Department of Veterans Affairs (VA)

Medical Appointment Scheduling System (MASS)

Contract #VA118-15-D-0129 (IDIQ)  
Task Order # TO 0002

MASS System Design Document (SDD)

Version 2.0



Prepared by:



8521 Leesburg Pike, Suite 600

Vienna, VA 22182

December 20, 2017

Revision History

| Date | Version | Description | Author |
| --- | --- | --- | --- |
| 12/20/2017 | 2.0 | Final | Robert (Reilly) Maginn |
| 8/15/2017 | 1.0 | Final (Initial) | Robert (Reilly) Maginn |

Table of Contents

[1.0 Introduction 1](#_Toc501524290)

[1.1 Scope 1](#_Toc501524291)

[2.0 Overview 2](#_Toc501524292)

[2.1 Overview of the System 2](#_Toc501524293)

[2.1.1 Overview of the Business Process 3](#_Toc501524294)

[2.1.2 Overview of Technical, Security, and Single Sign On (SSO) 5](#_Toc501524295)

[2.1.3 Overview of Pivotal Technical Requirements 5](#_Toc501524296)

[2.1.4 Overview of Security and Privacy Requirements 6](#_Toc501524297)

[2.1.5 Overview of Single Sign On Requirements 6](#_Toc501524298)

[2.1.6 Overview of Use of Enterprise Portals 6](#_Toc501524299)

[3.0 Conceptual Design 7](#_Toc501524300)

[3.1 Conceptual Application Design 7](#_Toc501524301)

[3.1.1 Application Context Overview 7](#_Toc501524302)

[3.1.2 Application Context Description (Configurations) 8](#_Toc501524303)

[3.1.3 Application Context Description (Interfaces) 11](#_Toc501524304)

[3.2 High-Level Application Design 16](#_Toc501524305)

[3.3 Application Locations 19](#_Toc501524306)

[3.4 Conceptual Data Design 20](#_Toc501524307)

[3.4.1 Project Conceptual Data Model 20](#_Toc501524308)

[3.4.2 Database Information 20](#_Toc501524309)

[3.5 User Interface Data Mapping 21](#_Toc501524310)

[3.6 Conceptual Infrastructure Design 21](#_Toc501524311)

[3.7 System Criticality and High Availability 21](#_Toc501524312)

[3.8 Special Technology 21](#_Toc501524313)

[3.9 Technology Locations 21](#_Toc501524314)

[3.10 Conceptual Infrastructure Diagram 21](#_Toc501524315)

[4.0 System Architecture 22](#_Toc501524316)

[4.1 Hardware Architecture 22](#_Toc501524317)

[4.1.1 Compute 23](#_Toc501524318)

[4.1.2 Citrix XenApp Environment 26](#_Toc501524319)

[4.1.3 Management Systems 29](#_Toc501524320)

[4.1.4 MyChart Environment 30](#_Toc501524321)

[4.1.5 Storage 30](#_Toc501524322)

[4.1.6 Cloud Management 34](#_Toc501524323)

[4.1.7 Backup 34](#_Toc501524324)

[4.2 Software Architecture 37](#_Toc501524325)

[4.2.1 MASS Adapter 38](#_Toc501524326)

[4.3 Network and Wide Area Network (WAN) Architecture and Connectivity 46](#_Toc501524327)

[4.3.1 Storage Connectivity 49](#_Toc501524328)

[4.3.2 Load Balancing 49](#_Toc501524329)

[4.3.3 Management 51](#_Toc501524330)

[5.0 Data Design 51](#_Toc501524331)

[5.1 Database Management System Files 51](#_Toc501524332)

[5.2 Data View 51](#_Toc501524333)

[6.0 Detail Design 52](#_Toc501524334)

[6.1 Hardware Detailed Design 52](#_Toc501524335)

[6.2 Software Detailed Design 53](#_Toc501524336)

[6.3 Conceptual Design 53](#_Toc501524337)

[6.3.1 User Interfaces 54](#_Toc501524338)

[6.3.2 User Characteristics 54](#_Toc501524339)

[6.3.3 Software Interfaces 54](#_Toc501524340)

[6.3.4 Memory Constraints 59](#_Toc501524341)

[6.3.5 Special Operations 59](#_Toc501524342)

[6.3.6 Product Features 59](#_Toc501524343)

[6.3.7 Dependencies and Constraints 60](#_Toc501524344)

[6.4 Specific Requirements 60](#_Toc501524345)

[6.4.1 Database Repository 60](#_Toc501524346)

[6.4.2 System Features 60](#_Toc501524347)

[6.4.3 Routines (Entry Points) 60](#_Toc501524348)

[6.5 Network Detailed Design 60](#_Toc501524349)

[6.6 Security and Privacy 60](#_Toc501524350)

[6.6.1 Security 60](#_Toc501524351)

[6.6.2 Privacy 62](#_Toc501524352)

[6.7 Service Oriented Detailed Design 62](#_Toc501524353)

[6.7.1 Patient Sequence 63](#_Toc501524354)

[6.7.2 Orders Sequence 63](#_Toc501524355)

[6.7.3 Scheduling Sequence 64](#_Toc501524356)

[6.7.4 Scheduling to VistA Sequence 64](#_Toc501524357)

[6.7.5 Application Error to MASS Sequence 65](#_Toc501524358)

[6.7.6 PCMM Sequence 65](#_Toc501524359)

[6.7.7 AudioCare Sequence 66](#_Toc501524360)

[7.0 External System Interface Design 67](#_Toc501524361)

[7.1 MyChart 67](#_Toc501524362)

[7.2 Ensemble 67](#_Toc501524363)

[7.3 Interface Architecture and Detailed Design 67](#_Toc501524364)

[8.0 Human-Machine Interface 67](#_Toc501524365)

[8.1 Inputs 67](#_Toc501524366)

[8.2 Outputs 67](#_Toc501524367)

[Appendix A - Signature Page 68](#_Toc501524368)

[Appendix B – Acronyms 69](#_Toc501524369)

[Appendix C – Reference Materials 71](#_Toc501524370)

[Appendix D – Business Needs Matrix 73](#_Toc501524371)

[Appendix E – Business Success Criteria 118](#_Toc501524372)

[Appendix F – Routines (Entry Points) 120](#_Toc501524373)

# Introduction

The Medical Appointment Scheduling Systems (MASS) program will deploy a commercial scheduling application to support schedulers, and ultimately Veterans, at the pilot site, Chalmers P. Wylie Department of Veterans Affairs (VA) Ambulatory Care Center (hereafter referred to as the Columbus ACC) and its associated Community Based Outpatient Clinics in Columbus, OH. This deployment and integration will allow VA to move from a clinic-based scheduling system to a resource-based scheduling system, thereby standardizing and improving scheduling processes, providing resource supply management, and delivering improved access to care for Veterans.

TeamSMS will incrementally capture requirements, design, develop, deploy, implement, and train users across the VA health care enterprise. The MASS program involves the initiation, design, and execution of the Columbus ACC pilot site to demonstrate the business value, the integration of MASS within the VA enterprise, the impact MASS will have on various clinical and administrative processes and workflows, and the effort necessary for affected stakeholders to adopt these changes.

## Scope

This System Design Document (SDD) provides the technical details related to infrastructure, integration, cloud-hosting, development, and deployment of the MASS solution at the Columbus ACC pilot site.

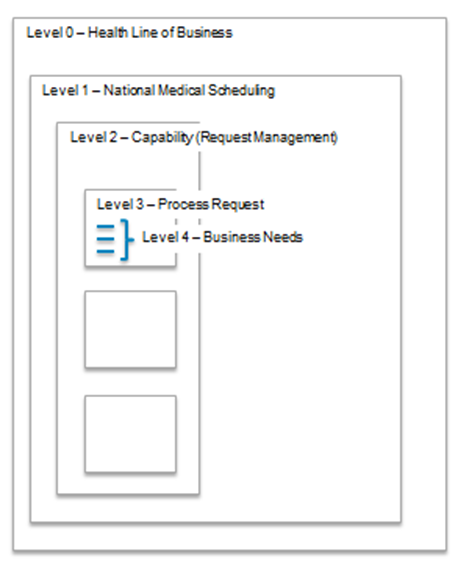
# Overview

## Overview of the System

Veterans Health Administration (VHA) outpatient scheduling functions are defined within a framework which draws a boundary around scheduling and clusters similar capabilities and their sub-capabilities.

*Figure 1: Scheduling Operation, Capabilities, and Business Functions Framework* shows the framework that represents the scheduling operation, capabilities, and business functions at four levels.

Figure : Scheduling Operation, Capabilities, and Business Functions Framework



Each level in the framework is described in detail below:

* **Level 0:** Represents the Health Care Line of Business (LOB). It identifies the operational model within VA related specifically to providing health care. Other LOBs might be the Financial or Burial and Cemetery
* **Level 1:** Represents VHA’s vision of the scheduling operation that encompasses outpatient scheduling
* **Level 2:** Represents business capabilities for specific grouping of activities and functions (i.e., Manage Appointment). The MASS team identified the following capabilities within the framework:
  + MASS Setup (i.e., Configuration)
  + Veteran Information Management
  + Request Management (i.e., Demand)
  + Appointment Management (i.e., Services and Delivery)
  + Coordinate Associated and Occasion of Services (i.e., Delivery and Services)
  + Encounter of Care Management (i.e., Episode of Care)
  + Report Management
* **Level 3:** Represents a group of functions for each capability. For example, the Manage Appointment capability has four functions; each function has a set of business needs
* **Level 4:** Represents the business needs, as described in the end-to-end scheduling process activities, integration between the capabilities, and flow of information

The framework serves as VHA’s highest level artifact and is used as a guide to identify high-priority and VHA-unique scheduling characteristics. The business needs matrix, located in *Appendix D – Business Needs Matrix*, defined by Access and Clinic Administrative Program and VA Office of Information and Technology (OI&T), shall take precedence for all MASS business requirements from Level 4 down.

### Overview of the Business Process

MASS supports the implementation of business needs detailed below in *Table 1: MASS Business Needs*. Supporting documents for each business need are stored in the Business Process folder on SharePoint.

Table : MASS Business Needs

| Capability | Business Need(s) |
| --- | --- |
| Medical Scheduling Setup | * Provide VHA with ability to manage resources throughout the scheduling process * Provide VHA with ability to manage providers as resources * Provide VHA with ability to manage facilities within a specific VA health care system as resources * Provide VHA with ability to manage equipment as resources * Provide VHA with ability to manage components of scheduling, such as service lines and modes of delivery * Provide data synchronization rules/triggers for external systems * Provide ability to create local (i.e., Facility, Service Line, Clinic) care coordination agreements * Provide VHA with ability to manage, store, and access standard notification templates (e.g., forms and letters) used for various patient communications (e.g., phone, mass mailing, and secure messaging) * Allow users with appropriate security privileges to establish standard parameters, business rules, alerts, workflows, etc. to ensure compliance with VHA scheduling practices * Provide VHA with ability to establish user access roles and rules at multiple organizational levels |
| Veteran Information Management | * Provide VHA with ability to view near real-time eligibility information from Veterans Integrated System Technology Architecture (VistA) * Provide VHA with ability to view and use Veteran information * Provide VHA with ability to manage patient healthcare preferences (e.g., appointment date and time, preferred provider, and language assistance) * Provide VHA with ability to manage patient healthcare special needs (e.g., transportation, escort services, and handicap assistance) when scheduling an appointment |
| Request Management | * Provide VHA with ability to enter requests for care from patients, providers, individuals authorized to act on behalf of the patient * Provide VHA with an all-inclusive master list designated for requests (including waiting for available appointments, recalls, etc.) in a consolidated standard format used for request processing and appointment management * Provide VHA with ability to perform request oversight activities by providing data for analysis regarding request activities |
| Appointment Management | * Provide VHA with ability to search available healthcare resources and reserve resources to meet the Veteran's needs in accordance with preferred date for care from a single integrated view * Provide VHA with ability to assess the impact of care due to a cancellation by tracking the cancellation request to the resolution (e.g., rescheduling, backfilling resources, providing alternate modes of healthcare delivery options, or cancelling the appointment) * Provide VHA with ability to streamline the notification process by providing an automatically populated, standard set of templates (e.g., letters, post cards, emails, or one-on-one phone calls) for all types of communication * Provide VHA with ability to monitor appointment, cancellation, reschedule and notification processes by reporting (i.e., structured, user defined, and ad-hoc) operational efficiencies or deficiencies in near-real time |
| Coordinated Occasions of Care | * Provide VHA with ability to coordinate services and access to care at facilities under the purview of the Columbus ACC |
| Encounter Management | * Provide VHA with ability to perform check-in activities for appointments * Provide VHA with ability to capture information about encounter events to include metrics for wait-time studies * Provide VHA with ability to perform check-out activities for appointments * Provide VHA with ability to monitor the check-in process and check-out activities for rapid decision making and identify operational efficiencies or deficiencies in near-real time |
| Reporting | * Provide VHA with ability to generate capacity management reports to illustrate capacity planning for resources and work force utilization throughout the scheduling operation * Provide VHA with ability to generate operational reports which will provide a variety of operational performance and audit reports of daily healthcare scheduling activities |
| User Interfaces | * Provide an intuitive, seamless user experience. The solution shall optimize qualities such as accessibility, responsiveness, consistency, aesthetics, and ease-of-use |
| Business Rules Management | * Allow VHA to create and maintain fluid business rules so that IT involvement is minimized in modifying these rules |
| Workflow Management | * Allow the user to define, create, and move workflows for different types of jobs or processes, in order to manage the end-to-end work items or tasks that need to be performed as part of the scheduling process |

### Overview of Technical, Security, and Single Sign On (SSO)

The following sub-sections provide an overview of significant technical, security, and single sign on-related requirements.

### Overview of Pivotal Technical Requirements

Pivotal technical requirements are items that satisfy business needs while providing thresholds to measure the success of the MASS program.

Specific pivotal technical requirements for the infrastructure component of MASS are defined using Epic and Microsoft-specified best practices, specifically in relation to Microsoft Azure for Government (MAG), based on their commercial customer experiences.

Specific pivotal technical requirements, as related to business needs, are defined initially for the MASS pilot site, Columbus ACC, and will be used as the basis for success criteria through the rollout targets. These requirements are specified in *Appendix D – Business Needs Matrix* and *Appendix E – Business Success Criteria*.

*Table 2: MASS Tool Summary* describes the tools used to build MASS.

Table : MASS Tool Summary

| Tier | Platform | Component | Version | Description |
| --- | --- | --- | --- | --- |
| Adapter | Linux | Java JDK | 1.8 | Programming tool |
| Tomcat | 8.0.23 | Embedded server |
| Spring Boot | 1.5 | Microservices management and configuration management |
| MS SQL | 2016 | Database |
| Docker | 17.10.0 | Docker Container |
| Jenkins |  |  |
| Java Runtime |  |  |
| ActiveMO |  |  |
| Maven | 3.3 | Build and Version Management |
| Epic | Linux | Cadence |  |  |
| Cogito |  |  |
| Chronicles |  |  |

### Overview of Security and Privacy Requirements

MASS inherits the following privacy requirements:

* MAG FedRAMP high accreditation package
* Federal Information Security Management Act (FISMA) moderate system, as detailed in the Privacy Act of 1974, 5 U.S.C. § 552a
* Health Insurance Portability and Accountability Act (HIPAA) of 1996
* National Institute of Standards and Technology (NIST) Special Publication 800-66 Revision 1
* Federal Register 45 CFR Parts 160, 162, and 164 Health Insurance Reform: Security Standards; Final Rule

### Overview of Single Sign On Requirements

Cadence will use pass-through authentication to log in end users based on information from Windows. This option is a light-weight alternative to Clinical Context Object Workgroup or third-party SSO software to enable access to Cadence without a password or other authentication beyond what Windows requires.

Pass-through authentication does not apply to revalidation; users must still enter their passwords or use another authentication method if a workflow in Cadence requires additional validation.

Integration with card-based authentication solutions, like the VA Personal Identity Verification (PIV) cards. In the case of the pilot site, PIV card integration will be done at the Windows OS level. MASS will assume those credentials at login and no PIV integration will be expected for the pilot.

Patients also will have logins into the system for MyChart and MyChart mobile access. The patient database in Cadence is separate from the user database, and the previous section does not apply to patients. Patient authentication will occur via Security Assertion Markup Language (SAML)-based (i.e., token-based authentication) integration with the single sign on external (SSOe) Identity and Access Management (IAM) solution to conform to the standard recommended patient authentication approach for VA.

### Overview of Use of Enterprise Portals

This section is not applicable at this time.

# 

# Conceptual Design

The following sub-sections describe the MASS conceptual design.

## Conceptual Application Design

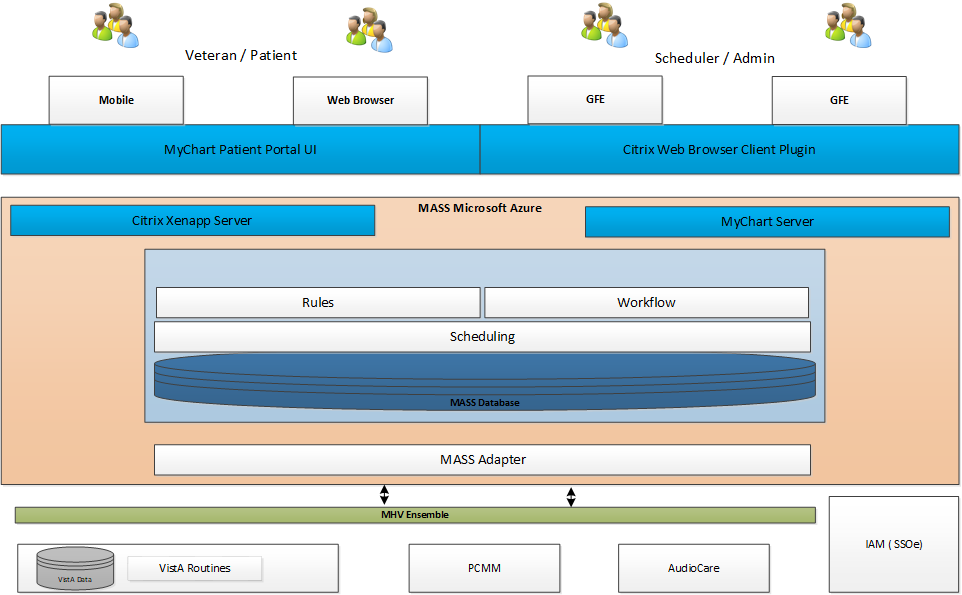
The following sub-sections provide information regarding the MASS conceptual application design.

### Application Context Overview

*Figure 2: MASS Conceptual View* provides the conceptual context for MASS with respect to other VA systems. The MASS application contains three main components:

* **Epic**
* **MASS Adapter** (to communicate with VA systems such as MyHealtheVet (MHV) Ensemble which subsequently communicates data to AudioCare, VistA, and Primary Care Management Module (PCMM))
* **VistA routines** (at the local VA Medical Centers (VAMC) VistA instance)

Figure : MASS Conceptual View

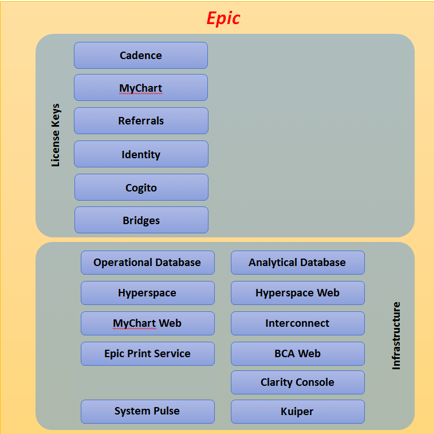


The application context focuses on the commercial application purchased from Epic. MASS Adapter services and VistA routines are considered helper services. These helper services will facilitate transforming VA scheduling, patient, and order/consult information into industry standard messaging and vice versa. This methodology will provide VA with a commercial application that has not been refactored at the software development level for specific VA use. It will also provide a scheduling interface to VistA that will not break existing downstream scheduling integrations with VistA.

### Application Context Description (Configurations)

The Epic cluster, as shown in *Figure 3: Epic Functionality Purchased by VA*, contains many configurations purchased by VA in support of scheduling. These configurations enable various components within Epic. Enabled components are limited by VA to only include licensing for scheduling-related features. Any features not specifically related to scheduling, such as display of clinical information in the MyChart portal, fall outside the scope of MASS and therefore, outside the scope of VA’s Epic license. Although this functionality can be easily turned on in Epic, it requires VA to purchase additional licenses from Epic and to re-evaluate the data center requirements and integration and conversion requirements of the system.

Figure : Epic Functionality Purchased by VA

**

The VA’s purchase of Epic is divided into the infrastructure components that come with every instance of Epic and the specific license keys that enable the MASS-specific functionality.

Epic’s infrastructure components, shown in *Figure 3: Epic Functionality Purchased by VA*, are described further detail below:

* **Operational Database:** Core to Epic’s single integrated record for each patient. The open Chronicles Extended Relational Database Management System (Chronicles), a high-performance data engine at the heart of the integrated product suite, is a foundational component of Epic. It has no built-in capacity limits, thereby supporting thousands of concurrent users and millions of records while still providing excellent response times
* **Hyperspace:** Epic’s integrated product suite user interface. The user interface is service-driven, separating the presentation layer from business logic, thereby enabling the presentation to be configured consistently across all of Epic’s products
* **Response Time Tracking:** Embedded in Hyperspace and integrated with System Pulse, the utility records response time data for over 300 pre-defined workflow steps
* **MyChart Web:** Epic’s browser-based, iOS-based, or Android-based, patient access portal. Its self-serve online functions can activate patients to improve their own health, reduce the cost of customer service, and provide a vital communication link to support accountable care
* **Epic Print Service:** A server solution for transporting rich text documents from Epic to a printer
* **Web BLOB Service:** A web application that Hyperspace connects to that is used to save and load media files
* **System Pulse:** Epic’s monitoring package, which consolidates system messages from various sources (e.g., database servers, Epic environments, Windows servers, XenApp servers) and presents them in a single web-based dashboard. It provides operations and maintenance services the ability to use a dashboard to monitor the overall health and performance of the scheduling system and execute any necessary corrective actions
* **Analytical Database:** Extracts data from Chronicles and stores it on a dedicated analytical reporting server in a relational format, organized as tables and columns
* **Hyperspace Web:** The next generation of Hyperspace, which will be web delivered
* **Interconnect:** Epic’s web service layer used to connect to the VA’s Enterprise Services
* **BCA:** A combination of Interconnect and Web BLOB Service. It stores reports coming from the database before they are distributed to the client for printing
* **BCA Web:** An application that enables users to view and print downtime reports through any web browser on the network
* **Clarity Console:** Serves as a communication box between the Cache environment and reporting server
* **Kuiper:** A web-based tool that system administrators can use to manage and deploy Epic products remotely to multiple servers and environments

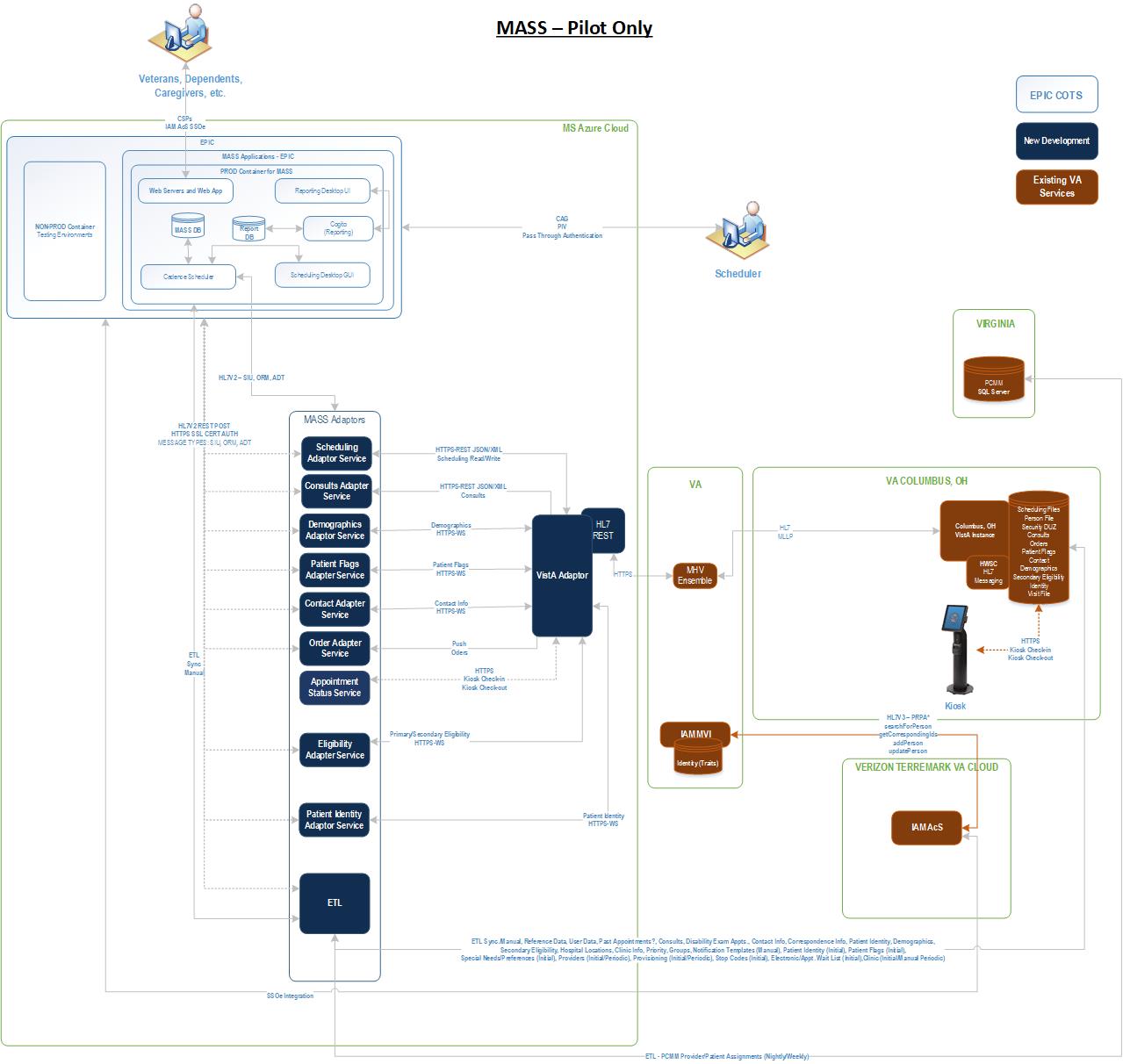
Epic’s License Key components depicted in *Figure 3: Epic Functionality Purchased by VA*, are described below:

* **Cadence:** Epic’s enterprise, resource-based, advanced rules-based scheduling and referral tracking system. It will enable VA to track patient throughput at every stage of the appointment and care process, and ensure that patients are scheduled in a timely manner with the appropriate resources. Cadence makes it easy for staff to schedule visits while providing advanced reporting tools, and includes a patient portal that help improves patient services, keeps appropriate slots open, and makes the most efficient use of staff and other resources
* **MyChart:** Epic’s browser-based, iOS-based, or Android-based patient access portal. Its self-serve online functions enable patients to improve their own health, reduce the cost of customer service and provide a vital communication links to support accountable care
* **Identity:** Epic's ID and duplicate record management application. It can be used to prevent issues, such as multiple records for the same patient or single records for multiple patients with identical names and similar data
* **Cogito:** Epic's integrated analytics and reporting system that delivers current clinical intelligence and business intelligence based on role and workflow. Intelligence can mean something different to each user, so Epic provides a combination of flexible tools, content, data sources, distribution, training, and process to support decisions throughout the health system with the best information available. The following are examples of how Cogito supports different users:
  + **For the Clinician/User:** Cogito presents clinicians with information that can improve the care they give – and lets them see how well they are doing. The dashboard informs users with daily trends, tasks, and messages pertinent to the user’s role and configurable to the user’s preferences. More than 500 roles are currently in the Cogito library, including more than 100 specialized physician roles alone. Physicians and other members of the care team, empowered by analytics, hold the potential to help create a learning healthcare organization
  + **For the Manager:** Cogito tools provide insight into departmental performance. Dashboards and benchmark metrics help identify areas for improvement, whether the opportunity lies in personnel training, process redesign, or a change to the care model
  + **For the Clinical Leader:** Embedded clinical intelligence can guide quality improvement with dashboards that aggregate and compare clinical performance across departments and facilities. Dashboards in development will provide visual comparisons to anonymized benchmarks of organizational key performance indicators, both clinical and financial
* **Bridges Interface Toolkit:** Facilitates interface maintenance with a real-time interface monitor, a usage statistics utility, an error logging utility, and an interactive error-analysis report. It enables customers to manage the interfaces purchased from Epic’s standard library of Health Level 7 (HL7), American National Standards Institute X12, and XML interfaces. It also enables Epic to maintain flexibility in data mapping and formatting. Bridges Interface Toolkit can be used to support interface transactions in a fixed format or other non-standard data formats

### Application Context Description (Interfaces)

*Figure 4: MASS System Interfaces* depicts conceptual design of the Adapter layer of MASS at the pilot site.

Figure : MASS System Interfaces



*Figure 5: MASS System Interfaces* depicts the conceptual MASS system interfaces and the conceptual path to those systems; although routing and transformation analysis is still ongoing.

Figure : MASS System Interfaces

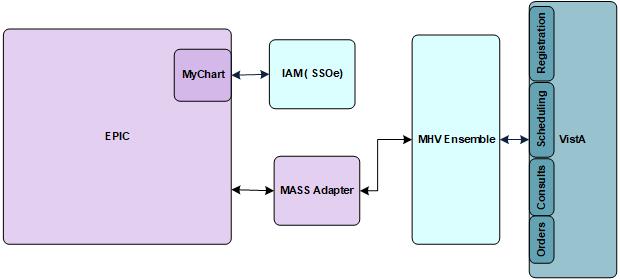


Table 3: System Interfaces lists the primary system interfaces that have been identified and specified as integration requirements.

Table : System Interfaces

| ID | Description | Interface Name | Interface System |
| --- | --- | --- | --- |
| 1 | Epic pulls user information from IAM (Veteran) | Epic-IAM | IAM |
| 2 | Epic push scheduling data to VistA | Epic-VistA | VistA |
| 3 | Kiosk pushes scheduling data to Epic | VistA-Epic | VistA |
| 4 | Eligibility and Enrollment, demographic, contact information pushed to Epic | VistA-Epic | VistA |
| 5 | VistA pushes appointment requests to Epic | VistA-Epic | VistA |
| 6 | Patient creation, Patient identity pushed to Epic | VistA-Epic | VistA |
| 7 | Patient care team information pushed to Epic | PCMM-Epic | PCMM |
| 8 | AudioCare information updated to and from Epic | AudioCare-Epic | AudioCare |

There are multiple system interfaces identified. With each system interface, there are multiple domains of information that have been identified along that interface. A specific domain, communicating along an interface, may use a different communication method than other domains communicating along the same system interface. Analysis to determine the specific communication method per domain is ongoing.

The primary method of interfacing to the Epic system is to re-use the current interface mechanisms (i.e., HL7v2). *Table 4: Scheduling Events* depicts the HL7v2 events that are planned to be used to communicate to the Epic system.

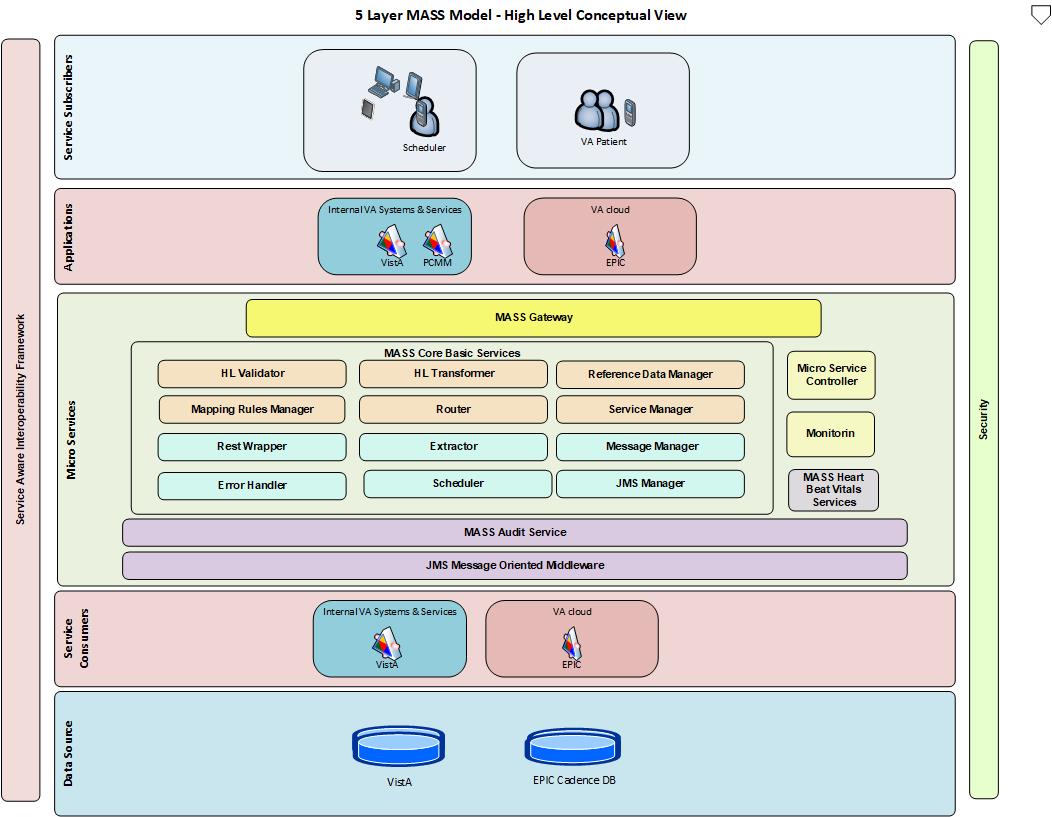
Table : Scheduling Events

|  |  |  |
| --- | --- | --- |
| Scheduling Events | HL7v2.x Event from VistA | HL7v2.x Event to Epic |
| Schedule | SIU S12 | SIU S12 |
| Update | SIU S14 | SIU S14 |
| Cancel | SIU S15 | SIU S15 |
| Check In | SIU S14 | SIU S14 |
| Sign In | SIU S14 | SIU S14 |
| Check Out | SIU S14 | SIU S14 |
| Cancel Check In | SIU S14 | SIU S14 |
| Cancel Check Out | SIU S14 | SIU S14 |
| Cancel Sign In | SIU S14 | SIU S14 |
| Notes Update | SIU S14 | SIU S14 |
| Reassign | SIU S14 or S15/S12 | SIU S14 or S15/S12 |
| Reschedule | SIU S15 followed by an SIU S12 | SIU S15 followed by an SIU S12 |
| No Show | SIU S26 | SIU S26 |
| Change Appointment | SIU S14 or S15/S12 | SIU S14 or S15/S12 |
| Add Provider Resource | SIU S14 or S15/S12 | SIU S14 or S15/S12 |
| Remove Provider Resource | SIU S14 or S15/S12 | SIU S14 or S15/S12 |
| Add Provider Resource Request | SIU S14 or S15/S12 | SIU S14 or S15/S12 |
| Remove Provider Resource Request | SIU S14 or S15/S12 | SIU S14 or S15/S12 |
| Appointment Edit Statistics Update x2 | SIU S14 | SIU S14 |
| Appointment Edit Statistics Cancel | SIU S15 | SIU S15 |
| Appointment Edit Statistics No Show | SIU S26 | SIU S26 |
| Appointment Edit Statistics Complete | SIU S14 | SIU S14 |
| End of Day No Show | SIU S26 | SIU S26 |
| End of Day Cancel | SIU S15 | SIU S15 |
| End of Day Complete | SIU S14 | SIU S14 |
| End of Day Update | SIU S14 | SIU S14 |
| Link Order | SIU S14 or S15/S12 | SIU S14 or S15/S12 |
| Consult Events | HL7v2.x Event from VistA | HL7v2.x Event to Epic |
| Create Consult in VistA | ORM^O01 | ORM^O01 |
| Update Consult in VistA | ORM^O01 | ORM^O01 |
| Cancel Consult in VistA | ORM^O01 | ORM^O01 |
| Link Consult to Appointment in Cadence | SIU S14 or S15/S12 | SIU S14 or S15/S12 |
| Order Events | HL7v2.x Event from VistA | HL7v2.x Event to Epic |
| Create Order in VistA | ORM^O01 | ORM^O01 |
| Update Order in VistA | ORM^O01 | ORM^O01 |
| Cancel order in VistA | ORM^O01 | ORM^O01 |
| Link Order to Appointment in Cadence (Not a current need) | SIU S14 or S15/S12 | SIU S14 or S15/S12 |
| Patient ID Events | HL7v2.x Event from VistA | HL7v2.x Event to Epic |
| Identifier Add | ADT^A28 or ADT^A31 | ADT^A28 or ADT^A31 |
| Identifier Update | ADT^A08 | ADT^A31 |
| Merge Patients | ADT^A40 | ADT^A40 |
| Demographic Events | HL7v2.x Event from VistA | HL7v2.x Event to Epic |
| Demographics Add | ADT^A08 | ADT^A31 |
| Demographics Update | ADT^A08 | ADT^A31 |
| Contact Information Events | HL7v2.x Event from VistA | HL7v2.x Event to Epic |
| Contact Add | ADT^A08 | ADT^A31 |
| Contact Update | ADT^A08 | ADT^A31 |
| Patient Flag Events | HL7v2.x Event from VistA | HL7v2.x Event to Epic |
| Patient Flag Add | ORU^R01 | ADT^A31 |
| Patient Flag Update | ORU^R01 | ADT^A31 |
| **Scheduling Events** | **HL7v2.x Event from VistA** | **HL7v2.x Event to Epic** |
| Schedule | SIU S12 | SIU S12 |
| Update | SIU S14 | SIU S14 |
| Cancel | SIU S15 | SIU S15 |
| Check In | SIU S14 | SIU S14 |
| Sign In | SIU S14 | SIU S14 |
| Check Out | SIU S14 | SIU S14 |
| Cancel Check In | SIU S14 | SIU S14 |
| Cancel Check Out | SIU S14 | SIU S14 |
| Cancel Sign In | SIU S14 | SIU S14 |
| Notes Update | SIU S14 | SIU S14 |
| Reassign | SIU S14 or S15/S12 | SIU S14 or S15/S12 |
| Reschedule | SIU S15 followed by an SIU S12 | SIU S15 followed by an SIU S12 |
| No Show | SIU S26 | SIU S26 |
| Change Appointment | SIU S14 or S15/S12 | SIU S14 or S15/S12 |
| Add Provider Resource | SIU S14 or S15/S12 | SIU S14 or S15/S12 |
| Remove Provider Resource | SIU S14 or S15/S12 | SIU S14 or S15/S12 |
| Add Provider Resource Request | SIU S14 or S15/S12 | SIU S14 or S15/S12 |
| Remove Provider Resource Request | SIU S14 or S15/S12 | SIU S14 or S15/S12 |
| Appointment Edit Statistics Update x2 | SIU S14 | SIU S14 |
| Appointment Edit Statistics Cancel | SIU S15 | SIU S15 |
| Appointment Edit Statistics No Show | SIU S26 | SIU S26 |
| Appointment Edit Statistics Complete | SIU S14 | SIU S14 |
| End of Day No Show | SIU S26 | SIU S26 |
| End of Day Cancel | SIU S15 | SIU S15 |
| End of Day Complete | SIU S14 | SIU S14 |
| End of Day Update | SIU S14 | SIU S14 |
| Link Order | SIU S14 or S15/S12 | SIU S14 or S15/S12 |
| **Consult Events** | **HL7v2.x Event from VistA** | **HL7v2.x Event to Epic** |
| Create Consult in VistA | ORM^O01 | ORM^O01 |
| Update Consult in VistA | ORM^O01 | ORM^O01 |
| Cancel Consult in VistA | ORM^O01 | ORM^O01 |
| Link Consult to Appointment in Cadence | SIU S14 or S15/S12 | SIU S14 or S15/S12 |
| **Order Events** | **HL7v2.x Event from VistA** | **HL7v2.x Event to Epic** |
| Create Order in VistA | ORM^O01 | ORM^O01 |
| Update Order in VistA | ORM^O01 | ORM^O01 |
| Cancel order in VistA | ORM^O01 | ORM^O01 |
| Link Order to Appointment in Cadence (Not a current need) | SIU S14 or S15/S12 | SIU S14 or S15/S12 |
| **Patient ID Events** | **HL7v2.x Event from VistA** | **HL7v2.x Event to Epic** |
| Identifier Add | ADT^A28 or ADT^A31 | ADT^A28 or ADT^A31 |
| Identifier Update | ADT^A08 | ADT^A31 |
| Merge Patients | ADT^A40 | ADT^A40 |
| **Demographic Events** | **HL7v2.x Event from VistA** | **HL7v2.x Event to Epic** |
| Demographics Add | ADT^A08 | ADT^A31 |
| Demographics Update | ADT^A08 | ADT^A31 |
| **Contact Information Events** | **HL7v2.x Event from VistA** | **HL7v2.x Event to Epic** |
| Contact Add | ADT^A08 | ADT^A31 |
| Contact Update | ADT^A08 | ADT^A31 |
| **Patient Flag Events** | **HL7v2.x Event from VistA** | **HL7v2.x Event to Epic** |
| Patient Flag Add | ORU^R01 | ADT^A31 |
| Patient Flag Update | ORU^R01 | ADT^A31 |

## High-Level Application Design

*Figure 6: MASS High-Level Design* describes the overall MASS solution, including items that will need to be built to achieve full operating capability.

Figure : MASS High-Level Design



MASS is comprised of five layers: Service Subscribers, Applications, MicroServices, Service Consumers and Data Sources.

Service Subscription layer is composed of MASS users. Stakeholders include VA schedulers, administrators and patients.

Application layer includes systems such as Epic, VistA, PCMM, and IAM, which support the functional needs of MASS. Cadence is a scheduling application with modern scheduling capabilities. The VA has purchased this commercial software as-is; custom software development of this product suite is not in scope for the MASS program. Epic will be configured to use HL7v2 messages as the primary integration mechanism to communicate with VA systems. *Section 3.1.3: Application Context Description (Interfaces)* explores these interfaces and the HL7v2 planned communication to Epic. VistA is the legacy scheduling application that supports, stores and integrates with other VHA applications to support their scheduling needs, PCMM provides primary care provider information of the patient application in MASS. IAM supports the SSOe authentication of VA patients into MyChart portal.

MicroServices layer is a set of services that will support all the MASS Adapter needs. Needs are broken down into microservices to reduce duplication and increase reusability across multiple adapters. MASS Adapters are interfaced with MHV Ensemble to enable communication to and from Epic with VistA via HL7.

Service Consumers layer includes Epic and VistA. Epic consumes Patient information andOrder information from VistA through MASS Adapter and VistA consumes scheduling information from Epic to support other VistA applications. Additionally, Epic consumes scheduling status updates from the VetLink Kiosk via VistA.

Data Source/Store layer allows MASS to obtain data from authoritative sources and store it appropriately to meet MASS application needs. *Table 3: System Interfaces* defines MASS system interfaces. *Table 5: MASS Enterprise Services to be Built* details the planned development in support of these interfaces.

Table : MASS Enterprise Services to be Built

| ID | Name | Description | Service or Legacy Code |
| --- | --- | --- | --- |
| 1 | Epic-VistA Data Exchange | Transform and route the communication between Epic and IAM between their various endpoints | Service |
| 2 | VistA-Epic Data Exchange | Transform and route the communication between VistA between their various endpoints | Service |
| 3 | VetLink-Epic Data Exchange | Route and transform the communication format between VetLink and Epic, through VistA | Service |
| 4 | PCMM-Epic Data Exchange | Transform and route the communication between PCMM and Epic between their various endpoints | Service |
| 5 | AudioCare-Epic Data Exchange | Transform and route the communication between AudioCare and Epic between their various endpoints | Service |

MASS Adapters are the services that standardize interfaces to VA systems for reading and writing scheduling data. These adapters are either incomplete for use by MASS or do not exist. *Table 6: MASS Adapters to be Built* describes the MASS Adapters to be built based on the interfaces described in *Table 3: System Interfaces*.

Table : MASS Adapters to be Built

| ID | Name | Description | Service or Legacy Code | External Interface Name | External Interface ID | Internal Interface Name | Internal Interface ID | SDP Sections 1&2 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | EPIC-VistA | Adapter to read and write scheduling and appointment request data to and from VistA | Legacy Code | TBD | TBD | TBD | TBD | TBD |
| 2 | VistA-epic | Adapter to read and write patient data to Epic from VistA | Legacy Code | TBD | TBD | TBD | TBD | TBD |
| 3 | VistA Orders-Epic | Adapter to read and write appointment request data (Orders/Consults) to and from VetLink | Legacy Code | TBD | TBD | TBD | TBD | TBD |
| 4 | PCMM-Epic | Adapter to read care team data from PCMM | Legacy Code | TBD | TBD | TBD | TBD | TBD |
| 5 | AudioCare-Epic | Adapter to read care team data from and send data to AudioCare | Legacy Code | TBD | TBD | TBD | TBD | TBD |

## Application Locations

MASS application will reside in the MAG Cloud. MASS Adapter will interface with MHV Ensemble, which is deployed in VA Terremark Cloud. Epic Cadence will integrate with IAM SSOe to authenticate VA patients into MyChart application.

Figure : Application Locations

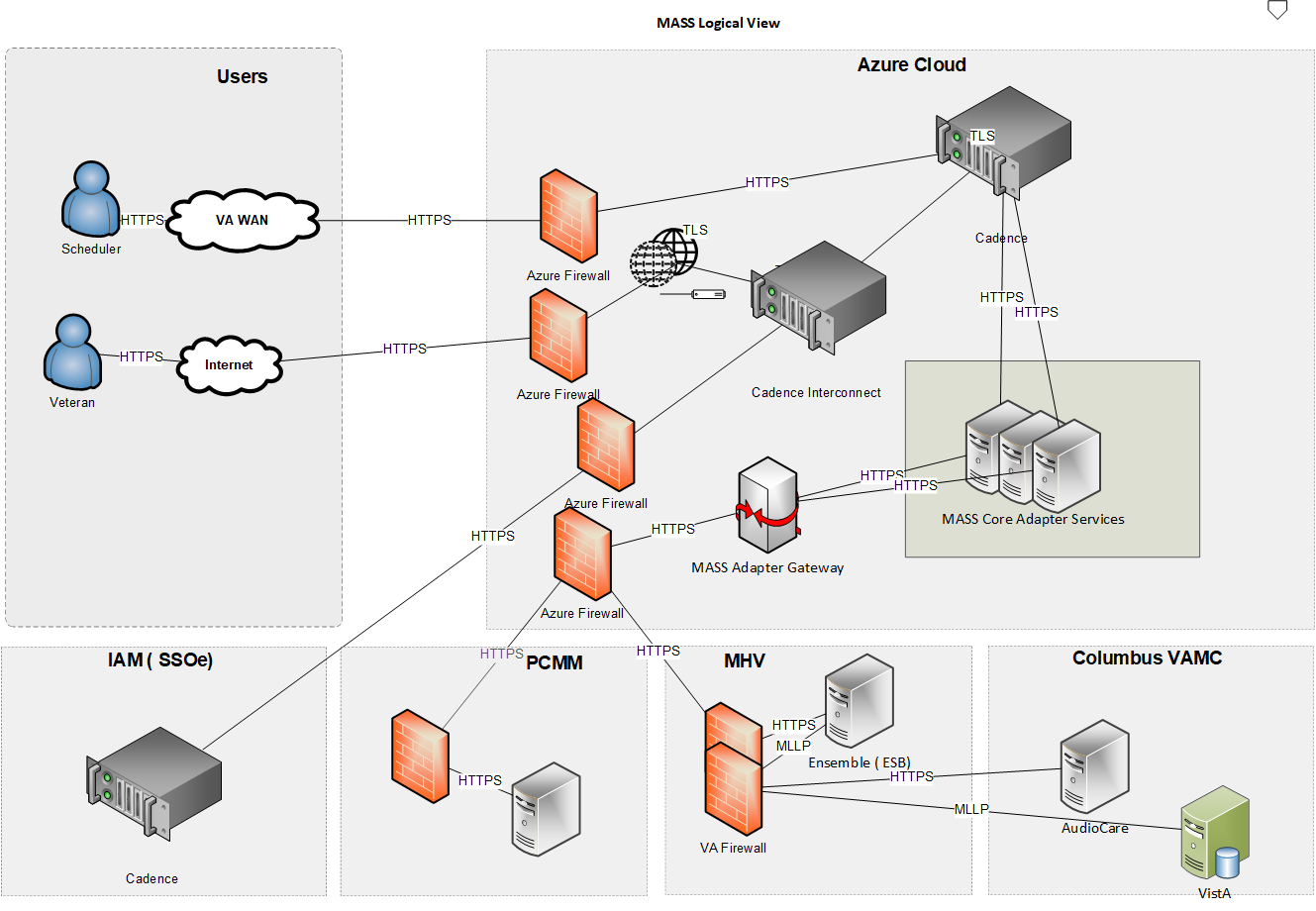


Table 7: Application Locations

| Application Component | Description | Location at Which Component is Run | Type |
| --- | --- | --- | --- |
| Epic Cadence | Scheduling Application | Microsoft Azure | Business Logic/Interface Logic/ Data Logic |
| MASS Adapter | Middleware apps/  Application Servers | Microsoft Azure | Interface Logic |
| VistA | Legacy Health Application | Columbus VistA | Business Logic/Interface Logic |
| AudioCare | Legacy Health Application | Columbus VistA | Business Logic/Interface Logic |
| MHV Ensemble | Enterprise Service Bus | Terremark | Interface Logic |

## Conceptual Data Design

The following sub-sections describe the MASS conceptual data design.

### Project Conceptual Data Model

MASS will use information and data that supports appointment creation, such as patient, location, provider, and enrollment. This data currently resides and will continue to reside in VA databases; however, the authoritative source for some of the data will switch from VistA to a new data store in Epic.

The existing conceptual data model for entities residing in VA databases will not be altered.

Near real-time data in motion between existing VA data centers and Epic will use Representational State Transfer (REST) compliant shared service and HL7 messaging to follow the conceptual data model relationships inherit to those formats, as applicable.

Epic will become the authoritative source for the scheduling data. Also, Epic will store multiple domains, primarily appointment information, with supporting data. The conceptual data model for information stored in Epic does not apply.

### Database Information

A database in this context is simply defined as data at rest. VA will continue to own the databases used for MASS; only the authoritative source, or stewardship, will change. *Table 7: Database Inventory* lists databases relevant to the MASS project, the type of database, and who will retain stewardship once MASS is operational within the scope of Task Order 2.

Table : Database Inventory

| Database Name | Description | Type (Create/Replace/Interface/Modify) | Steward |
| --- | --- | --- | --- |
| Epic | Will contain appointment information with supporting data. | Create, Interface | Epic |
| VistA | The VistA Appointment module is called out separately as MASS will modify the VistA appointments table to add the schedule primary key from Epic. This primary key will be used as a reference to coordinate data synchronization between systems. | Modify, Interface | VA |
| VistA | All domains referenced by MASS, including Locations, Providers, Patient Flags, Kernel Package for Authorization, User Provisioning, Special Needs, Notification templates, Recall List, Registration, Stop Codes, Wait List, and Computerized Patient Record System. | Interface | VA |
| PCMM | Providers assigned to teams for patient care | Interface | VA |
| AudioCare | Provides AudioCare information | Interface | VA |
| VistA | Provision users as records in the MASS Authentication for Patients | Interface | VA |

## User Interface Data Mapping

The following sub-sections describe the MASS interface data mapping.

**Application Screen Interface**

All screen interfaces will be delivered as part of Cadence. The data displayed on the screen will be mapped primarily to Epic storage, and in a few cases, from live data being pulled or embedded from services interfacing with VA databases. There will not be any new screens specifically built in Cadence for the MASS project.

**Application Report Interface**

All report interfaces will be delivered as part of Epic; the data in the reports will be mapped from Epic storage. MASS will use out-of-box reports, as applicable, and will likely configure new reports specific to MASS requirements, but those have not yet been defined.

**Unmapped Data Element**

Elements mapped to services and databases are not complete; therefore, unmapped data elements, if any, are not listed as part of this section.

## Conceptual Infrastructure Design

This is covered in *Section 4.0: System Architecture.*

## System Criticality and High Availability

This is covered in *Section 4.0: System Architecture.*

## Special Technology

This is covered in *Section 4.0: System Architecture.*

## Technology Locations

This is covered in *Section 4.0: System Architecture.*

## Conceptual Infrastructure Diagram

This is covered in *Section 4.0: System Architecture.*

**Location of Environments and External Interfaces**

This is covered in *Section 4.0: System Architecture.*

**Conceptual Production String Diagram**

This is covered in *Section 4.0: System Architecture.*

# System Architecture

The technical design details within this document collectively represent the system architecture for the infrastructure deployment in support of the DANMSS ilot site implementation; however, the design is not limited to pilot site implementation. It is designed to scale and support the nationwide implementation of the Epic Cadence scheduling system without requiring major architectural changes, such as those that would require re-certifying the environment. It supports the intent of the pilot site success evaluation to determine the feasibility of a national implementation by using the exact same system design intended to support national implementation. The only significant difference is the smaller sizing of the virtual machines (VMs) for the pilot implementation in accordance with the sizing analysis.

## Hardware Architecture

TeamSMS will leverage VA’s existing MAG Cloud environment, subscription and dev/test model. VA also has an existing ExpressRoute connection. The ExpressRoute connection routes through the Trusted Internet Connection (TIC) and is accredited by the VA.

TeamSMS will implement the solution in the PROD MAG for all production workloads. TeamSMS will also leverage other Azure subscriptions from MAG such as NPROD and DEV/TEST. The NPROD subscription will host all MASS non-production environments. The DEV/TEST subscription will host a development environment, so TeamSMS can develop VistA and Epic integration connectors.

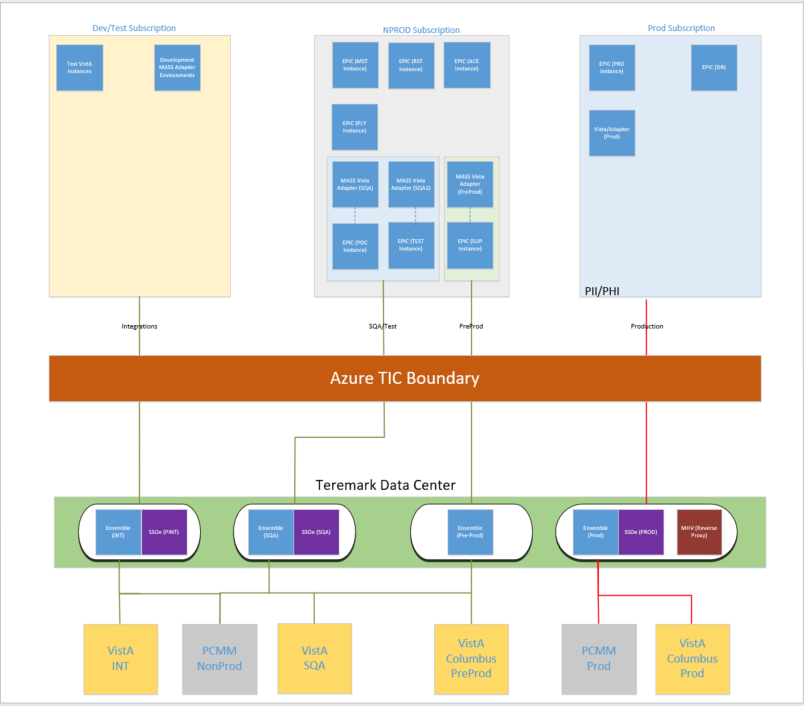
TeamSMS will develop and build multiple environments to ensure development efforts do not affect production systems. These environments will be used for testing, classroom training, support, etc.

These MAG environments which host Epic and MASS Adapter, will communicate with the Ensemble instance in the VA Terremark datacenter to act as a gateway and allow the MASS MAG environments to communicate with VA Columbus VistA instance and other systems such as AudioCare and PCMM.

TeamSMS will leverage VA’s EDE environment as a development environment while personnel on the program receive PIV. Once all personnel receive PIV access, TeamSMS will migrate EDE VMs to the MASS instance in the DEV/TEST environment.

*Figure 8: MASS High-Level Architecture Overview* provides an illustrated summary of the MASS architecture. Detailed architecture diagrams are included in other sections throughout this document.

Figure : MASS High-Level Architecture Overview



### Compute

MASS will use Azure VMs within the MAG cloud in VA’s subscription enrollment. This section outlines terms and important concepts used for designing MASS or other infrastructure VM details.

* **Azure VMs:** Provide the flexibility of virtualization without having to buy and maintain physical hardware. VMs still require maintenance tasks such as configuration, patch application, and software installation
* **Business Continuity:** Keeps data and workloads running to Service Level Agreement (SLA) standards in case of serious incidents or disasters. Azure uses a variety of business continuity methods (e.g., availability sets consisting of fault domains and update domains).
  + Availability Sets: A grouping system for VMs that minimizes downtime due to unplanned outages. It ensures that VM instances are placed far enough apart from one another to ensure continuity during an event that would otherwise bring that system down
  + Fault Domain: Defines the group of VMs that share a common power source and network switch. By default, VMs configured within an availability set are separated across up to three fault domains for Resource Manager deployments (two fault domains for Classic). While placing VMs into an availability set does not protect an application from operating system or application-specific failures, it does limit the impact of potential physical hardware failures, network outages, or power interruptions
  + Update Domains: For a given availability set, five non-user-configurable update domains are assigned by default to indicate groups of VMs and underlying physical hardware that can be rebooted at the same time. Resource Manager deployments can then be increased to provide up to 20 update domains. When more than five VMs are configured within a single availability set, the sixth virtual machine is placed into the same update domain as the first virtual machine, the seventh in the same update domain as the second virtual machine, and so on. The order of update domains being rebooted may not proceed sequentially during planned maintenance, but only one update domain is rebooted at a time. A rebooted update domain is given 30 minutes to recover before maintenance is initiated on a different update domain

TeamSMS will build MASS according to the VM specifications listed in *Table 8: VM Specifications.*

Table : VM Specifications

| Area | Type | Application/Service | vCPU | vRAM (GB) | Quantity | Total vCPU | Total vRAM (GB) |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Non-Prod | Cogito | BusinessObjects Enterprise Test Server | 4 | 16 | 1 | 4 | 16 |
| Non-Prod | Cogito | Clarity & Clarity Console Test Server | 4 | 32 | 1 | 4 | 32 |
| Non-Prod | ODB | Non-Production Database Server | 6 | 32 | 1 | 6 | 32 |
| Non-Prod | Presentation | Citrix Controllers | 2 | 4 | 1 | 2 | 4 |
| Non-Prod | Presentation | Citrix Storefront | 2 | 4 | 1 | 2 | 4 |
| Non-Prod | Presentation | Citrix License Server | 1 | 4 | 1 | 1 | 4 |
| Non-Prod | Presentation | Citrix XenApp Servers - Hyperspace | 6 | 30 | 3 | 18 | 90 |
| Non-Prod | Presentation | Citrix XenApp Servers - Multipurpose | 2 | 4 | 1 | 2 | 4 |
| Non-Prod | Presentation | Hyperspace Web Servers | 2 | 4 | 1 | 2 | 4 |
| Non-Prod | Presentation | Hyperspace Web Servers | 2 | 4 | 1 | 2 | 4 |
| Non-Prod | Presentation | Web Servers: MyChart | 1 | 4 | 1 | 1 | 4 |
| Non-Prod | Service | Business Continuity Access (BCA) Servers | 1 | 4 | 1 | 1 | 4 |
| Non-Prod | Service | Interconnect Servers | 2 | 4 | 1 | 2 | 4 |
| Non-Prod | Service | Kuiper Server | 1 | 4 | 1 | 1 | 4 |
| Non-Prod | Service | Multipurpose SQL Server | 1 | 4 | 1 | 1 | 4 |
| Non-Prod | Service | Print Format Servers | 2 | 4 | 1 | 2 | 4 |
| Non-Prod | Service | Print Format Servers | 2 | 4 | 1 | 2 | 4 |
| Non-Prod | Service | Reverse Proxy Servers | 1 | 4 | 1 | 1 | 4 |
| Non-Prod | Service | Web BLOB Servers | 1 | 4 | 1 | 1 | 4 |
| Production | Cogito | BusinessObjects Enterprise Server | 4 | 16 | 1 | 4 | 16 |
| Production | Cogito | Clarity & Clarity Console Server | 6 | 56 | 1 | 6 | 56 |
| Production | ODB | Disaster Recovery Server | 4 | 84 | 1 | 4 | 84 |
| Production | ODB | Production Database Server | 4 | 84 | 1 | 4 | 84 |
| Production | Presentation | Citrix Controllers | 2 | 4 | 2 | 4 | 8 |
| Production | Presentation | Citrix Storefront | 2 | 4 | 2 | 4 | 8 |
| Production | Presentation | Citrix License Server | 1 | 4 | 1 | 1 | 4 |
| Production | Presentation | Citrix XenApp Servers - Hyperspace | 6 | 30 | 20 | 120 | 600 |
| Production | Presentation | Citrix XenApp Servers - Multipurpose | 2 | 4 | 2 | 4 | 8 |
| Production | Presentation | Hyperspace Web Servers | 2 | 4 | 4 | 8 | 16 |
| Production | Presentation | Web Servers: MyChart | 1 | 4 | 2 | 2 | 8 |
| Production | Service | BCA Servers | 2 | 4 | 2 | 4 | 8 |
| Production | Service | Interconnect Servers | 2 | 4 | 4 | 8 | 16 |
| Production | Service | Kuiper Server | 2 | 4 | 1 | 2 | 4 |
| Production | Service | Multipurpose SQL Server | 2 | 8 | 1 | 2 | 8 |
| Production | Service | Print Format Servers | 4 | 4 | 4 | 16 | 16 |
| Production | Service | Reverse Proxy Servers | 1 | 4 | 2 | 2 | 8 |
| Production | Service | System Pulse Server | 4 | 4 | 1 | 4 | 4 |
| Production | Service | Web BLOB Servers | 1 | 4 | 2 | 2 | 8 |
| **Total** | | |  |  | **73** | **252** | **1186** |

### Citrix XenApp Environment

**Citrix Design**

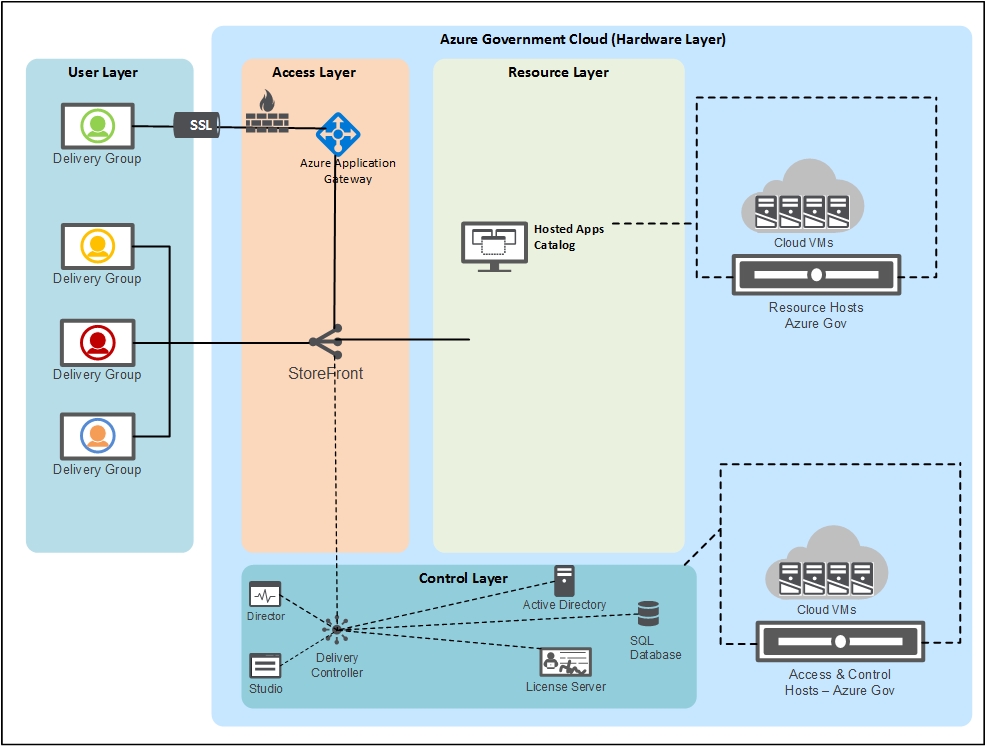
For the pilot site deployment, the resources required for Citrix XenApp are minimal; therefore, some of the design considerations in this section are not relevant. However, if the system is scaled for a national roll-out, the number of XenApp servers will far outpace the number of ancillary application servers, then performance in this tier becomes critical for maintaining a good end-user experience. For this reason, TeamSMS invested time to collaborate with Epic’s dedicated Client Systems engineers, Citrix directly, and Convergence (the largest Citrix reseller for federal market), to design a solution that will serve the needs of the pilot site as well as be scaled to support national loads without requiring a major re-design.

The Citrix XenApp system architecture includes careful consideration of the following:

* Specific CPU support requirements
* Large-scale deployment
* Non-siloed environment

Based on extensive in-house performance and scalability testing, and in cooperation with Citrix, Epic developed a list of supported CPUs and expected user sessions for each CPU. *Figure 9: Citrix XenApp Conceptual Design* shows the interaction between the component layers of the Citrix XenApp installation.

Figure : Citrix XenApp Conceptual Design

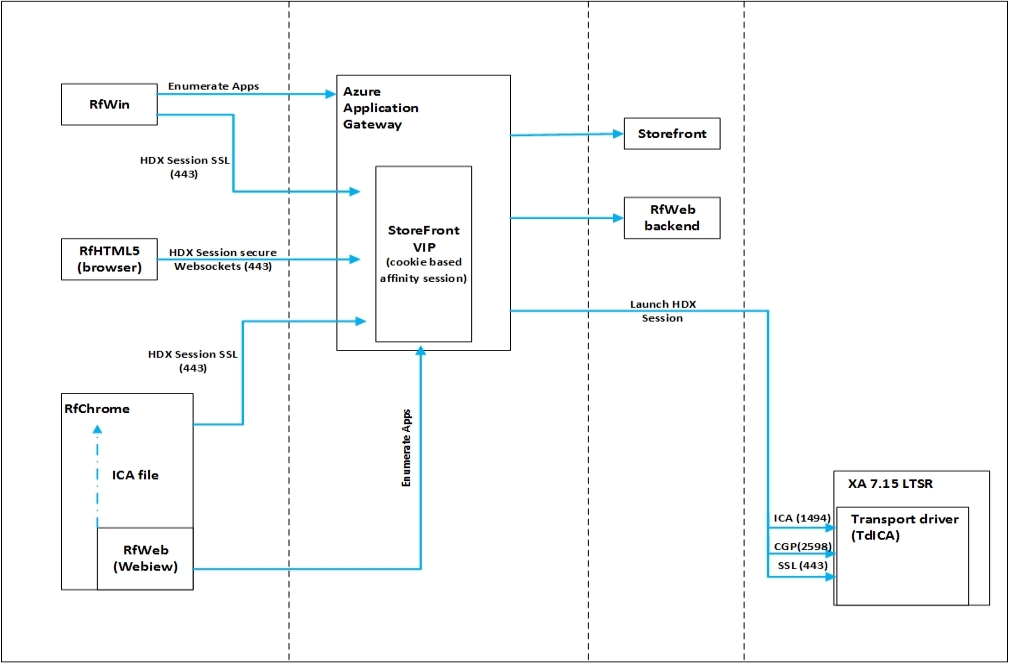


**Pilot Site Design**

Epic Cadence was extensively tested with Citrix, and based on those findings, TeamSMS will deploy XenApp 7.6 LTSR servers running with eight virtual CPUs and 28 gigabytes (GBs) of RAM, coinciding with the Azure DS4v2 component. This VM configuration can support approximately 36 users concurrently per VM. To support the 300-pilot site concurrent Citrix XenApp sessions and enable server redundancy and a test/maintenance XenApp server, TeamSMS will provision 20 XenApp 7.6 servers. TeamSMS will also provision an additional 6 VMs for the Citrix XenApp infrastructure servers (i.e., Delivery Controller/StoreFront/License servers).

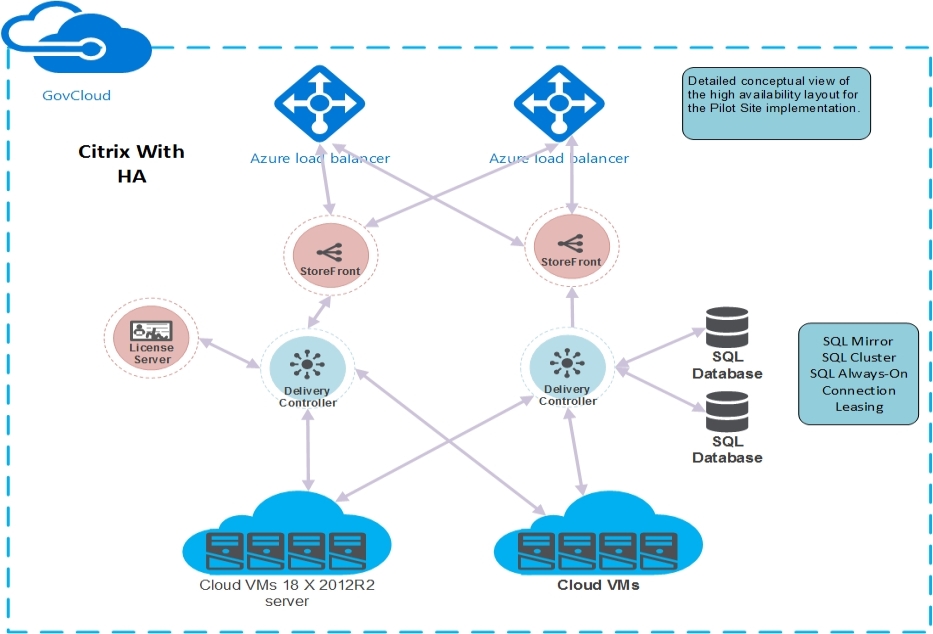
*Figure 10: Citrix Access Flow* shows the access flow from the schedulers to the Epic application.

Figure : Citrix Access Flow



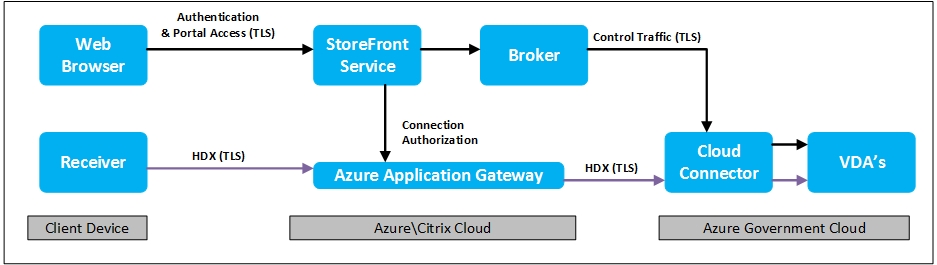
*Figure 11: Citrix Components for High Availability Design* shows the detailed conceptual view of the high availability layout for the pilot site implementation.

Figure : Citrix Components for High Availability Design



*Figure 12: Citrix Access High-Level Design* shows the detailed conceptual view of the pilot site implementation.

Figure : Citrix Access High-Level Design



### Management Systems

TeamSMS will use existing VA management systems, including all Continuous Readiness in Information Security Program services provided by VA. Additional cloud management systems will also be used upon availability from the VA Enterprise Cloud program.

### MyChart Environment

MyChart will use VA’s reverse proxy environment that is behind the TIC. All users coming in from the public internet to access MyChart will first be routed to the TIC, then to VA’s Reverse Proxy, then across the Business Partner Extranet to the MAG. TeamSMS will create an internet/public web server request through the Enterprise System Change Control Board (ESCCB) change request system for the MyChart environment.

*Figure 13: MyChart Environment* shows the high-level view of the MyChart environment and how the data flows.

Figure : MyChart Environment



### Storage

A general-purpose storage account includes access to Azure storage services, such as tables, queues, files, blobs and Azure VM disks. It also provides a unique namespace for working with blobs, queues, and tables.

Azure storage services may include the following:

* **Blob:** Stores file data. A blob can contain any type of text or binary data, such as a document, media file, or application installer. Every data blob is organized into an object called a container within each storage account. A storage account can contain any number of containers (but it must have at least one), and a container can contain any number of blobs. The types of blobs are as follows:
  + Block blobs are optimized for streaming and storing cloud objects, and are a good choice for storing documents, media files, and backups
  + Page blobs are optimized for representing Infrastructure as a Service (IaaS) disks and supporting random writes. An Azure VM IaaS disk is a virtual hard disk that is stored as a page blob
  + Append Blobs are optimized for storing data that needs to be added to at a later date. Adding data to an append blob always places the data at the end of the blob, without modifying the existing data within the blob. This is ideal for data such as logs and audits
* **Table:** Stores structured datasets. Table storage is a NoSQL key-attribute data store, which allows for development and fast access to large quantities of data
* **Queue:** Provides messaging for workflow processing and for communication between components of cloud services
* **File:** Offers shared storage for legacy applications that leverage the SMB 2.1/3.0 protocol. Azure VMs and cloud services can share file data across application components by using mounted shares. On-premises applications can also access data in a share via the File service REST Application Programming Interface (API)

A general-purpose storage account has two performance tiers:

* **Standard:** Allows users to store tables, queues, files, blobs and Azure VM disks. This tier is hosted on Hard Disk Drives
* **Premium:** Currently only supports Azure VM disks. This tier is hosted on Solid State Drives

A blob storage account is a specialized storage account for storing unstructured data as blobs (objects) in Azure Storage. Blob storage accounts are similar to general-purpose storage accounts and share all the durability, availability, scalability, and performance features and API consistency for block blobs and append blobs. Blob storage accounts are recommended for applications requiring only block or append blob storage.

A blob storage account has two tiers:

* **Hot access**: Indicates that the objects in the storage account will be more frequently accessed. This allows users to store data at a lower access cost.
* **Cool access**: Indicates that the objects in the storage account will be less frequently accessed. This allows users to store data at a lower data storage cost.

Storage accounts are available with four redundancy types:

* Locally redundant storage (LRS)
* Geo-redundant storage (GRS)
* Zone redundant storage (SRS)
* Read access geo-redundant storage (RA-GRS)

**Azure Managed Disks:** Azure Managed Disks simplifies disk management for Azure IaaS VMs by managing the [storage accounts](https://docs.microsoft.com/en-us/azure/storage/common/storage-introduction) associated with the VM disks. You only have to specify the type ([Premium](https://docs.microsoft.com/en-us/azure/virtual-machines/windows/premium-storage) or [Standard](https://docs.microsoft.com/en-us/azure/virtual-machines/windows/standard-storage)) and the size of disk you need, and Azure creates and manages the disk for you.

Azure Disks are designed for 99.999% availability. Rest easier knowing that you have three replicas of your data that enables high durability. If one or even two replicas experience issues, the remaining replicas help ensure persistence of your data and high tolerance against failures. This architecture has helped Azure consistently deliver enterprise-grade durability for IaaS disks, with an industry-leading ZERO% Annualized Failure Rate.

The MASS program will use Managed Disks for all virtual machine disks, and data disks. Storage accounts will be used for virtual machine diagnostic logging. All managed disks will be encrypted using Azure Disk Encryption outlined below. All diagnostic storage accounts will be encrypted using the Azure Storage Encryption Service also outlined below. PII and PHI data will be stored in managed disks while the storage account will contain no PII or PHI data.

**Azure Disk Encryption:** Provides Azure customers with the capability to encrypt their virtual machine data on a multitenant, shared infrastructure, and use Azure Key Vault to manage the encryption keys for their encrypted data.

**Azure Storage Service Encryption (SSE):** Helps protect and safeguard data at rest to meet security and compliance commitments. With this feature, Azure Storage automatically encrypts data prior to writing to storage and decrypts prior to retrieval. The encryption, decryption, and key management are transparent to users. SSE can be used for block blobs, page blobs and append blobs. It works for:

* General purpose storage accounts and blob storage accounts
* Standard storage and premium storage MS (separate storage for databases)
* Redundancy levels (e.g., LRS, ZRS, GRS, RA-GRS)
* Azure Resource Manager storage accounts (but not classic)

*Table 9: Storage Performance Metrics* details the storage performance metrics the MASS design will use.

Table : Storage Performance Metrics

|  | Production/DR ODB | Cogito Clarity |
| --- | --- | --- |
| Read IOPS | 800 | 2,400 |
| Read Throughput (MB/sec) | 100 | 150 |
| Write Operations per 80-Second Write Burst | 20,000 | N/A |
| Write Volume per 80-Second Burst (MB) | 400 | N/A |
| Write IOPS | N/A | 800 |
| Write Throughput (MB/sec) | N/A | 50 |
| Minimum Storage Area Network Write Cache Requirement (MB) | 400 | 300 |
| Average Read Response Time Requirement (MS) | < 12 | < 150 |
| Average Write Response Time Requirement (MS) | < 1.0 | < 5.0 |

For acceptable end-user performance, the production operational database (ODB) read and write time requirements are as follows:

* For randomly placed reads to database files (measured at the system call level):
  + Average read latencies must be 12 MS or less
  + 99% of read latencies must be below 60 MS
  + 99.9% of read latencies must be below 200 MS
  + 99.99% of read latencies must be below 600 MS
* For randomly placed writes to database files (measured at the system call level):
  + Average write latencies must be 1.0 MS or less

*Table 10: Storage Capacity Metrics* details the storage capacity metrics the MASS design will use.

Table : Storage Capacity Metrics

| Storage Use | Server | Usable Space (GB) |
| --- | --- | --- |
| Production Data | Production Database Server | 700 |
| Production Journal Files | Production Database Server | 300 |
| Production Application Files | Production Database Server | 200 |
| Production Support Environment  (Clone or Snapshot of Production) | Production Support Database Server | 700 |
| Test and Build Environments | Test Database Server | 400 |
| Training Environments | Training Database Server | 300 |
| Multipurpose Virtual Servers | Physical Hosts for Production and Non-Production Virtual Servers | 1000 |
| Operational Database Virtual Servers | Physical Hosts for Production and Non-Production Virtual Servers | 300 |
| Clone or Snapshot for Nightly Backups | N/A | 1200 |
| Cogito Clarity Data | Clarity Server | 700 |
| Cogito Clarity Test Data | Clarity Test Server | 700 |
| Multipurpose SQL Server Data | Multipurpose SQL Server | 200 |
| Citrix XenApp Virtual Servers | Citrix XenApp Servers | 1700 |
| Cogito Virtual Servers | Physical Hosts for Production and Non-Production Virtual Servers | 320 |
| DR Data | Disaster Recovery Server | 700 |
| DR Journal Files and Application Files | Disaster Recovery Server | 500 |
| DR ODB Virtual Server | Physical Hosts for Production and Non-Production Virtual Servers | 200 |
| DR Clone or Snapshot for Nightly Backups | N/A | 1200 |
| Web BLOB Data | Web BLOB Servers | 600 |
| BusinessObjects Enterprise Data | BusinessObjects Enterprise Servers | 100 |
| Production and Non-Production Service Servers | Production and Non-Production Service Servers | 300 |
| **Total (TB)** |  | **12.4** |

### Cloud Management

The Azure Resource Manager (ARM) REST API was developed to replace Azure Service Management (ASM) as the authoritative method to manage Azure resources. ARM supports both desired state configuration and role-based access control (RBAC), while providing a pluggable model allowing new Azure services to be cleanly integrated. The Azure Portal and the ARM mode of the Azure PowerShell cmdlets both use ARM.

ARM introduces the concept of a resource group which is a collection of individual Azure resources. A resource group is associated with a specific Azure region but may contain resources from more than one region.

ARM supports the use of a parameterized resource group JavaScript Object Notation (JSON) template file that can be used to create one or more resource groups along with their individual resources. The deployment of a resource group uses desired state configuration. ARM ensures that the resources are deployed in accordance with the appropriately parameterized template file for the resource group. Resource providers exist for many types of Azure resources, and more Azure services are currently adding ARM support, gradually migrating from the legacy ASM model.

ARM supports RBAC, and this support is expressed in the Azure Portal and the ARM mode of the Azure PowerShell and Azure command line interface (CLI) cmdlets. ARM provides several core roles – Owner, Contributor, and Reader.

VA developed a role-based access control (RBAC) model that TeamSMS will use to deploy MASS into MAG cloud.. In this multi-tenant environment, MASS administrators will only be able to affect changes to resources in the MASS resource group and will not have the ability to manage resources that exist outside of the MASS boundary.

Epic’s built-in tools will be used to perform additional MASS application management actions.

### Backup

The Epic architecture contains multiple servers, each with their own backup requirements. Requirements might vary for different types of data within a single server. For example, it's common for a single server to have a different backup scheme for operating system files versus database files. The backup requirements are largely dictated by the following considerations:

* Recovery Point Objective (RPO)
* Recovery Time Objective
* Point in time consistency requirements
* Tolerance for performance degradation or downtime for backups

The Epic system is a client-server architecture. There are separate servers for the interactive user (ODB) databases and the analytical reporting databases. There are also many Web & Service servers for print formatting, image storage, terminal services and other tasks. Some of these servers are optional, depending on functionality. *Figure 14: Epic System* Architectureshows Epic’s typical client-server, system architecture. The components that are most important to back up are highlighted with yellow and peach backgrounds.

Figure : Epic System Architecture

https://galaxy.epic.com/Handler/ImageHandler.ashx?filename=U:%5CImages%5C2010%20RELEASE%5C1300001To1400000%5C1306548.png&ver=1380525157310

The importance of backups for a given server and its data strongly mirrors the RPOs for that server.

As part of the standard change control architecture, there are non-production versions of most servers. These non-production servers typically have a reduced need for backups. This is particularly true of the non-production Web & Service servers, many of which can be rebuilt instead of requiring backups for a restore.

The content on the machines accessed by users for running Hyperspace, the Epic client software, is typically not backed up individually. In the event of an issue with these components, most sites chose to either reinstall or back up a single site-wide image for each unique component. The Hyperspace client itself doesn't store critical data on the local machine. However, users might use these machines for more than Epic software alone. In such cases, policies should be put in place that restrict users from saving other important content locally on these machines if they are not backed up.

Referencing the same storage capacity and areas in *Table 10: Storage Capacity Metrics*, the backup frequencies are detailed below in *Table 11: MASS Backup Frequencies*:

Table : MASS Backup Frequencies

|  |  |  |
| --- | --- | --- |
| Storage Use | Server | Backup Frequency |
| Production Data | Production Database Server | Nightly |
| Production Journal Files | Production Database Server | Nightly |
| Production Application Files | Production Database Server | Nightly |
| Production Support Environment  (Clone or Snapshot of Production) | Production Support Database Server | N/A |
| Test and Build Environments | Test Database Server | Nightly |
| Training Environments | Training Database Server | Nightly |
| Multipurpose Virtual Servers | Physical Hosts for Production and Non-Production Virtual Servers | After upgrades |
| Operational Database Virtual Servers | Physical Hosts for Production and Non-Production Virtual Servers | After upgrades |
| Clone or Snapshot for Nightly Backups | N/A | N/A |
| Cogito Clarity Data | Clarity Server | Nightly |
| Cogito Clarity Test Data | Clarity Test Server | Weekly |
| Multipurpose SQL Server Data | Multipurpose SQL Server | Nightly |
| Citrix XenApp Virtual Servers | Citrix XenApp Servers | After upgrades |
| Cogito Virtual Servers | Physical Hosts for Production and Non-Production Virtual Servers | After upgrades |
| DR Data | Disaster Recovery Server | Nightly |
| DR Journal Files and Application Files | Disaster Recovery Server | Nightly |
| DR ODB Virtual Server | Physical Hosts for Production and Non-Production Virtual Servers | After upgrades |
| DR Clone or Snapshot for Nightly Backups | N/A | N/A |
| Web BLOB Data | Web BLOB Servers | Nightly |
| BusinessObjects Enterprise Data | BusinessObjects Enterprise Servers | Nightly |
| Production and Non-Production Service Servers | Production and Non-Production Service Servers | Nightly |

Azure Backup is the Azure-based service MASS will use to back up (or protect) and restore data in the MAG cloud. It will replace existing on-premises or off-site backup solutions with a cloud-based solution that is reliable, secure, and cost-competitive. It offers multiple components that are downloadable and can be deployed on appropriate computers, servers, or in the cloud. The deployed component, or agent, depends on what needs to be protected. All Azure Backup components can be used to back up data to a Recovery Services vault in Azure.

Azure Backup feature that MASS will utilize include:

* **Multiple storage options:** An aspect of high-availability is storage replication. Azure Backup offers two types of replication: locally redundant storage and geo-redundant storage.
  + **LRS:** Replicates data three times (i.e., creates three copies of selected data) in a paired datacenter in the same region. It is a low-cost option for protecting data from local hardware failures
  + **GRS:** Replicates data to a secondary region (i.e., location hundreds of miles away from the primary location of the source data). It costs more than LRS, but GRS provides a higher level of durability for your data, even if there is a regional outage
* **Data encryption:** Allows for secure transmission and storage of data in the public cloud. A user stores the encryption passphrase locally, and it is never transmitted or stored in Azure. If it is necessary to restore any of the data, only the user has the encryption passphrase, or key
* **Application-consistent backup:** Whether backing up a file server, virtual machine, or SQL database, a specified recovery point has all required data to restore the backup copy. Azure Backup provides application-consistent backups, which ensures additional fixes are not needed to restore the data. Restoring application consistent data reduces the restoration time, allowing you to quickly return to a running state
* **Long-term retention:** Azure doesn't limit the length of time data remains in a Backup or Recovery Services vault. You can keep data in a vault for as long as you like. Azure Backup has a limit of PORT recovery points per protected instance

MASS will use Azure Backup to protect individual servers in the event of an application/operating system level crash or corruption of data, and to satisfy a limited DR requirement. Azure Backup will use GRS storage to copy backups to an alternate region where they will be available for restore as required.

## Software Architecture

MASS will provide Software as a Service in the form of Epic - a commercial off the shelf (COTS) product. Epic is composed of various applications and modules and is a patient-centered electronic medical record system.

To facilitate meeting the business needs of the VHA, TeamSMS will employ a multitiered environment strategy. As detailed in the Change Management processes adopted by MASS, configuration change migrations will be accomplished by different groups of people.

The envisioned solution architecture will follow the COTS-VistA scheduling archetype supporting Columbus VAMC--but with enterprise shared services-middleware abstracting the presentation layer from the rules engine and VistA data. The MASS architecture is a highly available platform designed to support scheduling needs; and exchange scheduling and patient data sets between VistA and Epic Cadence.

### MASS Adapter

The MASS Adapter framework addresses the need for growing data exchanges, unpredictable traffic patterns, and the demand for faster response times. It also meets the following requirements:

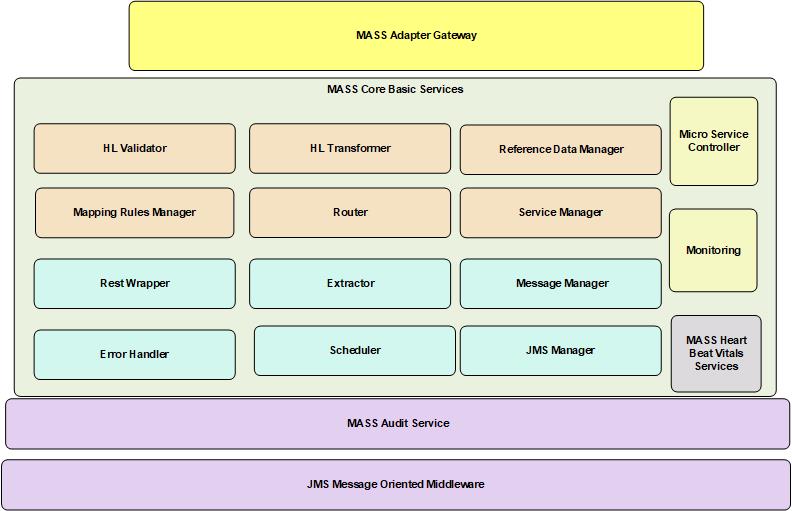
* security/scalability
* operational efficiency
* audit/analytics

MASS Adapter services are autonomous, reusable, and discoverable. It provides stateless functionality with necessary granularity and can also be part of a composite service.

MASS Adapter provides a highly reliable and scalable infrastructure to facilitate data exchange with minimal support, administration costs, and flexibility. It includes a variety of foundational microservices to facilitate data exchange. These services are developed as a suite of independently deployable, small, modular services in which each service runs a unique process and communicates through a well-defined, lightweight mechanism to serve a business goal. The services are thread safe with multiple asynchronous nodes that work through a load balancer. Secure data in transit and confidential data is encrypted. Services are developed in Node.js and Java.

*Figure 15: MASS Adapter Overview* shows a high-level overview of the MASS Adapter framework.

Figure : MASS Adapter Overview



Each MASS Adapter service is comprised of one or more microservices listed below.

* **ReceiveOverHAPI:** Verifies and forwards incoming data for further processing and communication. It’s most important role is providing security. The MASS Adapter ReceiveOverHAPI service uses HTTPS for all web-services invocations and acts as a proxy for communication, handles HL7 using HAPI API and acknowledges the sender for data received from Epic Cadence and MHV Ensemble
* **HL7 Validator:** Validates the incoming message based on the producing system standards and sends an acknowledgement accordingly. It validates the message if the message is handled/processed by the HL7 transformer, based on the consuming system standards, prior to posting the message to consuming system. Validation will be managed by HAPI and ER7 parsers
* **HL7 Transformer:** HL7 Transformer transforms the incoming message to meet the consuming system standards. Transformations will be managed by HAPI and ER7 parsers
* **Scheduler:** Calls a HTTP/HTTPS service based on a CRON Schedule. This would enable execution of CRON jobs that can execute a HTTP call with an asynchronous response or synchronization requests
* **File Transfer Service:** Extracts batch data from VistA and PCMM, process them and send to Epic
* **Data Store Service:** Stores the original and transformed HL7 data, in a database, to ensure all the transactions are stored temporarily
* **MASS Heart Beat Vitals and Monitoring Service:** Calculates throughput, processing speed, and alerts for any performance degradation or connection issues to ensure service stability
* **JMS Message Oriented Middleware**: Implemented in Spring Framework, Apache ActiveMQ configurations will be used to create the routing and brokering of messages. It is implemented in Java and deployed in a JEE container

The MASS service oriented architecture is implemented using microservices pattern, to ensure the application is light weight, reusable and loosely coupled for effective maintenance.

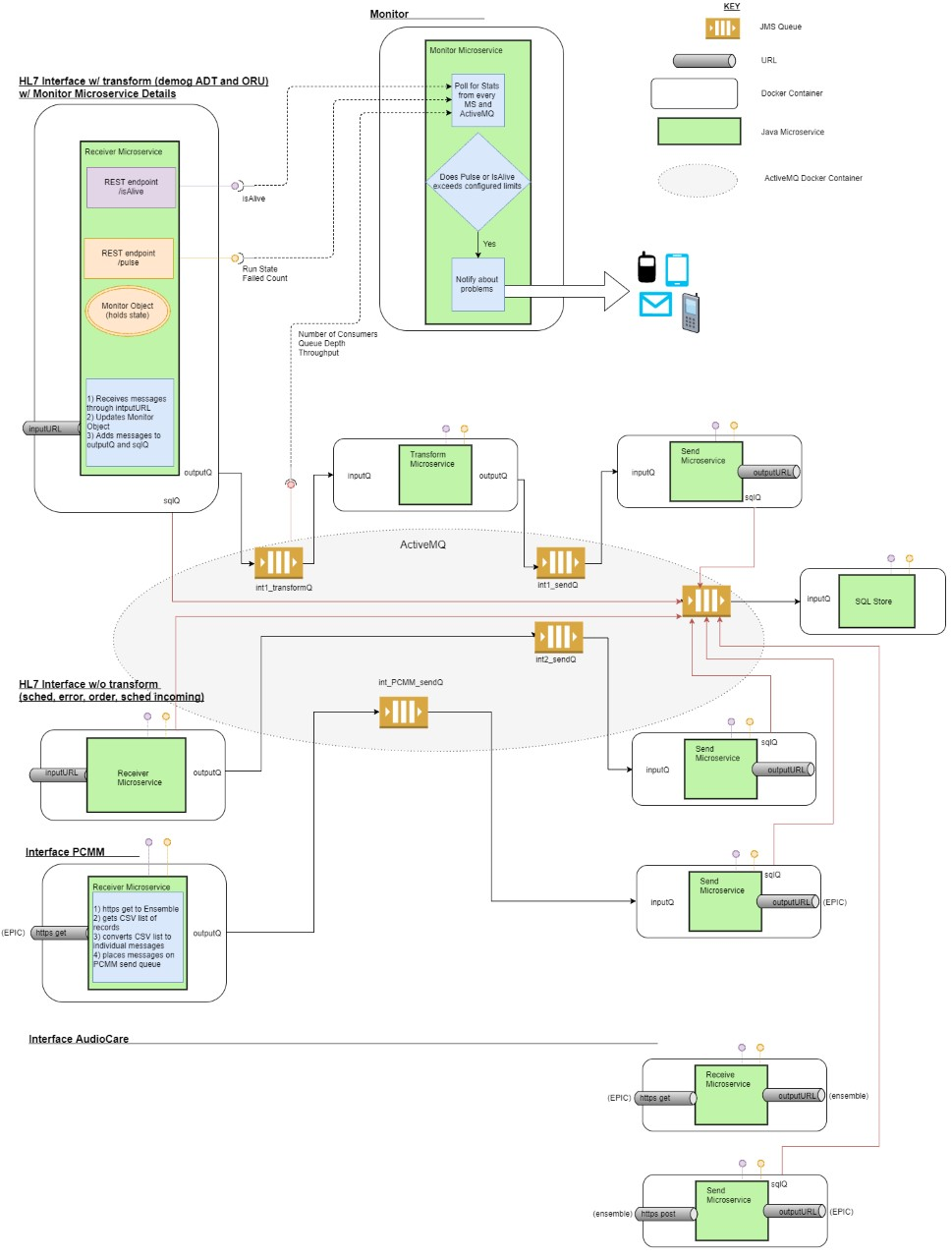
The microservices are combined to create workflows (Lego Model), to exchange Scheduling Data, Appointment Status Data, Eligibility Data, Patient Identity, Demographics, Consults Data, Patient data, Orders Data, Primary Care data and AudioCare information between MASS and VistA.

The microservices developed to meet the above exchanges are

* **ReceiveOverHAPI Service:** This Service will receive data from VistA/ Epic, depending on the soured system and act as Gateway, to ensure the TLS security is in place, and also to validate and acknowledge the Source system, once the data is stored to a queue for further processing. Each workflow will contain an instance of this microservice, to ensure the data received is from an identified and safe endpoint, using 2-way SSL handshake.
* **Message Oriented Middleware Service:** This service will receive the data, store the data temporarily in a JMS queue using ActiveMQ and allow the consuming application, to further process the data
* **Transform Service:** The message to be transformed is picked up from the configured input queue. The Data is converted to xml format using HAPI Package. Transform service uses the configured xslt to transform the xml message to the desired xml format. Finally, it converts the xml back into hl7 pipe based message and sends it to the output queue
* **Data Store Service:** This service stores HL7 message received by the adapter to a MS SQL Server Database. It also stores the acknowledgements from destination, to ensure replay and add temporary persistence (90 days) for maintaining the sequence of changes in Epic and VistA
* **File Transfer Service:** File Transfer Service is used to exchange batch/bulk data between VA systems such as AudioCare, PCMM and Epic
* **SendAndGetAck Service:** This service picks the message from the configured input queue and sends the HL7 messages to the configured destination endpoint using Https Over HAPI. Transformation of the message structure, if needed is done prior to the message being posted to input queue
* **Microservice Monitoring Service**: This service polls each of the other microservices to ensure each is alive as well as gather statistics for each microservice. In addition, it talks directly to ActiveMQ to get queue depths and throughput. The monitor uses this gathered information to decide whether emails need to be sent based on limits that can be configured in the service

*Figure 16: MASS Adapter Orchestration* represents microservices orchestration in the MASS Adapter application:

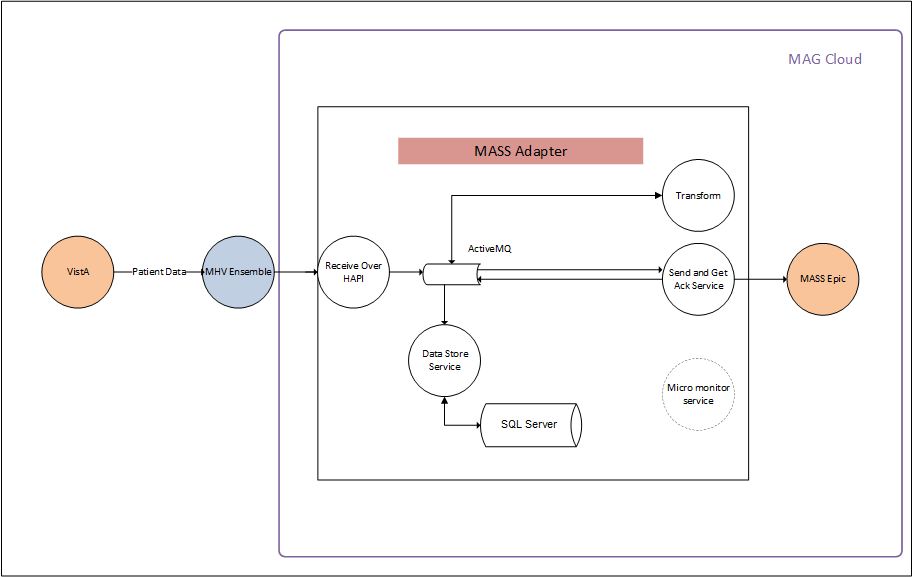
Figure : MASS Adapter Orchestration

****

These microservices are combined to create several workflows and deployed as workflows, to meet MASS Data Requirements and Business Requirements. Workflows include:

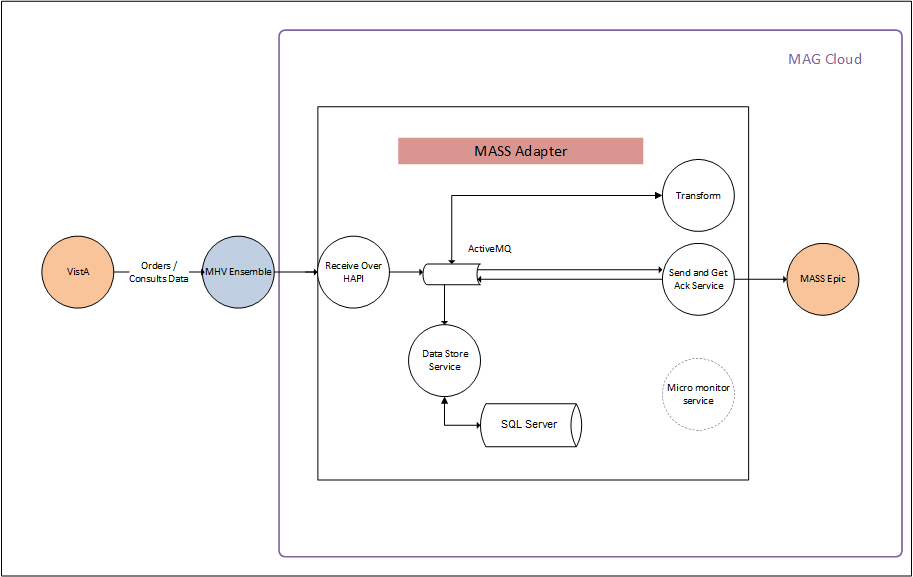
* **Patient Data Workflow:** This workflow, as illustrated in *Figure 17: Patient Data Workflow*, will receive patient related updates information from VistA in HL7 format and send to Epic Cadence. Patient data comprises of patient Identity, Demographics, Patient flags, contacts and Eligibility information of a Veteran/Patient, through Ensemble

Figure : Patient Data Workflow



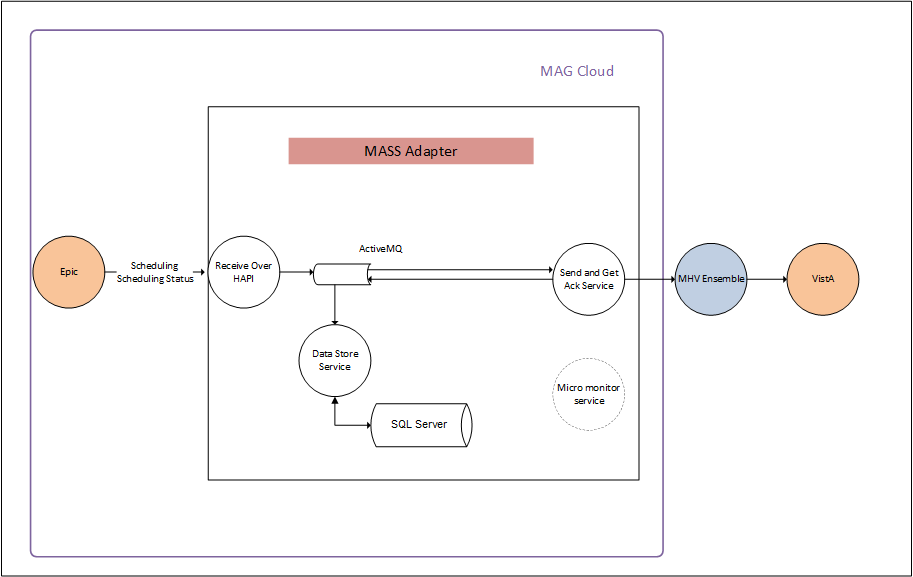
* **Orders Data Workflow:** This workflow, as illustrated in *Figure 18: Orders Data Workflow*, will receive Orders/Consults information from VistA in HL7 format and send to Epic Cadence, through Ensemble

Figure : Orders Data Workflow



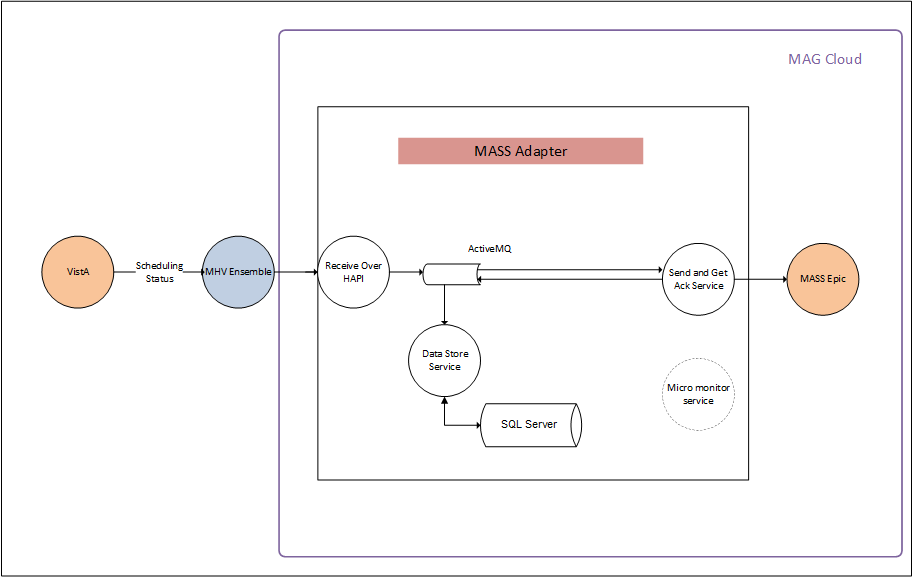
* **Scheduling Epic Data Workflow:** This workflow, as illustrated in *Figure 19: Scheduling Data Workflow*, will receive Scheduling information from Epic in HL7 format and send to Columbus VistA, through Ensemble

Figure : Scheduling Data Workflow



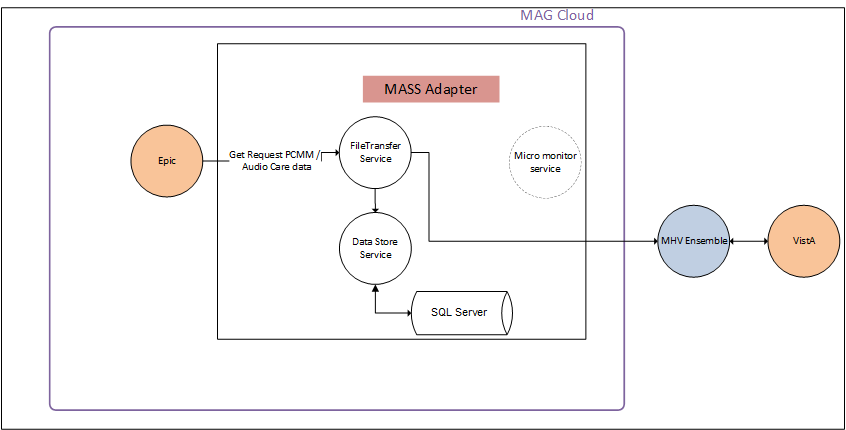
* **Scheduling VistA Data Workflow:** This workflow, as illustrated in *Figure 20: Scheduling VistA Data Workflow,* will receive Scheduling information from Epic in HL7 format and send to Columbus VistA, through Ensemble

Figure : Scheduling VistA Data Workflow



* **PCMM/AudioCare Data Workflow:** This workflow, as illustrated in *Figure 21: PCMM/AudioCare Data Workflow*, will receive information from PCMM/AudioCare as a batch file and send to Epic Cadence. This workflow will also send information from Epic as a batch file to AudioCare as well, through Ensemble

Figure : PCMM/AudioCare Data Workflow



MASS interfaces with MHV Ensemble to exchange data with VA systems:

* **MHV Ensemble:** MASS will use MHV Ensemble to communicate data to and receive data from VistA, PCMM and AudioCare

## Network and Wide Area Network (WAN) Architecture and Connectivity

The MAG cloud is connected to VA’s network backbone via ExpressRoute. This connectivity passes through the TIC before getting to the MAG cloud. Network connectivity between subscriptions within the same region is enabled with Azure Virtual Network Peering.

ExpressRoute is an Azure service that lets users create private connections between Microsoft datacenters and infrastructure that’s on premises or in a co-location facility. ExpressRoute connections do not go over the public Internet; and offer higher security, reliability, and speeds with lower latencies than typical connections over the Internet.

Virtual network peering enables users to connect two virtual networks in the same region through the Azure backbone network. Once peered, the two virtual networks appear as one, for connectivity purposes. The two virtual networks are still managed as separate resources, but VMs in the peered virtual networks can communicate with each other directly by using private IP addresses.

The traffic between VMs in the peered virtual networks is routed through the Azure infrastructure, much like traffic is routed between VMs in the same virtual network. Benefits of using virtual network peering include:

* Low-latency, high-bandwidth connection between resources in different virtual networks
* Ability to use resources, such as network appliances and Virtual Private Network (VPN) gateways, as transit points in a peered virtual network
* Ability to peer two virtual networks created through the ARM deployment model, or to peer one virtual network created through Resource Manager, to a virtual network created through the classic deployment model. Read the [Understand Azure Deployment Models](http://searchcloudapplications.techtarget.com/tip/Understanding-Azure-deployment-methods) article to learn more about the differences between the two Azure deployment models

*Figure 22: VA-Azure Connectivity* shows connectivity from the VA network to the Azure region via ExpressRoute.

Figure : VA-Azure Connectivity



### Storage Connectivity

Data within Azure Storage can be accessed and managed in a variety of ways, using numerous tools and processes. The following mechanisms support managing Azure Storage:

* **Graphical Interface (GUI) Tools:** Access Azure Storage using an interface that mimics File Explorer. It includes functionality such as drag-and-drop-based tools that allows users to view and access data as if on a local or network drive on a server. These tools are easy to use and understand, and they are the best option for those who are new to Azure Storage.
* **CLI Tools:** Access Azure Storage from a command line, such as Azure PowerShell. Allows users to include data operations (such as move, copy, and delete) within automation scripts. These interfaces have many options and switches to allow for a variety of options in working with the data. They are best used by advanced users and those who are already familiar with Windows PowerShell and require automation as part of their Azure-based solution
* **REST Interfaces:** Offers programmatic access to the blob, queue, table, and file services in Azure Storage, or in the development environment, via the storage emulator. All storage services are accessible via REST APIs. Storage services can be accessed from within a service running in Azure Storage, or directly over the Internet from any application that can send an HTTP/HTTPS request and receive an HTTP/HTTPS response. These interfaces are best suited for developers or solutions that require detailed information or control over Azure Storage services
* **Client Libraries:** The Azure Storage Client Library reference for .NET contains the current version of the Storage Client Library for .NET. Users can install the Storage Client Library for .NET from NuGet or from the Azure SDK for .NET. The source code for the Storage Client Library for .NET is publicly available in GitHub. The Azure Storage Native Client Library is a C++ library for working with the Azure Storage services
* **Cross-Platform Options:** The Azure Cross-Platform Command-Line Interface (xplat-cli) provides a set of open source, cross-platform commands for working with the Azure platform. The xplat-cli provides much of the same functionality found in the Azure portal, such as the ability to manage websites, VMs, storage, and SQL databases. The xplat-cli is written in JavaScript, and it requires Node.js

### Load Balancing

Azure Load Balancer delivers high availability and network performance to your applications. It is a Layer 4 (TCP, UDP) load balancer that distributes incoming traffic among healthy instances of services defined in a load-balanced set. Azure Load Balancer can be configured to load balance traffic between VMs in a virtual network, between virtual machines in cloud services, or between on-premises computers and VMs in a cross-premises virtual network. This configuration is known as [internal load balancing](https://docs.microsoft.com/en-us/azure/load-balancer/load-balancer-internal-overview).

Microsoft Azure Application Gateway is a dedicated virtual appliance providing application delivery controller (ADC) as a service. It offers various Layer 7 load balancing capabilities for applications. It allows web farm productivity optimization by offloading CPU intensive SSL termination to the application gateway. It also provides other Layer 7 routing capabilities including round robin distribution of incoming traffic, cookie-based session affinity, URL path-based routing, and the ability to host multiple websites behind a single Application Gateway.

Microsoft Azure Application Gateway includes the following features:

* [**Web Application Firewall (WAF)**](https://docs.microsoft.com/en-us/azure/application-gateway/application-gateway-webapplicationfirewall-overview)**:** Protects web applications from common web-based attacks like SQL injection, cross-site scripting attacks, and session hijacks. It can be configured as internet facing gateway, internal only gateway, or a combination of both
* **HTTP Load Balancing:** Provides round robin load balancing. It is done at Layer 7 and is used for HTTP(S) traffic only
* **Cookie-based Session Affinity:** Useful when you want to keep a user session on the same backend. By using gateway-managed cookies, the Application Gateway can direct subsequent traffic from a user session to the same backend for processing. This feature is important in cases where session state is saved locally on the backend server for a user session
* [**Secure Sockets Layer (SSL) Offload**](https://docs.microsoft.com/en-us/azure/application-gateway/application-gateway-ssl-arm)**:** Takes the costly task of decrypting HTTPS traffic off web servers. By terminating the SSL connection at the Application Gateway and forwarding the request to the server unencrypted, the web server is unburdened by decryption. Application Gateway re-encrypts the response before sending it back to the client. This feature is useful in scenarios where the backend is in the same secured virtual network as the Azure Application Gateway
* [**End to End SSL**](https://docs.microsoft.com/en-us/azure/application-gateway/application-gateway-backend-ssl)**:** Supports end to end encryption of traffic by terminating the SSL connection at the application gateway. The gateway then applies the routing rules to the traffic, re-encrypts the packet, and forwards the packet to the appropriate backend based on the routing rules defined. Any response from the web server goes through the same process back to the end user
* [**URL-based Content Routing**](https://docs.microsoft.com/en-us/azure/application-gateway/application-gateway-url-route-overview)**:** Provides the capability to use different backend servers for different traffic. Traffic for a folder on the web server or for a Content Delivery Network could be routed to a different back-end. This capability reduces unneeded load on backends that don't serve specific content
* [**Multi-site Routing**](https://docs.microsoft.com/en-us/azure/application-gateway/application-gateway-multi-site-overview)**:** Allows users to consolidate up to 20 websites on a single application gateway
* [**Websocket Support**](https://docs.microsoft.com/en-us/azure/application-gateway/application-gateway-websocket)**:** Provides native support for Websocket
* [**Health Monitoring**](https://docs.microsoft.com/en-us/azure/application-gateway/application-gateway-probe-overview)**:** Provides default health monitoring of backend resources and custom probes to monitor for more specific scenarios
* [**SSL Policy and Ciphers**](https://docs.microsoft.com/en-us/azure/application-gateway/application-gateway-ssl-policy-overview)**:** This feature provides the ability to limit the SSL protocol versions and the ciphers suites that are supported and the order in which they are processed
* [**Request Redirect**](https://docs.microsoft.com/en-us/azure/application-gateway/application-gateway-redirect-overview)**:** Provides the capability to redirect HTTP requests to an HTTPS listener
* [**Multi-tenant Backend Support**](https://docs.microsoft.com/en-us/azure/application-gateway/application-gateway-web-app-overview)**:** Supports configuring multi-tenant backend services like Azure Web Apps and API Gateway as backend pool members
* [**Advanced Diagnostics**](https://docs.microsoft.com/en-us/azure/application-gateway/application-gateway-diagnostics)**:** Provides full diagnostics and access logs. Firewall logs are available for application gateway resources that have WAF enabled

All virtual machines for Citrix Storefront, and Epic applications and services, except Business Continuity Access Web, Kuiper, and System Pulse need to be load balanced using a physical or virtual appliance capable of Layer 7 network load balancing and capable of supporting cookie-based session affinity, with vendor software support. Azure Application Gateways will be used where SSL offloading is a requirement. Azure Internal Load Balancers are used when SSL offloading is not required but general load balancing is required.

### Management

Network and WAN connectivity management will be handled via ESCCB requests, which follow VA’s policies and procedures. The network connectivity between the VA and MAG is controlled by Network & Security Operations Center (NSOC); connectivity and port requests are managed though the ESCCB.

TeamSMS will implement the following to ensure MASS adheres to the VA enterprise architecture:

* MASS uses Epic Cadence as its scheduling application. TeamSMS and VA’s MASS team currently have efforts underway to add Epic to the VA TRM
* Standards compliant HL7 V2.x messaging and Representational State Transfer (RESTful) interfaces are exposed via a MASS Adapter
* Plan to utilize IAM SSOe for user identification to access MyChart via SAML authentication integration
* VistA access through a VA-approved VistA Adapter by leveraging MHV Ensemble integration
* Utilize existing VA interfaces and systems

# Data Design

Epic’s database, called Chronicles, runs on top of Intersystems Caché database. This allows for scalable, patient-centric data storage that is quickly accessible to users via Hyperspace.

## Database Management System Files

Chronicles is not a relational database system. Epic is a COTS product; therefore, additional details about the database are

and will not be included in this document.

## Data View

As Chronicles is not a relational database, data is extracted, transformed, and loaded into a sister database called Clarity. This data model conforms to Epic-maintained strategies. Additional information about the data model can be provided upon request.

# Detail Design

## Hardware Detailed Design

Since MASS will be implemented in the MAG cloud, there will be no MASS-managed hardware; however, storage, VMs, networking, load balancing, etc. will still be managed via the Azure Portal/Powershell/CLI.

For hardware architecture information, refer to *Section 4.1: Hardware Architecture*.

**Azure Storage in Relation to IaaS Deployment**

Every solution deployed in Microsoft Azure leverages an aspect of Azure Storage, making storage a common component and critical to planning any Azure-based solution design. Each Azure VM created needs to access to Storage, and Networking for full functionality.

**Storage for Azure IaaS (Compute)**

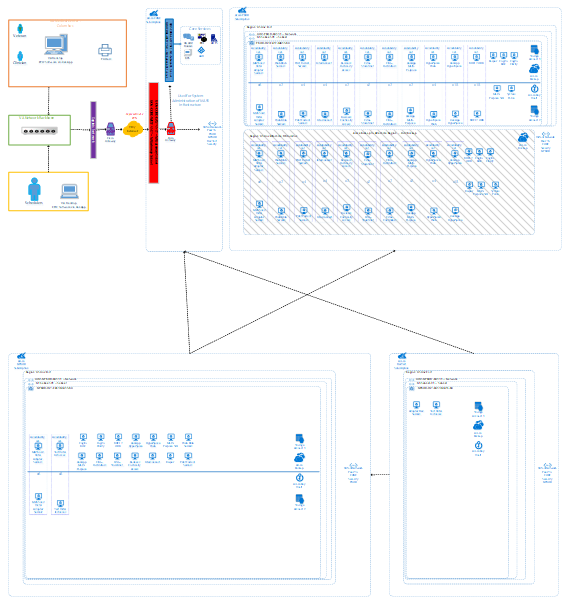
A Microsoft Azure VM is created from an image or a disk. All VMs use one operating system disk, a temporary local disk, and enable the use of multiple data disks depending on the selected size of the VM. All images and disks, except the temporary local disk, are created from virtual hard disk files that are stored as page blobs in a storage account in Microsoft Azure.

**Networking for Azure IaaS (Compute)**

Azure Virtual Networks provide a key building block for establishing virtual private networks. Virtual networks can be used to allow isolated network communication within the Azure environment or establish cross-premises network communication between an organization’s network infrastructure and Azure. By default, when VMs are created and connected to Azure Virtual Network, they can route to any subnet within the virtual network, and outbound access to the Internet is provided by Azure’s Internet connection.

*Figure 23: MASS Detailed Design,* on the next page, depicts all the servers, network connectivity, and represents a detailed design diagram for MASS.

Figure : MASS Detailed Design



## Software Detailed Design

For software architecture information, refer to *Section 4.2: Software Architecture*.

## Conceptual Design

For conceptual design information, refer to *Section 0:*

*Conceptual* Design and its corresponding sub-sections.

### User Interfaces

For conceptual design, please refer to section 3.0 Hardware Interfaces

There are no hardware interfaces as MASS will be deployed to the cloud. Please refer to *Section 4.0: System Architecture* and *Section 6.0: Detail Design* for cloud design information.

### User Characteristics

For conceptual design, please refer to *Section 3.0: Conceptual Design*.

### Software Interfaces

The MASS pilot deployment will support user interactions from both VistA and Epic. Depending upon the action taken, data must be transferred between the two systems to maintain consistency of information.

**VistA Operations**

Users within VistA will initiate the following interactions. These actions will automatically trigger a data exchange from VistA to Epic to maintain consistency between the systems. Data exchanges include:

* New Patient or Update Patient's Demographic
  + Eligibility and Enrollment, Demographic, Contact information
  + Patient creation, Patient identity
  + Patient care team information
* New Orders and Consults
* The Appointment Status Service kiosk will communicate check-in and check-out information

All data transmitted between Epic Cadence, MASS Adapter and MHV HealthConnect (Ensemble) will be over the VA TCP/IP transport layer using HTTPS. Data transfers between MHV HealthConnect (Ensemble) and VistA will be over TCP/IP transport layer using Minimal Lower Layer Protocol (MLLP). The HL7 message will be exchanged securely between MASS and MHV using HTTPS. The message-body (if any) of the HTTP post request will carry the HL7 message and the response message body will contain the Commit ACK/Commit Reject.

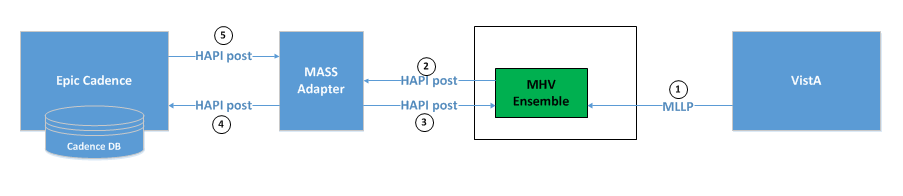
HL7v2 messages to and from MASS must be sent over the internet securely. To accomplish this Ensemble and MASS will use the HAPI standard utilizing SSL/TLS encryption and X.509 certificate authentication to traverse the MAG computing network to the Terremark cloud network. Ensemble and MASS will decrypt the communications streams and enqueue messages to pass on to other systems within the VA and MASS networks.

Further, VistA and Ensemble will communicate using MLLP over TCP/IP. The following data transfers will take place between VistA and Ensemble:

* HAPI incoming from MASS Adapter to Ensemble
* MLLP outgoing to VistA from Ensemble
* MLLP incoming from VistA to Ensemble
* HAPI outgoing from Ensemble to MASS Adapter
* Security for HAPI transactions
* Encryption via SSL/TLS via port 443
* Client authentication via X.509 certificates
* Server authentication via X.509 certificates
* URL Routing

*Figure 24: VistA to Epic Data Transfer* depicts data transfers from VistA to Epic.

Figure : VistA to Epic Data Transfer



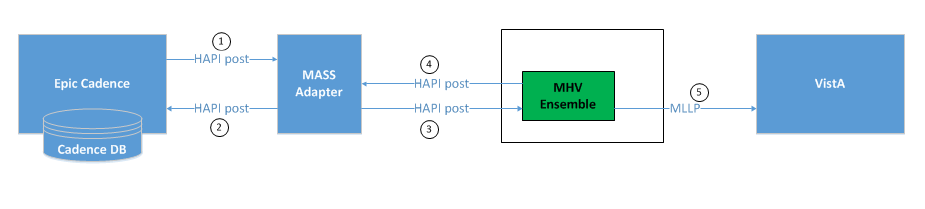
1. VistA application creates the HL7 messages for patient, consults, orders and appointment status updates, MHV HealthConnect (Ensemble) receives the original HL7 message from the VistA HL7 (HLO) Message Engine in MASS Namespace.
2. MHV HealthConnect (Ensemble) sends the original request HL7 message, as received from VistA HL7 (HLO) Message Engine, to MASS Adapter using HTTPS Post method
3. MASS Adapter sends HL7 ACK to MHV HealthConnect (Ensemble) Ensemble as response
4. MASS Adapter queues, processes if needed and sends the data in HL7 format to EPIC Cadence using HTTPS Post method
5. Epic Cadence sends HL7 ACK to MASS Adapter as response

**Epic Operations**

Users will interact directly with Epic to create new Appointment Scheduling and update Appointment Scheduling. These actions will automatically trigger a data exchange from Epic to VistA to maintain consistency between the systems.

*Figure 25: Epic to VistA Data Transfer* depicts data transfers from Epic to VistA.

Figure : Epic to VistA Data Transfer



1. Epic Cadence sends Scheduling data in HL7 format message to MASS Adapter using HTTPS post method
2. MASS Adapter send HL7 ACK to Cadence as response
3. MASS Adapter queues, processes if needed and sends the data in HL7 format to MHV HealthConnect (Ensemble) using HTTPS Post method
4. MHV HealthConnect (Ensemble) send HL7 ACK to MASS Adapter as response
5. MHV HealthConnect (Ensemble) sends the original HL7 message as received from MASS Adapter to VistA HL7 (HLO) Messaging Engine in MASS namespace. VistA receives the HL7 message, extracts the scheduling details, and performs the necessary actions to store the data.

**PCMM to MASS Adapter**

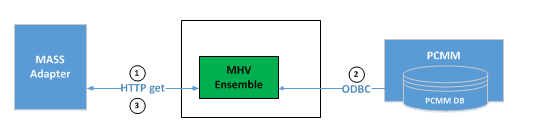
To maintain the patient care team in MASS, a periodic ETL must run from MASS to request the complete patient care team managed for a facility or an incremental subset. The PCMM system contains a database of the patient's care team. To access this system a secure connection must be created between MASS and PCMM. This will be accomplished via a web service exposed by Ensemble that further queries PCMM and returns the complete patient care team set for the facility.

HTTP GET messages from MASS must be sent over the internet securely. To accomplish this Ensemble and MASS will utilize SSL/TLS encryption and X.509 certificate authentication to traverse the MAG computing network to the Terremark cloud network. Ensemble and MASS will decrypt the communications streams and enqueue messages to pass on to other systems within the VA and MASS networks. Additionally, PCMM and Ensemble will communicate using an ODBC connection. The following is an outline of this communication:

* HTTP GET web service in Ensemble
* Build an SQL Query in Ensemble to query for the treatment team in the PCMM database
* Support basic parameters to modify the SQL Query (e.g. facility, time range, patient, range)
* Parse the SQL Query response in Ensemble and return in the HTTP GET Response as csv
* Return each line as: <patient local IEN>,<facility of IEN>,<patient name>,<patient date of birth>,<patient sex>,<provider role>,<provider ID>,<provider ID system>,<provider id type>,<provider name>
* Security
  + Encryption via SSL/TLS via port 443
  + Client authentication via X.509 certificates
* Server authentication via X.509 certificates
* ODBC connection to PCMM

*Figure 26: PCMM to MASS Adapter Data Transfer* shows how patient care team data is transferred from PCMM to the MASS Adapter.

Figure : PCMM to MASS Adapter Data Transfer



1. MASS Adapter will request for patient care information using HTTPS get method. Request should support basic parameters to modify the SQL Query (e.g. facility, time range, patient, range).
2. MHV HealthConnect (Ensemble) will query the data from PCMM database.
3. Ensemble sends the queried results in the response body of the get method. Return response will be in the following CSV format:
4. <patient local IEN>,<facility of IEN>,<patient name>,<patient date of birth>,<patient sex>,<provider role>,<provider ID>,<provider ID system>,<provider id type>,<provider name>

**AudioCare and MASS Adapter**

Interactive Voice Response (IVR) systems can be used to deliver appointment reminders to patients prior to their visits, asking them to confirm. Once the appointment is confirmed, the IVR system sends this information back to Epic Cadence so clinic staff can estimate no-show probabilities. The clinic staff will have a work list of appointments that have cancel requests by the IVR system to drive rescheduling efforts.

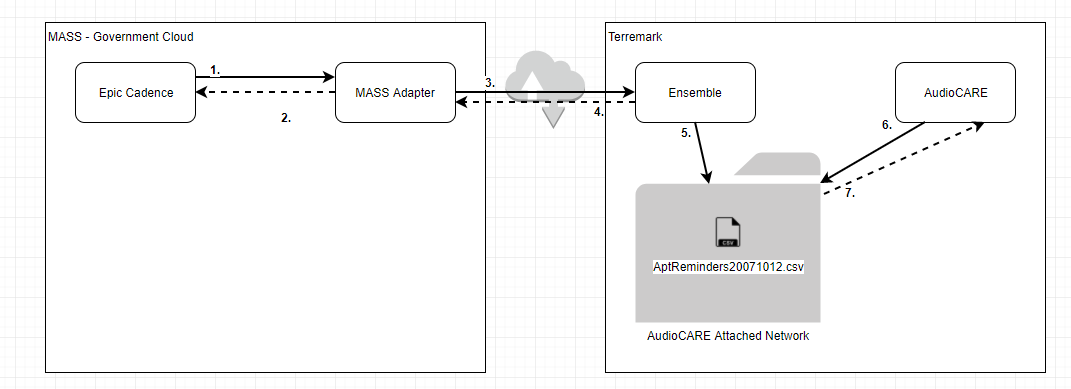
The files to and from MASS must be sent over the internet securely. To accomplish this, Ensemble and MASS will use HTTPS with SSL/TLS encryption and X.509 certificate authentication to traverse the MAG computing network to the Terremark cloud network. Ensemble and MASS will decrypt the communications streams and enqueue messages to pass on to other systems within the VA and MASS networks. Additionally, AudioCare and Ensemble will communicate over SSH.

At the VA the IVR system currently in place is AudioCare. The following outlines this process:

* MASS Adapter web client service to send appointment notification
* MASS Adapter web client service to request appointment notification responses
* MASS Adapter batch job to kick off web service requests at specific times (e.g., daily)
* Ensemble web server service to receive appointment notifications
* Ensemble web server service to send appointment notification responses

*Figure 27: MASS to AudioCare - Appointment Notifications* depicts the process flow from MASS to AudioCare for appointment notifications.

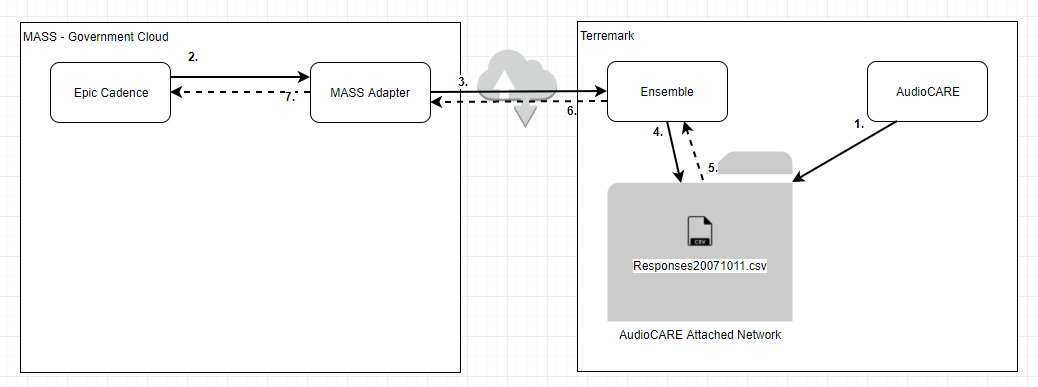
Figure : MASS to AudioCare - Appointment Notifications



1. Cadence will run a batch job (default: nightly) to export appointment notification information as an HTTPS POST using TLS/SSL for encryption and X.509 Mutual Auth for certificates for authentication to an MASS Adapter web service containing the notification content as the post body.
2. The MASS Adapter will respond with an HTTP 200-ok or other appropriate HTTP response code.
3. The MASS Adapter will execute a HTTPS POST using TLS/SSL for encryption and X.509 Mutual Auth for certificates for authentication to an Ensemble web service containing the notification content as the post body.
4. The Ensemble Engine will respond with an HTTP 200-ok or other appropriate HTTP response code.
5. The Ensemble Engine will place the notification content in the AudioCare secure network storage location via SSH.
6. AudioCare will run a batch job (default: nightly) to pick up the flat file from the network location. AudioCare will notify patients as appropriate.
7. AudioCare will process the file and delete or move it as appropriate.

*Figure 28: AudioCare to MASS - Appointment Notification Responses* depicts the process flow from MASS to AudioCare for appointment notification responses.

Figure : AudioCare to MASS - Appointment Notification Responses



1. AudioCare will run a batch job (default: nightly) to export a flat file (csv) of appointment notification responses to a secure network storage location via SSH and tag the file with a date.
2. Cadence will run a batch job (default: nightly) to execute a HTTPS GET using TLS/SSL for encryption and X.509 Mutual Auth for certificates for authentication to an MASS Adapter web service requesting today's notification responses.
3. The MASS Adapter will execute a HTTPS GET using TLS/SSL for encryption and X.509 Mutual Auth for certificates for authentication to an Ensemble web service requesting the notification responses.
4. The Ensemble Engine will retrieve the AudioCare file via SSH.
5. The Ensemble Engine will parse the AudioCare file.
6. The Ensemble Engine will respond to the MASS Adapter with an HTTP 200-ok and the content of the notification responses in the get response body or other appropriate HTTP response code.
7. The MASS Adapter will respond to Cadence with an HTTP 200-ok and the content of the notification responses in the get response body or other appropriate HTTP response code.
8. Cadence will process the notification content and update the status of the patient request in Epic.

### Memory Constraints

At this time there are no known memory constraints with the MASS application.

### Special Operations

For any special operations instructions please refer to the MASS Production Operations Manual.

### Product Features

The major product features for MASS can be found in the User Stories.

### Dependencies and Constraints

None at this time.

## Specific Requirements

### Database Repository

MASS will utilize the existing VistA repository to extract Patient information and Orders information and update the Epic system. MASS will update the VistA repository for any new scheduled appointments/appointment status to VistA as well.

### System Features

For specific system features please refer to the User Stories and/or Technical Stories for the respective MASS releases.

### Routines (Entry Points)

Please refer to *Appendix F – Routines (Entry Points)* for a detailed list*.*

## Network Detailed Design

For network architecture information, refer to *Section 4.3: Network and Wide Area Network (*WAN) Architecture and Connectivity.

## Security and Privacy

The following sub-sections describe the security and privacy design for MASS.

### Security

The MASS design will comply with specific security mechanisms across the infrastructure and application implementations in accordance with the NIST 800-53 Rev 4 Security Controls. The governing document for this shall be the MASS System Security Plan (SSP). A Security Risk Assessment was performed as part of the Assessment and Authorization package. MASS is categorized as a FISMA moderate system.

Encryption of data at rest is protected using the Azure Disk Encryption service in MAG.

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

Multifactor user authentication will be employed as PIV cards are implemented throughout the VA staff user population. The multifactor authentication will consist of something a user has (i.e., PIV card), and something they know (i.e., Personal Identification Number). These factors will enhance the integrity controls by making it exponentially more difficult for a user account to be compromised. Once a user is identified and authenticated, access to system resources and application functionality will be predicated upon role-based access controls.

The MASS production environment will be hosted within the MAG cloud. The system access necessary to use and manage MASS resources will employ the following technical resources and methods to ensure the integrity of the production environment:

* Administrator access will require a user account that is authorized and granted administrator privileges for MASS resources
* Administrators will be limited to accessing only those resources necessary to accomplish their assigned tasks
* Administrator access will be subject to an audit of all privileged actions to include logon, logoff, user account management, and administrative access to system servers and sensitive resources such as database access
* Every system user must have a User Record defined in the underlying Epic database. Without this record, the user cannot log in to the Epic system. Authentication is done directly in Epic or through an external system such as the lightweight directory access protocol. The IAM will provide the initial authentication and authorization used by the Epic system. Subsequent IAM interactions will be designed to ensure compliance with the VA’s IAM and security policies and procedures
* Once a user is authenticated, Security Classifications act like keys that determine what actions a user can perform in the system and which functionality and screens the user can access
* User roles control the look and feel of the application and are customized for groups of users, typically by functional role. The user role determines, generally, what parts of the application a user can see and how long he or she can stay in the system. If there are parts of the system that the user does not use, these features and screens can be hidden by the user role so that the user is not even aware of them

A user assigned administrative duties will be required to authenticate via a VA-issued PIV. To establish an administrative connection with MASS, an administrator must use the VA Citrix Access Gateway, VA Secure Mobility Client, or VA Rescue VPN. Each administrator must complete the Enterprise Operations production access request process to gain administrative access to MASS. The MAG cloud infrastructure is already connected to the VA using Express Route, and MASS will utilize this same connection as it is accredited with VA NSOC policies.

Additional responses to these required items about security controls will be found in the MASS SSP. The entire list of security controls which MASS has been assessed and will be contained in the Inventory of Security Controls document, which also includes inherited controls provided by the VA infrastructure.

### Privacy

MASS is designed to hold patient protected health information (PHI) and patient personally identifiable information (PII). As a result, MASS will comply with all security and privacy controls as detailed appropriate for a FISMA moderate system as detailed in the Privacy Act of 1974, 5 U.S.C. § 552a, HIPAA, NIST Special Publication 800-66 Revision 1, as well as the Federal Register 45 CFR Parts 160, 162, and 164 Health Insurance Reform: Security Standards; Final Rule.

MASS takes a defense in depth approach to protecting PII/PHI including the following mechanisms:

* MASS complies with privacy controls per NIST 800-53 & 800-66 as documented in SSP
* MAG VA subscription is only accessible from VA network
* API protected by a policy enforcement/policy decision point
* Data encryption at rest for any partition where PII/PHI will be contained
* Data encryption in transit using TLS on any network traffic beyond the local enclave
* Software security program will be built into the Software Development Lifecycle to ensure vulnerabilities are found and remediated during development

MASS will work with the VA Privacy Office to establish annual Privacy Threshold Analysis requirements and submit them to the VA through the VA’s Governance, Risk Management, and Compliance tool, Risk Vision.

## Service Oriented Detailed Design

All data transmitted between Epic Cadence, MASS Adapter and MHV HealthConnect (Ensemble) will be over RESTful web services using HTTPS. Data transfers between MHV HealthConnect (Ensemble) and VistA will be over TCP/IP transport layer using MLLP.

The data being exchanged from Epic to VistA and VistA to Epic will be in the format of HL7 messages. *Table 12: VistA and Epic Data Exchanges* lists the events that will initiate each Data Exchange.

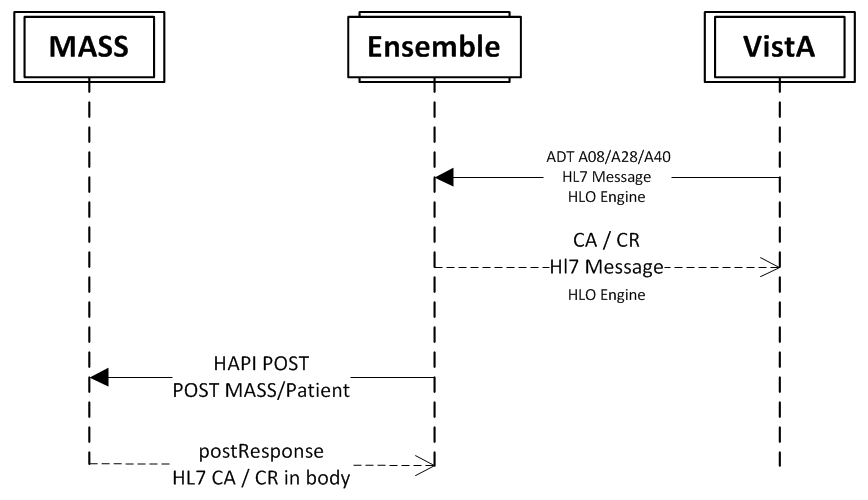
Table : VistA and Epic Data Exchanges

| ID | Description | Source System | Destination System |
| --- | --- | --- | --- |
| 1 | Patient Sequence - Eligibility and Enrollment, Demographic, Contact information, Patient creation, Merge, and Patient identity | VistA | Epic |
| 2 | Orders Sequence – New, Cancel, Update | VistA | Epic |
| 3 | Scheduling Sequence – Check-In and Check-Out | VistA | Epic |
| 4 | Scheduling Sequence – Appointment events Make, Check-In, Check-Out, No Show, Cancel | Epic | VistA |
| 5 | Application Errors | VistA | Epic |
| 6 | Patient Care Team – Update Treatment Team | PCMM | Epic |
| 7 | AudioCare – Patient Notifications | Epic | AudioCare |
| 8 | AudioCare – Responses | AudioCare | Epic |

### Patient Sequence

*Figure 29: Patience Sequence Diagram* illustrates the messages exchanged in the Patient Sequence. The Patient post from Ensemble to MASS will use the HAPI.

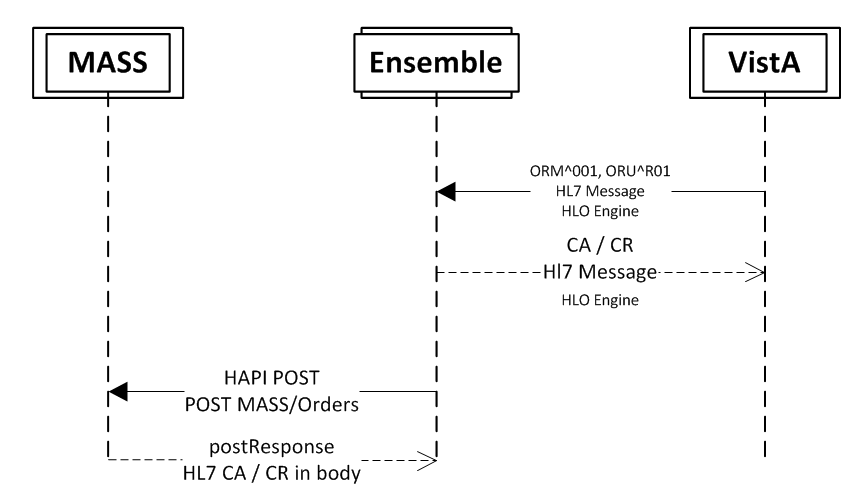
Figure : Patience Sequence Diagram



### Orders Sequence

*Figure 30: Orders Sequence Diagram* illustrates the messages exchanged in the Orders Sequence. The Orders post from Ensemble to MASS will use the HAPI.

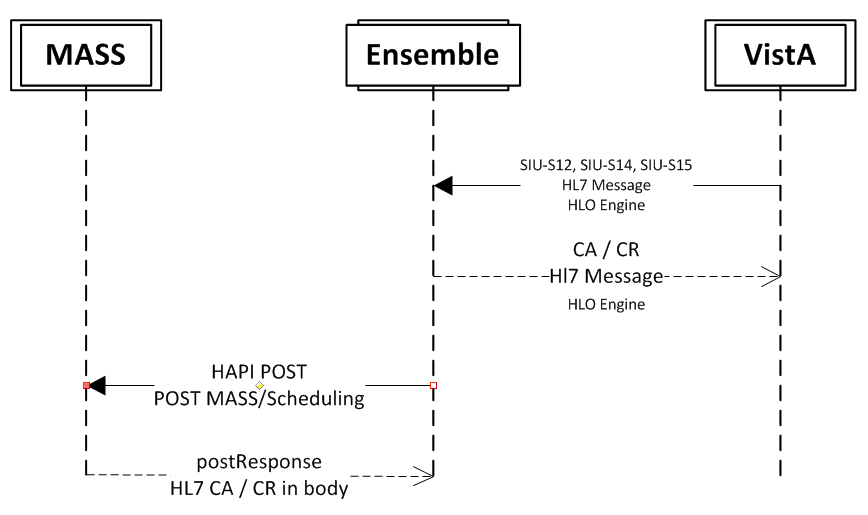
Figure : Orders Sequence Diagram



### Scheduling Sequence

*Figure 31: Scheduling Sequence Diagram* illustrates the messages exchanged in the Scheduling Sequence. The Scheduling post from Ensemble to MASS will use the HAPI.

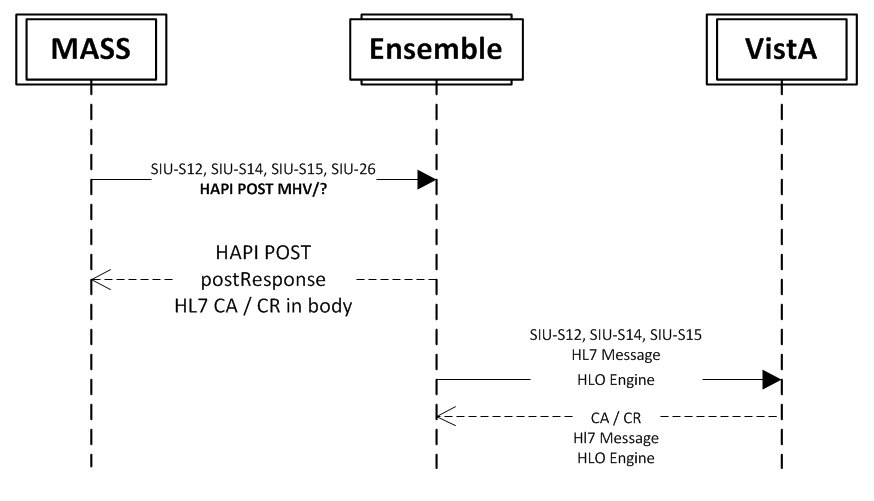
Figure : Scheduling Sequence Diagram



### Scheduling to VistA Sequence

*Figure 32: Scheduling to VistA Sequence Diagram* illustrates the messages exchanged in the Scheduling to VistA Sequence. The Scheduling post and response between Ensemble and MASS will use the HAPI.

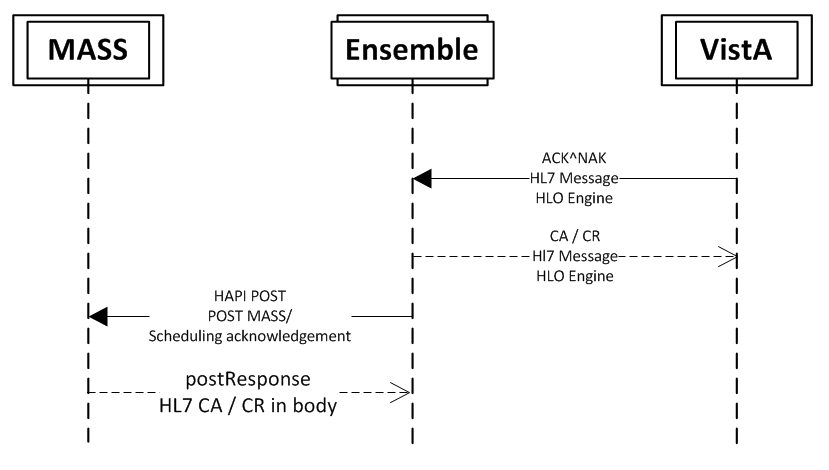
Figure : Scheduling to VistA Sequence Diagram



### Application Error to MASS Sequence

*Figure 33: Application Ack to MASS Sequence Diagram* illustrates the messages exchanged in the Application Error to MASS Sequence.

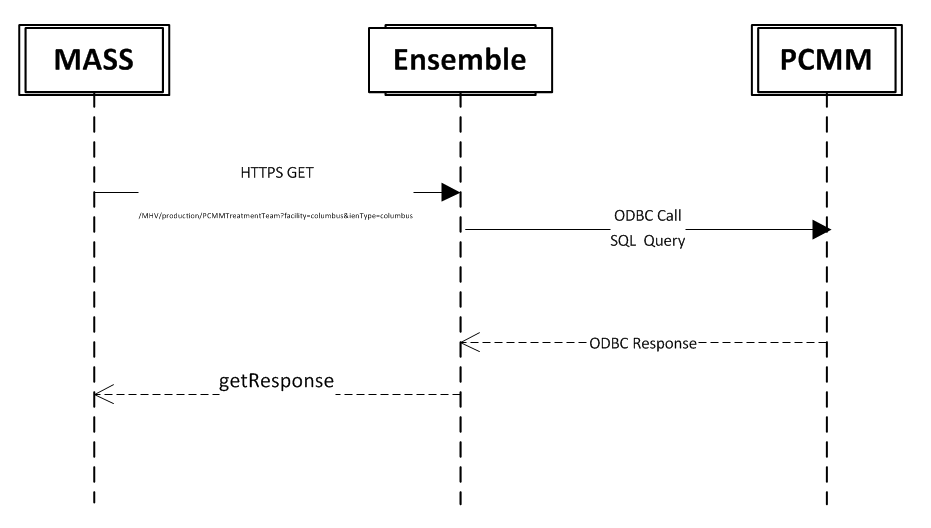
Figure : Application Ack to MASS Sequence Diagram



### PCMM Sequence

*Figure 34: PCMM Sequence Diagram* illustrates the messages exchanged in the PCMM Sequence.

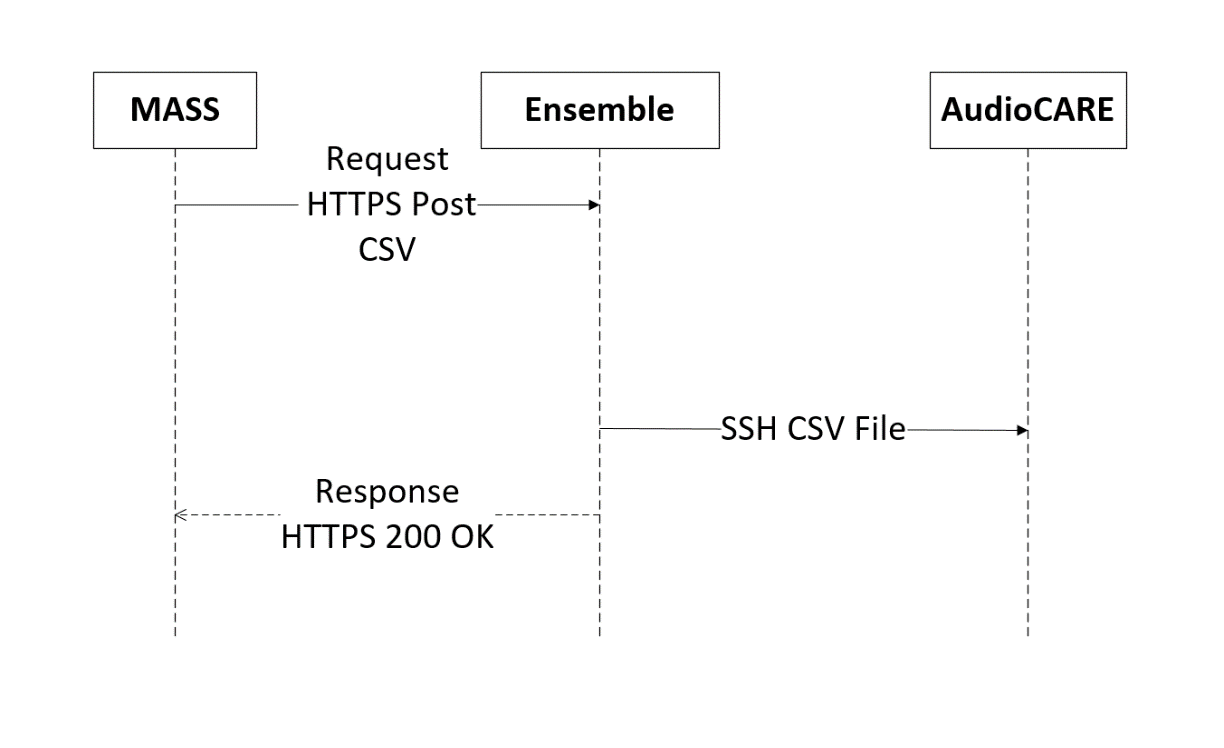
Figure : PCMM Sequence Diagram



### AudioCare Sequence

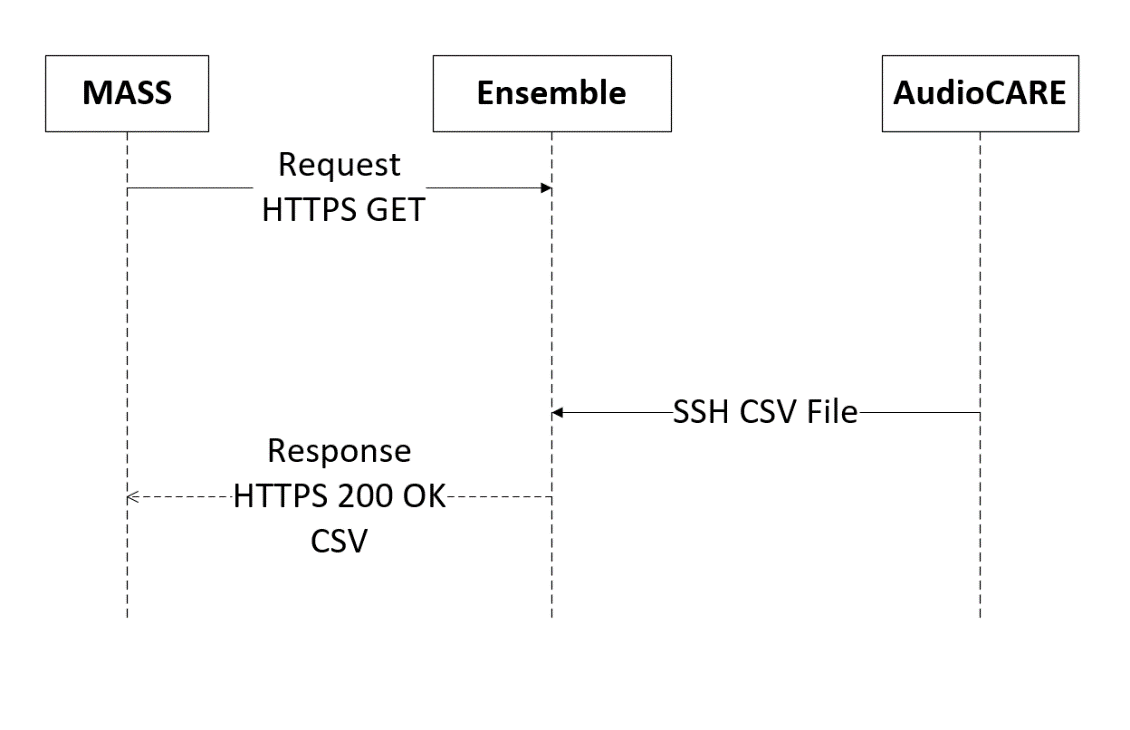
*Figure 35: AudioCare Notification Sequence Diagram* illustrates the communications path for AudioCare notifications.

Figure : AudioCare Notification Sequence Diagram



*Figure 36: AudioCare Response Sequence Diagram* illustrates the communications path for AudioCare responses.

Figure : AudioCare Response Sequence Diagram



# External System Interface Design

## MyChart

For MyChart design information, refer to *Section 4.1.4: MyChart Environment*.

## Ensemble

This section will be updated based on Interface agreements, Interface Control documents with IAM, SSOe, and MHV.

## Interface Architecture and Detailed Design

For Interface Architecture and design information about MASS Adapters, refer to *Section 4.2.1: MASS Adapter.*

# Human-Machine Interface

Epic’s end user GUI program, Hyperspace, was developed by user interface experts. These experts followed industry standards and best practices when designing the GUI for Hyperspace. Users can navigate Hyperspace with both mouse and keyboard.

## Inputs

Most Hyperspace workflows can be accomplished solely via keyboard and keyboard shortcuts. Additionally, Hyperspace supports mouse input as well. Access to various workflows is controlled via security so that users do not have access to tools they should not.

## Outputs

Various reports can be generated by Hyperspace. There are various Epic-released templates for reports that serve as a starting point. These can be tailored and run on the fly by users with appropriate security access to dive into the various data elements in Epic. Most of the configuration of these reports has not occurred; however, this will be an ongoing process throughout the life of the MASS project.

# Appendix A - Signature Page

I have reviewed and accept this document.

|  |  |
| --- | --- |
| **Signed:**  Carrie McCaslin | 12/20/17 |
| Carrie McCaslin,MASS Program Manager | Date |

|  |  |
| --- | --- |
| **Signed:** |  |
| Ruth Beltran-West,Contracting Officer Representative | Date |

# Appendix B – Acronyms

*Table 13: Acronyms* provides a complete list of acronyms used in this document.

Table : Acronyms

| Acronym | Description |
| --- | --- |
| ADC | Application Delivery Controller |
| API | Application Programming Interface |
| ARM | Azure Resource Manager |
| ASM | Azure Service Management |
| CLI | Command Line Interface |
| COTS | Commercial Off the Shelf |
| DR | Disaster Recovery |
| ESCCB | Enterprise System Change Control Board |
| FIPS | Federal Information Processing Standard |
| FISMA | Federal Information Security Management Act |
| GRS | Geo Redundant Storage |
| GUI | Graphical User Interface |
| HIPAA | Health Insurance Portability and Accountability Act |
| HL7 | Health Level 7 |
| IaaS | Infrastructure as a Service |
| IAM | Identity and Access Management |
| IVR | Interactive Voice Response |
| JSON | JavaScript Object Notation |
| LRS | Locally Redundant Storage |
| LOB | Line of Business |
| MAG | Microsoft Azure for Government |
| MASS | Medical Appointment Scheduling |
| MHV | MyHealtheVet |
| NIST | National Institute of Standards and Technology |
| NSOC | Network & Security Operations Center |
| ODB | Operational Database |
| OI&T | Office of Information and Technology |
| PCMM | Primary Care Management Module |
| PHI | Protected Health Information |
| PII | Personally Identifiable Information |
| PIN | Personal Identification Number |
| PIV | Personal Identity Verification |
| PMAS | Project Management Accountability System |
| RA-GRS | Read Access Geo Redundant Storage |
| RBAC | Role Based Access Control |
| REST | Representational State Transfer |
| RPO | Recovery Point Objective |
| SAML | Security Assertion Markup Language |
| SDD | Software Design Document |
| SLA | Service Level Agreement |
| SP | Special Publication |
| SRS | Zone Redundant Storage |
| SSE | Storage Service Encryption |
| SSL | Secure Sockets Layer |
| SSO | Single Sign-On |
| SSP | System Security Plan |
| TBD | To Be Determined |
| TIC | Trusted Internet Connection |
| TLS | Transport Layer Security |
| TO | Task Order |
| TRM | Technical Reference Model |
| VA | Department of Veterans Affairs |
| VAMC | VA Medical Centers |
| DHN | Se erans Health Administration |
| VistA | Veterans Integrated System Technology Architecture |
| VM | Virtual Machines |
| VPN | Virtual Private Network |
| WAF | Web Application Firewall |
| WAN | Wide Area Network |
| XML | Extensible Markup Language |

# Appendix C – Reference Materials

In addition to constraining policies, directives, procedures, requirements and design specifications within this document, development of MASS shall follow all VA regulations, mandates, security requirements, and standard operating procedures for software development and all other applicable Federal regulations. Applicable documents include:

* VA procedures such as PMAS and ProPath
* Federal mandates such as the Health Insurance Portability and Accountability Act (HIPAA)
* VA Directive 6508 - Privacy Impact Assessments
* VA Directive 6500 – Information Security Program
* VACA (Veterans Access to Care Act)
* VA Directive 0735, Homeland Security Presidential Directive 12 (HSPD-12) Program, February 17, 2011
* VA Handbook 0735, Homeland Security Presidential Directive 12 (HSPD-12) Program, March 20, 2014
* [OMB Memorandum M-06-18](https://www.whitehouse.gov/sites/default/files/omb/assets/omb/memoranda/fy2006/m06-18.pdf), Acquisition of Products and Services for Implementation of HSPD-12, June 30, 2006
* [OMB Memorandum 05-24](https://www.whitehouse.gov/sites/default/files/omb/memoranda/fy2005/m05-24.pdf), Implementation of Homeland Security Presidential Directive (HSPD) 12 – Policy for a Common Identification Standard for Federal Employees and Contractors, August 5, 2005
* [OMB memorandum M-11-11](https://www.whitehouse.gov/sites/default/files/omb/memoranda/2011/m11-11.pdf), “Continued Implementation of Homeland Security Presidential Directive (HSPD) 12 – Policy for a Common Identification Standard for Federal Employees and Contractors, February 3, 2011
* [OMB Memorandum](https://www.whitehouse.gov/sites/default/files/omb/assets/egov_docs/memotocioshspd-12guideance2008.pdf), Guidance for Homeland Security Presidential Directive (HSPD) 12 Implementation, May 23, 2008
* [Federal Identity, Credential, and Access Management (FICAM) Roadmap and Implementation Guidance](http://www.idmanagement.gov/sites/default/files/documents/FICAM_Roadmap_and_Implementation_Guidance_v2%200_20111202_0.pdf), December 2, 2011
* [NIST SP 800-116](http://csrc.nist.gov/publications/nistpubs/800-116/SP800-116.pdf), A Recommendation for the Use of Personal Identity Verification (PIV) Credentials in Physical Access Control Systems, November 20, 2008
* [OMB Memorandum M-07-16](https://www.whitehouse.gov/sites/default/files/omb/memoranda/fy2007/m07-16.pdf), Safeguarding Against and Responding to the Breach of Personally Identifiable Information, May 22, 2007
* [NIST SP 800-63-2](http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-63-2.pdf), Electronic Authentication Guideline, August 2013
* [Draft NIST Special Publication 800-157](http://csrc.nist.gov/publications/drafts/800-157/sp800_157_draft.pdf), Guidelines for Derived PIV Credentials, March 2014
* [NIST Special Publication 800-164](http://csrc.nist.gov/publications/drafts/800-164/sp800_164_draft.pdf), Guidelines on Hardware-Rooted Security in Mobile Devices (Draft), October 2012
* [Draft National Institute of Standards and Technology Interagency Report (NISTIR) 7981 Mobile, PIV, and Authentication](http://csrc.nist.gov/publications/drafts/nistir-7981/nistir7981_draft.pdf), March 2014
* VA Memorandum, VAIQ #7100147, Continued Implementation of Homeland Security Presidential Directive 12 (HSPD-12), April 29, 2011
* VA Memorandum, VAIQ # 7011145, VA Identity Management Policy, June 28, 2010 (reference Enterprise Architecture Section, PIV/IAM
* IAM Identity Management Business Requirements Guidance document, May 2013, (reference Enterprise Architecture Section, PIV/IAM
* [Trusted Internet Connections (TIC) Reference Architecture Document](https://www.fedramp.gov/files/2015/04/TIC_Ref_Arch_v2-0_2013.pdf), Version 2.0, Federal Interagency Technical Reference Architectures, Department of Homeland Security, October 1, 2013
* [OMB Memorandum M-08-05](https://www.whitehouse.gov/sites/default/files/omb/assets/omb/memoranda/fy2008/m08-05.pdf), “Implementation of Trusted Internet Connections (TIC), November 20, 2007
* [OMB Memorandum M-08-23](https://www.whitehouse.gov/sites/default/files/omb/memoranda/fy2008/m08-23.pdf), Securing the Federal Government’s Domain Name System Infrastructure, August 22, 2008
* VA Memorandum, VAIQ #7497987, Compliance – Electronic Product Environmental Assessment Tool (EPEAT) – IT Electronic Equipment, August 11, 2014 (reference Document Libraries, EPEAT/Green Purchasing Section
* [Sections 524 and 525 of the Energy Independence and Security Act of 2007](http://www.gpo.gov/fdsys/pkg/PLAW-110publ140/html/PLAW-110publ140.htm), (Public Law 110–140), December 19, 2007
* [Section 104 of the Energy Policy Act of 2005](http://energy.gov/sites/prod/files/2013/10/f3/epact_2005.pdf), (Public Law 109–58), August 8, 2005
* [Executive Order 13514](https://www.whitehouse.gov/assets/documents/2009fedleader_eo_rel.pdf), “Federal Leadership in Environmental, Energy, and Economic Performance,” October 5, 2009
* [Executive Order 13423](http://www.gpo.gov/fdsys/pkg/FR-2007-01-26/pdf/07-374.pdf), “Strengthening Federal Environmental, Energy, and Transportation Management,” January 24, 2007
* [Executive Order 13221](http://www.gpo.gov/fdsys/pkg/WCPD-2001-08-06/pdf/WCPD-2001-08-06-Pg1123.pdf), “Energy-Efficient Standby Power Devices,” August 2, 2001
* VA Directive 0058, “VA Green Purchasing Program”, July 19, 2013
* VA Handbook 0058, “VA Green Purchasing Program”, July 19, 2013
* Office of Information Security (OIS) VAIQ #7424808 Memorandum, “Remote Access”, January 15, 2014
* [Clinger-Cohen Act of 1996](http://www.gpo.gov/fdsys/pkg/USCODE-2011-title40/pdf/USCODE-2011-title40-subtitleIII.pdf), 40 U.S.C. §11101 and §11103
* VA Directive 6071, Project Management Accountability System (PMAS, February 20, 2013

[[1]](#footnote-2)

# Appendix D – Business Needs Matrix

*Table 14: Business Needs Matrix* includes a master list of business needs provided by VA that serves as a reference guide for this implementation of MASS.

Table : Business Needs Matrix

| ID | Business Capabilities | COTS Product out of the box1 | Describe the effort required/complexity | Government Clarifications |
| --- | --- | --- | --- | --- |
| BN1 | System Set up |  |  |  |
| BN1A1 | The MASS shall provide VHA with ability to manage resources throughout the scheduling process. |  |  |  |
| BN1A1.1 | Create groups of resources, assigning different resources to one group or more (e.g., MRI machine, MRI technician, Room ABC). | X |  |  |
| BN1A1.2 | Maintain groups of resources with the ability to move/reassign resources to an already established group of resources. Maintain original group configuration for historical reporting and auditing purposes. | X |  |  |
| BN1A1.3 | Schedule a group of resources (resource set such as Provider, Nurse, etc.) as one entity for a scheduled event. | X |  | The expectation is that these groups are created and maintained in the COTS product and do not require synchronization with VistA or other systems in the first two years. |
| BN1A1.4 | Configure available time for resources (resource set such as provider, support staff, equipment and facilities). | X |  | Facility and equipment are expected to be created and maintained in the COTS product and do not require synchronization with VistA or other systems in the first two years. |
| BN1A1.5 | Ability to block time slots on the calendar for services at the resource (provider, equipment) level. | X |  | Mapping of time slots to resources is expected to be created and maintained in the COTS product and do not require synchronization with VistA or other systems in the first two years. |
| BN1A1.6 | Ability to block time slots on the calendar for services at the service (primary care) level. | X |  |  |
| BN1A1.7 | Ability to unblock time slots on the calendar for services (primary care) and resource at the provider/ equipment level. | X |  |  |
| BN1A1.8 | Ability to unblock time slots on the calendar for services at the resource (provider, equipment) level. | X |  |  |
| BN1A1.9 | Ability to define types of requests for care. | X |  |  |
| BN1A1.10 | Ability to configure types of requests for care. | X |  |  |
| BN1A1.11 | Ability to standardize requests for care. | X |  | The expectation is that requests for care are managed in the COTS product and do not require synchronization with VistA or other systems in the first two years. |
| BN1A2 | The MASS shall provide VHA with ability to manage providers as resources |  |  |  |
| BN1A2.1 | Ability to maintain a master list of providers for Columbus only in the system (MASS). | X |  | The expectation is that providers are maintained in both the COTS and VistA. Linking and synchronization of providers is required in the first two years. |
| BN1A2.1.1 | Ability to maintain a master list of all providers for Columbus. | X |  |  |
| BN1A2.2 | Ability to activate a provider to make them available for scheduling. | X |  | Provider activation and deactivation is expected to be linked to VistA Provider attributes and requires synchronization in the first two years. |
| BN1A2.3 | Ability to deactivate a provider to make them unavailable for scheduling. | X |  | Provider activation and deactivation is expected to be linked to VistA Provider attributes and requires synchronization in the first two years. |
| BN1A2.4 | Ability to create groups of providers. | X |  | Groups of providers are expected to be created and maintained in the COTS product and do not require synchronization with VistA or other systems in the first two years. |
| BN1A2.5 | Ability to maintain groups of providers. | X |  | Groups of providers are expected to be created and maintained in the COTS product and do not require synchronization with VistA or other systems in the first two years. |
| BN1A2.6 | Activate groups of providers to make them available for scheduling. | X |  |  |
| BN1A2.7 | Deactivate a group of providers to make them unavailable for scheduling. | X |  |  |
| BN1A2.8 | Ability to assign individual providers to calendar time slots. | X |  |  |
| BN1A2.9 | Ability to assign groups of providers to calendar time slots. | X |  |  |
| BN1A2.10 | Ability to modify individual providers assigned to calendar time slots. | X |  |  |
| BN1A2.11 | Ability to modify groups of providers assigned to calendar time slots. | X |  |  |
| BN1A3 | The MASS Shall provide VHA with the ability to manage facilities within a specific VA health care system as resources |  |  |  |
| BN1A3.1 | Ability to create lists (reference tables) of VA/VHA facilities and their relationships (e.g., CBOC associated with a VAMC). | X |  | Facilities are expected to be created and maintained in the COTS product and do not require synchronization with VistA or other systems in the first two years. |
| BN1A3.2 | Ability to maintain a list of VA/VHA facilities and their relationships (e.g., CBOC associated with a VAMC). | X |  | Facilities are expected to be created and maintained in the COTS product and do not require synchronization with VistA or other systems in the first two years. |
| BN1A3.3 | Ability to maintain a list of facilities and rooms associated with each VA/VHA/CBOC as they apply to scheduling resources. | X |  | Facilities and rooms are expected to be created and maintained in the COTS product and do not require synchronization with VistA or other systems in the first two years. |
| BN1A3.4 | Ability to activate facilities and rooms designated for services required for scheduling. | X |  | Facilities and rooms are expected to be created and maintained in the COTS product and do not require synchronization with VistA or other systems in the first two years. |
| BN1A3.5 | Ability to deactivate facilities and rooms designated for services required for scheduling. | X |  | Facilities and rooms are expected to be created and maintained in the COTS product and do not require synchronization with VistA or other systems in the first two years. |
| BN1A3.6 | Ability to assign facilities to calendar time slots. | X |  |  |
| BN1A3.7 | Ability to modify assignment of facilities/rooms to calendar time slots. | X |  |  |
| BN1A4 | The MASS shall provide VHA with ability to manage equipment as resources |  |  | Equipment resources are expected to be created and maintained in the COTS product and do not require synchronization with VistA or other systems in the first two years. |
| BN1A4.1 | Ability to create a master list of equipment necessary for scheduling (such as MRI machines). | X |  |  |
| BN1A4.2 | Ability to maintain a master list of equipment necessary for scheduling (such as MRI machines). | X |  |  |
| BN1A4.3 | Ability to activate equipment required for scheduling. | X |  |  |
| BN1A4.4 | Ability to deactivate equipment required for scheduling. | X |  |  |
| BN1A4.5 | Ability to assign equipment to calendar time slots. | X |  |  |
| BN1A4.6 | Ability to modify assignment of equipment to calendar time slots. | X |  |  |
| BN1A5 | The MASS shall provide VHA with the ability to manage components of scheduling such as service lines, modes of delivery). |  |  |  |
| BN1A5.1 | Ability to continue supporting current VistA national reporting and DSS coding to support non-scheduling business processes as it currently does today by interfacing data to VistA Scheduling. This is applicable for national reporting only. Local reporting will be in MASS. | X |  |  |
| BN1A5.2 | Ability to create a list of modes of delivery. | X |  | Modes of delivery will only be maintained in the COTS. Epic: Yes, but mobile health and home health not in scope for pilot |
| BN1A5.3 | Ability to associate modes of care to services and sub services. | X |  |  |
| BN5D1.6 | Ability to search for the availability of providers who accept telephone appointments (mode of care delivery). | X |  | Moved from the telehealth section as this applies to other appointments as well. |
| BN5D1.7 | Ability to modify mode of care related to an appointment. | X |  |  |
| BN5D1.8 | Ability to report when a mode of care for an appointment has been modified. | X |  |  |
| BN5D1.12 | Ability to search by "mode of care" when scheduling an appointment. | X |  |  |
| BN1A5.4 | Ability to define a standard list of service/sub-service types (primary type/subtype). | X |  | Service and sub-Service types will only be maintained in the COTS. |
| BN1A5.5 | Ability to modify a standard list of service/sub-service types (primary type/subtype). | X |  |  |
| BN1A5.6 | Ability to create a hierarchy of primary services and related sub-services groups. | X |  |  |
| BN1A5.7 | Ability to modify a hierarchy of primary services and related sub-services groups. | X |  |  |
| BN1A5.8 | Ability to define a standard set of appointment purpose types and assign default appointment lengths for each service/sub-service. | X |  | There is a need to provide standardized updates of appointment purpose type codes through a patch process. |
| BN1A5.9 | Ability to modify a standard set of appointment purpose types with default appointment lengths for each service/sub-service | X |  |  |
| BN1A5.10 | Allow for the creation of multiple appointments in the same timeslot with the same resource ("overbooking"). | X |  |  |
| BN1A5.11 | Ability to establish and apply business rules for "overbooking". | X |  |  |
| BN1A5.12 | Ability to define an appointment purpose type/service type as "care for a single patient" appointment. | X |  |  |
| BN1A5.13 | Ability for appointments within MASS to map back to DSS stop codes in VistA Scheduling. (See definition of DSS Stop codes.) |  | Epic will add read-only data elements for primary and secondary stop codes to our COTS solution's existing data structures and provide the ability to populate these codes from an external data file. This update is of a type customarily available in the commercial marketplace and will be performed at private expense outside the scope of the contract. |  |
| BN1A5.14 | Ability to define certain appointment purpose types/service types as "care of multiple patients" appointments (group appointment) | X |  |  |
| BN1A6 | The MASS shall provide data synchronization rules/triggers for external systems |  |  |  |
| BN1A6.1 | For the Pilot, there will need to be some synchronization between VistA and Epic. VistA will not update Epic. Epic will update VistA. Below is a high-level description of known data exchanges/flows by system: AudioCARE - Outgoing Appointment Reminders Kiosk - Incoming Scheduling Status CPRS - Incoming Orders and Consults to Epic Scheduling - Outgoing to VistA Demographics, Identity and Eligibility - Incoming to Epic | X |  | See VA AudioCARE Audit document for more information on existing features. |
| BN1A6.2 | MASS will be used exclusively for all scheduling during the pilot (excluding Radiology, Telehealth, and HBPC). | X |  |  |
| BN1A6.3 | Enable data synchronization with VistA, when MASS is the authoritative source of record. | X |  |  |
| BN1A6.4 | Create the DSS relationship with service type and appointment purpose type, count / no-count association and other combinations of required information when an appointment is created or modified. |  | Epic will create read-only data elements for count/no-count in our COTS solution and include these elements in outgoing interface transactions that occur upon the check in and check out of Veteran appointments. This update is of a type customarily available in the commercial marketplace and will be performed at private expense outside the scope of the contract. | The expectation is that the resources and appointments in COTS are pushed to the VistA scheduling application and that the relationships of resources, appointments and appointment attributes are appropriately maintained to VistA scheduling clinics to ensure that dependencies of other VistA packages on scheduling is preserved. |
| BN1B1.1 | Ability to create local (Facility, Service Line, Clinic) care coordination agreements. |  | Users can be guided by the Provider Steering feature to refer Veterans to the appropriate services and providers based on rules. Care coordination agreements can be linked from dashboards within the COTS solution. Any needed modifications would be of a type customarily available in the commercial marketplace and will be performed at private expense outside the scope of the contract. | The expectation is that Care Coordination Agreements are created and maintained in the COTS and no synchronization is required in the first two years.  Was BN1B1.4 previously. |
| BN1B1.3 | Ability to modify care coordination agreements by facility. |  | Users can be guided by the Provider Steering feature to refer Veterans to the appropriate services and providers based on rules. Care coordination agreements can be linked from dashboards within the COTS solution. Any needed modifications would be of a type customarily available in the commercial marketplace and will be performed at private expense outside the scope of the contract. | The expectation is that Care Coordination Agreements are created and maintained in the COTS and no synchronization is required in the first two years. |
| BN1C1 | The MASS shall provide VHA with the ability to manage, store and access standard notification templates (forms/letters) used for various patient communications (such as phone, mass mailing, and secure messaging). |  |  |  |
| BN1C1.1 | Ability to create patient notification templates. | X |  |  |
| BN1C1.2 | Ability to publish patient notification templates. | X |  |  |
| BN1C1.3 | Ability to share patient notification templates. | X |  |  |
| BN1C1.4 | Ability to modify notification templates according to business rules to accommodate facility and service requirements (i.e., directions, parking instructions, number of day prior to appointment the notification is sent, etc.). | X |  |  |
| BN1C1.5 | Ability to activate notification templates, one at a time a group or based on rules. | X | The COTS solution meets this need by allowing users to send patient notifications based on a patient's recorded preferences. |  |
| BN1C1.6 | Ability to deactivate notification templates, one at a time, as a group or based on rules of templates. | X | The COTS solution meets this need by allowing users to send patient notifications based on a patient's recorded preferences. |  |
| BN1C1.7 | Ability to set up notifications to be distributed automatically according to specified criteria/business rules, e.g., a configurable schedule | X |  |  |
| BN1D1 | The system shall allow users with the appropriate security privileges to establish standard parameters, business rules, alerts, workflows, etc. to ensure compliance with VHA scheduling practices. |  |  | The expectation is that the system is integrated with identity and access management to ensure compliance with enterprise architecture and security policies. VetsEZ: Integration with IAM SSOi for the Pilot |
| BN1D1.8 | Ability to create business rules (such as clinical service level, resource, and appointment purpose type). | X |  |  |
| BN1D1.9 | Ability to modify business rules (such as clinical service level, resource, and appointment purpose type). | X |  |  |
| BN1D1.12 | Allow facilities to modify their local workflow per security and policy constraints. | X | TBD on April 20, 2017, The Pilot will ensure compliance with Enterprise Architecture and VA Security Policies but may not integrate with IAM SSOi for the Pilot. MASS will integrate with IAM SSOi for National Rollout. User level authentication on interface messaging is not in scope | VetsEZ: Integration with IAM SSOi for the Pilot |
| BN1E1 | The MASS shall provide VHA with the ability to establish user access roles and rules at multiple organizational levels. |  |  | VetsEZ: Integration with IAM SSOi for the Pilot |
| BN1E1.2 | Ability to configure user privileges for scheduling functions, for sets of data (service line, facility) and based on patient flags and employee flags; such as restricting access to specific patients' data based on established security roles. | X | Security and configuration will be standardized nationally, and will be maintained within MASS. | The expectation is that MASS will be able to read and use patient and employee flags from VistA registration. VetsEZ: Integration with VistA REE Package |
| BN1E1.3 | Support user configuration preferences for data display and entry screens according to VA security directives and policies, and identity management and access applications. | X | Security and configuration will be standardized nationally, and will be maintained within MASS. | The expectation is that the provisioning of users and privileges is integrated with enterprise identity and access management and with VistA kernel. Synchronization of users and privileges with VistA and enterprise services is required in the first two years. |
| BN1E1.4 | Provide access in accordance with user authorization parameters for patients who manage their own appointments and requests through the portal. This will be a manual process for the pilot, although for national it will be automatic. | X | Business process will keep MASS in sync with the national system, not an automated integration for the scope of the pilot | The expectation is that the provisioning of users and privileges is integrated with enterprise identity and access management and with VistA kernel. Synchronization of users and privileges with VistA and enterprise services is required in the first two years. |
| BN1E1.5 | Log changes at the system-level. | X |  |  |
| BN1E1.6 | Ability to report on system-level changes. | X |  |  |
| BN2 | Veteran Information Management |  |  |  |
| BN2A1 | The MASS shall provide VHA with the ability to view near real-time eligibility information from VistA. |  |  | Epic: VistA will maintain eligibility information, not from ESR |
| BN2A1.1 | Provide authorized users with visibility to data from the authoritative source for patient identity. | X |  | VetsEZ: ESR integration is not in scope for the pilot. Eligibility information will be from VistA. Epic: Patient data will be cached locally. |
| BN2A1.2 | Demographics viewed in MASS will include eligibility and enrollment data, including priority groupings as necessary to support scheduling functions. | X | The COTS solution meets this need. We will build a display of eligibility information to appear during scheduling workflows. | VetsEZ: ESR integration is not in scope for the pilot. Eligibility information will be from VistA. Epic: Patient data will be cached locally. |
| BN2B1 | The MASS shall provide VHA with ability to view and use Veteran information. |  |  |  |
| BN2B1.1 | Ability to view patient scheduling data (demographics, special needs and preferences, registration, PACT assignment, etc.) based on the physical location of the delivery of care. | X | The COTS solution can display patient scheduling and demographic information regardless of physical location. | VetsEZ: ESR integration is not in scope for the pilot. Eligibility information will be from VistA. Epic: Patient data will be cached locally. |
| BN2B1.2 | Ability to view all patient scheduling demographics and VA demographics in a single screen/window. | X |  |  |
| BN2B1.3 | Ability to view Veteran provider assignments in MASS by populating data extracted weekly from PCMM. Real time integration is not in scope. | X |  | The PACT assignment is managed by the primary care management module in VistA and does not require COTS integration with PCMM in the first two years.  The scheduler needs to view the assignment, if nothing else; The scheduler needs to see the teams; Inability to do this efficiently could impact performance standards such as ASA.  VetsEZ: PCMM is not in scope for the pilot. Epic: We are not integrating with PCMM as part of the pilot |
| BN2B1.6 | Ability to view and use current patient demographic information in near real-time. (System needs the demographic information when performing certain functions such as generating patient notifications.) | X |  | VetsEZ: ESR integration is not in scope for the pilot. Eligibility information will be from VistA. Epic: Patient data will be cached locally. |
| BN2B1.8 | Ability to access and display a Veteran's Primary Care Provider and Primary Mental Health Provider for each facility in which they are seen from the authoritative source. | X |  | VetsEZ: PCMM is not in scope for the pilot. Epic: We are not integrating with PCMM as part of the pilot. To fulfill this need, a weekly extract from PCMM could be loaded in MASS. |
| BN2B1.10 | Ability to identify facilities/divisions associated with a Veteran. (A Veteran has to be "registered" in a VistA instance before they can make an appointment at the facility. Since we were planning for a national implementation, we wanted to make sure that this relationship was established in the database.) | X | A facility specific medical record number (MRN) could be assigned, which could be visible to the scheduler. The veteran wouldn’t be filtered out of the lookup results if the scheduler didn’t first register the veteran at that VAMC. (I'm not sure what "first register the Veteran" means.) | When making appointments, the scheduler would need to see the Veteran's pending appointments regardless of which VAMC the appointments are scheduled at. This is especially important for VAMCs that are close together, i.e. Danville IL and Indianapolis since Veterans go to both and, at times, Veterans can be scheduled at both VAMCs on the same day or too close together for the Veteran to make both appointments. |
| BN2B1.11 | Ability to set up alerts triggered by user-defined rules, e.g., when the need to coordinate additional services is required. | X |  |  |
| BN2B1.12 | Ability to migrate/view future appointments in MASS. (Appointment history will not be migrated to MASS.) | X |  |  |
| BN2C1 | The MASS shall provide VHA with the ability to manage patient healthcare preferences (such as appointment date and time, preferred provider, and language assistance). |  |  |  |
| BN2C1.1 | Ability to capture/store patient healthcare preferences (such as appointment date and time, preferred provider, and language assistance) | X |  |  |
| BN2C1.2 | Ability to search data regarding preferences related to a patient. | X |  |  |
| BN2C1.3 | Ability to capture (date/time stamp) modifications to a Veteran’s preferences data. | X |  |  |
| BN2C1.4 | Ability to use patient healthcare preferences information when scheduling an appointment | X |  |  |
| BN2C2 | The MASS shall provide VHA with the ability to manage patient healthcare special needs (such as transportation, escort services, and handicap assistance) when scheduling an appointment. |  |  |  |
| BN2C2.1 | Ability to select one or more patient special needs from a predefined list. | X |  |  |
| BN2C2.2 | Ability to search data regarding special needs related to a patient. | X |  |  |
| BN2C2.3 | Ability to use patient healthcare special needs (such as transportation, escort services, and handicap assistance) information when scheduling an appointment | X |  |  |
| BN2C2.4 | Ability to capture (date/time stamp) modifications to a Veteran’s special needs data. | X |  |  |
| BN3 | Request Management |  |  |  |
| BN3A1 | The MASS shall provide VHA with the ability to enter requests for care from patients, providers, individuals authorized to act on behalf of the patient. |  |  | The expectation is that requests can be electronically submitted from other VA systems such as mobile applications, portals and call centers. |
| BN3A1.1 | Ability to identify persons other than the Veteran who are requesting care on behalf of the Veteran, e.g., patients, providers, and individuals authorized to act on behalf of the patient. | X |  |  |
| BN3A1.2 | Ability to maintain a single request list, which replaces VistA lists such as New Enrollee Appointment Request list (NEAR), Electronic Wait List (EWL), Consults and Service Orders (Vista interface - CPRS) and Recall/Reminder list. |  | Epic will modify one of our COTS solution's existing appointment request structures to a single request structure. This update is of a type customarily available in the commercial marketplace and will be performed at private expense outside the scope of the contract. |  |
| BN3A1.7 | Ability to auto populate patient data derived through the identity verification process on the request for care | X |  | VetsEZ: Integration with IAM SSOe and myChart Integration will populate patient data from IAM |
| BN3A1.8 | Ability to capture the creation date of a request. | X |  |  |
| BN3A1.9 | Ability to enter the Veteran’s preferred date for an appointment. | X | The COTS solution's single request structure supports the ability to store the Veteran's preferred date. |  |
| BN3A1.10 | Ability to capture and store request data to include date request created, mode of request (mobile, phone, in-person). | X | The COTS solution's single request structure supports the ability to record this information in a free-text field associated with the request. |  |
| BN3A1.11 | Ability to assign workflow based on types of requests (such as such as urgent, new, change existing or Veteran is not yet established in the system). | X |  |  |
| BN3A1.12 | Ability to use request tracking information for other scheduling functions (calculating preferred date, wait times, etc.) | X |  |  |
| BN3A1.13 | Ability to track and consolidate request status and appointment status related to the request (such as the request on waiting list, or progressed to scheduled appointment, and encounter completed). | X |  |  |
| BN3A1.14 | Ability to enter the preferred appointment date, identify the source of the preferred date, username if user-entered (such as patient-id/scheduler-id) or system-generated. Once entered, the date will be preserved and available for reporting and other processes | X |  |  |
| BN3A1.15 | The system shall automatically identify the first, second and third next available appointments as calculated by the preferred date. | X |  |  |
| BN3A1.16 | Ability to capture dates associated with VA care/ services. Dates/times should be system-protected (not changeable), reportable, and auditable. | X |  |  |
| BN3B1 | The MASS shall provide VHA with an all-inclusive master list designated for requests (including waiting for available appointments, recalls, etc.) in a consolidated standard format used for request processing and appointment management. |  | The COTS product vendor will modify one of our COTS solution's existing appointment request structures to a single request structure. This update is of a type customarily available in the commercial marketplace and will be performed at private expense outside the scope of the contract. |  |
| BN3B1.1 | Ability to sort and filter the consolidated request list by request attributes, such as type of request (Urgent, Non-Urgent), source of input (email, web access, mobile applications, phone), Veteran, service, date or resource. | X | Users can view request lists using the COTS solution's Reporting Workbench module. |  |
| BN3B1.2 | Ability to track activities for processing request list activities and capture the date/time stamped. | X |  |  |
| BN3B1.3 | Ability to capture the type of request received and track the workflow routing requirements of a request for care (such as urgent, new, change existing, or Veteran is not yet established in the system). | X |  |  |
| BN3B1.4 | Ability to route a task/work flow item to triage personnel for clinical care or urgent care decisions as a result of a request. | X |  |  |
| BN3B1.5 | Ability to create notes about attempts to contact the patient. | X |  |  |
| BN3B1.6 | System shall store notes regarding attempts to contact the patient. | X |  |  |
| BN3B1.7 | Ability to assign reminders/alerts to call a patient in the Request list if an earlier appointment becomes available prior to their scheduled appointment. | X | When a user cancels an appointment, the solution can bring up the wait list activity to offer the newly available slot to patients on the wait list. |  |
| BN3B1.8 | Ability to control access to the request list according to security roles. | X |  |  |
| BN3C1 | The MASS shall provide VHA with ability to perform request oversight activities by providing data for analysis regarding request activities. |  |  |  |
| BN3C1.1 | Ability to generate report data regarding requests (such as who submitted the request, date submitted.) | X |  |  |
| BN3C1.2 | Ability to generate report data about the method used to submit the request (such as mobile, phone, walk-in, other VA data sources such as C&P and consults, and other government network partners). | X |  |  |
| BN3C1.3 | Ability to generate report data for length of time it takes to process a request in queue from the time the request was submitted until an appointment is scheduled. | X |  |  |
| BN3C1.4 | Ability to create audit reports to monitor requests | X |  |  |
| BN3C1.5 | Ability to create audit reports to monitor frequency of actions by user. (Add, delete, modify a record, user logins/logouts) |  | Using the COTS solution's existing reporting infrastructure, Epic will add audit reports that monitor frequency of actions by user in the solution. This update is of a type customarily available in the commercial marketplace and will be performed at private expense outside the scope of the contract. |  |
| BN3C1.6 | Ability to create audit reports to monitor age of requests. | X |  |  |
| BN3C1.7 | Ability to create audit reports to monitor requests by preferred date. | X |  |  |
| BN4 | Appointment Management |  |  |  |
| BN4A1 | The Mass shall provide VHA with the ability to search available healthcare resources and to reserve resources to meet the Veteran's needs in accordance with preferred date for care from a single integrated view |  |  |  |
| BN4A1.1 | Ability to search for availability based on user-selected items (such as preferred date, date range, service, patient preferences, appointment availability, geographic considerations, facility, date range, resource, CVT equipment and special needs). | X |  |  |
| BN4A1.2 | Resources shall be searchable based on availability status. (Show as available or not available based on the search criteria) | X |  |  |
| BN4A1.3 | Search for availability must be provided in a single view. | X |  |  |
| BN4A1.4 | Ability to search for dates and times when resources will be available beyond a one-year period. | X |  |  |
| BN4A1.5 | Ability to search for the first, second and third next available appointments as calculated by the preferred date. | X |  |  |
| BN4A1.6 | Ability to generate a complete list of a single patient’s appointments while highlighting any that are in close proximity to the preferred date, so the appointments might be coordinated. | X |  |  |
| BN4A1.7 | Display all appointments at the Pilot Site for a patient to prevent double booking of appointments. | X |  |  |
| BN4A1.8 | Ability to view patient special needs when searching for availability | X |  | Duplicate of BN2C2.3 |
| BN4A1.9 | Ability to view patient preferences when searching for availability. | X |  |  |
| BN4A1.10 | Request list shall be viewable, sortable and searchable when creating or canceling appointments. | X |  |  |
| BN4A1.11 | The system shall populate the appointment scheduling screen with Veteran information from VistA. | X | We are not doing any MVI integration for the pilot site rollout. |  |
| BN4A1.12 | Ability to view availability of associated services such as laboratory tests when scheduling appointments to satisfy an order set. | X |  | The expectation is that associated services use the COTS product to check in and check out patients. Integration with Laboratory, Radiology, Physical Medicine is not required in the first two years. |
| BN4A1.13 | Ability to link appointments to the associated consult and update the consult status to scheduled. Also, to update the consult status is the appointment is cancelled, etc. | X |  |  |
| BN4A1.14 | Ability to modify appointment information. | X |  |  |
| BN4A1.15 | Ability to define business rules for scheduling appointments. | X |  |  |
| BN4A1.15.1 | Ability to associate appointments so that if a lab appt associated with a specialty care appointment is cancelled, the appointment is flagged for possible cancellation. | X |  | The expectation is that associated services use the COTS product to check in and check out patients. Integration with Laboratory, Radiology, Physical Medicine is not required in the first two years. |
| BN4A1.15.2 | Ability to define business rules to identify valid and invalid timeslots for scheduling appointments. | X |  |  |
| BN4A1.16 | Determine the order of appointments to allow patients time to transition from one appointment to another. | X |  |  |
| BN4A1.17 | Ability to schedule appointments with providers at any facility that is listed in the system (MASS) within user access and business rules constraints. | X |  | For the Pilot, this will only be those CBOCs/facilities associated with Columbus VA Amublatory Care Center (on same VistA instance). |
| BN4A1.18 | Ability to establish recurring appointments. | X |  |  |
| BN4A1.19 | The system shall provide an indicator when a mandatory association exists between appointments. | X |  | Related to BN4A1.15.1 |
| BN4A1.20 | Ability to fast-track a walk-in Veteran – scheduling into an appointment with minimum information. | X |  |  |
| BN4A1.21 | Ability to change the appointment mode of delivery (such as clinic to telephone) at the time the appointment is being made or after the appointment is made. | X |  |  |
| BN4A1.23 | Ability to schedule an appointment for a user-defined length of time. | X |  |  |
| BN4A1.24 | Ability to view the status of specialty consults related to an appointment. | X |  | The expectation is that appointments can be linked to consults and this will require integration with VistA in the first two years. |
| BN4A1.25 | Ability to link two or more appointments and maintain the association of those appointments. | X |  |  |
| BN4A1.26 | Ability to assign an alert to linked appointments. | X |  |  |
| BN4A1.27 | Ability to disposition (update the status of) all linked and associated appointments simultaneously or one at a time. | X |  |  |
| BN4A1.28 | Ability to indicate appointments for which the patient would like to be called if something comes up sooner. | X |  | Possible duplicate of BN3B1.7 |
| BN4A1.29 | System shall display an appointment status, including the status of "confirmed," which means the appointment has been confirmed with the patient. | X |  |  |
| BN4A1.30 | Ability to generate a report on status of appointments. | X |  |  |
| BN4A1.31 | Ability to access a library of stored patient instructions for specified procedures and specific circumstances (such as patient should be fasting.) | X |  |  |
| BN4B1 | The MASS shall provide VHA with the ability to assess the impact of care due to a cancellation by tracking the cancellation request to the resolution (such as rescheduling, backfilling resources, providing alternate modes of healthcare delivery options, or cancelling the appointment). |  |  |  |
| BN4B1.1 | Appointments cancelled or rescheduled must be associated with a predefined reason code. | X |  |  |
| BN4B1.2 | Ability to change the status of the patient's request, if they cancel an appointment but cannot be rescheduled at the time and a call back reminder will be required. | X |  |  |
| BN4B1.3 | Ability to generate a list of patients associated with a scheduled resource that is now unavailable (i.e., a provider or equipment is unavailable.) | X |  |  |
| BN4B1.4 | Ability to establish a highly-visible alert if a patient is identified as "high risk" based on VistA patient flags. | X |  | The expectation is that MASS will be able to read and use patient and employee flags from VistA registration. |
| BN4B1.5 | Ability to re-schedule one appointment in a series of recurring appointments. | X |  |  |
| BN4B1.6 | Ability to cancel one appointment in a series of recurring appointments. | X |  |  |
| BN4B1.7 | Ability to re-schedule a series of recurring appointments as a group. | X |  |  |
| BN4B1.8 | Ability to cancel a series of recurring appointments as a group. | X |  |  |
| BN4C1 | The MASS shall provide VHA with the ability to streamline the notification process by providing an automatically populated, standard set of templates (for letters, post cards, emails, or one-on-one phone calls) for all types of communication. |  |  |  |
| BN4C1.1 | The ability to generate manually or automatically through business rules patient notifications to communicate upcoming events. (Note: The notifications must be in formats to include email, letter, text, and postcard). | X |  | Generate an extract to send to a vendor. |
| BN4C1.2 | Provide templates for the development of patient notifications | X |  |  |
| BN4C1.3 | Ability to store patient notification preferences (i.e. one person wants email only, another text only). | X |  |  |
| BN4C1.4 | Store Notifications in a common repository (e.g.,) library. | X |  |  |
| BN4C1.5 | Track the processing of notifications with date/time stamped. | X |  |  |
| BN4C1.6 | Generate reminder notifications to send to patients for upcoming appointments. | X |  |  |
| BN4C1.7 | Generate reminder notifications to send to patients to schedule a new appointment. | X |  |  |
| BN4C1.8 | Ability to assign alerts to notification processing. | X |  |  |
| BN4C1.9 | Ability to categorize notifications by type of notification. | X |  |  |
| BN4C1.10.1 | The scheduler can enter an alternate address when making an appointment, which will be stored with the patient information in MASS. (VistA registration has multiple addresses for the Veteran.) | X | The COTS solution can record four Veteran addresses, including a temporary address with a specific date range, for patients.  Addresses are stored at the patient level. | We are not doing any ADR integration for the pilot site rollout. Epic will receive address data from initial patient loads, but nothing will be sent back to VistA and VistA will remain authoritative source.   If a user wants to add another address in MASS while scheduling, that would not feed back to VistA and they will have to manually update that back in VistA |
| BN4C1.12 | Ability to search notifications by various fields and use the search results as input to process the notifications. | X |  |  |
| BN4C1.13 | Auto-populate appropriate fields from patient demographics and appointment data for the notification process. | X |  |  |
| BN4C1.14 | Automatically use Veteran information when processing notifications. | X |  |  |
| BN4C1.15 | Ability to print an appointment confirmation letter immediately after making an appointment or at any time during the appointment management process. | X |  |  |
| BN4C1.16 | Ability to provide data for external print and mail processing systems via data extract. | X |  | Columbus has a contract with Xerox so an extract will need to be produced and sent to Xerox. |
| BN4C1.17 | Ability to send secure messages as notifications (secure messaging via portals). | X |  |  |
| BN4C1.18 | Generate reports to indicate notification messages were sent via secure messaging. | X |  |  |
| BN4C1.19 | Ability to record the date, time, and notes associated with calls to Veterans regarding appointments. | X | Epic: Scheduling related activity only |  |
| BN4C1.20 | Provide an audit log to track patient contact history related to appointments. | X | VetsEZ: Not in scope for pilot. Contact information maintained in VistA. Epic: Scheduling related activity only |  |
| BN4C1.22 | Ability to record multiple calls to a single Veteran for a given appointment or recall. | X |  |  |
| BN4D1 | The MASS shall provide VHA with the ability to monitor appointment, cancellation, reschedule and notification processes by reporting (structured, user defined, and ad-hoc) operational efficiencies or deficiencies in near-real time. |  |  |  |
| BN4D1.1 | Ability to report data regarding booked appointments based on user-selected items (such as specific dates, patient preferences, appointment availability, geographic considerations, facility, date range, resource, and special needs). | X |  |  |
| BN4D1.2 | Ability to report data regarding the availability and status of resources. | X |  |  |
| BN4D1.4 | Ability to report data regarding resource utilization (reserved or not reserved) related to appointments. | X |  |  |
| BN4D1.5 | Ability to report data regarding appointments that are rescheduled including the identification of the person (patient, provider, advocate) requesting the reschedule and the system user performing the action. | X |  |  |
| BN4D1.6 | Ability to report data regarding appointments that are cancelled including the reason, identification of the person (patient, provider, advocate) requesting the cancellation and the system user performing the action. | X |  |  |
| BN4D1.7 | Report a history of resource allocation. | X |  |  |
| BN4D1.9 | Ability to generate statistics about cancellations including: cancellations that result in a new appointment (rescheduled), who requested the cancellation, how many patients were affected by the cancellation, and the reason for the change (such as if by provider or broken equipment.) | X |  |  |
| BN4D1.11 | Ability to generate various reports at the clinic level to aid in resource management | X |  |  |
| BN4D1.11.1 | Ability to generate a report of patients with scheduled appointments with a history of high no-show rates | X |  |  |
| BN4D1.11.2 | Ability to generate reports on appointments scheduled outside of policy (e.g. delta between appointment date and preferred date) | X |  |  |
| BN5 | Coordinate Occasions of Care |  |  |  |
| BN5A1 | The MASS shall provide VHA with the ability to coordinate services and access to care at facilities under the purview of the Columbus VA Ambulatory Care Center. |  |  |  |
| BN5A1.4 | Ability to enter the reason an appointment is not scheduled at the Veteran's local facility. | X |  |  |
| BN5A1.5 | Ability to link appointments with C&P requests in VistA using the 2507 file. There is a C&P appointment type in VistA. No integration with DEAP, AMIE or CAPRI is expected. | X | Epic does not know the way in which C&P requests are sent. If they utilize same structure as CPRS, we can integrate. If not , this will be part of national rollout | VetsEZ: Integration with DEAP/CUI is not in scope for Pilot. Will integrate for National Release. |
| BN5D | Telehealth |  |  |  |
| BN5D1.1 | Ability to identify an appointment as a Telehealth appointment. Available resources at the Columbus facility will be scheduled and the information about the provider/service at the other facility will be documented in the Columbus appointments. | X |  |  |
| BN6 | Encounter Management |  |  |  |
| BN6A1 | The MASS shall provide VHA with ability to perform check-in activities for appointments. |  |  |  |
| BN6A1.1 | Display a list of daily appointments filtered by provider, PACT, service, Veteran or facility. | X |  |  |
| BN6A1.2 | Print a list of daily appointments sorted by provider, PACT, service, Veteran or facility. | X |  |  |
| BN6A1.3 | Ability for Veterans to check-in via kiosk and automatically record the mode of check-in, time checked-in, and location of check-in kiosk. | X | Check-in via third-party kiosks is available using an HL7 interface. |  |
| BN6A1.4 | Ability to modify the checked-in appointment status. | X |  |  |
| BN6A1.5 | Ability to enter a checked-in appointment status. | X |  |  |
| BN6A1.6 | Ability to create a "just in time" (unscheduled) appointments as a result of a visit to any clinic, including urgent care or walk-ins \and the ability to link the appointment to the service (DSS code). | X |  |  |
| BN6A1.7 | Ability to capture and store timestamps associated with check-in, check-out and other appointment status updates, including if the Veteran leaves without being seen, or does not show for an appointment. | X |  |  |
| BN6A1.10 | Ability to move an overbooked appointment into a cancellation time slot. | X |  |  |
| BN6A1.11 | Ability to reuse an appointment slot when there is a cancellation. | X |  |  |
| BN6B1 | The MASS shall provide VHA with the ability to capture information about encounter events to include metrics for wait-time studies. |  |  |  |
| BN6B1.1 | Ability to generate a report including the check in and check out date/time stamps. | X |  |  |
| BN6B1.3 | Ability to generate reports to track resource usage (provider). This includes: Capacity Planning (identifying required capacity to meet demand) and Capacity Control (ensuring the plan is met by managing resources). | X | The COTS solution supports reporting on demand through standard appointment, wait list, and recall reports. The solution can show available capacity through standard provider utilization and accessibility reports. |  |
| BN6C1 | The MASS shall provide VHA with the ability to perform check-out activities for appointments. |  |  |  |
| BN6C1.1 | Ability to update the appointment status in MASS, along with date/time checked out and timestamps. | X |  |  |
| BN6C1.3 | Ability to manually create follow-on appointments from the current patient appointment data (demographics, mode of care, service line, etc.) based on business rules. | X |  |  |
| BN6C1.4 | Enter a Veteran’s future appointment request with a minimum set of required fields which is stored in the system when the date is outside the scheduling horizon. | X |  |  |
| BN6C1.5 | Provide reason codes to categorize appointments for patients that check-in but are not seen by any provider (for example, left without being seen). | X |  |  |
| BN6D1 | The MASS shall provide VHA with the ability to monitor the check-in process and check-out activities for rapid decision making and identify operational efficiencies or deficiencies in near-real time. |  |  |  |
| BN6D1.1 | Ability to generate a report regarding the appointment to include patients that are checked-in as a walk-in, and when an unscheduled appointment is created. | X |  |  |
|  | Ability generate reports for wait-time metrics. |  |  |  |
| BN6D1.3 | Ability to generate a report regarding appointments with a required reference to other VA services (such as C&P). This will be limited to those services stored within MASS. | X | Only able to generate report with services that are in scope for MASS. |  |
| BN6D1.4 | Ability to generate a report regarding how resource time is utilized. | X | We can report on same day cancellations and provider cancellation. We would need more clarity on other items as there is not a standard way to report on things like moving overbook appointment or administrative work. |  |
| BN6D1.6 | Ability to capture data regarding orders for new care or services and/or follow-up care during check-out. | X |  |  |
| BN6D1.9 | Ability to capture metrics on appointment statuses such as no-shows and LWBS. | X |  |  |
| BN6D1.11 | Ability to track a Veteran’s check-in/check-out status from one appointment to the next on days with multiple same day appointments. | X |  |  |
| BN6D1.12 | Ability to capture user/time stamp data for a Veteran’s check-in/check-out status from one appointment to the next on days with multiple same day appointments. | X |  |  |
| BN7 | Reporting |  |  |  |
| BN7A1 | The MASS shall provide VHA with the ability to generate capacity management reports to illustrate capacity planning for resources and work force utilization throughout the scheduling operation. |  | Epic will do forecasting based on existing data, but predictive analytics is not scope for pilot site. |  |
| BN7A1.1 | Ability to report resource availability and utilization based on actual demand and fulfilled appointments. |  | Epic will add projected demand reporting capability based on the COTS solution's existing reporting infrastructure. This update is of a type customarily available in the commercial marketplace and will be performed at private expense outside the scope of the contract. |  |
| BN7A1.2 | Ability to generate resource and utilization reports that will include aggregated data at the Service Line level. | X |  |  |
| BN7A1.3 | Ability to produce forecasting reports based on existing data. (Predictive analytics is not in scope for pilot site.) |  | Epic will deliver metrics to forecast demand and resource utilization to help with capacity planning reports, based on our existing reporting infrastructure. This update is of a type customarily available in the commercial marketplace and will be performed at private expense outside the scope of the contract. |  |
| BN7A1.4 | Provide tools (such as a dashboards and other types of formats) to support planning and leveling. | X |  |  |
| BN7A1.5 | Provide tools such as a dashboard to monitor 'just-in-time' utilization of staff and resources to meet demand (such as backlogged, current, and future requests). | X |  |  |
| BN7A1.6 | Ability to report data to reconcile unmatched demand and supply. | X |  |  |
| BN7D1 | The MASS shall provide VHA with the ability to generate operational reports which will provide a variety of operational performance and audit reports of daily healthcare scheduling activities. |  |  |  |
| BN7D1.1 | Ability to report on operational performance of daily healthcare scheduling activities which can be aggregated from the service-line level. | X |  |  |
| BN7D1.2.1 | Ability to generate reports of performance against plans (such as planned availability against utilization) | X |  |  |
| BN8 | User Interfaces |  |  |  |
| BN8 | The MASS solution shall provide an intuitive, seamless user experience. The solution shall optimize qualities such as Accessibility, Responsiveness, Consistency, Aesthetics, and Ease-of-Use. The VA requires vendors to adhere to User Interface standards that ensure that the solution can support a large number of users with varying degrees of skills, experience, and capabilities. |  |  | The expectation is that single sign-on capabilities will be provided |
| BN8.1 | The system shall keep users informed as to where they are in the process, context of the patient and the action through appropriate on-screen feedback | X |  |  |
| BN8.2 | The system shall have configurable prompts and messages to the extent possible within the limitations of commercially available product. | X |  | Epic: As a COTS product, not all prompts and messages will be configurable to the VA's specifications |
| BN8.3 | User actions must be recoverable such that common inadvertent errors can be undone or backed out. Users will receive prompts and/or confirmation when taking an action that is not revocable - such as deleting items. Support undo and redo. | X |  |  |
| BN8.4 | The user interface shall use consistent terminology across functionality within the COTS application. Users should not have to wonder whether different words, situations, or actions mean the same thing. | X |  |  |
| BN8.5 | The user interface design will eliminate error-prone conditions and, when appropriate, check for potential errors and present users with a confirmation option before they commit to the action. | X |  |  |
| BN8.6 | Make objects, actions, and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate. Tailor able context sensitive help should be available throughout the system. | X |  |  |
| BN8.7 | Provide standard keyboard shortcuts to help with productivity. | X |  |  |
| BN8.8 | Dialogues should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility. | X |  |  |
| BN8.9 | Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution. Error messaging language shall be tailor able to VA language and processes. | X |  |  |
| BN8.10 | Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large. Tailor able context sensitive help shall be available throughout the system. | X |  |  |
| BN9 | Business Rules Management |  |  |  |
| BN9 | Business Rules Management: The MASS solution will need to include Business Rules management features. Business Rules management features are a critical component to ensure that the solutions allows for flexible changes to rules and guidelines. Business Rule Engine allows VHA to create and maintain fluid business rules so that IT involvement is minimized in modifying these rules. Business rules are typically maintained by business analysts, and can be modified to satisfy evolving business needs. |  |  |  |
| BN9.1 | Ability for business users (nonprogrammers) to define, design, document and edit business rules using an intuitive user interface.  Ideally, the business rules language should be at a high-enough level that it can be used to communicate with business owners and stakeholders either textually or visually (e.g. models). | X |  |  |
| BN9.2 | Ability to configure business rules at the field level | X | Many, but not all fields, in Epic can be configured with business rules. |  |
| BN9.3 | Ability for business rules to trigger calls to external systems and data resources as needed. | X |  |  |
| BN9.4 | Ability to model business rules in a visual environment. | X |  | Want to be able to use non-programming language to model business rules. |
| BN9.5 | Ability to store and version business rules in a repository (i.e. Change Management). | X | This can be performed by extracting business rules into a spreadsheet to store a snapshot. |  |
| BN9.6 | Ability to reuse business rules from the repository. | X |  |  |
| BN9.7 | Ability to track the utilization of rules in order to perform analysis (e.g. performance, correctness, and optimization) | X |  |  |
| BN9.9 | Ability to generate a report on all the business rules such as the context of their use, documentation, hierarchy relationship to other rules, dependencies to other rules, etc. | X | The COTS solution's flexible Reporting Workbench tool can be used to report on any Epic master file. Users can create reports on any data element in a given master file.   More sophisticated reporting may require changes to existing reporting functionality. |  |
| BN9.10 | Ability to import/export business rules to external development environment (e.g. Rational Requirements Composer, Rational Software Architect). | X | The COTS solution's master file settings can be reported upon using our Reporting Workbench tool. The report output can be extracted to a flat-file for consumption by an external tool. |  |
| BN10 | Workflow Management |  |  |  |
| BN10 | Workflow Management: A workflow management component manages, defines, coordinates and administers a series of tasks within an organization to produce a final outcome or outcomes. Workflow management systems allow the user to define, create and move different workflows for different types of jobs or processes. The MASS requires a Workflow Management component in order to manage the end-to-end work items/tasks that need to be performed as part of the scheduling process. |  |  |  |
| BN10.1 | Ability to create work items/tasks and manage these throughout the scheduling lifecycle. | X |  |  |
| BN10.2 | Ability to edit and update work items/tasks - e.g. reassign tasks to other resources. | X |  |  |
| BN10.3 | Ability to send notifications to resources participating in a work item/task and to customers. | X |  |  |
| BN10.4 | Ability to track the status of a work item/task. | X |  |  |
| BN10.5 | Ability to take automated actions on work items/tasks based on triggers or events. | X |  |  |
| BN10.6 | Ability to log work item/task, event, trigger, and action information. | X |  |  |
| BN10.7 | Ability to route work items to other resources/queues. | X |  |  |
| 1COTS out of the box: Includes COTS application and configuration. | | | |  |

# Appendix E – Business Success Criteria

*Table 15: Business Success Criteria* includes a master list of business success criteria provided by VA that serves as a reference guide for this implementation of MASS.

Table : Business Success Criteria

| Category | Measures | Metrics | Comments | Baseline Source |
| --- | --- | --- | --- | --- |
| Veterans | Decrease % of no show appointments | No Show Rate |  | VSSC |
| Veterans | Wait time – 3rd next available | Third Next Available | # of days in which a patient is able to be seen from date of request | Clinic Based Measure Only |
| Veterans | Increase % of appointments completed within 30 days of the Veteran's Preferred Date |  | Stratify: % of time we saw the patient within 7 days of the PD |  |
| Veterans | Increase % of same day appointments |  | consider Create Date = Appointment Date |  |
| Veterans | # of appointments in MyChart |  | Count of non-scheduler created appointments |  |
| Providers | Average number of completed appointments per day | Completed Appointments |  |  |
| Providers | Increased % of completed appointments; ensure data is adjusted for weather/other variables |  | consider contact time with patients if appointment length varies |  |
| Providers | # times appointment cancelled | Cancelled by Patient and Cancelled by Clinic | Phillip said they can create a cancellation reason for lack of pre-work |  |
| Schedulers | Scheduler usability/satisfaction | Survey | Survey | Need Survey |
| Schedulers | Improve accuracy of scheduled appointments (e.g., CID) | Appointment Error Rate | Need to look at the things we are measuring today in VistA & Scheduling Audit; Debbie will get report from Manny and Phillip will review it and provide us the Cadence equivalent | National Audit Tool |
| Schedulers | Average time to train a new scheduler |  | Consider time to become competent |  |
| Schedulers | Improved scheduler compliance with policies |  | CID error rate is one measure. See report metrics and audit results |  |
| Resource Managers | Amount of available FTF Clinic time and/or # of appointments |  | Over the last 30 days expect appointment number to increase |  |
| Resource Managers | Number of Fast Pass offers were sent and number accepted (versus rejected) |  |  |  |
| Resource Managers | Number of Fast Pass (the offers) that allow the patient to be seen sooner and the saved waiting time |  |  |  |

# Appendix F – Routines (Entry Points)

This section includes a detailed list of routines used for MASS.

Table : SDAM Routine

| Routines | Activities | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Routine Name | SDAM | | | | | | | | |
| Enhancement Category | New | Modify | | Delete | | No Change | | | |
| RTM | BN1A | | | | | | | | |
| Related Options |  | | | | | | | | |
| Related Routines | Routines “Called By” | | | | Routines “Called” | | | |
|  | SDVLT  SDCOAM  SDCO1 | | | | %DTC  DIR  VALM  VADPT  SDAM1  SDMXFLAG  SDUTL2  SDAM10  SDPPTEM  SDAM3  VALM1 | | | |
| Routines | Activities | | | | | | | | |
| Data Dictionary (DD) References | ^TMP  ^SD(409.62  ^SD(409.63 | | | | | | | | |
| Related Protocols | N/A | | | | | | | | |
| Related Integration Control Registrations (ICRs) | 1304, 2579 | | | | | | | | |
| Data Passing | Input | | Output Reference | | Both | | Global Reference | Local | | |
| Input Attribute Name and Definition |  | | | | | | | | |
| Output Attribute Name and Definition |  | | | | | | | | |
| Current Logic | | | | | | | | | |
| SDAM ;MJK/ALB -Appt Mgt ; 8/30/99 9:09am  ;;5.3;Scheduling;\*\*149,177,76,242,380\*\*;Aug 13, 1993  ;  D HDLKILL^SDAMEVT EN ; -- main entry point  N XQORS,VALMEVL D EN^VALM("SDAM APPT MGT")  Q  ; INIT ; -- set up appt man vars  K I,X,SDBEG,SDEND,SDB,XQORNOD,SDFN,SDCLN,DA,DR,DIE,DNM,DQ,%B,SDRES  S DIR(0)="43,213",DIR("A")="Select Patient name or Clinic name"  D ^DIR K DIR I $D(DIRUT) S VALMQUIT="" G INITQ  S SDY=Y  I SDY["DPT(" S DFN=+SDY D 2^VADPT I +VADM(6) D G:SDUP="^" INIT  . W !!,"WARNING ",VADM(7),!!  . R "Press Return to Continue or ^ to Quit: ",SDUP:DTIME  I SDY["DPT(" S SDAMTYP="P",SDFN=+SDY D INIT^SDAM1  I SDY["SC(" S SDRES=$$CLNCK^SDUTL2(+SDY,1) I 'SDRES D G INIT  . W !,?5,"Clinic MUST be corrected before continuing."  I SDY["SC(" S SDAMTYP="C",SDCLN=+SDY D INIT^SDAM3 INITQ Q | | | | | | | | | |
| Modified Logic (Changes are in bold) | | | | | | | | | |
| SDAM ;MJK/ALB - Appt Mgt ; 8/30/99 9:09am  ;;5.3;Scheduling;\*\*149,177,76,242,380,**676**\*\*;Aug 13, 1993  ;  D HDLKILL^SDAMEVT EN ; -- main entry point  N XQORS,VALMEVL D EN^VALM("SDAM APPT MGT")  Q  ; INIT ; -- set up appt man vars  K I,X,SDBEG,SDEND,SDB,XQORNOD,SDFN,SDCLN,DA,DR,DIE,DNM,DQ,%B,SDRES  S DIR(0)="43,213",DIR("A")="Select Patient name or Clinic name"  D ^DIR K DIR I $D(DIRUT) S VALMQUIT="" G INITQ  S SDY=Y  I SDY["DPT(" S DFN=+SDY D 2^VADPT I +VADM(6) D G:SDUP="^" INIT  . W !!,"WARNING ",VADM(7),!!  . R "Press Return to Continue or ^ to Quit: ",SDUP:DTIME  I SDY["DPT(" S SDAMTYP="P",SDFN=+SDY D INIT^SDAM1  **I SDY["SC",$$MSG^SDMXFLAG(+Y) G INIT ; SD/676**  I SDY["SC(" S SDRES=$$CLNCK^SDUTL2(+SDY,1) I 'SDRES D G INIT  . W !,?5,"Clinic MUST be corrected before continuing."  I SDY["SC(" S SDAMTYP="C",SDCLN=+SDY D INIT^SDAM3 INITQ Q | | | | | | | | | |

Table : SDAMEX Routine

| Routines | Activities | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Routine Name | SDAMEX | | | | | | | | |
| Enhancement Category | New | | Modify | Delete | | | No Change | | |
| RTM | BN1A | | | | | | | | |
| Related Options | Make Appointment  Multiple Appointment Booking | | | | | | | | |
| Related Routines | Routines “Called By” | | | | | Routines “Called” | | | |
|  | SDAM10 | | | | | SDAMEX1  SDM1A  DIR  DIC  SDMXFLAG  SDAM2  VALM1 | | | |
| Routines | Activities | | | | | | | | |
| Data Dictionary (DD) References | ^SC(IEN  ^DD(2.98  ^DPT(DFN  ^RT | | | | | | | | |
| Related Protocols | N/A | | | | | | | | |
| Related Integration Control Registrations (ICRs) | N/A | | | | | | | | |
| Data Passing | Input | Output Reference | | | Both | | | Global Reference | Local |
| Input Attribute Name and Definition |  | | | | | | | | |
| Output Attribute Name and Definition |  | | | | | | | | |
| Current Logic | | | | | | | | | |
| SDAMEX ;ALB/MJK,RMO - Appointment Check In/Check Out ; 12/1/91  ;;5.3;Scheduling;\*;Aug 13, 1993  CL W !,$S($D(SDCL):"Next",1:"Select")\_" Clinic: "  S SDDEF=$S($P($O(^SC(+$G(^DISV(DUZ,"^SC(")),"S",SDATE)),".")=SDATE:+$G(^DISV(DUZ,"^SC(")),1:0)  I '$D(SDCL),$G(^SC(SDDEF,0))]"" W $P(^(0),U)\_"// "  R X:DTIME  I X="",SDDEF,'$D(SDCL) S X="`"\_SDDEF  I "^"[X S SDCL=0 G CLINICQ  S:X?1" "1N.N X="`"\_$E(X,2,99)  S DIC(0)="NEMQ",DIC="^SC("  S DIC("S")="I $P(^(0),U,3)[""C"",$P($O(^(""S"",SDATE)),""."")=SDATE"  D ^DIC K DIC G CL:Y<1 S SDCL=+Y CLINICQ Q SDCL>0 | | | | | | | | | |
| Modified Logic (Changes are in bold) | | | | | | | | | |
| SDAMEX ;ALB/MJK,RMO - Appointment Check In/Check Out ; 12/1/91  ;;5.3;Scheduling;\***676**\*;Aug 13, 1993  CL W !,$S($D(SDCL):"Next",1:"Select")\_" Clinic: "  S SDDEF=$S($P($O(^SC(+$G(^DISV(DUZ,"^SC(")),"S",SDATE)),".")=SDATE:+$G(^DISV(DUZ,"^SC(")),1:0)  I '$D(SDCL),$G(^SC(SDDEF,0))]"" W $P(^(0),U)\_"// "  R X:DTIME  I X="",SDDEF,'$D(SDCL) S X="`"\_SDDEF  I "^"[X S SDCL=0 G CLINICQ  S:X?1" "1N.N X="`"\_$E(X,2,99)  S DIC(0)="NEMQ",DIC="^SC("  S DIC("S")="I $P(^(0),U,3)[""C"",$P($O(^(""S"",SDATE)),""."")=SDATE"  D ^DIC K DIC G CL:Y<1 S SDCL=+Y  **I $$MSG^SDMXFLAG(+Y) G CL ; SD/676** CLINICQ Q SDCL>0 | | | | | | | | | |

Table : SDC Routine

| Routines | Activities | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Routine Name | SDC | | | | | | | | |
| Enhancement Category | New | | Modify | Delete | | | No Change | | |
| RTM | BN1A | | | | | | | | |
| Related Options |  | | | | | | | | |
| Related Routines | Routines “Called By” | | | | | Routines “Called” | | | |
|  |  | | | | | %DT  SDC1  SDUTL  DGUTL  SDMXFLAG  SDC4  SDC2  SDC0  SDAMEVT  SDCNSLT  SDCNP0  SDOE | | | |
| Routines | Activities | | | | | | | | |
| Data Dictionary (DD) References | ^SC(IEN  ^DPT(DFN  ^DG(40.8  ^SCE | | | | | | | | |
| Related Protocols | N/A | | | | | | | | |
| Related Integration Control Registrations (ICRs) | N/A | | | | | | | | |
| Data Passing | Input | Output Reference | | | Both | | | Global Reference | Local |
| Input Attribute Name and Definition |  | | | | | | | | |
| Output Attribute Name and Definition |  | | | | | | | | |
| Current Logic | | | | | | | | | |
| SDM ;SF/GFT,ALB/BOK - MAKE AN APPOINTMENT ; 22 Jul 2016 4:33 PM  ;;5.3;Scheduling;\*\*15,32,38,41,44,79,94,167,168,218,223,250,254,296,380,478,441,619\*\*;Aug 13, 1993;Build 35  ; If defined...  ; appt mgt vars: SDFN := DFN of patient....will not be asked  ; SDCLN := ifn of clinic.....will not be asked   ; SDAMERR := returned if error occurs  ;   ; Reference to LANGDEL^DGRPE supported by DBIA #6405  ; Reference to ^DPT(DFN,.207) supported by DBIA #6406  ;  S:'$D(SDMM) SDMM=0 EN1 L W !! D I^SDUTL I '$D(SDCLN) S DIC="^SC(",DIC(0)="AEMZQ",DIC("A")="Select CLINIC: ",DIC("S")="I $P(^(0),U,3)=""C"",'$G(^(""OOS""))" D ^DIC K DIC G:Y<0!'$D(^("SL")) END  N SDRES S:$D(SDCLN) Y=+SDCLN S SDRES=$$CLNCK^SDUTL2(+Y,1)  I 'SDRES W !,?5,"Clinic MUST be corrected before continuing." G END:$D(SDCLN),SDM  K SDAPTYP,SDIN,SDRE,SDXXX S:$D(SDCLN) Y=+SDCLN  S TMPYCLNC=Y,STPCOD=$P($G(^SC(+TMPYCLNC,0)),U,7) ;SD/478 | | | | | | | | | |
| Modified Logic (Changes are in bold) | | | | | | | | | |
| SDM ;SF/GFT,ALB/BOK - MAKE AN APPOINTMENT ; 22 Jul 2016 4:33 PM  ;;5.3;Scheduling;\*\*15,32,38,41,44,79,94,167,168,218,223,250,254,296,380,478,441,619**,676**\*\*;Aug 13, 1993;Build 35  ; If defined...  ; appt mgt vars: SDFN := DFN of patient....will not be asked  ; SDCLN := ifn of clinic.....will not be asked   ; SDAMERR := returned if error occurs  ;   ; Reference to LANGDEL^DGRPE supported by DBIA #6405  ; Reference to ^DPT(DFN,.207) supported by DBIA #6406  ;  S:'$D(SDMM) SDMM=0 EN1 L W !! D I^SDUTL I '$D(SDCLN) S DIC="^SC(",DIC(0)="AEMZQ",DIC("A")="Select CLINIC: ",DIC("S")="I $P(^(0),U,3)=""C"",'$G(^(""OOS""))" D ^DIC K DIC G:Y<0!'$D(^("SL")) END  N SDRES S:$D(SDCLN) Y=+SDCLN S SDRES=$$CLNCK^SDUTL2(+Y,1)  I 'SDRES W !,?5,"Clinic MUST be corrected before continuing." G END:$D(SDCLN),SDM  **I $$MSG^SDMXFLAG(+Y) G END:$D(SDCLN),SDM ;SD/676**  K SDAPTYP,SDIN,SDRE,SDXXX S:$D(SDCLN) Y=+SDCLN  S TMPYCLNC=Y,STPCOD=$P($G(^SC(+TMPYCLNC,0)),U,7) ;SD/478; | | | | | | | | | |

Table : SDM Routine

| Routines | Activities | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Routine Name | SDM | | | | | | | | |
| Enhancement Category | New | | Modify | Delete | | | No Change | | |
| RTM | BN1A | | | | | | | | |
| Related Options | Appointment Check-in/Check-out  Multiple Clinic Display/Book | | | | | | | | |
| Related Routines | Routines “Called By” | | | | | Routines “Called” | | | |
|  | DVBCSCHD  IBDFDE1  SDCO5  SDAM2 | | | | | SDM0  SDM4  SDM1A  SDMXFLAG  SDUTL2  SDUTL  SDUTL  EASMTCHK  DGSAUTL  DICN  DIE  SDCNSLT  SCMCCON  SCAPMCU4  DGRPE | | | |
| Routines | Activities | | | | | | | | |
| Data Dictionary (DD) References | ^SC(IEN  ^DPT(DFN  ^DGS(41.1 | | | | | | | | |
| Related Protocols | N/A | | | | | | | | |
| Related Integration Control Registrations (ICRs) | 261, 739, 1584, 10042, 747 | | | | | | | | |
| Data Passing | Input | Output Reference | | | Both | | | Global Reference | Local |
| Input Attribute Name and Definition |  | | | | | | | | |
| Output Attribute Name and Definition |  | | | | | | | | |
| Current Logic | | | | | | | | | |
| SDM ;SF/GFT,ALB/BOK - MAKE AN APPOINTMENT ; 22 Jul 2016 4:33 PM  ;;5.3;Scheduling;\*\*15,32,38,41,44,79,94,167,168,218,223,250,254,296,380,478,441,619\*\*;Aug 13, 1993;Build 35  ; If defined...  ; appt mgt vars: SDFN := DFN of patient....will not be asked  ; SDCLN := ifn of clinic.....will not be asked   ; SDAMERR := returned if error occurs  ;   ; Reference to LANGDEL^DGRPE supported by DBIA #6405  ; Reference to ^DPT(DFN,.207) supported by DBIA #6406  ;  S:'$D(SDMM) SDMM=0 EN1 L W !! D I^SDUTL I '$D(SDCLN) S DIC="^SC(",DIC(0)="AEMZQ",DIC("A")="Select CLINIC: ",DIC("S")="I $P(^(0),U,3)=""C"",'$G(^(""OOS""))" D ^DIC K DIC G:Y<0!'$D(^("SL")) END  N SDRES S:$D(SDCLN) Y=+SDCLN S SDRES=$$CLNCK^SDUTL2(+Y,1)  I 'SDRES W !,?5,"Clinic MUST be corrected before continuing." G END:$D(SDCLN),SDM  K SDAPTYP,SDIN,SDRE,SDXXX S:$D(SDCLN) Y=+SDCLN  S TMPYCLNC=Y,STPCOD=$P($G(^SC(+TMPYCLNC,0)),U,7) ;SD/478 | | | | | | | | | |
| Modified Logic (Changes are in bold) | | | | | | | | | |
| SDM ;SF/GFT,ALB/BOK - MAKE AN APPOINTMENT ; 22 Jul 2016 4:33 PM  ;;5.3;Scheduling;\*\*15,32,38,41,44,79,94,167,168,218,223,250,254,296,380,478,441,619**,676**\*\*;Aug 13, 1993;Build 35  ; If defined...  ; appt mgt vars: SDFN := DFN of patient....will not be asked  ; SDCLN := ifn of clinic.....will not be asked   ; SDAMERR := returned if error occurs  ;   ; Reference to LANGDEL^DGRPE supported by DBIA #6405  ; Reference to ^DPT(DFN,.207) supported by DBIA #6406  ;  S:'$D(SDMM) SDMM=0 EN1 L W !! D I^SDUTL I '$D(SDCLN) S DIC="^SC(",DIC(0)="AEMZQ",DIC("A")="Select CLINIC: ",DIC("S")="I $P(^(0),U,3)=""C"",'$G(^(""OOS""))" D ^DIC K DIC G:Y<0!'$D(^("SL")) END  N SDRES S:$D(SDCLN) Y=+SDCLN S SDRES=$$CLNCK^SDUTL2(+Y,1)  I 'SDRES W !,?5,"Clinic MUST be corrected before continuing." G END:$D(SDCLN),SDM  **I $$MSG^SDMXFLAG(+Y) G END:$D(SDCLN),SDM ;SD/676**  K SDAPTYP,SDIN,SDRE,SDXXX S:$D(SDCLN) Y=+SDCLN  S TMPYCLNC=Y,STPCOD=$P($G(^SC(+TMPYCLNC,0)),U,7) ;SD/478; | | | | | | | | | |

Table : SDMULT Routine

| Routines | Activities | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Routine Name | SDMULT | | | | | | | | |
| Enhancement Category | New | | Modify | Delete | | | No Change | | |
| RTM | BN1A | | | | | | | | |
| Related Options | Make Appointment  Appointment Check-in/Check-out | | | | | | | | |
| Related Routines | Routines “Called By” | | | | | Routines “Called” | | | |
|  | SDM0  SDNEXT | | | | | %ZIS  SDUTL  DIC  VADPT  SDMULT0  SDUTL2  SDNEXT  SDMXFLAG  SDM1 | | | |
| Routines | Activities | | | | | | | | |
| Data Dictionary (DD) References | ^SC(IEN | | | | | | | | |
| Related Protocols | N/A | | | | | | | | |
| Related Integration Control Registrations (ICRs) | N/A | | | | | | | | |
| Data Passing | Input | Output Reference | | | Both | | | Global Reference | Local |
| Input Attribute Name and Definition |  | | | | | | | | |
| Output Attribute Name and Definition |  | | | | | | | | |
| Current Logic | | | | | | | | | |
| SDMULT ;ALB/TMP - MAKE MULTI-CLINIC APPOINTMENTS ; 02 Jan 2000 6:30 PM  ;;5.3;Scheduling;\*\*63,168,380,478\*\*;Aug 13, 1993  I '$D(DT) D DT^SDUTL  S IOP=$S($D(ION):ION,1:"HOME") D ^%ZIS K SDNEXT,SDC1,IOP 1 K SDAPTYP S SDMLT="",DIC="^DPT(",DIC(0)="AQZME" D ^DIC S DFN=+Y I "^"[X K FND S SDNEXT="" K SDMLT,SDAPTYP G END^SDMULT0  G:Y<0 1 D 2^VADPT I +VADM(6) W !?10,\*7,"PATIENT HAS DIED." G 1  S SDW=$S('$D(^DPT(DFN,.1)):"",^(.1)]"":^(.1),1:""),(SDMM,COLLAT)=0  S SDXXX="" D EN^SDM I $D(SDMLT1) K FND G END^SDMULT0  D:'$D(DT) DT^SDUTL S SDCT=0,SDMAX=DT K SDC W !!,"YOU MAY SELECT FROM 2-4 CLINICS",! RD S DIC="^SC(",DIC(0)="AEMQZ",DIC("S")="I $P(^(0),""^"",3)=""C"",'$G(^(""OOS""))",DIC("A")="Select CLINIC: " D ^DIC K DIC("S"),DIC("A") I X="",SDCT>1 G START^SDMULT0  I $S(X["^":1,'$D(DTOUT):0,$D(DTOUT)&DTOUT:1,1:0) K FND G END^SDMULT0  I $D(SDNEXT) S SDMAX=DT G:X]"" C G END^SDMULT0  I X']"" W !,\*7,"MUST HAVE MORE THAN 1 CLINIC" G RD  N SDRES S SDRES=$$CLNCK^SDUTL2(+Y,1)  I 'SDRES W !,?5,"Clinic MUST be corrected before continuing." G RD  G:Y'>0 RD I $D(SDC1(+Y)) W !,\*7,"This clinic has already been selected" G RD | | | | | | | | | |
| Modified Logic (Changes are in bold) | | | | | | | | | |
| SDMULT ;ALB/TMP - MAKE MULTI-CLINIC APPOINTMENTS ; 02 Jan 2000 6:30 PM  ;;5.3;Scheduling;\*\*63,168,380,478**,676**\*\*;Aug 13, 1993  I '$D(DT) D DT^SDUTL  S IOP=$S($D(ION):ION,1:"HOME") D ^%ZIS K SDNEXT,SDC1,IOP 1 K SDAPTYP S SDMLT="",DIC="^DPT(",DIC(0)="AQZME" D ^DIC S DFN=+Y I "^"[X K FND S SDNEXT="" K SDMLT,SDAPTYP G END^SDMULT0  G:Y<0 1 D 2^VADPT I +VADM(6) W !?10,\*7,"PATIENT HAS DIED." G 1  S SDW=$S('$D(^DPT(DFN,.1)):"",^(.1)]"":^(.1),1:""),(SDMM,COLLAT)=0  S SDXXX="" D EN^SDM I $D(SDMLT1) K FND G END^SDMULT0  D:'$D(DT) DT^SDUTL S SDCT=0,SDMAX=DT K SDC W !!,"YOU MAY SELECT FROM 2-4 CLINICS",! RD S DIC="^SC(",DIC(0)="AEMQZ",DIC("S")="I $P(^(0),""^"",3)=""C"",'$G(^(""OOS""))",DIC("A")="Select CLINIC: " D ^DIC K DIC("S"),DIC("A") I X="",SDCT>1 G START^SDMULT0  I $S(X["^":1,'$D(DTOUT):0,$D(DTOUT)&DTOUT:1,1:0) K FND G END^SDMULT0  I $D(SDNEXT) S SDMAX=DT G:X]"" C G END^SDMULT0  I X']"" W !,\*7,"MUST HAVE MORE THAN 1 CLINIC" G RD  N SDRES S SDRES=$$CLNCK^SDUTL2(+Y,1)  I 'SDRES W !,?5,"Clinic MUST be corrected before continuing." G RD  **I $$MSG^SDMXFLAG(+Y) G RD ; SD/676**  G:Y'>0 RD I $D(SDC1(+Y)) W !,\*7,"This clinic has already been selected" G RD | | | | | | | | | |

Table : SDNEXT Routine

| Routines | Activities | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Routine Name | SDNEXT | | | | | | | | |
| Enhancement Category | New | | Modify | Delete | | | No Change | | |
| RTM | BN1A | | | | | | | | |
| Related Options | Make Appointment  Multiple Appointment Booking  Appointment Check-in/Check-out | | | | | | | | |
| Related Routines | Routines “Called By” | | | | | Routines “Called” | | | |
|  | SDMULT | | | | | %DT  %ZIS  DIR  SDMULT0  SDUTL  SDNEXT1  SCRPU1  SCRPU2  SCAPMC24 | | | |
| Routines | Activities | | | | | | | | |
| Data Dictionary (DD) References | ^SC(IEN  ^SCTM(404.57 | | | | | | | | |
| Related Protocols | N/A | | | | | | | | |
| Related Integration Control Registrations (ICRs) | N/A | | | | | | | | |
| Data Passing | Input | Output Reference | | | Both | | | Global Reference | Local |
| Input Attribute Name and Definition |  | | | | | | | | |
| Output Attribute Name and Definition |  | | | | | | | | |
| Current Logic | | | | | | | | | |
| SDNEXT ;ALB/TMP - FIND NEXT AVAILABLE APPOINTMENT FOR A CLINIC ; 18 APR 86  ;;5.3;Scheduling;\*\*41,45,165,549\*\*;AUG 13, 1993;Build 2  NEW ;entry point to be use for next available appt. 3/29/96  K VAUTT,VAUTC,SCUP  N SCOKNULL  S SCOKNULL=1  S IOP=$S($D(ION):ION,1:"HOME") D ^%ZIS K IOP  S SDNEXT="",SDCT=0  S VAUTNA="" ;don't allow all to be selected  S VAUTCA="" ;allow any clinic to be selected  S VAUTD=1 ;all divisions  D CLINIC^SCRPU1 ;prompt for clinics (none,one,many)  Q:$D(SCUP) ; "^" SELECTED | | | | | | | | | |
| Modified Logic (Changes are in bold) | | | | | | | | | |
| SDNEXT ;ALB/TMP - FIND NEXT AVAILABLE APPOINTMENT FOR A CLINIC ; 18 APR 86  ;;5.3;Scheduling;\*\*41,45,165,549,**676**\*\*;AUG 13, 1993;Build 2  NEW ;entry point to be use for next available appt. 3/29/96  K VAUTT,VAUTC,SCUP  N SCOKNULL  S SCOKNULL=1  S IOP=$S($D(ION):ION,1:"HOME") D ^%ZIS K IOP  S SDNEXT="",SDCT=0  S VAUTNA="" ;don't allow all to be selected  S VAUTCA="" ;allow any clinic to be selected  S VAUTD=1 ;all divisions  **D CLINIC^SDNEXT1 ;prompt for clinics (none,one,many) ; SD\*5.3\*676  ;;D CLINIC^SCRPU1 ;prompt for clinics (none,one,many)**  Q:$D(SCUP) ; "^" SELECTED | | | | | | | | | |

Table : SDNEXT1 Routine

| Routines | Activities | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Routine Name | SDNEXT1 | | | | | | | | |
| Enhancement Category | New | | Modify | Delete | | | No Change | | |
| RTM | BN1A | | | | | | | | |
| Related Options |  | | | | | | | | |
| Related Routines | Routines “Called By” | | | | | Routines “Called” | | | |
|  | SDNEXT | | | | | DIC  DDIOL  VAUTOMA  SCRPU1  SCRPU3  SDMXFLAG | | | |
| Routines | Activities | | | | | | | | |
| Data Dictionary (DD) References | ^SCTM(404.51  ^SCTM(404.57  ^SCTM(404.52  ^SD(404.91  ^USR(8930  ^VA(200 | | | | | | | | |
| Related Protocols | N/A | | | | | | | | |
| Related Integration Control Registrations (ICRs) | N/A | | | | | | | | |
| Data Passing | Input | Output Reference | | | Both | | | Global Reference | Local |
| Input Attribute Name and Definition |  | | | | | | | | |
| Output Attribute Name and Definition |  | | | | | | | | |
| Current Logic | | | | | | | | | |
|  | | | | | | | | | |
| Modified Logic (Changes are in bold) | | | | | | | | | |
| SDNEXT1 ;MASS/JEO - CLINIC PROMPTS FOR Next Available Appointment ;11/14/17  ;;5.3;Scheduling;\*\*676\*\*;NOV 14, 2017  ;;VA DIRECTIVE 6402, this routine should not be modified.  ; Copied from SCRPU1  ; INST ;Prompt for VAUTCinstitution  S VAUTVB="VAUTD",DIC="^DIC(4,",DIC("S")="I $D(^SCTM(404.51,""AINST"",+Y))"  S VAUTNI=2,VAUTSTR="Division"  G FIRST^VAUTOMA  ; PRMTT ;Prompt for team. Set VAUTTN to allow not assigned to a team as a selection  I '$D(VAUTD) G ERR  S VAUTVB="VAUTT",DIC="^SCTM(404.51,",VAUTNI=2,VAUTSTR="Team",DIC("B")=""  S DIC("S")="I VAUTD=1!($D(VAUTD(+$P(^(0),U,7))))"  G FIRST  ; CLINIC ;Prompt for Clinic  I '$D(VAUTT)&'$D(VAUTCA) G ERR  S VAUTVB="VAUTC",VAUTSTR="Clinic",VAUTNI=2,DIC="^SC("  ;Set screen to only allow clinics and clinics that are associated to the teams selected  I '$D(VAUTCA) S DIC("S")="I $$CLSC^SDNEXT1()"  ;VAUTCA allows for selection of any clinic in the selected  I $D(VAUTCA) S DIC("S")="I $$CLSC2^SDNEXT1()"  G FIRST  ; USER ;Prompt for User Class  I '$D(VAUTT) G ERR  I $P($G(^SD(404.91,1,"PCMM")),"^")'=1 Q ;user class turned off  S VAUTVB="VAUTUC",DIC="^USR(8930,",VAUTSTR="User Class",VAUTNI=2  S DIC("S")="I $$USRCL^SCRPU1"  G FIRST  ; USRCL() ;Screen for user class - must be related to teams selected  N STOP,ENT,NODE,TIEN  I '+$P(^(0),U,3) Q 0  ;check for active/exiting user class  S ENT=0,STOP=0  F S ENT=$O(^SCTM(404.57,"AUSR",+Y,ENT)) Q:ENT=""!(STOP) D  .S NODE=$G(^SCTM(404.57,ENT,0))  .I NODE="" S STOP=0 Q  .S TIEN=+$P(NODE,"^",2) ;team ien  .I $D(VAUTT(TIEN))!(VAUTT=1) S STOP=1 Q  .I VAUTT=""&(TIEN="") S STOP=1 Q ;no team selected, no team assigned  .I VAUTT'=1&('$D(VAUTT(TIEN))) S STOP=0  Q STOP  ; ROLE ;Prompt for Role  I '$D(VAUTT) G ERR  S VAUTVB="VAUTR",DIC="^SD(403.46,",VAUTSTR="Role",VAUTNI=2  S DIC("S")="I $$RL^SCRPU1()"  G FIRST  ; RL() ;Screen for Role - screen on team  N EN,STOP,ACT,TEAM  S EN="",STOP=0  I $D(^SCTM(404.57,"AC",+Y)) D  .F S EN=$O(^SCTM(404.57,"AC",+Y,EN)) Q:EN=""!(STOP) D  ..S ACT=+$$ACTTP^SCMCTPU(EN) ;currently active?  ..I 'ACT!('$D(^SCTM(404.57,EN,0))) Q  ..S TEAM=$P(^SCTM(404.57,EN,0),"^",2)  ..I $D(VAUTT(TEAM))!(VAUTT=1) S STOP=1  ..I VAUTT=""&(TEAM="") S STOP=1  Q STOP  ; PRACT ; Prompt for One (set VAUTPO) or One,Many,All,None Practitioner(s)  I '$D(VAUTT) G ERR  S VAUTVB="VAUTP",VAUTSTR="Practitioner",VAUTNI=2,DIC="^VA(200,"  S DIC("S")="I $$PRACS^SCRPU1()"  G FIRST  ; PRACS() ;Practitioner screen - off of team selection  N EN,STOP,NODE,TEAM  S EN="",STOP=0  I '$D(^SCTM(404.52,"C",+Y)) Q 0  ;Position Assignment History file  F S EN=$O(^SCTM(404.52,"C",+Y,EN)) Q:EN=""!(STOP) D  .I '$D(^SCTM(404.52,EN)) Q  .S NODE=$G(^SCTM(404.52,EN,0))  .S TEAM=+$P($G(^SCTM(404.57,$P(NODE,"^"),0)),"^",2)  .I $P(NODE,"^",4),$D(VAUTT(TEAM)) S STOP=1  .I VAUTT=1 S STOP=1  Q STOP  ; FIRST ;  S DIC(0)="EQMNZ",DIC("A")="Select "\_VAUTSTR\_": " K @VAUTVB  S (@VAUTVB,Y)=0 REDO W !,DIC("A") R X:DTIME G ERR:(X="^")!'$T D:X["?"!(X=""&('$G(SCOKNULL))) HELP^SCRPU3  G:$G(SCOKNULL)&(X="") QUIT  I X="A"!(X="ALL")&'$D(VAUTNA) S @VAUTVB=1 G QUIT  ;VAUTNA doesn't allow all to be selected  ;VAUTTN allows 'Not assigned to a team' as a selection  I X="N"!(X="NOT")!(X="NONE") I $D(VAUTTN)!($D(VAUTPP)) S @VAUTVB="" G QUIT  ;VAUTPP allows 'Not assigned to a practitioner' as a selection  S DIC("A")="Select another "\_VAUTSTR\_": " D ^DIC G:Y'>0 FIRST D SET  I '$D(VAUTPO) F VAI=1:0:19 W !,DIC("A") R X:DTIME G ERR:(X="")!(X="^")!'$T K Y D HELP^SCRPU3:X["?" S:$E(X)="-" VAUTX=X,X=$E(VAUTX,2,999) D ^DIC I Y>0 D SET G:VAX REDO S:'VAERR VAI=VAI+1  ;VAUTPO - only one practitioner allowed to be selected  G QUIT SET S VAX=0 I $D(VAUTX) S J=$S(VAUTNI=2:+Y,1:$P(Y(0),"^")) K VAUTX S VAERR=$S($D(@VAUTVB@(J)):0,1:1) W $S('VAERR:"...removed from list...",1:"...not on list...can't remove") Q:VAERR S VAI=VAI-1 K @VAUTVB@(J) S:$O(@VAUTVB@(0))']"" VAX=1 Q  S VAERR=0 I $S($D(@VAUTVB@($P(Y(0),U))):1,$D(@VAUTVB@(+Y)):1,1:0) W !?3,\*7,"You have already selected that ",VAUTSTR,". Try again." S VAERR=1  Q:$$MSG^SDMXFLAG(+Y)  S @VAUTVB@(+Y)=$P(Y(0),U)  Q  ; ERR S Y=-1 I $O(@VAUTVB@(0))="" K @VAUTVB I X="^" S SCUP="" QUIT S:'$D(Y) Y=1  I $D(@VAUTVB),VAUTSTR="Team",@VAUTVB=1 D:'$G(DGQUIET) EN^DDIOL("All Teams selected, this report may take some time...","","!,?10")  K DIC,J,VAERR,VAI,VAJ,VAJ1,VAX,VAUTNI,VAUTSTR,VAUTVB,X  Q  ; CLSC() ;screen on clinic selection, must be related to team prompt  I $P(^(0),U,3)'="C" Q 0  N TRUE,EN,TEAM  S TRUE=0,EN=""  F S EN=$O(^SCTM(404.57,"E",+Y,EN)) Q:EN=""!(TRUE) D  .S TEAM=+$P($G(^SCTM(404.57,EN,0)),"^",2)  .I $D(VAUTT(TEAM))!(VAUTT=1) S TRUE=1  I VAUTT="" S TRUE=1  Q TRUE  ; CLSC2() ;screen on clinic selection, must be a clinic  I $P(^(0),U,3)'="C" Q 0  Q 1  ; CLSC2OLD() ;screen on clinic selection, must be related to division prompt  I $P(^(0),U,3)'="C" Q 0  N TRUE,EN,INST,TDIV  S TRUE=0,EN=""  S TDIV=+$P(^(0),U,15) ;clinic's division  Q:TDIV=0 0  S INST=+$P(^DG(40.8,TDIV,0),U,7)  I '$D(VAUTD(INST))&(VAUTD'="") S TRUE=0  I $D(VAUTD(INST)) S TRUE=1  I VAUTD=1 S TRUE=1  Q TRUE | | | | | | | | | |

Table : ORMXCONV Routine

| Routines | Activities | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Routine Name | ORMXCONV | | | | | | | | |
| Enhancement Category | New | | Modify | Delete | | | No Change | | |
| RTM | BN1A5.13, BN1A6.1, BN1A6.3, BN3A1.2, BN3A1.7, BN3B1, BN3C1.2, BN4A1.13, BN4A1.24,BN5A1.5 | | | | | | | | |
| Related Options |  | | | | | | | | |
| Related Routines | Routines “Called By” | | | | | Routines “Called” | | | |
|  | None | | | | | DIK  DIR  ORMXFMT  ORMXTR  SDMXCORE  SDMXERRO  VALM  VALM10  XQORM1 | | | |
| Routines | Activities | | | | | | | | |
| Data Dictionary (DD) References | ^DGEN(27.11  ^DPT(  ^DVB(396.3  ^DVB(396.4  ^GMR(123  ^OR(100  ^SD(403.5 | | | | | | | | |
| Related Protocols | N/A | | | | | | | | |
| Related Integration Control Registrations (ICRs) | N/A | | | | | | | | |
| Data Passing | Input | Output Reference | | | Both | | | Global Reference | Local |
| Input Attribute Name and Definition |  | | | | | | | | |
| Output Attribute Name and Definition |  | | | | | | | | |
| Current Logic | | | | | | | | | |
| N/A – New routine | | | | | | | | | |
| Modified Logic (Changes are in bold) | | | | | | | | | |
| **ORMXCONV ;MASS/AWS - ORDER CONVERSION;10/10/17  ;;3.0;Order Entry/Results Reporting;\*\*470\*\*;OCTOBER 10,2017  ;;Per VA DIRECTIVE 6402, this routine should not be modified.  ; EN ; Main entry point for ORMX Conversion Utility  D EN^VALM("ORMX CONVERSION")  Q  ; HDR ; Header code for the ORMX Conversion Utility  S VALMHDR(1)="Orders Conversion"  Q  ; INIT ; Set initial variables and list array for the ORMX Conversion Utility  D SET^VALM10(1,1\_" Orders Conversion")  D SET^VALM10(2,2\_" C&P Conversion")  D SET^VALM10(3,3\_" NEAR Conversion")  D SET^VALM10(4,4\_" Consults Conversion")  D SET^VALM10(5,5\_" Recall Reminder Conversion")  D SET^VALM10(6,6\_" Complete All Return To Clinic Orders")  D SET^VALM10(7,7\_" Complete All Recall Reminders")  ;  S VALMCNT=7  Q  ; HELP ; Help code for the ORMX Conversion Utility  S X="?" D DISP^XQORM1 W !!  Q  ; EXIT ; Exit code for the ORMX Conversion Utility  Q  ; EXPND ; Expand code for the ORMX Conversion Utility  Q  ; RUN ; Run code for the ORMX Conversion Utility  N DIR,DIRUT,DUOUT,X,Y ; Needed for call to ^DIR  ;  S DIR(0)="N^1:"\_VALMCNT\_":0"  D ^DIR  ;  S Y=$G(Y)  I Y=1 S VALMSG="Orders conversion complete." D ORDCONV^ORMXCONV  I Y=2 S VALMSG="C&P conversion complete." D CPCONV^ORMXCONV  I Y=3 S VALMSG="NEAR conversion complete." D NLCONV^ORMXCONV  I Y=4 S VALMSG="Consult conversion complete." D GMRCONV^ORMXCONV  I Y=5 S VALMSG="Recall conversion complete." D RCCONV^ORMXCONV  I Y=6 S VALMSG="All Return to Clinic orders completed." D RTCCOMP  I Y=7 S VALMSG="All recall reminders completed." D RCCOMP^ORMXCONV  ;  S VALMBCK="R"  Q  ; ORDCONV ; Orders conversion  N ORDIEN,HLOARR,ERROR,SUCCESS  ;  S ORDIEN=""  F S ORDIEN=$O(^OR(100,ORDIEN)) Q:ORDIEN="" D  . ;  . ; Don't trigger messages for consult orders since they will be handled in GMRCONV  . I $D(^GMR(123,"AC",ORDIEN)) Q  . ;  . ; Only trigger messages for PENDING [5] and ACTIVE [6] orders.  . ; The list of statuses is found in ^ORD(100.01)  . I '$$INSTRING^SDMXCORE($P($G(^OR(100,ORDIEN,3)),"^",3),"5,6",",") Q  . ;  . ; Don't trigger a message if its a type of order we won't schedule in Epic  . I $$FILTEROR^ORMXTR(ORDIEN) Q  . ;  . ; Don't trigger a message if the patient associated with the order is dead  . I $$ISPTDEAD($P($P($G(^OR(100,ORDIEN,0)),"^",2),";",1)) Q  . ;  . ; Format the HLOARR  . S SUCCESS=$$FMTOR^ORMXFMT("NW",ORDIEN,"",.HLOARR,.ERROR)  . I SUCCESS=-1 D ERRLOG^SDMXERRO(312,ERROR,1) Q  . ;  . ; Trigger the message  . S SUCCESS=$$BLDORMHL^ORMXTR(.HLOARR,.ERROR)  . I SUCCESS=-1 D ERRLOG^SDMXERRO(314,ERROR,1) Q  Q  ; GMRCONV ; Consults conversion  N GMRIEN,HLOARR,ERROR,SUCCESS  ;  S GMRIEN=""  F S GMRIEN=$O(^GMR(123,GMRIEN)) Q:GMRIEN="" D  . ;  . ; Only trigger messages for PENDING [5] and ACTIVE [6] consults.  . ; The list of statuses is found in ^ORD(100.01)  . I '$$INSTRING^SDMXCORE($P($G(^GMR(123,GMRIEN,0)),"^",12),"5,6",",") Q  . ;  . ; Don't trigger a message if the patient associated with the order is dead  . I $$ISPTDEAD($P($G(^GMR(123,GMRIEN,0)),"^",2)) Q  . ;  . ; Format the HLOARR  . S SUCCESS=$$FMTGMRC^ORMXFMT("NW",GMRIEN,"",.HLOARR,.ERROR)  . I SUCCESS=-1 D ERRLOG^SDMXERRO(312,ERROR,1) Q  . ;  . ; Trigger the message  . S SUCCESS=$$BLDORMHL^ORMXTR(.HLOARR,.ERROR)  . I SUCCESS=-1 D ERRLOG^SDMXERRO(314,ERROR,1) Q  Q  ; CPCONV ; C&P conversion  N DVBIEN,DVB0,HLOARR,ERROR,SUCCESS  ;  S DVBIEN=""  F S DVBIEN=$O(^DVB(396.4,DVBIEN)) Q:DVBIEN="" D  . ;  . S DVB0=$G(^DVB(396.4,DVBIEN,0))  . I DVB0="" Q  . ;  . ; Only trigger messages for OPEN ["O"] requests.  . I $P(DVB0,"^",4)'="O" Q  . ;  . ; Don't trigger a message if the patient associated with the request is dead  . I $$ISPTDEAD($P($G(^DVB(396.3,$P(DVB0,"^",2),0)),"^",1)) Q  . ;  . ; Format the HLOARR  . S SUCCESS=$$FMTCP^ORMXFMT("NW",DVBIEN,.HLOARR,.ERROR)  . I SUCCESS=-1 D ERRLOG^SDMXERRO(312,ERROR,1) Q  . ;  . ; Trigger the message  . S SUCCESS=$$BLDORMHL^ORMXTR(.HLOARR,.ERROR)  . I SUCCESS=-1 D ERRLOG^SDMXERRO(314,ERROR,1) Q  Q  ; NLCONV ; NEAR (new enrollee appointment request) list conversion  N DFN,DGENIEN,ERROR,SUCCESS  ;  S DFN=""  F S DFN=$O(^DGEN(27.11,"C",DFN)) Q:DFN="" D  . ;  . S DGENIEN=$O(^DGEN(27.11,"C",DFN,""),-1)  . I DGENIEN="" Q  . ;  . ; Only trigger messages for UNVERIFIED [1], VERIFIED [2], and PENDING [9] requests.  . ; The list of statuses is found in ^DGEN(27.15)  . I '$$INSTRING^SDMXCORE($P($G(^DGEN(27.11,DGENIEN,0)),"^",4),"1,2,9",",") Q  . ;  . ; Don't trigger a message if the patient associated with the request is dead  . I $$ISPTDEAD(DFN) Q  . ;  . ; Format the HLOARR  . S SUCCESS=$$FMTNL^ORMXFMT("NW",DFN,.HLOARR,.ERROR)  . I SUCCESS=-1 D ERRLOG^SDMXERRO(312,ERROR,1) Q  . ;  . ; Trigger the message  . S SUCCESS=$$BLDORMHL^ORMXTR(.HLOARR,.ERROR)  . I SUCCESS=-1 D ERRLOG^SDMXERRO(314,ERROR,1) Q  Q  ; RCCONV ; Recall Reminder conversion  N DFN,RCIFN,ERROR,SUCCESS  ;  S DFN=""  F S DFN=$O(^SD(403.5,"B",DFN)) Q:DFN="" D  . ;  . ; Don't trigger a message if the patient associated with the request is dead  . I $$ISPTDEAD(DFN) Q  . ;  . S RCIFN=""  . F S RCIFN=$O(^SD(403.5,"B",DFN,RCIFN)) Q:RCIFN="" D  . . ;  . . ; Format the HLOARR  . . S SUCCESS=$$FMTRC^ORMXFMT("NW",RCIFN,.HLOARR,.ERROR)  . . I SUCCESS=-1 D ERRLOG^SDMXERRO(312,ERROR,1) Q  . . ;  . . ; Trigger the message  . . S SUCCESS=$$BLDORMHL^ORMXTR(.HLOARR,.ERROR)  . . I SUCCESS=-1 D ERRLOG^SDMXERRO(314,ERROR,1) Q  Q  ; RCCOMP ;  N DFN,SDIEN,SDRRFTR  ;  S DFN="",SDIEN=""  F S DFN=$O(^SD(403.5,"B",DFN)) Q:DFN="" D  . F S SDIEN=$O(^SD(403.5,"B",DFN,SDIEN)) Q:SDIEN="" D  . . S DA=SDIEN,SDRRFTR=7,DIK="^SD(403.5," D ^DIK K DA,DIK  Q  ; RTCCOMP ;  N ISRTC,ORDIEN  ;  S ORDIEN=""  F S ORDIEN=$O(^OR(100,ORDIEN)) Q:ORDIEN="" D  . ;  . ; Need to make sure the order is an RTC order  . I '$$ISRTCORD^ORMXTR(ORDIEN) Q  . ;  . ; Update the status of the RTC order to "Complete"  . D COMPRTC^ORMXTR(ORIEN)  Q ISPTDEAD(DFN) ; Determine if the patient is dead  ; DFN (I,REQ) - Patient IEN  ;  ; Returns 1 if the patient is dead; 0 otherwise.  ;  I $G(^DPT(DFN,.35)) Q 1  Q 0  ;** | | | | | | | | | |

Table : ORMXFMT Routine

| Routines | Activities | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Routine Name | ORMXFMT | | | | | | | | |
| Enhancement Category | New | | Modify | Delete | | | No Change | | |
| RTM | BN1A5.13, BN1A6.1, BN1A6.3, BN3A1.2, BN3A1.7, BN3B1, BN3C1.2, BN4A1.13, BN4A1.24,BN5A1.5 | | | | | | | | |
| Related Options |  | | | | | | | | |
| Related Routines | Routines “Called By” | | | | | Routines “Called” | | | |
|  | ORMXCONV  ORMXTR  SDMXSCHI  SDMXSCHT | | | | | ORCDLG2  ORMXFMT1  ORMXTR  SDMXCORE  XPAR | | | |
| Routines | Activities | | | | | | | | |
| Data Dictionary (DD) References | ^DGEN(27.11  ^DVB(396.3  ^DVB(396.4  ^DVB(396.6  ^GMR(123  ^OR(100  ^ORD(101  ^ORD(101.43  ^SC(  ^SD(403.5 | | | | | | | | |
| Related Protocols | N/A | | | | | | | | |
| Related Integration Control Registrations (ICRs) | N/A | | | | | | | | |
| Data Passing | Input | Output Reference | | | Both | | | Global Reference | Local |
| Input Attribute Name and Definition |  | | | | | | | | |
| Output Attribute Name and Definition |  | | | | | | | | |
| Current Logic | | | | | | | | | |
| N/A – New routine | | | | | | | | | |
| Modified Logic (Changes are in bold) | | | | | | | | | |
| **ORMXFMT ;MASS/BHC - ORDER FORMATTING;8/29/17  ;;3.0;Order Entry/Results Reporting;\*\*470\*\*;AUGUST 29,2017  ;;Per VA DIRECTIVE 6402, this routine should not be modified.  ; FMTOR(CNTLCODE,ORDIEN,ORDABLE,HLOARR,ERROR) ; Format HLOARR for an order  ; Format and build the HLOARR array from the Order Entry Package. See  ; the format of the HLOARR array at the top of the routine.  ;  ; CNTLCODE (I,REQ) - The message control code (i.e. NW, DC, CA, etc.)  ; ORDIEN (I,REQ) - Order IEN  ; ORDABLE (I,OPT) - Orderable  ; HLOARR (O,REQ) - Array containing the order information  ; ERROR (O,OPT) - Contains the error reason if an error occurs  ;  ; Returns 1 if we were sucessfully able to build and format the HLOARR;  ; returns -1 otherwise.  ;  N CLINIC,DIAGCNT,DIAGSEQ,DXIEN,RSNNODE,CMTNODE,CMTCNT,DXR,TOTLINES,LINECNT  N ORTYPE,NODE,OR0,NLTFLAG,FACILITY,ISSCHBLE,DIAG,X  ;  K ERROR S ERROR=""  ;  ; Validate parameters  S CNTLCODE=$G(CNTLCODE),ORDIEN=$G(ORDIEN),ORDABLE=$G(ORDABLE)  I (CNTLCODE="")!(ORDIEN="") D Q -1  . S ERROR="Missing required parameters for FMTOR^ORMXFMT."  ;  ; Initialize HLOARR  D INITARR^ORMXFMT1(.HLOARR)  ;  ; Add the order information in HLOARR  S OR0=$G(^OR(100,ORDIEN,0))  I OR0="" D Q -1  . S ERROR="The order does not exist in ^OR. The 0 node is missing."  ;  I ORDABLE="" S ORDABLE=$G(^OR(100,ORDIEN,.1,1,0))  S ORTYPE=$$TRANORCD(2,"^OR(100")  ;  S HLOARR("ORDER ID")=$P(OR0,"^",1)\_$S(ORTYPE=-1:"^",1:"^"\_ORTYPE)  S HLOARR("PATIENT")=$$FMTPAT^ORMXFMT1($P($P(OR0,"^",2),";"))  S HLOARR("ORDERING PHYSICIAN")=$$FMTPROV^ORMXFMT1($P(OR0,"^",4))  S HLOARR("ORDER DATE")=$P(OR0,"^",7)  S HLOARR("ORDER CODE")=CNTLCODE  S HLOARR("STOP DATE")=$P(OR0,"^",9)  ;  ; RTC specific processing if the order is an RTC order  I $$ISRTCORD^ORMXTR(ORDIEN) D  . S NODE=$O(^OR(100,ORDIEN,4.5,"ID","LOCATION",0))  . ;  . ; ORDERABLE will be the clinic ID where the patient should return  . S FACILITY=$$GET^XPAR("SYS","ORMX FACILITY ID")  . S CLINIC=+$G(^OR(100,ORDIEN,4.5,+NODE,1))  . I CLINIC="" S CLINIC=$P($P(OR0,"^",10),";")  . S HLOARR("ORDERABLE")=CLINIC\_"^"\_$P($G(^SC(CLINIC,0)),"^",1)\_"^RTC"\_FACILITY  . ;  . ; Set the NLT flag  . S NODE=$O(^OR(100,ORDIEN,4.5,"ID","YN",0))  . S NLTFLAG=+$G(^OR(100,ORDIEN,4.5,+NODE,1))  . S HLOARR("NO LATER THAN")=$S(NLTFLAG:"YES",1:"") ; should be a 1 for yes if no later than  . ;  . ; Set the recurrecnces (if populated)  . S NODE=$O(^OR(100,ORDIEN,4.5,"ID","QTY",0))  . S HLOARR("RECUR")=+$G(^OR(100,ORDIEN,4.5,+NODE,1))  . ;  . ; Set the frequency (if populated)  . S NODE=$O(^OR(100,ORDIEN,4.5,"ID","MISC",0))  . S HLOARR("FREQUENCY")=$G(^OR(100,ORDIEN,4.5,+NODE,1))  . ;  . S NODE=$O(^OR(100,ORDIEN,4.5,"ID","CLINICALLY",0))  . S X=$G(^OR(100,ORDIEN,4.5,+NODE,1))  . D DT^ORCDLG2  . I Y>0 S HLOARR("START DATE")=$G(Y)  E D  . S CLINIC=$P($P(OR0,"^",10),";")  . I ORDABLE'="" S HLOARR("ORDERABLE")=ORDABLE\_"^"\_$P($G(^ORD(101.43,ORDABLE,0)),"^")\_$S(ORTYPE=-1:"",1:"^"\_ORTYPE)  . ;  . S HLOARR("START DATE")=$P(OR0,"^",8)  ;  S HLOARR("CLINIC")=$$FMTCLIN^ORMXFMT1(CLINIC)  ;  ; Add the diagnosis information in HLOARR  S DIAGCNT=1,DIAGSEQ=""  F S DIAGSEQ=$O(^OR(100,ORDIEN,5.1,DIAGSEQ)) Q:DIAGSEQ="" D  . ;  . ; Get the diagnoses for an order  . S DXIEN=$P($G(^OR(100,ORDIEN,5.1,DIAGSEQ,0)),"^") ; DXIEN=pointer to diagnosis (ICD9) file #80  . S DIAG=$$FMTDIAG^ORMXFMT1(DXIEN)  . I DIAG'="" D  . . S HLOARR("DIAGNOSIS",DIAGCNT)=DIAG  . . S DIAGCNT=DIAGCNT+1  ;  S HLOARR("DIAGNOSIS",0)=DIAGCNT-1  ;  ; Add the reason for study information in HLOARR  S LINECNT=1  S RSNNODE=$O(^OR(100,ORDIEN,4.5,"ID","REASON",""))  I RSNNODE'="" D  . ;  . ; Reason for study will only have 1 line of text.  . S HLOARR("REASON FOR STUDY",LINECNT)="REASON FOR STUDY",LINECNT=LINECNT+1  . S HLOARR("REASON FOR STUDY",LINECNT)=$G(^OR(100,ORDIEN,4.5,RSNNODE,1)),LINECNT=LINECNT+1  ;  S HLOARR("REASON FOR STUDY",0)=LINECNT-1  ;  ; Add the comments in HLOARR  D FMTORCM^ORMXFMT1(ORDIEN,.HLOARR)  ;  Q 1  ; FMTGMRC(CNTLCODE,GMRIEN,ORDABLE,HLOARR,ERROR) ; Format HLOARR for a consult  ; Format and build the HLOARR array from the Consult Package. See  ; the format of the HLOARR array at the top of the routine.  ;  ; CNTLCODE (I,REQ) - The message control code (i.e. NW, DC, CA, etc.)  ; GMRIEN (I,REQ) - Consult IEN  ; ORDABLE (I,OPT) - Orderable  ; HLOARR (IO,REQ) - Array containing the order information  ; ERROR (O,OPT) - Contains the error reason if an error occurs  ;  ; Returns 1 if we were sucessfully able to build and format the HLOARR;  ; returns -1 otherwise.  ;  N CLINIC,ORDIEN,RSNCNT,NTECNT,CMTNODE,CMTCNT,TOTLINES,DIAGCNT,DXIEN,CMTSTS,ORTYPE,DIAG,URGENCY  ;  K ERROR S ERROR=""  ;  ; Validate parameters  S CNTLCODE=$G(CNTLCODE),GMRIEN=$G(GMRIEN),ORDABLE=$G(ORDABLE)  I (CNTLCODE="")!(GMRIEN="") D Q -1  . S ERROR="Missing required parameters for FMTGMRC^ORMXFMT."  ;  ; Initialize HLOARR  D INITARR^ORMXFMT1(.HLOARR)  ;  ; Add the order information in HLOARR  S ORDIEN=$P($G(^GMR(123,GMRIEN,0)),"^",3)  I ORDIEN="" D Q -1  . S ERROR="Can not find the associated order for the consult in ^OR."  ;  S OR0=$G(^OR(100,ORDIEN,0))  I OR0="" D Q -1  . S ERROR="The order does not exist in ^OR. The 0 node is missing."  ;  S ORTYPE=$$TRANORCD(2,"^OR(100")  ;  S HLOARR("ORDER ID")=ORDIEN\_$S(ORTYPE=-1:"",1:"^"\_ORTYPE)  S HLOARR("PATIENT")=$$FMTPAT^ORMXFMT1($P($P(OR0,"^",2),";"))  I ORDABLE'="" S HLOARR("ORDERABLE")=ORDABLE\_"^"\_$P($G(^ORD(101.43,ORDABLE,0)),"^")\_$S(ORTYPE=-1:"",1:"^"\_ORTYPE)  ;  S HLOARR("ORDERING PHYSICIAN")=$$FMTPROV^ORMXFMT1($P(OR0,"^",4))  S HLOARR("ORDER DATE")=$P(OR0,"^",7)  S HLOARR("ORDER CODE")=CNTLCODE  ;  ; If the control code is "CA" (or cancel), set the cancel reason  I CNTLCODE="CA" D  . ;  . ; Set a specific cancel reason for forwards  . I $$INSTRING^SDMXCORE($P($G(^GMR(123,$G(GMRIEN),0)),"^",13),"17,25",",") S HLOARR("CANCEL REASON")="FWD"  . E S HLOARR("CANCEL REASON")=""  ;  S HLOARR("START DATE")=$P($G(^GMR(123,GMRIEN,0)),"^",24)  I HLOARR("START DATE")="" S HLOARR("START DATE")=$P(OR0,"^",8) ; Default the order start date if no CID  S HLOARR("STOP DATE")=$P(OR0,"^",9)  ;  ; URGENCY  S URGENCY=$P(^GMR(123,GMRIEN,0),"^",9)  I URGENCY'="" S HLOARR("URGENCY")=$P(^ORD(101,URGENCY,0),"^",2)  ;  ; Add the clinic information in HLOARR  S CLINIC=$P($P(OR0,"^",10),";")  S HLOARR("CLINIC")=$$FMTCLIN^ORMXFMT1(CLINIC)  ;  ; Add the diagnosis information in HLOARR  S DIAGCNT=1  I $D(^GMR(123,"AC",ORDIEN)) D  . S DXIEN=$P($G(^GMR(123,GMRIEN,30.1)),"^") ; DXIEN=pointer to diagnosis (ICD9) file #80  . S DIAG=$$FMTDIAG^ORMXFMT1(DXIEN)  . I DIAG'="" D  . . S HLOARR("DIAGNOSIS",DIAGCNT)=DIAG  . . S DIAGCNT=DIAGCNT+1  ;  S HLOARR("DIAGNOSIS",0)=DIAGCNT-1  ;  ; Add the reason for study information in HLOARR  S LINECNT=1  S TOTLINES=$P($G(^GMR(123,GMRIEN,20,0)),"^",3)  I TOTLINES'="" D  . ;  . ; Loop through the reason for study and add to the HLOARR  . S HLOARR("REASON FOR STUDY",LINECNT)="REASON FOR STUDY",LINECNT=LINECNT+1  . F RSNCNT=1:1:TOTLINES D  . . S HLOARR("REASON FOR STUDY",LINECNT)=$G(^GMR(123,GMRIEN,20,RSNCNT,0)),LINECNT=LINECNT+1  ;  S HLOARR("REASON FOR STUDY",0)=LINECNT-1  ;  ; Add the comments in HLOARR  D FMTGMRCM^ORMXFMT1(ORDIEN,.HLOARR)  ;  Q 1  ; FMTCP(CNTLCODE,DVBIEN,HLOARR,ERROR) ; Format HLOARR for a C&P order  ; Format and build the HLOARR array from the C&P Package. See  ; the format of the HLOARR array at the top of the routine.  ;  ; CNTLCODE (I,REQ) - The message control code (i.e. NW, DC, CA, etc.)  ; DVBIEN (I,REQ) - C&P IEN  ; HLOARR (IO,REQ) - Array containing the order information  ; ERROR (O,OPT) - Contains the error reason if an error occurs  ;  ; Returns 1 if we were sucessfully able to build and format the HLOARR;  ; returns -1 otherwise.  ;  N DVB0,CP0,ORCODE,ORTYPE,LINE,FACILITY,LINECNT,REQIEN  ;  K ERROR S ERROR=""  ;  ; Validate parameters  S CNTLCODE=$G(CNTLCODE),DVBIEN=$G(DVBIEN)  I (CNTLCODE="")!(DVBIEN="") D Q -1  . S ERROR="Missing required parameters for FMTCP^ORMXFMT."  ;  ; Initialize HLOARR  D INITARR^ORMXFMT1(.HLOARR)  ;  S DVB0=$G(^DVB(396.4,DVBIEN,0))  I DVB0="" D Q -1  . S ERROR="The C&P request does not exist in ^DVB. The 0 node is missing."  ;  S CP0=$G(^DVB(396.3,$P(DVB0,"^",2),0))  S FACILITY=$$GET^XPAR("SYS","ORMX FACILITY ID")  S ORTYPE=$$TRANORCD(2,"^DVB(396.4")  ;  S HLOARR("ORDER ID")=DVBIEN\_$S(ORTYPE=-1:"",1:"^"\_ORTYPE)  S HLOARR("ORDERABLE")=$P(DVB0,"^",3)\_"^"\_$P($G(^DVB(396.6,$P(DVB0,"^",3),0)),"^",1)\_$S(ORTYPE=-1:"",1:"^"\_ORTYPE)  S HLOARR("CLINIC")=$$GET^XPAR("SYS","ORMX C&P DEPARTMENT ID")\_"^"\_FACILITY\_"^^SC"\_FACILITY  S HLOARR("PATIENT")=$$FMTPAT^ORMXFMT1($P(CP0,"^",1))  S HLOARR("ORDER CODE")=CNTLCODE  ;  S REQIEN=$P(DVB0,"^",2)  I REQIEN'="" D  . S LINECNT=$P($G(^DVB(396.3,REQIEN,2,0)),"^",3)  . F LINE=1:1:LINECNT D  . . S HLOARR("COMMENT",LINE)=^DVB(396.3,REQIEN,2,LINE,0)  . ;  . S HLOARR("COMMENT",0)=LINECNT  ;  Q 1  ; FMTNL(CNTLCODE,DFN,HLOARR,ERROR) ; Format HLOARR for a NEAR order  ; Format and build the HLOARR array from the NEAR Package. See  ; the format of the HLOARR array at the top of the routine.  ;  ; CNTLCODE (I,REQ) - The message control code (i.e. NW, DC, CA, etc.)  ; DFN (I,REQ) - Patient IEN  ; HLOARR (IO,REQ) - Array containing the order information  ; ERROR (O,OPT) - Contains the error reason if an error occurs  ;  ; Returns 1 if we were sucessfully able to build and format the HLOARR;  ; returns -1 otherwise.  ;  N DGENIEN,ORTYPE,FACILITY  ;  K ERROR S ERROR=""  ;  ; Validate parameters  S CNTLCODE=$G(CNTLCODE),DFN=$G(DFN)  I (CNTLCODE="")!(DFN="") D Q -1  . S ERROR="Missing required parameters for FMTNL^ORMXFMT."  ;  ; Initialize HLOARR  D INITARR^ORMXFMT1(.HLOARR)  ;  ; Get the most recent enrollment record for the patient  S DGENIEN=$O(^DGEN(27.11,"C",DFN,""),-1)  I DGENIEN="" D Q -1  . S ERROR="Unable to retrieve the most recent enrollment record for the patient."  ;  S ORTYPE=$$TRANORCD(2,"^DGEN(27.11")  S FACILITY=$$GET^XPAR("SYS","ORMX FACILITY ID")  ;  S HLOARR("PATIENT")=$$FMTPAT^ORMXFMT1(DFN)  S HLOARR("ORDER CODE")=CNTLCODE  S HLOARR("ORDER ID")=DGENIEN\_$S(ORTYPE=-1:"",1:"^"\_ORTYPE)  S HLOARR("ORDERABLE")="NEAR^New Enrollee Appointment Request"\_$S(ORTYPE=-1:"",1:"^"\_ORTYPE)  S HLOARR("CLINIC")=$$GET^XPAR("SYS","ORMX NEAR DEPARTMENT ID")\_"^"\_FACILITY\_"^^SC"\_FACILITY  S HLOARR("ORDER DATE")=$P($G(^DGEN(27.11,DGENIEN,0)),"^",1)  S HLOARR("ORDERING PHYSICIAN")=$$FMTPROV^ORMXFMT1($P($G(^DGEN(27.11,DGENIEN,"U")),"^",2))  S HLOARR("START DATE")=$P($G(^DGEN(27.11,DGENIEN,"U")),"^",1)  ;  Q 1  ; FMTRC(CNTLCODE,RCIFN,HLOARR,ERROR) ; Format HLOARR for a recall reminder  ; Format and build the HLOARR array from the recall reminder Package. See  ; the format of the HLOARR array at the top of the routine.  ;  ; CNTLCODE (I,REQ) - The message control code (i.e. NW, DC, CA, etc.)  ; RCIFN (I,REQ) - Recall reminder IEN  ; HLOARR (IO,REQ) - Array containing the order information  ; ERROR (O,OPT) - Contains the error reason if an error occurs  ;  ; Returns 1 if we were sucessfully able to build and format the HLOARR;  ; returns -1 otherwise.  ;  N SD0,ORTYPE,CLINIC,COMMENT  ;  K ERROR S ERROR=""  ;  ; Validate parameters  S CNTLCODE=$G(CNTLCODE),RCIFN=$G(RCIFN)  I (CNTLCODE="")!(RCIFN="") D Q -1  . S ERROR="Missing required parameters for FMTRC^ORMXFMT."  ;  ; Initialize HLOARR  D INITARR^ORMXFMT1(.HLOARR)  ;  S SD0=$G(^SD(403.5,RCIFN,0))  I SD0="" D Q -1  . S ERROR="The recall reminder does not exist in ^SD. The 0 node is missing."  ;  S ORTYPE=$$TRANORCD(2,"^SD(403.5")  ;  S HLOARR("PATIENT")=$$FMTPAT^ORMXFMT1($P(SD0,"^",1))  S HLOARR("ORDER CODE")=CNTLCODE  S HLOARR("ORDER ID")=RCIFN\_$S(ORTYPE=-1:"",1:"^"\_ORTYPE)  S HLOARR("ORDER DATE")=$P(SD0,"^",14)  S HLOARR("ORDERING PHYSICIAN")=$$FMTPROV^ORMXFMT1($P(SD0,"^",5))  S HLOARR("START DATE")=$P(SD0,"^",6)  ;  ; Add the clinic information in HLOARR  S CLINIC=$P(SD0,"^",2)  S HLOARR("CLINIC")=$$FMTCLIN^ORMXFMT1(CLINIC)  ;  ; ORDERABLE will be the clinic ID where the patient should return  S HLOARR("ORDERABLE")=CLINIC\_"^"\_$S(CLINIC'="":$P($G(^SC(CLINIC,0)),"^",1),1:"")\_"^RTC"  ;  ; Add the comments in HLOARR  S COMMENT=$P(SD0,"^",7)  I COMMENT'="" D  . ;  . ; There's only line for a comment in the global, so we will hardcode the line count  . S HLOARR("COMMENT",1)=COMMENT  . S HLOARR("COMMENT",0)=1  ;  Q 1  ; TRANORCD(DIRECT,VALUE) ; Translates a file string to its HLO order type code and visa-versa  ; DIRECT (I,REQ) - 1 for outbound, 2 for inbound  ; VALUE (I,REQ) - the value to be translated. File for outbound  ; or the HLO order type code for inbound  ;  ; Returns the translated value if one is found, otherwise returns -1  ;  N TRANARY,FACILITY  ;  ; Data validation  I ($G(DIRECT)="")!($G(VALUE)="") Q -1  ;  ; Get Facility ID  S FACILITY=$$GET^XPAR("SYS","ORMX FACILITY ID")  ;  ; Setting up translation array  S TRANARY(1,"OR"\_FACILITY)="^OR(100" ; Orders  S TRANARY(1,"GMR"\_FACILITY)="^GMR(123" ; Consults  S TRANARY(1,"CP"\_FACILITY)="^DVB(396.4" ; C&P exams  S TRANARY(1,"NEAR"\_FACILITY)="^DGEN(27.11" ; NEAR list entries  S TRANARY(1,"RC"\_FACILITY)="^SD(403.5" ; Recall reminders  S TRANARY(2,"^OR(100")="OR"\_FACILITY  S TRANARY(2,"^GMR(123")="GMR"\_FACILITY  S TRANARY(2,"^DVB(396.4")="CP"\_FACILITY  S TRANARY(2,"^DGEN(27.11")="NEAR"\_FACILITY  S TRANARY(2,"^SD(403.5")="RC"\_FACILITY  ;  I $D(TRANARY(DIRECT,VALUE)) Q TRANARY(DIRECT,VALUE)  ;  Q -1  ;** | | | | | | | | | |

Table : ORMXTR Routine

| Routines | Activities | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Routine Name | ORMXTR | | | | | | | | |
| Enhancement Category | New | | Modify | Delete | | | No Change | | |
| RTM | BN1A5.13, BN1A6.1, BN1A6.3, BN3A1.2, BN3A1.7, BN3B1, BN3C1.2, BN4A1.13, BN4A1.24, BN5A1.5 | | | | | | | | |
| Related Options |  | | | | | | | | |
| Related Routines | Routines “Called By” | | | | | Routines “Called” | | | |
|  | ORMXCONV  ORMXFMT | | | | | DGMXHL7  HLOAPI  HLOAPI1  HLOAPI4  ORCSAVE2  ORM  ORMXFMT  SDMXCORE  SDMXERRO  SDMXLKRQ  XPAR | | | |
| Routines | Activities | | | | | | | | |
| Data Dictionary (DD) References | ^DPT(  ^DVB(396.4  ^GMR(123  ^OR(100  ^ORD(101.43 | | | | | | | | |
| Related Protocols | N/A | | | | | | | | |
| Related Integration Control Registrations (ICRs) | N/A | | | | | | | | |
| Data Passing | Input | Output Reference | | | Both | | | Global Reference | Local |
| Input Attribute Name and Definition |  | | | | | | | | |
| Output Attribute Name and Definition |  | | | | | | | | |
| Current Logic | | | | | | | | | |
| N/A – New routine | | | | | | | | | |
| Modified Logic (Changes are in bold) | | | | | | | | | |
| **ORMXTR ;MASS/AWS - ORDER TRIGGERING;08/22/17  ;;3.0;Order Entry/Results Reporting;\*\*470\*\*;AUGUST 22,2017  ;;Per VA DIRECTIVE 6402, this routine should not be modified.  ; TRORDMSG ; Trigger order message  ; Triggers an order message out of VistA using HLO APIs.  ;  ; Variables assumed to be set from calling function/protocol/cross reference:  ; CODE - The message control code (i.e. NW, DC, CA, etc.)  ; ORIFN - Order ID  ;  N HLOARR,ERROR,SUCCESS,OR0,ORIEN,CNTLCODE,ORDABLE  ;  ; If MASS isn't enabled, don't trigger messages  I '$$GET^XPAR("SYS","SDMX MASS ENABLED") Q  ;  S ORIEN=$P($G(ORIFN),";",1)  I ORIEN="" Q  S ORDABLE=$G(^OR(100,ORIEN,.1,1,0))  ;  ; Filter out unnecessary triggers  I $$INCINTF^SDMXCORE() Q ; Prevents rebounding messages from incoming scheduling  I $D(^GMR(123,"AC",ORIEN)) Q ; Prevents consults from triggering duplicate messages  I '$D(CODE) Q ; Prevents triggering messages for batch header and taiprevler event calls  I $$FILTEROR(ORDABLE) Q ; Prevent certain orders from triggering a message  ;  S CNTLCODE=$S($G(CODE)="DC":"CA",1:$G(CODE))  ;  ; Build the HLOARR with the necessary information  S SUCCESS=$$FMTOR^ORMXFMT(CNTLCODE,ORIEN,"",.HLOARR,.ERROR)  I SUCCESS=-1 D ERRLOG^SDMXERRO(312,ERROR,1) Q  ;  ; Trigger the message  S SUCCESS=$$BLDORMHL(.HLOARR,.ERROR)  I SUCCESS=-1 D ERRLOG^SDMXERRO(314,ERROR,1) Q  ;  ; If the order is an RTC order, set the status to "Complete"  I $$ISRTCORD(ORIEN) D  . S SUCCESS=$$COMPRTC(ORIEN)  . I SUCCESS=-1 D ERRLOG^SDMXERRO(314,ERROR,1) Q  ;  Q  ; TRGMRMSG(UPDATE) ; Trigger consult message  ; Triggers a consult message out of VistA using HLO APIs.  ;  ; UPDATE (I,OPT) - Set to 1 to force an update message (control code is "XO")  ;  ; Variables assumed to be set from calling function/protocol/cross reference:  ; GMRCO - Consult ID  ;  N HLOARR,ERROR,SUCCESS,ACTION,CNTLCODE,CNTLLIST,CURR,ORDABLE,LASTNODE,FWDFROM,ORIEN  ;  ; If MASS isn't enabled, don't trigger messages  I '$$GET^XPAR("SYS","SDMX MASS ENABLED") Q  ;  ; Filter out unnecessary triggers  I $$INCINTF^SDMXCORE() Q ; Prevents rebounding messages from incoming scheduling  ;  I $G(GMRCO)="" Q  S ORIEN=$P($G(^GMR(123,$G(GMRCO),0)),"^",3)  ;  ; Determine the event (i.e. control code). We can force an update message by  ; setting update to 1  I $D(UPDATE) D  . S CNTLLIST="XO"  E D  . S ACTION=$P($G(^GMR(123,$G(GMRCO),0)),"^",13) ; Last action taken  . S CNTLLIST=$S(ACTION=2:"NW",ACTION=6:"CA",ACTION=17:"CA,NW",ACTION=19:"CA",ACTION=20:"XO",ACTION=25:"CA,NW",1:"")  I CNTLLIST="" Q  ;  S CURR=""  F CURR=1:1:$L(CNTLLIST,",") D  . S CNTLCODE=$P(CNTLLIST,",",CURR)  . ;  . ; For consult forwards, we want to send a message for the old orderable  . I ($$INSTRING^SDMXCORE(ACTION,"17,25",",")),(CNTLCODE="CA") D  . . S LASTNODE=$P(^GMR(123,$G(GMRCO),40,0),"^",3)  . . S FWDFROM=$P(^GMR(123,$G(GMRCO),40,LASTNODE,0),"^",6)  . . S ORDABLE=$$ORDITEM^ORM("^^^"\_FWDFROM\_"^^99CON")  . E S ORDABLE=$G(^OR(100,ORIEN,.1,1,0)) ; Lookup the orderable in the FMT routine  . ;  . ; Prevent certain orders from triggering a message  . I $$FILTEROR(ORDABLE) Q  . ;  . ; Build the HLOARR with the necessary information  . S SUCCESS=$$FMTGMRC^ORMXFMT(CNTLCODE,$G(GMRCO),ORDABLE,.HLOARR,.ERROR)  . I SUCCESS=-1 D ERRLOG^SDMXERRO(312,ERROR,1) Q  . ;  . ; Trigger the message  . S SUCCESS=$$BLDORMHL(.HLOARR,.ERROR)  . I SUCCESS=-1 D ERRLOG^SDMXERRO(314,ERROR,1) Q  ;  Q  ; TRCPMSG(UPDATE) ; Trigger C&P message  ; Triggers a C&P order message out of VistA using HLO APIs.  ;  ; UPDATE (I,OPT) - Set to 1 to force an update message (control code is "XO")  ;  ; Variables assumed to be set from calling function/protocol/cross reference:  ; D0 - C&P ID  ;  N HLOARR,ERROR,SUCCESS,CNTLCODE,DVB0  ;  ; If MASS isn't enabled, don't trigger messages  I '$$GET^XPAR("SYS","SDMX MASS ENABLED") Q  ;  ; Filter out unnecessary triggers  I $$INCINTF^SDMXCORE() Q ; Prevents rebounding messages from incoming scheduling  ;  I $G(D0)="" Q  ;  ; Build the HLOARR with the necessary information  S DVB0=$G(^DVB(396.4,$G(D0),0))  S CNTLCODE=$S($D(UPDATE):"XO",$P(DVB0,"^",4)="O":"NW",$$INSTRING^SDMXCORE($P(DVB0,"^",4),"X,RX,T",","):"CA",1:"")  I CNTLCODE="" Q  ;  S SUCCESS=$$FMTCP^ORMXFMT(CNTLCODE,$G(D0),.HLOARR,.ERROR)  I SUCCESS=-1 D ERRLOG^SDMXERRO(312,ERROR,1) Q  ;  ; Trigger the message  S SUCCESS=$$BLDORMHL(.HLOARR,.ERROR)  I SUCCESS=-1 D ERRLOG^SDMXERRO(314,ERROR,1) Q  ;  Q  ; TRNLMSG(SENDCANC) ; Trigger NEAR message  ; Triggers a NEAR order message out of VistA using HLO APIs.  ;  ; SENDCANC (I,OPT) - Set to 1 to send a cancel message  ;  ; Variables assumed to be set from calling function/protocol/cross reference:  ; DFN - Patient ID  ;  N HLOARR,ERROR,SUCCESS,CURR,CNTLLIST,CNTLCODE,FLAG  ;  ; If MASS isn't enabled, don't trigger messages  I '$$GET^XPAR("SYS","SDMX MASS ENABLED") Q  ;  ; If the NEAR is filled, don't trigger messages  S FLAG=$P(^DPT($G(DFN),1010.16),"^",1)  I $G(FLAG)="F" Q  ;  I $D(SENDCANC) S CNTLLIST="CA,NW"  E S CNTLLIST="NW"  ;  S CURR=""  F CURR=1:1:$L(CNTLLIST,",") D  . S CNTLCODE=$P(CNTLLIST,",",CURR)  . ;  . ; Build the HLOARR with the necessary information  . S SUCCESS=$$FMTNL^ORMXFMT(CNTLCODE,$G(DFN),.HLOARR,.ERROR)  . I SUCCESS=-1 D ERRLOG^SDMXERRO(312,ERROR,1) Q  . ;  . ; Trigger the message  . S SUCCESS=$$BLDORMHL(.HLOARR,.ERROR)  . I SUCCESS=-1 D ERRLOG^SDMXERRO(314,ERROR,1) Q  ;  Q  ; TRCMTMSG ; Trigger an "Add comment" message  ; Determines whether or not the order is a consult order or not.  ; If the order is a consult order, we will call TRGMRMSG to trigger  ; the message. If not, we will call TRORDMSG  ;  ; Variables assumed to be set from calling function/protocol/cross reference:  ; CODE - The message control code (i.e. NW, DC, CA, etc.)  ; ORIFN - Order ID  ;  N CODE,GMRCO  ;  I $G(ORIFN)="" Q  ;  ; Check if the order is a consult order  I $D(^GMR(123,"AC",$P($G(ORIFN),";",1))) D  . ; Set variables TRORDMSG assumes to be set before calling  . S GMRCO=$O(^GMR(123,"AC",$P($G(ORIFN),";",1),""))  . ;  . D TRGMRMSG(1)  E D  . ; Set variables TRORDMSG assumes to be set before calling  . S CODE="XO"  . ;  . D TRORDMSG  ;  Q  ; TRCPCMT ; Trigger a comment update for each exam when the comments  ; on the request are updated  N D0  F S D0=$O(^DVB(396.4,"C",DA,D0)) Q:D0="" D ;loop through all exams on request  . D TRCPMSG(1) ; trigger a message  Q  ; FILTEROR(ORDABLE) ; Filter certain orders from triggering  ; Filters orders from triggering messages based on the type of the order and  ; whether or not the order is schedulable.  ;  ; ORDABLE - Orderable ID  ;  ; Returns 1 if the order should be filtered from triggering a message; 0  ; otherwise  ;  N ISSCHBLE,STATUS  ;  I ORDABLE'="" S ISSCHBLE=$P($G(^ORD(101.43,$G(ORDABLE),0)),"^",9)  E S ISSCHBLE=""  ;  I ISSCHBLE="" S ISSCHBLE=$$GET^XPAR("SYS","ORMX SCH DEFAULT FLAG")  I ISSCHBLE=0 Q 1  ;  S STATUS=$P($G(^OR(100,ORIEN,3)),"^",3)  I $$INSTRING^SDMXCORE(STATUS,"11",",") Q 1  ;  Q 0  ; ISRTCORD(ORIEN) ; Determine if the order is an RTC order  ;  ; ORIEN - Order ID  ;  ; Returns 1 if the order is an RTC order; 0 otherwise  ;  N ORDABLE  ;  S ORIEN=$G(ORIEN)  I ORIEN="" Q 0  ;  S ORDABLE=$G(^OR(100,$G(ORIEN),.1,1,0))  Q $$INSTRING^SDMXCORE(ORDABLE,$$GET^XPAR("SYS","ORMX RTC ORDERABLE ID"),",")  ; COMPRTC(ORIEN) ; Determine if the order is an RTC order  ;  ; ORIEN - Order ID  ;  ; Returns 1 if the RTC order was completed; -1 otherwise  ;  S ORIEN=$G(ORIEN)  I ORIEN="" Q -1  ;  I '$$LOCKORD^SDMXLKRQ(ORIEN) Q -1  D STATUS^ORCSAVE2(ORIEN,2)  I '$$ULOCKORD^SDMXLKRQ(ORIEN) Q -1  Q 1  ; BLDORMHL(HLOARR,ERROR) ; Build order HLO  ; Builds an ORM message using HLO APIs.  ;  ; HLOARR (I,OPT) - Array containing the order information  ; ERROR (O,OPT) - Contains the error reason if an error occurs  ;  ; Returns the message number on success; -1 otherwise  ;  N PARMS,HLMSTATE,SEG,DFN,DIAGSEQ,NTECNT,WHOTO,RSNCNT,CMTCNT,OK  ;  K ERROR S ERROR=""  ;  ; Set necessary variables  S PARMS("EVENT")="O01"  S PARMS("MESSAGE TYPE")="ORM"  S PARMS("MESSAGE STRUCTURE")="ORM\_O01"  S PARMS("VERSION")="2.4"  ;  I '$$NEWMSG^HLOAPI(.PARMS,.HLMSTATE,.ERROR) Q -1  ;  ; Build PID segment  S DFN=$P($G(HLOARR("PATIENT")),"^",1)  ;  D PID^DGMXHL7(DFN,1,.SEG)  I '$$ADDSEG^HLOAPI(.HLMSTATE,.SEG,.ERROR) Q -1  ;  ; Build PV1 segment  D PV1(.HLOARR,.SEG)  I '$$ADDSEG^HLOAPI(.HLMSTATE,.SEG,.ERROR) Q -1  ;  ; Build ORC segment  D ORC(.HLOARR,.SEG)  I '$$ADDSEG^HLOAPI(.HLMSTATE,.SEG,.ERROR) Q -1  ;  ; Build OBR segment  D OBR(.HLOARR,1,.SEG)  I '$$ADDSEG^HLOAPI(.HLMSTATE,.SEG,.ERROR) Q -1  ;  ; Build NTE segments (contains reason for study, reoccurence, and comments)  S NTECNT=1  F RSNCNT=1:1:$G(HLOARR("REASON FOR STUDY",0)) D  . D NTE($G(HLOARR("REASON FOR STUDY",RSNCNT)),"CMT",NTECNT,.SEG)  . I $$ADDSEG^HLOAPI(.HLMSTATE,.SEG,.ERROR) S NTECNT=NTECNT+1  ;  ; Add a blank line between comments  I (NTECNT>1),($G(HLOARR("RECUR"))'="") D  . D NTE(" ","CMT",NTECNT,.SEG)  . I $$ADDSEG^HLOAPI(.HLMSTATE,.SEG,.ERROR) S NTECNT=NTECNT+1  ;  ; Recurrance information  I $G(HLOARR("RECUR"))'="" D  . D NTE("RECURRANCE INFORMATION","CMT",NTECNT,.SEG)  . I $$ADDSEG^HLOAPI(.HLMSTATE,.SEG,.ERROR) S NTECNT=NTECNT+1  . ;  . D NTE("Number of appointments to schedule: "\_$G(HLOARR("RECUR"))\_", Frequency: "\_$G(HLOARR("FREQUENCY")),"CMT",NTECNT,.SEG)  . I $$ADDSEG^HLOAPI(.HLMSTATE,.SEG,.ERROR) S NTECNT=NTECNT+1  ;  ; Add a blank line between comments  I (NTECNT>1),($G(HLOARR("COMMENT",0))>0) D  . D NTE(" ","CMT",NTECNT,.SEG)  . I $$ADDSEG^HLOAPI(.HLMSTATE,.SEG,.ERROR) S NTECNT=NTECNT+1  ;  F CMTCNT=1:1:$G(HLOARR("COMMENT",0)) D  . D NTE($G(HLOARR("COMMENT",CMTCNT)),"CMT",NTECNT,.SEG)  . I $$ADDSEG^HLOAPI(.HLMSTATE,.SEG,.ERROR) S NTECNT=NTECNT+1  ;  I $G(HLOARR("NO LATER THAN"))'="" D  . D NTE($G(HLOARR("NO LATER THAN")),"NLT",NTECNT,.SEG)  . I $$ADDSEG^HLOAPI(.HLMSTATE,.SEG,.ERROR) S NTECNT=NTECNT+1  ;  ; Build DG1 segment  F DIAGSEQ=1:1:$G(HLOARR("DIAGNOSIS",0)) D  . D DG1(.HLOARR,DIAGSEQ,.SEG)  . S OK=$$ADDSEG^HLOAPI(.HLMSTATE,.SEG,.ERROR)  ;  ; Send the message  S PARMS("SENDING APPLICATION")="OR-ORM-OUT"  S WHOTO("RECEIVING APPLICATION")="EPIC"  S WHOTO("FACILITY LINK NAME")="EPICORDIN"  ;  S SUCCESS=$$SENDONE^HLOAPI1(.HLMSTATE,.PARMS,.WHOTO,.ERROR)  Q $S(SUCCESS=0:-1,1:SUCCESS)  ; PV1(HLOARR,SEG) ; Build PV1 segment  ; Builds the PV1 segment using the HLO segment building APIs.  ;  ; HLOARR (I,REQ) - The array containing the order information  ; SEG (IO,REQ) - The segment, returned as a list of fields.  ;  N CLINIC  ;  K SEG S SEG="" ; The segment should start off blank.  ;  D SET^HLOAPI(.SEG,"PV1",0) ; Set the segment type.  ;  ; Set Clinic in into PV1-3 (format: <Clinic ID> <Clinic facility ID>^^^<Clinic facility ID>^^^^^^^<Assigning authority>)  S CLINIC=$G(HLOARR("CLINIC"))  D SET^HLOAPI(.SEG,$P(CLINIC,"^",1)\_" "\_$P(CLINIC,"^",2),3,1,1,1) ; Clinic ID  D SET^HLOAPI(.SEG,$P(CLINIC,"^",2),3,4,1,1) ; Clinic facility ID  D SET^HLOAPI(.SEG,$P(CLINIC,"^",4),3,11,1,1) ; Clinic assigning authority  Q  ; ORC(HLOARR,SEG) ; Build ORC segment  ; Builds the ORC segment using the HLO segment building APIs.  ;  ; HLOARR (I,REQ) - The array containing the order information  ; SEG (IO,REQ) - The segment, returned as a list of fields.  ;  K SEG S SEG="" ; The segment should start off blank.  ;  D SET^HLOAPI(.SEG,"ORC",0) ; Set the segment type.  ;  ; Set order control code into ORC-1  D SET^HLOAPI(.SEG,$G(HLOARR("ORDER CODE")),1)  ;  ; Set ORC-9  D SETDT^HLOAPI4(.SEG,$G(HLOARR("ORDER DATE")),9)  ;  ; Set ORC-16  D SET^HLOAPI(.SEG,$G(HLOARR("CANCEL REASON")),16)  Q  ; OBR(HLOARR,SEGCNT,SEG) ; Build OBR segment  ; Builds the OBR segment using the HLO segment building APIs.  ;  ; HLOARR (I,REQ) - The array containing the order information  ; SEGCNT (I,OPT) - Value for OBR-1. For the first occurrence of the segment  ; it should be set to 1, for the second 2, etc. If no value  ; is passed in, a default value of 1 is assigned.  ; SEG (IO,REQ) - The segment, returned as a list of fields.  ;  N ORDERABL,ORDERID,PROVIDER,TYPE  K SEG S SEG="" ; The segment should start off blank.  ;  D SET^HLOAPI(.SEG,"OBR",0) ; Set the segment type.  ;  ; Set OBR-1  I SEGCNT="" D SET^HLOAPI(.SEG,1,1)  E D SET^HLOAPI(.SEG,SEGCNT,1)  ;  ; Set order number in into OBR-2 (format: <ID>^<Application ID>)  S ORDERID=$G(HLOARR("ORDER ID"))  D SET^HLOAPI(.SEG,$P(ORDERID,"^",1),2,1,1,1) ; Order ID  D SET^HLOAPI(.SEG,$P(ORDERID,"^",2),2,2,1,1) ; Application ID  ;  ; Set orderable into OBR-4 (format: <ID>^<Name>^<Coding system name>)  S ORDERABL=$G(HLOARR("ORDERABLE"))  D SET^HLOAPI(.SEG,$P(ORDERABL,"^",1),4,1,1,1) ; Order ID  D SET^HLOAPI(.SEG,$P(ORDERABL,"^",2),4,2,1,1) ; Order name  D SET^HLOAPI(.SEG,$P(ORDERABL,"^",3),4,3,1,1) ; Order coding system name  ;  ; Set Provider in into OBR-16 (format: <ID>^<Last name>^<First name>^^^^^^<Assigning authority>^^^^<ID type>)  S PROVIDER=$G(HLOARR("ORDERING PHYSICIAN"))  S TYPE=$P(PROVIDER,"^",4)  D SET^HLOAPI(.SEG,$P(PROVIDER,"^",1),16,1,1,1) ; Provider ID  D SET^HLOAPI(.SEG,$P(PROVIDER,"^",2),16,2,1,1) ; Provider last name  D SET^HLOAPI(.SEG,$P(PROVIDER,"^",3),16,3,1,1) ; Provider first name  D SET^HLOAPI(.SEG,TYPE,16,9,1,1) ; Provider assigning authority  D SET^HLOAPI(.SEG,TYPE,16,13,1,1) ; Provider ID type  ;  ; Set in into OBR-27.4  D SETDT^HLOAPI4(.SEG,$G(HLOARR("START DATE")),27,4,1)  ;  ; Set in into OBR-27.5  D SETDT^HLOAPI4(.SEG,$G(HLOARR("STOP DATE")),27,5,1)  ;  ;Add in reccurency information (OBR-27.2, OBR-27.8, OBR-27.12)  I HLOARR("RECUR")'="" D  . D SET^HLOAPI(.SEG,$G(HLOARR("FREQUENCY")),27,2,1,1)  . D SET^HLOAPI(.SEG,"STANDING",27,8,1,1)  . D SET^HLOAPI(.SEG,$G(HLOARR("RECUR")),27,12,1,1)  Q  ; NTE(NTECMT,NTETYPE,SEGCNT,SEG) ; Builds NTE segment  ; Builds the NTE segment using the HLO segment building APIs.  ;  ; NTECMT (I,REQ) - Free text comment  ; NTETYPE (I,OPT) - The comment type  ; SEGCNT (I,REQ) - Value for NTE-1. For the first occurrence of the segment  ; it should be set to 1, for the second 2, etc.  ; SEG (IO,REQ) - The segment, returned as a list of fields.  ;  K SEG S SEG="" ; The segment should start off blank.  ;  D SET^HLOAPI(.SEG,"NTE",0) ; Set the segment type.  ;  ; Set NTE-1  D SET^HLOAPI(.SEG,SEGCNT,1)  ;  ; Set NTE-3  D SET^HLOAPI(.SEG,$G(NTECMT),3)  ;  ; Set NTE-4  D SET^HLOAPI(.SEG,$G(NTETYPE),4)  Q  ; DG1(HLOARR,SEGCNT,SEG) ; Build DG1 segment  ; Builds the DG1 segment using the HLO segment building APIs.  ;  ; HLOARR (I,REQ) - The array containing the diagnosis information  ; SEGCNT (I,REQ) - Value for DG1-1. For the first occurrence of the segment  ; it should be set to 1, for the second 2, etc.  ; SEG (IO,REQ) - The segment, returned as a list of fields.  ;  N DIAGNSIS  K SEG S SEG="" ;The segment should start off blank.  ;  D SET^HLOAPI(.SEG,"DG1",0) ;Set the segment type.  ;  ; Set DG1-1  D SET^HLOAPI(.SEG,SEGCNT,1)  ;  ; Set DG1-3 (format: <ID>^<Name>^<Code set>)  S DIAGNSIS=$G(HLOARR("DIAGNOSIS",SEGCNT))  D SET^HLOAPI(.SEG,$P(DIAGNSIS,"^",1),3,1,1,1) ; Diagnosis ID  D SET^HLOAPI(.SEG,$P(DIAGNSIS,"^",2),3,2,1,1) ; Diagnosis name  D SET^HLOAPI(.SEG,$P(DIAGNSIS,"^",3),3,3,1,1) ; Diagnosis code set  Q  ;** | | | | | | | | | |

Table : SDMXCANC Routine

| Routines | Activities | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Routine Name | SDMXCANC | | | | | | | | |
| Enhancement Category | New | | Modify | Delete | | | No Change | | |
| RTM |  | | | | | | | | |
| Related Options |  | | | | | | | | |
| Related Routines | Routines “Called By” | | | | | Routines “Called” | | | |
|  | SDMXSCHP | | | | | DIE  DIK  SDAM2  SDAMEVT  SDCNSLT  SDMXERRO  SDMXGAPT  XLFDT | | | |
| Routines | Activities | | | | | | | | |
| Data Dictionary (DD) References | ^DPT(  ^SC(  ^SD(409.2 | | | | | | | | |
| Related Protocols | N/A | | | | | | | | |
| Related Integration Control Registrations (ICRs) | N/A | | | | | | | | |
| Data Passing | Input | Output Reference | | | Both | | | Global Reference | Local |
| Input Attribute Name and Definition |  | | | | | | | | |
| Output Attribute Name and Definition |  | | | | | | | | |
| Current Logic | | | | | | | | | |
| N/A – New routine | | | | | | | | | |
| Modified Logic (Changes are in bold) | | | | | | | | | |
| SDMXCANC ; MASS/AWS,DAP - APPOINTMENT CANCEL API;08/22/017  ;;5.3;Scheduling;\*\*676\*\*;AUGUST 22,2017  ;;Per VA directive 6402, this routine should not be modified.  Q  CANCAPPT(PATIEN,CLINIEN,APTDTTM,USERID,CNREASON,CNNOTES) ; Cancel an appointment  ; This is the main entry point in this routine. Will only cancel appointments  ; with a status of "Scheduled." This tag was inspired by CANCEL^SDEC08.  ; PATIEN (I,REQ) - Patient IEN  ; CLINIEN (I,OPT) - Clinic IEN (will be looked up if not passed in)  ; APTDTTM (I,REQ) - Appointment time (in VistA format)  ; DUZ (I,REQ) - User who canceled the appointment  ; CNREASON (I,OPT) - Cancelation reason. Code is from ^SD(409.2)  ; CNNOTES (I,OPT) - Cancelation remarks  N ERRTXT,SUCCESS,CNTYPE  ; Check the required input parameters since these are required to cancel the appointment  I ($G(PATIEN)="")!($G(APTDTTM)="")!($G(USERID)="") D Q 0  . S ERRTXT="Missing required parameters"  . D ERRLOG^SDMXERRO(101,ERRTXT,0) ; Log an error but don't send a message over the interface  ; Get the Clinic ID if it hasn't been passed in.  I $G(CLINIEN)="" S CLINIEN=$$APTNODEP^SDMXGAPT($G(PATIEN),$G(APTDTTM),0,1)  I $G(CLINIEN)="" D Q 0  . S ERRTXT="Unable to cancel appointment. No clinic has the scheduled appointment."  . D ERRLOG^SDMXERRO(300,ERRTXT,1)  ; Default the cancel type if it hasn't been passed in  S CNTYPE=$P($G(^SD(409.2,CNREASON,0)),"^",2)  I (CNTYPE="B")!(CNTYPE="") S CNTYPE="C"  ; Validate parameters  S SUCCESS=""  S SUCCESS=$$VALDPARM($G(PATIEN),$G(CLINIEN),$G(APTDTTM),$G(CNTYPE),$G(USERID),$G(CNREASON),$G(CNNOTES))  I SUCCESS=0 Q 0 ; Error is logged in tag to support different error IDs  ; Validate appointment status  S SUCCESS=""  S SUCCESS=$$VALDAPT($G(PATIEN),$G(APTDTTM))  I SUCCESS'="" D ERRLOG^SDMXERRO(309,SUCCESS,1) Q 0  Q $$FILECANC($G(PATIEN),$G(CLINIEN),$G(APTDTTM),$G(CNTYPE),$G(USERID),$G(CNREASON),$G(CNNOTES))  FILECANC(PATIEN,CLINIEN,APTDTTM,CNTYPE,DUZ,CNREASON,CNNOTES) ; File the data to the database to cancel the appointment  ; This tag will file data to the database to cancel the appointment. We assume  ; the input parameters are all valid parameters that match to a valid, scheduled  ; appointment. We will not perform validation on the parameters.  ; PATIEN (I,REQ) - Patient IEN  ; CLINIEN (I,REQ) - Clinic IEN  ; APTDTTM (I,REQ) - Appointment time (in VistA format)  ; CNTYPE (I,REQ) - "PC" is patient-canceled, "C" if clinic canceled  ; DUZ (I,REQ) - User who canceled the appointment  ; CNREASON (I,OPT) - Cancelation reason. Code is from ^SD(409.2)  ; CNNOTES (I,OPT) - Cancelation remarks  ; Returns 1 if appointment was successfully canceled; 0 otherwise.  N SDDA,SDCPHDL,SDATA,SDFDA,DA,DIK,SDECIENS,DURATION,USER,DATE,ERRTXT,SUCCESS,AVAIL  ; Get the appointment IFN for the patient  S SDDA=$$FIND^SDAM2(PATIEN,APTDTTM,CLINIEN)  ; Get the event handler for the calls to BEFORE^SDAMEVT, AFTER^SDAMEVT, and EVT^SDAMEVT  S SDCPHDL=$$HANDLE^SDAMEVT(1)  S SDATA=SDDA\_"^"\_PATIEN\_"^"\_APTDTTM\_"^"\_CLINIEN  D BEFORE^SDAMEVT(.SDATA,PATIEN,APTDTTM,CLINIEN,SDDA,SDCPHDL)  ; Get the user who made appt and date appt made from ^SC  ; because data in ^SC will be deleted  S USER=$P($G(^SC(CLINIEN,"S",APTDTTM,1,SDDA,0)),"^",6)  S DATE=$P($G(^SC(CLINIEN,"S",APTDTTM,1,SDDA,0)),"^",7)  ; Update information in 2.98 file  S SDECIENS=APTDTTM\_","\_PATIEN\_","  S SDFDA(2.98,SDECIENS,"3")=CNTYPE  S SDFDA(2.98,SDECIENS,"14")=""  S SDFDA(2.98,SDECIENS,"15")=$$NOW^XLFDT()  I CNREASON'="" S SDFDA(2.98,SDECIENS,"16")=CNREASON  I CNNOTES'="" S SDFDA(2.98,SDECIENS,"17")=$E(CNNOTES,1,160) ; Truncate to 160 chars  S SDFDA(2.98,SDECIENS,"19")=USER  S SDFDA(2.98,SDECIENS,"20")=DATE  ; Update the database  D FILE^DIE("","SDFDA","DIEFOUT")  ; Update ^GMR and ^OR globals if appointment was scheduled on an order or consult.  S SUCCESS=""  I +$D(^SC(CLINIEN,"S",APTDTTM,1,SDDA,"CONS"))=1 S SUCCESS=$$UPCONREQ(CLINIEN,APTDTTM,CNTYPE,SDDA,DUZ,CNNOTES)  I SUCCESS'="" D ERRLOG^SDMXERRO(207,SUCCESS,1) Q 0  ; Grab the duration of the appointment before deleting the data in the ^SC global  S DURATION=$P($G(^SC(CLINIEN,"S",APTDTTM,1,SDDA,0)),"^",2)  ; Delete data in ^SC global  S DIK="^SC("\_CLINIEN\_",""S"","\_APTDTTM\_",1,"  S DA(2)=CLINIEN  S DA(1)=APTDTTM  S DA=SDDA  D ^DIK  D AFTER^SDAMEVT(.SDATA,PATIEN,APTDTTM,CLINIEN,SDDA,SDCPHDL)  ; Call the event driver to send an HL7 message  D EVT^SDAMEVT(.SDATA,2,0,SDCPHDL) ; SDAMEVT=2, SDMODE=0  Q 1  ;  VALDPARM(PATIEN,CLINIEN,APTDTTM,CNTYPE,DUZ,CNREASON,CNNOTES) ; Validate parameters  ; Determine if we can cancel the appointment based on the parameters passed  ; into the cancel API.  ; PATIEN (I,REQ) - Patient IEN  ; CLINIEN (I,OPT) - Clinic IEN (will be looked up if not passed in)  ; APTDTTM (I,REQ) - Appointment time (in VistA format)  ; CNTYPE (I,REQ) - "PC" is patient-canceled, "C" if clinic canceled  ; DUZ (I,REQ) - User who canceled the appointment  ; CNREASON (I,OPT) - Cancelation reason. Code is from ^SD(409.2)  ; CNNOTES (I,OPT) - Cancelation remarks  ; Returns 1 if we can cancel the appointment; 0 otherwise.  N TYPE,ERRTXT  ; Validate patient  I '$D(^DPT(PATIEN)) D Q 0  . S ERRTXT="Unable to cancel appointment. Invalid patient."  . D ERRLOG^SDMXERRO(204,ERRTXT,1)  I '$D(^SC(CLINIEN)) D Q 0  . S ERRTXT="Unable to cancel appointment. Invalid clinic."  . D ERRLOG^SDMXERRO(300,ERRTXT,1)  I (CNREASON'=""),('$D(^SD(409.2,CNREASON))) D Q 0  . S ERRTXT="Unable to cancel appointment. Invalid cancelation reason."  . D ERRLOG^SDMXERRO(302,ERRTXT,1)  Q 1  ;  VALDAPT(PATIEN,APTDTTM) ; Validate appointment status  ; Determine if we can cancel the appointment based on the status of the  ; appointment. Only an appointment with a status of "Scheduled" is considered  ; valid to cancel.  ; PATIEN - Patient IEN  ; APTDTTM - Appointment time  ; Returns "" if we can cancel the appointment; otherwise, the reason the  ; appointment cannot be canceled will be returned.  N APTSTS  S APTSTS=$$APTSTAT^SDMXGAPT($G(PATIEN),$G(APTDTTM))  ; Does the appointment even exist?  I APTSTS="" Q "Appointment does not exist"  ; Have we already canceled the appointment?  I APTSTS="CANCELLED" Q "Appointment already cancelled"  ; Has the appointment already been checked in?  I APTSTS["CHECKED IN" Q "Appointment already checked-in"  ; Has the appointment already been checked out?  I APTSTS["CHECKED OUT" Q "Appointment already checked-out"  Q ""  ;  UPCONREQ(CLINIEN,APTDTTM,CNTYPE,SDDA,DUZ,CNNOTES) ; Update consult request  ; Update the status of the consult to no longer be "Scheduled" if the  ; appointment with the linked consult was canceled. This will update the status  ; of the consult to be "Active."  ; CLINIEN (I,OPT) - Clinic IEN (will be looked up if not passed in)  ; APTDTTM (I,REQ) - Appointment time (in VistA format)  ; CNTYPE (I,REQ) - "PC" is patient-canceled, "C" if clinic canceled  ; SDDA (I,REQ) - Appointment IFN  ; DUZ (I,REQ) - User who canceled the appointment  ; CNNOTES (I,OPT) - Cancelation remarks  ; Returns "" if ^GRM and ^OR globals were sucessfully updated; otherwise,  ; the reason the consult request wasn't updated will be returned.  N SDSC,SDTTM,SDWH,SDPL,TMPD,SDADM,SDERR  ; Set assumed variables for call to CANCEL^SDCNSLT  S SDSC=CLINIEN  S SDTTM=APTDTTM  S SDWH=CNTYPE  S SDPL=SDDA  S SDADM=$P($G(^SC(CLINIEN,"S",APTDTTM,1,SDDA,0)),"^",7)  S TMPD=CNNOTES  ; Update the ^GMR global with the canceled appointment status  D CANCEL^SDCNSLT  I $G(SDERR)'["0" Q $P($G(SDERR),"^",2) ; Return reason consult request wasn't updated  Q "" | | | | | | | | | |

Table : SDMXCHKI Routine

| Routines | Activities | | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Routine Name | SDMXCHKI | | | | | | | | | |
| Enhancement Category | New | | Modify | Delete | | | No Change | | | |
| RTM |  | | | | | | | | | |
| Related Options |  | | | | | | | | | |
| Related Routines | Routines “Called By” | | | | | Routines “Called” | | | | |
|  | SDMXSCHP | | | | | %DTC  DIE  DILF  SDAM2  SDAMEVT | | | | |
| Routines | Activities | | | | | | | | | |
| Data Dictionary (DD) References | ^SC( | | | | | | | | | |
| Related Protocols | N/A | | | | | | | | | |
| Related Integration Control Registrations (ICRs) | N/A | | | | | | | | | |
| Data Passing | Input | Output Reference | | | Both | | | Global Reference | Local |
| Input Attribute Name and Definition |  | | | | | | | | | |
| Output Attribute Name and Definition |  | | | | | | | | | |
| Current Logic | | | | | | | | | | |
| N/A – New routine | | | | | | | | | | |
| Modified Logic (Changes are in bold) | | | | | | | | | | |
| **SDMXCHKI ;MASS/BB - Epic Appointment Checkin API;8/17/17**  **;;5.3;Scheduling;\*\*676\*\*;AUGUST 22,2017**  **;;Per VA directive 6402, this routine should not be modified.**  **Q**  **CHKIN(PATID,CLINID,APPTDT,CHECKINDT,USERDUZ) ;Check-in Appointment API**  **; SCOPE: PUBLIC**  **; DESCRIPTION: Public API to check-in an appointment in VistA**  **; PARAMETERS:**  **; PATID (I,REQ) - Patient IEN (internal VistA ID)**  **; CLINID (I,REQ) - Clinic IEN (internal VistA ID**  **; APPTDT (I,REQ) - Date and time of appointment in vista format**  **; CHECKINDT (I,REQ) - Date and time of the check in action**  **; USERDUZ (I,REQ) - Data Entry Clerk**  **; RETURNS: success/failure array - see CHK^XMASSCHKIN for details**  **N RET**  **D CHK(.RET,$$FMTSTR(PATID,CLINID,APPTDT),CHECKINDT,USERDUZ)**  **Q $G(RET) ;**  **CCHKIN(PATID,CLINID,APPTDT) ;Kills the Check-in node of the clinic global, reverting status to "no action yet".**  **; PATID (I,REQ) - Patient IEN (internal VistA ID)**  **; CLINID (I,REQ) - Clinic IEN (internal VistA ID**  **; APPTDT (I,REQ) - Date and time of appointment in vista format**  **N APPTIEN**  **S APPTIEN=$$FIND^SDAM2(PATID,APPTDT,CLINID)**  **K ^SC(CLINID,"S",APPTDT,1,APPTIEN,"C")**  **Q**  **FMTSTR(PATID,CLINID,APPTDT) ;Performs string concatenation**  **; PATID(I,REQ) - patient IEN (internal VistA ID)**  **; CLINID (I,REQ) - clinic IEN (internal VistA ID**  **; APPTDT (I,REQ) - date and time of appointment in VistA format**  **Q PATID\_"-"\_CLINID\_"-"\_APPTDT\_";"**  **CHK(VPSRES,VPSAPPT,CHECKINDT,USERDUZ) ;Set-up routine for check-in for an appointment in VistA. Can take multiple appointments at once,**  **; but we might not need that - currently only doing one at a time.**  **; VPSRES (O,OPT) - Output array containing the success/failure state of attempted check-ins.**  **; RETurn value = 1 if check-in successful or '99' if appointment was not checked in**  **; Syntax:**  **; DFN\_"-"\_clinicIEN\_"-"\_date/timestamp of appt\_"-"\_RETurn value\_";"**  **; Example of data output (example represents the result of 2 checked in appts):**  **; VPSRES(0)="308165-1218-3120420.1215-1**  **; VPSRES(1)="308165-4569-3120420.1030-99"**  **; VPSAPPT (I,REQ) - Input string containing appointment details to be checked in. Format:**  **; DFN\_"-"\_clinic IEN\_"-"\_date/timestamp of scheduled appt\_";"**  **; USERDUZ (I,REQ) - Data Entry Clerk**  **N VPSCIEN,VPSI,DFN,VPSDT,VPSCLIN,RESULT,VPSREC,U**  **S U="^"**  **I '+$G(VPSAPPT) S VPSRES(0)="---99-appt record not sent" Q**  **F VPSI=1:1 S VPSREC=$P(VPSAPPT,";",VPSI) Q:VPSREC']"" D**  **. S DFN=$P(VPSREC,"-")**  **. S VPSCLIN=$P(VPSREC,"-",2)**  **. S VPSDT=$P(VPSREC,"-",3)**  **. I '+DFN S VPSRES(VPSI)=VPSREC\_"-99-patient DFN not sent" Q**  **. I '+VPSDT S VPSRES(VPSI)=VPSREC\_"-99-date/timestamp not sent" Q**  **. I '+VPSCLIN S VPSRES(VPSI)=VPSREC\_"-99-clinic identifier not sent" Q**  **. D DT^DILF("T",VPSDT,.VPSDT)**  **. S VPSCIEN=$$FIND^SDAM2(DFN,VPSDT,VPSCLIN)**  **. I +VPSCIEN'>0 S VPSRES(VPSI)=VPSREC\_"-99-Appt not found." Q**  **. D HDLKILL^SDAMEVT ;CLEAR PRE-EXISTING HANDLES**  **. N SDATA,SDCIHDL S SDATA=VPSCIEN\_U\_DFN\_U\_VPSDT\_U\_VPSCLIN,SDCIHDL=$$HANDLE^SDAMEVT(1) ;CALL TO EVENT HANDLER**  **. D BEFORE^SDAMEVT(.SDATA,DFN,VPSDT,VPSCLIN,VPSCIEN,SDCIHDL) ;CAPTURE CURRENT APT DATA IN ^TMP("SDAMEVT",$J**  **. S RESULT=$$CHECKIN(VPSCLIN,VPSDT,VPSCIEN,CHECKINDT,USERDUZ)**  **. D AFTER^SDAMEVT(.SDATA,DFN,VPSDT,VPSCLIN,VPSCIEN,SDCIHDL) ;CAPTURE CHECK-IN DATA IN ^TMP("SDAMEVT",$J**  **. D EVT^SDAMEVT(.SDATA,4,1,SDCIHDL) ; 4 := CI EVT , 1:= COMPUTER MONLOGUE ;CALL EVT HANDLER**  **. D HDLKILL^SDAMEVT ;CLEAR HANDLES**  **. S VPSRES(VPSI)=VPSREC\_"-"\_RESULT**  **Q**  **CHECKIN(CLIN,DTM,CIEN,CHECKINDT,USERDUZ) ;Update appropriate fields for check-in (HOSPITAL LOCATION file(#44). Actually checks patient in.**  **; CLIN (I,REQ) - clinic IEN**  **; DTM (I,REQ) - VistA date/time**  **; CIEN (I,REQ) - "Contact" (appointment) entry to check in**  **; USERDUZ (I,REQ) - Data Entry Clerk**  **N VPSFDA,VERR,VPSRET,USERFILE**  **N %,VPSNOW D NOW^%DTC S VPSNOW=%**  **S VPSFDA(44.003,CIEN\_","\_DTM\_","\_CLIN\_",",309)=CHECKINDT ;PATIENT MULTIPLE/APPOINTMENT MULTIPLE OF HOSPITAL LOCATION FILE**  **D FILE^DIE("","VPSFDA","VERR")**  **I $D(VERR) S VPSRET="99-APPT COULD NOT BE CHECKED IN"**  **S USERFILE(44.003,CIEN\_","\_DTM\_","\_CLIN\_",",302)=USERDUZ ;File Check In DUZ**  **D FILE^DIE("","USERFILE","VERR")**  **E S VPSRET=1**  **K VPSFDA,VERR**  **Q VPSRET** | | | | | | | | | | |

Table : SDMXCHKO Routine

| Routines | Activities | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Routine Name | SDMXCHKO | | | | | | | | |
| Enhancement Category | New | | Modify | Delete | | | No Change | | |
| RTM |  | | | | | | | | |
| Related Options |  | | | | | | | | |
| Related Routines | Routines “Called By” | | | | | Routines “Called” | | | |
|  | SDMXSCHP | | | | | DIE  SDAM2  SDAPI  SDAPIAP  SDAPIER  SDCODEL  SDMXERRO  SDMXGAPT | | | |
| Routines | Activities | | | | | | | | |
| Data Dictionary (DD) References | N/A | | | | | | | | |
| Related Protocols | N/A | | | | | | | | |
| Related Integration Control Registrations (ICRs) | N/A | | | | | | | | |
| Data Passing | Input | Output Reference | | | Both | | | Global Reference | Local |
| Input Attribute Name and Definition |  | | | | | | | | |
| Output Attribute Name and Definition |  | | | | | | | | |
| Current Logic | | | | | | | | | |
| N/A – New routine | | | | | | | | | |
| Modified Logic (Changes are in bold) | | | | | | | | | |
| **SDMXCHKO ;MASS/JMM - APPOINTMENT CHECKOUT API;8/24/17**  **;;5.3;Scheduling;\*\*676\*\*;AUGUST 22,2017**  **;;Per VA directive 6402, this routine should not be modified.**  **Q**  **CHKOUT(PATIENTIEN,CLINICIEN,APPTDT,CHKOUTDT,USERDUZ) ;Check out appointment API**  **; PATIENTIEN (I,REQ) - Patient IEN**  **; CLINICIEN (I,REQ) - Clinic IEN**  **; APPTDT (I,REQ) - appointment date/time in VA DT dormat**  **; CHKOUTDT (I,REQ) - check out date/time in VA DT format**  **; USERDUZ (I,REQ) - user ID**  **N RESULT,SDUZ,SDMODE,SDEVENT,SDROOT,SDMODE,SDRET,ENCIEN,ERRCOUNT,USERFILE,VERR,CTRL,VPSCIEN,SDERROOT,SDVIEN,U**  **;Validate Inputs**  **I $G(PATIENTIEN)="" Q 0 ;Quit if no patient**  **I $G(CLINICIEN)="" Q 0 ;Quit if no clinic**  **I $G(APPTDT)="" Q 0 ;Quit if no appt d/t**  **I $G(CHKOUTDT)="" Q 0 ;Quit if no checkout d/t**  **;Setup Event and Root Arrays**  **S SDERROOT=""**  **S SDVIEN=""**  **S SDEVENT=$NA(^XTMP("MASSAPPT"))**  **S SDROOT=$NA(^XTMP("MASSAPPT"))**  **S @SDEVENT@("EVENT")="CHECK-OUT"**  **S @SDROOT@("EVENT")="CHECK-OUT"**  **S @SDROOT@("DATE/TIME")=$G(CHKOUTDT)**  **S @SDROOT@("USER")=$G(USERDUZ) ;DUZ should always exist, but check anyways**  **S U="^"**  **S SDMODE=0**  **D INIT^SDAPI(PATIENTIEN,APPTDT,CLINICIEN,.SDEVENT,.SDROOT,.SDMODE,.SDRET,.SDUZ)**  **I $$ERRCHK^SDAPIER() D Q 0 ;Communicate internally-logged vista errors up to MASS**  **. S ERRCOUNT=$G(@SDERROOT@("ERROR"))**  **. F CTRL=1:1:ERRCOUNT D**  **. . I $P($G(@SDERROOT@("ERROR",CTRL)),"^",1)=1 D ERRLOG^SDMXERRO(201,"No event data",1)**  **. . I $P($G(@SDERROOT@("ERROR",CTRL)),"^",1)=2 D ERRLOG^SDMXERRO(204,"Invalid Patient ID",1)**  **. . I $P($G(@SDERROOT@("ERROR",CTRL)),"^",1)=3 D ERRLOG^SDMXERRO(308,"Invalid User ID",1)**  **. . I $P($G(@SDERROOT@("ERROR",CTRL)),"^",1)=4 D ERRLOG^SDMXERRO(300,"Invalid Clinic ID",1)**  **. . I $P($G(@SDERROOT@("ERROR",CTRL)),"^",1)=5 D ERRLOG^SDMXERRO(305,"Invalid Encounter ID",1)**  **. . E D ERRLOG^SDMXERRO(207,$G(@SDERROOT@("ERROR",CTRL)),1)**  **S SDMODE=0 ;0-Non-interactive, 1-Interactive**  **S ENCIEN=$$EN^SDAPIAP(PATIENTIEN,APPTDT,CLINICIEN,SDUZ,SDMODE,.SDVIEN)**  **I $$ERRCHK^SDAPIER() D Q 0 ;Communicate internally-logged vista errors up to MASS**  **. S ERRCOUNT=$G(@SDERROOT@("ERROR"))**  **. F CTRL=1:1:ERRCOUNT D**  **. . I $P($G(@SDERROOT@("ERROR",CTRL)),"^",1)=100!$P($G(@SDERROOT@("ERROR",CTRL)),"^",1)=102 D ERRLOG^SDMXERRO(307,"Invalid Checkout D/T",1)**  **. . I $P($G(@SDERROOT@("ERROR",CTRL)),"^",1)=101 D ERRLOG^SDMXERRO(204,"Invalid Patient ID",1)**  **. . I $P($G(@SDERROOT@("ERROR",CTRL)),"^",1)=103 D ERRLOG^SDMXERRO(309,"Current status won't allow checking-out",1)**  **. . I $P($G(@SDERROOT@("ERROR",CTRL)),"^",1)=104 D ERRLOG^SDMXERRO(307,"Appointment date is after today",1)**  **. . E D ERRLOG^SDMXERRO(207,$G(@SDERROOT@("ERROR",CTRL)),1)**  **S RESULT=$$FINAL^SDAPI(ENCIEN)**  **S VPSCIEN=$$FIND^SDAM2(PATIENTIEN,APPTDT,CLINICIEN)**  **S USERFILE(44.003,VPSCIEN\_","\_APPTDT\_","\_CLINICIEN\_",",304)=USERDUZ ;File Check Out DUZ**  **D FILE^DIE("","USERFILE","VERR")**  **Q RESULT ;Returns encounter IEN and status**  **CHKODEL(PATIENTIEN,APPTDT) ;Undo a checked-out appointment**  **; PATIENTIEN (I,REQ) - patient ien**  **; APPTDT (I,REQ) - vista appointment date/time**  **N ENCOUNTERID**  **;Validate Inputs**  **I $G(PATIENTIEN)="" Q ;Quit if no patient**  **I $G(APPTDT)="" Q ;Quit if no appt d/t**  **S ENCOUNTERID=$$APTNODEP^SDMXGAPT(PATIENTIEN,APPTDT,0,20)**  **I $G(ENCOUNTERID)="" D ERRLOG^SDMXERRO(305,"No encounter found",1) Q**  **D EN^SDCODEL(ENCOUNTERID,0,1) ;**  **Q** | | | | | | | | | |

Table : SDMXCORE Routine

| Routines | Activities | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Routine Name | SDMXCORE | | | | | | | | |
| Enhancement Category | New | | Modify | Delete | | | No Change | | |
| RTM |  | | | | | | | | |
| Related Options |  | | | | | | | | |
| Related Routines | Routines “Called By” | | | | | Routines “Called” | | | |
|  | DGMXHL7  DGMXVLD  ORMXCONV  ORMXFMT  ORMXTR  SDMXMAKE  SDMXSCHI  SDMXSCHP  SDMXSCHT | | | | | %ZTER  DIQ  SDMXERRO  SDMXGAPT  XLFDT  XLFSTR | | | |
| Routines | Activities | | | | | | | | |
| Data Dictionary (DD) References | ^DPT(  ^OR(100  ^SC(  ^SD(409.2  ^SD(409.67  ^VA(200 | | | | | | | | |
| Related Protocols | N/A | | | | | | | | |
| Related Integration Control Registrations (ICRs) | N/A | | | | | | | | |
| Data Passing | Input | Output Reference | | | Both | | | Global Reference | Local |
| Input Attribute Name and Definition |  | | | | | | | | |
| Output Attribute Name and Definition |  | | | | | | | | |
| Current Logic | | | | | | | | | |
| N/A – New routine | | | | | | | | | |
| Modified Logic (Changes are in bold) | | | | | | | | | |
| **SDMXCORE ;MASS/RPC,DAP- Core Tags;8/22/17**  **;;5.3;Scheduling;\*\*676\*\*;AUGUST 22,2017**  **;;Per VA directive 6402, this routine should not be modified.**  **HL72VATS(HL7TS) ; Converts HL7 formatted timestamps to VA format**  **; HL7TS - date/time stamp in 24H HL7 format (YYYYMMDDHHMMSS)**  **Q $$HL7TFM^XLFDT($G(HL7TS))**  **VA2HL7TS(VATS) ; Converts VA formatted timestamps to HL7 format**  **; VATS - date/time stamp in VA format (YYYMMDD.HHMMSS)**  **Q $$FMTHL7^XLFDT($G(VATS))**  **GETPTIEN(PATNAME) ; Returns patient ID or null, name must be perfect match**  **; PATNAME - Patient name - must be exact LAST,FIRST**  **N IEN**  **I $G(PATNAME)="" Q ""**  **I $D(^DPT("B",PATNAME)) D**  **. S IEN=$O(^DPT("B",PATNAME,""))**  **Q $G(IEN)**  **GETPTNM(DFN) ; Returns patient name from ^DPT global, given a valid DFN**  **; DFN - Patient ID to look for**  **N PATNAME**  **I $G(DFN)="" Q ""**  **I $D(^DPT(DFN,0)) D**  **. S PATNAME=$P(^DPT(DFN,0),"^",1)**  **Q $G(PATNAME)**  **GETLCIEN(LOCNAME) ; Returns Location ID or null, name must be perfect match**  **; PATNAME - Location name - must be exact**  **N IEN**  **I $G(LOCNAME)="" Q ""**  **I $D(^SC("B",LOCNAME)) D**  **. S IEN=$O(^SC("B",LOCNAME,""))**  **Q $G(IEN)**  **GETLCNM(LOCID) ; Returns location name from clinic file 44 given a valid clinic IEN**  **; LOCID - Location ID to look for**  **Q $$GET1^DIQ(44,$G(LOCID),.01)**  **GETNMPRV(CLINIC) ; Returns the number of providers associated with a clinic**  **; CLINIC - The Clinic IEN (first piece of DPT 0 node)**  **Q $P(^SC($G(CLINIC),"PR",0),"^",4) ;Piece 3 is most recently assigned number, piece 4 is total active.**  **GETCNGNM(CLINICGROUP) ; Returns the Name of a Clinic's group**  **; CLINICGROUP - The Clinic Group IEN**  **Q $P(^SD(409.67,$G(CLINICGROUP),0),"^",1)**  **GETPRVNM(PROVIEN) ; Returns the provider name, given a provider ID**  **; PROVIEN - The Provider IEN**  **Q $P(^VA(200,$G(PROVIEN),0),"^")**  **ICLNDPRV(CLINIC,PROVIEN) ; Determines if the provider is the default provider for the clinic**  **; CLINIC - The Clinic IEN (first piece of DPT 0 node)**  **; PROVIEN - The Provider IEN**  **Q $P(^SC($G(CLINIC),"PR",$$CLNPVIND($G(CLINIC),$G(PROVIEN)),0),"^",2)**  **CLNPVIND(CLINIC,PROVIEN) ; Determines the line number the provider is listed on for a clinic**  **; CLINIC - The Clinic IEN (first piece of DPT 0 node)**  **; PROVIEN - The Provider IEN**  **Q +$QS($Q(^SC($G(CLINIC),"PR","B",$G(PROVIEN))),5)**  **GTCANRSN(PATIENTIEN,APPTDT) ; Returns the discrete cancellation reason**  **; PATIEN (I,REQ)- Patient ID as in DPT(PATIEN,"S",APPTDAT**  **; APPTDAT (I,REQ) - Appointment date**  **Q $P(^SD(409.2,$$APTNODEP^SDMXGAPT($G(PATIENTIEN),$G(APPTDT),0,15),0),"^",1)**  **GTCNRNTP(PATIENTIEN,APPTDT) ; Gets the cancelation reason type.**  **; PATIEN - Patient ID as in DPT(PATIEN,"S",APPTDAT**  **; APPTDAT - Appointment date**  **N VAL,CANTYPE**  **S CANTYPE=$$APTNODEP^SDMXGAPT($G(PATIENTIEN),$G(APPTDT),0,15)**  **S VAL=$P($G(^SD(409.2,$G(CANTYPE),0)),"^",2)**  **Q $S($G(VAL)="B":"C",1:$G(VAL))**  **ORD2CONS(ORDERID) ;Returns the consult ID linked to the given order**  **; ORDERID - Order ID**  **N CNSLTLNK**  **I $G(ORDERID)="" Q ""**  **S CNSLTLNK=$G(^OR(100,ORDERID,4))**  **I $P(CNSLTLNK,";",2)="GMRC" Q $P(CNSLTLNK,";",1)**  **Q ""**  **INSTRING(VALUE,LIST,DELIM) ; compare a string value to see if it is a list given a particular delimiter**  **; VALUE - value to find in the list.**  **; LIST - The list to check**  **; DELIM - Delimiter that separates the data in the list. Default = ","**  **Q $S($G(DELIM)="":(","\_$G(LIST)\_",")[(","\_$G(VALUE)\_","),1:($G(DELIM)\_$G(LIST)\_$G(DELIM))[($G(DELIM)\_$G(VALUE)\_$G(DELIM)))**  **INITINC ; Sets temp global that indicates this process is filing an incoming message**  **S ^TMP($J,"INCINTF")=1**  **Q**  **DONEINC ; Clears temp global that indicates this process is filing an incoming message**  **K ^TMP($J,"INCINTF")**  **Q**  **INCINTF() ; Checks temp global that indicates whether the process is filing an incoming message**  **Q +$G(^TMP($J,"INCINTF"))**  **SETMSGET() ;SEND AN ERROR MESSAGE OUT AND LOG THE CACHE ERROR+STACK TO ^ERRORS**  **N $ETRAP**  **S $ETRAP="LOGSEND^SDMXCORE"**  **Q**  **LOGSEND() ;**  **S $ETRAP="Q:$QUIT """" Q"**  **D ERRLOG^SDMXERRO(-1,"Cache Error: "\_$ZE,1)**  **D ^%ZTER**  **Q**  **;**  **FMTPHONE(PHONE,EXT) ; Formats a VistA telephone number into an HL7-compliant format**  **; Formats include: (nnn)nnn-nnnn and nnn-nnnn, depending on whether or not there is an area code.**  **; If the number is not in an a valid format, does not attempt to do any formatting.**  **; Returns 1 if the number was formatted, 0 otherwise.**  **;**  **; PHONE - Phone number to be formatted**  **; EXT - Phone number extension (if specified)**  **;**  **I $G(PHONE)="" Q 0**  **N TEMP,LENGTH**  **;**  **; Extract phone number**  **S TEMP=$$STRIP^XLFSTR(PHONE,"-()") ; Strip certain delimiters**  **S TEMP=$TR(TEMP,"x","X") ; Standardize extension delimiter**  **S EXT=$P(TEMP,"X",2) ; Pull out the extension (if it exists)**  **S TEMP=$P(TEMP,"X",1)**  **;**  **; Format based on length**  **S LENGTH=$L(TEMP)**  **I '$$INSTRING^SDMXCORE(LENGTH,"7,10",",") Q 0 ; Length not 7 or 10**  **I LENGTH=7 S TEMP=$E(TEMP,1,3)\_"-"\_$E(TEMP,4,7) ; No area code: nnn-nnnn**  **I LENGTH=10 S TEMP="("\_$E(TEMP,1,3)\_")"\_$E(TEMP,4,6)\_"-"\_$E(TEMP,7,10) ; Area code: (nnn)nnn-nnnn**  **;**  **; Save output**  **S PHONE=TEMP**  **Q 1** | | | | | | | | | |

Table : SDMXERRO Routine

| Routines | Activities | | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Routine Name | SDMXERRO | | | | | | | | | |
| Enhancement Category | New | | Modify | Delete | | | No Change | | | |
| RTM |  | | | | | | | | | |
| Related Options |  | | | | | | | | | |
| Related Routines | Routines “Called By” | | | | | Routines “Called” | | | | |
|  | DGMXVLD  ORMXCONV  ORMXTR  SDMXCANC  SDMXCHKO  SDMXCORE  SDMXSCHI | | | | | HLOAPI  HLOAPI1 | | | | |
| Routines | Activities | | | | | | | | | |
| Data Dictionary (DD) References | N/A | | | | | | | | | |
| Related Protocols | N/A | | | | | | | | | |
| Related Integration Control Registrations (ICRs) | N/A | | | | | | | | | |
| Data Passing | Input | Output Reference | | | Both | | | Global Reference | Local |
| Input Attribute Name and Definition |  | | | | | | | | | |
| Output Attribute Name and Definition |  | | | | | | | | | |
| Current Logic | | | | | | | | | | |
| N/A – New routine | | | | | | | | | | |
| Modified Logic (Changes are in bold) | | | | | | | | | | |
| **SDMXERRO ;MASS/MJK - Outgoing Error Interface;08/17/2017**  **;;5.3;Scheduling;\*\*676\*\*;AUGUST 17,2017**  **;;Per VA directive 6402, this routine should not be modified.**  **Q**  **ERRLOG(ERRORID,ERRORTEXT,SENDMSG) ;LOG AN ERROR TO ^XERROR AND OPTIONALLY TRIGGER ERROR INTERFACE**  **N OK**  **I ($G(ERRORID)'="")!($G(SENDMSG)'="")**  **I $G(SENDMSG) S OK=$$SENDERR("","","",$G(ERRORID),$G(ERRORTEXT))**  **Q**  **;**  **SENDERR(SEGID,SEQUENCE,FIELDPOS,ERRCODE,ERRTEXT,ERROR) ;TRIGGER ERROR INTERFACE**  **;**  **; FROM HLO processing code.**  **; HDR: For incoming interfaces to VistA that process an error through SENDERR the HDR variable will hold the header information**  **; for the message. This information can be used to get the source message to return on the error.**  **;**  **N PARMS,MSG,ERROR,SEG,WHOTO**  **;**  **S PARMS("MESSAGE TYPE")="ACK"**  **S PARMS("EVENT")="NAK"**  **;**  **I '$$NEWMSG^HLOAPI(.PARMS,.MSG,.ERROR) S ERROR="COULD NOT INITIALIZE MESSAGE" Q 0**  **;**  **;MSA SEGMENT**  **D SET^HLOAPI(.SEG,"MSA",0)**  **D SET^HLOAPI(.SEG,$G(HDR("MESSAGE CONTROL ID")),2) ;MSA-2**  **I '$$ADDSEG^HLOAPI(.MSG,.SEG,.ERROR) S ERROR="COULD NOT ADD MSA SEGMENT" Q 0**  **;**  **;ERR SEGMENT**  **D SET^HLOAPI(.SEG,"ERR",0) ;ERR-0**  **D SET^HLOAPI(.SEG,$G(SEGID),1,1) ;ERR-1.1**  **D SET^HLOAPI(.SEG,$G(SEQUENCE),1,2) ;ERR-1.2**  **D SET^HLOAPI(.SEG,$G(FIELDPOS),1,3) ;ERR-1.3**  **D SET^HLOAPI(.SEG,$G(ERRCODE),1,4,1) ;ERR-1.4.1**  **D SET^HLOAPI(.SEG,$G(ERRTEXT),1,4,2) ;ERR-1.4.2**  **D SET^HLOAPI(.SEG,"HL70357",1,4,3) ;ERR-1.4.3**  **I '$$ADDSEG^HLOAPI(.MSG,.SEG,.ERROR) S ERROR="COULD NOT ADD ERR SEGMENT" Q 0**  **;**  **S PARMS("SENDING APPLICATION")="VISTA"**  **S WHOTO("RECEIVING APPLICATION")="MASS"**  **S WHOTO("FACILITY LINK NAME")="EPICERRIN"**  **;**  **Q $$SENDONE^HLOAPI1(.MSG,.PARMS,.WHOTO,.ERROR)** | | | | | | | | | | |

Table : SDMXFLAG Routine

| Routines | Activities | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Routine Name | SDMXFLAG | | | | | | | |
| Enhancement Category | New | | Modify | Delete | | No Change | | |
| RTM |  | | | | | | | |
| Related Options | View/Modify MASS Clinic Flag [SDMXFL] | | | | | | | |
| Related Routines | Routines “Called By” | | | | Routines “Called” | | | |
|  | None | | | | DIC  DICN  DIQ  DIR  VALM1  XPAR | | | |
| Routines | Activities | | | | | | | |
| Data Dictionary (DD) References | ^SC( | | | | | | | |
| Related Protocols | N/A | | | | | | | |
| Related Integration Control Registrations (ICRs) | N/A | | | | | | | |
| Data Passing | Input | Output Reference | | | Both | | Global Reference | Local | |
| Input Attribute Name and Definition |  | | | | | | | |
| Output Attribute Name and Definition |  | | | | | | | |
| Current Logic | | | | | | | | |
| N/A – New routine | | | | | | | | |
| Modified Logic (Changes are in bold) | | | | | | | | |
| **SDMXFLAG ;BP/JEO - Clinic Flag modify(cont) ;SEP 23, 2017  ;;5.3;Scheduling;\*\*676\*\*;AUGUST 22,2017  ;;Per VA directive 6402, this routine should not be modified.  ;  ; input:  ; SDCL := clinic#  ; S:'$D(^SC(SDCL,"S","MX"))  ; IND N DIC,X,Y,TIUAPDT,CLID  K DIC  S DIC=44,DIC(0)="AEMO"  S DIC("A")="Select CLINIC:// "  D ^DIC K DIC("S") S:+Y>0 CLID=+Y  I $G(CLID)="" G ASKCLIN  D DISPLAY(CLID)  G IND  Q  ; ASKCLIN ;  N DIR,X,Y,LINE  W #,!?6,"MASS CLINIC FLAG UTILITY"  W !!,"Now you'll select the clinics."  S DIR(0)="SO^A:Show Flag for All Clinics;I:Show/Modify Flag for an Individual Clinic;"  S DIR("A")="Select Clinics by"  S DIR("?",1)="Show/Modify MASS Flag for Clinic."  D ^DIR  K DIR  I Y="I" G IND  I Y="A" G ALL  Q:Y="^"  Q  ; ALL ;  N IEN,CLFL,FLAG,Y,%,%Y  W !!?5,"CLINIC NAME",?40,"MASS FLAG",!  W ?5,"-----------",?40,"---------",!!  S LINE=7,IEN=0,Y=1  S IEN=0  F S IEN=$O(^SC(IEN)) Q:('IEN)!(Y=0) D  . S CLFL=$$GETFLAG(IEN)  . S FLAG=$S(CLFL=1:"YES",CLFL=0:"NO",1:"ERROR")  . W ?5,$P(^SC(IEN,0),"^"),?43,FLAG,!  . S LINE=LINE+1  . I LINE=22 D PAUSE^VALM1 S LINE=0  I Y=0 G ASKCLIN  ;  Q  ; DISPLAY(CLID) ; Displays the individual clinic MASS flag  ; and prompt to update it.  N CLFL,FLAG,CLNAME,FLG,CHANGE,%,%Y  S CLNAME=$P(^SC(CLID,0),"^")  S CLFL=$P($G(^SC(CLID,"MX")),"^")  S FLAG=$S(CLFL=1:"YES",CLFL=0:"NO",CLFL="":"NOT SET, USING DEFAULT",1:"ERROR")  W #,!!?5,"CLINIC NAME",?40,"MASS FLAG",!  W ?5,"-----------",?40,"---------",!  W !?5,CLNAME,?40,FLAG  W !!,"DO YOU WANT TO CHANGE THE MASS FLAG?: "  S %=2 ;default to yes  D YN^DICN ;Yes No question  I %'=1 Q  ; handle setting new value  W !!,"SET THE MASS FLAG TO YES OR NO?: "  S %=$S(CLFL=1:1,CLFL=0:2,1:"")  D YN^DICN ;Yes No question  ;determine if we are changing  S CHANGE=$S(%=1:1,%=2:0,1:"")  I CHANGE'="",CHANGE'=CLFL D  . S $P(^SC(CLID,"MX"),"^")=CHANGE,FLAG=$S(CHANGE=1:"YES",CHANGE=0:"NO")  . W !!?5,"THE MASS FLAG HAS BEEN UPDATED TO ",FLAG,!  E W !!?5,"THE MASS FLAG WAS NOT CHANGED."  Q  ; MSG(IEN) ;Displays message in the menus  N FLAG  S FLAG=$$GETFLAG(IEN)  D:FLAG  . W !!!?12,"This clinic is no longer schedulable in VistA."  . W !?12,"Please view or schedule to it in MASS.",!!  Q FLAG  ; GETFLAG(IEN) ;gets the clinic flag. If not set gets the default.  N FLAG  I $G(IEN)="" Q ""  S FLAG=$$GET1^DIQ(44,IEN\_",",22902,"I")  I FLAG="" S FLAG=$$GET^XPAR("SYS","SDMX CLINIC RO FLAG DEFAULT") ;Check System level default  I FLAG="" S FLAG=0  Q FLAG** | | | | | | | | |

Table : SDMXGAPT Routine

| Routines | Activities | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Routine Name | SDMXGAPT | | | | | | | |
| Enhancement Category | New | | Modify | Delete | | No Change | | |
| RTM |  | | | | | | | |
| Related Options |  | | | | | | | |
| Related Routines | Routines “Called By” | | | | Routines “Called” | | | |
|  | SDMXCANC  SDMXCHKO  SDMXCORE  SDMXMAKE  SDMXSCHP  SDMXSCHT | | | | DIQ  SDAM1  SDAM2 | | | |
| Routines | Activities | | | | | | | |
| Data Dictionary (DD) References | ^DPT(  ^SC( | | | | | | | |
| Related Protocols | N/A | | | | | | | |
| Related Integration Control Registrations (ICRs) | N/A | | | | | | | |
| Data Passing | Input | Output Reference | | | Both | | Global Reference | Local | |
| Input Attribute Name and Definition |  | | | | | | | |
| Output Attribute Name and Definition |  | | | | | | | |
| Current Logic | | | | | | | | |
| N/A – New routine | | | | | | | | |
| Modified Logic (Changes are in bold) | | | | | | | | |
| **SDMXGAPT ;MASS/RPC - Appointment retrieval API;08/17/2017**  **;;5.3;Scheduling;\*\*676\*\*;AUGUST 17,2017**  **;;Per VA directive 6402, this routine should not be modified.**  **Q**  **GETAPPT(PATIEN,APPTDT,APPTARY) ;Gets the appointment details form the database and returns it in an array.**  **; See parameter descriptions for details of the available nodes.**  **; PATIEN(I,REQ) - The patient IEN**  **; APPTDT(I,REQ) - the appointment date/time in vista format.**  **; Found in the ^DPT(<IEN>,"S",<APPTDT> node.**  **; APPTARY(O,REQ) - Array of appointment data.**  **;**  **; APPTARY("PATIENT IEN") - IEN**  **; APPTARY("APPTDT") = date/time of the appointment in VistA format**  **; APPTARY("APPTTYPE") - appointment type**  **; APPTARY("CANCEL REASON") - Cancellation reason (discrete)**  **; APPTARY("CANCEL REMARK") - Cancellation remark (freetext)**  **; APPTARY("CHECKIN DT") - date/time of the check in action**  **; APPTARY("CHECKIN USER") - check in user**  **; APPTARY("CHECKOUT DT") - date/time of the check out action**  **; APPTARY("CHECKOUT USER") - check out user**  **; APPTARY("CLINIC") = the clinic of this appointment**  **; APPTARY("CLINIC NAME") = Clinic name**  **; APPTARY("CLINIC PROVIDER",0) - number of providers on the clinic**  **; APPTARY("CLINIC PROVIDER",n,"IEN") - Provider IEN**  **; APPTARY("CLINIC PROVIDER",n,"NAME") - Name of provider**  **; APPTARY("CLINIC PURGED") - Flag to say the clinic has reached its**  **; purge days so the data is no longer available.**  **; APPTARY("CLINIC GROUP") - This is heavily covering our specialties.**  **; APPTARY("COMMENT") - Appointment comment.**  **; APPTARY("CONSULT") - linked consult ID**  **; APPTARY("COV") - The COV. Looking into what this is.**  **; APPTARY("DURATION") - appointment duration**  **; APPTARY("ENCOUNTER IEN") - Encounter ID**  **; APPTARY("ELIGIBILITY") - Appointment eligibility if different than primary**  **; APPTARY("Next Available") - If the appt was scheduled as next available**  **; APPTARY("PAT INDICATED DATE") - Date the start of search was performed on.**  **; APPTARY("STATUS") - APPT status calculated to current state**  **; (SCHEDULED/CHECKED IN/CHECKED OUT/CANCELLED/NO-SHOW)**  **; APPTARY("USER") - Scheduling user**  **;**  **N DELIM,CLNODE0,SNODE0,CLINIC,PROVARY,CLNODEC,CLNODECN**  **K APPTARY ;Force output only parameter**  **;**  **I $G(PATIEN)="" Q**  **I $G(APPTDT)="" Q**  **;**  **S DELIM="^"**  **S SNODE0=$$APPTNODE(PATIEN,APPTDT,0)**  **I $G(SNODE0)="" Q ;no appointment found**  **;**  **; Get clinic**  **S CLINIC=$P(SNODE0,DELIM,1)**  **S CLNODE0=$$CLINNODE(PATIEN,APPTDT,CLINIC,0)**  **S CLNODEC=$$CLINNODE(PATIEN,APPTDT,CLINIC,"C")**  **S CLNODECN=$$CLINNODE(PATIEN,APPTDT,CLINIC,"CONS")**  **;**  **; Get providers off of clinic**  **I $G(CLINIC)'="" D CLINPROV(CLINIC,.PROVARY)**  **;**  **; Build Array of patient information from the known nodes**  **S APPTARY("PATIENT IEN")=PATIEN ;IEN**  **S APPTARY("APPTDT")=APPTDT ;date/time of the appointment in VistA format**  **S APPTARY("APPTTYPE")=$P(SNODE0,DELIM,16) ;appointment type**  **S APPTARY("CLINIC")=$G(CLINIC) ;the clinic of this appointment ;**  **I $G(CLNODE0)="" S APPTARY("CLINIC PURGED")=1**  **I $D(PROVARY) M APPTARY("CLINIC PROVIDER")=PROVARY**  **S APPTARY("CLINIC NAME")=$$GET1^DIQ(44,$G(CLINIC),.01)**  **S APPTARY("CLINIC GROUP")=$$CLINGRP($G(CLINIC))**  **S APPTARY("CLINIC GROUP NAME")=$$GET1^DIQ(409.67,$G(APPTARY("CLINIC GROUP")),.01)**  **S APPTARY("CHECKIN DT")=$P($G(CLNODEC),DELIM,1)**  **S APPTARY("CHECKIN USER")=$P($G(CLNODEC),DELIM,2)**  **S APPTARY("CHECKOUT DT")=$P($G(CLNODEC),DELIM,3)**  **S APPTARY("CHECKOUT USER")=$P($G(CLNODEC),DELIM,4)**  **S APPTARY("COMMENT")=$P(CLNODE0,DELIM,4)**  **S APPTARY("CONSULT")=$$CLINNODE(PATIEN,APPTDT,CLINIC,"CONS") ;consult ID**  **S APPTARY("COV")=$P(SNODE0,DELIM,7) ;The COV if c&P/Scheduled/or walked in**  **S APPTARY("DURATION")=$P(CLNODE0,DELIM,2)**  **s APPTARY("ELIGIBILITY")=$P(CLNODE0,DELIM,10)**  **S APPTARY("PAT INDICATED DATE")=$$APTNODEP(PATIEN,APPTDT,1,1)**  **; APPT status - Computed and translated**  **S APPTARY("STATUS")=$$APTSTAT(PATIEN,APPTDT,0)**  **S APPTARY("STATUS PIECE")=$P(SNODE0,DELIM,2) ;Status from the status node**  **S APPTARY("USER")=$P(SNODE0,DELIM,18) ;Scheduling user**  **S APPTARY("CANCEL REASON")=$P(SNODE0,DELIM,15)**  **S APPTARY("CANCEL REMARK")=$$APTNODEP(PATIEN,APPTDT,"R",1)**  **S APPTARY("ENCOUNTER IEN")=$P(SNODE0,DELIM,20)**  **Q**  **APPTNODE(PATIEN,APPTDT,NODE) ;For a given patient we will return their appointment node in the ^DPT file.**  **; PATIEN(I,REQ) - The patient IEN**  **; APPTDT(I,REQ) - the appointment date/time in vista format. Found in the ^DPT(<IEN>,"S",<APPTDT> node.**  **; NODE(I,REQ) - node number to pull**  **;**  **I ($G(APPTDT)="")!($G(PATIEN)="")!($G(NODE)="") Q**  **Q $G(^DPT(PATIEN,"S",APPTDT,NODE))**  **APTNODEP(PATIEN,APPTDT,NODE,PIECE) ;For a given patient we will return a piece of their appointment node in the ^DPT file.**  **; PATIEN(I,REQ) - The patient IEN**  **; APPTDT(I,REQ) - the appointment date/time in vista format. Found in the ^DPT(<IEN>,"S",<APPTDT> node.**  **; NODE(I,REQ) - node number to pull**  **; PIECE(I,REQ) - piece in the node to return**  **; DATA(I,OPT) - data node may be passed in to bypass extracting.**  **N DATA**  **S DATA=$$APPTNODE($G(PATIEN),$G(APPTDT),$G(NODE))**  **Q $P($G(DATA),"^",$G(PIECE))**  **CLINNODE(PATIEN,APPTDT,CLINICIEN,NODE,COUNT) ;For a given patient we will find their clinic node in the ^SC file.**  **; May need to loop through the overbooks on that time.**  **; PATIEN(I,REQ) - The patient IEN**  **; APPTDT(I,REQ) - the appointment date/time in vista format. Found in the ^DPT(<IEN>,"S",<APPTDT> node.**  **; CLINICIEN(I,REQ) - clinic record IEN to search through**  **; NODE(I,REQ) - last node for the clinic. 0 node has appointment info. "C" node has check in/out info.**  **; COUNT(O,OPT) - subscript # of appointment**  **N RET**  **; Loop through the possibly multiple appointments scheduled into this clinic slot**  **S COUNT=$$FIND^SDAM2($G(PATIEN),$G(APPTDT),$G(CLINICIEN))**  **I $G(COUNT)'="" S RET=$G(^SC($G(CLINICIEN),"S",$G(APPTDT),1,COUNT,$G(NODE))) ;Get the node from the SC global**  **Q $G(RET)**  **CLINPROV(CLINIC,ARRAY) ;Sets an array filled with clinic provider data**  **; CLINIC (I,REQ) - The Clinic IEN (first piece of DPT 0 node)**  **; ARRAY (O,REQ) - APPTARY("CLINIC PROVIDER",0) - number of providers on the clinic**  **; APPTARY("CLINIC PROVIDER",N,"IEN") - Provider IEN**  **; APPTARY("CLINIC PROVIDER",N,"NAME") - Name of provider**  **N NUM**  **K ARRAY**  **I $G(CLINIC)="" Q**  **S NUM=0**  **S ARRAY(0)=0**  **F S NUM=$O(^SC(CLINIC,"PR",NUM)) Q:'NUM D**  **. S ARRAY(NUM,"IEN")=$$PROVIEN(CLINIC,NUM) ;-Provider IEN File 200**  **. I ARRAY(NUM,"IEN")="" Q**  **. S ARRAY(NUM,"NAME")=$$GET1^DIQ(200,(ARRAY(NUM,"IEN")),.01) ;- Name of provider File 200**  **. S ARRAY(0)=ARRAY(0)+1**  **Q**  **PROVIEN(CLINIC,NODE) ;Returns the Nth provider ID for a Clinic**  **; CLINIC (I,REQ) - The Clinic IEN (first piece of DPT 0 node)**  **; NODE (I,REQ) - The count of the node being examined**  **Q $P(^SC($G(CLINIC),"PR",$G(NODE),0),"^",1)**  **CLINGRP(CLINIC) ;Returns the ID of a Clinic's group**  **; CLINIC (I,REQ) - The Clinic IEN (first piece of DPT 0 node)**  **Q $P(^SC($G(CLINIC),0),"^",31)**  **;---------**  **; DESCRIPTION:**  **; PARAMETERS:**  **;**  **;**  **APTSTAT(PATIEN,APPTDT,FULLSTAT) ;Returns current computed appointment status which**  **; includes checked in/out which the "S"0;2 node does not.**  **; PATIEN (I,REQ)- Patient ID as in DPT(PATIEN,"S",APPTDAT**  **; APPTDT (I,REQ) - Appointment date**  **; FULLSTAT (I,OPT,DEFAULT:"") - Set to 1 return full STATUS string**  **; OUTPUT: Appointment current Status values:**  **; SCHEDULED (Default)**  **; CHECKED IN**  **; CHECKED OUT**  **; CANCELLED**  **; NO-SHOW**  **; "" if the appointment does not exist.\**  **;**  **N RET,DPT0,CLINICID**  **I $G(PATIEN)="" Q ""**  **I $G(APPTDT)="" Q ""**  **S DPT0=$$APPTNODE(PATIEN,APPTDT,0)**  **I $G(DPT0)="" Q ""**  **;CLINIC ID IS FIRST PART OF DPT "S" 0 NODE**  **S CLINICID=+$G(DPT0)**  **I $G(CLINICID)="" Q ""**  **S RET=$$STATUS^SDAM1(PATIEN,APPTDT,CLINICID,DPT0)**  **I '$G(FULLSTAT) D**  **. S RET=$P(RET,";",3) ;PRINT STATUS- NOTE THAT THESE HAVE THE POSSIBILITY OF RETURNING MULTIPLE STATUSES**  **. S RET=$S(RET["CANCELLED":"CANCELLED",RET["NO-SHOW":"NO SHOW",RET["CHECKED OUT":"CHECKED OUT",RET["CHECKED IN":"CHECKED IN",1:"SCHEDULED")**  **Q RET**  **;** | | | | | | | | |

Table : SDMXLKRQ Routine

| Routines | Activities | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Routine Name | SDMXLKRQ | | | | | | | |
| Enhancement Category | New | | Modify | Delete | | No Change | | |
| RTM |  | | | | | | | |
| Related Options |  | | | | | | | |
| Related Routines | Routines “Called By” | | | | Routines “Called” | | | |
|  | ORMXTR  SDMXSCHP | | | | HLOAPI3  XLFDT | | | |
| Routines | Activities | | | | | | | |
| Data Dictionary (DD) References | ^DPT(  ^GMR(123  ^HLB("QUEUE"  ^OR(100  ^SC( | | | | | | | |
| Related Protocols | N/A | | | | | | | |
| Related Integration Control Registrations (ICRs) | N/A | | | | | | | |
| Data Passing | Input | Output Reference | | | Both | | Global Reference | Local | |
| Input Attribute Name and Definition |  | | | | | | | |
| Output Attribute Name and Definition |  | | | | | | | |
| Current Logic | | | | | | | | |
| N/A – New routine | | | | | | | | |
| Modified Logic (Changes are in bold) | | | | | | | | |
| **SDMXLKRQ ;MASS/DAP - Locking and Resequencing Tags;11/05/17**  **;;5.3;Scheduling;\*\*676\*\*;NOVEMBER 25,2017**  **;;Per VA directive 6402, this routine should not be modified.**  **Q**  **;**  **;**  **SEQUENCE(MSGARY,LOCKARY) ;Entry point- Called from SDMX scheduling**  **N PATIEN,CLINIEN,APPTDT,ORDID,CONSID,MSGIEN**  **K LOCKARY**  **D ARY2VAR(.MSGARY,.PATIEN,.CLINIEN,.APPTDT,.ORDID,.CONSID,.MSGIEN) ;Puts the Array's values in the local variables.**  **I $$INHOLDQ(PATIEN,CLINIEN,APPTDT,ORDID,CONSID,MSGIEN) D RESEQ(PATIEN,CLINIEN,APPTDT,ORDID,CONSID,MSGIEN) Q 0**  **I '$$LOCKALL(PATIEN,CLINIEN,APPTDT,ORDID,CONSID) D RESEQ(PATIEN,CLINIEN,APPTDT,ORDID,CONSID,MSGIEN) Q 0**  **D CACHEARY(.LOCKARY,PATIEN,CLINIEN,APPTDT,ORDID,CONSID)**  **I $$SHOULDRQ() D REQUEUE()**  **Q 1**  **AFTRPROC(LOCKARY) ;Entry point after message processing to release locks. Sequence sets the input via CACHEARY.**  **N PATIEN,CLINIEN,APPTDT,ORDID,CONSID,MSGIEN**  **D ARY2VAR(.LOCKARY,.PATIEN,.CLINIEN,.APPTDT,.ORDID,.CONSID,.MSGIEN)**  **D ULOCKALL(PATIEN,CLINIEN,APPTDT,ORDID,CONSID)**  **I $$SHOULDRQ() D REQUEUE()**  **Q**  **INHOLDQ(PATIEN,CLINIEN,APPTDT,ORDID,CONSID,MSGIEN) ;Evaluates if the current message has any lockable IDs in the hold queue**  **I PATIEN,APPTDT,$G(@$$HQGLO()@("PATIEN",PATIEN,APPTDT))'="" Q 1**  **I CLINIEN,APPTDT,$G(@$$HQGLO()@("CLINIEN",CLINIEN,APPTDT))'="" Q 1**  **I ORDID,$G(@$$HQGLO()@("ORDID",ORDID))'="" Q 1**  **I CONSID,$G(@$$HQGLO()@("CONSID",CONSID))'="" Q 1**  **I MSGIEN,$G(@$$HQGLO()@("MSGIEN",MSGIEN))'="" Q 1**  **Q 0**  **;**  **LOCKALL(PATIEN,CLINIEN,APPTDT,ORDID,CONSID) ;**  **;PATIEN - Patient IEN**  **;CLINIEN - Clinic IEN**  **;APPDT - Appointment date and time in vista format. For example: 3171024.0915**  **;ORDID - Order ID**  **;CONSID - Consult ID**  **I PATIEN,APPTDT,'$$LOCKPAT(PATIEN,APPTDT) Q 0**  **I CLINIEN,APPTDT,'$$LOCKCLIN(CLINIEN,APPTDT) Q 0**  **I ORDID,'$$LOCKORD(ORDID) Q 0**  **I CONSID,'$$LOCKCONS(CONSID) Q 0**  **Q 1**  **RESEQ(PATIEN,CLINIEN,APPTDT,ORDID,CONSID,MSGIEN) ;Resequences a message**  **N OK**  **D ULOCKALL(PATIEN,CLINIEN,APPTDT,ORDID,CONSID)**  **D ADDHOLDQ(PATIEN,CLINIEN,APPTDT,ORDID,CONSID,MSGIEN)**  **I $$SHOULDRQ() D REQUEUE()**  **Q**  **ULOCKALL(PATIEN,CLINIEN,APPTDT,ORDID,CONSID) ;**  **;PATIEN - Patient IEN**  **;CLINIEN - Clinic IEN**  **;APPDT - Appointment date and time in vista format. For example: 3171024.0915**  **;ORDID - Order ID**  **I PATIEN,APPTDT,'$$ULOCKPAT(PATIEN,APPTDT) Q 0**  **I CLINIEN,APPTDT,'$$ULCKCLIN(CLINIEN,APPTDT) Q 0**  **I ORDID,'$$ULOCKORD(ORDID) Q 0**  **I CONSID,'$$ULCKCONS(CONSID) Q 0**  **Q 1**  **ADDHOLDQ(PATIEN,CLINIEN,APPTDT,ORDID,CONSID,MSGIEN) ;Adds a message entry to the hold queue**  **S @$$HQGLO()@(0)=$$HTFM^XLFDT($H+31,1)\_"^"\_$$HTFM^XLFDT($H,1) ;Purge date^Create date used by XQ82**  **I $I(@$$HQGLO())=1 S @$$HQGLO()@("OLDEST$H")=$H**  **I PATIEN,APPTDT,$I(@$$HQGLO()@("PATIEN",PATIEN,APPTDT))**  **I CLINIEN,APPTDT,$I(@$$HQGLO()@("CLINIEN",CLINIEN,APPTDT))**  **I ORDID,$I(@$$HQGLO()@("ORDID",ORDID))**  **I CONSID,$I(@$$HQGLO()@("CONSID",CONSID))**  **I $I(@$$HQGLO()@("MSGIEN",MSGIEN))=1 S OK=$I(@$$HQGLO()@("UNIQUEMSGS"))**  **Q 1**  **SHOULDRQ() ;Determines if the hold queue should be requeued**  **N HELDMSGS,QUETIME,RQUETIME,TDIFF**  **S HELDMSGS=$G(@$$HQGLO())**  **I +HELDMSGS=0 Q 0**  **;I $$QUEEMPTY() H 3 Q 1 ;Checks the HLO queue**  **S QUETIME=$P(@$$HQGLO()@("OLDEST$H"),",",2)**  **S RQUETIME=$P($H,",",2)**  **S TDIFF=(RQUETIME-QUETIME)**  **I TDIFF>10 Q 1**  **I TDIFF<0 Q 1 ;Midnight and DST can get a bonus requeue**  **Q 0**  **REQUEUE() ;Merges the hold queue back into the HLO queue**  **N HELDMSGS,MSGIEN,ERROR**  **F S MSGIEN=$O(@$$HQGLO()@("MSGIEN",$G(MSGIEN))) Q:(MSGIEN="") D**  **. D REPROC^HLOAPI3(MSGIEN,.ERROR)**  **. D CLRHQNDE("MSGIEN",MSGIEN)**  **D CLRHQGLO**  **Q**  **QUEEMPTY() ;Returns if the HLO interface queue is empty.**  **Q ($O(^HLB("QUEUE","IN",$G(QUEUE("FROM")),$G(QUEUE("QUEUE")),""))="")**  **LOCKPAT(PATIEN,APPTDT) ;**  **Q $$LOCK($NA(^DPT(PATIEN,"S",APPTDT)))**  **LOCKCLIN(CLINIEN,APPTDT) ;**  **Q $$LOCK($NA(^SC(CLINIEN,"S",APPTDT)))**  **LOCKORD(ORDID) ;**  **Q $$LOCK($NA(^OR(100,ORDID)))**  **LOCKCONS(CONSID) ;**  **Q $$LOCK($NA(^GMR(123,CONSID)))**  **ULOCKPAT(PATIEN,APPTDT) ;**  **Q $$ULOCK($NA(^DPT(PATIEN,"S",APPTDT)))**  **ULCKCLIN(CLINIEN,APPTDT) ;**  **Q $$ULOCK($NA(^SC(CLINIEN,"S",APPTDT)))**  **ULOCKORD(ORDID) ;**  **Q $$ULOCK($NA(^OR(100,ORDID)))**  **ULCKCONS(CONSID) ;**  **Q $$ULOCK($NA(^GMR(123,CONSID)))**  **LOCKOGR(FR) ; LOCKS AN OPEN GLOBAL ROOT**  **; FR - AN OPEN GLOBAL ROOT AS USED BY FILEMAN (DIE, DIC)**  **S FR=$E(FR,"",$L(FR)-1)\_")"**  **Q $$LOCK(FR)**  **ULOCKOGR(FR) ; UNLOCKS AN OPEN GLOBAL ROOT**  **; FR - AN OPEN GLOBAL ROOT AS USED BY FILEMAN (DIE, DIC)**  **S FR=$E(FR,"",$L(FR)-1)\_")"**  **Q $$ULOCK(FR)**  **LOCK(GLO) ;Locks a lock name**  **;GLO -Lock name**  **S GLO=$NA(@GLO)**  **L +@GLO:0 E Q 0**  **Q 1**  **ULOCK(GLO) ;Unlocks a lock name**  **;GLO -Lock name**  **S GLO=$NA(@GLO)**  **L -@GLO:0 E Q 0**  **Q 1**  **HQGLO() ;**  **;Returns a reference to the global used for resequencing**  **Q $NA(^XTMP("SDMX HOLDQUEUE"))**  **CLRHQGLO() ;Clears the entire hold queue global**  **K @$$HQGLO() ;**  **Q**  **CLRHQNDE(NODE,VAL) ;Clears a specified node in the hold queue global**  **K @$$HQGLO()@(NODE,VAL)**  **Q**  **ARY2VAR(ARY,PATIEN,CLINIEN,APPTDT,ORDID,CONSID,MSGIEN) ;Pulls IDs out of the SDMX standard scheduling array**  **S PATIEN=$G(ARY("PATIENT IEN"))**  **S CLINIEN=$G(ARY("CLINIC"))**  **S APPTDT=$G(ARY("APPTDT"))**  **S ORDID=$G(MSGARY("ORDER ID"))**  **S CONSID=$G(ARY("CONSULT ID"))**  **S MSGIEN=$G(ARY("MESSAGE IEN"))**  **Q**  **CACHEARY(ARY,PATIEN,CLINIEN,APPTDT,ORDID,CONSID) ;Sets the IDs used for locking in a SDMX standard scheduling array**  **S ARY("PATIENT IEN")=PATIEN**  **S ARY("CLINIC")=CLINIEN**  **S ARY("APPTDT")=APPTDT**  **S ARY("ORDER ID")=ORDID**  **S ARY("CONSULT ID")=CONSID**  **Q** | | | | | | | | |

Table : DGMXADT Routine

| Routines | Activities | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Routine Name | DGMXADT | | | | | | | | |
| Enhancement Category | New | | Modify | Delete | | | No Change | | |
| RTM | BN1A6.1  BN1A6.3  BN1E1.2  BN2A1  BN2A1.1  BN2A1.2  BN2B1  BN2B1.1  BN2B1.6  BN2B1.10  BN3A1.7  BN3B1.8  BN4A1.11  BN4B1.4 | | | | | | | | |
| Related Options |  | | | | | | | | |
| Related Routines | Routines “Called By” | | | | | Routines “Called” | | | |
|  | VAFCMSG | | | | | DIC  DICN  DGMXHL7  HLOAPI  XLFDT  XQOR | | | |
| Routines | Activities | | | | | | | | |
| Data Dictionary (DD) References | ^DPT(  ^ORD(101 | | | | | | | | |
| Related Protocols | N/A | | | | | | | | |
| Related Integration Control Registrations (ICRs) | 4716,4717,10063,10103 | | | | | | | | |
| Data Passing | Input | Output Reference | | | Both | | | Global Reference | Local |
| Input Attribute Name and Definition |  | | | | | | | | |
| Output Attribute Name and Definition |  | | | | | | | | |
| Current Logic | | | | | | | | | |
| N/A | | | | | | | | | |
| Modified Logic (Changes are in bold) | | | | | | | | | |
| DGMXADT ; MASS/RPR - ADT Trigger Code;8/24/17  ;;5.3;Registration;\*\*948\*\*;SEPTEMBER 5,2017;Build 7  ;VA DIRECTIVE 6402, this routine should not be modified.  ;  ; ICR# Supported References  ; 4716 $$ADDSEG^HLOAPI  ; 4716 $$NEWMSG^HLOAPI  ; 4717 $$SENDONE^HLOAPI1  ; 10063 D ^%ZTLOAD  ; 10103 $$NOW^XLFDT  ;  Q MONITOR(DGMXDFN,DGMXTYPE,DGMXFMX,DGMXFMX1,DGMXFMX2) ; Send ADT messages to subscribers when demographics change  ; DGMXDFN - Patient edited (required)  ; DGMXTYPE - Type of cross-reference action ("SET" or "KILL") (required)  ; DGMXFMX - X array for FileMan new style x-refs (required)  ; DGMXFMX1 - X1 array for FileMan new style x-refs (required)  ; DGMXFMX2 - X2 array for FileMan new style x-refs (required)  ;  ; Validate Inputs  I $G(DGMXDFN)="" Q ; Quit if no patient  I $G(DGMXFMX)="" Q ; Quit if setting or killing null value  I $G(DGMXTYPE)="" Q ; Quit if no edit type  ;  I (DGMXTYPE="KILL"),(DGMXFMX2'="") Q ; Skip kill action on data update  ;  ; Set edit type  S DGMXTYPE=$S(DGMXTYPE="KILL":"DELETE",DGMXTYPE="SET":"ADD",1:DGMXTYPE)  I ($G(DGMXFMX1)'=""),($G(DGMXFMX2)'="") S DGMXTYPE="UPDATE"  ;  ; Task off (Taskman) driver routine.  N ZTRTN,ZTDESC,ZTIO,ZTDTH,ZTSAVE,ZTSK,DGMXVAR,BXREF,SUBSCR,ZTREQ  S ZTRTN="INIT^DGMXADT",ZTDESC="DGMX DEMOGRAPHICS MONITOR"  S ZTIO="DGMX DEMOGRAPHICS MONITOR",ZTDTH=$$NOW^XLFDT()  F DGMXVAR="DGMXDFN","DGMXTYPE","DGMXFMX","DGMXFMX1","DGMXFMX2" S ZTSAVE(DGMXVAR)=""  ; If there are no subscribers, do not call Taskman  S BXREF=0,BXREF=$O(^ORD(101,"B","DGMX DEMOGRAPHICS MONITOR",BXREF))  S SUBSCR=0,SUBSCR=$O(^ORD(101,BXREF,10,SUBSCR))  I 'SUBSCR Q  D ^%ZTLOAD  Q INIT N X ; Loop through the subscribers to DGMX DEMOGRAPHICS MONITOR and execute them  S X=$O(^ORD(101,"B","DGMX DEMOGRAPHICS MONITOR",0))\_";ORD(101," D EN1^XQOR  I $D(ZTQUEUED) S ZTREQ="@"  K DGMXDFN,DGMXTYPE,DGMXFMX,DGMXFMX1,DGMXFMX2  Q TRIGGER(DFN,EDITTYPE) ; Trigger HL7 messages when patient edits are made  ; DFN - Patient IEN (required)  ; EDITTYPE - Type of modification made to patient ("ADD", "UPDATE", or "DELETE")  ;  I ($G(DFN)="")!($G(EDITTYPE)="") Q ; validate inputs  I EDITTYPE="ADD" D TRIGDEM(DFN,"A28") Q  I EDITTYPE="UPDATE" D TRIGDEM(DFN,"A08") Q  I EDITTYPE="DELETE" D TRIGA40(DFN)  Q TRIGDEM(DFN,EVENT) ; Triggers a Patient Update Message  ; DFN - Patient IEN  ; EVENT - Event type to trigger. Trigger an A28 for new patients.  ;  I $G(DFN)="" Q  N SEG,MSG,PARAMS,ERROR,WHOTO,OK,ENR  S PARAMS("MESSAGE TYPE")="ADT"  I $G(EVENT)="" S PARAMS("EVENT")="A08"  E S PARAMS("EVENT")=EVENT  S PARAMS("SENDING APPLICATION")="DG-REG-OUT"  S WHOTO("RECEIVING APPLICATION")="MASS"  S WHOTO("FACILITY LINK NAME")="EPICADTIN" ; HL Logical Link  S ENR=$P($G(^DPT(DFN,"ENR")),"^",1) ; Get Enrollment information  ;  ; Start new message  I '$$NEWMSG^HLOAPI(.PARAMS,.MSG) Q  D PID^DGMXHL7(DFN,1,.SEG) ; Build PID (required)  I '$$ADDSEG^HLOAPI(.MSG,.SEG) Q  D PV1^DGMXHL7(DFN,1,.SEG) ; Build PV1  S OK=$$ADDSEG^HLOAPI(.MSG,.SEG)  D ZCT^DGMXHL7(DFN,1,.SEG) ; Build ZCT  S OK=$$ADDSEG^HLOAPI(.MSG,.SEG)  D BLDZEL^DGMXHL7(DFN,.SEG,.MSG) ; Build ZEL (repeating)  D ZEN^DGMXHL7(ENR,1,.SEG) ; Build ZEN  S OK=$$ADDSEG^HLOAPI(.MSG,.SEG)  D ZSP^DGMXHL7(DFN,1,.SEG) ; Build ZSP  S OK=$$ADDSEG^HLOAPI(.MSG,.SEG)  ;  ; Send message  S OK=$$SENDONE^HLOAPI1(.MSG,.PARAMS,.WHOTO,.ERROR)  Q TRIGA40(PREVID) ; Triggers an A40 Message (Patient Merge), given the source patient.  ; The source patient record is merged INTO the (surviving) target patient record.  ; PREVID - Source Patient IEN  ;  I $G(PREVID)="" Q  N SEG,MSG,PARAMS,ERROR,WHOTO,DFN,OK  S PARAMS("MESSAGE TYPE")="ADT"  S PARAMS("EVENT")="A40"  S PARAMS("SENDING APPLICATION")="DG-REG-OUT"  S WHOTO("RECEIVING APPLICATION")="MASS"  S WHOTO("FACILITY LINK NAME")="EPICADTIN" ; HL Logical Link  S DFN=$G(^DPT(PREVID,-9)) ; Target Patient  I DFN="" Q  ;  ; Start new message  I '$$NEWMSG^HLOAPI(.PARAMS,.MSG) Q  D PID^DGMXHL7(DFN,1,.SEG) ; Build PID (required)  I '$$ADDSEG^HLOAPI(.MSG,.SEG) Q  D MRG^DGMXHL7(PREVID,.SEG) ; Build MRG (required)  I '$$ADDSEG^HLOAPI(.MSG,.SEG) Q  ;  ; Send message  S OK=$$SENDONE^HLOAPI1(.MSG,.PARAMS,.WHOTO,.ERROR)  Q MANLTRIG ; Manually trigger ADT messages for all patients or a subset of patients  N DFN,Y,LIST,DONE,LENGTH,IND  ;  W !,"Trigger ADT for ALL patients"  S %=2 ; Default no  D YN^DICN  ;  ; Yes - Trigger ADT for ALL patients  I %=1 D Q  . K %  . W !,"This will trigger approximately "\_$P(^DPT(0),"^",4)\_" messages. Are you sure"  . S %=2 ; Default no  . D YN^DICN  . I %=1 D IMPORTPT Q  . Q  ;  ; No - Trigger ADT for subset of patients  I %=2 D  . ; Select patients  . S DONE=0  . F Q:DONE=1 D  . . K DIC  . . S DIC=2,DIC(0)="QEAMIZ",DIC("A")="Select PATIENT: "  . . D ^DIC  . . S DFN=+Y  . . I DFN<0 S DONE=1 Q  . . I LIST="" S LIST=DFN Q  . . S LIST=LIST\_","\_DFN  . ;  . ; Trigger messages  . S LENGTH=$L(LIST,",")  . I LENGTH<1 Q  . F IND=1:1:LENGTH D  . . S DFN=$P(LIST,",",IND)  . . D TRIGDEM(DFN,"A28")  Q IMPORTPT ; Loops through ^DPT and triggers an A08. Excludes stub patient records.  N DFN  F S DFN=$O(^DPT(DFN)) Q:DFN="" D  . I ^DPT(DFN,-9)'="" Q  . D TRIGDEM^DGMXADT(DFN,"A28")  Q TRIGADDL(DFN) ; Trigger additional messages  I $$GET1^DIQ(2,DFN,.121)'="" D TRIGBAD^DGMXORU(DFN,"ACTIVE")  Q  ; | | | | | | | | | |

Table : DGMXHL7 Routine

| Routines | Activities | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Routine Name | DGMXHL7 | | | | | | | | |
| Enhancement Category | New | | Modify | Delete | | | No Change | | |
| RTM | BN1A6.1  BN1A6.3  BN1E1.2  BN2A1  BN2A1.1  BN2A1.2  BN2B1  BN2B1.1  BN2B1.6  BN2B1.10  BN3A1.7  BN3B1.8  BN4A1.11  BN4B1.4 | | | | | | | | |
| Related Options |  | | | | | | | | |
| Related Routines | Routines “Called By” | | | | | Routines “Called” | | | |
|  | DGMXADT | | | | | DIQ  HLOAPI  HLOAPI4  MPIF001  SDMXCORE  VADPT  XLFNAME  XPAR | | | |
| Routines | Activities | | | | | | | | |
| Data Dictionary (DD) References | ^DIC(5  ^DPT( | | | | | | | | |
| Related Protocols | N/A | | | | | | | | |
| Related Integration Control Registrations (ICRs) | 4716,4717 | | | | | | | | |
| Data Passing | Input | Output Reference | | | Both | | | Global Reference | Local |
| Input Attribute Name and Definition |  | | | | | | | | |
| Output Attribute Name and Definition |  | | | | | | | | |
| Current Logic | | | | | | | | | |
| N/A | | | | | | | | | |
| Modified Logic (Changes are in bold) | | | | | | | | | |
| DGMXHL7 ; MASS/DXL - HL7 Segment Build Code;8/22/17  ;;5.3;Registration;\*\*948\*\*;AUGUST 17,2017  ;VA DIRECTIVE 6402, this routine should not be modified.  ;  ; ICR# Supported References  ; 2056 $$GET1^DIQ  ; 2056 $$GETS^DIQ  ; 2701 $$GETICN^MPIF001  ; 3065 STDNAME^XLFNAME  ; 4716 $$ADDSEG^HLOAPI  ; 4716 SET^HLOAPI  ; 4853 SETAD^HLOAPI4  ; 4853 SETDT^HLOAPI4  ; 4853 SETXPN^HLOAPI4  ; $$GETALL^DGPFAA  ; 10061 ADD^VADPT Patient Address  ; 10061 DEM^VADPT Patient Demographics  ; 10061 ELIG^VADPT Patient Eligibility  ;  Q  PID(DFN,SEQ,SEG) ; Builds a PID segment  ; DFN - VistA Internal Entry Number (IEN)  ; SEQ - Segment Repetition index  ; SEG - Segment output  ;  I $G(DFN)="" Q  I $G(SEQ)="" S SEQ=1  K SEG S SEG=""  ;  ; Get Patient Demographics, Address, and Telecommunication Information  N NAME,DOB,SSN,ICN,ADDRESS,USEPERM  N SEX,ALIAS,TEMPADDR,CONFADDR,HPHONE,WPHONE,TPHONE,DOD,EMAIL,HPEXT,WPEXT,TPEXT  D GETDEMOG(DFN,.NAME,.DOB,.SSN,.SEX,.ICN,.DOD,.ALIAS)  D GETADDR(DFN,.ADDRESS,.CONFADDR,.TEMPADDR,.USEPERM)  D GETTELE(DFN,.HPHONE,.HPEXT,.WPHONE,.WPEXT,.TPHONE,.TPEXT,.EMAIL,USEPERM)  ;  ; Sanitize Variables  S HPHONE=$G(HPHONE),WPHONE=$G(WPHONE),TPHONE=$G(TPHONE)  S HPEXT=$G(HPEXT),WPEXT=$G(WPEXT),TPEXT=$G(TPEXT)  ;  ; Build Segment  D SET^HLOAPI(.SEG,"PID",0) ; PID-0 ("PID")  D SET^HLOAPI(.SEG,SEQ,1) ; PID-1 (Repetition)  D BLDPID3(.SEG,ICN,DFN,3) ; PID-3 (Patient Identifiers)  D SETXPN^HLOAPI4(.SEG,.NAME,5) ; PID-5 (Patient Name)  D SETDT^HLOAPI4(.SEG,DOB,7) ; PID-7 (Date of Birth)  D SET^HLOAPI(.SEG,SEX,8) ; PID-8 (Gender)  D BLDPID9(.SEG,ALIAS,9) ; PID-9 (Aliases)  D BLDPID11(.SEG,.ADDRESS,.CONFADDR,.TEMPADDR,11) ; PID-11 (Address)  D BLDPID13(.SEG,HPHONE,HPEXT,EMAIL,TPHONE,TPEXT,13) ; PID-13 (Home Phone Number/Email)  D BLDPID14(.SEG,WPHONE,WPEXT,14) ; PID-14 (Work Phone)  D SET^HLOAPI(.SEG,SSN,19) ; PID-19 (Social Security Number)  D SETDT^HLOAPI4(.SEG,DOD,29) ; PID-29 (Date of Death)  Q  PV1(DFN,SEQ,SEG) ; Builds a PV1 segment  ; DFN - VistA Internal Entry Number (IEN)  ; SEQ - Segment Repetition index  ; SEG - Segment output  ;  I $G(DFN)="" Q  I $G(SEQ)="" S SEQ=1  K SEG S SEG=""  ;  ; Get Patient Type  N PTYPE  S PTYPE=$$GET1^DIQ(2,DFN,391)  ;  ; Build Segment  I $G(PTYPE)="" Q ; No Data - Don't build segment  D SET^HLOAPI(.SEG,"PV1",0) ; PV1-0 ("PV1")  D SET^HLOAPI(.SEG,SEQ,1) ; PV1-1 (Repetition)  D SET^HLOAPI(.SEG,PTYPE,18) ; PV1-18 (Patient type)  Q  MRG(DFN,SEG) ; Builds an MRG segment  ; DFN - VistA Internal Entry Number (IEN) - This should be the source (old) patient  ; SEG - Segment output  ;  I $G(DFN)="" Q  K SEG S SEG=""  ;  ; Build Segment  D SET^HLOAPI(.SEG,"MRG",0) ; MRG-0 ("MRG")  D BLDPID3(.SEG,-1,DFN,1) ; MRG-1 (Old Patient ID)  Q  OBR(SEQ,SEG,SUMMARY) ; Builds an OBR segment  ; SEQ - Segment Repetition index  ; SEG - Segment output  ; SUMMARY - Name of the flag  ;  I $G(SEQ)="" S SEQ=1  K SEG S SEG=""  ;  I $G(SUMMARY)="" Q ; No Data - Don't build segment  D SET^HLOAPI(.SEG,"OBR",0) ; OBR-0 ("OBR")  D SET^HLOAPI(.SEG,SEQ,1) ; OBR-1 (Repetition)  D SET^HLOAPI(.SEG,SUMMARY,4) ; OBX-4 (Flag Name)  Q  OBX(SEQ,SEG,TYPE,TEXT) ; Builds an OBX segment  ; SEQ - Segment Repetition index  ; SEG - Segment output  ; TYPE - Observation type  ; TEXT - Observation value  ;  I $G(SEQ)="" S SEQ=1  K SEG S SEG=""  ;  ; Get Observation Types  N VALTYPE  S TYPE=$G(TYPE)  S VALTYPE=$S(TYPE="NARRATIVE":"TX",TYPE="STATUS":"ST",1:TYPE)  ;  I $G(TEXT)="" Q ; No Data - Don't build segment  D SET^HLOAPI(.SEG,"OBX",0) ; OBX-0 ("OBX")  D SET^HLOAPI(.SEG,SEQ,1) ; OBX-1 (Repetition)  D SET^HLOAPI(.SEG,VALTYPE,2) ; OBX-2 (Value Type)  D SET^HLOAPI(.SEG,TYPE,3) ; OBX-3 (Observation Type - Narrative/Status)  D SET^HLOAPI(.SEG,TEXT,5) ; OBX-5 (Observation Value)  Q  ZCT(DFN,SEQ,SEG) ; Builds a ZCT segment  ; DFN - VistA Internal Entry Number (IEN)  ; SEQ - Segment Repetition index  ; SEG - Segment output  ;  I $G(DFN)="" Q  I $G(SEQ)="" S SEQ=1  K SEG S SEG=""  ;  ; Get Emergency Contact Info  N ENAME,EREL,EADDR,EPHONE  D GETEINFO(DFN,.ENAME,.EREL,.EADDR,.EPHONE)  ;  ; Build Segment  I ENAME="",EREL="",'$$HASADDR(.EADDR),EPHONE="" Q ; No Data - Don't build segment  D SET^HLOAPI(.SEG,"ZCT",0) ; ZCT-0 ("ZCT")  D SET^HLOAPI(.SEG,SEQ,1) ; ZCT-1 (Repetition)  D SETXPN^HLOAPI4(.SEG,.ENAME,3) ; ZCT-3 (Emergency Contact Name)  D SET^HLOAPI(.SEG,EREL,4) ; ZCT-4 (Emergency Contact Relation to Patient)  D SETAD^HLOAPI4(.SEG,.EADDR,5) ; ZCT-5 (Emergency Contact Address)  D SET^HLOAPI(.SEG,EPHONE,6) ; ZCT-6 (Emergency Contact Phone)  Q  ZEL(SEQ,SEG,ELIGCODE,ELIGSTAT,COMBSTAT,COMBDATE) ; Builds a ZEL segment  ; SEQ - Segment Repetition index  ; SEG - Segment output  ; ELIGCODE - Eligibility code (File 8)  ; ELIGSTAT - Eligibility status  ; COMBSTAT - Combat service indicated?  ; COMBDATE - Combat veteran end date  ;  I $G(ELIGCODE)="",$G(COMBSTAT)="" Q  I $G(SEQ)="" S SEQ=1  K SEG S SEG=""  ;  ; Build Segment  D SET^HLOAPI(.SEG,"ZEL",0) ; ZEL-0 ("ZEL")  D SET^HLOAPI(.SEG,SEQ,1) ; ZEL-1 (Repetition)  I $G(ELIGCODE)'="" D SET^HLOAPI(.SEG,ELIGCODE,2) ; ZEL-2 (Eligibility Code)  I $G(ELIGSTAT)'="" D SET^HLOAPI(.SEG,ELIGSTAT,10) ; ZEL-10 (Eligibility Status)  I $G(COMBSTAT)'="" D SET^HLOAPI(.SEG,COMBSTAT,37) ; ZEL-37 (Combat Veteran Eligibility Indicator)  I $G(COMBDATE)'="" D SET^HLOAPI(.SEG,COMBDATE,38) ; ZEL-38 (Combat Veteran Eligibility End Date)  Q  ZEN(ENR,SEQ,SEG) ; Builds a ZEN segment  ; ENR - Enrollment (File 27.11)  ; SEQ - Segment Repetition index  ; SEG - Segment output  ;  I $G(ENR)="" Q  I $G(SEQ)="" S SEQ=1  K SEG S SEG=""  ;  ; Get Enrollment Information  N ENSTATUS,ENPRIORI,ENSUBGRP  D GETENR(ENR,.ENSTATUS,.ENPRIORI,.ENSUBGRP)  ;  ; Build Segment  I ENSTATUS="",ENPRIORI="",ENSUBGRP="" Q ; No Data - Don't build segment  D SET^HLOAPI(.SEG,"ZEN",0) ; ZEN-0 ("ZEN")  D SET^HLOAPI(.SEG,SEQ,1) ; ZEN-1 (Repetition)  D SET^HLOAPI(.SEG,ENSTATUS,4) ; ZEN-4 (Enrollment Status)  D SET^HLOAPI(.SEG,ENPRIORI,9) ; ZEN-9 (Enrollment Priority)  D SET^HLOAPI(.SEG,ENSUBGRP,13) ; ZEN-13 (Enrollment Subgroup)  Q  ZSP(DFN,SEQ,SEG) ; Builds a ZSP segment if Service Connected  ; DFN - VistA Internal Entry Number (IEN)  ; SEQ - Segment Repetition index  ; SEG - Segment output  ;  I $G(DFN)="" Q  I $G(SEQ)="" S SEQ=1  K SEG S SEG=""  ;  ; Get Service Connected Information  N ISSERCON,SERCONPC,DENTCLAS  D GETSERVC(DFN,.ISSERCON,.SERCONPC,.DENTCLAS)  ;  ; Build Segment  I ISSERCON="",SERCONPC="",DENTCLAS="" Q ; No Data - Don't build segment  D SET^HLOAPI(.SEG,"ZSP",0) ; ZSP-0 ("ZSP")  D SET^HLOAPI(.SEG,SEQ,1) ; ZSP-1 (Repetition)  D SET^HLOAPI(.SEG,ISSERCON,2) ; ZSP-2 (Service Connected?)  D SET^HLOAPI(.SEG,SERCONPC,3) ; ZSP-3 (Service Connected Percentage)  D SET^HLOAPI(.SEG,DENTCLAS,12) ; ZSP-12 (Dental Classification)  Q  BLDPID3(SEG,ICN,DFN,FIELD) ; Builds patient identifier list  ; SEG - Segment output  ; ICN - Integration Control Number (ICN)  ; DFN - VistA Internal Entry Number (IEN)  ; FIELD - Field Index  ;  ; Internal Entry Number (IEN)  N ASSIGNING  S ASSIGNING=$$GET^XPAR("SYS","DGMX PATIENT IEN ASSIGNING") ; Get configurable HL7 Assigning Authority  I $G(ASSIGNING)="" S ASSIGNING="USVHA" ; Default to "USVHA" assigning authority  D SET^HLOAPI(.SEG,DFN,FIELD,1,1,1)  D SET^HLOAPI(.SEG,ASSIGNING,FIELD,4,1,1)  D SET^HLOAPI(.SEG,"IEN",FIELD,5,1,1)  ;  ; Integration Control Number (ICN)  I $P(ICN,"^",1)=-1 Q ; No ICN  D SET^HLOAPI(.SEG,ICN,FIELD,1,1,2)  D SET^HLOAPI(.SEG,"USVHA",FIELD,4,1,2)  D SET^HLOAPI(.SEG,"NI",FIELD,5,1,2)  Q  BLDPID9(SEG,ALIAS,FIELD) ; Builds patient aliases  ; SEG - Segment output  ; ALIAS - "^" delimited list of patient aliases  ; FIELD - Field Index  ;  N REP,LEN,IND,VAL  S REP=1  S LEN=$L(ALIAS,"^")  ;  ; Loop through aliases  F IND=1:1:LEN D  . S VAL=$P(ALIAS,"^",IND)  . I VAL="NONE" Q  . D STDNAME^XLFNAME(.VAL,"C") ; Format Alias  . D SETXPN^HLOAPI4(.SEG,.VAL,FIELD,"",REP) ; Build repetition  . S REP=REP+1  Q  BLDPID11(SEG,PERMADDR,CONFADDR,TEMPADDR,FIELD) ; Builds patient addresses  ; SEG - Segment output  ; PERMADDR - Patient permanent address  ; CONFADDR - Patient confidential address  ; TEMPADDR - Patient temporary address  ; FIELD - Field Index  ;  N REP  S REP=1  I $$HASADDR(.PERMADDR) D SETAD^HLOAPI4(.SEG,.PERMADDR,FIELD,"",REP) S REP=REP+1  I $$HASADDR(.TEMPADDR) D  . D SETAD^HLOAPI4(.SEG,.TEMPADDR,FIELD,"",REP)  . D SET^HLOAPI(.SEG,TEMPADDR("STARTDT"),FIELD,13,"",REP) ; Start Date  . D SET^HLOAPI(.SEG,TEMPADDR("ENDDT"),FIELD,14,"",REP) ; End Date  . S REP=REP+1  I $$HASADDR(.CONFADDR) D SETAD^HLOAPI4(.SEG,.CONFADDR,FIELD,"",REP)  Q  BLDPID13(SEG,HPHONE,HPEXT,EMAIL,TPHONE,TPEXT,FIELD) ; Builds patient home telecoms  ; SEG - Segment output  ; HPHONE - Home phone  ; HPEXT - Home phone extension  ; EMAIL - Email address  ; TPHONE - Temporary phone (send instead of the permanent phone, if active.)  ; TPEXT - Temporary phone extension  ; FIELD - Field Index  ;  D  . I $G(TPHONE)'="" D Q ; Temp Phone  . . D SET^HLOAPI(.SEG,TPHONE,FIELD,1) ; PID-13.1 (Phone Number)  . . D SET^HLOAPI(.SEG,"T",FIELD,2) ; PID-13.2 (Phone Use Code)  . . I $G(TPEXT)'="" D SET^HLOAPI(.SEG,TPEXT,FIELD,8) ; PID-13.8 (Phone Extension)  . I $G(HPHONE)'="" D Q ; Home Phone  . . D SET^HLOAPI(.SEG,HPHONE,FIELD,1) ; PID-13.1 (Phone Number)  . . D SET^HLOAPI(.SEG,"P",FIELD,2) ; PID-13.2 (Phone Use Code)  . . I $G(HPEXT)'="" D SET^HLOAPI(.SEG,HPEXT,FIELD,8) ; PID-13.8 (Phone Extension)  I $G(EMAIL)'="" D SET^HLOAPI(.SEG,EMAIL,FIELD,4) ; PID-13.4 (Email)  Q  BLDPID14(SEG,WPHONE,WPEXT,FIELD) ; Builds patient work telecoms  ; SEG - Segment output  ; WPHONE - Work phone  ; WPEXT - Work phone extension  ; FIELD - Field Index  ;  I $G(WPHONE)="" Q ; Work Phone  D SET^HLOAPI(.SEG,WPHONE,FIELD,1) ; PID-14.1 (Phone Number)  I $G(WPEXT)'="" D SET^HLOAPI(.SEG,WPEXT,FIELD,8) ; PID-14.8 (Phone Extension)  Q  BLDZEL(DFN,SEG,MSG,ONLYPRIM) ; Builds eligibility information in ZEL segments.  ; These segments can repeat, but the first repetition  ; contains the primary eligibility.  ; DFN - VistA Internal Entry Number (IEN) - Used by ELIG^VADPT  ; SEG - Segment output  ; MSG - Message output  ; ONLYPRIM - Set to 1 to only build the primary eligibility  ;  ; Get Eligibility and Combat Eligibility information  N PECODE,PESTAT,SEQ,LINE,VAHOW,VAEL,VAL,COMBSTAT,COMBDATE  S SEQ=1,LINE=0,VAHOW=0  D ELIG^VADPT  D GETCOMB(DFN,.COMBSTAT,.COMBDATE)  ;  ; Primary Eligibility  S PECODE=$P(VAEL(1),"^",2) ; Primary Eligibility Code  S PESTAT=$P(VAEL(8),"^",2) ; Primary Eligibility Status  D ZEL(SEQ,.SEG,PECODE,PESTAT,COMBSTAT,COMBDATE)  I '$$ADDSEG^HLOAPI(.MSG,.SEG) Q  I $G(ONLYPRIM)=1 Q ; Don't build secondary eligibilites  ;  ; Secondary Eligibility  F S LINE=$O(VAEL(1,LINE)) Q:(LINE="") D  . S VAL=$P(VAEL(1,LINE),"^",2)  . I VAL=PECODE Q ; Do not build primary eligibility again  . S SEQ=SEQ+1  . D ZEL(SEQ,.SEG,VAL,"")  . I '$$ADDSEG^HLOAPI(.MSG,.SEG) Q  Q  GETALIAS(DFN) ; Get Patient Aliases  ; DFN - VistA Internal Entry Number (IEN)  ; LIST - Output contains patient aliases in "^" delimited string. "NONE" if no aliases.  ;  N LINE,LIST,NAME  S LINE=0,LIST=""  ;  ; Get the patient aliases  D GETS^DIQ(2,DFN,"1\*","E","ALIAS")  ;  ; Loop through and build return string  F S LINE=$O(ALIAS(2.01,LINE)) Q:LINE="" D  . S NAME=$G(ALIAS(2.01,LINE,.01,"E"))  . I NAME="" Q  . I LIST="" S LIST=NAME  . E S LIST=LIST\_"^"\_NAME  I LIST="" S LIST="NONE"  Q LIST  GETDEMOG(DFN,NAME,DOB,SSN,SEX,ICN,DOD,ALIAS) ; Get Patient Demographics  ; DFN - VistA Internal Entry Number (IEN)  ; NAME - Patient name  ; DOB - Patient date of birth  ; SSN - Patient SSN  ; SEX - Patient sex  ; ICN - Integration Control Number (ICN)  ; DOD - Patient date of death  ; ALIAS - Patient aliases  ;  N VAHOW,VADM  S VAHOW=1  D DEM^VADPT  S NAME=VADM("NM") ; Name  D STDNAME^XLFNAME(.NAME,"C") ; Format the Name  S NAME("SECOND")=$G(NAME("MIDDLE")) ; Format the Middle Name  S ALIAS=$$GETALIAS(DFN) ; Alias(es)  S DOB=$P(VADM("DB"),"^") ; Date of Birth  S SSN=$P(VADM("SS"),"^") ; SSN  S SEX=$P(VADM("SX"),"^") ; Gender  S ICN=$P($$GETICN^MPIF001(DFN),"V") ; Integration Control Number (ICN)  S DOD=$P(VADM("EX"),"^") ; Date of Death  Q  GETADDR(DFN,PADDR,CADDR,TADDR,USEPERM) ; Get Patient Addresses given a DFN  ; DFN - VistA Internal Entry Number (IEN)  ; PADDR - Patient permanent address  ; CADDR - Patient confidential address  ; TADDR - Patient temporary address  ; USEPERM - Use permanent address flag  ;  N VAHOW,VAPA,TADSTDT,TADENDT,TADFLAG  S VAHOW=""  K USEPERM  D ADD^VADPT ; Populates VAPA array  ;  ; Temporary Address Checks  S TADSTDT=$$GET1^DIQ(2,DFN,.1217,"I") ; Temp Address Start Date  S TADENDT=$$GET1^DIQ(2,DFN,.1218,"I") ; Temp Address End Date  S TADFLAG=$$GET1^DIQ(2,DFN,.12105,"I") ; Temp Address Active?  S TADSTDT=$G(TADSTDT),TADENDT=$G(TADENDT),TADFLAG=$G(TADFLAG) ; Sanitize variables  ;  D ; Waterfall logic to determine whether or not to use the permanent address (Comes from 3^VADPT1)  . I $D(VAPA("P")) S USEPERM=1 Q ; Permanent address flag set - Permanent  . I '$D(^DPT(DFN,.121)) S USEPERM=1 Q ; No temporary address - Permanent  . I TADFLAG="N" S USEPERM=1 Q ; Temp address not active - Permanent  . I TADSTDT="" S USEPERM=1 Q ; No start date - Permanent  . I TADSTDT>DT S USEPERM=1 Q ; Start date AFTER today? - Permanent  . I TADENDT="" S USEPERM=0 Q ; No end date - Temporary  . I TADENDT<DT S USEPERM=1 Q ; End date BEFORE today? - Permanent  ;  ; Build Address Arrays  I USEPERM D ; Permanent Address  . I $$GET1^DIQ(2,DFN,.121)'="" Q ; Don't send if marked as a bad address  . S PADDR("STREET1")=VAPA(1)  . S PADDR("STREET2")=VAPA(2)  . S PADDR("STREET3")=VAPA(3)  . S PADDR("CITY")=VAPA(4)  . S PADDR("STATE")=$$GET1^DIQ(5,$P(VAPA(5),"^",1),1)  . S PADDR("ZIP")=VAPA(6)  . S PADDR("TYPE")="P"  E D ; Temporary Address  . S TADDR("STREET1")=VAPA(1)  . S TADDR("STREET2")=VAPA(2)  . S TADDR("STREET3")=VAPA(3)  . S TADDR("CITY")=VAPA(4)  . S TADDR("STATE")=$$GET1^DIQ(5,$P(VAPA(5),"^",1),1)  . S TADDR("ZIP")=VAPA(6)  . S TADDR("TYPE")="T"  . S TADDR("STARTDT")=$$VA2HL7TS^SDMXCORE(TADSTDT)  . S TADDR("ENDDT")=$$VA2HL7TS^SDMXCORE(TADENDT)  ; Confidential Address  S CADDR("STREET1")=VAPA(13)  S CADDR("STREET2")=VAPA(14)  S CADDR("STREET3")=VAPA(15)  S CADDR("CITY")=VAPA(16)  S CADDR("STATE")=$$GET1^DIQ(5,$P(VAPA(17),"^",1),1)  S CADDR("ZIP")=$P(VAPA(18),"^")  S CADDR("TYPE")="C"  Q  GETTELE(DFN,HPHONE,HPEXT,WPHONE,WPEXT,TPHONE,TPEXT,EMAIL,USEPERM) ; Get Patient Telecommunication information  ; DFN - VistA Internal Entry Number (IEN)  ; HPHONE - Home Phone  ; HPEXT - Home Phone Extension  ; WPHONE - Work Phone  ; WPEXT - Work Phone Extension  ; TPHONE - Temporary Phone  ; TPEXT - Temporary Phone Extension  ; EMAIL - Email Address  ; USEPERM - Flag that the temp phone is active  ;  N OK  S HPHONE=$$GET1^DIQ(2,DFN,.131) ; Home Phone  S OK=$$FMTPHONE^SDMXCORE(.HPHONE,.HPEXT)  S WPHONE=$$GET1^DIQ(2,DFN,.132) ; Work Phone  S OK=$$FMTPHONE^SDMXCORE(.WPHONE,.WPEXT)  S TPHONE=$S($G(USEPERM)'=1:$$GET1^DIQ(2,DFN,.1219),1:"") ; Temporary Phone  S OK=$$FMTPHONE^SDMXCORE(.TPHONE,.TPEXT)  S EMAIL=$$GET1^DIQ(2,DFN,.133) ; Email  Q  GETEINFO(DFN,ENAME,EREL,EADDR,EPHONE) ; Gets the patient's emergency contact information  ; DFN - VistA Internal Entry Number (IEN)  ; ENAME - Emergency contact name  ; EREL - Emergency contact relation to patient  ; EADDR - Emergency contact address  ; EPHONE - Emergency contact phone number  ;  N STATE,EPEXT,OK  S ENAME=$$GET1^DIQ(2,DFN,.331) ; Name  D STDNAME^XLFNAME(.ENAME,"C") ; Format Name  S ENAME("SECOND")=$G(ENAME("MIDDLE"))  S EREL=$$GET1^DIQ(2,DFN,.332) ; Relation  S EPHONE=$$GET1^DIQ(2,DFN,.339) ; Phone  S OK=$$FMTPHONE^SDMXCORE(.EPHONE,.EPEXT) ; Format Phone  I $G(EPEXT)'="" S EPHONE=EPHONE\_"x"\_EPEXT ; Append extension, if specified  ;  ; Address  S EADDR("STREET1")=$$GET1^DIQ(2,DFN,.333)  S EADDR("STREET2")=$$GET1^DIQ(2,DFN,.334)  S EADDR("STREET3")=$$GET1^DIQ(2,DFN,.335)  S EADDR("CITY")=$$GET1^DIQ(2,DFN,.336)  S STATE=$P($G(^DPT(DFN,.33)),"^",7)  I STATE'="" S EADDR("STATE")=$P($G(^DIC(5,STATE,0)),"^",2)  S EADDR("ZIP")=$$GET1^DIQ(2,DFN,.338)  Q  GETSERVC(DFN,ISSERCON,SERCONPC,DENTCLAS) ; Gets the patient's service connected information  ; DFN - VistA Internal Entry Number (IEN)  ; ISSERCON - Patient service connected?  ; SERCONPC - Service connected percentage  ; DENTCLAS - Patient dental class  ;  N VAHOW,VAEL  S VAHOW=0 D ELIG^VADPT  S ISSERCON=$S($P(VAEL(3),"^",1)=1:"Y",1:"N")  S SERCONPC=$P(VAEL(3),"^",2)  S DENTCLAS=$$GET1^DIQ(2,DFN,220)  Q  GETENR(ENR,ENSTATUS,ENPRIORI,ENSUBGRP) ; Gets the enrollment information  ; DFN - VistA Internal Entry Number (IEN)  ; ENSTATUS - Enrollment status  ; ENPRIORI - Enrollment priority  ; ENSUBGRP - Enrollment subgroup  ;  I $G(ENR)="" Q  S ENSTATUS=$$GET1^DIQ(27.11,ENR,.04)  S ENPRIORI=$$GET1^DIQ(27.11,ENR,.07)  S ENSUBGRP=$$GET1^DIQ(27.11,ENR,.12)  Q  GETCOMB(DFN,COMBSTAT,COMBDATE) ; Gets the combat service eligibility information  ; DFN - VistA Internal Entry Number (IEN)  ; COMBSTAT - Combat service indicated?  ; COMBDATE - Combat veteran end date  ;  I $G(DFN)="" Q  S COMBSTAT=$$GET1^DIQ(2,DFN,.5291)  S COMBDATE=$$VA2HL7TS^SDMXCORE($$GET1^DIQ(2,DFN,.5295,"I"))  Q  HASADDR(ADDR) ; Checks nodes of an address array for data.  ; ADDR - Address in standard array structure  ;  I $G(ADDR("STREET1")) Q 1  I $G(ADDR("STREET2")) Q 1  I $G(ADDR("STREET3")) Q 1  I $G(ADDR("CITY")) Q 1  I $G(ADDR("ZIP")) Q 1  I $G(ADDR("STATE")) Q 1  Q 0 ; No data - Return 0  SETXCN(SEG,DFN,FIELD) ; Sets an XCN field in an HLO message  ; SEG - HLO Segment  ; DFN - VistA Internal Entry Number (IEN)  ; FIELD - Field number  ;  N NAME  S NAME=$$GET1^DIQ(200,DFN,.01)  D STDNAME^XLFNAME(.NAME,"C")  D SET^HLOAPI(.SEG,DFN,FIELD,1)  D SET^HLOAPI(.SEG,$G(NAME("FAMILY")),FIELD,2)  D SET^HLOAPI(.SEG,$G(NAME("GIVEN")),FIELD,3)  D SET^HLOAPI(.SEG,$G(NAME("SECOND")),FIELD,4)  D SET^HLOAPI(.SEG,$G(NAME("SUFFIX")),FIELD,5)  D SET^HLOAPI(.SEG,$G(NAME("PREFIX")),FIELD,6)  D SET^HLOAPI(.SEG,$G(NAME("DEGREE")),FIELD,7)  Q  ; | | | | | | | | | |

Table : DGMXORU Routine

| Routines | Activities | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Routine Name | DGMXORU | | | | | | | |
| Enhancement Category | New | | Modify | Delete | | No Change | | |
| RTM | BN1A6.1  BN1A6.3  BN1E1.2  BN2A1  BN2A1.1  BN2A1.2  BN2B1  BN2B1.1  BN2B1.6  BN2B1.10  BN3A1.7  BN3B1.8  BN4A1.11  BN4B1.4 | | | | | | | |
| Related Options |  | | | | | | | |
| Related Routines | Routines “Called By” | | | | Routines “Called” | | | |
|  | Cross-references type MUMPS in file #38.1 field #.01 and file #41.41 field #1 | | | | DGMXHL7  DGPFAA  DIQ  HLOAPI  HLOAPI1 | | | |
| Routines | Activities | | | | | | | |
| Data Dictionary (DD) References | ^DGPF(26.13, | | | | | | | |
| Related Protocols | N/A | | | | | | | |
| Related Integration Control Registrations (ICRs) | 4716,4717 | | | | | | | |
| Data Passing | Input | Output Reference | | | Both | | Global Reference | Local | |
| Input Attribute Name and Definition |  | | | | | | | |
| Output Attribute Name and Definition |  | | | | | | | |
| Current Logic | | | | | | | | |
| N/A | | | | | | | | |
| Modified Logic (Changes are in bold) | | | | | | | | |
| DGMXORU ; MASS/DXL - ORU Trigger/Build Code;10/05/17  ;;5.3;Registration;\*\*948\*\*;OCTOBER 05,2017  ;VA DIRECTIVE 6402, this routine should not be modified.  ;  ; ICR# Supported References  ; $$GETALL^DGPFAA  ; $$FMTE^DILIBF  ; 2056 $$GET1^DIQ  ; 2056 GETS^DIQ  ; 4716 $$ADDSEG^HLOAPI  ; 4716 $$NEWMSG^HLOAPI  ; 4717 $$SENDONE^HLOAPI1  ;  Q  TRIGBAD(DFN,STATUS) ; Wrapper function for Bad address indicator  ; DFN - VistA Internal Entry Number (IEN)  ; STATUS - Active or inactive flag  ;  I ($G(DFN)="")!($G(STATUS)="") Q  N TRIGDATA  S TRIGDATA("STATUS")=STATUS  D TRIGFLAG(DFN,"BAD",.TRIGDATA)  Q  TRIGPRF(FLAGID) ; Wrapper function for patient record flag  ; FLAGID - Identifier of the patient record flag (Field .001 in File 26.13)  ;  N DFN  S DFN=$P(^DGPF(26.13,FLAGID,0),"^",1) ; Get the patient DFN  I $G(DFN)="" Q  D TRIGFLAG(DFN,"PRF")  Q  TRIGSEN(DFN,STATUS) ; Wrapper function for patient sensitivity flag  ; DFN - VistA Internal Entry Number (IEN)  ; STATUS - Active or inactive flag  ;  I $G(DFN)="" Q  N TRIGDATA  S TRIGDATA("STATUS")=STATUS  D TRIGFLAG(DFN,"SEN",.TRIGDATA)  Q  TRIGVER(DFN) ; Wrapper function for patient demographics verification date  ; DFN - VistA Internal Entry Number (IEN)  ;  I $G(DFN)="" Q  N PDFN,TRIGDATA  S PDFN=$$GET1^DIQ(41.41,DFN,.01,"I") ; Get the patient DFN  S TRIGDATA("VERDFN")=DFN  D TRIGFLAG(PDFN,"VER",.TRIGDATA)  Q  TRIGFLAG(DFN,TYPE,TRIGDATA) ; Triggers an ORU message containing patient flag information.  ; This currently includes Patient Record Flags, Bad Address Indicators, Patient Sensitivity,  ; and Demographic Verification Date.  ; DFN - VistA Internal Entry Number (IEN)  ; TYPE - Type of flag ("PRF","BAD","SEN","VER")  ; TRIGDATA - Array containing parameters for the build code. Nodes include:  ; ("STATUS") - For Bad Address/Sensitivity - Active or inactive flag  ; ("VERDFN") - For Verification - DFN for file 41.41  ;  I $G(DFN)="" Q  I $G(TYPE)="" Q  N SEG,MSG,PARAMS,ERROR,WHOTO,OK  S PARAMS("MESSAGE TYPE")="ORU"  S PARAMS("EVENT")="R01"  S PARAMS("SENDING APPLICATION")="DG-REG-OUT"  S WHOTO("RECEIVING APPLICATION")="MASS"  S WHOTO("FACILITY LINK NAME")="EPICADTIN2" ; HL Logical Link  ;  ; Start new message  I '$$NEWMSG^HLOAPI(.PARAMS,.MSG) Q  D PID^DGMXHL7(DFN,1,.SEG) ; Build PID (required)  I '$$ADDSEG^HLOAPI(.MSG,.SEG) Q  I TYPE="PRF" D BLDPRF(DFN,.SEG,.MSG) ; Patient record flags  I TYPE="BAD" D BLDBAD(DFN,.SEG,.MSG,.TRIGDATA) ; Bad address indicator  I TYPE="SEN" D BLDSEN(DFN,.SEG,.MSG,.TRIGDATA) ; Patient sensitivity flag  I TYPE="VER" D BLDVER(DFN,.SEG,.MSG,.TRIGDATA) ; Demographics verification date  ;  ; Send message  S OK=$$SENDONE^HLOAPI1(.MSG,.PARAMS,.WHOTO,.ERROR)  Q  BLDPRF(DFN,SEG,MSG) ; Build patient record flags in OBR-OBX segments  ; DFN - VistA Internal Entry Number (IEN)  ; SEG - Segment output  ; MSG - Message output  ;  N IND,FLAGS,OK,LINE,NARR,TEXT,SUMMARY,OBRIND,OBXIND,STATUS  S IND=0  I '$$GETALL^DGPFAA(DFN,.FLAGS,"","") Q ; National and Local Flags  ;  S OBRIND=1  F S IND=$O(FLAGS(IND)) Q:IND="" D ; Loop Through Patient Flags  . ;  . ; Get Flag Info  . S SUMMARY=$$GET1^DIQ(26.13,IND,.02) ; Summary  . S STATUS=$$GET1^DIQ(26.13,IND,.03) ; Status  . D GETS^DIQ(26.13,IND,"1","","NARR") ; Narrative Text  . ;  . ; Build OBR Segment  . D OBR^DGMXHL7(OBRIND,.SEG,SUMMARY)  . I '$$ADDSEG^HLOAPI(.MSG,.SEG) Q  . S OBRIND=OBRIND+1  . ;  . ; Build Status OBX  . S OBXIND=1  . D OBX^DGMXHL7(OBXIND,.SEG,"STATUS",STATUS)  . S OK=$$ADDSEG^HLOAPI(.MSG,.SEG)  . ;  . ; Build Narrative OBX(s)  . S LINE=0,OBXIND=OBXIND+1  . F S LINE=$O(NARR(26.13,IND\_",",1,LINE)) Q:LINE="" D  . . S TEXT=NARR(26.13,IND\_",",1,LINE)  . . D OBX^DGMXHL7(OBXIND,.SEG,"NARRATIVE",TEXT)  . . S OK=$$ADDSEG^HLOAPI(.MSG,.SEG)  . . S OBXIND=OBXIND+1  Q  BLDBAD(DFN,SEG,MSG,TRIGDATA) ; Build bad address flag in OBR-OBX segments  ; DFN - VistA Internal Entry Number (IEN)  ; SEG - Segment output  ; MSG - Message output  ; TRIGDATA - Parameter array containing the action  ; ("STATUS") - Active or inactive status  ;  N OK  D OBR^DGMXHL7(1,.SEG,"BAD ADDRESS") ; Build OBR  S OK=$$ADDSEG^HLOAPI(.MSG,.SEG)  D OBX^DGMXHL7(1,.SEG,"STATUS",$G(TRIGDATA("STATUS"))) ; Build Status OBX  S OK=$$ADDSEG^HLOAPI(.MSG,.SEG)  D OBX^DGMXHL7(2,.SEG,"NARRATIVE",$$GET1^DIQ(2,DFN,.121)) ; Build Narrative OBX  S OK=$$ADDSEG^HLOAPI(.MSG,.SEG)  Q  BLDSEN(DFN,SEG,MSG,TRIGDATA) ; Build patient sensitivity flag in OBR-OBX segments  ; DFN - VistA Internal Entry Number (IEN)  ; SEG - Segment output  ; MSG - Message output  ; TRIGDATA - Parameter array containing the action  ; ("STATUS") - Active or inactive status  ;  N OK  D OBR^DGMXHL7(1,.SEG,"SENSITIVITY") ; Build OBR  S OK=$$ADDSEG^HLOAPI(.MSG,.SEG)  D OBX^DGMXHL7(1,.SEG,"STATUS",$G(TRIGDATA("STATUS"))) ; Build Status OBX  S OK=$$ADDSEG^HLOAPI(.MSG,.SEG)  Q  BLDVER(DFN,SEG,MSG,TRIGDATA) ; Builds the patient demographics verification date in OBR-OBX  ; DFN - VistA Internal Entry Number (IEN) - Used by ELIG^VADPT  ; SEG - Segment output  ; MSG - Message output  ; TRIGDATA - Parameter array containing the verification DFN  ; ("VERDFN") - DFN of the Verification record in file 41.41  ;  N OK,VERDATE,NARR  S VERDATE=$$FMTE^DILIBF($P($$GET1^DIQ(41.41,$G(TRIGDATA("VERDFN")),1,"I"),".",1),2) ; Format Verification date (MM/DD/YY)  I $G(VERDATE)="" Q  S NARR="Pre-Register Date: "\_VERDATE  D OBR^DGMXHL7(1,.SEG,"DEMOGRAPHIC VERIFICATION") ; Build OBR  S OK=$$ADDSEG^HLOAPI(.MSG,.SEG)  D OBX^DGMXHL7(1,.SEG,"STATUS","ACTIVE") ; Build Status OBX (Always send ACTIVE)  S OK=$$ADDSEG^HLOAPI(.MSG,.SEG)  D OBX^DGMXHL7(2,.SEG,"NARRATIVE",NARR) ; Build Narrative OBX  S OK=$$ADDSEG^HLOAPI(.MSG,.SEG)  Q  ; | | | | | | | | |

Table : DGMXVLD Routine

| Routines | Activities | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Routine Name | DGMXVLD | | | | | | | | |
| Enhancement Category | New | | Modify | Delete | | | No Change | | |
| RTM | BN1A6.1  BN1A6.3  BN1E1.2  BN2A1  BN2A1.1  BN2A1.2  BN2B1  BN2B1.1  BN2B1.6  BN2B1.10  BN3A1.7  BN3B1.8  BN4A1.11  BN4B1.4 | | | | | | | | |
| Related Options |  | | | | | | | | |
| Related Routines | Routines “Called By” | | | | | Routines “Called” | | | |
|  | SDMXSCHI | | | | | DGMXHL7  DILIBF  HLOPRS  SDMXCORE  XLFSTR | | | |
| Routines | Activities | | | | | | | | |
| Data Dictionary (DD) References | NONE | | | | | | | | |
| Related Protocols | N/A | | | | | | | | |
| Related Integration Control Registrations (ICRs) |  | | | | | | | | |
| Data Passing | Input | Output Reference | | | Both | | | Global Reference | Local |
| Input Attribute Name and Definition |  | | | | | | | | |
| Output Attribute Name and Definition |  | | | | | | | | |
| Current Logic | | | | | | | | | |
| N/A | | | | | | | | | |
| Modified Logic (Changes are in bold) | | | | | | | | | |
| DGMXVLD ; MASS/DXL - Incoming Patient Validation;10/05/17  ;;5.3;Registration;\*\*948\*\*;OCTOBER 05,2017  ;VA DIRECTIVE 6402, this routine should not be modified.  ;  ; ICR# Supported References  ; $$FMTE^DILIBF  ; 4718 $$GET^HLOPRS  ; $$STRIP^XLFSTR  ; $$UP^XLFSTR  ;  Q  VLDPAT(PID,DFN) ; Validates patient information from an incoming HL7 message. Returns '1' for valid, '0' for invalid.  ; PID - Contains the PID segment from the HL7 message (in standard HLO array)  ; DFN - VistA Internal Entry Number (IEN)  ;  N NAME,FNAME,LNAME,DOB,SEX,SSN,MSGFNAME,MSGLNAME,MSGDOB,MSGSEX,MSGSSN,ERRLIST,RET  S RET=1  D GETDEMOG^DGMXHL7(DFN,.NAME,.DOB,.SSN,.SEX) ; Get internal patient demographics  S LNAME=$P(NAME,",",1) ; Piece internal name  S FNAME=$P(NAME,",",2)  D GETPID(.PID,.MSGLNAME,.MSGFNAME,.MSGDOB,.MSGSEX,.MSGSSN) ; Get demographics from PID  ;  ; Sanitize variables  S LNAME=$G(LNAME),MSGLNAME=$G(MSGLNAME)  S FNAME=$G(FNAME),MSGFNAME=$G(MSGFNAME)  S DOB=$G(DOB),MSGDOB=$G(MSGDOB)  S SEX=$G(SEX),MSGSEX=$G(MSGSEX)  S SSN=$G(SSN),MSGSSN=$G(MSGSSN)  ;  ; Check demographics  I (LNAME'=""),(MSGLNAME'=""),($E(LNAME,1,3)'=$E(MSGLNAME,1,3)) D BLDERR(.ERRLIST,"Last Name",LNAME,MSGLNAME) ; Last Name (first three characters)  I (FNAME'=""),(MSGFNAME'=""),($E(FNAME,1,3)'=$E(MSGFNAME,1,3)) D BLDERR(.ERRLIST,"First Name",FNAME,MSGFNAME) ; First Name (first three characters)  I (DOB'=""),(MSGDOB'=""),(DOB'=MSGDOB) D BLDERR(.ERRLIST,"DOB",$$FMTE^DILIBF(DOB),$$FMTE^DILIBF(MSGDOB)) ; DOB (readable format)  I (SEX'=""),(MSGSEX'=""),(SEX'=MSGSEX) D BLDERR(.ERRLIST,"Gender",SEX,MSGSEX) ; Sex  I (SSN'=""),(MSGSSN'=""),(SSN'=MSGSSN) D BLDERR(.ERRLIST,"SSN",SSN,MSGSSN) ; Social Security Number  I $G(ERRLIST)'="" D ERRLOG^SDMXERRO(313,ERRLIST,1) S RET=0  Q RET  GETPID(PID,MSGLNAME,MSGFNAME,MSGDOB,MSGSEX,MSGSSN) ; Pulls demographic data from the PID segment on an incoming message  ; PID - Contains the PID segment from the HL7 message (in standard HLO array)  ; MSGLNAME - Last name from the message  ; MSGFNAME - First name from the message  ; MSGDOB - Date of birth from the message  ; MSGSEX - Sex from the message  ; MSGSSN - Social security number from the message  ;  ; Get data from PID  S MSGLNAME=$$GET^HLOPRS(.PID,5,1) ; PID-5.1 (Last Name)  S MSGFNAME=$$GET^HLOPRS(.PID,5,2) ; PID-5.2 (First Name)  S MSGDOB=$$GET^HLOPRS(.PID,7) ; PID-7 (Date of Birth)  S MSGSEX=$$GET^HLOPRS(.PID,8) ; PID-8 (Sex)  S MSGSSN=$$GET^HLOPRS(.PID,19) ; PID-19 (Social Security Number)  ;  ; Format data from PID  S MSGLNAME=$$UP^XLFSTR(MSGLNAME) ; First three characters (uppercase)  S MSGFNAME=$$UP^XLFSTR(MSGFNAME) ; First three characters (uppercase)  S MSGDOB=$$HL72VATS^SDMXCORE(MSGDOB) ; HL7 timestamp to internal time  S MSGSEX=$$UP^XLFSTR(MSGSEX)  S MSGSEX=$S(MSGSEX="MALE":"M",MSGSEX="FEMALE":"F",1:MSGSEX) ; "M" or "F" gender  S MSGSSN=$$STRIP^XLFSTR(MSGSSN,"-") ; Strip '-'s  Q  BLDERR(ERRLIST,TYPE,INTVAL,MSGVAL) ; Builds a string containing error information for patient validation  ; The format of the error text is repeating. Types include: "Last Name","First Name","DOB","Gender","SSN"  N DROW,DCOL  S DROW=" / "  S DCOL=":"  I $G(ERRLIST)="" S ERRLIST="Type"\_DCOL\_"Internal Value"\_DCOL\_"Msg Value"\_DROW  S ERRLIST=ERRLIST\_TYPE\_DCOL\_INTVAL\_DCOL\_MSGVAL\_DROW  Q  ; | | | | | | | | | |

Table : VAFCDD01 Routine

| Routines | Activities | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Routine Name | VAFCDD01 | | | | | | | |
| Enhancement Category | New | | Modify | Delete | | No Change | | |
| RTM | BN1A6.1  BN1A6.3  BN1E1.2  BN2A1  BN2A1.1  BN2A1.2  BN2B1  BN2B1.1  BN2B1.6  BN2B1.10  BN3A1.7  BN3B1.8  BN4A1.11  BN4B1.4 | | | | | | | |
| Related Options |  | | | | | | | |
| Related Routines | Routines “Called By” | | | | Routines “Called” | | | |
|  | VAFCA04  VAFCMGA  VAFCMS02  VAFCMSG  VAFCMSG0  VAFCOTFU  VAFCTFMF  VAFCTFU  VAFCTR | | | | DIE  VAFHPIVT  VAFHREG  VAFHUTL  VAFHUTL2 | | | |
| Routines | Activities | | | | | | | |
| Data Dictionary (DD) References | ^DD(391.71,  ^DPT(  ^VAT(391.71, | | | | | | | |
| Related Protocols | N/A | | | | | | | |
| Related Integration Control Registrations (ICRs) |  | | | | | | | |
| Data Passing | Input | Output Reference | | | Both | | Global Reference | Local | |
| Input Attribute Name and Definition |  | | | | | | | |
| Output Attribute Name and Definition |  | | | | | | | |
| Current Logic | | | | | | | | |
| VAFCDD01 ;ALB/JRP,LTL-DATA DICTIONARY UTILITIES ;05-SEP-1996  ;;5.3;Registration;\*\*91,149\*\*;Jun 06, 1996;Build 5  ;Input : IFN - Pointer to entry in PATIENT file (#2)  ;Hard set fields edited  S:$G(VAFCF)&($P($G(^VAT(391.71,DA,2)),U)'[$G(VAFCF)) $P(^VAT(391.71,DA,2),U)=$P($G(^VAT(391.71,DA,2)),U)\_$G(VAFCF) | | | | | | | | |
| Modified Logic (Changes are in bold) | | | | | | | | |
| VAFCDD01;ALB/JRP,LTL-DATA DICTIONARY UTILITIES ;05-SEP-1996  ;;5.3;Registration;\*\*91,149,**948**\*\*;Jun 06, 1996;Build 5  ;Input : IFN - Pointer to entry in PATIENT file (#2)  **; Variable VAFCEPIC may be passed to update field**  **; ENTRY WAS CREATED BY MASS EPIC (.11) in the**  **; ADT/HL7 PIVOT file (#391.71)**  ;Hard set fields edited  S:$G(VAFCF)&($P($G(^VAT(391.71,DA,2)),U)'[$G(VAFCF)) $P(^VAT(391.71,DA,2),U)=$P($G(^VAT(391.71,DA,2)),U)\_$G(VAFCF)  **S $P(^VAT(391.71,DA,0),U,11)=+$G(VAFCEPIC)** | | | | | | | | |

Table : VAFCMSG Routine

| Routines | Activities | | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Routine Name | VAFCMSG | | | | | | | | | |
| Enhancement Category | New | | Modify | Delete | | | No Change | | | |
| RTM | BN1A6.1  BN1A6.3  BN1E1.2  BN2A1  BN2A1.1  BN2A1.2  BN2B1  BN2B1.1  BN2B1.6  BN2B1.10  BN3A1.7  BN3B1.8  BN4A1.11  BN4B1.4 | | | | | | | | | |
| Related Options |  | | | | | | | | | |
| Related Routines | Routines “Called By” | | | | | Routines “Called” | | | | |
|  | Taskman | | | | | DIE  VAFHPIVT  VAFHREG  VAFHUTL  VAFHUTL2 | | | | |
| Routines | Activities | | | | | | | | | |
| Data Dictionary (DD) References | ^DD(391.71,  ^DPT(  ^VAT(391.71, | | | | | | | | | |
| Related Protocols | N/A | | | | | | | | | |
| Related Integration Control Registrations (ICRs) |  | | | | | | | | | |
| Data Passing | Input | Output Reference | | | Both | | | Global Reference | Local |
| Input Attribute Name and Definition |  | | | | | | | | | |
| Output Attribute Name and Definition |  | | | | | | | | | |
| Current Logic | | | | | | | | | | |
| VAFCMSG ;ALB/JRP-BACKGROUND JOB TO TRANSMIT ENTRIES IN PIVOT FILE ; 3/2/04 12:54pm  ;;5.3;Registration;\*\*91,149,530,578\*\*;Jun 06, 1996;Build 5  N SEND  S SEND=$P($$SEND^VAFHUTL(),"^",2)  ;Send changes to patient's demographical data (ADT-A08)  D BCSTA08  ;Send changes to patient's treating facility list (MFU-M05)  D BCKTFMFU^VAFCTFMF  ;unlock global  .I ('$D(^DPT(DFN,0)))!($G(^DPT(DFN,-9))) D XMITFLAG^VAFCDD01(PIVOTPTR,"  ",1) Q  .;Store info into event information array  .I '$D(^DPT("B",$P(^DPT(DFN,0),U),DFN)) D XMITFLAG^VAFCDD01(PIVOTPTR,"",1) Q  .;Broadcast ADT-A04 message | | | | | | | | | | |
| Modified Logic (Changes are in bold) | | | | | | | | | | |
| VAFCMSG ;ALB/JRP-BACKGROUND JOB TO TRANSMIT ENTRIES IN PIVOT FILE ; 3/2/04 12:5  4pm  ;;5.3;Registration;\*\*91,149,530,578**,948**\*\*;Jun 06, 1996;Build 5  N **VAFCEPIC,**SEND  S **VAFCEPIC=0,**SEND=$P($$SEND^VAFHUTL(),"^",2)  ;Send changes to patient's demographical data (ADT-A08)  **D:'VAFCEPIC BCSTA08**  ;Send changes to patient's treating facility list (MFU-M05)  **D:'VAFCEPIC BCKTFMFU^VAFCTFMF**  ;unlock global  .I '$D(^DPT("B",$P(^DPT(DFN,0),U),DFN)) D XMITFLAG^VAFCDD01(PIVOTPTR,"",1) Q  **.D TRIGDEM^DGMXADT(DFN)**  **.;Check if the Pivot File entry was created by MASS/EPIC  .I $P(NODE,U,11)=1 S VAFCEPIC=1 D XMITFLAG^VAFCDD01(PIVOTPTR,"",1) Q**  .;Store info into event information array  .I '$D(^DPT("B",$P(^DPT(DFN,0),U),DFN)) D XMITFLAG^VAFCDD01(PIVOTPTR,"",1) Q  **.D TRIGDEM^DGMXADT(DFN)  .;Check if the Pivot File entry was created by MASS/EPIC  .I $P(NODE,U,11)=1 S VAFCEPIC=1 D XMITFLAG^VAFCDD01(PIVOTPTR,"",1) Q**  .;Broadcast ADT-A04 message | | | | | | | | | | |

Table : VAFHLZSP Routine

| Routines | Activities | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Routine Name | VAFHLZSP | | | | | | | | |
| Enhancement Category | New | | Modify | Delete | | | No Change | | |
| RTM | BN1A6.1  BN1A6.3  BN1E1.2  BN2A1  BN2A1.1  BN2A1.2  BN2B1  BN2B1.1  BN2B1.6  BN2B1.10  BN3A1.7  BN3B1.8  BN4A1.11  BN4B1.4 | | | | | | | | |
| Related Options |  | | | | | | | | |
| Related Routines | Routines “Called By” | | | | | Routines “Called” | | | |
|  | VAFCA04  VAFCADT2  VAFCMSG3  VAFCQRY | | | | | DIQ  HLFNC  VADPT  VAFHLFNC | | | |
| Routines | Activities | | | | | | | | |
| Data Dictionary (DD) References | ^DIC(21,  ^DPT( | | | | | | | | |
| Related Protocols | N/A | | | | | | | | |
| Related Integration Control Registrations (ICRs) |  | | | | | | | | |
| Data Passing | Input | Output Reference | | | Both | | | Global Reference | Local |
| Input Attribute Name and Definition |  | | | | | | | | |
| Output Attribute Name and Definition |  | | | | | | | | |
| Current Logic | | | | | | | | | |
| VAFHLZSP ;ALB/RJS,TDM,PJH - ZSP SEGMENT - 3/18/96 ; 5/30/07 4:21pm  ;;5.3;Registration;\*\*94,106,122,220,653,754\*\*;Aug 13, 1993;Build 46  S $P(VAFY,HLFS,6)=$S($P(VAFNODE,"^",14)]"":$$HLDATE^HLFNC($P(VAFNODE,"^",14)),1:HLQ) ; Combined SC percent Effective Date  ;  S RETURN=RETURN\_HLFS\_$G(VAFY) | | | | | | | | | |
| Modified Logic (Changes are in bold) | | | | | | | | | |
| VAFHLZSP ;ALB/RJS,TDM,PJH - ZSP SEGMENT - 3/18/96 ; 5/30/07 4:21pm  ;;5.3;Registration;\*\*94,106,122,220,653,754**,948**\*\*;Aug 13, 1993;Build 5  S $P(VAFY,HLFS,6)=$S($P(VAFNODE,"^",14)]"":$$HLDATE^HLFNC($P(VAFNODE,"^",14)),1:HLQ) ; Combined SC percent Effective Date  **S $P(VAFY,HLFS,7)=$$GET1^DIQ(2,DFN\_",",220) ;Dental Classification**  ;  S RETURN=RETURN\_HLFS\_$G(VAFY) | | | | | | | | | |

Table : SDMXMAKE Routine

| Routines | Activities | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Routine Name | **SDMXMAKE** | | | | | | | | |
| Enhancement Category | New | | Modify | Delete | | | No Change | | |
| RTM | SBN1A5.1  BN1A5.13  BN1A6  BN1A6.1  BN1A6.3  BN1A6.4  BN2B1.12  BN3B1.5  BN4A1.7  BN4A1.13  BN4A1.24  BN4B1.1  BN6A1.6  BN6C1.5 | | | | | | | | |
| Related Options |  | | | | | | | | |
| Related Routines | Routines “Called By” | | | | | Routines “Called” | | | |
|  | SDMXSCHP | | | | | DICRW  XLFDT  DIC  DIE  DICN  SDMXGAPT  SDMXCORE  SDCNSLT  ORCSAVE2  SDAMEVT  SCRPW71 | | | |
| Routines | Activities | | | | | | | | |
| Data Dictionary (DD) References | ^DPT(DFN  ^SC(IEN  ^OR(100  ^ORD(101.43  ^TMP("SDAMEVT"  ^SCE("ADFN",DFN | | | | | | | | |
| Related Protocols | N/A | | | | | | | | |
| Related Integration Control Registrations (ICRs) | N/A | | | | | | | | |
| Data Passing | Input | Output Reference | | | Both | | | Global Reference | Local |
| Input Attribute Name and Definition |  | | | | | | | | |
| Output Attribute Name and Definition |  | | | | | | | | |
| Current Logic | | | | | | | | | |
| N/A - New routine | | | | | | | | | |
| Modified Logic (Changes are in bold) | | | | | | | | | |
| SDMXMAKE ;MASS/BB,DAP - Make Appointment API;9/7/2017  ;;5.3;Scheduling;\*\*676\*\*;SEPTEMBER 07,2017  ;;Per VA directive 6402, this routine should not be modified.  Q  MAKE(PATIEN,CLINIEN,APPTTYPE,APPTDT,DURATION,USERDUZ,PATELIG,COMMENT,PID,ORDERID,WLKINFLG,CSLTIEN) ;FILES APPOINTMENT DATE  ; WILL CREATE AN APPOINTMENT IF THE APPOINTMENT DOES NOT EXIST OR UPDATE IF THE APPOINTMENT DOES EXIST  ; PASSED IN PARAMETERS  ; PATIEN (I,REQ KEY) - PATIENT ID  ; CLINIEN (I,REQ KEY) - CLINIC ID  ; APPTTYPE (I,OPT) - APPOINTMENT TYPE INTENAL VALUE  ; APPTDT (I,REQ KEY) - APPOINTMENT DATE+TIME  ; DURATION (I,REQ) - APPOINTMENT LENGTH IN MINUTES  ; USERDUZ (I,REQ) - DATA ENTRY CLERK  ; PATELIG (I,OPT) - PATIENT ELIGIBILITY  ; COMMENT (I,OPT) - REASON FOR APPOINTMENT COMMENT  ; PID (I,OPT) - PATIENT INDICATED DATE  ; ORDERID (I,OPT) - CONSULT'S ORDER ID  ; CSLTIEN (I,OPT) - CONSULT ID  ; Check Requirements  I ($G(PATIEN)="")!($G(CLINIEN)="")!($G(APPTDT)="")!($G(DURATION)="")!($G(USERDUZ)="") Q "Missing Parameters"  ;  N NOW,CLINLINE,DINUM,DIC,DA,X,Y,DLAYGO,DIE,DR,TMPYCLNC,TMPD,FILES,OLDCMMT  N COUNT,ORDPERF,MASSMSG,U  S U="^"  S COUNT=""  ;Create entry in Patient File  D DT^DICRW ;Sets DT = date of action  S NOW=$$NOW^XLFDT()  I $G(PID)="" S PID=NOW  N MASSFDA,MASSIENS,MASSMSG  I $D(^DPT(PATIEN,"S",APPTDT,0)) D  . S MASSIENS=APPTDT\_","\_PATIEN\_","  . S MASSFDA(2.98,MASSIENS,.01)=CLINIEN ;CLINIC  . S MASSFDA(2.98,MASSIENS,"9")=3 ;PURPOSE OF VISIT \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  . S MASSFDA(2.98,MASSIENS,"9.5")=$G(APPTTYPE) ;APPOINTMENT TYPE  . S MASSFDA(2.98,MASSIENS,"19")=USERDUZ ;DATA ENTRY CLERK  . S MASSFDA(2.98,MASSIENS,"20")=$P(APPTDT,".") ;DATE APPT. MADE  . S MASSFDA(2.98,MASSIENS,"26")="0" ;NEXT AVA. APPT. INDICATOR - Always "Other than Next Available"  . S MASSFDA(2.98,MASSIENS,"27")=$P(PID,".") ;DESIRED DATE OF APPOINTMENT  . S MASSFDA(2.98,MASSIENS,"28")=$$PTFU(PATIEN,CLINIEN) ;FOLLOW-UP VISIT  . D FILE^DIE("","MASSFDA","MASSMSG")  E D  . S MASSIENS="?+1,"\_PATIEN\_"," ;? MEANS LOOK UP, +1 MEANS ADD NODE FOR SUBSCRIPT 1  . S MASSIENS(1)=APPTDT  . S MASSFDA(2.98,MASSIENS,".01")=CLINIEN ;CLINIC  . S MASSFDA(2.98,MASSIENS,"9")=3 ;PURPOSE OF VISIT \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  . S MASSFDA(2.98,MASSIENS,"9.5")=$G(APPTTYPE) ;APPOINTMENT TYPE  . S MASSFDA(2.98,MASSIENS,"16")=""  . S MASSFDA(2.98,MASSIENS,"19")=USERDUZ ;DATA ENTRY CLERK  . S MASSFDA(2.98,MASSIENS,"20")=$P(APPTDT,".") ;DATE APPT. MADE  . S MASSFDA(2.98,MASSIENS,"25")=$S($G(WLKINFLG):"W",1:"O") ;SCHEDULING REQUEST TYPE  . S MASSFDA(2.98,MASSIENS,"26")="0" ;NEXT AVA. APPT. INDICATOR - Always "Other than Next Available"  . S MASSFDA(2.98,MASSIENS,"27")=$P(PID,".") ;DESIRED DATE OF APPOINTMENT  . S MASSFDA(2.98,MASSIENS,"28")=$$PTFU(PATIEN,CLINIEN) ;FOLLOW-UP VISIT  . D UPDATE^DIE("","MASSFDA","MASSIENS","MASSMSG")  ;  ;Set up Clinic file for appointment entries  K DIC,DA,X,Y,DLAYGO  I '$D(^SC(CLINIEN,"S",0)) S ^SC(CLINIEN,"S",0)="^44.001DA^^"  I '$D(^SC(CLINIEN,"S",APPTDT,0)) D I $G(Y)<1 Q "Error in 44.001"  . S DIC="^SC("\_CLINIEN\_",""S"","  . S DA(1)=CLINIEN  . S (X,DINUM)=APPTDT  . S DIC("P")="44.001DA",DIC(0)="L",DLAYGO=44.001  . S Y=1 I '$D(@(DIC\_X\_")")) D FILE^DICN  ;  ;Consult Linking  I $$CLINNODE^SDMXGAPT(PATIEN,APPTDT,CLINIEN,0,.COUNT)'="" S OLDCMMT=$P($G(^SC(CLINIEN,"S",APPTDT,1,COUNT,0)),"^",4) ;K ^SC(CLINIEN,"S",APPTDT,1,COUNT)  S COMMENT=$S($L($G(COMMENT))>150:$E($G(COMMENT),1,149)\_"\*",1:$G(COMMENT)) ;truncate if longer han 150 characters  S TMPD=COMMENT,TMPYCLNC=CLINIEN\_$G(U)\_$P($G(^SC(CLINIEN,0)),"^")  I ($G(OLDCMMT)'=COMMENT)!(COUNT'="") D ;Do we need to update anything?  . I COUNT'="",$D(^SC(CLINIEN,"S",APPTDT,1,COUNT,"CONS")) Q ;consult already linked  . I CSLTIEN="",ORDERID'="" S CSLTIEN=$$ORD2CONS^SDMXCORE(ORDERID)  . I CSLTIEN'="" D EDITCS^SDCNSLT(APPTDT,TMPD,TMPYCLNC,CSLTIEN) I 1  I ORDERID'="",($$ORD2CONS^SDMXCORE(ORDERID)="") D ;not a consult but still an order  . S ORDPERF=$G(^OR(100,ORDERID,.1,1,0)) ;what is the performable?  . I ($G(ORDPERF)'=""),($P($G(^ORD(101.43,ORDPERF,0)),"^",9)=2) D STATUS^ORCSAVE2(ORDERID,2) ;performable item 11 says if interface should complete order or not  ;Order Completion  ;  ;Set up filing data arrays and find if the appointment already exists  K DIC,DA,X,Y,DLAYGO,DINUM  S FILES="1////"\_DURATION\_";3////"\_COMMENT\_";7////"\_USERDUZ\_";8////"\_NOW\_";30////"\_$G(PATELIG)\_";688////"\_$G(CSLTIEN)  S DIC="^SC("\_CLINIEN\_",""S"","\_APPTDT\_",1,"  S DA(2)=CLINIEN,DA(1)=APPTDT,X=PATIEN ;Set node levels 2, 1 and index  S DIC("P")="44.003PA",DIC(0)="L",DLAYGO=44.003  D ^DIC  S CLINLINE=+$G(Y)  ;File the data  I CLINLINE=-1 D  . S DIC("DR")=FILES  . D FILE^DICN  E D  . S DIE=DIC ;DIE now holds the subfile's root  . S DA=CLINLINE ;+Y contains the internal entry number of subentry chosen  . S DR=FILES  . D ^DIE  ;  D EVT(PATIEN,APPTDT,CLINIEN,CLINLINE) ;Call event driver  Q 1  EVT(DFN,SD,SC,SDY) ;handle event logging directly to avoid CI/CO prompt  N SDATA,SDMKHDL,SDHDL  K ^TMP("SDAMEVT",$J)  S SDMKHDL=$$HANDLE^SDAMEVT(1)  S (^TMP("SDAMEVT",$J,"BEFORE","DPT"),^TMP("SDAMEVT",$J,"BEFORE","SC"),SDATA("BEFORE","STATUS"),^TMP("SDAMEVT",$J,"BEFORE","STATUS"),^TMP("SDEVT",$J,SDMKHDL,1,"DPT",0,"BEFORE"),^TMP("SDEVT",$J,SDMKHDL,1,"SC",0,"BEFORE"))=""  D AFTER^SDAMEVT(.SDATA,DFN,SD,SC,$G(SDY),SDMKHDL)  S SDATA=$G(SDY)\_U\_DFN\_U\_SD\_U\_SC  D EVT^SDAMEVT(.SDATA,1,+$G(SDAMODE),SDMKHDL)  Q  PTFU(DFN,SC) ;Determine if this is a follow-up (return to clinic within 24 months)  ;Input: DFN=patient ifn  ;Input: SC=clinic ifn  ;Output: '1' if seen within 24 months, '0' otherwise  ;  I ($G(DFN)="")!($G(SC)="") Q 0 ;variable check  N SDBDT,SDT,SDX,SDY,SDCP,SDCP1,SC0,SDENC,SDENC0,SDCT  ;set up variables  S SDBDT=($G(DT)-20000)+.24,SDT=$G(DT)\_.PORT99,(SDCT,SDY)=0  S SC0=$G(^SC(+SC,0)),SDX=$$CPAIR^SCRPW71($G(SC0),.SDCP) ;get credit pair for this clinic  ;Iterate through encounters  F S SDT=$O(^SCE("ADFN",DFN,SDT),-1) Q:SDT<SDBDT!SDY D  . S SDENC=0 F S SDENC=$O(^SCE("ADFN",DFN,SDT,SDENC)) Q:'SDENC!SDY D  . . S SDENC0=$G(^SCE(SDENC,0)) ;get encounter node  . . Q:$P(SDENC0,U,6) ;parent encounters only  . . S SDX=$P(SDENC0,U,4) Q:'SDX ;get clinic  . . S SC0=$G(^SC(SDX,0))  . . S SDX=$$CPAIR^SCRPW71(SC0,.SDCP1) ;get credit pair for encounter  . . S SDY=SDCP=SDCP1 ;compare credit pairs  . . S SDCT=SDCT+1 W:SDCT#10=0 "."  . . Q  . Q  Q SDY  ; | | | | | | | | | |

Table : SDMXMAKE Routine

| Routines | Activities | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Routine Name | **SDMXNS** | | | | | | | | |
| Enhancement Category | New | | Modify | Delete | | | No Change | | |
| RTM | SBN1A5.1  BN1A5.13  BN1A6  BN1A6.1  BN1A6.3  BN1A6.4  BN2B1.12  BN3B1.5  BN4A1.7  BN4A1.13  BN4A1.24  BN4B1.1  BN6A1.6  BN6C1.5 | | | | | | | | |
| Related Options |  | | | | | | | | |
| Related Routines | Routines “Called By” | | | | | Routines “Called” | | | |
|  | SDMXSCHP | | | | | DIE  SDAMEVT  SDAM2  SDCNSLT  XLFDT | | | |
| Routines | Activities | | | | | | | | |
| Data Dictionary (DD) References | ^OR(100  ^SC(IEN | | | | | | | | |
| Related Protocols | N/A | | | | | | | | |
| Related Integration Control Registrations (ICRs) | N/A | | | | | | | | |
| Data Passing | Input | Output Reference | | | Both | | | Global Reference | Local |
| Input Attribute Name and Definition |  | | | | | | | | |
| Output Attribute Name and Definition |  | | | | | | | | |
| Current Logic | | | | | | | | | |
| N/A - New routine | | | | | | | | | |
| Modified Logic (Changes are in bold) | | | | | | | | | |
| SDMXNS ;MASS/PJS - Appointment No-Show API;8/30/17  ;;5.3;Scheduling;\*\*676\*\*;AUGUST 30,2017  ;;Per VA directive 6402, this routine should not be modified.  Q  NOSHOW(DFN,APPDT,USER,CLINIC) ; turns an appointment into a no-show  ; DFN (I,REQ) - Internal ID for the patient  ; APPDT (I,REQ) - VistA date time for the appointment  ; USER (I,REQ) - User Id for who is no-showing the order  ; CLINIC (I,REQ) - Clinic IEN that the appointment occurs in  N MASSFDA,MASSIENS,MASSMSG,APPTSTRING,SDATA,SDDA,SDNSHDL,U,SDENC0,SDAMODE  S DFN=$G(DFN),APPDT=$G(APPDT),USER=$G(USER),U="^"  S SDNSHDL=$$HANDLE^SDAMEVT(1) ; setup event handle  S SDDA=$$FIND^SDAM2(DFN,APPDT,CLINIC) ;setup SDDA - number of appointment node in SC  D BEFORE^SDAMEVT(.SDATA,DFN,APPDT,CLINIC,SDDA,SDNSHDL) ; Set up before state  S MASSIENS=APPDT\_","\_DFN\_","  S MASSFDA(2.98,MASSIENS,"3")="N" ;STATUS. COMMENTED WOULD HANDLE INPATIENT  S MASSFDA(2.98,MASSIENS,"14")=USER ; USER  S MASSFDA(2.98,MASSIENS,"15")=$$NOW^XLFDT()  D FILE^DIE("","MASSFDA","MASSMSG") ;  D EVT(.SDATA,DFN,APPDT,CLINIC,SDDA,SDNSHDL) ; call event protocol/after state  ;  Q $G(MASSMSG)  UNNOSHOW(DFN,APPDT,ORDID,CLINIC) ; reverts an appointment from a no show to a scheduled status  ; DFN (I,REQ) - ID for thepatient  ; APPDT (I,REQ) - VistA date time for the appointment  ; ORDID (I,OPT) - IEN for the order that the appointment was made on.  ; API will relink consults, if this was an order for a consult  N MASSFDA,MASSIENS,MASSMSG,SDNSHDL,SDDA,SDATA,TMPYCLNC,TMPD,SCNAME,CNSLTLNK,U  S DFN=$G(DFN),APPDT=$G(APPDT),ORDID=$G(ORDID),U="^"  S SDNSHDL=$$HANDLE^SDAMEVT(1) ; setup event handle  S SDDA=$$FIND^SDAM2(DFN,APPDT,CLINIC)  D BEFORE^SDAMEVT(.SDATA,DFN,APPDT,CLINIC,SDDA,SDNSHDL)  S MASSIENS=APPDT\_","\_DFN\_","  S MASSFDA(2.98,MASSIENS,"3")="@" ;STATUS. COMMENTED WOULD HANDLE INPATIENT  S MASSFDA(2.98,MASSIENS,"14")="@" ; USER  S MASSFDA(2.98,MASSIENS,"15")="@"  D FILE^DIE("","MASSFDA","MASSMSG")  D EVT(.SDATA,DFN,APPDT,CLINIC,SDDA,SDNSHDL)  ;  ;Relink consult to appointment  S CNSLTLNK=$P($G(^OR(100,ORDID,4)),";")  I CNSLTLNK="" Q MASSMSG  D LINK^SDCNSLT(CLINIC,SDDA,APPDT,CNSLTLNK)  S SCNAME=$P($G(^SC(CLINIC,0)),"^")  S TMPYCLNC=CLINIC\_U\_SCNAME  S TMPD=$P($G(^SC(CLINIC,"S",APPDT,1,SDDA,0)),"^",4)  D EDITCS^SDCNSLT(APPDT,TMPD,TMPYCLNC,CNSLTLNK)  Q MASSMSG  EVT(SDATA,DFN,SDDTM,CLINIC,SDDA,SDNSHDL) ; calls into the consult linking and no show event driver  ; SDATA (IO,REQ) - State data about the appointment  ; DFN (I,REQ) - IEN for patient  ; SDDTM (I,REQ) - internal appointment date/time  ; CLINIC (I,REQ) - clinic IEN  ; SDDA (I,REQ) - number of appointment node in ^SC global  ; SDNSHDL (I,REQ) - even handle  N I,SDINP,Y,SDSTAT,SDTIME,SDYES,SM,SM1  N SD1,SD2,SDMSG,SDT,SDCT,CNSTLNK,CN,CNPAT,DONE  D NOSHOW^SDAMEVT(.SDATA,DFN,SDDTM,CLINIC,SDDA,0,SDNSHDL)  S CNSTLNK="",CN=0,DONE=0  F S CN=$O(^SC(CLINIC,"S",SDDTM,1,CN)) Q:'+CN D Q:DONE  . S CNPAT=$P($G(^SC(CLINIC,"S",SDDTM,1,CN,0)),U)  . I CNPAT=DFN S CNSTLNK=$P($G(^SC(CLINIC,"S",SDDTM,1,CN,"CONS")),U),DONE=1  D:+CNSTLNK NOSHOW^SDCNSLT(CLINIC,SDDTM,CNPAT,CNSTLNK,CN)  Q  ; | | | | | | | | | |
|  | | | | | | | | | |

Table : SDMXSCHI Routine

| Routines | Activities | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Routine Name | **SDMXSCHI** | | | | | | | | |
| Enhancement Category | New | | Modify | Delete | | | No Change | | |
| RTM | SBN1A5.1  BN1A5.13  BN1A6  BN1A6.1  BN1A6.3  BN1A6.4  BN2B1.12  BN3B1.5  BN4A1.7  BN4A1.13  BN4A1.24  BN4B1.1  BN6A1.6  BN6C1.5 | | | | | | | | |
| Related Options |  | | | | | | | | |
| Related Routines | Routines “Called By” | | | | | Routines “Called” | | | |
|  |  | | | | | %ETN  DIC  DIQ  DGMXVLD  SDMXCORE  HLOPRS  ORMXFMT  SDMXERRO  SDMXSCHP  XPAR  XLFDT | | | |
| Routines | Activities | | | | | | | | |
| Data Dictionary (DD) References | DPT(DFN  SC(IEN  SC(“B” | | | | | | | | |
| Related Protocols | N/A | | | | | | | | |
| Related Integration Control Registrations (ICRs) | N/A | | | | | | | | |
| Data Passing | Input | Output Reference | | | Both | | | Global Reference | Local |
| Input Attribute Name and Definition |  | | | | | | | | |
| Output Attribute Name and Definition |  | | | | | | | | |
| Current Logic | | | | | | | | | |
| N/A - New routine | | | | | | | | | |
| Modified Logic (Changes are in bold) | | | | | | | | | |
| SDMXSCHI ;MASS/RPC,DAP - Appointment retrieval API;08/22/2017  ;;5.3;Scheduling;\*\*676\*\*;AUGUST 22,2017  ;;Per VA directive 6402, this routine should not be modified.  Q  PARSEMSG() ;Primary entry routine for HLO based scheduling processing.  ; Will take all scheduling messages through this one point.  ;  ; FROM HLO processing code.  ; HLMSGIEN: At the point HLO calls this message handler, the variable  ; is set to the IEN of the message in the HLO MESSAGE ADMINISTRATION file, #779.2.  ; OUTPUT:  ; MSGARY - data layer of message data formatted and ready to file to vista. The layout is as follows:  ; MSGARY("PATIENT IEN") - IEN  ; MSGARY("APPTDT") = date/time of the appointment in VistA format  ; MSGARY("APPTTYPE") - appointment type  ; MSGARY("CANCEL REASON") - reason for an appointment cancel ;  ; MSGARY("CHECKINDT") - date/time of check in  ; MSGARY("CHECKOUTDT") - date/time of check out  ; MSGARY("COMMENT") - appointment creation comment  ; MSGARY("CLINIC") - appointment clinic  ; MSGARY("DURATION") - appointment duration  ; MSGARY("ELIGIBILITY") - VistA eligibility code  ; MSGARY("EVENT") - appointment message event (SCHEDULE/RESCHEDULE/UPDATE/CANCEL/NOSHOW)  ; MSGARY("PAT INDICATED DATE") - Date the start of search was performed on.  ; MSGARY("PROVIDERS",0) - number of providers  ; MSGARY("PROVIDERS",cnt,"IEN") - provider identifier  ; MSGARY("PROVIDERS",cnt,"NAME") - provider identifier  ; MSGARY("ORDER ID") - Order ID on appointment/consult requests  ; MSGARY("STATUS") - APPT status (SCHEDULED,CHECKED IN,CHECKED OUT,CANCELLED,NO SHOW)  ; MSGARY("USER") - Scheduling user  ; MSGARY("WALK IN") - Walk in boolean flag to denote the appointment being made is being walked in.  D SETMSGET^SDMXCORE ;Sets and error trap  D INITINC^SDMXCORE ;Indicate that this is filing an incoming message  N MSG,HDR,SEG,SEGTYPE,MSGARY,LASTSEG,HDRTIME,ABORT,BASEDT,CLINARY,COUNT,PROVDTL  S ABORT="",BASEDT=""  ; initialize message from queue  I '$$STARTMSG^HLOPRS(.MSG,$G(HLMSGIEN),.HDR) D ERRLOG^SDMXERRO(PORT9,"Message ID "\_$G(HLMSGIEN)\_"no longer on queue")  I $G(HDR("MESSAGE TYPE"))'="SIU" D ERRLOG^SDMXERRO(200,"Scheduling message type not received on scheduling interface. Message type received: "\_$G(HDR("MESSAGE TYPE"))\_" With message control ID of: "\_$G(HDR("MESSAGE CONTROL ID")))  S HDRTIME=$P($G(HDR(2)),"|",2)  S MSGARY("MESSAGE IEN")=HLMSGIEN  ; determine scheduling action event from message event  D SETEVENT($G(HDR("EVENT")),.MSGARY)  I $G(MSGARY("EVENT"))="" D ERRLOG^SDMXERRO(201,"EVENT COULD NOT BE DETERMINED",1) Q  ; Process segments  F Q:('$$NEXTSEG^HLOPRS(.MSG,.SEG)!$G(ABORT)) D  . S SEGTYPE=$G(SEG("SEGMENT TYPE"))  . I SEGTYPE'="NTE" S LASTSEG=SEGTYPE  . ; Process segments by type  . I SEGTYPE="SCH" D SCH(.SEG,.MSGARY,.ABORT,.BASEDT) Q ;SCH MUST BE PROCESSED FIRST SOME VALIDATION DEPENDS ON APPOINTMENT STATUS IN SCH-25  . I SEGTYPE="NTE" D NTE(.SEG,.MSGARY,LASTSEG,.CLINARY,.ABORT,.PROVDTL) Q  . I SEGTYPE="PID" D PID(.SEG,.MSGARY,.ABORT) Q  . I SEGTYPE="PV1" D PV1(.SEG,.MSGARY,HDRTIME,.ABORT) Q  . ;Repeating segment  . I SEGTYPE="OBX" D OBX(.SEG,.MSGARY) Q  . ;Repeating segment block  . I SEGTYPE="RGS" D RGS(.SEG,.MSGARY) Q  . I SEGTYPE="AIS" D AIS(.SEG,.MSGARY) Q  . I SEGTYPE="AIG" D AIG(.SEG,.MSGARY,.PROVDTL,BASEDT) Q  . I SEGTYPE="AIP" D AIP(.SEG,.MSGARY,.PROVDTL,BASEDT) Q  . ;Repeating segment block  F COUNT=1:1:$G(CLINARY(0)) D  . S MSGARY("CLINIC")=$G(CLINARY(COUNT))  . I $$GET^XPAR("SYS","SDMX PROVIDER TIME")=1 D ;1 = SCH DT and AIP/AIG duration  . . S MSGARY("APPTDT")=$G(BASEDT)+(0.0001\*(COUNT-1))  . . S MSGARY("DURATION")=$G(CLINARY(COUNT,"LN"))  . E I $$GET^XPAR("SYS","SDMX PROVIDER TIME")=3 D ;3 = AIP/AIG DT and duration  . . S MSGARY("APPTDT")=$G(CLINARY(COUNT,"DT"))  . . S MSGARY("DURATION")=$G(CLINARY(COUNT,"LN"))  . E D ;2 = SCH DT and duration, fall on SCH if config not set  . . S MSGARY("APPTDT")=$G(BASEDT)+(0.0001\*(COUNT-1)) ;Increment time by 1 minute for each additional clinic, for DPT nodes  . D BUILD^SDMXSCHP(.MSGARY)  D DONEINC^SDMXCORE  Q  ;  SETEVENT(EVENT,MSGARY) ;Takes the scheduling event and sets a message event to process.  ; EVENT (I/REQ) - Message event from the MSH header. EX. S12, S14  ; MSGARY (I/O,REQ) message array structure with deformated and translated data ready for filing. See PARSEMSG for details.  I $G(EVENT)="" Q  I EVENT="S12" S MSGARY("EVENT")="SCHEDULE"  I EVENT="S14" S MSGARY("EVENT")="UPDATE"  I EVENT="S15" S MSGARY("EVENT")="CANCEL"  I EVENT="S26" S MSGARY("EVENT")="NOSHOW"  Q  ;  SCH(SCH,MSGARY,ABORT,BASEDT) ;SCH segment processing.:  ; SEG (I/REQ) - SCH message segment data  ; MSGARY (I/O,REQ) message array structure with deformated and translated data ready for filing. See PARSEMSG for details.  ; ABORT (O,OPT) - Error parameter if we did not receive an appointment date and time. Fatal case to this message.  ; BASEDT (O,REQ) - appointment base date/time to use. May be incremented later if processing multiple joint clinic scheduling  N ORDIDTYP  ; Appointment Identifiers  S MSGARY("PLACER ID")=$$GET^HLOPRS(.SCH,1,1) ;SCH-1.1  S MSGARY("FILLER ID")=$$GET^HLOPRS(.SCH,2,1) ;SCH-2.1  ; Cancel Reason  I MSGARY("EVENT")="CANCEL" S MSGARY("CANCEL REASON")=$$GETRSN(.SCH) ;SCH-6  ; Duration  S MSGARY("DURATION")=$$GETLEN(.SCH) ;SCH-9,10  I 'MSGARY("DURATION") S ABORT=1 Q  ; Appointment Date  S BASEDT=$$HL7TFM^XLFDT($$GET^HLOPRS(.SCH,11,4),"L") ;SCH-11.3  I $G(BASEDT)="" D ERRLOG^SDMXERRO(305,"NO APPOINTMENT DATE AND TIME",1) S ABORT=1 Q  ; User  S MSGARY("USER")=$$GETUSER(.SCH) ;SCH-20  ; Status  S MSGARY("STATUS")=$$GETSTAT(.SCH) ;SCH-25  ; Linked Consults/Orders  S ORDIDTYP=$$GET^HLOPRS(.SCH,27,2) ;Placer ID Type  I (ORDIDTYP=$$TRANORCD^ORMXFMT(2,"^OR(100"))!(ORDIDTYP="") D  . S MSGARY("ORDER ID")=$$GET^HLOPRS(.SCH,27,1) ;Placer ID;  . S MSGARY("ORDER ID TYPE")=ORDIDTYP  . S MSGARY("CONSULT ID")=$$ORD2CONS^SDMXCORE(MSGARY("ORDER ID"))  Q  ;  NTE(NTE,MSGARY,LASTSEG,CLINARY,ABORT,PROVDTL) ;NTE segment processing.  ; NTE (I/REQ) - NTE message segment data  ; MSGARY (I/O,REQ) - message array structure with deformated and translated data ready for filing. See PARSEMSG for details.  ; LASTSEG (I,REQ) - segment previous to the NTE to determine context of note.  ; CLINARY (I/O,REQ) - List of Clinics to be scheduled. Could contain more than one for joint appointments  ; ABORT (O,REQ) - quit parameter to the whole tag. Having one clinic unmapped must stop filing.  ; PROVDTL (I/OPT) - passed when NTE concerns a preceding AIP or AIG segment  N NOTE,NOTETYPE,CLINIC  S LASTSEG=$G(LASTSEG)  S NOTE=$$GET^HLOPRS(.NTE,3,1) ;NTE-3.1  S NOTETYPE=$$GET^HLOPRS(.NTE,4,1) ;NTE-4.1  ; Process NTE following SCH for scheduling comments.  S NOTE=$TR(NOTE,"^","?") ;FILEMAN can't handle "^"  I LASTSEG="SCH" D  . I ($G(MSGARY("COMMENT"))'=""),(NOTE'="") S MSGARY("COMMENT")=$G(MSGARY("COMMENT"))\_" "  . S MSGARY("COMMENT")=NOTE  ;  ; Process NTE following AIG/AIP for getting clinics  I (LASTSEG="AIP")!(LASTSEG="AIG") D  . I NOTETYPE="CLINIC" D  . . S CLINIC=$$GETCLIN(NOTE)  . . I CLINIC="" D ERRLOG^SDMXERRO(300,"CLINIC MAPPING ERROR VALUE") S ABORT=1 Q  . . S CLINARY(0)=$G(CLINARY(0))+1  . . S CLINARY(CLINARY(0))=CLINIC  . . S CLINARY(CLINARY(0),"DT")=$G(PROVDTL("DT"))  . . S CLINARY(CLINARY(0),"LN")=$G(PROVDTL("LN"))  Q  ;  PID(PID,MSGARY,ABORT) ;PID segment processing.  ; PID (I/REQ) - PID message segment data  ; MSGARY (I/O,REQ) message array structure with deformated and translated data ready for filing. See PARSEMSG for details.  ; ABORT (O,OPT) - Error parameter if we failed to find a valid patient. Fatal case to this message.  N IDENTIFIERS,IENCHECK,OK  ; Get IDs for this namespace from the identifier list  D GETIDS(.PID,.IDENTIFIERS)  ; Validate IDs  S IENCHECK=$$GTIENICN($G(IDENTIFIERS("PATIENT ICN")))  I IENCHECK'="" D I 1  . I IENCHECK=$G(IDENTIFIERS("PATIENT IEN")) S MSGARY("PATIENT IEN")=$G(IDENTIFIERS("PATIENT IEN")) Q  . I $G(MSGARY("PATIENT IEN"))="" S MSGARY("PATIENT IEN")=IENCHECK Q  . D ERRLOG^SDMXERRO(204,"ICN/IEN patient mismatch") S ABORT=1 Q  E S MSGARY("PATIENT IEN")=$G(IDENTIFIERS("PATIENT IEN"))  ; Handle error conditions  I $G(ABORT) Q ;an error occurred in identifier validation  I $G(MSGARY("PATIENT IEN"))="" D ERRLOG^SDMXERRO(4050,"No Identifiers found to lookup a patient") S ABORT=1 Q  I $G(^DPT(MSGARY("PATIENT IEN"),0))="" D ERRLOG^SDMXERRO(204,"Patient IEN does not exist in the system") S ABORT=1 Q  ; Perform Patient Validation  S OK=$$VLDPAT^DGMXVLD(.PID,MSGARY("PATIENT IEN")) ; This can send error messages  Q  ;  PV1(PV1,MSGARY,HDRTIME,ABORT) ;PV1 segment processing.  ; PV1 (I/REQ) - PV1 message segment data  ; MSGARY (I/O,REQ) message array structure with deformated and translated data ready for filing. See PARSEMSG for details.  ; HDRTIME (I,OPT) - TIME FROM MSH-7, USED AS A DEFAULTING OPTION  ; ABORT (O,OPT) - Error parameter if we failed to find a valid patient. Fatal case to this message.  N ERROR  ;  ;ONLY LOG DEFAULTING ERRORS  ;CHECK IN DATE/TIME  S MSGARY("CHECKINDT")=$$DETTIME($$GET^HLOPRS(.PV1,44,1),$G(HDRTIME),.ERROR) ;PV1-44.1  I ($G(ERROR)'=""),($G(MSGARY("STATUS"))="CHECKED IN") D ERRLOG^SDMXERRO(306,"NO CHECK IN TIME IN PV1-44 "\_ERROR,1)  ;  ;CHECK OUT DATE/TIME  S MSGARY("CHECKOUTDT")=$$DETTIME($$GET^HLOPRS(.PV1,45,1),$G(HDRTIME),.ERROR) ;PV1-45.1  I ($G(ERROR)'=""),($G(MSGARY("STATUS"))="CHECKED OUT") D ERRLOG^SDMXERRO(307,"NO CHECK IN TIME IN PV1-45 "\_ERROR,1)  Q  OBX(OBX,MSGARY,HDRTIME) ;OBX segment processing.  ; OBX (I/REQ) - OBX message segment data  ; MSGARY (I/O,REQ) message array structure with deformated and translated data ready for filing. See PARSEMSG for details.  ; HDRTIME (I,OPT) - TIME FROM MSH-7, USED AS A DEFAULTING OPTION  N OBSTYPE,ERROR  ;  ; Get Observation Type  S OBSTYPE=$$GET^HLOPRS(.OBX,3,1) ;OBX-3.1  ;  ; Process appointment eligibility  I OBSTYPE="APPT ELIGIBILITY" S MSGARY("ELIGIBILITY")=$$GETELIG(.OBX) ;OBX-5  ;  ; Process Appointment Type  I OBSTYPE="APPT TYPE" S MSGARY("APPTTYPE")=$$GETTYPE(.OBX) ;OBX-5  ;  ; Process Walk in flag  I OBSTYPE="WALK-IN" D  . S MSGARY("WALK IN")=$$GET^HLOPRS(.OBX,5,1) ;OBX-5.1 for boolean flag  . I '$$INSTRING^SDMXCORE(MSGARY("WALK IN"),"0,1") D ERRLOG^SDMXERRO(311,"WALKIN FLAG MAPPING ERROR",1)  ;  ; Process Patient Indicated Date  I OBSTYPE="REQUESTED DATE" D  . S MSGARY("PAT INDICATED DATE")=$$DETTIME($$GET^HLOPRS(.OBX,5,1),$G(HDRTIME),.ERROR) ;OBX-5.1 for boolean flag  . I $G(ERROR)'="" D ERRLOG^SDMXERRO(310,"NO PATIENT INDICATED DATE "\_ERROR,1)  Q  RGS(RGS,MSGARY) ; RGS segment processing.  ; Per HL7 this segment repeats and has multiple AIS/AIG/AIP segments underneath.  ; RGS (I/REQ) - RGS message segment data  ; MSGARY (I/O,REQ) message array structure with deformated and translated data ready for filing. See PARSEMSG for details.  Q  AIS(AIS,MSGARY) ;AIS segment processing.  ; Per HL7 this field can repeat within each RGS group.  ; AIS (I/REQ) - AIS message segment data  ; MSGARY (I/O,REQ) message array structure with deformated and translated data ready for filing. See PARSEMSG for details.  Q  AIP(AIP,MSGARY,PROVDTL,BASEDTE) ;AIP segment processing.  ; Per HL7 this field can repeat within each RGS group.  ; AIP (I/REQ) - AIP message segment data  ; MSGARY (I/O,REQ) message array structure with deformated and translated data ready for filing. See PARSEMSG for details.  ; PROVDTL (O,REQ) - AIP date/time and length  ; BASEDTE (I,REQ) - Appt D/T from SCH  I $$HL7TFM^XLFDT($$GET^HLOPRS(.AIP,6,1),"L")'="" S PROVDTL("DT")=$$HL7TFM^XLFDT($$GET^HLOPRS(.AIP,6,1),"L") ;AIP-6  E S PROVDTL("DT")=BASEDTE  I $$GETLEN(,.AIP)'="" S PROVDTL("LN")=$$GETLEN(,.AIP) ;AIP-9  E S PROVDTL("LN")=MSGARY("DURATION")  Q  AIG(AIG,MSGARY,PROVDTL,BASEDTE) ;AIG segment processing.  ; Per HL7 this field can repeat within each RGS group.  ; AIG (I/REQ) - AIG message segment data  ; MSGARY (I/O,REQ) message array structure with deformated and translated data ready for filing. See PARSEMSG for details.  ; PROVDTL (O,REQ) - AIG date/time and length  ; BASEDTE (I,REQ) - Appt D/T from SCH  I $$HL7TFM^XLFDT($$GET^HLOPRS(.AIG,8,1),"L")'="" S PROVDTL("DT")=$$HL7TFM^XLFDT($$GET^HLOPRS(.AIG,8,1),"L") ;AIG-8  E S PROVDTL("DT")=BASEDTE  I $$GETLEN(,,.AIG)'="" S PROVDTL("LN")=$$GETLEN(,,.AIG) ;AIG-11  E S PROVDTL("LN")=MSGARY("DURATION")  Q  ;  GETRSN(SCH) ; Collects appointment reason and translates into internal format.  ; Tries using the Title to lookup the reason. If that fails uses the ID to lookup  ; the reason against the title. If that fails tries using the ID against the ID.  ; SCH (I/REQ) - SCH message segment data  Q $$DATALKUP(.SCH,"409.2","^SD(409.2,",6,302,"APPOINTMENT REASON MAPPING ERROR")  GETTYPE(OBX) ;translates appointment type into internal format  ; OBX (I/REQ) - OBX message segment data  N APPTTYPE  S APPTTYPE=$$DATALKUP(.OBX,"409.1","^SD(409.1,",5,303,"APPOINTMENT TYPE MAPPING ERROR")  I $G(APPTTYPE)="" S APPTTYPE=9  Q APPTTYPE  GETLEN(SCH,AIP,AIG) ;Translates duration into Minutes. Assumes minutes unless set to S or SEC for the units  ; Only one parameter at a time should be passed-in, depending on what segment is calling this tag  ; SCH (I/OPT) - SCH message segment data  ; AIP (I/OPT) - AIP message segment data  ; AIG (I/OPT) - AIG message segment data  N DURATION,UNIT  I $D(SCH) D  . S DURATION=+$$GET^HLOPRS(.SCH,9,1) ;SCH-9  . I DURATION=0 D ERRLOG^SDMXERRO(304,"NO APPOINTMENT DURATION RECIEVED IN SCH",1) Q  . S UNIT=$$GET^HLOPRS(.SCH,10,1) ;SCH-10  E I $D(AIP) D  . S DURATION=+$$GET^HLOPRS(.AIP,9,1) ;AIP-9  . I DURATION=0 D ERRLOG^SDMXERRO(304,"NO APPOINTMENT DURATION RECIEVED IN AIP",1) Q  . S UNIT=$$GET^HLOPRS(.AIP,10,1) ;AIP-10  E I $D(AIG) D  . S DURATION=+$$GET^HLOPRS(.AIG,11,1) ;AIG-11  . I DURATION=0 D ERRLOG^SDMXERRO(304,"NO APPOINTMENT DURATION RECIEVED IN AIG",1) Q  . S UNIT=$$GET^HLOPRS(.AIG,12,1) ;AIG-12  ; Translate to minutes  I $$INSTRING^SDMXCORE(UNIT,"S,SEC") S DURATION=DURATION/60  Q $G(DURATION)  GETUSER(SCH) ;collects appointment entered by user and confirms they are a user in the 200 file  ; SCH (I/REQ) - SCH message segment data  N USER  S USER=$$GET^HLOPRS(.SCH,20,1)  I $$GET1^DIQ(200,USER,".01")'="" Q USER  E S USER=.5 D ERRLOG^SDMXERRO(308,"USER MAPPING ERROR",1) ;Default to FileMan User .5  Q USER  GETSTAT(SCH) ; Translates status into appropriate scheduling statuses  ; Options: (SCHEDULED,CHECKED IN,CHECKED OUT,CANCELLED,NO SHOW)  ; SCH (I/REQ) - SCH message segment data  N STATUS,ID,TITLE  S ID=$$GET^HLOPRS(.SCH,25,1)  S TITLE=$$GET^HLOPRS(.SCH,25,2)  I $$INSTRING^SDMXCORE(TITLE,"SCHEDULED,CHECKED IN,CHECKED OUT,CANCELLED,NO SHOW") Q TITLE  I $$INSTRING^SDMXCORE(ID,"SCHEDULED,CHECKED IN,CHECKED OUT,CANCELLED,NO SHOW") Q ID  I (ID'="")!(TITLE'="") D ERRLOG^SDMXERRO(309,"SCHEDULING STATUS MAPPING ERROR",1)  Q "NA"  GETIDS(PID,IDENTIFIERS) ;Loops over PID-3 and extracts all IDs out into an array. Currently will identify ICN and IEN identifiers only  ; PID (I,REQ) - PID message segment data  ; IDENTIFIERS (O,REQ) - Identifier array to return  ;  K IDS ;force output parameter  N REP,ID,ASSIGN,IDTYPE  ; Loop through all repetitions of PID-3  F REP=1:1:$O(PID(3,""),-1) D  . S ID=$$GET^HLOPRS(.PID,3,1,1,REP) ;PID-3.1  . I ID="" Q ;Only check the ID if it exists  . S ASSIGN=$$GET^HLOPRS(.PID,3,4,1,REP) ;PID-3.4  . S IDTYPE=$$GET^HLOPRS(.PID,3,5,1,REP) ;PID-3.5  . I $$ISPATIEN(ASSIGN,IDTYPE) S IDENTIFIERS("PATIENT IEN")=ID Q  . I $$ISPATICN(ASSIGN,IDTYPE) S IDENTIFIERS("PATIENT ICN")=ID  Q  ISPATIEN(ASSIGN,IDTYPE) ;Determines if given id descriptors are the IEN for this instance  ; ASSIGN (I,OPT) - Assigning Authority of this identifier  ; IDTYPE (I,OPT) - ID Type of this identifier  I $G(IDTYPE)="IEN" Q 1  Q 0  ISPATICN(ASSIGN,IDTYPE) ;Determines if given id descriptors are the ICN for this instance  ; ASSIGN (I,OPT) - Assigning Authority of this identifier  ; IDTYPE (I,OPT) - ID Type of this identifier  I $G(IDTYPE)="ICN" Q 1  Q 0  GTIENICN(PATICN) ;Lookup the IEN for a given ICN  ; PATICN (I,REQ) - Patient ICN  ; IDTYPE (I,OPT) - ID Type of this identifier  N PATIEN  S PATIEN=""  Q PATIEN  GETCLIN(ID) ;Collects clinic from the PV1-3.1 segment. There is no title component to this data type.  ; ID (I/REQ) - Clinic string to lookup clinic with  ;  ; Check Requirements  I $G(ID)="" Q ""  ;  N CLINIC  ; Try robust mutli tier lookup  S CLINIC=$O(^SC("B",ID,""))  I CLINIC'="" Q CLINIC  I $G(^SC(ID,0))'="" Q ID  Q ""  GETELIG(OBX) ;Collects appointment eligibility and translates into internal format  ; Tries using the Title to lookup the eligibility. If that fails uses the  ; ID to lookup the reason against the title. If that fails tries using the ID against the ID.  ; OBX (I/REQ) - OBX message segment data  Q $$DATALKUP(.OBX,"8","^DIC(8,",5)  DETTIME(PV1TIME,HDRTIME,ERROR) ;RETURNS THE BEST CHECK IN/OUT TIME AVAILABLE IN THE MESSAGE OR DEFAULTS TO NOW  ; PV1TIME (I,OPT) - HIGHEST PRIORITY TIME TO RETURN FROM EITHER PV1-44 OR PV1-45  ; HDRTIME (I,OPT) - TIME FROM MSH-7  ; ERROR (O,OPT) - ERROR OUTPUT PARAMETER  K ERROR  I $G(PV1TIME)'="" Q $$HL7TFM^XLFDT(PV1TIME,"L")  I $G(HDRTIME)'="" S ERROR="FALLING BACK TO MSH-7" Q $$HL7TFM^XLFDT(HDRTIME,"L")  S ERROR="FALLING BACK TO FILING TIME" Q $$NOW^XLFDT()  CERROR() ;SEND AN ERROR MESSAGE OUT AND LOG THE ERROR WITHIN VISTA TO ^ERRORS  D ERRLOG^SDMXERRO(-1,"Cache Error: "\_$ZE,1)  D BACK^%ETN  Q  DATALKUP(SEG,FILE,FILEPATH,FIELD,ERRCODE,ERRTEXT) ; Translates a data element for a given fileman file in an HL7 field  ; Tries using the Title to lookup the data. If that fails uses the ID to lookup  ; the reason against the title. If that fails tries using the ID against the ID.  ; SEG (I,REQ) - Message segment to parse  ; FILE (I,REQ) - Fileman File to lookup  ; FILEPATH (I,REQ) - global path to the file's storage location for DIC lookup. Make sure to end with a comma ^<glo>(<File>,  ; FIELD (I,REQ) - message field to look in  ; ERRCODE (I,OPT) - error to log if failure  ; ERRTEXT (I,OPT) - error text to log if failure  ;  ; Check Requirements  I ($G(FILE)="")!($G(FIELD)="") Q  ;  N ID,TITLE,DATA,X,Y,DIC  ;  S DATA=""  S ID=$$GET^HLOPRS(.SEG,FIELD,1) ;component 1 HL7 ID field  S TITLE=$$GET^HLOPRS(.SEG,FIELD,2) ;component 2 HL7 Title field  I (ID=""),(TITLE="") Q "" ;No data to translate  ;  ; Try robust mutli tier lookup  I TITLE'="" S DIC=FILEPATH,DIC(0)="B",X=TITLE D ^DIC S DATA=$P(Y,"^",1) ;lookup "B" node with the second component  I DATA'="",DATA'=-1 Q DATA  I ID'="" d  . S DIC=FILEPATH,DIC(0)="B",X=ID D ^DIC S DATA=$P(Y,"^",1) ;lookup "B" node with the first component  . I DATA'="",DATA'=-1 Q  . I $$GET1^DIQ(FILE,ID,".01")'="" S DATA=ID ;check if the ID matches a record in the File. if so use it.  I DATA'="" Q DATA  I $G(ERRCODE)'="" D ERRLOG^SDMXERRO(ERRCODE,ERRTEXT,1) ;All lookups have failed and data exists so send an error  Q ""  ; | | | | | | | | | |

Table : SDMXSCHP Routine

| Routines | Activities | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Routine Name | **SDMXSCHP** | | | | | | | | |
| Enhancement Category | New | Modify | Delete | | | No Change | | | |
| RTM | SBN1A5.1  BN1A5.13  BN1A6  BN1A6.1  BN1A6.3  BN1A6.4  BN2B1.12  BN3B1.5  BN4A1.7  BN4A1.13  BN4A1.24  BN4B1.1  BN6A1.6  BN6C1.5 | | | | | | | | |
| Related Options |  | | | | | | | | |
| Related Routines | Routines “Called By” | | | | Routines “Called” | | | | |
|  | SDMXSCHI | | | | SDMXLKRQ  SDMXCORE  SDMXMAKE  SDMXCHKI  SDMXCHKO  SDMXCANC  SDMXNS  SDMXUCAN | | | | |
| Routines | Activities | | | | | | | | |
| Data Dictionary (DD) References |  | | | | | | | | |
| Related Protocols | N/A | | | | | | | | |
| Related Integration Control Registrations (ICRs) | N/A | | | | | | | | |
| Data Passing | Input | Output Reference | | Both | | | Global Reference | Local |
| Input Attribute Name and Definition |  | | | | | | | | |
| Output Attribute Name and Definition |  | | | | | | | | |
| Current Logic | | | | | | | | | |
| N/A - New routine | | | | | | | | | |
| Modified Logic (Changes are in bold) | | | | | | | | | |
| SDMXSCHP ;MASS/RPC,DAP - Appointment Processing routine using SAFE processor;09/05/2017  ;;5.3;Scheduling;\*\*676\*\*;SEPTEMBER 05,2017  ;;Per VA directive 6402, this routine should not be modified.  Q  BUILD(MSGARY) ; Appointment building tag. Receives data from an inbound message formatted to VistA elements  ; and makes appropriate decisions as to next steps to file it to VistA.  ; PARAMETERS:  ; MSGARY (I,REQ) - Appointment data array from the message in the VistA data structure format.  ; Array format below:  ;  ; MSGARY("PATIENT IEN") - IEN  ; MSGARY("APPTDT") - date/time of the appointment in VistA format  ; MSGARY("APPTTYPE") - appointment type  ; MSGARY("CANCEL REASON") - reason for an appointment. Will be the reason for cancel if being canceled;  ; MSGARY("CANCEL REMARK") - cancel reason comment. (Not currently supported by HL7 SIU messaging. Could be added later if needed. Will be in source system.  ; MSGARY("CHECKOUTDT") - date/time of check out  ; MSGARY("CHECKINDT") - date/time of check in  ; MSGARY("COMMENT") - appointment creation comment  ; MSGARY("CLINIC") - appointment clinic  ; MSGARY("DURATION") - appointment duration  ; MSGARY("ELIGIBILITY") - VistA eligibility code  ; MSGARY("EVENT") - appointment message event (SCHEDULE/UPDATE/CANCEL/NOSHOW)(S12/S14/S15/S26)  ; MSGARY("FILLER ID") - appointment filler ID  ; MSGARY("NEXT AVAILABLE") - If the appointment was scheduled as next available  ; MSGARY("ORDER ID") - Order ID on appointment/consult requests  ; MSGARY("PAT INDICATED DATE") - Date the start of search was performed on.  ; MSGARY("PLACER ID") - appointment placer ID  ; MSGARY("PROVIDERS",0) - number of providers  ; MSGARY("PROVIDERS",cnt,"IEN") - provider identifier  ; MSGARY("PROVIDERS",cnt,"NAME") - provider identifier  ; MSGARY("STATUS") - APPT status (SCHEDULED,CHECKED IN,CHECKED OUT,CANCELLED,NO SHOW)  ; MSGARY("USER") - Scheduling user  ;  N ACTIONLIST,LOCKARY  I '$$SEQUENCE^SDMXLKRQ(.MSGARY,.LOCKARY) Q ;LOCKARY contains a copy of MSGARY's locked IDs in case something happens to MSGARY  ; Determine scheduling actions to take  S ACTIONLIST=$$ACTIONS(.MSGARY)  ; Process the action list  D PROCESS(ACTIONLIST,.MSGARY)  ;  D AFTRPROC^SDMXLKRQ(.LOCKARY)  Q  ;  ACTIONS(MSGARY,PATIENTIEN,APPTDT) ; Compares message data against the existing appointment in VistA to determine  ; the actions that need to be taken on the appointment.  ;  ; MSGARY (I,REQ) - Appointment data array from the message in the VistA data structure format.  ; See BUILD header above for array structure:  ; PATIENTIEN (I,OPT) - Patient IEN  ; APPTDT (I,OPT) - appointment DateTime in VistA format  ;  ; Output: Comma delimited list of actions to perform.  ; Possible actions include:  ; SCHEDULE  ; CHECKIN  ; CHECKOUT  ; CANCEL  ; NOSHOW  ; UNDOCHECKIN  ; UNDOCHECKOUT  ; UNDONOSHOW  ; UNDOCANCEL  ;  N APPTARY,ACTIONLIST,MESSAGESTATUS,APPTSTATUS,CONTEXTARY,EVENT  ;  ; check requirements  I $G(PATIENTIEN)="" S PATIENTIEN=MSGARY("PATIENT IEN")  I $G(PATIENTIEN)="" Q ""  I $G(APPTDT)="" S APPTDT=MSGARY("APPTDT")  I $G(APPTDT)="" Q ""  ;  ;  ; get Appointment from VistA to compare to the message  D GETAPPT^SDMXGAPT(PATIENTIEN,APPTDT,.APPTARY)  ;  ; setup primary keys to the context logic  S MESSAGESTATUS=$G(MSGARY("STATUS"))  S APPTSTATUS=$G(APPTARY("STATUS"))  I $G(APPTSTATUS)="" S APPTSTATUS="NA" ;No appointment status  S EVENT=MSGARY("EVENT")  ;  ; setup event. Context for cancel and no show are specific  I '$$INSTRING^SDMXCORE(EVENT,"CANCEL,NOSHOW") S EVENT="APPT"  ;  ; Get context Logic  D CONTEXT(.CONTEXTARY)  ;  ; Determine Scheduling action list  S ACTIONLIST=$G(CONTEXTARY(EVENT,APPTSTATUS,MESSAGESTATUS))  I $G(ACTIONLIST)'="" Q ACTIONLIST  S ACTIONLIST=$G(CONTEXTARY(EVENT,APPTSTATUS))  I $G(ACTIONLIST)'="" Q ACTIONLIST  S ACTIONLIST=$G(CONTEXTARY(EVENT))  Q ACTIONLIST  CONTEXT(CONTEXT) ; SAFE processor array for comparisons of scheduling message state against  ; appointment message state to make sure the appropriate action is taken  ; to get the appointment in Sync with what is in the message.  ;  ; CONTEXT (O,REQ) - Context array for the appointment action list  ; CONTEXT(EVENT,APPTSTATUS,MESSAGESTATUS)  ;  ; Available Events are as follows:  ; APPT - use for S12 and S14 events as they can be treated similar for appointment actions to take  ; CANCEL - S15 events have special precedent to just perform the action  ; NOSHOW - S26 events have special precedent to just perform the action.  ;  ; Available statuses are as follows:  ; SCHEDULED  ; CHECKED IN  ; CHECKED OUT  ; CANCELLED  ; NO SHOW  ; NA - This status is used for APPTSTATUS only when no appointment exists in VistA  ;  ;  K CONTEXT S CONTEXT="" ;force output parameter  ;  ; Load context to the array.  S CONTEXT("APPT","SCHEDULED","SCHEDULED")="UPDATE"  S CONTEXT("APPT","SCHEDULED","CHECKED IN")="CHECKIN"  S CONTEXT("APPT","SCHEDULED","CHECKED OUT")="CHECKIN,CHECKOUT"  S CONTEXT("APPT","SCHEDULED","CANCELLED")="CANCEL"  S CONTEXT("APPT","SCHEDULED","NO SHOW")="NOSHOW"  S CONTEXT("APPT","CHECKED IN","SCHEDULED")="UNDOCHECKIN"  S CONTEXT("APPT","CHECKED IN","CHECKED IN")="UPDATE"  S CONTEXT("APPT","CHECKED IN","CHECKED OUT")="CHECKOUT"  S CONTEXT("APPT","CHECKED IN","CANCELLED")="UNDOCHECKIN,CANCEL"  S CONTEXT("APPT","CHECKED IN","NO SHOW")="UNDOCHECKIN,NOSHOW"  S CONTEXT("APPT","CHECKED OUT","SCHEDULED")="UNDOCHECKOUT,UNDOCHECKIN"  S CONTEXT("APPT","CHECKED OUT","CHECKED IN")="UNDOCHECKOUT"  S CONTEXT("APPT","CHECKED OUT","CHECKED OUT")="UPDATE"  S CONTEXT("APPT","CHECKED OUT","CANCELLED")="UNDOCHECKOUT,UNDOCHECKIN,CANCEL"  S CONTEXT("APPT","CHECKED OUT","NO SHOW")="UNDOCHECKOUT,UNDOCHECKIN,NOSHOW"  S CONTEXT("APPT","CANCELLED","SCHEDULED")="UNDOCANCEL"  S CONTEXT("APPT","CANCELLED","CHECKED IN")="UNDOCANCEL,CHECKIN"  S CONTEXT("APPT","CANCELLED","CHECKED OUT")="UNDOCANCEL,CHECKIN,CHECKOUT"  S CONTEXT("APPT","CANCELLED","CANCELLED")="UPDATE"  S CONTEXT("APPT","CANCELLED","NO SHOW")="UNDOCANCEL,NOSHOW"  S CONTEXT("APPT","NO SHOW","SCHEDULED")="UNDONOSHOW"  S CONTEXT("APPT","NO SHOW","CHECKED IN")="UNDONOSHOW,CHECKIN"  S CONTEXT("APPT","NO SHOW","CHECKED OUT")="UNDONOSHOW,CHECKIN,CHECKOUT"  S CONTEXT("APPT","NO SHOW","CANCELLED")="UNDONOSHOW,CANCEL"  S CONTEXT("APPT","NO SHOW","NO SHOW")="UPDATE"  S CONTEXT("APPT","NA","SCHEDULED")="SCHEDULE"  S CONTEXT("APPT","NA","CHECKED IN")="SCHEDULE,CHECKIN"  S CONTEXT("APPT","NA","CHECKED OUT")="SCHEDULE,CHECKIN,CHECKOUT"  S CONTEXT("APPT","NA","CANCELLED")="SCHEDULE,CANCEL"  S CONTEXT("APPT","NA","NO SHOW")="SCHEDULE,NOSHOW"  S CONTEXT("APPT","NA")="SCHEDULE"  S CONTEXT("APPT")="UPDATE"  S CONTEXT("CANCEL")="CANCEL"  S CONTEXT("CANCEL","NA")="SCHEDULE,CANCEL"  S CONTEXT("CANCEL","CHECKED IN")="UNDOCHECKIN,CANCEL"  S CONTEXT("CANCEL","CHECKED OUT")="UNDOCHECKOUT,UNDOCHECKIN,CANCEL"  S CONTEXT("CANCEL","NO SHOW")="UNDONOSHOW,CANCEL"  S CONTEXT("NOSHOW")="NOSHOW"  S CONTEXT("NOSHOW","NA")="SCHEDULE,NOSHOW"  S CONTEXT("NOSHOW","CHECKED IN")="UNDOCHECKIN,NOSHOW"  S CONTEXT("NOSHOW","CHECKED OUT")="UNDOCHECKOUT,UNDOCHECKIN,NOSHOW"  S CONTEXT("NOSHOW","CANCELLED")="UNDOCANCEL,NOSHOW"  S CONTEXT("NOSHOW","SCHEDULED","NO SHOW")="NOSHOW"  ;  Q  PROCESS(ACTIONLIST,MSGARY) ; Context logic array for comparisons  ;  ; ACTIONLIST (I,REQ) - Comma delimited list of actions to perform  ; MSGARY (I,REQ) - Appointment data array from the message in the VistA data structure format.  ;  N ACTION,COUNT  ;  ; Process action  F COUNT=1:1:$L($G(ACTIONLIST),",") D  . S ACTION=$P($G(ACTIONLIST),",",COUNT)  . I $G(ACTION)="" Q  . ; ACTION PROCESSING  . I ACTION="SCHEDULE" D SCHEDULE(.MSGARY) Q  . ;  . I ACTION="UPDATE" D SCHEDULE(.MSGARY) Q  . ;  . I ACTION="CHECKIN" D CHECKIN(.MSGARY) Q  . ;  . I ACTION="CHECKOUT" D CHECKOUT(.MSGARY) Q  . ;  . I ACTION="CANCEL" D CANCEL(.MSGARY) Q  . ;  . I ACTION="NOSHOW" D NOSHOW(.MSGARY) Q  . ;  . I ACTION="UNDOCHECKIN" D UNCHKIN(.MSGARY) Q  . ;  . I ACTION="UNDOCHECKOUT" D UNCHKOUT(.MSGARY) Q  . ;  . I ACTION="UNDONOSHOW" D UNNOSHOW(.MSGARY) Q  . ;  . I ACTION="UNDOCANCEL" D UNCANCEL(.MSGARY) Q  Q  SCHEDULE(MSGARY) ; Process Schedule messages  ; MSGARY (I,REQ) - Appointment data array from the message in the VistA data structure format.  N PATIEN,CLINIEN,APTYP,APDT  S PATIEN=$G(MSGARY("PATIENT IEN"))  S CLINIEN=$G(MSGARY("CLINIC"))  S APTYP=$G(MSGARY("APPTTYPE"))  S APDT=$G(MSGARY("APPTDT"))  D MAKE^SDMXMAKE(PATIEN,CLINIEN,APTYP,APDT,$G(MSGARY("DURATION")),$G(MSGARY("USER")),$G(MSGARY("ELIGIBILITY")),$G(MSGARY("COMMENT")),$G(MSGARY("PAT INDICATED DATE")),$G(MSGARY("ORDER ID")),$G(MSGARY("WALK IN")),$G(MSGARY("CONSULT ID"))) ;  Q  CHECKIN(MSGARY) ; Process Check In messages  ; MSGARY (I,REQ) - Appointment data array from the message in the VistA data structure format.  ;  D CHKIN^SDMXCHKI($G(MSGARY("PATIENT IEN")),$G(MSGARY("CLINIC")),$G(MSGARY("APPTDT")),$G(MSGARY("CHECKINDT")),$G(MSGARY("USER")))  Q  CHECKOUT(MSGARY) ; Process Check Out messages  ; MSGARY (I,REQ) - Appointment data array from the message in the VistA data structure format.  ;  D CHKOUT^SDMXCHKO($G(MSGARY("PATIENT IEN")),$G(MSGARY("CLINIC")),$G(MSGARY("APPTDT")),$G(MSGARY("CHECKOUTDT")),$G(MSGARY("USER")))  Q  CANCEL(MSGARY) ; Process Cancel messages  ; MSGARY (I,REQ) - Appointment data array from the message in the VistA data structure format.  ;  D CANCAPPT^SDMXCANC($G(MSGARY("PATIENT IEN")),$G(MSGARY("CLINIC")),$G(MSGARY("APPTDT")),$G(MSGARY("USER")),$G(MSGARY("CANCEL REASON")),$G(MSGARY("CANCEL REMARK")))  Q ;  NOSHOW(MSGARY) ; Process No Show messages  ; MSGARY (I,REQ) - Appointment data array from the message in the VistA data structure format.  ;  D NOSHOW^SDMXNS($G(MSGARY("PATIENT IEN")),$G(MSGARY("APPTDT")),$G(MSGARY("USER")),$G(MSGARY("CLINIC")))  Q  UNCHKIN(MSGARY) ; Process Undo Check In messages  ; MSGARY (I,REQ) - Appointment data array from the message in the VistA data structure format.  ;  D CCHKIN^SDMXCHKI($G(MSGARY("PATIENT IEN")),$G(MSGARY("CLINIC")),$G(MSGARY("APPTDT"))) ;  Q ;  UNCHKOUT(MSGARY) ; Process Undo Check Out messages  ; MSGARY (I,REQ) - Appointment data array from the message in the VistA data structure format.  ;  D CHKODEL^SDMXCHKO($G(MSGARY("PATIENT IEN")),$G(MSGARY("APPTDT"))) ;  Q  UNNOSHOW(MSGARY) ; Process undo No Show messages  ; MSGARY (I,REQ) - Appointment data array from the message in the VistA data structure format.  ;  D UNNOSHOW^SDMXNS($G(MSGARY("PATIENT IEN")),$G(MSGARY("APPTDT")),"",$G(MSGARY("CLINIC")))  Q  UNCANCEL(MSGARY) ; Process undo cancel messages  ; MSGARY (I,REQ) - Appointment data array from the message in the VistA data structure format.  ;  D UNCANCEL^SDMXUCAN($G(MSGARY("PATIENT IEN")),$G(MSGARY("CLINIC")),$G(MSGARY("APPTDT")),$G(MSGARY("USER")),$G(MSGARY("DURATION")),$G(MSGARY("CONSULT ID")),$G(MSGARY("ORDER ID")))  Q  ; | | | | | | | | | |

Table : SDMXSCHT Routine

| Routines | Activities | | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Routine Name | **SDMXSCHT** | | | | | | | | | |
| Enhancement Category | New | | Modify | Delete | | | No Change | | | |
| RTM | SBN1A5.1  BN1A5.13  BN1A6  BN1A6.1  BN1A6.3  BN1A6.4  BN2B1.12  BN3B1.5  BN4A1.7  BN4A1.13  BN4A1.24  BN4B1.1  BN6A1.6  BN6C1.5 | | | | | | | | | |
| Related Options |  | | | | | | | | | |
| Related Routines | Routines “Called By” | | | | | Routines “Called” | | | | |
|  | SDMXTRCT | | | | | SDMXFLAG  SDMXGAPT  SDMXCORE  HLOAPI  HLOAPI1  HLOAPI4  DIQ  DGMXHL7  ORMXFMT  XPAR | | | | |
| Routines | Activities | | | | | | | | | |
| **Data Dictionary (DD) References** |  | | | | | | | | | |
| Related Protocols | N/A | | | | | | | | | |
| Related Integration Control Registrations (ICRs) | N/A | | | | | | | | | |
| Data Passing | Input | Output Reference | | | Both | | | Global Reference | Local |
| Input Attribute Name and Definition |  | | | | | | | | | |
| Output Attribute Name and Definition |  | | | | | | | | | |
| Current Logic | | | | | | | | | | |
| N/A - New routine | | | | | | | | | | |
| Modified Logic (Changes are in bold) | | | | | | | | | | |
| SDMXSCHT ;MASS/SEL - Appointment Trigger;8/17/17  ;;5.3;Scheduling;\*\*676\*\*;AUGUST 17,2017  ;;Per VA directive 6402, this routine should not be modified.  MAIN ; Triggers scheduling messages out of VistA when an appointment event occurs.  ; Received from SDAM APPOINTMENT EVENTS PROTOCOL  ; SDATA - uses structure <Appointment IEN>^<DFN>^<Appointment Date/Time>^<Clinic IEN>  ; SDAMEVT - is the scheduling event  ; 1 = MAKE  ; 2 = CANCEL  ; 3 = NO-SHOW  ; 4 = CHECK-IN  ; 5 = CHECK-OUT  ; 6 = STOP CODE ADD  ; 7 = STOP CODE CHANGE  ; 8 = DISPOSITION ADD  ; 9 = DISPOSITION CHANGE  ; If MASS isn't enabled, don't trigger messages  I '$$GET^XPAR("SYS","SDMX MASS ENABLED") Q  ;  N CLINIC,FLAG  S CLINIC=$P($G(SDATA),"^",4)  S FLAG=$$GETFLAG^SDMXFLAG(CLINIC)  I FLAG=0 Q ;not a MASS clinic  ;  I $$INCINTF^SDMXCORE() Q ;Do not send messages from incoming interface  I ($G(SDAMEVT)="")!($G(SDATA)="") Q  I SDAMEVT=4 D CHKIN(SDATA) ;Check In  E I SDAMEVT=5 D CHKOUT(SDATA) ;Check Out  ;E I SDAMEVT=2 D CANCEL(SDATA) ;Cancel  Q  CHKIN(SDATA) ; Triggers appointment check in message out of VistA  ; SDATA (I,REQ) - <Appointment IEN>^<DFN>^<Appointment Date/Time>^<Clinic IEN>  ;  N OK  S OK=$$BUILDHLO(SDATA,"S14")  Q  CHKOUT(SDATA) ;Triggers appointment check out message out of VistA  ; SDATA (I,REQ) - <Appointment IEN>^<DFN>^<Appointment Date/Time>^<Clinic IEN>  ;  N OK  S OK=$$BUILDHLO(SDATA,"S14")  Q  CANCEL(SDATA) ; Triggers appointment cancel message out of VistA  ; SDATA (I,REQ) - <Appointment IEN>^<DFN>^<Appointment Date/Time>^<Clinic IEN>  ;  N OK  S OK=$$BUILDHLO(SDATA,"S15")  Q  BUILDHLO(SDATA,EVENT,LINK) ; Builds scheduling message using HLO and queues it  ; SDATA (I,REQ) - <Appointment IEN>^<DFN>^<Appointment Date/Time>^<Clinic IEN>  ; EVENT (I,REQ,DEFAULT:S14) - Scheduling event (Example: S14)  ; LINK (I,OPT,DEFAULT:SD SIU OUT) - HL Logical Link to use.  ;  N PARMS,MSG,ERROR,SEG,HL,SCIEN,DFN,SDT,CLINIEN,APPTARY,PROVNUM,RET,OBXNUM  S RET=""  S OBXNUM=0  I $G(LINK)="" S LINK="SD SIU OUT"  S SCIEN=$P(SDATA,"^",1) ;Appointment IEN  S DFN=$P(SDATA,"^",2) ;Patient DFN  S SDT=$P(SDATA,"^",3) ;Appointment Date/Time  S CLINIEN=$P(SDATA,"^",4) ;Clinic IEN  I $G(EVENT)="" S EVENT="S14"  S PARMS("EVENT")=EVENT  S PARMS("MESSAGE TYPE")="SIU"  D GETAPPT^SDMXGAPT(DFN,SDT,.APPTARY)  I '$$NEWMSG^HLOAPI(.PARMS,.MSG,.ERROR) S RET="FAIL!" Q RET  D SCH(SDATA,.APPTARY,EVENT,.SEG)  I '$$ADDSEG^HLOAPI(.MSG,.SEG,.ERROR) S RET="FAIL!" Q RET  I $G(APPTARY("COMMENT"))'="" D I RET'="" Q RET  . D NTE(.APPTARY,.SEG)  . I '$$ADDSEG^HLOAPI(.MSG,.SEG,.ERROR) S RET="FAIL!"  D PID^DGMXHL7(DFN,1,.SEG)  I '$$ADDSEG^HLOAPI(.MSG,.SEG,.ERROR) S RET="FAIL!" Q RET  D PV1(SDATA,.APPTARY,.SEG)  I '$$ADDSEG^HLOAPI(.MSG,.SEG,.ERROR) S RET="FAIL!" Q RET  I $G(APPTARY("PAT INDICATED DATE"))'="" D  . S OBXNUM=OBXNUM+1  . D OBXPID(OBXNUM,APPTARY("PAT INDICATED DATE"),.SEG)  . I '$$ADDSEG^HLOAPI(.MSG,.SEG,.ERROR) S RET="FAIL!"  I $G(APPTARY("ELIGIBILITY"))'="" D  . S OBXNUM=OBXNUM+1  . D OBXELIG(OBXNUM,APPTARY("ELIGIBILITY"),.SEG)  . I '$$ADDSEG^HLOAPI(.MSG,.SEG,.ERROR) S RET="FAIL!"  I RET'="" Q RET  D RGS(.APPTARY,.SEG)  I '$$ADDSEG^HLOAPI(.MSG,.SEG,.ERROR) S RET="FAIL!" Q RET  D AIP(.APPTARY,.SEG)  I '$$ADDSEG^HLOAPI(.MSG,.SEG,.ERROR) S RET="FAIL!" Q RET  S PARMS("SENDING APPLICATION")="SD-SIU-OUT"  S WHOTO("RECEIVING APPLICATION")="MASS"  S WHOTO("FACILITY LINK NAME")=LINK  S RET=$$SENDONE^HLOAPI1(.MSG,.PARMS,.WHOTO,.ERROR)  Q RET  SCH(SDATA,APPTARY,EVENT,SEG) ;Builds SCH segment  ; SCH-6 Appointment Reason - APPTARY("CANCEL REASON") for S15, APPTARY("STATUS") for everything else  ; SCH-8 Appointment Type - APPTARY("APPTTYPE")  ; SCH-9 Appointment Duration - APPTARY("DURATION")  ; SCH-10 Apointment Duration Unit - "MIN"  ; SCH-11.4 Appointment Date/Time - APPTARY("APPTDT")  ; SCH-16 Appointment Scheduling User - APPTARY("USER"/"CHECKIN USER"/"CHECKOUT USER")  ; SCH-20 Appointment Scheduling User - APPTARY("USER"/"CHECKIN USER"/"CHECKOUT USER")  ; SCH-25 Appointment Status - APPTARY("STATUS")  ; SCH-27 Appointment linked consult order (Will be the ID from the OR file  ;  ; SDATA (I,REQ) - <Appointment IEN>^<DFN>^<Appointment Date/Time>^<Clinic IEN>  ; APPTARY (IO,REQ) - Appointment array from GETAPPT^SDMXGAPT  ; EVENT (I,REQ) - message event (Example: S15)  ; SEG (O,REQ) - HLO Segment  ;  N SCIEN,APPTTYPE,APPTDUR,DURUNIT,STODAY,SDSTAT,USER,CONSORD,CONS,CLINIEN,DFN  K SEG S SEG=""  S SCIEN=$P(SDATA,"^",1)  S DFN=$P(SDATA,"^",2)  S SDT=$P(SDATA,"^",3)  S CLINIEN=$P(SDATA,"^",4)  I SDAMEVT=4 S USER=$G(APPTARY("CHECKIN USER"))  I SDAMEVT=5 S USER=$G(APPTARY("CHECKOUT USER"))  I $G(USER)="" S USER=$G(APPTARY("USER"))  D SET^HLOAPI(.SEG,"SCH",0) ;SCH  S APPTTYPE=$G(APPTARY("APPTTYPE"))  I EVENT="S15" D ;SCH-6  . D SET^HLOAPI(.SEG,$G(APPTARY("CANCEL REASON")),6,1)  . D SET^HLOAPI(.SEG,$$GET1^DIQ(409.2,$G(APPTARY("CANCEL REASON")),.01),6,2)  E D SET^HLOAPI(.SEG,$G(APPTARY("STATUS")),6)  D SET^HLOAPI(.SEG,APPTTYPE,8,1) ;SCH-8  D SET^HLOAPI(.SEG,$$GET1^DIQ(409.1,$G(APPTARY("APPTTYPE")),.01),8,2)  S APPTDUR=$G(APPTARY("DURATION"))  D SET^HLOAPI(.SEG,APPTDUR,9) ;SCH-9  S DURUNIT="MIN"  D SET^HLOAPI(.SEG,DURUNIT,10) ;SCH-10  D SETDT^HLOAPI4(.SEG,$G(APPTARY("APPTDT")),11,4) ;SCH-11.4  D SETXCN^DGMXHL7(.SEG,USER,16) ;SCH-16  D SETXCN^DGMXHL7(.SEG,USER,20) ;SCH-20  D SET^HLOAPI(.SEG,$G(APPTARY("STATUS")),25) ;SCH-25  S CONS=$G(APPTARY("CONSULT"))  I CONS'="" D  . S CONSORD=$$GET1^DIQ(123,CONS,.03)  I $G(CONSORD)'="" D  . D SET^HLOAPI(.SEG,CONSORD,27,1) ;SCH-27.1  . D SET^HLOAPI(.SEG,$$TRANORCD^ORMXFMT(2,"^OR(100"),27,2) ;SCH-27.2  Q  PV1(SDATA,APPTARY,SEG) ;Builds PV1 segment  ; PV1-1 Set ID - 1  ; PV1-2 Patient Class - O  ; PV1-3 Assigned Patient Location - Clinic IEN  ;  ; SDATA (I,REQ) - <Appointment IEN>^<DFN>^<Appointment Date/Time>^<Clinic IEN>  ; APPTARY (IO,REQ) - Appointment array from GETAPPT^SDMXGAPT  ; SEG (O,REQ) - HLO Segment  ;  K SEG S SEG=""  N CLINIEN  S CLINIEN=$P(SDATA,"^",4)  D SET^HLOAPI(.SEG,"PV1",0) ;PV1  D SET^HLOAPI(.SEG,"O",2) ;PV1-2  D SET^HLOAPI(.SEG,APPTARY("CLINIC NAME"),3) ;PV1-3  I $G(APPTARY("CHECKIN DT"))'="" D SETDT^HLOAPI4(.SEG,APPTARY("CHECKIN DT"),44)  I $G(APPTARY("CHECKOUT DT"))'="" D SETDT^HLOAPI4(.SEG,APPTARY("CHECKOUT DT"),45)  Q  NTE(APPTARY,SEG) ;Builds NTE segment  ; NTE-3 Comment  ;  ; APPTARY (IO,REQ) - Appointment array from GETAPPT^XMASSGETAPPT  ; SEG (O,REQ) - HLO Segment  ;  K SEG S SEG=""  D SET^HLOAPI(.SEG,"NTE",0) ;NTE  D SET^HLOAPI(.SEG,$G(APPTARY("COMMENT")),3,1) ;NTE-3.1  Q  RGS(APPTARY,SEG) ;Builds RGS segment  ; RGS-1 Set ID - 1  ; RGS-3 Resource Group ID - Clinic IEN  ;  ; APPTARY (IO,REQ) - Appointment array from GETAPPT^XMASSGETAPPT  ; SEG (O,REQ) - HLO Segment  ;  K SEG S SEG=""  D SET^HLOAPI(.SEG,"RGS",0) ;RGS  D SET^HLOAPI(.SEG,1,1) ;RGS-1  D SET^HLOAPI(.SEG,APPTARY("CLINIC"),3,1) ;RGS-3.1  D SET^HLOAPI(.SEG,APPTARY("CLINIC NAME"),3,2) ;RGS-3.2  Q  AIP(APPTARY,SEG) ;Builds AIP segment  ; AIP-3 Schedulable Resource - Clinic IEN  ;  ; APPTARY (IO,REQ) - Appointment array from GETAPPT^XMASSGETAPPT  ; SEG (O,REQ) - HLO Segment  ;  K SEG S SEG=""  D SET^HLOAPI(.SEG,"AIP",0) ;AIP  D SET^HLOAPI(.SEG,APPTARY("CLINIC"),3,1) ;AIP-3.1  D SET^HLOAPI(.SEG,APPTARY("CLINIC NAME"),3,2) ;AIP-3.2  Q  OBXELIG(SETID,ELIG,SEG) ;Builds OBX segment for eligibility  ; OBX-1 Set ID  ; OBX-3 "APPT ELIGIBILITY"  ; OBX-5 Eligibility  ;  ; SETID (I,REQ) - Set ID that's set in OBX-1  ; ELIG (I,REQ) - Eligibility value from appointment array  ; SEG (O,REQ) - HLO Segment  ;  K SEG S SEG=""  D SET^HLOAPI(.SEG,"OBX",0) ;OBX  D SET^HLOAPI(.SEG,SETID,1) ;OBX-1  D SET^HLOAPI(.SEG,"APPT ELIGIBILITY",3) ;OBX-3  D SET^HLOAPI(.SEG,ELIG,5,1) ;OBX-5.1  D SET^HLOAPI(.SEG,"NAME",5,2) ;OBX-5.2  Q  OBXPID(SETID,PID,SEG) ;Builds OBX segment for patient indicated date  ; OBX-1 Set ID  ; OBX-3 "REQUESTED DATE"  ; OBX-5 Patient Indicated Date  ;  ; SETID (I,REQ) - Set ID that's set in OBX-1  ; PID (I,REQ) - Patient indicated date  ; SEG (O,REQ) - HLO Segment  ;  K SEG S SEG=""  D SET^HLOAPI(.SEG,"OBX",0) ;OBX  D SET^HLOAPI(.SEG,SETID,1) ;OBX-1  D SET^HLOAPI(.SEG,"REQUESTED DATE",3) ;OBX-3  D SETDT^HLOAPI4(.SEG,PID,5) ;OBX-5  Q  ; | | | | | | | | | | |

Table : SDMXTRCT Routine

| Routines | Activities | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Routine Name | **SDMXTRCT** | | | | | | | | |
| Enhancement Category | New | | Modify | Delete | | | No Change | | |
| RTM | BN1A6, BN2B1.11, BN9.3, BN10.6 | | | | | | | | |
| Related Options |  | | | | | | | | |
| Related Routines | Routines “Called By” | | | | | Routines “Called” | | | |
|  |  | | | | | DICRW  DIC  DICN  SDMXSCHT  SDMXFLAG | | | |
| Routines | Activities | | | | | | | | |
| Data Dictionary (DD) References | ^DPT(DFN  ^SC(IEN | | | | | | | | |
| Related Protocols | N/A | | | | | | | | |
| Related Integration Control Registrations (ICRs) | N/A | | | | | | | | |
| Data Passing | Input | Output Reference | | | Both | | | Global Reference | Local |
| Input Attribute Name and Definition |  | | | | | | | | |
| Output Attribute Name and Definition |  | | | | | | | | |
| Current Logic | | | | | | | | | |
| N/A - New routine | | | | | | | | | |
| Modified Logic (Changes are in bold) | | | | | | | | | |
| SDMXTRCT ;MASS/JEO,RPC - EXTRACTS THE SCHEDULED APPOINTMENTS FROM THE PATIENT FILE AND TRIGGERS SIU messages;9/15/2017  ;;5.3;Scheduling;\*\*676\*\*;9/15/2017  ;;Per VA directive 6402, this routine should not be modified.  ; This utility will trigger future appointments out the SD-SIU-OUT HLO SIU interface.  ; The future appointments may be extracted started from the current, or given future day.  Q  ;  FUTAP ;EXTRACTS THE FUTURE APPOINTMENTS FOR THE ACTIVE CLINICS  ; DT - Current date in VistA format  ; Y - The start date in VistA format  N Y,SDT,SDT1,CLINIC,DFN,MASSAPT,ISMASS,CLCOUNT,SDATA,U,DT,PATCOUNT,TCOUNT,CLINARY,FIRST,DIR,ACTION  W #  ; Get the start date for the conversion  S Y="",TCOUNT=0,U="^"  D DT^DICRW ;Today  W !,"This utility triggers future active appointments to the",!  W "SD\_SIU\_OUT HL7 appointment interface",!!!  K DIR  S DIR(0)="D^"\_DT,DIR("A")="Enter a start date"  D ^DIR  I Y="^" Q  S SDT=Y-.1  ; Get Action  K DIR  S DIR(0)="SB^1:SEND ONLY APPOINTMENTS FOR CLINICS WITH THE MASS FLAG;2:SEND ALL APPOINTMENTS IN THE SYSTEM;3:CHOOSE CLINICS TO TRIGGER APPOINTMENT FOR"  S DIR("A")="Select a trigger action: 1 - Send MASS appts, 2 - Send all appts , 3 - select individual clinics"  D ^DIR  S ACTION=$G(Y)  I '$$INSTRING^SDMXCORE(ACTION,"1,2,3",",") W !,"NO ACTION WILL BE TAKEN" Q  ; SEND ALL  I ACTION=1 D LOOP(.TCOUNT,SDT,1)  I ACTION=2 D LOOP(.TCOUNT,SDT)  I ACTION=3 D MANUAL(.TCOUNT,SDT)  ;  ; Final output  W !,"TOTAL APPOINTMENTS TRIGGERED: "\_TCOUNT  Q  MANUAL(TCOUNT,SDT) ;  N CLCOUNT,FIRST,STOP,CLINIC,CLINARY,SDT1,DFN,THREADS,LINK,COUNT  W !  S DIC="^SC(",DIC(0)="AEMZQ",DIC("A")="SELECT CLINIC: ",DIC("S")="I $P(^(0),U,3)=""C"",'$G(^(""OOS""))"  S CLCOUNT=0,STOP=0,FIRST=1  F D Q:STOP  . D ^DIC  . I Y=-1 S STOP=1 Q  . S CLINIC=$P(Y,U,1)  . I $G(CLINARY(CLINIC))'="" Q  . S CLINARY(CLINIC)=1  . S CLCOUNT=CLCOUNT+1  . I FIRST=1 S DIC("A")="AND CLINIC: ",FIRST=""  ;  I CLCOUNT=0 Q  S THREADS=$$GET^XPAR("SYS","SDMX CONV THREADS")  ; Process the list  S CLINIC=""  F S CLINIC=$O(CLINARY(CLINIC)) Q:CLINIC="" D  . ;  . ; Loop over all appointment DATE/TIME on this clinic.  . S SDT1=SDT  . F S SDT1=$O(^SC(CLINIC,"S",SDT1)) Q:SDT1="" D  . . ;  . . ; Loop over all appointments at this time  . . S CLCOUNT=0  . . F S CLCOUNT=$O(^SC(CLINIC,"S",SDT1,1,CLCOUNT)) Q:CLCOUNT="" D  . . . ; set app if multi threading  . . . I $G(THREADS) D  . . . . S COUNT=TCOUNT#THREADS+1  . . . . S LINK="SD SIU O"\_COUNT  . . . . S ^XRPC(LINK)=1  . . . . ;  . . . ; GET PATIENT COUNTER  . . . S DFN=$P($G(^SC(CLINIC,"S",SDT1,1,CLCOUNT,0)),"^",1)  . . . I DFN="" Q  . . . ;  . . . ; TRIGGER APPOINTMENT  . . . S SDATA=CLCOUNT\_U\_DFN\_U\_SDT1\_U\_CLINIC  . . . D BUILDHLO^SDMXSCHT(SDATA,"S12",LINK)  . . . S TCOUNT=TCOUNT+1  Q  ;  LOOP(TCOUNT,SDT,MASSAPT) ;All appointment loop  N CLINIC,ISMASS,SDT1,CLCOUNT,DFN,SDATA,THREADS,LINK,COUNT  S CLINIC=""  S THREADS=$$GET^XPAR("SYS","SDMX CONV THREADS")  ; Loop through all clinics  F S CLINIC=$O(^SC(CLINIC)) Q:CLINIC="" D  . ;  . ; Check if Clinic is for MASS  . S ISMASS=""  . I MASSAPT=1 S ISMASS=$$ISMASS(CLINIC)  . I ISMASS=0 Q ;Set to not send CLINICs not scheduled in MASS  . ;  . ; Loop over all appointment DATE/TIME on this clinic.  . S SDT1=SDT  . F S SDT1=$O(^SC(CLINIC,"S",SDT1)) Q:SDT1="" D  . . ;  . . ; Loop over all appointments at this time  . . S CLCOUNT=0  . . F S CLCOUNT=$O(^SC(CLINIC,"S",SDT1,1,CLCOUNT)) Q:CLCOUNT="" D  . . . ; set app if multi threading  . . . I $G(THREADS) D  . . . . S COUNT=TCOUNT#THREADS+1  . . . . S LINK="SD SIU O"\_COUNT  . . . . S ^XRPC(LINK)=1  . . . ;  . . . ; GET PATIENT COUNTER  . . . S DFN=$P($G(^SC(CLINIC,"S",SDT1,1,CLCOUNT,0)),"^",1)  . . . I DFN="" Q  . . . ;  . . . ; TRIGGER APPOINTMENT  . . . S SDATA=CLCOUNT\_U\_DFN\_U\_SDT1\_U\_CLINIC  . . . D BUILDHLO^SDMXSCHT(SDATA,"S12",LINK)  . . . S TCOUNT=TCOUNT+1  Q  ISMASS(CLINIC) ;Get MASS clinic flag  N RET  S RET=$$GETFLAG^SDMXFLAG(CLINIC)  Q RET  ; | | | | | | | | | |

Table : SDMXUCAN Routine

| Routines | Activities | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Routine Name | **SDMXUCAN** | | | | | | | | |
| Enhancement Category | New | | Modify | Delete | | | No Change | | |
| RTM | BN4A1.14 | | | | | | | | |
| Related Options |  | | | | | | | | |
| Related Routines | Routines “Called By” | | | | | Routines “Called” | | | |
|  | SDMXSCHP | | | | | DIE  DIC  DICN  SDMXCORE  SDCNSLT  SDMXCANC  XLFDT | | | |
| Routines | Activities | | | | | | | | |
| Data Dictionary (DD) References | ^DPT(DFN  ^OR(100  ^SC(IEN | | | | | | | | |
| Related Protocols | N/A | | | | | | | | |
| Related Integration Control Registrations (ICRs) | N/A | | | | | | | | |
| Data Passing | Input | Output Reference | | | Both | | | Global Reference | Local |
| Input Attribute Name and Definition |  | | | | | | | | |
| Output Attribute Name and Definition |  | | | | | | | | |
| Current Logic | | | | | | | | | |
| N/A - New routine | | | | | | | | | |
| Modified Logic (Changes are in bold) | | | | | | | | | |
| SDMXUCAN ;MASS/DAP - Epic Uncancel appointment;8/17/17  ;;5.3;Scheduling;\*\*676\*\*;AUGUST 22,2017  ;;Per VA directive 6402, this routine should not be modified.  Q  UNCANCEL(PATIEN,CLINIEN,APPTDT,USER,DURATION,CSLTIEN,ORDERID) ;ENTRY POINT TO UNCANCEL A PREVOIUSLY EXISTING APPOINTMENT  ; WRAPPER FOR THE FILING TAG. DOES LIGHT VALIDATION/GETTING  ; PASSED IN PARAMETERS  ; REQUIRED FOR FUNCTIONALITY:  ; PATIEN (I,REQ) - PATIENT ID  ; CLINIEN (I,REQ) - CLINIC ID  ; APPTDT (I,REQ) - APPOINTMENT DATE+TIME  ; REQUIRED BY WORKFLOW:  ; USER (I,REQ) - DATA ENTRY CLERK  ; DURATION (I,REQ) - APPOINTMENT LENGTH IN MINUTES  ; CSLTIEN (I,OPT) - CONSULT ID. WILL LOOK UP VIA ORDER ID IF ""  ; ORDERID (I,OPT) - ORDER ID  N OK ;  ;CHECK INPUT PARAMETERS  I ((PATIEN="")!(CLINIEN="")!(APPTDT="")!(USER="")!(DURATION="")) Q "MISSING REQUIRED PARAMETERS" ;REQUIRED FIELDS TO FILE.  ;LOOK UP CONSULT ID IF NOT PASSED IN  I $G(CSLTIEN)="",$G(ORDERID)'="" S CSLTIEN=$G(^OR(100,ORDERID,4)) ;  S OK=$$FILEUNCN($G(PATIEN),$G(CLINIEN),$G(APPTDT),$G(USER),$G(DURATION),$G(CSLTIEN))  Q OK  FILEUNCN(PATIEN,CLINIEN,APPTDT,USER,DURATION,CSLTIEN) ;FILES VALIDATED UNCANCEL APPOINTMENT DATA  ; PASSED IN PARAMETERS  ; REQUIRED FOR FUNCTIONALITY:  ; PATIEN (I,REQ) - PATIENT ID  ; CLINIEN (I,REQ) - CLINIC ID  ; APPTDT (I,REQ) - APPOINTMENT DATE+TIME  ; REQUIRED BY WORKFLOW:  ; USER (I,REQ) - DATA ENTRY CLERK  ; DURATION (I,REQ) - APPOINTMENT LENGTH IN MINUTES  ; CSLTIEN (I,OPT) - CONSULT ID. WILL LOOK UP VIA ORDER ID IF ""  ; ORDERID (I,OPT) - ORDER ID  ;  N NOW,CLINLINE,DIC,DA,X,Y,DLAYGO,DIE,DR,SCNAME,TMPYCLNC,TMPD,FILES,MASSFDA,MASSIENS,MASSMSG  S NOW=$$NOW^XLFDT()  I '$D(^DPT(PATIEN,"S",APPTDT,0)) Q "APPOINTMENT DOESN'T EXIST"  ; FILE 2.98  S MASSIENS=APPTDT\_","\_PATIEN\_"," ;  S MASSFDA(2.98,MASSIENS,"0.01")=CLINIEN ;CLINIEN  S MASSFDA(2.98,MASSIENS,"3")="" ;STATUS  S MASSFDA(2.98,MASSIENS,"14")="" ;NO-SHOW/CANCELLED BY  S MASSFDA(2.98,MASSIENS,"15")="" ;NO-SHOW/CANCEL DATE/TIME  S MASSFDA(2.98,MASSIENS,"16")="" ;CANCELLATION REASON  S MASSFDA(2.98,MASSIENS,"17")="" ;CANCELLATION REMARKS  S MASSFDA(2.98,MASSIENS,"18")="" ;APPT. CANCELLED  S MASSFDA(2.98,MASSIENS,"19")=USER ;DATA ENTRY CLERK  S MASSFDA(2.98,MASSIENS,"20")=NOW ;DATE APPT. MADE  D FILE^DIE("K","MASSFDA","MASSMSG")  ;  ; ADD APPT TO FILE 44  ;  ;Use DIC to set up Y which will find the appointment in SC  K DIC,DA,X,Y,DLAYGO  S FILES="1////"\_DURATION\_";7////"\_USER\_";8////"\_NOW\_";688////"\_CSLTIEN  S DIC="^SC("\_CLINIEN\_",""S"","\_APPTDT\_",1,"  S DA(2)=CLINIEN  S DA(1)=APPTDT  S X=PATIEN ;Set node levels 2, 1 and the lookup index  S DIC("P")="44.003PA"  S DIC(0)="L"  S DLAYGO=44.003  D ^DIC  S CLINLINE=+$G(Y)  ;File the data  I CLINLINE=-1 D ;No existing line should be the usual case  . S DIC("DR")=FILES  . D FILE^DICN  S DIE=DIC ;The subfile's root  S DA=CLINLINE  S DR=FILES  D ^DIE  ;  ;Update OR and GMR with consult link  I $G(CSLTIEN)="" Q $G(MASSMSG)  S SCNAME=$P($G(^SC(CLINIEN,0)),"^")  S TMPYCLNC=CLINIEN\_"^"\_SCNAME  S TMPD=$P($G(^SC(CLINIEN,"S",APPTDT,1,CLINLINE,0)),"^",4)  D EDITCS^SDCNSLT(APPTDT,TMPD,TMPYCLNC,CSLTIEN)  ;  Q 1  ; | | | | | | | | | |

Table : ORMXFMT1 Routine

| Routines | Activities | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Routine Name | ORMXFMT1 | | | | | | | |
| Enhancement Category | New | | Modify | Delete | | No Change | | |
| RTM |  | | | | | | | |
| Related Options |  | | | | | | | |
| Related Routines | Routines “Called By” | | | | Routines “Called” | | | |
|  | ORMXFMT | | | | ICDCODE  ORWDBA3  SDMXCORE  XPAR | | | |
| Routines | Activities | | | | | | | |
| Data Dictionary (DD) References | ^DPT(  ^GMR(123  ^OR(100  ^SC(  ^VA(200 | | | | | | | |
| Related Protocols | N/A | | | | | | | |
| Related Integration Control Registrations (ICRs) | N/A | | | | | | | |
| Data Passing | Input | Output Reference | | | Both | | Global Reference | Local |
| Input Attribute Name and Definition |  | | | | | | | |
| Output Attribute Name and Definition |  | | | | | | | |
| Current Logic | | | | | | | | |
| N/A - New routine | | | | | | | | |
| Modified Logic (Changes are in bold) | | | | | | | | |
| **ORMXFMT1 ;MASS/AWS - ORDER FORMATTING;11/30/17  ;;3.0;Order Entry/Results Reporting;\*\*470\*\*;November 30,2017  ;;Per VA DIRECTIVE 6402, this routine should not be modified.  ; INITARR(HLOARR) ; Initialize the HLO array containing order information  ; The format of the HLOARR is as follows:  ;  ; HLOARR("ORDER ID") - <Unique order ID (per global)>^<A string to identify the global in  ; VistA that the ID comes from>  ; HLOARR("PATIENT") - <Patient Identifier (e.g. IEN)>^<A string to identify Patient ID type>  ; HLOARR("ORDERABLE") - <Identifier for itemordered>^<Free text name of orderable>^<A string to  ; identify the global in VistA that the Orderable ID comes from>  ; HLOARR("ORDERING PHYSICIAN") - <Physician Identifier>^<Physician Last Name>^<Physician First Name>^<A string to  ; identify Physician ID type>  ; HLOARR("ORDER DATE") - <VistA format for Date/Time the order was placed>  ; HLOARR("ORDER CODE") - <Unique order code, e.g."NW","CA", etc.>  ; HLOARR("START DATE") - <VistA format for Date/Time the order is to be performed>  ; HLOARR("STOP DATE") - <VistA format for Date/Time the order expires>  ; HLOARR("DIAGNOSIS",0) - <Integer number of diagnoses>  ; HLOARR("DIAGNOSIS",#) - <Diagnosis ID>^<Free text name of diagnosis>^<A string to identify the  ; global in VistA that the Diagnosis ID comes from>  ; HLOARR("CLINIC") - <Clinic ID where order was placed>^<Clinic facility ID>^<Clinic Name>^<A string  ; to identify the clinic ID type>  ; HLOARR("REASON FOR STUDY",0) - <Number of lines for the reason for study>  ; HLOARR("REASON FOR STUDY",#) - <Free text reason for study>  ; HLOARR("COMMENT",0) - <Number of lines for the comments>  ; HLOARR("COMMENT",#) - <Free text comments>  ; HLOARR("NO LATER THAN") - <No later than flag>  ; HLOARR("RECUR") - <Total occurrences>  ; HLOARR("FREQUENCY") - <Frequency>  ; HLOARR("CANCEL REASON") - <Cancel reason for canceled orders>  ; HLOARR("URGENCY") - <Urgency of a consult>  ;  K HLOARR S HLOARR=""  ;  S HLOARR("ORDER ID")=""  S HLOARR("PATIENT")=""  S HLOARR("ORDERABLE")=""  S HLOARR("ORDERING PHYSICIAN")=""  S HLOARR("ORDER DATE")=""  S HLOARR("ORDER CODE")=""  S HLOARR("START DATE")=""  S HLOARR("STOP DATE")=""  S HLOARR("DIAGNOSIS",0)=0  S HLOARR("CLINIC")=""  S HLOARR("REASON FOR STUDY",0)=0  S HLOARR("COMMENT",0)=0  S HLOARR("NO LATER THAN")=""  S HLOARR("FREQUENCY")=""  S HLOARR("RECUR")=""  S HLOARR("CANCEL REASON")=""  S HLOARR("URGENCY")=""  ;  Q  ; FMTPAT(IEN) ; Formats the patient ID given the IEN for the patient  ;  ; IEN (I,REQ) - IEN for the patient  ;  ; Returns the formatted patient ID. If the IEN passed in does not match  ; to a patient, the null string will be returned. Format is "<ID>^<TYPE>"  ;  S IEN=$G(IEN)  I IEN="" Q ""  ;  I '$D(^DPT(IEN)) Q ""  ;  Q IEN\_"^IEN"  ; FMTPROV(IEN) ; Formats the provider ID given the IEN for the provider  ;  ; IEN (I,REQ) - IEN for provider to match from file #200  ;  ; Returns either the NPI ID (if one exists) or the IEN for the provider.  ; If the IEN passed in does not match to a provider from file #200, the  ; null string will be returned. Format is "<ID>^<LAST NAME>^<FIRST NAME>^<TYPE>"  ;  N NPI,NAME,FIRSTNM,LASTNM,FACILITY  ;  S IEN=$G(IEN)  I IEN="" Q ""  ;  I '$D(^VA(200,IEN)) Q ""  ;  ; Set the name of the provider  S NAME=$P($G(^VA(200,IEN,0)),"^",1)  I NAME'="" D  . S LASTNM=$P(NAME,",",1)  . S FIRSTNM=$P(NAME,",",2)  ;  ; If the NPI ID exists, use that ID for the provider ID  S NPI=$P($G(^VA(200,IEN,"NPI")),"^",1) ; NPI ID  I NPI'="" Q NPI\_"^"\_LASTNM\_"^"\_FIRSTNM\_"^"\_"NPI"  ;  ; Get Facility ID  S FACILITY=$$GET^XPAR("SYS","ORMX FACILITY ID")  ;  ; Use the VA ID as the default ID  Q IEN\_"^"\_LASTNM\_"^"\_FIRSTNM\_"^"\_"VA"\_FACILITY  ; FMTCLIN(IEN) ; Formats the clinic ID given the IEN for the clinic  ;  ; IEN (I,REQ) - IEN for clinic  ;  ; Returns either the clinic ID. If the IEN passed in does not match to a clinic, the  ; null string will be returned. Format is "<CLINIC ID>^<CLINIC FACILITY ID>^<CLINIC NAME>^<TYPE>"  ;  N FACILITY  ;  S IEN=$G(IEN)  I IEN="" Q ""  ;  I '$D(^SC(IEN)) Q ""  ;  ; Get Facility ID  S FACILITY=$$GET^XPAR("SYS","ORMX FACILITY ID")  ;  Q IEN\_"^"\_FACILITY\_"^"\_$P($G(^SC(IEN,0)),"^",1)\_"^"\_"SC"\_FACILITY  ; FMTDIAG(IEN) ; Formats the diagnosis given the IEN for the diagnosis  ;  ; IEN (I,REQ) - IEN for diagnosis to match from file #80  ;  ; Returns the formatted diagnosis. If the IEN passed in does not match  ; to a provider from file #80, the null string will be returned. Format  ; is "<DIAGNOSIS ID>^<DIAGNOSIS NAME>^<TYPE>"  ;  N ICD9,DXV,ORDDTTM,DXR  ;  S IEN=$G(IEN)  I IEN="" Q ""  ;  S ORDDTTM=$$ORFMDAT^ORWDBA3(ORDIEN)  S DXR=$$ICDDX^ICDCODE(IEN,ORDDTTM)  I +DXR=-1 Q "" ; An error occured when retrieving codes  ;  ; Get diagnosis verbiage and ICD code  S ICD9=$P(DXR,"^",2)  S DXV=$P(DXR,"^",4)  ;  I (ICD9="")!(DXV="") Q ""  ;  Q ICD9\_"^"\_DXV\_"^ICD9"  ; FMTORCM(ORDIEN,HLOARR) ; Formats the order comments given the order ID  ; This will format the HLOARR("COMMENT") array for orders. This should  ; not be called for consult orders.  ;  ; ORDIEN (I,REQ) - IEN for the order  ; HLOARR (IO,REQ) - Array containing the order information  ;  N LINECNT,CMTNODE,TOTLINES,CMTCNT  ;  S ORDIEN=$G(ORDIEN)  I ORDIEN="" Q  ;  ; Add the comments in HLOARR  S LINECNT=1  S CMTNODE=$O(^OR(100,ORDIEN,4.5,"ID","COMMENT",""))  I CMTNODE'="" D  . ;  . S TOTLINES=$P($G(^OR(100,ORDIEN,4.5,CMTNODE,2,0)),"^",3)  . I LINECNT>1 S HLOARR("COMMENT",LINECNT)=" ",LINECNT=LINECNT+1  . S HLOARR("COMMENT",LINECNT)="COMMENTS",LINECNT=LINECNT+1  . ;  . ; Comments will have multiple lines so we have to loop through them  . F CMTCNT=1:1:TOTLINES D  . . S HLOARR("COMMENT",LINECNT)=$G(^OR(100,ORDIEN,4.5,CMTNODE,2,CMTCNT,0)),LINECNT=LINECNT+1  ;  S TOTLINES=$P($G(^OR(100,ORDIEN,8,1,5,0)),"^",3)  I TOTLINES'="" D  . ;  . ; Add a blank line to deliniate between comments  . I LINECNT>1 S HLOARR("COMMENT",LINECNT)=" ",LINECNT=LINECNT+1  . E S HLOARR("COMMENT",LINECNT)="COMMENTS",LINECNT=LINECNT+1  . ;  . F CMTNODE=1:1:TOTLINES D  . . S HLOARR("COMMENT",LINECNT)=$G(^OR(100,ORDIEN,8,1,5,CMTNODE,0)),LINECNT=LINECNT+1  ;  S HLOARR("COMMENT",0)=LINECNT-1  ;  Q  ; FMTGMRCM(ORDIEN,HLOARR) ; Formats the consult comments given the order ID  ; This will format the HLOARR("COMMENT") array for consults. This should  ; not be called for orders other than consults.  ;  ; ORDIEN (I,REQ) - IEN for the consult order  ; HLOARR (IO,REQ) - Array containing the order information  ;  N GMRIEN,LINECNT,CMTNODE,TOTLINES,CMTCNT,CMTSTS  ;  S ORDIEN=$G(ORDIEN)  I ORDIEN="" Q  ;  S GMRIEN=$O(^GMR(123,"AC",ORDIEN,""))  I GMRIEN="" Q  ;  S LINECNT=1  I $D(^GMR(123,GMRIEN,30)) D  . ; Add a blank line to deliniate between comments  . I LINECNT>1 S HLOARR("COMMENT",LINECNT)="",LINECNT=LINECNT+1  . S HLOARR("COMMENT",LINECNT)="PROVISIONAL DIAGNOSIS",LINECNT=LINECNT+1  . S HLOARR("COMMENT",LINECNT)=$G(^GMR(123,GMRIEN,30)),LINECNT=LINECNT+1  ;  S TOTLINES=$P($G(^OR(100,ORDIEN,8,1,5,0)),"^",3)  I TOTLINES'="" D  . ;  . I LINECNT>1 S HLOARR("COMMENT",LINECNT)=" ",LINECNT=LINECNT+1  . S HLOARR("COMMENT",LINECNT)="COMMENTS",LINECNT=LINECNT+1  . ;  . F CMTNODE=1:1:TOTLINES D  . . S HLOARR("COMMENT",LINECNT)=$G(^OR(100,ORDIEN,8,1,5,CMTNODE,0)),LINECNT=LINECNT+1  . ;  . S HLOARR("COMMENT",0)=LINECNT-1  ;  F CMTNODE=1:1:$P($G(^GMR(123,GMRIEN,40,0)),"^",3) D  . ;  . ; Only add comments if its an "Update", "Forward", or "Cancel" action  . S CMTSTS=$P($G(^GMR(123,GMRIEN,40,CMTNODE,0)),"^",2)  . I '$$INSTRING^SDMXCORE(CMTSTS,"17,19,20",",") Q  . ;  . ; Quit out if there are no comments  . S TOTLINES=$P($G(^GMR(123,GMRIEN,40,CMTNODE,1,0)),"^",3) I TOTLINES="" Q  . ;  . ; Add a blank line to deliniate between comments  . I LINECNT>1 S HLOARR("COMMENT",LINECNT)=" ",LINECNT=LINECNT+1  . ;  . ; Set the "comment header"  . D  . . I CMTSTS=17 S HLOARR("COMMENT",LINECNT)="FORWARD SERVICE COMMENT",LINECNT=LINECNT+1 Q  . . I CMTSTS=19 S HLOARR("COMMENT",LINECNT)="CANCEL COMMENT",LINECNT=LINECNT+1 Q  . . I CMTSTS=20 S HLOARR("COMMENT",LINECNT)="ADD COMMENT",LINECNT=LINECNT+1 Q  . ;  . ; Loop through the comments and add to the HLOARR  . F CMTCNT=1:1:TOTLINES D  . . S HLOARR("COMMENT",LINECNT)=$G(^GMR(123,GMRIEN,40,CMTNODE,1,CMTCNT,0)),LINECNT=LINECNT+1  ;  S HLOARR("COMMENT",0)=LINECNT-1  ;  Q  ;** | | | | | | | | |

Table : ORMXUTIL Routine

| Routines | Activities | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Routine Name | ORMXUTIL | | | | | | | |
| Enhancement Category | New | | Modify | Delete | | No Change | | |
| RTM |  | | | | | | | |
| Related Options |  | | | | | | | |
| Related Routines | Routines “Called By” | | | | Routines “Called” | | | |
|  | None | | | | DIE  DIR  SDMXCORE  VALM  VALM1  VALM10  VALM4  XQORM1 | | | |
| Routines | Activities | | | | | | | |
| Data Dictionary (DD) References | ^ORD(100.98  ^ORD(101.43 | | | | | | | |
| Related Protocols | N/A | | | | | | | |
| Related Integration Control Registrations (ICRs) | N/A | | | | | | | |
| Data Passing | Input | Output Reference | | | Both | | Global Reference | Local |
| Input Attribute Name and Definition |  | | | | | | | |
| Output Attribute Name and Definition |  | | | | | | | |
| Current Logic | | | | | | | | |
| N/A - New routine | | | | | | | | |
| Modified Logic (Changes are in bold) | | | | | | | | |
| **ORMXUTIL ;MASS/BHC - ORDER UTILITIES;11/28/17  ;;3.0;Order Entry/Results Reporting;\*\*470\*\*;NOVEMBER 28,2017;Build 1  ;;Per VA directive 6402, this routine should not be modified.  ; SFEN ; Main entry point for ORMX Schedulable Action Utility  D EN^VALM("ORMX SCHEDULABLE ACTION")  Q SFHDR ; Header code for the ORMX Schedulable Action Utility  S VALMHDR(1)="ORMX Schedulable Action Utility"  Q  ; SFINIT ; Set initial variables and list array for the ORMX Schedulable Action Utility  ;  ; Variables assumed to be set from calling List Manager Application:  ; DISPGRP - Internal Code for Display Group (#100.98). If blank, will display all  N LINE,ORDABLE,ORDNAME,LINEVAR,SHRTNAME,ACTION,NAME,ORD0  S DISPGRP=$G(DISPGRP)  S SHRTNAME=$P($G(^ORD(100.98,DISPGRP,0)),"^",3)  W !,SHRTNAME  I SHRTNAME="" D  . S ORDABLE=0  . F S ORDABLE=$O(^ORD(101.43,ORDABLE)) Q:'+ORDABLE D  . . S LINE=LINE+1  . . S ORD0=$G(^ORD(101.43,ORDABLE,0)) Q:ORD0=""  . . S NAME=$P(ORD0,"^",1),ACTION=$P(ORD0,"^",9)  . . S LINEVAR=$$SETFLD^VALM1(LINE\_".",LINEVAR,"LINENO")  . . S LINEVAR=$$SETFLD^VALM1(ORDABLE,LINEVAR,"ORDERABLE")  . . S LINEVAR=$$SETFLD^VALM1(NAME,LINEVAR,"NAME")  . . S LINEVAR=$$SETFLD^VALM1(ACTION,LINEVAR,"ACTION")  . . D SET^VALM10(LINE,LINEVAR,ORDABLE)  E D  . F S ORDNAME=$O(^ORD(101.43,"S."\_SHRTNAME,ORDNAME)) Q:ORDNAME="" D  . . S LINE=LINE+1  . . S ORDABLE=$O(^ORD(101.43,"S."\_SHRTNAME,ORDNAME,""))  . . S ORD0=$G(^ORD(101.43,ORDABLE,0)) Q:ORD0=""  . . S NAME=$P(ORD0,"^",1),ACTION=$P(ORD0,"^",9)  . . S LINEVAR=$$SETFLD^VALM1(LINE\_".",LINEVAR,"LINENO")  . . S LINEVAR=$$SETFLD^VALM1(ORDABLE,LINEVAR,"ORDERABLE")  . . S LINEVAR=$$SETFLD^VALM1(NAME,LINEVAR,"NAME")  . . S LINEVAR=$$SETFLD^VALM1(ACTION,LINEVAR,"ACTION")  . . D SET^VALM10(LINE,LINEVAR,ORDABLE)  S VALMCNT=LINE  Q  ; SFHELP ; Help code for the ORMX Schedulable Action Utility  N X  S X="?" D DISP^XQORM1 W !!  Q  ; SFEXIT ; Exit code for the ORMX Schedulable Action Utility  K @VALMAR,DISPGRP  Q  ; SFEXPND ; Expand code for the ORMX Schedulable Action Utility  Q  ; SFRUN ; Run code for the ORMX Schedulable Action Utility  N ORDABLE,LINES,LINE,ITER,NODE,Y  ; Get Orderables to set action  K DIR  S DIR(0)="L^1:"\_VALMCNT  D ^DIR  I Y="^" S VALMBCK="R" Q  S LINES=$G(Y)  S NODE=""  F S NODE=$O(Y(NODE)) Q:NODE="" F ITER=1:1:$L(Y(NODE),",") D  . S LINE=$P(Y(NODE),",",ITER) I LINE="" Q  . S ORDABLE(LINE)=$O(@VALMAR@("IDX",LINE,""))  ;  ; Get Action  K DIR  S DIR(0)="SB^0:DO NOT SCHEDULE;1:SCHEDULE ONLY;2:SCHEDULE AND COMPLETE"  D ^DIR  S Y=$G(Y)  I Y="^" S VALMBCK="R" Q  ;  ; Set Action  I $$INSTRING^SDMXCORE(Y,"0,1,2",",") D SFSET(.ORDABLE,Y)  ;  S VALMBCK="R"  Q  ; SFSET(ORDABLE,SFVAL) ; Set 0;9 of ^ORD(101.43,ORDABLE to SFVAL  ;Update file entry  N ORDARR,LINE  F S LINE=$O(ORDABLE(LINE)) Q:'+LINE D  . S ORDARR(101.43,ORDABLE(LINE)\_",",11)=SFVAL  . D FILE^DIE("","ORDARR")  . ;repaint list manager lines  . D FLDTEXT^VALM10(LINE,"ACTION",SFVAL)  . D WRITE^VALM10(LINE)  . K ORDARR(101.43,ORDABLE(LINE)\_",",11)  Q  ; SFCHDGRP ; prompts user for a orderable item display group and  ; redraws the list area with only orderable items in that  ; display group.  N Y  S DIR(0)="P^100.98"  D ^DIR  S DISPGRP=$P(Y,"^",1)  D CLEAN^VALM10  D SFINIT  D RE^VALM4  Q** | | | | | | | | |

1. New regulatory requirements for contract v2 [↑](#footnote-ref-2)