**Inbound ePrescribing**

System Design Document



**June 2017**

Version 2.2

Department of Veterans Affairs

Office of Information and Technology (OI&T)

.

Revision History

| Date | Version | Description | Author |
| --- | --- | --- | --- |
| 06/20/2017 | 2.2 | Updates to Section 6.5, Figure 36, and Section 3.2, and Appendix A.3; updates to VistA screens in Section 8.4.1. | J. Wandling |
| 05/08/2017 | 2.1 | Updates to Section 6.2.2 for Options, Templates, Protocols and RPCs; added folder link to the current E&E Service Description; updated GUI and VistA screenshots; updated TRM links; updated for Section 508 compliance. | B. Fisher |
| 04/27/2017 | 2.0 | Updates to Section 6.2.2 to include Routines, Options, Templates, Protocols and RPCs; added VistA OP screenshots to Section 8.4.1; updated web-based GUI screenshots in Section 3.2; added VistA File Numbers to Section 5.6; removed references to PMAS and Milestone reviews; added List of Tables sand List of Figures. | B. Fisher |
| 02/20/2017 | 1.09 | Technical Updates to Software Used Versions and Document Links | J. Wandling, S. Butt |
| 02/08/2017 | 1.08 | Technical Updates DAS “Plan B” | J. Wandling, S. Butt |
| 06/17/2016 | 1.07 | TW edits throughout | L. Morahan |
| 05/24/2016 | 1.07 | Technical Updates | S. Butt |
| 04/04/2016 | 1.06 | Technical Updates DAS, Change Healthcare related | J. Wandling, S. Butt |
| 03/02/2016 | 1.05 | Technical Updates VistA OP related | J. Wandling, S. Butt, B. Fisher |
| 02/17/2016 | 1.04 | VistA OP design elements updated in Section 6 | J. Wandling, S. Butt, B. Fisher |
| 02/08/2016 | 1.03 | AERB Constraints Updates | J. Wandling, S. Butt |
| 01/15/2016 | 1.02 | Technical Updates | J. Wandling, S. Butt |
| 01/03/2016 | 1.01 | Updated Architecture Diagram Section 4 | J. Wandling |
| 12/31/2015 | 1.0 | Final revisions and edits | J. Wandling, S. Butt, B. Fisher, M. Murph, H. Trenkov |
| 12/2015 | 0.2 | Technical Updates: Sections 1, 2, 3, 4, 5, 6, 7 and 8 | J. Wandling, S. Butt, B. Fisher, M. Murph, H. Trenkov |
| 11/2015 | 0.1 | Initial Version Development | J. Wandling, S. Butt, B. Fisher, M. Murph, H. Trenkov |

Process Asset Library (PAL) (formerly ProPath) SDD Template 2.10 – June 2015

Artifact Rationale

The System Design Document (SDD) is a dual-use document that provides the conceptual design as well as the as-built design. This document will be updated as the product is built, to reflect the as-built product.

Table of Contents

[1. Introduction 1](#_Toc485810489)

[1.1. Scope 1](#_Toc485810490)

[1.1.1. Inbound ePrescribing Scope Inclusions 1](#_Toc485810491)

[1.1.2. Inbound ePrescribing Scope Exclusions 2](#_Toc485810492)

[1.2. User Profiles 2](#_Toc485810493)

[2. Background 3](#_Toc485810494)

[2.1. Overview of the System 3](#_Toc485810495)

[2.1.1. Business Benefits 4](#_Toc485810496)

[2.2. Overview of the Business Process 5](#_Toc485810497)

[2.2.1. Overview of Key Terms 5](#_Toc485810498)

[2.2.2. Overview of the Inbound eRx Process Flow 5](#_Toc485810499)

[2.3. Overview of the Significant Requirements 9](#_Toc485810500)

[2.3.1. Functional Requirements 9](#_Toc485810501)

[2.3.2. Functional Workload/Performance Requirements 10](#_Toc485810502)

[2.3.3. Operational Requirements 11](#_Toc485810503)

[2.3.4. Pivotal Technical Requirements 13](#_Toc485810504)

[2.3.5. Security and Privacy Requirements 14](#_Toc485810505)

[2.3.6. System Criticality and High Availability Requirements 15](#_Toc485810506)

[2.3.7. Single Sign-On Requirements 15](#_Toc485810507)

[2.3.8. Use of Enterprise Portals 16](#_Toc485810508)

[3. Conceptual Design 16](#_Toc485810509)

[3.1. Conceptual Application Design 16](#_Toc485810510)

[3.1.1. Application Context 16](#_Toc485810511)

[3.1.2. High-Level Application Design 17](#_Toc485810512)

[3.1.3. Application Locations 25](#_Toc485810513)

[3.2. Conceptual Data Design 26](#_Toc485810514)

[3.2.1. Pharma Data Formats 28](#_Toc485810515)

[3.2.2. High-Level eRx Processing Hub Logic (Example Transaction) 28](#_Toc485810516)

[3.2.3. Project Conceptual Data Model 29](#_Toc485810517)

[3.2.4. Database Information 30](#_Toc485810518)

[3.2.5. User Interface Data Mapping 30](#_Toc485810519)

[3.3. Conceptual Infrastructure Design 45](#_Toc485810520)

[3.3.1. System Criticality and High Availability 45](#_Toc485810521)

[3.3.2. Special Technology 45](#_Toc485810522)

[3.3.3. Technology Locations 48](#_Toc485810523)

[3.3.4. Conceptual Infrastructure Diagram 49](#_Toc485810524)

[4. System Architecture 50](#_Toc485810525)

[4.1. Hardware Architecture 52](#_Toc485810526)

[4.2. Software Architecture 52](#_Toc485810527)

[4.2.1. Components 53](#_Toc485810528)

[4.2.2. VistA OP Components 62](#_Toc485810529)

[4.3. Network Architecture 64](#_Toc485810530)

[4.4. Service Oriented Architecture / ESS 65](#_Toc485810531)

[4.5. Enterprise Architecture 66](#_Toc485810532)

[5. Data Design 66](#_Toc485810533)

[5.1. DBMS Files – eRx Processing Hub 66](#_Toc485810534)

[5.1.1. Physical Database Organization 67](#_Toc485810535)

[5.1.2. Physical Data Structures and Files 67](#_Toc485810536)

[5.1.3. NCPDP XML Data Analysis 68](#_Toc485810537)

[5.1.4. Query Patterns 69](#_Toc485810538)

[5.1.5. Indexing 69](#_Toc485810539)

[5.2. DBMS Files – VistA Outpatient Pharmacy 69](#_Toc485810540)

[5.3. Non-DBMS Files 70](#_Toc485810541)

[5.4. Data View 70](#_Toc485810542)

[5.4.1. eRx Processing Hub Tier 70](#_Toc485810543)

[5.4.2. VistA OP eRx Holding Queue (#52.49) 79](#_Toc485810544)

[5.4.3. eRx External Pharmacy File (#52.47) 83](#_Toc485810545)

[5.4.4. eRx External Patient File (#52.46) 83](#_Toc485810546)

[5.4.5. eRx External Person File (#52.48) 84](#_Toc485810547)

[5.4.6. eRx Service Reason Codes File (#52.45) 84](#_Toc485810548)

[6. Detailed Design 85](#_Toc485810549)

[6.1. Hardware Detailed Design 85](#_Toc485810550)

[6.2. Software Detailed Design 86](#_Toc485810551)

[6.2.1. Conceptual Design 86](#_Toc485810552)

[6.2.2. Specific Requirements 88](#_Toc485810553)

[6.3. Network Detailed Design 205](#_Toc485810554)

[6.4. Security and Privacy 205](#_Toc485810555)

[6.4.1. Security 205](#_Toc485810556)

[6.4.2. Privacy 207](#_Toc485810557)

[6.5. Service Oriented Architecture / ESS Detailed Design 208](#_Toc485810558)

[6.5.1. Service Description for MVI 208](#_Toc485810559)

[6.5.2. Service Description for E&E 208](#_Toc485810560)

[6.5.3. Service Design for Inbound eRx Process Hub Services 208](#_Toc485810561)

[7. External System Interface Design 211](#_Toc485810562)

[7.1. Interface Architecture 212](#_Toc485810563)

[7.2. Interface Detailed Design 212](#_Toc485810564)

[7.2.1. Connectivity Methods 212](#_Toc485810565)

[7.2.2. SSL Certificate 212](#_Toc485810566)

[7.2.3. Timeout Limits 212](#_Toc485810567)

[7.2.4. Two-way Mutual Certification Authentication 212](#_Toc485810568)

[7.2.5. Network Communications Header 212](#_Toc485810569)

[7.2.6. eRx XML Wrapper 213](#_Toc485810570)

[8. Human-Machine Interface 216](#_Toc485810571)

[8.1. Interface Design Rules 216](#_Toc485810572)

[8.2. Inputs 216](#_Toc485810573)

[8.3. Outputs 216](#_Toc485810574)

[8.4. Navigation Hierarchy 216](#_Toc485810575)

[8.4.1. Vista Outpatient Pharmacy eRx Holding Queue Screens 217](#_Toc485810576)

[A.1. Relationship to Other Documents and Plans 222](#_Toc485810577)

[A.2. Acronyms and Abbreviations 222](#_Toc485810578)

[A.3. Identification of Technology and Standards 225](#_Toc485810579)

[A.4. Constraining Policies, Directives and Procedures 226](#_Toc485810580)

[A.5. Requirements Traceability Matrix 226](#_Toc485810581)

[A.6. Packaging and Installation 227](#_Toc485810582)

[A.7. Design Metrics 227](#_Toc485810583)

List of Figures

[Figure 1: Overall Inbound ePrescribing Process 6](#_Toc485810584)

[Figure 2: Inbound eRx Application Context Diagram 16](#_Toc485810585)

[Figure 3: Inbound eRx High-Level Application Diagram 21](#_Toc485810586)

[Figure 4: Conceptual Data Flows and Designs 27](#_Toc485810587)

[Figure 5: eRx Message Processing Logic – Example 28](#_Toc485810588)

[Figure 6: Single Sign-On Login Screen 31](#_Toc485810589)

[Figure 7: Login Accept Screen 31](#_Toc485810590)

[Figure 8: Home Screen 32](#_Toc485810591)

[Figure 9: Pharmacy Management Screen 32](#_Toc485810592)

[Figure 10: Add Pharmacy Screen 33](#_Toc485810593)

[Figure 11: Track/Audit Screen (Search Criteria & Results) 36](#_Toc485810594)

[Figure 12: New eRx Detail Screen 37](#_Toc485810595)

[Figure 13: Error Detail Screen 40](#_Toc485810596)

[Figure 14: Verify Detail Screen 41](#_Toc485810597)

[Figure 15: User Management Screen 43](#_Toc485810598)

[Figure 16: Help Screen 44](#_Toc485810599)

[Figure 17: Reports Screen 44](#_Toc485810600)

[Figure 18: Example Flow with Multiple Threads for a Single Step (Row Denormalizer) 46](#_Toc485810601)

[Figure 19: Clustering in Pentaho Data Integration 47](#_Toc485810602)

[Figure 20: Conceptual Fail-Over PDI Cluster Architecture 48](#_Toc485810603)

[Figure 21: Conceptual Networks and Environments 49](#_Toc485810604)

[Figure 22: Conceptual Production String Diagram 50](#_Toc485810605)

[Figure 23: Inbound eRx System Architecture 51](#_Toc485810606)

[Figure 24: Inbound eRx – Incoming eRx Flow 54](#_Toc485810607)

[Figure 25: Inbound eRx – PatientEligibility Check Flow 55](#_Toc485810608)

[Figure 26: Inbound eRx – Prescription Provider Check Flow 56](#_Toc485810609)

[Figure 27: Inbound eRx – Prescription Drug Check Flow 57](#_Toc485810610)

[Figure 28: Inbound eRx – Pharmacy Transfer Flow 58](#_Toc485810611)

[Figure 29: Inbound eRx – VistAErxTransmission Flow 60](#_Toc485810612)

[Figure 30: Inbound eRx – eRx Status Transmission Flow 61](#_Toc485810613)

[Figure 31: Inbound eRx – Pharmacy Information Update Transmission Flow 62](#_Toc485810614)

[Figure 32: ErxWSCaller 63](#_Toc485810615)

[Figure 33: eRx Processing Hub (Manual) 64](#_Toc485810616)

[Figure 34: Network Architecture 65](#_Toc485810617)

[Figure 35: Conceptual Data Flow 66](#_Toc485810618)

[Figure 36: eRx Processing Hub Entity Relationship Diagram 70](#_Toc485810619)

[Figure 37: SiteMinder Authentication Flowchart 207](#_Toc485810620)

[Figure 38: eRx List View 217](#_Toc485810621)

[Figure 39: Search Action 218](#_Toc485810622)

[Figure 40: Search Action 218](#_Toc485810623)

[Figure 41: Sort Entries 219](#_Toc485810624)

[Figure 42: Sort Criteria 219](#_Toc485810625)

[Figure 43: Holding Queue Summary Screen 220](#_Toc485810626)

[Figure 44: Patient Validation Screen 220](#_Toc485810627)

[Figure 45: Provider Validation Screen 221](#_Toc485810628)

[Figure 46: Drug/SIG Validation Screen 221](#_Toc485810629)

**List of Tables**

[Table 1: User Characteristics 3](#_Toc485810630)

[Table 2: Functional Workload/Performance Requirements 10](#_Toc485810631)

[Table 3: Operational Requirements 12](#_Toc485810632)

[Table 4: Technical Requirements 13](#_Toc485810633)

[Table 5: Security and Privacy Requirements 15](#_Toc485810634)

[Table 6 (Grouping): Application Context Description 16](#_Toc485810635)

[Table 7: Objects in the High Level Application Design 21](#_Toc485810636)

[Table 8: Application Locations 26](#_Toc485810637)

[Table 9: Application Users 26](#_Toc485810638)

[Table 10: Database Inventory 30](#_Toc485810639)

[Table 11: eRx GUI Login Screen Description 31](#_Toc485810640)

[Table 12: eRx Login Accept Screen Description 31](#_Toc485810641)

[Table 13: eRx Home Screen Description 32](#_Toc485810642)

[Table 14: Pharmacy Management Screen Description (Pharmacy Table) 32](#_Toc485810643)

[Table 15: Add Pharmacy Screen Description 34](#_Toc485810644)

[Table 16: Track/Audit Search Criteria Screen Description (Search Criteria & Results) 36](#_Toc485810645)

[Table 17: View New eRx Detail Screen Description 37](#_Toc485810646)

[Table 18: Error Detail Screen Description 40](#_Toc485810647)

[Table 19: Verify Detail Screen Description 41](#_Toc485810648)

[Table 20: User Management Screen Description 43](#_Toc485810649)

[Table 21: Help Screen Description 44](#_Toc485810650)

[Table 22: Summary Report Screen Description 45](#_Toc485810651)

[Table 23: Special Technology Requirements 46](#_Toc485810652)

[Table 24: Technology Location Details 48](#_Toc485810653)

[Table 25: Inbound\_Ncpdp\_Msg Table Fields 71](#_Toc485810654)

[Table 26: Outbound\_Ncpdp\_MsgTable Fields 73](#_Toc485810655)

[Table 27: Pharmacy Table Fields 74](#_Toc485810656)

[Table 28: Erx\_Status Table Fields 77](#_Toc485810657)

[Table 29: Ncpdp\_Code Table Fields 77](#_Toc485810658)

[Table 30: Ecl\_CodeTable Fields 78](#_Toc485810659)

[Table 31: App\_Configuration Table Fields 78](#_Toc485810660)

[Table 32: Va\_User Table Fields 78](#_Toc485810661)

[Table 33: Development/SQA Detailed VM Requirements 85](#_Toc485810662)

[Table 34: Staging Detailed VM Requirements 85](#_Toc485810663)

[Table 35: Pre-Production Detailed VM Requirements 85](#_Toc485810664)

[Table 36: Production Detailed VM Requirements 86](#_Toc485810665)

[Table 37: Server Software List 86](#_Toc485810666)

[Table 38: Routine: PSOERX 88](#_Toc485810667)

[Table 39: Routine: PSOERX1 94](#_Toc485810668)

[Table 40: Routine: PSOERX1A 98](#_Toc485810669)

[Table 41: Routine: PSOERX1B 112](#_Toc485810670)

[Table 42: Routine: PSOERXA0 119](#_Toc485810671)

[Table 43: Routine: PSOERXA1 125](#_Toc485810672)

[Table 44: Routine: PSOERXA2 137](#_Toc485810673)

[Table 45: Routine: PSOERXA3 143](#_Toc485810674)

[Table 46: Routine: PSOERXD1 152](#_Toc485810675)

[Table 47: Routine: PSOERXH1 161](#_Toc485810676)

[Table 48: Routine: PSOERXO1 164](#_Toc485810677)

[Table 49: Routine: PSOERXP1 171](#_Toc485810678)

[Table 50: Routine: PSOERXR1 175](#_Toc485810679)

[Table 51: Routine: PSO467PST 179](#_Toc485810680)

[Table 52: Template: PSO ERX DRUG VALIDATION 182](#_Toc485810681)

[Table 53: Template: PSO ERX HOLDING QUEUE 182](#_Toc485810682)

[Table 54: Template: PSO ERX HQ DISPLAY 182](#_Toc485810683)

[Table 55: Template: PSO ERX PATIENT VALIDATION 183](#_Toc485810684)

[Table 56: Template: PSO ERX PROVIDER VALIDATION 183](#_Toc485810685)

[Table 57: Bulletins 184](#_Toc485810686)

[Table 58: Data Entries Affected by the Design 184](#_Toc485810687)

[Table 59: Unique Record ID 184](#_Toc485810688)

[Table 60: File or Global Size Changes 184](#_Toc485810689)

[Table 61: Mail Groups 185](#_Toc485810690)

[Table 62: Security Key: PSO ERX RPH 185](#_Toc485810691)

[Table 63: Security Key: PSO ERX VIEW 186](#_Toc485810692)

[Table 64: Security Key: PSO ERX TECH 187](#_Toc485810693)

[Table 65: Security Key: PSO ERX MGR 187](#_Toc485810694)

[Table 66: Security Key: PSO ERX PBM 188](#_Toc485810695)

[Table 67: Option: PSO ERX FINISH 189](#_Toc485810696)

[Table 68: Option: PSO RX 189](#_Toc485810697)

[Table 69: Option: PSO WEB SERVICES OPTION 190](#_Toc485810698)

[Table 70: Protocol: PSO ERX ACCEPT ERX 191](#_Toc485810699)

[Table 71: Protocol: PSO ERX ACCEPT VALIDATION 191](#_Toc485810700)

[Table 72: Protocol: PSO ERX DISPLAY MENU 191](#_Toc485810701)

[Table 73: Protocol: PSO ERX EDIT 192](#_Toc485810702)

[Table 74: Protocol: PSO ERX HOLD 193](#_Toc485810703)

[Table 75: Protocol: PSO ERX HQ MENU 193](#_Toc485810704)

[Table 76: Protocol: PSO ERX HQ SEARCH 194](#_Toc485810705)

[Table 77: Protocol: PSO ERX HQ SELECT 194](#_Toc485810706)

[Table 78: Protocol: PSO ERX HQ SORT 195](#_Toc485810707)

[Table 79: Protocol: PSO ERX PRINT 195](#_Toc485810708)

[Table 80: Protocol: PSO ERX REMOVE 196](#_Toc485810709)

[Table 81: Protocol: PSO ERX SELECT BY NUMBER 196](#_Toc485810710)

[Table 82: Protocol: PSO ERX UNHOLD 196](#_Toc485810711)

[Table 83: Protocol: PSO ERX VALIDATE DRUG 197](#_Toc485810712)

[Table 84: Protocol: PSO ERX VALIDATE PATIENT 197](#_Toc485810713)

[Table 85: Protocol: PSO ERX VALIDATE PROVIDER 198](#_Toc485810714)

[Table 86: Protocol: PSO ERX VALIDATION MENU 198](#_Toc485810715)

[Table 87: RPC: PSOERXA0 DRGMTCH 199](#_Toc485810716)

[Table 88: RPC: PSOERXA0 PRVMTCH 199](#_Toc485810717)

[Table 89: RPC: PSOERXA1 INCERX 200](#_Toc485810718)

[Table 90: Constants Defined in Interface 200](#_Toc485810719)

[Table 91: Variables Defined in Interface 200](#_Toc485810720)

[Table 92: Types Defined in Interface 200](#_Toc485810721)

[Table 93: GUI 200](#_Toc485810722)

[Table 94: GUI Classes 201](#_Toc485810723)

[Table 95: Components on Form 201](#_Toc485810724)

[Table 96: Events 201](#_Toc485810725)

[Table 97: Methods 201](#_Toc485810726)

[Table 98: Class Events 202](#_Toc485810727)

[Table 99: Class Methods 202](#_Toc485810728)

[Table 100: Class Properties 202](#_Toc485810729)

[Table 101: Functions 202](#_Toc485810730)

[Table 102: Dialog 203](#_Toc485810731)

[Table 103: Help Frame 203](#_Toc485810732)

[Table 104: HL7 Application Parameter 204](#_Toc485810733)

[Table 105: HL7 Logical Link 204](#_Toc485810734)

[Table 106: COTS Interface 205](#_Toc485810735)

[Table 107: Header Segment 213](#_Toc485810736)

[Table 108: eRx Segment 215](#_Toc485810737)

[Table 109: Body Segment 215](#_Toc485810738)

# Introduction

Pharmacy applications and systems are some of the oldest technologies in the Veterans Health Information Systems and Technology Architecture (VistA) system. The Veterans Health Administration (VHA) has identified system limitations and cumbersome, inconsistent pharmacy processes as a weakness in its ability to provide efficient pharmaceutical services across the VHA continuum. The Department of Veterans Affairs (VA) approved the Pharmacy Reengineering (PRE) project to address several fundamental problems with the current system. The objective of the overall PRE project was to facilitate the improvement of pharmacy operations, customer service, and patient safety for VHA.

VHA, Patient Care Services (PCS), and Pharmacy Benefits Management (PBM) has requested a new capability as part of the PRE program to receive inbound electronic prescriptions (e-prescriptions or eRxs) from an external provider (e.g., a doctor not associated with VA, medical staff at a Department of Defense [DoD] military treatment facility, etc.). Once received, these prescriptions will then be fed into the existing VistA Outpatient Pharmacy (OP) for processing and dispensing.

## Scope

This document focuses on the design of Inbound ePrescribing (eRx) web-based application and enhancements to the VistA OP module, including the design of a new VistA OP Inbound eRx Holding Queue.

The purpose of this document is to describe the design of the new web-based application, the design of the new VistA OP Inbound eRx Holding Queue, and any modifications to other VistA modules, where applicable. This System Design Document (SDD) translates the requirement specifications into a document from which the developers can create the actual system. It identifies the top-level system architecture, and identifies hardware, software, communication, and interface components relevant to the affected application and modules.

The high-level scope of this project (as well as what is not included in the scope) is indicated below, with further detail outlined in the *Inbound ePrescribing Requirements Specification Document (RSD)*.

### Inbound ePrescribing Scope Inclusions

The Inbound eRx project’s scope includes the following:

* Receiving and processing inbound eRxs, where “inbound” refers to the ordering of medication or medical related supplies for a VA patient by a non-VA provider to be filled at a VA pharmacy.
* Electronically receiving and processing outpatient prescriptions only (includes prescriptions created for a VA patient upon discharge from a non-VA hospital to be filled on an outpatient basis by a VA pharmacy).
* Receiving and processing inbound eRxs from non-VA providers for controlled substances.
* Receiving and processing inbound eRxs from non-VA providers that currently prescribe medications and medical related supplies for Civilian Health and Medical Program of the VA (CHAMPVA) beneficiaries and which are currently handled by the Medications by Mail (MbM)[[1]](#footnote-2) program.
* Electronically receiving and processing cancellations and changes for previously received eRxs from non-VA providers. (**NOTE:** This functionality will not be delivered in Version 2.0.)
* Electronically sending refill requests to non-VA providers who previously sent eRxs to VA pharmacies, receiving responses from the non-VA providers, and processing those responses (i.e., eRx refill requests initiated from VA Pharmacy to a non-VA provider).
* Electronically receiving and processing refill requests from non-VA providers who previously sent eRxs to VA pharmacies.
* Electronically receiving and processing medication history[[2]](#footnote-3) requests initiated by any entity (e.g., non-VA provider or VA provider, Personal Health Record [PHR] system, or Electronic Medical Record [EMR] system, etc.) to a VA Pharmacy, as well as electronically sending medication history requests from a VA Pharmacy to any entity that supports receiving and processing medication history requests. (**NOTE:** This functionality will not be delivered in Version 2.0.)
* Sending outbound electronic notifications from a VA pharmacy (that received an inbound eRx) to the non-VA provider that originally sent the eRx.
* Electronic transfers of prescriptions to VA pharmacies from VA pharmacies. (**NOTE:** This functionality will not be delivered in Version 2.0.)

### Inbound ePrescribing Scope Exclusions

The Inbound eRx effort scope does not include:

* VA providers generating Rxs at one VAMC location to be electronically transmitted to and processed by (filled, dispensed, etc.) a different VAMC location’s pharmacy.
* Initiating outbound eRxs (generation of an eRx by a VA provider to be filled at a non-VA pharmacy).
* Electronic receipt and processing of any VA and non-VA inpatient medication orders.
* Electronic receipt and processing of any VA and non-VA orders for Durable Medical Equipment (DME), such as wheel chairs.
* Electronic receipt and processing of eRx refill requests from a VA patient’s non-VA PHR system.

## User Profiles

The users of the Inbound ePrescribing application have been categorized into primary and secondary users. Users can easily request prescriptions or medical supplies to be electronically transmitted from non-VA providers and pharmacies to VA pharmacies for processing.

The table below lists the primary and secondary users and their responsibilities.

Table 1: User Characteristics

| Type of User | Description | Responsibilities |
| --- | --- | --- |
| Primary Users | VA Pharmacists  VA Pharmacy Technicians/Assistants  Change Healthcare | Can view inbound eRxs received from an external provider  Can manually validate an inbound eRx  Can transfer existing eRxs to another VA pharmacy and receive electronically transferred eRxs from another VA pharmacy (**NOTE:** This functionality will not be delivered in Version 2.0.)  Can send prescriptions and medication history requests (**NOTE:** This functionality will not be delivered in Version 2.0.) |
| Secondary Users | VA Pharmacy Managers  PBM personnel | View reports on inbound eRx |
| Secondary Users | Non-VA Providers | Send/transmit inbound eRx and associated requests to VA  View inbound eRx notifications from VA  Respond to eRx requests (e.g., a refill request) regarding inbound eRxs from VA |
| Secondary Users | VA Administrators  Support Staff | Full Control  Turn on/off a pharmacy from receiving inbound eRxs |

# Background

## Overview of the System

The Inbound eRx system aims to meet the business needs identified in the Inbound eRx Business Requirements Document (BRD) titled *Pharmacy Reengineering Inbound ePrescribing Work Effort Unique Identifying #20140401*. To meet those needs, the Inbound eRx system will:

* Allow external providers to select VA pharmacies that support inbound electronic prescriptions.
* Receive electronic prescriptions from non-VA providers to be processed by a VA Pharmacy.
* Allow VA pharmacy staff to view and interact with received inbound electronic prescriptions after they have gone through an automatic validation process by the system.
* Allow VA pharmacy staff to process (manually validate, manage, dispense) inbound electronic prescriptions.
* Provide tracking and auditing functions for the inbound eRx capability.
* Help standardize terminology for all medications at the VA by implementing and using the National Council for Prescription Drug Programs (NCPDP) SCRIPT Standard.
* Reduce time from when an Rx is written or created by a non-VA provider to when it is entered into or received by VistA OP.
* Increase satisfaction of Veterans/VA beneficiaries, VA pharmacy personnel, and non-VA providers.

### Business Benefits

A typical scenario of what happens today is a Veteran visits a non-VA provider specialist who hand-writes an Rx for a medication and gives it to the Veteran. Veterans then physically visits their VA pharmacy and submit the paper Rx to be processed and dispensed by that facility to optimize their VA benefits that offer them low, or no, copays. The VA pharmacy dispenses the Rx and the Veteran picks it up. Issues may arise in this scenario, such as the VA pharmacy staff misinterpretation of the provider’s hand-written Rx, which could lead to the wrong medication being dispensed, a dosing error, or other patient safety issues. It is also time consuming and inconvenient for Veterans to physically visit their VA pharmacy in order to submit the Rx, and inefficient for the VA pharmacy staff to manually input the Rx information into VistA OP so that it can be processed.

Additionally, today, VA pharmacies cannot accept transferred Rxs from other pharmacies nor can they transfer prescriptions to other pharmacies. This presents an inconvenience for Veterans who travel or move frequently in that they must visit a provider in order to get a new Rx so that they can take the new Rx to a different pharmacy for dispensing.

Providing the capability to receive and process Inbound eRxs from outside sources and providers will improve the effectiveness and efficiency with which health care benefits are delivered to VA beneficiaries. Some of the high-level business benefits provided by the Inbound ePrescribing system include:

* **Improved efficiency:** The Inbound ePrescribing system leads to more efficient use of VA pharmacy resources and non-VA provider resources. For example: fewer transcribing/translation errors and clear/error-free communications equates to time savings by not having to communicate back and forth regarding the content of a prescription.
* **Improved Veteran/beneficiary satisfaction:** The Inbound ePrescribing system makes an existing manual process easier, more efficient, and more effective through automation of the prescription process. This automation helps improve the satisfaction of the Veteran community by:
* Reducing the risk of losing paper Rxs
* Enabling more secure communication of Rx data

And most importantly:

* Providing timelier dispensing of Rxs prescribed by non-VA providers
* **Improved patient safety:** The Inbound ePrescribing not only improves on the manual process from an efficiency standpoint, but from a safety one as well. By automating the process the risk of interpreting hand-written or transcribed prescriptions is drastically reduced, while Veteran health-safety is improved.
* **Improved data accuracy:** The Inbound ePrescribing system provides enhanced functionality within VistA OP by improving the accuracy and potential use of the data it collects. By automating data transmission from providers to the VA (and between other pharmacies), the need for VA pharmacy personnel to manually input Rx data from non-VA providers is largely eliminated, reducing the chance for data to be entered incorrectly or missed.

## Overview of the Business Process

The Inbound ePrescribing system is intended to provide the ability for the VA to receive and process eRxs from external to VA providers. Inbound eRx prescription data encompasses incoming to the VA prescription messages, as well as incoming/outgoing status, error and verification messages as defined in the NCPDP Script 10.6 XML.

### Overview of Key Terms

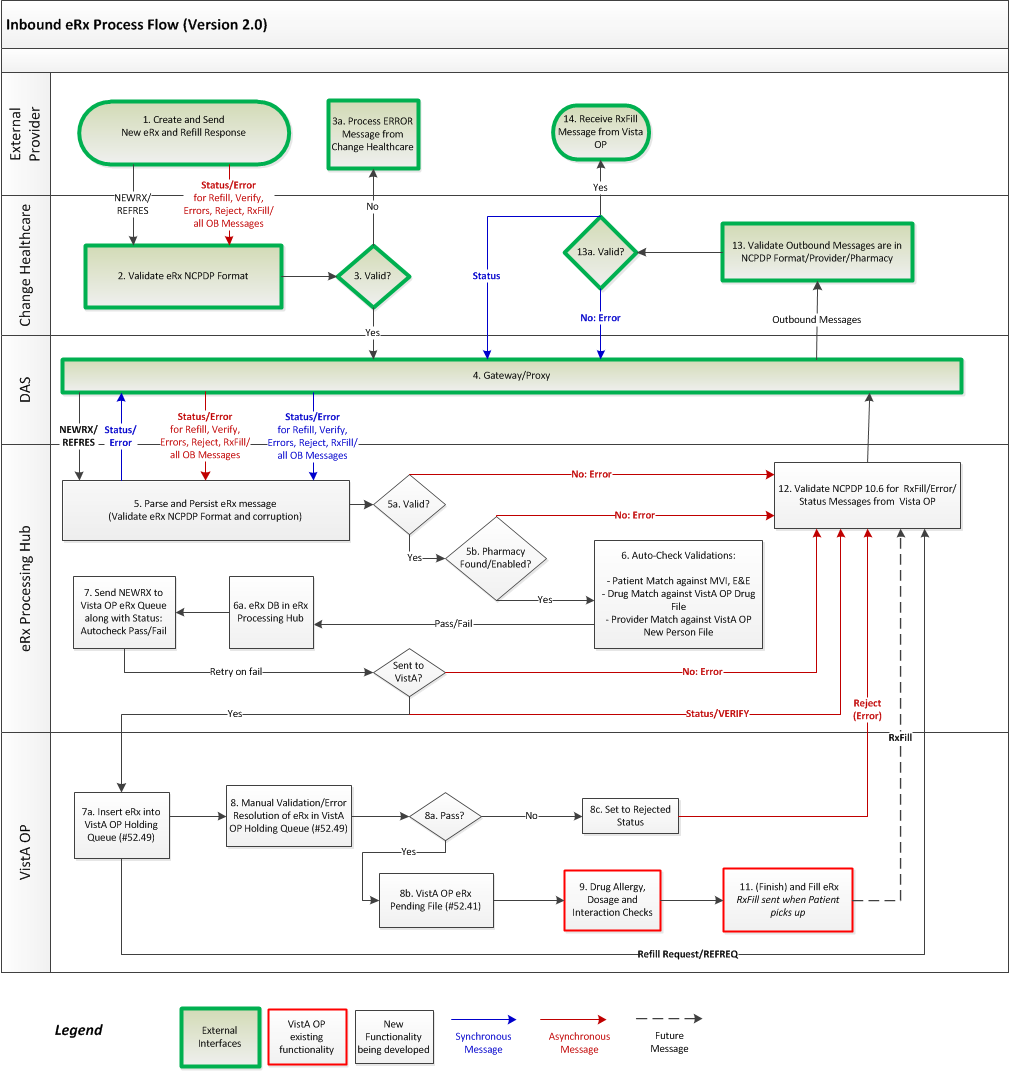
This section contains key terms used to describe the Inbound ePrescribing process, as well as other aspects of the solution as depicted in this document. This is not the complete list of terms and is rather a sub-set of the most widely used ones:

* **Inbound eRx or Inbound ePrescribing** – refers to the end-to-end solution and process from external provider-to-fulfilled prescription
* **Clearinghouse** – refers to Change Healthcare’s pharmacy data exchange
* **Network Gateway** – refers to VA’s Data Access Service (DAS) infrastructure proxies messages between Change Healthcare and the eRx Processing Hub
* **eRx Processing Hub** – refers to the main processing component of the Inbound eRx solution; leverages Pentaho Data Integration (PDI) and custom web services
* **eRx Processing Table** – refers to the data container in the eRx Processing Hub storing all inbound and outbound pharmacy messages and their statuses
* **eRx UI** – refers to the eRx web application providing eRx tracking, pharmacy enable/disable and reports
* **eRx Holding Queue** – the Inbound eRx staging file in VistA OP for processing auto-validated prescriptions from external providers

### Overview of the Inbound eRx Process Flow

A high-level overview of the Inbound eRx process for pharmacy data messages is outlined in Figure 1 below.

Figure 1: Overall Inbound ePrescribing Process



The Inbound eRx Process Flow depicts five (5) swim-lanes – one external to Inbound eRx (External Provider) and four (4) Inbound eRx processing tiers (Change Healthcare, DAS, eRx Processing Hub and VistA OP):

* (External to Inbound eRx) External Provider:
  + External to VA physicians who, with the use of a 3rd party Electronic Medical Records (EMR) software, issue a prescription for a Veteran
  + The EMR system is registered with Change Healthcare and is responsible for creating and sending the eRx in NCPDP XML format
  + The external physician is registered with Change Healthcare and their provider information (e.g., NPI number) is known and verified by Change Healthcare
* (Tier 1) Change Healthcare:
  + Serve as proxy for all messages between the external providers and the VA infrastructure (i.e., the DAS/eRx Processing Hub)
  + Support and validate NCPDP XML format and structure
* (Tier 2) DAS:
  + Serve as secured-layer gateway/proxy for all messages (in NCPDP XML) between Change Healthcare and the eRx Processing Hub
* (Tier 3) eRx Processing Hub
  + Receive, persist, validate, manipulate, construct/send NCPDP XMLs
  + Validate designated pharmacy from NCPDP XML and match to VistA OP instance.
  + Perform auto-validation and matching (including patient, enrollment/registration, provider, drug)
  + Transform prescription data and send to VistA eRx Holding Queue format
  + Maintain processing statuses and perform error/exception handling
  + Provide admin UI to track, enable/disable transmission, validate eRx and run reports
* (Tier 4) VistA OP (1-n)
  + Provide UI for eRx Users (manual steps, review)
  + Process eRx Holding Queue transactions (including drug-to-drug, PAs, DEA rules, dosage, drug dispensation, order processing, prescription file and fulfillment)
  + Once the eRx is validated it will be processed into the PENDING OUTPATIENT ORDERS (#52.41) file
  + Perform error handling and status updating in eRx Holding Queue (and the eRx Processing Hub processing statuses)

The Inbound eRx processing flow is sequential in nature as depicted in Figure 1 (above):

* Step 1: The Inbound eRx process flow begins with the External Provider, using their EMR system, creates and sends to Change Healthcare eRx message data in NCPDP XML format
* Step 2: In Change Healthcare (Tier 1), the eRx message is validated against the NCPDP Format to ensure that message is in valid construct without any corruption
* Step 3: If the message is valid, Change Healthcare (Tier 1) routes the message to the VA infrastructure via DAS (Tier 2) for further processing
  + Step 3a: If the eRx message is invalid, Error message is sent back to the external provider (as per the NCPDP specifications) without sending the message to the VA
* Step 4: DAS (Tier 2) proxies the message to the eRx Processing Hub (Tier 3)
* Step 5: The eRx Processing Hub (Tier 3) validates the NCPDP XML format to ensure that the message is in valid construct without any corruption, and “shreds” the XML into its data elements, structure and stores the message; the message is recorded in a Transaction/Processing table which tracks the processing status of the message, as well as the coordinates auto-validations (step 6) and the synchronization (steps 7 and 11) with VistA OP instance (Tier 4)
* Step 6: The eRx Processing Hub (Tier 3) performs patient, provider and drug auto-validations – once all auto-validations are completed and the eRx is ready to be sent to VistA OP (step 7), the Transaction/Processing table prescription record in the eRx Processing Hub (Tier 3) is updated to reflect (a) successful auto-validation, (b) failed auto-validation (i.e., patient match was unsuccessful), or (c) partial auto-validation (i.e., patient match was successful, but other auto-validation(s) failed); furthermore, this steps identifies the respective VistA OP instance the message needs to be sent to
  + The VA pharmacy needs to be enabled to accept Inbound eRx or an error message will be sent to the External Provider – for simplicity purposes, this uncommon scenario is not explicitly depicted in the Inbound eRx process flow
* Step 7: The eRx Processing Hub (Tier 3) constructs the eRx data into the format of the eRx Holding Queue of the respective VistA OP (1-n) instance (Tier 4)
* Step 8: In the respective VistA OP instance (tier 4), pharmacy personnel performs manual validation and eRx Processing Hub (tier 3) auto-validation error resolution of the eRx (e.g., patient match, eligibility, drug dosage match, prior authorization received)
  + Any change to the status of the eRx in VistA OP (Tier 4) is synchronized with the Transaction/Processing table in the eRx Processing Hub (Tier 3); errors (and some successes) initiate NCPDP messages in the eRx Processing Hub (Tier 3) to be sent back to the External Provider
* Step 9: Upon successful patient and drug validation, additional auto-validation checks in VistA OP (Tier 4) occur (e.g., drug-to-drug); successful and unsuccessful auto-validations update the eRx Holding Queue transaction status
  + Any change to the status of the eRx in VistA OP (Tier 4) is synchronized with the Transaction/Processing table in the eRx Processing Hub (Tier 3); errors (and some successes) initiate NCPDP messages in the eRx Processing Hub (Tier 3) to be sent back to the External Provider
* Step 10: Once all the validations are completed successfully, the prescription is fulfilled in VistA OP (Tier 4) based on the existing fulfillment routines
* Step 11: Upon success, VistA OP (Tier 4) updates the eRx Processing Hub (Tier 3) and triggers a message update to the External Provider
* Step 12: In the eRx Processing Hub (Tier 3), success (and error) status updates from VistA OP (Tier 4) trigger message(s) to the External Provider; those messages (in NCPDP XML format) are sent through DAS (Tier 2) to Change Healthcare (Tier 1) and ultimately to the External Provider
* Step 13: Change Healthcare (Tier 1) validates the received message against the NCPDP Format to ensure that the message is in valid construct without any corruption, and sends it to the External Provider
* Step 14: The Inbound eRx process flow ends with the External Provider receiving the message update from VA

Technical details of the Inbound eRx processing architecture, data flow, exception/error handling, monitoring and logging, user interfaces, and reporting are included in Sections 3, 4, 5, 6, 7 and 8 of this document.

## Overview of the Significant Requirements

### Functional Requirements

The functional business requirements identified correspond to the original business requirements, as documented in the corresponding RSD for this effort*.* For purposes of design and architecture, this set of requirements expands and elaborates upon the original requirements, and provides sequential detail:

* From the initiation of an eRx  inbound to a VA Pharmacy
* To the receipt of the eRx within the VA Pharmacy
* To the validation and processing of the eRx within the VA Pharmacy
* To the dispensing of the eRx to the Veteran Patient.

In support of this, new requirements were added to fill gaps where additional functionality was needed within the inbound ePrescribing system. These requirements were developed hierarchically, starting at the top with:

* An **EPIC** overarching major requirement
  + To a **User Story** who, what, and why requirement
    - To the **Tasks** that support that user story requirement.
      * **Business Acceptance Criteria** needed to support that requirement

Using the NCPDP script standard 10.6, along with analogous system functionality found in the Defense Health Agency’s (DHA) Composite Health Care System (CHCS), as guides, the functional requirements detailed in the RSD overview the major functional needs regarding:

* Patient, drug and external provider validation;
* Script routing through the new system; and,
* Interface/coordination with any existing functionality within VistA OP.

Note that some areas may require additional elaboration (e.g. requirements regarding the transfer of eRxs from external pharmacies), but the RSD extensively covers the baseline process of a new inbound eRx from an external provider, as well as the majority of the alternative flows that were identified. Please refer to the following high-level requirements document on the Technical Services Project Repository (TSPR):

* *Inbound eRx Requirements Specification Document:*
* *Inbound eRx Business Requirements Document***:**

**NOTE:** As of May 26, 2017, TSPR is down until further notice. Please refer to the following SharePoint links for access to the above documents. For SharePoint access requests, please contact

* *Inbound eRx Requirements Specification Document:*
* *Inbound eRx Business Requirements Document***:**

### Functional Workload/Performance Requirements

The following Inbound ePrescribing functional workload/performance requirements are detailed in the corresponding RSD for this effort.

Table 2: Functional Workload/Performance Requirements

| ID from BRD | Requirement |
| --- | --- |
| NONF2811 | The Inbound ePrescribing system shall provide the capability to measure the reporting requirements for Responsiveness, Capacity, and Availability. |
| NONF2812 | The actual system performance data shall be displayed to the Information Technology (IT) Performance Dashboard for customers and IT staff to view. |
| New | The Inbound ePrescribing system shall provide operating capacity100% of the time, 24 hours/day, 365 days/year to support inbound electronic prescription transactions |
| New | The Inbound ePrescribing system response times and page load times shall be consistent with VistA OP standards. |
| New | The Inbound ePrescribing system shall support the different National Council for Prescription Drug Programs (NCPDP) SCRIPT transaction types. |
| New | The Inbound ePrescribing system shall support new, refilled, and partially-filled prescriptions. |
| New | The Inbound ePrescribing system shall provide the capability to process current and future inbound electronic prescription transactions |
| New | The Inbound ePrescribing system shall provide the capability to capture the inbound electronic prescription transaction speed. |
| New | The Inbound ePrescribing system shall provide the capability to capture data retrieval times from the Holding Queues. |
| New | The Inbound ePrescribing system shall provide the capability to capture the dates and times of the inbound electronic prescription. |
| New | The Inbound ePrescribing system shall provide the capability to capture page refresh times. |
| New | The Inbound ePrescribing system shall provide the capability to capture page load times. |
| New | The Inbound ePrescribing system shall provide the capability to display performance data using different report formats. |
| New | The Inbound ePrescribing system shall provide the VA user with the ability to view performance summary data. |
| New | The Inbound ePrescribing system shall provide the VA user with the ability to view performance data in details by category. |
| New | The Inbound ePrescribing system shall provide the VA user with the ability to select tabular and graphical data views of the performance data. |
| New | The Inbound ePrescribing system shall provide the capability to capture message response times. |
| New | The Inbound ePrescribing system shall support the existing VistA OP users. Current Vista OP user count is 12,300 users (8,000 pharmacists and 4,300 pharmacy technicians) |
| New | The system is expected to support 1,000 concurrent users |
| New | Response time:  Running queries: five seconds or less  Running reports: five seconds or less  Synchronous web service response: 27 seconds or less  Transmit eRx to VistA OP: 5 minutes or less |
| New | Usage peak times: Monday through Friday, 8:00 a.m. Eastern Time – 9:00 p.m. Eastern Time |
| New | Raw single eRx message data size is estimated to be 100 KB |
| New | Total raw message data storage is estimated to be (100 KB \*4M)/106 = 400GB per year |

### Operational Requirements

The following Inbound ePrescribing operational requirements are detailed in the corresponding RSD/BRD for this effort.

Table 3: Operational Requirements

| ID from BRD | Requirement |
| --- | --- |
| NONF3297 | The Inbound ePrescribing system response times and page load times shall be consistent with VistA Outpatient Pharmacy (OP) standards. |
| NONF1780 | Maintenance, including maintenance of externally developed software incorporated into the VistA OP application, must be scheduled during off peak hours or in conjunction with relevant maintenance schedules. The business owner should provide specific requirements for establishing system maintenance windows when planned service disruptions can occur in support of periodic maintenance. |
| NONF1608 | The Inbound ePrescribing system shall disseminate unscheduled system outages or other events that impact the response time to the user community within 30 minutes of the occurrence. |
| NONF1609 | The Inbound ePrescribing system shall provide a real-time monitoring solution to report agreed/identified critical system performance parameters agreed/identified critical system performance parameters. |
| NONF2820 | Critical business performance parameters shall be captured and identified to support metric reporting, the performance meters are displayed to the Office of Information and Technology (OI&T) Performance Dashboard, and the system solution shall comply with all Service Level Agreements (SLAs). |
| NONF2811 | The Inbound ePrescribing system implementation includes Disaster Recovery (DR) procedures and 24/7 support consistent with organizationally established expectations relative to system availability. |
| NONF3314 | The Inbound ePrescribing solution provides a contingency plan that includes procedures for disable and enables receiving inbound electronic prescriptions. |
| NONF3213 | The Inbound ePrescribing system shall provide data protection measures, such as back-up intervals and redundancy that is consistent with systems categorized as mission critical (12 hour restoration, 2 hour recover point objective). |
| NONF1610 | Notification of scheduled maintenance periods that require the service to be offline or that may degrade system performance shall be disseminated to the business user community a minimum of 48 hours prior to the scheduled event. |
|  | The Inbound ePrescribing solution allows functional requirements to be integrated into the VistA Outpatient Pharmacy (OP) system without changes to the hardware. |
| NONF2229 | A monitoring process shall be provided to ensure that data is accurate and up-to-date and provides accurate alerts for malfunctions while minimizing false alarms. |
|  | Maintenance, including maintenance of externally developed software incorporated into the Inbound ePrescribing utilities use system (contingency/failover, etc.) redundancy to ensure that scheduled maintenance does not cause system down time. |
| NONF1615 | The Inbound ePrescribing solution provides a back-up and data recovery process for when the system is brought off-line for maintenance or technical issues/problems off-line. |
| NONF3085 | Technical Help Desk support for the application shall be provided for VA users to obtain assistance with receiving and processing inbound eRxs, and sending and receiving eRx transfers. |

### Pivotal Technical Requirements

Table 4: Technical Requirements

| ID from BRD/RSD | Requirement |
| --- | --- |
| EPIC 20 | Auto-validation in Inbound eRx is based on precise determination of pass/fail criteria – i.e., match or no-match; Patient identity match (MVI match) needs to pass successfully (i.e., exact match) before any further auto-checks (e.g., provider, registration, drug) can be performed; Patient mandatory attributes for matching (NCPDP required fields) are: first name, last name, DOB and gender; eRx Processing Hub will perform incrementally narrowing criteria until an exact (i.e., one) match or no-match occurs |
| OWNR170 | Provide the ability to express all content using nationally recognized reference and authoritative terminology standards (e.g., Logical Observation Identifiers, Names, and Codes [LOINC], Systematized Nomenclature of Medicine Clinical Terms [SNOMED CT], RxNORM, etc.). |
| OWNR13409 | Provide the ability for external providers to specify the VA pharmacy to which they can send an electronic prescription. |
| OWNR13410 | Provide the ability for a VA Pharmacy to initiate receiving inbound electronic prescriptions (i.e., turn “on”) and the ability to stop receiving inbound electronic prescriptions (i.e., turn “off”). |
| OWNR13411 | Provide the ability for a VA Pharmacy to receive and respond to an electronically transmitted new prescription for medication (including controlled substances) in accordance with the most current version supported of the NCPDP SCRIPT Standard (10.6 at minimum) from an external provider. | |
| OWNR13412 | Provide the ability for a VA Pharmacy to receive and respond to an electronically transmitted new prescription for expendable medical supplies (other than durable medical equipment) in accordance with the most current version supported of the NCPDP SCRIPT Standard (10.6 at minimum) from an external provider. | |
| OWNR13413 | Provide the ability for a VA Pharmacy to send an electronically transmitted prescription change request in accordance with the most current version supported of the NCPDP SCRIPT Standard (10.6 at minimum) to an external provider and receive an associated response in relation to a previously received eRx from this provider.  **NOTE: This is currently out of scope for Version 2.0.** | |
| OWNR13414 | Provide the ability for a VA Pharmacy to receive and respond to an electronically transmitted prescription cancellation request in accordance with the most current version supported of the NCPDP SCRIPT Standard (10.6 at minimum) from an external provider in relation to a previously received eRx from this provider.  **NOTE: This is currently out of scope for Version 2.0.** | |
| OWNR13415 | Provide the ability for a VA Pharmacy to send an electronically transmitted prescription refill request in accordance with the most current version supported of the NCPDP SCRIPT Standard (10.6 at minimum) to an external provider and receive an associated response for a previously received eRx from this provider. | |
| OWNR13416 | Provide the ability for a VA Pharmacy to receive and respond to an electronically transmitted prescription refill request in accordance with the most current version supported of the NCPDP SCRIPT Standard (10.6 at minimum) from an external provider in relation to a previously received eRx from this provider. | |
| OWNR13417 | Provide the ability for a VA Pharmacy to receive and respond to an electronically transmitted medication history request in accordance with the most current version supported of the NCPDP SCRIPT Standard (10.6 at minimum) from an external entity.  **NOTE: This is currently out of scope for Version 2.0.** | |
| OWNR13418 | Provide the ability for a VA Pharmacy to send an electronically transmitted medication history request in accordance with the most current version supported of the NCPDP SCRIPT Standard (10.6 at minimum) to an external entity and receive an associated response.  **NOTE: This is currently out of scope for Version 2.0** | |
| OWNR13419 | Provide the ability for a VA Pharmacy to send electronic notifications in accordance with the most current version supported of the NCPDP SCRIPT Standard (10.6 at minimum) to an external provider that inbound eRx requests and responses received from it by VA were successfully received or resulted in error. | |
| OWNR13420 | Provide the ability for a VA Pharmacy to receive electronic notifications in accordance with the most current version supported of the NCPDP SCRIPT Standard (10.6 at minimum) from an external provider indicating that inbound eRx requests and responses sent to it from VA were successfully received or resulted in error. | |
| OWNR13421 | Provide the ability for VA pharmacy staff to interact with received inbound eRxs after the system has automatically validated the patient specified on the eRx is a known/identifiable individual to VA and the individual is able to receive pharmacy benefits rendered by VA. | |
| OWNR13422 | Provide the ability for VA pharmacy staff to interact with received inbound eRxs after the system has automatically validated the external provider that sent the eRx is a valid provider authorized to write prescriptions and/or provide services to the specified VA patient. | |
| OWNR13423 | Provide the ability for VA pharmacy staff to interact with received inbound eRxs after the system has automatically validated the medication or medical supply specified on the eRx is recognized by VA and allowed to be prescribed to a VA patient. | |
| OWNR13426 | Provide the ability for VA pharmacy staff to manually validate an inbound eRx (to include identifying valid patient, provider, and medication or medical supply) before the eRx can be further processed for dispensing. | |
| OWNR13427 | Provide the ability for VA pharmacy staff to perform existing VistA OP functions on an eRx once it is automatically received into the VistA OP queue. | |
| OWNR13430 | Provide the ability for VA pharmacies to track and account for the number of inbound eRxs received and processed in order to accommodate any future needs for reporting, auditing, or fulfilling contractual obligations, (e.g., for paying intermediaries). | |
| OWNR13433 | Provide the ability to electronically receive a transferred Rx for medication (including controlled substances) or medical supplies in accordance with the most current, appropriate NCPDP Standard(s) from an external pharmacy so that the VA pharmacy may dispense the prescription.  **NOTE: This is currently out of scope for Version 2.0.** | |

### Security and Privacy Requirements

The Federal Information Processing Standard 199 (FIPS 199), Standards for Security Categorization of Federal Information and Information Systems, defines the security categories, security objectives, and impact levels to which National Institute of Standards and Technology (NIST) Special Publication (SP) 800-60 Volume 1 Revision 1, maps information types. A FIPS 199 analysis was completed for the Inbound ePrescribing system, and it has been determined that the security categorization for this system is a MODERATE in accordance with FIPS 199.

This classification of MODERATE ensures that data is only received by the persons and applications that it is intended for; that data is not subject to unauthorized or accidental alterations; and that the system’s resources are available when needed. This classification also means that each end-user with direct or indirect access must have a valid clearance and be on a need-to-know basis for all information within the Inbound ePrescribing system (as well as any shared or networked systems). It should also be noted that personnel with system administration rights and roles will be required to undergo an elevated background investigation to fulfill their duties.

In terms of privacy requirements, the Inbound ePrescribing system will adhere to all that are required by the VA. The requisite Privacy Impact Assessment (PIA) System of Records Notice (SORN) will cover the system’s collection and maintenance of individually identifiable information. This, along with other documentation related to security and privacy, will be continually updated to ensure compliance with the latest VA guidance. An overview of the current security and privacy requirements that the system adheres to are outlined in the following table, and detailed to a much a greater degree in the Inbound ePrescribing RSD in the section titled “Security Specifications”.

Table 5: Security and Privacy Requirements

| ID | Requirement |
| --- | --- |
| 1 | All VA security requirements will be adhered to. Based on Federal Information Processing Standard (FIPS) 199 and National Institute of Standards and Technology (NIST) SP 800-60, recommended Security Categorization is Moderate. |
| 2 | The Security Categorization will drive the initial set of minimal security controls required for the information system. Minimum security control requirements are addressed in NIST SP 800-53 and VA Handbook 6500, Appendix D (BRD). |
| 3 | All VA Privacy requirements will be adhered to. Efforts that involve the collection and maintenance of individually identifiable information must be covered by a Privacy Act system of records notice. |

### System Criticality and High Availability Requirements

The Inbound eRx is considered a mission critical system and requires high availability. It follows the same standard procedures used for all critical VistA and VistA associated systems.

The system shall be available 24/7, with exceptions made for required planned and well communicated system maintenance during off-peak times.

### Single Sign-On Requirements

The Inbound eRx web UI requires users to authenticate using Personal Identification Verification (PIV) card and Active Directory. This requirement is met using the VA standard Identity Access Management (IAM) Single Sign On Implementation (SSOi) to allow seamless authentication between systems.

### Use of Enterprise Portals

The use of Enterprise Portals is not a current requirement of Inbound eRx as it does not currently expose suitable Enterprise Portal content.

# Conceptual Design

This section describes the conceptual design of the Inbound eRx system.

## Conceptual Application Design

This section provides the conceptual application design of the Inbound eRx system. The Inbound eRx design includes a newly designed Java Enterprise Edition (JEE) application working together with a VistA OP module enhanced and modified to handle eRxs and related NCPDP messages.

### Application Context

Figure 2 depicts the Inbound eRx application and the external systems that it interacts with.

Figure 2: Inbound eRx Application Context Diagram

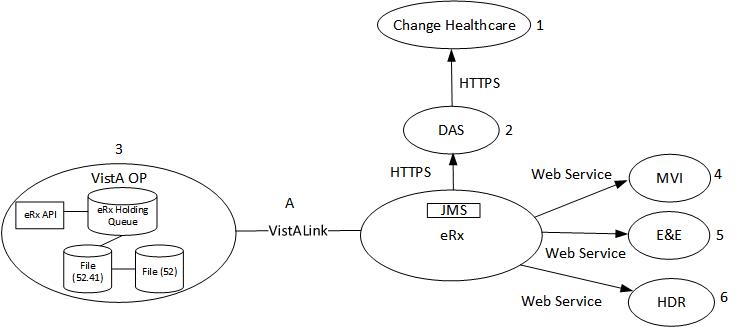


Table 6 (Grouping): Application Context Description

| ID | Name | Description | Interface Name | Interface System |
| --- | --- | --- | --- | --- |
| 2 | DAS | Acts as a network gateway between Change Healthcare and Inbound eRx | HTTPS | Inbound eRx, Change Healthcare |
| 3 | VistA OP | Allows VistA to receive prescription information from eRx | VistALink | Inbound eRx |
| 4 | MVI | Allows a system like Inbound eRx to confirm patient identity. | Web Service | Inbound eRx |
| 5 | E&E | Allows a system like Inbound eRx to confirm patient’s enrollment and eligibility. | Web Service | Inbound eRx |
| 6 | HDR | Allows a system like Inbound eRx to request patient’s medication history.  **NOTE:** This is currently out of scope for Version 2.0. | Web Service | Inbound eRx |

Interfaces External to OI&T

| ID | Name | Related Object | Input Messages | Output Messages | External Party |
| --- | --- | --- | --- | --- | --- |
| 1 | Change Healthcare | Allows system like Inbound eRx to receive prescription data. | Web Services | Inbound eRx | Change Healthcare |

Interfaces Internal to OI&T

| ID | Name | Related Object | Input Messages | Output Messages | External Party |
| --- | --- | --- | --- | --- | --- |
| A | VistALink | Inbound eRx | N/A | Messages to VistA for updating prescription data. | N/A |

Externally Shared Data Stores

| ID | Name | Data Stored | Owner | Access |
| --- | --- | --- | --- | --- |
| N/A |  |  |  |  |

### High-Level Application Design

This section describes the high-level application design of the system.

#### VistA Outpatient Pharmacy Interface/Enhancements

Currently, VistA OP is not able to accept outside prescriptions electronically, due to the lack of software and RPCs to directly handle electronically transmitted prescriptions. There are, however, APIs and other software processes in place that can be leveraged to assist in accepting and processing electronic prescriptions.

Within the current functionality, a prescription can be entered into CPRS, at which time the prescription is stored in the ORDER file (#100). Once the order is signed, the prescription information is sent to the PENDING OUTPATIENT ORDERS file (#52.41). After the prescription has been finished, the prescription is logged into the PRESCRIPTION file (#52), and the entry in the PENDING OUTPATIENT ORDERS file is removed (deleted).

A provider name (located in NEW PERSON file [#200]) is required in order to facilitate this process, which is not readily available within VistA in the appropriate format for completing the above actions electronically. For example, providers outside of the VA do not have entries in the NEW PERSON file, an electronic signature code, or a populated VA# or DEA value, which is required for certain medications.

The VistA OP package uses similar functionality when processing a new order from the option PSO LM BACKDOOR ORDERS. Using this process, entries are placed into the ORDER (#100) and PRESCRIPTION (#52) files. The order information itself is actually sent through PS EVSEND OR protocol. This is the means by which the order is created in the Order file. The key driving mechanism for the creation of the prescription and order entry information is an OP API, named EN^PSON52. Sample HL7 sent through PS EVSEND OR:

GOLDDEV 16d1>ZW MSG

MSG(1)="MSH|^~\&|PHARMACY|500|||||ORM"

MSG(2)="PID|||100808||ONEHUNDRED,INPATIENT"

MSG(3)="PV1||O|||||||||||||||||"

MSG(4)="ORC|SN||404584^PS||CM||||20151208115942-0500|10000000226^FISHER,BRADLEY

W||10000000224^PROVIDER,ONE|||20151208|W^Written^99ORN^^^"

MSG(5)="RXO|^^^5^ACETAMINOPHEN TAB^99PSP"

MSG(6)="RXE|325\T\MG\T\1\T\TABLET^BID^^20151208^20161208^^^325MG^|1338.6642^ACET

AMINOPHEN 325MG TAB^99NDF^5591^ACETAMINOPHEN 325MG TAB^99PSD|||^^^20^MG^99PSU|^^

^63^TAB^99PSF||||60||11||10000000224|501379|||||||30|||325|^^^20^MG^99PSU"

MSG(7)="NTE|21||TAKE ONE TABLET BY MOUTH TWICE A DAY"

MSG(8)="RXR|^^^1^ORAL (BY MOUTH)^99PSR"

MSG(9)="ZRX||W|N|W|10000000226^USER NAME/CLERK NAME^99NP|"

MSG(10)="ZSC||||||||"

MSG(11)="ZCL|1|1|"

MSG(12)="ZCL|1|2|"

MSG(13)="ZCL|1|3|"

MSG(14)="ZCL|1|4|"

MSG(15)="ZCL|1|5|"

MSG(16)="ZCL|1|6|"

MSG(17)="ZCL|1|7|"

MSG(18)="ZCL|1|8|"

A new VistA Inbound eRx Holding Queue will be created that will allow a pharmacist to view all inbound eRxs, rectify errors, review information, and ultimately, ‘submit’ or ‘approve’ the eRx. New RPCs will be created within the OP package to accept incoming HL7 messages containing all the needed elements for a prescription from a non-VA medical facility. Using the inbound HL7 message, a new entry will be placed in the eRx Holding Queue file.

This queue will use List Manager for user interaction. The eRx Holding Queue will show eRxs in a list format, with extended options being available to see all of the details about the prescriptions. Additional extended options will be created to allow the pharmacist to fix issues with the prescription, such as provider, drug, and patient identification errors.

After all automatic and manual checks are complete, a pharmacist will need to choose to ‘Submit’ or ‘Approve’ the eRx from the Holding Queue. At this point, there will be enough information about the prescription, and the eRx will be sent to the either the PENDING OUTPATIENT ORDERS file (#52.41), or processed using the EN^PSON52 API, which will add the prescription to the prescription file, and create an order in OE/RR via PS EVSEND OR. The decision will be based on the difficulty encountered when attempting to interface with OE/RR to create the Order. If using the PENDING OUTPATIENT ORDERS file (#52.41), the .01 field (placer order number) is the pointer to the order in the ORDER file (#100). It may make more sense to use the existing ‘New Order’ API to reduce development time/effort as well as additional needs for ICR’s. Once the eRx has been sent to the PENDING OUTPATIENT ORDERS file, a user will be able to ‘finish’ the prescription the same way as it is done today. This will leverage existing process, and reduce the need for duplicating logic. The PSO LMOE FINISH functionality will be modified to include eRx specific information.

Routine PSONEW1 is the driver for the entry of new orders through PSO LM BACKDOOR ORDERS. This routine prompts the user for a series of items. These items are: Patient Status, Drug, ICD diagnosis codes for the order (only prompted if user holds the provider key), Dosing, Patient Instructions, Days Supply, Quantity, label copies, # of Refills, Provider, Clinic, Mail/Window information, Remarks, Issue Date, and Fill Date. This routine calls several other routines, and requires user interaction to answer the prompts that are displayed.

Much of this data will already be available through the eRx Holding Queue, but may need to be verified against the logic within VistA before a submitted eRxs can be fully processed. Depending on where it makes the most sense, underlying logic within each of these prompting sequences will be leveraged to ensure the same checks are being done on eRx entries, as are being done for ‘New Orders’ within PSO LM BACKDOOR ORDERS. This may mean that the checking is done via verification options within the eRx Holding Queue. If the eRx Holding Queue is not the ideal place for such checks, the verification will occur through a series of ‘new’ prompts once the pharmacist chooses to ‘submit’ or ‘approve’ the eRx for completion. The eRx processing functionality will aim to keep the number of user interactive prompts to a bare minimum, and cannot be guaranteed to be able to remove user interaction entirely. There may be issues with dosing or other fields that may need to be resolved through user prompts after the ‘submit’/’approve’ process. These would be final steps to ensure the data is appropriate.

During the entry of a new order within PSO LM BACKDOOR orders, checks are performed based on the drug that has been selected. Once the order has been added to the PRESCRIPTION and ORDER files, it will follow the existing order checks that are built into the current pharmacy functionality. Order checking will not occur from the eRx Queue. All order checking will be done by the existing processes in place within the pharmacy package.

Some new processes/RPCs will need to be created to potentially ‘bypass’ the order signing process, or make the order signing process happen automatically. The signing process requires an Electronic Signature code to be in place in the NEW PERSON file (#200) in order to complete the signature action.

Once the order is in the PENDING OUTPATIENT ORDERS file or the Prescription and Orders files, pharmacists can then use existing processes (PSO LMOE FINISH/PSO VR) to finish and verify the medication (if needed). Due to the use of the existing file structures, minimal change will be needed in these areas to facilitate the release/finishing of a medication.

New incoming partial prescriptions will always require that a new Rx be created in the Prescription file (#52). The partial indicator is a subfile within the Prescription file (#52), and will need to be utilized to identify that the prescription is a partial. This can be a tricky process due to needing a new prescription with 0 (zero) refills. The prescription itself would then not be able to be ‘filled’ per the standard filling process, rather we will have to leverage the partial fill logic from the PSORRPA\* routines.

#### Inbound ePrescribingSystem

Figure 3 provides a high-level application diagram for the system. The circles/ovals within the “eRx” oval represent the business services that comprise the Inbound eRx application Service-Oriented Architecture (SOA) approach.

Figure 3: Inbound eRx High-Level Application Diagram

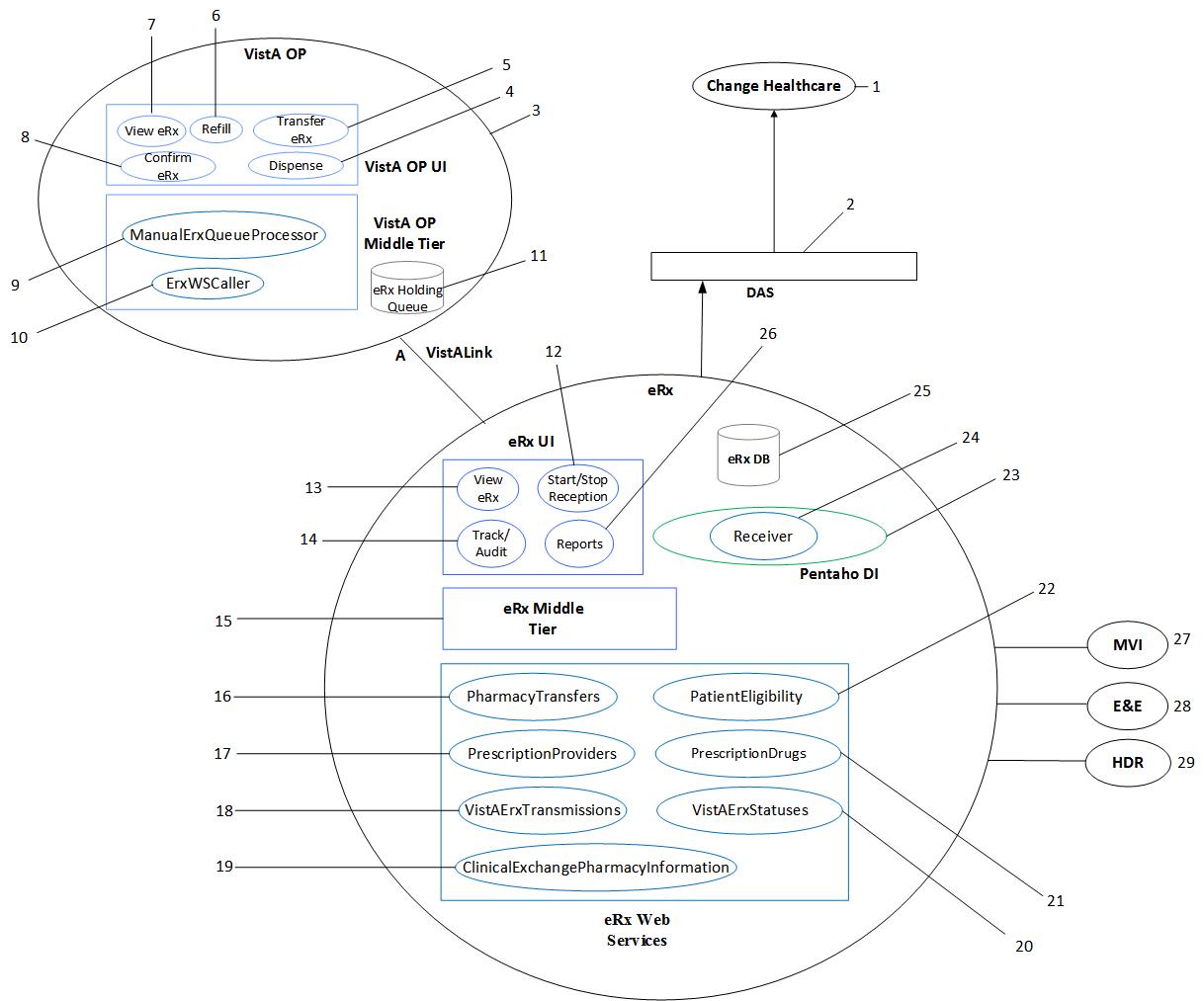


Table 7: Objects in the High Level Application Design

| ID | Name | Description | Service or Legacy Code | External Interface Name | External Interface ID | Internal Interface Name | Internal Interface ID | SDP Sections 1&2 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| A | VistA Link | It is used to authorize user in VistA and to send authenticated RPC requests to VistA. | Service | VistALink, the interface mechanism between VistA and Inbound eRx | A | Internal calls to this service are performed via direct Java method calls. | N/A | N/A |
| 4 | Dispense | This functionality allows dispensing of an eRx. It is part of VistA OP UI. | N/A | N/A | N/A | Internal calls to this service are via MUMPs routines, which are invoked when this service is selected from VistA OP UI options. | N/A | N/A |
| 5\* | Transfer eRx | This functionality allows transferring of an eRx.  It is part of VistA OP UI. | N/A | N/A | N/A | Internal calls to this service are via MUMPs routines, which are invoked when this service is selected from VistA OP UI options. | N/A | N/A |
| 6 | Refill | This functionality allows refilling an existing prescription. It is part of VistA OP UI. | N/A | N/A | N/A | Internal calls to this service are via MUMPs routines, which are invoked when this service is selected from VistA OP UI options. | N/A | N/A |
| 7 | View eRx | This functionality allows viewing an eRx from the VistA eRx Holding Queue. It is part of VistA OP UI. | N/A | N/A | N/A | Internal calls to this service are via MUMPs routines, which are invoked when this service is selected from VistA OP UI options. | N/A | N/A |
| 8 | Confirm eRx | This functionality allows confirming an eRx. It is part of VistA OP UI. | N/A | N/A | N/A | Internal calls to this service are via MUMPs routines, which are invoked when this service is selected from VistA OP UI options. | N/A | N/A |
| 9 | ManualErxQueueProcessor | It enables the processing of the prescriptions from the eRx Holding Queue. It is Internal to VistA OP. | Service | N/A | N/A | Internal calls to this service are via MUMPs routines, which are invoked when manual validation step of eRx Prescription data starts. | N/A | N/A |
| 10 | ErxWSCaller | It sends the prescription status to the eRx middle by calling VistAErxStatuses web service. | Service | N/A | N/A | Internal calls to this service are via MUMPs routines, which are invoked when Prescriptions are manually processed and statues are generated and are passed to this. | N/A | N/A |
| 11 | eRx Holding Queue | A temporary storage file which will hold pending eRxs which were transmitted from eRx processing hub. | Service | N/A | N/A | Internal calls to this service are via MUMPs routines, which are invoked when manual validation step of eRx Prescription data starts. | N/A | N/A |
| 12 | Start/Stop Reception | This is part of eRx UI. It allows a VA pharmacy to start/stop eRx transmission from Change Healthcare. | Service | N/A | N/A | Internal calls to this service are performed via direct Java method calls. | N/A | N/A |
| 13 | View eRx | This service allows viewing eRx on eRx GUI. | Service | N/A | N/A | Internal calls to this service are performed via direct Java method calls. | N/A | N/A |
| 14 | Track/Audit | This service allows tracking/auditing an eRx. | Service | N/A | N/A | Internal calls to this service are performed via direct Java method calls. | N/A | N/A |
| 16\* | PharmacyTransfers | This is a RESTful Web Service. It converts the data into XML format and sends to Pentaho DI. | Service | N/A | N/A | It is called by VistA with prescription pharmacy transfer requests. | N/A | N/A |
| 17 | PrescriptionProviders | This is a RESTful Web Service. It validates the provider from the new eRxs. | Service | N/A | N/A | It is called by Pentaho DI Job. | N/A | N/A |
| 18 | VistAErxTransmissions | This is a RESTful Web Service. It transmits the eRxs to VistA OP via RPC call. | Service | N/A | N/A | It is called by Pentaho DI Job. | N/A | N/A |
| 19 | ClinicalExchangePharmacyInformation | This is a RESTful Web Service. It is used to send pharmacy data to be updated in Change Healthcare. | Service | N/A | N/A | It is called by Pentaho DI Job. | N/A | N/A |
| 20 | VistAErxStatuses | This is a RESTful Web Service. It gets the status of a prescription while it goes through processing. | Service | N/A | N/A | It is called by VistA OP ErxWSCaller. | N/A | N/A |
| 21 | PrescriptionDrugs | This is a RESTful Web Service. It checks drug information in local VistA OP via an RPC call. | Service | N/A | N/A | It is called by Pentaho DI Job. | N/A | N/A |
| 22 | PatientEligibility | This is a RESTful Web Service. It searches for patient and their eligibility | Service | N/A | N/A | It is called by Pentaho DI Job. | N/A | N/A |
| 24 | Receiver | This is a job inside Pentaho DI which picks up new eRxs messages from eRx Processing Hub JMS queue. | Service | N/A | N/A | N/A | N/A | N/A |
| 26 | Reports | This service allows running of reports on eRx. | Service | N/A | N/A | Internal calls to this service are performed via direct Java method calls. | N/A | N/A |
| 27 | MVI | Allows a system like Inbound eRx to confirm patient identity. | Service | IdMWebService | N/A | N/A |  |  |
| 28 | E&E | Allows a system like Inbound eRx to confirm patient’s enrollment and eligibility. | Service | getEESummary | N/A | N/A | N/A | N/A |
| 29 | HDR | Allows a system like Inbound eRx to request patient’s medication history.  **NOTE:** This is currently out of scope for Version 2.0. | Service | Unknown in this version. | N/A | N/A | N/A | N/A |

\* Future functionality; will not be released in Version 2.0.

Internal Data Stores

| ID | Name | Data Stored | Steward | Access |
| --- | --- | --- | --- | --- |
| 4 | eRx Database | The eRx database persists all of the information needed by the Inbound eRx application. | *Intermediate Service layer classes perform the CRUD operations needed by the application.* | *The Inbound eRx application performs entity create, read, update and delete operations on the eRx database.* |

### Application Locations

This sections specifies the locations at which the application components will be hosted.

Table 8: Application Locations

| Application Component | Description | Location at Which Component is Run | Type |
| --- | --- | --- | --- |
| View, confirm, validate eRx, manage, fill, print labels, dispense, etc. an inbound eRx. Send/transmit inbound eRx. | VA Pharmacists, VA Pharmacy Technicians/Assistants, VA Pharmacy Managers will be allowed to access these Inbound eRx application’s services. | AITC – Inbound eRx server | View, confirm, validate eRx, manage, fill, print, dispense, etc. an inbound eRx. Send/transmit inbound eRx. |
| Business Services | Supports presentation logic, business logic, and data CRUD services for information in the Inbound eRx application. | AITC – Inbound eRx server | Presentation Logic/ Business Logic/Data Logic (via Hibernate) |
| VistA Link Service | Supports the Inbound eRx application interface to VistA. | AITC – Inbound eRx server | Interface Code |

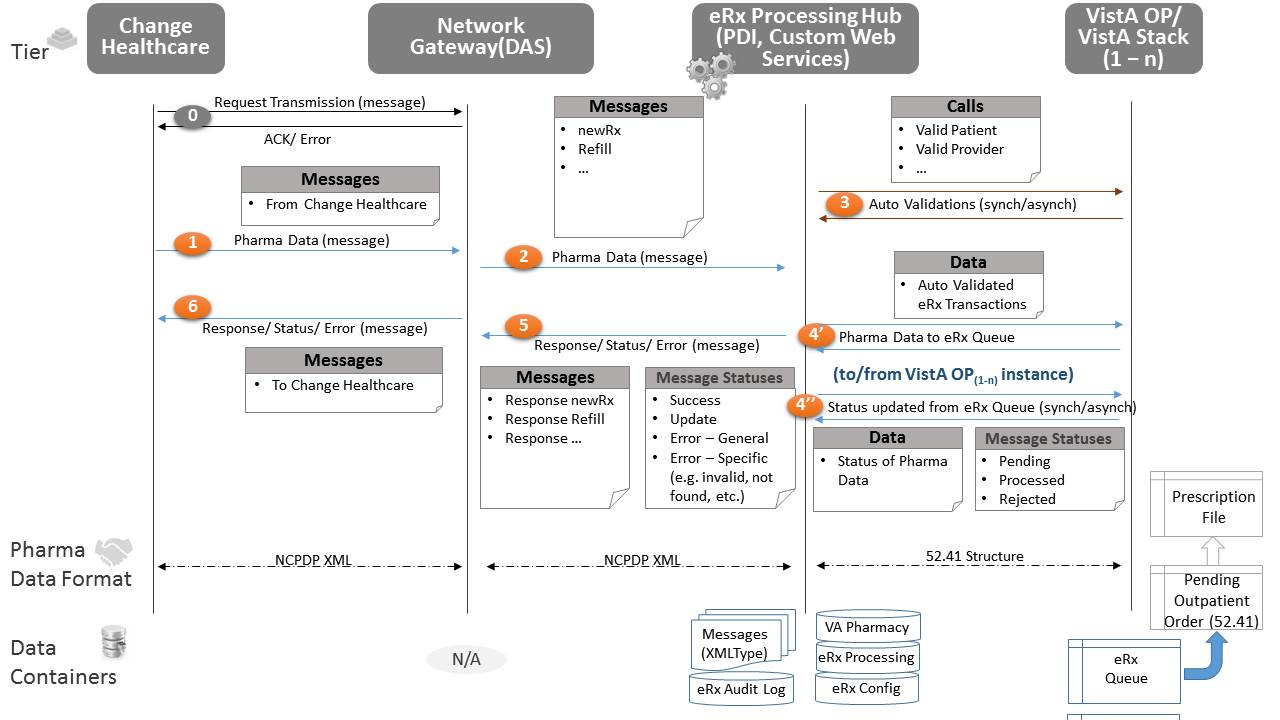
Table 9: Application Users

| Application Component | Location | User |
| --- | --- | --- |
| View, confirm, validate eRx, manage, fill, print labels, dispense, etc. an inbound eRx. Send/transmit inbound eRx. | AITC – Inbound eRx server | VA Pharmacists, VA Pharmacy Technicians/Assistants, VA Pharmacy Managers will be allowed to access these Inbound eRx application’s services. |

## Conceptual Data Design

The conceptual data design presented in this section includes eRx database design concepts and functionality, as well as conceptual details related to VistA OP files involved in the end-to-end Inbound eRx solution. Figure 4 depicts the conceptual data design and the data flows between the four (4) architectural tiers – Change Healthcare (external to VA), Network Gateway (DAS), eRx Processing Hub and VistA OP/VistA Stack.

Figure 4: Conceptual Data Flows and Designs



The main communication messages include:

* Step 0 – “handshake” messages between Change Healthcare and eRx Processing Hub.
* Step 1 – Pharma Data messages containing eRx message from Change Healthcare in NCPDP XML format; Change Healthcare supported messages are pre-defined and it is anticipated that new messages are not going to be added often.
* Step 2 – the eRx Processing Hub “listens” to the configured eRx Processing Hub JMS Message Queue and receives the eRx messages via DAS as proxy; structural validation is performed on the eRx message; messages are of different types (e.g., newRx, Refill, etc.).
* Step 3 – the eRx Processing Hub coordinates auto-validations of the received eRx message (e.g., MVI call to identify and validate the patient).
* Step 4’ – validated eRx transaction is sent to the staging area in VistA OP (the eRx Holding Queue).
* Step 4’’ –eRx Processing Hub data is synchronized with the status of the transmitted eRx transactions in the eRx Holding Queue in VistA OP; the eRx processing logic in VistA OP is responsible for updating the status of each eRx transaction in the eRx Holding Queue.
* Step 5 – a response message is constructed and sent through DAS to Change Healthcare; response messages are of type corresponding to the incoming eRx messages (i.e., Response newRx, Response Refill, etc.); each response message is associated with a message status such as Success, Update, Error (general or specific – invalid, not found, or other).

**NOTE:** The step sequence described above is representative. Depending on the type of message and its content, some of the steps may be repeated multiple times (e.g., different match criteria are sent for patient identification and validation) or skipped entirely (e.g., message structure validation error occurs).

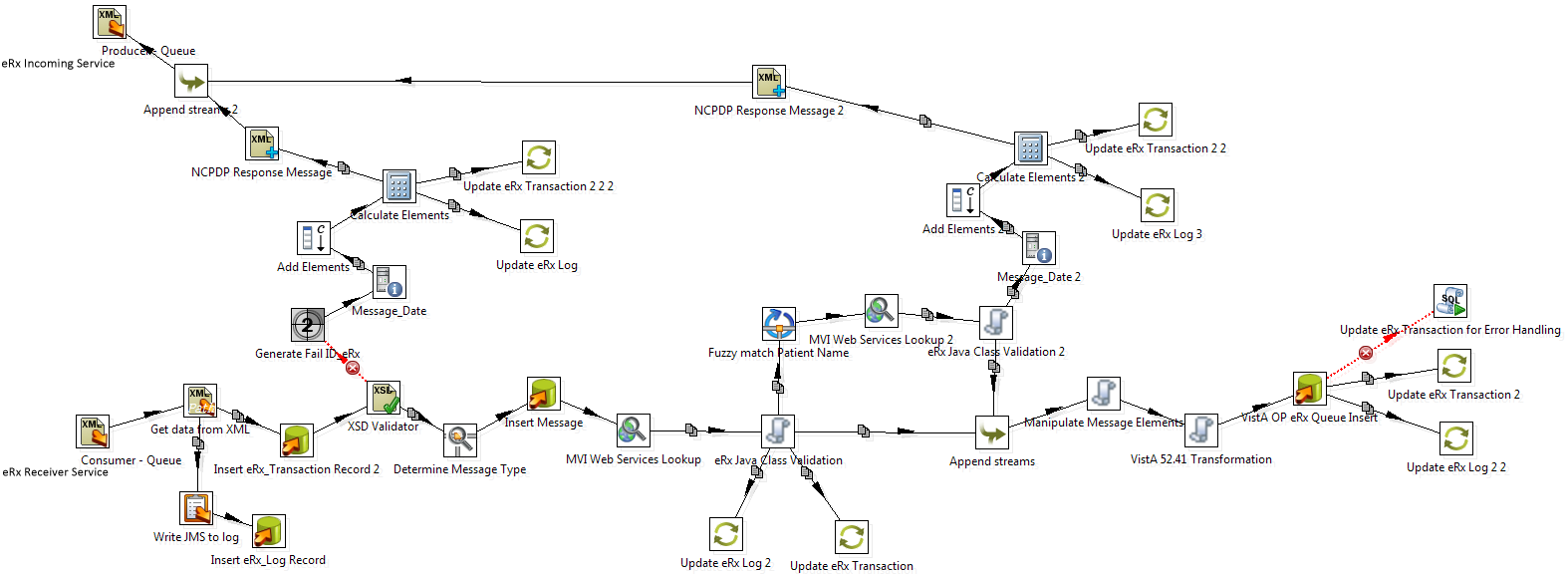
### Pharma Data Formats

* The data format between the architectural Tiers Change Healthcare and eRx Processing Hub is based on NCPDP XML messages as supported by Change Healthcare.
* The data format between the architectural Tiers Network Gateway (DAS) and eRx Processing Hub (Pentaho DI) is based on NCPDP XML messages as sent by Change Healthcare; no XML message manipulation is anticipated in the Network Gateway (DAS) tier.
* The data format between the architectural Tiers eRx Processing Hub (Pentaho DI) and VistA OP/ VistA Stack is synchronous (i.e., not message based); auto validation calls are in the format expected by the respective APIs/RPCs or other interfaces (e.g., patient MVI validation); Pharma data is “sent” and status data is “received” to/from the eRx Holding Queue in VistA OP (the staging area) via SOA data connection.

### High-Level eRx Processing Hub Logic (Example Transaction)

This section depicts the high-level eRx Processing Hub logic for newRx message transaction with Pentaho Data Integration (Figure 5). Please note that this logic is for illustrative purposes and the actual processing logic will likely be more complex.

Figure 5: eRx Message Processing Logic – Example



#### Integration with DAS

VA’s DAS is a messaging, queuing middleware provider and network gateway. Inbound eRx will use DAS as a network gateway for data communication with Change Healthcare.

The eRx Processing Hub will utilize Oracle WebLogic JMS Queue hosted on Inbound eRx AITC servers.

#### High Level Processing Flow (for illustrative purposes)

* Message from Change Healthcare arrives at the JMS queue on the Processing Hub.
* Message is being read with the XML Get Data Utility; the JMS log is being read and the eRx\_Log and eRx\_Transaction records are created.
* The structure of the message is validated with the XSD Validator Utility.
  + If Error, NCPDP XML message is constructed, transaction and log tables are updated, and a response message is sent via DAS to Change Healthcare
  + If success, message type is determined (e.g., newRx) and raw message is stored in the respective XMLMessage table
* MVI validation is performed using the web services validation utility.
  + If Error, for illustrative purposes, we have depicted a scenario where we do not reject the message; instead, a fuzzy match is performed based on “like” names where traditional lookup matches do not work and Levenshtein matching algorithm is used instead; if error, response message is constructed and sent via DAS to Change Healthcare
  + If success, the eRx transaction is (in this simplified example) considered auto-validated and ready to be sent to VistA OP eRx Holding Queue
* Message is manipulated and transformed to match the VistA OP expected structure; message is transferred to the VistA OP via an RPC.
  + For simplicity purposes, if case or error, the transaction status is updated allowing a separate (not depicted here) error handling process to identify and process the failed transaction again

As stated earlier, the purpose of this section is to outline a high level example of a design approach to processing eRx messages. As with all ETL transformations, the design can be finalized during a pilot phase and after extensive testing.

### Project Conceptual Data Model

Figure 4 depicts the main data containers in each architectural tier. Refer to Section 5.4.1 for the Entity Relationship Diagram (ERD).

* Network Gateway (DAS):
  + Data Containers: N/A
* *Context:* DAS is a pass-through proxy with no data storage. eRx Processing Hub (Pentaho DI):
  + Data Containers:
    - XML Messages – stores eRx inbound and outbound messages as XMLType (one table per message type).
    - eRx\_Transaction – stores processing data for managing the lifecycle of each eRx message.
    - VA\_Pharmacy – stores VA Pharmacy information.
    - eRx\_Config – stores eRx Processing Hub configuration and translation data.
    - eRx\_Log – stores audit log data and status changes for each eRx transaction.
  + *Context:* Data containers are stored in an Oracle 11g database.
* VistA OP:
  + Data Containers:
    - eRx – Holding Queue VistA OP – staging area for the validated by the *eRx Processing Hub* eRx transaction data; has a similar structure as the Pending Outpatient Order (#52.41) file + the eRx status flag(s) and “link(s)” with the eRx Transaction Hub records.
    - eRx – Medication History Rcvd – staging area to receive patient Medication History information. (**NOTE:** Out of scope in Version 2.0)
  + *Context:* VistA OP eRx functionality, which is developed as part of this initiative, processes the received eRx transactions similarly as the existing VistA OP functionality to perform drug-to-drug, DEA special handling and other validations; as part of the processing, the eRx transaction is loaded into the PENDING OUTPATIENT ORDERS (52.41) and PRESCRIPTION file (#52).

Section 5 of this document depicts further technical details about the data model.

### Database Information

Table 10: Database Inventory

| Database Name | Description | Type | Steward |
| --- | --- | --- | --- |
| Inbound eRx | Oracle 11g Relational Database used to store, process and archive eRx data and application metadata | Create | Inbound eRx |
| eRx Holding Queue VistA OP | New data file with similar to 52.41 structure for staging eRx transactions for processing within VistA OP | Create | VistA OP |

### User Interface Data Mapping

This section describes and defines the information that will be available for users of the system to be able to enter data into the database or to retrieve information from the database, if applicable.

#### Application Screen Interface

This section describes the eRx screen interfaces.

##### Login Screen

The figure below depicts the eRx login screen. If the authentication and authorization is successful, the user is taken to the home screen with a welcome message (shown in home screen).

Figure 6: Single Sign-On Login Screen

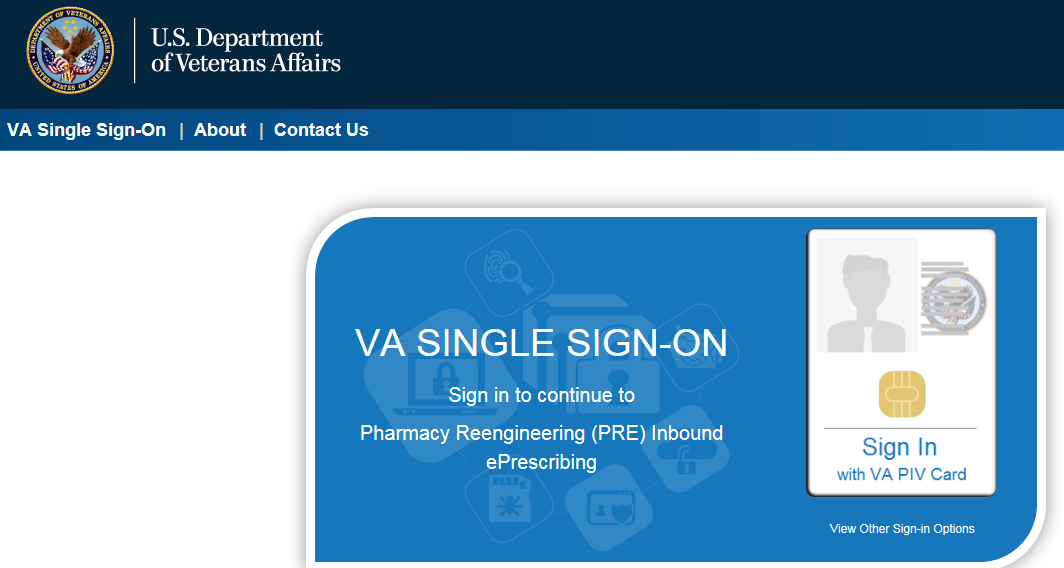


Table 11: eRx GUI Login Screen Description

| Graphical User Interface (GUI) Field | Table (Database Table that field connects to) | Field (Field in Table that the GUI field connects to) | Comments |
| --- | --- | --- | --- |
| No fields defined on this screen. | N/A | N/A | N/A |

##### Home Screen

The figure below illustrates shows the eRx application home screen after the user is authenticated successfully. It shows a welcome message.

Figure 7: Login Accept Screen

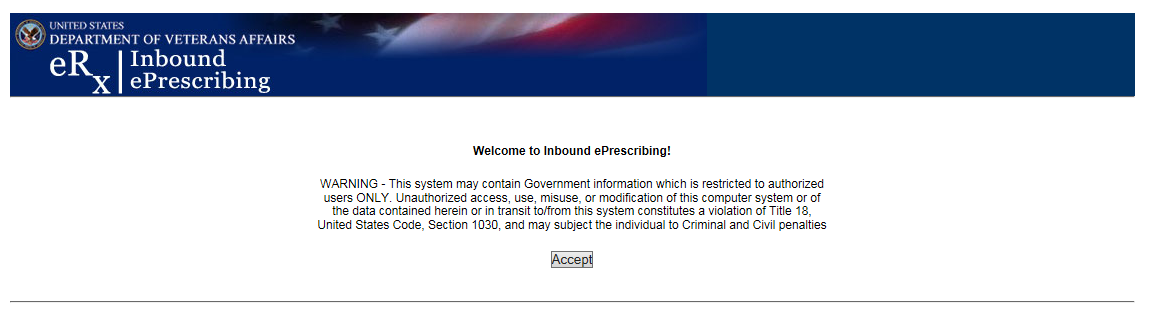


Table 12: eRx Login Accept Screen Description

| Graphical User Interface (GUI) Field | Table (Database Table that field connects to) | Field (Field in Table that the GUI field connects to) | Comments |
| --- | --- | --- | --- |
| No fields defined on this screen. | N/A | N/A | N/A |

Figure 8: Home Screen

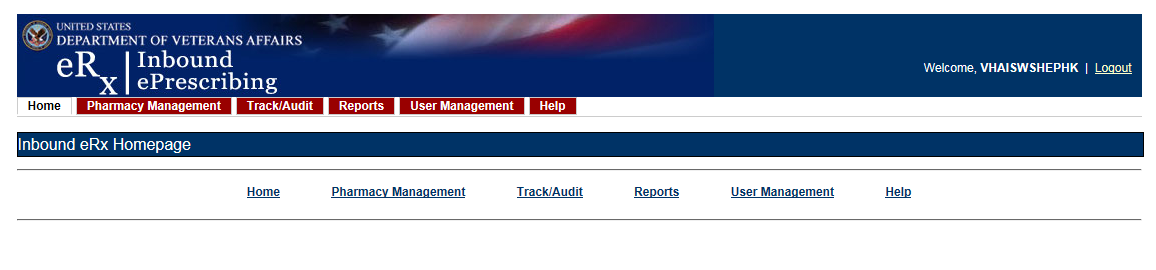


Table 13: eRx Home Screen Description

| Graphical User Interface (GUI) Field | Table (Database Table that field connects to) | Field (Field in Table that the GUI field connects to) | Comments |
| --- | --- | --- | --- |
| No fields defined on this screen. | N/A | N/A | N/A |

##### Pharmacy Management Screen

The figure below illustrates the Pharmacy Management screen in the application. This screen allows administrators to add new pharmacies, update pharmacy information, and enable/disable pharmacies.

Figure 9: Pharmacy Management Screen

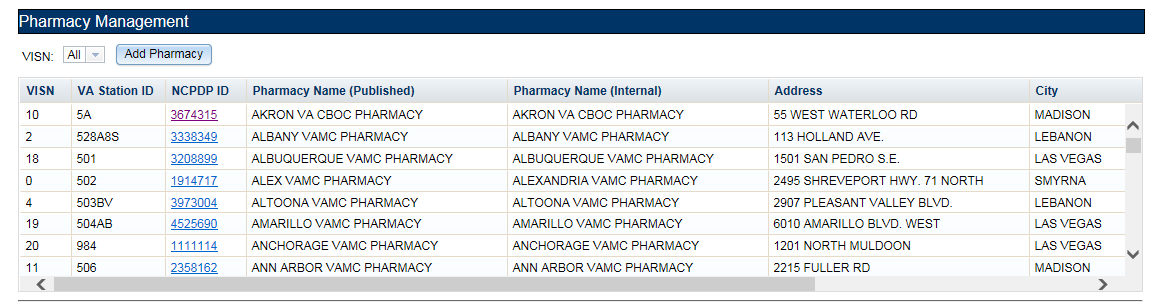


Table 14: Pharmacy Management Screen Description (Pharmacy Table)

| Graphical User Interface (GUI) Field | Table (Database Table that field connects to) | Field (Field in Table that the GUI field connects to) | Comments |
| --- | --- | --- | --- |
| VISN | PHARMACY | VISN |  |
| VA Station ID | PHARMACY | VA\_STATION\_ID |  |
| NCPDP ID | PHARMACY | NCPDP\_ID |  |
| Pharmacy Name (Published) | PHARMACY | STORE\_NAME |  |
| Pharmacy Name (Internal) | PHARMACY | DIVISION\_NAME |  |
| Address | PHARMACY | PHARMACY\_ADDRESS\_LINE\_1  PHARMACY\_ADDRESS\_LINE\_2 |  |
| City | PHARMACY | PHARMACY\_CITY |  |
| State | PHARMACY | PHARMACY\_STATE |  |
| Phone Number | PHARMACY | PHARMACY\_PHONE\_NUMBER |  |
| eRx Enabled | PHARMACY | INBOUND\_ERX\_ENABLED |  |
| Effective Date | PHARMACY | ACTIVE\_START\_TIME |  |

##### Add Pharmacy Screen

The figure below illustrates the Add Pharmacy screen in the application.

Figure 10: Add Pharmacy Screen

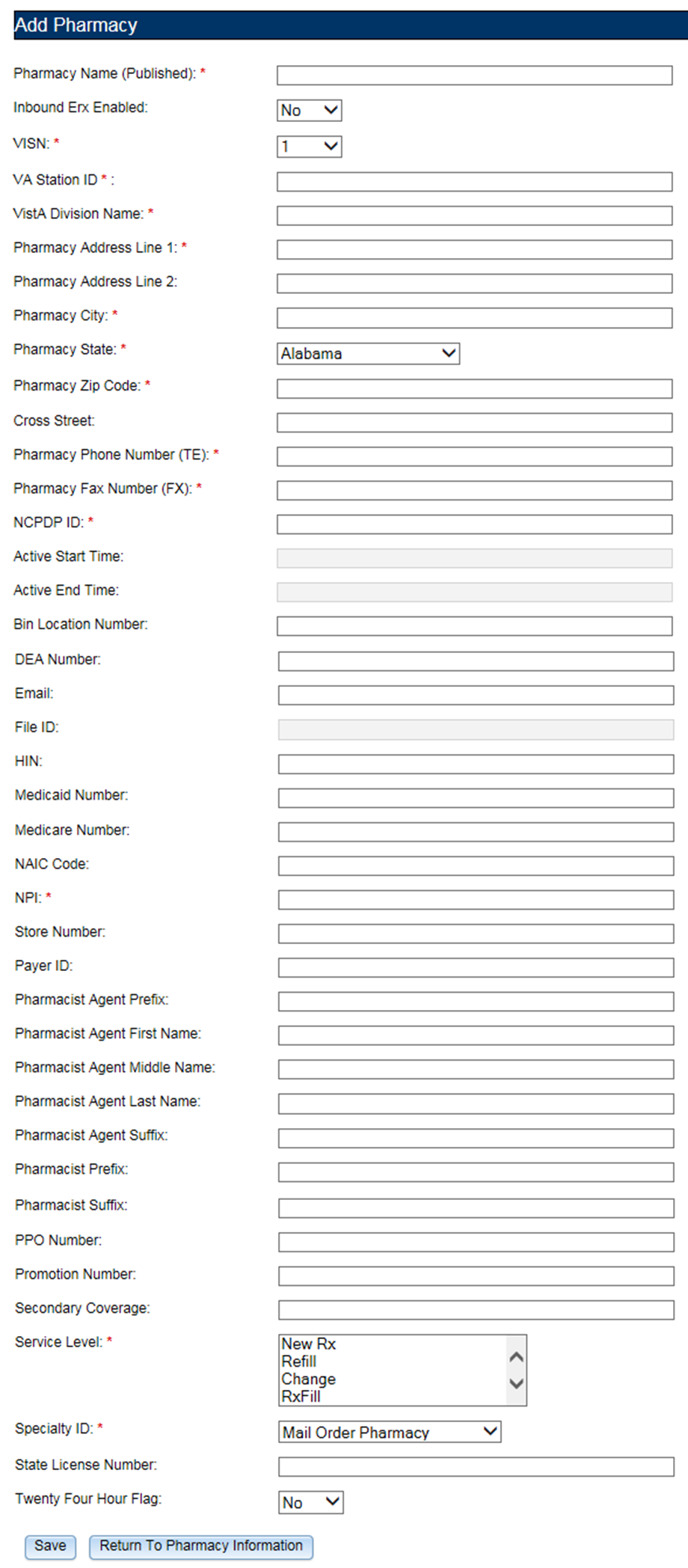


Table 15: Add Pharmacy Screen Description

| Graphical User Interface (GUI) Field | Table (Database Table that field connects to) | Field (Field in Table that the GUI field connects to) | Comments |
| --- | --- | --- | --- |
| Pharmacy Name (Published) | Pharmacy | STORE\_NAME |  |
| Inbound eRx Enabled | Pharmacy | INBOUND\_ERX\_ENABLED |  |
| VISN | Pharmacy | VISN |  |
| VA Station ID | Pharmacy | VA\_STATION\_ID |  |
| VistA Division Name | Pharmacy | DIVISION\_NAME |  |
| Pharmacy Address Line 1 | Pharmacy | PHARMACY\_ADDRESS\_LINE\_1 |  |
| Pharmacy Address Line 2 | Pharmacy | PHARMACY\_ADDRESS\_LINE\_2 |  |
| Pharmacy City | Pharmacy | PHARMACY\_CITY |  |
| Pharmacy State | Pharmacy | PHARMACY\_STATE |  |
| Pharmacy Zip Code | Pharmacy | PHARMACY\_ZIPCODE |  |
| Cross Street | Pharmacy | CROSS\_STREET |  |
| Pharmacy Phone Number (TE) | Pharmacy | PHARMACY\_PHONE\_NUMBER |  |
| Pharmacy Fax Number (FX) | Pharmacy | PHARMACY\_FAX\_NUMBER |  |
| NCPDP ID | Pharmacy | NCPDP\_ID |  |
| Active Start Time | Pharmacy | ACTIVE\_START\_TIME |  |
| Active End Time | Pharmacy | ACTIVE\_END\_TIME |  |
| Bin Location Number | Pharmacy | BIN\_LOCATION\_NUMBER |  |
| DEA Number | Pharmacy | DEA\_NUMBER |  |
| Email | Pharmacy | EMAIL |  |
| File ID | Pharmacy | FILE\_ID |  |
| HIN | Pharmacy | HIN |  |
| Medicaid Number | Pharmacy | MEDICAID\_NUMBER |  |
| Medicare Number | Pharmacy | MEDICARE\_NUMBER |  |
| NAIC Code | Pharmacy | NAIC\_CODE |  |
| NPI | Pharmacy | NPI |  |
| Store Number | Pharmacy | STORE\_NUMBER |  |
| Payer ID | Pharmacy | PAYER\_ID |  |
| Pharmacist Agent Prefix | Pharmacy | PHARMACIST\_AGENT\_PREFIX |  |
| Pharmacist Agent First Name | Pharmacy | PHARMACIST\_AGENT\_FIRST\_NAME |  |
| Pharmacist Agent Middle Name | Pharmacy | PHARMACIST\_AGENT\_MIDDLE\_NAME |  |
| Pharmacist Agent Last Name | Pharmacy | PHARMACIST\_AGENT\_LAST\_NAME |  |
| Pharmacist Agent Suffix | Pharmacy | PHARMACIST\_AGENT\_SUFFIX |  |
| PPO Number | Pharmacy | PPO\_NUMBER |  |
| Promotion Number | Pharmacy | PROMOTION\_NUMBER |  |
| Secondary Coverage | Pharmacy | SECONDARY\_COVERAGE |  |
| Service Level | Pharmacy | SERVICE\_LEVEL |  |
| Specialty ID | Pharmacy | SPECIALITY\_ID |  |
| State License Number | Pharmacy | STATE\_LICENSE\_NUMBER |  |
| Twenty Four Hour Flag | Pharmacy | TWENTY\_FOUR\_HOUR\_FLAG |  |

##### Track/Audit Screen

The figure below depicts the Inbound eRx Track/Audit screen. It allows the pharmacist or pharmacy technician to track a prescription or see an audit trail of a prescription.

Figure 11: Track/Audit Screen (Search Criteria & Results)

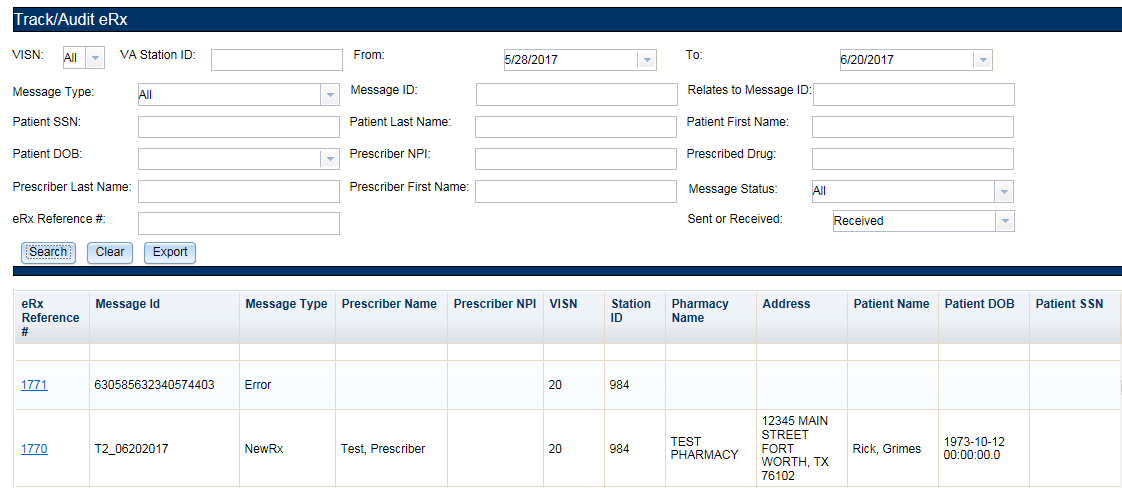


Table 16: Track/Audit Search Criteria Screen Description (Search Criteria & Results)

| Graphical User Interface (GUI) Field | Table (Database Table that field connects to) | Field (Field in Table that the GUI field connects to) | Com-ments |
| --- | --- | --- | --- |
| VISN | Pharmacy | VISN |  |
| VA Station ID | PHARMACY | VA\_STATION\_ID |  |
| From | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| To | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| Message Type | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE\_TYPE |  |
| Message ID | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE\_ID |  |
| Relates to Message ID | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | REL\_TO\_MESSAGE\_ID |  |
| Patient SSN | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| Patient Last Name | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| Patient First Name | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| Patient DOB | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| Prescriber NPI | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| Prescribed Drug | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| Prescriber Last Name | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| Prescriber First Name | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| Message Status | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE\_STATUS |  |
| eRx Reference # | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| Sent or Received | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |

##### View eRx Detail Screen

The figure below shows the View eRx Detail Screen.

Figure 12: New eRx Detail Screen

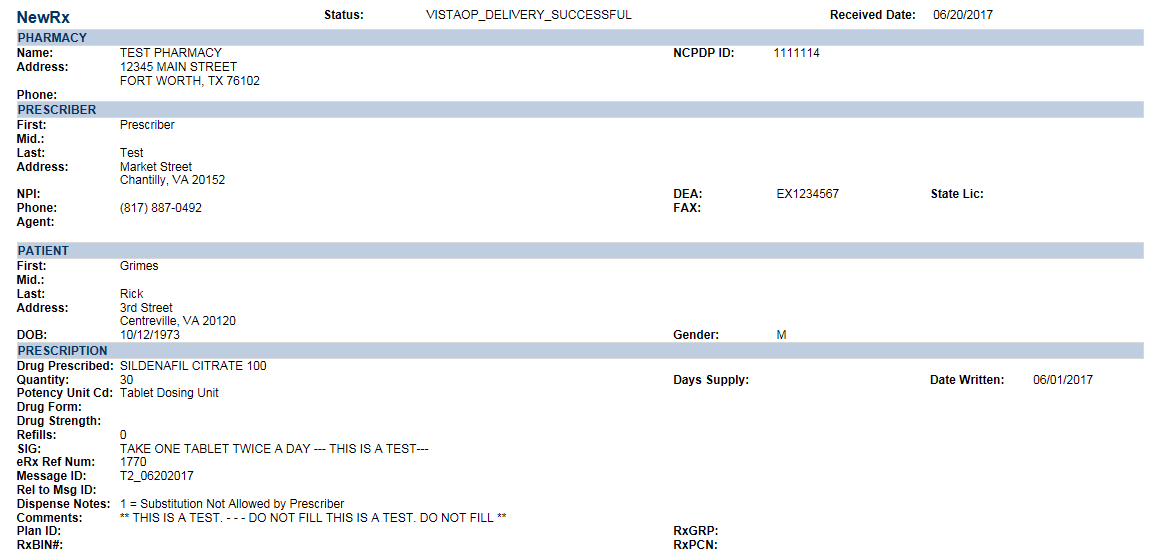


Table 17: View New eRx Detail Screen Description

| Graphical User Interface (GUI) Field | Table (Database Table that field connects to) | Field (Field in Table that the GUI field connects to) | Comments |
| --- | --- | --- | --- |
| New eRx | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE\_TYPE |  |
| Status | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE\_STATUS |  |
| Received Date | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | RECEIVED\_DATE |  |
| Pharmacy Info | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| Name | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| Address | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE | AddressLine1+AddressLine2+City+State+ZipCode |
| Phone | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE | Code Qualifier=="TE" |
| NCPDP ID | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| Prescriber Info | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| First | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| Mid. | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| Last | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| Address | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE | AddressLine1+AddressLine2+City+State+ZipCode |
| NPI | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| DEA | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| State Lic | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| Phone | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE | Qualifier=="TE" |
| FAX | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE | Qualifier=="FX" |
| Agent | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE | PrescriberAgent.First Name+ PrescriberAgent.LastName |
| Patient Info | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| First | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| Mid | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| Last | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| Address | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE | AddressLine1+AddressLine2+City+State+ZipCode |
| DOB | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| Gender | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| Prescription | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| Drug Prescribed | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| Quantity | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| Days Supply | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| Date Written | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| Potency Unit Cd | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| Drug Form | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| Drug Strength | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| Refills | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| SIG | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| eRx Ref Num | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | INBOUND\_NCPDP\_MSG\_ID/OUTBOUND\_NCPDP\_MSG\_ID |  |
| Message ID | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE\_ID |  |
| Rel to Msg ID | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | REL\_TO\_MESSAGE\_ID |  |
| Dispense Note | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| Comments | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| Plan ID | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| RxGRP | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| RxBIN# | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| RxPCN | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |

##### Error Detail Screen

The figure below illustrates the Error Detail Screen.

Figure 13: Error Detail Screen

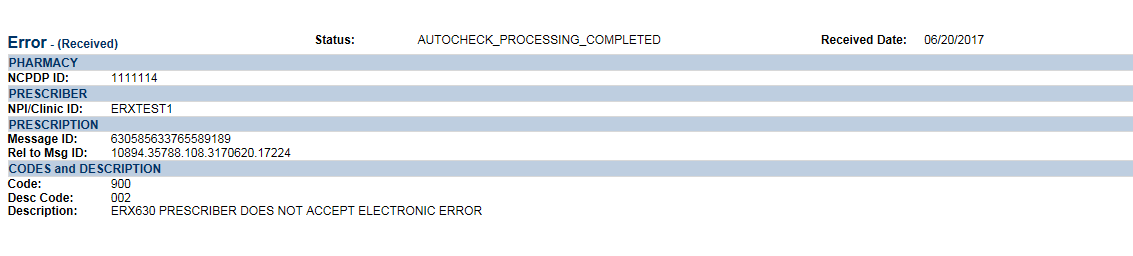


Table 18: Error Detail Screen Description

| Graphical User Interface (GUI) Field | Table (Database Table that field connects to) | Field (Field in Table that the GUI field connects to) | Comments |
| --- | --- | --- | --- |
| Error – (Received) | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | Messagetype |  |
| Status | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE\_STATUS |  |
| Received Date | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | RECEIVED\_DATE |  |
| PHARMACY | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| NCPDP ID | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| PRESCRIBER | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| NPI/Clinic ID | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| PRESCRIPTION | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| Message ID | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE\_ID |  |
| Rel to Msg ID | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | REL\_TO\_MESSAGE\_ID |  |
| CODES AND DESCRIPTION | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| Code | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| Desc Code | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| Description | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |

##### Verify Detail Screen

The figure below illustrates the Verify Detail Screen, followed by details of the fields and descriptions in the table below.

Figure 14: Verify Detail Screen

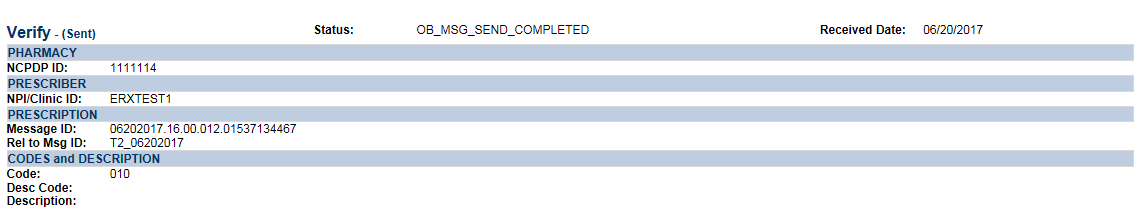


Table 19: Verify Detail Screen Description

| Graphical User Interface (GUI) Field | Table (Database Table that field connects to) | Field (Field in Table that the GUI field connects to) | Comments |
| --- | --- | --- | --- |
| Verify – (Sent) | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE\_TYPE |  |
| Status | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE\_STATUS |  |
| Received Date | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | RECEIVED\_DATE |  |
| PHARMACY | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| NCPDP ID | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| PRESCRIBER | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| NPI/Clinic ID | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| PRESCRIPTION | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| Message ID | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| Rel to Msg ID | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | REL\_TO\_MESSAGE\_ID |  |
| CODES AND DESCRIPTION | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| Code | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| Desc Code | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |
| Description | INBOUND\_NCPDP\_MSG/OUTBOUND\_NCPDP\_MSG | MESSAGE |  |

##### User Management

The User Management screen of the application allows users to manage user access to the application’s functionality, including updating roles/permissions.

Figure 15: User Management Screen

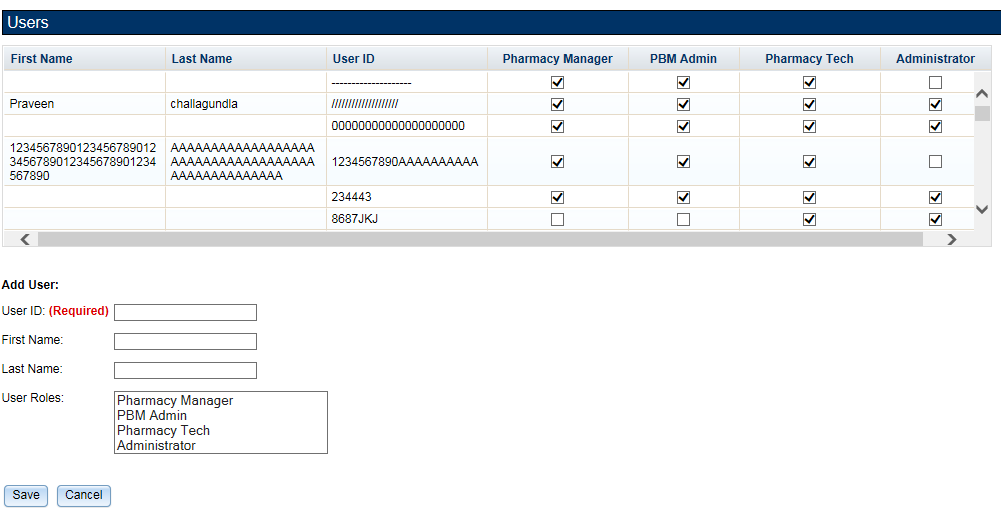


Table 20: User Management Screen Description

| Graphical User Interface (GUI) Field | Table (Database Table that field connects to) | Field (Field in Table that the GUI field connects to) | Comments |
| --- | --- | --- | --- |
| First Name | VA\_USER | FIRST\_NAME |  |
| Last Name | VA\_USER | LAST\_NAME |  |
| User ID | VA\_USER | USERS\_ID |  |
| Pharmacy Manager | VA\_USER | IS\_PHARM\_MGR |  |
| PBM Admin | VA\_USER | IS\_PBM\_ADMIN |  |
| Pharmacy Tech | VA\_USER | IS\_PHARM\_TECH |  |
| Administrator | VA\_USER | IS\_SYSTEM\_ADMIN |  |

##### Help Screen

The Help screen is a general purpose screen, which provides help topics and production support information.

Figure 16: Help Screen

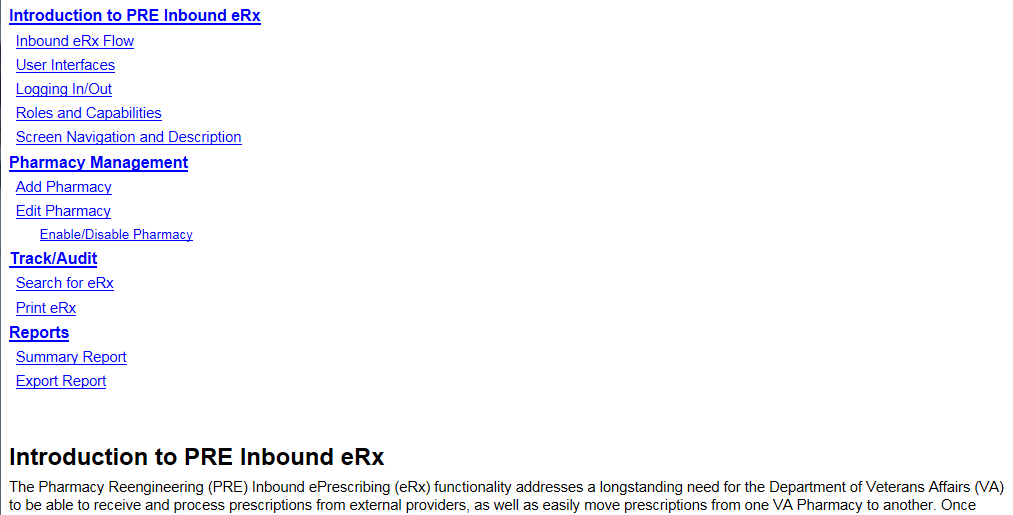


Table 21: Help Screen Description

| Graphical User Interface (GUI) Field | Table (Database Table that field connects to) | Field (Field in Table that the GUI field connects to) | Comments |
| --- | --- | --- | --- |
| N/A | N/A | N/A | N/A |

#### Application Report Interface

This section describes and defines the reports that will be available in the application.

##### Reports Screen

Figure 17 shows the application’s Reports screen. It allows the pharmacist or pharmacy technician to view a summary report of eRx auto-validation checks. Table 16 outlines the fields on the Reports screen.

Figure 17: Reports Screen

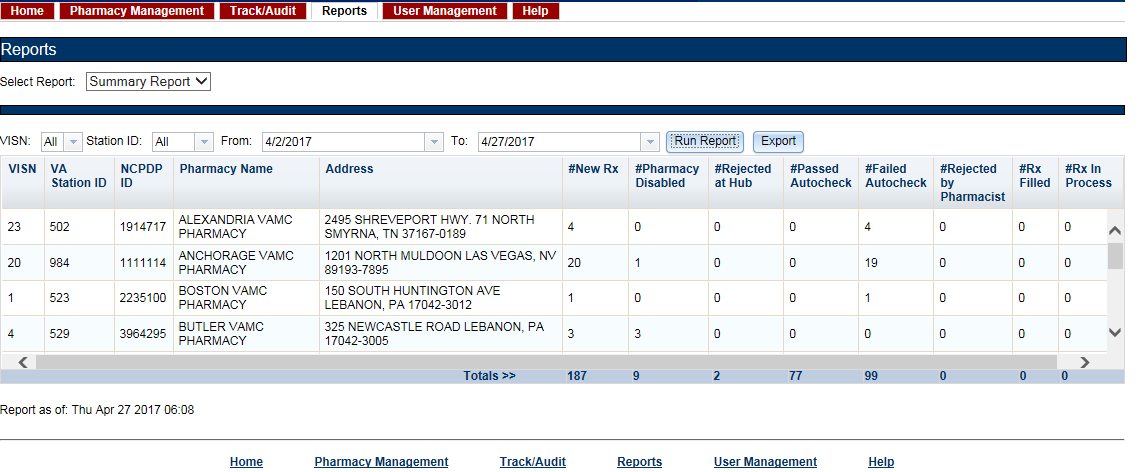


Table 22: Summary Report Screen Description

| Report Column | Data Source |
| --- | --- |
| VISN | SUMMARY\_REPORT\_VW (PHARMACY) |
| VA Station ID | SUMMARY\_REPORT\_VW (PHARMACY) |
| NCPDP ID | SUMMARY\_REPORT\_VW (PHARMACY) |
| Pharmacy Name | SUMMARY\_REPORT\_VW (PHARMACY) |
| Address | SUMMARY\_REPORT\_VW (PHARMACY) |
| #New Rx | SUMMARY\_REPORT\_VW (INBOUND\_NCPDP\_MSG) |
| #Pharmacy Disabled | SUMMARY\_REPORT\_VW (INBOUND\_NCPDP\_MSG) |
| #Rejected at Hub | SUMMARY\_REPORT\_VW (INBOUND\_NCPDP\_MSG) |
| #Passed Autocheck | SUMMARY\_REPORT\_VW (INBOUND\_NCPDP\_MSG) |
| #Failed Autocheck | SUMMARY\_REPORT\_VW (INBOUND\_NCPDP\_MSG) |
| #Rejected by Pharmacist | SUMMARY\_REPORT\_VW (OUTBOUND\_NCPDP\_MSG) |
| #Rx Filled | SUMMARY\_REPORT\_VW (OUTBOUND\_NCPDP\_MSG) |
| #Rx In Process | SUMMARY\_REPORT\_VW (OUTOUND\_NCPDP\_MSG) |

## Conceptual Infrastructure Design

The conceptual infrastructure design is a high-level overview of the infrastructure that will be used to support the Inbound eRx application. Primary emphasis is on the environments that will be required and the locations at which they will be installed.

### System Criticality and High Availability

The Inbound eRx project is part of the overall PRE program and will not have separate system criticality and high availability requirements. As such, Inbound eRx will follow the same criteria that the rest of the PRE applications use.

The Pentaho Data Integration (PDI) (component of the eRx Processing Hub) scalability and high-availability considerations are depicted in Section 3.3.2 below.

Disaster recovery specifications and services will be managed under the current *Austin Information Technology Center (AITC) Disaster Recovery Plan (DRP)*, which is managed by AITC, with limited access to external teams. Please contact AITC directly for a copy of this document.

In line with the requirements established by Pharmacy Benefits Management (PBM) in the product’s Requirements Specification Document (RSD), the system receives “essential support” as defined in the *AITC DRP*.

### Special Technology

This section contains the special technology requirements for the *eRx Processing Hub*.

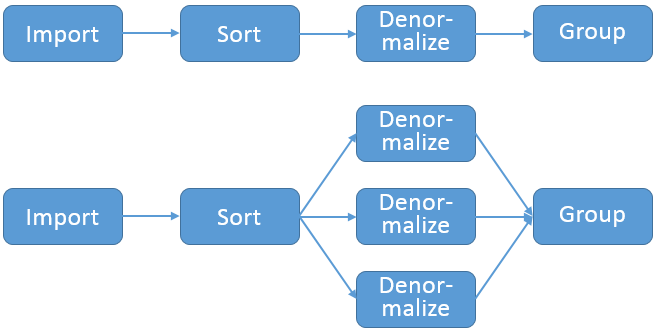
Table 23: Special Technology Requirements

| Special Technology | Description | Notional Location | TRM Status |
| --- | --- | --- | --- |
| Pentaho Data Integration\* | Comprehensive ETL platform with the ability to efficiently manipulate XML messages, out-of-the-box integration with Oracle WebLogic JMS | Co-located as the eRx Processing Hub components | TRM approved, version 6.1 (current version) |

#### Pentaho Data Integration Multi-threaded Architecture

PDI’s streaming engine architecture provides the ability to work with extremely large data volumes, and provides enterprise-class performance and scalability with a broad range of deployment options including dedicated, clustered, and/or cloud-based ETL servers. The architecture allows both vertical and horizontal scaling. The engine executes tasks in parallel and across multiple CPUs on a single machine as well as across multiple servers via clustering and partitioning. Figure 13 depicts a multi-thread example of a single step.

Figure 18: Example Flow with Multiple Threads for a Single Step (Row Denormalizer)



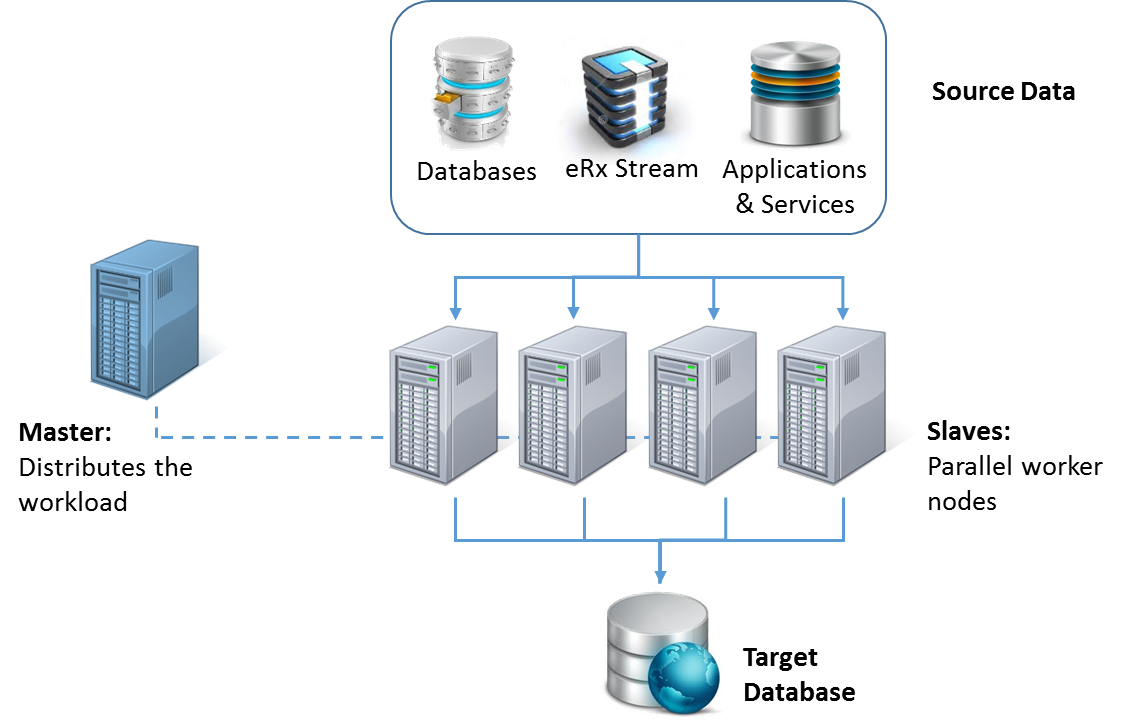
#### Transformation Processing Engine

PDI’s transformation processing engine starts and executes all steps within a transformation in parallel (multi-threaded) allowing maximum usage of available CPU resources. Done by default this allows processing of an unlimited number of rows and columns in a streaming fashion. Furthermore, the engine is 100% metadata driven (no code generation) resulting in reduced deployment complexity. PDI also provides different processing engines that can be used to influence thread priority or limit execution to a single thread which is useful for parallel performance tuning of large transformations. Additional tuning options include the ability to configure streaming buffer sizes, reduce internal data type conversions (lazy conversion), leverage high performance non-blocking I/O (NIO) for read large blocks at a time and parallel reading of files, and support for multiple step copies to allowing optimization of Java Virtual Machine multi-thread usage.

#### Clustering and Partitioning

PDI provides advanced clustering and partitioning capabilities that allow Inbound eRx to scale up the processing of incoming external prescriptions. PDI clusters are built for increasing performance and throughput of data transformations; in particular they are built to perform classic “divide and conquer” processing of data sets in parallel. PDI clusters have a strong master/slave topology. There is one master in cluster, but there can be many slaves. This cluster scheme can be used to distribute the processing workload in parallel appropriately across these multiple systems. Transformations are broken into master/slaves topology and deployed to all servers in a cluster – where each server in the cluster is running a PDI engine to listen, receive, execute, and monitor transformations. It is also possible to define dynamic clusters where the Slave servers are only known at run-time.

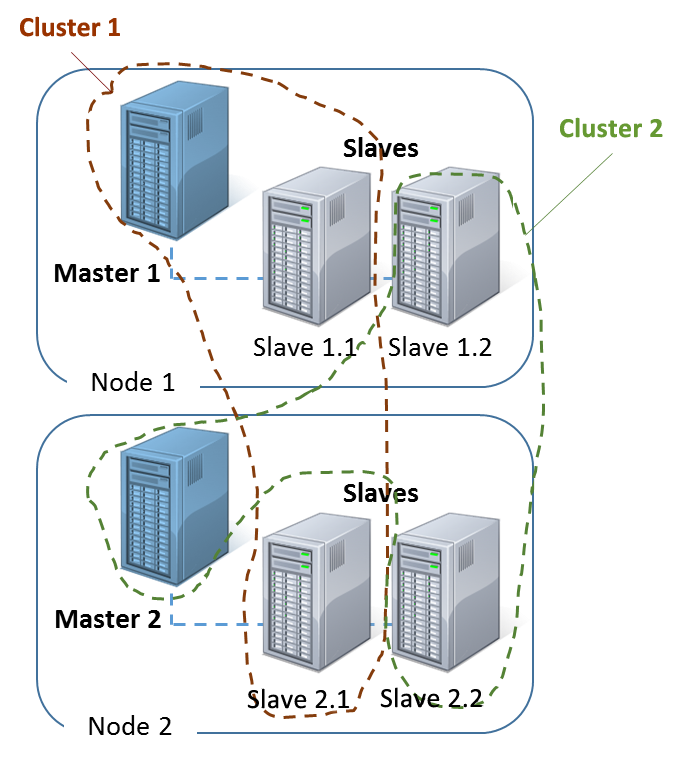
Figure 19: Clustering in Pentaho Data Integration



PDI clusters are completely software based and can be deployed as physical servers or as virtual servers. As described above, PDI clusters consist of a master service and one or more slave services. A master service can be configured to be dynamic, allowing new slave services to self-register with the master service without interrupting transformations currently running on the cluster.

Each master and slave service is implemented by a light-weight web service called Carte. Furthermore, multiple slave services can run on the same server as the master (master server) or on separate servers (slave servers). The figure below depicts a contextual design of fail-over architecture based on two (2) nodes, running two (2) clusters with each cluster having a master service (Master 1 and Master 2, respectively) and two (2) slave services (Slave 1.1 and Slave 1.2 for Master 1, and Slave 2.1 and Slave 2.2 for Master 2). Note that, this depiction is for illustrative purposes and that the PDI component will be implemented within the overall eRx Processing Hub fail-over architecture.

Figure 20: Conceptual Fail-Over PDI Cluster Architecture



### Technology Locations

IEP technology is housed at AITC in Austin, TX.

Table 24: Technology Location Details

| Technology Component  Production 1 - | | Location | | Usage |
| --- | --- | --- | --- | --- |
| Workstations | | Local sites | |  |
| Special Hardware | | None | |  |
| Interface Processors | | None | |  |
| Legacy Mainframe | | None | |  |
| Legacy Application Server | | None | |  |
| Legacy Databases | | None | |  |
| Other | | None | |  |
| Technology Component  Certification | | Location | | Usage |
| N/A | |  | |  |
| Technology Component  Education | | Location | | Usage |
| N/A | |  | |  |
| Technology Component  Test | Location | | Usage | |
| Linux VMs | AITC | |  | |
| Technology Component  Development | Location | | Usage | |
| Linux VMs | AITC | |  | |

### Conceptual Infrastructure Diagram

This section illustrates the conceptual environment diagram.

#### Location of Environments and External Interfaces

Figure 21 is a conceptual environment diagram.

Figure 21: Conceptual Networks and Environments



#### Conceptual Production String Diagram

Figure 22 illustrates the conceptual Production String.

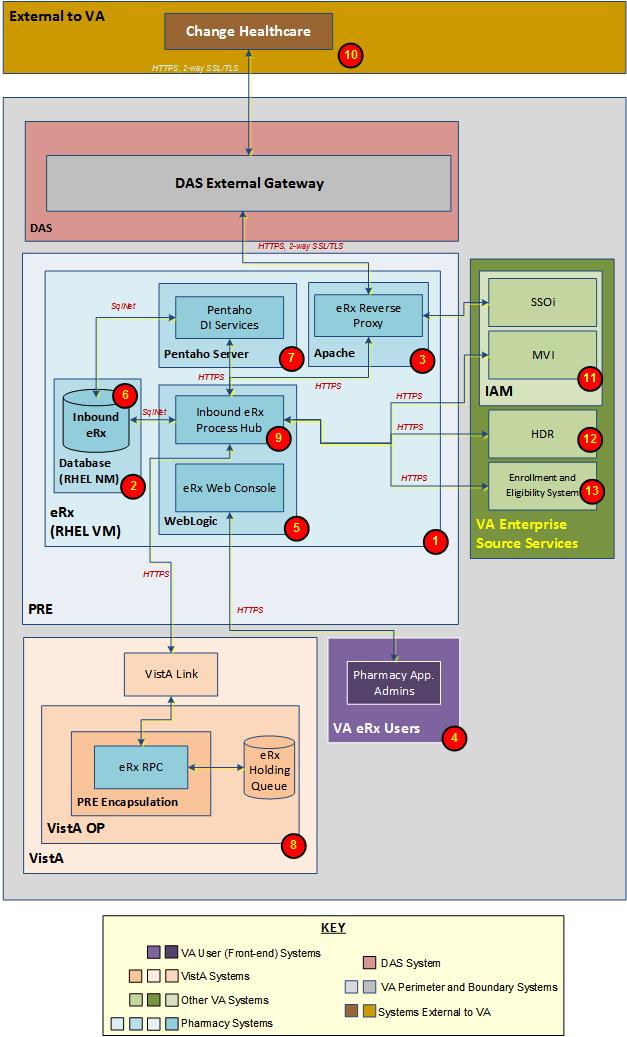
Figure 22: Conceptual Production String Diagram

This image depicts the conceptual Production String

# System Architecture

The system architecture is comprised of the hardware, software, and communication architectures. The hardware architecture describes the physical components needed in the system and their relationship to one another. The software architecture describes the software needed to support the system and what hardware component each software component will exist on. The communication architecture describes the connections needed between the hardware components.

Figure 23: Inbound eRx System Architecture



## Hardware Architecture

The table below references Figure 23 and further describes the hardware components.

| ID | System Component | Comments |
| --- | --- | --- |
| 1 | Redhat Enterprise Linux VM | Apache HTTP, WebLogic 12c, Pentaho Data Integration |
| 2 | Redhat Enterprise Linux VM | Oracle Database 11g |

## Software Architecture

This section describes the overall system software and organization, building upon the high-level application design. The architecture for Inbound eRx is defined as a multi-tiered system consisting of the following logical layers:

* Presentation Layer – Contains user interface components
* Service Layer – Contains business logic components
* Domain Layer – Contains data and database interfacing components
* Interface Layer – Contains interface components, to systems external to the Inbound eRx application

The table below references Figure 23 and further describes the software components.

| ID | System Component | Software | Function |
| --- | --- | --- | --- |
| 3 | Web Server | Apache HTTP 2.2 | Reverse proxy, load balancing and WebLogic high availability clustering support, static content |
| 4 | Client Workstation/Laptop | Internet Explorer 11 or greater | User Interface |
| 5 | Application Server | WebLogic 12.1 | Web Service Orchestration, eRx Initial Validation, RESTful Services supporting UI |
| 6 | Database Server | Oracle 11g | Storage of Inbound eRx messages and application metadata |
| 7 | Data Integration Server | Pentaho Data Integration 6.0 | XML Extraction, Transformation and Loading |
| 8 | VistA Outpatient Pharmacy (OP) | VistA/MUMPS | Existing VistA OP module extended to support Inbound eRx |
| 9 | Processing Hub | JEE Services | Provides data processing and web services |
| 10 | Change Healthcare eRx Network | Externally hosted (non-VA) | Verifies and transmits eRx transactions to/from external provider EHR systems and the Inbound eRx system |
| 11 | Master Veteran Index | Existing VA system | Provides patient lookup service for incoming eRx message validation |
| 12 | Health Data Repository (HDR) | Existing VA system | Provides list of all prescribed medications for individual to fulfill eRx medication history requests.  **NOTE: This is currently out of scope for Version 2.0.** |
| 13 | Enrollment and Eligibility System | Existing VA system | Provides patient enrollment and eligibility status for incoming eRx message validation |

### Components

Inbound eRx interacts with Change Healthcare using DAS as a gateway.

**Inbound eRx Middle Tier Components**

The Inbound eRx middle tier/processing hub consists of the following components and services:

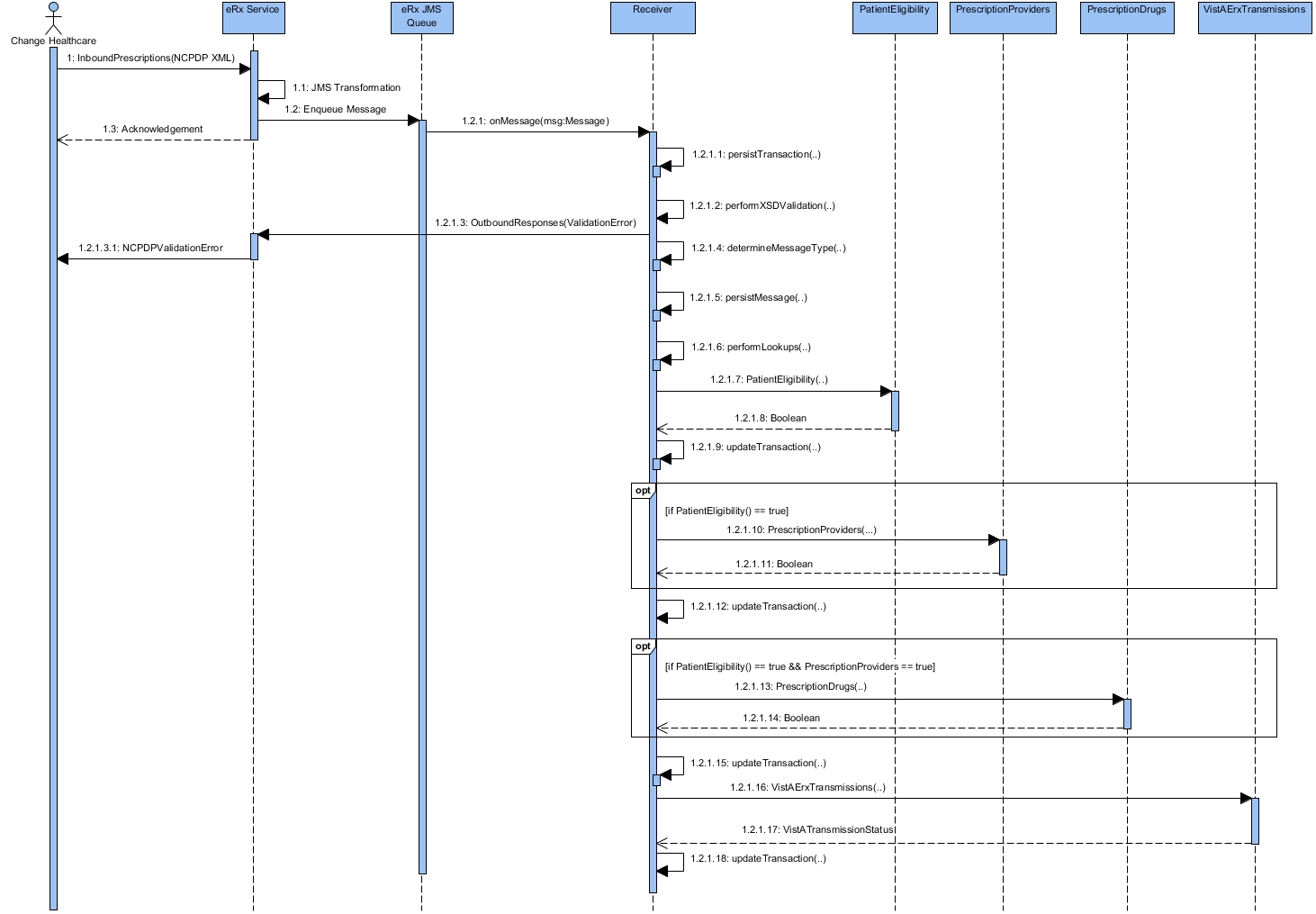
**Receiver**

It is a message processing service hosted on the WebLogic server when the application starts.

**Steps**

1. It picks up messages from Processing Hub JMS message queue and forwards them to Pentaho processing hub, where Pentaho parses the message, stores the prescription into its own XML type table and inserts a record into eRx\_Transaction table.
2. Pentaho DI then calls PatientEligibility, PrescriptionProviders PrescriptionDrugs for each prescription message being processed and updates the eRx\_Transaction table in the process.
3. Finally Pentaho DI calls VistAErxTransmission to transmit that prescription to VistA. Figure 19 shows the sequence diagram for the Receiver.

Figure 24: Inbound eRx – Incoming eRx Flow



**PatientEligibility**

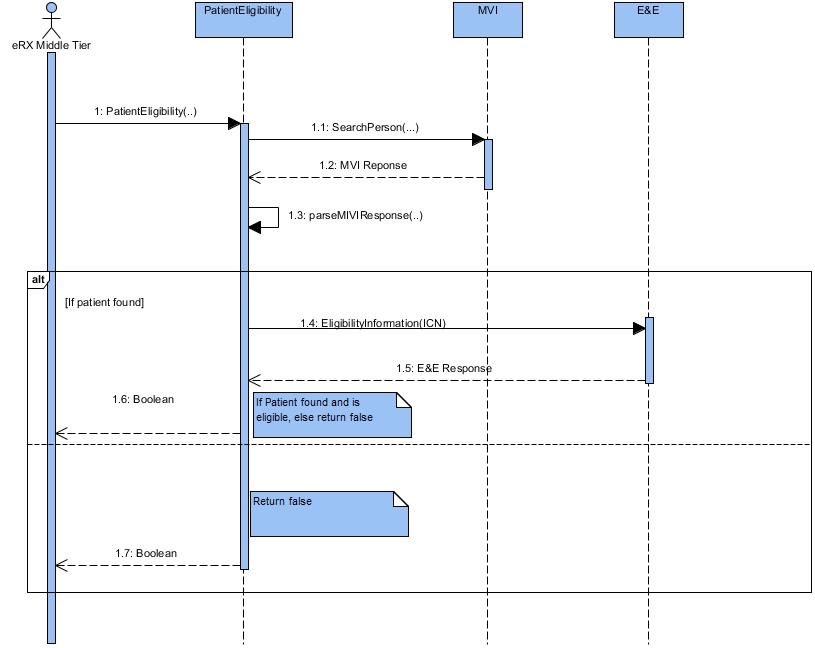
* 1. **Service Description**

This service is called by Pentaho DI and is implemented as a RESTful web service. It is a provider service. Figure 25 shows the sequence diagram for this service.

* 1. **Steps**
     1. It receives patient information Name (FirstName, Middle, Last Name), DOB, Gender, Address and Phone number.
     2. It calls the MVI RESTful web service by passing patient information.
     3. If the patient is not found, it stops further processing, and returns a Boolean flag of false.
     4. If the patient is found, it calls the E&E (Enrollment and Eligibility) web service passing the ICN returned by the MVI web service response and gets the E&E web service response back with the enrollment and eligibility information of the patient and returns a Boolean flag of true.
     5. It returns a Boolean flag of either true or false.

| **Parameter** | **Type** |
| --- | --- |
| patientFirstName | String |
| patientMiddleName | String |
| patientLastName | String |
| gender | String |
| address | String |
| telephone | String |
| dateofBirth | String |

Figure 25: Inbound eRx – PatientEligibility Check Flow



**PrescriptionProviders**

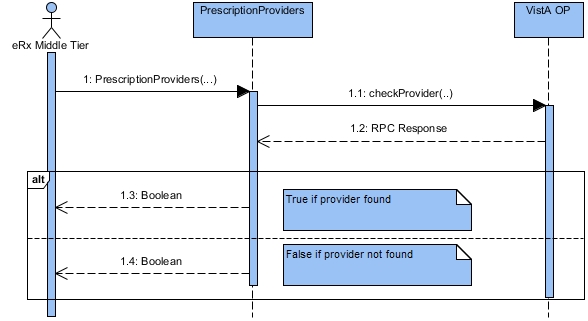
* 1. **Service Description**

This service is called by Pentaho DI and is implemented as a RESTful web service. It is a provider service.It is called even if the PatientEligibility web service returns false in the response as per requirements. Figure 26 shows the sequence diagram for this service.

* 1. **Steps**
     1. It receives provider information i.e., ID, Name, Address and Phone number
     2. It does a RPC call to VistA, passing in the provider information and gets a response back.
     3. If provider is found, it returns a Boolean flag of true.
     4. If provider is not found, it returns a Boolean flag of false.
     5. It returns a Boolean flag of either true or false.

|  |  |
| --- | --- |
| **Parameter** | **Type** |
| providerID | String |
| providerName | String |
| proivderAddress | String |
| telephone | String |

Figure 26: Inbound eRx – Prescription Provider Check Flow



**PrescriptionDrugs**

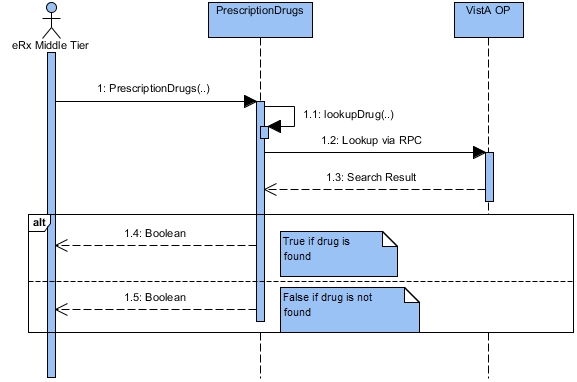
* 1. **Service Description**

This service is called by Pentaho DI and is implemented as a RESTful web service. It is a provider service. It is called even if the PatientEligibility and PrescriptionProviders web service return a false flag in their responses as per requirements. Figure 27 shows the sequence diagram for this service.

* 1. **Steps**
     1. It receives drug information i.e., drugName, NDC, drugClass, generic, ingredients.
     2. It searches local VistA OP for the drug with given information via RPC call and if the drug is found, it will return a Boolean flag of true.
     3. If drug is not found, it will return a Boolean flag of false.
     4. It returns a Boolean flag of either true or false.

| **Parameter** | **Type** |
| --- | --- |
| drugName | String |
| ndc | String |
| drugClass | String |
| generic | String |
| ingredients | String |

Figure 27: Inbound eRx – Prescription Drug Check Flow



**PharmacyTransfers**

**NOTE:** This functionality is currently out of scope in Version 2.0.

* 1. **Service Description**

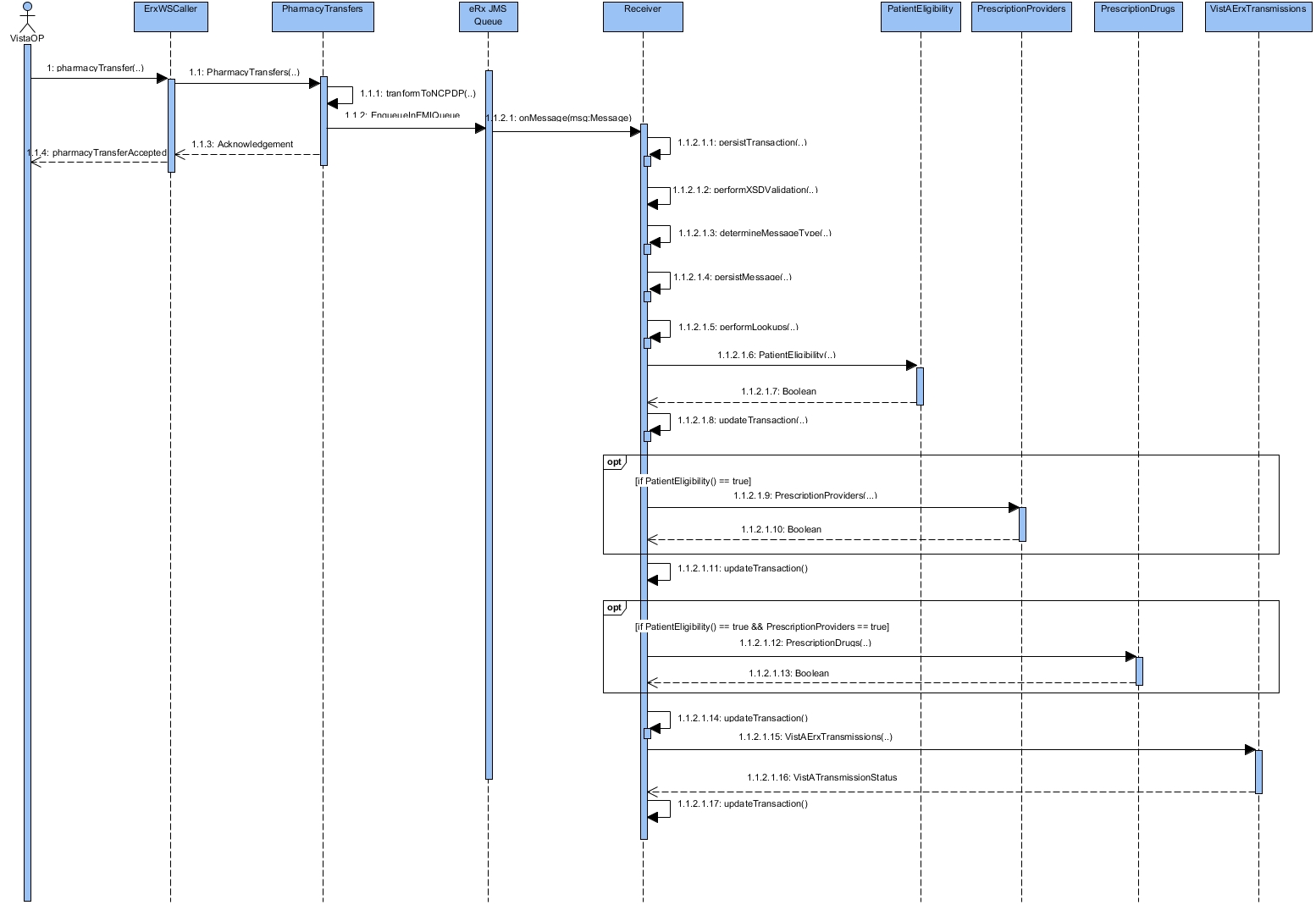
This service is called by VistA OP and is implemented as a RESTful web service. It is a processing service. Figure 28 shows the sequence diagram for this service.

**Steps**

* + 1. It receives prescription details, which is to be transferred to another pharmacy, via VistA OP.
    2. It builds the XML conforming to NCPDP prescription transfers and puts it into Processing Hub JMS Queue where Receiver picks it up and sends to Pentaho middle tier for processing which finally calls VistAErxTransmissions to transfer it.
    3. It returns success in the response; if all processing is successful otherwise it returns an error.

| **Parameter** | **Type** |
| --- | --- |
| pharmacyID | String |
| prescriptonID | String |
| prescriptionDetails | String |
| patientID | String |
| drugID | String |
| prescriptionDate | String |

Figure 28: Inbound eRx – Pharmacy Transfer Flow



**VistAErxTransmissions**

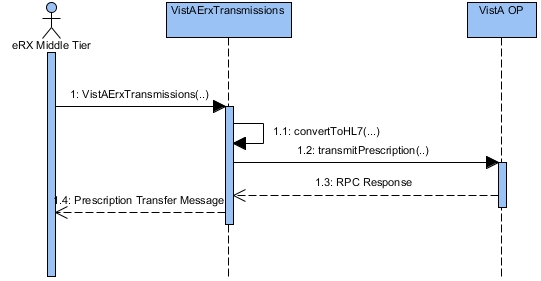
1. **Service Description**

This service is called by Pentaho DI and is implemented as a RESTful web service. It is a processing service. It is called only after PatientEligibility, PrescriptionProviders and PrescriptionDrugs have been called and returned a response. Figure 29 shows the sequence diagram for this service.

1. **Steps**
   * 1. It receives prescription information (shown below) to be transferred to VistA OP.
     2. It translates/converts the data to HL7 for VistA OP with status indicating success or error.
     3. It transmits to VistA OP via RPC call and gets a response back.
     4. It returns success in the response; if all processing is successful otherwise it returns an error.

| **Parameter** | **Type** |
| --- | --- |
| pharmacyName | String |
| pharmacyID | String |
| address | String |
| city | String |
| state | String |
| zipcode | String |
| telephone | String |
| patientID | String |
| patientName | String |
| patientAddress | String |
| patientTelephone | String |
| drugID | String |
| dosageDetails | String |
| prescriptionDate | String |
| prescriptonID | String |
| prescriptionDetails | String |

Figure 29: Inbound eRx – VistAErxTransmission Flow



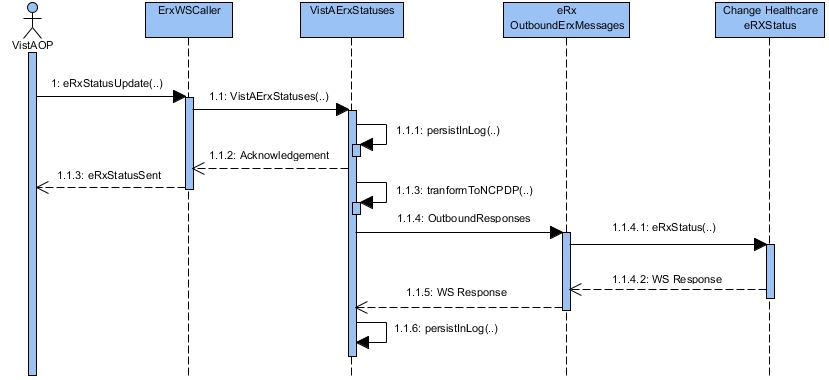
**VistAErxStatuses**

1. **Service Description**

This service is called by VistA OP after a prescription has either been dispensed, updated, cancelled or it causes an error while processing in VistA OP. It sends the data to Change Healthcare and is implemented as a RESTful web service. It is a processing service. Figure 30 shows the sequence diagram for this service.

1. **Steps**
2. It is called by the VistA OP passing the data for a particular prescription after the processing from VistA is finished and status of that particular prescription needs to be sent outbound to the provider.
3. It stores the data into eRX\_Log table.
4. It translates the data into NCPDP format xml.
5. It transmits the data to Change Healthcare via DAS and gets a response back.
6. It returns success in the response; if all processing is successful otherwise it returns an error.

Figure 30: Inbound eRx – eRx Status Transmission Flow



**ClinicalExchangePharmacyInformation**

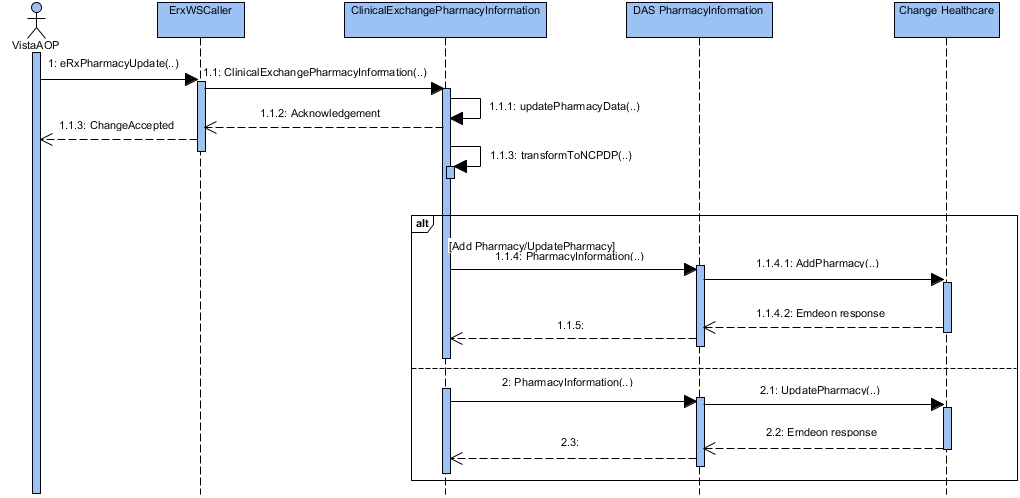
* 1. **Service Description**

This service can be called by VistA OP and is implemented as a RESTful web service. It is a processing service. It is used to send updates to Change Healthcare regarding updating a pharmacy in the Change Healthcare system. Figure 31 shows the sequence diagram for this service.

* 1. **Steps**
     1. It gets the pharmacy information i.e. pharmacyName, pharmacyID, address, city, state, zip code, telephone to be update/added from VistA OP.
     2. It updates the pharmacy information in the VA Pharmacy table in eRx.
     3. It builds the message from the data in NCPDP format and transmits to Change Healthcare via DAS and gets a response back.
     4. It returns success in the response, if all processing is successful, otherwise it returns an error.

| **Parameter** | **Type** |
| --- | --- |
| pharmacyName | String |
| pharmacyID | String |
| address | String |
| city | String |
| state | String |
| zipcode | String |
| telephone | String |

Figure 31: Inbound eRx – Pharmacy Information Update Transmission Flow



### VistA OP Components

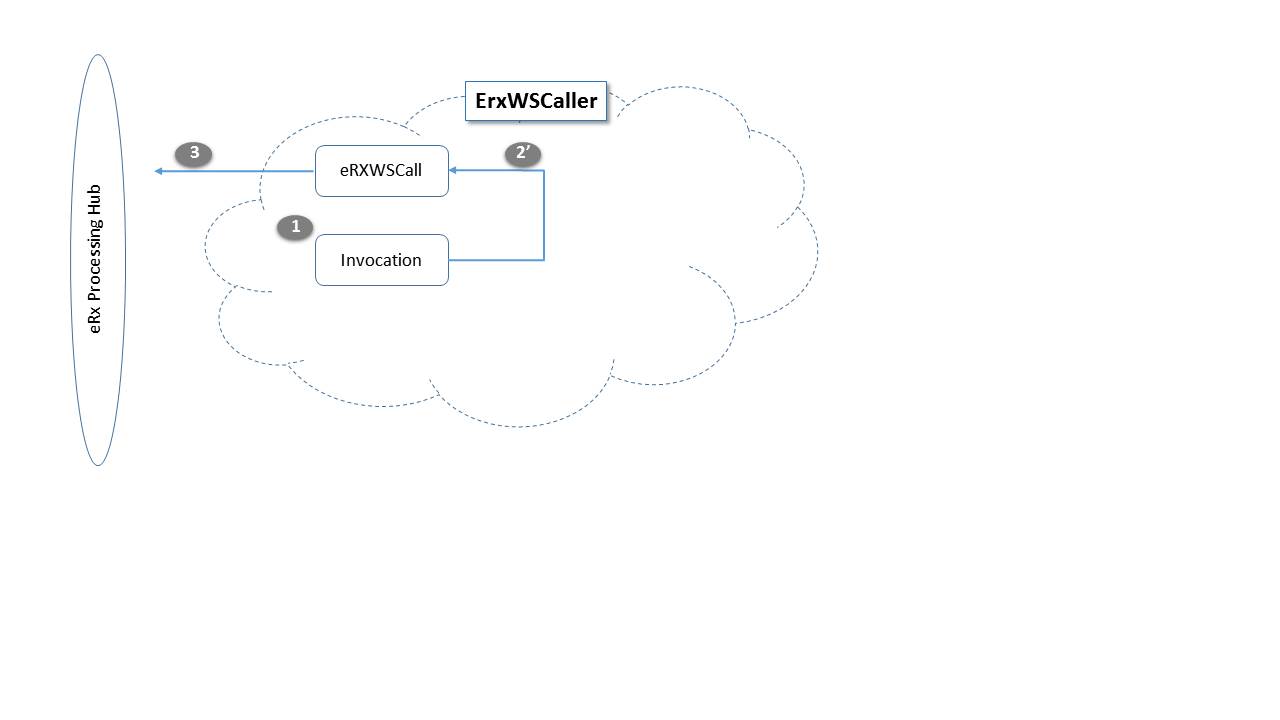
**ErxWSCaller**

1. **Service Description**

It is a MUMPs routine for calling web services in eRx e.g., for updating pharmacy information in Change Healthcare and for updating the status in eRx middle tier during the processing of prescriptions in VistA OP. Upon status change in the eRx Holding Queue, the prescription’s status is changed accordingly in the transaction/processing table in the eRx Processing Hub. It is used as wrapper for other routines to call the VistAErxStatuses web service in eRx Processing Hub. Figure 32 shows the flow for this service.

1. **Steps**
2. Call the VistAErxStatuses – the web service passing in the prescription details
3. Update Status in the eRx Processing Hub – the status of the prescription (C\_TransactionStatus) is updated to status = [the specific status – e.g., fulfilled, rejected, etc.]; no additional services are invoked

Figure 32: ErxWSCaller



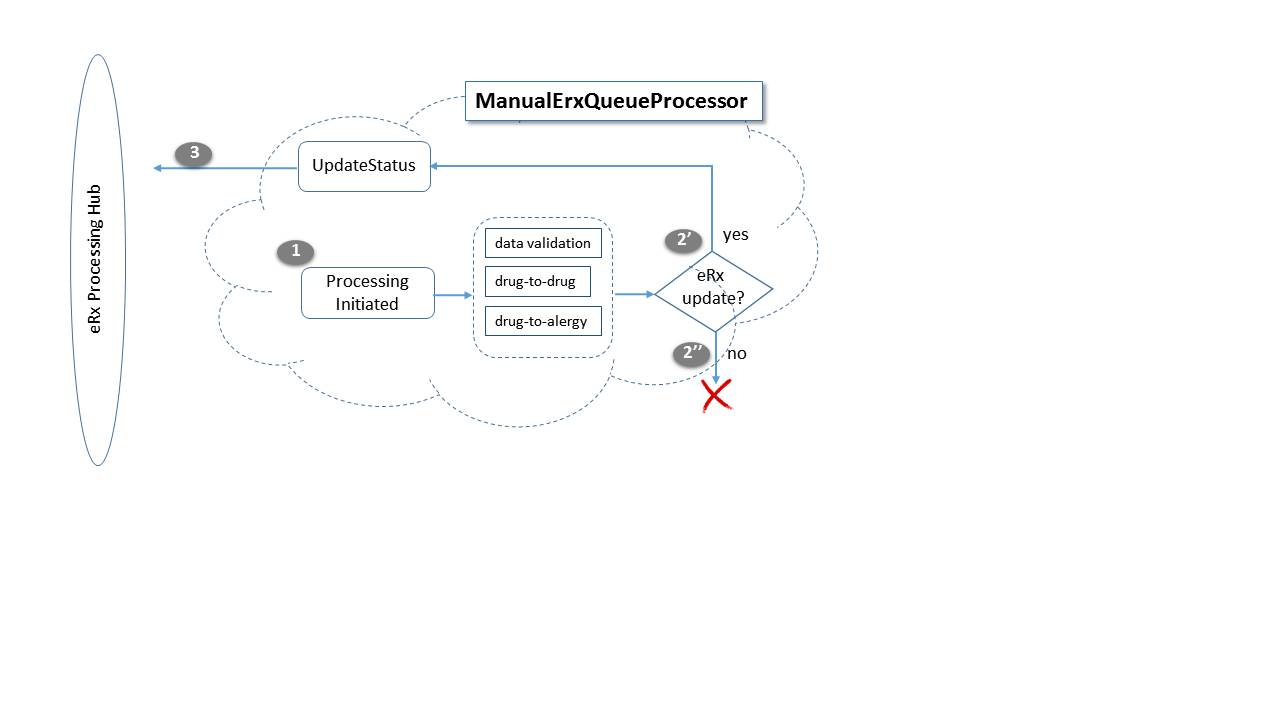
**ManualErxQueueProcessor**

1. **Service Description**

It is a MUMPs routine triggered from the manual validation step for initiating the existing VistA OP processing logic - prescription validation, approval and fulfillment processing; sub-routines include patient/provider/drug/data validation, drug-to-drug check, and fulfillment; eRx Processing Hub status update is invoked, as well. Figure 33 shows the flow for this service.

1. **Steps**
2. Process Prescription Data – upon manual validation, prescriptions are processed in VistA OP; the main functions invoked are validation, approval and fulfillment processing; sub-routines include patient/ provider/ drug/ data validation, drug-to-drug check, and fulfillment
3. Update Status in the eRx Processing Hub – the status of the prescription (C\_TransactionStatus) is updated to status = [the specific status – e.g., fulfilled, rejected, etc.]; no additional services are invoked

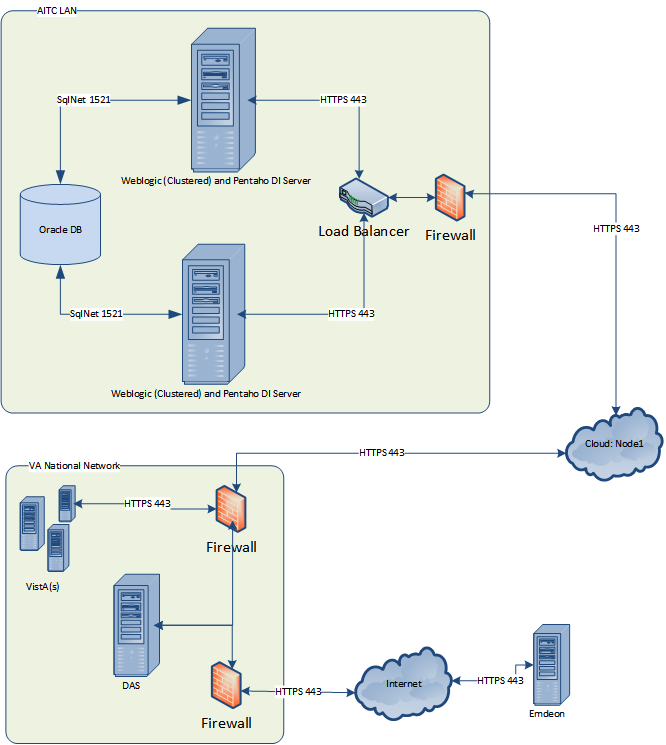
Figure 33: eRx Processing Hub (Manual)



## Network Architecture

The communications architecture for eRx will use a combination of wide area networks (WAN) coupled with local area networks (LAN) as depicted in Figure 34. The LANs will use a star topology with transportation communication protocol and internet protocol (TCP/IP). The LAN will use one gigabit or greater switches where required. All nodes within the confines of the AITC will be connected to the AITC LAN. A firewall is used to provide security and connect each LAN to the cloud. The requirements for each firewall are defined by the system administrator for the respective LANs. The cloud is used to connect the two LANs resulting in a WAN. Access to the eRx is accomplished via a standalone laptop or workstation connected to the cloud using the appropriate browser.

Figure 34: Network Architecture



## Service Oriented Architecture / ESS

Use of Enterprise Shared Services (ESS) is prevalent in the design when noting the use of DAS, MVI, HDR and the E&E Web services. SOA) is also implemented with a design consideration being the potential sharing of Inbound eRx services in the Enterprise. Web services are designed, when practical, to be of potential use to other systems in the Enterprise.

## Enterprise Architecture

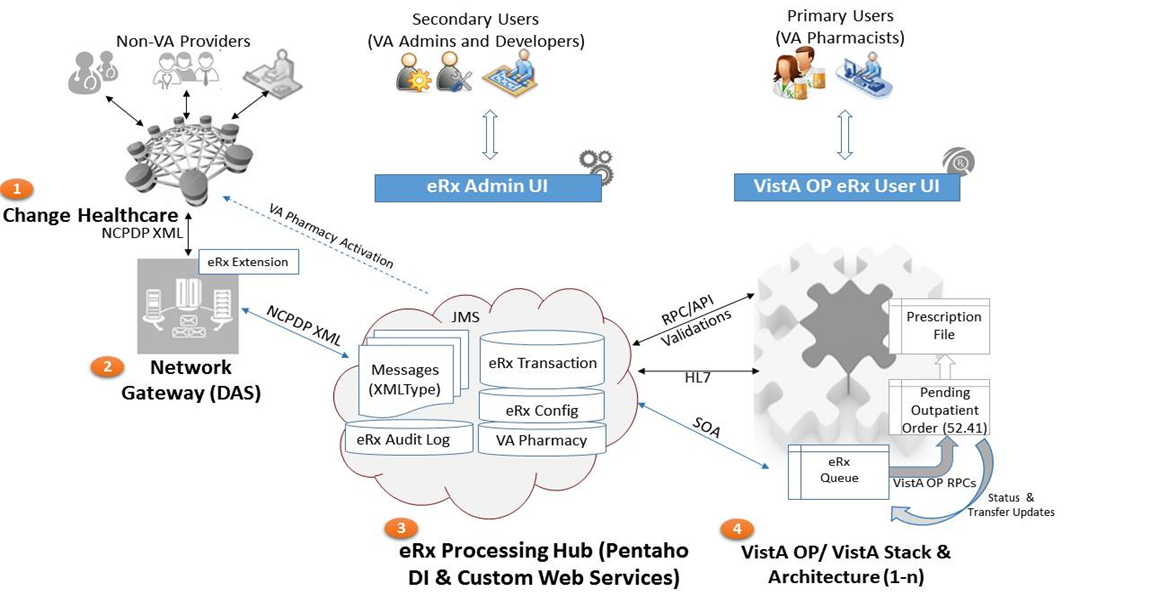
Inbound eRx will target Java SE 8 and WebLogic 12c, Pentaho Data Integration and Oracle Database 11g. The major components planned include Java, Spring, and Hibernate, along with various other VA Technical Reference Model (TRM) approved libraries. A server software list with links to the TRM approvals is available in Appendix A.3.

Inbound eRx will follow the recent Software Assurance (SwA) process of statically scanning the application code using HP Fortify SCA to uncover potential security and privacy issues. In addition to flagging potential security vulnerabilities, Fortify has the ability to recognized potential “leaks” of PII and PHI in application source code. One example being the potential logging of PHI and PII to unsecured application log files.

# Data Design

The Inbound eRx Architecture is composed of four (4) processing tiers: Change Healthcare, Network Gateway (DAS), eRx Processing Hub (Pentaho DI) and VistA OP/ VistA Stack.

Figure 35: Conceptual Data Flow



*Change Healthcare* tier is external to the Inbound eRx architecture and is not required to be covered in this document. The *Network Gateway (DAS)* tier is a straight pass-through proxy with no data storage or message buffering. The focus of the design is the data model of the remaining two (2) tiers – *eRx Processing Hub (Pentaho DI)* and *VistA OP*.

## DBMS Files – eRx Processing Hub

This section depicts the database design for the *eRx Processing Hub* and discusses the physical database organization, the physical data structures and files, the use of indexing as it relates to performance of the database, querying and accessing of the data, and the estimated database size and update frequency.

### Physical Database Organization

A database is a collection of tables. These tables, however, must ultimately be deployed on storage devices and server technology consistent with Oracle 11g - the Relational Database Management System (RDBMS) chosen for deployment.

Oracle RDBMS technology maintains a clear separation between the database (viewed as a set of tables) and the actual physical structures that implement those tables. The following sections outline some of the considerations that eventually drive the low-level physical design for deployment of the database portion of the eRx Processing Hub, noting that these implementation considerations are properly placed in the purview of a DBA. The DBA also has the responsibility to tune the database to improve performance after actual transactional and data-volume metrics are analyzed.

### Physical Data Structures and Files

While the set of physical tables presented in Section 5 of the document represent a relational database design, those models do not constitute a complete physical design of the database. This representation can be used to generate the basic American National Standards Institute (ANSI) SQL Data Definition Language (DDL) that would create the tables, keys, and constraints. That basic DDL would not include specific instructions about actual physical storage disks and servers. For the RDBMS selected for deployment (Oracle 11g), detailed design work is needed in order for precise physical representations to be included in the DDL code base. That is, low-level physical representation is the one area of RDBMS technology that is not vendor neutral.

Using Oracle 11g, some of these low-level design decisions would include:

* Tablespace organization
* Physical file organization

Oracle allows a DBA to organize tables, as well as constructs such as indexes, rollback segments, and temporary work areas, into a set of tablespaces. A tablespace is a parameterized set of one or more physical files in which database tables and other database constructs are stored. One tablespace, called SYSTEM always exists and an Inbound eRx tablespace will be needed.

Design considerations for selecting the set of tablespaces to create and table parameters include:

* The level or degree of stability or volatility of the data for the tables within the tablespace;
* The transaction volume expected on the tables;
* The projected growth rate of the data in the tables;
* The degree of concurrency with which multiple users might access the tables.

General design principles suggest that it is desirable to keep like “things” together and different “things” apart to minimize contention in an Optimal Flexible Architecture (OFA), which is a standard for Oracle database instances. However, in order to make sound decisions with respect to OFA, detailed sizing analysis work must be accomplished. The following high level analysis has been performed:

* VA has estimated that ~1.4M new eRx prescriptions and ~2.6M refills from external providers will be processed annually by Inbound eRx (total of 4M eRx transactions);
* For example, newRx message contains 238 and refill has 395 nodes; we are estimated average message size is 100KB (including response messages);
* Total raw message data size is estimated to be (100 KB \*4M)/106 = 400GB;
* We estimate Oracle 11g compression will reduce the message size 3x (i.e. 67% size reduction) – total annual growth of estimated 132GB for storing eRx raw message data;
* The eRx Processing Hub metadata size is a linear function of the raw data message size; we anticipate it to be 5-10% - expected metadata annual growth of ~6.6 – 13.2GB;
* Based on the above assessment, it is estimated that the annual eRx Processing Hub data growth will not exceed 145GB; higher compression methods (e.g., OLTP, warehouse and archive) can reduce the total annual data further.

In Oracle 11g, each tablespace is a collection of one or more files. Design decisions must be made about how many files are needed and where they are to be located. A file size of 30GB may be appropriate for Inbound eRx. Furthermore, tablespace or SAN encryption of AES256 (or as per the VA security policy) is anticipated to be required to protect PHI/PII data.

Along with the files required for the tablespaces, file space is needed to support redo logs, archive logs, and other database artifacts that allow the RDBMS optimally. Furthermore, the DBA must also consider the nature of the physical hardware disk and server technology. As the differing hardware configurations at different deployment levels are defined, the physical database design can be finalized.

### NCPDP XML Data Analysis

Inbound eRx is primarily processing NCPDP XML messages received from and send to Change Healthcare. XML is being used in a variety of ways – e.g., sometimes XML is constructed from relational data sources, so it is relatively structured, sometimes it is used in the Extract, Transform, Load (ETL) scenario, which is also very structured, sometimes it is used for storing free-form documents that do not have a pre-defined structure. This diversity in XML data leads to significantly different retrieval patterns. The data-centric processes usually have a fixed set of queries, whereas the document-centric users issue more ad-hoc queries.

Since XML usage falls in a broad spectrum, there is no one-size-fits-all storage to offer the best performance and flexibility. Oracle 11g offers different storage models for handling XML data via XMLType – primarily Binary and Object Relational. XMLType is an abstract data type that provides different storage and indexing models to best fit the eRx data and use cases. As an abstract data type, the Inbound eRx applications and database queries gain in flexibility leveraging the same interface for all XMLType operations.

The VA Inbound eRx use case is data-centric and falls in the ETL scenario where XML is used as a staging area for producing relational values from XML, as well as generating XML from relational data. Furthermore, the NCPDP XML data is highly structured and conforms to an XML schema.

As such, the Inbound eRx XMLs are best stored utilizing XMLType Object Relational storage model (or Structured Storage). The data model of Inbound eRx is designed accordingly.

The structured storage is an entity-relationship decomposition of the XML. With Structured Storage, the XMLType data is stored relationally which results in significant performance advantages matching the performance of relational tables. It also provides relational-like schema evolution capability which will allow VA to accommodate changes as they occur in the future.

### Query Patterns

Another important consideration in designing the data model and indexing for Inbound eRx is the query patterns. The VA use case for eRx data is a multi-root hierarchy query pattern. The NCPDP messages are uniquely coded by the MessageID and (for most messages) the RelatedMessageID. It is anticipated that the search queries will be predominantly based on the MessageID and RelatedMessageID. The Object Relational storage model gives the VA the best performance as it performs relational-style lookups starting from any storage table to the parent/child tables.

### Indexing

One of the key topics in database design, especially with respect to SQL-based relational databases, is data integrity; however, database performance is an equally important topic especially for transactional systems such as Inbound eRx.

One of the most common techniques for increasing performance in RDBMS-based database systems is indexing. Assuming an Oracle 11g implementation, the Inbound eRx Processing Hub data model will use primarily B-tree indexing with Structured Storage. This gives VA the best performance for data-centric transactions as the metadata (i.e. tags) is pulled out into column level, and hence queries can do a metadata lookup, which is extremely fast. Furthermore, Oracle supports XMLIndex and other types of indexes with Structured Storage which will be considered as performance tuning tools when the queries are known ahead of time and the list of Xpaths queried is known.

## DBMS Files – VistA Outpatient Pharmacy

This section contains special data design considerations for the *VistA OP eRx Holding Queue*. These considerations are specific to the VistA OP Intersystem’s Cache database.

The VistA’s Cache database is a key-value database engine optimized for high-throughput transaction processing. As such it is in the class of "schema-less", "schema-free," or NoSQL databases. Internally, it stores data in multidimensional hierarchical sparse arrays (also known as key-value nodes, sub-trees, or associative memory). Each array may have up to a predefined number of subscripts, or dimensions. A scalar can be thought of as an array element with zero subscripts. Nodes with varying numbers of subscripts (including one node with no subscripts) can freely co-exist in the same array.

Inbound eRx will create a new eRx Holding Queue file to hold the staging data of auto-validated eRx transactions. Detailed data sizing task is yet to be completed, but the following analysis has been performed:

* The eRx Holding Queue in VistA OP will need to house similar size data as the Messages table in the eRx Processing Hub;
* This is data that is otherwise entered manually in VistA when the paper eRx is processed as per the current process;
* We anticipate the annual data growth to be in-line with the current VistA OP annual growth projections for the paper-based eRx processing.
* Furthermore, we anticipate that performance tuning will be performed with similar considerations as the Pending Outpatient Orders (52.41).

## Non-DBMS Files

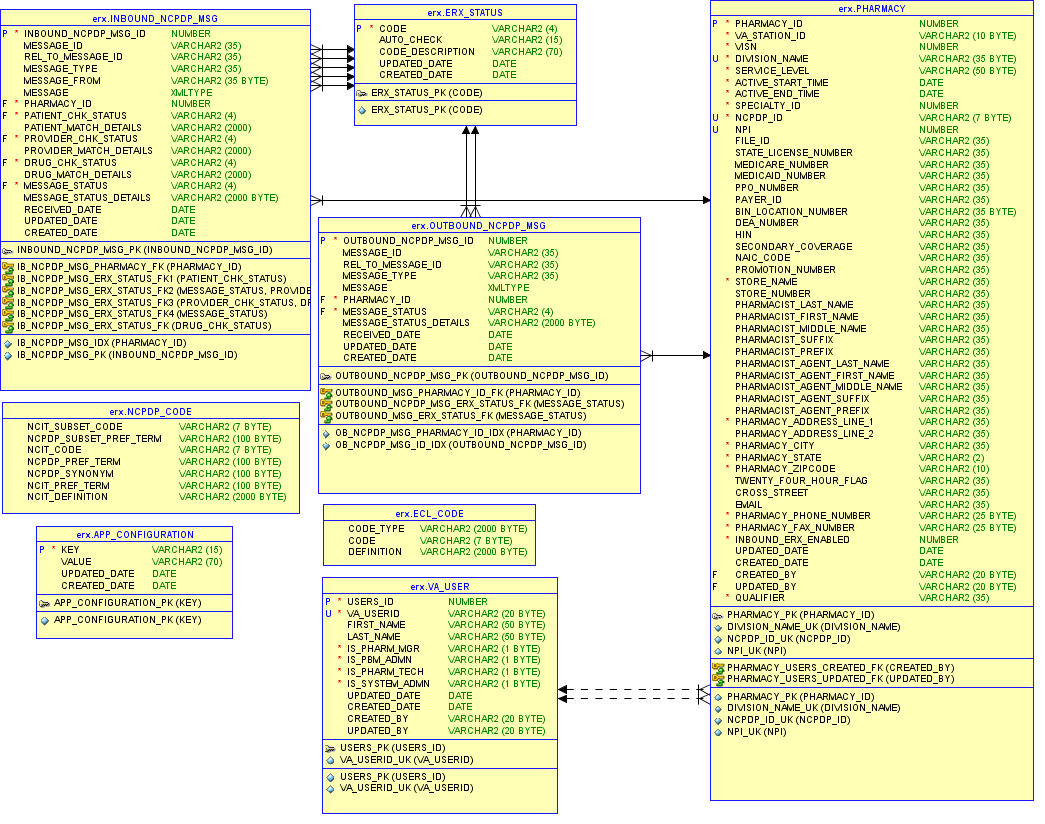
All data within Inbound eRx is saved in the eRx Processing Hub and VistA OP data models, as described in Sections 5.1 and 5.2.

## Data View

### eRx Processing Hub Tier

Figure 36 depicts the entity relationship diagram (ERD) of the eRx Processing Hub Data Model.

Figure 36: eRx Processing Hub Entity Relationship Diagram



The ERD diagram depicts the following tables:

* INBOUND\_NCPDP\_MSG
* OUTBOUND\_NCPDP\_MSG
* PHARMACY
* ERX\_STATUS
* NCPDP\_CODE
* ECL\_CODE
* APP\_CONFIGURATION
* VA\_USER

**Inbound\_Ncpdp\_Msg**

The Inbound\_Ncpdp\_Msg table stores all the necessary data elements to receive, auto-validate and coordinate the processing of the incoming eRx messages. Each transaction has an auto-generated primary key, as well as is coded with the values of MessageID and RelatedMessageID (if exists) message element. The Message column hold the intact NCPDP message as delivered from Change Healthcare. This enables an XQuery to be constructed to pull drug, provider and patient data needed for validation. The table below depicts the fields of the Inbound\_Ncpdp\_Msg table.

Table 25: Inbound\_Ncpdp\_Msg Table Fields

| **#** | **Name** | **Type** | **Sample Values** | **Notes** |
| --- | --- | --- | --- | --- |
| 1 | INBOUND\_NCPDP\_MSG\_ID | NUMBER | 123 | Internally generated unique identifer |
| 2 | MESSAGE\_ID | VARCHAR2(35 BYTE) | 289433527047421978 | Message Identifier |
| 3 | REL\_TO\_MESSAGE\_ID | VARCHAR2(35 BYTE) | 289433527047421978 | Related to Message Identifier |
| 4 | MESSAGE\_TYPE | VARCHAR2(35 BYTE) | NewRx | NCPDPD Message Type |
| 5 | MESSAGE\_FROM | VARCHAR2(35 BYTE) | 1234568594 | Identifier for the sender of the message |
| 6 | MESSAGE | XMLTYPE | <NewRx></NewRx> | Stores the NCPDP XML messages as XMLType Object Relational types |
| 7 | PHARMACY\_ID | NUMBER | 123 | Pharmacy unique identifier from pharmacy table |
| 8 | PATIENT\_CHK\_STATUS | VARCHAR2(4 BYTE) | 0001, 0002, 0101 | Status code for patient auto checking |
| 9 | PATIENT\_MATCH\_DETAILS | VARCHAR2(2000 BYTE) | <patientCheck>  <mvi>  <patientICN></patientICN>  <errorMessage>Error while searching patient: No Record Found</errorMessage>  <patientFirstName></patientFirstName>  <patientMiddleName></patientMiddleName>  <patientLastName></patientLastName>  <patientSSN></patientSSN>  <patientDateOfBirth></patientDateOfBirth>  <patientStationIds></patientStationIds>  <runCount>1</runCount>  </mvi>  <success>false</success>  </patientCheck> | Patient auto checking details, including MVI and E&E checks |
| 10 | PROVIDER\_CHK\_STATUS | VARCHAR2(4 BYTE) | 2001, 2002, 2003 | Status code for provider auto check |
| 11 | PROVIDER\_MATCH\_DETAILS | VARCHAR2(2000 BYTE) | <providerCheck><success>false</success><providerFirstName></providerFirstName><providerLastName></providerLastName><providerIEN></providerIEN><errorMessage>No matching NPI.</errorMessage><runCount>6</runCount></providerCheck> | Provider auto check details from VistA RPC call |
| 12 | DRUG\_CHK\_STATUS | VARCHAR2(4 BYTE) | 1001, 1002, 1003 | Status code for drug auto check |
| 13 | DRUG\_MATCH\_DETAILS | VARCHAR2(2000 BYTE) | <drugCheck><success>false</success><drugName></drugName><drugIEN></drugIEN><errorMessage>No matches found.</errorMessage><runCount>6</runCount></drugCheck> | Drug auto check details from VistA RPC call |
| 14 | MESSAGE\_STATUS | VARCHAR2(4 BYTE) | 0000, 3001, 3002, 3006 | Overall message status |
| 15 | MESSAGE\_STATUS\_DETAILS | VARCHAR2(2000 BYTE) | <vistaSendRetry></vistaSendRetry> | Indicates any details of message delivery attempt to VistA. |
| 16 | RECEIVED\_DATE | DATE | 07-APR-17 | Date/Time message received by eRx Processing Hub |
| 17 | UPDATED\_DATE | DATE | 07-APR-17 | Date/Time record last updated |
| 18 | CREATED\_DATE | DATE | 07-APR-17 | Date/Time record inserted into table |

**Outbound\_Ncpdp\_Msg**

The Outbound\_Ncpdp\_Msg table stores all the necessary data elements to receive, auto-validate and coordinate the processing of the outgoing eRx messages. Each transaction has an auto-generated primary key, as well as is coded with the values of MessageID and RelatedMessageID (if exists) message element. The Message column hold the intact NCPDP message as delivered from VistA or the eRx Processing Hub. The table below depicts the fields of the Outbound\_Ncpdp\_Msg table.

Table 26: Outbound\_Ncpdp\_MsgTable Fields

| **#** | **Name** | **Type** | **Sample Values** | **Notes** |
| --- | --- | --- | --- | --- |
| 1 | OUTBOUND\_NCPDP\_MSG\_ID | NUMBER | 123 | Internally generated unique identifer |
| 2 | MESSAGE\_ID | VARCHAR2(35 BYTE) | 289433527047421978 | Message Identifier |
| 3 | REL\_TO\_MESSAGE\_ID | VARCHAR2(35 BYTE) | 289433527047421978 | Related to Message Identifier |
| 4 | MESSAGE\_TYPE | VARCHAR2(35 BYTE) | RefillRequest | NCPDPD Message Type |
| 5 | MESSAGE | XMLTYPE | <RefillRequest></RefillRequest> | Stores the NCPDP XML messages as XMLType Object Relational types |
| 6 | PHARMACY\_ID | NUMBER | 123 | Pharmacy unique identifier from pharmacy table |
| 7 | MESSAGE\_STATUS | VARCHAR2(4 BYTE) | 4001, 4002 | Overall message status |
| 8 | MESSAGE\_STATUS\_DETAILS | VARCHAR2(2000 BYTE) | <sendRetry></sendRetry> | Indicates any details of message delivery attempt to Change Healthcare. Not currently used in this version. |
| 9 | RECEIVED\_DATE | DATE | 07-APR-17 | Date/Time message received by eRx Processing Hub |
| 10 | UPDATED\_DATE | DATE | 07-APR-17 | Date/Time record last updated |
| 11 | CREATED\_DATE | DATE | 07-APR-17 | Date/Time record inserted into table |

**Pharmacy**

The VA\_Pharmacy stores VA Pharmacy data and the corresponding VistA instance. Each transaction has an auto-generated ID\_Pharmacy primary key. The following depicts the fields of the VA\_Pharmacy table.

Table 27: Pharmacy Table Fields

| **#** | **Name** | **Type** | **Sample Values** | **Notes** |
| --- | --- | --- | --- | --- |
| 1 | PHARMACY\_ID | NUMBER | 123 | Internally generated unique identifer |
| 2 | VA\_STATION\_ID | VARCHAR2(10 BYTE) | 984AB | The station ID of each pharmacy |
| 3 | VISN | NUMBER | 12 | The Veterans Integrated Service Network of the pharmacy |
| 4 | DIVISION\_NAME | VARCHAR2(35 BYTE) | DANVILLE VAMC PHARMACY | VA internal Pharmacy Name |
| 5 | SERVICE\_LEVEL | VARCHAR2(50 BYTE) | 3 | Indicates which electronic message types are supported |
| 6 | ACTIVE\_START\_TIME | DATE | 01-JAN-99 | Date pharmacy became active in the clearinghouse |
| 7 | ACTIVE\_END\_TIME | DATE | 01-JAN-36 | Date pharmacy became inactive in the clearinghouse |
| 8 | SPECIALTY\_ID | NUMBER | 8 | Indicates the nature of the pharmacy (Retail, Mail Order, etc.) |
| 9 | NCPDP\_ID | VARCHAR2(7 BYTE) | 6464536 | Seven-digit NCPDP Provider ID Number |
| 10 | NPI | NUMBER | 8376254598 |  |
| 11 | FILE\_ID | VARCHAR2(35 BYTE) | 234234 | Pharmacy Identifier |
| 12 | STATE\_LICENSE\_NUMBER | VARCHAR2(35 BYTE) | NC0230823 | State license number of pharmacy |
| 13 | MEDICARE\_NUMBER | VARCHAR2(35 BYTE) | 213123123 | Medicare number of pharmacy |
| 14 | MEDICAID\_NUMBER | VARCHAR2(35 BYTE) | 745734788 | Medicaid number of pharmacy |
| 15 | PPO\_NUMBER | VARCHAR2(35 BYTE) | 12341235125 | Pharmacy Identifier |
| 16 | PAYER\_ID | VARCHAR2(35 BYTE) | 08970961243 | Pharmacy Identifier |
| 17 | BIN\_LOCATION\_NUMBER | VARCHAR2(35 BYTE) | 648363 | Pharmacy Identifier |
| 18 | DEA\_NUMBER | VARCHAR2(35 BYTE) | CW738493 | Pharmacy's DEA Number |
| 19 | HIN | VARCHAR2(35 BYTE) | YU567890FR9 | Health Industry Number |
| 20 | SECONDARY\_COVERAGE | VARCHAR2(35 BYTE) |  | Pharmacy Identifier |
| 21 | NAIC\_CODE | VARCHAR2(35 BYTE) | 446110 | NAICS Code of Pharmacy |
| 22 | PROMOTION\_NUMBER | VARCHAR2(35 BYTE) | 234 | Pharmacy Identifier |
| 23 | STORE\_NAME | VARCHAR2(35 BYTE) | Danville Outpatient VA Pharmacy | Name of Pharmacy |
| 24 | STORE\_NUMBER | VARCHAR2(35 BYTE) | 34234 | Pharmacy Identifier |
| 25 | PHARMACIST\_LAST\_NAME | VARCHAR2(35 BYTE) | Smith | Pharmacist’s last name |
| 26 | PHARMACIST\_FIRST\_NAME | VARCHAR2(35 BYTE) | John | Pharmacist’s first name |
| 27 | PHARMACIST\_MIDDLE\_NAME | VARCHAR2(35 BYTE) | Doe | Pharmacist’s middle name |
| 28 | PHARMACIST\_SUFFIX | VARCHAR2(35 BYTE) | Jr. | Pharmacist’s suffix |
| 29 | PHARMACIST\_PREFIX | VARCHAR2(35 BYTE) | Dr., Mr. | Pharmacist’s prefix |
| 30 | PHARMACIST\_AGENT\_LAST\_NAME | VARCHAR2(35 BYTE) | Smith | Pharmacist’s agent last name |
| 31 | PHARMACIST\_AGENT\_FIRST\_NAME | VARCHAR2(35 BYTE) | John | Pharmacist’s agent first name |
| 32 | PHARMACIST\_AGENT\_MIDDLE\_NAME | VARCHAR2(35 BYTE) | Doe | Pharmacist’s agent middle name |
| 33 | PHARMACIST\_AGENT\_SUFFIX | VARCHAR2(35 BYTE) | Jr. | Pharmacist’s agent suffix |
| 34 | PHARMACIST\_AGENT\_PREFIX | VARCHAR2(35 BYTE) | Dr. | Pharmacist’s agent prefix |
| 35 | PHARMACY\_ADDRESS\_LINE\_1 | VARCHAR2(35 BYTE) | 123 South St. | Pharmacy’s first line |
| 36 | PHARMACY\_ADDRESS\_LINE\_2 | VARCHAR2(35 BYTE) | Suite 1000 | Pharmacy’s address second line |
| 37 | PHARMACY\_CITY | VARCHAR2(35 BYTE) | Madison | Pharmacy’s city |
| 38 | PHARMACY\_STATE | VARCHAR2(2 BYTE) | WI | Pharmacy’s state |
| 39 | PHARMACY\_ZIPCODE | VARCHAR2(10 BYTE) | 99999 | Pharmacy’s zipcode |
| 40 | TWENTY\_FOUR\_HOUR\_FLAG | VARCHAR2(35 BYTE) | Y,N | Indicates if pharmacy is open 24hours |
| 41 | CROSS\_STREET | VARCHAR2(35 BYTE) | Main Street | Major intersection nearest to pharmacy |
| 42 | EMAIL | VARCHAR2(35 BYTE) |  | Pharmacy’s email address |
| 43 | PHARMACY\_PHONE\_NUMBER | VARCHAR2(25 BYTE) |  | Pharmacy's phone number |
| 44 | PHARMACY\_FAX\_NUMBER | VARCHAR2(25 BYTE) |  | Pharmacy's fax number |
| 45 | INBOUND\_ERX\_ENABLED | NUMBER | Y, N | Indicates whether or not pharmacy is currently receiving and processing electronic messages |
| 46 | UPDATED\_DATE | DATE | 07-APR-17 | Date/Time record last updated |
| 47 | CREATED\_DATE | DATE | 07-APR-17 | Date/Time record inserted |
| 48 | CREATED\_BY | VARCHAR2(20 BYTE) | 07-APR-17 | User who inserted record |
| 49 | UPDATED\_BY | VARCHAR2(20 BYTE) | 07-APR-17 | User who updated record |

**Erx\_Status**

The Erx\_status stores status codes and descriptions used for each eRx transaction. These are used on the inbound\_ncpdp\_msg and outbound\_ncpdp\_msg tables to indicate drug, patient and provider auto check status as well as general message processing and delivery status. The following below depicts the fields of the Erx\_Status table.

Table 28: Erx\_Status Table Fields

| **#** | **Name** | **Type** | **Sample Values** | **Notes** |
| --- | --- | --- | --- | --- |
| 1 | CODE | VARCHAR2(4 BYTE) | 0000, 0106, 3005 | Code value |
| 2 | AUTO\_CHECK | VARCHAR2(15 BYTE) | MESSAGE, PATIENT, DRUG | Code type |
| 3 | CODE\_DESCRIPTION | VARCHAR2(70 BYTE) | PATIENT\_MATCH\_NOT\_FOUND, VISTAOP\_DELIVERY\_SUCCESSFUL | Description of code |
| 4 | UPDATED\_DATE | DATE | 07-APR-17 | Date/Time record last updated |
| 5 | CREATED\_DATE | DATE | 07-APR-17 | Date/Time record inserted |

**Ncpdp\_Code**

The Ncpdp\_Code stores NCIT (National Cancer Institute component terminology within FMT(Federal Medication Terminologies)) codes and values that are needed to translate incoming code values prior to sending to VistA and for dislay in the message detail screen of the eRx Processing Hub GUI. The following below depicts the fields of the Ncpdp\_Code table.

Table 29: Ncpdp\_Code Table Fields

| **#** | **Name** | **Type** | **Sample Values** | **Notes** |
| --- | --- | --- | --- | --- |
| 1 | NCIT\_SUBSET\_CODE | VARCHAR2(7 BYTE) | C89510 | Code subset |
| 2 | NCPDP\_SUBSET\_PREF\_TERM | VARCHAR2(100 BYTE) | NCPDP QuantityUnitOfMeasure Terminology | Terminology subset |
| 3 | NCIT\_CODE | VARCHAR2(7 BYTE) | C64696 | Code value |
| 4 | NCPDP\_PREF\_TERM | VARCHAR2(100 BYTE) | Caplet, Gram, Inhaler | NCPDP Preferred Term |
| 5 | NCPDP\_SYNONYM | VARCHAR2(100 BYTE) | Pen, Unit, mEq | Synonym |
| 6 | NCIT\_PREF\_TERM | VARCHAR2(100 BYTE) | Caplet Dosing Unit | NCIt preferred term |
| 7 | NCIT\_DEFINITION | VARCHAR2(2000 BYTE) | A dosing unit equal to the amount of active ingredient(s) contained in a caplet. | NCIt definition |

**Ecl\_Code**

The ecl\_code stores ECL (NCPDP External Code List) codes and definitions needed to translate ECL codes to definitions for dislay in the message detail screen of the eRx Processing Hub GUI. The following below depicts the fields of the Ecl\_Code table.

Table 30: Ecl\_CodeTable Fields

| **#** | **Name** | **Type** | **Sample Values** | **Notes** |
| --- | --- | --- | --- | --- |
| 1 | CODE\_TYPE | VARCHAR2(2000 BYTE) | 1131 – Code List Qualifier – Response Code - RES Segment | Type of code |
| 2 | CODE | VARCHAR2(7 BYTE) | AA | Code value |
| 3 | DEFINITION | VARCHAR2(2000 BYTE) | Patient unknown to the Prescriber | Code definition |

**App\_Configuration**

The App\_Configuration stores key/values pairs used internally for configuration within the eRx Processing Hub. The following below depicts the fields of the App\_Configuration table:

Table 31: App\_Configuration Table Fields

| **#** | **Name** | **Type** | **Sample Values** | **Notes** |
| --- | --- | --- | --- | --- |
| 1 | KEY | VARCHAR2(15 BYTE) | ERX\_URL | Key code |
| 2 | VALUE | VARCHAR2(70 BYTE) | http://sample.erx.url/ | Key value |
| 3 | UPDATED\_DATE | DATE | 07-APR-17 | Date/Time record last updated |
| 4 | CREATED\_DATE | DATE | 07-APR-17 | Date/Time record inserted into table |

**Va\_User**

The eRx\_Log stores audit log data and status changes for each eRx transaction. Each transaction has an auto-generated ID\_Log primary key and ID\_eRx foreign key. The following below depicts the fields of the VA\_User table:

Table 32: Va\_User Table Fields

| **#** | **Name** | **Type** | **Sample Values** | **Notes** |
| --- | --- | --- | --- | --- |
| 1 | USERS\_ID | NUMBER | 123 | Internally generated Id |
| 2 | VA\_USERID | VARCHAR2(20 BYTE) | VHAXXXXXXXX | VA assigned network Id |
| 3 | FIRST\_NAME | VARCHAR2(50 BYTE) | John | First name of user |
| 4 | LAST\_NAME | VARCHAR2(50 BYTE) | Smith | Last name of user |
| 5 | IS\_PHARM\_MGR | VARCHAR2(1 BYTE) | N, Y | Pharmacy Manager role indicator |
| 6 | IS\_PBM\_ADMN | VARCHAR2(1 BYTE) | N, Y | PBM role indicator |
| 7 | IS\_PHARM\_TECH | VARCHAR2(1 BYTE) | N, Y | Pharmacist/Pharmacist Technician role indicator |
| 8 | IS\_SYSTEM\_ADMN | VARCHAR2(1 BYTE) | N, Y | System Administrator role indicator |
| 9 | UPDATED\_DATE | DATE | 07-APR-17 | Date/Time record last updated |
| 10 | CREATED\_DATE | DATE | 07-APR-17 | Date/Time record inserted |
| 11 | CREATED\_BY | VARCHAR2(20 BYTE) | VHAXXXXXXXX | User who inserted record |
| 12 | UPDATED\_BY | VARCHAR2(20 BYTE) | VHAXXXXXXXX | User who updated record |

### VistA OP eRx Holding Queue (#52.49)

This section depicts the structure of the VistA OP eRx Holding Queue File (#52.49). The following data elements are expected to be required in the VistA OP eRx Holding Queue:

.01 - NCPDP or Emdeon message id/number

.02 - Related or Parent message ID

.03 - Message Date/Time

.04 - External patient (pointer to eRx External Patient file #tbd)

.041 - Vista (linked) Patient (muiltiple pointer #2 or eRx External Patient file)

.05 - Hospital Location

.06 - Pharmacy (pointer to the OUTPATIENT SITE file #59, matched on NCPDPID)

.07 - Message Type (refill, partial, cancel, change etc.)

.08 - External Order Number (Prescriber Order Number)

.09 - Vista Pending Order (pointer to the PENDING OUTPATIENT ORDER FILE 52.41 - if needed)

.091 - Vista Order Number (pointer to the ORDERS file #100 - once established)

.092 - Vista Prescription Number (pointer to the PRESCRIPTION file #52)

1 - eRx Order status (pending/approved/processed etc)

1.1 - Provider Status (auto validation)

1.11 - Provider status (Manual validation)

1.2 - Drug Status (auto validation)

1.21 - Drug Status (manual validation)

1.3 - Patient Status (auto validation)

1.31 - Patient status (manual Validation)

2 - External Provider (pointer to the eRx External User file)

2.1 - External Pharmacist (pointer to the eRx External User file)

2.2 - VA (matched) provider (pointer?)

2.3 - TO/FROM Qualifier (P,C,M,D,ZZZ - Pharmacy, Clinic, Mailbox, Prescriber, Mutually Defined)

3 - Order Type (similar to order type in 52.41?)

4 - Quantity Timing sub-file (similar to 52.41?)

5 - Effective Date

6 - Nature of Order?? This may not be available in the same format.

7 - Provider Comments

8 - DRUG INFORMATION

.01 - External Drug/Supply (Medication Prescribed) (Drug Description) (alpha numeric 105 char max)

.02 - Matched Drug/Supply (VA)

1 - Drug Coded

.01 - Product Code (AN - 35)

.02 - Product Code Qualifier (ND - NATIONAL DRUG CODE (NDC), MF - MANUFACTURER, UP - UNIVERSAL PRODUCT CODE (UPC)/ - SET OF CODES)

.03 - Strength (AN - 70)

.04 - DrugDBCode (AN - 35)

.05 - Form Source Code

.06 - Form Code

.07 - Strength Source Code

.08 - Strength Code (AA, AB, AC)

.09 - DEA Schedule

1 - Quantity

.01 - Quantity

.02 - Code List Qualifier (38-Original Qty, 40-Remaining Qty, 87-Quantity Received, QS- Quantity sufficient as determined by the dispensing pharmacy. Quantity to be based on established dispensing protocols between the prescriber and pharmacy/pharmacist.)

.03 - Unit Source Code (Strength Code (aa, ab, ac))

.04 - Potency Unit Code (AN 15 MAX)

2.1 - Days Supply (numeric (3))

2.2 - Directions (AN 140 MAX)

3 - NOTE (AN 210 MAX)

4.1 - REFILLS (NUMERIC)

4.2 - REFILL QUALIFIER (R - Number of Refills 'R' implies an Original Dispensing in addition to the Quantity specified in DRU 060-I009-02- 6060 or PRN - as many refills as that patient needs for 1 year)

4.3 - Substitutions (0 - No product selection indicated, 1 - Substitution Not allowed by Prescriber)

4.4 - Written Date (Date or Date/time)

4.5 - Last Fill Date

4.6 - Expiration Date

4.7 - Effective Date

4.8 - Period End

4.9 - Delivered On Date

4.11- Date Validated

5 - Diagnoses (multiple)

.01 - Diagnosis

.02 - Clinical Information Qualifier (1 - Prescriber/presriber supplied, 2 - Pharmacy Inferred)

.03 - Diagnosis Type (Primary/Secondary)

.04 - Qualifier (E|F|M|ABF|DX|ICD9|ICD10 - per definition)

6 - Drug Use evaluation (do we even need this in VistA? if so i will map it out)

.01 -

7.1 - Drug Coverage status (PR|AP|PA|NF|NR|DC|UN|ST|SI|CP|SP) - may create a drug coverage status file since there are quite a few definitions, we will then point this field to that file.

8.1 - Prior Authorization (value)

8.2 - Prior Authorization Qualifier (G1|PD|D3|94|0B|1C|1D|1E|1G|1M|2U|BO|DH|HI|NC|NF|PD|SY|HPI|G1|ZZ) - - may create a prior authorization qualifier file since there are quite a few definitions, we will then point this field to that file.

8.3 - Prior Authoriztion Status (A|D|R|N|F)

8.4 - DO NOT FILL (Y/N - 1/0)

8.5 - NEEDED NO LATER THAN (DATE/TIME?)

8.6 - TIMEZONE

8.7 - TIMZONE DIFFERENCE QUANTITY

9 - Structured SIG (word processing)

10 - Order Checks (if applicable - multiple)

11.1 - Service Connected

11.2 - Pickup Routing

11.3 - Rx refill request (indicates the rx number for refill request from CPRS - may need to be modified or removed)

11.4 - reason order created (may not need this. could possibly load 52.41 based on programmatic check)

11.5 - Date/Time Flagged

12.1 - Patient Instructions

12.2 - Patient Instructions Flag

12.3 - Expanded Patient Instructions

13.1 - Signature Status?? will we need this, or will it get signed through CPRS?

14.1 - External Pharmacy (pointer to the eRx external pharmacy file #tbd)

14.2 - External Pharmacist (pointer to the eRx external Pharmacist file #tbd)

15 - Store Name (this may be able to go into the eRx Pharmacy file)

16 - Transferred to other VA Pharmacy (Pointer to Institution file #4) - for future use when transferring from VA to VA.

17 - Transferred to External Pharamcy (pointer to eRx External Pharamcy file) - for future use when transferring from VA to VA.

18 - Observation Notes (MULTIPLE?)

.01 - Observation ID (COUNTER OR NUMBER)

.02 - Observation Source code (AA|AB|AC)

.03 - Measurement Unit code

1 - Observation notes (word processing)

19 - Payer Identification

.01 - BIN Location Number

.02 - Mutually Defined?? what is this?

.03 - Payer Name

.04 - Card Holder ID

.05 - Card Holder Last name

.06 - Card Holder First Name

.07 - Card Holder Middle Name

.08 - Card Holder Suffix

.09 - Card Holder Prefix

1.1 - Group ID

100 - Processing Errors (multiple)

.01 - Error ID

.02 - Error type (Drug/Provider/patient/transfer etc)

.03 - Error Source (Vista/eRx Processing Hub)

.04 - Status

.05 - Resolved by

.06 - Date/time resolved

### eRx External Pharmacy File (#52.47)

The ERX External Pharmacy File # 52.47 is comprised of provider information from each incoming New eRx.

.01 - External pharmacy name/identifier

.02 - NCPDP ID

.03 - NPI

.04 - DEANumber

.05 – Store Name

1.1 - Address Line 1

1.2 - Address Line 2

1.3 - City

1.4 - State

1.5 - Zip

1.6 - Type ('VA' FOR VA FACILITY; 'EF' FOR EXTERNAL FACILITY)

1.7 - Place/Location Qualifier (if applicable)

1.8 - Specialty

3 - Communication Numbers (multiple)

3.01 - Number

3.02 - Type (BN - BEEPER, CP - CELLULAR, EM - ELECTRONIC MAIL, FX - FAX, HP - HOME, NP - NIGHT, TE - TELEPHONE, WP - WORK, SET OF CODES)

### eRx External Patient File (#52.46)

The ERX External Patient file # 52.46 is comprised of patient information from each incoming New eRx. Each patient record here is unique based on the combination of parameters: Name, DOB and Gender. The parameter list may be enhanced to add Address Line 1.

The following information about each patient will be captured: (Note that fields may be added / removed as necessary).

.01 - NAME

.02 - LAST NAME

.03 - FIRST NAME

.04 - MIDDLE NAME

.05 - SUFFIX

.06 - PREFIX

.07 - GENDER

.08 - DATE OF BIRTH

.09 - ERX EXTERNAL PHARMACY

1.1 - FILE ID

1.2 - MEDICAL RECORD ID #

1.3 - ACCOUNT NUMBER

1.4 - SSN

1.5 - LINKED VISTA PATIENT

1.6 - PLACE/LOCATION QUALIFIER

2 - COMMUNICATION

3.1 - ADDRESS LINE 1

3.2 - ADDRESS LINE 2

3.3 - CITY

3.4 - STATE

3.5 - ZIP CODE

### eRx External Person File (#52.48)

The ERX External Person File # 52.48 is comprised of provider information from each incoming New eRx. Each provider record here is unique based on the combination of parameters: Name and NPI

.01 - Name

.02 – NPI

### eRx Service Reason Codes File (#52.45)

The ERX Service Reason Codes File #52.45 is comprised of Service Reason Codes and their corresponding translations.

eRx Service Reason Codes file (#52.45)

.01 - code

.02 - text

# Detailed Design

This section describes the proposed design in detail.

## Hardware Detailed Design

Table 33: Development/SQA Detailed VM Requirements

| **VM** | **RAM (GB)** | **Space\* (GB)** | **CPUs** | **OS** | **VM Description/Use/DNS Required** |
| --- | --- | --- | --- | --- | --- |
| 1 | 16 | 300 | 4 | RHEL 7.X | DEV 1 DB Server running Oracle |
| 2 | 16 | 300 | 4 | RHEL 7.X | DEV 2 DB Server running Oracle |
| 3 | 16 | 300 | 4 | RHEL 7.X | SQA 1 DB Server running Oracle |
| 4 | 16 | 300 | 4 | RHEL 7.X | SQA 2 DB Server running Oracle |
| 5 | 16 | 300 | 4 | RHEL 7.X | DEV1 AP Server running Apache/WebLogic |
| 6 | 16 | 300 | 4 | RHEL 7.X | DEV 2 AP Server running Apache/WebLogic |
| 7 | 16 | 300 | 4 | REHL 7.X | SQA 1 AP Server running Apache/WebLogic |
| 8 | 16 | 300 | 4 | REHL 7.X | SQA 2 AP Server running Apache/WebLogic |
| Total | 128 | 2400 | 32 | 8 |  |

Table 34: Staging Detailed VM Requirements

| **VM** | **RAM (GB)** | **Space\* (GB)** | **CPUs** | **OS** | **VM Description/Use/DNS Required** |
| --- | --- | --- | --- | --- | --- |
| 1 | 16 | 800 | 4 | RHEL 7.X | STAGING DB Server running Oracle |
| 2 | 16 | 300 | 4 | RHEL 7.X | STAGING Application Server running Apache/WebLogic |
| 3 | 16 | 300 | 4 | RHEL 7.X | STAGING Application Server running Apache/WebLogic |
| Total | 48 | 1400 | 16 | 3 |  |

Table 35: Pre-Production Detailed VM Requirements

| **VM** | **RAM (GB)** | **Space\* (GB)** | **CPUs** | **OS** | **VM Description/Use/DNS Required** |
| --- | --- | --- | --- | --- | --- |
| 1 | 16 | 1300 | 4 | RHEL 7.X | PRE-PRODUCTION DB Server running Oracle |
| 2 | 16 | 300 | 4 | RHEL 7.X | PRE-PRODUCTION Application Server running Apache/WebLogic |
| 3 | 16 | 300 | 4 | RHEL 7.X | PRE-PRODUCTION Application Server running Apache/WebLogic |
| Total | 48 | 1900 | 12 | 3 |  |

Table 36: Production Detailed VM Requirements

| **VM** | **RAM (GB)** | **Space\* (GB)** | **CPUs** | **OS** | **VM Description/Use/DNS Required** |
| --- | --- | --- | --- | --- | --- |
| 1 | 16 | 1300 | 4 | RHEL 7.X | PRODUCTION DB Server running Oracle |
| 2 | 16 | 300 | 4 | RHEL 7.X | PRODUCTION Application Server running Apache/WebLogic |
| 3 | 16 | 300 | 4 | RHEL 7.X | PRODUCTION Application Server running Apache/WebLogic |
| Total | 48 | 1900 | 12 | 3 |  |

Table 37: Server Software List

| **#** | **Product Name** | **Version** | **TRM Approved (Y/N)** | **Description/Use/TRM link** |
| --- | --- | --- | --- | --- |
| 1 | Apache HTTP Server | 2.2 | Y |  |
| 2 | WebLogic Server | 12.1 | Y |  |
| 3 | Oracle Database | 11.2 | Y |  |
| 4 | Pentaho Data Integration | 6.0 | Y |  |

## Software Detailed Design

This section provides conceptual and detailed information associated with the design of the software being delivered.

### Conceptual Design

Please refer to [Section 3](#_Conceptual_Design) for details on the conceptual design.

#### Dependencies and Constraints

The following list details the dependencies and constraints assumed by the Inbound eRx project:

* The solution must comply with the appropriate business architecture.
* The application developers must refactor components into partitioned logical layers (i.e. presentation, business, logical, data access) with each layer containing functionality specifically related to that layer.
* The application layers must expose interface components that promote loose coupling between layers.
* The application developers must refactor components so that business logic and data management processes are fully decoupled from each other.
* The application user interfaces must follow enterprise common UI templates and style guidelines.
* The application must store data on enterprise servers and not on end-user devices or workstations.
* Inbound eRx development team must develop unit tests for application functions and publicly exposed methods.
* The application must implement procedures for communicating and resolving unhandled exceptions.
* The application shall scale horizontally and operate on a series of loosely coupled commodity platforms.
* The application must scale out based on the need.
* Application business logic must exist statelessly (i.e., session information is not stored within the business logic layer).
* Application user interfaces must comply with Section 508.
* The application must comply with VA Enterprise Architecture published data standards (HL7, NCPDP).
* Inbound eRx must identify and leverage authoritative information sources for data retrieval and manipulation.
* Information captured by the application must syntactically and semantically harmonize with the VA Enterprise CDM.
* The application must operate optimally using information from the authoritative source or receive permission for caching data locally.
* The team must configure system/and server platforms used by the application using standard system images published in the current VA Release Architecture.
* The team must publish relational and object oriented databases utilized by the solution in the current VA Release Architecture.
* The solution design must operate on the standard OI&T defined virtual environments.
* The team must base application production capacity requirements on workload analysis, simulated workload benchmark tests, or application performance models.
* The team must base application storage capacity requirements on detailed capacity analysis and/or models.
* The team must design the solution to operate within the current VA LAN and WAN network configurations.
* The deployment environment must meet the performance and downtime monitoring requirements of the solution.
* The team and data center must develop and provision a disaster recovery plan.
* All critical infrastructure components (including data) must be located at multiple physical locations.
* The application backup and restore solution must meet data recovery requirements and recovery time objectives.
* The application UIs must exist as browser-based UIs and roll and scroll in VistA.
* The application must adhere to all applicable information security rules contained within the VA Policy Handbook (6500).
* If the application is deployed externally, it must follow all guidelines for using commercial partners.
* The application must establish secure access paths for accessing the application and application data.
* The solution must document specific reasons for all limited, external access to data, including the need to know along with security, privacy and other legal restrictions.
* The solution must implement appropriate controls that prevent unwarranted disclosure of sensitive, PII, or PHI.
* The team must base all system interfaces (both external and internal) implemented by the solution on open standards such as SOAP, REST, JMS, MQ, HTTPS and standard message formats such as HL7and NCPDP.
* The solution must utilize the Enterprise IAM service for authentication and authorization.
* The solution must access available enterprise information through services.
* The VA TRM must identify all products and standards used by this solution as permissible for usage.

### Specific Requirements

#### Database Repository

Refer to Section 5 of this document for details on the database design.

#### System Features

Please refer to the *Inbound eRx Requirements Specification Document* at the following SharePoint link for the system features and functional requirements for the project:

##### Routines (Entry Points)

This section lists the new routines developed as part of the Inbound eRx functionality.

Table 38: Routine: PSOERX

| Routines | Activities | | | |
| --- | --- | --- | --- | --- |
| **Routine Name** | PSOERX | | | |
| **Enhancement Category** | New | Modify | Delete | No Change |
| **RTM** | List View of the Holding Queue – Story # 417222  Search Functionality on the List View – Story # 417233  Sort Functionality on the List View – Story # 417238 | | | |
| **Related Options** |  | | | |

| Related Routines | Routines “Called By” | Routines “Called” |
| --- | --- | --- |
|  |  |  |

| Routines | Activities | | | | |
| --- | --- | --- | --- | --- | --- |
| **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) |
| --- |
| **PSOERX ;ALB/BWF - eRx Utilities/RPC's ; 8/3/2016 5:14pm**  **;;7.0;OUTPATIENT PHARMACY;\*\*467\*\*;DEC 1997;Build 42**  **;**  **EN(SRCH,SORTT) ; -- main entry point for PSO ERX HOLDING QUEUE**  **S PSOSRCH=$S($D(SRCH):1,1:0)**  **S PSOSRT=$S($D(SORTT):1,1:0)**  **D:'$D(PSOPAR) ^PSOLSET I '$D(PSOPAR) D MSG^PSODPT G EX**  **I '$G(PSOPINST) D INST^PSOORFI2 I $G(PSOIQUIT) K PSOIQUIT G EX**  **D EN^VALM("PSO ERX HOLDING QUEUE")**  **Q**  **;**  **HDR ; -- header code**  **S VALMHDR(1)="PSO ERX HOLDING QUEUE"**  **; TODO?? - bwf - setting valmbck="R" and trying to use RE^VALM4 was not**  **refreshing the list area.. dev environment problem?**  **I VALMBCK="R" K @VALMAR S VALMBCK="" D INIT**  **Q**  **;**  **INIT ; -- init variables and list array**  **N LINE,LINEVAR,X,SGLOB,RXSTAT,ERXIEN,PATIEN,PATNM,PTDOB,EXDS,EXPRIEN,EX**  **PRNM,AUTOST,MANST,PTLOOP,ERXDAT**  **N DEASCH,ERX,ERXDT,ERXEDT,ERXPR,PTDOBE,ERXSTAT,ERXISTAT,EF,ERXIENS,RXST**  **ATN**  **S EF=52.49**  **S SGLOB=$NA(^TMP("PSOERX",$J)) K @SGLOB**  **;S PSOSRCH=$S($G(SRCHT):1,1:0)**  **;S PSOSRT=$S($G(SORTT):1,1:0)**  **; initialize LINE**  **I '$G(PSOPINST) Q**  **S LINE=0**  **; TODO - look into date or inverse date cross reference for processing.**  **Look back 18 months.**  **S RXSTAT=0 F S RXSTAT=$O(^PS(52.49,"E",PSOPINST,RXSTAT)) Q:'RXSTAT D**  **.S RXSTATN=$$GET1^DIQ(52.45,RXSTAT,.01,"E")**  **.; do not show reject, remove, or processed prescriptions.**  **.I RXSTATN="RJ"!(RXSTATN="RM")!(RXSTATN="PR") Q**  **.;Q:$D(^PS(52.45,"C",RXSTATN,RXSTAT))**  **.;Q:$D(^PS(52.45,"C",RXSTATN,RXSTAT))**  **.;Q:RXSTATN="PR"**  **.S ERXIEN=0 F S ERXIEN=$O(^PS(52.49,"C",RXSTAT,ERXIEN)) Q:'ERXIEN D**  **..S ERXIENS=ERXIEN\_","**  **..K ERXDAT D GETS^DIQ(52.49,ERXIENS,".03;.04;4.9;3.1;2.1;1;1.2;1.3","IE**  **","ERXDAT")**  **..S PATIEN=$G(ERXDAT(EF,ERXIENS,.04,"I"))**  **..I $D(SRCH(1)),PATIEN'=$P($G(SRCH(1)),U) Q**  **..; patient ien missing, do not process??**  **..Q:'PATIEN**  **..S DEASCH=$G(ERXDAT(EF,ERXIENS,4.9,"E"))**  **..I $D(SRCH(7)),+DEASCH'="","C48672C48675C48676C48677C48678"'[DEASCH Q**  **..I $D(SRCH(8)),"C48672C48675C48676C48677C48678"[DEASCH Q**  **..; patient name/dob**  **..S PATNM=$G(ERXDAT(EF,ERXIENS,.04,"E"))**  **..;S PATNM=$$GET1^DIQ(52.49,ERXIEN,.04,"E")**  **..S PTDOB=$$GET1^DIQ(52.46,PATIEN,.08,"I")**  **..I PTDOB]"" S PTDOBE=$$FMTE^XLFDT(PTDOB,"2D")**  **..I 'PTDOB S PTDOB="N/A"**  **..I $D(SRCH(2)),PTDOB'=$P($G(SRCH(2)),U,2) Q**  **..; external drug/supply**  **..S EXDS=$G(ERXDAT(EF,ERXIENS,3.1,"E"))**  **..;S EXDS=$$GET1^DIQ(52.49,ERXIEN,3.1,"E")**  **..I $D(SRCH(6)),EXDS'[$P($G(SRCH(6)),U) Q**  **..; external provider ien and name**  **..S EXPRIEN=$G(ERXDAT(EF,ERXIENS,2.1,"I"))**  **..;S EXPRIEN=$$GET1^DIQ(52.49,ERXIEN,2.1,"I")**  **..I $D(SRCH(4)),EXPRIEN'=$P($G(SRCH(4)),U,2) Q**  **..; if there is no external provider, quit - TODO - may need to find a**  **way to view and deal with these instances.**  **..S EXPRNM=$G(ERXDAT(EF,ERXIENS,2.1,"E")) Q:'$L(EXPRNM)**  **..; auto and manual validation status**  **..S AUTOST=$G(ERXDAT(EF,ERXIENS,1.2,"E"))**  **..S MANST=$G(ERXDAT(EF,ERXIENS,1.3,"E"))**  **..S ERXSTAT=$G(ERXDAT(EF,ERXIENS,1,"E"))**  **..S ERXISTAT=$G(ERXDAT(EF,ERXIENS,1,"I"))**  **..I $D(SRCH(5)),ERXSTAT'=$P(SRCH(5),U,2) Q**  **..; date ERX was received**  **..S ERXDT=$G(ERXDAT(EF,ERXIENS,.03,"I")),ERXEDT=$$FMTE^XLFDT($P(ERXDT,"**  **."),"2D")**  **..I $D(SRCH(3)),(ERXDT<$P($G(SRCH(3)),U,1))!(ERXDT>$P($G(SRCH(3)),U,2))**  **Q**  **..I $G(SORTT) D Q**  **...I SORTT=1 S @SGLOB@(PATNM,EXPRNM,ERXIEN)=PATNM\_U\_PTDOBE\_U\_EXDS\_U\_EXP**  **RNM\_U\_AUTOST\_U\_MANST\_U\_ERXEDT\_U\_ERXSTAT**  **...I SORTT=2 S @SGLOB@(PTDOB,PATNM,ERXIEN)=PATNM\_U\_PTDOBE\_U\_EXDS\_U\_EXPR**  **NM\_U\_AUTOST\_U\_MANST\_U\_ERXEDT\_U\_ERXSTAT**  **...I SORTT=3 S @SGLOB@(ERXDT,PATNM,ERXIEN)=PATNM\_U\_PTDOBE\_U\_EXDS\_U\_EXPR**  **NM\_U\_AUTOST\_U\_MANST\_U\_ERXEDT\_U\_ERXSTAT**  **...I SORTT=4 S @SGLOB@(EXPRNM,PATNM,ERXIEN)=PATNM\_U\_PTDOBE\_U\_EXDS\_U\_EXP**  **RNM\_U\_AUTOST\_U\_MANST\_U\_ERXEDT\_U\_ERXSTAT**  **...; TODO PHARMACY LOCATION - NOT READY YET MAY NEED TO ADD FIELD/INDEX**  **??**  **...I SORTT=5 S @SGLOB@(ERXSTAT,PATNM,ERXIEN)=PATNM\_U\_PTDOBE\_U\_EXDS\_U\_EX**  **PRNM\_U\_AUTOST\_U\_MANST\_U\_ERXEDT\_U\_ERXSTAT**  **...I SORTT=6 S @SGLOB@(EXDS,PATNM,ERXIEN)=PATNM\_U\_PTDOBE\_U\_EXDS\_U\_EXPRN**  **M\_U\_AUTOST\_U\_MANST\_U\_ERXEDT\_U\_ERXSTAT**  **...I SORTT=7,DEASCH]"" S @SGLOB@("CS",PATNM,ERXIEN)=PATNM\_U\_PTDOBE\_U\_EX**  **DS\_U\_EXPRNM\_U\_AUTOST\_U\_MANST\_U\_ERXEDT\_U\_ERXSTAT**  **...I SORTT=8 S @SGLOB@("NCS",PATNM,ERXIEN)=PATNM\_U\_PTDOBE\_U\_EXDS\_U\_EXPR**  **NM\_U\_AUTOST\_U\_MANST\_U\_ERXEDT\_U\_ERXSTAT**  **..S @SGLOB@(PATNM,EXPRNM,ERXIEN)=PATNM\_U\_PTDOBE\_U\_EXDS\_U\_EXPRNM\_U\_AUTOS**  **T\_U\_MANST\_U\_ERXEDT\_U\_ERXSTAT**  **S PTLOOP="" F S PTLOOP=$O(@SGLOB@(PTLOOP)) Q:PTLOOP="" D**  **.S ERXPR="" F S ERXPR=$O(@SGLOB@(PTLOOP,ERXPR)) Q:ERXPR="" D**  **..S ERX=0 F S ERX=$O(@SGLOB@(PTLOOP,ERXPR,ERX)) Q:'ERX D**  **...S LINE=LINE+1,LINEVAR=""**  **...S ERXDAT=$G(@SGLOB@(PTLOOP,ERXPR,ERX))**  **...S LINEVAR=$$SETFLD^VALM1(LINE\_".",LINEVAR,"#")**  **...S LINEVAR=$$SETFLD^VALM1($P(ERXDAT,U),LINEVAR,"PATIENT NAME")**  **...S LINEVAR=$$SETFLD^VALM1($P(ERXDAT,U,2),LINEVAR,"DOB")**  **...S LINEVAR=$$SETFLD^VALM1($P(ERXDAT,U,3),LINEVAR,"DRUG")**  **...S LINEVAR=$$SETFLD^VALM1($P(ERXDAT,U,4),LINEVAR,"PROVIDER")**  **...S LINEVAR=$$SETFLD^VALM1($P(ERXDAT,U,8),LINEVAR,"STA")**  **...S LINEVAR=$$SETFLD^VALM1($P(ERXDAT,U,7),LINEVAR,"RECORD DATE")**  **...D SET^VALM10(LINE,LINEVAR,ERX)**  **I LINE=0 S LINE=LINE+1 D SET^VALM10(LINE,"No results for current query.**  **")**  **S VALMCNT=LINE**  **Q**  **;**  **HELP ; -- help code**  **S X="?" D DISP^XQORM1 W !!**  **Q**  **;**  **EXIT ; -- exit code**  **K PSOSRCH,PSOSRT,@VALMAR,PSOPAR,PSOPINST**  **D FULL^VALM1**  **Q**  **;**  **EX ; early exit logic**  **K PSOSRCH,PSOSRT,PSOPAR,PSOPINST**  **D EX^PSOORFI1**  **Q**  **EXPND ; -- expand code**  **Q**  **;**  **; search list and display results**  **SEARCH ;**  **N RES,STYP,SVAL,I,DONE,SRCHARY**  **D FULL^VALM1**  **S DONE=0**  **F I=1:1 D Q:DONE**  **.S RES=$$DIR(,I,.SRCHARY)**  **.I '+RES S DONE=1 Q**  **.S SRCHARY(+RES)=$P(RES,U,2,99)**  **D EN(.SRCHARY)**  **;S VALMBCK="R"**  **Q**  **; sort target list**  **SORT ;**  **N RES,STYP,SVAL**  **D FULL^VALM1**  **S RES=$$DIR(1,0)**  **I '+$P(RES,U) Q**  **S STYP=$P(RES,U)**  **D EN(,STYP)**  **Q**  **DIR(SORT,CNT,SLIST) ;**  **N DIR,Y,RLINE,STAG,SVAL**  **S DIR(0)="SO^1:PATIENT NAME;2:DATE OF BIRTH;3:RECEIVED DATE RANGE;4:PRO**  **VIDER NAME;5:VA PHARAMCY LOCATION;6:DRUG NAME;7:CONTROLLED SUBSTANCES;8:NON-CONT**  **ROLLED SUBSTANCES"**  **I CNT<2 S DIR("L",1)="Select one of the following "\_$S($G(SORT):"sort",**  **1:"search")\_" criteria:"**  **I CNT>1 S DIR("L",1)="Select another search criteria or '^' to exit:"**  **S DIR("S")="I '$D(SLIST(Y))"**  **S DIR("L",2)=""**  **S DIR("L",3)=" "\_$S($D(SLIST(1)):"\*",1:"")\_"1.) PATIENT NAME"**  **S DIR("L",4)=" "\_$S($D(SLIST(2)):"\*",1:"")\_"2.) DATE OF BIRTH"**  **S DIR("L",5)=" "\_$S($D(SLIST(3)):"\*",1:"")\_"3.) RECEIVED DATE RANGE"**  **S DIR("L",6)=" "\_$S($D(SLIST(4)):"\*",1:"")\_"4.) PROVIDER NAME"**  **S DIR("L",7)=" "\_$S($D(SLIST(5)):"\*",1:"")\_"5.) ERX STATUS"**  **S DIR("L",8)=" "\_$S($D(SLIST(6)):"\*",1:"")\_"6.) DRUG NAME"**  **S DIR("L",9)=" "\_$S($D(SLIST(7)):"\*",1:"")\_"7.) CONTROLLED SUBSTANCES"**  **S DIR("L")=" "\_$S($D(SLIST(8)):"\*",1:"")\_"8.) NON-CONTROLLED SUBSTANCES**  **"**  **D ^DIR K DIR Q:'Y 0**  **S RES=Y I $G(SORT) Q RES**  **S RLINE=$S(RES=1:"PAT",RES=2:"DOB",RES=3:"RDT",RES=4:"PRVNM",RES=5:"EST**  **AT",RES=6:"DNAME",RES=7:"CS",RES=8:"NCS",1:"")**  **I RLINE']"" Q 0**  **S STAG=RLINE**  **S SVAL=$$@STAG I SVAL="" Q 0**  **Q RES\_U\_SVAL**  **PAT() ;**  **N Y,DIC**  **S DIC=52.46,DIC(0)="AEQ" D ^DIC**  **I Y<1 Q 0**  **Q Y**  **DOB() ;**  **N %DT,Y**  **S %DT="A"**  **S %DT("A")="Enter the Date of Birth (DOB): "**  **D ^%DT**  **I Y<0 Q 0**  **Q Y**  **RDT() ;**  **N BDATE,EDATE,%DT,Y**  **S %DT="A"**  **S %DT("A")="Enter the beginning date: "**  **D ^%DT**  **I Y<0 Q 0**  **S BDATE=Y K Y,%DT**  **S %DT="A"**  **S %DT("A")="Enter the ending date: "**  **S %DT("B")="T"**  **D ^%DT**  **I Y<0 Q 0**  **S EDATE=Y\_".999999"**  **Q BDATE\_U\_EDATE**  **PRVNM() ;**  **N Y,DIC**  **S DIC=52.48,DIC(0)="AEQ" D ^DIC**  **I Y<1 Q 0**  **Q Y**  **ESTAT() ;**  **; todo - prompt for erx status**  **N Y,DIC**  **S DIC=52.45,DIC(0)="AEQ",DIC("S")="I $D(^PS(52.45,""C"",""ERX"",X))" D**  **^DIC**  **I Y<1 Q 0**  **Q Y**  **DNAME() ;**  **N DIR,Y**  **S DIR(0)="FO"**  **S DIR("A")="Enter the name or partial name of the incoming eRx drug"**  **D ^DIR**  **I Y=""!(Y="^") Q ""**  **Q Y**  **; may not need the following tags. just use the value and a 1 or 0 for**  **on/off?**  **CS ;**  **Q**  **NCS ;**  **Q** |

Table 39: Routine: PSOERX1

| Routines | Activities | | | |
| --- | --- | --- | --- | --- |
| **Routine Name** | PSOERX1 | | | |
| **Enhancement Category** | New | Modify | Delete | No Change |
| **RTM** | Summary Screen of the eRx Holding Queue – Story #417226 | | | |
| **Related Options** |  | | | |

| Related Routines | Routines “Called By” | Routines “Called” |
| --- | --- | --- |
|  |  |  |

| Routines | Activities | | | | |
| --- | --- | --- | --- | --- | --- |
| **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) |
| --- |
| **PSOERX1 ;ALB/BWF - eRx Utilities/RPC's ; 8/3/2016 5:14pm**  **;;7.0;OUTPATIENT PHARMACY;\*\*467\*\*;DEC 1997;Build 42**  **;**  **EN(PSOIEN) ; -- main entry point for PSO ERX HOLDING QUEUE**  **D EN^VALM("PSO ERX HQ DISPLAY")**  **Q**  **;**  **HDR ; -- header code**  **;S VALMHDR(1)="eRx Processing"**  **S VALMHDR(1)="eRx Patient: "\_$$GET1^DIQ(52.49,PSOIEN,.04,"E")**  **S VALMHDR(2)="eRx Reference #: "\_$$GET1^DIQ(52.49,PSOIEN,25,"E")**  **; TODO?? - bwf - setting valmbck="R" and trying to use RE^VALM4 was not**  **refreshing the list area.. dev environment problem?**  **I VALMBCK="R" K @VALMAR S VALMBCK="" D INIT**  **Q**  **;**  **INIT ;**  **Q:'$G(PSOIEN)**  **N PATIEN,PHARMIEN,PRVIEN,PHDAT,PATDAT,SUPDAT,PRVDAT,LINE,PRVLNM,PRVMI,P**  **RVFN,EPAT,EPATDOB,VPATIEN**  **N VPATNM,VPATDOB,EPRVIEN,EPRVNM,EPRVNPI,VAPRVIEN,VAPRVNM,VAPRVNPI,SUPIE**  **N,ERXDRUG,ERXQTY,ERXFLS**  **N ERXDS,ERXDT,VADRG,LINETXT,ERXCOMM,ERXRFLS,PATIENS,VAHREA,VAQTY,VAREF,**  **VASIG,PATDAT,CURSTATE,CURSTATI**  **N LHFOUND,LHMATCH,LHSTATI,VAPDEAEX,ERXCOMM,COMFRST,COMARY,SIGDATA,SIGLO**  **OP,SFIRST,SGLOOP,SIGARY**  **S LINE=0**  **;L +^PS(52.49,PSOIEN):0**  **S PATIEN=$$GET1^DIQ(52.49,PSOIEN,.04,"I"),PATIENS=PATIEN\_","**  **D GETS^DIQ(52.46,PATIENS,"\*\*","IE","PATDAT")**  **S EPAT=$G(PATDAT(52.46,PATIENS,.01,"E"))**  **S EPATDOB=$G(PATDAT(52.46,PATIENS,.08,"I")),EPATDOB=$$FMTE^XLFDT(EPATDO**  **B,"2D")**  **; TODO - should we look at the patient linkage on the eRx or on 52.46?**  **for now use 52.49**  **;S VPATIEN=$G(PATDAT(52.46,PATIENS,1.5,"I"))**  **S VPATIEN=$$GET1^DIQ(52.49,PSOIEN,.05,"I")**  **S VPATNM=$S(VPATIEN:$$GET1^DIQ(2,VPATIEN,.01,"E"),1:"NOT LINKED")**  **S VPATDOB=$S(VPATIEN:$$GET1^DIQ(2,VPATIEN,.03,"I"),1:"N/A")**  **I VPATDOB S VPATDOB=$$FMTE^XLFDT(VPATDOB,"2D")**  **S PHARMIEN=$$GET1^DIQ(52.49,PSOIEN,2.5,"I")**  **D GETS^DIQ(52.47,PHARMIEN,"\*\*","E","PHDAT")**  **S EPRVIEN=$$GET1^DIQ(52.49,PSOIEN,2.1,"I")**  **S EPRVNM=$$GET1^DIQ(52.48,EPRVIEN,.01,"E")**  **S EPRVNPI=$$GET1^DIQ(52.48,EPRVIEN,1.5,"E")**  **S VAPRVIEN=$$GET1^DIQ(52.49,PSOIEN,2.3,"I")**  **S VAPRVNM=$S(VAPRVIEN:$$GET1^DIQ(200,VAPRVIEN,.01,"E"),1:"NOT LINKED")**  **S VAPRVNPI=$S(VAPRVIEN:$$GET1^DIQ(200,VAPRVIEN,41.99,"E"),1:"N/A")**  **D GETS^DIQ(52.48,EPRVIEN,"\*\*","E","PRVDAT")**  **S SUPIEN=$$GET1^DIQ(52.49,PSOIEN,2.6,"I")**  **D GETS^DIQ(52.48,SUPIEN,"\*\*","E","SUPDAT")**  **S LINETXT=""**  **S LINE=LINE+1**  **D ADDITEM^PSOERX1A(.LINETXT,"eRx Patient: ",$E(EPAT,1,55),1,55)**  **D ADDITEM^PSOERX1A(.LINETXT,"DOB: ",EPATDOB,57,20)**  **D SET^VALM10(LINE,LINETXT) S LINETXT=""**  **S LINETXT=""**  **S LINE=LINE+1**  **D ADDITEM^PSOERX1A(.LINETXT,"Vista Patient: ",$E(VPATNM,1,55),1,55)**  **D ADDITEM^PSOERX1A(.LINETXT,"DOB: ",VPATDOB,57,20)**  **D SET^VALM10(LINE,LINETXT) S LINETXT=""**  **S LINE=LINE+1 D SET^VALM10(LINE,"")**  **S LINE=LINE+1**  **D ADDITEM^PSOERX1A(.LINETXT,"eRx Provider: ",$E(EPRVNM,1,55),1,55)**  **D ADDITEM^PSOERX1A(.LINETXT,"NPI: ",EPRVNPI,57,20)**  **D SET^VALM10(LINE,LINETXT) S LINETXT=""**  **S LINE=LINE+1**  **D ADDITEM^PSOERX1A(.LINETXT,"Vista Provider: ",$E(VAPRVNM,1,55),1,55)**  **D ADDITEM^PSOERX1A(.LINETXT,"NPI: ",VAPRVNPI,57,20)**  **D SET^VALM10(LINE,LINETXT) S LINETXT=""**  **S LINE=LINE+1 D SET^VALM10(LINE,"")**  **S ERXDRUG=$$GET1^DIQ(52.49,PSOIEN,3.1,"E")**  **S ERXQTY=$$GET1^DIQ(52.49,PSOIEN,5.1,"E")**  **S ERXRFLS=$$GET1^DIQ(52.49,PSOIEN,5.6,"E")**  **S ERXDS=$$GET1^DIQ(52.49,PSOIEN,5.5,"E")**  **S ERXDT=$$GET1^DIQ(52.49,PSOIEN,.03,"E")**  **S VADRG=$$GET1^DIQ(52.49,PSOIEN,3.2,"E")**  **S VAREF=$$GET1^DIQ(52.49,PSOIEN,20.5,"E")**  **S VAQTY=$$GET1^DIQ(52.49,PSOIEN,20.1,"E")**  **; only set the hold reason if the eRx has a hold status**  **S CURSTATE=$$GET1^DIQ(52.49,PSOIEN,1,"E")**  **S VAHREA=""**  **I $E(CURSTATE,1)="H" D**  **.S CURSTATI=$$GET1^DIQ(52.49,PSOIEN,1,"I")**  **.S LHMATCH=999999,LHFOUND=0 F S LHMATCH=$O(^PS(52.49,PSOIEN,19,LHMATCH**  **),-1) Q:'LHMATCH!(LHFOUND) D**  **..S LHSTATI=$$GET1^DIQ(52.4919,LHMATCH\_","\_PSOIEN\_",",.02,"I") I LHSTAT**  **I=CURSTATI D S LHFOUND=LHMATCH Q**  **...S VAHREA=$$GET1^DIQ(52.4919,LHMATCH\_","\_PSOIEN\_",",1)**  **I VADRG']"" S VADRG="NOT LINKED"**  **S LINE=LINE+1 D SET^VALM10(LINE,"eRx Drug: "\_$E(ERXDRUG,1,65))**  **S LINE=LINE+1**  **D ADDITEM^PSOERX1A(.LINETXT,"Qty: ",ERXQTY,1,8)**  **D ADDITEM^PSOERX1A(.LINETXT,"eRx Refills: ",ERXRFLS,14,15)**  **D ADDITEM^PSOERX1A(.LINETXT,"Days Supply: ",ERXDS,32,15)**  **D ADDITEM^PSOERX1A(.LINETXT,"eRx Date: ",$P(ERXDT,"@"),52,25)**  **D SET^VALM10(LINE,LINETXT) S LINETXT=""**  **S SIGDATA=""**  **S SIGLOOP=0 F S SIGLOOP=$O(^PS(52.49,PSOIEN,11,SIGLOOP)) Q:'SIGLOOP D**  **.I '$L(SIGDATA) S SIGDATA=$$GET1^DIQ(52.4911,SIGLOOP\_","\_PSOIEN,1,"E")**  **Q**  **.S SIGDATA=SIGDATA\_" "\_SIGDATA**  **D TXT2ARY^PSOERXD1(.SIGARY,SIGDATA,,70)**  **S SFIRST=$O(SIGARY(0))**  **S SGLOOP=0 F S SGLOOP=$O(SIGARY(SGLOOP)) Q:'SGLOOP D**  **.S LINE=LINE+1 D SET^VALM10(LINE,$S(SGLOOP=SFIRST:"eRx Sig: ",1:"**  **")\_$G(SIGARY(SGLOOP)))**  **S LINE=LINE+1 D SET^VALM10(LINE,"")**  **S LINE=LINE+1 D SET^VALM10(LINE,"Vista Drug: "\_VADRG)**  **S LINE=LINE+1**  **D ADDITEM^PSOERX1A(.LINETXT,"Vista Qty: ",$G(VAQTY),1,14)**  **D ADDITEM^PSOERX1A(.LINETXT,"Vista Refills: ",$G(VAREF),25,17)**  **D ADDITEM^PSOERX1A(.LINETXT,"Vista Days Supply: ",$G(ERXDS),52,23)**  **D SET^VALM10(LINE,LINETXT) S LINETXT=""**  **S LINE=LINE+1 D SET^VALM10(LINE,"Vista Sig: "\_$G(VASIG))**  **S LINE=LINE+1 D SET^VALM10(LINE,"Hold Reason: "\_$G(VAHREA))**  **S LINE=LINE+1 D SET^VALM10(LINE,"")**  **S ERXCOMM=$$GET1^DIQ(52.49,PSOIEN,8,"E")**  **D TXT2ARY^PSOERXD1(.COMARY,ERXCOMM," ",68)**  **S COMFRST=$O(COMARY(0)) S LINE=LINE+1 D SET^VALM10(LINE,"Comments: "\_$G**  **(ERXCOMM))**  **F S COMFRST=$O(COMARY(COMFRST)) Q:'COMFRST D**  **.S LINE=LINE+1 D SET^VALM10(LINE," "\_$G(COMARY(COMFRST)))**  **S VALMCNT=LINE**  **Q**  **;**  **HELP ; -- help code**  **S X="?" D DISP^XQORM1 W !!**  **Q**  **;**  **EXIT ; -- exit code**  **K @VALMAR**  **;D INIT^PSOERX**  **Q**  **;**  **EXPND ; -- expand code**  **Q** |

Table 40: Routine: PSOERX1A

| Routines | Activities | | | |
| --- | --- | --- | --- | --- |
| **Routine Name** | PSOERX1A | | | |
| **Enhancement Category** | New | Modify | Delete | No Change |
| **RTM** | Validate Patient – Story #417240  Validate Provider – Story #417243  Validate Drug – Story #417244 | | | |
| **Related Options** |  | | | |

| Related Routines | Routines “Called By” | Routines “Called” |
| --- | --- | --- |
|  |  |  |

| Routines | Activities | | | | |
| --- | --- | --- | --- | --- | --- |
| **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) |
| --- |
| **PSOERX1A ;ALB/BWF - eRx Utilities/RPC's ; 8/3/2016 5:14pm**  **;;7.0;OUTPATIENT PHARMACY;\*\*467\*\*;DEC 1997;Build 42**  **;**  **Q**  **; select an item**  **SI ;**  **N RESP,ERXIEN,ERXDAT,LINE,LINEVAR**  **S DIR(0)="N^"\_VALMBG\_":"\_VALMLST\_":0" D ^DIR**  **I 'Y S VALMBCK="R" Q**  **S RESP=Y**  **S ERXIEN=$O(@VALMAR@("IDX",RESP,"")) Q:'ERXIEN**  **L +^PS(52.49,ERXIEN):0 I '$T W !,"Another person is editing this order.**  **" S VALMBCK="R" S DIR(0)="E" D ^DIR K DIR Q**  **D EN^PSOERX1(ERXIEN)**  **L -^PS(52.49,ERXIEN)**  **S VALMBCK="R"**  **Q**  **SBN ;**  **N Y,ERXIEN**  **S Y=+$P(XQORNOD(0),"=",2)**  **I 'Y S VALMBCK="R" Q**  **S ERXIEN=$O(@VALMAR@("IDX",Y,"")) Q:'ERXIEN**  **L +^PS(52.49,ERXIEN):0 I '$T W !,"Another person is editing this order.**  **" S VALMBCK="R" S DIR(0)="E" D ^DIR K DIR Q**  **D EN^PSOERX1(ERXIEN)**  **L -^PS(52.49,ERXIEN)**  **S VALMBCK="R"**  **;D RE^VALM4**  **Q**  **;**  **; TEXT - variable where text is stored (passed by reference)**  **; HDR - header text**  **; DATA - data associated with the header**  **; STRT - start location (column)**  **; LEN - total length for header and data**  **ADDITEM(TEXT,HDR,DATA,STRT,LEN) ;**  **N LLEN,FULLDAT,L**  **S FULLDAT=$G(HDR)\_$G(DATA)**  **S TEXT=$G(TEXT,"") I STRT=1 S TEXT=TEXT\_$E(FULLDAT,1,LEN) Q**  **S LLEN=$L(TEXT)**  **I LLEN<STRT D**  **.F L=$L(TEXT):1:STRT-1 D**  **..S TEXT=TEXT\_" "**  **S TEXT=TEXT\_$E(FULLDAT,1,LEN)**  **Q**  **; provider information display**  **PROV ;**  **D EN^PSOERXR1**  **Q**  **; patient information display**  **PAT ;**  **D EN^PSOERXP1**  **Q**  **; drug information display**  **DRUG ;**  **D EN^PSOERXD1**  **Q**  **; edit validation**  **; EDTYPE - D=drug, P=patient, PR=provider**  **EDIT(EDTYP) ;**  **N DIR,Y,ITEM,RES,TAG,PQUIT**  **D FULL^VALM1**  **S VALMBCK="R"**  **Q:'$G(PSOIEN)**  **S PSOIENS=PSOIEN\_","**  **Q:'$D(EDTYP)**  **I EDTYP="D" D Q**  **.D PLSTRNG(1,12,.RES)**  **.I $G(RES)=0 Q**  **.I '$$GET1^DIQ(52.49,PSOIEN,3.2,"I"),'$D(RES(1)) D Q**  **..W !,"The VA drug must be matched before editing any further data elem**  **ents."**  **..S DIR(0)="E" D ^DIR K DIR**  **.S (ITEM,PQUIT)=0 F S ITEM=$O(RES(ITEM)) Q:'ITEM!(PQUIT) D**  **..S TAG="VDRG"\_ITEM\_"(PSOIEN,PSOIENS)" D @TAG**  **.K @VALMAR D INIT^PSOERXD1**  **I EDTYP="P" D VPAT K @VALMAR D INIT^PSOERXP1 Q**  **I EDTYP="PR" D VPROV K @VALMAR D INIT^PSOERXR1 Q**  **Q**  **; validate provider**  **VPROV ;**  **N EXPRVIEN,VAPRVIEN,MANVAL,PRVDAT,EXPRVDEA,EXPRVNPI,EXPRNAME,EXPRLNAM,E**  **XPRFNAM,PSOIENS**  **N EXPRIENS,SELPRV,QUIT**  **S PSOIENS=PSOIEN\_","**  **S EXPRVIEN=$$GET1^DIQ(52.49,PSOIEN,2.1,"I")**  **S EXPRIENS=EXPRVIEN\_","**  **D GETS^DIQ(52.48,EXPRIENS,".01;.02;.03;1.5;1.6","E","PRVDAT")**  **S EXPRVDEA=$G(PRVDAT(52.48,EXPRIENS,1.6,"E"))**  **S EXPRVNPI=$G(PRVDAT(52.48,EXPRIENS,1.5,"E"))**  **S EXPRNAME=$G(PRVDAT(52.48,EXPRIENS,.01,"E"))**  **S EXPRLNAM=$G(PRVDAT(52.48,EXPRIENS,.02,"E"))**  **S EXPRFNAM=$G(PRVDAT(52.48,EXPRIENS,.03,"E"))**  **S MANVAL=$$GET1^DIQ(52.49,PSOIEN,1.3,"I")**  **S VAPIEN=$$GET1^DIQ(52.49,PSOIEN,2.3,"I")**  **I VAPIEN D Q**  **.W !,"Current Vista provider: "\_EXPRNAME,!**  **.S DIR(0)="YO",DIR("A")="Would you like to modify the validated provide**  **r"**  **.S DIR("A",1)="This provider has already been validated."**  **.S DIR("B")="NO" D ^DIR**  **.Q:'Y**  **.;S DIC=200,DIC(0)="AEMQ",DIC("S")="I $$CHKPRV^PSOERX1A("\_""""\_EXPRLNAM**  **\_""""\_","\_""""\_EXPRVNPI\_""""\_","\_""""\_EXPRVDEA\_""""\_")" D ^DIC**  **.;K DIC**  **.;Q:Y<1**  **.S DIC=200,DIC(0)="AEMQ",DIC("S")="I $$CHKPRV2^PSOERX1A(Y)" D ^DIC**  **.Q:Y<1**  **.S NEWPIEN=$P(Y,U) K Y**  **.S DIR(0)="Y",DIR("A")="Would you like to use this provider."**  **.S DIR("A",1)="You have selected provider: "\_$$GET1^DIQ(200,NEWPIEN,.01**  **,"E")**  **.S DIR("B")="YES" D ^DIR**  **.I Y<1 S QUIT=1 Q**  **.; change existing entry**  **.S FDA(52.49,PSOIENS,2.3)=NEWPIEN**  **.S FDA(52.49,PSOIENS,1.3)=1**  **.D FILE^DIE(,"FDA") K FDA**  **.; user chose to delete the mapping**  **.;Q:QUIT**  **.;S FDA(52.49,PSOIENS,2.1)="@"**  **.;S FDA(52.49,PSOIENS,1.3)=0**  **.;D FILE^DIE(,"FDA") K FDA**  **;S DIC=200,DIC(0)="AEMQ",DIC("S")="I $$CHKPRV^PSOERX1A("\_""""\_EXPRLNAM\_**  **""""\_","\_""""\_EXPRVNPI\_""""\_","\_""""\_EXPRVDEA\_""""\_")" D ^DIC**  **; for now, only list providers that are authorized to write med orders**  **and whose dea is not expired**  **S DIC=200,DIC(0)="AEMQ",DIC("S")="I $$CHKPRV2^PSOERX1A(Y)" D ^DIC**  **Q:Y<1**  **S SELPRV=Y**  **S DIR(0)="Y",DIR("A")="Would you like to use this provider."**  **S DIR("A",1)="You have selected provider: "\_$$GET1^DIQ(200,Y,.01,"E")**  **S DIR("B")="YES" D ^DIR**  **Q:Y<1**  **S FDA(52.49,PSOIENS,2.3)=$P(SELPRV,U)**  **S FDA(52.49,PSOIENS,1.3)=1**  **D FILE^DIE(,"FDA","ERR") K FDA**  **Q**  **CHKPRV2(Y) ;**  **;S DEAEXP=$P($G(^VA(200,Y,"QAR")),U,9)**  **I '$P($G(^VA(200,Y,"PS")),U) Q 0**  **;I DEAEXP]"",DEAEXP<DT Q 0**  **Q 1**  **; validate drug**  **; prompt list or range**  **; LOW - lowest number to prompt for**  **; HIGH - highest number to prompt for**  **; EDIT - returned list of selected values**  **; EDIT(n1)=""**  **; EDIT(n2)=""**  **; EDIT(n3)=""**  **PLSTRNG(LOW,HIGH,EDIT) ;**  **N DIR,DONE,DONE2,Y,NUMCHK,NUM,VAL,I,LIST**  **I '$G(LOW)!'$G(HIGH) S LIST=0 Q**  **S DONE=0**  **F D Q:DONE**  **.S DIR(0)="FO^",DIR("A")="Which field(s) would you like to edit? ("\_LOW**  **\_"-"\_HIGH\_") or 'A'll"**  **.S DIR("?")="Enter a number, range, or a list of numbers (i.e. 3, 1-5,**  **3,7,9, or 'A'll)"**  **.S DIR("B")="A"**  **.D ^DIR**  **.I Y="A" D Q**  **..F I=LOW:1:HIGH D**  **...S EDIT(I)=""**  **..S DONE=1**  **.; check for a range or list of numbers**  **.I Y'["-",Y'[",",Y'<LOW,Y'>HIGH S EDIT(Y)="" S DONE=1 Q**  **.I Y?1.2N1"-"1.2N D**  **..F I=$P(Y,"-"):1:$P(Y,"-",2) D**  **...Q:I<LOW!(I>HIGH)**  **...S EDIT(I)=""**  **.I $D(EDIT) S DONE=1 Q**  **.I Y["," D**  **..; check to see if there are alpha-numerics if there are, quit and rep**  **rompt**  **..S NUMCHK=$TR(Y,",","") I 'NUMCHK W !,"Invalid response." Q**  **..S DONE2=0**  **..F NUM=1:1 D Q:DONE2**  **...S VAL=$P(Y,",",NUM)**  **...I VAL<LOW!(VAL>HIGH) Q**  **...I 'VAL S DONE2=1 Q**  **...S EDIT(VAL)=""**  **Q**  **VDRG1(PSOIEN,PSOIENS) ;**  **N VANDRG,VAODRG,DIE,DA,DR,AUTOVAL,DIC,Y**  **Q:'$G(PSOIEN)**  **S VAODRG=$$GET1^DIQ(52.49,PSOIEN,3.2,"I")**  **S AUTOVAL=$$GET1^DIQ(52.49,PSOIEN,1.4,"I")**  **I VAODRG D**  **.S DIR(0)="S^E:EDIT;D:DELETE",DIR("A")="Would you like to 'E'dit or 'D'**  **elete this drug?" D ^DIR**  **Q:$G(Y)="^"**  **I $G(Y)="D" D Q**  **.S DIE="^PS(52.49,",DA=PSOIEN,DR="3.2///@;1.5///@;1.11///@;1.12///@" D**  **^DIE K DIE**  **.S QTLOOP=0 F S QTLOOP=$O(^PS(52.49,PSOIEN,21,QTLOOP)) Q:'QTLOOP D**  **..S FDA(52.4921,QTLOOP\_","\_PSOIENS,.01)="@" D FILE^DIE(,"FDA") K FDA**  **.K DIR S DIR(0)="E" W !,"Vista drug deleted." D ^DIR**  **; for now allow user to search by drug name. may enhance screening in t**  **he future.**  **S DIC(0)="AEMQ",DIC=50,DIC("S")="I $$ACTIVE^PSOERXA0(Y),($$OUTPAT^PSOER**  **XA0(Y)),('$$INVCOMP^PSOERXA0(Y)),('$$CS^PSOERXA0(Y))" D ^DIC**  **K DIC**  **Q:$P(Y,U)<1**  **S DIE="^PS(52.49,",DA=PSOIEN,DR="3.2///"\_$P(Y,U,1) D ^DIE**  **; set the manual validation flag if the drug has been selected.**  **S VANDRG=$$GET1^DIQ(52.49,PSOIEN,3.2,"I")**  **S FDA(52.49,PSOIENS,1.5)=$S(VANDRG:1,1:"")**  **; if the drug has changed, update the manual validation by and date/tim**  **e**  **I VANDRG'="",VAODRG'=VANDRG D**  **.S FDA(52.49,PSOIENS,1.11)=$G(DUZ)**  **.S FDA(52.49,PSOIENS,1.12)=$$NOW^XLFDT**  **.S QTLOOP=0 F S QTLOOP=$O(^PS(52.49,PSOIEN,21,QTLOOP)) Q:'QTLOOP D**  **..S FDA(52.4921,QTLOOP\_","\_PSOIENS,.01)="@" D FILE^DIE(,"FDA") K FDA**  **.;TODO - if the autoval parameter is set to true, and a user changed th**  **e drug, should we remove the autoval parameter??**  **.;I AUTOVAL S FDA(52.49,PSOIENS,1.4)="@"**  **D FILE^DIE(,"FDA") K FDA**  **Q**  **VDRG2(PSOIEN,PSOIENS) ;**  **N PSOQTS**  **N PSODOSE,PSONEW,PSODRUG,PSORXED,VERB,QTIEN,SFIENS,DOSE,PSODFN**  **; get patient IEN**  **; TODO - even though the dfn is the same in file 2 as in file 55, shoul**  **d this be collected from file 55 instead?**  **S PSODFN=$$GET1^DIQ(52.49,PSOIEN,.05,"I")**  **I 'PSODFN D Q**  **.W !,"Cannot continue with dosing instructions until patient has been m**  **atched."**  **.S DIR(0)="E" D ^DIR Q**  **; get drug ien and orderable item ien**  **S PSODRUG("IEN")=$$GET1^DIQ(52.49,PSOIEN,3.2,"I") Q:'PSODRUG("IEN")**  **S PSODRUG("OI")=$$GET1^DIQ(50,PSODRUG("IEN"),2.1,"I") Q:'PSODRUG("OI")**  **S (PSORXED("ENT"),ENT)=1**  **D DOSE^PSSORPH(.DOSE,PSODRUG("IEN"),"O",PSODFN)**  **D ASK^PSOBKDED**  **I $G(VERB)]"" S PSORXED("VERB",ENT)=VERB G DUPD**  **VER ;**  **D VER^PSOOREDX I X[U,$L(X)>1 S FIELD="VER" G JUMP**  **G:$D(DTOUT)!($D(DUOUT)) EXQ**  **I X="@" K PSORXED("VERB",ENT),VERB G DUPD**  **S:X'="" (PSORXED("VERB",ENT),VERB)=X**  **DUPD ;**  **I $G(PSORXED("DOSE",ENT))'?.N&($G(PSORXED("DOSE",ENT))'?.N1".".N)!'DOSE**  **("LD") K PSORXED("DOSE ORDERED",ENT),DUPD G NOU1**  **D DUPD^PSOOREDX**  **S DIR("B")=$S($G(PSORXED("DOSE ORDERED",ENT))]"":PSORXED("DOSE ORDERED"**  **,ENT),1:1) S:$E($G(DIR("B")),1)="." DIR("B")="0"\_$G(DIR("B")) K:DIR("B")="" DIR(**  **"B")**  **D ^DIR I X[U,$L(X)>1 S FIELD="DUPD" G JUMP**  **G:$D(DTOUT)!($D(DUOUT)) EXQ**  **I X="@"!(X=0) W !,"Dispense Units Per Dose is Required!!",! G DUPD**  **D STR^PSOOREDX**  **NOU1 ;**  **G:'$G(PSORXED("DOSE ORDERED",ENT)) RTE**  **D CNON^PSOORED3**  **N PSONDEF**  **I $G(NOUN)]"" S PSORXED("NOUN",ENT)=NOUN**  **NOU ;**  **D NOU^PSOOREDX I X[U,$L(X)>1 S FIELD="NOU" G JUMP**  **G:$D(DTOUT)!($D(DUOUT)) EXQ**  **I X="@" K PSORXED("NOUN",ENT),NOUN G RTE**  **I X'="",$G(PSONDEF)="" S NOUN=X**  **I X'="",$G(PSONDEF)'=X S NOUN=X**  **S:X'="" PSORXED("NOUN",ENT)=X**  **;**  **RTE ;**  **K JUMP S ROU="PSOORED4" D RTE^PSOBKDED K ROU**  **I $G(JUMP) K JUMP G JUMP**  **G:$D(DTOUT)!($D(DUOUT)) EXQ**  **;**  **SCH ;**  **D SCH^PSOBKDED I X[U,$L(X)>1 S FIELD="SCH" G JUMP**  **G:$D(DTOUT)!($D(DUOUT)) EXQ**  **S SCH=Y D SCH^PSOSIG I $G(SCH)']"" G SCH**  **S PSORXED("SCHEDULE",ENT)=SCH W " ("\_SCHEX\_")" K SCH,SCHEX,X,Y,PSOSCH**  **S:PSORXED("ENT")<ENT PSORXED("ENT")=ENT**  **;**  **DUR ;**  **N SIG,SIGDAT,SLOOP,P01**  **D KV K EXP S DIR(0)="52.0113,4",DIR("A")="LIMITED DURATION (IN DAYS, HO**  **URS OR MINUTES)"**  **S DIR("B")=$S($G(PSORXED("DURATION",ENT))]"":PSORXED("DURATION",ENT),1:**  **"") K:DIR("B")="" DIR("B")**  **D ^DIR I X[U,$L(X)>1 S FIELD="DUR" G JUMP**  **G:$D(DTOUT)!($D(DUOUT)) EXQ**  **D DUR1^PSOOREDX**  **;**  **; REMOVE CONJUNCTION FOR NOW, SINCE WE WILL NOT HAVE COMPLEX DOSING**  **;CON D CON^PSOOREDX I X[U,$L(X)>1 S FIELD="CON" G JUMP**  **;G:$D(DTOUT)!($D(DUOUT)) EXQ**  **;I X="@",$G(PSORXED("CONJUNCTION",ENT))="" W !,?10,"Invalid Entry - not**  **hing to delete!!" G CON**  **;S:X'=""&(X'="@") PSORXED("CONJUNCTION",ENT)=Y**  **;I X="@",$D(PSORXED("CONJUNCTION",ENT)) D CON1^PSOOREDX G:$D(DIRUT) EXQ**  **G:'Y CON N CKX S CKX=1 D UPD^PSOOREDX G CON**  **;;**  **;N PSODLBD4 S PSOSAVX=X,PSODLBD4=1**  **;;\*437**  **;I '$$DUROK^PSOORED3(.PSORXED,ENT) D G DUR**  **;. W !!,"Duration is required for the dosage entered prior to the THEN**  **conjunction.",$C(7),!**  **;I $G(PSORXED("CONJUNCTION",ENT))]"" S PSOCKCON=1 D DCHK1^PSODOSUT G:$G**  **(PSONEW("DFLG")) EX S ENT=ENT+1 K DIR G ASK**  **;E K PSOCKCON D DCHK1^PSODOSUT I $D(DTOUT)!($D(DUOUT)) S PSORX("DFLG")**  **=1,PSONEW("DFLG")=1 G EX ;don't need to print the full summary, just the last s**  **equence.**  **;I PSOSAVX="",$G(PSORXED) K PSOCKCON,PSOEDDOS**  **;K PSOSAVX**  **;;**  **;S X=$G(PSORXED("INS")) D SIG^PSOHELP S:$G(INS1)]"" PSORXED("SIG")=$E(I**  **NS1,2,9999999)**  **;D EN^PSOFSIG(.PSORXED),VERI I $G(CKX),'$G(PSOSIGFL) D MP1 K CKX**  **;I $G(PSOSIGFL)=1 D I '$G(PSOSIGFL) Q**  **;.I $D(OR0),$P(OR0,"^",24)=1 S VALMSG="Digitally Signed Order - No such**  **changes allowed." K PSORXED,PSOSIGFL M PSORXED=PSODOSE D EN^PSOFSIG(.PSORXED) D**  **Q**  **;..I $D(PSOBDR) K PSODRUG M PSODRUG=PSOBDR K PSOBDR,PSOBDRG**  **;.S PSORXED("ENT")=ENT,SIGOK=1,VALMSG="This change will create a new pr**  **escription!",NCPDPFLG=1**  **;K QTYHLD S:$G(PSORXED("QTY")) QTYHLD=PSORXED("QTY") D QTY^PSOSIG(.PSOR**  **XED) I $G(PSORXED("QTY")) S QTY=1**  **;I $G(QTYHLD),'$G(PSORXED("QTY")) S PSORXED("QTY")=QTYHLD**  **;K QTYHLD**  **;I '$G(PSORXED("QTY")),$P(OR0,"^",10) S PSORXED("QTY")=$P(OR0,"^",10)**  **S PSORXED("INS")="TESTING TO SEE HOW LONG WE CAN MAKE THE SIG AND GET T**  **HIS TO FORMAT CORRECTLY"**  **S X=$G(PSORXED("INS")) D SIG^PSOHELP S:$G(INS1)]"" PSORXED("SIG")=$E(IN**  **S1,2,9999999)**  **D EN^PSOFSIG(.PSORXED)**  **; delete existing SIG**  **S SLOOP=0 F S SLOOP=$O(^PS(52.49,PSOIEN,"SIG",SLOOP)) Q:'SLOOP D**  **.S FDA(52.4926,SLOOP\_","\_PSOIEN\_",",.01)="@"**  **I $D(FDA) D FILE^DIE(,"FDA") K FDA**  **; file sig into appropriate VA DISPENSING INSTRUCTIONS multiple**  **S SLOOP=0 F S SLOOP=$O(SIG(SLOOP)) Q:'SLOOP D**  **.S SIGDAT=$G(SIG(SLOOP)) Q:SIGDAT=""**  **.S SFDA(52.4926,"+1,"\_PSOIENS,.01)=SIGDAT D UPDATE^DIE(,"SFDA",,"SERR")**  **K SFDA**  **S QTIEN=$O(^PS(52.49,PSOIEN,21,0))**  **S SFIENS=$S(QTIEN:QTIEN\_",",1:"+1,")**  **S P01=$S($L($G(PSORXED("DOSE ORDERED",ENT))):$G(PSORXED("DOSE ORDERED",**  **ENT))\_"&"\_$G(PSORXED("NOUN",ENT)),1:$G(PSORXED("DOSE",ENT)))**  **I '$L(P01) D Q**  **.W !,"Dosage is required. Please re-enter the dosing instructions." S D**  **IR(0)="W" D ^DIR D EX Q**  **S FDA(52.4921,SFIENS\_PSOIENS,.01)=$G(PSORXED("DOSE ORDERED",ENT))\_"&"\_$**  **G(PSORXED("NOUN",ENT))**  **S FDA(52.4921,SFIENS\_PSOIENS,1)=$G(PSORXED("SCHEDULE",ENT))**  **S FDA(52.4921,SFIENS\_PSOIENS,2)=$G(PSORXED("DURATION",ENT))**  **S FDA(52.4921,SFIENS\_PSOIENS,8)=$G(PSORXED("DOSE",ENT))**  **S FDA(52.4921,SFIENS\_PSOIENS,9)=$G(PSORXED("DOSE ORDERED",ENT))**  **S FDA(52.4921,SFIENS\_PSOIENS,10)=$G(PSORXED("ROUTE",ENT))**  **S FDA(52.4921,SFIENS\_PSOIENS,11)=$G(PSORXED("UNITS",ENT))**  **S FDA(52.4921,SFIENS\_PSOIENS,12)=$G(PSORXED("NOUN",ENT))**  **S FDA(52.4921,SFIENS\_PSOIENS,13)=$G(PSORXED("VERB",ENT))**  **I SFIENS["+" D UPDATE^DIE(,"FDA",,"ERR") K FDA**  **; if there is a quantity/timing entry, overwrite the data (we are not u**  **sing complex dosing in this phase)**  **I SFIENS'["+" D FILE^DIE(,"FDA") K FDA**  **; todo - how do we update the vista sig???**  **EX ;**  **K PSOBDR,PSOBDRG,PSOSCH,DUPD,STRE,UNITN,SCH,VERB,NOUN,DOSEOR,RTE,DUR,X,**  **Y,ENTS,PSODOSE,OLENT,FIELD,FLDNM,AR,NM,ENT,STRE,UNITN,PSODOSE,ERTE,ROU,AR1,INS1**  **KV K DTOUT,DUOUT,DIR,DIRUT**  **Q**  **EXQ ;**  **K PSORXED,PSOSIGFL M PSORXED=PSODOSE D EN^PSOFSIG(.PSORXED) D MP1**  **I $D(PSOBDR) M PSODRUG=PSOBDR K PSOBDR,PSOBDRG**  **G EX Q**  **MP1 ;**  **S VALMSG="eRx Not Updated!"**  **Q**  **JUMP ;jump to fields**  **I $L($E(X,2,99))<3 W !,"Field Name Must Be At Least 3 Characters in Len**  **gth",! G @FIELD**  **D FNM^PSOOREDX**  **I FLDNM']"" K X,NM,FLDNM W !,"INVALID FIELD NAME. PLEASE TRY AGAIN!",!**  **G @FIELD**  **F AR=1:1:PSORXED("ENT") W !,AR\_". "\_$P(FLDNM,"^",2)\_": "\_$S(NM="ROU"&($**  **G(PSORXED($P(FLDNM,"^"),AR))):$P(^PS(51.2,PSORXED($P(FLDNM,"^"),AR),0),"^"),1:$G**  **(PSORXED($P(FLDNM,"^"),AR))) S AR1=AR**  **D KV S DIR("A",1)="\* Indicates which fields will create a New Order",DI**  **R("A")="Select Field to Edit by number",DIR(0)="NO^1:"\_AR1 D ^DIR G:$D(DIRUT) @F**  **IELD**  **D JFN^PSOOREDX G:FLDNM="" @FIELD G @FLDNM**  **G EX**  **Q**  **VDRG3(PSOIEN,PSOIENS) ;**  **S DIE="^PS(52.49,",DA=PSOIEN,DR="27" D ^DIE K DIE,DR,DA**  **Q**  **VDRG4(PSOIEN,PSOIENS) ;**  **Q**  **VDRG5(PSOIEN,PSOIENS) ;**  **N PATIEN,PS55IEN**  **S PATIEN=$$GET1^DIQ(52.49,PSOIEN,.05,"I")**  **I 'PATIEN W !!,"Patient has not been validated, cannot edit patient sta**  **tus",! Q**  **S PS55IEN=$O(^PS(55,"B",PATIEN,0)) Q:'PS55IEN**  **S DIE="^PS(55,",DA=PS55IEN,DR="1" D ^DIE K DIE,DR,DA**  **Q**  **VDRG6(PSOIEN,PSOIENS) ;**  **S DIE="^PS(52.49,",DA=PSOIEN,DR="20.2" D ^DIE K DIE,DR,DA**  **Q**  **VDRG7(PSOIEN,PSOIENS) ;**  **S DIE="^PS(52.49,",DA=PSOIEN,DR="20.1" D ^DIE K DIE,DR,DA**  **Q**  **VDRG8(PSOIEN,PSOIENS) ;**  **S DIE="^PS(52.49,",DA=PSOIEN,DR="20.5" D ^DIE K DIE,DR,DA**  **Q**  **VDRG9(PSOIEN,PSOIENS) ;**  **S DIE="^PS(52.49,",DA=PSOIEN,DR="20.4" D ^DIE K DIE,DR,DA**  **Q**  **VDRG10(PSOIEN,PSOIENS) ;**  **N DIC,Y**  **S DIC="^SC(",DIC(0)="QEAMZ",DIC("A")="Select CLINIC: " D ^DIC K DIC I Y**  **<1!($D(DTOUT))!($D(DUOUT)) Q**  **; todo - add patient location or vista clinic field to 52.41. This maps**  **to patient location in 52.41 (field #1.1)**  **; ADD FIELD NUMBER INTO DR TO COMPLETE**  **S VACLIN=$P(Y,U)**  **S DIE="^PS(52.49,",DA=PSOIEN,DR="20.6///"\_VACLIN D ^DIE K DIE,DR,DA**  **K DTOUT,DUOUT**  **Q**  **VDRG11(PSOIEN,PSOIENS) ;**  **D VPROV**  **Q**  **VDRG12(PSOIEN,PSOIENS) ;**  **S DIE="^PS(52.49,",DA=PSOIEN,DR="8" D ^DIE K DIE,DR,DA**  **Q**  **; validate patient**  **VPAT ;**  **N ERXPIEN,VAPIEN,MANVAL,ERXSSN,ERXLNAME,ERXFNAME,DIR,Y,PSOIENS,ERXDOB,V**  **APIEN,MANVAL,NEWPIEN,DIR,DIC,SELPAT,PDONE**  **S PSOIENS=PSOIEN\_","**  **S ERXPIEN=$$GET1^DIQ(52.49,PSOIEN,.04,"I")**  **S ERXSSN=$$GET1^DIQ(52.46,ERXPIEN,1.4,"E"),ERXSSN=$TR(ERXSSN,"-","")**  **S ERXLNAME=$$GET1^DIQ(52.46,ERXPIEN,.02,"E")**  **S ERXFNAME=$$GET1^DIQ(52.46,ERXPIEN,.03,"E")**  **S ERXDOB=$$GET1^DIQ(52.46,ERXPIEN,.08,"I")**  **S ERXGEN=$$GET1^DIQ(52.46,ERXPIEN,.07,"I")**  **S VAPIEN=$$GET1^DIQ(52.49,PSOIEN,.05,"I")**  **S MANVAL=$$GET1^DIQ(52.49,PSOIEN,1.7,"I")**  **; if there is a patient currently defined**  **I VAPIEN D Q**  **.W !,"Current Vista patient: "\_$$GET1^DIQ(2,VAPIEN,.01,"E"),!**  **.S DIR(0)="YO",DIR("A")="Would you like to edit the validated patient"**  **.S DIR("A",1)="This patient has already been validated."**  **.S DIR("B")="NO" D ^DIR**  **.Q:'Y**  **.S PDONE=0**  **.F D Q:PDONE**  **..S SELPAT=""**  **..K DIC S DIC=2,DIC(0)="AEQ" D ^DIC**  **..I Y<1 S PDONE=1 Q**  **..S SELPAT=$P(Y,U) K VADM S DFN=SELPAT D DEM^VADPT**  **..I $P($G(VADM(6)),U)]"" W "[PATIENT DIED ON "\_$P($G(VADM(6)),U,2)\_"]"**  **Q**  **..I ERXSSN,'$D(^DPT(2,"SSN",ERXSSN)) W !,"SSN mismatch."**  **..I ERXDOB,'$D(^DPT(2,"ADOB",ERXDOB)) W !,"Date of Birth mismatch."**  **..I ERXGEN]"",$P($G(VADM(5)),U)'=ERXGEN W !,"Gender mismatch."**  **..W !,"Are you sure you wish to use this patient?"**  **..K DIR,Y S DIR(0)="Y",DIR("B")="NO" D ^DIR**  **..I Y=1 S PDONE=1 Q**  **.Q:+Y<1!('SELPAT)**  **.S NEWPIEN=SELPAT K Y**  **.S DIR(0)="Y",DIR("A")="Would you like to use this patient."**  **.S DIR("A",1)="You have selected patient: "\_$$GET1^DIQ(2,NEWPIEN,.01,"E**  **")**  **.S DIR("B")="YES" D ^DIR**  **.Q:Y<1**  **.; change existing entry**  **.S FDA(52.49,PSOIENS,.05)=NEWPIEN**  **.S FDA(52.49,PSOIENS,1.7)=1**  **.D FILE^DIE(,"FDA") K FDA**  **.; user chose to delete the mapping**  **.;S FDA(52.49,PSOIENS,.05)="@"**  **.;S FDA(52.49,PSOIENS,1.7)=0**  **.;D FILE^DIE(,"FDA") K FDA**  **.;D DISPEXP**  **.;S DIE="^PS(52.49,",DA=VAPIEN,DR=".05"**  **.;D ^DIE**  **.; current mapped patient**  **; if no vista patient has been matched.**  **S PDONE=0**  **F D Q:PDONE**  **.S SELPAT=""**  **.K DIC S DIC=2,DIC(0)="AEQ" D ^DIC**  **.I Y<1 S PDONE=1 Q**  **.S SELPAT=$P(Y,U) K VADM S DFN=SELPAT D DEM^VADPT**  **.I $P($G(VADM(6)),U)]"" W "[PATIENT DIED ON "\_$P($G(VADM(6)),U,2)\_"]" Q**  **.I ERXSSN,'$D(^DPT(2,"SSN",ERXSSN)) W !,"SSN mismatch."**  **.I ERXDOB,'$D(^DPT(2,"ADOB",ERXDOB)) W !,"Date of Birth mismatch."**  **.I ERXGEN]"",$P($G(VADM(5)),U)'=ERXGEN W !,"Gender mismatch."**  **.W !,"Are you sure you wish to use this patient?"**  **.K DIR,Y S DIR(0)="Y",DIR("B")="NO" D ^DIR**  **.I Y=1 S PDONE=1 Q**  **;D LIST^DIC(2,,".01","","","","","","I $$CHKPAT^PSOERX1A("\_""""\_ERXLNAM**  **E\_""""\_","\_""""\_ERXSSN\_""""\_","\_""""\_ERXDOB\_""""\_")","","BWF")**  **;S DIC=2,DIC(0)="AEMQ",DIC("S")="I $$CHKPAT^PSOERX1A("\_""""\_ERXLNAME\_""**  **""\_","\_""""\_ERXSSN\_""""\_","\_""""\_ERXDOB\_""""\_")" D ^DIC**  **;I +Y<1 D**  **;.K Y,DIR S DIR(0)="YO",DIR("A")="Would you like to search patients by**  **name?",DIR("B")="YES" D ^DIR**  **;.Q:'Y**  **;.K DIC S DIC=2,DIC(0)="AEMQ",DIC("S")="I '$P($G(^DPT(Y,.35)),U)" D ^DI**  **C**  **;Q:+Y<1**  **S NEWPIEN=SELPAT**  **S DIR(0)="Y",DIR("A")="Would you like to use this patient."**  **S DIR("A",1)="You have selected patient: "\_$$GET1^DIQ(2,NEWPIEN,.01,"E"**  **)**  **S DIR("B")="YES" D ^DIR**  **Q:Y<1**  **S FDA(52.49,PSOIENS,.05)=NEWPIEN**  **S FDA(52.49,PSOIENS,1.7)=1**  **D FILE^DIE(,"FDA") K FDA**  **Q**  **CHKPAT(NAME,SSN,DOB,GEN) ;**  **; for testing just quit 1**  **N SMATCH,NMATCH,DMATCH,CSSN**  **S CSSN=$TR(SSN,"-","")**  **; if deceased, do not include**  **Q:$P($G(^DPT(Y,.35)),U) 0**  **; filter out deceased patients**  **S (SMATCH,NMATCH,DMATCH)=0**  **; if ssn matches, go ahead and include in display**  **I ($L(CSSN)),($D(^DPT("SSN",CSSN,Y))) S SMATCH=1 Q 1**  **; TODO - NEED IA**  **I $P($P(^DPT(Y,0),U),",")[NAME S NMATCH=1**  **I $P(^DPT(Y,0),U,3)=DOB S DMATCH=1**  **;I $L(SSN),SMATCH,NMATCH,DMATCH Q 1**  **; making this less restrictive for now**  **I DMATCH Q 1**  **;I NMATCH,DMATCH Q 1**  **;I $P(^DPT(Y,0),U,3)'=DOB S VAL=0**  **Q 0**  **CHKPRV(NAME,NPI,DEA) ;**  **; not authorized to write medication orders**  **I $P($G(^VA(200,Y,"PS")),U)'=1 Q 0**  **I $G(NPI)]"",$D(^VA(200,"ANPI",NPI,Y)) Q 1**  **I $G(DEA)]"",$D(^VA(200,"PS1",DEA,Y)) Q 1**  **I $L($G(NAME)),$P(^VA(200,Y,0),U,1)[NAME Q 1**  **Q 0** |

Table 41: Routine: PSOERX1B

| Routines | Activities | | | |
| --- | --- | --- | --- | --- |
| **Routine Name** | PSOERX1B | | | |
| **Enhancement Category** | New | Modify | Delete | No Change |
| **RTM** | Accept functionality (Story #417254) | | | |
| **Related Options** |  | | | |

| Related Routines | Routines “Called By” | Routines “Called” |
| --- | --- | --- |
|  |  |  |

| Routines | Activities | | | | |
| --- | --- | --- | --- | --- | --- |
| **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) |
| --- |
| **PSOERX1B ;ALB/BWF - Accept eRx function ; 8/3/2016 5:14pm**  **;;7.0;OUTPATIENT PHARMACY;\*\*467\*\*;DEC 1997;Build 42**  **;**  **Q**  **;PSOHY("LOC")=IEN of hospital location file (#44) - NOT USED,**  **;PSOHY("CHNUM")=EXTERNAL PLACER ORDER NUMBER (NEED TO FIND OUT HOW WE S**  **HOULD SET THIS) (25)**  **;PSOHY("PICK")=MAIL/WINDOW (20.4)**  **;PSOHY("ENTER")=ENTERED BY IEN (2.1)**  **;PSOHY("PROV")=PROVIDER IEN (2.3)**  **;PSOHY("SDT")=EFFECTIVE DATE (6.3)**  **;PSOHY("ITEM")=PHARMACY ORDERABLE ITEM (DERIVED FROM THE DRUG IEN) - NO**  **MAPPING TO 52.49**  **;PSOHY("DRUG")=DRUG IEN (3.2)**  **;PSOHY("QTY")=QUANTITY (20.1)**  **;PSOHY("REF")=# OF REFILLS (20.5)**  **;PSOHY("PAT")=PATIENT IEN (.05)**  **;PSOHY("OCC")=ORDER TYPE (ALWAYS 'NW') - NO MAPPING TO 52.49**  **;PSOHY("EDT")=LOGIN DATE (TODAYS DATE) - NO MAPPING TO 52.49**  **;PSOHY("PRIOR")=PRIORITY (SET OF CODES, 52.41,25 - STAT, EMERGENCY, ROU**  **TINE)**  **;PSOHY("EXAPP")=EXTERNAL APPLICATION (FREE TEXT), LIKELY "PSO" - NO MAP**  **PING TO 52.49**  **;PSOHY("PRCOM",#)=PROVIDER COMMENTS (8- NOTES)**  **;PSOHY("SIG",#)=SIG (52.4911 (STRUCTURED SIG), FIELD 1 (SIG FREE TEXT)**  **;PSOHY("QTSUB",CNT)=QUANTITY TIMING SUBFILE DATA. MERGED IN, FULL SUBFI**  **LE DATA**  **; QUANTITY/TIMING MAPS DIRECTLY TO QUANTITY TIMING IN 52.41**  **SETUP ;**  **N PSOIENS,PSODAT,F,PATIEN,PROVIEN,OC,VQTY,EFFDT,VADRUG,VAOI,VAREF,VAROU**  **T,VAPRIOR,PSOHY,LOC,ERXNUM,PRVARY,PRVCOMM**  **N PLOOP,PCNT,QTLOOP,QTCNT,PSOEXMS,DIR,ORDERTYP,PSOEXCNT,SCNT,SIGDAT,SLO**  **OP,POORD**  **S F=52.49**  **Q:'$G(PSOIEN)**  **S VALMBCK="R"**  **S PSOIENS=PSOIEN\_","**  **D GETS^DIQ(F,PSOIENS,".05;.07;1;2.1;2.3;3.2;6.3;8;20.1;20.4;20.5;25;25.**  **2","IE","PSODAT")**  **S PSOEXCNT=0**  **S ERXSTA=$G(PSODAT(F,PSOIENS,1,"E")) I ERXSTA="E"!($E(ERXSTA)="H") S PS**  **OEXCNT=PSOEXCNT+1,PSOEXMS(PSOEXCNT)="eRx is in an 'Error' or 'Hold' status." D M**  **SGDIR(.PSOEXMS) Q**  **S POORD=$G(PSODAT(F,PSOIENS,25.2,"I")) I POORD S PSOEXCNT=PSOEXCNT+1,PS**  **OEXMS(PSOEXCNT)="Pending outpatient order already exists."**  **S PATIEN=$G(PSODAT(F,PSOIENS,.05,"I")) I 'PATIEN S PSOEXCNT=PSOEXCNT+1,**  **PSOEXMS(PSOEXCNT)="No matched vista patient."**  **S PROVIEN=$G(PSODAT(F,PSOIENS,2.3,"I")) I 'PROVIEN S PSOEXCNT=PSOEXCNT+**  **1,PSOEXMS(PSOEXCNT)="Provider not matched to a vista provider."**  **S VADRUG=$G(PSODAT(F,PSOIENS,3.2,"I")) I 'VADRUG S PSOEXCNT=PSOEXCNT+1,**  **PSOEXMS(PSOEXCNT)="Drug not matched to a vista drug."**  **S LOC=$G(PSODAT(F,PSOIENS,20.6,"I")) ;I 'LOC S PSOEXCNT=PSOEXCNT+1,PSOE**  **XMS(PSOEXCNT)="Hospital location missing."**  **S ERXNUM=$G(PSODAT(F,PSOIENS,25,"E")) I 'ERXNUM S PSOEXCNT=PSOEXCNT+1,P**  **SOEXMS(PSOEXCNT)="eRx number missing."**  **S VQTY=$G(PSODAT(F,PSOIENS,20.1,"E")) I 'VQTY S PSOEXCNT=PSOEXCNT+1,PSO**  **EXMS(PSOEXCNT)="Quantity missing."**  **S EFFDT=$G(PSODAT(F,PSOIENS,6.3,"I")) I 'EFFDT S PSOEXCNT=PSOEXCNT+1,PS**  **OEXMS(PSOEXCNT)="Effective date missing."**  **S VAOI=$$GET1^DIQ(50,VADRUG,2.1,"I") I 'VAOI S PSOEXCNT=PSOEXCNT+1,PSOE**  **XMS(PSOEXCNT)="Orderable item not associted with drug."**  **S VAREF=$G(PSODAT(F,PSOIENS,20.5,"E"))**  **S VAROUT=$G(PSODAT(F,PSOIENS,20.4,"I")) I '$L(VAROUT) S PSOEXCNT=PSOEXC**  **NT+1,PSOEXMS(PSOEXCNT)="Pickup routing missing."**  **; TODO - ensure this is the correct field to pull provider comments fro**  **m**  **S PRVCOMM=$G(PSODAT(F,PSOIENS,8,"E"))**  **D TXT2ARY^PSOERXD1(.PRVARY,PRVCOMM)**  **S (PLOOP,PCNT)=0 F S PLOOP=$O(PRVARY(PLOOP)) Q:'PLOOP D**  **.S PCNT=PCNT+1,PSOHY("PRCOM",PCNT)=$G(PRVARY(PLOOP))**  **I '$O(^PS(52.49,PSOIEN,21,0)) S PSOEXCNT=PSOEXCNT+1,PSOEXMS(PSOEXCNT)="**  **Dosing information missing."**  **S (QTLOOP,QTCNT)=0 F S QTLOOP=$O(^PS(52.49,PSOIEN,21,QTLOOP)) Q:'QTLOO**  **P D**  **.S QTCNT=QTCNT+1**  **.M PSOHY("QTSUB",QTCNT)=^PS(52.49,PSOIEN,21,QTLOOP)**  **; always 'routine' for now**  **S VAPRIOR="R"**  **; always 'new' for this version**  **S ORDERTYP="NW"**  **S PSOHY("LOC")=LOC**  **S PSOHY("CHNUM")=$G(ERXNUM)**  **S PSOHY("PICK")=VAROUT**  **S PSOHY("ENTER")=PROVIEN**  **S PSOHY("PROV")=PROVIEN**  **S PSOHY("SDT")=EFFDT**  **S PSOHY("ITEM")=VAOI**  **S PSOHY("DRUG")=VADRUG**  **S PSOHY("QTY")=VQTY**  **S PSOHY("REF")=VAREF**  **S (PSOHY("PAT"),DFN)=PATIEN**  **S PSOHY("OCC")=ORDERTYP**  **; login date will always be the date the prescription was processed fro**  **m the erx holding queue.**  **S PSOHY("EDT")=DT**  **S PSOHY("PRIOR")=VAPRIOR**  **; ALWAYS PSO as the external application**  **S PSOHY("EXAPP")="PHARMACY"**  **; sig from eRx**  **S (SLOOP,SCNT)=0 F S SLOOP=$O(^PS(52.49,PSOIEN,"SIG",SLOOP)) Q:'SLOOP**  **D**  **.S SIGDAT=$G(^PS(52.49,PSOIEN,"SIG",SLOOP,0))**  **.S SCNT=SCNT+1**  **.S PSOHY("SIG",SCNT)=SIGDAT**  **; if provider, patient or drug is missing, no need to continue.**  **; TODO - do we need to check more fields? i would suspect we need to ch**  **eck them all**  **I $D(PSOEXMS) D MSGDIR(.PSOEXMS) Q**  **D ADD**  **I $G(PSOEXMS)]"" W !,PSOEXMS S DIR(0)="E" D ^DIE**  **K DFN**  **Q**  **ADD ;Add CHCS message to Outpatient Pending Orders file**  **N PSOHQ,PSOHQT,PSOCPEND,PSOHINI,PSOHINLO,ERXSTA**  **; TODO - reactive this logic - BWF**  **;S (PSOHINI,PSOHINLO)=0 D**  **;.I $G(PSOHY("LOC")) S PSOHINLO=$P($G(^SC(PSOHY("LOC"),0)),"^",4) I PSO**  **HINLO Q**  **;.I $G(PSOHY("LOC")) S PSOHINI=$P($G(^SC(PSOHY("LOC"),0)),"^",15)**  **;.I '$G(PSOHINI) S PSOHINI=$O(^DG(40.8,0))**  **;.S PSOHINLO=+$$SITE^VASITE(PSOHINI)**  **; get institution from 52.49**  **S PSOHINLO=$$GET1^DIQ(52.49,PSOIEN,24.1,"I")**  **I +$G(PSOHINLO)<1 S PSOEXMS="Unable to derive Institution from Clinic."**  **Q**  **K DD,DO,DIC S X=PSOHY("CHNUM"),DIC="^PS(52.41,",DIC(0)="L"**  **S:$G(PSOHY("PICK"))="" PSOHY("PICK")="W"**  **S DIC("DR")="4////"\_$G(PSOHY("ENTER"))\_";5////"\_PSOHY("PROV")\_";6////"\_**  **$G(PSOHY("SDT"))\_";8////"\_PSOHY("ITEM")\_";11////"\_PSOHY("DRUG")\_";12////"\_$G(PSO**  **HY("QTY"))\_";13////"\_$G(PSOHY("REF"))**  **D FILE^DICN K DD,DIC,DO I Y<0 S PSOEXMS="Unable to add order to Pending**  **file." Q**  **S PSOCPEND=+Y**  **;S $P(^PS(52.41,PSOCPEND,0),"^",2)=PSOHY("PAT"),$P(^(0),"^",3)=PSOHY("O**  **CC"),$P(^(0),"^",12)=$G(PSOHY("EDT")),$P(^(0),"^",13)=PSOHY("LOC")**  **S $P(^PS(52.41,PSOCPEND,0),"^",2)=PSOHY("PAT"),$P(^(0),"^",3)=PSOHY("OC**  **C"),$P(^(0),"^",12)=$G(PSOHY("EDT"))**  **S $P(^PS(52.41,PSOCPEND,0),"^",14)=$G(PSOHY("PRIOR")),$P(^(0),"^",17)=$**  **G(PSOHY("PICK"))**  **S $P(^PS(52.41,PSOCPEND,"EXT"),"^")=PSOHY("CHNUM"),$P(^("EXT"),"^",2)=0**  **,$P(^("EXT"),"^",3)=PSOHY("EXAPP")**  **N DA,DIK S DA=PSOCPEND,DIK="^PS(52.41,",DIK(1)="114^C" D EN1^DIK**  **I $O(PSOHY("PRCOM",0)) D I PSOHQT S ^PS(52.41,PSOCPEND,3,0)="^^"\_PSOHQ**  **T\_"^"\_PSOHQT\_"^"\_DT\_"^"**  **.S PSOHQ="",PSOHQT=0 F S PSOHQ=$O(PSOHY("PRCOM",PSOHQ)) Q:PSOHQ="" I**  **$G(PSOHY("PRCOM",PSOHQ))'="" S PSOHQT=PSOHQT+1,^PS(52.41,PSOCPEND,3,PSOHQT,0)=$G**  **(PSOHY("PRCOM",PSOHQ))**  **I $O(PSOHY("SIG",0)) D I PSOHQT S ^PS(52.41,PSOCPEND,"SIG",0)="^52.412**  **4A^"\_PSOHQT\_"^"\_PSOHQT**  **.S PSOHQ="",PSOHQT=0 F S PSOHQ=$O(PSOHY("SIG",PSOHQ)) Q:PSOHQ="" I $G**  **(PSOHY("SIG",PSOHQ))'="" S PSOHQT=PSOHQT+1,^PS(52.41,PSOCPEND,"SIG",PSOHQT,0)=$G**  **(PSOHY("SIG",PSOHQ))**  **S $P(^PS(52.41,PSOCPEND,"INI"),"^")=$G(PSOHINLO)**  **; add quantity/timing subfile information**  **I $O(PSOHY("QTSUB",0)) D I PSOHQT S ^PS(52.41,PSOCPEND,1,0)="^52.413^"**  **\_PSOHQT\_"^"\_PSOHQT**  **.S PSOHQ="",PSOHQT=0 F S PSOHQ=$O(PSOHY("QTSUB",PSOHQ)) Q:PSOHQ="" D**  **..I $O(PSOHY("QTSUB",PSOHQ,0)) S PSOHQT=PSOHQT+1**  **..S ^PS(52.41,PSOCPEND,1,PSOHQT,0)=PSOHY("QTSUB",PSOHQ,0)**  **..S ^PS(52.41,PSOCPEND,1,PSOHQT,1)=PSOHY("QTSUB",PSOHQ,1)**  **..S ^PS(52.41,PSOCPEND,1,PSOHQT,2)=PSOHY("QTSUB",PSOHQ,2)**  **;Cross references not set yet preventing Pharmacy from finishing order**  **D EN^PSOHLSNC(PSOCPEND,"SN","IP")**  **D FULL^VALM1**  **;Just set to DC, don't delete because 52.41 entry would be re-used**  **;I '$P($G(^PS(52.41,PSOCPEND,"EXT")),"^",2) S DA=PSOCPEND,DIK="^PS(52.4**  **1," D ^DIK K DIK,DA S PSOEXMS="Unable to send CHCS order to CPRS." D NAK^PSOHLEX**  **C Q**  **I '$P($G(^PS(52.41,PSOCPEND,"EXT")),"^",2) D S $P(^PS(52.41,PSOCPEND,0**  **),"^",3)="DC" S PSOEXMS="Unable to send CHCS order to CPRS." Q**  **.;x-ref shouldn't be set, but we'll kill them just in case**  **.K ^PS(52.41,"AOR",$P(^PS(52.41,PSOCPEND,0),"^",2),+$P($G(^("INI")),"^"**  **),PSOCPEND),^PS(52.41,"AD",$P(^PS(52.41,PSOCPEND,0),"^",12),+$P($G(^("INI")),"^"**  **),PSOCPEND)**  **.K ^PS(52.41,"ACL",+$P(^PS(52.41,PSOCPEND,0),"^",13),$P(^(0),"^",12),PS**  **OCPEND),^PS(52.41,"AQ",+$P($G(^PS(52.41,PSOCPEND,0)),"^",21),PSOCPEND)**  **.S $P(^PS(52.41,PSOCPEND,4),"^")="External order, unable to successfull**  **y transmit to CPRS."**  **.W !!,"External order, unable to successfully transmit to CPRS."**  **.S DIR(0)="E" D ^DIR**  **;Successful transmission to CPRS**  **; add the pending outpatient order number to 52.49 and update status of**  **eRx to PR (Processed)**  **S ERXSTA=$O(^PS(52.45,"C","ERX","PR",0))**  **S ORDNUM=$P($G(^PS(52.41,PSOCPEND,0)),U)**  **S DIE="^PS(52.49,",DR="25.2///"\_PSOCPEND\_";1///"\_ERXSTA\_";.12///"\_ORDNU**  **M,DA=PSOIEN D ^DIE K DIE**  **S DA=PSOCPEND,DIK="^PS(52.41," D IX^DIK**  **W !!,"eRx #"\_PSOHY("CHNUM")\_" sent to PENDING OUTPATIENT ORDERS!"**  **S DIR(0)="E" D ^DIR**  **Q**  **;SIGS ;**  **;N PSOHZZZ,PSOHZZZZ**  **;S PSOHZZZ=1,PSOHZZZZ=""**  **;F S PSOHZZZZ=$O(^PS(52.41,ORD,"SIG",PSOHZZZZ)) Q:PSOHZZZZ="" I $D(^(**  **PSOHZZZZ,0)) S SIG(PSOHZZZ)=$G(^(0)),PSOHZZZ=PSOHZZZ+1**  **;I $O(PSONEW("SIG",""))'="" S PSOHZZZZ="" F S PSOHZZZZ=$O(PSONEW("SIG"**  **,PSOHZZZZ)) Q:PSOHZZZZ="" S SIG(PSOHZZZ)=$G(PSONEW("SIG",PSOHZZZZ)),PSOHZZZ=PSO**  **HZZZ+1**  **;I $O(PSONEW("SIG",""))'="" Q**  **;I $O(PRC(""))'="" S PSOHZZZZ="" F S PSOHZZZZ=$O(PRC(PSOHZZZZ)) Q:PSOH**  **ZZZZ="" I $D(PRC(PSOHZZZZ)) S SIG(PSOHZZZ)=$G(PRC(PSOHZZZZ)),PSOHZZZ=PSOHZZZ+1**  **;Q**  **MSGDIR(MSG) ;**  **N DIR,MLOOP,MTXT**  **S VALMBCK="R"**  **D FULL^VALM1**  **W !!,"Errors encountered during processing:",!**  **S MLOOP=0 F S MLOOP=$O(MSG(MLOOP)) Q:'MLOOP D**  **.S MTXT=$G(MSG(MLOOP)) W !,MLOOP\_".) ",MTXT**  **W !!,"Cannot process eRx.",!**  **S DIR(0)="E" D ^DIR**  **Q**  **; remove eRx from holding queue**  **REM ;**  **N DIR,Y,PSSRET,PSOIENS,REMIEN,REMSTA,REMTXT,ERXRMIEN**  **D FULL^VALM1**  **S PSOIENS=PSOIEN\_","**  **S VALMBCK="R"**  **S DIR(0)="YO",DIR("A")="Would you like to 'Remove' eRx #"\_$$GET1^DIQ(52**  **.49,PSOIEN,.01,"E") D ^DIR K DIR**  **Q:'Y**  **S DIR(0)="P^52.45",DIR("S")="I $D(^PS(52.45,""C"",""REM"",X))" D ^DIR K**  **DIR**  **I $P(Y,U)<1 W !,"Remove reason required!" Q**  **S REMIEN=$P(Y,U)**  **S REMSTA=$P(Y,U,2)**  **S DIR(0)="FO^1:70",DIR("A")="Enter removal reason" D ^DIR K DIR**  **I '$L(Y) W !,"Removal reason required. eRx NOT removed." S DIR(0)="E" D**  **^DIR K DIR Q**  **S REMTXT=Y**  **S FDA(52.4919,"+1,"\_PSOIENS,.01)=$$NOW^XLFDT**  **S FDA(52.4919,"+1,"\_PSOIENS,.02)=REMIEN**  **S FDA(52.4919,"+1,"\_PSOIENS,.03)=DUZ**  **S FDA(52.4919,"+1,"\_PSOIENS,1)=REMTXT**  **D UPDATE^DIE(,"FDA") K FDA**  **; SET THE ERX STATUS TO THE REMOVAL REASON**  **S ERXRMIEN=$O(^PS(52.45,"C","ERX","RM",0))**  **S DIE="^PS(52.49,",DR="1///"\_ERXRMIEN,DA=PSOIEN D ^DIE**  **K DIE,DA,DR**  **Q**  **; reject eRx**  **REJ ;**  **N DIR,Y,PSSRET,PSOIENS,REMTXT,REJSTA,FULLTXT,ERXRJIEN,REJDESC,REJIEN,RE**  **JTXT**  **D FULL^VALM1**  **S PSOIENS=PSOIEN\_","**  **S VALMBCK="R"**  **S DIR(0)="YO",DIR("A")="Would you like to 'Reject' eRx #"\_$$GET1^DIQ(52**  **.49,PSOIEN,.01,"E") D ^DIR K DIR**  **Q:'Y**  **S DIR(0)="P^52.45",DIR("S")="I $D(^PS(52.45,""C"",""REJ"",X))" D ^DIR K**  **DIR**  **I $P(Y,U)<1 W !,"Reject reason required!" Q**  **S REJIEN=$P(Y,U)**  **S REJSTA=$P(Y,U,2)**  **K Y**  **S DIR(0)="FO^1:70",DIR("A")="Enter reject reason" D ^DIR K DIR**  **I '$L(Y)!(Y="^") W !,"Reject reason required. eRx NOT rejected." S DIR(**  **0)="E" D ^DIR K DIR Q**  **; if reject reason was entered, log the activity.**  **S REJTXT=Y**  **S REJDESC=$$GET1^DIQ(52.45,REJIEN,.02,"E")**  **S FULLTXT=REJSTA\_"-"\_REJDESC**  **D POST^PSOERXO1(PSOIEN,.PSSRET,900,"",$E(FULLTXT,1,70))**  **; if the post was unsuccessful, inform the user and quit.**  **I $P(PSSRET(0),U)<1 W !,$P(PSSRET(0),U,2) S DIR(0)="E" D ^DIR K DIR Q**  **I $D(PSSRET("errorMessage")) W !,PSSRET("errorMessage") S DIR(0)="E" D**  **^DIR K DIR Q**  **W !!,"Rejection message sent." S DIR(0)="E" D ^DIR K DIR**  **; if post is successful, change the eRx status and log the status activ**  **ity.**  **S FDA(52.4919,"+1,"\_PSOIENS,.01)=$$NOW^XLFDT**  **S FDA(52.4919,"+1,"\_PSOIENS,.02)=REJIEN**  **S FDA(52.4919,"+1,"\_PSOIENS,.03)=DUZ**  **; todo - may concatenate the code to the reject text here**  **S FDA(52.4919,"+1,"\_PSOIENS,1)=REJTXT**  **D UPDATE^DIE(,"FDA") K FDA**  **; SET THE ERX STATUS TO THE REJECT REASON**  **S ERXRJIEN=$O(^PS(52.45,"C","ERX","RJ",0))**  **S DIE="^PS(52.49,",DR="1///"\_ERXRJIEN,DA=PSOIEN D ^DIE**  **K DIE,DR,DA**  **Q** |

Table 42: Routine: PSOERXA0

| Routines | Activities | | | |
| --- | --- | --- | --- | --- |
| **Routine Name** | PSOERXA0 | | | |
| **Enhancement Category** | New | Modify | Delete | No Change |
| **RTM** | eRx Utilities and RPCs | | | |
| **Related Options** |  | | | |

| Related Routines | Routines “Called By” | Routines “Called” |
| --- | --- | --- |
|  |  |  |

| Routines | Activities | | | | |
| --- | --- | --- | --- | --- | --- |
| **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) |
| --- |
| **PSOERXA0 ;ALB/BWF - eRx Utilities/RPC's ; 8/3/2016 5:14pm**  **;;7.0;OUTPATIENT PHARMACY;\*\*467\*\*;DEC 1997;Build 42**  **Q**  **; test ndc/upn match**  **SQA1 ; FOR NDC**  **N RES,NDCUPN**  **R "ENTER THE NDC: ",NDCUPN D DRGMTCH(.RES,"N^"\_NDCUPN) W !,RES**  **Q**  **SQA2 ; FOR UPN**  **N RES,NDCUPN**  **R "ENTER THE UPN: ",NDCUPN D DRGMTCH(.RES,"U^"\_NDCUPN) W !,RES**  **Q**  **SQA3 ; FOR DRUG NAME/VA PRODUCT NAME/VA GENERIC**  **N RES,DRUG**  **R "ENTER DRUG NAME: ",DRUG D DRGMTCH(.RES,"",DRUG) W !,RES**  **Q**  **TNDCUPN ;**  **N X,NDC,UPN,PID,RES,STR**  **S X=0 F S X=$O(^PSNDF(50.67,X)) Q:'X D**  **.S NDC=$$GET1^DIQ(50.67,X,1,"E"),UPN=$$GET1^DIQ(50.67,X,2,"E")**  **.S PID=$$GET1^DIQ(50.67,X,5,"I") Q:'PID**  **.;I NDC]"" K RES D DRGMTCH(.RES,NDC) I $P(RES,U)>0 W !,"NDC: "\_NDC\_"^",**  **RES**  **.;I UPN]"" K RES D DRGMTCH(.RES,NDC) I $P(RES,U)>0 W !,"UPN: "\_UPN\_"^",**  **RES**  **.;I NDC]"" S NDC="N"\_U\_NDC K RES D DRGMTCH(.RES,NDC) S STR=$S($P(RES,U)**  **>0:" MATCHED TO ",1:" NOT MATCHED -") W !,"NDC: "\_NDC\_STR,RES**  **.I UPN]"" S UPN="U"\_U\_UPN K RES D DRGMTCH(.RES,UPN) S STR=$S($P(RES,U)>**  **0:" MATCHED TO ",1:" NOT MATCHED -") W !,"UPN: "\_UPN\_STR,RES**  **Q**  **; test generic name match**  **TGEN ;**  **;"METHADONE HCL 5MG TAB" - FINDS A MATCH USING 50.68,"B" - DIRECT 1:1**  **; "DOCUSATE SODIUM 100MG CAP" - FINDS A MATCH USING 50,"B" - DIRECT 1:1**  **W !,"Matching - METHADONE HCL 5MG TAB" D DRGMTCH(.RES,"","METHADONE HCL**  **5MG TAB") W !,RES**  **W !!!,"Matching - DOCUSATE SODIUM 100MG CAP" D DRGMTCH(.RES,"","DOCUSAT**  **E SODIUM 100MG CAP") W !,RES**  **Q**  **; test Branded drug name match**  **TBRND ;**  **Q**  **; All parameters are optional, however at least one needs to be passed**  **in for processing to be sucessful.**  **; NDCUPN - This is the NDC/UPN for the drug (optional)**  **; DGDESC - Drug description (optional)**  **DRGMTCH(PSORES,NDCUPN,DGDESC) ;**  **N VAPRID,NDCUPNT,NDCUPNV,NDCUXREF,NUIEN,NDCERR,PSMIEN,NDCERR,PSDRG,PSDR**  **GCNT,PSDGLST**  **S (PSORES,NDCERR)=0**  **I $G(DGDESC)]"" D**  **.S DGDESC=$$UP^XLFSTR(DGDESC)**  **.I $D(^PSDRUG("B",DGDESC)) D**  **..S (PSDRG,PSDRGCNT)=0 F S PSDRG=$O(^PSDRUG("B",DGDESC,PSDRG)) Q:'PSDR**  **G D**  **...I '$$ACTIVE(PSDRG)!('$$OUTPAT(PSDRG))!($$INVCOMP(PSDRG))!($$CS(PSDRG**  **)) Q**  **...S PSDRGCNT=PSDRGCNT+1,PSDGLST(PSDRG)=""**  **..I PSDRGCNT>1 S PSORES="0^More than one possible drug match found. Pha**  **rmacist review required."**  **..I PSDRGCNT=1 S PSMIEN=$O(^PSDRUG("B",DGDESC,0)),PSORES=PSMIEN\_U\_$$GET**  **1^DIQ(50,PSMIEN,.01,"E") Q**  **.I $D(^PSNDF(50.68,"B",DGDESC)) S VAPRID=$O(^PSNDF(50.68,"B",DGDESC,0))**  **Q**  **.; is it possible to have more than one drug or va product match here?**  **.01 fields are unique!**  **; direct match in DRUG file**  **I $P(PSORES,U)=1 Q**  **; direct match in VA PRODUCT file**  **I $G(VAPRID) D VAPRID(.PSORES,VAPRID) I $P(PSORES,U) Q**  **I $G(NDCUPN)']"",$P(PSORES,U,2)]"" Q**  **; check the NDC/UPN if passed in**  **I $G(NDCUPN)]"" D**  **.S NDCUPNT=$P(NDCUPN,U),NDCUPNV=$P(NDCUPN,U,2),NDCUXREF=$S(NDCUPNT="N":**  **"NDC",NDCUPNT="U":"UPN",1:"")**  **.S NUIEN=$O(^PSNDF(50.67,NDCUXREF,NDCUPNV,0)) I 'NUIEN S NDCERR=1 Q**  **.S VAPRID=$$GET1^DIQ(50.67,NUIEN,5,"I")**  **.I VAPRID D VAPRID(.PSORES,VAPRID)**  **I $P(PSORES,U) Q**  **I $G(NDCERR) S PSORES="0^NDC/UPN match not found." Q**  **I $P(PSORES,U)=0 S PSORES="0^No matches found."**  **Q**  **VAPRID(PSORES,VAPID) I '$G(VAPRID) S PSORES="0^No VA PRODUCT associated with thi**  **s NDC/UPN." Q**  **N VAPMTCH,VAPCNT,VAPDRG,PSODRG**  **;N VAPSTREN**  **; get the product strenth**  **;S VAPSTREN=$$GET1^DIQ(50.68,VAPID,2,"E")**  **S (VAPMTCH,VAPCNT)=0 F S VAPMTCH=$O(^PSDRUG("APR",VAPRID,VAPMTCH)) Q:'**  **VAPMTCH D**  **.; ONLY GET MEDICATIONS FOR OUTPATIENT USE, AND ARE NOT MARKED INACTIVE**  **.I '$$OUTPAT(VAPMTCH)!('$$ACTIVE(VAPMTCH))!($$INVCOMP(VAPMTCH))!($$CS(V**  **APMTCH)) Q**  **.;S DRGSTREN=$$GET1^DIQ(50,VAPMTCH,901,"E") I VAPSTREN]"",VAPSTREN'=DRG**  **STREN Q**  **.S VAPDRG(VAPMTCH)="",VAPCNT=VAPCNT+1**  **I VAPCNT=1 S PSODRG=$O(VAPDRG(0)),PSORES=PSODRG\_U\_$$GET1^DIQ(50,PSODRG,**  **.01,"E") Q**  **I VAPCNT>1 S PSORES="0^Multiple matched drugs found. Pharmacist review**  **required." Q**  **;I $G(DGENNM)]""**  **Q**  **; active drug check**  **ACTIVE(DIEN) ;**  **N INACTDT**  **;S INACTDT=$$GET1^DIQ(50,DIEN,100,"I") I INACTDT,INACTDT<DT Q 0**  **S INACTDT=$P($G(^PSDRUG(DIEN,"I")),U) I INACTDT,INACTDT<DT Q 0**  **Q 1**  **; check to see if this is drug is marked for outpatient use**  **OUTPAT(DIEN) ;**  **I $P($G(^PSDRUG(DIEN,2)),U,3)["O" Q 1**  **;I $$GET1^DIQ(50,DIEN,63,"E")["O" Q 1**  **Q 0**  **; check to see if the drug is investigational or compond**  **INVCOMP(DIEN) ;**  **N X**  **;S X=$$GET1^DIQ(50,DIEN,3,"E")**  **S X=$P($G(^PSDRUG(DIEN,0)),U,3)**  **; if a supply, not controlled substance**  **I X="S" Q 0**  **I X["I"!(X["0")!(X["M") Q 1**  **Q 0**  **CS(DIEN) ;**  **N X**  **S X=$P($G(^PSDRUG(DIEN,0)),U,3)**  **I X["S" Q 0**  **I X]"",+X<6 Q 1**  **Q 0**  **CHKSTR() ;**  **Q**  **TPRVMTCH ;**  **N X,Y,TRES**  **S X="" F S X=$O(^VA(200,"PS1",X)) Q:X="" D**  **.S Y=0 F S Y=$O(^VA(200,"PS1",X,Y)) Q:'Y D**  **..K TRES D PRVMTCH(.TRES,"",X) I $P(TRES,U)=0 W !,TRES\_" "\_X Q**  **..I $P(TRES,U) W !,X,?20,$$GET1^DIQ(200,Y,.01,"E")**  **Q**  **; Match provider given NPI, DEA, or provider name.**  **; NPI - NPI value for the provider**  **; DEA - Providers' DEA number**  **; CS - controlled substance (1-yes, 0 or "" - no)**  **PRVMTCH(PSORES,NPI,DEA,CS) ;**  **N NPIEN,MATCH,VAL,NVAL,INDEX,NPCNT,NPLIST,DEACNT,SRCH,DEACNT,DEAMTCH,ND**  **MTCH,DEAIEN**  **N DEACHK**  **S (PSORES,MATCH)=0**  **S NPI=$G(NPI,""),DEA=$G(DEA,"")**  **I NPI="",DEA="" S PSORES="0^NPI and DEA# missing." Q**  **I $G(CS),DEA="" S PSORES="0^DEA # must be provided with controlled subs**  **tances." Q**  **I $G(CS),NPI="" S PSORES="0^NPI must be provided with controlled substa**  **nces." Q**  **I $G(CS),'$D(^VA(200,"ANPI",NPI)) S PSORES="0^NPI# does not exist in th**  **is system." Q**  **I $G(CS),'$D(^VA(200,"PS1",DEA)) S PSORES="0^DEA# does not exist in thi**  **s system." Q**  **I '$G(CS),NPI="" D Q**  **.I DEA="" S PSORES="0^Missing DEA number." Q**  **.I '$D(^VA(200,"PS1",DEA)) S PSORES="0^DEA# does not exist at this loca**  **tion." Q**  **.S (DEACHK,DEACNT)=0 F S DEACHK=$O(^VA(200,"PS1",DEA,DEACHK)) Q:'DEACH**  **K D**  **..S DEACNT=$G(DEACNT)+1**  **.I DEACNT>1 S PSORES="0^Multiple DEA matches found." Q**  **.I DEACNT=1 S DEAIEN=$O(^VA(200,"PS1",DEA,0))**  **.I '$$MEDAUTH(DEAIEN) S PSORES="0^DEA match, not authorized to write me**  **dication orders." Q**  **.S PSORES=DEAIEN\_U\_$$GET1^DIQ(200,DEAIEN,.01,"E")**  **I '$D(^VA(200,"ANPI",NPI)) S PSORES="0^No matching NPI." Q**  **; get a list of providers that match the NPI#**  **S (NPIEN,NPCNT)=0 F S NPIEN=$O(^VA(200,"ANPI",NPI,NPIEN)) Q:'NPIEN D**  **.S NPLIST(NPIEN)="",NPCNT=$G(NPCNT)+1**  **; no matches**  **I '$D(NPLIST) S PSORES="0^Could not match provided NPI." Q**  **I '$G(CS),NPCNT>1 S PSORES="0^Multiple provider matches found." Q**  **I NPCNT=0 S PSORES="0^No NPI match found." Q**  **I '$G(CS),NPCNT=1 D Q**  **.S NDMTCH=$O(NPLIST(0))**  **.I '$$MEDAUTH(NDMTCH) S RES="0^NPI match found, not authorized to write**  **medication orders." Q**  **.S PSORES=NDMTCH\_U\_$$GET1^DIQ(200,$O(NPLIST(0)),.01,"E")**  **; if this is a controlled substance, we must match both the NPI and the**  **DEA#**  **S (SRCH,DEACNT)=0 F S SRCH=$O(NPLIST(SRCH)) Q:'SRCH D**  **.I '$D(^VA(200,"PS1",DEA,SRCH)) Q**  **.S DEAMTCH(SRCH)="",DEACNT=$G(DEACNT)+1**  **I DEACNT>1 S PSORES="0^Multiple DEA matches found." Q**  **I DEACNT=0 S PSORES="0^NPI match, DEA mismatch." Q**  **S NDMTCH=$O(DEAMTCH(0))**  **I '$$MEDAUTH(NDMTCH) S PSORES="0^NPI/DEA match, not authorized to write**  **medication orders." Q**  **I NDMTCH S PSORES=NDMTCH\_U\_$$GET1^DIQ(200,NDMTCH,.01,"E") Q**  **S PSORES="0^Matching procedure completed with no results."**  **Q**  **; ensure the dea# is active**  **DEACTIVE(USER) ;**  **N EXPDT**  **S EXPDT=$$GET1^DIQ(200,USER,747.44,"I")**  **I EXPDT,EXPDT<DT Q 0**  **Q 1**  **; check to ensure the provider is authorized to write med orders**  **MEDAUTH(USER) ;**  **Q $$GET1^DIQ(200,USER,53.1,"I")**  **;**  **; Match patient**  **; PSORES - results returned to calling application**  **; PATNM - Patient Name**  **; SOCIAL - Patient SSN**  **; ICN - Patient ICN from MPI??**  **PATMTCH(PSORES,PATNM,SOCIAL,ICN) ;**  **N PSOCIAL,PSOPAT**  **S PSORES=0**  **I $G(SOCIAL)]"" S PSOCIAL=$TR(SOCIAL,"-","")**  **I $L(PSOCIAL)'=9 S PSORES="-1^Invalid SSN."**  **S PID=$O(^DPT("SSN",PSOCIAL,0)) I PID Q PID\_U\_$$GET1^DIQ(2,PID,.01,"E")**  **I $G(PATNM)]""**  **Q**  **NEWERX(RES,PSINC) ;**  **Q** |

Table 43: Routine: PSOERXA1

| Routines | Activities | | | |
| --- | --- | --- | --- | --- |
| **Routine Name** | PSOERXA1 | | | |
| **Enhancement Category** | New | Modify | Delete | No Change |
| **RTM** | eRx Utilities and RPCs  Files incoming XML into appropriate file: Provider check, patient check, and drug check. | | | |
| **Related Options** |  | | | |

| Related Routines | Routines “Called By” | Routines “Called” |
| --- | --- | --- |
|  |  |  |

| Routines | Activities | | | | |
| --- | --- | --- | --- | --- | --- |
| **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) |
| --- |
| **PSOERXA1 ;ALB/BWF - eRx Utilities/RPC's ; 8/3/2016 5:14pm**  **;;7.0;OUTPATIENT PHARMACY;\*\*467\*\*;DEC 1997;Build 42**  **Q**  **; File incoming XML into appropriate file**  **; XML - xml text**  **; PRCHK - provider check information**  **; PACHK - patient check information**  **; DACHK - drug auto check**  **; STATION - station #**  **; DIV - division**  **; ERXHID - eRx processing hub id**  **INCERX(RES,XML,PRCHK,PACHK,DACHK,STATION,DIV,ERXHID) ;**  **N CURREC,FDA,EIEN,ERRTXT,ERRSEQ,PACNT,PASCNT,PAICN,PAIEN,VAINST**  **S CURREC=$$PARSE(.XML)**  **I $P(CURREC,U)<1 D Q**  **.I $L($P(CURREC,U,2)) S RES=CURREC Q**  **.S RES="0^XML received. Error creating or finding associated record in**  **the ERX Holding queue." Q**  **S EIEN=CURREC**  **S CURREC=CURREC\_","**  **; TODO - check to ensure this works and is the correct way to get the i**  **nstitution**  **; - create cross reference to filter eRx records by institution, o**  **r outpatient site, or both.**  **S VAINST=$$FIND1^DIC(4,,"O",STATION,"D")**  **I '$G(VAINST) S RES="-1^Institution could not be resolved. eRx not file**  **d." Q**  **I $G(VAINST) S FDA(52.49,CURREC,24.1)=VAINST D FILE^DIE(,"FDA") K FDA**  **; Process auto-validation results. only log positive results for now -**  **bwf 2/1/2017**  **K FDA**  **; set up the erx hub id**  **S FDA(52.49,CURREC,25)=ERXHID**  **I DACHK("success")="true" D**  **.I $G(DACHK("IEN")) D**  **..S FDA(52.49,CURREC,1.4)=1**  **..S FDA(52.49,CURREC,3.2)=DACHK("IEN")**  **I DACHK("success")="false" D**  **.S ERRTXT=$G(DACHK("error"))**  **.S ERRSEQ=$$ERRSEQ() Q:'ERRSEQ**  **.D FILERR(CURREC,ERRSEQ,"D","E",ERRTXT)**  **.; todo - add error to subfile 100 (processing errors)**  **I $G(PRCHK("success"))="true" D**  **.I PRCHK("IEN") D**  **..S FDA(52.49,CURREC,1.2)=1**  **..S FDA(52.49,CURREC,2.3)=PRCHK("IEN")**  **I $G(PRCHK("success"))="false" D**  **.S ERRTXT=$G(PRCHK("error"))**  **.S ERRSEQ=$$ERRSEQ() Q:'ERRSEQ**  **.D FILERR(CURREC,ERRSEQ,"PR","E",ERRTXT)**  **I PACHK("success")="true" D**  **.S PAICN=+$P($G(PACHK("ICN")),"V")**  **.I PAICN D**  **..S (PAIEN,PACNT)=0 F S PAIEN=$O(^DPT("AICN",PAICN,PAIEN)) Q:'PAIEN D**  **...S PACNT=PACNT+1**  **...; TODO - if we find more than one match in the local system, do we l**  **og some sort of an error?**  **.I $G(PACNT)=1 D Q**  **..S FDA(52.49,CURREC,1.6)=1**  **..S FDA(52.49,CURREC,.05)=$O(^DPT("AICN",PAICN,0))**  **.I $L(PACHK("ssn")) D**  **..S (PASCNT,PAIEN)=0 F S PAIEN=$O(^DPT("SSN",$TR(PACHK("ssn"),"-",""),**  **PAIEN)) Q:'PAIEN D**  **...S PASCNT=PASCNT+1**  **.I $G(PASCNT)=1 D Q**  **..S FDA(52.49,CURREC,1.6)=1**  **..S FDA(52.49,CURREC,.05)=$O(^DPT("SSN",PACHK("ssn"),0))**  **I $G(PACHK("success"))="false" D**  **.; file e&e error**  **.S ERRTXT=$G(PACHK("EandEerror")) I ERRTXT]"" D**  **..S ERRSEQ=$$ERRSEQ() Q:'ERRSEQ**  **..D FILERR(CURREC,ERRSEQ,"PA","E",ERRTXT)**  **.; file mvi error**  **.S ERRTXT=$G(PACHK("MVIerror")) I ERRTXT]"" D**  **..S ERRSEQ=$$ERRSEQ() Q:'ERRSEQ**  **..D FILERR(CURREC,ERRSEQ,"PA","E",ERRTXT)**  **I $D(FDA) D FILE^DIE(,"FDA") K FDA**  **S RES="1^Erx Received."**  **Q**  **ERRSEQ() ;**  **N SEQ**  **I '$O(^PS(52.49,EIEN,100,0)) S SEQ=1**  **I '$G(SEQ) S SEQ=$O(^PS(52.49,EIEN,100,999999),-1)+1**  **Q SEQ**  **;**  **; IENS - ien of the erx holding queue entry**  **; SEQ - sequence number of the error**  **; TYPE - type (d-drug, pr-provider, pa-patient, a-allergy, o-order chec**  **k, t-transfer)**  **; SRC - v - vista, e - erx processing hub**  **; TXT - error text**  **FILERR(IENS,SEQ,TYPE,SRC,TXT) ;**  **N FDA**  **S FDA(52.49101,"+1,"\_IENS,.01)=SEQ**  **S FDA(52.49101,"+1,"\_IENS,.02)=TYPE**  **S FDA(52.49101,"+1,"\_IENS,.03)=SRC**  **; all new errors are set to pending**  **S FDA(52.49101,"+1,"\_IENS,.04)="P"**  **S FDA(52.49101,"+1,"\_IENS,1)=TXT**  **D UPDATE^DIE(,"FDA") K FDA**  **; now set the eRx entry status to error**  **S FDA(52.49,IENS,1)=$O(^PS(52.45,"C","ERX","E",0)) D FILE^DIE(,"FDA") K**  **FDA**  **Q**  **PARSE(STREAM) ;**  **N %XML,GL**  **S GL=$NA(^TMP($J,"PSOERXO1"))**  **K @GL**  **N STATUS,READER,XOBERR,S,ATTR**  **S STATUS=##class(%XML.TextReader).ParseStream(STREAM,.READER,,,,,1)**  **I $$STATCHK^XOBWLIB(STATUS,.XOBERR,1) D**  **.N BREAK**  **.S BREAK=0 F Q:BREAK||READER.EOF||'READER.Read() D**  **..N X,PUSHED,PARENT**  **..I READER.AttributeCount D**  **...S PARENT=READER.LocalName**  **...D SPUSH(.S,PARENT) S PUSHED=1**  **...F ATTR=1:1:READER.AttributeCount D**  **....D READER.MoveToAttributeIndex(ATTR)**  **....I READER.NodeType="attribute" D APUT(.S,READER.Value,READER.LocalNa**  **me)**  **..I READER.NodeType="element",'$G(PUSHED) D SPUSH(.S,READER.LocalName)**  **..I READER.NodeType="endelement" D SPOP(.S,.X)**  **..I READER.NodeType="chars" D SPUT(.S,READER.Value)**  **N NERXIEN,ERR**  **S NERXIEN=$$HDR()**  **I $P(NERXIEN,U)<1 Q NERXIEN**  **D PAT(NERXIEN),BFC(NERXIEN),PHR^PSOERXA2(NERXIEN),PRE^PSOERXA2(NERXIEN)**  **D MED^PSOERXA3(NERXIEN),OBS(NERXIEN),SUP^PSOERXA2(NERXIEN)**  **; facility/request have no where to go at this point in time??**  **;D FAC(ERXIEN),REQ(ERXIEN)**  **Q NERXIEN**  **HDR() ; header information**  **N GL,GL2,FQUAL,TQUAL,FROM,TO,MID,PONUM,SRTID,SSTID,SENTTIME,RTMID,FDA,E**  **RXIEN,FMID,NEWERX,MES,ERXIENS**  **S GL=$NA(^TMP($J,"PSOERXO1","Message",0,"Header",0))**  **S GL2=$NA(^TMP($J,"PSOERXO1","Message","A","Qualifier","Header","A","Qu**  **alifier"))**  **; from and to qualifiers**  **S FQUAL=$G(@GL2@("From","A","Qualifier"))**  **S TQUAL=$G(@GL2@("To","A","Qualifier"))**  **; from, to, message id, prescriber order number**  **S FROM=$G(@GL@("From",0))**  **S TO=$G(@GL@("To",0))**  **S MID=$G(@GL@("MessageID",0))**  **; set up the full message id**  **S FMID=MID**  **; This logic was intended to handle 'updates' to and existing message i**  **f the same message id was received**  **; however, updates should probably occur through a change request, so i**  **f the same message id is received, we**  **; are returning an error.**  **;I $D(^PS(52.49,"FMID",FMID)) D**  **;.S ERXIEN=$O(^PS(52.49,"FMID",FMID,0)),UPDATE=1**  **;I '$G(ERXIEN) S ERXIEN="+1"**  **S ERXIENS="+1,"**  **; todo - FOR NOW, quit and return a message back if this eRx exists.**  **I $D(^PS(52.49,"FMID",FMID)) S MES="-1^This message already exists. Cha**  **nges must occur via a change request XML message." Q MES**  **S PONUM=$G(@GL@("PrescriberOrderNumber",0))**  **; security receiver tertiary identification**  **S SRTID=$G(@GL@("Security",0,"Receiver",0,"TertiaryIdentification,",0))**  **; security sender tertiary identification**  **S SSTID=$G(@GL@("Security",0,"Sender",0,"TertiaryIdentification,",0))**  **; convert senttime to file manager dt/tm**  **S SENTTIME=$G(@GL@("SentTime",0)),SENTTIME=$$CONVDTTM^PSOERXA1(SENTTIME**  **)**  **S RTMID=$G(@GL@("RelatesToMessageID",0))**  **S FDA(52.49,ERXIENS,.01)=FMID**  **S FDA(52.49,ERXIENS,.02)=RTMID**  **; todo - do we use the message sentTime, or now? if we use now, do we a**  **dd a message sentTime field?**  **S FDA(52.49,ERXIENS,.03)=SENTTIME**  **S FDA(52.49,ERXIENS,.09)=PONUM**  **S FDA(52.49,ERXIENS,1)=$$PRESOLV^PSOERXA1("N","ERX")**  **S FDA(52.49,ERXIENS,22.1)=FROM**  **S FDA(52.49,ERXIENS,22.2)=FQUAL**  **S FDA(52.49,ERXIENS,22.3)=TO**  **S FDA(52.49,ERXIENS,22.4)=TQUAL**  **S FDA(52.49,ERXIENS,22.5)=$$NOW^XLFDT**  **; if this is an existing record, file the updates to the erx and return**  **the IEN**  **;I $G(UPDATE) D FILE^DIE(,"FDA") K FDA Q ERXIEN**  **D UPDATE^DIE(,"FDA","NEWERX","EERR") K FDA**  **S ERXIEN=""**  **S ERXIEN=$O(NEWERX(0)),ERXIEN=$G(NEWERX(ERXIEN))**  **I 'ERXIEN Q ""**  **; TODO - XSD shows digital signature. Do we need to collect this?**  **Q ERXIEN**  **BFC(ERXIEN) ; benefits coordination**  **N GL,BFCCNT,CHFN,CHLN,CHMN,CHPRE,CHSUFF,CHID,GRPID,PAYIDTYP,PAYIDVAL,CH**  **FN,F,PIEN,NEWPAYER,BFCERR,IENS,CHFULLN,FDA**  **S F=52.4918**  **S GL=$NA(^TMP($J,"PSOERXO1","Message",0,"Body",0,"NewRx",0,"BenefitsCoo**  **rdination"))**  **; cannot start at 0, since the first entry is on the 0 subscript.**  **S BFCCNT=-1 F S BFCCNT=$O(@GL@(BFCCNT)) Q:BFCCNT="" D**  **.S CHFN=$$UP^XLFSTR($G(@GL@(BFCCNT,"CardHolderName",0,"FirstName",0)))**  **.S CHLN=$$UP^XLFSTR($G(@GL@(BFCCNT,"CardHolderName",0,"LastName",0)))**  **.S CHMN=$$UP^XLFSTR($G(@GL@(BFCCNT,"CardHolderName",0,"MiddleName",0)))**  **.; todo - do we need suffix and prefix in the full name?**  **.; set up full name - last, first, mi**  **.S CHFULLN=CHLN\_","\_CHFN\_$S(CHMN]"":" "\_CHMN,1:"")**  **.S CHPRE=$$UP^XLFSTR($G(@GL@(BFCCNT,"CardHolderName",0,"Prefix",0)))**  **.S CHSUFF=$$UP^XLFSTR($G(@GL@(BFCCNT,"CardHolderName",0,"Suffix",0)))**  **.S CHID=$G(@GL@(BFCCNT,"CardholderID",0))**  **.S GRPID=$G(@GL@(BFCCNT,"GroupID",0))**  **.S IENS="+1,"\_ERXIEN\_","**  **.S FDA(F,IENS,.01)=CHID**  **.S FDA(F,IENS,.02)=GRPID**  **.S FDA(F,IENS,.03)=CHFULLN**  **.S FDA(F,IENS,1)=CHLN,FDA(F,IENS,2)=CHFN,FDA(F,IENS,3)=CHMN,FDA(F,IENS,**  **4)=CHSUFF,FDA(F,IENS,5)=CHPRE**  **.D UPDATE^DIE(,"FDA","NEWPAYER","ERR") K FDA**  **.S PIEN=$O(NEWPAYER(0)),PIEN=$G(NEWPAYER(PIEN))**  **.I 'PIEN Q**  **.; todo - file payer ID types - requires modification to the current pa**  **yer information subfile**  **.; - THIS REQUIRES RESOLUTION OF THE PAYERID TYPE ISSUE ALONG WITH**  **PRIOR AUTH VALUES, ETC.**  **.S PAYIDTYP="" F S PAYIDTYP=$O(@GL@(BFCCNT,"PayerIdentification",0,PAY**  **IDTYP)) Q:PAYIDTYP="" D**  **..S PAYIDVAL=$G(@GL@(BFCCNT,"PayerIdentification",0,PAYIDTYP,0))**  **..S FDA(52.49186,"+1,"\_PIEN\_","\_ERXIEN\_",",.01)=PAYIDTYP**  **..S FDA(52.49186,"+1,"\_PIEN\_","\_ERXIEN\_",",.02)=PAYIDVAL**  **..D UPDATE^DIE(,"FDA") K FDA**  **.K NEWPAYER,PIEN**  **Q**  **OBS(ERXIEN) ; Observation**  **N GL,I,LAST,DIM,MSOURCE,MUNIT,OBSDT,MVAL,OBSNOTE,OBSCNT,F,EIENS,FDA**  **S GL=$NA(^TMP($J,"PSOERXO1","Message",0,"Body",0,"NewRx",0,"Observation**  **",0))**  **S F=52.4914,EIENS=ERXIEN\_","**  **S I=-1,OBSCNT=0 F S I=$O(@GL@("Measurement",I)) Q:I="" D**  **.S OBSCNT=OBSCNT+1,FDA(F,"+1,"\_EIENS,.01)=OBSCNT**  **.S DIM=$G(@GL@("Measurement",I,"Dimension",0)),FDA(F,"+1,"\_EIENS,.02)=D**  **IM**  **.S MSOURCE=$G(@GL@("Measurement",I,"MeasurementSourceCode",0)),FDA(F,"+**  **1,"\_EIENS,.06)=MSOURCE**  **.S MUNIT=$G(@GL@("Measurement",I,"MeasurementUnitCode",0)),FDA(F,"+1,"\_**  **EIENS,.07)=MUNIT**  **.; TODO - we have a field for MeasurementDataQualifier (.05) - is this**  **valid?**  **.S OBSDT=$G(@GL@("Measurement",I,"ObservationDate",0,"Date",0)),OBSDT=$**  **$CONVDTTM^PSOERXA1(OBSDT),FDA(F,"+1,"\_EIENS,.04)=OBSDT**  **.S MVAL=$G(@GL@("Measurement",I,"Value",0)),FDA(F,"+1,"\_EIENS,.03)=MVAL**  **.D UPDATE^DIE(,"FDA") K FDA**  **S OBSNOTE=$G(@GL@("ObservationNotes",0)),FDA(52.49,EIENS,15)=OBSNOTE D**  **FILE^DIE(,"FDA") K FDA**  **Q**  **PAT(ERXIEN) ; patient**  **N GL,AL1,AL2,CITY,STATE,ZIP,LN,FN,MN,PREF,SUFF,COMQUAL,COMVAL,PLQUAL,DO**  **B,GEN,PRELATE,IDDONE,CDONE,I,C,CQUAL,CVAL**  **N IDNM,IDVAL,PFN,ERXPAT,NEWPAT,F,EIENS,FDA,IDFND,SRCH,PIENS,NPIEN,PATSS**  **N,PREL**  **S F=52.46**  **S EIENS=ERXIEN\_","**  **S GL=$NA(^TMP($J,"PSOERXO1","Message",0,"Body",0,"NewRx",0,"Patient",0)**  **)**  **S PREL=$G(@GL@("PatientRelationship",0))**  **S FN=$$UP^XLFSTR($G(@GL@("Name",0,"FirstName",0)))**  **S LN=$$UP^XLFSTR($G(@GL@("Name",0,"LastName",0)))**  **S MN=$$UP^XLFSTR($G(@GL@("Name",0,"MiddleName",0)))**  **S PFN=LN\_","\_FN\_$S(MN]"":" "\_MN,1:"")**  **S SUFF=$$UP^XLFSTR($G(@GL@("Name",0,"Suffix",0)))**  **S PREF=$$UP^XLFSTR($G(@GL@("Name",0,"Prefix",0)))**  **S PRELATE=$G(@GL@("PatientRelationship",0))**  **S GEN=$G(@GL@("Gender",0))**  **S DOB=$G(@GL@("DateOfBirth",0,"Date",0)),DOB=$$CONVDTTM^PSOERXA1(DOB)**  **S AL1=$G(@GL@("Address",0,"AddressLine1",0))**  **S AL2=$G(@GL@("Address",0,"AddressLine2",0))**  **S CITY=$G(@GL@("Address",0,"City",0))**  **S STATE=$G(@GL@("Address",0,"State",0))**  **S ZIP=$G(@GL@("Address",0,"ZipCode",0)),ZIP=$E(ZIP,1,5)**  **; need to check for SSN before trying to match the patient. This needs**  **to be stored in an array for later processing**  **; check 52.46 for a match before filing**  **S PATSSN=$G(@GL@("Identification",0,"SocialSecurity",0))**  **S ERXPAT=$$FINDPAT^PSOERXA1(PFN,DOB,GEN,$G(PATSSN),$G(AL1)) S PIENS=$S(**  **ERXPAT:ERXPAT\_",",1:"+1,")**  **; first, lets set up the main part**  **S FDA(F,PIENS,.01)=PFN,FDA(F,PIENS,.02)=LN,FDA(F,PIENS,.03)=FN,FDA(F,PI**  **ENS,.04)=MN,FDA(F,PIENS,.05)=SUFF,FDA(F,PIENS,.06)=PREF**  **S FDA(F,PIENS,.07)=GEN,FDA(F,PIENS,.08)=DOB**  **S FDA(F,PIENS,1.4)=PATSSN,FDA(F,PIENS,1.7)=PREL**  **S FDA(F,PIENS,3.1)=AL1,FDA(F,PIENS,3.2)=AL2,FDA(F,PIENS,3.3)=CITY**  **S FDA(F,PIENS,3.4)=$$FIND1^DIC(5,,,STATE,"C")**  **S FDA(F,PIENS,3.5)=$$FZIP(ZIP,CITY,STATE)**  **I PIENS["+" D Q**  **.D UPDATE^DIE(,"FDA","NEWPAT") K FDA**  **.S NPIEN=$O(NEWPAT(0)),NPIEN=$G(NEWPAT(NPIEN))**  **.Q:'NPIEN**  **.S NPIEN=NPIEN**  **.D PATC(NPIEN)**  **.S FDA(52.49,EIENS,.04)=NPIEN D FILE^DIE(,"FDA") K FDA**  **D FILE^DIE(,"FDA") K FDA D PATC(ERXPAT)**  **S FDA(52.49,EIENS,.04)=ERXPAT D FILE^DIE(,"FDA") K FDA**  **; if this is an existing record loop through the communication values a**  **nd do a compare to see what needs to be updated**  **; otherwise, build the FDA and file the communication values.**  **Q**  **PATC(IEN) ; patient communication**  **N IENS,CQUAL,CVAL,COMARY,FDA,SRCH,IDFND,IDNM,IDVAL,IDARY,PATSSN**  **Q:'IEN**  **S IENS=IEN\_","**  **; Kill off existing communication values**  **K ^PS(52.46,IEN,3)**  **S C=-1 F S C=$O(@GL@("CommunicationNumbers",0,"Communication",C)) Q:C=**  **"" D**  **.S CQUAL=$G(@GL@("CommunicationNumbers",0,"Communication",C,"Qualifier"**  **,0))**  **.S CVAL=$G(@GL@("CommunicationNumbers",0,"Communication",C,"Number",0))**  **.; TODO - can we have more than one of the same type??**  **.S COMARY(CQUAL)=CVAL**  **.S FDA(52.462,"+1,"\_IENS,.01)=CVAL**  **.S FDA(52.462,"+1,"\_IENS,.02)=CQUAL**  **.D UPDATE^DIE(,"FDA") K FDA**  **S IDNM="" F S IDNM=$O(@GL@("Identification",0,IDNM)) Q:IDNM="" D**  **.S IDVAL=$G(@GL@("Identification",0,IDNM,0))**  **.I IDNM="SocialSecurity" S PATSSN=IDVAL**  **.S IDARY(IDNM)=IDVAL**  **.S IDFND=0**  **.S SRCH=0 F S SRCH=$O(^PS(52.46,IEN,5,SRCH)) Q:'SRCH D**  **..I $$GET1^DIQ(52.465,SRCH\_","\_IEN\_",",.01)=IDNM D**  **...S IDFND=1**  **...S FDA(52.465,SRCH\_","\_IEN\_",",.02)=IDVAL D FILE^DIE(,"FDA") K FDA**  **.Q:IDFND**  **.S FDA(52.465,"+1,"\_IEN\_",",.01)=IDNM**  **.S FDA(52.465,"+1,"\_IEN\_",",.02)=IDVAL**  **.D UPDATE^DIE(,"FDA") K FDA**  **I $G(PATSSN)]"" S FDA(52.46,IENS,1.4)=PATSSN D FILE^DIE(,"FDA") K FDA**  **Q**  **REQ ; request**  **N GL,CRTYPE,RETREC,RRNUM**  **S GL=$NA(^TMP($J,"PSOERXO1","Message",0,"Body",0,"NewRx",0,"Request",0)**  **)**  **S CRTYPE=$G(@GL@("ChangeRequestType",0))**  **S RETREC=$G(@GL@("ReturnReceipt",0))**  **S RRNUM=$G(@GL@("RequestReferenceNumber",0))**  **; TODO - What needs to be done with these?**  **Q**  **;**  **SPUSH(S,X) ;places X on the stack S and returns the current level of the stack**  **N I S I=$O(S(""),-1)+1,S(I)=X**  **Q I**  **;**  **SPOP(S,X) ;removes the top item from the stack S and put it into the variable X**  **and returns the level that X was at**  **N I S I=$O(S(""),-1)**  **I I S X=S(I) K S(I)**  **N J S J=$O(S(I),-1) I J S S(J,X)=$G(S(J,X))+1**  **Q I**  **;**  **SPEEK(S,X) ;same as SPOP except the top item is not removed**  **N I S I=$O(S(""),-1)**  **I I S X=S(I)**  **Q I**  **;**  **SPUT(S,X) ;implementation specific, uses the stack to form a global node**  **N I,STR**  **S STR=$P(GL,")")**  **S I=0 F S I=$O(S(I)) Q:'I D**  **.S STR=STR\_","\_""""\_S(I)\_""""\_","**  **.N NUM S NUM=0**  **.I $D(S(I-1,S(I))) S NUM=+$G(S(I-1,S(I)))**  **.S STR=STR\_NUM**  **S STR=STR\_")"**  **I $D(@STR) S @STR=@STR\_X**  **I '$D(@STR) S @STR=X**  **Q STR**  **APUT(S,X,LN) ; what am i doing here?**  **N I,STR**  **S STR=$P(GL,")")**  **S I=0 F S I=$O(S(I)) Q:'I D**  **.S STR=STR\_","\_""""\_S(I)\_""""\_","**  **.N NUM S NUM="""A"""**  **.;I $D(S(I-1,S(I))) S NUM=+$G(S(I-1,S(I)))**  **.;S STR=STR\_NUM**  **.S STR=STR\_NUM\_","\_""""\_LN\_""""**  **S STR=STR\_")"**  **I $D(@STR) S @STR=@STR\_X**  **I '$D(@STR) S @STR=X**  **Q STR**  **;**  **; find zip code**  **; ZIP - zip code from xml**  **; CITY - city from xml**  **; STATE - state abbreviation or name from xml**  **FZIP(ZIP,CITY,STATE) ;**  **N SZIP,SIEN,MSTATE,DIITEM,ZDATA,ZIEN,ZZIP,ZCITY,ZSTATE,MTCH,ZMATCH**  **I '$L(ZIP) Q ""**  **I ZIP["-" S ZIP=$P(ZIP,"-")**  **I '$D(^VIC(5.11,"C",ZIP)) Q:""**  **I $L(STATE)=2 S SIEN=$$FIND1^DIC(5,,,STATE,"C")**  **I $G(SIEN) S MSTATE=$$GET1^DIQ(5,SIEN,.01,"E")**  **I '$D(MSTATE) S MSTATE=STATE**  **; if a single match is found, return it, no need to search further**  **S SZIP=$$FIND1^DIC(5.11,,"B",ZIP) I SZIP Q SZIP**  **D FIND^DIC(5.11,,"1;3","P",ZIP,,,,,"ZMATCH")**  **I '$D(ZMATCH) Q ""**  **S MTCH=""**  **S DIITEM=0 F S DIITEM=$O(ZMATCH("DILIST",DIITEM)) Q:'DIITEM!(MTCH) D**  **.S ZDATA=$G(ZMATCH("DILIST",DIITEM,0))**  **.S ZIEN=$P(ZDATA,U),ZZIP=$P(ZDATA,U,2),ZCITY=$P(ZDATA,U,3),ZSTATE=$P(ZD**  **ATA,U,4)**  **.I ZCITY=$G(CITY),ZSTATE=$G(MSTATE) S MTCH=ZIEN Q**  **Q MTCH**  **; VAL - value to resolve**  **; TYPE - This is the code type, which will tell which 'C' index type to**  **get the code from**  **PRESOLV(VAL,TYPE) ;**  **N MATCH**  **S MATCH=""**  **Q:'$L(TYPE)!('$L(VAL)) "" ; avoid null subscript**  **S MATCH=$O(^PS(52.45,"C",TYPE,VAL,0))**  **;S MATCH=$$FIND1^DIC(52.45,"","O",VAL,"C","I $D(^PS(52.45,""C"",TYPE,VA**  **L))","ERR")**  **;S MATCH=$$FIND1^DIC(52.45,"","O",.VAL,"C","","ERR")**  **; return the match found, null if no match**  **Q MATCH**  **; look for existing patient**  **; NAME - PATIENT FULL NAME**  **; IDOB - INCOMING PATIENT DOB**  **; IDGEN - INCOMING PATIENT GENDER**  **; SSN - INCOMING PATIENT SSN**  **; AL1 - INCOMING PATIENT ADDRESS LINE 1**  **FINDPAT(NAME,IDOB,IGEN,SSN,AL1) ;**  **N MPAT,MTCHCNT,PIEN,MATCH,PDOB,PGEN,PSSN,PAL1**  **;D FIND^DIC(52.46,"",".01;.07;.08","",NAME,,"BN",,,"PAT")**  **;S MPAT=$$FIND1^DIC(52.46,"","",NAME,"BN",,"PAT")**  **; TODO - dont quit simply because name match does not occur.**  **I '$D(^PS(52.46,"BN",NAME)) Q ""**  **S MTCHCNT=0**  **S PIEN=0 F S PIEN=$O(^PS(52.46,"BN",NAME,PIEN)) Q:'PIEN D**  **.S PDOB=$$GET1^DIQ(52.46,PIEN,.08,"I"),PGEN=$$GET1^DIQ(52.46,PIEN,.07,"**  **I")**  **.S PSSN=$$GET1^DIQ(52.46,PIEN,1.4),PAL1=$$GET1^DIQ(52.46,PIEN,3.1,"E")**  **.; if the ssn exists, and does not match, quit**  **.I $L(SSN),SSN'=PSSN Q**  **.I PDOB=IDOB,PGEN=IGEN,AL1=PAL1 S MTCHCNT=MTCHCNT+1,MATCH(PIEN)=""**  **I MTCHCNT'=1 Q ""**  **S MPAT=$O(MATCH(0))**  **I MPAT Q MPAT**  **;S SVAL=$G(FDA(52.46,IENS,1.4))**  **;D FIND^DIC(52.46,"",".01;.02;.07;.08;1.4","",SVAL,,"SSN",,,"PAT")**  **Q ""**  **; look for exising provider/prescriber**  **FINDPRE(NAME,NPI,DEA) ;**  **N MPRV,MTCHCNT,PIEN,PNPI,MATCH**  **; TODO - dont quit if name match fails. may need to check npi instead**  **;I '$D(^PS(52.48,"BN",NAME)) Q ""**  **; First see if there is a single record that matches both dea and npi v**  **alues**  **;S FRSTPRV=$$FIND1^DIC(52.48,**  **S MTCHCNT=0**  **S PIEN=0 F S PIEN=$O(^PS(52.46,"BN",NAME,PIEN)) Q:'PIEN D**  **.S PNPI=$$GET1^DIQ(52.48,PIEN,1.5,"E")**  **.I $L(NPI),PNPI'=NPI Q**  **.S MTCHCNT=MTCHCNT+1,MATCH(PIEN)=""**  **; no matches, or more than 1 match**  **I MTCHCNT'=1 Q ""**  **S MPRV=$O(MATCH(0))**  **I MPRV Q MPRV**  **;S SVAL=$G(FDA(52.48,IENS,1.2,"E"))**  **;D FIND^DIC(52.48,"",".01;1.1;1.2;1.3;1.4","M",SVAL,,,,,"PROV")**  **Q ""**  **CONVDTTM(VAL) ;**  **N EDATE,ETIME,X,ETZ,Y**  **S EDATE=$P(VAL,"T"),ETIME=$P(VAL,"T",2)**  **; split off time zone**  **S ETZ=$P(ETIME,".",2)**  **S ETIME=$P(ETIME,".")**  **S X=EDATE D ^%DT I 'Y Q ""**  **S VAL=Y\_$S($L(ETIME):"."\_$TR(ETIME,":",""),1:"")**  **Q VAL** |

Table 44: Routine: PSOERXA2

| Routines | Activities | | | |
| --- | --- | --- | --- | --- |
| **Routine Name** | PSOERXA2 | | | |
| **Enhancement Category** | New | Modify | Delete | No Change |
| **RTM** | eRx Utilities and RPCs | | | |
| **Related Options** |  | | | |

| Related Routines | Routines “Called By” | Routines “Called” |
| --- | --- | --- |
|  |  |  |

| Routines | Activities | | | | |
| --- | --- | --- | --- | --- | --- |
| **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) |
| --- |
| **PSOERXA2 ;ALB/BWF - eRx Utilities/RPC's ; 8/3/2016 5:14pm**  **;;7.0;OUTPATIENT PHARMACY;\*\*467\*\*;DEC 1997;Build 46**  **Q**  **FAC ; facility**  **N GL,IDTYPE,IDVAL**  **S GL=$NA(^TMP($J,"PSOERXO1","Message",0,"Body",0,"NewRx",0,"Facility",0**  **))**  **; todo - file ID types - requires modification to the current payer inf**  **ormation subfile**  **; - THIS REQUIRES RESOLUTION OF THE PAYERID TYPE ISSUE ALONG WITH**  **PRIOR AUTH VALUES, ETC.**  **S IDTYPE="" F S IDTYPE=$O(@GL@("Identification",0,IDTYPE)) Q:IDTYPE=""**  **D**  **.S IDVAL=$G(@GL@("Identification",0,IDTYPE,0))**  **Q**  **PHR(ERXIEN) ; pharamcy**  **N GL,SNAME,AL1,AL2,CIT,STATE,ZIP,PLQUAL,COMTYP,COMVAL,I,F,EIENS,PHIEN,C**  **CNT,NEW,SPEC,FDA,NEWPHIEN**  **S GL=$NA(^TMP($J,"PSOERXO1","Message",0,"Body",0,"NewRx",0,"Pharmacy",0**  **))**  **S F=52.47,PHIEN=""**  **S EIENS=ERXIEN\_","**  **S SNAME=$G(@GL@("StoreName",0))**  **I $D(^PS(52.47,"B",SNAME)) S PHIEN=$O(^PS(52.47,"B",SNAME,0)) I PHIEN S**  **PHIEN=PHIEN\_",",NEW=0**  **; if we found a match, clear out the existing communication numbers and**  **identification**  **I PHIEN K ^PS(52.47,$P(PHIEN,","),3),^PS(52.47,$P(PHIEN,","),2)**  **I '$G(PHIEN) S PHIEN="+1,",NEW=1,FDA(F,PHIEN,.01)=SNAME**  **S FDA(F,PHIEN,.05)=SNAME**  **S SPEC=$G(@GL@("Specialty",0)),FDA(F,PHIEN,1.8)=SPEC**  **; TODO - WHY DO WE HAVE THE NAME TWICE IN THE FILE?**  **S AL1=$G(@GL@("Address",0,"AddressLine1",0)),FDA(F,PHIEN,1.1)=AL1**  **S AL2=$G(@GL@("Address",0,"AddressLine2",0)),FDA(F,PHIEN,1.2)=AL2**  **S CITY=$G(@GL@("Address",0,"City",0)),FDA(F,PHIEN,1.3)=CITY**  **S STATE=$G(@GL@("Address",0,"State",0)),FDA(F,PHIEN,1.4)=$$FIND1^DIC(5,**  **,,STATE,"C")**  **S ZIP=$G(@GL@("Address",0,"ZipCode",0)),FDA(F,PHIEN,1.5)=ZIP**  **S PLQUAL=$G(@GL@("Address",0,"PlaceLocationQualifier",0)),FDA(F,PHIEN,1**  **.7)=PLQUAL**  **; if this is an existing pharmacy entry, file the updates, communicatio**  **n values, and identification, then link to 52.49**  **I 'NEW D Q**  **.D FILE^DIE(,"FDA") K FDA D PHRIC($P(PHIEN,","))**  **.S FDA(52.49,EIENS,2.5)=$P(PHIEN,",") D FILE^DIE(,"FDA") K FDA**  **; for a new entry, file the entry, then file communication/identificati**  **on and link to 52.49**  **D UPDATE^DIE(,"FDA","NEWPHIEN") K FDA**  **S PHIEN=$O(NEWPHIEN(0)),PHIEN=$G(NEWPHIEN(PHIEN))**  **Q:'PHIEN**  **D PHRIC(PHIEN)**  **S FDA(52.49,EIENS,2.5)=PHIEN D FILE^DIE(,"FDA") K FDA**  **Q**  **PHRIC(IEN) ; pharmacy identification and communication information**  **N IDTYP,IDVAL,FDA,I,CCNT,PHIEN,FDA,IDCNT**  **S PHIEN=IEN\_","**  **S IDCNT=0**  **S IDNM="" F S IDNM=$O(@GL@("Identification",0,IDNM)) Q:IDNM="" D**  **.S IDVAL=$G(@GL@("Identification",0,IDNM,0))**  **.I IDNM="NCPDPID" S NCPDPID=$G(IDVAL)**  **.S IDARY(IDNM)=IDVAL**  **.S IDFND=0**  **.S SRCH=0 F S SRCH=$O(^PS(52.47,IEN,2,SRCH)) Q:'SRCH D**  **..I $$GET1^DIQ(52.472,SRCH\_","\_IEN\_",",.01)=IDNM D**  **...S IDFND=1**  **...S FDA(52.472,SRCH\_","\_IEN\_",",.02)=IDVAL D FILE^DIE(,"FDA") K FDA**  **.Q:IDFND**  **.S FDA(52.472,"+1,"\_IEN\_",",.01)=IDNM**  **.S FDA(52.472,"+1,"\_IEN\_",",.02)=IDVAL**  **.D UPDATE^DIE(,"FDA") K FDA**  **I $G(NCPDPID)]"" S FDA(52.47,PHIEN,.02)=NCPDPID D FILE^DIE(,"FDA") K FD**  **A**  **; clear out existing communication Numbers**  **K ^PS(52.47,IEN,3)**  **S I=-1 F S I=$O(@GL@("CommunicationNumbers",0,"CommunicationNumber",I)**  **) Q:I="" D**  **.S CCNT=$G(CCNT)+1**  **.S COMVAL=$G(@GL@("CommunicationNumbers",0,"CommunicationNumber",I,"Num**  **ber",0))**  **.S COMTYP=$G(@GL@("CommunicationNumbers",0,"CommunicationNumber",I,"Qua**  **lifier",0))**  **.S FDA(52.473,"+"\_CCNT\_","\_PHIEN,.01)=COMVAL**  **.S FDA(52.473,"+"\_CCNT\_","\_PHIEN,.02)=COMTYP**  **D UPDATE^DIE(,"FDA") K FDA**  **Q**  **PRE(ERXIEN) ; prescriber**  **N GL,FN,LN,MN,SUFF,PREF,AL1,AL2,CITY,STATE,ZIP,IDDONE,I,IDNM,IDVAL,C,CQ**  **UAL,CVAL,SPEC,AFN,ALN,AMN,APREF,ASUFF,FULLNM**  **N EIENS,FDA,NPIEN,NEW,PRVIEN,PRVIENS,NEWIEN,IDFND,SRCH,PNPI,PDEA**  **S GL=$NA(^TMP($J,"PSOERXO1","Message",0,"Body",0,"NewRx",0,"Prescriber"**  **,0))**  **S F=52.48**  **S EIENS=ERXIEN\_","**  **S FN=$$UP^XLFSTR($G(@GL@("Name",0,"FirstName",0)))**  **S LN=$$UP^XLFSTR($G(@GL@("Name",0,"LastName",0)))**  **S MN=$$UP^XLFSTR($G(@GL@("Name",0,"MiddleName",0)))**  **S FULLNM=LN\_","\_FN\_$S(MN]"":" "\_MN,1:"")**  **S SUFF=$$UP^XLFSTR($G(@GL@("Name",0,"Suffix",0)))**  **S PREF=$$UP^XLFSTR($G(@GL@("Name",0,"Prefix",0)))**  **S AL1=$G(@GL@("Address",0,"AddressLine1",0))**  **S AL2=$G(@GL@("Address",0,"AddressLine2",0))**  **S CITY=$G(@GL@("Address",0,"City",0))**  **S STATE=$G(@GL@("Address",0,"State",0))**  **S ZIP=$G(@GL@("Address",0,"ZipCode",0))**  **S SPEC=$G(@GL@("Specialty",0))**  **S AFN=$$UP^XLFSTR($G(@GL@("PrescriberAgent",0,"FirstName",0)))**  **S ALN=$$UP^XLFSTR($G(@GL@("PrescriberAgent",0,"LastName",0)))**  **S AMN=$$UP^XLFSTR($G(@GL@("PrescriberAgent",0,"MiddleName",0)))**  **S APREF=$$UP^XLFSTR($G(@GL@("PrescriberAgent",0,"Prefix",0)))**  **S ASUFF=$$UP^XLFSTR($G(@GL@("PrescriberAgent",0,"Suffix",0)))**  **; try to match the provider/supervisor. if no match, create a new entr**  **y for this provider**  **; TODO - DO NOT HAVE THE NPI, SO NO MATCH CAN BE FOUND. NEED TO SEARCH**  **THE IDENTIFICATION MULTIPLE**  **S PNPI=$G(@GL@("Identification",0,"NPI",0))**  **S PDEA=$G(@GL@("Identification",0,"DEANumber",0))**  **; TODO - enhance incoming provider matching logic**  **S PRVIEN=$$FINDPRE^PSOERXA1(FULLNM,$G(PNPI),PDEA) I PRVIEN S NEW=0**  **I 'PRVIEN S NEW=1,PRVIEN="+1"**  **S PRVIENS=PRVIEN\_","**  **S FDA(F,PRVIENS,.01)=FULLNM,FDA(F,PRVIENS,.02)=LN,FDA(F,PRVIENS,.03)=FN**  **,FDA(F,PRVIENS,.04)=MN,FDA(F,PRVIENS,.05)=SUFF**  **S FDA(F,PRVIENS,.06)=PREF,FDA(F,PRVIENS,4.1)=AL1,FDA(F,PRVIENS,4.2)=AL2**  **,FDA(F,PRVIENS,4.3)=CITY**  **S FDA(F,PRVIENS,4.4)=$$FIND1^DIC(5,,,STATE,"C"),FDA(F,PRVIENS,4.5)=ZIP**  **S FDA(F,PRVIENS,5.1)=ALN,FDA(F,PRVIENS,5.2)=AFN,FDA(F,PRVIENS,5.3)=AMN,**  **FDA(F,PRVIENS,5.4)=ASUFF,FDA(F,PRVIENS,5.5)=APREF**  **S FDA(F,PRVIENS,1.2)=SPEC**  **S FDA(F,PRVIENS,1.8)=$G(STLIC)**  **S FDA(F,PRVIENS,1.1)="PR"**  **I 'NEW D Q**  **.D FILE^DIE(,"FDA")**  **.D PRVCI(PRVIEN)**  **.S FDA(52.49,EIENS,2.1)=PRVIEN D FILE^DIE(,"FDA") K FDA**  **D UPDATE^DIE(,"FDA","NEWIEN") K FDA**  **S NPIEN=$O(NEWIEN(0)),NPIEN=$G(NEWIEN(NPIEN))**  **D PRVCI(NPIEN)**  **S FDA(52.49,EIENS,2.1)=NPIEN D FILE^DIE(,"FDA") K FDA**  **Q**  **PRVCI(IEN) ;**  **N IENS,C,CQUAL,CVAL,FDA,IDNM,IDVAL,IDARY,IDFND,SRCH,NCPDPID,PHIN,DEA,ST**  **LIC,PNPI**  **S IENS=IEN\_","**  **; kill off existing data**  **K ^PS(52.48,IEN,3)**  **S C=-1 F S C=$O(@GL@("CommunicationNumbers",0,"Communication",C)) Q:C=**  **"" D**  **.S CQUAL=$G(@GL@("CommunicationNumbers",0,"Communication",C,"Qualifier"**  **,0))**  **.S CVAL=$G(@GL@("CommunicationNumbers",0,"Communication",C,"Number",0))**  **.S FDA(52.483,"+1,"\_IENS,.01)=CVAL**  **.S FDA(52.483,"+1,"\_IENS,.02)=CQUAL**  **.D UPDATE^DIE(,"FDA") K FDA**  **S IDNM="" F S IDNM=$O(@GL@("Identification",0,IDNM)) Q:IDNM="" D**  **.S IDVAL=$G(@GL@("Identification",0,IDNM,0))**  **.S IDARY(IDNM)=IDVAL**  **.I IDNM="NCPDPID" S NCPDPID=IDVAL**  **.I IDNM="HIN" S PHIN=IDVAL**  **.I IDNM="DEANumber" S DEA=IDVAL**  **.I IDNM="StateLicenseNumber" S STLIC=IDVAL**  **.I IDNM="NPI" S PNPI=IDVAL**  **.S IDFND=0**  **.S SRCH=0 F S SRCH=$O(^PS(52.48,IEN,6,SRCH)) Q:'SRCH D**  **..I $$GET1^DIQ(52.486,SRCH\_","\_IEN\_",",.01)=IDNM D**  **...S IDFND=1**  **...S FDA(52.486,SRCH\_","\_IEN\_",",.02)=IDVAL D FILE^DIE(,"FDA") K FDA**  **.Q:IDFND**  **.S FDA(52.486,"+1,"\_IEN\_",",.01)=IDNM**  **.S FDA(52.486,"+1,"\_IEN\_",",.02)=IDVAL**  **.D UPDATE^DIE(,"FDA") K FDA**  **S FDA(52.48,IENS,1.4)=$G(NCPDPID),FDA(52.48,IENS,1.5)=$G(PNPI),FDA(52.4**  **8,IENS,1.6)=$G(DEA),FDA(52.48,IENS,1.7)=$G(PHIN)**  **D FILE^DIE(,"FDA") K FDA**  **Q**  **REQ ; request**  **N GL,CRTYPE,RETREC,RRNUM**  **S GL=$NA(^TMP($J,"PSOERXO1","Message",0,"Body",0,"NewRx",0,"Request",0)**  **)**  **S CRTYPE=$G(@GL@("ChangeRequestType",0))**  **S RETREC=$G(@GL@("ReturnReceipt",0))**  **S RRNUM=$G(@GL@("RequestReferenceNumber",0))**  **; TODO - What needs to be done with these?**  **Q**  **SUP(ERXIEN) ; supervisor**  **N GL,FN,LN,MN,SUFF,PREF,AL1,AL2,CITY,STATE,ZIP,IDDONE,I,IDNM,IDVAL,C,CQ**  **UAL,CVAL,SPEC,AFN,ALN,AMN,APREF,ASUFF,EIENS**  **N FDA,NPIEN,NEW,PRVIEN,PRVIENS,NEWIEN,FDA,IDFND,SRCH**  **S GL=$NA(^TMP($J,"PSOERXO1","Message",0,"Body",0,"NewRx",0,"Supervisor"**  **,0))**  **S EIENS=ERXIEN\_","**  **S FN=$$UP^XLFSTR($G(@GL@("Name",0,"FirstName",0)))**  **S LN=$$UP^XLFSTR($G(@GL@("Name",0,"LastName",0)))**  **S MN=$$UP^XLFSTR($G(@GL@("Name",0,"MiddleName",0)))**  **S FULLNM=LN\_","\_FN\_$S(MN]"":" "\_MN,1:"")**  **S SUFF=$$UP^XLFSTR($G(@GL@("Name",0,"Suffix",0)))**  **S PREF=$$UP^XLFSTR($G(@GL@("Name",0,"Prefix",0)))**  **S AL1=$G(@GL@("Address",0,"AddressLine1",0))**  **S AL2=$G(@GL@("Address",0,"AddressLine2",0))**  **S CITY=$G(@GL@("Address",0,"City",0))**  **S STATE=$G(@GL@("Address",0,"State",0))**  **S ZIP=$G(@GL@("Address",0,"ZipCode",0))**  **S SPEC=$G(@GL@("Specialty",0))**  **S PRVIEN=$$FINDPRE^PSOERXA1(FULLNM,$G(PNPI)) I PRVIEN S NEW=0**  **I 'PRVIEN S NEW=1,PRVIEN="+1"**  **S PRVIENS=PRVIEN\_","**  **S FDA(F,PRVIENS,.01)=FULLNM,FDA(F,PRVIENS,.02)=LN,FDA(F,PRVIENS,.03)=FN**  **,FDA(F,PRVIENS,.04)=MN,FDA(F,PRVIENS,.05)=SUFF**  **S FDA(F,PRVIENS,.06)=PREF,FDA(F,PRVIENS,4.1)=AL1,FDA(F,PRVIENS,4.2)=AL2**  **,FDA(F,PRVIENS,4.3)=CITY**  **; STATE AND POINTER RESOLUTION**  **S FDA(F,PRVIENS,4.4)=$$FIND1^DIC(5,,,STATE,"C"),FDA(F,PRVIENS,4.5)=ZIP**  **S FDA(F,PRVIENS,1.2)=SPEC**  **;S FDA(F,PRVIENS,1.4)=$G(NCPDPID),FDA(F,PRVIENS,1.5)=$G(PNPI),FDA(F,PRV**  **IENS,1.6)=$G(DEA),FDA(F,PRVIENS,1.7)=$G(HIN)**  **;S FDA(F,PRVIENS,1.8)=$G(STLIC)**  **S FDA(F,PRVIENS,1.1)="S"**  **I 'NEW D Q**  **.D FILE^DIE(,"FDA") K FDA**  **.D PRVCI(PRVIEN)**  **.S FDA(52.49,EIENS,2.6)=PRVIEN D FILE^DIE(,"FDA") K FDA**  **D UPDATE^DIE(,"FDA","NEWIEN") K FDA**  **S NPIEN=$O(NEWIEN(0)),NPIEN=$G(NEWIEN(NPIEN))**  **D PRVCI(NPIEN)**  **S FDA(52.49,EIENS,2.6)=NPIEN D FILE^DIE(,"FDA") K FDA**  **Q** |

Table 45: Routine: PSOERXA3

| Routines | Activities | | | |
| --- | --- | --- | --- | --- |
| **Routine Name** | PSOERXA3 | | | |
| **Enhancement Category** | New | Modify | Delete | No Change |
| **RTM** | eRx Utilities and RPCs  Medication Prescribed | | | |
| **Related Options** |  | | | |

| Related Routines | Routines “Called By” | Routines “Called” |
| --- | --- | --- |
|  |  |  |

| Routines | Activities | | | | |
| --- | --- | --- | --- | --- | --- |
| **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) |
| --- |
| **PSOERXA3 ;ALB/BWF - eRx Utilities/RPC's ; 8/3/2016 5:14pm**  **;;7.0;OUTPATIENT PHARMACY;\*\*467\*\*;DEC 1997;Build 46**  **Q**  **MED(ERXIEN) ; medication prescribed**  **N GL,VALDT,DAYS,CIQUAL,PQUAL,PDIAG,SQUAL,DIAG,SDIAG,DIRECT,DCSTAT,DCSCO**  **DE,DDESC,DUE,CAID,CAQUAL,PSC,SREACODE,SRESCODE**  **N EFFDT,EXPDT,NOTE,PEDT,PAUTHQ,PAUTHV,PAUTHS,QCLQ,QPUC,QUSC,QTY,REFQUAL**  **,REFILLS,WRITDT,SUBS,STOPIND,CLINSIG,ACKREA**  **N F,SF,EIENS,SFDA,FDA,DIENS,DCCNT,VALDATE,DCDBCOD,DCDBCODQ,DCDEASCH,DCF**  **C,DCFSC,DCPCODE,DCPCQUAL,DCSTR,DCSTRC,DCSTSC**  **I 'ERXIEN Q**  **S EIENS=ERXIEN\_","**  **S F=52.49**  **S SF=52.4911**  **S GL=$NA(^TMP($J,"PSOERXO1","Message",0,"Body",0,"NewRx",0,"MedicationP**  **rescribed",0))**  **S VALDT=$G(@GL@("DateValidated",0,"Date",0)),VALDATE=$$CONVDTTM^PSOERXA**  **1(VALDT),FDA(52.49,EIENS,6.6)=VALDATE**  **S DAYS=$G(@GL@("DaysSupply",0)),FDA(F,EIENS,5.5)=DAYS**  **; TODO - MAKE SURE VISTA DAYS SUPPLY CAN BE THE DAYS SUPPLY PASSED IN**  **; force to numeric**  **S FDA(F,EIENS,20.2)=+$G(DAYS)**  **S DIRECT=$G(@GL@("Directions",0)),FDA(F,EIENS,7)=DIRECT**  **S DDESC=$$UP^XLFSTR($G(@GL@("DrugDescription",0))),FDA(52.49,EIENS,3.1)**  **=DDESC**  **; drugCoded**  **S DCPCODE=$G(@GL@("DrugCoded",0,"ProductCode",0)),FDA(F,EIENS,4.1)=DCPC**  **ODE**  **; TODO - convert pc qualifier**  **S DCPCQUAL=$G(@GL@("DrugCoded",0,"ProductCodeQualifier",0)),FDA(F,EIENS**  **,4.2)=DCPCQUAL**  **S DCSTR=$G(@GL@("DrugCoded",0,"Strength",0)),FDA(F,EIENS,4.3)=DCSTR**  **S DCDBCOD=$G(@GL@("DrugCoded",0,"DrugDBCode",0)),FDA(F,EIENS,4.4)=DCDBC**  **OD**  **S DCDBCODQ=$G(@GL@("DrugCoded",0,"DrugDBCodeQualifier",0)),FDA(F,EIENS,**  **4.11)=$$PRESOLV^PSOERXA1(DCDBCODQ,"DDB")**  **S DCFSC=$G(@GL@("DrugCoded",0,"FormSourceCode",0)),FDA(F,EIENS,4.5)=DCF**  **SC**  **S DCFC=$G(@GL@("DrugCoded",0,"FormCode",0)),FDA(F,EIENS,4.6)=DCFC**  **S DCSTSC=$G(@GL@("DrugCoded",0,"StrengthSourceCode",0)),FDA(F,EIENS,4.7**  **)=DCSTSC**  **S DCSTRC=$G(@GL@("DrugCoded",0,"StrengthCode",0)),FDA(F,EIENS,4.8)=DCST**  **RC**  **S DCDEASCH=$G(@GL@("DrugCoded",0,"DEASchedule")),FDA(F,EIENS,4.9)=DCDEA**  **SCH**  **; end drugCoded**  **S EFFDT=$G(@GL@("EffectiveDate",0,"Date",0)),EFFDT=$$CONVDTTM^PSOERXA1(**  **EFFDT),FDA(F,EIENS,6.3)=EFFDT**  **S EXPDT=$G(@GL@("ExpirationDate",0,"Date",0)),EXPDT=$$CONVDTTM^PSOERXA1**  **(EXPDT),FDA(F,EIENS,6.2)=EXPDT**  **S NOTE=$G(@GL@("Note",0)),FDA(F,EIENS,8)=NOTE**  **; todo - no place to store period end date as of now**  **S PEDT=$G(@GL@("PeriodEnd",0,"Date",0)),PEDT=$$CONVDTTM^PSOERXA1(PEDT),**  **FDA(F,EIENS,6.4)=PEDT**  **S WRITDT=$G(@GL@("WrittenDate",0,"Date",0)),WRITDT=$$CONVDTTM^PSOERXA1(**  **WRITDT),FDA(F,EIENS,5.9)=WRITDT**  **; dispense notes**  **S SUBS=$G(@GL@("Substitutions",0)),FDA(F,EIENS,5.8)=SUBS**  **; todo - no current field for stop indicator**  **;S STOPIND=$G(@GL@("Stop",0,"StopIndicator",0)),FDA(F,EIENS,12.7)=STOPI**  **ND**  **; prior authorization**  **S PAUTHQ=$G(@GL@("PriorAuthorization",0,"Qualifier",0)),PAUTHQ=$$PRESOL**  **V^PSOERXA1(PAUTHQ,"PAV"),FDA(F,EIENS,10.3)=PAUTHQ**  **S PAUTHV=$G(@GL@("PriorAuthorization",0,"Value",0)),FDA(F,EIENS,10.2)=P**  **AUTHV**  **S PAUTHS=$G(@GL@("PriorAuthorizationStatus",0)),FDA(F,EIENS,10.4)=PAUTH**  **S**  **; quantity**  **S QCLQ=$G(@GL@("Quantity",0,"CodeListQualifier",0)),FDA(F,EIENS,5.2)=QC**  **LQ**  **S QPUC=$G(@GL@("Quantity",0,"PotencyUnitCode",0)),FDA(F,EIENS,5.4)=QPUC**  **S QUSC=$G(@GL@("Quantity",0,"UnitSourceCode",0)),FDA(F,EIENS,5.3)=QUSC**  **S QTY=$G(@GL@("Quantity",0,"Value",0)),FDA(F,EIENS,5.1)=QTY**  **; TODO - need to look into quantity multiplier (is this the NCPDP QUANT**  **ITY MULTIPLIER?) It looks like the**  **; ncpdp quantity multiplier is for billing purposes only.**  **S FDA(F,EIENS,20.1)=QTY**  **; refills**  **S REFQUAL=$G(@GL@("Refills",0,"Qualifier",0)),FDA(F,EIENS,5.7)=REFQUAL**  **S REFILLS=$G(@GL@("Refills",0,"Value",0))**  **; PRN - refills for 1 year, set to 11 for vista**  **I '$G(REFILLS),REFQUAL="PRN" S REFILLS=11**  **S FDA(F,EIENS,5.6)=REFILLS**  **; todo - ensure vista refills can be exacly what is passed in, as long**  **as it is not greater than 11**  **S FDA(F,EIENS,20.5)=$S(REFILLS<12:REFILLS,1:11)**  **; file what we currently have**  **D FILE^DIE(,"FDA") K FDA**  **; diagnosis - primary and secondary**  **N DIAGCNT,STORCODE**  **S DIAGCNT=0**  **S DIAG=-1 F S DIAG=$O(@GL@("Diagnosis",DIAG)) Q:DIAG="" D**  **.S DIAGCNT=DIAGCNT+1**  **.S DIENS="+"\_DIAGCNT\_","\_EIENS**  **.S CIQUAL=$G(@GL@("Diagnosis",DIAG,"ClinicalInformationQualifier",0))**  **.S PQUAL=$G(@GL@("Diagnosis",DIAG,"Primary",0,"Qualifier",0))**  **.S PDIAG=$G(@GL@("Diagnosis",DIAG,"Primary",0,"Value",0))**  **.S SQUAL=$G(@GL@("Diagnosis",DIAG,"Secondary",0,"Qualifier",0))**  **.S SDIAG=$G(@GL@("Diagnosis",DIAG,"Secondary",0,"Value",0))**  **; drug coverage status codes**  **S DCCNT=0**  **S DCSTAT=-1 F S DCSTAT=$O(@GL@("DrugCoverageStatusCode",DCSTAT)) Q:DCS**  **TAT="" D**  **.S DCSCODE=$G(@GL@("DrugCoverageStatusCode",DCSTAT))**  **.S STORCODE=$$PRESOLV^PSOERXA1(DCSCODE,"DCS") Q:'$L(STORCODE)**  **.S DCCNT=DCCNT+1**  **.S DCFDA(52.4928,"+1,"\_EIENS,.01)=DCCNT**  **.S DCFDA(52.4928,"+1,"\_EIENS,.02)=STORCODE D UPDATE^DIE(,"DCFDA") K DCF**  **DA**  **; todo - do we need a counter here or sequence? can the service reason**  **code (.01) be the same value more than once?**  **; - lacking a specific unique field at this point in time.**  **S DUE=-1 F S DUE=$O(@GL@("DrugUseEvaluation",DUE)) Q:DUE="" D**  **.S CAID=$G(@GL@("DrugUseEvaluation",DUE,"CoAgent",0,"CoAgentID",0)),DUE**  **FDA(52.4916,"+1,"\_EIENS,.04)=CAID**  **.S CAQUAL=$G(@GL@("DrugUseEvaluation",DUE,"CoAgent",0,"CoAgentQualifier**  **",0)),CAQUAL=$$PRESOLV^PSOERXA1(CAQUAL,"CAQ"),DUEFDA(52.4916,"+1,"\_EIENS,.05)=CA**  **QUAL**  **.S PSC=$G(@GL@("DrugUseEvaluation",DUE,"ProfessionalServiceCode",0)),PS**  **C=$$PRESOLV^PSOERXA1(PSC,"PSC"),DUEFDA(52.4916,"+1,"\_EIENS,.02)=PSC**  **.S SREACODE=$G(@GL@("DrugUseEvaluation",DUE,"ServiceReasonCode",0)),SRE**  **ACODE=$$PRESOLV^PSOERXA1(SREACODE,"REA"),DUEFDA(52.4916,"+1,"\_EIENS,.01)=SREACOD**  **E**  **.S SRESCODE=$G(@GL@("DrugUseEvaluation",DUE,"ServiceResultCode",0)),SRE**  **SCODE=$$PRESOLV^PSOERXA1(SRESCODE,"RES"),DUEFDA(52.4916,"+1,"\_EIENS,.03)=SRESCOD**  **E**  **.S CLINSIG=$G(@GL@("DrugUseEvaluation",DUE,"ClinicalSignificanceCode",0**  **)),DUEFDA(52.4916,"+1,"\_EIENS,.06)=CLINSIG**  **.S ACKREA=$G(@GL@("DrugUseEvaluation",DUE,"AcknowledgementReason",0)),D**  **UEFDA(52.4916,"+1,"\_EIENS,1)=ACKREA**  **.D UPDATE^DIE(,"DUEFDA") K DUEFDA**  **; Structured Sig**  **N SSGL,STSIG,SIGSEQ,MSIGMOD,FTSIG,FTSIGSI,SNOMEDV,FMTV,DCIND,DDMC,DDMCQ**  **,DDMMC,DDMMCQ,DDMTEXT,DFC,DFCQ,DFTEXT,DQTY,DRNGMOD**  **N DCBMQ,DCBMV,DCCDN,DCCDUMC,DCCDUMCQ,DCCDUMT,DCDBNV,DCDBRM,DCDBUMC,DCDB**  **UMCQ,DCDBUMT,DURNV,DURTXT,DURTXTC,DURTXTCQ**  **N IIPC,IIPCQ,IIPCT,IIT,IITC,IITCQ,IIVT,IIVU,IIVUMC,IIVUMCQ,IIVUMT,IIVM,**  **MDRNV,MDRNCQ,MDRUC,MDRUT,MDRVDM,MDRVNV,MDRVUC**  **N MDRVUCQ,MDRVUT,ROAMRAM,ROAMC,ROAMCQ,ROAMT,SAMATM,SASAC,SASACQ,TATC,TA**  **TCQ,TATT,TFNV,TFUC,TFUCQ,TFUT,TINV,TIUC,TIUCQ**  **N TIUT,TMATM,TRUMC,TRUMCQ,TRUMT,TRA,TTPBC,TTPBCQ,TTPBT,TVFM,TVIM,VMVM,V**  **N,VNC,VNCQ,VQTY,VUMC,VUMCQ,VUMT,STSCNT,SAT**  **N STIENS**  **S SSGL=$NA(^TMP($J,"PSOERXO1","Message",0,"Body",0,"NewRx",0,"Medicatio**  **nPrescribed",0,"StructuredSIG"))**  **S STSCNT=0**  **S STSIG=-1 F S STSIG=$O(@SSGL@(STSIG)) Q:STSIG="" D**  **.S STSCNT=STSCNT+1**  **.S STIENS="+"\_STSCNT\_","\_EIENS**  **.S SIGSEQ=$G(@SSGL@(STSIG,"RepeatingSIG",0,"SigSequencePositionNumber",**  **0)),SFDA(SF,STIENS,.01)=SIGSEQ**  **.S MSIGMOD=$G(@SSGL@(STSIG,"RepeatingSIG",0,"MultipleSigModifier",0)),S**  **FDA(SF,STIENS,.02)=MSIGMOD**  **.S FTSIG=$G(@SSGL@(STSIG,"FreeText",0,"SigFreeText",0)),SFDA(SF,STIENS,**  **1)=FTSIG**  **.S FTSIGSI=$G(@SSGL@(STSIG,"FreeText",0,"SigFreeTextStringIndicator",0)**  **),SFDA(SF,STIENS,.05)=FTSIGSI**  **.S SNOMEDV=$G(@SSGL@(STSIG,"CodeSystem",0,"SNOMEDVersion",0)),SFDA(SF,S**  **TIENS,.03)=SNOMEDV**  **.S FMTV=$G(@SSGL@(STSIG,"CodeSystem",0,"FMTVersion",0)),SFDA(SF,STIENS,**  **.04)=FMTV**  **.; dose section**  **.S DCIND=$G(@SSGL@(STSIG,"Dose",0,"DoseCompositeIndicator",0)),SFDA(SF,**  **STIENS,2.1)=DCIND**  **.S DDMC=$G(@SSGL@(STSIG,"Dose",0,"DoseDeliveryMethodCode",0)),SFDA(SF,S**  **TIENS,2.4)=DDMC**  **.S DDMCQ=$G(@SSGL@(STSIG,"Dose",0,"DoseDeliveryMethodCodeQualifier",0))**  **,SFDA(SF,STIENS,2.3)=DDMCQ**  **.S DDMMC=$G(@SSGL@(STSIG,"Dose",0,"DoseDeliveryMethodModifierCode",0)),**  **SFDA(SF,STIENS,2.6)=DDMMC**  **.S DDMMCQ=$G(@SSGL@(STSIG,"Dose",0,"DoseDeliveryMethodModifierCodeQuali**  **fier",0)),SFDA(SF,STIENS,2.7)=DDMMCQ**  **.S DDMTEXT=$G(@SSGL@(STSIG,"Dose",0,"DoseDeliveryMethodText",0)),SFDA(S**  **F,STIENS,2.2)=DDMTEXT**  **.S DFC=$G(@SSGL@(STSIG,"Dose",0,"DoseFormCode",0)),SFDA(SF,STIENS,3.3)=**  **DFC**  **.S DFCQ=$G(@SSGL@(STSIG,"Dose",0,"DoseFormCodeQualifier",0)),SFDA(SF,ST**  **IENS,3.4)=DFCQ**  **.S DFTEXT=$G(@SSGL@(STSIG,"Dose",0,"DoseFormText",0)),SFDA(SF,STIENS,3.**  **2)=DFTEXT**  **.S DQTY=$G(@SSGL@(STSIG,"Dose",0,"DoseQuantity",0)),SFDA(SF,STIENS,3.1)**  **=DQTY**  **.S DRNGMOD=$G(@SSGL@(STSIG,"Dose",0,"DoseRangeModifier",0)),SFDA(SF,STI**  **ENS,3.5)=DRNGMOD**  **.; dose calcuation section**  **.S DCBMQ=$G(@SSGL@(STSIG,"DoseCalculation",0,"BodyMetricQualifier",0)),**  **SFDA(SF,STIENS,4.5)=DCBMQ**  **.S DCBMV=$G(@SSGL@(STSIG,"DoseCalculation",0,"BodyMetricValue",0)),SFDA**  **(SF,STIENS,4.6)=DCBMV**  **.S DCCDN=$G(@SSGL@(STSIG,"DoseCalculation",0,"CalculatedDoseNumeric",0)**  **),SFDA(SF,STIENS,4.7)=DCCDN**  **.S DCCDUMC=$G(@SSGL@(STSIG,"DoseCalculation",0,"CalculatedDoseUnitofMea**  **sureCode",0)),SFDA(SF,STIENS,4.9)=DCCDUMC**  **.S DCCDUMCQ=$G(@SSGL@(STSIG,"DoseCalculation",0,"CalculatedDoseUnitofMe**  **asureCodeQualifier",0)),SFDA(SF,STIENS,4.11)=DCCDUMCQ**  **.S DCCDUMT=$G(@SSGL@(STSIG,"DoseCalculation",0,"CalculatedDoseUnitofMea**  **sureText",0)),SFDA(SF,STIENS,4.8)=DCCDUMT**  **.S DCDBNV=$G(@SSGL@(STSIG,"DoseCalculation",0,"DosingBasisNumericValue"**  **,0)),SFDA(SF,STIENS,4.1)=DCDBNV**  **.S DCDBRM=$G(@SSGL@(STSIG,"DoseCalculation",0,"DosingBasisRangeModifier**  **",0)),SFDA(SF,STIENS,4.12)=DCDBRM**  **.S DCDBUMC=$G(@SSGL@(STSIG,"DoseCalculation",0,"DosingBasisUnitofMeasur**  **eCode",0)),SFDA(SF,STIENS,4.3)=DCDBUMC**  **.S DCDBUMCQ=$G(@SSGL@(STSIG,"DoseCalculation",0,"DosingBasisUnitofMeasu**  **reCodeQualifier",0)),SFDA(SF,STIENS,4.4)=DCDBUMCQ**  **.S DCDBUMT=$G(@SSGL@(STSIG,"DoseCalculation",0,"DosingBasisUnitofMeasur**  **eText",0)),SFDA(SF,STIENS,4.2)=DCDBUMT**  **.; duration**  **.S DURNV=$G(@SSGL@(STSIG,"Duration",0,"DurationNumericValue",0)),SFDA(S**  **F,STIENS,9.6)=DURNV**  **.S DURTXT=$G(@SSGL@(STSIG,"Duration",0,"DurationText",0)),SFDA(SF,STIEN**  **S,9.7)=DURTXT**  **.S DURTXTC=$G(@SSGL@(STSIG,"Duration",0,"DurationTextCode",0)),SFDA(SF,**  **STIENS,9.8)=DURTXTC**  **.S DURTXTCQ=$G(@SSGL@(STSIG,"Duration",0,"DurationTextCodeQualifier",0)**  **),SFDA(SF,STIENS,9.9)=DURTXTCQ**  **.; indication**  **.S IIPC=$G(@SSGL@(STSIG,"Indication",0,"IndicationPrecursorCode",0)),SF**  **DA(SF,STIENS,11.2)=IIPC**  **.S IIPCQ=$G(@SSGL@(STSIG,"Indication",0,"IndicationPrecursorCodeQualifi**  **er",0)),SFDA(SF,STIENS,11.3)=IIPCQ**  **.S IIPCT=$G(@SSGL@(STSIG,"Indication",0,"IndicationPrecursorText",0)),S**  **FDA(SF,STIENS,11.1)=IIPCT**  **.S IIT=$G(@SSGL@(STSIG,"Indication",0,"IndicationText",0)),SFDA(SF,STIE**  **NS,11.4)=IIT**  **.S IITC=$G(@SSGL@(STSIG,"Indication",0,"IndicationTextCode",0)),SFDA(SF**  **,STIENS,11.5)=IITC**  **.S IITCQ=$G(@SSGL@(STSIG,"Indication",0,"IndicationTextCodeQualifier",0**  **)),SFDA(SF,STIENS,11.6)=IITCQ**  **.S IIVT=$G(@SSGL@(STSIG,"Indication",0,"IndicationValueText",0)),SFDA(S**  **F,STIENS,12.1)=IIVT**  **.S IIVU=$G(@SSGL@(STSIG,"Indication",0,"IndicationValueUnit",0)),SFDA(S**  **F,STIENS,12.2)=IIVU**  **.S IIVUMC=$G(@SSGL@(STSIG,"Indication",0,"IndicationValueUnitofMeasureC**  **ode",0)),SFDA(SF,STIENS,12.4)=IIVUMC**  **.S IIVUMCQ=$G(@SSGL@(STSIG,"Indication",0,"IndicationValueUnitofMeasure**  **CodeQualifier",0)),SFDA(SF,STIENS,12.5)=IIVUMCQ**  **.S IIVUMT=$G(@SSGL@(STSIG,"Indication",0,"IndicationValueUnitofMeasureT**  **ext",0)),SFDA(SF,STIENS,12.3)=IIVUMT**  **.S IIVM=$G(@SSGL@(STSIG,"Indication",0,"IndicationVariableModifier",0))**  **,SFDA(SF,STIENS,12.6)=IIVM**  **.; Maximum Dose Restriction**  **.S MDRNV=$G(@SSGL@(STSIG,"MaximumDoseRestriction",0,"MaximumDoseRestric**  **tionNumericValue",0)),SFDA(SF,STIENS,10.1)=MDRNV**  **.S MDRNCQ=$G(@SSGL@(STSIG,"MaximumDoseRestriction",0,"MaximumDoseRestri**  **ctionCodeQualifier",0)),SFDA(SF,STIENS,10.3)=MDRNCQ**  **.S MDRUC=$G(@SSGL@(STSIG,"MaximumDoseRestriction",0,"MaximumDoseRestric**  **tionUnitsCode",0)),SFDA(SF,STIENS,10.4)=MDRUC**  **.S MDRUT=$G(@SSGL@(STSIG,"MaximumDoseRestriction",0,"MaximumDoseRestric**  **tionUnitsText",0)),SFDA(SF,STIENS,10.2)=MDRUT**  **.S MDRVDM=$G(@SSGL@(STSIG,"MaximumDoseRestriction",0,"MaximumDoseRestri**  **ctionVariableDurationModifier",0)),SFDA(SF,STIENS,10.9)=MDRVDM**  **.S MDRVNV=$G(@SSGL@(STSIG,"MaximumDoseRestriction",0,"MaximumDoseRestri**  **ctionVariableNumericValue",0)),SFDA(SF,STIENS,10.5)=MDRVNV**  **.S MDRVUC=$G(@SSGL@(STSIG,"MaximumDoseRestriction",0,"MaximumDoseRestri**  **ctionVariableUnitsCode",0)),SFDA(SF,STIENS,10.7)=MDRVUC**  **.S MDRVUCQ=$G(@SSGL@(STSIG,"MaximumDoseRestriction",0,"MaximumDoseRestr**  **ictionVariableUnitsCodeQualifier",0)),SFDA(SF,STIENS,10.8)=MDRVUCQ**  **.S MDRVUT=$G(@SSGL@(STSIG,"MaximumDoseRestriction",0,"MaximumDoseRestri**  **ctionVariableUnitsText",0)),SFDA(SF,STIENS,10.6)=MDRVUT**  **.; Route of Administration**  **.S ROAMRAM=$G(@SSGL@(STSIG,"RouteOfAdministration",0,"MultipleRouteofAd**  **ministrationModifier",0)),SFDA(SF,STIENS,6.4)=ROAMRAM**  **.S ROAMC=$G(@SSGL@(STSIG,"RouteOfAdministration",0,"RouteofAdministrati**  **onCode",0)),SFDA(SF,STIENS,6.2)=ROAMC**  **.S ROAMCQ=$G(@SSGL@(STSIG,"RouteOfAdministration",0,"RouteofAdministrat**  **ionCodeQualifier",0)),SFDA(SF,STIENS,6.3)=ROAMCQ**  **.S ROAMT=$G(@SSGL@(STSIG,"RouteOfAdministration",0,"RouteofAdministrati**  **onText",0)),SFDA(SF,STIENS,6.1)=ROAMT**  **.; site of Administration**  **.S SAMATM=$G(@SSGL@(STSIG,"SiteofAdministration",0,"MultipleAdministrat**  **ionTimingModifier",0)),SFDA(SF,STIENS,6.8)=SAMATM**  **.S SASAC=$G(@SSGL@(STSIG,"SiteofAdministration",0,"SiteofAdministration**  **Code",0)),SFDA(SF,STIENS,6.6)=SASAC**  **.S SASACQ=$G(@SSGL@(STSIG,"SiteofAdministration",0,"SiteofAdministratio**  **nCodeQualifier",0)),SFDA(SF,STIENS,6.7)=SASACQ**  **.S SAT=$G(@SSGL@(STSIG,"SiteofAdministration",0,"SiteofAdministrationTe**  **xt",0)),SFDA(SF,STIENS,6.5)=SAT**  **.; Timing**  **.S TATC=$G(@SSGL@(STSIG,"Timing",0,"AdministrationTimingCode",0)),SFDA(**  **SF,STIENS,7.2)=TATC**  **.S TATCQ=$G(@SSGL@(STSIG,"Timing",0,"AdministrationTimingCodeQualifier"**  **,0)),SFDA(SF,STIENS,7.3)=TATCQ**  **.S TATT=$G(@SSGL@(STSIG,"Timing",0,"AdministrationTimingText",0)),SFDA(**  **SF,STIENS,7.1)=TATT**  **.S TFNV=$G(@SSGL@(STSIG,"Timing",0,"FrequencyNumericValue",0)),SFDA(SF,**  **STIENS,8.4)=TFNV**  **.S TFUC=$G(@SSGL@(STSIG,"Timing",0,"FrequencyUnitsCode",0)),SFDA(SF,STI**  **ENS,8.6)=TFUC**  **.S TFUCQ=$G(@SSGL@(STSIG,"Timing",0,"FrequencyUnitsCodeQualifier",0)),S**  **FDA(SF,STIENS,8.7)=TFUCQ**  **.S TFUT=$G(@SSGL@(STSIG,"Timing",0,"FrequencyUnitsText",0)),SFDA(SF,STI**  **ENS,8.5)=TFUT**  **.S TINV=$G(@SSGL@(STSIG,"Timing",0,"IntervalNumericValue",0)),SFDA(SF,S**  **TIENS,9.1)=TINV**  **.S TIUC=$G(@SSGL@(STSIG,"Timing",0,"IntervalUnitsCode",0)),SFDA(SF,STIE**  **NS,9.3)=TIUC**  **.S TIUCQ=$G(@SSGL@(STSIG,"Timing",0,"IntervalUnitsCodeQualifier",0)),SF**  **DA(SF,STIENS,9.4)=TIUCQ**  **.S TIUT=$G(@SSGL@(STSIG,"Timing",0,"IntervalUnitsText",0)),SFDA(SF,STIE**  **NS,9.2)=TIUT**  **.S TMATM=$G(@SSGL@(STSIG,"Timing",0,"MultipleAdministrationTimingModifi**  **er",0)),SFDA(SF,STIENS,7.4)=TMATM**  **.S TRUMC=$G(@SSGL@(STSIG,"Timing",0,"RateUnitofMeasureCode",0)),SFDA(SF**  **,STIENS,7.7)=TRUMC**  **.S TRUMCQ=$G(@SSGL@(STSIG,"Timing",0,"RateUnitofMeasureCodeQualifier",0**  **)),SFDA(SF,STIENS,7.8)=TRUMCQ**  **.S TRUMT=$G(@SSGL@(STSIG,"Timing",0,"RateUnitofMeasureText",0)),SFDA(SF**  **,STIENS,7.6)=TRUMT**  **.S TRA=$G(@SSGL@(STSIG,"Timing",0,"RateofAdministration",0)),SFDA(SF,ST**  **IENS,7.5)=TRA**  **.S TTPBC=$G(@SSGL@(STSIG,"Timing",0,"TimePeriodBasisCode",0)),SFDA(SF,S**  **TIENS,8.2)=TTPBC**  **.S TTPBCQ=$G(@SSGL@(STSIG,"Timing",0,"TimePeriodBasisCodeQualifier",0))**  **,SFDA(SF,STIENS,8.3)=TTPBCQ**  **.S TTPBT=$G(@SSGL@(STSIG,"Timing",0,"TimePeriodBasisText",0)),SFDA(SF,S**  **TIENS,8.1)=TTPBT**  **.S TVFM=$G(@SSGL@(STSIG,"Timing",0,"VariableFrequencyModifier",0)),SFDA**  **(SF,STIENS,8.8)=TVFM**  **.S TVIM=$G(@SSGL@(STSIG,"Timing",0,"VariableIntervalModifier",0)),SFDA(**  **SF,STIENS,9.5)=TVIM**  **.; Vehicle**  **.S VMVM=$G(@SSGL@(STSIG,"Vehicle",0,"MultipleVehicleModifier",0)),SFDA(**  **SF,STIENS,5.8)=VMVM**  **.S VN=$G(@SSGL@(STSIG,"Vehicle",0,"VehicleName",0)),SFDA(SF,STIENS,5.1)**  **=VN**  **.S VNC=$G(@SSGL@(STSIG,"Vehicle",0,"VehicleNameCode",0)),SFDA(SF,STIENS**  **,5.2)=VNC**  **.S VNCQ=$G(@SSGL@(STSIG,"Vehicle",0,"VehicleNameCodeQualifier",0)),SFDA**  **(SF,STIENS,5.3)=VNCQ**  **.S VQTY=$G(@SSGL@(STSIG,"Vehicle",0,"VehicleQuantity",0)),SFDA(SF,STIEN**  **S,5.4)=VQTY**  **.S VUMC=$G(@SSGL@(STSIG,"Vehicle",0,"VehicleUnitofMeasureCode",0)),SFDA**  **(SF,STIENS,5.6)=VUMC**  **.S VUMCQ=$G(@SSGL@(STSIG,"Vehicle",0,"VehicleUnitofMeasureCodeQualifier**  **",0)),SFDA(SF,STIENS,5.7)=VUMCQ**  **.S VUMT=$G(@SSGL@(STSIG,"Vehicle",0,"VehicleUnitofMeasureText",0)),SFDA**  **(SF,STIENS,5.5)=VUMT**  **.D UPDATE^DIE(,"SFDA") K SFDA**  **Q** |

Table 46: Routine: PSOERXD1

| Routines | Activities | | | |
| --- | --- | --- | --- | --- |
| **Routine Name** | PSOERXD1 | | | |
| **Enhancement Category** | New | Modify | Delete | No Change |
| **RTM** | eRx Drug Display & Actions  Drug Validation Screen | | | |
| **Related Options** |  | | | |

| Related Routines | Routines “Called By” | Routines “Called” |
| --- | --- | --- |
|  |  |  |

| Routines | Activities | | | | |
| --- | --- | --- | --- | --- | --- |
| **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) |
| --- |
| **PSOERXD1 ;ALB/BWF - eRx Drug display/actions ; 8/3/2016 5:14pm**  **;;7.0;OUTPATIENT PHARMACY;\*\*467\*\*;DEC 1997;Build 46**  **;**  **EN ; -- main entry point for PSO ERX HOLDING QUEUE**  **D EN^VALM("PSO ERX DRUG VALIDATION")**  **Q**  **;**  **HDR ; -- header code**  **S VALMHDR(1)="eRx Patient: "\_$$GET1^DIQ(52.49,PSOIEN,.04,"E")**  **S VALMHDR(2)="eRx Reference #: "\_$$GET1^DIQ(52.49,PSOIEN,25,"E")**  **I $G(VALMBCK)="R" D INIT**  **Q**  **;**  **INIT ;**  **Q:'$G(PSOIEN)**  **N LINE,ERXDRG,LINETXT,ERXQTY,ERXQQ,ERXDF,ERXSC,ERXRFLS,ERXDS,ERXDT,ERXD**  **W,ERXDSUB,VADRG,NXTLINE,ORDITEM**  **N ERXCOMM,ERXDRUG,ERXSIG,FILLDT,ISSDT,MAILWIN,PATSTAT,SGLOOP,SIGDATA,SI**  **GLINE,SIGLOOP,VACLIN,VACOPIES,VADAYS**  **N VADOSE,VADRGIEN,VAENBY,VAENDATE,VAPINST,VAPROV,VAPUS,VAQTY,VAREF,VARE**  **MARK,VAROUTE,VASCHED,VASIG,VAVERB**  **N PSODAT,PSONEW,SIGDATA,SGLOOP,PRVCARY,PCRFST,PCLOOP,INSTARY,IFRST,INLO**  **OP,SEPLN,PATINST,PIARY,PILOOP**  **N SFIRST,SIGARY,PSOIENS,PSODAT2,PFRST,FN,FSSIG,INST,PATIEN,PCFRST,PRVCO**  **MM,PSODFN,SLOOP**  **S LINE=0**  **S PSOIENS=PSOIEN\_","**  **S FN=52.49**  **;D GETS^DIQ(52.49,PSOIEN,".05;2.3;3.1;5.1;5.2;4.6;4.8;5.6;5.5;.03;5.9;5**  **.8;3.2;7;8;20.1;20.2;20.3;20.6","IE","PSODAT")**  **D GETS^DIQ(52.49,PSOIEN,"\*\*","IE","PSODAT")**  **S PATIEN=$G(PSODAT(FN,PSOIENS,.05,"I"))**  **S ERXDRUG=$G(PSODAT(FN,PSOIENS,3.1,"E"))**  **; quantity**  **S ERXQTY=$G(PSODAT(FN,PSOIENS,5.1,"E"))**  **; quantity qualifier**  **S ERXQQ=$G(PSODAT(FN,PSOIENS,5.2,"E"))**  **;S ERXQQ=$$GET1^DIQ(52.49,PSOIEN,5.2,"E")**  **; drug form code**  **S ERXDF=$G(PSODAT(FN,PSOIENS,4.6,"E"))**  **; strength code**  **S ERXSC=$G(PSODAT(FN,PSOIENS,4.8,"E"))**  **; refills**  **S ERXRFLS=$G(PSODAT(FN,PSOIENS,5.6,"E"))**  **; days supply**  **S ERXDS=$G(PSODAT(FN,PSOIENS,5.5,"E"))**  **; erx date**  **S ERXDT=$G(PSODAT(FN,PSOIENS,.03,"E"))**  **; written date**  **S ERXDW=$G(PSODAT(FN,PSOIENS,5.9,"E"))**  **; substitions**  **S ERXDSUB=$G(PSODAT(FN,PSOIENS,5.8,"E"))**  **; va (matched) drug**  **S VADRG=$G(PSODAT(FN,PSOIENS,3.2,"E"))**  **I VADRG']"" S VADRG="NOT LINKED"**  **S LINE=LINE+1 D SET^VALM10(LINE,"eRx Drug: "\_$E(ERXDRUG,1,60))**  **S LINE=LINE+1**  **D ADDITEM^PSOERX1A(.LINETXT,"Qty: ",ERXQTY,1,26)**  **D ADDITEM^PSOERX1A(.LINETXT,"Days Supply: ",ERXDS,28,26)**  **D ADDITEM^PSOERX1A(.LINETXT,"Date Written: ",$P(ERXDW,"@"),54,26)**  **D SET^VALM10(LINE,LINETXT) S LINETXT=""**  **S LINE=LINE+1**  **D ADDITEM^PSOERX1A(.LINETXT,"Qty Qualifier: ",ERXQQ,1,26)**  **D ADDITEM^PSOERX1A(.LINETXT,"Drug Form: ",ERXDF,28,26)**  **D ADDITEM^PSOERX1A(.LINETXT,"Strength: ",ERXSC,54,26)**  **D SET^VALM10(LINE,LINETXT) S LINETXT=""**  **S LINE=LINE+1**  **D ADDITEM^PSOERX1A(.LINETXT,"Refills: ",ERXRFLS,1,15)**  **D ADDITEM^PSOERX1A(.LINETXT,"Do not sub: ",ERXDSUB,28,50)**  **D SET^VALM10(LINE,LINETXT) S LINETXT=""**  **S SIGDATA=""**  **S SIGLOOP=0 F S SIGLOOP=$O(^PS(52.49,PSOIEN,11,SIGLOOP)) Q:'SIGLOOP D**  **.I '$L(SIGDATA) S SIGDATA=$$GET1^DIQ(52.4911,SIGLOOP\_","\_PSOIEN,1,"E")**  **Q**  **.S SIGDATA=SIGDATA\_" "\_SIGDATA**  **D TXT2ARY(.SIGARY,SIGDATA,,57)**  **S SFIRST=$O(SIGARY(0))**  **S SGLOOP=0 F S SGLOOP=$O(SIGARY(SGLOOP)) Q:'SGLOOP D**  **.S LINE=LINE+1 D SET^VALM10(LINE,$S(SGLOOP=SFIRST:"eRx Sig: ",1:"**  **")\_$G(SIGARY(SGLOOP)))**  **S LINE=LINE+1 D SET^VALM10(LINE,"")**  **S $P(SEPLN,"-",80)="-" D SET^VALM10(LINE,SEPLN)**  **; vista drug information**  **S VADRGIEN=$G(PSODAT(52.49,PSOIENS,3.2,"I"))**  **S VADRG=$S(VADRGIEN:$$GET1^DIQ(50,VADRGIEN,.01,"E"),1:"NOT MATCHED")**  **S LINE=LINE+1 D SET^VALM10(LINE," (1) Vista Drug: "\_VADRG)**  **; TODO - may not need clinic or remarks.**  **S (VACLIN,VAREMARK)=""**  **S VAPROV=$G(PSODAT(52.49,PSOIENS,2.3,"E"))**  **S INST=$G(PSODAT(52.49,PSOIENS,7,"E"))**  **S PRVCOMM=$G(PSODAT(52.49,PSOIENS,8,"E"))**  **S VADAYS=$G(PSODAT(52.49,PSOIENS,20.2,"E"))**  **S VAQTY=$G(PSODAT(52.49,PSOIENS,20.1,"E"))**  **S VAVERB=$G(PSODAT(52.49,PSOIENS,20.3,"E"))**  **S PATSTAT=$$GET1^DIQ(55,PATIEN,3,"E")**  **S MAILWIN=$G(PSODAT(52.49,PSOIENS,20.4,"E"))**  **S VAREF=$G(PSODAT(52.49,PSOIENS,20.5,"E"))**  **S PATINST=$G(PSODAT(52.49,PSOIENS,27,"E"))**  **S VACLIN=$G(PSODAT(52.49,PSOIENS,20.6,"E"))**  **D TXT2ARY(.PIARY,PATINST,,55)**  **D DOSE**  **; pat instructions built from dosage multiple**  **S PFRST="",PFRST=$O(PIARY(0))**  **S PILOOP=0 F S PILOOP=$O(PIARY(PILOOP)) Q:'PILOOP D**  **.S LINE=LINE+1 D SET^VALM10(LINE,$S(PFRST=PILOOP:" (3) Pat Instructions**  **: "\_$G(PIARY(PILOOP)),1:" "\_$G(PIARY(PILOOP))))**  **; provider comments come from the 'notes' field in 52.49 (#8)**  **D TXT2ARY(.PRVCARY,PRVCOMM,,57)**  **S PCFRST=$O(PRVCARY(0))**  **S PCLOOP=0 F S PCLOOP=$O(PRVCARY(PCLOOP)) Q:'PCLOOP D**  **.S LINE=LINE+1 D SET^VALM10(LINE,$S(PCFRST=PCLOOP:" (4) Provider Commen**  **ts: ",1:" ")\_PRVCARY(PCLOOP))**  **; instructions come from the 'DIRECTIONS' field in 52.49**  **; if the instructions are longer than 57, convert to array for display**  **I $L(INST)>57 D**  **.D TXT2ARY(.INSTARY,INST,,57)**  **S IFRST=$O(INSTARY(0))**  **S INLOOP=0 F S INLOOP=$O(INSTARY(INLOOP)) Q:'INLOOP D**  **.S LINE=LINE+1 D SET^VALM10(LINE,$S(IFRST=INLOOP:" Instruction**  **s: ",1:" ")\_$G(INSTARY(INLOOP)))**  **S FSSIG=$O(^PS(52.49,PSOIEN,"SIG",0))**  **S SLOOP=0 F S SLOOP=$O(^PS(52.49,PSOIEN,"SIG",SLOOP)) Q:'SLOOP D**  **.S VASIG=$G(^PS(52.49,PSOIEN,"SIG",SLOOP,0))**  **.S LINE=LINE+1 D SET^VALM10(LINE,$S(SLOOP=FSSIG:" SIG**  **: "\_VASIG,1:" "\_VASIG))**  **S LINETXT=""**  **S LINE=LINE+1 D SET^VALM10(LINE," (5) Patient Status: "\_PATSTAT)**  **S LINE=LINE+1**  **D ADDITEM^PSOERX1A(.LINETXT," (6) Days Supply: ",VADAYS,1,25)**  **D ADDITEM^PSOERX1A(.LINETXT,"(7) QTY: ",VAQTY,45,30)**  **D SET^VALM10(LINE,LINETXT) S LINETXT=""**  **S LINE=LINE+1 D SET^VALM10(LINE," Provider ordered '"\_$S(VAREF:VAREF**  **,1:0)\_"' refills")**  **S LINE=LINE+1**  **D ADDITEM^PSOERX1A(.LINETXT," (8) # of Refills: ",VAREF,1,30)**  **; routing defaults to 'M'ail**  **D ADDITEM^PSOERX1A(.LINETXT,"(9) Routing: ",MAILWIN,45,25)**  **D SET^VALM10(LINE,LINETXT) S LINETXT=""**  **; TODO - CLINIC and remarks SHOULD PROBABLY BE REMOVED - doesn't seem t**  **o be a need for these from an erx perspective**  **S LINE=LINE+1 D SET^VALM10(LINE," (10) Clinic: "\_VACLIN)**  **S LINE=LINE+1 D SET^VALM10(LINE," (11) Provider: "\_VAPROV)**  **S LINE=LINE+1 D SET^VALM10(LINE," (12) Remarks: "\_VAREMARK)**  **S LINE=LINE+1**  **;D ADDITEM^PSOERX1A(.LINETXT,"Entry by: ",VAENBY,4,40)**  **;D ADDITEM^PSOERX1A(.LINETXT,"Entry Date: ",VAENDATE,45,30)**  **;D SET^VALM10(LINE,LINETXT) S LINETXT=""**  **; todo - comments may need to be in a loop. DO WE NEED ERX COMMENTS? TH**  **EY DO NOT GET SENT TO 52.41**  **;S LINE=LINE+1 D SET^VALM10(LINE,"")**  **;S ERXCOMM=""**  **;S LINE=LINE+1 D SET^VALM10(LINE,"eRx Comments: "\_ERXCOMM)**  **S VALMCNT=LINE**  **S EDTYP="D"**  **K PSODAT,PSONEW,DOENT,TDUR**  **Q**  **;SIG ;**  **;N D**  **;S SIG=0,PSOFINFL=1 F S SIG=$O(^PS(52.41,ORD,"SIG",SIG)) Q:'SIG D**  **;.S (MIG,SIG(SIG))=^PS(52.41,ORD,"SIG",SIG,0)**  **;.F SG=1:1:$L(MIG," ") S:$L(^TMP("PSOPO",$J,IEN,0)\_" "\_$P(MIG," ",SG))>**  **80 IEN=IEN+1,$P(^TMP("PSOPO",$J,IEN,0)," ",20)=" " S ^TMP("PSOPO",$J,IEN,0)=$G(^**  **TMP("PSOPO",$J,IEN,0))\_" "\_$P(MIG," ",SG) D**  **;..I $E(^TMP("PSOPO",$J,IEN,0),$L(^TMP("PSOPO",$J,IEN,0)))=" " S ^TMP("**  **PSOPO",$J,IEN,0)=$E(^TMP("PSOPO",$J,IEN,0),1,($L(^TMP("PSOPO",$J,IEN,0))-1))**  **;S:$O(SIG(0)) SIGOK=1 K MIG**  **;F D=0:0 S D=$O(^PS(52.41,ORD,"INS1",D)) Q:'D S PSONEW("INS",D)=^PS(52**  **.41,ORD,"INS1",D,0)**  **;Q**  **;**  **HELP ; -- help code**  **S X="?" D DISP^XQORM1 W !!**  **Q**  **;**  **EXIT ; -- exit code**  **K @VALMAR,EDTYP**  **Q**  **;**  **EXPND ; -- expand code**  **Q**  **; ARY - array to store the output (pass by reference)**  **; TEXT - the text to convert into the array format**  **; DELIM - delimiter for text (default is space)**  **; MAXLEN - maximum lenth of each array items text, defaults to 80**  **TXT2ARY(ARY,TEXT,DELIM,MAXLEN) ;**  **N WORD,I,LCNT,LINETXT,S**  **S S=$S($D(DELIM):DELIM,1:" ")**  **I '$G(MAXLEN) S MAXLEN=80**  **S LCNT=1,LINETXT=""**  **F I=1:1:$L(TEXT,S) D**  **.S WORD=$P(TEXT,S,I)**  **.I $L(LINETXT)+$L(S)+$L(WORD)>MAXLEN D Q**  **..S ARY(LCNT)=LINETXT**  **..S LCNT=LCNT+1,LINETXT=WORD**  **.I '$L(LINETXT) S LINETXT=WORD Q**  **.S LINETXT=LINETXT\_S\_WORD**  **; if there is information left, set it into the array**  **I $L(LINETXT) S ARY(LCNT)=$G(LINETXT)**  **Q**  **DOSE ;displays dosing info for pending orders. called from psoorfi1**  **K II,UNITS S DS=1**  **;I '$O(^PS(52.49,PSOIEN,21,I,0)) S IEN=IEN+1,^TMP("PSOPO",$J,IEN,0)=" (**  **3) \*Dosage:" G DOSEX**  **I '$O(^PS(52.49,PSOIEN,21,0)) S LINE=LINE+1,LINETXT="" D ADDITEM^PSOERX**  **1A(.LINETXT," (2) \*Dosage:",,1,30) D SET^VALM10(LINE,LINETXT) S LINETXT**  **="" G DOSEX**  **F I=0:0 S I=$O(^PS(52.49,PSOIEN,21,I)) Q:'I S DOSE=$G(^PS(52.49,PSOIEN**  **,21,I,1)),DOSE1=$G(^(2)) D D DOSE1**  **.S II=$G(II)+1 K PSONEW("UNITS",II)**  **.S PSONEW("DOSE",II)=$P(DOSE1,"^"),PSONEW("DOSE ORDERED",II)=$P(DOSE1,"**  **^",2),PSONEW("UNITS",II)=$P(DOSE,"^",9),PSONEW("NOUN",II)=$P(DOSE,"^",5)**  **.S:$P(DOSE,"^",9) UNITS=$P(^PS(50.607,$P(DOSE,"^",9),0),"^")**  **.S PSONEW("VERB",II)=$P(DOSE,"^",10),PSONEW("ROUTE",II)=$P(DOSE,"^",8)**  **.S ROUTE="" S:$P(DOSE,"^",8) ROUTE=$P(^PS(51.2,$P(DOSE,"^",8),0),"^")**  **.S PSONEW("SCHEDULE",II)=$P(DOSE,"^"),PSONEW("DURATION",II)=$P(DOSE,"^"**  **,2)**  **.S DOENT=$G(DOENT)+1 I $P(DOSE,"^",6)]"" S PSONEW("CONJUNCTION",II)=$S(**  **$P(DOSE,"^",6)="S":"T",$P(DOSE,"^",6)="X":"X",1:"A")**  **.;I 'PSONEW("DOSE ORDERED",II),$G(PSONEW("VERB",II))]"" S IEN=IEN+1,^TM**  **P("PSOPO",$J,IEN,0)=" Verb: "\_$G(PSONEW("VERB",II))**  **.I 'PSONEW("DOSE ORDERED",II),$G(PSONEW("VERB",II))]"" D**  **..S LINETXT="" S LINE=LINE+1 D ADDITEM^PSOERX1A(.LINETXT,"**  **Verb: ",$G(PSONEW("VERB",II)),1,40)**  **..D SET^VALM10(LINE,LINETXT) S LINETXT=""**  **.S:$G(DS) LINE=LINE+1 D ADDITEM^PSOERX1A(.LINETXT," (2)","",1,5)**  **DOSEX S PSONEW("ENT")=+$G(II) K DOSE,DOSE1,II,I,UNITS,ROUTE,DG**  **Q**  **DOSE1 ;**  **;I $G(DS)=1 S ^TMP("PSOPO",$J,IEN,0)=^TMP("PSOPO",$J,IEN,0)\_" \*D**  **osage:" D FMD^PSOORFI3 G DU**  **I $G(DS)=1 D ADDITEM^PSOERX1A(.LINETXT," \*Dosage:","",4,30) D FM**  **D G DU**  **;S IEN=IEN+1,^TMP("PSOPO",$J,IEN,0)=" \*Dosage:" D FMD^PSOORF**  **I3**  **S LINE=LINE+1 D ADDITEM^PSOERX1A(.LINETXT," \*Dosage:","",1,3**  **0) D FMD**  **DU ;**  **;I 'PSONEW("DOSE ORDERED",I),$P($G(^PS(55,PSODFN,"LAN")),"^") S IEN=IEN**  **+1,^TMP("PSOPO",$J,IEN,0)=" \*Oth. Lang. Dosage: "\_$G(PSONEW("ODOSE",I))**  **; TODO - PSODFN IS NOT DEFINED\*!!!!!!**  **S PSODFN=1**  **I 'PSONEW("DOSE ORDERED",I),$P($G(^PS(55,PSODFN,"LAN")),"^") D**  **.S LINE=LINE+1,LINETXT="" D ADDITEM^PSOERX1A(.LINETXT," Oth. Lang. Dos**  **age: ",$G(PSONEW("ODOSE",I)),1,50) D SET^VALM10(LINE,LINETXT) S LINETXT=""**  **I PSONEW("DOSE ORDERED",II),$G(PSONEW("VERB",II))]"" D**  **.S LINE=LINE+1 D ADDITEM^PSOERX1A(.LINETXT," Verb: ",$G(P**  **SONEW("VERB",II)),1,50) D SET^VALM10(LINE,LINETXT) S LINETXT=""**  **.S LINE=LINE+1 D ADDITEM^PSOERX1A(.LINETXT," Dispense Units: ",$S($**  **E(PSONEW("DOSE ORDERED",II),1)=".":"0",1:"")\_PSONEW("DOSE ORDERED",II),1,50) D S**  **ET^VALM10(LINE,LINETXT) S LINETXT=""**  **.;S IEN=IEN+1,^TMP("PSOPO",$J,IEN,0)=" Verb: "\_$G(PSONEW(**  **"VERB",II))**  **.;S IEN=IEN+1,^TMP("PSOPO",$J,IEN,0)=" Dispense Units: "\_$S($E(PSON**  **EW("DOSE ORDERED",II),1)=".":"0",1:"")\_PSONEW("DOSE ORDERED",II)**  **;I PSONEW("NOUN",II)]"" S IEN=IEN+1,^TMP("PSOPO",$J,IEN,0)="**  **Noun: "\_PSONEW("NOUN",II)**  **I PSONEW("NOUN",II)]"" D**  **.S LINE=LINE+1 D ADDITEM^PSOERX1A(.LINETXT," Noun: ",PSON**  **EW("NOUN",II),1,50) D SET^VALM10(LINE,LINETXT) S LINETXT=""**  **;I $G(ROUTE)]"" S IEN=IEN+1,^TMP("PSOPO",$J,IEN,0)=" \*Route**  **: "\_$G(ROUTE)**  **I $G(ROUTE)]"" D**  **.S LINE=LINE+1 D ADDITEM^PSOERX1A(.LINETXT," \*Route: ",$G(R**  **OUTE),1,50) D SET^VALM10(LINE,LINETXT) S LINETXT=""**  **;S IEN=IEN+1,^TMP("PSOPO",$J,IEN,0)=" \*Schedule: "\_PSONEW("SCH**  **EDULE",II)**  **S LINE=LINE+1 D ADDITEM^PSOERX1A(.LINETXT," \*Schedule: ",PSONE**  **W("SCHEDULE",II),1,50) D SET^VALM10(LINE,LINETXT) S LINETXT=""**  **I $G(PSONEW("DURATION",II))]"" D**  **.S PSONEW("DURATION",II)=$S($E(PSONEW("DURATION",II),1)'?.N:$E(PSONEW("**  **DURATION",II),2,99)\_$E(PSONEW("DURATION",II),1),1:PSONEW("DURATION",II))**  **.;S IEN=IEN+1,^TMP("PSOPO",$J,IEN,0)=" \*Duration: "\_PSONEW("DU**  **RATION",II)\_" ("\_$S(PSONEW("DURATION",II)["M":"MINUTES",PSONEW("DURATION",II)["H**  **":"HOURS",PSONEW("DURATION",II)["L":"MONTHS",PSONEW("DURATION",II)["W":"WEEKS",1**  **:"DAYS")\_")"**  **.S TDUR=PSONEW("DURATION",II)\_" ("\_$S(PSONEW("DURATION",II)["M":"MINUTE**  **S",PSONEW("DURATION",II)["H":"HOURS",PSONEW("DURATION",II)["L":"MONTHS",PSONEW("**  **DURATION",II)["W":"WEEKS",1:"DAYS")\_")"**  **.S LINE=LINE+1 D ADDITEM^PSOERX1A(.LINETXT," \*Duration: ",TDUR**  **,1,50) D SET^VALM10(LINE,LINETXT) S LINETXT=""**  **;I $G(PSONEW("CONJUNCTION",II))]"" S IEN=IEN+1,^TMP("PSOPO",$J,IEN,0)="**  **\*Conjunction: "\_$S(PSONEW("CONJUNCTION",II)="T":"THEN",PSONEW("CONJUNCTIO**  **N",II)="X":"EXCEPT",1:"AND")**  **I $G(PSONEW("CONJUNCTION",II))]"" D**  **.S LINE=LINE+1 D ADDITEM^PSOERX1A(.LINETXT," \*Conjunction: ",$S(P**  **SONEW("CONJUNCTION",II)="T":"THEN",PSONEW("CONJUNCTION",II)="X":"EXCEPT",1:"AND"**  **),1,50) D SET^VALM10(LINE,LINETXT) S LINETXT=""**  **Q**  **;DOSE2 ;displays pending order after edits. called from psoornew**  **;I '$O(PSONEW("DOSE",0))!($O(PSONEW("DOSE",0))="") S IEN=IEN+1,^TMP("PS**  **OPO",$J,IEN,0)=" (3) \*Dosage:" Q**  **;S DS=1**  **;F I=1:1:PSONEW("ENT") Q:'I D D DOSE3 K COJ**  **;.S:$G(PSONEW("UNITS",I))]"" UNITS=$P(^PS(50.607,PSONEW("UNITS",I),0),"**  **^")**  **;.I $G(PSONEW("ROUTE",I))]"",$G(^PS(51.2,PSONEW("ROUTE",I),0))]"" S ROU**  **TE=$P(^PS(51.2,PSONEW("ROUTE",I),0),"^")**  **;.S DUR=$G(PSONEW("DURATION",I)) S:$G(PSONEW("CONJUNCTION",I))]"" COJ=P**  **SONEW("CONJUNCTION",I)**  **;.S NOUN=$G(PSONEW("NOUN",I)),VERB=$G(PSONEW("VERB",I))**  **;.I '$G(PSONEW("DOSE ORDERED",I)),$G(PSONEW("VERB",I))]"" S IEN=IEN+1,^**  **TMP("PSOPO",$J,IEN,0)=" Verb: "\_$G(PSONEW("VERB",I))**  **;.S:$G(DS) IEN=IEN+1,^TMP("PSOPO",$J,IEN,0)=" (3)"**  **;K I,UNITS,ROUTE,DUR,COJ,VERB,NOUN,DG**  **;Q**  **;DOSE3 I $G(DS)=1 S II=I,^TMP("PSOPO",$J,IEN,0)=^TMP("PSOPO",$J,IEN,0)\_**  **" \*Dosage:" D FMD^PSOORFI3 G DO**  **;S II=I,IEN=IEN+1,^TMP("PSOPO",$J,IEN,0)=" \*Dosage:" D FMD^P**  **SOORFI3**  **;DO I '$G(PSONEW("DOSE ORDERED",I)),$P($G(^PS(55,PSODFN,"LAN")),"^") S**  **IEN=IEN+1,^TMP("PSOPO",$J,IEN,0)=" \*Oth. Lang. Dosage: "\_$G(PSONEW("ODOSE",I))**  **;I $G(PSONEW("DOSE ORDERED",I)),$G(PSONEW("VERB",I))]"" S IEN=IEN+1,^TM**  **P("PSOPO",$J,IEN,0)=" Verb: "\_$G(PSONEW("VERB",I))**  **;I $G(PSONEW("DOSE ORDERED",I)) S IEN=IEN+1,^TMP("PSOPO",$J,IEN,0)="**  **Dispense Units: "\_$S($E(PSONEW("DOSE ORDERED",I),1)=".":"0",1:"")\_PSONEW("DOSE**  **ORDERED",I)**  **;I $G(PSONEW("NOUN",I))]"" S IEN=IEN+1,^TMP("PSOPO",$J,IEN,0)="**  **NOUN: "\_PSONEW("NOUN",I)**  **;I $G(ROUTE)]"" S IEN=IEN+1,^TMP("PSOPO",$J,IEN,0)=" \*Route**  **: "\_$G(ROUTE)**  **;S IEN=IEN+1,^TMP("PSOPO",$J,IEN,0)=" \*Schedule: "\_PSONEW("SCH**  **EDULE",I)**  **;I $G(PSONEW("DURATION",I))]"" D**  **;.S PSONEW("DURATION",I)=$S($E(PSONEW("DURATION",I),1)'?.N:$E(PSONEW("D**  **URATION",I),2,99)\_$E(PSONEW("DURATION",I),1),1:PSONEW("DURATION",I))**  **;.S IEN=IEN+1,^TMP("PSOPO",$J,IEN,0)=" \*Duration: "\_PSONEW("DU**  **RATION",I)\_" ("\_$S(PSONEW("DURATION",I)["M":"MINUTES",PSONEW("DURATION",I)["H":"**  **HOURS",PSONEW("DURATION",I)["L":"MONTHS",PSONEW("DURATION",I)["W":"WEEKS",1:"DAY**  **S")\_")"**  **;I $G(PSONEW("CONJUNCTION",I))]"" S IEN=IEN+1,^TMP("PSOPO",$J,IEN,0)="**  **\*Conjunction: "\_$S(PSONEW("CONJUNCTION",I)="T":"THEN",PSONEW("CONJUNCTION"**  **,I)="X":"EXCEPT",1:"AND")**  **;Q**  **;**  **FMD ;**  **Q:$G(PSONEW("DOSE",II))']"" S MIG=PSONEW("DOSE",II)**  **I $E(MIG,1)=".",$G(PSONEW("DOSE ORDERED",II)) S MIG="0"\_MIG**  **;F SG=1:1:$L(MIG," ") S:$L(^TMP("PSOPO",$J,IEN,0)\_" "\_$P(MIG," ",SG))>8**  **0 IEN=IEN+1,$P(^TMP("PSOPO",$J,IEN,0)," ",20)=" " S ^TMP("PSOPO",$J,IEN,0)=$G(^T**  **MP("PSOPO",$J,IEN,0))\_" "\_$P(MIG," ",SG)**  **F SG=1:1:$L(MIG," ") D**  **.I $L(LINETXT\_" "\_$P(MIG," ",SG))>80 D Q**  **..S LINE=LINE+1 D ADDITEM^PSOERX1A(.LINETXT," ","",20,1)**  **..D ADDITEM^PSOERX1A(.LINETXT," ",$P(MIG," ",SG),$L(LINETXT),$L($P(MIG,**  **" ",SG))+1)**  **..;S ^TMP("PSOPO",$J,IEN,0)=$G(^TMP("PSOPO",$J,IEN,0))\_" "\_$P(MIG," ",S**  **G)**  **.D ADDITEM^PSOERX1A(.LINETXT," ",$P(MIG," ",SG),$L(LINETXT),$L($P(MIG,"**  **",SG))+1)**  **D SET^VALM10(LINE,LINETXT) S LINETXT=""**  **;I $G(UNITS)]"" S:$L(^TMP("PSOPO",$J,IEN,0)\_" ("\_UNITS\_")")>80 IEN=IEN+**  **1,$P(^TMP("PSOPO",$J,IEN,0)," ",20)=" " S ^TMP("PSOPO",$J,IEN,0)=$G(^TMP("PSOPO"**  **,$J,IEN,0))\_" ("\_UNITS\_")"**  **I $G(UNITS)]"" D**  **.;I $L(^TMP("PSOPO",$J,IEN,0)\_" ("\_UNITS\_")")>80 D**  **.I $L(LINETXT\_" ("\_UNITS\_")")>80 D Q**  **..S LINE=LINE+1,LINETXT=""**  **..D ADDITEM^PSOERX1A(.LINETXT," ","",20,1)**  **..D ADDITEM^PSOERX1A(.LINETXT," ("\_UNITS\_")",22,50)**  **..D SET^VALM10(LINE,LINETXT) S LINETXT=""**  **..;S $P(^TMP("PSOPO",$J,IEN,0)," ",20)=" " S ^TMP("PSOPO",$J,IEN,0)=$G(**  **^TMP("PSOPO",$J,IEN,0))\_" ("\_UNITS\_")"**  **K DS,MIG,SG**  **; TODO - PSODFN NOT DEFINED.**  **;I '$G(PSONEW("DOSE ORDERED",II)),$P($G(^PS(55,PSODFN,"LAN")),"^") D LA**  **N^PSOORED5**  **Q** |

Table 47: Routine: PSOERXH1

| Routines | Activities | | | |
| --- | --- | --- | --- | --- |
| **Routine Name** | PSOERXH1 | | | |
| **Enhancement Category** | New | Modify | Delete | No Change |
| **RTM** | eRx Utilities/RPCs  Story #453522: Hold functionality in List View | | | |
| **Related Options** |  | | | |

| Related Routines | Routines “Called By” | Routines “Called” |
| --- | --- | --- |
|  |  |  |

| Routines | Activities | | | | |
| --- | --- | --- | --- | --- | --- |
| **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) |
| --- |
| **PSOERXH1 ;ALB/BWF - eRx Utilities/RPC's ; 8/3/2016 5:14pm**  **;;7.0;OUTPATIENT PHARMACY;\*\*467\*\*;DEC 1997;Build 46**  **;**  **Q**  **; place eRx on Hold**  **HOLD ;**  **N DIE,DA,DR,CURSTAT,CSTATI,LMATCH,LSTAT,SUBFIEN,NEWSTAT,RESP**  **Q:'$G(PSOIEN)**  **D FULL^VALM1**  **; check to see if the erx order status is a hold status**  **S CSTATI=$$GET1^DIQ(52.49,PSOIEN,1,"I")**  **S CURSTAT=$$GET1^DIQ(52.49,PSOIEN,1,"E")**  **S VALMBCK="R"**  **I $E(CURSTAT,1)="H" D Q**  **.S DIR(0)="YO",DIR("B")="NO"**  **.S DIR("A",1)="This eRx is already in a 'HOLD' status."**  **.S DIR("A")="Would you like to change the hold status and comments?"**  **.D ^DIR**  **.Q:'Y**  **.K DIR**  **.S RESP=$$HDIR() Q:'RESP**  **.S SUBFIEN=$$NSTAT(PSOIEN,RESP)**  **.S DIE="^PS(52.49,"\_PSOIEN\_",19,",DA=SUBFIEN,DA(1)=PSOIEN,DR="1" D ^DIE**  **.K DA,DR,DIE S DIE="^PS(52.49,",DA=PSOIEN,DR="1///"\_RESP D ^DIE**  **.K @VALMAR D INIT^PSOERX1**  **S RESP=$$HDIR() Q:'RESP**  **S FDA(52.4919,"+1,"\_PSOIEN\_",",.01)=$$NOW^XLFDT()**  **S FDA(52.4919,"+1,"\_PSOIEN\_",",.02)=RESP**  **S FDA(52.4919,"+1,"\_PSOIEN\_",",.03)=$G(DUZ)**  **D UPDATE^DIE(,"FDA","NEWSTAT") K FDA**  **S SUBFIEN=$O(NEWSTAT(0)) Q:'SUBFIEN**  **S SUBFIEN=$G(NEWSTAT(SUBFIEN))**  **S DIE="^PS(52.49,"\_PSOIEN\_",19,",DA=SUBFIEN,DA(1)=PSOIEN,DR="1" D ^DIE**  **K DA,DR,DIE**  **S DIE="^PS(52.49,",DA=PSOIEN,DR="1///"\_RESP D ^DIE K DA,DR,DIE**  **K @VALMAR D INIT^PSOERX1**  **Q**  **NSTAT(IEN,STAT) ;**  **N SUBFIEN**  **S FDA(52.4919,"+1,"\_IEN\_",",.01)=$$NOW^XLFDT()**  **S FDA(52.4919,"+1,"\_IEN\_",",.02)=STAT**  **S FDA(52.4919,"+1,"\_IEN\_",",.03)=$G(DUZ)**  **D UPDATE^DIE(,"FDA","NEWSTAT") K FDA**  **S SUBFIEN=$O(NEWSTAT(0)) Q:'SUBFIEN**  **S SUBFIEN=$G(NEWSTAT(SUBFIEN))**  **Q SUBFIEN**  **HDIR() ;**  **N DIR,Y**  **S DIR(0)="PO^PS(52.45,",DIR("S")="I $D(^PS(52.45,""C"",""ERX"",X)),$E($**  **P(^PS(52.45,Y,0),U),1)=""H""" D ^DIR**  **Q:'+$P(Y,U) 0**  **Q $P(Y,U)**  **; remove hold from eRx**  **HDIR2() ;**  **N DIR,Y**  **S DIR(0)="PO^PS(52.45,",DIR("S")="I $D(^PS(52.45,""C"",""ERX"",X)),$E($**  **P(^PS(52.45,Y,0),U),1)'=""H""" D ^DIR**  **Q:'+$P(Y,U) 0**  **Q $P(Y,U)**  **UNHOLD ;**  **N Y,DIR,DIE,DA,DR,NSTAT,NEWSIEN**  **D FULL^VALM1**  **S VALMBCK="R"**  **W !**  **I $E($$GET1^DIQ(52.49,PSOIEN,1,"E"),1)'="H" D Q**  **.W !,"This eRx is not currently on hold. Please use the 'Hold'",!,"func**  **tion to update the hold status and comments.",!!**  **.S DIR(0)="E"**  **.D ^DIR**  **.K @VALMAR D INIT^PSOERX1**  **;S NSTAT=$$HDIR2() Q:'NSTAT**  **S NSTAT=$O(^PS(52.45,"C","ERX","I",0))**  **S DIE="^PS(52.49,",DA=PSOIEN,DR="1///"\_NSTAT D ^DIE**  **;S NEWSIEN=$$NSTAT(PSOIEN,NSTAT)**  **W !,"eRx removed from hold status, and placed to 'In Progress'."**  **S DIR(0)="E" D ^DIR K DIR**  **K @VALMAR D INIT^PSOERX1**  **;S DIE="^PS(52.49,",DA=PSOIEN,DR="1///"\_NSTAT\_";19.1///"""""\_";19.2///@**  **;19.4///"\_$$NOW^XLFDT\_";19.5///"\_DUZ D ^DIE**  **Q** |

Table 48: Routine: PSOERXO1

| Routines | Activities | | | |
| --- | --- | --- | --- | --- |
| **Routine Name** | PSOERXO1 | | | |
| **Enhancement Category** | New | Modify | Delete | No Change |
| **RTM** | eRx Outbound Error messages | | | |
| **Related Options** |  | | | |

| Related Routines | Routines “Called By” | Routines “Called” |
| --- | --- | --- |
|  |  |  |

| Routines | Activities | | | | |
| --- | --- | --- | --- | --- | --- |
| **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) |
| --- |
| **PSOERXO1 ;ALB/BWF - eRx Outbound Error messages ; 8/3/2016 5:14pm**  **;;7.0;OUTPATIENT PHARMACY;\*\*467\*\*;DEC 1997;Build 46**  **;**  **Q**  **;**  **; todo - should this be modified to check errorSection from our XML?**  **MSGERR() ;check errors from XML return**  **;returns empty string "" if there was no error**  **;returns empty string "" if the only error was "ALL\_PATIENT\_IDS\_EXCLUDE**  **D"**  **;otherwise returns the exceptionMessage string from the errorSection**  **N ORRET S ORRET=""**  **I $D(^TMP($J,"ORRDI","ClinicalData",0,"errorSection")) D**  **.N I F I="fatalErrors","errors","warnings" D**  **..N J S J="" F S J=$O(^TMP($J,"ORRDI","ClinicalData",0,"errorSection",**  **0,I,J)) Q:J="" D**  **...N ORSTR S ORSTR=$G(^TMP($J,"ORRDI","ClinicalData",0,"errorSection",0**  **,I,J,"errorCode",0))**  **...I ORSTR'="ALL\_PATIENT\_IDS\_EXCLUDED" S ORRET=ORSTR**  **Q ORRET**  **ERRHNDL(DFN) ;handle any errors that may get thrown in call to GET^ORRDI1**  **K ^TMP($J,"ORRDI"),^XTMP("ORRDI","PSOO",DFN),^XTMP("ORRDI","ART",DFN)**  **D UNWIND^%ZTER**  **Q**  **POST(ERXIEN,PSSOUT,ECODE,DESCODE,DESC) ;**  **N PSS,PSSERR,PSSFDBRT,PSREQ,INST,GBL,C**  **N TOQUAL,FRQUAL,TO,FROM,MID,RTMID,ERXIENS,F,PSODAT**  **S F=52.49,C=0**  **S PSSFDBRT=1**  **S GBL=$NA(^TMP("POST^PSOERXO1",$J)) K @GBL**  **Q:'$G(ERXIEN)**  **S ERXIENS=ERXIEN\_","**  **D GETS^DIQ(F,ERXIENS,".01;.02;22.1:22.4;24.1","IE","PSODAT")**  **S INST=$G(PSODAT(F,ERXIENS,24.1,"I")) I 'INST S PSSOUT(0)=-1\_U\_"Unable**  **to identify instution. Cannot send message." Q**  **; TODO - message ID needs to be unique from vista - Site#.DUZ.erxIEN.da**  **te.time??**  **S MID=INST\_"."\_DUZ\_"."\_ERXIEN\_"."\_$$NOW^XLFDT**  **;S MID=$G(PSODAT(F,ERXIENS,.01,"E"))**  **; relates to message ID is the incoming message id from CH for outbound**  **messages.**  **S RTMID=$G(PSODAT(F,ERXIENS,.01,"E"))**  **; from is TO from the erx.**  **S FROM=$G(PSODAT(F,ERXIENS,22.3,"E"))**  **S FRQUAL=$G(PSODAT(F,ERXIENS,22.4,"I"))**  **; to is FROM from the erx**  **S TO=$G(PSODAT(F,ERXIENS,22.1,"E"))**  **S TOQUAL=$G(PSODAT(F,ERXIENS,22.2,"I"))**  **;**  **D C S @GBL@(C,0)="<?xml version = '1.0' encoding = 'UTF-8'?><Message ve**  **rsion=""010"" release=""006"" HighestVersionSupported="""" xmlns=""http://www.nc**  **pdp.org/schema/SCRIPT"">"**  **D C S @GBL@(C,0)="<Header><To Qualifier="""\_TOQUAL\_""">"\_TO\_"</To><From**  **Qualifier="""\_FRQUAL\_""">"\_FROM\_"</From><MessageID>"\_MID\_"</MessageID>"**  **D C S @GBL@(C,0)="<RelatesToMessageID>"\_RTMID\_"</RelatesToMessageID><Se**  **ntTime>"\_$$EXTIME()\_"</SentTime></Header>"**  **D C S @GBL@(C,0)="<Body><Error><Code>"\_$G(ECODE)\_"</Code>"**  **I $L(DESCODE) D C S @GBL@(C,0)="<DescriptionCode>"\_$G(DESCODE)\_"</Descr**  **iptionCode>"**  **D C S @GBL@(C,0)="<Description>"\_$G(DESC)\_"</Description>"**  **D C S @GBL@(C,0)="</Error></Body></Message>"**  **;**  **;S PSREQ="<?xml version = '1.0' encoding = 'UTF-8'?><Message version=""**  **010"" release=""006"" HighestVersionSupported="""" xmlns=""http://www.ncpdp.org/**  **schema/SCRIPT"">"**  **;S PSREQ=PSREQ\_"<Header><To Qualifier="""\_TOQUAL\_""">"\_TO\_"</To><From Q**  **ualifier="""\_FRQUAL\_""">"\_FROM\_"</From><MessageID>"\_MID\_"</MessageID>"**  **;S PSREQ=PSREQ\_"<RelatesToMessageID>"\_RTMID\_"</RelatesToMessageID><Sent**  **Time>"\_$$SENTTIME()\_"</SentTime></Header>"**  **;S PSREQ=PSREQ\_"<Body><Error><Code>"\_$G(ECODE)\_"</Code>"**  **;I $L(DESCODE) S PSREQ=PSREQ\_"<DescriptionCode>"\_$G(DESCODE)\_"</Descrip**  **tionCode>"**  **;S PSREQ=PSREQ\_"<Description>"\_$G(DESC)\_"</Description>"**  **;S PSREQ=PSREQ\_"</Error></Body></Message>"**  **D RESTPOST(.PSSOUT,.GBL)**  **K @GBL,C**  **Q**  **C ;**  **S C=C+1**  **Q**  **RESTPOST(PSSOUT,GBL) ;**  **;N $ETRAP,$ESTACK,PSREQ**  **N PSREQ,PSS,PSSERR,GLOOP,GDAT**  **; Set error trap**  **;SET $ETRAP="DO ERROR^PSSHTTP"**  **K ^TMP($J,"OUT") ; if exists from previous runs, posting would not e**  **xecute.**  **SET PSS("server")="PSO WEB SERVER"**  **SET PSS("webserviceName")="PSO ERX WEB SERVICE"**  **SET PSS("path")="services/rest/vistaoutboundMsg/processXMLMessage"**  **SET PSS("parameterName")="xmlRequest"**  **;SET PSS("parameterValue")=PSREQ**  **;**  **; get instance of client REST request object**  **SET PSS("restObject")=$$GETREST^XOBWLIB(PSS("webserviceName"),PSS("serv**  **er"))**  **IF $DATA(^TMP($JOB,"OUT","EXCEPTION"))>0 S PSSOUT(0)="-1^"\_^TMP($JOB,"O**  **UT","EXCEPTION") K ^TMP($JOB,"OUT","EXCEPTION") Q PSSOUT**  **;**  **; insert XML as parameter**  **;DO PSS("restObject").InsertFormData(PSS("parameterName"),PSS("paramete**  **rValue"))**  **S PSS("restObject").ContentType="application/xml"**  **S GLOOP=0 F S GLOOP=$O(@GBL@(GLOOP)) Q:'GLOOP D**  **.S GDAT=$G(@GBL@(GLOOP,0))**  **.;SET PSS("parameterValue")=$G(PSS("parameterValue"))\_$G(@GBL@(GLOOP,0)**  **)**  **.DO PSS("restObject").EntityBody.Write(GDAT)**  **;DO PSS("restObject").InsertFormData(PSS("parameterName"),PSS("paramete**  **rValue"))**  **IF $DATA(^TMP($JOB,"OUT","EXCEPTION"))>0 S PSSOUT(0)="-1^"\_^TMP($JOB,"O**  **UT","EXCEPTION") K ^TMP($JOB,"OUT","EXCEPTION") QUIT PSSOUT**  **;**  **; execute HTTP Post method**  **SET PSS("postResult")=$$POST^XOBWLIB(PSS("restObject"),PSS("path"),.PSS**  **ERR)**  **IF $DATA(^TMP($JOB,"OUT","EXCEPTION"))>0 S PSSOUT(0)="-1^"\_^TMP($JOB,"O**  **UT","EXCEPTION") K ^TMP($JOB,"OUT","EXCEPTION") QUIT PSSOUT**  **;**  **; response coming back**  **;<vistaOutboundResponse><success>true</success></vistaOutboundResponse>**  **; error handling**  **DO:'PSS("postResult")**  **. SET PSSOUT(0)=-1\_U\_"Unable to make http request."**  **. SET PSS("result")=0**  **. QUIT**  **;**  **; if every thing is ok parse the returned xml result**  **I PSS("postResult") S PSS("result")=1 D PRSSTRM(PSS("restObject"),.PSSO**  **UT) S PSSOUT(0)=1**  **; TODO - do we need to pass back the message ID for storage into 52.49?**  **Q PSS("result")**  **;**  **PRSSTRM(RESTOBJ,PSSOUT) ; parse the XML into token**  **; input: RESTOBJ--a rest object**  **; output: PSSOUT - array containing the list of route names for the giv**  **en drug.**  **;**  **; parse the XML into tokens. the first part of the token is the type of**  **node being read.**  **; the second part is the data--either the name of the node, or data. th**  **ese fields are delimited using "<>".**  **; if the node is type attribute, each attribute is separated by a caret**  **("^").**  **;**  **N AREADER**  **S AREADER=$$GETREADR(RESTOBJ)**  **D PARSXML(AREADER,.PSSOUT)**  **Q**  **;**  **PARSXML(AREADER,PSSOUT) ; extract the list of routes from XML results**  **; input: AREADER-%XML.TextReader object.**  **; output: PSSOUT - array containing the list of route names for the giv**  **en drug.**  **;**  **N ATOKEN,NODETYPE**  **F D Q:AREADER.EOF**  **.S ATOKEN=$$GETTOKEN(AREADER)**  **.I '$L(ATOKEN) Q**  **.S NODETYPE=$P(ATOKEN,"<>"),ATOKEN=$P(ATOKEN,"<>",2)**  **.I ATOKEN="errorMessage" D POSTERR(AREADER,.PSSOUT)**  **.I ATOKEN="success" D POSTRES(AREADER,.PSSOUT)**  **Q**  **;**  **POSTRES(AREADER,PSSOUT) ; get value of success/failure**  **N TOKEN,TYPE**  **F D Q:TOKEN="/success"**  **.S TOKEN=$$GETTOKEN(AREADER)**  **.S TYPE=$P(TOKEN,"<>"),TOKEN=$P(TOKEN,"<>",2)**  **.Q:'$L(TOKEN)!(TOKEN="/success")**  **.S PSSOUT("success")=TOKEN**  **Q**  **POSTERR(AREADER,PSSOUT) ; get error message**  **N TOKEN,TYPE**  **F D Q:TOKEN="/errorMessage"**  **.S TOKEN=$$GETTOKEN(AREADER)**  **.S TYPE=$P(TOKEN,"<>"),TOKEN=$P(TOKEN,"<>",2)**  **.Q:'$L(TOKEN)!(TOKEN="/errorMessage")**  **.S PSSOUT("errorMessage")=$TR(TOKEN,$C(10)," ")**  **Q**  **;**  **GETREADR(RESTOBJ) ; set up and return a Textreader object to be used to parse th**  **e XML stream**  **; input: RESTOBJ- REST object**  **; output: returns a %XML.TextReader object.**  **;**  **N AREADER**  **S AREADER=##class(%XML.TextReader).%New("%XML.TextReader")**  **D ##class(%XML.TextReader).ParseStream(RESTOBJ.HttpResponse.Data,.AREAD**  **ER)**  **Q AREADER**  **;**  **GETTOKEN(READER) ; get a token at a time**  **; input: AREADER-%XML.TextReader object**  **; Output: returns a token**  **;**  **; this is the key to the parsing of the XML stream.**  **; each element is parsed with its associated data (if any)**  **; the nodetype is concatenated with "<>" with the Token**  **; which can be the tag or the data.**  **; for example each time is called return one of the following:**  **; . . .**  **; . . .**  **; drug(attributes)<>gcnSeqNo=17240**  **; element<>routes**  **; element<>route**  **; element<>id**  **; chars<>006**  **; endelement<>/id**  **; element<>name**  **; chars<>CONTINUOUS INFUSION**  **; endelement<>/name**  **; endelement<>/route**  **; . . .**  **; . . .**  **;**  **N TOKEN,NODETYPE,SUBTOKEN,ALLTOKEN**  **S TOKEN="",SUBTOKEN="",NODETYPE="",ALLTOKEN=""**  **D**  **.Q:READER.EOF**  **.D READER.Read() ; go to first node**  **.Q:READER.EOF ; try before and after read**  **.;W !,READER.NodeTypeGet()**  **.;S NODETYPE=READER.NodeTypeGet()**  **.I READER.HasAttributes D**  **..S NODETYPE=READER.Name\_"(attributes)"**  **..S TOKEN=$$GETATTS(READER)**  **.I '$L(TOKEN) S NODETYPE=READER.NodeTypeGet() D**  **..I NODETYPE="element" S TOKEN=READER.Name Q**  **..I NODETYPE="chars" S TOKEN=READER.Value Q**  **..I NODETYPE="endelement" S TOKEN="/"\_READER.Name Q**  **..I NODETYPE="comment" S TOKEN="^"**  **..I NODETYPE="processinginstruction" S TOKEN=READER.Value Q**  **..I NODETYPE="ignorablewhitespace" S TOKEN="^" Q**  **..I NODETYPE="startprefixmapping" S TOKEN=READER.Value Q**  **..I NODETYPE="warning" S TOKEN=READER.Value Q**  **..I '$L(TOKEN) S TOKEN="^"**  **..;**  **.I $L(NODETYPE) S ALLTOKEN=NODETYPE\_"<>"\_TOKEN**  **;W !,"TOKEN="\_ALLTOKEN**  **Q ALLTOKEN**  **;**  **GETATTS(AREADER) ; get attributes**  **; input: AREADER-%XML.TextReader object**  **; Output: returns the attributes**  **;**  **N I,INDEX,TOKEN,SUBTOKEN,ATTRB**  **S (TOKEN,SUBTOKEN)=""**  **S INDEX=AREADER.AttributeCountGet()**  **FOR I=1:1:INDEX D**  **.S ATTRB=AREADER.MoveToAttributeIndex(I) D**  **.S SUBTOKEN=AREADER.Name\_"="\_AREADER.Value**  **.I '$L(TOKEN) S TOKEN=SUBTOKEN Q**  **.S TOKEN=TOKEN\_"^"\_SUBTOKEN**  **;W !," ATT=",TOKEN**  **Q TOKEN**  **EXTIME(IDTTM) ;**  **N YY,MM,DD,TIME,EXDT,TLEN,I,TZONE,DTTM**  **S IDTTM=$G(IDTTM,"")**  **S DTTM=$S($L(IDTTM):$$FMTHL7^XLFDT(IDTTM),1:$$FMTHL7^XLFDT($$NOW^XLFDT(**  **)))**  **S TZONE=$P(DTTM,"-",2)**  **I 'TZONE S TZONE=$P($$FMTHL7^XLFDT($$NOW^XLFDT()),"-",2)**  **S DTTM=$P(DTTM,"-"),TZONE=$E(TZONE,1,2)\_":"\_$E(TZONE,3,4)**  **S YY=$E(DTTM,1,4),MM=$E(DTTM,5,6),DD=$E(DTTM,7,8),TIME=$E(DTTM,9,$L(DTT**  **M))**  **I $L(TIME)<6 D**  **.S TLEN=$L(TIME)**  **.F I=TLEN:1:6 D**  **..S TIME=TIME\_0**  **; now construct the date**  **S EXDT=YY\_"-"\_MM\_"-"\_DD\_"T"\_$E(TIME,1,2)\_":"\_$E(TIME,3,4)\_":"\_$E(TIME,5**  **,6)\_$S($L(TZONE):"-"\_TZONE,1:"")**  **Q EXDT** |

Table 49: Routine: PSOERXP1

| Routines | Activities | | | |
| --- | --- | --- | --- | --- |
| **Routine Name** | PSOERXP1 | | | |
| **Enhancement Category** | New | Modify | Delete | No Change |
| **RTM** | eRx Patient Validation Display / Actions  Story # 417240 | | | |
| **Related Options** |  | | | |

| Related Routines | Routines “Called By” | Routines “Called” |
| --- | --- | --- |
|  |  |  |

| Routines | Activities | | | | |
| --- | --- | --- | --- | --- | --- |
| **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) |
| --- |
| **PSOERXP1 ;ALB/BWF - eRx Patient Display/Actions ; 8/3/2016 5:14pm**  **;;7.0;OUTPATIENT PHARMACY;\*\*467\*\*;DEC 1997;Build 46**  **;**  **EN ; -- main entry point for PSO ERX HOLDING QUEUE**  **D EN^VALM("PSO ERX PATIENT VALIDATION")**  **Q**  **;**  **HDR ; -- header code**  **S VALMHDR(1)="eRx Patient: "\_$$GET1^DIQ(52.49,PSOIEN,.04,"E")**  **S VALMHDR(2)="eRx Reference #: "\_$$GET1^DIQ(52.49,PSOIEN,25,"E")**  **I $G(VALMBCK)="R" D INIT**  **Q**  **;**  **INIT ;**  **Q:'$G(PSOIEN)**  **N LINE,PATDAT,EXPIEN,EXPIENS,EXPNM,EXPDOB,EXPSSN,EXPADD,EXPGEN,EXPCTY,E**  **XPST,EXPZIP,HFIEN,EXPHPH,CPIEN,EXPCPH,LINETXT**  **N MANVAL,VAPATIEN,SEPLN,COMM,COMLINE,DONE,TEXT,VAEL,VAELIEN,VAELIG,VAEL**  **NM,WORD**  **S LINE=0,LINETXT=""**  **S EXPIEN=$$GET1^DIQ(52.49,PSOIEN,.04,"I")**  **S EXPIENS=EXPIEN\_","**  **D GETS^DIQ(52.46,EXPIENS,"\*\*","E","PATDAT")**  **S EXPNM=$G(PATDAT(52.46,EXPIENS,.01,"E"))**  **S EXPDOB=$G(PATDAT(52.46,EXPIENS,.08,"E"))**  **S EXPSSN=$G(PATDAT(52.46,EXPIENS,1.4,"E"))**  **S EXPADD=$G(PATDAT(52.46,EXPIENS,3.1,"E"))**  **S EXPGEN=$G(PATDAT(52.46,EXPIENS,.07,"E"))**  **S EXPCTY=$G(PATDAT(52.46,EXPIENS,3.3,"E"))**  **; todo - is the state pointer not working? why am i having to use inter**  **nal?**  **S EXPST=$G(PATDAT(52.46,EXPIENS,3.4,"E"))**  **; TODO - zip code internal and external seem to be broken??**  **S EXPZIP=$G(PATDAT(52.46,EXPIENS,3.5,"E")),EXPZIP=$E(EXPZIP,1,5)**  **S HFIEN=$O(^PS(52.46,EXPIEN,3,"C","HP",0))**  **; home phone**  **S EXPHPH=$$GET1^DIQ(52.462,HFIEN\_","\_EXPIENS,.01,"E")**  **; cellular phone**  **S CPIEN=$O(^PS(52.46,EXPIEN,3,"C","CP",0))**  **S EXPCPH=$$GET1^DIQ(52.462,CPIEN\_","\_EXPIENS,.01,"E")**  **S LINE=LINE+1**  **D ADDITEM^PSOERX1A(.LINETXT,"External Patient: ","",1,20)**  **D ADDITEM^PSOERX1A(.LINETXT,"DOB: ",EXPDOB,55,17)**  **D SET^VALM10(LINE,LINETXT) S LINETXT=""**  **S LINE=LINE+1**  **D ADDITEM^PSOERX1A(.LINETXT,"Patient: ",$E(EXPNM,1,40),1,50)**  **D ADDITEM^PSOERX1A(.LINETXT,"SSN: ",EXPSSN,55,15)**  **D SET^VALM10(LINE,LINETXT) S LINETXT=""**  **S LINE=LINE+1**  **D ADDITEM^PSOERX1A(.LINETXT,"Addr: ",$E(EXPADD,1,40),1,50)**  **D ADDITEM^PSOERX1A(.LINETXT,"Sex: ",EXPGEN,55,15)**  **D SET^VALM10(LINE,LINETXT) S LINETXT=""**  **S LINE=LINE+1**  **D ADDITEM^PSOERX1A(.LINETXT,"City: ",$E(EXPCTY,1,20),1,20)**  **D ADDITEM^PSOERX1A(.LINETXT,"St: ",$E(EXPST,1,20),35,20)**  **D ADDITEM^PSOERX1A(.LINETXT,"Zip: ",EXPZIP,65,10)**  **D SET^VALM10(LINE,LINETXT) S LINETXT=""**  **S LINE=LINE+1**  **D ADDITEM^PSOERX1A(.LINETXT,"Home Phone: ",EXPHPH,1,25)**  **D ADDITEM^PSOERX1A(.LINETXT,"Cell Phone: ",EXPCPH,40,25)**  **D SET^VALM10(LINE,LINETXT) S LINETXT=""**  **S COMM=$$GET1^DIQ(52.49,PSOIEN,8,"E"),COMM="Comments: "\_COMM**  **S COMLINE="",DONE=0**  **I $L(COMM)<80!$L(COMM)=80 D SET^VALM10(LINE,COMM)**  **I $L(COMM)>80 D**  **.F WORD=1:1:$L(COMM," ") D**  **..S TEXT=$P(COMM," ",WORD)**  **..; if the text is to long, quit**  **..I $L(COMLINE)+$L(TEXT)>80 D Q**  **...S LINE=LINE+1 D SET^VALM10(LINE,COMLINE)**  **...S COMLINE=TEXT**  **..I '$L(COMLINE) S COMLINE=TEXT Q**  **..S COMLINE=$G(COMLINE)\_" "\_TEXT**  **S LINE=LINE+1 D SET^VALM10(LINE,"")**  **S LINE=LINE+1 D SET^VALM10(LINE,"")**  **S $P(SEPLN,"-",80)="-" D SET^VALM10(LINE,SEPLN)**  **; vista patient information**  **S VAPATIEN=$$GET1^DIQ(52.49,PSOIEN,.05,"I")**  **I VAPATIEN D**  **.S DFN=VAPATIEN D DEM^VADPT,ADD^VADPT,ELIG^VADPT**  **.S VAELIG=$G(VAEL(1)),VAELIEN=$P(VAELIG,U),VAELNM=$P(VAELIG,U,2)**  **S MANVAL=$$GET1^DIQ(52.49,PSOIEN,1.7,"I")**  **; TODO - MOVE PATIENT NAME behind Vista Patient:**  **S LINE=LINE+1 D SET^VALM10(LINE,"Vista Patient: "\_$S(VAPATIEN:$G(VADM(1**  **)),1:"NOT VALIDATED"))**  **I 'VAPATIEN S LINE=LINE+1 D SET^VALM10(LINE,"PATIENT NOT MATCHED")**  **I VAPATIEN D**  **.S LINE=LINE+1**  **.;D ADDITEM^PSOERX1A(.LINETXT,"Name: ",$G(VADM(1)),1,35)**  **.D ADDITEM^PSOERX1A(.LINETXT,"Sex: ",$P($G(VADM(5)),U,2),1,20)**  **.D ADDITEM^PSOERX1A(.LINETXT,"DOB: ",$P($G(VADM(3)),U,2),55,20)**  **.D SET^VALM10(LINE,LINETXT) S LINETXT=""**  **.S LINE=LINE+1**  **.;D ADDITEM^PSOERX1A(.LINETXT,"VA Number?: ",$G(VA("BID")),1,25)**  **.D ADDITEM^PSOERX1A(.LINETXT,"SSN: ",$P($G(VADM(2)),U,2),55,20)**  **.D SET^VALM10(LINE,LINETXT) S LINETXT=""**  **.S LINE=LINE+1**  **.D ADDITEM^PSOERX1A(.LINETXT,"Address: ",$P($G(VAPA(1)),U),1,25)**  **.;D ADDITEM^PSOERX1A(.LINETXT,"Sex: ",$P($G(VADM(5)),U,2),55,20)**  **.D SET^VALM10(LINE,LINETXT) S LINETXT=""**  **.S LINE=LINE+1**  **.D ADDITEM^PSOERX1A(.LINETXT,"City: ",$P($G(VAPA(4)),U),1,25)**  **.D ADDITEM^PSOERX1A(.LINETXT,"Zip: ",$P($G(VADM(6)),U),55,20)**  **.D SET^VALM10(LINE,LINETXT) S LINETXT=""**  **.S LINE=LINE+1**  **.D ADDITEM^PSOERX1A(.LINETXT,"Home Phone: ",$P($G(VAPA(8)),U),1,25)**  **.D ADDITEM^PSOERX1A(.LINETXT,"Cell Phone: ",$$GET1^DIQ(2,VAPATIEN,.134,**  **"E"),55,25)**  **.D SET^VALM10(LINE,LINETXT) S LINETXT=""**  **.S LINE=LINE+1 D SET^VALM10(LINE,"")**  **.S LINE=LINE+1 D SET^VALM10(LINE,"Eligibility: "\_$G(VAELNM))**  **.S LINE=LINE+1 D SET^VALM10(LINE,"Pharmacy Narrative: "\_$$GET1^DIQ(55,V**  **APATIEN,1,"E"))**  **S VALMCNT=LINE**  **S EDTYP="P"**  **K VAPA,VADM,DFN,VA**  **Q**  **HELP ; -- help code**  **S X="?" D DISP^XQORM1 W !!**  **Q**  **;**  **EXIT ; -- exit code**  **K EDTYP,@VALMAR**  **Q**  **;**  **EXPND ; -- expand code**  **Q** |

Table 50: Routine: PSOERXR1

| Routines | Activities | | | |
| --- | --- | --- | --- | --- |
| **Routine Name** | PSOERXR1 | | | |
| **Enhancement Category** | New | Modify | Delete | No Change |
| **RTM** | Validate Provider Display/Actions  Story # 417243 | | | |
| **Related Options** |  | | | |

| Related Routines | Routines “Called By” | Routines “Called” |
| --- | --- | --- |
|  |  |  |

| Routines | Activities | | | | |
| --- | --- | --- | --- | --- | --- |
| **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) |
| --- |
| **PSOERXR1 ;ALB/BWF - eRx Provider Display/actions ; 8/3/2016 5:14pm**  **;;7.0;OUTPATIENT PHARMACY;\*\*467\*\*;DEC 1997;Build 46**  **;**  **EN ; -- main entry point for PSO ERX HOLDING QUEUE**  **D EN^VALM("PSO ERX PROVIDER VALIDATION")**  **Q**  **;**  **HDR ; -- header code**  **;S VALMHDR(1)="eRx Processing"**  **S VALMHDR(1)="eRx Provider: "\_$$GET1^DIQ(52.49,PSOIEN,2.1,"E")**  **S VALMHDR(2)="eRx Message ID: "\_$$GET1^DIQ(52.49,PSOIEN,25,"E")**  **I $G(VALMBCK)="R" D INIT**  **;S VALMHDR(2)="This is the second line"**  **Q**  **;**  **INIT ;**  **Q:'$G(PSOIEN)**  **N LINE,PARVDAT,EXPIEN,EXPIENS,EXPNM,EXPDOB,EXPSSN,EXPADD,EXPGEN,EXPCTY,**  **EXPST,EXPZIP,HFIEN,EXPHPH,CPIEN,EXPCPH,LINETXT**  **N EXPADD1,EXPADD2,EXPAGNT,EXPDEA,EXPFN,EXPLIC,EXPNPI,EXPSUPER,EXPWPH,FN**  **IEN,MANVAL,PRVDAT,VAPADD1,VAPADD2,VAPCIT**  **N VAPDEA,VAPFAX,VAPIENS,VAPLIC,VAPNM,VAPNPI,VAPROV,VAPRVIEN,VAPST,VAPTE**  **L,VAPZIP,WPIEN,SEPLN,VAPDEAEX**  **;N VAPA,VADM,DFN,VA**  **S LINE=0,LINETXT=""**  **S EXPIEN=$$GET1^DIQ(52.49,PSOIEN,2.1,"I")**  **S EXPIENS=EXPIEN\_","**  **D GETS^DIQ(52.48,EXPIENS,"\*\*","E","PRVDAT")**  **S EXPNM=$G(PRVDAT(52.48,EXPIENS,.01,"E"))**  **S EXPNPI=$G(PRVDAT(52.48,EXPIENS,1.5,"E"))**  **S EXPDEA=$G(PRVDAT(52.48,EXPIENS,1.6,"E"))**  **S EXPLIC=$G(PRVDAT(52.48,EXPIENS,1.8,"E"))**  **S EXPAGNT=$G(PRVDAT(52.48,EXPIENS,5.1,"E"))\_", "\_$G(PRVDAT(52.48,EXPIEN**  **S,5.2,"E"))**  **S EXPSUPER=$$GET1^DIQ(52.49,PSOIEN,2.6,"E")**  **S EXPADD1=$G(PRVDAT(52.48,EXPIENS,4.1,"E"))**  **S EXPADD2=$G(PRVDAT(52.48,EXPIENS,4.2,"E"))**  **S EXPCTY=$G(PRVDAT(52.48,EXPIENS,4.3,"E"))**  **S EXPST=$G(PRVDAT(52.48,EXPIENS,4.4,"E"))**  **S EXPZIP=$G(PRVDAT(52.48,EXPIENS,4.5,"E")),EXPZIP=$E(EXPZIP,1,5)**  **S WPIEN=$O(^PS(52.48,EXPIEN,3,"C","WP",0))**  **; home phone**  **S EXPWPH=$$GET1^DIQ(52.483,WPIEN\_","\_EXPIENS,.01,"E")**  **; fax number**  **S FNIEN=$O(^PS(52.48,EXPIEN,3,"C","FX",0))**  **S EXPFN=$$GET1^DIQ(52.483,FNIEN\_","\_EXPIENS,.01,"E")**  **S LINE=LINE+1 D SET^VALM10(LINE,"External Provider: "\_$E(EXPNM,1,50))**  **S LINE=LINE+1 D SET^VALM10(LINE,"Address: "\_$E(EXPADD1,1,50))**  **S LINE=LINE+1 D SET^VALM10(LINE," "\_$E(EXPADD2,1,50))**  **S LINE=LINE+1 D SET^VALM10(LINE," "\_EXPCTY\_", "\_EXPST\_" "\_EXPZI**  **P)**  **S LINE=LINE+1**  **D ADDITEM^PSOERX1A(.LINETXT,"NPI: ",EXPNPI,1,26)**  **D ADDITEM^PSOERX1A(.LINETXT,"DEA: ",EXPDEA,28,26)**  **D ADDITEM^PSOERX1A(.LINETXT,"State Lic: ",EXPLIC,54,26)**  **D SET^VALM10(LINE,LINETXT) S LINETXT=""**  **;S LINE=LINE+1 D SET^VALM10(LINE,"NPI: "\_EXPNPI\_$J("DEA: "\_EXPDEA,20)\_$**  **J("State Lic: "\_EXPLIC,20))**  **S LINE=LINE+1**  **D ADDITEM^PSOERX1A(.LINETXT,"Tel: ",EXPWPH,1,26)**  **D ADDITEM^PSOERX1A(.LINETXT,"Fax: ",EXPFN,28,26)**  **D SET^VALM10(LINE,LINETXT) S LINETXT=""**  **;S LINE=LINE+1 D SET^VALM10(LINE,"Tel: "\_EXPWPH\_$J("Fax: "\_EXPFN,47))**  **S LINE=LINE+1**  **D ADDITEM^PSOERX1A(.LINETXT,"Agent: ",$S($L(EXPAGNT):EXPAGNT,1:""),1,35**  **)**  **D ADDITEM^PSOERX1A(.LINETXT,"Supervisor: ",$S($L(EXPSUPER):EXPSUPER,1:"**  **"),40,40)**  **D SET^VALM10(LINE,LINETXT) S LINETXT=""**  **;S LINE=LINE+1 D SET^VALM10(LINE,"Agent: "\_$E(EXPAGNT,1,25)\_$J("Supervi**  **sor: "\_$E(EXPSUPER,1,25),40))**  **S LINE=LINE+1 D SET^VALM10(LINE,"")**  **S LINE=LINE+1 D SET^VALM10(LINE,"")**  **S $P(SEPLN,"-",80)="-" D SET^VALM10(LINE,SEPLN)**  **; vista patient information**  **S VAPRVIEN=$$GET1^DIQ(52.49,PSOIEN,2.3,"I")**  **; INITIALIZE variables**  **S (VAPNM,VAPADD1,VAPADD2,VAPCIT,VAPST,VAPZIP,VAPNPI,VAPDEA,VAPLIC,VAPTE**  **L,VAPFAX)=""**  **S MANVAL=$$GET1^DIQ(52.49,PSOIEN,1.3,"I")**  **S LINE=LINE+1 D SET^VALM10(LINE,"VA Provider: "\_$S(MANVAL:"VALIDATED",1**  **:"NOT VALIDATED"))**  **I 'VAPRVIEN S LINE=LINE+1 D SET^VALM10(LINE,"NO MATCHED PROVIDER")**  **I VAPRVIEN D**  **.S VAPIENS=VAPRVIEN\_","**  **.; 41.99 - NPI, 53.2 - DEA, 54.2 - STATE LICENSING DEA NUMBER, .132 - O**  **FFICE PHONE, .136 - FAX**  **.D GETS^DIQ(200,VAPIENS\_",",".01;.111;.112;.113;.114;.115;.116;.132;.13**  **6;41.99;53.2;54.2","IE","VAPROV")**  **.S VAPNM=$G(VAPROV(200,VAPIENS,.01,"E"))**  **.S VAPADD1=$G(VAPROV(200,VAPIENS,.111,"E"))**  **.S VAPADD2=$G(VAPROV(200,VAPIENS,.112,"E"))**  **.S VAPCIT=$G(VAPROV(200,VAPIENS,.114,"E"))**  **.S VAPST=$G(VAPROV(200,VAPIENS,.115,"E"))**  **.S VAPZIP=$G(VAPROV(200,VAPIENS,.116,"E"))**  **.S VAPNPI=$G(VAPROV(200,VAPIENS,41.99,"E"))**  **.S VAPDEA=$G(VAPROV(200,VAPIENS,53.2,"E"))**  **.S VAPDEAEX=$$GET1^DIQ(200,VAPIENS,747.44,"I") I VAPDEAEX,VAPDEAEX<DT S**  **VAPDEAEX=1**  **.S VAPLIC=$G(VAPROV(200,VAPIENS,54.2,"E"))**  **.S VAPTEL=$G(VAPROV(200,VAPIENS,.132,"E"))**  **.S VAPFAX=$G(VAPROV(200,VAPIENS,.136,"E"))**  **.S MANVAL=$$GET1^DIQ(52.49,PSOIEN,1.3,"I")**  **.;S LINE=LINE+1 D SET^VALM10(LINE,"VA Provider: "\_$S(MANVAL:"VALIDATED"**  **,1:"NOT VALIDATED"))**  **.S LINE=LINE+1 D SET^VALM10(LINE,"Name: "\_VAPNM)**  **.S LINE=LINE+1 D SET^VALM10(LINE,"Address: "\_$S($L(VAPADD1):VAPADD1,1:"**  **No street address on file."))**  **.I $L(VAPADD2) S LINE=LINE+1 D SET^VALM10(LINE," "\_VAPADD2)**  **.S LINE=LINE+1 D SET^VALM10(LINE," "\_VAPCIT\_", "\_VAPST\_" "\_VAPZ**  **IP)**  **.S LINE=LINE+1**  **.D ADDITEM^PSOERX1A(.LINETXT,"NPI: ",VAPNPI,1,26)**  **.D ADDITEM^PSOERX1A(.LINETXT,"DEA: ",VAPDEA\_$S(VAPDEAEX:" (Expired)",1:**  **""),28,26)**  **.D ADDITEM^PSOERX1A(.LINETXT,"State Lic: ",VAPLIC,54,26)**  **.D SET^VALM10(LINE,LINETXT) S LINETXT=""**  **.S LINE=LINE+1**  **.D ADDITEM^PSOERX1A(.LINETXT,"Tel: ",VAPTEL,1,26)**  **.D ADDITEM^PSOERX1A(.LINETXT,"Fax: ",VAPFAX,28,26)**  **.D SET^VALM10(LINE,LINETXT) S LINETXT=""**  **S LINE=LINE+1 D SET^VALM10(LINE,"")**  **S VALMCNT=LINE**  **S EDTYP="PR"**  **Q**  **HELP ; -- help code**  **S X="?" D DISP^XQORM1 W !!**  **Q**  **;**  **EXIT ; -- exit code**  **K EDTYP,@VALMAR**  **Q**  **;**  **EXPND ; -- expand code**  **Q** |

Table 51: Routine: PSO467PST

| Routines | Activities | | | |
| --- | --- | --- | --- | --- |
| **Routine Name** | PSO467PST | | | |
| **Enhancement Category** | New | Modify | Delete | No Change |
| **RTM** | Patch 467 post-install | | | |
| **Related Options** |  | | | |

| Related Routines | Routines “Called By” | Routines “Called” |
| --- | --- | --- |
|  |  |  |

| Routines | Activities | | | | |
| --- | --- | --- | --- | --- | --- |
| **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) |
| --- |
| **PSO467PST ;ALB/BWF - patch 467 post-install ; 8/17/2016 1:14pm**  **;;7.0;OUTPATIENT PHARMACY;\*\*467\*\*;DEC 1997;Build 46**  **;**  **; ICR #4677 = $$CREATE^XUSAP (API for Application Proxy)**  **; ICR #10141 = BMES^XPDUTL & MES^XPDUTL**  **;**  **; Application Proxy name = "PSOAPPLICATIONPROXY,PSO"**  **; Secondary Menu Option name = "PSO WEB SERVICES OPTION"**  **;**  **RUNALL ; Entry point in the (Patch - PSO\*7.0\*467)**  **D BMSG("Starting Post-Init")**  **D PROXY**  **D MSG("Post-Init Complete")**  **D APRIND**  **D WSERV**  **Q**  **;**  **PROXY ; Create an Application Proxy for PSO application**  **N X**  **S X=$$CREATE^XUSAP("PSOAPPLICATIONPROXY,PSO","","PSO WEB SERVICES OPTIO**  **N")**  **;**  **I +X=0 D Q**  **.D BMSG(" Application Proxy User - 'PSOAPPLICATIONPROXY,PSO'")**  **.D MSG(" already exists in the NEW PERSON file (#200)"),MSG("")**  **;**  **I +X=-1 D Q**  **.D BMSG(" Application Proxy User - 'PSOAPPLICATIONPROXY,PSO'")**  **.D MSG(" Unsuccessful; could not create Application Proxy User")**  **.D MSG(" OR error in call to UPDATE^DIE)"),MSG("")**  **;**  **D BMSG(" \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***  **\*\*\*\*\*")**  **D MSG(" \*\* Application Proxy User - 'PSOAPPLICATIONPROXY,PSO' = creat**  **ed \*\*")**  **D MSG(" \*\* Secondary Menu Option - 'PSO WEB SERVICES OPTION' = linked**  **\*\*")**  **D MSG(" \*\* to the Application Proxy.**  **\*\*")**  **D MSG(" \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***  **\*\*\*\*\*")**  **D MSG("")**  **Q**  **;**  **; A message is also recorded in INSTALL file**  **; (#9.7) entry for the installation.**  **;**  **; Output a message.**  **MSG(MSG) ; Integration Agreement #10141**  **D MES^XPDUTL(MSG)**  **Q**  **;**  **; Output a message with a blank line added.**  **BMSG(MSG) ; Integration Agreement #10141**  **D BMES^XPDUTL(MSG)**  **Q**  **; create the APR index**  **APRIND ;**  **N DIK,DA**  **S DIK="^PSDRUG(",DIK(1)="22^APR"**  **D ENALL^DIK**  **Q**  **; create web server and service**  **WSERV ;**  **N FDA,WSIEN,NWSIEN,WSERVIEN,WSRVIEN**  **; if the web service already exists, this has been configured**  **I $$FIND1^DIC(18.02,,"B","PSO ERX WEB SERVICE") Q**  **; set up the web service**  **S FDA(18.02,"+1,",.01)="PSO ERX WEB SERVICE"**  **S FDA(18.02,"+1,",.02)=2**  **S FDA(18.02,"+1,",200)="/INB-ERX/"**  **D UPDATE^DIE(,"FDA","WSIEN") K FDA**  **S NWSIEN=$O(WSIEN(0)),NWSIEN=$G(WSIEN(NWSIEN)) Q:'NWSIEN**  **S FDA(18.12,"+1,",.01)="PSO WEB SERVER"**  **S FDA(18.12,"+1,",.03)=80**  **;S FDA(18.12,"+1,",.04)="SERVER.aac.DOMAIN"**  **S FDA(18.12,"+1,",.06)=1**  **S FDA(18.12,"+1,",.07)=30**  **S FDA(18.12,"+1,",1.01)=0**  **D UPDATE^DIE(,"FDA","WSERVIEN") K FDA**  **S WSRVIEN=$O(WSERVIEN(0)) Q:'WSRVIEN**  **S WSRVIEN=$G(WSERVIEN(WSRVIEN)) Q:'WSRVIEN**  **S FDA(18.121,"+1,"\_WSRVIEN\_",",.01)=NWSIEN**  **S FDA(18.121,"+1,"\_WSRVIEN\_",",.06)=1**  **D UPDATE^DIE(,"FDA",,"ERR") K FDA**  **Q**  **CALL ;**  **N I,X**  **F I=52.46,52.47,52.48,52.49 D**  **.S X=0 F S X=$O(^PS(I,X)) Q:'X D**  **..S FDA(I,X\_",",.01)="@"**  **D FILE^DIE(,"FDA","FERR") K FDA**  **Q** |

##### Templates

This section outlines the templates developed for the Inbound eRx functionality.

Table 52: Template: PSO ERX DRUG VALIDATION

| Templates | Description | | | | |
| --- | --- | --- | --- | --- | --- |
| Template Name | PSO ERX DRUG VALIDATION | | | | |
| Enhancement Category | New | Modify | | Delete | No Change |
| Template Type | Sort | Input | | Print | Other |
| Related Options | PSO ERX VALIDATION MENU | | | | |
| **Related Routines** | **Routines “Called By”** | | **Routines “Called”** | | |
|  |  | |  | | |
| Routines | Description | | | | |
| Data Dictionary (DD) References |  | | | | |
| Global References |  | | | | |

Table 53: Template: PSO ERX HOLDING QUEUE

| Templates | Description | | | | |
| --- | --- | --- | --- | --- | --- |
| Template Name | PSO ERX HOLDING QUEUE | | | | |
| Enhancement Category | New | Modify | | Delete | No Change |
| Template Type | Sort | Input | | Print | Other |
| Related Options | PSO ERX VALIDATION MENU | | | | |
| **Related Routines** | **Routines “Called By”** | | **Routines “Called”** | | |
|  |  | |  | | |
| Routines | Description | | | | |
| Data Dictionary (DD) References |  | | | | |
| Global References |  | | | | |

Table 54: Template: PSO ERX HQ DISPLAY

| Templates | Description | | | | |
| --- | --- | --- | --- | --- | --- |
| Template Name | PSO ERX HQ DISPLAY | | | | |
| Enhancement Category | New | Modify | | Delete | No Change |
| RSD |  | | | | |
| Template Type | Sort | Input | | Print | Other |
| Related Options |  | | | | |
| **Related Routines** | **Routines “Called By”** | | **Routines “Called”** | | |
|  |  | |  | | |
| Routines | Description | | | | |
| Data Dictionary (DD) References |  | | | | |
| Global References |  | | | | |

Table 55: Template: PSO ERX PATIENT VALIDATION

| Templates | Description | | | | |
| --- | --- | --- | --- | --- | --- |
| Template Name | PSO ERX PATIENT VALIDATION | | | | |
| Enhancement Category | New | Modify | | Delete | No Change |
| RSD |  | | | | |
| Template Type | Sort | Input | | Print | Other |
| Related Options |  | | | | |
| **Related Routines** | **Routines “Called By”** | | **Routines “Called”** | | |
|  |  | |  | | |
| Routines | Description | | | | |
| Data Dictionary (DD) References |  | | | | |
| Global References |  | | | | |

Table 56: Template: PSO ERX PROVIDER VALIDATION

| Templates | Description | | | | |
| --- | --- | --- | --- | --- | --- |
| Template Name | PSO ERX PROVIDER VALIDATION | | | | |
| Enhancement Category | New | Modify | | Delete | No Change |
| RSD |  | | | | |
| Template Type | Sort | Input | | Print | Other |
| Related Options |  | | | | |
| **Related Routines** | **Routines “Called By”** | | **Routines “Called”** | | |
|  |  | |  | | |
| Routines | Description | | | | |
| Data Dictionary (DD) References |  | | | | |
| Global References |  | | | | |

##### Bulletins

This section will be added in a future version of this document, where applicable.

Table 57: Bulletins

| Bulletins | Description | | | | |
| --- | --- | --- | --- | --- | --- |
| Bulletin Name |  | | | | |
| Enhancement Category | New | Modify | | Delete | No Change |
| RTM |  | | | | |
| Related Routines | Routines “Called By” | | Routines “Called” | | |
|  |  | |  | | |
| Routines | Description | | | | |
| Mail Subject |  | | | | |
| Mail Group |  | | | | |
| Parameters |  | | | | |
| Data Dictionary (DD) References |  | | | | |

##### Data Entries Affected by the Design

This section will be added in a future version of this document, where applicable.

Table 58: Data Entries Affected by the Design

| Field Name | Current Value | New Value |
| --- | --- | --- |
| TBD |  |  |

##### Unique Record(s)

This section will be added in a future version of this document, where applicable.

Table 59: Unique Record ID

| Field Name(s) | Current Value | New Value |
| --- | --- | --- |
| TBD |  |  |

##### File or Global Size Changes

The file/global size related to the Inbound eRx design will be added in a future version of this document. The initial Capacity Planning for storage needs, memory, Central Processing Unit (CPU) requirements was completed by the Inbound eRx and the Enterprise Infrastructure Engineering (EIE) teams as per the application requirements for maintaining optimal performance.

Table 60: File or Global Size Changes

| File/Global Name(s) | Estimated Increase | Estimated Decrease |
| --- | --- | --- |
| TBD |  |  |

##### Mail Groups

This section outlines the mail groups affected by the Inbound eRx functionality being designed. Mail groups will be added in a future version of this document.

Table 61: Mail Groups

| Mail Groups | Activities | | | | | |
| --- | --- | --- | --- | --- | --- | --- |
| Mail Group Name | TBD | | | | | |
| Enhancement Category | New | | Modify | | Delete | No Change |
| Related Options |  | | | | | |
| Related Routines | Routines “Called By” | | | Routines “Called” | | |
|  |  | | |  | | |
| Mail Groups | Instructions | | | | | |
| Data Dictionary (DD) References |  | | | | | |
| Related Protocols |  | | | | | |
| Mail Group Description |  | | | | | |
| Self-Enrollment Allowed | Yes | No | | | | |
| Type | Public | Private | | | | |

##### Security Keys

This section outlines the security keys affected by the Inbound eRx functionality being designed. Additional details on the security keys will be added in a future version of this document once finalized by PBM.

Table 62: Security Key: PSO ERX RPH

| Security Keys | Activities | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Security Key Name | PSO ERX RPH | | | | | | | | |
| Enhancement Category | New | | Modify | | | Delete | | No Change | |
| Related Options |  | | | | | | | | |
| Related Routines | Routines “Called By” | | | | Routines “Called” | | | | |
|  |  | | | |  | | | | |
| Security Keys | Activities | | | | | | | | |
| Data Passing | Input | Output | | Both | | | Global Reference | | Local Reference |
| Security Key Description | This key identifies an eRx pharmacist. | | | | | | | | |
| Subordinate Keys |  | | | | | | | | |
| Mutually Exclusive Keys |  | | | | | | | | |
| Granting Condition Logic |  | | | | | | | | |
| Current Logic | | | | | | | | | |
|  | | | | | | | | | |
| Modified Logic (Changes are in bold) | | | | | | | | | |
|  | | | | | | | | | |
| Security Keys | Activities | | | | | | | | |
| Hierarchical Precedence |  | | | | | | | | |

Table 63: Security Key: PSO ERX VIEW

| Security Keys | Activities | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Security Key Name | PSO ERX VIEW | | | | | | | | |
| Enhancement Category | New | | Modify | | | Delete | | No Change | |
| Related Options |  | | | | | | | | |
| Related Routines | Routines “Called By” | | | | Routines “Called” | | | | |
|  |  | | | |  | | | | |
| Security Keys | Activities | | | | | | | | |
| Data Passing | Input | Output | | Both | | | Global Reference | | Local Reference |
| Security Key Description | Read only access to eRxs in the eRx Holding Queue | | | | | | | | |
| Subordinate Keys |  | | | | | | | | |
| Mutually Exclusive Keys |  | | | | | | | | |
| Granting Condition Logic |  | | | | | | | | |
| Current Logic | | | | | | | | | |
|  | | | | | | | | | |
| Modified Logic (Changes are in bold) | | | | | | | | | |
|  | | | | | | | | | |
| Security Keys | Activities | | | | | | | | |
| Hierarchical Precedence |  | | | | | | | | |

Table 64: Security Key: PSO ERX TECH

| Security Keys | Activities | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Security Key Name | PSO ERX TECH | | | | | | | | |
| Enhancement Category | New | | Modify | | | Delete | | No Change | |
| Related Options |  | | | | | | | | |
| Related Routines | Routines “Called By” | | | | Routines “Called” | | | | |
|  |  | | | |  | | | | |
| Security Keys | Activities | | | | | | | | |
| Data Passing | Input | Output | | Both | | | Global Reference | | Local Reference |
| Security Key Description | This security key identifies an eRx Technician. | | | | | | | | |
| Subordinate Keys |  | | | | | | | | |
| Mutually Exclusive Keys |  | | | | | | | | |
| Granting Condition Logic |  | | | | | | | | |
| Current Logic | | | | | | | | | |
|  | | | | | | | | | |
| Modified Logic (Changes are in bold) | | | | | | | | | |
|  | | | | | | | | | |
| Security Keys | Activities | | | | | | | | |
| Hierarchical Precedence |  | | | | | | | | |

Table 65: Security Key: PSO ERX MGR

| Security Keys | Activities | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Security Key Name | PSO ERX MGR | | | | | | | | |
| Enhancement Category | New | | Modify | | | Delete | | No Change | |
| Related Options |  | | | | | | | | |
| Related Routines | Routines “Called By” | | | | Routines “Called” | | | | |
|  |  | | | |  | | | | |
| Security Keys | Activities | | | | | | | | |
| Data Passing | Input | Output | | Both | | | Global Reference | | Local Reference |
| Security Key Description | TBD | | | | | | | | |
| Subordinate Keys |  | | | | | | | | |
| Mutually Exclusive Keys |  | | | | | | | | |
| Granting Condition Logic |  | | | | | | | | |
| Current Logic | | | | | | | | | |
|  | | | | | | | | | |
| Modified Logic (Changes are in bold) | | | | | | | | | |
|  | | | | | | | | | |
| Security Keys | Activities | | | | | | | | |
| Hierarchical Precedence |  | | | | | | | | |

Table 66: Security Key: PSO ERX PBM

| Security Keys | Activities | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Security Key Name | PSO ERX PBM | | | | | | | | |
| Enhancement Category | New | | Modify | | | Delete | | No Change | |
| Related Options |  | | | | | | | | |
| Related Routines | Routines “Called By” | | | | Routines “Called” | | | | |
|  |  | | | |  | | | | |
| Security Keys | Activities | | | | | | | | |
| Data Passing | Input | Output | | Both | | | Global Reference | | Local Reference |
| Security Key Description | TBD | | | | | | | | |
| Subordinate Keys |  | | | | | | | | |
| Mutually Exclusive Keys |  | | | | | | | | |
| Granting Condition Logic |  | | | | | | | | |
| Current Logic | | | | | | | | | |
|  | | | | | | | | | |
| Modified Logic (Changes are in bold) | | | | | | | | | |
|  | | | | | | | | | |
| Security Keys | Activities | | | | | | | | |
| Hierarchical Precedence |  | | | | | | | | |

##### Options

This section lists the new VistA Options developed for the Inbound eRx Holding Queue and other options modified for the Inbound eRx Holding Queue.

Table 67: Option: PSO ERX FINISH

| Options | Activities | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Option Name | PSO ERX FINISH | | | | | | | |
| Enhancement Category | New | | Modify | | Delete | | No Change | |
| Associated Menu Options that will invoke this reference |  | | | | | | | |
| Data Passing | Input | Output | | Both | | Global Reference | | Local Reference |
| Menu Text Description | Complete Orders from eRx  This menu option allows users to complete incoming eRx prescriptions. | | | | | | | |
| 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) | | | | | | | | |
| **D EN^PSOERX** | | | | | | | | |

Table 68: Option: PSO RX

| Options | Activities | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Option Name | PSO RX | | | | | | | |
| Enhancement Category | New | | Modify | | Delete | | No Change | |
| Associated Menu Options that will invoke this reference |  | | | | | | | |
| Data Passing | Input | Output | | Both | | Global Reference | | Local Reference |
| Menu Text Description | Rx Prescriptions menu modified to include PSO ERX FINISH FERX. | | | | | | | |
| 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) | | | | | | | | |
|  | | | | | | | | |

Table 69: Option: PSO WEB SERVICES OPTION

| Options | Activities | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Option Name | PSO WEB SERVICES OPTION | | | | | | | |
| Enhancement Category | New | | Modify | | Delete | | No Change | |
| Associated Menu Options that will invoke this reference |  | | | | | | | |
| Data Passing | Input | Output | | Both | | Global Reference | | Local Reference |
| Menu Text Description | Inbound eRx Web Services Option  This option is required by the Kernel Broker to give access to the RPCs used by the Inbound eRx team. | | | | | | | |
| Option Type | Edit | | Print | | Menu | | Inquire | |
| Action | | Run Routine | | Other | |  | |
| Associated RPCs | PSOERXA0 DRGMTCH; PSOERXA0 PRVMTCH; PSOERXA1 INCERX | | | | | | | |
| Option Definition |  | | | | | | | |
| Current Entry Action Logic | | | | | | | | |
|  | | | | | | | | |
| Modified Entry Action Logic (Changes are in bold) | | | | | | | | |
|  | | | | | | | | |

##### Protocols

This section outlines the protocols created by the Inbound eRx functionality. A protocol is a single entry point referencing multiple routine entry points to execute several inter-related, required processes which perform specific functions.

Table 70: Protocol: PSO ERX ACCEPT ERX

| Protocols | Activities | | | |
| --- | --- | --- | --- | --- |
| Protocol Name | PSO ERX ACCEPT ERX | | | |
| Enhancement Category | New | Modify | Delete | No Change |
| Associated Protocols |  | | | |
| Data Passing | Input  Output  Both  Global Reference  Local Reference | | | |
| Item Text Description | Accept eRx | | | |
| Protocol Type | Action  Menu  Protocol  Protocol Menu  Limited Protocol  Extended Action  Dialog  Other | | | |
| Screen | I $D(^XUSEC("PSO ERX RPH",DUZ)) | | | |

| Current Entry Action Logic |
| --- |
|  |
| Modified Entry Action Logic (Changes are in bold) |
| **D SETUP^PSOERX1B** |

Table 71: Protocol: PSO ERX ACCEPT VALIDATION

| Protocols | Activities | | | |
| --- | --- | --- | --- | --- |
| Protocol Name | PSO ERX ACCEPT VALIDATION | | | |
| Enhancement Category | New | Modify | Delete | No Change |
| Associated Protocols |  | | | |
| Data Passing | Input  Output  Both  Global Reference  Local Reference | | | |
| Item Text Description | Accept Validation | | | |
| Protocol Type | Action  Menu  Protocol  Protocol Menu  Limited Protocol  Extended Action  Dialog  Other | | | |
| Screen |  | | | |

| Current Entry Action Logic |
| --- |
|  |
| Modified Entry Action Logic (Changes are in bold) |
| **D ACVAL^PSOERX1B(PSOIEN,EDTYP)** |

Table 72: Protocol: PSO ERX DISPLAY MENU

| Protocols | Activities | | | |
| --- | --- | --- | --- | --- |
| Protocol Name | PSO ERX DISPLAY MENU | | | |
| Enhancement Category | New | Modify | Delete | No Change |
| Associated Protocols |  | | | |
| Data Passing | Input  Output  Both  Global Reference  Local Reference | | | |
| Item Text Description | eRx Display | | | |
| 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) |
| **ITEM: PSO ERX VALIDATE PATIENT MNEMONIC: VP**  **SEQUENCE: 1**  **ITEM: PSO ERX VALIDATE PROVIDER MNEMONIC: VM**  **SEQUENCE: 4**  **ITEM: PSO ERX VALIDATE DRUG MNEMONIC: VD**  **SEQUENCE: 7**  **ITEM: PSO ERX PRINT MNEMONIC: P**  **SEQUENCE: 2**  **ITEM: PSO ERX REJECT MNEMONIC: RJ**  **SEQUENCE: 5**  **ITEM: PSO ERX ACCEPT ERX MNEMONIC: AC**  **SEQUENCE: 8**  **ITEM: PSO ERX HOLD MNEMONIC: H**  **SEQUENCE: 3**  **ITEM: PSO ERX UNHOLD MNEMONIC: UH**  **SEQUENCE: 6**  **ITEM: PSO ERX REMOVE MNEMONIC: RM**  **SEQUENCE: 9**  **SCREEN: I 1 X:$D(^ORD(101,+$P(^ORD(101,DA(1),10,DA,0),"""^""",1),24)) ^(24)**  **HEADER: D SHOW^VALM W ""** |

Table 73: Protocol: PSO ERX EDIT

| Protocols | Activities | | | |
| --- | --- | --- | --- | --- |
| Protocol Name | PSO ERX EDIT | | | |
| Enhancement Category | New | Modify | Delete | No Change |
| Associated Protocols |  | | | |
| Data Passing | Input  Output  Both  Global Reference  Local Reference | | | |
| Item Text Description | Edit eRx | | | |
| 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) |
| **D EDIT^PSOERX1A(EDTYP)** |

Table 74: Protocol: PSO ERX HOLD

| Protocols | Activities | | | |
| --- | --- | --- | --- | --- |
| Protocol Name | PSO ERX HOLD | | | |
| Enhancement Category | New | Modify | Delete | No Change |
| Associated Protocols |  | | | |
| Data Passing | Input  Output  Both  Global Reference  Local Reference | | | |
| Item Text Description | Hold eRx | | | |
| 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) |
| **D HOLD^PSOERXH1** |

Table 75: Protocol: PSO ERX HQ MENU

| Protocols | Activities | | | |
| --- | --- | --- | --- | --- |
| Protocol Name | PSO ERX HQ MENU | | | |
| Enhancement Category | New | Modify | Delete | No Change |
| Associated Protocols |  | | | |
| Data Passing | Input  Output  Both  Global Reference  Local Reference | | | |
| Item Text Description | ERX QUEUE MENU  This is the eRx Holding Queue Menu. | | | |
| 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) |
| **ITEM: PSO ERX HQ SELECT MNEMONIC: SI**  **SEQUENCE: 1**  **ITEM: PSO ERX HQ SEARCH MNEMONIC: SR**  **SEQUENCE: 2**  **ITEM: PSO ERX HQ SORT MNEMONIC: SO**  **SEQUENCE: 3**  **SCREEN: I 1 X:$D(^ORD(101,+$P(^ORD(101,DA(1),10,DA,0),"""^""",1),24)) ^(24)**  **HEADER: D SHOW^VALM S XQORM("#")=$O(^ORD(101,"B","PSO ERX SELECT BY NUMBER",0)**  **)\_"^1:"\_VALMCNT MENU PROMPT: Select Action:**  **TIMESTAMP: 64411,28968** |

Table 76: Protocol: PSO ERX HQ SEARCH

| Protocols | Activities | | | |
| --- | --- | --- | --- | --- |
| Protocol Name | PSO ERX HQ SEARCH | | | |
| Enhancement Category | New | Modify | Delete | No Change |
| Associated Protocols |  | | | |
| Data Passing | Input  Output  Both  Global Reference  Local Reference | | | |
| Item Text Description | Search Queue  This protocol allows a user to search the eRx Holding Queue. | | | |
| Protocol Type | Action  Menu  Protocol  Protocol Menu  Limited Protocol  Extended Action  Dialog  Other | | | |
| Screen | I '$G(PSOSRCH) | | | |

| Current Entry Action Logic |
| --- |
|  |
| Modified Entry Action Logic (Changes are in bold) |
| **D SEARCH^PSOERX** |

Table 77: Protocol: PSO ERX HQ SELECT

| Protocols | Activities | | | |
| --- | --- | --- | --- | --- |
| Protocol Name | PSO ERX HQ SELECT | | | |
| Enhancement Category | New | Modify | Delete | No Change |
| Associated Protocols |  | | | |
| Data Passing | Input  Output  Both  Global Reference  Local Reference | | | |
| Item Text Description | Select Item | | | |
| 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) |
| **D SI^PSOERX1A** |

Table 78: Protocol: PSO ERX HQ SORT

| Protocols | Activities | | | |
| --- | --- | --- | --- | --- |
| Protocol Name | PSO ERX HQ SORT | | | |
| Enhancement Category | New | Modify | Delete | No Change |
| Associated Protocols |  | | | |
| Data Passing | Input  Output  Both  Global Reference  Local Reference | | | |
| Item Text Description | Sort Entries  This action protocol allows a user to sort the eRx entries. | | | |
| Protocol Type | Action  Menu  Protocol  Protocol Menu  Limited Protocol  Extended Action  Dialog  Other | | | |
| Asscoiated Routine |  | | | |

| Current Entry Action Logic |
| --- |
|  |
| Modified Entry Action Logic (Changes are in bold) |
| **I '$G(PSOSRT)** |

Table 79: Protocol: PSO ERX PRINT

| Protocols | Activities | | | |
| --- | --- | --- | --- | --- |
| Protocol Name | PSO ERX PRINT | | | |
| Enhancement Category | New | Modify | Delete | No Change |
| Associated Protocols |  | | | |
| Data Passing | Input  Output  Both  Global Reference  Local Reference | | | |
| Item Text Description | Print  NOTE: This action will be disabled in the first production release in 2017. | | | |
| 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) |
| TBD |

Table 80: Protocol: PSO ERX REMOVE

| Protocols | Activities | | | |
| --- | --- | --- | --- | --- |
| Protocol Name | PSO ERX REMOVE | | | |
| Enhancement Category | New | Modify | Delete | No Change |
| Associated Protocols |  | | | |
| Data Passing | Input  Output  Both  Global Reference  Local Reference | | | |
| Item Text Description | Remove eRx | | | |
| Protocol Type | Action  Menu  Protocol  Protocol Menu  Limited Protocol  Extended Action  Dialog  Other | | | |
| Screen | I $D(^XUSEC("PSO ERX RPH",DUZ)) | | | |

| Current Entry Action Logic |
| --- |
|  |
| Modified Entry Action Logic (Changes are in bold) |
| **D REM^PSOERX1B** |

Table 81: Protocol: PSO ERX SELECT BY NUMBER

| Protocols | Activities | | | |
| --- | --- | --- | --- | --- |
| Protocol Name | PSO ERX SELECT BY NUMBER | | | |
| Enhancement Category | New | Modify | Delete | No Change |
| Associated Protocols |  | | | |
| Data Passing | Input  Output  Both  Global Reference  Local Reference | | | |
| Item Text Description | Select eRx by the record number. | | | |
| 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) |
| **D SBN^PSOERX1A** |

Table 82: Protocol: PSO ERX UNHOLD

| Protocols | Activities | | | |
| --- | --- | --- | --- | --- |
| Protocol Name | PSO ERX UNHOLD | | | |
| Enhancement Category | New | Modify | Delete | No Change |
| Associated Protocols |  | | | |
| Data Passing | Input  Output  Both  Global Reference  Local Reference | | | |
| Item Text Description | Un-Hold eRx | | | |
| Protocol Type | Action  Menu  Protocol  Protocol Menu  Limited Protocol  Extended Action  Dialog  Other | | | |
| Screen | I $D(^XUSEC("PSO ERX RPH",DUZ)) | | | |

| Current Entry Action Logic |
| --- |
|  |
| Modified Entry Action Logic (Changes are in bold) |
| **D UNHOLD^PSOERXH** |

Table 83: Protocol: PSO ERX VALIDATE DRUG

| Protocols | Activities | | | |
| --- | --- | --- | --- | --- |
| Protocol Name | PSO ERX VALIDATE DRUG | | | |
| Enhancement Category | New | Modify | Delete | No Change |
| Associated Protocols |  | | | |
| Data Passing | Input  Output  Both  Global Reference  Local Reference | | | |
| Item Text Description | Validate Drug | | | |
| 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) |
| **D DRUG^PSOERX1A** |

Table 84: Protocol: PSO ERX VALIDATE PATIENT

| Protocols | Activities | | | |
| --- | --- | --- | --- | --- |
| Protocol Name | PSO ERX VALIDATE PATIENT | | | |
| Enhancement Category | New | Modify | Delete | No Change |
| Associated Protocols |  | | | |
| Data Passing | Input  Output  Both  Global Reference  Local Reference | | | |
| Item Text Description | Validate Patient  This protocol hangs off of the PSO ERX HQ DISPLAY PROTOCOL as an action. This protocol allows the user to manually validate the patient for a given eRx prescription. | | | |
| 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) |
| **D PAT^PSOERX1A** |

Table 85: Protocol: PSO ERX VALIDATE PROVIDER

| Protocols | Activities | | | |
| --- | --- | --- | --- | --- |
| Protocol Name | PSO ERX VALIDATE PROVIDER | | | |
| Enhancement Category | New | Modify | Delete | No Change |
| Associated Protocols |  | | | |
| Data Passing | Input  Output  Both  Global Reference  Local Reference | | | |
| Item Text Description | Validate Provider | | | |
| 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) |
| **D PROV^PSOERX1A** |

Table 86: Protocol: PSO ERX VALIDATION MENU

| Protocols | Activities | | | |
| --- | --- | --- | --- | --- |
| Protocol Name | PSO ERX VALIDATION MENU | | | |
| Enhancement Category | New | Modify | Delete | No Change |
| Associated Protocols |  | | | |
| Data Passing | Input  Output  Both  Global Reference  Local Reference | | | |
| Item Text Description | Validation Menu | | | |
| 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) |
| **ITEM: PSO ERX PRINT MNEMONIC: P**  **SEQUENCE: 1**  **ITEM: PSO ERX REJECT MNEMONIC: RJ**  **ITEM: PSO ERX ACCEPT VALIDATION MNEMONIC: AV**  **SEQUENCE: 7**  **ITEM: PSO ERX EDIT MNEMONIC: E**  **SEQUENCE: 2**  **ITEM: PSO ERX HOLD MNEMONIC: H**  **SEQUENCE: 5**  **ITEM: PSO ERX UNHOLD MNEMONIC: UH**  **SEQUENCE: 8**  **HEADER: D SHOW^VALM S XQORM("#")=$O(^ORD(101,"B","PSO ERX SELECT BY NUMBER",0)**  **)\_"^1:12"** |

##### Remote Procedure Call (RPC)

This section outlines the Remote Procedure Calls (RPC) affected by the functionality being designed.

Table 87: RPC: PSOERXA0 DRGMTCH

| RPCs | Activities | | |
| --- | --- | --- | --- |
| Name | PSOERXA0 DRGMTCH | | |
| TAG^RTN | DRGMTCH | | |
| Routine | PSOERXA0 | | |
| Input Parameters | NDCUPN; DGDESC | | |
| Input Description | Both parameters are 'optional', but to receive any legitimate results, at least one of the two must be passed in.    NDCUPN - NDC or UPN value to be match to the NDC/UPN file (#50.67).  If NDC, pass "N^ndc value"  If UPN, pass "U^upn value"    DGDESC - Textual drug description. Pass in the drug description or 'drug name'. (ex. ACETAMINOPHEN 400 MG TABLET). | | |
| Results Array | Single Value | Array | Word Processing |
| Global Array | Global Instance |  |
| Description | This RPC is used to match incoming eRx drugs/supply items to the VistA equivalent drug/supply items. | | |

Table 88: RPC: PSOERXA0 PRVMTCH

| RPCs | Activities | | |
| --- | --- | --- | --- |
| Name | PSOERXA0 PRVMTCH  Provides logic for matching a provider identified for an incoming eRx prescription. | | |
| TAG^RTN | PRVMTCH | | |
| Routine | PSOERXA0 | | |
| Input Parameters | NPI; DEA; CSUB | | |
| Results Array | Single Value | Array | Word Processing |
| Global Array | Global Instance |  |
| Description | Provides logic for matching a provider identified for an incoming eRx prescription. | | |

Table 89: RPC: PSOERXA1 INCERX

| RPCs | Activities | | |
| --- | --- | --- | --- |
| Name | PSOERXA1 INCERX | | |
| TAG^RTN | INCERX | | |
| Input Parameters | XML; PRCHK; PACHK  XML: The full XML text, as a 'stream' or 'literal'.  PRCHK: This is the provider check results from the eRx middleware.  PACHK: This is the patient check information from the eRx middleware. | | |
| Results Array | Single Value | Array | Word Processing |
| Global Array | Global Instance |  |
| Description | This RPC receives and processes an incoming ERX XML message, provider check information, and patient check information. | | |

##### Constants Defined in Interface

This section will be added in a future version of this document, where applicable.

Table 90: Constants Defined in Interface

| Name | Description |
| --- | --- |
| TBD |  |

##### Variables Defined in Interface

This section will be added in a future version of this document, where applicable.

Table 91: Variables Defined in Interface

| Name | Type | Description |
| --- | --- | --- |
|  |  |  |

##### Types Defined in Interface

This section will be added in a future version of this document, where applicable.

Table 92: Types Defined in Interface

| Name | Type | Description |
| --- | --- | --- |
|  |  |  |

##### GUI

This section will be added in a future version of this document, where applicable.

Table 93: GUI

| Unit Name | Description |
| --- | --- |
|  |  |

##### GUI Classes

This section will be added in a future version of this document, where applicable.

Table 94: GUI Classes

| GUI Classes | Instructions |
| --- | --- |
| Class Name |  |
| Derived From Class |  |
| Purpose |  |

##### Current Form

As the Inbound eRx Holding Queue is new functionality, this section is not applicable.

##### Modified Form

This section will be added in a future version of this document, where applicable.

##### Components on Form

This section will be added in a future version of this document, where applicable.

Table 95: Components on Form

| Name | Type | Description |
| --- | --- | --- |
| TBD |  |  |

##### Events

This section will be added in a future version of this document, where applicable.

Table 96: Events

| Name | Type | Description |
| --- | --- | --- |
|  |  |  |

##### Methods

This section will be added in a future version of this document, where applicable.

Table 97: Methods

| Method Name | Procedure/Function | Description |
| --- | --- | --- |
|  |  |  |

##### Special References

This section will be added in a future version of this document, if applicable.

| Special Reference Name | Type | Description |
| --- | --- | --- |
|  |  |  |

##### Class Events

This section will be added in a future version of this document, if applicable.

Table 98: Class Events

| Name | Type | Description |
| --- | --- | --- |
|  |  |  |

##### Class Methods

This section will be added in a future version of this document, if applicable.

Table 99: Class Methods

| Name | Procedure/Function | Description |
| --- | --- | --- |
|  |  |  |

##### Class Properties

This section will be added in a future version of this document, ifapplicable.

Table 100: Class Properties

| Class Properties Name | Type | Visibility | Description |
| --- | --- | --- | --- |
|  |  |  |  |

##### Uses Clause

This section will be added in a future version of this document, if applicable.

##### Forms

This section is not applicable to Inbound eRx.

##### Functions

This section will be added in a future version of this document, if applicable.

Table 101: Functions

| Function Name | Activities | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Short Description |  | | | | | | | | |
| Enhancement Category | New | | Modify | | Delete | | | No Change | |
| Related Options |  | | | | | | | | |
| Related Routines | Routines “Called By” | | | | | Routines “Called” | | | |
|  |  | | | | |  | | | |
| Function Name | Activities | | | | | | | | |
| 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) | | | | | | | | | |
|  | | | | | | | | | |

##### Dialog

This section will be added in a future version of this document, where applicable.

Table 102: Dialog

| Dialog | Instructions | | | |
| --- | --- | --- | --- | --- |
| Dialog Message (Description) |  | | | |
| Enhancement Category | New | Modify | Delete | No Change |
| Dialog Message (Description) Condition |  | | | |
| Current Dialog Message (Description) |  | | | |
| Modified Dialog Message (Description)  (Changes are in bold) |  | | | |

##### Help Frame

This section will be added in a future version of this document.

Table 103: Help Frame

| Help Frame | Description | | | |
| --- | --- | --- | --- | --- |
| Help Frame Text |  | | | |
| Enhancement Category | New | Modify | Delete | No Change |
| Help Frame Text Calling Mechanism |  | | | |
| Current Help Frame Text | | | | |
|  | | | | |
| Modified Help Frame Text (Changes are in bold) | | | | |
|  | | | | |

##### HL7 Application Parameter

This section will be added in a future version of this document.

Table 104: HL7 Application Parameter

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

##### HL7 Logical Link

This section will be added in a future version of this document, if applicable.

Table 105: HL7 Logical Link

| HL7 Logical Link | Description | | | | |
| --- | --- | --- | --- | --- | --- |
| HL7 Logical Link Parameter Name |  | | | | |
| Enhancement Category | New | Modify | | Delete | No Change |
| Enhancement Category | Current | | Modified | | |
| Node |  | |  | | |
| Institution |  | |  | | |
| Domain |  | |  | | |
| Autostart |  | |  | | |
| Queue Size |  | |  | | |
| LLP Type |  | |  | | |

##### COTS Interface

This section will be added in a future version of this document, where applicable.

Table 106: COTS Interface

| COTS Interface | Description |
| --- | --- |
| Communication Method |  |
| Application Interface |  |

## Network Detailed Design

Refer to [Section 4.3](#_Network_Architecture) for the network detailed design.

## Security and Privacy

### Security

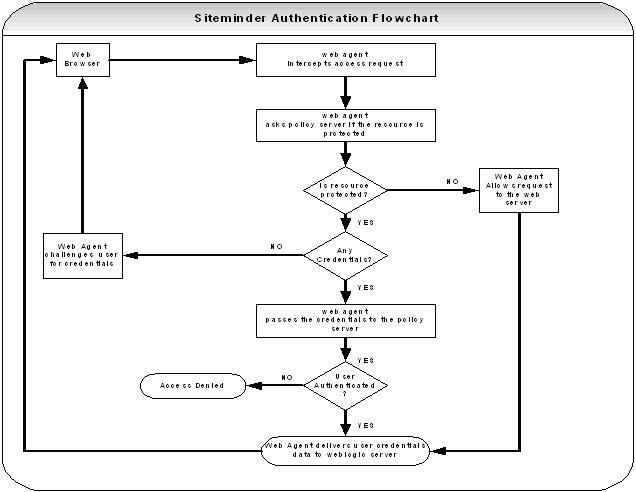
The Inbound eRx security controls have been implemented to comply with VA Handbook 6500. The following describes at a high-level the key security controls required for technical and operational implementation. For additional information related to full details of security, refer to the ATO documentation in Risk Vision.

| Security Control | Implementation |
| --- | --- |
| Access Control, Identification, Authentication, and Authorization | **NOTE**: Inbound eRx assumes that all users will be authenticated to the One VA network through VA SSOi service using their PIV card and Identity and Access Management (SAC) service using role-based (RBAC) access functionality.  **NOTE:** Inbound eRx assumes that external connections (i.e. prescription clearinghouse, such as Change Healthcare) will follow the DAS security controls.  Privileged users will access Inbound eRx through Enterprise Operations (EO) generated accounts using 9957 process, which will be specific to their function and are issued and monitored from EO.  Messages will be encrypted using two-way SSL at the transport/session layer and via IP/Port identification at the network layer.  Inbound eRx shall follow VA Authentication Federation Infrastructure (VAAFI) standards when interacting with the Master Veteran Index (MVI) and other VA interfaces. |
| Auditing | All auditing to include infrastructure and application will be monitored by EO. |
| Media Protection, Transmission Confidentiality and Integrity | ePHI/PII data stored in the eRx Processing Hub is encrypted using oracle’s Transparent Data Encryption (TDE) and protected as per the VA VM and EO policies.  Messages are encrypted using two-way SSL at the transport/session layer and via IP/Port identification at the network layer. |
| Contingency planning /Disaster Recovery | EO is responsible for the Infrastructure and Inbound eRx is responsible for the Application. |
| Incident Response, Security Alerting | EO is responsible for Incident response and will notify Inbound eRx if there are any incidents. |
| Maintenance | EO is responsible for the maintenance of the Inbound eRx infrastructure. Inbound eRx is responsible for the application maintenance. |
| Physical and Environmental | EO is responsible for the Physical and environmental security controls. |
| Boundary Protection, network | EO is responsible for the boundary protection using Load balancers, firewalls, etc. |
| Cryptography | DAS processing tier is responsible for all external to VA communications. DAS has implemented two-way SSL which will be decrypted at the Layer 7 and IB. Certificates are provided by VA Public Key Infrastructure office and are compliant with VA Handbook 6500.  Internal to VA Cryptography (including database) is provided by EO. |
| Patch Management, Vulnerability Assessments | EO is responsible for patch management of the infrastructure and Inbound eRx is responsible for the applications. Both Inbound eRx and EO CM processes are followed. Vulnerability scanning is the responsibility of the National Security Operations Center, and the remediation of findings is the responsibility of both EO and Inbound eRx. |

#### Authentication and Authorization

Administrator UI in eRx Processing Hub uses the VASingle Sign-On (SSOi) service and VA Identity Access Management (IAM) solution for authentication (SiteMinder) (Figure 37). Authorization is role-based. The eRx UI application extracts HTTP Header variables passed from SiteMinder to determine the VA User ID of the currently authentication user. The eRx Processing Hub maintains a mapping of VA User IDs to application roles in the VA\_USER table. Spring Security is used to implement authorization and protect resources

Figure 37: SiteMinder Authentication Flowchart



### Privacy

The Inbound eRx complies with all VA privacy policies. The following describes at a high-level the key privacy controls required for technical and operational implementation.

| Privacy Control | Implementation |
| --- | --- |
| Governance and Privacy | A governance program has been established. Please refer to governance documentation for compliance. |
| Privacy Impact and Risk Assessment | As part of the ATO, a Privacy Impact Assessment (PIA) and Risk Assessment were performed for Inbound eRx. |
| Privacy Enhanced System Design | The system design for Inbound eRx employs technologies and system capabilities that allow PII/PHI data to flow through an ESB to eliminate additional copies of this type of data, automate collection, use, retention, and disclosure of PII. |
| Data Integrity | Data Integrity is handled through two-way SSL, service IP/Port identification, and, in through IAM. |
| Data Retention and Disposal | For stored data, encryption methods will be deployed according to VA Handbook 6500. Currently, there are no defined data archiving requirements for Inbound eRx. |
| PII Used in Testing | No PII data is using in testing scenarios. |
| Incident Response | Incidents for PII are reported to the VA Privacy Officer. |

## Service Oriented Architecture / ESS Detailed Design

Inbound eRx is designed as a service oriented architecture. It involves loose coupling between services, service reusability, service autonomy, service abstraction and utilizes services that are self-contained. Service-orientation is independent of any vendor, product or technology.

This section provides details of provided and consumed services as follows:

### Service Description for MVI

Please refer to the following Rational Team Concert (RTC) folder in the Inbound ePrescribing team area for the E&E Service Description. For access to RTC, please contact the Rational Tools Team at

INB\_ERX Documentation > PHARM\_INB\_ERX\_Documentation > Design > Service Descriptions

This document is also available on the Inbound eRx SharePoint site at the following link:

### Service Description for E&E

Please refer to the following RTC folder in the Inbound ePrescribing team area for the E&E Service Description. For access to RTC, please contact the Rational Tools Team at

INB\_ERX Documentation > PHARM\_INB\_ERX\_Documentation > Design > Service Descriptions

This document is also available on the Inbound eRx SharePoint site at the following link:

### Service Design for Inbound eRx Process Hub Services

This section is not applicable to all of the Inbound eRx Processing Hub services (see below) due to the fact that they are not considered enterprise services. For more information on the Inbound eRx Processing Hub services, please refer to following links below.

* [PatientEligibility](#_PatientEligibility)
* [PrescriptionProviders](#_PrescriptionProviders)
* [PrescriptionDrugs](#_PrescriptionDrugs)
* [PharmacyTransfers](#_PharmacyTransfers)
* [VistAErxTransmissions](#_VistAErxTransmissions)
* [VistAErxStatuses](#_VistAErxStatuses)
* [ClinicalExchangePharmacyInformation](#_ClinicalExchangePharmacyInformation)

#### Introduction

N/A

##### Purpose and Scope of Service

N/A

##### Links to Other Documents

N/A

#### Service Details

N/A

##### Service Identification

N/A

| Service Attribute | Value |
| --- | --- |
| N/A | N/A |
| Overview | N/A |
| Version | N/A |
| Latest Status | N/A |
| Service Type | N/A |
| Architecture Layer | N/A |
| Business Domain | N/A |
| Service Domain | N/A |
| Business Organization and Owner | N/A |
| Technical Organization and Owner | N/A |
| Development Organization and Owner | N/A |
| Support Organization and Owner | N/A |
| Target Consumer Organization(s) and Owner(s) | N/A |

##### Service Versions

N/A

|  |  |  |
| --- | --- | --- |
| Version Numbers | Current Status of Version | A Brief Description of the change implemented in that version |
| N/A |  |  |
|  |  |  |
|  |  |  |

##### Summary of Design and Platform Details

N/A

###### SOA Pattern(s) Implemented

N/A

###### COTS Platform vendor names and versions for hosting platform

N/A

#### Dependencies

N/A

#### Service Design Details

N/A

##### Interface Technical Specs

N/A

###### Service Invocation Type

N/A

###### Service Interface Type

N/A

###### Service Name

N/A

###### Interface

N/A

###### End Points

N/A

###### Operations or Methods

N/A

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Operation Name | Inputs | Outputs | Transactional Qualities if relevant (Updating?, Atomic?, Can participate in transaction?) | Pre and Post Conditions | Exception (s) |
| N/A |  |  |  |  |  |
|  |  |  |  |  |  |

###### Message Schemas

N/A

##### Information Model

N/A

###### Class Diagram and Description of Entities Involved

N/A

###### Mappings from ELDM to Standards Based Schemas

N/A

##### Behavior Model (AKA Use Case Realization)

N/A

###### Use Cases (Use Case Model)

N/A

###### Interaction Diagrams

N/A

#### Gap Analysis

N/A

| Design Elements🡪  Policies / SLD elements etc.↓ | Design  Element A | Design  Element B | Design  Element C | Comment for non-conformance |
| --- | --- | --- | --- | --- |
| N/A |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

##### Variances from Enterprise Target Architecture

N/A

##### Variances from SLDs

N/A

##### Variances from Standards and Policies

N/A

##### Justification for Exceptions and Mitigation

N/A

# External System Interface Design

This section details interfaces external to the Inbound eRx system.

## Interface Architecture

The Inbound eRx Processing Hub will interface with a “clearinghouse” system hosted outside of the VA network in order to facilitate eRx message retrieval and delivery between non-VA prescribers and the VA Pharmacies. Change Healthcare is the clearinghouse provider used for this purpose in the current design. Change Healthcare provides commercial ePrescribing solutions, and for the purposes of the Inbound eRx implementation serves as a gateway to all ePrescribing providers nationwide. The Change Healthcare system will interface to the Inbound eRx Processing Hub through DAS as shown in Section 4.3. DAS and Change Healthcare will communicate using https requests over a secured network.

## Interface Detailed Design

Below is the description of the Change Healthcare connection, authentication, authorization and interface specifications. Recognizing the PHI and PII nature of the messages on its network, Change Healthcare will support and adhere to standard VA guidelines on message encryption and service authentication/authorization.

### Connectivity Methods

The recommended network connection method for eRx Network is HTTPS (an SSL Certificate is required). HTTPS is reliable and widely used for the exchange of information over TCP/IP.

### SSL Certificate

An SSL certificate/Fully qualified Domain Name is required if the connection will be made via a public HTTPS connection. The SSL certificate must meet the following guidelines:

1. The SSL certificate must have at least 1024-but key and must be issued from a trusted certificate authority (GeoTrust, GlobalSign etc.).
2. The SSL certificate cannot be self-signed.

### Timeout Limits

Timeout limits (maximum wait times) for:

* Submitted transactions – 30 seconds before timeout.
* Received transactions – 27 seconds before timeout.

### Two-way Mutual Certification Authentication

Change Healthcare will support mutual authentication between DAS and the Change Healthcare servers.

### Network Communications Header

Transactions sent across eRx Network must use the eRx Network Communications Header format to allow Change Healthcare to properly identify and route ePrescribing transactions.

This protocol can be either synchronous or asynchronous. It is possible to send each transaction via a separate socket connection if desired.

### eRx XML Wrapper

Transactions sent across eRx Network require a standard eRx XML wrapper. For XML transactions, the wrapper information is included in the XML header.

Snippet of XML Wrapper Code:

<?xml version="1.0" encoding="UTF-8"?>

<Message xmlns="https://erxpad.erxnetwork.com">

<Header>

<To/>

<From/>

<MessageID/>

<RelatesToMessageID/>

<SentTime/>

<Security/>

<Mailbox/>

<TestMessage/>

<DigitalSignature>

<DigestValue/>

<SignatureValue/>

<X509Data/>

</DigitalSignature>

</Header>

<erx>

<Application/>

<SendingEntity/>

<SendingEntityType/>

<ERXMessageID/>

<BodyType/>

<BodyVersion/>

<MessageType/>

<ControlledDrug/>

<CustomerID/>

</erx>

<Body/>

</Message>

The eRx XML wrapper consists of these elements:

Table 107: Header Segment

| **Element** | **Description** |
| --- | --- |
| <To> | Identifies the intended receiver of the message. Usually a pharmacy or prescriber.  **SCRIPT Reference: UIB-070-01 for EDIFACT**  **XML Reference: Header>To** |
| <From> | Identifies the sender of the message. Usually a pharmacy or prescriber. **SCRIPT Reference: UIB-060-01 for EDIFACT**  **XML Reference: Header>From** |
| <MessageID> | Unique message identifier used to track messages and identify duplicate messages.  **SCRIPT Reference: UIB-030-01 for EDIFACT**  **XML Reference: Header>MessageID** |
| <RelatesToMessageID> | Identifies previously sent messages related to the message being sent.  **SCRIPT Reference: UIB-030-02 for EDIFACT**  **XML Reference: Header>RelatesToMessageID** |
| <SentTime> | Date and time message was sent.  Expressed as an XML UTC time string.  Example: 2007-01-01T11:12:13Z  **SCRIPT Reference: UIB-080-01 and UIB-080-02 for EDIFACT**  **XML Reference: Header>SentTime** |
| <Security> | Not used. |
| <Mailbox> | Not used. |
| <TestMessage> | Indicates whether or not the message is a test.  0 = False  1 = True  **SCRIPT Reference: UIB-100 for EDIFACT**  **XML Reference: Header>TestMessage** |
| <DigitalSignature> | EPCS only – Digital code used to authenticate the identity of the sender.  The DigitalSignature is made up of these elements:  DigestValue  SignatureValue  X509Data  The data contained in each of these elements must be base64 encoded.  For additional EPCS and Digital Signature information and suggestions, see Steps for Creating a Digital Signature. |
| <DigestValue> | EPCS only – All required message fields encoded. |
| <SignatureValue> | EPCS only – DigestValue that has been signed with a Private Key, base64 encoded |
| <X509Data> | EPCS only – Base64-encoded raw bytes of the X509 certificate (which contains the Public Key related to the Private Key in the SignatureValue element). |

Table 108: eRx Segment

| **Element** | **Description** |
| --- | --- |
| <Application> | Set to “ESCRIPTS” |
| <SendingEntity> | Agreed upon value to be obtained by eRx. |
| <SendingEntityType> | Set to type of entity sending the message:  Pharmacy = “PHARMACY”  Prescriber = “CLINIC” |
| <ERXMessageID> | eRx Internal Message ID – eRx Network supplies this on responses or new messages it sends. |
| <BodyType> | Defines the format of the message inside the <Body>.  Set to the appropriate message format:  EDIFACT = “SCRIPT”  XML = “XML” |
| <BodyVersion> | Defines the version of the message being sent.  Example: “10.6” or “8.1” |
| <MessageType> | Message Function  **Script Reference: UIH-010-040**  **XML reference: Body>MessageType** |
| <ControlledDrug> | Not supported |
| <CustomerID> | ID provided by the customer.  eRx returns the provided CustomerID value in responses |

Table 109: Body Segment

| **Element** | **Description** |
| --- | --- |
| <Body> | Contains the entire EDIFACT or XML message.  Includes the UNA, UIB, and UIH segments |

eRx Network Certification can include the following transaction types:

* NEWRX/NewRx
* REFREQ/RefillRequest
* REFRES/RefillResponse
* CANRX/CancelRx (**NOTE:** Will not be delivered in 2017 production release.)
* CANRES/CancelRxResponse (**NOTE:** Will not be delivered in 2017 production release.)
* RXCHG/RxChange (**NOTE:** Will not be delivered in 2017 production release.)
* CHGRES/RxChangeResponse (**NOTE:** Will not be delivered in 2017 production release.)
* RXFILL/RxFill (**NOTE:** Will not be delivered in 2017 production release.)
* VERIFY/Verify
* ERROR/Error (synchronous and asynchronous)
* STATUS/Status
* GETMSG/GetMessage
* PASCHG/PasswordChange
* RESUPP/Resupply
* CENSUS/Census

**Sender and Receiver Fields**

The Sender and Receiver fields appear in all supported transaction types. These fields must contain specific values to ensure transactions are routed properly. The sender and receiver fields are determined and set by Change Healthcare during the implementation process.

# Human-Machine Interface

The Inbound eRx application is hosted on a WebLogic application server and provides a web-based front end for user control. This section details the design of this web-based user interface. The VistA OP enhancements will include new “roll and scroll” screens using List Manager as well as modifications to existing screens.

## Interface Design Rules

The layout of the Inbound eRx screens follows that of other VA web-based interfaces. All web pages in the eRx application must be 508 compliant. 508 compliance testing will be performed during development testing to ensure a readily 508 certified system. The system is designed to work with a screen resolution of 1280 X 1024 and to work with the Internet Explorer version 11 or greater or other TRM compliant web browsers. All the user interface mockups in this section are notional and may change in future builds.

## Inputs

All data input by the VA Pharmacists is collected from the web pages served by the eRx application. All the functionality defined for eRx is controlled via the eRx processing hub, eRx GUI screens and VistA OP. Only users authenticated with the VA’s SSOi service and authorized by the application are allowed access. Authorization is role-based and is supported by SiteMinder via SAML2. The eRx Processing Hub maintains a mapping of roles to application functions in the eRx\_Config table. Spring Security is used to implement authentication. Additionally VA Identity and Access Management (SAC) service using role based (RBAC) can be used for role based authentication.

## Outputs

All data output to the users is presented via web pages served by the eRx application and VistA OP.

## Navigation Hierarchy

This section focuses on the navigation hierarchy of the VistA OP Inbound eRx Holding Queue. Refer to Section 3.2.5 for the navigation heirarchy of the Inbound eRx web-based UI.

### Vista Outpatient Pharmacy eRx Holding Queue Screens

The VistA OP Inbound eRx Holding Queue user interface will include new “roll and scroll” screens. The figures below illustrate the following screens:

* eRx List View
  + Search Queue
  + Sort Entries
* Summary Screen
* Patient Validation Screen
* Provider Validation Screen
* Drug/SIG Validation Screen

Figure 38: eRx List View

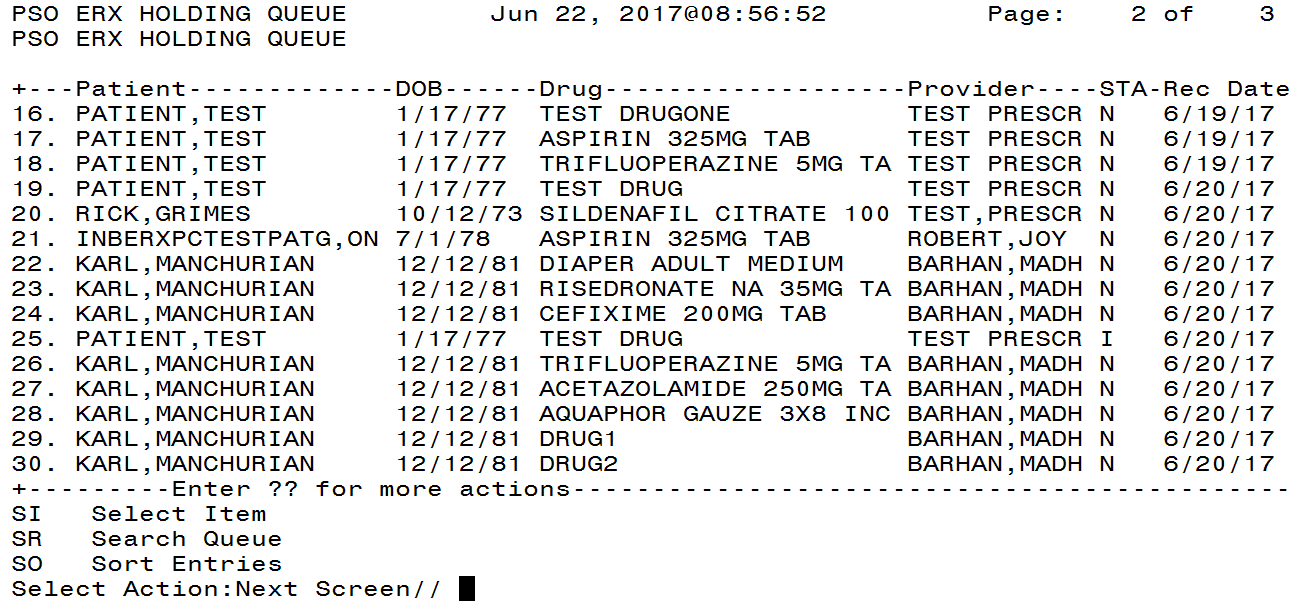


Figure 39: Search Action

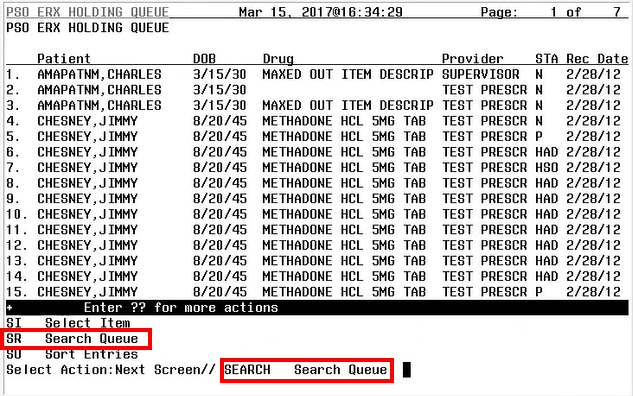


Figure 40: Search Action

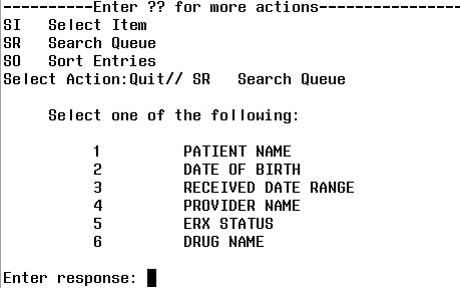


Figure 41: Sort Entries

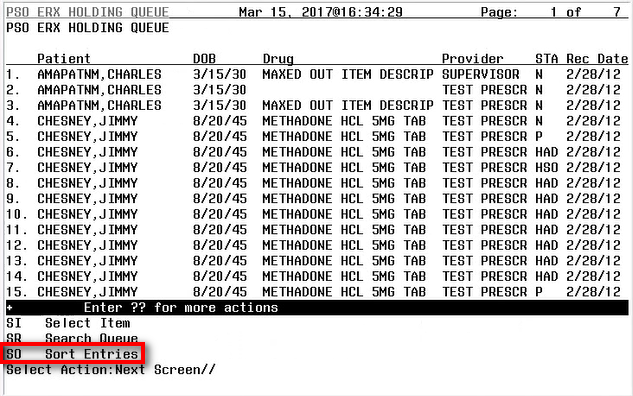


Figure 42: Sort Criteria

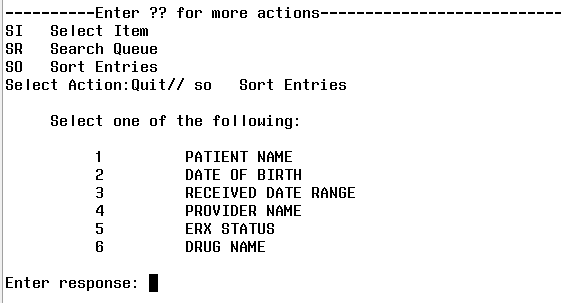


Figure 43: Holding Queue Summary Screen

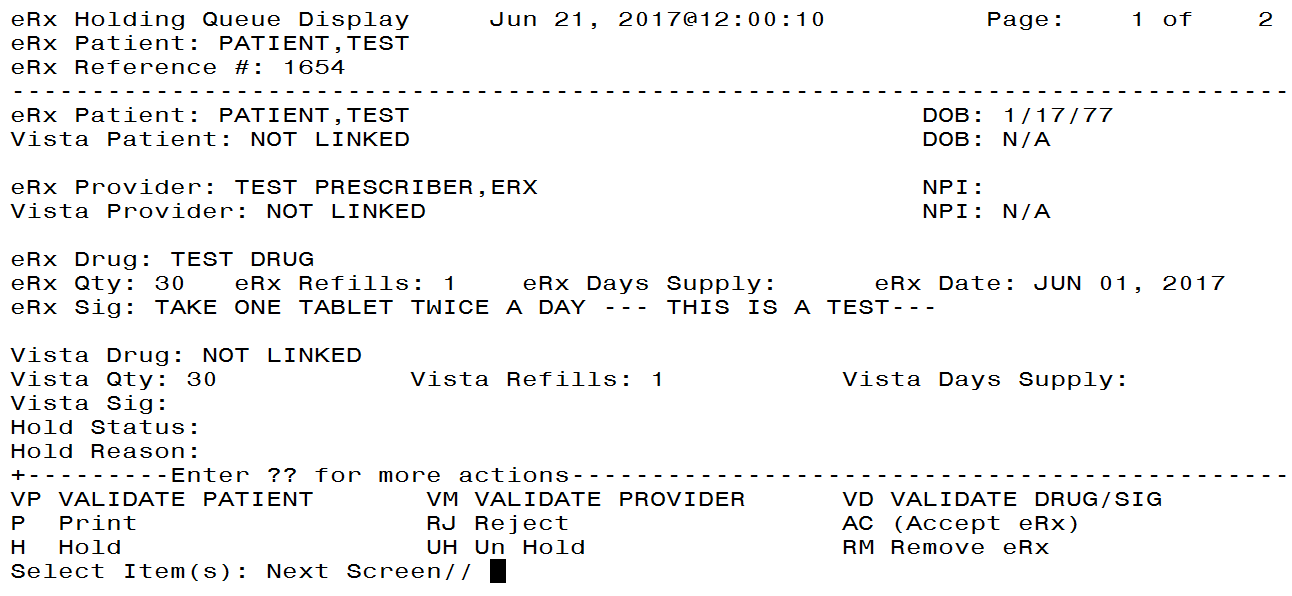


Figure 44: Patient Validation Screen

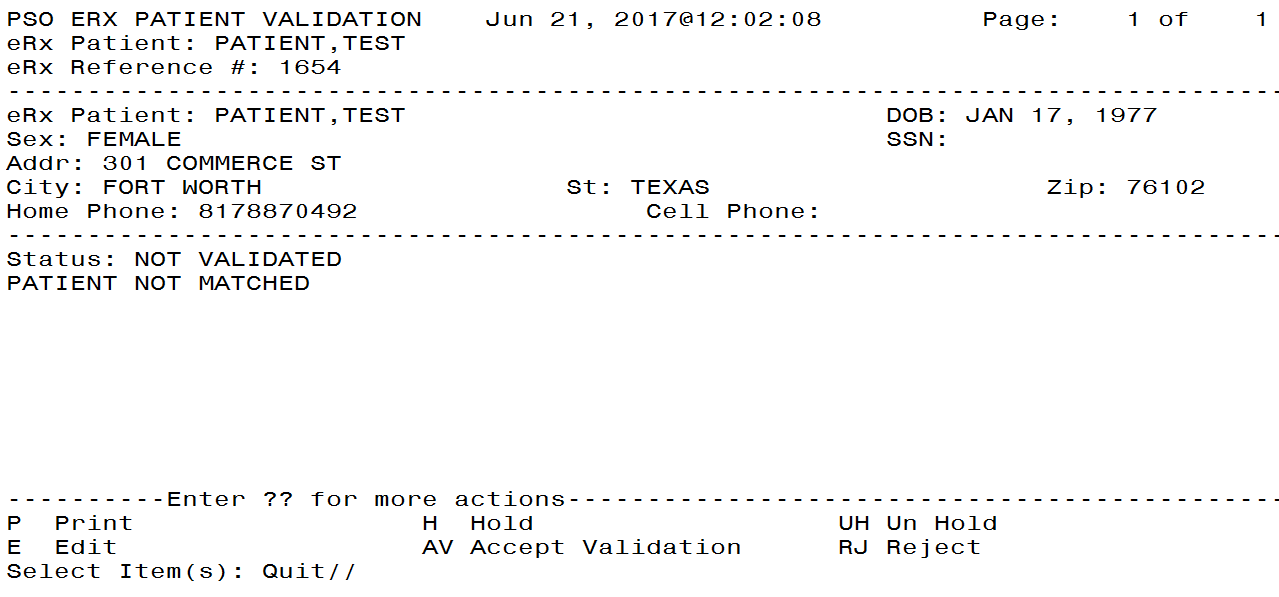


Figure 45: Provider Validation Screen

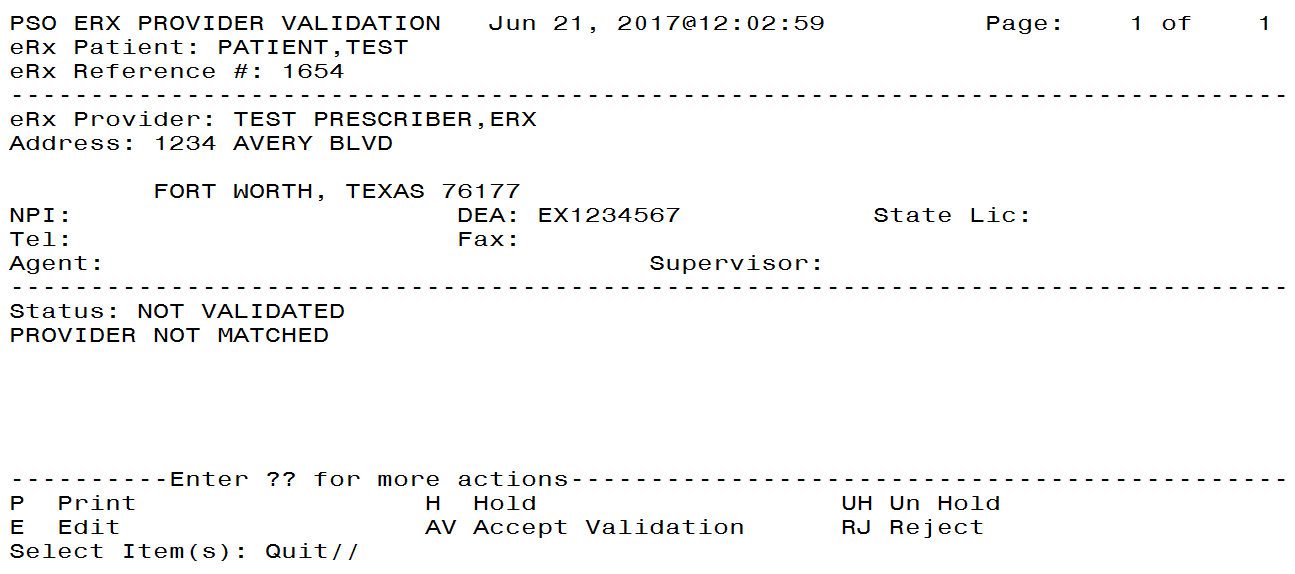
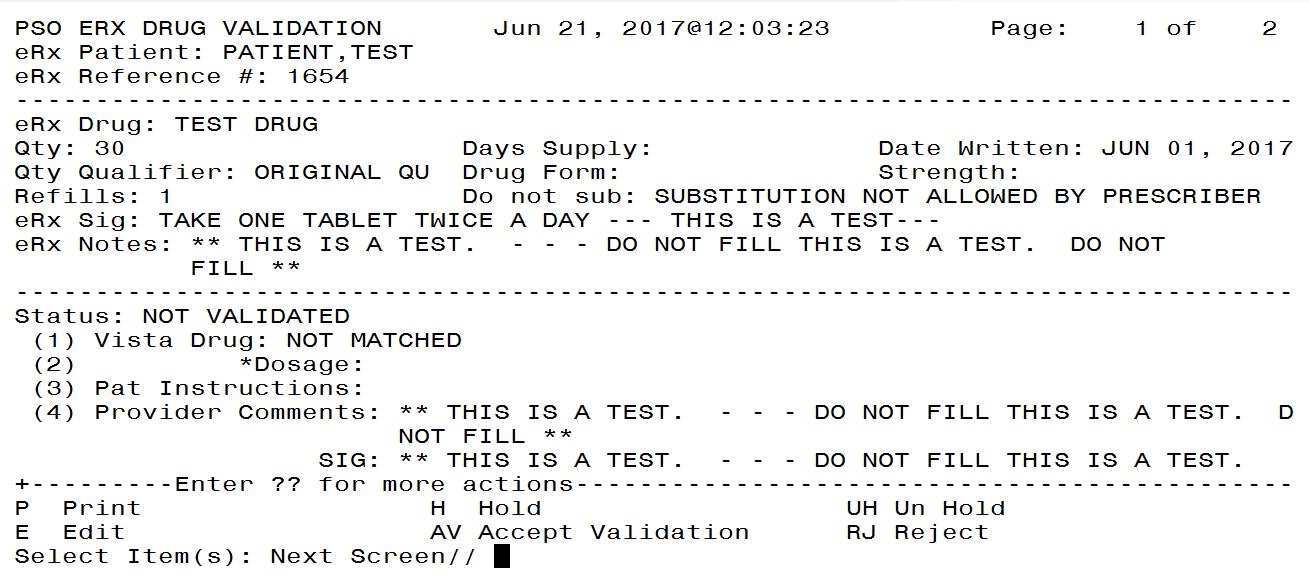


Figure 46: Drug/SIG Validation Screen



1. Additional Information
   1. Relationship to Other Documents and Plans

This design document was developed according to the specification and functionality as found in the following documents:

* *Inbound ePrescribing Requirements Specification Document, Version 2.1*, November 2015
* *Inbound ePrescribing Business Requirements Document, Version 1.0*, October 2014

All documents are available on the VA Inbound eRx SharePoint site and RTC.

* 1. Acronyms and Abbreviations

Relevant definitions, acronyms, and abbreviations for the Inbound eRx application are outlined below.

| **Term** | **Definition** |
| --- | --- |
| ANR | Automated Notification Reporting |
| API | Application Program Interface |
| ARRA | American Recovery and Reinvestment Act |
| BN | Business Need |
| BRD | Business Requirements Document |
| CBO | Central Business Office |
| CCD | Continuity of Care Document |
| CHAMPVA | Civilian Health and Medical Program of the VA |
| COTS | Commercial-Off-the-Shelf |
| CPRS | Computerized Patient Record System |
| CPU | Central Processing Unit |
| DEA | Drug Enforcement Administration |
| DME | Durable Medical Equipment |
| DoD | Department of Defense |
| EDES | Emergency Department Encounter Summary |
| EES | Employee Education System |
| EHR | Electronic Health Record |
| DAS | Data Access Services |
| EMR | Electronic Medical Record |
| ENTR | Enterprise Requirement |
| ePrescribing | Electronic Prescribing |
| ERD | Entity Relationship Diagram |
| eRx | Electronic Prescribing or Electronic Prescription |
| FB | Fee Basis |
| FIPS | Federal Information Processing Standard |
| FTE | Full Time Equivalent |
| FY | Fiscal Year |
| GUI | Graphical User Interface |
| HARB | Health Architecture Review Board |
| HIPAA | Health Insurance Portability and Accountability Act |
| HITSP | Health Information Technology Standards Panel |
| HL7 | Health Level Seven |
| IAM | Identity & Access Management |
| IEC | International Electrochemical Commission |
| IHE | Integrating the Healthcare Enterprise |
| ISO | International Organization for Standardization |
| IT | Information Technology |
| JEE | Java Enterprise Edition |
| JEC | Joint Executive Council |
| JSP | Joint Strategic Plan |
| LOINC | Logical Observation Identifiers, Names, and Codes |
| MbM | Medications by Mail |
| MOCHA | Medication Order Check Healthcare Application |
| MTF | Military Treatment Facility |
| MU | Meaningful Use |
| MVI | Master Veteran Index |
| NCHS | National Center for Health Statistics |
| NCPDP | National Council for the Prescription Drug Programs |
| NDF | National Drug File |
| NIST | National Institute of Standards and Technology |
| NISTIR | National Institute of Standards and Technology Interagency Report |
| nonf | Non-Functional Requirement |
| NSR | New Service Request |
| NTRT | New Term Rapid Turnaround |
| OIA | Office of Informatics and Analytics |
| OI&T | Office of Information and Technology |
| ONCHIT | Office of the National Coordinator for Health Information Technology |
| OP | Outpatient Pharmacy |
| OWNR | Owner Requirement |
| RPC | Remote Procedure Call |
| PBM | Pharmacy Benefits Management |
| PCS | Patient Care Services |
| PD | Product Development |
| PHIS | Pharmacy Hospital Information System |
| PHR | Personal Health Record |
| PRE | Pharmacy Re-engineering |
| RDM | Requirements Development and Management |
| RED | Requirements Elaboration Document |
| RQM | Rational Quality Manager |
| RSD | Requirements Specification Document |
| RTC | Rational Team Concert |
| Rx | Prescription |
| SDS | Standard Data Services |
| SIM | Strategic Investment Management |
| SLA | Service Level Agreement |
| SME | Subject Matter Expert |
| SNOMED CT | Systematized Nomenclature of Medicine Clinical Terms |
| SOA | Service-Oriented Architecture |
| SSOi | Single Sign On Implementation |
| UCD | User Centered Design |
| UI | User Interface |
| VA | Department of Veterans Affairs |
| VAMC | VA Medical Center |
| VE | VistA Evolution |
| VETS | VA Enterprise Terminology Services |
| VHA | Veterans Health Administration |
| VistA | Veterans Health Information Systems and Technology Architecture |
| JEE | Java Enterprise Edition |
| E&E | Eligibility And Enrollment |
| HDR | Health Data Repository |
| EAR | Enterprise Archive, A file format used by Java EE for packaging one or more modules. |
| OP | Outpatient |

* 1. Identification of Technology and Standards
* 2013 DoD/VA Target Health Standards Profile

<http://www.gao.gov/assets/320/315525.html>

* American Recovery and Reinvestment Act of 2009  
  <http://www.gpo.gov/fdsys/pkg/BILLS-111hr1enr/pdf/BILLS-111hr1enr.pdf>
* [Centers for Medicare and Medicaid Services  
  http://www.cms.gov/Medicare/E-Health/Eprescribing/index.html](http://www.cms.gov/Medicare/E-Health/Eprescribing/index.html)
* Department of Veterans Affairs FY2014-2020 Strategic Plan
* National Center for Health Statistics (NCHS) Data Brief No. 143, January 2014  
  <http://www.cdc.gov/nchs/data/databriefs/db143.htm>
* NCPDP Prescription Transfer Standard 3.2  
  <http://www.ncpdp.org/Standards/Standards-Info>
* NCPDP SCRIPT Standard 10.6
* NSR #20130905 VistA Evolution
* One-VA Enterprise Architecture Enterprise Technical Architecture
* One-VA Technical Reference Model
* VA Handbook 6500 – Information Security Program
* VHA’s Defining Excellence in the 21st Century
* VHA Handbook 1108.05 Outpatient Pharmacy Services

The tools to be utilized for the implementation of the Inbound eRx solution are all verified as being on the current TRM approved list.

| **Technology/Software/Tool** | **TRM** |
| --- | --- |
| Apache CXF (version 3.1.7) |  |
| Apache HTTP Server (version 2.3) |  |
| Apache log4j (version 2.3) |  |
| Apache Maven (version 3.3.9) |  |
| Apache Tiles (version 3.0.5) |  |
| Dojo Toolkit (version 1.10) |  |
| Eclipse Mars (version 4.5.1) |  |
| Hibernate ORM (version 5.2.10) |  |
| Hibernate Validator (version 5.4.0) |  |
| HP Fortify (version 17.10) |  |
| Java 8u131 |  |
| Oracle Database 11.2.0.4.0 |  |
| Oracle SQL Developer (version 4.1.5) |  |
| Oracle WebLogic 12c (version 12.1.3) |  |
| Pentaho Data Integration (version 6.1.0.1) |  |
| Slf4j (version 1.7) |  |
| SoapUI (version 5.2.1) |  |
| Spring Framework version (version 4.3.9) |  |
| Spring Security (version 4.2.0) |  |

* 1. Constraining Policies, Directives and Procedures

This design document is based on the system design document template standardized by VA.

From the corresponding BRD for this effort, the development of the Inbound ePrescribing system has the following constraints:

* Joint health standards profile defined by the DoD/VA Health Architecture Review Board (HARB) in support of the DoD/VA Joint Executive Council (JEC) Joint Strategic Plan (JSP)
* NCPDP SCRIPT Standard 10.6 or subsequent current version
* NCPDP Prescription Transfer Standard 3.2 or subsequent current version
* Utilizing DAS as a gateway for incoming prescription data and outgoing statuses to Change Healthcare
  1. Requirements Traceability Matrix

Please refer to the PHARM (QM) Inbound ePrescribing in Rational Quality Manager (RQM) for the RTM. For access to the RTM, please contact the Inbound eRx Project Manager.

* 1. Packaging and Installation

Application code targeted for the WebLogic Server on the Inbound eRx Processing Hub will conform to the standard JEE specification for resources run inside a packaged EAR file. The EAR file will package one or more JEE modules into a single module to align class loading and deployment into a single server. VistA OP packaging and installation will conform to standardized VA practices details of which will be provided in a future increment.

* 1. Design Metrics

All software development by Inbound eRx developers shall conform to technology standards as defined in the VA TRM.

Inbound ePrescribing will be developed and maintained using industry standard technologies. This project will add functionality that complements the existing VistA OP package. All necessary security protocols and requirements necessary for interacting with this package as well as others are outlined in this SDD and RSD.

1. Meds by Mail Brochure [↑](#footnote-ref-2)
2. A thorough history of all regular medication use (prescribed and non-prescribed) by an individual [↑](#footnote-ref-3)