Alert Watch and Response Engine (AWARE)

System Design Document



October 2014

Prepared by Harris Corporation

Revision History

| **Creation Date** | **Version No.** | **Description/Comments** | **Author(s)** | **Reviewer(s)** | **Review Type** | **Issue Date** |
| --- | --- | --- | --- | --- | --- | --- |
| 03/31/2014 | 0.01 | Initial | I. Valdes | S. Greenacre B. Tjahjo L. Glance M. Becker S. Wang B. Jan | Peer | 05/27/2014 |
| 07/15/2014 | 0.02 | Added new SQL Version table to section 6.10.2.3 and ordered the tables in alphabetical order. | S. Greenacre |  |  |  |
| 10/16/2014 | 0.03 | Updated table 8. | S. Greenacre |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

Table of Contents

[1 Introduction 1](#_Toc388968230)

[1.1 Purpose 1](#_Toc388968231)

[1.2 Identification 1](#_Toc388968232)

[1.3 Scope 1](#_Toc388968233)

[1.4 Relationship to Other Plans 2](#_Toc388968234)

[1.5 Methodology, Tools, and Techniques 2](#_Toc388968235)

[1.6 Policies, Directives, and Procedures 3](#_Toc388968236)

[1.7 Constraints 3](#_Toc388968237)

[1.8 Design Trade-Offs 3](#_Toc388968238)

[1.9 User Characteristics 3](#_Toc388968239)

[1.9.1 User Problem Statement 3](#_Toc388968240)

[1.9.2 User Objectives 3](#_Toc388968241)

[2 Background 4](#_Toc388968242)

[2.1 Overview of the System 4](#_Toc388968243)

[2.2 Overview of the Business Process 5](#_Toc388968244)

[2.3 Business Benefits 6](#_Toc388968245)

[2.4 Assumptions and Constraints 7](#_Toc388968246)

[2.4.1 Design Assumptions 7](#_Toc388968247)

[2.5 Overview of the Significant Requirements 7](#_Toc388968248)

[2.5.1 Functional Workload and Functional Performance Requirements 8](#_Toc388968249)

[2.5.2 Operational Requirements 8](#_Toc388968250)

[2.5.3 Overview of the Technical Requirements 9](#_Toc388968251)

[2.5.4 Overview of the Security or Privacy Requirements 9](#_Toc388968252)

[2.5.5 System Criticality and High Availability Requirements 9](#_Toc388968253)

[2.5.6 Special Device Requirements 9](#_Toc388968254)

[2.6 Legacy System Retirement 9](#_Toc388968255)

[3 Conceptual Design 10](#_Toc388968256)

[3.1 Conceptual Application Design 10](#_Toc388968257)

[3.1.1 Application Context 10](#_Toc388968258)

[3.1.2 High-Level Application Design 11](#_Toc388968259)

[3.1.3 Application Locations 12](#_Toc388968260)

[3.1.4 Application Users 13](#_Toc388968261)

[3.2 Conceptual Data Design 13](#_Toc388968262)

[3.2.1 Project Conceptual Data Model 13](#_Toc388968263)

[3.2.2 Database Information 13](#_Toc388968264)

[3.2.3 User Interface Data Mapping 14](#_Toc388968265)

[3.3 Conceptual Infrastructure Design 17](#_Toc388968266)

[3.3.1 System Criticality and High Availability 17](#_Toc388968267)

[3.3.2 Special Technology 17](#_Toc388968268)

[3.3.3 Conceptual Infrastructure Diagram 17](#_Toc388968269)

[3.3.4 Conceptual Production String Diagram 17](#_Toc388968270)

[4 System Architecture 18](#_Toc388968271)

[4.1 Hardware Architecture 18](#_Toc388968272)

[4.2 Software Architecture 18](#_Toc388968273)

[4.3 Communications Architecture 21](#_Toc388968274)

[5 Data Design 22](#_Toc388968275)

[5.1 Database Management System Files 22](#_Toc388968276)

[5.2 Non-Database Management System Files 22](#_Toc388968277)

[6 Detailed Design 22](#_Toc388968278)

[6.1 Hardware Detailed Design 22](#_Toc388968279)

[6.2 Software Detailed Design 23](#_Toc388968280)

[6.2.1 CPRS (1) and AWARE DLL (1a) Modules 23](#_Toc388968281)

[6.2.2 KB Editor 28](#_Toc388968282)

[6.2.3 Alert Cache Collector 30](#_Toc388968283)

[6.2.4 Alert Cache Viewer 32](#_Toc388968284)

[6.2.5 VistA to SQL Transporter 34](#_Toc388968285)

[6.2.6 QI Tool 36](#_Toc388968286)

[6.2.7 Local Data Structures 38](#_Toc388968287)

[6.3 Hardware Interfaces 64](#_Toc388968288)

[6.4 Software Interfaces 65](#_Toc388968289)

[6.5 Communications Interfaces 65](#_Toc388968290)

[6.6 Memory Constraints 65](#_Toc388968291)

[6.7 Special Operations 65](#_Toc388968292)

[6.8 User Characteristics 65](#_Toc388968293)

[6.9 Dependencies and Constraints 66](#_Toc388968294)

[6.10 Database Repository 66](#_Toc388968295)

[6.10.1 VistA VA FileMan 66](#_Toc388968296)

[6.10.2 MS SQL Server Database 70](#_Toc388968297)

[6.11 System Features 73](#_Toc388968298)

[6.11.1 CPRS/AWARE Design Element Tables 74](#_Toc388968299)

[6.11.2 Mail groups 96](#_Toc388968300)

[6.11.3 Security Keys 97](#_Toc388968301)

[6.11.4 Options 98](#_Toc388968302)

[6.11.5 Protocols 99](#_Toc388968303)

[6.11.6 Remote Procedure Call (RPC) 99](#_Toc388968304)

[6.11.7 Constants Defined Interface 101](#_Toc388968305)

[6.11.8 Variables Defined Interface 101](#_Toc388968306)

[6.11.9 Types Defined Interface 101](#_Toc388968307)

[6.11.10 CPRS/AWARE DLL Graphical User Interface (GUI) 101](#_Toc388968308)

[6.11.11 GUI Classes 102](#_Toc388968309)

[6.11.12 Current Form 103](#_Toc388968310)

[6.11.13 Modified Form 103](#_Toc388968311)

[6.11.14 Components on Form 103](#_Toc388968312)

[6.11.15 Events 103](#_Toc388968313)

[6.11.16 Methods 104](#_Toc388968314)

[6.11.17 Special References 104](#_Toc388968315)

[6.11.18 Class Events 104](#_Toc388968316)

[6.11.19 Class Methods 104](#_Toc388968317)

[6.11.20 Class Properties 104](#_Toc388968318)

[6.11.21 Uses Clause 104](#_Toc388968319)

[6.11.22 Forms 104](#_Toc388968320)

[6.11.23 Functions 104](#_Toc388968321)

[6.11.24 Dialog 105](#_Toc388968322)

[6.11.25 Help Frame 105](#_Toc388968323)

[6.11.26 Alert Cache Viewer Web Application Interface 106](#_Toc388968324)

[6.11.27 KB Editor Web Application Interface 108](#_Toc388968325)

[6.11.28 QI ReportingTool Web Application Interface 118](#_Toc388968326)

[6.11.29 VistA to SQL Transporter (Console Application) 118](#_Toc388968327)

[6.11.30 HL7 Application Parameter 118](#_Toc388968328)

[6.11.31 HL7 Logical Link 118](#_Toc388968329)

[6.11.32 COTS Interface 118](#_Toc388968330)

[6.12 Communications Detailed Design 119](#_Toc388968331)

[7 External Interface Design 121](#_Toc388968332)

[7.1 Interface Architecture 121](#_Toc388968333)

[8 Human Machine Interface 121](#_Toc388968334)

[8.1 Interface Design Rules 121](#_Toc388968335)

[8.2 Inputs 121](#_Toc388968336)

[8.3 Outputs 122](#_Toc388968337)

[8.4 Navigation Hierarchy 123](#_Toc388968338)

[8.4.1 CPRS/AWARE Screens 123](#_Toc388968339)

[8.4.2 Alert Cache Viewer Screens 126](#_Toc388968340)

[8.4.3 KB Editor Screens 127](#_Toc388968341)

[8.4.4 QI Tool Screens 127](#_Toc388968342)

[9 System Integrity Controls 127](#_Toc388968343)

[10 Appendix A 128](#_Toc388968344)

[10.1 Requirements Traceability Matrix 128](#_Toc388968345)

[10.2 See AWARE Requirements Traceability Matrix. 128](#_Toc388968346)

[10.3 Packaging and Installation 128](#_Toc388968347)

[10.4 Design Metrics 128](#_Toc388968348)

[10.5 Acronyms, Abbreviations, Terms, and Definitions 129](#_Toc388968349)

[10.6 Required Technical Documents 129](#_Toc388968350)

[11 Attachment A - Approval Signatures 130](#_Toc388968351)

List of Tables

[Table 1- Business Process Components 5](#_Toc388968352)

[Table 2 - Functional Requirements 7](#_Toc388968353)

[Table 3 - Technical Requirements 9](#_Toc388968354)

[Table 4 - AWARE Context Description 10](#_Toc388968355)

[Table 5 - Application Locations 12](#_Toc388968356)

[Table 6 - Database Inventory 14](#_Toc388968357)

[Table 7 - Reminder Dialog Screen Description 15](#_Toc388968358)

[Table 8 - Data Mapping for QI Tool 16](#_Toc388968359)

[Table 9 - AWARE System Description 19](#_Toc388968360)

[Table 10 - CPRS Objects in CPRS/AWARE DLL Process Design 26](#_Toc388968361)

[Table 11 - AWARE DLL Objects in CPRS/AWARE DLL Process Design 27](#_Toc388968362)

[Table 12 - KB Editor Objects in KB Editor Process Design 29](#_Toc388968363)

[Table 13 - Objects In Alert Cache Collector Design 31](#_Toc388968364)

[Table 14 - Objects In Alert Cache Viewer Process Design 33](#_Toc388968365)

[Table 15 - Objects In VistA to SQL Transporter Process Design 35](#_Toc388968366)

[Table 16 - Objects In QI Tool Process Design 37](#_Toc388968367)

[Table 17 –Get Alert Type RPC 74](#_Toc388968368)

[Table 18 - Alert Tracker, Renew Tracked alert, Reporting Routines 75](#_Toc388968369)

[Table 19 - AWARE Patient Closed-out COM Object RPCs 77](#_Toc388968370)

[Table 20 - Continuation AWARE Patient Closed-out COM Object RPCs 79](#_Toc388968371)

[Table 21 - Alert Cache Collector Procedure& entry points to other procedures 80](#_Toc388968372)

[Table 22 - Continued AWARE Alert Cache Collector Procedure 81](#_Toc388968373)

[Table 23 - Continuation Alert Cache Collector Procedure 82](#_Toc388968374)

[Table 24 - VistA to SQL Transport RPC 83](#_Toc388968375)

[Table 25 - Continued VistA to SQL Transport RPC 84](#_Toc388968376)

[Table 26 - Security Key Routines 85](#_Toc388968377)

[Table 27 - Collection of Alert Cache Viewer Procedures 86](#_Toc388968378)

[Table 28 - Get All Alert Categories Procedure 87](#_Toc388968379)

[Table 29 - Save Alert Category Procedure 87](#_Toc388968380)

[Table 30 - Get All Alert Types Procedure 88](#_Toc388968381)

[Table 31 - Get Alert Type - Order/Follow-up/Comments Dialog Text Procedure 88](#_Toc388968382)

[Table 32 - Get Alert Type Name Function 89](#_Toc388968383)

[Table 33 - Get Alert Type – Reminder Dialog Record Procedure 89](#_Toc388968384)

[Table 34 - Get All dialogs (Order/Follow-up/Comment) for an Alert type Procedure 90](#_Toc388968385)

[Table 35 - Save Alert Type – Reminder Dialog Procedure 90](#_Toc388968386)

[Table 36 - Save Order/Follow-up/Comments Dialog Procedure 91](#_Toc388968387)

[Table 37 - Deactivate Alert Type Procedure 91](#_Toc388968388)

[Table 38 - Update Initiation Date Procedure 92](#_Toc388968389)

[Table 39 - Delete Alert Type Procedure 92](#_Toc388968390)

[Table 40 - Delete Dialog Box text (Order/Follow-up/Comments) Procedure 93](#_Toc388968391)

[Table 41 - Alert Type Validator 93](#_Toc388968392)

[Table 42 - Continued Alert Type Validator 94](#_Toc388968393)

[Table 43 - Collection of procedures/functions used by KB Editor/Alert Cache Viewer 95](#_Toc388968394)

[Table 44 - Unique Record(s) 96](#_Toc388968395)

[Table 45 - Security Keys 97](#_Toc388968396)

[Table 46 - Options 98](#_Toc388968397)

[Table 47 - Types Defined Interface 101](#_Toc388968398)

[Table 48 - Delphi Units and Forms for AWARE DLL 101](#_Toc388968399)

[Table 49 - Changes in CPRS Delphi Units 101](#_Toc388968400)

[Table 50 - New GUI Classes in AWARE DLL 102](#_Toc388968401)

[Table 51 - Components on Form 103](#_Toc388968402)

[Table 52 - Events 103](#_Toc388968403)

[Table 53 - Methods 104](#_Toc388968404)

[Table 54 - Special References 104](#_Toc388968405)

[Table 55 - Class Events 104](#_Toc388968406)

[Table 56 - Class Methods 104](#_Toc388968407)

[Table 57 - Class Properties 104](#_Toc388968408)

[Table 58 - Save Access and Verify Codes 106](#_Toc388968409)

[Table 59 - Validate and Login to Alert Cache Viewer 106](#_Toc388968410)

[Table 60 - Populate Web Page with Alert Cache Contents 107](#_Toc388968411)

[Table 61 - Show Alert Cache Detail Information 108](#_Toc388968412)

[Table 62 - Save Access and Verify Codes 108](#_Toc388968413)

[Table 63 - Validate and Login to KB Editor 109](#_Toc388968414)

[Table 64 - Fill Web page with Alert Categories and Alert Types 109](#_Toc388968415)

[Table 65 - Populating Alert Category Form- Get Delete Options 110](#_Toc388968416)

[Table 66 - Populating Alert Category Form- Notification Type Options 110](#_Toc388968417)

[Table 67 - Saving Alert Category 111](#_Toc388968418)

[Table 68 - Disable Alert Category Fields 111](#_Toc388968419)

[Table 69 - Populate Alert Type Form 112](#_Toc388968420)

[Table 70 - Saving Alert Type 112](#_Toc388968421)

[Table 71 - Populate Order Dialog Form 113](#_Toc388968422)

[Table 72 - Populate Follow-up Dialog Form 113](#_Toc388968423)

[Table 73 - Populate Comments Dialog Form 114](#_Toc388968424)

[Table 74 - Saving Alert Type’s Order Dialog Record 115](#_Toc388968425)

[Table 75 - Saving Alert Type’s Follow-up Dialog Record 115](#_Toc388968426)

[Table 76 - Saving Alert Type’s Comment Dialog Record 116](#_Toc388968427)

[Table 77 - Saving Alert Type’s Dialog Record Method 117](#_Toc388968428)

[Table 78 - Deleting Alert Type’s Dialog Record Method 117](#_Toc388968429)

[Table 79 - VistA Login Web Service 118](#_Toc388968430)

[Table 80 – Server Hardware 120](#_Toc388968431)

[Table 81 – Server Software 121](#_Toc388968432)

List of Figures

[Figure 1 - AWARE System Overview 4](#_Toc388968433)

[Figure 2 - AWARE Business Process 5](#_Toc388968434)

[Figure 3 - Related Business Processes 6](#_Toc388968435)

[Figure 4 - AWARE Application Context 10](#_Toc388968436)

[Figure 5 - AWARE High Level Application Design 12](#_Toc388968437)

[Figure 6 - AWARE Conceptual Data Model 13](#_Toc388968438)

[Figure 7 - Follow-Up Action Dialog 14](#_Toc388968439)

[Figure 8 - Reminder Dialog 15](#_Toc388968440)

[Figure 9 - AWARE Conceptual Infrastructure 17](#_Toc388968441)

[Figure 10 – Hardware Architecture 18](#_Toc388968442)

[Figure 11 - AWARE Software Components 19](#_Toc388968443)

[Figure 12 – Communications Architecture 21](#_Toc388968444)

[Figure 13 – Application Database Schema from the MS SQL Server Management Studio 22](#_Toc388968445)

[Figure 14 - CPRS/AWARE DLL Process Flow 24](#_Toc388968446)

[Figure 15 - CPRS/AWARE DLL Process Design 25](#_Toc388968447)

[Figure 16 - KB Editor Process Design 28](#_Toc388968448)

[Figure 17 - Alert Cache Collector Process Design 30](#_Toc388968449)

[Figure 18 - Alert Cache Viewer Process Design 32](#_Toc388968450)

[Figure 19 - VistA to SQL Transporter Process Design 34](#_Toc388968451)

[Figure 20 - QI Tool Process Design 36](#_Toc388968452)

[Figure 21 - “Prompt Screen” User Interface 38](#_Toc388968453)

[Figure 22 - Reminder Dialog 39](#_Toc388968454)

[Figure 23 - User Setup 40](#_Toc388968455)

[Figure 24 - User Setup Continued (above) 40](#_Toc388968456)

[Figure 25 - List of Alert Categories 41](#_Toc388968457)

[Figure 26 - Alert Category Form 42](#_Toc388968458)

[Figure 27 - List of Alert Types 42](#_Toc388968459)

[Figure 28 –Alert Type Form 43](#_Toc388968460)

[Figure 297 –List of Order Dialogs 43](#_Toc388968461)

[Figure 30 –Order Dialog Text Form 44](#_Toc388968462)

[Figure 31 –List of Follow-up Dialogs 45](#_Toc388968463)

[Figure 32 - Follow-up Dialog Text Form 46](#_Toc388968464)

[Figure 33 - List of Comments Dialogs 46](#_Toc388968465)

[Figure 34 - Comment Dialog Text Form 47](#_Toc388968466)

[Figure 35 - Order Dialog Text Form 48](#_Toc388968467)

[Figure 36 - List of Cached Critical Alerts 49](#_Toc388968468)

[Figure 37 - Detail Cached Critical Alert Information (GENERAL) 50](#_Toc388968469)

[Figure 38 - Detail Cached Critical Alert (FAT/STATUS) 50](#_Toc388968470)

[Figure 39 - Detail Cached Critical Alert (RESULT) 51](#_Toc388968471)

[Figure 40 - Detail Cached Critical Alert (FOLLOW-UP ACTIONS) 51](#_Toc388968472)

[Figure 41 – QI Tool Logon Screen 52](#_Toc388968473)

[Figure 42 – QI Tool Logon Screen 2 53](#_Toc388968474)

[Figure 43 - Validated as Super User 54](#_Toc388968475)

[Figure 44 - Validated as Local Application User 55](#_Toc388968476)

[Figure 45 - Validated as VistA User 55](#_Toc388968477)

[Figure 46 - Dynamic Reports Parameter panel 56](#_Toc388968478)

[Figure 47 - Selected Reports Generated 57](#_Toc388968479)

[Figure 48 - QI Tool Administration Tools 58](#_Toc388968480)

[Figure 49 - QI Group Panel – Add Mode 58](#_Toc388968481)

[Figure 50 - Available Reports Drop Down List 59](#_Toc388968482)

[Figure 51 - Group Members Drop Down List 59](#_Toc388968483)

[Figure 52 - Delete QI Group Warning 60](#_Toc388968484)

[Figure 53 - VistA Groups Panel – Add Mode 60](#_Toc388968485)

[Figure 54 - VistA Groups – AWARE Group Drop Down List 61](#_Toc388968486)

[Figure 55 - VistA Group Deletion Warning 62](#_Toc388968487)

[Figure 56 - Users Panel – Add Mode 62](#_Toc388968488)

[Figure 57 - Delete User Warning Prompt 63](#_Toc388968489)

[Figure 58 - Reports Panel 64](#_Toc388968490)

[Figure 59 - MS SQL Application Data Tables 71](#_Toc388968491)

[Figure 60 - MS SQL Application Data Tables (Continued) 72](#_Toc388968492)

[Figure 61 - Prompting Screen 103](#_Toc388968493)

[Figure 62 - Help Screen 105](#_Toc388968494)

[Figure 63 - Communication Diagram 119](#_Toc388968495)

[Figure 64 - CPRS Prompting Screen 122](#_Toc388968496)

[Figure 65 - CPRS Patient Selection Screen 122](#_Toc388968497)

[Figure 66 - Select New Patient 123](#_Toc388968498)

[Figure 67 - Patient Selection 124](#_Toc388968499)

[Figure 68 - FAT Prompt 124](#_Toc388968500)

[Figure 69 - Clinic Selection (above) 125](#_Toc388968501)

[Figure 70 - Clinic Selection (continued) 125](#_Toc388968502)

[Figure 71 - Progress Note Title Selection 126](#_Toc388968503)

[Figure 72 - Reminder Dialog Present 126](#_Toc388968504)

# Introduction

The Computerized Patient Record System (CPRS) View Alerts package delivers notification of non-life threatening critical test results; however, it does not track whether providers take appropriate follow-up actions in response to the alerts. Currently, the only way to track follow-up actions on critical alerts is through manual review of individual patient records. The innovation Alert, Watch and Response Engine (AWARE) will track and monitor follow-up actions, and will identify certain critical lab and imaging test result alerts that lack timely follow-up.

## Purpose

The purpose of this document is to describe how AWARE will be integrated with the current version of CPRS and the system design elements that are required therein. CPRS integration functions as the core part of the AWARE system for specific patient alerts involving patient safety. It is the basis for re-direction prompting of a patient’s provider(s) for processing timely follow-up actions on specific alerts. When these follow-up actions are completed, they help serve Veterans Affairs (VA)-desired goals for providing excellent clinical care and matching patient safety measures for serving Veterans.

This document also describes how the Knowledge-Based (KB) Editor is utilized to define AWARE (critical) alerts and follow-up actions, and how providers and safety officer could use the Alert Cache Viewer and Quality Improvement (QI) Reporting Tool to view AWARE alerts.

## Identification

This document applies to CPRS V29 and AWARE v1.0.

## Scope

The scope of this document is describing the AWARE integration with CPRS and other components of AWARE. The breadth of the scope is as follows:

Inclusions:

* Dynamic-link library (DLL)’s for normal, abnormal, and incomplete test result critical alert detection.
* CPRS Modifications.
* VistA server files and data structure.
* Microsoft Structured Query Language (MS SQL) Server tables and data structure.
* Alert Cache Collector.
* VistA to SQL Transporter.
* Knowledge-Based (KB) Editor.
* QI Reporting tool.
* Alert Cache Viewer.

Exclusions:

* Non-VistA systems.

## Relationship to Other Plans

The following related documents apply:

* Requirements Specification Document
* CPRS Integration Specification
* User Interface Document
* Version Description Document
* KIDS and CSP Installation Guide
* Server Installation Guide

## Methodology, Tools, and Techniques

Methodologies:

* Agile development user stories
* Wireframe mock-ups
* Requirements engineering
* Stakeholder identification
* Stakeholder interviews
* Requirements Lists

Tools:

* Visio
* Microsoft Office
* VistA Development Library (VDL)
* Delphi 2007
* Microsoft Visual Studio 2010
* SQL Server Reporting Services (SSRS)

Techniques:

* Data flow modeling
* Goal Modeling
* Joint Requirements Analysis Sessions

## Policies, Directives, and Procedures

This document must go through a series of informal and formal reviews, and all accepted feedback will be incorporated into the document prior to release. This document will comply with all applicable VA policies, directives, and procedures.

## Constraints

The major constraints as it relates to CPRS and VistA are the processes, workflow, and software itself, surrounding the VA health care system embodied in the VistA system. There is a considerable amount of existing software, workflow, and processes that, by definition, constrain this effort. Examples are the ability to make changes to CPRS constrained by its release schedule, centralized planning, and rollout nature. Therefore, changes to the CPRS core should be as limited as possible. Other examples are the inability to use alert color-coding because of 508 compliance, single typeface, and manual nature of reminder dialog box building, lack of order and medication sub-classification, inability to specify a Text Integration Utility (TIU) note title programmatically further constrain this effort.

## Design Trade-Offs

This effort has to take into consideration all of the existing VA information systems, procedures and their limitations which have a potential to affect this project. While there is potential to make major modifications to the current VistA and CPRS applications to meet business requirements there is a significant cost to budget and schedule. Therefore, the trade-off documented within this System Design Document is to utilize a Dynamic Link Library (DLL), which allows for modifications to the current systems with little effect on the current functionality

## User Characteristics

This solution will be utilized by various users. Users authenticated on the VistA system, which may consist of clinicians, clinical staff, and support staff, will have access to the solution. Additionally, users required to support the applications are as follows: Clinical Application Coordinators (CAC), Automated Data Processing Application Coordinators (ADPAC), Information Resource Management (IRM) staff.

### User Problem Statement

At present, there is little to no special notification of normal, abnormal, and incomplete test results such as those indicative of a cancer. Critical alerts occur in the system, but little to no tracking, follow up information, special notification or decision support occurs other than what is present in the base system.

### User Objectives

The AWARE system shall:

* Provide an AWARE prompt to notify a user that abnormal lab results exist that has not been followed-up.
* Decision-support workflow that guides the user towards appropriate follow-up.
* Provide the reporting capabilities needed to track and monitor abnormal test results being tracked by the aware system.
* Provide tool for Clinical Application Coordinator (CAC) to define critical alerts and follow-ups interface through Reminder Dialogs.

# Background

## Overview of the System

Currently, there is no effective way to monitor or track critical abnormal test results in CPRS, which can potentially cause unnecessary delay in, and lack of appropriate follow-up, care for patients. In addition, there is no system in place within CPRS that has the capacity to identify, monitor, and track milestones in the diagnostic work-up and treatment of critical abnormal results. Paired with national clinical reminders and the proposed tracking of abnormal and incomplete test results capability, the functionality AWARE provides will improve detection, diagnosis, treatment, and management of patient conditions suspicious for malignancy in the Veterans Health Administration (VHA).

The figure below provides an overview of the AWARE system.

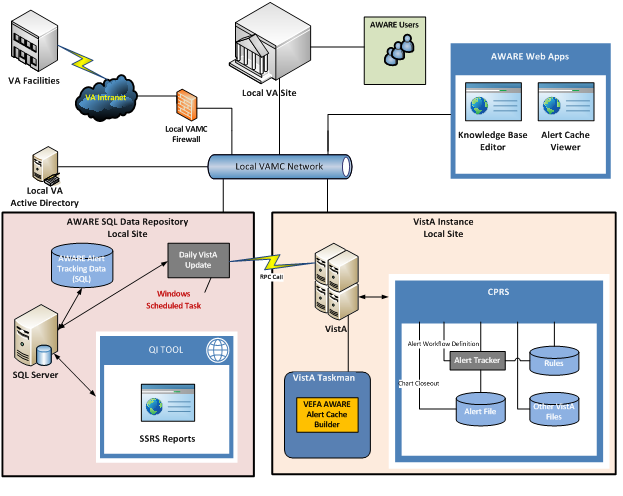


Figure - AWARE System Overview

## Overview of the Business Process

This section provides an overview of the business processes that AWARE will support. As shown in the following diagrams, the logical business flow is easily traceable because each modular component is sequentially identified. The business flow is further clarified in refined granularity in the table, where each business processes is classified, assigned an owner, and given a detailed description.



Figure - AWARE Business Process

Table - Business Process Components

| **Business Process ID** | **Business Process Name** | **Type** | **Owner** | **Description** |
| --- | --- | --- | --- | --- |
| 1 | CPRS | Existing | Clinicians and Quality Management personnel | CPRS is the primary GUI for VistA. Patient closeout chart, Alert follow-up prompt are called from VistA and displayed in CPRS (as in 1a) |
| 1a | Patient closeout prompt, View notification alert, reminders | New | CPRS users | Alerting users that test results exist for the patient test results and prompts users to take action on making a follow-up/treatment recommendation.  If the provider takes an action on the abnormal result while working in the chart then this prompt will be suppressed on chart close out and it will NOT be displayed to the provider. |
| 2 | Reminder Dialog Data Store Tracked Abnormal Alert type and Alert Categories | New | CPRS users and Quality Management personnel | This data store contains tracked normal, abnormal, and incomplete alerts types and alert categories as well as corresponding reminder dialogs and associated data to be used for AWARE. |
| 4 | Tracking Collector | New | CPRS users and Quality Management personnel and population demographers | The Tracking Collector will query alert follow-up information tracked from business process 1a Patient closeout chart, Alert follow-up prompt in addition to generalizing to other conditions such as occult blood, normal, abnormal, and incomplete PSA and other alert types. |
| 5 | Reminders, and associated information Data Store | Enhanced | CPRS users and Quality Management personnel and population demographers | Reminders, Alerts, Health Factors, procedures, and orders. |
| 6 | Reports | Existing/Enhanced | CPRS users and Quality Management personnel and population demographers | Tracked Alert follow-up performance statistics.  Also predefined and ad hoc reports will be generated at users request to display information collected and stored in the system. |

A related business flow diagram for the AWARE project is shown in the following figure for reference.

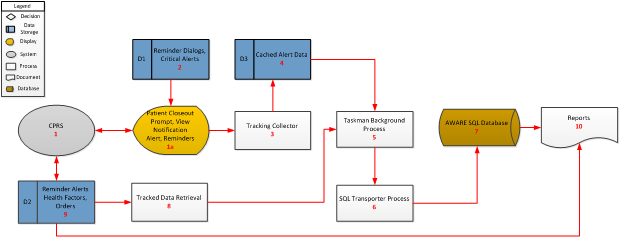


Figure - Related Business Processes

## Business Benefits

The project will furnish providers and care coordination staff with the ability to collaboratively increase the quality and timeliness of care through availability of all diagnostic test results with specific emphasis on conditions suspicious for malignancy and population management tools to enhance patient-centered care coordination.

## Assumptions and Constraints

This section describes the assumptions and constraints that impact the AWARE design.

### Design Assumptions

* An entry point can be made to exist within CPRS for on-chart closeout calls to a .dll file.
* Integration will occur with CPRS v29.
* Microsoft SQL Server 2012 Standard Edition to store AWARE Alert historical data.
* Microsoft Internet Information Services (IIS 7.x) to host AWARE web applications.
* Microsoft Windows 2008 R2 Service Pack 1 (Application, Database, IIS Web Server)
* Microsoft .NET Framework 4.x applied on (Application, Database, IIS Web Server)
* Microsoft SQL Server Reporting Services (SSRS) installed on the same instance as SQL Server.
* Microsoft Visual Studio 2010
* Microsoft Report Viewer 2010 Redistributable Package applied to IIS Web Server.
* VistA access from Web Server.
* Intersystem’s Cache Server Page (CSP) Gateway installed on the same instance as IIS.
* Limitations of the VistA system itself.
* The ability to deploy a new version of CPRS and a new .dll.

## Overview of the Significant Requirements

The figure below provides an overview of major functional requirements for the system from the Requirements Specification Document (RSD). The listed requirements drive the design detailed throughout the document. This table embodies the major functions to be performed and the major requirements that drive the design. The goal is not to include the full set of requirements. The impact that identified requirements have on the design are detailed with a synopsis. The CPRS/AWARE DLL integration is the core part of overall AWARE system, which includes additional enhancements.

Table - Functional Requirements

| **RSD ID** | **Specific Requirement / Synopsis** | **Requirement Overview** |
| --- | --- | --- |
| 2.6.1 | QI Tool | Convert text-based knowledge editor to web-based GUI, the “QI tool”. |
| 2.6.2 | Use SQL for critical alert tracking. | Use SQL for the purpose of aggregating and reporting critical alert data. |
| 2.6.3 | AWARE/CPRS Integration | CPRS will be extended with “Alert Watch and Response Engine” to inform the user of unacknowledged, un-followed up critical alerts and provide workflow guidance. |

|  | **AWARE Components to satisfy Functional Requirements** | **Discussion** |
| --- | --- | --- |
|  | CPRS AWARE | Standard CPRS is extended to be made aware of critical alerts. |
|  | CPRS Patient Closed-out COM (DLL) | To minimize code changes in the standard CPRS, AWARE call VistA Delphi COM interface during patient close-out event. |
|  | Alert Cache Collector | This VistA TaskMan process stores critical alerts into VistA Alert Cache file. |
|  | VistA to SQL Transporter | This Windows C# application transports VistA Alert Cache to MS SQL Server table. |
|  | KB Editor | This Content Security Policy (CSP) Web application provides tool for CAC to define critical alerts and follow-up actions. |
|  | Alert Cache Viewer | This CSP Web application provides provider and patient safety officer/admin to view critical alert cache. |
|  | QI Tool | This asp.net application is leveraging SQL Server Reporting Service to provide provider and patient safety officer/admin to view historical critical alerts. |

### Functional Workload and Functional Performance Requirements

The CPRS and VistA enhancements and changes required by the functional requirements outlined above rely on the pre-existing compliance of these applications with VA’s Information Technology (IT) Infrastructure Standards. No separate compliance ratification is expected to be necessary. The modifications will not impact the performance of the system.

Like other TaskMan processes, the impact of the performance caused by Alert Cache Collector will depend on the frequency and when the process is executed.

The VistA to SQL Transporter will be launched once per day, or according to another acceptable schedule using Windows Scheduler. It should have minimal impact on the performance if it is scheduled during off-hours.

KB Editor and Alert Cache Viewer are using CSP technology. The performance impact caused by those CSP applications will be minimal compared to Remote Procedure Call (RPC) driven CPRS application. The QI Tool is solely interacting with the SQL Server so there will be no impact on the VistA Server.

### Operational Requirements

The CPRS and VistA enhancements and changes required by the requirements outlined above in the Table rely on the pre-existing compliance of these applications with VA’s IT Infrastructure Standards. No separate compliance ratification is expected to be necessary. The proposed AWARE .DLL will only exist in the context of CPRS.

With the exception of the QI Tool, all AWARE components rely on VistA instance running on Intersystem’s Cache Server.

All AWARE web applications such as the KB Editor, Alert Cache Viewer, and QI tool rely on an IIS web server. There is an IIS CSP Gateway module running on the web server to route request to the VistA instance.

AWARE is also using an MS SQL Server to store Historical alert data. The VistA to SQL Transporter transports the VistA Alert Cache to a SQL Server table and the QI Tool gives users a reporting tool to view the historical alert data.

### Overview of the Technical Requirements

The enhancements and changes required by the technical requirements outlined below rely on the pre-existing compliance of these applications with VA’s Technical Reference Model (One TRM). No separate compliance ratification is expected to be necessary. The table below provides the major technical requirements that drive the conceptual design. The requirement overview summarizes the collective requirements of the RSD IDs provided. The CPRS/AWARE DLL integration is the core part of overall AWARE system, which includes additional enhancements.

Table - Technical Requirements

|  |  |
| --- | --- |
| **RSD ID** | **Requirement Overview** |
| 2.6.3 | AWARE/CPRS Integration |
|  | Alert Cache Collector |
| 2.6.2.1.2 | VistA to SQL Transporter |
| 2.6.1 | KB Editor |
| 2.6.2.2.2. | Alert Cache Viewer |
| 2.6.2.2.1 | QI Tool |

### Overview of the Security or Privacy Requirements

The CPRS and VistA enhancements and changes required by the requirements outlined above rely on the pre-existing compliance of these applications with the security and privacy requirements and their resulting Authority to Operate (ATOs). As such, no separate Certification and Accreditation Process and ATO are expected to be necessary.

Web applications such as KB Editor, Alert Cache Viewer, and QI Tool rely on similar VistA authentication mechanism. VistA security keys are utilized to give user access to KB Editor, Alert Cache Viewer, and QI Tool. These applications shall also comply with all privacy and security features as mandated by VA, Federal, State, and Local regulations.

### System Criticality and High Availability Requirements

The CPRS and VistA enhancements and changes required by the requirements outlined above rely on the pre-existing system criticality and availability requirements of these applications.

Alert Cache Collector, KB Editor, and Alert Cache Viewer have similar availability requirements as CPRS. VistA to SQL Transporter and QI Tool require both VistA instance and SQL Server to be available.

### Special Device Requirements

The current Performance Work Statement (PWS) requires a Windows with MS SQL server at each site.

## Legacy System Retirement

Not applicable. There are no legacy systems to be retired.

# Conceptual Design

## Conceptual Application Design

### Application Context

This is a broad overview figure of AWARE.

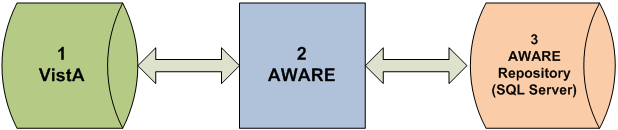


Figure - AWARE Application Context

The table below in this section is used to describe the information in the Application Context Diagram. Note that because the system for which this design applies is represented by a single object (in the center of the diagram, here as CPRS as ID 1 and AWARE DLL as ID 1a) it is not referred to in the table.

Table - AWARE Context Description

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Objects** | | | |
| **ID** | **Name** | **Description** | **Interface Name(s)** | **Interface System** |
| 3 | SQL Database table solution for alert Follow-up Action Tracking (FAT) data | Storage and Retrievals of tracked alert follow-up activities including whether follow-ups are made on a timely basis. This is by facility, clinic, provider, etc. over a period of time to hopefully provide incentives for improving outcomes with improvements in subsequent detection, diagnosis, treatment and management of cancer. | Vista Extraction Input Transmitter into  SQL Database table solution    SQL Database table Output Receiver | Alert Tracking Collector  Alert Tracking Data Retriever |

|  | **Interfaces External to Vista/CPRS** | | | | |
| --- | --- | --- | --- | --- | --- |
| **ID** | **Interface Name** | **Related Object** | **Input Messages** | **Output Messages** | **External Party** |
| 2 | VistA/CPRS data Input Transmitter into SQL database tables | From Extracted, transfer data into SQL database table solution  Using FTP transfers to build/update data directly into SQL server database. | Single Business Objects:  Alert Monitoring activity data from VISTA/CPRS including data if follow-up actions were taken | N/A | N/A |
| 2 | SQL database tables Output Receiver | Retrieve data from SQL database table for reports for clinical and administrative staff  Use SQL Server Reporting Services as web reports | N/A | Displays of historical Alert Monitoring activity data from VISTA/CPRS including data including statistical data on how well follow-up actions have been taken. | N/A |

|  | **Interfaces Internal to VISTA/CPRS** | | | | |
| --- | --- | --- | --- | --- | --- |
| **ID** | **Interface Name** | **Related Object** | **Input Messages** | **Output Messages** | **Other CBP Party** |
| 1 | Alert Tracker Collector of follow up action activity | Combined multiple data collections and subsequent storage of tracked alert follow-up action data along with tracked alert status and demographic data | Combined Business Objects to include subsets of the following:  Follow up action tracking activities from VISTA ALERT AND ALERT TRACKING files whose data is derived from actions taken from CPRS | Combined Business Objects to include subsets of the following:  Cache Data built/updated into ALERT tracking audit or log file. | N/A |
| 2 | User Tracked Alert Data Retriever | Combined multiple data retrievals via authorized user access to relevant alert tracking data by clinical or administrative staff for a facility such as clinician or patient safety officers to determine if follow-up actions had been taken and whether these follow-up actions were timely in | Combined Business Objects to include subsets of the following:  Collected Follow-up action data | Displays of relevant user alert tracking data for a facility for review/corrective action purposes | N/A |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Externally Shared Data Stores** | | | |
| **ID** | **Name** | **Data Stored** | **Owner** | **Access** |
| 3 | SQL Server database table | Historical Tracked alert follow-up action taken data | VHA/Facility ownership of data | Create, Retrieve, Update, Delete (CRUD) operations.  Transfer of transferred data with Create, Insert, Update, and Read, and operations. Inserts (Updates) automatically through storage of collected file data transferred on a periodic basis directly into SQL Database tables.  Dashboard web displays available to VA Executive Administrative and Clinical staff. |

### High-Level Application Design

The figure below provides a high-level Application Design of AWARE architecture. This conceptual architecture identifies modules within AWARE. The KB Editor application defines critical alerts by identifying notification types and reminder dialogs for follow-up actions. CPRS AWARE uses the KB Editor rules to identify critical alerts. It performs follow-up actions during patient chart closed-out events. A TaskMan process collects critical alerts and stores them into the VistA Alert Cache file. The Alert Cache Viewer allows providers and patient safety officers to view his/her critical alerts. The VistA to SQL Transporter (a Windows Scheduled task) transmits the VistA Alert Cache to SQL Server. The QI Tool allows provides providers and/or patient safety officers to use a SSRS reporting tool to view critical alerts stored in the SQL Server database.

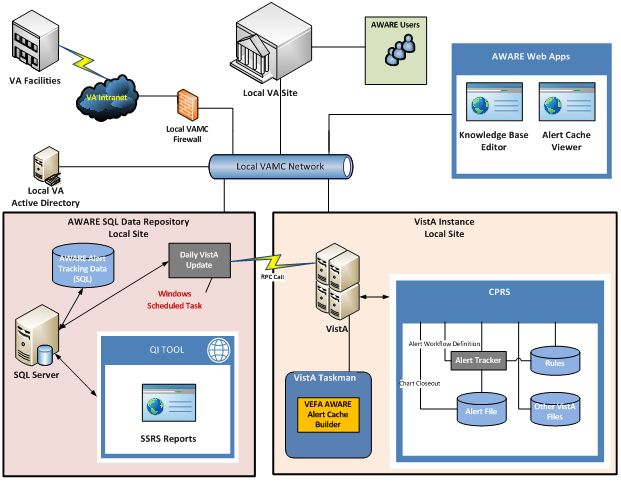


Figure - AWARE High Level Application Design

### Application Locations

The CPRS/AWARE DLL integration is currently made from client based workstations connected to VA facility VISTA systems. Web applications (KB Editor and QI Tool/Alert Cache Viewer) can be launched using their URLs.

The location of the application components is described in the table below:

Table - Application Locations

| Application Component | Description | Location at Which Component is Run |
| --- | --- | --- |
| AWARE/CPRS DLL (vefaalerttracking.dll) | CPRS Patient Close-Out COM Object (DLL) | User Workstation |
| KB Editor / Alert Cache Viewer | Cache Server Pages | Application/Database server IIS |
| Alert Cache Collector | VistA TaskMan Process | Local VistA Instance |
| SQL Transporter | Windows C# application calling VistA RPC to update SQL Server Tables | Database Server |
| Databases | SQL Server Databases | Database server |
| QI Manager | Asp.net web application managing user logins and displaying SSRS reports | Application/Database server IIS |
| QI Reports | SQL Server Reporting Service Reports | Database server |

### Application Users

The main users of the CPRS/AWARE DLL integration are clinicians. Reporting of tracked alert follow-up activities are of interest to clinicians as well administrative staff (patient safety officers, health system specialist, and executive staff).

## Conceptual Data Design

### Project Conceptual Data Model

The project conceptual data model is presented below. It is intended to be a high-level representation of the data entities and their relationships.

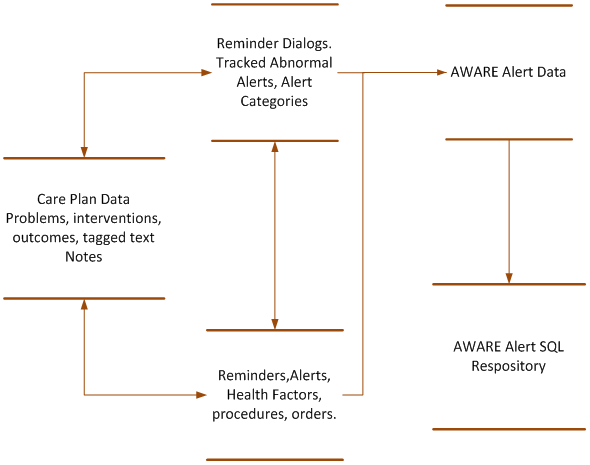


Figure - AWARE Conceptual Data Model

### Database Information

The table below identifies all databases that will be created, replaced, interfaced with, or whose structure will be modified (i.e., add or delete tables or add or delete columns to a table) as part of the AWARE effort. The VistA database will have an additional file or file added to its namespace for storing of tracked normal, abnormal and incomplete results, associated order sets and actions for corresponding reminder dialog boxes and other needed information to support AWARE. Modifications are expected to be largely limited to addition to the existing database.

Table - Database Inventory

|  |  |  |  |
| --- | --- | --- | --- |
| **Database Name** | **Description** | **Type** | **Steward** |
| MUMPS/VistA | MUMPS (Massachusetts General Hospital Utility Multi-Programming System) VistA system. Addition of tracking file via FileMan. | Modify | Department of Veterans Affairs |
| Microsoft SQL Server 2012 Standard Edition | SQL Server to store Critical Alerts Historical Data and to provide data to QI Reporting Tool. | New | Department of Veterans Affairs |

### User Interface Data Mapping

#### Reminder Dialog User Interface Data Mapping

VA Reminders are the tools for providers to enter for alert follow-up actions. These include orders, consults, text orders and comments/observations, which are mapped to equivalent VA FileMan files.

##### Reminder Dialog Screen

The AWARE DLL Follow-up Action tracking module provides a redirected prompt screen for a provider to consider follow-up action for their patient upon the occurrence of a tracked alert while the patient is being seen in CPRS. An example of this user interface is shown as below:

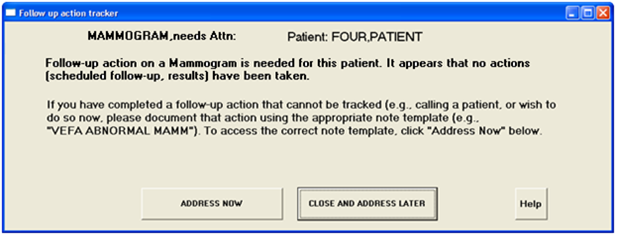


Figure - Follow-Up Action Dialog

Upon clicking the “ADDRESS NOW” button, the user will be re-redirected automatically to a specific Reminder dialog associated with that type of tracked alert. An example of what occurs when the user presses “ADDRESS NOW” is shown as below.

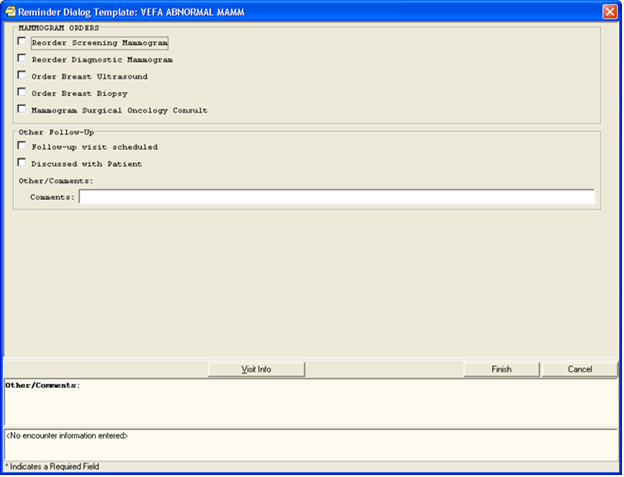


Figure - Reminder Dialog

Table - Reminder Dialog Screen Description

| **Graphical User Interface (GUI) Field** | **Table (Database Table that field connects to)** | **Field (Field in Table that the GUI field connects to)** | **Comments** |
| --- | --- | --- | --- |
| Reminder Dialog Element checkboxes. | Equivalent Follow-up Action File Man files such  As orders, consults text orders, comments, and observations in the progress note file. | Corresponding Reminder dialog element |  |

#### QI Tool User Interface Data Mapping

The table below shows a one to one correspondence from elements of the VistA database to elements of the SQL server.

Table - Data Mapping for QI Tool

| **Data Element in VistA** | **Data Element in SQL Server Database (Table. Column)** | **Data Element in QI Reports** |
| --- | --- | --- |
| 19008.2,.01/ALERTID | Alerts$.ALERTID | Alert ID |
| 19008.2,1/DATETIME | Alerts$.DATETIME1 | Alert Date Time |
| 19008.2,2/FACILITY NAME | Alerts$.FACILITYNAME | Facility |
| 19008.2,3/SERVICE | Alerts$.SERVICE1 | Service |
| 19008.2,4/ORDERING PROVIDER | Alerts$.ORDERINGPROVIDER | Ordering Provider |
| 19008.2,5/ALERT RECIPIENTS | Alerts$.ALERTRECIPIENTS | N/A |
| 19008.2,6/PATIENT | N/A | The Patient Name field is not mapped to SQL due to Privacy Rules. |
| 19008.2,7/ALERT CATEGORY | Alerts$.ALERTCATEGORY | Alert Category |
| 19008.2,8/ALERT TYPE | Alerts$.ALERTTYPE | Alert Type |
| 19008.2,23/ALERTVALUE | Alerts$.VALUE1 | Alert Value |
| 19008.2,9/UNACKSTATUS | Alerts$.UNACKSTATUS | Unack Status |
| 19008.2,10/ACKRENEWDATE | Alerts$.ACKRENEWDATE | Ack Renew Date |
| 19008.2,11/DELETEDATE | Alerts$.DELETEDATE | Ack Date |
| 19008.2,12/FAT STATUS | Alerts$.FATSTATUS | FAT Status |
| 19008.2,13/FAT PROVIDER | Alerts$.FATPROVIDER | FAT Provider |
| 19008.2,14/FOLLOW-UP PROVIDER ID | Alerts$.FOLLOW\_UPPROVIDERID | N/A |
| 19008.2,16/CLINIC | Alerts$.CLINIC | Clinic |
| 19008.2,17/PATIENTID | Alerts$.PATIENTID | Patient ID |
| 19008.2,18/ALERT RESULTOR | Alerts$.ALERTRESULTOR | N/A |
| 19008.2,19/RESULTOR PERSON CLASS | Alerts$.RESULTORPERSONCLASS | N/A |
| 19008.2,20/ALERT TYPE ORIG STATION | Alerts$.ALERTTYPEORIGSTATION | N/A |
| 19008.2,21/FOLLOWUP>7D | Alerts$.FOLLOWUPGT7D | Follow-up > 7 Days |
| 19008.2,22/ACK>7D | Alerts$.ACKGT7D | Ack > 7 Days |
| 19008.215,.01/FOLLOWUP ACTIONS | Followups$.FOLLOWUP | Follow-up |
| 19008.215,1/DATE/TIME | Followups$.FOLLOWUPDATETIME | Follow-up Date Time |
| N/A | Alerts$.STATION\_DATETIME\_ALERTID  This field is calculated during table loading. | N/A |
| 19008.2,25 ORIG ALERT TYPE | Alerts$.SPARE | N/A |
| N/A | Alerts$.FOLLOWUPLT7D  This field is calculated during table loading. | N/A |

#### Application Performance

Application performance is in large part governed by the performance of the overall VistA system. Report generation performance is also governed by the complexity of the report. Reporting is not part of the CPRS/AWARE DLL integration.

#### Unmapped Data Element

There is no unmapped data element.

## Conceptual Infrastructure Design

### System Criticality and High Availability

Microsoft SQLServer for long-term reporting purposes as identified in the table is not part of the CPRS/AWARE DLL integration, but is part of the enhancements for the overall AWARE system.

### Special Technology

The KB Editor and Alert Cache Viewer are using Intersystem’s Cache Server Page (CSP) while the QI Tool is utilizing ASP.NET and Microsoft SQL Server Reporting Service (SSRS).

The CPRS/AWARE DLL integration is currently from client based workstations connected to VA facility VISTA systems. The VA facility VISTA systems may be located remotely from the facilities such as VA IT Regional centers. The AWARE SQL Server and Web Server will be in the same location as the VistA system. A VistA to SQL transporter will retrieve data from VistA by RPC and store it in SQL Server database.

### Conceptual Infrastructure Diagram

The figure below shows the location of environments and external interfaces.

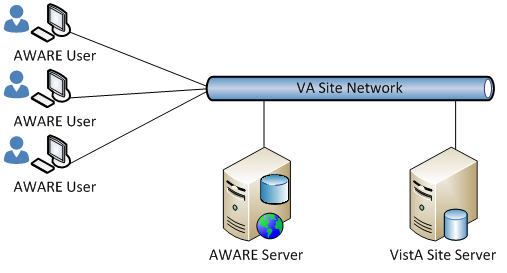


Figure - AWARE Conceptual Infrastructure

### Conceptual Production String Diagram

See the AWARE Conceptual Infrastructure above.

# System Architecture

## Hardware Architecture

The diagram below shows a VistA instance and an AWARE server which handles reporting by MS SQL Server.

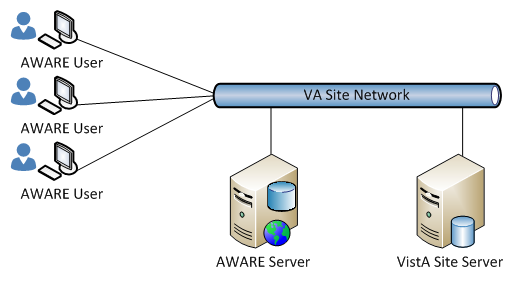


Figure – Hardware Architecture

## Software Architecture

Software to develop and maintain the CPRS/AWARE DLL is written in DELPHI. MUMPS is used to develop and maintain routines on the VISTA system with which the CPRS program communicates. The Alert Cache Collector is a TaskMan process. A VistA to SQL Transporter program is written in .NET C# and connects to both a VistA instance and SQL Server. The web applications KB Editor and Alert Cache Viewer are written in both JavaScript and CSP while the QI Tool is written in .NET C# and utilizes the SSRS Report Viewer. The following figure and table describe the software components that make up the AWARE system.

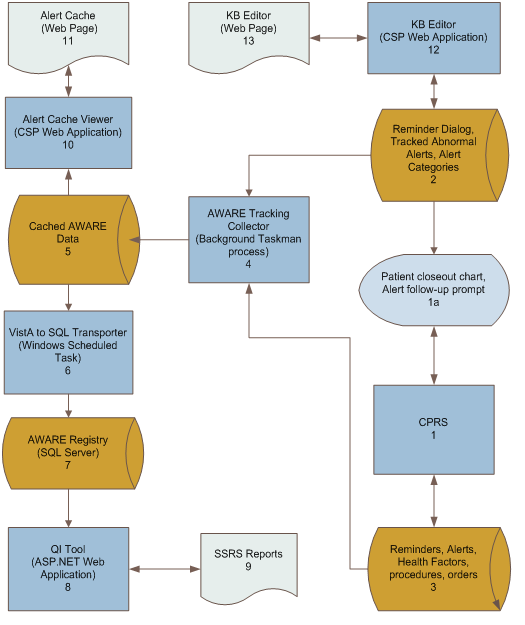


Figure - AWARE Software Components

Brief descriptions of the overall AWARE system are shown in the following table.

Table - AWARE System Description

| **Business Process ID** | **Business Process Name** | **Type** | **Owner** | **Description** |
| --- | --- | --- | --- | --- |
| 1 | CPRS | Existing | Clinicians and Quality Management personnel | CPRS is the primary GUI for VistA. Patient closeout chart and the Alert follow-up prompt are called from VistA and displayed in CPRS (as in 1a). |
| 1a | Patient closeout prompt, View notification alert, Reminders | New | CPRS users | Alerts users that results exist for the patient test results and prompts users to take action on completing a follow-up/treatment recommendation.  If the provider takes an action on the abnormal result while working in the chart, this prompt will be suppressed on the chart close out and it will NOT be displayed to the provider. |
| 2 | Reminder Dialog, Data Store, Tracked Abnormal Alert type and Alert Categories | New | CPRS users and Quality Management personnel | The data store contains tracked normal, abnormal, and incomplete alert types and alert categories, as well as, corresponding reminder dialogs and associated data to be used for AWARE. |
| 3 | Reminders and associated information Data Store | Enhanced | CPRS users, Quality Management personnel and population demographers | Reminders, Alerts, Health Factors, procedures, orders |
| 4 | Aware Tracking Collector | New | CPRS users, Quality Management personnel and population demographers | This TaskMan Tracking Collector will query alert follow-up information tracked from business process 1a) Patient closeout chart, Alert follow-up prompt) in addition to generalizing to other conditions such as occult blood, normal, abnormal and incomplete PSA and other alert types. |
| 5 | Cached AWARE data | New | CPRS users and Quality Management personnel | The collector stores AWARE alerts in this VistA file. Any alerts beyond 2 weeks old will be truncated. |
| 6 | VistA to SQL Transporter | New | CPRS users, Quality Management personnel and population demographers | This C# application connects to VistA to retrieve cached AWARE data and transports to SQL Server. |
| 7 | AWARE Registry (SQL Server) | New | CPRS users, Quality Management personnel and population demographers | Alert Cache data is stored in SQL AWARE Registry for historical and statistical purposes. Other QI Tool related tables are also stored in SQL Server. |
| 8 | QI Tool | New | CPRS users, Quality Management personnel and population demographers | This ASP.NET application is using SSRS Report Viewer to allow provider and patient safety officer to view AWARE Alerts historical and statistical reports. |
| 9 | SSRS Reports | Existing/Enhanced | CPRS users, Quality Management personnel and population demographers | Tracked Alert follow-up performance statistics.  Predefined and ad hoc reports will be generated as users request to display information collected and stored in the system. |
| 10 | Alert Cache Viewer (CSP) | New | CPRS users and Quality Management personnel | This CSP application serves browser request to retrieves VistA Alert Cache file and returns the contents to the browser. |
| 11 | Alert Cache (Web Page) | New | CPRS users and Quality Management personnel | Providers and patient safety officers use this web interface to view Alert Cache data. |
| 12 | KB Editor (CSP) | New | CAC | This CSP application server browser request to add, edit and validate VistA Alert Category and Alert Type files. |
| 13 | KB Editor  (Web Page) | New | CAC | This web interface allows CAC to create and/or edit Alert Categories and Alert Types. |

## Communications Architecture

The QI tool communicates with VistA by RPC calls.

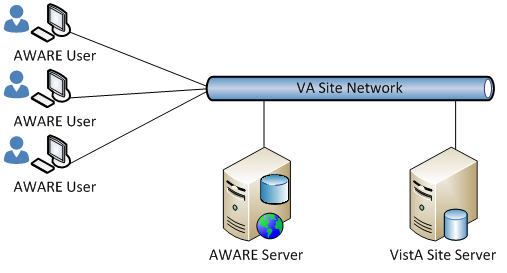


Figure – Communications Architecture

# Data Design

## Database Management System Files

The CPRS/AWARE.DLL integration is the core part of the AWARE system. See section 6.10 Database Repository for details on enhancements of a reporting database outside the CPRS/AWARE integration.

The figure below shows the application database schema from the MS SQL server management studio.

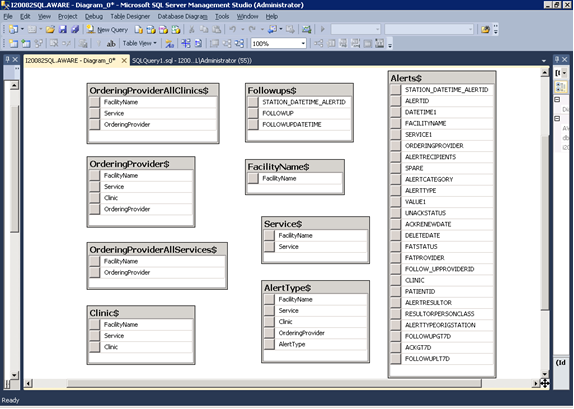


Figure – Application Database Schema from the MS SQL Server Management Studio

## Non-Database Management System Files

There are no non-database management system files that apply.

# Detailed Design

The design for the CPRS/AWARE DLL integration is described in detail in this section.

## Hardware Detailed Design

The hardware will consist of client workstations running an AWARE customized CPRS connected to VistA servers. It will also consist of VA approved devices to launch web applications (KB Editor, Alert Cache Viewer and QI Tool) on IIS Web Server and SQL Server machine.

## Software Detailed Design

### CPRS (1) and AWARE DLL (1a) Modules

The “Patient closeout chart, Alert follow-up prompt” is from an AWARE Dynamic Link Library (DLL) integrated with CPRS (noted as 1a in the figure below). The AWARE DLL is a COM object called from CPRS at patient closeout during the selection of a different patient. Before this new patient selection is actually allowed, an alert tracker function in the AWARE DLL determines if a follow-up action(s) has been made for any of the tracked alert types for that user. If none has been made, a resulting prompting screen is presented to the user for another opportunity to address this issue. A choice can be made by the provider to do a follow-up action via subsequent re-direction to a specific CPRS reminder dialog. This is called Follow-up Action Tracking (FAT).

The CPRS/AWARE DLL Modules Process Flow is shown below:

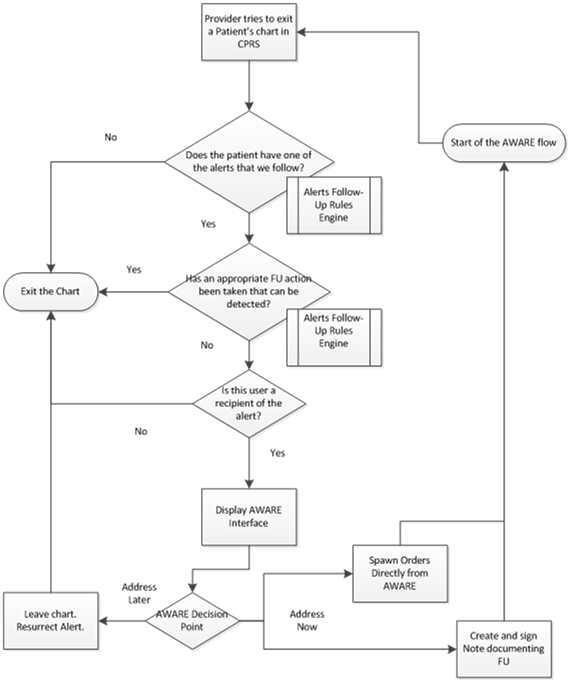


Figure - CPRS/AWARE DLL Process Flow

The figure below shows how the .dll, CPRS and web components work together.

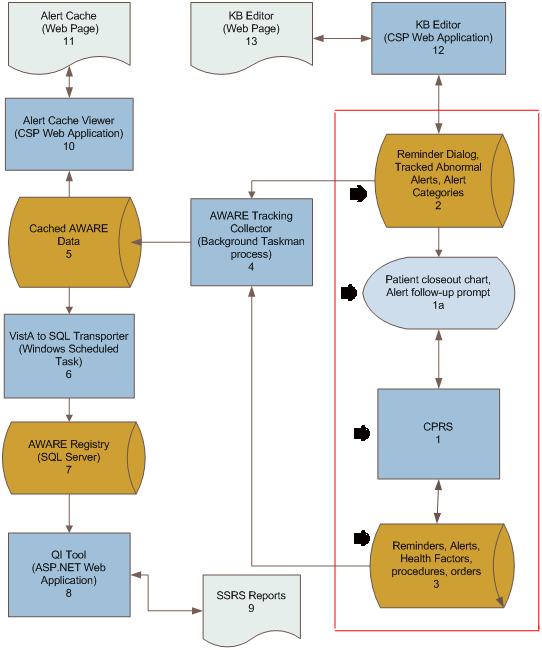
**

Figure - CPRS/AWARE DLL Process Design

The table below is the object mapping from the diagram for CPRS/AWARE process design.

Table - CPRS Objects in CPRS/AWARE DLL Process Design

| **Objects** | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ID | Name | Description | Service or Legacy Code | External Interface Name | External Interface ID | Internal Interface Name | Internal Interface ID | RSD |
| 1 | CPRS | Source for clinician supplied data including reminders (screening, other, etc., procedures, orders, health factors, etc.) | Alert tracking monitoring including abnormal alert results, reminder dialogs. | N/A | N/A | RPC access to progress notes, reminder dialogs.  ICPRSBROKER COM object communication with AWARE DLL | 3  1a | 2.6.3.3,  2.6.3.4 |

| **Internal Data Stores** | | | | |
| --- | --- | --- | --- | --- |
| **ID** | **Name** | **Data Stored** | **Steward** | **Access** |
| 3 | Data store for merge of real-time data from CPRS. | Merge of CPRS reminder dialog resulting for screenings, additional monitored real-time data as required during course of alert tracking, procedures, orders, and health factors merged with data retrieved from cached collections. | Clinicians/Authorized users for reminder dialogs, procedures, orders directly and data retrieved from cached collections. | CRUD (Creation, Retrieval, Updating, and Deletion) operations -Write/Read. |

The table below shows the object mapping for the diagram to the process design.

Table - AWARE DLL Objects in CPRS/AWARE DLL Process Design

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Objects** | | | | | | | | |
| **ID** | **Name** | **Description** | **Service or Legacy Code** | **External Interface Name** | **External ID** | **Internal Interface Name** | **Internal Interface ID** | **RSD** |
| 1a | Patient Closeout Chart. Alert Follow-up prompt  (AWARE DLL) | Abnormal Test Results Follow-up initiation (prompting)  And redirection for provider Follow-up actions in a Reminder Dialog | Prompting as Follow-up initiation opportunity | N/A | N/A | ICPRSBROKER COM object communication with CPRS  RPC Access to Reminder Dialogs, associated tracked abnormal alerts and alert categories | 1  2 | 2.6.3.1,  2.6.3.2 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Internal Data Stores** | | | | |
| **ID** | **Name** | **Data Stored** | **Steward** | **Access** |
| 2 | Reminder Dialogs, associated tracked abnormal alerts and alert categories | Reminder Dialogs, associated tracked abnormal alerts and alert categories | Providers ( clinicians) and via Web editor by CAC use to edit tracked alert categories and alert types | CRUD operations. Read tracked  Read of tracked abnormal resulting alerts, alert categories.  Generate/Display Follow-up Action  Tracking Prompts |

### KB Editor

KB Editor is a web application designed to create alert categories and alert types to define critical alert and follow-up actions. The following figure and table describe KB Editor process design.

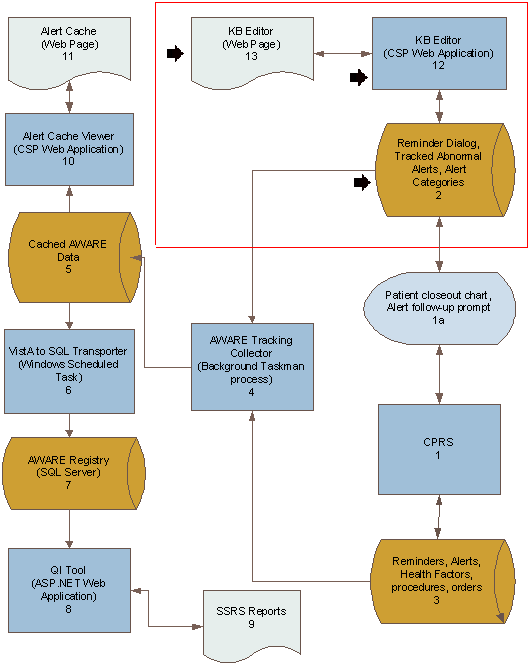


Figure - KB Editor Process Design

The table below shows the object mapping for the diagram to the process design for the KB editor.

Table - KB Editor Objects in KB Editor Process Design

| **Objects** | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Service or Legacy Code** | **External Interface Name** | **External Interface ID** | **Internal Interface Name** | **Internal Interface ID** | **RSD** |
| 12 | KB Editor | Define critical alerts and follow-up actions. | Store Alert categories and Alert Types on VISTA system. | N/A | N/A | CSP access to Alert Category  And Alert Type data | 2 | 2.6.1 |
| 13 | KB Editor Web Page | Web Interfaces for KB Editor | Provide web interfaces | N/A | N/A | N/A | N/A | 2.6.1 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Internal Data Stores** | | | | |
| **ID** | **Name** | **Data Stored** | **Steward** | **Access** |
| 2 | Reminder Dialog, Alert Category  Alert Type files | Alert Category defines critical alert category.  Alert Type defines critical alert.  Alert Cache Collector stores critical alerts into Alert Cache file. KB editor defines the rules of what data is stored. | Each Facility for their authorized users (clinician’s , patient safety officers) | CRUD operations of create/update for cached data store |

### Alert Cache Collector

The Alert Cache Collector serves as a staging area for recent Alerts. The figure below diagrams the process flow for that process.

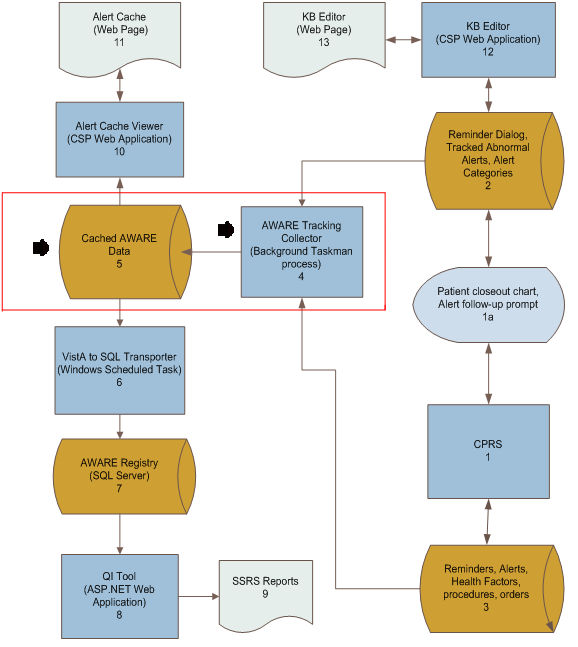
****

Figure - Alert Cache Collector Process Design

The table below shows the object mapping for the diagram to the process design for the Alert Cache Collector.

Table - Objects In Alert Cache Collector Design

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Objects** | | | | | | | | |
| **ID** | **Name** | **Description** | **Service or Legacy Code** | **External Interface Name** | **External Interface ID** | **Internal Interface Name** | **Internal Interface ID** | **RSD** |
| 4 | Alert Tracking Collector | Combined multiple data collections and subsequent storage of alert tracking data | Collector of alert tracking data as cache on each VISTA system. | N/A | N/A | Vista access to data from Alert and Alert tracking files, reminder dialog data with health factors, procedures, orders, etc  Vista FileMan write/update of Cached Tracked Alert Audit/Log File | 3  5 | 2.6.2.1.1  2.6.2.1.1 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Internal Data Stores** | | | | |
| **ID** | **Name** | **Data Stored** | **Steward** | **Access** |
| 5 | Cached Tracked Alert Audit/Log file | Cached Tracked Alert follow-up activities for up to 2 weeks back, and staging for transfer to longer term SQL storage.  Use for pro-active info for provider use with their follow-up work | Each Facility for their authorized users (clinician’s , patient safety officers) | CRUD operations of create/update for cached data store |

### Alert Cache Viewer

Alert Cache Viewer is a web application designed for provider and patient safety officers to view critical alerts stored in Alert Cache file. The following figure and table describes each objects in the Alert Cache Viewer process design.

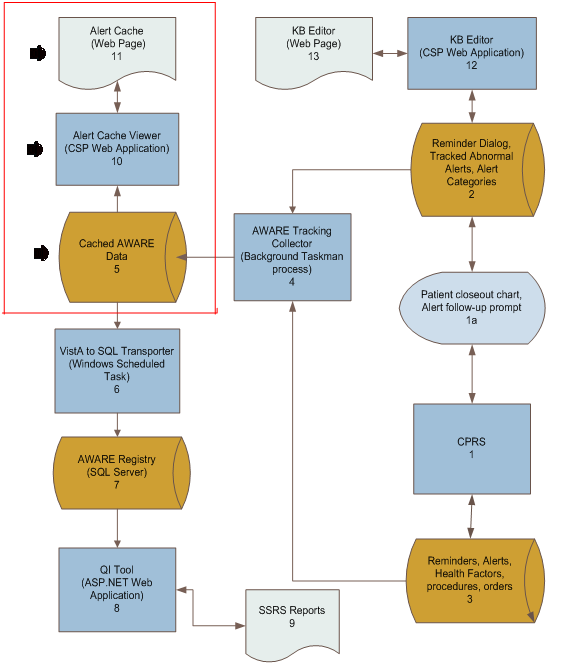


Figure - Alert Cache Viewer Process Design

The table below shows the object mapping for the diagram to the process design for the Alert Cache Viewer.

Table - Objects In Alert Cache Viewer Process Design

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Objects** | | | | | | | | |
| **ID** | **Name** | **Description** | **Service or Legacy Code** | **External Interface Name** | **External Interface ID** | **Internal Interface Name** | **Internal Interface ID** | **RSD** |
| 10 | Alert Cache Viewer | Web app for viewing recent alert tracking activity for which timely pro-active actions can be made from providers or team providers, and monitored by patient safety officers | Web viewer of Cached Tracked Alert Audit/Log file  Web displays for authorized users | N/A | N/A | CSP access to Cached Tracked Alert Audit/Log file | 4 | 2.6.1 |
| 11 | Alert Cache Web Page | N/A | N/A | N/A | N/A | N/A | N/A | 2.6.1 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Internal Data Stores** | | | | |
| **ID** | **Name** | **Data Stored** | **Steward** | **Access** |
| 4 | Cached Tracked Alert Audit/Log file | Cached Tracked Alert follow-up activities for up to 2 weeks back, and staging for transfer to longer term SQL storage.  Use for pro-active info for provider use with their follow-up work | Each Facility for their authorized users (clinician’s , patient safety officers) | CRUD operations of create/update for cached data store |

### VistA to SQL Transporter

VistA to SQL Transporter is a window scheduled process with its main function to retrieve VistA’s AWARE Alert Cache data and store them into MS SQL Tables. The following Figure and table show how the components interact in VistA to SQL Transporter Process design.



Figure - VistA to SQL Transporter Process Design

The table below shows the object mapping for the diagram to the process design for the VistA to SQL transporter.

Table - Objects In VistA to SQL Transporter Process Design

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Objects** | | | | | | | | |
| **ID** | **Name** | **Description** | **Service or Legacy Code** | **External Interface Name** | **External Interface ID** | **Internal Interface Name** | **Internal Interface ID** | **RSD** | |
| 6 | Vista to SQL Transporter (Windows Scheduled Task) | From cached collector collections, build daily transfers of accumulated data for transfer to external Windows server machine for subsequent long term SQL storage. | Gathering of snapshots of Alert Cache Data to combine into accumulative SQL storage for subsequent review and statistical analysis. | RPC access of Vista Alert Cache data | 5 | SQL calls to SQL Server tables | 7 | 2.6.2.1. | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Internal Data Stores** | | | | | |
| **ID** | **Name** | **Data Stored** | **Steward** | **Access** |
| 5 | Assemblies from Tracked Alert audit/log file cache. | Daily Tracked Alert audit/log file cache. | Eventually VA owners including administrative staff on a facility/service/ provider level | CRUD operations. Read |
| 7 | Tracked Alert Follow-up activities as stored in SQL Server tables | Long term historical use for dashboards web use. | VA staff including administrative staff | CRUD operations. Write/Read |

### QI Tool

The QI Tool is the reporting tool of the AWARE application. At its core as an ASP.NET Web Application, it is tightly integrated with MS SQL Server and MS Reporting Services. This provides a complete platform for the delivery of alert reports from the AWARE database. The QI Tool is capable of authenticating and authorizing users in the viewing of the reports packaged with the AWARE solution. In addition, any custom reports that may occur post installation can be accommodated provided they are installed in the AWARE reports folder.

The figure below outlines the process flow for the QI tool.

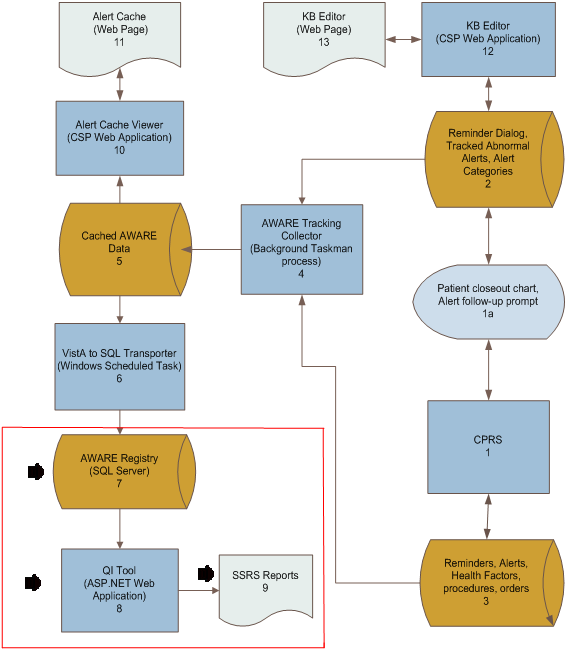


Figure - QI Tool Process Design

The table below shows the object mapping for the diagram to the process design for the QI tool.

Table - Objects In QI Tool Process Design

| **Objects** | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Service or Legacy Code** | **External Interface Name** | **External Interface ID** | **Internal Interface Name** | **Internal Interface ID** | **RSD** | |
| 8 | Reporting Tool Web Application | Reporting of Historical Tracked Alert Follow-up Alert activities  1. Web based displays using SSRS(SQL Server Reporting Services)  2. Web displays for authorized users with QI Manager | Long term trend reporting | N/A | N/A | Access to SSRS Report Manager Reports and their subsequent access to SQL Server table Alert tracking data stores, and to Administrative SQL table stores | 7 | 2.6.2.2.1. | |
| 9 | SSRS Reports | Web Interface for SSRS Report | N/A | N/A | N/A | N/A | N/A | N/A | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Internal Data Stores** | | | | | |
| **ID** | **Name** | **Data Stored** | **Steward** | **Access** |
| 7 | SQL Server Long Term Tracked Alert Follow-up Activity stores | Storage for long term web dashboard displays | VA staff including administrators | CRUD operations- Read |
|  | QI Manager Administrative SQL table stores | Table store for authorized users | Facility users for access to particular reports and SQL data | CRUD operations- Read/Write |

### Local Data Structures

Local data structures, which are part of the business knowledge base rules for guiding re-direction opportunities to addressing follow-up actions, include the following for the CPRS/AWARE DLL integration:

* A FileMan file for a various tracked alert categories of eligible alert notifications ( i.e., AA critical lab alert category)
* A FileMan file of specific tracked alert types ( i.e., critical PSA lab result alert)
* For each specific tracked alert type, a link to an associated Reminder dialog
* For Reminder dialog an associated Text Integration Utilities (TIU) template
* The VA FileMan National Alert and Alert Tracking files

The actual details for connected follow-up actions via the associated Reminder dialog/TIU template are defined through Clinical Application Coordinators (CACs) who are responsible for their functioning via proper design using a VA CPRS reminder dialog builder. These have many sub-components, such as use of various VA FileMan files including that for orderable items, pharmacy orderable items, facility lab test names, consults, text comments, health factors, procedures, orders, and other available data structure elements that are naturally used by CACs in the design of Reminder dialogs as composite Reminder dialog elements. Guidance will be provided as part of AWARE project for specific features needed in the design of these Reminder Dialog designs for proper FAT workings.

In addition, as an auxiliary function (but outside this CPRS/AWARE DLL integration), a new file for Tracked Alert auditing/logging is used as a cache for local facility access by authorized users as a source of recent alert tracking activity for daily or other periodic transfer and storage into SQL Server database tables as historical data to be used for further reporting purposes.

#### CPRS AWARE User Interfaces

The AWARE DLL Follow-up Action tracking module provides a redirected prompt screen for a provider to consider follow-up action for their patient upon the occurrence of a tracked alert while the patient is being seen in CPRS. An example of this user interface is shown below.

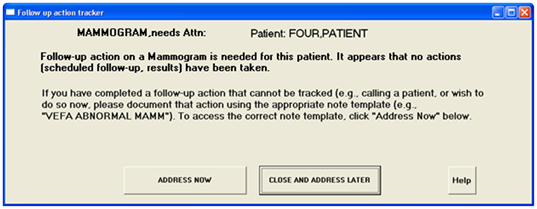


Figure - “Prompt Screen” User Interface

Upon clicking the “ADDRESS NOW” button, the user will be re-redirected automatically to a specific Reminder dialog associated with that type of tracked alert. An example of this is shown below:

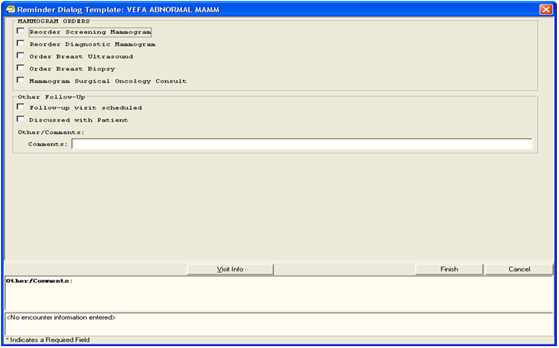


Figure - Reminder Dialog

The logical configuration necessary per user to allow for this screen prompting and choice for follow-up actions depends on the following:

* Each user of the AWARE system must have the AWARE DLL registered on their desktop.
* Setup of alert notification types to be monitored per user should include those alert types the AWARE system is monitoring. Usually these are set up as default for all users, but they can be customized for each user so they can be aware and see these actual tracked alert notifications. These notifications are configurable within CPRS per user as shown below. They are also configurable on a system-wide or default basis through the CPRS options made available by CACs.

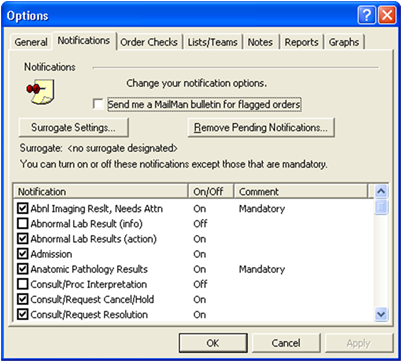


Figure - User Setup

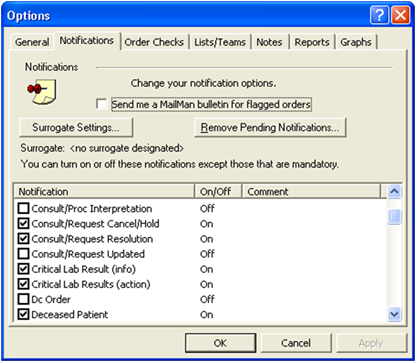


Figure - User Setup Continued (above)

* Allows specific Reminder Dialogs to be connected with the user by a defined TIU Template for an associated tracked alert type. This connection can be based on a specific user, or by service, division, or system level. This user configuration is done by CACs as part of their normal duties for using Reminder Dialogs. The TIU template should also be a Shared TIU template available to all or certain groups of users.

#### KB Editor User Interfaces

KB Editor is a web application designed to create alert categories and alert types to define critical alert and follow-up actions.

The following figure shows that after a successful login, the KB Editor main page containing Alert Categories and Alert Types appears. On this page a user can add or update an Alert Category and/or Alert Type.

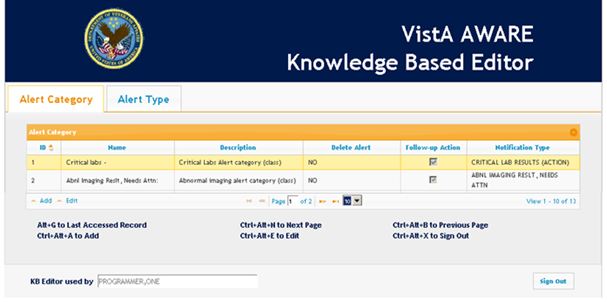


Figure - List of Alert Categories

The figure below shows how to create an alert category.



Figure - Alert Category Form

The following figure shows a list of alert types.

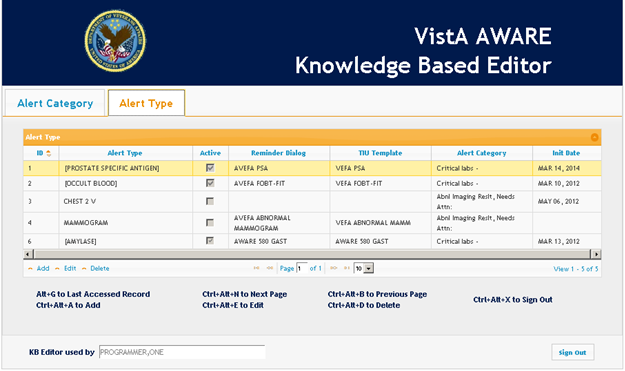


Figure - List of Alert Types

The figure below shows the creation of an Alert Type.

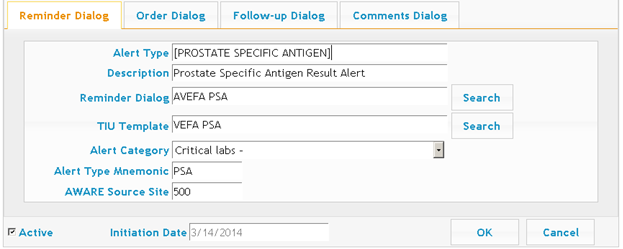
**

Figure –Alert Type Form

When an alert is identified as an AWARE (critical) alert, follow-up actions will be presented during CPRS Patient Closed-out event. Those follow-up actions are defined in Order, Follow-up and Comments dialog tabs.

The figure below shows a list of previously created order dialog elements. These will be displayed to the clinician user.

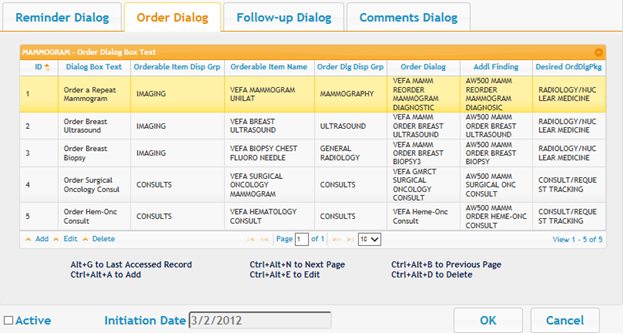


Figure 7 –List of Order Dialogs

The figure below shows the creation of an order dialog text. This will be presented to the clinician user in a reminder dialog.

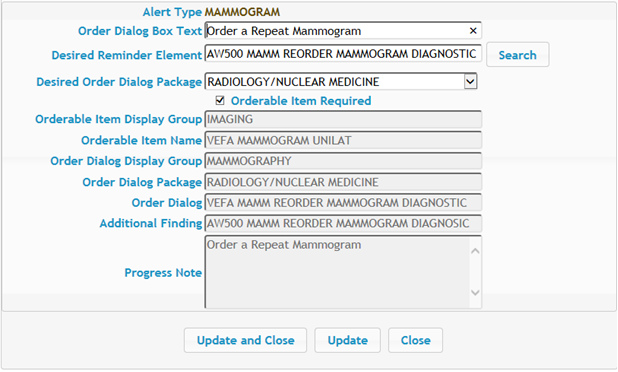


Figure –Order Dialog Text Form

The figure below shows a follow-up dialog element definitions list. These will be displayed to the clinician user.

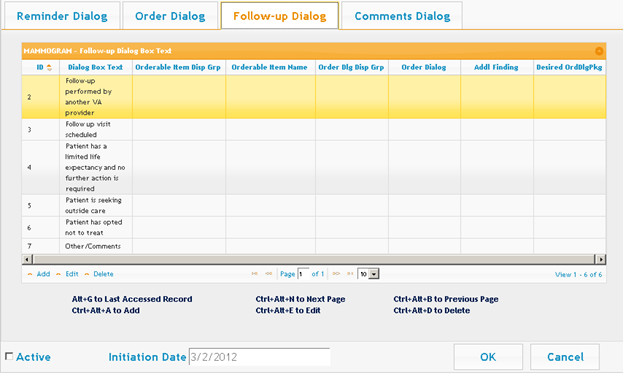
**

Figure –List of Follow-up Dialogs

The following figure shows creation of follow-up dialog text.

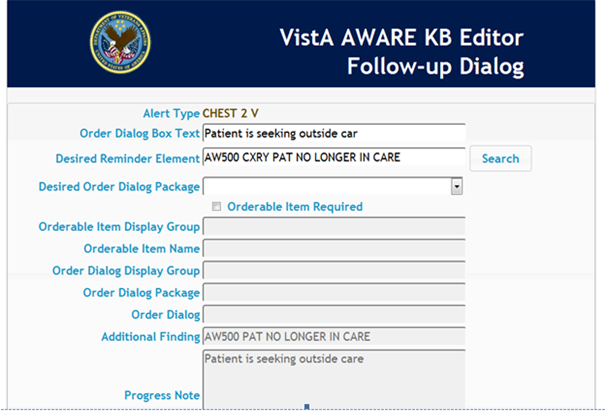
**

Figure - Follow-up Dialog Text Form

The figure below shows the follow-up comments dialog element.

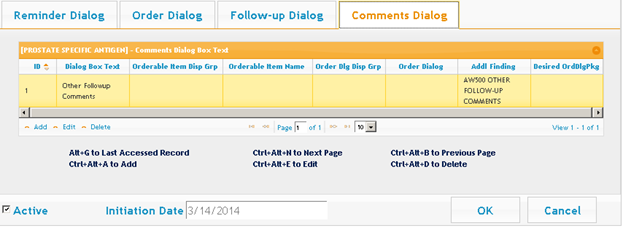


Figure - List of Comments Dialogs

The figure below shows the follow up comments element being created.

**

Figure - Comment Dialog Text Form

The figure below shows order dialog text being created that will be presented to a clinician user.

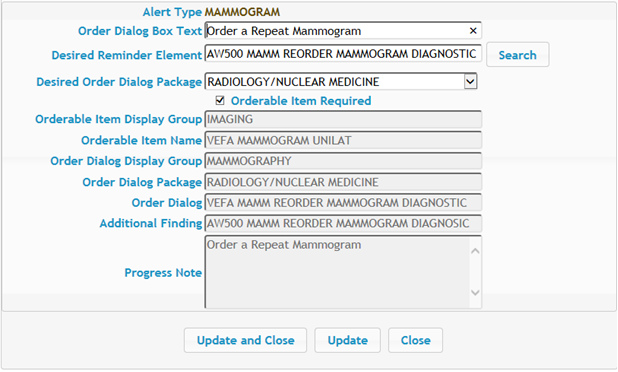


Figure - Order Dialog Text Form

#### Alert Cache Viewer User Interfaces

Alert Cache Viewer is a web application designed for provider and patient safety officers to view critical alerts stored in Alert Cache file. Figures below show the user interfaces.

The figure below shows a report of alerts being shown by the Alerts Cache Viewer. The Alerts Cache Viewer shows recent alerts quickly.

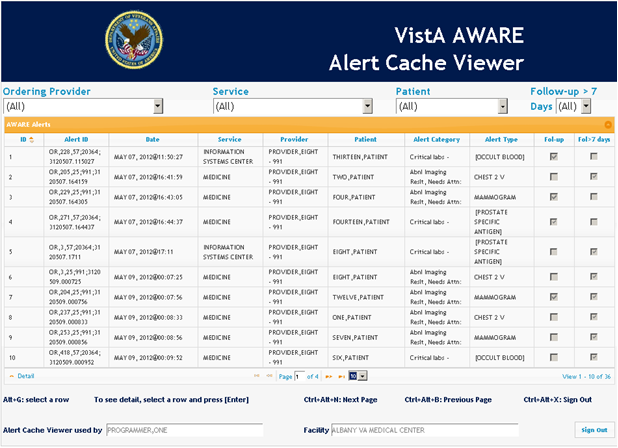


Figure - List of Cached Critical Alerts

The figure below shows details of a critical alert in the Alerts Cache Viewer.

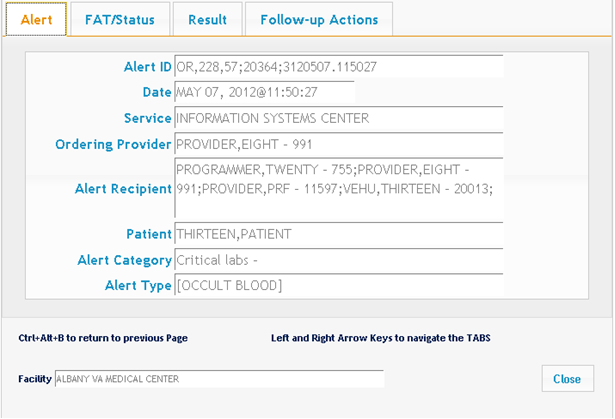


Figure - Detail Cached Critical Alert Information (GENERAL)

The figure below shows more critical alert information.

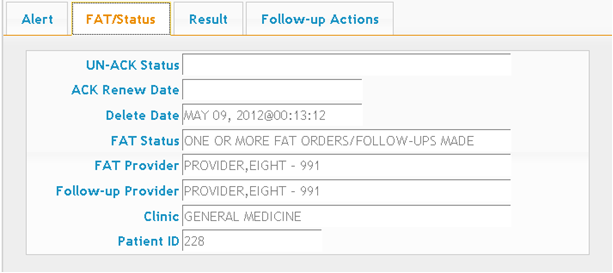


Figure - Detail Cached Critical Alert (FAT/STATUS)

The figure below shows the results detail of a critical alert.

**

Figure - Detail Cached Critical Alert (RESULT)

The figure below shows the follow-up actions taken.

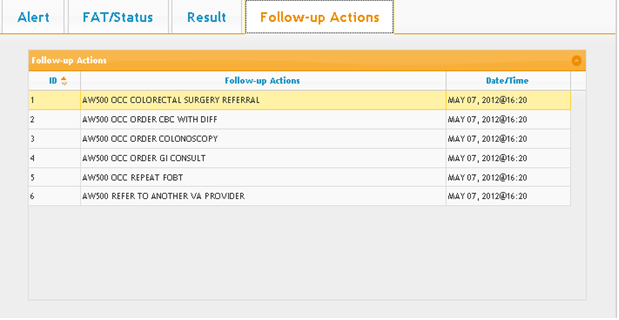


Figure - Detail Cached Critical Alert (FOLLOW-UP ACTIONS)

#### QI Tool User Interfaces

The QI Tool is the reporting tool of the AWARE application. At its core as an ASP.NET Web Application, it is tightly integrated with MS SQL Server and MS Reporting Services. This provides a complete platform for the delivery of alert reports from the AWARE database. The QI Tool is capable of authenticating and authorizing users in the viewing of the reports packaged with the AWARE solution in addition to any custom reports that may result post installation provided they are installed to the AWARE reports folder.

##### Logon Screen

When the user first launches the QI Report Manager Tool the system will check to see if a current session is available for the user to browse. If there is no session available to the current http user context then the system will redirect the user to the Login screen where the user may enter their credentials to access the system. The screen is shown in the following figure.

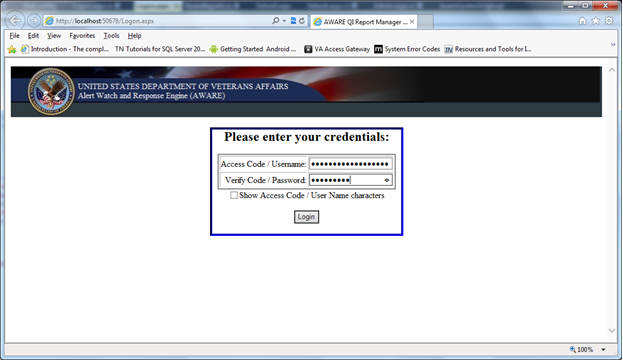


Figure – QI Tool Logon Screen

On this screen, the user will enter their credentials for accessing the system.

**Access Code / Username:** The access code / username name by default can either be masked as shown in the figure above, or unmasked as shown in the following figure below. The setting for controlling the masking by default is defined in the web.config file by the name of “LogonUserNameIsMasked”. When this setting is set to true the access code/user name will be masked and the “Show Access Code / User Name characters” check box will be unchecked.

The access code / user name text box can also be set to default to the current user whom has logged into Windows. When this setting is enabled the access code / user name text box becomes non-editable, the setting is stored in the web.config file by the name of “SharedComputer”. This setting is set to true by default to allow multiple users and multiple type users to access the system from a single computer without logging in and out of the system.

**Verify Code / Password:** The verify code / password text box will always mask the users input as to keep it private.

**Show Access Code / User Name characters:** The show access code / user name checkbox allows a user to verify the input of their access code or user name. When the checkbox is checked the user will be able to see the input in readable characters as shown in the figure below, when the checkbox is not check the input will be masked as shown in the previous figure above..

**Login:** When the user clicks the Login button, the system will verify the provided credentials against 1 of 3 different types of logons.

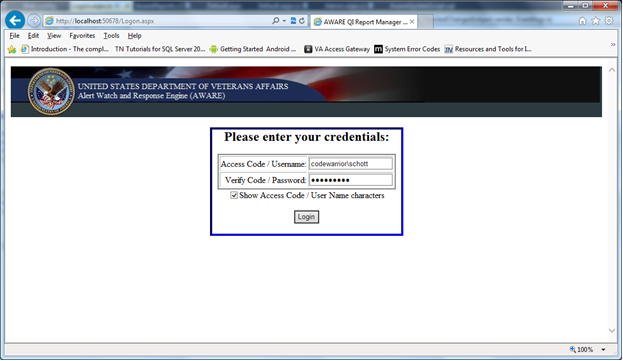


Figure – QI Tool Logon Screen 2

After the user has clicked the login button, the provided credentials will be validated in the following waterfall fashion.

**Active Directory:** The active directory is queried for the existence of the account provided in the Access Code / User Name text box. If the account is found, the credentials are then passed to the credential store to determine if the credentials are valid. Active directory logon is only valid for super users at the time of this writing. If there are some common active directory groups among the users then that role can be expanded. Upon initial installation of the application there are no users or groups set up in the system; because of this the system needs a way to validate the user for initial setup. In the web.config file there is a setting by the name of “SuperUsers”, and in this list there needs to be at least one name to access the system for configuration. Although this list can host multiple names it is recommended only for those who will be responsible to maintain the system. If a valid active directory account is used in the logon process and that account does not exists in the super users list, then access will not be granted.

**Local Application User:** If the active directory verification failed then the credentials are processed against the local application users. These are users that a super user would set up to have designated rights for the specific user. A local application user can also be a member of the super users list. These credentials are stored in the USERS table of the AWARE database.

**Vista Access Verify codes:** If the local application user verification failed the final check would be to validate the supplied credentials to VistA via a web service. If this verification succeeds then the user will gain access to the QI Tool based upon the group retrieved from VistA. If this verification fails, the user is returned to the logon screen with an error message identifying why the verification failed.

After successful verification of the supplied credentials, the user will be directed to the QI Tool main screen as shown in the following figures.



Figure - Validated as Super User

In the following figure, the logged on user was validated as a super user. When a user is designated as a super user they are able to move around in the application without limitation; they have access to everything the application can do. When the application is run under the super user context there will be a “Tools” link in the yellow banner bar. The super user would click the Tools link to configure the QI Tool application.

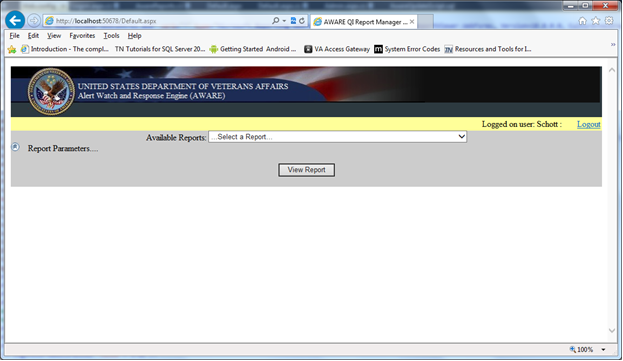


Figure - Validated as Local Application User

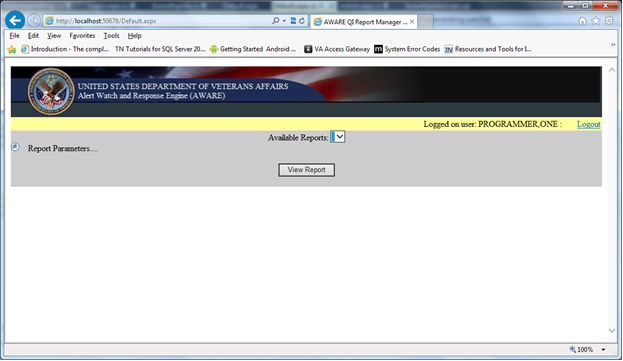


Figure - Validated as VistA User

The figures below show the screen as authenticated as a local application user and VistA user. All users will receive Logged on user identifier and logout link on the right side of the yellow banner bar.

**Logout:** When the super user clicks the “Logout” link their session will be closed and will return the user to the logon screen.

**Available Reports:** Just below the yellow banner bar is drop down list of available reports based on the credentials entered during the logon process. A super user will see all reports on the reporting services server. During the initialization of the application the QI Tool will query the Reporting Service for all reports located in the defined root folder. In the web.config application there is a setting by the name of “ReportsRootFolder” that defines where the root folder is located.

**Report Parameters:** The report parameters are a dynamic panel of parameters to pass to the reporting service engine for report generation.They are updated every time a different report is selected from the available reports drop down list. The figure above shows the dynamic report parameters panel.

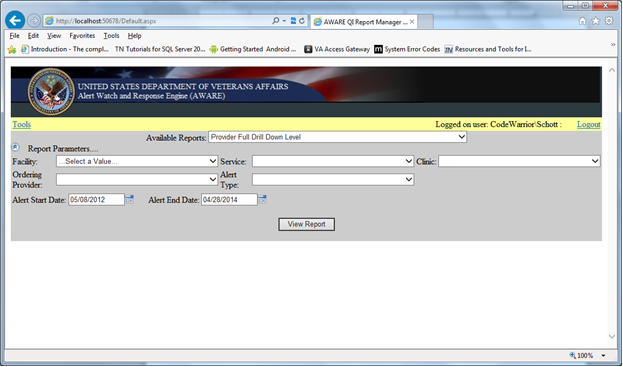


Figure - Dynamic Reports Parameter panel

Each parameter for any given report starts at the facility level and progressively narrows the criteria of each report. The parameters shown in the figure above shows the maximum number of parameters for any report.

**Start and End Dates:** All reports will have an Alert Start and End Date parameters; the default for the Alert Start Date can be set by a setting in the web configuration file by the name of “DefaultStartDateSpan”. This Alert Start Date shown above is set at 720 days. The default is 90.

**View Report:** When the user clicks the “View report” button, the application will gather the parameters and pass them to the reporting services web service to generate the selected report in the Available Reports drop down list as shown in the following figure.

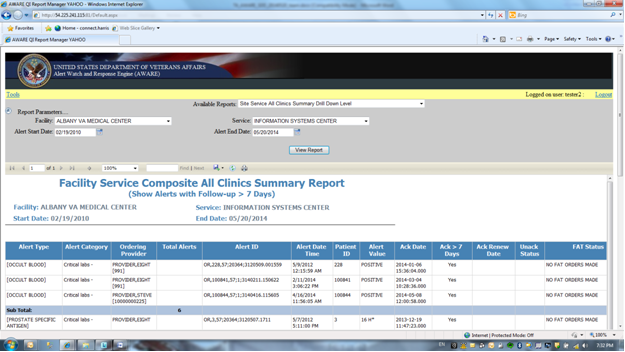


Figure - Selected Reports Generated

When the report is generated by the reporting service, an additional toolbar will be present which facilitates the user to navigate the pages of the report, to zoom in the visibility of the report, search for text within the report, options for exporting the report, refresh the report based on the selected parameters the report was initially executed with, and the ability to print the document.

The reporting service web service is configurable in the web.config file through a setting by the name of “AwareQIManager\_Reportingservice2010\_ReportingService2010“. Through this setting the user can set the http / https identity and port. It is important that matches the installation endpoint. It can be discovered through the Reporting services Configuration Manager installed when MS SQL Server was installed.

##### QI Tool – Tools Link

When the super user clicks the Tools link in the yellow status bar the user will be presented with the QI Tool Administration Tools screen as shown in the following figure.

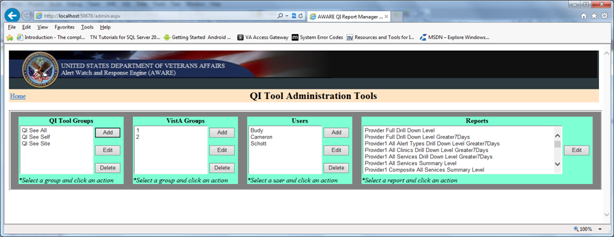


Figure - QI Tool Administration Tools

On the QI Tool AdministrationTools, the super user is able to create QI Tool based groups, define expected VistA Groups, create QI Tool based users, and modify the presentation name for the discovered reports.

**QI Tool Groups:** At the time of this writing the security authorization for the QI Tool begins at the QI Tool Groups. It is here that the super user may define a group, specify which reports the group is authorized to view, and assign local application users (Users) to the groups. There is no limit on the number of groups that can be defined. The super user may add, edit or delete QI Tool Groups from this screen.

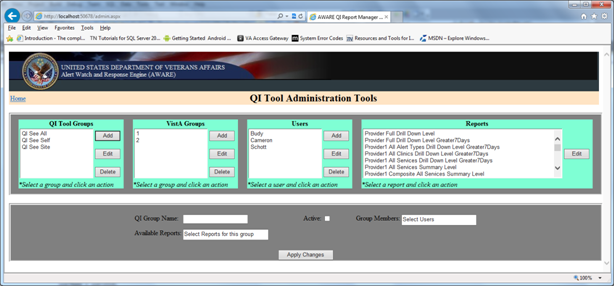


Figure - QI Group Panel – Add Mode

In the QI Group panel as shown above, the super user may define the name, and mark the group as being active or inactive. The Available Reports drop down list displays all the reports that were discovered when the application was initialized. The Available Reports list has checkboxes for the super user to select. As shown in the following figure below, multiple report selections are permitted. The Group Members drop down list contains all application defined users where specific authorizations are needed. The Group Members list has checkboxes for the super user to select. As shown in Figure 51, multiple user selections are permitted.

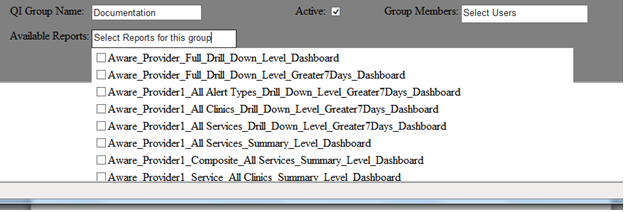


Figure - Available Reports Drop Down List



Figure - Group Members Drop Down List

If the super user chose to “edit” a selected QI Group, the name would be populated, the active checkbox would be marked to its current status, and the available reports and group members would be selected.

When the super user clicks the ‘Apply” button the new group will be added to the database if the user was in the “Add” mode. If the user was in the “Edit” mode, the changes for the selected group will be updated to the database. All fields in the QI Group panel will be re-initialized.

If a super user chooses to “Delete” a selected group, a warning prompt appears as shown in the following figure.

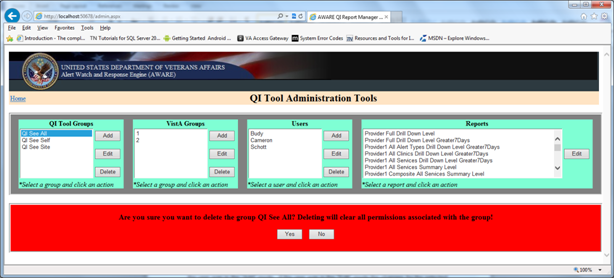


Figure - Delete QI Group Warning

**Yes:** When the super user clicks the “Yes” button the selected QI Group will be deleted from the database along with the associated group authorizations. The users of these groups will remain without authorizations.

**No:** When the super user clicks the “No” button, the delete action is cancelled and nothing is done affecting the selected QI Group.

**VistA Groups:** When a user logs in using their Access/Verify key pair, those credentials are passed to the VistA web service for verification. If the verification is successful, the web service will return the VistA user group for this user. It is here where the super user defines the expected Vista Group and maps it to an existing QI Tool Group as shown in the figure below. The super user may add, edit or delete Vista Groups from this screen.

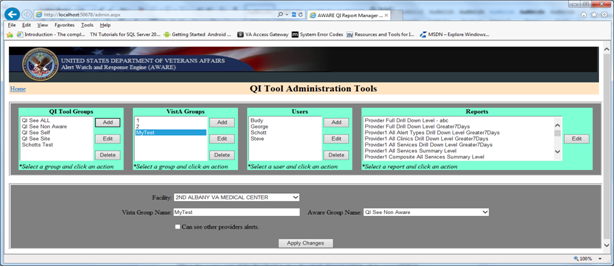


Figure - VistA Groups Panel – Add Mode

When the super user clicks the “Add” button the VistA Groups panel will be presented as shown in the figure above.

The Facility drop down list will list all the facilities which are available for this installation and stored in the database in the FACILITIES table. This field is read only.

The Vista Group Name is the name from VistA for the intended group. It must be exactly the same name as passed from the VistA web service. This field is used as a mapping, and readability is only of concern to the application.

The Aware Group Name is a drop down list with all the available QI Tool Groups which are active as shown below. This drop down list allows a single selection only.

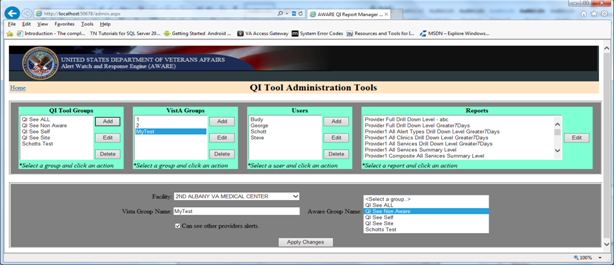


Figure - VistA Groups – AWARE Group Drop Down List

The “Can see other provider’s alerts” checkbox identifies whether this specific Vista Group can see other providers alerts or just their own. If the checkbox is checked then any provider belonging to this group will be able to see the other providers’ alerts. If the checkbox is not checked then the provider will only be able to see their own alerts.

If the user clicks the “Edit” button the details for the selected VistA group will be pre-populated.

When the super user clicks the “Apply” button, if in the “Add” mode a new group mapping will be written to the database: if in the ‘Edit” mode the changes will be updated to the database. These mappings are stored in the database in the AWARE\_VISTA\_GROUP\_MAPPINGS table.

If the super user clicks the “Delete” button they will be presented with a warning prompt as shown in the following figure.

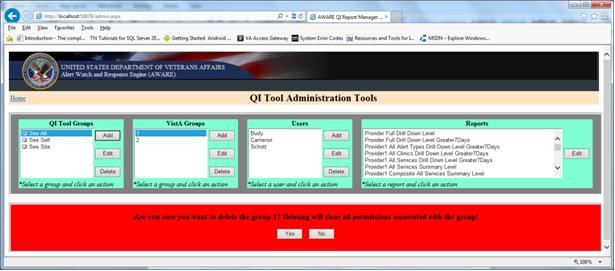


Figure - VistA Group Deletion Warning

**Yes:** When the super user clicks the “Yes” button, the selected VistA Group will be deleted from the database along with the associated group mapping. The providers who belong to this VistA group will be denied access as there is no mapping to authorizations.

**No:** When the super user clicks the “No” button the delete action is cancelled and nothing is done affecting the selected VistA Group.

**Users:** The super user can define local application users for any situation where a user may not have a valid access/verify key pair without making that user a super user. These users are stored in the database in the USERS table. The super user may add, edit or delete users from this screen. When the Add or Edit button is click the Users panel will be presented as shown in the following figure.

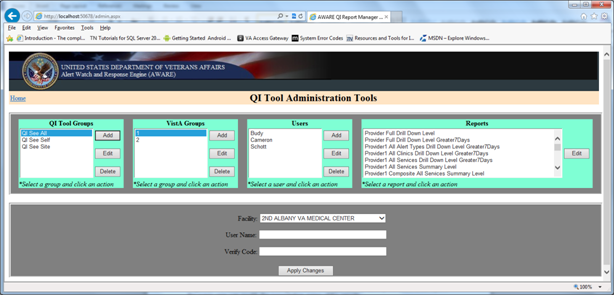


Figure - Users Panel – Add Mode

The Facility drop down list will list all the facilities which are available for this installation and stored in the database in the FACILITIES table. This field is read only.

The User Name field will display the specific user.

The Verify Code is where the users verify code /password should be entered. This field is encrypted in the AWARE database stored in the USERS table. The characters are not masked on the UI to facilitate easy retrieval and accuracy when entering.

When the super user clicks the “Apply Changes”, if in “Add” mode the new user would be added to the database, if in “Edit” mode the selected users details would be updated to the database.

If the super user clicks the ‘Delete” button after selecting a user a warning would prompt the user for confirmation as shown in the following figure.

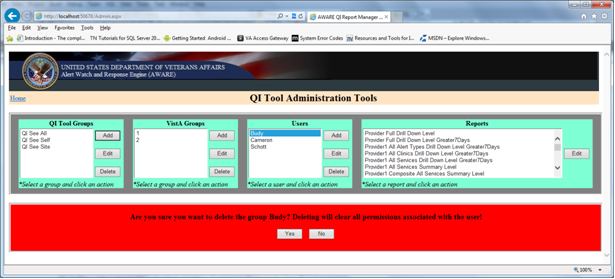


Figure - Delete User Warning Prompt

**Yes:** When the super user clicks the “Yes” button the selected User will be deleted from the database along with any authorizations associated with the selected user.

**No:** When the super user clicks the “No” button the delete action is cancelled and nothing is done affecting the selected User.

**Reports:** To facilitate easier reading and allow the super user to identify the discovered reports the super user can modify the presentation name of the reports. When the user selects a report from the report listing and clicks the “Edit” button Reports panel will be displayed as shown in the following figure.

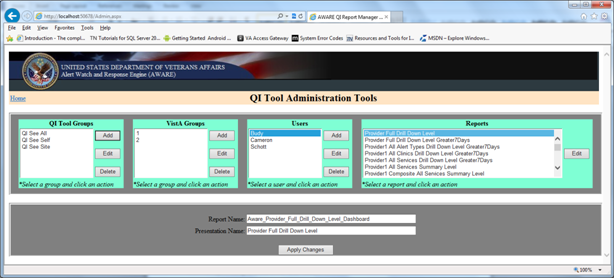


Figure - Reports Panel

**Report Name:** The report name is the actual name of the report as known to the application and reporting services.

**Presentation Name:** The name which appears within the application for the user to see. This name can be anything that makes it easier to use, with the only constraint being that it has to be unique.

When the super user clicks “Apply Changes” the new presentation name is updated to the AWARE database in the SECURITY\_ITEMS table.

**Home:** The “Home” link in the left hand side of the tool bar allows the user to navigate back to the home page of the application.

## Hardware Interfaces

Summary of the hardware interfaces for the AWARE project.

Asterisk (\*) indicates the only hardware interfaces for the basic CPRS/AWARE integration specifications for this document.

1. \*Desktop PCs for running CPRS/AWARE DLL with an AWARE DLL interface with VistA server machine(s).
2. Desktop PCs or other VA approved devices to run web applications (KB Editor, Alert Cache Viewer, and QI Tool).
3. \*MUMPS VistA server machine(s) with underlying Linux, VMS, or MS Windows operating systems for communications between CPRS/AWARE DLL and VISTA server data.
4. Windows Server machine per facility with configured SQL Server and IIS Internet Information System for storage of data received from a VistA server, and for authorized user viewing, long term historical reporting, and other configuration purposes.

The other additional aforementioned interfaces mentioned here are parts of the overall AWARE system specification.

## Software Interfaces

Summary of the software interfaces for the AWARE project.

(\*\*) indicates the software interfaces for the core CPRS AWARE Integration part of this document.

1. \*\*Client interface using CPRSwith AWARE DLL software interfaces with VistA data using Remote Procedure Calls (RPCs).
2. \*\*VistA data retrieved and written from an AWARE DLL using VA Remote Procedure Calls (RPCs).
3. VistA TaskMan process stores critical alerts into Alert Cache file.
4. Web interface using Intersystem’s CSP to communicate with VistA data through CSP Gateway.
5. CSP Gateway module running on IIS Web server to request and retrieve CSP page to and from VistA Cache Server.
6. SQL Server data storage of received VistA data, and web display for authorized user viewing, long term historical reporting, and other configuration purposes.
   1. The long-term storage retrieval is by RPC with transfer into SQL Server tables from a scheduled Windows daily task.
   2. Reporting functions through web server includes those for historical reporting such as those employed with MS SQL Server Reporting Services (web dashboards). It also includes authorized user viewing of current or recent tracked alert activity from a facility such as via web displays with access to VISTA data via RPCs.

The other additional aforementioned interfaces are enhancements that are part of the overall AWARE System.

## Communications Interfaces

Communications via Remote Procedure Calls (RPCS) and HTTP web server/client functions are as discussed in section 6.4 Software Interfaces above.

## Memory Constraints

The applications are mostly I/O bound. No known memory constraints.

## Special Operations

Backup and recovery operations will be performed by VA personnel using current standards.

## User Characteristics

Users of the CPRS/AWARE DLL include clinician staff as these users are the ones who perform alert follow-up actions to be tracked. Refer to the AWARE RSD Section 2.6, Functional Specification sections 2.6.3.1 and 2.6.3.2.

CPRS/AWARE DLL integration includes clinicians, authorized users, administrative and executive level staff, patient safety officers, which are users who do additional viewing and reporting analysis of this data.

Clinical Application Coordinators are targeted users for the KB Editor user with expertise defining Reminder Dialogs.

## Dependencies and Constraints

Section 508 compliance is required for CPRS usage in CPRS/AWARE DLL integration as well as other AWARE components. CPRS is a VA Class 1 product.

## Database Repository

### VistA VA FileMan

VISTA VA FileMan Alert Tracking audit/log file for tracked alerts, their status, and any follow-up actions taken. Data retention is parameter defined such as from 2 weeks up to 30 days for current or recent follow-up action activities. Storage and updating is on a periodic basis.

File VEFA AWARE ALERT CACHE 19008.2,^DIZ(19008.2, Field elements as follows:

* ALERT ID
* ALERT TEXT
* FACILITY/STATION
* SERVICE\_CLINIC
* ORDERING\_PROVIDER & ID
* ALL\_RECIPIENTS
* PATIENT ID
* ALERT\_TYPE
* ALERT CATEGORY
* ALERT\_DATE\_TIME
* FAT\_ORDER\_STATUS
* FAT\_PROVIDER
* UNACKED STATE
* ACKED\_RENEWED STATE
* ACKED\_DELETED STATE
* ACKED\_BY PROVIDER ID
* FOLLOWUP > 7DAYS
* FOLLOWUP
* Multiple Sub-file FAT\_ORDER/FOLLOW\_UPS\_TAKEN

VistA VA FileMan **VEFA Critical Alert Category** file stores the definition of a critical alert category.

File **VEFA Critical Alert Category** ,19008,^DIZ(19008, Field Elements as follows:

* DELETE ALERT
* DESCRIPTION
* NAME
* NOTIFICATION TYPE
* UPON FOLLOW-UP ACTION

VistA VA FileMan **VEFA Tracked Critical Alert Type** file stores the definition of critical alert as well as Reminder Dialogs used during AWARE follow-ups.

File 19007, **VEFA Tracked Critical Alert Type,**,^DIZ(19007,Field Elements as follows:

* NAME
* REMINDER DIALOG
* TIU TEMPLATE
* ALERT CATEGORY
* INITIATION DATE
* ACTIVE
* AWARE SOURCE SITE
* ALERT TYPE MNEMONIC
* DESCRIPTION
* ORDER DIALOG BOX TEXT

ORDERABLE ITEM DISPLAY GROUP

ORDERABLE ITEM NAME

ORDER DIALOG DISPLAY GROUP

ORDER DIALOG

ADDITIONAL FINDING

DESIRED ORDER DIALOG PACKAGE

COMMENT TEXT BOX

DESIRED ELEMENT NAME

ORDERABLE ITEM REQUIRED

ACTUAL ORDER DIALOG PACKAGE

PROGRESS NOTE TEXT

* FOLLOWUP DIALOG BOX TEXT

ORDERABLE ITEM DISPLAY GROUP

ORDERABLE ITEM NAME

ORDER DIALOG DISPLAY GROUP

ORDER DIALOG

ADDITIONAL FINIDING

DESIRED ORDER DIALOG PACKAGE

COMMENT TEXT BOX

DESIRED ELEMENT NAME

ORDERABLE ITEM REQUIRED

ACTUAL ORDER DIALOG PACKAGE

PROGRESS NOTE TEXT

* COMMENTS DIALOG BOX TEXT

ORDERABLE ITEM DISPLAY GROUP

ORDERABLE ITEM NAME

ORDER DIALOG DISPLAY GROUP

ORDER DIALOG

ADDITIONAL FINIDING

DESIRED ORDER DIALOG PACKAGE

COMMENT TEXT BOX

DESIRED ELEMENT NAME

ORDERABLE ITEM REQUIRED

ACTUAL ORDER DIALOG PACKAGE

PROGRESS NOTE TEXT

VistA VA FileMan **VEFA AWARE Alert Parameter** file to store AWARE Site parameters.

File **VEFA AWARE Alert Parameter** 19008.1 ,^DIZ(19008.1, Field Elements as follows:

* NAME
* LAST PROCESS START DATE TIME
* CACHE WINDOW
* LAST EXTRACT DATE TIME

VistA VA FileMan **ALERT** file to retrieve provider Alert data.

File **ALERT** ,8992 ,^XTV(8992, Field Elements as follows:

* RECIPIENT
* SURROGATE FOR ALERTS
* SURROGATE START DATE/TIME
* SURROGATE END DATE/TIME
* ALERT DATE/TIME (multiple
* ALERT DATE/TIME SUB-FIELD Choose from:

ALERT DATE/TIME

ALERT ID

MESSAGE TEXT

NEW ALERT FLAG

ACTION FLAG

RESERVED1

ENTRY POINT

AROUTINE NAME

RESERVED

CAN DELETE WITHOUT PROCESSING

DAYS FOR SURROGATE

DAYS FOR SUPERVISOR

DAYS FOR BACKUP REVIEWER

DATA FOR ALERT

COMMENT FOR DISPLAY

GUID FOR GUI

PATIENT

LONG INFO TEXT (word-processing)

* SURROGATE (multiple)

VistA VA FileMan **ALERT TRACKING** file to retrieve current Alert Information.

File **ALERT TRACKING** ,8992.1, ^XTV(8992.1, Field Elements as follows:

* NAME
* DATE CREATED
* PKG ID
* PATIENT
* GENERATED BY
* GENERATED WHILE QUEUED
* STATUS
* RETENTION DATE
* DISPLAY TEXT
* OPTION FOR PROCESSING
* ROUTINE TAG
* ROUTINE FOR PROCESSING
* DATA FOR PROCESSING
* GUID FOR GUI
* LONG INFO TEXT (word-processing)
* RECIPIENT (multiple)

RECIPIENT SUB-FIELD List? Y (Yes)

.01 RECIPIENT

.02 ALERT FIRST DISPLAYED

.03 FIRST SELECTED ALERT

.04 PROCESSED ALERT

.05 DELETED ON

.06 AUTO DELETED

.07 FORWARDED BY

.08 DATE/TIME FORWARDED

.09 DELETED BY USER

1 RECIPIENT TYPE (multiple)

2 FORWARDED DATE/TIME (multiple)

3 SURROGATE FOR (multiple)

### MS SQL Server Database

**MS SQL Server Database** with tables residing on a MS Windows Server machine. Tables for storing tracked alerts, their status, and any follow-up actions taken reside in the MS SQL Server Database. Retention occurs for up to 180 days. Storage occurs on a periodic basis for a facility with a daily update. There are various tables of alert tracking data.

#### MS SQL Application Data Tables

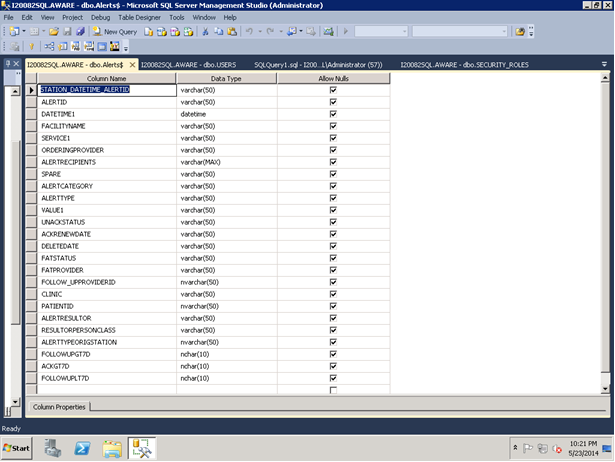


Figure - MS SQL Application Data Tables

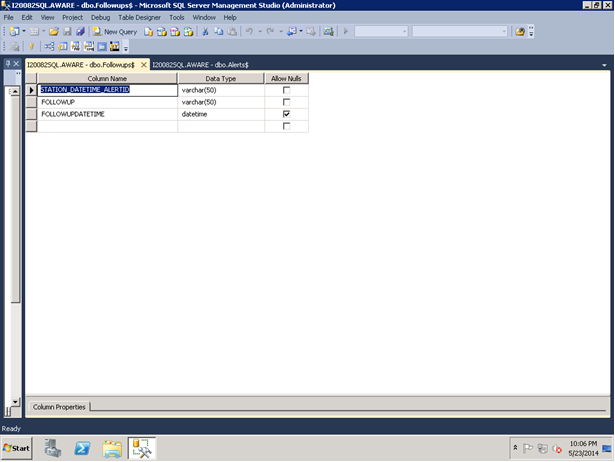


Figure - MS SQL Application Data Tables (Continued)

#### MS SQL Application Lookup Tables

* FacilityName$
  + FacilityName
* Service$
  + FacilityName
  + Service
* Clinic$
  + FacilityName
  + Service
  + Clinic
* OrderingProvider$
  + FacilityName
  + Service
  + Clinic
  + OrderingProvider
* AlertType$
  + FacilityName
  + Service
  + Clinic
  + OrderingProvider
  + AlertType
* OrderingProviderAllServices$
  + FacilityName
  + OrderingProvider
* OrderingProviderAllClinics$
  + FacilityName
  + Service
  + OrderingProvider

#### MS SQL QI Manager System Tables:

* APPLICATION\_VERSION
* AWARE\_VISTA\_GROUP\_MAPPINGS
* FACILITIES
* PROVIDERS
* SECURITY\_GROUPS
* SECURITY\_ITEMS
* SECURITY\_RIGHTS
* SECURITY\_ROLES
* USERS

## System Features

The system features, functional requirements, sub-requirements, etc. of the AWARE system are organized in an outline format in the AWARE Requirements Specifications Document (RSD).The CPRS/AWARE DLL integration is the core part of AWARE system.

### CPRS/AWARE Design Element Tables

Table –Get Alert Type RPC

| Routine Name | VEFAALRE | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| RTM(RSD) | RSD 2.6.3 AWARE/CPRS INTEGRATION  Alert tracker(2.6.3.1,2.6.3,2), FAT Prompt in CPRS from AWARE DLL (2.6.3.3.1) | | | | | | |
| Related Options |  | | | | | | |
| **Related Routines** | **Routines “Called By”** | | | | **Routines “Called”** | | |
|  | VEFAALR1 | | | | **Label Reference**  **LKPORD** | | |
| Data Dictionary & Global References | ^AUPNVHF(  ^AUPNVHF("C"  ^AUPNVSIT(  ^AUTTHF(  ^DIZ(19007  ^OR(100  ^ORD(101.43  ^PXRMD(801.41  ^TMP($J | | | | | | |
| Data Passing Details | Remote Procedure Calls (RPCs)  NAME: VEFALKPORD TAG: LKPORD  ROUTINE: VEFAALRE RETURN VALUE TYPE: SINGLE VALUE  AVAILABILITY: PUBLIC VERSION: 1  DESCRIPTION:  Do a lookup for specific critical alert TYPE, and associated date/time of unacknowledged alert, and check whether any ORDERS by user (DUZ)have been made from list of orders in a possible  e defined VEFA ALERT TRACKING entry's Reminder Dialog  ;(field), the dialog elements thereof containing the ORDER types (orderable  items) for such list to be compared with any actual orders  ; by this user (DUZ) made after the date/time of the actual critical alert.  ;First Verify also that type of alert passed is in VEFA ALERT TRACKING file  for "follow-up" purposes, and has defined REMINDER DIALOG.  INPUT PARAMETER: INPUT VALUE PARAMETER TYPE: LITERAL  MAXIMUM DATA LENGTH: 120 REQUIRED: YES | | | | | | |
| Related Integration Agreements |  | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |
| Input Attribute Name and Definition | Name: ;  Verify alert is tracked alert type, | | | | | | |
| Output Attribute Name and Definition | ; Output : None | | | | | | |

Table - Alert Tracker, Renew Tracked alert, Reporting Routines

| *Get Tracked Alerts RPC* Routine Name | VEFAALR1 | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| RTM(RSD) | RSD 2.6.3, AWARE/CPRS INTEGRATION 2.6.3.1,2.6.3.2 and 2.6.3.3.2 and 2.6.3.3.3 (renew alert)  Alert tracker, Re-new deleted alert in AWARE DLL via RPC routine  Assist with collection of data for viewing/subsequent reporting purposes | | | | | | |
| Related Options |  | | | | | | |
| **Related Routines** | **Routines “Called By”** | | | | **Routines “Called”** | | |
| VEFAALR9  VEFAALR6 | | | | External References  C^%DTC  NOW^%DTC  ^DIC  YN^DICN  $$CAT^VEFAALR1  $$SERVICE^VEFAALR1  $$STATUS^VEFAALR1  LKPORD^VEFAALR2  LKPORDCK^VEFAALRE  UPDATE^DIE | | |
| Data Dictionary & Global References | ^DD("DD"  ^DIC(4  ^DIC(49  ^DIZ(19007  ^DIZ(19008  ^DPT(  ^VA(200  ^XTV(8989.3  ^XTV(8992  ^XTV(8992.1 | | | | | | |
| Data Passing Details | Remote Procedure Calls (RPCs)  NAME: VEFA CRIT ALERT TRACKED TAG: LOOKUP  ROUTINE: VEFAALR1 RETURN VALUE TYPE: SINGLE VALUE  AVAILABILITY: PUBLIC VERSION: 1  INPUT PARAMETER: INPUT VALUE PARAMETER TYPE: LITERAL  MAXIMUM DATA LENGTH: 120 REQUIRED: YES  DESCRIPTION: LIST OF TRACKED ALERTS  Collection/assembly of Alert tracking Data for viewing/subsequent viewing/reporting  CODE:  MUMPS CODE: S X=$$CAT^VEFAALR1(X)  EXPLANATION: IS ALERT TYPE IN VEFA ALERT TRACKING CATEGORY/NOTIFICATION TYPE  MUMPS CODE: S X=$$DATEFMT^VEFAALR1(X)  EXPLANATION: RETURN DATE FORMATTED  MUMPS CODE: S X=$$FOLLOWU1^VEFAALR1(X)  EXPLANATION: FOLLOWUP ACTION (S) COMPLETED FOR SPECIFIC PASSED FAT ALERT (AS X)  MUMPS CODE: S X=$$ORDERCK1^VEFAALR1(X)  EXPLANATION: RETURN A CONCATENATED LIST (";") DELIMITED OF FAT ORDER/FOLLOW-UPS MADE FOR AN ALERT  MUMPS CODE: S X=$$STATUS^VEFAALR1(X)  EXPLANATION: RETURN UNACKNOWLEDGED STATUS^RENEW(PROCESSED) DATE^ACKNOWLEDGED OR DELETED DATE  MUMPS CODE: S X=$$SERVICE^VEFAALR1(X)  EXPLANATION: RETURN SYSTEM/CLINIC (SERVICE/SECTION FILE ENTRY) OF PASSED ALERT TRACKING ENTRY | | | | | | |
| Related Integration Agreements |  | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |
| Input Attribute Name and Definition | Name: Sorting arrays for collection/assembly of follow-up actions | | | | | | |
| Output Attribute Name and Definition | ; Output Renew Alert in Alert Tracking file | | | | | | |

Table - AWARE Patient Closed-out COM Object RPCs

| Routine Name | VEFAALR2 | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| RTM(RSD) | RSD 2.6.3 AWARE/CPRS INTEGRATION 2.6.3.1 and 2.6.3.2,  And 2.6.3.3.1  Alert tracker, FAT Prompt in AWARE DLL RPC routine | | | | | | |
| Related Options |  | | | | | | |
| **Related Routines** | **Routines “Called By”** | | | | **Routines “Called”** | | |
|  | VEFAALR1 | | | | External References | | |
| Data Dictionary & Global References | ^AUPNVHF(  ^AUPNVHF("C"  ^AUPNVSIT(  ^DIZ(19007  ^DIZ(19008  ^OR(100  ^PXRMD(801.41  ^TIU(8927  ^TMP("ORVEFACAT"  ^TMP("ORVEFAFOL"  ^TMP("ORVEFAORD"  ^TMP($J | | | | | | |
| Data Passing Details | Remote Procedure Calls (RPCs):  NAME: VEFA CRIT ALERT VALUES TAG: SPECALRT  ROUTINE: VEFAALR2 RETURN VALUE TYPE: SINGLE VALUE  AVAILABILITY: RESTRICTED INACTIVE: ACTIVE  WORD WRAP ON: TRUE VERSION: 1  DESCRIPTION:  RETURN REMINDER DIALOG AND TIU TEMPLATE FROM PASSED CRIT ALERT IN FILE 19007  INPUT PARAMETER: INPUT VALUE PARAMETER TYPE: LITERAL  MAXIMUM DATA LENGTH: 120 REQUIRED: YES  NAME: VEFA ALERT DOC FOLLOWUPS TAG: ALERTFOL  ROUTINE: VEFAALR2 RETURN VALUE TYPE: GLOBAL ARRAY  AVAILABILITY: RESTRICTED INACTIVE: ACTIVE  WORD WRAP ON: TRUE  DESCRIPTION:  RETURN LIST OF FOLLOWUPS AS DOCS OF FOLLOWUPS IN REMINDER DIALOG  INPUT PARAMETER: INPUT VALUE PARAMETER TYPE: LITERAL  MAXIMUM DATA LENGTH: 120 REQUIRED: YES  NAME: VEFA ALERT DOC ORDERS TAG: ALERTORD  ROUTINE: VEFAALR2 RETURN VALUE TYPE: GLOBAL ARRAY  AVAILABILITY: RESTRICTED INACTIVE: ACTIVE  WORD WRAP ON: TRUE  DESCRIPTION:  RETURN ORDERS IN ORDERS GROUP DIALOG AS DOCS OF REMINDER DIALOG  INPUT PARAMETER: INPUT VALUE PARAMETER TYPE: LITERAL  MAXIMUM DATA LENGTH: 120 REQUIRED: YES  DESCRIPTION:  NAME: VEFAALERTCAT TAG: ALERTCAT  ROUTINE: VEFAALR2 RETURN VALUE TYPE: GLOBAL ARRAY  AVAILABILITY: RESTRICTED INACTIVE: ACTIVE  WORD WRAP ON: TRUE  DESCRIPTION:  RETURN ARRAY OF CRITICAL ALERT CATEGORIES AND THEIT NOTIFICATION TYPES | | | | | | |
| Related Integration Agreements |  | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |
| Input Attribute Name and Definition | Name:   Pass Critical Alert to receive reminder dialog, TIU template, order and follow-ups, Date ranges for collection/assembly of follow-up action, | | | | | | |
| Output Attribute Name and Definition | ; Output : None | | | | | | |

Table - Continuation AWARE Patient Closed-out COM Object RPCs

| Routine Name | VEFAALR3 | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| RTM(RSD) | RSD 2.6.3 AWARE/CPRS INTEGRATION 2.6.3.1 and 2.6.3.2,  And 2.6.3.3.1  Alert tracker, Test Imaging Alert Types | | | | | | |
| Related Options |  | | | | | | |
| **Related Routines** | **Routines “Called By”** | | | | **Routines “Called”** | | |
|  | | | | External References  C^%DTC  NOW^%DTC  ^DIC  YN^DICN  UPDATE^DIE  $$CAT^VEFAALR1  $$SERVICE^VEFAALR1  $$STATUS^VEFAALR1 | | |
| Data Dictionary & Global References | ^DD("DD"  ^DPT(  ^XTV(8992.1 | | | | | | |
| Data Passing Details |  | | | | | | |
| Related Integration Agreements |  | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |
| Input Attribute Name and Definition | Name:   Pass Critical Alert to receive reminder dialog, TIU template, order and follow-ups, Date ranges for collection/assembly of follow-up action, | | | | | | |
| Output Attribute Name and Definition | ; Output : None | | | | | | |

#### Alert Cache Collector Design Element Table

Table - Alert Cache Collector Procedure& entry points to other procedures

| Routine Name | VEFAALR4 | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| RTM(RSD) | 2.6.2 | | | | | | |
| Related Options | VEFA AWARE ALERT CACHE BUILDER | | | | | | |
| **Related Routines** | **Routines “Called By”** | | | | **Routines “Called”** | | |
| TaskMan VEFA AWARE ALERT CACHE BUILDER  Entry points to RPC for Extracting Alert Cache data into SQL tables  Entry Point for Validator for KB Editor  VEFAALR9 | | | | C^%DTC  NOW^%DTC  EXTALERT^VEFAALR7  BASICCHK^VEFAALR8  COLLECT^VEFAALR9 | | |
| Data Dictionary & Global References | ^AUPNVHF(  ^AUPNVHF("C"  ^AUPNVSIT(  ^AUTTHF(  ^DIZ(19007  ^DIZ(19008  ^OR(100  ^ORD(100.9  ^ORD(101.43  ^PXRMD(801.  ^TMP($J | | | | | | |
| Data Passing Details | TAG: COLLECT TAG : BASICCHK TAG EXTALERT  INPUT: None INPUT: None INPUT: None  RETURN: None  OUTPUT: Store Critical alerts into Alert Cache | | | | | | |
| Related Integration Agreements |  | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |

Table - Continued AWARE Alert Cache Collector Procedure

| Routine Name | VEFAALR6 | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| RTM(RSD) | RSD 2.6..2.1.1 | | | | | | |
| Related Options |  | | | | | | |
| **Related Routines** | **Routines “Called By”** | | | | **Routines “Called”** | | |
| VEFAALR9 | | | | External References  $$SERVIC1^VEFAALR1  $$STATUS1^VEFAALR1  LKPORD^VEFAALRA  Label References  $$FOLLOWUP  $$ORDERCK  LKPORDCK | | |
| Data Dictionary & Global References | ^AUPNVHF(  ^AUPNVHF("C"  ^AUPNVSIT  ^AUTTHF  ^DIZ(19007  ^OR(100  ^ORD(101.43  ^PXRMD(801.41  ^TMP($J  ^VA(200  ^XTV(8992.1 | | | | | | |
| Data Passing Details | TAG: $$FOLLOWUP  INPUT: Alert Type ,Patient, Alert ID  RETURN: Followups made  TAG: $$ORDERCK  INPUT: Alert Type ,Patient, Alert ID  RETURN: Order checks made  TAG: LPKORDCK  INPUT: Alert Type, Patient , Alert ID  RETURN: Status order checks made | | | | | | |
| Related Integration Agreements |  | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |
| Input Attribute Name and Definition | Name:   Pass Critical Alert to receive reminder dialog, TIU template, order and follow-ups, Date ranges for collection/assembly of follow-up action, | | | | | | |
| Output Attribute Name and Definition | ; Output : None | | | | | | |

Table - Continuation Alert Cache Collector Procedure

| Routine Name | VEFAALR9 | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| RTM(RSD) | 2.6.2.1.1 | | | | | | |
| Related Options | VEFA AWARE ALERT CACHE BUILDER | | | | | | |
| **Related Routines** | **Routines “Called By”** | | | | **Routines “Called”** | | |
| TaskMan VEFA AWARE ALERT CACHE BUILDER | | | | C^%DTC  NOW^%DTC  UPDATE^DIE  RR^LR7OR1  $$CAT^VEFAALR1  $$SERVIC1^VEFAALR1  $$SERVICE^VEFAALR1  $$STATUS1^VEFAALR1  LKPORDCK^VEFAALR4  $$FOLLOWU1^VEFAALR6  $$ALERT2^VEFAALR9 $$ORDERCK2^VEFAALR9  $$FMTE^XLFDT | | |
| Data Dictionary & Global References | ^DIZ(19007  ^DIZ(19008  ^DIZ(19008.1  ^DPT(  ^OR(100  ^SC(  ^TMP("LRRR"  ^VA(200  ^XTV(8992  ^XTV(8992.1 | | | | | | |
| Data Passing Details | TAG: COLLECT  INPUT: None  RETURN: None  OUTPUT: Store Critical alerts into Alert Cache | | | | | | |
| Related Integration Agreements |  | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |

#### VistA to SQL Transport Design Element Table

Table - VistA to SQL Transport RPC

| Routine Name | VEFAALR4 | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| RTM(RSD) | 2.6.2.1.2 | | | | | | |
| Related Options |  | | | | | | |
| **Related Routines** | **Routines “Called By”** | | | | **Routines “Called”** | | |
| VistA to SQL Transporter (AWARE\_SQL\_Transporter.exe) | | | | C^%DTC  NOW^%DTC  EXTALERT^VEFAALR7 | | |
| Data Dictionary & Global References | ^TMP(“VEFAFOL”,$J)  ^DIZ(19007  ^DIZ(19008.1  ^DIZ(19008.2  ^XTV(8989.3) | | | | | | |
| Data Passing Details | Remote Procedure Calls (RPCs):  NAME: VEFA AWARE ALERT CACHE TAG: EXTALERT  ROUTINE: VEFAALR4 RETURN VALUE TYPE: GLOBAL ARRAY  AVAILABILITY: RESTRICTED INACTIVE: ACTIVE  WORD WRAP ON: TRUE  DESCRIPTION:  AWARE ALERT CACHE RETRIEVER FROM VISTA INTO SQL SERVER TABLES  RETURN PARAMETER DESCRIPTION:  ARRAY OF ALERTS FOR EXTRACTED TIME PERIOD | | | | | | |
| Related Integration Agreements |  | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |

Table - Continued VistA to SQL Transport RPC

| Routine Name | VEFAALR7 | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| RTM(RSD) | 2.6.2.1.2 | | | | | | |
| Related Options |  | | | | | | |
| **Related Routines** | **Routines “Called By”** | | | | **Routines “Called”** | | |
| VistA to SQL Transporter (AWARE\_SQL\_Transporter.exe)  VEFAALR4 | | | | C^%DTC  NOW^%DTC  ^DIK  $$FMTE^XLFDT  $$FMTHL7^XLFDT  Label References  DELCACHE | | |
| Data Dictionary & Global References | ^DIC(4  ^DIZ(19007  ^DIZ(19008.1  ^DIZ(19008.2  ^TMP("VEFAFOL"  ^TMP("VEFAKB  ^XTV(8989.3 | | | | | | |
| Data Passing Details | RPC NAME: VEFA AWARE ALERT CACHE TAG: EXTALERT  ROUTINE: VEFAALR4 RETURN VALUE TYPE: GLOBAL ARRAY  AVAILABILITY: RESTRICTED INACTIVE: ACTIVE  WORD WRAP ON: TRUE  DESCRIPTION:  AWARE ALERT CACHE RETRIEVER FROM VISTA INTO SQL SERVER TABLES  RETURN PARAMETER DESCRIPTION:  ARRAY OF ALERTS FOR EXTRACTED TIME PERIOD  ALSO TRIMS VEFA AWARE DATA CACHE file BASED ON AWARE VEFA PARAMETER FILE “WINNDOWS” size. | | | | | | |
| Related Integration Agreements |  | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |
|  |  |  | |  | |  |  |

#### Security Keys for Alert Cache Viewer/QI Tool and KB Editor Design Element Table

Table - Security Key Routines

| Routine Name | VEFAALR5 | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| RTM(RSD) | 2.6.2.2.2 & 2.6.1 | | | | | | |
| Related Options |  | | | | | | |
| **Related Routines** | **Routines “Called By”** | | | | **Routines “Called”** | | |
| Alert Cache Viewer and KB Editor Cache Server Pages. QI Tool Cache Web Service | | | | $$DEA^XUSER | | |
| Data Dictionary & Global References | ^XUSEC | | | | | | |
| Data Passing Details | TAG: USRKEY  DESCRIPTION: Get security level to view Alert Cache using Alert Cache Viewer  INPUT: DUZ  OUTPUT Reference: LEVEL (User Key Security Level)  0 : User has no right to view alert cache  1 : User has right to view his/her own alert cache  2 : User has right to view all alert cache  TAG: HASKBKEY  DESCRIPTION: Get flag to indicate whether or not user has access to KB Editor  INPUT: DUZ  RETURN:  0: User has no access to KB Editor,  1: User is permitted to use KB Editor  TAG: ISPROG  RETURN: flag to indicate whether or not user has programmer key | | | | | | |
|  |  | | | | | | |
| Related Integration Agreements | #10076 (^XUSEC - Supported)  #2343 (DEA^XUSER – Supported) | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |

#### Alert Cache Viewer Design Element Table

Table - Collection of Alert Cache Viewer Procedures

| Routine Name | VEFAALC1 | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| RTM(RSD) | 2.6.2.2.2 | | | | | | |
| Related Options |  | | | | | | |
| **Related Routines** | **Routines “Called By”** | | | | **Routines “Called”** | | |
| Alert Cache Viewer CSP Pages | | | | USRKEY^VEFAALR5  GETS^DIQ  LIST^DIC  $$LOW^XLFSTR  $$UP^XLFSTR  $$QUOTE^VEFACSP1 | | |
| Data Dictionary & Global References | ^DIZ(19008.2 | | | | | | |
| Data Passing Details | **TAG**: GETALCH - Return filtered Alert Cache records  **INPUT**  USERNAME : User Name - if user has key to see his/her own alert cache, filter result array with this user name  DUZ : This User DUZ will be used to get the security key  PROV : Provider - if not All, filter result array with this provider  SERV : Service - if not All, filter result array with this service  PAT : Patient - if not All, filter result array with this patient  FOL7 : Follow-up > 7 Days - if not All, filter result array with true/false  **OUTPUT**  VEFARSLT : Result Array (global reference) of Alert Cache based on criteria  SELPROV : Result Array of providers  SELSERV : Result Array of services  SELPAT : Result Array of patients  FACILITY : Facility  **TAG:** GETDATA – Return a single Alert Cache record for IENS  **INPUT**  IENS: Internal Alert Cache Record ID  **OUTPUT**  DATA : Alert Cache Record  **TAG**: GETFOL – Return Alert Cache Follow-up records  **INPUT**  IENS: Internal Alert Cache Record ID  **OUTPUT**  VEFARSLT: Alert Cache Follow-up Records for IENS | | | | | | |
| Related Integration Agreements | 2056, 2051, 10104 | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |

#### KB Editor Design Element Table

Table - Get All Alert Categories Procedure

| Routine Name | VEFAKB01 | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| RTM(RSD) | 2.6.1 | | | | | | |
| Related Options |  | | | | | | |
| **Related Routines** | **Routines “Called By”** | | | | **Routines “Called”** | | |
| KB Editor CSP Pages | | | | LIST^DIC  $$QUOTE^VEFACSP1  $$UP^XLFSTR | | |
| Data Dictionary & Global References | ^DIZ(19008 | | | | | | |
| Data Passing Details | **TAG**: GETALCAT – Return all alert categories  **INPUT**  NONE  **OUTPUT**  VEFARSLT : Reference Array containing all alert categories | | | | | | |
| Related Integration Agreements | 2051, 10104 | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |

Table - Save Alert Category Procedure

| Routine Name | VEFAALC2 | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| RTM(RSD) | 2.6.1 | | | | | | |
| Related Options |  | | | | | | |
| **Related Routines** | **Routines “Called By”** | | | | **Routines “Called”** | | |
| KB Editor CSP Pages | | | | FILE^DIE  UPDATE^DIE | | |
| Data Dictionary & Global References | ^DIZ(19008 | | | | | | |
| Data Passing Details | **TAG**: SAVE – Save Alert Category Record  **INPUT**  TODO : add or edit  IEN : ID of Alert Category Record to save  RECORD : Alert Category Record to save  **OUTPUT**  ERR : Error Message  RETURN  1 - success  0 – failed | | | | | | |
| Related Integration Agreements | 2053 | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |

Table - Get All Alert Types Procedure

| Routine Name | VEFAKB01 | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| RTM(RSD) | 2.6.1 | | | | | | |
| Related Options |  | | | | | | |
| **Related Routines** | **Routines “Called By”** | | | | **Routines “Called”** | | |
| KB Editor CSP Pages | | | | LIST^DIC  $$QUOTE^VEFACSP1  $$UP^XLFSTR | | |
| Data Dictionary & Global References | ^DIZ(19007 | | | | | | |
| Data Passing Details | **TAG**: GETALTYP – Return all alert types  **INPUT**  NONE  **OUTPUT**  VEFARSLT : Reference Array containing all alert types | | | | | | |
| Related Integration Agreements | 2051, 10104 | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |

Table - Get Alert Type - Order/Follow-up/Comments Dialog Text Procedure

| Routine Name | VEFAKB01 | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| RTM(RSD) | 2.6.1 | | | | | | |
| Related Options |  | | | | | | |
| **Related Routines** | **Routines “Called By”** | | | | **Routines “Called”** | | |
| KB Editor CSP Pages | | | | GETS^DIQ  $$QUOTE^VEFACSP1 | | |
| Data Dictionary & Global References | ^DIZ(19007 | | | | | | |
| Data Passing Details | **TAG**: GETREC – Return AWARE order/follow-up/comments Dialog Text  **INPUT**  FILE : Sub File Number for AWARE Order/Follow-up/Comments dialog text  IENS : Record ID for the Sub File  **OUTPUT**  VEFARSLT : Reference Array containing dialog text record  PRGNOTES : Progress Note by reference | | | | | | |
| Related Integration Agreements | 2056 | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |

Table - Get Alert Type Name Function

| Routine Name | VEFAKB01 | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| RTM(RSD) | 2.6.1 | | | | | | |
| Related Options |  | | | | | | |
| **Related Routines** | **Routines “Called By”** | | | | **Routines “Called”** | | |
| KB Editor CSP Pages | | | | GET1^DIQ | | |
| Data Dictionary & Global References | ^DIZ(19007 | | | | | | |
| Data Passing Details | **TAG**: GETNAME – Given IEN, return Alert Type Name  **INPUT**  IEN : Record ID of the Alert Type  **RETURN**  Alert Type name | | | | | | |
| Related Integration Agreements | 2056 | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |

Table - Get Alert Type – Reminder Dialog Record Procedure

| Routine Name | VEFAKB02 | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| RTM(RSD) | 2.6.1 | | | | | | |
| Related Options |  | | | | | | |
| **Related Routines** | **Routines “Called By”** | | | | **Routines “Called”** | | |
| KB Editor CSP Pages | | | | GETS^DIQ  $$FMTE^XLFDT  $$QUOTE^VEFACSP1 | | |
| Data Dictionary & Global References | ^DIZ(19007 | | | | | | |
| Data Passing Details | **TAG**: GETATFRM – Get alert Type record  **INPUT**  IENS : Record ID of the Alert Type  **OUTPUT**  VEFARSLT : Reference Array by field containing the field value  DTINIT : Initiation Date  ACTIVE : Alert Type active (true/false) indicator  SITE : Site Number where the Alert Type is created | | | | | | |
| Related Integration Agreements | 2056, 10103 | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |

Table - Get All dialogs (Order/Follow-up/Comment) for an Alert type Procedure

| Routine Name | VEFAKB02 | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| RTM(RSD) | 2.6.1 | | | | | | |
| Related Options |  | | | | | | |
| **Related Routines** | **Routines “Called By”** | | | | **Routines “Called”** | | |
| KB Editor CSP Pages | | | | LIST^DIC  $$QUOTE^VEFACSP1 | | |
| Data Dictionary & Global References | ^DIZ(19007 | | | | | | |
| Data Passing Details | **TAG**: GETATLST – Get all dialogs (order/follow-up/comments) for an alert type  **INPUT**  FILE : Alert Type Sub file (Order/follow-up/comments)  ; IENS : Alert Type Record ID  **OUTPUT**  VEFARSLT : Reference Array containing all dialog text records | | | | | | |
| Related Integration Agreements | 2051 | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |

Table - Save Alert Type – Reminder Dialog Procedure

| Routine Name | VEFAALC2 | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| RTM(RSD) | 2.6.1 | | | | | | |
| Related Options |  | | | | | | |
| **Related Routines** | **Routines “Called By”** | | | | **Routines “Called”** | | |
| KB Editor CSP Pages | | | | FILE^DIE  UPDATE^DIE | | |
| Data Dictionary & Global References | ^DIZ(19007 | | | | | | |
| Data Passing Details | **TAG**: SAVEATFRM – Save Alert Type’s Reminder Dialog section  **INPUT**  TODO : add or edit  FILE : File Number  IEN : ID of Alert Type to save  RECORD : Alert Type Record to save  **OUTPUT**  MSG : Error Message  RETURN  1 - success  0 – failed | | | | | | |
| Related Integration Agreements | 2053 | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |

Table - Save Order/Follow-up/Comments Dialog Procedure

| Routine Name | VEFAALC2 | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| RTM(RSD) | 2.6.1 | | | | | | |
| Related Options |  | | | | | | |
| **Related Routines** | **Routines “Called By”** | | | | **Routines “Called”** | | |
| KB Editor CSP Pages | | | | FILE^DIE  UPDATE^DIE | | |
| Data Dictionary & Global References | ^DIZ(19007 | | | | | | |
| Data Passing Details | **TAG**: SAVEDLG – Save Order/Follow-up/Comments Dialog  **INPUT**  TODO : add or edit  FILE : Sub File Number  RECORD : Alert Type Record to save  TYPEIEN : Record ID of Alert Type to save  FILEIDX : Subscript Index representing the Sub File  IEN : Record ID of the dialog text  **OUTPUT**  MSG : Error Message  RETURN  1 - success  0 – failed | | | | | | |
| Related Integration Agreements | 2053 | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |

Table - Deactivate Alert Type Procedure

| Routine Name | VEFAALC2 | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| RTM(RSD) | 2.6.1 | | | | | | |
| Related Options |  | | | | | | |
| **Related Routines** | **Routines “Called By”** | | | | **Routines “Called”** | | |
| KB Editor CSP Pages | | | | FILE^DIE | | |
| Data Dictionary & Global References | ^DIZ(19007 | | | | | | |
| Data Passing Details | **TAG**: INACTV – Deactivate Alert Type  **INPUT**  NAME : Alert Type Name  **OUTPUT**  ACTIVE field of Alert Type set to false | | | | | | |
| Related Integration Agreements | 2053 | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |

Table - Update Initiation Date Procedure

| Routine Name | VEFAALC2 | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| RTM(RSD) | 2.6.1 | | | | | | |
| Related Options |  | | | | | | |
| **Related Routines** | **Routines “Called By”** | | | | **Routines “Called”** | | |
| KB Editor CSP Pages | | | | FILE^DIE  DT^XLFDT | | |
| Data Dictionary & Global References | ^DIZ(19007 | | | | | | |
| Data Passing Details | **TAG**: UPINITDT – Update Alert Type’s Initiation Date field with the current date  **INPUT**  NAME : Alert Type Name  **OUTPUT**  Initiation Date field of Alert Type set to the current date | | | | | | |
| Related Integration Agreements | 2053, 10103 | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |

Table - Delete Alert Type Procedure

| Routine Name | VEFAALC2 | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| RTM(RSD) | 2.6.1 | | | | | | |
| Related Options |  | | | | | | |
| **Related Routines** | **Routines “Called By”** | | | | **Routines “Called”** | | |
| KB Editor CSP Pages | | | | FILE^DIE | | |
| Data Dictionary & Global References | ^DIZ(19007 | | | | | | |
| Data Passing Details | **TAG**: DELALTYP – Delete Alert Type  **INPUT**  TYPIEN : Alert Type Record ID (IEN)  **OUTPUT**  Alert Type with TYPIEN removed from Alert Type file | | | | | | |
| Related Integration Agreements | 2053 | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |

Table - Delete Dialog Box text (Order/Follow-up/Comments) Procedure

| Routine Name | VEFAALC2 | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| RTM(RSD) | 2.6.1 | | | | | | |
| Related Options |  | | | | | | |
| **Related Routines** | **Routines “Called By”** | | | | **Routines “Called”** | | |
| KB Editor CSP Pages | | | | FILE^DIE | | |
| Data Dictionary & Global References | ^DIZ(19007 | | | | | | |
| Data Passing Details | **TAG**: DELORDLG – Delete Dialog Box Text (Order/follow-up/Comments)  **INPUT**  FILE : Alert Type Sub File for (Order/Follow-up/Comments) dialog  TYPIEN : Alert Type Record ID (IEN)  ORDIEN : Alert Type Dialog Record ID  **OUTPUT**  Alert Type Dialog Record with IENS (TYPIEN, ordien) removed from Alert Type file | | | | | | |
| Related Integration Agreements | 2053 | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |

Table - Alert Type Validator

| Routine Name | VEFAALR8 | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| RTM(RSD) | 2.6.1 | | | | | | |
| Related Options |  | | | | | | |
| **Related Routines** | **Routines “Called By”** | | | | **Routines “Called”** | | |
| CSP KB Editor Critical Tracked Alert Type Validator  VEFAALR4 | | | | External References  FILE^DIE  $$GET1^DIQ  GETPROG^VEFAALRA  WRITPROG^VEFAALRA | | |
| Data Dictionary & Global References | ^AUTTHF(  ^DIC(9.4  ^DIZ(19007  ^ORD(101.41  ^ORD(101.43  ^PXRMD(801.41  ^TMP($J | | | | | | |
| Data Passing Details | TAG: BASICCHK  INPUT: None  RETURN: Status and Next element to Process  OUTPUT: Update Orders Tab element values for KB Critical Treacked Alert Type file from Reminder Dialog Data | | | | | | |
| Related Integration Agreements |  | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |

Table - Continued Alert Type Validator

| Routine Name | VEFAALRA | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| RTM(RSD) | 2.6.1 | | | | | | |
| Related Options |  | | | | | | |
| **Related Routines** | **Routines “Called By”** | | | | **Routines “Called”** | | |
| CSP KB Editor Critical Tracked Alert Type Validator  VEFAALR8  VEFAALR6 | | | | Label References  GETPROG  WRITPROG  LKPORD | | |
| Data Dictionary & Global References | ^AUPNVHF(  ^AUPNVHF("C"  ^AUPNVSIT(  ^DIZ(19007  ^OR(100  ^PXRMD(801.41  ^TMP($J | | | | | | |
| Data Passing Details | TAG: GETPROG, WRITPROG  INPUT: None  RETURN: Retrieve and Write Reminder Dialog progress note data into KB VEFA Critical tracked Alert Type dialog elements as part of Validator for KB database  OUTPUT:  TAG: LKPORD  INPUT: None  RETURN: Retrieve orders for analysis for reporting Cache data  OUTPUT: | | | | | | |
| Related Integration Agreements |  | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |

#### Shared API for KB Editor and Alert Cache Viewer Design Element Table

Table - Collection of procedures/functions used by KB Editor/Alert Cache Viewer

| Routine Name | VEFACSP1 | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| RTM(RSD) | 2.6.1 & 2.6.2.2.2 | | | | | | |
| Related Options |  | | | | | | |
| **Related Routines** | **Routines “Called By”** | | | | **Routines “Called”** | | |
| KB Editor CSP Pages  Alert Cache Viewer CSP Pages | | | | GETENV^%ZOSV  DT^XLFDT  $$PROD^XUPROD  NAME^XUSER  $$ENCRYPT^XUSRB1  CHECKAV^XUSRB  VALIDAV^XUSRB | | |
| Data Dictionary & Global References | ^XTV(8989.3,1,”XUS”) | | | | | | |
| Data Passing Details | **TAG**: GETENV – Get VistA Environment Variables  **INPUT**  NONE  **OUTPUT**  SERVER : Local VistA Server  VOLBOX : Volume BOX pair  UCI : VistA Instance  PROD : Production Account Indicator  **TAG:** GETDUZ – Validate Access/Verify codes and return DUZ  **INPUT**  ACCESS : User Access Code  VERIFY : User Verify Code  **OUTPUT**  MSG : Error Message  **RETURN**  DUZ (User ID)  **TAG:** GETNAMES – Return all name values start with value in SEARCH in a FILE  **INPUT**  FILE : File Number  SEARCH : Start With value  **OUTPUT**  VEFARSLT : Reference Array containing the filtered names for the FILE  **TAG:** QUOTE – Replace quotes and linefeed with equivalent HTML in given string and return the replaced string  **INPUT**  STR : String with potential quotes and linefeed  **RETURN**  The transformed String with html chars  **TAG**: GETSRCH – return name values for a file start with a given string and filter the records with given FileMan screen value  **INPUT**  FILE : File Number  SEARCH : Start with value  FILTER ; FILEMAN screen value  **OUTPUT**  VEFARSLT : Reference Array containing the “filtered” name values  **TAG**: GETNAME – Return User Name  **INPUT**  DUZ : User ID  **OUTPUT**  User Name | | | | | | |
| Related Integration Agreements | 2240, 2343, 2882, 4054, 4440, 4762, 10097, 10103, 10104 | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |

#### Templates

Not applicable.

#### Bulletins

Not applicable.

#### Data Entries Affected by the Design

Not applicable.

#### Unique Record(s)

Table - Unique Record(s)

| **Field Name(s)** | **Current Value** | **New Value** |
| --- | --- | --- |
| ORWCOM VEFA PAT CHART CLOSEOUT parameter value as defined in the PARAMETER DEFINITION file. Add new record | None | TRUE |

### Mail groups

Not applicable.

### Security Keys

Table - Security Keys

| **Security** | **Description** | | | |
| --- | --- | --- | --- | --- |
| **Key Names** | VEFA AWARE ADD/EDIT KB RULE : With this key, user is permitted to add/edit KB rule using KB Editor  VEFA AWARE ALL ALERT CACHE – With this key, user is permitted to view all alert cache using Alert Cache Viewer  VEFA AWARE OWN ALERT CACHE – With this key, user is permitted to view OWN alert cache using Alert Cache Viewer (\*\*Physician providers do not need this security key in the production account) | | | |
| **Enhancement Category** | New | Modify | Delete | No Change |
| **Applications** | Knowledge-Based Editor  Alert Cache Viewer/QI Tool | | | |
| **Definition** | AWARE Security Keys for Web Applications | | | |

### Options

Table - Options

| **Options** | **Activities** | | | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Option Name** | Select OPTION NAME: VEFAALRE VEFA ALRE CONTEXT  ANOTHER ONE:  STANDARD CAPTIONED OUTPUT? Yes// (Yes)  Include COMPUTED fields: (N/Y/R/B): NO// - No record number (IEN), no Computed  Fields  NAME: VEFAALRE MENU TEXT: VEFA ALRE CONTEXT  TYPE: Broker (Client/Server) CREATOR: PROGRAMMER,ONE  TIMESTAMP OF PRIMARY MENU: 62583,83322  RPC: ORWORB FASTUSER  RPC: VEFALKPORD  RPC: VEFA CRIT ALERT TRACKED  RPC: VEFAALERTCAT  RPC: VEFA CRIT ALERT VALUES  RPC: VEFA ALERT DOC ORDERS  RPC: VEFA ALERT DOC FOLLOWUPS  UPPERCASE MENU TEXT: VEFA ALRE CONTEXT  AME: VEFA AWARE ALERT CACHE BUILDER MENU TEXT: AWARE ALERT CACHE BUILDER  TYPE: run routine CREATOR: INNOVATIONS,VHA  ROUTINE: COLLECT^VEFAALR4  UPPERCASE MENU TEXT: AWARE ALERT CACHE BUILDER  AME: VEFA AWARE GET ALERT CACHE  MENU TEXT: Context for Alert Cache Retrieve  TYPE: Broker (Client/Server) CREATOR: INNOVATIONS,VHA  TIMESTAMP OF PRIMARY MENU: 63172,62834  RPC: VEFA AWARE ALERT CACHE  UPPERCASE MENU TEXT: CONTEXT FOR ALERT CACHE RETRIE | | | | | | | | | | |
| **Enhancement Category** | New | Modify | | | | Delete | | | No Change | | |
| **Associated Menu Options that will invoke this reference** | Added to CPRS context option OR CPRS GUI CHART | | | | | | | | | | |
| **Data Passing** | Input | | Output | | Both | | | Global Reference | | | Local Reference |
| **Menu Text Description** |  | | | | | | | | | | |
| **Option Type** | Edit | | | Print | | | Menu | | | Inquire | |
|  | Action | | | Run Routine | | | Other | | | Broker (Client/Server) type for Context for DLL usage | |
| **Associated Routine** |  | | | | | | | | | | |
| **Option Definition** | Context for GUI usage of AWARE DLL | | | | | | | | | | |

| **Current Entry Action Logic** |
| --- |
|  |

| **Modified Entry Action Logic (Changes are in bold)** |
| --- |
|  |

| **Current Exit Action Logic** |
| --- |
|  |

| **Modified Exit Action Logic (Changes are in bold)** |
| --- |
|  |

### Protocols

Not applicable.

### Remote Procedure Call (RPC)

OUTPUT FROM WHAT FILE: OPTION// REMOTE PROCEDURE (3025 entries)

Select REMOTE PROCEDURE NAME: ORWCOM VEFA PT CLSCHART

ANOTHER ONE:

STANDARD CAPTIONED OUTPUT? Yes// (Yes)

Include COMPUTED fields: (N/Y/R/B): NO// - No record number (IEN), no Computed

Fields

NAME: ORWCOM VEFA PT CLSCHART TAG: PTCLS

ROUTINE: ORWCOM RETURN VALUE TYPE: SINGLE VALUE

AVAILABILITY: RESTRICTED WORD WRAP ON: TRUE

DESCRIPTION:

Returns COM Object entries from different parameters.

RETURN PARAMETER DESCRIPTION:

Zero Node from File 101.15

Note \*\*ORWCOM VEFA PT CLSCHART RPC is added to CPRS context option ORWCOM VEFA PT CLSCHART

REMOTE PROCEDURE LIST NOV 3,2013 19:18 PAGE 1

--------------------------------------------------------------------------------

NAME: VEFA ALERT DOC FOLLOWUPS TAG: ALERTFOL

ROUTINE: VEFAALR2 RETURN VALUE TYPE: GLOBAL ARRAY

AVAILABILITY: RESTRICTED INACTIVE: ACTIVE

WORD WRAP ON: TRUE

DESCRIPTION:

RETURN LIST OF FOLLOWUPS AS DOCS OF FOLLOWUPS IN REMINDER DIALOG

INPUT PARAMETER: INPUT VALUE PARAMETER TYPE: LITERAL

MAXIMUM DATA LENGTH: 120 REQUIRED: YES

NAME: VEFA ALERT DOC ORDERS TAG: ALERTORD

ROUTINE: VEFAALR2 RETURN VALUE TYPE: GLOBAL ARRAY

AVAILABILITY: RESTRICTED INACTIVE: ACTIVE

WORD WRAP ON: TRUE

DESCRIPTION:

RETURN ORDERS IN ORDERS GROUP DIALOG AS DOCS OF REMINDER DIALOG

INPUT PARAMETER: INPUT VALUE PARAMETER TYPE: LITERAL

MAXIMUM DATA LENGTH: 120 REQUIRED: YES

DESCRIPTION:

REMOTE PROCEDURE LIST NOV 3,2013 19:18 PAGE 2

--------------------------------------------------------------------------------

RECORD NUMBER OF VEFA CRIT ALERT TRACKING FILE "ALERT TYPE"

NAME: VEFA CRIT ALERT TRACKED TAG: LOOKUP

ROUTINE: VEFAALR1 RETURN VALUE TYPE: SINGLE VALUE

AVAILABILITY: PUBLIC VERSION: 1

INPUT PARAMETER: INPUT VALUE PARAMETER TYPE: LITERAL

MAXIMUM DATA LENGTH: 120 REQUIRED: YES

NAME: VEFA CRIT ALERT VALUES TAG: SPECALRT

ROUTINE: VEFAALR2 RETURN VALUE TYPE: SINGLE VALUE

AVAILABILITY: RESTRICTED INACTIVE: ACTIVE

WORD WRAP ON: TRUE VERSION: 1

DESCRIPTION:

RETURN REMINDER DIALOG AND TIU TEMPLATE FROM PASSED CRIT ALERT IN FILE 19007

INPUT PARAMETER: INPUT VALUE PARAMETER TYPE: LITERAL

MAXIMUM DATA LENGTH: 120 REQUIRED: YES

NAME: VEFAALERTCAT TAG: ALERTCAT

ROUTINE: VEFAALR2 RETURN VALUE TYPE: GLOBAL ARRAY

AVAILABILITY: RESTRICTED INACTIVE: ACTIVE

REMOTE PROCEDURE LIST NOV 03, 2013@19:18 PAGE 3

--------------------------------------------------------------------------------

WORD WRAP ON: TRUE

DESCRIPTION:

RETURN ARRAY OF CRITICAL ALERT CATEGORIES AND THEIT NOTIFICATION TYPES

NAME: VEFALKPORD TAG: LKPORD

ROUTINE: VEFAALRE RETURN VALUE TYPE: SINGLE VALUE

AVAILABILITY: PUBLIC VERSION: 1

DESCRIPTION:

Do a lookup for specific critical alert TYPE, and associated date/time of a rea

l unacknowledged alert, and check whether

;any ORDERS by user (DUZ)have been made from list of orders in a possible

e defined VEFA ALERT TRACKING entry's Reminder Dialog

;(field), the dialog elements thereof containing the ORDER types (orderable

iems) for such list to be compared with any actual orders

;by this user (DUZ) made after the date/time of the actual critical alert.

;First Verify also that type of alert passed is in VEFA ALERT TRACKING file

for "followup" purposes, and has defined REMINDER DIALOG.

INPUT PARAMETER: INPUT VALUE PARAMETER TYPE: LITERAL

MAXIMUM DATA LENGTH: 120 REQUIRED: YES

AME: VEFA AWARE ALERT CACHE TAG: EXTALERT

ROUTINE: VEFAALR4 RETURN VALUE TYPE: GLOBAL ARRAY

AVAILABILITY: RESTRICTED INACTIVE: ACTIVE

WORD WRAP ON: TRUE

DESCRIPTION:

AWARE ALERT CACHE RETRIEVER FROM VISTA INTO SQL SERVER TABLES

RETURN PARAMETER DESCRIPTION:

ARRAY OF ALERTS FOR EXTRACTED TIME PERIOD

### Constants Defined Interface

Not applicable.

### Variables Defined Interface

Not applicable.

### Types Defined Interface

Table - Types Defined Interface

| ***Name*** | ***Description*** |
| --- | --- |
| ORWCOM VEFA PAT CHART CLOSEOUT | Parameter Definition type for allowing parameterization per user, service, system to allow new Patient Closeout Com object entry point in CPRS |

### CPRS/AWARE DLL Graphical User Interface (GUI)

The AWARE DLL is written in Delphi.

Table - Delphi Units and Forms for AWARE DLL

| Unit Name | Description |
| --- | --- |
| write4cprsext\_TLB.pas | COM object interface library |
| CPRSChart\_TLB.pas | COM object CPRS interface |
| writecomobject.pas | Main DLL Alert Tracker module |
| FAT4\_textok.dfm | FAT prompt data form |
| FAT4\_textok.pas | FAT prompt module |
| AlertIntercept1.pas | Intercept tracked alert module |
| FATHelpScreen.dfm | FAT help screen form |
| FATHelpScreen.pas | FAT help screen module |
| FAT3\_textok.dfm | FAT prompt data form |
| FAT3\_textok.dfm | FAT prompt module |
| AlertIntercept.pas | Intercept tracked alert module |

CPRS is written in Delphi (Delphi code). Customization required for CPRS/AWARE integration.

Table - Changes in CPRS Delphi Units

|  |  |
| --- | --- |
| **Unit Name** | **Description** |
| fEncnt.pas | Provider & Location for Current Activities |
| fNotes.pas | Progress Note Page |
| fptSel.pas | Patient Selection |
| fptSelDemog.pas | Patient Demographics |
| rCore.pas | Collection of Core RPC record types and common RPC related APIs. |
| rEventHooks.pas | Collection of COM related APIs. |
| uCore.pas | Collection of common core classes and Notification APIs |
| uEventHooks.pas | Collection of COM related classes and APIs. |

### GUI Classes

Table - New GUI Classes in AWARE DLL

| **GUI Classes** | **Instructions** |
| --- | --- |
| Twrite4comobject | Twrite4comobject = class(TAutoObject, Iwrite4comobject, ICPRSExtension)  protected  function Execute(const CPRSBroker: ICPRSBroker;  const CPRSState: ICPRSState; const Param1, Param2,  Param3: WideString; var Data1, Data2: WideString): WordBool;  safecall;  procedure Free ();  procedure AppMessage(var Msg: tagMSG; var Handled: Boolean);  { Protected declarations }  end; |
| **Derived From Class** |  |
| **Purpose** |  |

| **GUI Classes** | **Instructions** |
| --- | --- |
| TFATFormS | TFATFormS = class(TForm) AlertLabel: TLabel;  ButtonLeave: TButton; Label2: TLabel;  Label3: TLabel; PatientName: TLabel;  Label4: TLabel; ReminderInstructions: TLabel;  Button1: TButton; Label9: TLabel;  Button2: TButton; NoteTitle: TLabel;  Button3: TButton;  procedure OnShow(Sender: TObject);  procedure ButtonLeaveClick(Sender: TObject);  procedure Button1Click(Sender: TObject);  procedure Button2Click(Sender: TObject);  procedure Button3Click(Sender: TObject); |
| **Derived From Class** |  |
| **Purpose** |  |

### Current Form

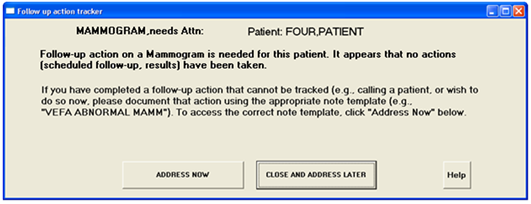


Figure - Prompting Screen

### Modified Form

There is no modified form.

### Components on Form

Name of Form (screen) above is the FAT Prompt screen at time of a Patient closeout (when a new patient selection is made).

Table - Components on Form

| **Name** | **Type** | **Description** |
| --- | --- | --- |
| Type of Alert | Field | Type of tracked alert |
| Patient | Text box | Patient name |
| Prompt and prompt Note template | Dynamic Text box | Description of opportunity to do a follow-up action |
| Address Now | Button | Re-direct CRPS flow into tracked alert associated Reminder Dialog |
| Close and Address Later | Button | **Continue with CPRS normal patient selection** |
| Help | Screen (Form) | Help instructions |

### Events

Table - Events

| **Name** | **Type** | **Description** |
| --- | --- | --- |
| AWARE DLL call | COM interface (return parameter) | Return dynamic dialog to CPRS |

### Methods

Table - Methods

| **Name** | **Type** | **Description** |
| --- | --- | --- |
| ICPRSBROKER.Execute | AWARE DLL COM object interface call | **CPRS/AWARE DLL communication interface at time of patient closeout** |

### Special References

Table - Special References

| Special Reference Name | Type | Description |
| --- | --- | --- |
| N/A |  |  |

### Class Events

Table - Class Events

| Name | Type | Description |
| --- | --- | --- |
| N/A |  |  |

### Class Methods

Table - Class Methods

| Name | Procedure/Function | Description |
| --- | --- | --- |
| N/A |  |  |

### Class Properties

Table - Class Properties

| Class Properties Name | Type | Visibility | Description |
| --- | --- | --- | --- |
| N/A |  |  |  |

### Uses Clause

Not applicable.

### Forms

No existing CPRS forms are to be modified**.**

### Functions

Not applicable.

### Dialog

Not applicable.

### Help Frame

When the user clicks the Help button from the FAT prompt, the following window appears.

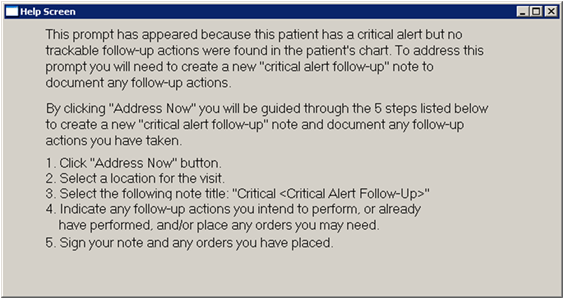


Figure - Help Screen

Table 58 - Help Frames

| *AWARE DLL FAT GUI Help Frame* | *Help for Answering FAT Prompt Form* | | | |
| --- | --- | --- | --- | --- |
| *Help Frame Text* | ***Help Screen*** | | | |
| *Enhancement Category* | *New* | *Modify* | *Delete* | *No Change* |
| *Help Frame Text Calling Mechanism* | *Help Button click* | | | |

| *Current Help Frame Text* |
| --- |
| *As described above* |

| *Modified Help Frame Text (Changes are in bold)* |
| --- |
| *None* |

### Alert Cache Viewer Web Application Interface

Alert Cache Viewer is written in HTML/JavaScript (Browser) and Intersystems Cache Server Page (Server Side).

Table - Save Access and Verify Codes

| CSP Method | SaveAV(ACCESS, VEIFY) | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| CSP Class | VEFA.Login.cls | | | | | | |
| CSP class extends | %CSP.Page | | | | | | |
| **Related Methods/Routines** | **Methods “Called By”** | | | | **Routines “Called”** | | |
| CSP Web Page: AlertCache.csp | | | | %CSP.Page.Encrypt | | |
| CSP variable references | %session | | | | | | |
| Interface | **Method:** SaveAV – Save Access and Verify Code  **INPUT**  ACCESS : User Access Code  VERIFY : User Verify Code  **OUTPUT**  Store ACCESS and VERIFY codes into %session | | | | | | |
|  |  | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |

Table - Validate and Login to Alert Cache Viewer

| CSP Method | LoginAlertCache() | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| CSP Class | VEFA.Login.cls | | | | | | |
| CSP class extends | %CSP.Page | | | | | | |
| **Related Methods/Routines** | **Methods “Called By”** | | | | **Methods/Routines “Called”** | | |
| CSP Web Page: AlertCache.csp | | | | %CSP.Page.Decrypt  $$GETDUZ^VEFACSP1  $$GETNAME^VEFACSP1  VEFA.AlertCache.FillGrid | | |
| CSP variable references | %session | | | | | | |
| Interface | **Method:** LoginAlertCache – Validate Access/Verify codes and populate web page with Alert Cache contents  **INPUT**  None  **OUTPUT**  Fill HTML controls with data (see VEFA.AlertCache.FillGrid) | | | | | | |
|  |  | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |

Table - Populate Web Page with Alert Cache Contents

| CSP Method | FillGrid() | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| CSP Class | VEFA.AlertCache.cls | | | | | | |
| CSP class extends | %CSP.Page | | | | | | |
| **Related Methods/Routines** | **Methods “Called By”** | | | | **Methods/Routines “Called”** | | |
| CSP Web Page: AlertCache.csp | | | | %CSP.Page.UnescapeURL  GETALCH^VEFAALC1  USRKEY^VEFAALR5 | | |
| CSP variable references | %session, %request | | | | | | |
| Interface | **Method:** FillGrid – Populate Web Page with Alert Cache contents  **INPUT**  None  **OUTPUT**  Fill HTML Grid control with Alert Cache Contents  Fill Order Provider HTML select  Fill Service HTML select  Fill Patient HTML select  Fill “Follow-up > 7 days” select | | | | | | |
|  |  | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |

Table - Show Alert Cache Detail Information

| CSP Method | Show() | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| CSP Class | VEFA.AlertCacheDetail.cls | | | | | | |
| CSP class extends | %CSP.Page | | | | | | |
| **Related Methods/Routines** | **Methods “Called By”** | | | | **Methods/Routines “Called”** | | |
| CSP Web Page: AlertCacheDetail.csp | | | | VEFA.Login.InvalidPage  GETDATA^VEFAALC1  GETFOL^VEFAALC1 | | |
| CSP variable references | %request | | | | | | |
| Interface | **Method:** LoginAlertCache – Validate Access/Verify codes and populate web page with Alert Cache contents  **INPUT**  None  **OUTPUT**  Fill all HTML Control boxes with Alert Cache data  Fill HTML Grid control with Alert Cache’s follow-up data | | | | | | |
|  |  | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |

### KB Editor Web Application Interface

KB Editor is written in HTML/JavaScript (Browser) and Intersystems Cache Server Page (Server side).

Table - Save Access and Verify Codes

| CSP Method | SaveAV(ACCESS, VEIFY) | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| CSP Class | VEFA.Login.cls | | | | | | |
| CSP class extends | %CSP.Page | | | | | | |
| **Related Methods/Routines** | **Methods “Called By”** | | | | **Routines “Called”** | | |
| CSP Web Page: kbeditor.csp | | | | %CSP.Page.Encrypt | | |
| CSP variable references | %session | | | | | | |
| Interface | **Method:** SaveAV – Save Access and Verify Code  **INPUT**  ACCESS : User Access Code  VERIFY : User Verify Code  **OUTPUT**  Store ACCESS and VERIFY codes into %session | | | | | | |
|  |  | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |

Table - Validate and Login to KB Editor

| CSP Method | LoginKBEditor() | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| CSP Class | VEFA.Login.cls | | | | | | |
| CSP class extends | %CSP.Page | | | | | | |
| **Related Methods/Routines** | **Methods “Called By”** | | | | **Methods/Routines “Called”** | | |
| CSP Web Page: kbeditor.csp | | | | %CSP.Page.Decrypt  $$GETDUZ^VEFACSP1  $$GETNAME^VEFACSP1  $$HASKBKEY^VEFAALR5  VEFA.KBEditorMain.FillGrids | | |
| CSP variable references | %session | | | | | | |
| Interface | **Method:** LoginAlertCache – Validate Access/Verify codes and populate web page with Alert Cache contents  **INPUT**  None  **OUTPUT**  Fill HTML controls with data (see VEFA.KBEditorMain.FillGrids) | | | | | | |
|  |  | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |

Table - Fill Web page with Alert Categories and Alert Types

| CSP Method | FillGrids() | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| CSP Class | VEFA.KBEditorMain.cls | | | | | | |
| CSP class extends | %CSP.Page | | | | | | |
| **Related Methods/Routines** | **Methods “Called By”** | | | | **Methods/Routines “Called”** | | |
| CSP Web Page: showmain.csp | | | | $$GETALCAT^VEFAKB01  $$GETALTYP^VEFAKB01  $$ISPROG^VEFAALR5  VEFA.Login.InvalidPage | | |
| CSP variable references | %session, %request | | | | | | |
| Interface | **Method:** FillGrids – populate web page with Alert Categories and Alert Types  **INPUT**  None  **OUTPUT**  Fill HTML Alert Category grid control with all Alert Categories  Fill HTML Alert Type grid control with all Alert Types | | | | | | |
|  |  | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |

Table - Populating Alert Category Form- Get Delete Options

| CSP Method | DeleteAlertOptions() | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| CSP Class | VEFA.AlertCategory.cls | | | | | | |
| CSP class extends | %CSP.Page | | | | | | |
| **Related Methods/Routines** | **Methods “Called By”** | | | | **Methods/Routines “Called”** | | |
| CSP Web Page: showalertcategory.csp | | | | $$GETDELOP^VEFAALC2 | | |
| CSP variable references |  | | | | | | |
| Interface | **Method:** DeleteAlertOptions – Get Delete Alert Options  **INPUT**  None  **OUTPUT**  Provide HTML Delete Alert select element with Alert Category’s Delete Alert options | | | | | | |
|  |  | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |

Table - Populating Alert Category Form- Notification Type Options

| CSP Method | NotfTypeOptions() | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| CSP Class | VEFA.AlertCategory.cls | | | | | | |
| CSP class extends | %CSP.Page | | | | | | |
| **Related Methods/Routines** | **Methods “Called By”** | | | | **Methods/Routines “Called”** | | |
| CSP Web Page: showalertcategory.csp | | | | $$GETNOTF^VEFAALC2 | | |
| CSP variable references |  | | | | | | |
| Interface | **Method:** NotfTypeOptions – Get Notification Type Options  **INPUT**  None  **OUTPUT**  Provide HTML Notification Type select element with Alert Category’s Notification types | | | | | | |
|  |  | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |

Table - Saving Alert Category

| CSP Method | Save() | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| CSP Class | VEFA.AlertCategory.cls | | | | | | |
| CSP class extends | %CSP.Page | | | | | | |
| **Related Methods/Routines** | **Methods “Called By”** | | | | **Methods/Routines “Called”** | | |
| CSP Web Page: savealertcat.csp | | | | %CSP.Page.UnescapeURL  VEFA.Login.DisplayResult  VEFA.Login.InvalidPage  VEFA.Login.SetCookie  $$SAVE^VEFAALC2 | | |
| CSP variable references | %request | | | | | | |
| Interface | **Method:** Save – Save Alert Category  **INPUT**  None  **OUTPUT**  New/Modified alert category values saved into Alert Category file | | | | | | |
|  |  | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |

Table - Disable Alert Category Fields

| CSP Method | DisableControls() | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| CSP Class | VEFA.AlertCategory.cls | | | | | | |
| CSP class extends | %CSP.Page | | | | | | |
| **Related Methods/Routines** | **Methods “Called By”** | | | | **Methods/Routines “Called”** | | |
| CSP Web Page: showalertcategory.csp | | | | VEFA.Login.InvalidPage  $ISPROG^VEFAALR5 | | |
| CSP variable references | %session | | | | | | |
| Interface | **Method:** DisableControls – Disable all HTML elements except description for non-programmer  **INPUT**  None  **OUTPUT**  For non-programmer, all HTML elements in showalertcategory except description disabled | | | | | | |
|  |  | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |

Table - Populate Alert Type Form

| CSP Method | Populate() | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| CSP Class | VEFA.AlertType.cls | | | | | | |
| CSP class extends | %CSP.Page | | | | | | |
| **Related Methods/Routines** | **Methods “Called By”** | | | | **Methods/Routines “Called”** | | |
| CSP Web Page: showalerttype.csp | | | | %CSP.Page.UnescapeURL  VEFA.Login.InvalidPage  VEFA.Login.Selector  GETATFRM^VEFAKB02  GETATLST^VEFAKB02 | | |
| CSP variable references | %request | | | | | | |
| Interface | **Method:** Populate – Populate web page with Alert Type information (Reminder/Order/follow-up/comments dialogs)  **INPUT**  None  **OUTPUT**  Populate HTML Reminder Dialog elements with Alert type’s Reminder information  Populate HTML Grid element with Order dialog (multiple records)  Populate HTML Grid element with Follow-up dialog (multiple records)  Populate HTML Grid element with Comments dialog (multiple records) | | | | | | |
|  |  | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |

Table - Saving Alert Type

| CSP Method | Save() | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| CSP Class | VEFA.AlertType.cls | | | | | | |
| CSP class extends | %CSP.Page | | | | | | |
| **Related Methods/Routines** | **Methods “Called By”** | | | | **Methods/Routines “Called”** | | |
| CSP Web Page: savealerttype.csp | | | | %CSP.Page.UnescapeURL  VEFA.Login.DisplayResult  VEFA.Login.SetCookie  $$DELALTYP^VEFAKB02  $$SAVATFRM^VEFAKB02  $$GETATIEN^VEFAKB02  INACTV^VEFAKB02  UPINITDT^VEFAKB02  BASICCHK^VEFAALR4 | | |
| CSP variable references | %request | | | | | | |
| Interface | **Method:** Save – Validate and Save Alert Type or Delete Alert type  **INPUT**  None  **OUTPUT**  **User Action:**  Add: New alert type created in Alert Type file  Edit: Modified alert type saved into Alert Type file  Delete: Alert type is removed from alert type file | | | | | | |
|  |  | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |

Table - Populate Order Dialog Form

| CSP Method | FillOrderDlg | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| CSP Class | VEFA.OrderDialog.cls | | | | | | |
| CSP class extends | %CSP.Page | | | | | | |
| **Related Methods/Routines** | **Methods “Called By”** | | | | **Methods/Routines “Called”** | | |
| CSP Web Page: showorderdlg.csp | | | | VEFA.Login.InvalidPage  VEFA.Login.FillOptions  GETNAME^VEFAKB01  GETREC^VEFAKB01  $$JSBOOL^VEFACSP1 | | |
| CSP variable references | %request | | | | | | |
| Interface | **Method:** FillOrderDlg – Populate Order Dialog web page  **INPUT**  None  **OUTPUT**  Populate HTML Order Dialog elements with alert type’s order dialog group information | | | | | | |
|  |  | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |

Table - Populate Follow-up Dialog Form

| CSP Method | FillFollowupDlg | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| CSP Class | VEFA.OrderDialog.cls | | | | | | |
| CSP class extends | %CSP.Page | | | | | | |
| **Related Methods/Routines** | **Methods “Called By”** | | | | **Methods/Routines “Called”** | | |
| CSP Web Page: showfoldlg.csp | | | | VEFA.Login.InvalidPage  VEFA.Login.FillOptions  GETNAME^VEFAKB01  GETREC^VEFAKB01  $$JSBOOL^VEFACSP1 | | |
| CSP variable references | %request | | | | | | |
| Interface | **Method:** FillFollowupDlg – Populate Followup Dialog web page  **INPUT**  None  **OUTPUT**  Populate HTML Follow-up Dialog elements with alert type’s followup dialog group information | | | | | | |
|  |  | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |

Table - Populate Comments Dialog Form

| CSP Method | FillCommentDlg | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| CSP Class | VEFA.OrderDialog.cls | | | | | | |
| CSP class extends | %CSP.Page | | | | | | |
| **Related Methods/Routines** | **Methods “Called By”** | | | | **Methods/Routines “Called”** | | |
| CSP Web Page: showcmtdlg.csp | | | | VEFA.Login.InvalidPage  VEFA.Login.FillOptions  GETNAME^VEFAKB01  GETREC^VEFAKB01  $$JSBOOL^VEFACSP1 | | |
| CSP variable references | %request | | | | | | |
| Interface | **Method:** FillCommentDlg – Populate Comment Dialog web page  **INPUT**  None  **OUTPUT**  Populate HTML Comment Dialog elements with alert type’s comment dialog group information | | | | | | |
|  |  | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |

Table - Saving Alert Type’s Order Dialog Record

| CSP Method | SaveOrderDlg() | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| CSP Class | VEFA.OrderDialog.cls | | | | | | |
| CSP class extends | %CSP.Page | | | | | | |
| **Related Methods/Routines** | **Methods “Called By”** | | | | **Methods/Routines “Called”** | | |
| CSP Web Page: saveorderdlg.csp | | | | VEFA.Login.InvalidPage  VEFA.AlertType.DeleteDlgForm  VEFA.AlertType.SaveDlgForm | | |
| CSP variable references | %request | | | | | | |
| Interface | **Method:** SaveOrderDialog – Save or Delete Alert Type’s Order Dialog record  **INPUT**  None  **OUTPUT**  **User Action:**  Add: New order dialog created in Alert Type’s Order Dialog group subfile  Edit: Modified order dialog saved into Alert Type’s order dialog group subfile  Delete: Order Dialog is removed from alert type’s order dialog group sub file  (see VEFA.AlertType.SaveDlgForm and VEFA.AlertType.DeleteDlgForm) | | | | | | |
|  |  | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |

Table - Saving Alert Type’s Follow-up Dialog Record

| CSP Method | SaveFollowupDlg() | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| CSP Class | VEFA.OrderDialog.cls | | | | | | |
| CSP class extends | %CSP.Page | | | | | | |
| **Related Methods/Routines** | **Methods “Called By”** | | | | **Methods/Routines “Called”** | | |
| CSP Web Page: savefoldlg.csp | | | | VEFA.Login.InvalidPage  VEFA.AlertType.DeleteDlgForm  VEFA.AlertType.SaveDlgForm | | |
| CSP variable references | %request | | | | | | |
| Interface | **Method:** SaveFollowupDialog – Save or Delete Alert Type’s Followup Dialog record  **INPUT**  None  **OUTPUT**  **User Action:**  Add: New follow-up dialog created in Alert Type’s Follow-up Dialog group subfile  Edit: Modified follow-up dialog saved into Alert Type’s follow-up dialog group subfile  Delete: Follow-up Dialog is removed from alert type’s follow-up dialog group sub file  (see VEFA.AlertType.SaveDlgForm and VEFA.AlertType.DeleteDlgForm) | | | | | | |
|  |  | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |

Table - Saving Alert Type’s Comment Dialog Record

| CSP Method | SaveCommentDlg() | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| CSP Class | VEFA.OrderDialog.cls | | | | | | |
| CSP class extends | %CSP.Page | | | | | | |
| **Related Methods/Routines** | **Methods “Called By”** | | | | **Methods/Routines “Called”** | | |
| CSP Web Page: savecmtdlg.csp | | | | VEFA.Login.InvalidPage  VEFA.AlertType.DeleteDlgForm  VEFA.AlertType.SaveDlgForm | | |
| CSP variable references | %request | | | | | | |
| Interface | **Method:** SaveCommentDialog – Save or Delete Alert Type’s Comment Dialog record  **INPUT**  None  **OUTPUT**  **User Action:**  Add: New comment dialog created in Alert Type’s comment dialog group subfile  Edit: Modified comment dialog saved into Alert Type’s comment dialog group subfile  Delete: Comment Dialog is removed from alert type’s comment dialog group sub file  (see VEFA.AlertType.SaveDlgForm and VEFA.AlertType.DeleteDlgForm) | | | | | | |
|  |  | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |

Table - Saving Alert Type’s Dialog Record Method

| CSP Method | SaveDlgForm() | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| CSP Class | VEFA.AlertType.cls | | | | | | |
| CSP class extends | %CSP.Page | | | | | | |
| **Related Methods/Routines** | **Methods “Called By”** | | | | **Methods/Routines “Called”** | | |
| CSP Web Page:  savefoldlg.csp  savecmtdlg.csp  saveorderdlg.csp | | | | %CSP.Page.UnescapeURL  VEFA.Login.SetCookie  VEFA.Login.DisplayResult  SAVDLG^VEFAKB02  GTDLGIEN^VEFACSP1  INACTV^VEFAKB02  UPINITDT^VEFAKB02  BASICCHK^VEFAALR4 | | |
| CSP variable references | %request | | | | | | |
| Interface | **Method:** SaveDlgForm – Save Alert Type’s Dialog text record  **INPUT**  None  **OUTPUT**  **User Action:**  Add: New dialog text created in Alert Type’s dialog group subfile  Edit: Modified dialog text saved into Alert Type’s dialog group subfile | | | | | | |
|  |  | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |

Table - Deleting Alert Type’s Dialog Record Method

| CSP Method | DeleteDlgForm() | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| CSP Class | VEFA.AlertType.cls | | | | | | |
| CSP class extends | %CSP.Page | | | | | | |
| **Related Methods/Routines** | **Methods “Called By”** | | | | **Methods/Routines “Called”** | | |
| CSP Web Page:  savefoldlg.csp  savecmtdlg.csp  saveorderdlg.csp | | | | VEFA.Login.SetCookie  VEFA.Login.DisplayResult  DELORDLG^VEFAKB02 | | |
| CSP variable references | %request | | | | | | |
| Interface | **Method:** DeleteDlgForm – Delete Alert Type’s Dialog text record  **INPUT**  None  **OUTPUT**  **User Action:**  Delete: Dialog text is removed from alert type’s dialog group sub file | | | | | | |
|  |  | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |

### QI ReportingTool Web Application Interface

Table - VistA Login Web Service

| Web Method | Login() | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify | Delete | | | No Change | |
| Web Service Class | VEFA.AWARE.cls | | | | | | |
| WebService class extends | %SOAP.WebService | | | | | | |
| **Related Methods/Routines** | **Methods “Called By”** | | | | **Methods/Routines “Called”** | | |
| .NET <aspx> | | | | $$GETDUZ^VEFACSP1  USRKEY^VEFAALR5  GETNAME^VEFACSP1 | | |
| Service Name | WSAWARE | | | | | | |
| Interface | **Method:** Login – Validate Access/Verify codes and return SOAP Response  **INPUT**  Access : User Access Code  Verify : User verify Code  **OUTPUT** **( SOAP Response )**  LoginResult  Fail : 0 ^ Error Message  Success :  1 ^ provide name ^ provider DUZ  2 ^ patient safety or programmer name ^ programmer DUZ | | | | | | |
|  |  | | | | | | |
| Data Passing | Input | Output Reference | | Both | | Global Reference | Local |

### VistA to SQL Transporter (Console Application)

This is a Windows Server scheduled background task that is done by retrieving data from VistA by RPC and writes it to SQL tables.

### HL7 Application Parameter

Not applicable.

### HL7 Logical Link

Not applicable.

### COTS Interface

QI Tool is leveraging Microsoft SQL Server Reporting Service (SSRS) to show Critical Alert Historical data.

## Communications Detailed Design

The following information is described in the detailed designs of this SDD (Systems Design Document) as appropriate, and in the diagram below:

* Details of servers and clients included on area network
* Specifications for bus timing requirements and bus control
* Format(s) for data being exchanged between components
* Connectivity between components, data flow (as applicable), and distances between components
* LAN topology (see next page)

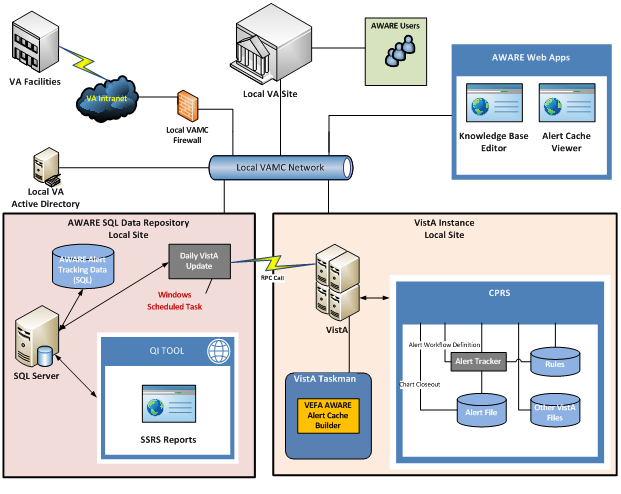


Figure - Communication Diagram

The server hardware/software is described below.

Table – Server Hardware

|  |
| --- |
| **Description** |
| PowerEdge T620, Intel Xeon E-26XX Processors (210-ABVO) |
| PowerEdge T620 Motherboard, TPM (591-BBBN) |
| Basic Hardware Services: Business Hours (5X10) Next Business Day On Site Hardware Warranty Repair Initial Year (989-9631) |
| Basic Hardware Services: Business Hours (5X10) Next Business Day On Site Hardware Warranty Repair 2 Year Exten (990-0291) |
| Dell Hardware Limited Warranty Plus On Site Service Initial Year (990-1521) |
| Dell Hardware Limited Warranty Plus On Site Service Extended Year (990-1631) |
| DECLINED CRITICAL BUSINESS SERVER OR STORAGE SOFTWARE SUPPORT PACKAGE-CALL YOUR DELL SALES REP IF UPGRADE NEED (990-1731) |
| SATA Hard Drive Ltd Warranty with Basic Support, 2 Year Extended (993-9412) |
| SATA Hard Drive Ltd Warranty with Basic Support, Initial Year (994-4500) |
| On-Site Installation Declined (900-9997) |
| Proactive Maintenance Service Declined (926-2979) |
| PowerEdge T620 Shipping (331-5592) |
| vFlash SD Slot Filler (318-2036) |
| iDRAC7 Express (331-3481) |
| Chassis with up to 8, 3.5" Hard Drives, Rack Configuration (331-5933) |
| T620 PERC Cable for 3.5in Chassis (331-6124) |
| Security Bezel (318-1544) |
| Power Saving Dell Active Power Controller (330-5116) |
| RAID 5 for H710P/H710/H310 (3-32 HDDs) (342-3955) |
| PERC H710p Adapter RAID Controller, 1GB NV Cache (342-4049) |
| Intel Xeon E5-2640 2.50GHz, 15M Cache, 7.2GT/s QPI, Turbo, 6C, 95W, Max Mem 1333MHz (317-9595) |
| T620 Heat Sink, 1 Proc, up to 115W (331-5602) |
| Intel Xeon E5-2640 2.50GHz, 15M Cache, 7.2GT/s QPI, Turbo, 6C, 95W (317-9609) |
| T620 Heat Sink, 1 Proc, up to 115W (331-5602) |
| 4GB RDIMM, 1333 MT/s, Low Volt, Single Rank, x4 Data Width (317-9649) |
| 1333 MHz RDIMMs (331-4422) |
| Performance Optimized (331-4428) |
| 600GB 15K RPM Self-Encrypting SAS 6Gbps 3.5in Hot-plug Hard Drive,FIPS140-2 (342-0550) |
| No System Documentation, No OpenManage DVD Kit (310-5171) |
| DVD-ROM, SATA, Internal (313-6765) |
| No Rack Rails or Cable Management Arm for Rack Chassis (330-3553) |
| Dual, Hot-plug, Redundant Power Supply (1+1), 495W (331-4603) |
| Power Cord, NEMA 5-15P to C13, 15 amp, wall plug, 10 feet / 3 meter (310-8509) |
| No Operating System (420-6320) |
| No Media Required (421-5736) |

Table – Server Software

|  |
| --- |
| **Description** |
| VLA WINDOWS SERVER STD 2012 PER 2 PROCESSORS (A6391092) |
| VLA SQL SERVER STD 2012 (A6401330) |
| VLA SQLCAL 2012 SNGL USRCAL (A6110684) |
| ELECTRONIC LICENSE CONFIRMATION electronic download only (A3458532) |

# External Interface Design

## Interface Architecture

The following AWARE components are interfaced with VistA:

* CPRS AWARE
* Patient Closed-out COM Object (DLL)
* Critical Alert Cache Collector
* Transporter
* KB Editor
* Alert Cache Viewer

The following AWARE components are interfaced with SQL Server:

* Transporter
* QI Reporting Tool

# Human Machine Interface

Input via CPRS (keyboard and mouse) with normal CPRS access required with normal CPRS entry Reminder dialog, order entry. The same occurs with the additional web applications including the KB Editor, Alert Cache Viewer, and QI tool.

## Interface Design Rules

Use of VA COM object CPRS interfacing described in document CPRS\_CreatingCustomCPRSGUIPlug-Ins(304)\_FC\_0208.pdf

Also, a new patient chart closeout function is added to the existing 3 COM object functions previously allowed by VA. New entry point logic for this function is provided in routine ORWCOM.

## Inputs

Keyboard/Mouse on Follow-up action prompting screen, and associated re-directed flow Reminder dialogs designed to perform follow-up actions as normal CPRS functionality. CPRS access needed.

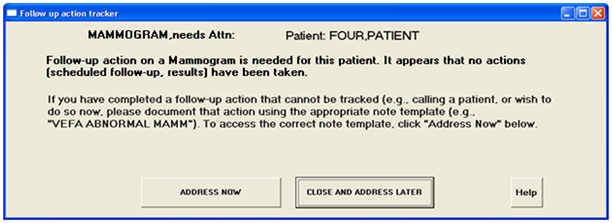


Figure - CPRS Prompting Screen

## Outputs

Indirectly, generation of outputs as orders, consults and text follow-up actions are the consequences of FAT prompting and follow-up Reminder dialog action(s).

Alert windows on patient selection dialog in CPRS reflects whether tracked acknowledged alerts have been renewed (as outputs), following acknowledge attempts when no associated follow-up actions have been done.

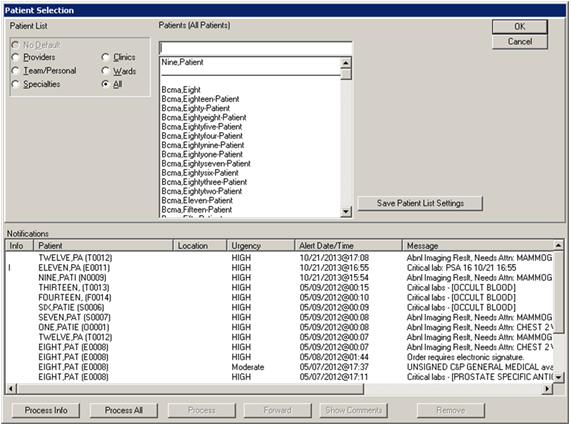


Figure - CPRS Patient Selection Screen

## Navigation Hierarchy

### CPRS/AWARE Screens

Navigation hierarchy starts with patient closeout at a desired selection of a new patient as below:

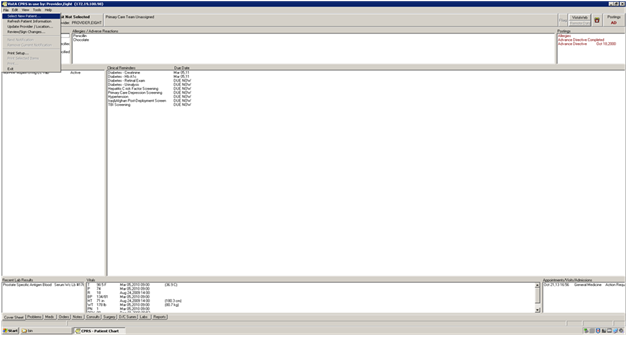


Figure - Select New Patient

A normal CPRS patient selection screen as below is not presented if a provider’s previous patient’s alert (under FAT) has been detected as needing follow-up action:

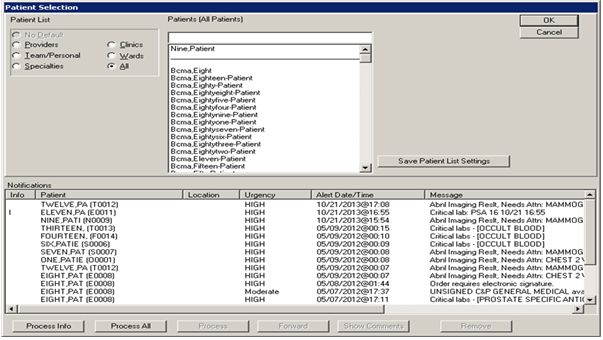


Figure - Patient Selection

Instead, a FAT prompt screen is presented as below to the provider for an opportunity to do follow-up action. The elements of such form are described in section 6.2.5.3.17 Components of Forms.

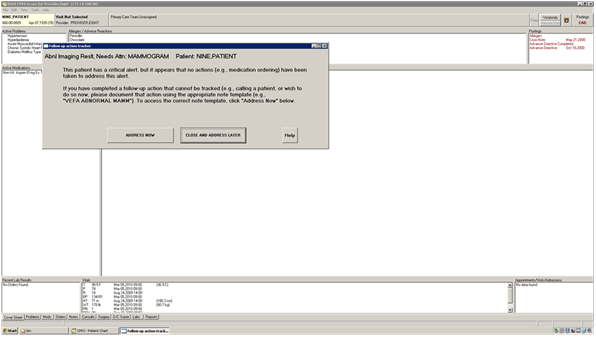


Figure - FAT Prompt

With opportunity to do a follow-up action on this prior patient, the provider can select the “ADDRESS NOW” button to bring up a Reminder Dialog with automatic re-direction of normal CPRS flow (as shown below) showing a particular template in the Template drawer:

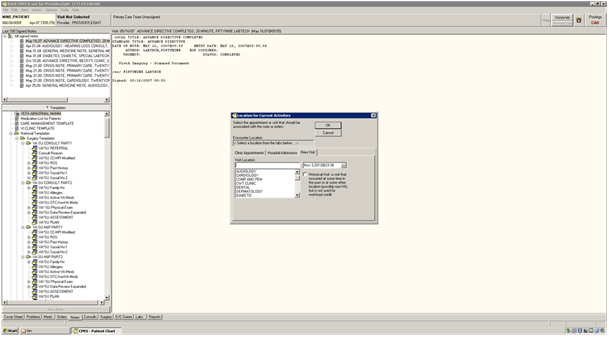


Figure - Clinic Selection (above)

After normal selection of a new visit at a chosen visit location and progress note title, a provider will then automatically be redirected with a specific Reminder Dialog to do follow-up action(s) shown in the sequence below:

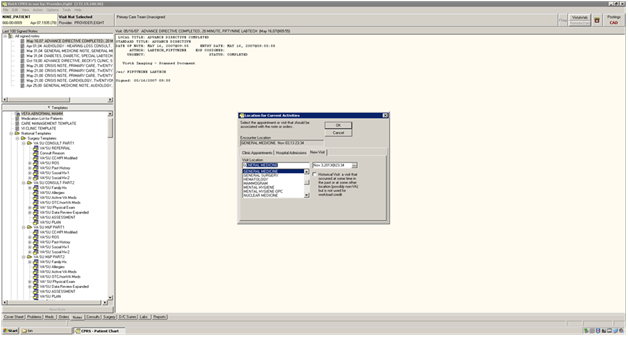


Figure - Clinic Selection (continued)

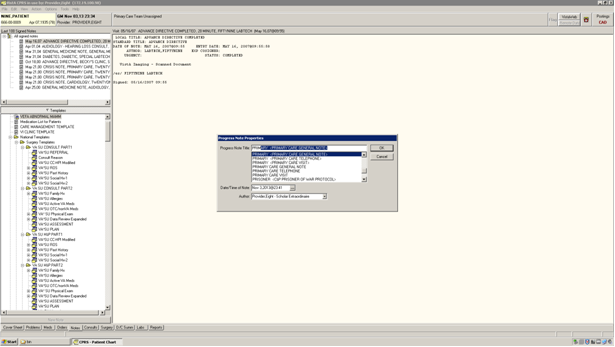


Figure - Progress Note Title Selection

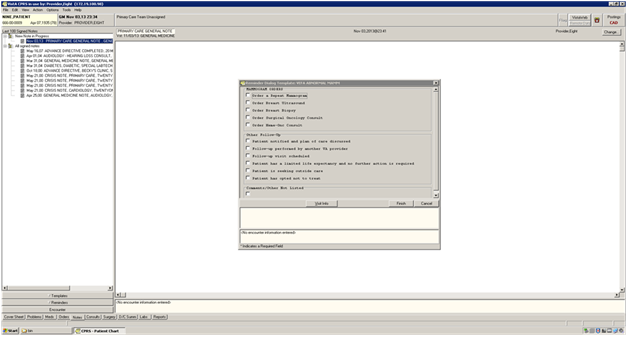


Figure - Reminder Dialog Present

### Alert Cache Viewer Screens

Navigation hierarchy for Alert Cache Viewer can be found in Section 6 – KB Editor User Interfaces.

### KB Editor Screens

Navigation hierarchy for KB Editor can be found in Section 6 – KB Editor User Interfaces.

### QI Tool Screens

Navigation hierarchy for QI Tool can be found in Section 6 – QI Tool User Interfaces.

# System Integrity Controls

Not applicable.

# Appendix A

## Requirements Traceability Matrix

## See AWARE Requirements Traceability Matrix.

## Packaging and Installation

* KIDS files
* AWARE DLL
* Customized CPRS executable
* Installation Instructions Guides and User Interface Guide and additional technical guide(s).
* Template Reminder Dialogs for further customization by each site’s CACs:
  + AVEFA1\_ABNORMAL\_CXR.PRD
  + AVEFA1\_ABNORMAL\_MAMMOGRAM.PRD
  + AVEFA1\_FOBT-FIT.PRD
  + AVEFA1\_PSA.PRD
* AWARE SQL Server Database
* SQL Transporter
* SSRS Reports
* QI Manager web application

## Design Metrics

The alert cache was designed to view recent critical alert type tracked activities, and pro-active site specific views of critical alerts that are being tracked and need follow-up actions (FAT). It was designed with allowance for initial collection and updating on a periodic basis. Subsequently it was to be uploaded into a longer term SQL storage. The criteria ( a nominal Cache window size, collection and uploading from Vista, and retrieval frequencies into SQL) was made so as to not place an undue burden on the system. Allowance was made for statistical use of a multi-facility SQL storage and dynamic look-up parameters to locate various data such as by facility, service, clinic, provider, alert type, and follow-up action data.

## Acronyms, Abbreviations, Terms, and Definitions

Table 81 - Acronyms, Abbreviations, Terms, and Definitions

| **Term** | **Definition** |
| --- | --- |
| AWARE | Alert Watch and Response Engine |
| CAC | Clinical Application Coordinator |
| COTS | Commercial Off The Shelf |
| CPRS | Computerized Patient Record System |
| CSP | Cache Server Page |
| D/T | Date and Time |
| FAT | Follow-up Action Tracking |
| FR | Functional Requirement |
| FTP | File Transfer Protocol |
| GUI | Graphical User Interface |
| HL7 | Health Level Seven |
| IIS | Internet Information Services |
| IT | Information Technology |
| JC | Joint Commission |
| OIT | Office of Information and Technology |
| PMAS | Project Management Accountability System |
| PWS | Performance Work Statement |
| RPC | Remote Procedure Call, |
| RSD | Requirements Specification Document |
| SDD | System Design Document |
| SQL | Structured Query Language |
| SSRS | SQL Server Reporting Service |
| TIU | Text Integration Utilities |
| TRM | VA’s Technical Reference Model |
| VA | Department of Veterans Affairs |
| VAMC | VA Medical Center |
| VHA | Veterans Health Administration |
| VISN | Veterans Integrated Service Network |
| VistA | Veterans Health Information Systems and Technology Architecture |

## Required Technical Documents

See section 1.4, Relationship to Other Plans..

# Attachment A - Approval Signatures

The following members of the governing IPT are required to sign. Please annotate signature blocks accordingly.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
Signed: Date:  
Blake Henderson   
Project Manager  
Innovation Coordinator

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
Signed: Date:  
Brian Stevenson   
Contracting Officer’s Representative  
Innovation Coordinator  
VHA OIA Innovation