access.

A user who is assigned administrative duties will be required to authenticate via a VA-issued PIV as described in sections 6.2.3, 9.1, and 9.2. In order to establish an administrative connection with the eHMP system, a user must be using the VA Citrix Access Gateway (CAG) and a valid, VA-issued PIV. Once authenticated, the administrative user will be joined to the local VLAN hosting the eHMP system resources, and the VPN client for Citrix Receiver will establish a FIPS 140-2 compliant, encrypted session that will enforce isolation of the administrator's laptop or desktop to prevent a compromise of the production environment.

9.3. Flaw Remediation

The eHMP system will handle remediation of software flaws by utilizing a number of tools to detect and remediate vulnerabilities such as Nessus, HP Fortify, and SonarQube. A yum repository that is controlled by AITC will exist within the environment to download and apply appropriate patches to the instances of Red Hat Linux OS. Patching of additional open source software components will be done manually, and any vulnerabilities will be identified through monthly VA Network Security Operations Center (NSOC) Nessus scans. VA NSOC Nessus scan results will be provided to the DevOps and Security teams for review and remediation. Any findings that require further testing and evaluation will be tracked in a Plan of Actions and Milestones (POA&M) until closed.

The eHMP source code will be scanned by the development team utilizing an HP Fortify Static Code Analyzer (SCA), prior to each build, and all critical and high vulnerabilities will be remediated before moving to the Enterprise Development Environment for testing on the VA network. Code quality will be evaluated using tools built into the SonarQube software. The eHMP program will support three environments to include Test, Pre-Production and Production. All changes to the source code will be tested thoroughly in each of the environments to identify and eliminate the risk of flaws prior to migrating to production. All servers within the test, pre-production and production environments are required by the VA to have McAfee Antivirus and Host Intrusion Protection Systems (HIPS) for continuous monitoring and protection from any malware and/or malicious code. Remediation processes for related projects such as JLV will be evaluated for incorporation of best practices.

VA NSOC performs monitoring on all AITC networks utilizing IDS/IPS sensors, and monitors the gateways, firewalls, and routers to conduct near real time analysis using HP OpenView software. eHMP will utilize a suite of open source tools that includes Sensu, ElasticSearch, Kibana and FluentD that will serve as a monitoring and logging solution for the environment.

9.4. Configuration Management (CM)

Please refer to the eHMP Configuration Management Plan and Process Module document for further details.

Attachment A – Approval Signatures

This section is used to document the approval of the System Design Document. The review should be conducted face to face where signatures can be obtained 'live' during the review. If unable to conduct a face-to-face meeting then it should be held via LiveMeeting and concurrence captured during the meeting. The Scribe should add /es/name by each position cited. Example provided below.

The Chair of the governing Integrated Project Team (IPT), Business Sponsor, IT Program Manager, and Project Manager are required to sign.

_____

Integrated Project Team Chair Date

_____

Business Sponsor Date:

_____

IT Program Manager Date:

_____

Project Manager Date:

_____

Enterprise Architecture Date:

_____

Service Delivery and Engineering Date:

A. Appendix 1 – Acronym List

Table A-1 Acronym List

Acronym	Description
A&A	Assessment and Authorization
ACID	Atomicity, Consistency, Isolation, Durability
AITC	Austin Information Technology Center
API	Application Programming Interface
ASCII	American Standard Code for Information Interchange
ASM	ASM Research
BHIE	Bidirectional Health Exchange
BSON	Binary JSON
CAG	Citrix Access Gateway
CBC	Cipher Block Chaining
CCB	Change Control Board
CCOW	Clinical Context Object Workgroup
CDA	Clinical Document Architecture
CDS	Clinical Data Service
CDW	Corporate Data Warehouse
CEM	Clinical Element Model
CHDR	Clinical Health Data Repository
CLIN	Contract Line Item Number
CMDB	Configuration Management Database
CPMP	Contractor Project Management Plan
CPRS	Computerized Patient Record System
CR	Change Request
DAO	Data Access Object
DAS	Data Access Service
DB	Database
DBMS	Database Management System
DCO	Data Center Operations
DHMSM	Defense Healthcare Management Systems Modernization
DICOM	Digital Imaging and Communications in Medicine
DFN	Data File Number

Acronym	Description
DMIX	Defense Medical Information Exchange
DMZ	Demilitarized Zone
DoD	Department of Defense
ECB	Electronic Codebook
EDE	Enterprise Development Environment
EHR	Electronic Health Record
EO	Enterprise Operations
eHMP	Enterprise Health Management Platform
eVPR	Enterprise Virtual Patient Record
FHIM	Federal Health Information Model
FHIR	Fast Healthcare Interoperability Resources
FIPS	Federal Information Processing Standards
GUI	Graphical User Interface
HDR	Health Data Repository
Hi2	Health Informatics Initiative
HIE	Health Information Exchange
HIPS	Host Intrusion Protection Systems
HITSP	Healthcare Information Technology Standards Panel
HL7	Health Level Seven International
HMP	Health Management Platform
HSDW	Health Services Data Warehouse
HTTP	Hypertext Transfer Protocol
IAM	Identity Access Management
ICN	Integration Control Number
iEHR	Integrated Electronic Health Record
JAR	Java Archive
JDS	JSON Data Store
JLV	Joint Legacy Viewer
JSON	JavaScript Object Notation
JVM	Java Virtual Machine
LOINC	Logical Observation Identifiers Names and Codes
MBI	Moderate Risk Background Investigation

Acronym	Description
MDHT	Model Driven Health Tools
MDR	MHS Data Repository
MDWS	Medical Domain Web Services
MHS	Military Health System
MHV	My HealtheVet
MVI	Master Veteran Index
MU	Meaningful Use
MVC	Model View Controller
NDAA	National Defense Authorization Act
NIEM	National Information Exchange Model
NwHIN	Nationwide Health Information Network
OHT	Open Health Tools
OI&T	Office of Information Technology
ONC	Office of National Coordinator
OSEHRA	Open Source Electronic Health Record Agent
PB	Petabyte
PDTS	Pharmacy Data Transaction Service
PITC	Philadelphia Information Technology Center
PDF	Portable Document Format
PMAS	Project Management Accountability System
POA&M	Plan of Actions and Milestones
PSI	Potentially Shippable Increment
QA	Quality Assurance
RDF	Resource Description Framework
REST	Representational State Transfer
RHEL	Red Hat Enterprise Linux
RPC	Remote Procedure Call
SAM	Scaled Agile Methodology
SAN	Storage Area Network
SCA	Static Code Analyzer
SDD	System Design Document
SDK	Software Development Kit

Acronym	Description
SDM	Service Desk Manager
SNOMED	Systematized Nomenclature of Medicine
SOA	Service-Oriented Architecture
SOR	System of Record
TCP/IP	Transmission Control Protocol/Internet Protocol
TLS	Transport Layer Security
TRM	Technical Reference Manual
UAT	User Acceptance Testing
URL	Uniform Resource Locator
UX	User Experience
VA	Department of Veterans Affairs
VDEF	VistA Data Extraction Framework
VHA	Veterans Health Administration
VistA	Veterans Health Information Systems and Technology Architecture
VLER	Veteran Lifetime Electronic Record
VM	Virtual Machine
VPR	Virtual Patient Record
WAR	Web Application Archive
W3C	World Wide Web Consortium
XACML	Extensible Access Control Markup
XML	Extensible Markup Language

B. Appendix 2 – FHIR Mappings

The most current version of the mappings to FHIR can be found on the project wiki: [redacted]

An example of an allergy represented via FHIR is listed below:


```
{
 resourceType:AdverseReaction,
 extension: [
 {
 url:[redacted]
 value:allergy
 },
 extension: [
 {
 url[redacted],
 value:2013-12-11T12:13:00
 }
 ],
 text:{
 status:generated,
 div:<div>PENICILLIN</div>
 },
 contained:[
 {
 resourceType:Practitioner,
 id:3702dba0-68cf-11e3-949a-0800200c9a66,
 text:{
 status:generated,
 div:<div>PROVIDER,ONE</div>
 },
 type:{
 text:PROVIDER,ONE
 }
 },
 {
 resourceType:Substance,
 id:b9f845b6-cce1-4f3b-bb9d-e87e9a7b9eb6,
 text:{
 status:generated,
 div:<div>PENICILLIN</div>
 },
 type:{
 coding:{
  system:urn:oid:2.16.840.1.113883.6.233,
  code:urn:va:vuid:4019880,
```

```
 display:PENICILLIN
},
text:PENICILLIN
}
}
],
identifier:{
use:official,
value:urn:va:B362:1:art:1
},
subject:{
reference:Patient/1
},
didNotOccurFlag: false,
recorder: {
reference:#3702dba0-68cf-11e3-949a-0800200c9a66
},
exposure:[
{
substance: {
reference:#b9f845b6-cce1-4f3b-bb9d-e87e9a7b9eb6
}
}
]
}
```

C. Appendix 3 - VistA Routines (Entry Points)

Routine Name	HMPACT	
Improvement Category	☒ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References	N/A	
Related Protocols	N/A	
Related Integration Control Registrations (ICRs)	N/A	
Data Passing	☒ Input ☐ Output Reference ☒ Both ☐ Global Reference ☐ Local	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPAP1	
Enhancement Category	☒ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD)		

Routine Name	HMPAP1
References	
Related Protocols	
Related Integration Control Registrations (ICRs)	
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Input Attribute Name and Definition	Name: Definition:
Output Attribute Name and Definition	Name: Definition:
Current Logic	
Modified Logic (Changes are in bold)	

Routine Name	**HMPATRG**	
Enhancement Category	☒ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name:	

Routine Name	**HMPATRG**
	Definition:
Current Logic	
Modified Logic (Changes are in bold)	

Routine Name	HMPPTDEM	
Enhancement Category	☒ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPMOCK
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change

Routine Name	HMPMOCK	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPCAC	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration		

Routine Name	HMPCAC
Control Registrations (ICRs)	
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Input Attribute Name and Definition	Name: Definition:
Output Attribute Name and Definition	Name: Definition:
Current Logic	
Modified Logic (Changes are in bold)	

Routine Name	HMPCORD	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		

Routine Name	HMPCORD
Modified Logic (Changes are in bold)	

Routine Name	HMPCORD1	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	**Name:** **Definition:**	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPCORD2	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**

Routine Name	HMPCORD2	
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPCORD3	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output Reference ☐ Both ☐ Global Reference ☐ Local	

Routine Name	HMPCORD3
Input Attribute Name and Definition	Name: Definition:
Output Attribute Name and Definition	Name: Definition:
Current Logic	
Modified Logic (Changes are in bold)	

Routine Name	HMPCORD4	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPCPAT	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPCPAT1	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		

Routine Name	HMPCPAT1
Related Integration Control Registrations (ICRs)	
Data Passing	☐ Input ☐ Output Reference ☐ Both ☐ Global Reference ☐ Local
Input Attribute Name and Definition	Name: Definition:
Output Attribute Name and Definition	Name: Definition:
Current Logic	
Modified Logic (Changes are in bold)	

Routine Name	HMPCPRS	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output Reference ☐ Both ☐ Global Reference ☐ Local	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		

Routine Name	HMPCPRS
Modified Logic (Changes are in bold)	

Routine Name	HMPCRPC	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPCRPC1
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change
Requirement Traceability Matrix	
Related Options	

Routine Name	HMPCRPC1	
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPD	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local	

Routine Name	HMPD
	Reference
Input Attribute Name and Definition	Name: Definition:
Output Attribute Name and Definition	Name: Definition:
Current Logic	
Modified Logic (Changes are in bold)	

Routine Name	HMPDERRH	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPDGMPL	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPDGMRA	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		

Routine Name	HMPDGMRA
Related Integration Control Registrations (ICRs)	
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Input Attribute Name and Definition	Name: Definition:
Output Attribute Name and Definition	Name: Definition:
Current Logic	
Modified Logic (Changes are in bold)	

Routine Name	HMPDGMRC	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		

Routine Name	HMPDGMRC
Modified Logic (Changes are in bold)	

Routine Name	HMPDGMV	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPDGPF	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**

Routine Name	HMPDGPF
Data Dictionary (DD) References	
Related Protocols	
Related Integration Control Registrations (ICRs)	
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Input Attribute Name and Definition	Name: Definition:
Output Attribute Name and Definition	Name: Definition:
Current Logic	
Modified Logic (Changes are in bold)	

Routine Name	HMPDIB	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name:	

Routine Name	HMPDIB
	Definition:
Current Logic	
Modified Logic (Changes are in bold)	

Routine Name	HMPDJ	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPDJ0
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change
Requirement Traceability Matrix	

Routine Name	HMPDJ0	
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPDJ00	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	

Routine Name	HMPDJ00
Input Attribute Name and Definition	Name: Definition:
Output Attribute Name and Definition	Name: Definition:
Current Logic	
Modified Logic (Changes are in bold)	

Routine Name	HMPDJ00A	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPDJ01

Routine Name	HMPDJ01	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPDJ02	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations		

Routine Name	HMPDJ02
(ICRs)	
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Input Attribute Name and Definition	Name: Definition:
Output Attribute Name and Definition	Name: Definition:
Current Logic	
Modified Logic (Changes are in bold)	

Routine Name	HMPDJ03	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPDJ04	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPDJ04A	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		

Routine Name	HMPDJ04A
Related Integration Control Registrations (ICRs)	
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Input Attribute Name and Definition	Name: Definition:
Output Attribute Name and Definition	Name: Definition:
Current Logic	
Modified Logic (Changes are in bold)	

Routine Name	HMPDJ04E	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		

Routine Name	HMPDJ04E
Modified Logic (Changes are in bold)	

Routine Name	HMPDJ05	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPDJ05V	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**

Routine Name	HMPDJ05V
Data Dictionary (DD) References	
Related Protocols	
Related Integration Control Registrations (ICRs)	
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Input Attribute Name and Definition	Name: Definition:
Output Attribute Name and Definition	Name: Definition:
Current Logic	
Modified Logic (Changes are in bold)	

Routine Name	HMPDJ06	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name:	

Routine Name	HMPDJ06
	Definition:
Current Logic	
Modified Logic (Changes are in bold)	

Routine Name	HMPDJ07	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPDJ08
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change
Requirement Traceability Matrix	

Routine Name	HMPDJ08	
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPDJ08A	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	

Routine Name	HMPDJ08A
Input Attribute Name and Definition	Name: Definition:
Output Attribute Name and Definition	Name: Definition:
Current Logic	
Modified Logic (Changes are in bold)	

Routine Name	HMPDJ09	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPDJ09M

Routine Name	HMPDJ09M	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output Reference ☐ Both ☐ Global Reference ☐ Local	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPDJ1	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations		

Routine Name	HMPDJ1
(ICRs)	
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Input Attribute Name and Definition	Name: Definition:
Output Attribute Name and Definition	Name: Definition:
Current Logic	
Modified Logic (Changes are in bold)	

Routine Name	HMPDJ2	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPDJFS	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPDJFSD	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		

Routine Name	HMPDJFSD
Related Integration Control Registrations (ICRs)	
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Input Attribute Name and Definition	Name: Definition:
Output Attribute Name and Definition	Name: Definition:
Current Logic	
Modified Logic (Changes are in bold)	

Routine Name	HMPDJFSG	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		

Routine Name	HMPDJFSG
Modified Logic (Changes are in bold)	

Routine Name	HMPDJFSM	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPDJFSP	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**

Routine Name	HMPDJFSP
Data Dictionary (DD) References	
Related Protocols	
Related Integration Control Registrations (ICRs)	
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Input Attribute Name and Definition	Name: Definition:
Output Attribute Name and Definition	Name: Definition:
Current Logic	
Modified Logic (Changes are in bold)	

Routine Name	HMPDJFST	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name:	

Routine Name	HMPDJFST
	Definition:
Current Logic	
Modified Logic (Changes are in bold)	

Routine Name	HMPDJX	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPDLR
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change
Requirement Traceability Matrix	

Routine Name	HMPDLR	
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPDLRA	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	

Routine Name	HMPDLRA
Input Attribute Name and Definition	Name: Definition:
Output Attribute Name and Definition	Name: Definition:
Current Logic	
Modified Logic (Changes are in bold)	

Routine Name	HMPDLRO	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPDMC

Routine Name	HMPDMC	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPDMDC	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration		

Routine Name	HMPDMDC
Control Registrations (ICRs)	
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Input Attribute Name and Definition	Name: Definition:
Output Attribute Name and Definition	Name: Definition:
Current Logic	
Modified Logic (Changes are in bold)	

Routine Name	HMPDOR	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPDOR

Routine Name	HMPDPROC	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPDPS	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD)		

Routine Name	HMPDPS
References	
Related Protocols	
Related Integration Control Registrations (ICRs)	
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Input Attribute Name and Definition	Name: Definition:
Output Attribute Name and Definition	Name: Definition:
Current Logic	
Modified Logic (Changes are in bold)	

Routine Name	HMPDPSI	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	

Routine Name	HMPDPSI
Current Logic	
Modified Logic (Changes are in bold)	

Routine Name	HMPDPSO	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPDPSOR
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change
Requirement Traceability Matrix	
Related Options	

Routine Name	HMPDPSOR	
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPDPT	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	

Routine Name	HMPDPT
Input Attribute Name and Definition	Name: Definition:
Output Attribute Name and Definition	Name: Definition:
Current Logic	
Modified Logic (Changes are in bold)	

Routine Name	HMPDPXAM	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPDPXED

Routine Name	HMPDPXED	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	**HMPDPXHF**	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations		

Routine Name	**HMPDPXHF**
(ICRs)	
Data Passing	☐ **Input** ☐ **Output Reference** ☐ **Both** ☐ **Global Reference** ☐ **Local**
Input Attribute Name and Definition	**Name:** **Definition:**
Output Attribute Name and Definition	**Name:** **Definition:**
Current Logic	
Modified Logic (Changes are in bold)	

Routine Name	HMPDPXIM	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPDPXSK	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPDRA	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		

Routine Name	HMPDRA
Related Integration Control Registrations (ICRs)	
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Input Attribute Name and Definition	Name: Definition:
Output Attribute Name and Definition	Name: Definition:
Current Logic	
Modified Logic (Changes are in bold)	

Routine Name	HMPDSDAM	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		

Routine Name	HMPDSDAM
Modified Logic (Changes are in bold)	

Routine Name	HMPDSR	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPDTIU	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called**	**Routines "Called"**

Routine Name	HMPDTIU	
	By"	
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPDTIUX	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and	Name:	

Routine Name	HMPDTIUX
Definition	Definition:
Output Attribute Name and Definition	Name: Definition:
Current Logic	
Modified Logic (Changes are in bold)	

Routine Name	HMPDVSIT	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPEASU
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change

Routine Name	HMPEASU	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPEF	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration		

Routine Name	HMPEF
Control Registrations (ICRs)	
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Input Attribute Name and Definition	Name: Definition:
Output Attribute Name and Definition	Name: Definition:
Current Logic	
Modified Logic (Changes are in bold)	

Routine Name	**HMPEFSG**	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	**HMPEFSG**

Routine Name	HMPEFSP	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPEFST	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**

Routine Name	HMPEFST
Data Dictionary (DD) References	
Related Protocols	
Related Integration Control Registrations (ICRs)	
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Input Attribute Name and Definition	Name: Definition:
Output Attribute Name and Definition	Name: Definition:
Current Logic	
Modified Logic (Changes are in bold)	

Routine Name	HMPEFX	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name:	

Routine Name	HMPEFX
	Definition:
Current Logic	
Modified Logic (Changes are in bold)	

Routine Name	HMPEHL7	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPELAB
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change
Requirement Traceability Matrix	

Routine Name	HMPELAB	
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPENSZ	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	

Routine Name	HMPENSZ
Input Attribute Name and Definition	Name: Definition:
Output Attribute Name and Definition	Name: Definition:
Current Logic	
Modified Logic (Changes are in bold)	

Routine Name	HMPENSZ1	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPEVNT

Routine Name	HMPEVNT	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPFPTC	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		

Routine Name	HMPFPTC
Related Integration Control Registrations (ICRs)	
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Input Attribute Name and Definition	Name: Definition:
Output Attribute Name and Definition	Name: Definition:
Current Logic	
Modified Logic (Changes are in bold)	

Routine Name	**HMPHTTP**	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		

Routine Name	**HMPHTTP**
Modified Logic (Changes are in bold)	

Routine Name	HMPIDX	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPLIST	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**

Routine Name	HMPLIST	
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output Reference ☐ Both ☐ Global Reference ☐ Local	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPMDUTL	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output Reference ☐ Both ☐ Global Reference ☐ Local	
Input Attribute Name and Definition	Name: Definition:	

Routine Name	HMPMDUTL
Output Attribute Name and Definition	Name: Definition:
Current Logic	
Modified Logic (Changes are in bold)	

Routine Name	HMPP3I	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPPANEL
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change

Routine Name	HMPPANEL	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPPARAM	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		

Routine Name	HMPPARAM
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Input Attribute Name and Definition	Name: Definition:
Output Attribute Name and Definition	Name: Definition:
Current Logic	
Modified Logic (Changes are in bold)	

Routine Name	HMPPATS	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPPI	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Reference ☐ Global Reference ☐ Local Both	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPPRODC	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		

Routine Name	HMPPRODC
Related Integration Control Registrations (ICRs)	
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Input Attribute Name and Definition	Name: Definition:
Output Attribute Name and Definition	Name: Definition:
Current Logic	
Modified Logic (Changes are in bold)	

Routine Name	HMPPXPR1	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		

Routine Name	HMPPXPR1
Modified Logic (Changes are in bold)	

Routine Name	HMPPXRM	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	**HMPROS2**	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**

Routine Name	**HMPROS2**	
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPROS3	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	

Routine Name	HMPROS3
Output Attribute Name and Definition	Name: Definition:
Current Logic	
Modified Logic (Changes are in bold)	

Routine Name	HMPROS4	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPROS5
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change

Routine Name	HMPROS5	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPROS6	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		

Routine Name	HMPROS6
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Input Attribute Name and Definition	Name: Definition:
Output Attribute Name and Definition	Name: Definition:
Current Logic	
Modified Logic (Changes are in bold)	

Routine Name	HMPROS7	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPROS8	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPSR	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		

Routine Name	HMPSR
Related Integration Control Registrations (ICRs)	
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Input Attribute Name and Definition	Name: Definition:
Output Attribute Name and Definition	Name: Definition:
Current Logic	
Modified Logic (Changes are in bold)	

Routine Name	HMPTRPC	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		

Routine Name	HMPTRPC
Modified Logic (Changes are in bold)	

Routine Name	HMPTRPC1	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPUPD	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**

Routine Name	HMPUPD
Data Dictionary (DD) References	
Related Protocols	
Related Integration Control Registrations (ICRs)	
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Input Attribute Name and Definition	Name: Definition:
Output Attribute Name and Definition	Name: Definition:
Current Logic	
Modified Logic (Changes are in bold)	

Routine Name	HMPUTILS	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name:	

Routine Name	HMPUTILS
	Definition:
Current Logic	
Modified Logic (Changes are in bold)	

Routine Name	HMPYCSI	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPYFRP
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change
Requirement Traceability	

Routine Name	HMPYFRP	
Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

Routine Name	HMPYFRP1	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		

Routine Name	HMPYFRP1
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Input Attribute Name and Definition	Name: Definition:
Output Attribute Name and Definition	Name: Definition:
Current Logic	
Modified Logic (Changes are in bold)	

Routine Name	HMPYPAR	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
Requirement Traceability Matrix		
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Input Attribute Name and Definition	Name: Definition:	
Output Attribute Name and Definition	Name: Definition:	
Current Logic		
Modified Logic (Changes are in bold)		

D. Appendix 4 - Data Dictionaries (DD)

File Name and Number	HMP SUBSCRIPTION # 800000
Enhancement Category	☒ New ☐ Modify ☐ Delete ☐ No Change
Requirements Traceability Matrix	
Related Options	
Data Dictionary (DD) References	
Related Protocols	
Related Integration Control Registrations (ICRs) Agreements	
Data Passing	☐ Input ☐ Output ☐ Both Reference ☐ Global Reference ☐ Local
File Documentation	
File Auditing, Security, and Archiving	

File Name and Number	HMP PATIENT OBJECT #800000.1
Enhancement Category	☒ New ☐ Modify ☐ Delete ☐ No Change
Requirements Traceability Matrix	
Related Options	
Data Dictionary (DD) References	
Related Protocols	
Related Integration Control Registrations (ICRs) Agreements	
Data Passing	☐ Input ☐ Output ☐ Both Reference ☐ Global Reference ☐ Local
File Documentation	
File Auditing, Security, and Archiving	

File Name and Number	HMP OBJECT #800000.11
Enhancement Category	☒ New ☐ Modify ☐ Delete ☐ No Change
Requirements Traceability Matrix	
Related Options	
Data Dictionary (DD) References	
Related Protocols	
Related Integration Control Registrations (ICRs) Agreements	
Data Passing	☐ Input ☐ Output ☐ Both Reference ☐ Global Reference ☐ Local
File Documentation	
File Auditing, Security, and Archiving	

File Name and Number	HMP LIST #800000.2
Enhancement Category	☒ New ☐ Modify ☐ Delete ☐ No Change
Requirements Traceability Matrix	
Related Options	
Data Dictionary (DD) References	
Related Protocols	
Related Integration Control Registrations (ICRs) Agreements	
Data Passing	☐ Input ☐ Output ☐ Both Reference ☐ Global Reference ☐ Local
File Documentation	
File Auditing, Security, and Archiving	

File Name and Number	HMP LIST DOMAIN #800000.21
Enhancement Category	☒ New ☐ Modify ☐ Delete ☐ No Change
Requirements Traceability Matrix	
Related Options	
Data Dictionary (DD) References	
Related Protocols	
Related Integration Control Registrations (ICRs) Agreements	
Data Passing	☐ Input ☐ Output ☐ Both Reference ☐ Global Reference ☐ Local
File Documentation	
File Auditing, Security, and Archiving	

File Name and Number	HMP LIST ATTRIBUTE #800000.22
Enhancement Category	☒ New ☐ Modify ☐ Delete ☐ No Change
Requirements Traceability Matrix	
Related Options	
Data Dictionary (DD) References	
Related Protocols	
Related Integration Control Registrations (ICRs) Agreements	
Data Passing	☐ Input ☐ Output ☐ Both Reference ☐ Global Reference ☐ Local
File Documentation	
File Auditing, Security, and Archiving	

File Name and Number	HMPPANEL #800001
Enhancement Category	☒ New ☐ Modify ☐ Delete ☐ No Change
Requirements	

Traceability Matrix	
Related Options	
Data Dictionary (DD) References	
Related Protocols	
Related Integration Control Registrations (ICRs) Agreements	
Data Passing	☒ Input ☐ Output ☐ Both Reference ☐ Global Reference ☐ Local
File Documentation	
File Auditing, Security, and Archiving	

File Name and Number	HMP ROSTER #800001.2
Enhancement Category	☒ New ☐ Modify ☐ Delete ☐ No Change
Requirements Traceability Matrix	
Related Options	
Data Dictionary (DD) References	
Related Protocols	
Related Integration Control Registrations (ICRs) Agreements	
Data Passing	☒ Input ☐ Output ☐ Both Reference ☐ Global Reference ☐ Local
File Documentation	
File Auditing, Security, and Archiving	

File Name and Number	HMP ACTIVITY 800001.5
Enhancement Category	☒ New ☐ Modify ☐ Delete ☐ No Change
Requirements Traceability Matrix	
Related Options	
Data Dictionary (DD)	

References	
Related Protocols	
Related Integration Control Registrations (ICRs) Agreements	
Data Passing	☐ Input ☐ Output ☐ Both Reference ☐ Global Reference ☐ Local
File Documentation	
File Auditing, Security, and Archiving	

E. Appendix 5 - Options

Option Name	HMP APPLICATION PROXY
Enhancement Category	☒ New ☐ Modify ☐ Delete ☐ No Change
Associated Menu Options, etc., that will invoke this reference	
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Menu Text Description	
Option Type	☐ Edit ☐ Print ☐ Menu ☐ Inquire ☐ Action ☐ Run Routine ☐ Other
Associated Routine	
Option Definition	
Current Entry Action Logic	
Modified Entry Action Logic (Changes are in bold)	
Current Exit Action Logic	
Modified Exit Action Logic (Changes are in bold)	

Option Name	HMP APPOINTMENTS
Enhancement Category	☒ New ☐ Modify ☐ Delete ☐ No Change
Associated Menu Options, etc., that will invoke this reference	
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Menu Text Description	

Option Type	☐ Edit ☐ Print ☐ Menu ☐ Inquire ☐ Action ☐ Run Routine ☐ Other
Associated Routine	
Option Definition	
Current Entry Action Logic	
Modified Entry Action Logic (Changes are in bold)	
Current Exit Action Logic	
Modified Exit Action Logic (Changes are in bold)	

Option Name	HMP PATIENT ACTIVITY
Enhancement Category	☒ New ☐ Modify ☐ Delete ☐ No Change
Associated Menu Options, etc., that will invoke this reference	
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Menu Text Description	
Option Type	☐ Edit ☐ Print ☐ Menu ☐ Inquire ☐ Action ☐ Run Routine ☐ Other
Associated Routine	
Option Definition	
Current Entry Action Logic	
Modified Entry Action Logic (Changes are in bold)	
Current Exit Action Logic	
Modified Exit Action Logic (Changes are in bold)	

Option Name	HMP PATIENT DATA MONITOR
Enhancement Category	☒ New ☐ Modify ☐ Delete ☐ No Change
Associated Menu Options, etc., that will invoke this reference	
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Menu Text Description	
Option Type	☐ Edit ☐ Print ☐ Menu ☐ Inquire ☐ Action ☐ Run Routine ☐ Other
Associated Routine	
Option Definition	
Current Entry Action Logic	
Modified Entry Action Logic (Changes are in bold)	
Current Exit Action Logic	
Modified Exit Action Logic (Changes are in bold)	

Option Name	HMP SYNCHRONIZATION CONTEXT
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change
Associated Menu Options, etc., that will invoke this reference	
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Menu Text	

Description	
Option Type	☐ Edit ☐ Print ☐ Menu ☐ Inquire ☐ Action ☐ Run Routine ☐ Other
Associated Routine	
Option Definition	
Current Entry Action Logic	
Modified Entry Action Logic (Changes are in bold)	
Current Exit Action Logic	
Modified Exit Action Logic (Changes are in bold)	

Option Name	HMP UI CONTEXT
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change
Associated Menu Options, etc., that will invoke this reference	
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Menu Text Description	
Option Type	☐ Edit ☐ Print ☐ Menu ☐ Inquire ☐ Action ☐ Run Routine ☐ Other
Associated Routine	
Option Definition	
Current Entry Action Logic	
Modified Entry Action Logic (Changes are in bold)	
Current Exit Action Logic	

Modified Exit Action Logic (Changes are in bold)

Option Name	HMP WB PTDEM
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change
Associated Menu Options, etc., that will invoke this reference	
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Menu Text Description	
Option Type	☐ Edit ☐ Print ☐ Menu ☐ Inquire ☐ Action ☐ Run Routine ☐ Other
Associated Routine	
Option Definition	
Current Entry Action Logic	
Modified Entry Action Logic (Changes are in bold)	
Current Exit Action Logic	
Modified Exit Action Logic (Changes are in bold)	

Option Name	HMP XU EVENTS
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change
Associated Menu Options, etc., that will invoke this reference	
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference

Menu Text Description	
Option Type	☐ Edit ☐ Print ☐ Menu ☐ Inquire ☐ Action ☐ Run Routine ☐ Other
Associated Routine	
Option Definition	
Current Entry Action Logic	
Modified Entry Action Logic (Changes are in bold)	
Current Exit Action Logic	
Modified Exit Action Logic (Changes are in bold)	

Option Name	HMPM ADD HMP PATIENT
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change
Associated Menu Options, etc., that will invoke this reference	
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Menu Text Description	
Option Type	☐ Edit ☐ Print ☐ Menu ☐ Inquire ☐ Action ☐ Run Routine ☐ Other
Associated Routine	
Option Definition	
Current Entry Action Logic	
Modified Entry Action Logic (Changes are in bold)	
Current Exit Action Logic	

Modified Exit Action Logic (Changes are in bold)

Option Name	HMPM ADD HMP USER
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change
Associated Menu Options, etc., that will invoke this reference	
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Menu Text Description	
Option Type	☐ Edit ☐ Print ☐ Menu ☐ Inquire ☐ Action ☐ Run Routine ☐ Other
Associated Routine	
Option Definition	
Current Entry Action Logic	
Modified Entry Action Logic (Changes are in bold)	
Current Exit Action Logic	
Modified Exit Action Logic (Changes are in bold)	

Option Name	HMPM EMERGENCY STOP
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change
Associated Menu Options, etc., that will invoke this reference	
Data Passing	☐ Input ☐ ☐ Both ☐ Global ☐ Local Reference

	Output Reference
Menu Text Description	
Option Type	☐ Edit ☐ Print ☐ Menu ☐ Inquire ☐ Action ☐ Run Routine ☐ Other
Associated Routine	
Option Definition	
Current Entry Action Logic	
Modified Entry Action Logic (Changes are in bold)	
Current Exit Action Logic	
Modified Exit Action Logic (Changes are in bold)	

Option Name	HMPM EXTRACT MONITOR
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change
Associated Menu Options, etc., that will invoke this reference	
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Menu Text Description	
Option Type	☐ Edit ☐ Print ☐ Menu ☐ Inquire ☐ Action ☐ Run Routine ☐ Other
Associated Routine	
Option Definition	
Current Entry Action Logic	
Modified Entry Action Logic (Changes are in bold)	

Current Exit Action Logic
Modified Exit Action Logic (Changes are in bold)

Option Name	HMPM RESTART FRESHNESS
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change
Associated Menu Options, etc., that will invoke this reference	
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Menu Text Description	
Option Type	☐ Edit ☐ Print ☐ Menu ☐ Inquire ☐ Action ☐ Run Routine ☐ Other
Associated Routine	
Option Definition	
Current Entry Action Logic	
Modified Entry Action Logic (Changes are in bold)	
Current Exit Action Logic	
Modified Exit Action Logic (Changes are in bold)	

Option Name	HMPMGR
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change
Associated Menu Options, etc., that will invoke this reference	

Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Menu Text Description	
Option Type	☐ Edit ☐ Print ☐ Menu ☐ Inquire ☐ Action ☐ Run Routine ☐ Other
Associated Routine	
Option Definition	
Current Entry Action Logic	
Modified Entry Action Logic (Changes are in bold)	
Current Exit Action Logic	
Modified Exit Action Logic (Changes are in bold)	

F. Appendix 6 - Security Keys

Security Key Name	HMP ADMIN	
Enhancement Category	☒ New ☐ Modify ☐ Delete ☐ No Change	
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Security Key Description		
Subordinate Keys		
Mutually Exclusive Keys		
Granting Condition Logic		
Current Logic		
Modified Logic (Changes are in bold)		
Hierarchical Precedence		

Security Key Name	HMP EXPERIMENTAL	
Enhancement Category	☒ New ☐ Modify ☐ Delete ☐ No Change	
Related Options		
Related Routines	**Routines "Called By"**	**Routines "Called"**
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference	
Security Key Description		
Subordinate Keys		
Mutually Exclusive Keys		

Granting Condition Logic	
Current Logic	
Modified Logic (Changes are in bold)	
Hierarchical Precedence	

G. Appendix 7 - Protocols

Protocol Name	HMP ADT-A04 CLIENT
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change
Associated Protocols	
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Item Text Description	N/A
Protocol Type	☐ Action ☐ Menu ☐ Protocol ☐ Protocol Menu ☐ Limited Protocol ☐ Extended Action ☐ Dialog ☐ Other
Associated Routine	
Current Entry Action Logic	
Modified Entry Action Logic (Changes are in bold)	
Current Exit Action Logic	
Modified Exit Action Logic (Changes are in bold)	

Protocol Name	HMP ADT-A08 CLIENT
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change
Associated Protocols	
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Item Text Description	N/A
Protocol Type	☐ Action ☐ Menu ☐ Protocol ☐ Protocol Menu ☐ Limited Protocol ☐ Extended Action ☐ Dialog

	☐ Other
Associated Routine	
Current Entry Action Logic	
Modified Entry Action Logic (Changes are in bold)	
Current Exit Action Logic	
Modified Exit Action Logic (Changes are in bold)	

Protocol Name	HMP APPT EVENTS
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change
Associated Protocols	
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Item Text Description	N/A
Protocol Type	☐ Action ☐ Menu ☐ Protocol ☐ Protocol Menu ☐ Limited Protocol ☐ Extended Action ☐ Dialog ☐ Other
Associated Routine	
Current Entry Action Logic	
Modified Entry Action Logic (Changes are in bold)	
Current Exit Action Logic	
Modified Exit Action Logic (Changes are in bold)	

Protocol Name	HMP DG UPDATES
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change
Associated Protocols	
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Item Text Description	N/A
Protocol Type	☐ Action ☐ Menu ☐ Protocol ☐ Protocol Menu ☐ Limited Protocol ☐ Extended Action ☐ Dialog ☐ Other
Associated Routine	
Current Entry Action Logic	
Modified Entry Action Logic (Changes are in bold)	
Current Exit Action Logic	
Modified Exit Action Logic (Changes are in bold)	

Protocol Name	HMP GMPL EVENT
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change
Associated Protocols	
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Item Text Description	N/A
Protocol Type	☐ Action ☐ Menu ☐ Protocol ☐ Protocol Menu ☐ Limited Protocol ☐ Extended Action ☐ Dialog ☐ Other

Associated Routine	
Current Entry Action Logic	
Modified Entry Action Logic (Changes are in bold)	
Current Exit Action Logic	
Modified Exit Action Logic (Changes are in bold)	

Protocol Name	HMP GMRA EVENTS
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change
Associated Protocols	
Data Passing	☐ Input ☐ Output ☐ **Both** ☐ Global Reference ☐ Local Reference
Item Text Description	N/A
Protocol Type	☐ Action ☐ Menu ☐ Protocol ☐ Protocol Menu ☐ Limited Protocol ☐ Extended Action ☐ Dialog ☐ Other
Associated Routine	
Current Entry Action Logic	
Modified Entry Action Logic (Changes are in bold)	
Current Exit Action Logic	
Modified Exit Action Logic (Changes are in bold)	

Protocol Name	HMP INPT EVENTS
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change
Associated Protocols	
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Item Text Description	N/A
Protocol Type	☐ Action ☐ Menu ☐ Protocol ☐ Protocol Menu ☐ Limited Protocol ☐ Extended Action ☐ Dialog ☐ Other
Associated Routine	
Current Entry Action Logic	
Modified Entry Action Logic (Changes are in bold)	
Current Exit Action Logic	
Modified Exit Action Logic (Changes are in bold)	

Protocol Name	HMP MDC EVENT
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change
Associated Protocols	
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Item Text Description	N/A
Protocol Type	☐ Action ☐ Menu ☐ Protocol ☐ Protocol Menu ☐ Limited Protocol ☐ Extended Action ☐ Dialog ☐ Other
Associated	

Routine	
Current Entry Action Logic	
Modified Entry Action Logic (Changes are in bold)	
Current Exit Action Logic	
Modified Exit Action Logic (Changes are in bold)	

Protocol Name	HMP NA EVENTS
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change
Associated Protocols	
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Item Text Description	N/A
Protocol Type	☐ Action ☐ Menu ☐ Protocol ☐ Protocol Menu ☐ Limited Protocol ☐ Extended Action ☐ Dialog ☐ Other
Associated Routine	
Current Entry Action Logic	
Modified Entry Action Logic (Changes are in bold)	
Current Exit Action Logic	
Modified Exit Action Logic (Changes are in bold)	

Protocol Name	HMP PATIENT APPT TRIGGER
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change
Associated Protocols	
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Item Text Description	N/A
Protocol Type	☐ Action ☐ Menu ☐ Protocol ☐ Protocol Menu ☐ Limited Protocol ☐ Extended Action ☐ Dialog ☐ Other
Associated Routine	
Current Entry Action Logic	
Modified Entry Action Logic (Changes are in bold)	
Current Exit Action Logic	
Modified Exit Action Logic (Changes are in bold)	

Protocol Name	HMP PCE EVENTS
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change
Associated Protocols	
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Item Text Description	N/A
Protocol Type	☐ Action ☐ Menu ☐ Protocol ☐ Protocol Menu ☐ Limited Protocol ☐ Extended Action ☐ Dialog ☐ Other
Associated	

Routine	
Current Entry Action Logic	
Modified Entry Action Logic (Changes are in bold)	
Current Exit Action Logic	
Modified Exit Action Logic (Changes are in bold)	

Protocol Name	HMP PCMM TEAM
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change
Associated Protocols	
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Item Text Description	N/A
Protocol Type	☐ Action ☐ Menu ☐ Protocol ☐ Protocol Menu ☐ Limited Protocol ☐ Extended Action ☐ Dialog ☐ Other
Associated Routine	
Current Entry Action Logic	
Modified Entry Action Logic (Changes are in bold)	
Current Exit Action Logic	
Modified Exit Action Logic (Changes are in bold)	

Protocol Name	HMP PCMM TEAM POSITION
Enhancement	☐ New ☐ Modify ☐ Delete ☐ No Change

Category	
Associated Protocols	
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Item Text Description	N/A
Protocol Type	☐ Action ☐ Menu ☐ Protocol ☐ Protocol Menu ☐ Limited Protocol ☐ Extended Action ☐ Dialog ☐ Other
Associated Routine	
Current Entry Action Logic	
Modified Entry Action Logic (Changes are in bold)	
Current Exit Action Logic	
Modified Exit Action Logic (Changes are in bold)	

Protocol Name	HMP PSB EVENTS
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change
Associated Protocols	
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Item Text Description	N/A
Protocol Type	☐ Action ☐ Menu ☐ Protocol ☐ Protocol Menu ☐ Limited Protocol ☐ Extended Action ☐ Dialog ☐ Other
Associated Routine	
Current Entry Action Logic	

Modified Entry Action Logic (Changes are in bold)
Current Exit Action Logic
Modified Exit Action Logic (Changes are in bold)

Protocol Name	HMP XQOR EVENTS
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change
Associated Protocols	
Data Passing	☐ Input ☐ Output ☐ Both ☐ Global Reference ☐ Local Reference
Item Text Description	N/A
Protocol Type	☐ Action ☐ Menu ☐ Protocol ☐ Protocol Menu ☐ Limited Protocol ☐ Extended Action ☐ Dialog ☐ Other
Associated Routine	
Current Entry Action Logic	
Modified Entry Action Logic (Changes are in bold)	
Current Exit Action Logic	
Modified Exit Action Logic (Changes are in bold)	

H. Appendix 8 - Remote Procedure Calls (RPC)

Name	HMP APPOINTMENTS

TAG^RTN	
Input Parameters	
Results Array	☐ Single Value ☐ Array ☐ Word Processing ☐ Global Array ☐ Global Instance
Description	

Name	HMP DATA VERSION
TAG^RTN	
Input Parameters	
Results Array	☐ Single Value ☐ Array ☐ Word Processing ☐ Global Array ☐ Global Instance
Description	

Name	HMP DELETE OBJECT
TAG^RTN	
Input Parameters	
Results Array	☐ Single Value ☐ Array ☐ Word Processing ☐ Global Array ☐ Global Instance
Description	

Name	HMP DELETE ROSTER
TAG^RTN	
Input Parameters	
Results Array	☐ Single Value ☐ Array ☐ Word Processing ☐ Global Array ☐ Global Instance
Description	

Name	HMP GET CHECKSUM
TAG^RTN	
Input Parameters	
Results Array	☐ Single Value ☐ Array ☐ Word Processing ☐ Global Array ☐ Global Instance

Description	

Name	HMP GET OBJECT
TAG^RTN	
Input Parameters	
Results Array	☐ Single Value ☐ Array ☐ Word Processing ☐ Global Array ☐ Global Instance
Description	

Name	HMP GET OPERATIONAL DATA
TAG^RTN	
Input Parameters	
Results Array	☐ Single Value ☐ Array ☐ Word Processing ☐ Global Array ☐ Global Instance
Description	

Name	HMP GET PATIENT DATA
TAG^RTN	
Input Parameters	
Results Array	☐ Single Value ☐ Array ☐ Word Processing ☐ Global Array ☐ Global Instance
Description	

Name	HMP GET PATIENT DATA JSON
TAG^RTN	
Input Parameters	
Results Array	☐ Single Value ☐ Array ☐ Word Processing ☐ Global Array ☐ Global Instance
Description	

Name	HMP GET REFERENCE DATA

TAG^RTN	
Input Parameters	
Results Array	☐ Single Value ☐ Array ☐ Word Processing ☐ Global Array ☐ Global Instance
Description	

Name	HMP GET ROSTER LIST
TAG^RTN	
Input Parameters	
Results Array	☐ Single Value ☐ Array ☐ Word Processing ☐ Global Array ☐ Global Instance
Description	

Name	HMP GET SOURCE
TAG^RTN	
Input Parameters	
Results Array	☐ Single Value ☐ Array ☐ Word Processing ☐ Global Array ☐ Global Instance
Description	

Name	HMP INPATIENTS
TAG^RTN	
Input Parameters	
Results Array	☐ Single Value ☐ Array ☐ Word Processing ☐ Global Array ☐ Global Instance
Description	

Name	HMP PATIENT ACTIVITY
TAG^RTN	
Input Parameters	
Results Array	☐ Single Value ☐ Array ☐ Word Processing ☐ Global Array ☐ Global Instance

Description	

Name	HMP PREVIEW ROSTER
TAG^RTN	
Input Parameters	
Results Array	☐ Single Value ☐ Array ☐ Word Processing ☐ Global Array ☐ Global Instance
Description	

Name	HMP PUT DEMOGRAPHICS
TAG^RTN	
Input Parameters	
Results Array	☐ Single Value ☐ Array ☐ Word Processing ☐ Global Array ☐ Global Instance
Description	

Name	**HMP PUT OBJECT**
TAG^RTN	
Input Parameters	
Results Array	☐ **Single Value** ☐ **Array** ☐ **Word Processing** ☐ **Global Array** ☐ **Global Instance**
Description	

Name	**HMP PUT PATIENT DATA**
TAG^RTN	
Input Parameters	
Results Array	☐ **Single Value** ☐ **Array** ☐ **Word Processing** ☐ **Global Array** ☐ **Global Instance**
Description	

Name	HMP ROSTER PATIENTS
TAG^RTN	
Input Parameters	
Results Array	☐ Single Value ☐ Array ☐ Word Processing ☐ Global Array ☐ Global Instance
Description	

Name	HMP ROSTERS
TAG^RTN	
Input Parameters	
Results Array	☐ Single Value ☐ Array ☐ Word Processing ☐ Global Array ☐ Global Instance
Description	

Name	HMP SUBSCRIBE
TAG^RTN	
Input Parameters	
Results Array	☐ Single Value ☐ Array ☐ Word Processing ☐ Global Array ☐ Global Instance
Description	

Name	HMP SUBSCRIBE ROSTERS
TAG^RTN	
Input Parameters	
Results Array	☐ Single Value ☐ Array ☐ Word Processing ☐ Global Array ☐ Global Instance
Description	

Name	HMP UPDATE ROSTER
TAG^RTN	
Input Parameters	
Results Array	☐ Single Value ☐ Array ☐ Word Processing ☐ Global Array ☐ Global Instance
Description	

Name	HMP WRITEBACK PT DEM
TAG^RTN	
Input Parameters	
Results Array	☐ Single Value ☐ Array ☐ Word Processing ☐ Global Array ☐ Global Instance
Description	

Name	HMPCORD RPC
TAG^RTN	
Input Parameters	
Results Array	☐ Single Value ☐ Array ☐ Word Processing ☐ Global Array ☐ Global Instance
Description	

Name	HMPCPAT RPC
TAG^RTN	
Input Parameters	
Results Array	☐ Single Value ☐ Array ☐ Word Processing ☐ Global Array ☐ Global Instance
Description	

Name	HMPCPRS RPC
TAG^RTN	
Input Parameters	
Results Array	☐ Single Value ☐ Array ☐ Word Processing ☐ Global Array ☐ Global Instance
Description	

Name	HMPCRPC RPC
TAG^RTN	
Input Parameters	
Results Array	☐ Single Value ☐ Array ☐ Word Processing ☐ Global Array ☐ Global Instance
Description	

Name	HMPCRPC RPCCHAIN
TAG^RTN	
Input Parameters	
Results Array	☐ Single Value ☐ Array ☐ Word Processing ☐ Global Array ☐ Global Instance
Description	

Name	HMPDJFS API
TAG^RTN	
Input Parameters	
Results Array	☐ Single Value ☐ Array ☐ Word Processing ☐ Global Array ☐ Global Instance
Description	

Name	HMPDJFS DELSUB
TAG^RTN	
Input Parameters	
Results Array	☐ Single Value ☐ Array ☐ Word Processing ☐ Global Array ☐ Global Instance
Description	

Name	HMPFPTC CHKS
TAG^RTN	
Input Parameters	
Results Array	☐ Single Value ☐ Array ☐ Word Processing ☐ Global Array ☐ Global Instance
Description	

Name	HMPFPTC LOG
TAG^RTN	
Input Parameters	
Results Array	☐ Single Value ☐ Array ☐ Word Processing ☐ Global Array ☐ Global Instance
Description	

I. Appendix 9 - Health Level 7 (HL7) Application Parameter

HL7 Application Parameter Name	HMP HL7	
Enhancement Category	☐ New ☐ Modify ☐ Delete ☐ No Change	
	Current	**Modified**
Application Status	☐ Active ☐ Inactive	☐ Active ☐ Inactive
Facility Name		
Country Code		
HL7 Field Separator		
HL7 Encoding Characters		
Mail Group		