Provider Utilities v1.1

System Design Document



August 2018

Version 1.6

Department of Veterans Affairs

Revision History

| Date | Version | Description | Author |
| --- | --- | --- | --- |
| 08/17/2018 | 1.6 | Updates to ORB3 routine to fix transfer to 3rd provider failure and updates to post install routine to slow it down, add checking of status capability and allowing stopping & resuming of the EPRACDT cross reference creation. | Rob Davis |
| 05/08/2018 | 1.5 | Updates to ORCSAVE routine which had overlap with OR\*3\*397 patch, therefore warranted a few updates. | Rob Davis |
| 03/05/2018 | 1.4 | Transfer Date/Time fix to allow times to be between 1 & 6 digits long. | Rob Davis |
| 10/6/2017 | 1.3 | Added another security related RPC and API into routine. | Rob Davis |
| 08/22/2017 | 1.2 | Added additionaly information to GUI section, relevant to post-review improvements. | Scott Stephenson |
| 08/16/2017 | 1.1 | Added new security related information into SDD | Scott Stephenson  Rob Davis |
| 06/21/2017 | 1.0 | Initial Draft | Scott Stephenson  Rob Davis |

Artifact Rationale

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

When to Complete Each Section of the SDD

| Section | Completed On or Before PMAS Phase | Rationale |
| --- | --- | --- |
| 1 – Introduction | MS 0 Review; updated thereafter | Conceptual design should inform evaluation of investments |
| 2 – Background | MS 0 Review; updated thereafter | Conceptual design should inform evaluation of investments |
| 3 – Conceptual Design | MS 0 Review; updated thereafter | Conceptual design should inform evaluation of investments |
| 4 – System Architecture | MS 0 Review; updated thereafter | Conceptual design should inform evaluation of investments |
| 5 – Data Design | MS 1 Review; updated thereafter | Design details should be elaborated upon during PMAS Planning phase and prior to development |
| 6 – Detailed Design | MS 1 Review; updated thereafter | Design details should be elaborated upon during PMAS Planning phase and prior to development |
| 7 – External System Interface Design | MS 1 Review; updated thereafter | Design details should be elaborated upon during PMAS Planning phase and prior to development |
| 8 – Human Machine Interfaces | MS 1 Review; updated thereafter | Design details should be elaborated upon during PMAS Planning phase and prior to development |
| Attachments | MS 1 Review; updated thereafter | Design details should be elaborated upon during PMAS Planning phase and prior to development |

A product’s system design should be defined conceptually prior to the allocation of personnel and resources that occur at project initiation. This gives the enterprise an opportunity to evaluate IT investments before project teams are stood up and funding is allocated. Sections 1- 4 which discuss the high-level design should be completed prior to MS 0. All sections should be completed and updated before MS 1. Projects will need to address all SDD approval constraints prior to the MS 2 review. In addition, the SDD should reflect the as-built product going into the MS 2 review.

Instructions

| Activity | New Capability (1) | Feature Enhancement (2) |
| --- | --- | --- |
| **Field Deployment (A)** | Yes | New Application |
| **Cloud/Web Deployment (B)** | No | No |
| **Mobile Application (C)** | No | No |

Table of Contents

[1. Introduction 1](#_Toc491162444)

[1.1. Scope 1](#_Toc491162445)

[1.2. User Profiles 1](#_Toc491162446)

[2. Background 1](#_Toc491162447)

[2.1. Overview of the System 1](#_Toc491162448)

[2.2. Overview of the Business Process 1](#_Toc491162449)

[2.3. Overview of the Significant Requirements 2](#_Toc491162450)

[3. Conceptual Design 2](#_Toc491162451)

[3.1. Conceptual Application Design 2](#_Toc491162452)

[3.1.1. Application Context 3](#_Toc491162453)

[3.1.2. High-Level Application Design 4](#_Toc491162454)

[3.1.3. Application Locations 5](#_Toc491162455)

[3.2. Conceptual Data Design 6](#_Toc491162456)

[3.2.1. Project Conceptual Data Model 6](#_Toc491162457)

[3.2.2. Database Information 6](#_Toc491162458)

[3.2.3. User Interface Data Mapping 7](#_Toc491162459)

[3.2.3.1. Application Screen Interface 7](#_Toc491162460)

[3.2.3.2. Application Report Interface 7](#_Toc491162461)

[3.2.3.3. Unmapped Data Element 7](#_Toc491162462)

[3.3. Conceptual Infrastructure Design 7](#_Toc491162463)

[3.3.1. System Criticality and High Availability 8](#_Toc491162464)

[3.3.2. Special Technology 8](#_Toc491162465)

[3.3.3. Technology Locations 8](#_Toc491162466)

[3.3.4. Conceptual Infrastructure Diagram 8](#_Toc491162467)

[3.3.4.1. Location of Environments and External Interfaces 8](#_Toc491162468)

[3.3.4.2. Conceptual Production String Diagram 8](#_Toc491162469)

[4. System Architecture 9](#_Toc491162470)

[4.1. Hardware Architecture 9](#_Toc491162471)

[4.2. Software Architecture 10](#_Toc491162472)

[4.3. Network Architecture 10](#_Toc491162473)

[4.4. Service Oriented Architecture / ESS 11](#_Toc491162474)

[4.5. Enterprise Architecture 11](#_Toc491162475)

[5. Data Design 11](#_Toc491162476)

[5.1. DBMS Files 11](#_Toc491162477)

[5.2. Non-DBMS Files 11](#_Toc491162478)

[5.3. Data View 12](#_Toc491162479)

[6. Detailed Design 12](#_Toc491162480)

[6.1. Hardware Detailed Design 12](#_Toc491162481)

[6.2. Software Detailed Design 12](#_Toc491162482)

[6.2.1. Delphi GUI Code Overview 12](#_Toc491162483)

[6.2.1.1. GUI Vista Remote Procedure Calls (RPC) 12](#_Toc491162484)

[6.2.1.1.1. ORWU DT [Now] 12](#_Toc491162485)

[6.2.1.1.2. ORQOR DETAIL [OrderIEN, PatientIEN] 12](#_Toc491162486)

[6.2.1.1.3. ORWU NEWPERS [Provider] 12](#_Toc491162487)

[6.2.1.1.4. ORQ3 EN [ProviderIEN, StartDate, EndDate] 12](#_Toc491162488)

[6.2.1.1.5. ORQ3 XFER [Order List] 12](#_Toc491162489)

[6.2.1.1.6. ORWU VERSRV 13](#_Toc491162490)

[6.2.1.1.7. ORWU USERINFO 13](#_Toc491162491)

[6.2.1.1.8. ORQ3 AUTHUSR 13](#_Toc491162492)

[6.2.1.2. Delphi Units 13](#_Toc491162493)

[6.2.2. Conceptual Design 14](#_Toc491162494)

[6.2.2.1. Product Perspective 14](#_Toc491162495)

[6.2.2.1.1. User Interfaces 14](#_Toc491162496)

[6.2.2.1.2. Hardware Interfaces 14](#_Toc491162497)

[6.2.2.1.3. Software Interfaces 15](#_Toc491162498)

[6.2.2.1.4. Communications Interfaces 15](#_Toc491162499)

[6.2.2.1.5. Memory Constraints 15](#_Toc491162500)

[6.2.2.1.6. Special Operations 15](#_Toc491162501)

[6.2.2.2. Product Features 15](#_Toc491162502)

[6.2.2.3. User Characteristics 15](#_Toc491162503)

[6.2.2.4. Dependencies and Constraints 15](#_Toc491162504)

[6.2.3. Specific Requirements 16](#_Toc491162505)

[6.2.3.1. Database Repository 16](#_Toc491162506)

[6.2.3.2. System Features 20](#_Toc491162507)

[6.2.3.2.1. NSR # 20130504 – Provider Role Change Enhancement 20](#_Toc491162508)

[6.2.3.2.1.1. OR3P453 20](#_Toc491162509)

[6.2.3.2.1.2. ORB3 Routine 22](#_Toc491162510)

[6.2.3.2.1.3. ORCSAVE Routine 24](#_Toc491162511)

[6.2.3.2.1.4. ORELR5 Routine 28](#_Toc491162512)

[6.2.3.2.1.5. ORQ2 Routine 31](#_Toc491162513)

[6.2.3.2.1.6. ORQ3 Routine 33](#_Toc491162514)

[6.2.3.2.2. Remote Procedure Call (RPC) 36](#_Toc491162515)

[6.3. Network Detailed Design 37](#_Toc491162516)

[6.4. Security and Privacy 38](#_Toc491162517)

[6.4.1. Security 38](#_Toc491162518)

[6.4.2. Privacy 38](#_Toc491162519)

[6.5. Service Oriented Architecture / ESS Detailed Design 39](#_Toc491162520)

[7. External System Interface Design 39](#_Toc491162521)

[7.1. Interface Architecture 39](#_Toc491162522)

[7.1.1. GUI Security 39](#_Toc491162523)

[7.2. Interface Detailed Design 39](#_Toc491162524)

[8. Human-Machine Interface 39](#_Toc491162525)

[8.1. Interface Design Rules 39](#_Toc491162526)

[8.2. Inputs 39](#_Toc491162527)

[8.3. Outputs 39](#_Toc491162528)

[8.4. Navigation Hierarchy 40](#_Toc491162529)

[8.4.1. Main Form: “Provider Utilities: Reassign Patient Orders” 40](#_Toc491162530)

[8.4.1.1. Usage (optimal program flow) 41](#_Toc491162531)

[8.4.1.2. Specific functionality 41](#_Toc491162532)

[8.4.2. Modal Form: “Select Current Provider” 42](#_Toc491162533)

[8.4.3. Modal Form: Qualifying Orders 43](#_Toc491162534)

[8.4.4. Modal Form: Reassign Orders (Review and Execute) 44](#_Toc491162535)

[8.4.5. Modal Form: About dialog 45](#_Toc491162536)

[9. Attachment A – Approval Signatures 45](#_Toc491162537)

[A. Additional Information 47](#_Toc491162538)

[A.1. Identification of Technology and Standards 47](#_Toc491162539)

[A.2. Constraining Policies, Directives and Procedures 47](#_Toc491162540)

[A.3. Requirements Traceability Matrix 47](#_Toc491162541)

[A.4. Packaging and Installation 47](#_Toc491162542)

[A.5. Design Metrics 47](#_Toc491162543)

# Introduction

This document is the System Design Document (SDD) for the Provider Utilities development project. Section 1 contains the purpose, scope, and references for the project.

This document will use the names “Provider Utilities” and “Provider Role Change” interchangeably. “Provider Utilities” is the file name of the executable application delivered to the customer. “Provider Role Change” is the CPRS menu item name by which the user executes the application. This menu item is found in the CPRS “Tools” menu,

This is a living document and will continue to evolve throughout the project. This project is implemented utilizing agile methods which include multiple sprints, or short development cycles. This document will be updated regularly during the development cycle to reflect the changes implemented in a sprint along with changes planned for future sprints.

## Scope

As determined by stakeholders during a series of interactive meetings, this product will allow a user to move one or more patients and associated qualifying orders from one provider to another (or several providers) for notification purposes. Scope and features shall not exceed that functionality.

## User Profiles

Provider Utilities can be used any authorized VA user tasked with supporting VA providers, but was primarily designed for use by the following user types:

* **Clinical Applications Coordinator(CAC).** A CAC may be tasked with conducting administrative support for one or more providers. For example, a CAC may be asked to reassign qualifying orders in the event of an unplanned reassignment of a provider.
* **Provider**. A provider may prefer to perform certain administrative tasks directly. An example would be a provider changing roles who wishes to examine a list of patients and orders, and then individually reassign qualifying orders based on the nature of each patient/order.

# Background

## Overview of the System

Subject Matter Experts (SMEs) from affected areas such as physicians, nurses and pharmacists as well as clinical application coordinators have been involved in determining the final requirements and will be involved in the field testing and approval of these enhancements for release.

## Overview of the Business Process

Provider Utilities supports the following business processes:

* Provider Role Change: The reassignment of qualifying orders to another provider in the event the ordering provider changes roles (and surrogacy is impracticable).



## Overview of the Significant Requirements

Link to the consolidated RSD: <http://DNS.URL/pasdocs/analysis/20150609_CPRS%20Enhancements%20Phase%20I_BRD.zip>

Link to the NSR: <http://DNS.URL/nsrd/Tab_GeneralInfoview.asp?RequestID=20130504>

Link to RTM: <http://DNS.URL/pasdocs/traceability/20150609_CPRS%20Enhancements%20Phase%20I_RTM.xlsx>

# Conceptual Design

## Conceptual Application Design

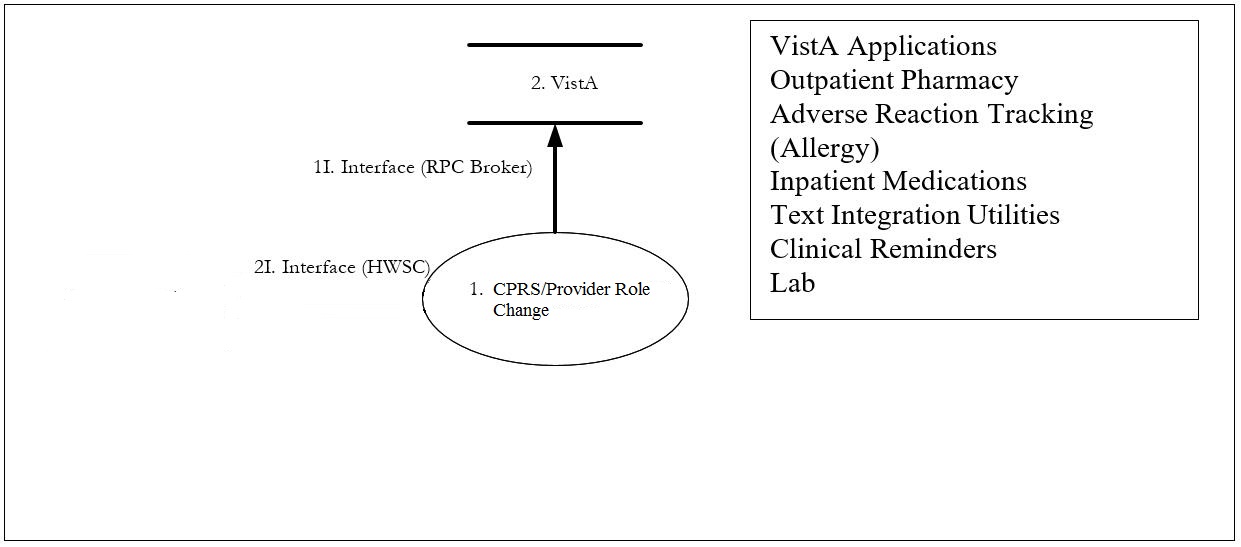
Provider Utilities is a standalone Windows application implemented in the Delphi XE8 language. It is completely independent of CPRS. CPRS and Provider Utilities do not communicate or share data. The sole relationship between the applications is that CPRS can “launch and forget” Provider Utilities from the Tools menu (Provider Role Change menu item).

Provider Utilities communicates with the Vista server using the same component library as CPRS. It logs into the server with the CPRS identification key. It validates the user via 2-factor authentication and the user ID and pass code.

This project comprises changes in both M and Delphi.

### VISTA Application Context

This application uses the newly created “OR PU GUI” Vista application context. For VISTA services, the existing CPRS broker framework is utilized.



Application Context Diagram

Table 5 (Grouping): Application Context Description

Object

| ID | Name | Description | Interface Name | Interface System |
| --- | --- | --- | --- | --- |
| 1 | Provider Utiliities  Computerized Patient Record System (CPRS) | Provider Utilities and CPRS (1) retrieves and sends clinical information to VistA (2). This includes order information such as Pharmacy Prescriptions. | RPC Broker | VistA |
| 2 | VistA | VistA is the primary data store for Patient and Health related information. | RPC Broker | CPRS |

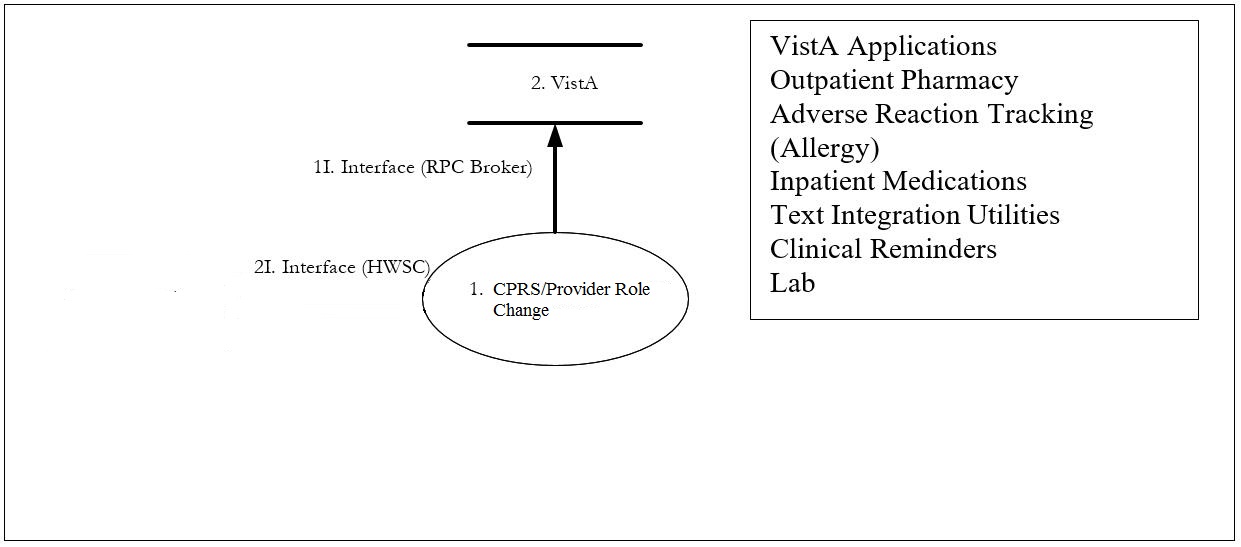
Interfaces External to OIT

CPRS does not contain any interfaces external to OIT.

Interfaces Internal to OIT

| ID | Name | Related Object | Input Messages | Output Messages | External Party |
| --- | --- | --- | --- | --- | --- |
| 1I | RPC Broker | CPRS (1)  VistA (2) | Message containing a request for information or order information | Requested information, or notification of success. | N/A |

### High-Level Application Design



High-Level Application Design

Table 6: Objects in the High Level Application Design

Objects / Components to be Built or Modified

Internal Data Stores

| ID | Name | Data Stored | Steward | Access |
| --- | --- | --- | --- | --- |
| 2 VistA (Order reassignment tracking) | Order Reassignment Tracking | From Provider  To Provider User Requesting Reassignment  Date Time of Reassignment | OR-Order Reassignment Tracking | Create  Retrieve |

### Application Locations

Provider Utilities is a CPRS adjunct application. As directed in the GUI SAC document, site administrators are expected to install Provider Utilities in the “C:\Program Files\VISTA\Provider Utilities” directory.

Even though Cloud technology is a goal in reforming Federal IT, this project will not utilize that technology (out of scope). In addition, regional deployments are used at various stations, however, it is not the decision of the development team. Regional deployment decisions are made by the regional IT staff.

Table 7: Application Locations

| Application Component | Description | Location at Which Component is Run | Type |
| --- | --- | --- | --- |
| CPRS  Provider Role Change | Executable | Workstations located throughout each VAMC. | Presentation Logic although there is some business logic contained in the executable. |
| VistA Server | Database which also contains the server side executable code of the VistA applications. | VistA servers are located either in local VAMCs or at regional data processing centers. | There are some applications that use the roll/scroll presentation logic. It is also the database and contains business logic and interface code. |

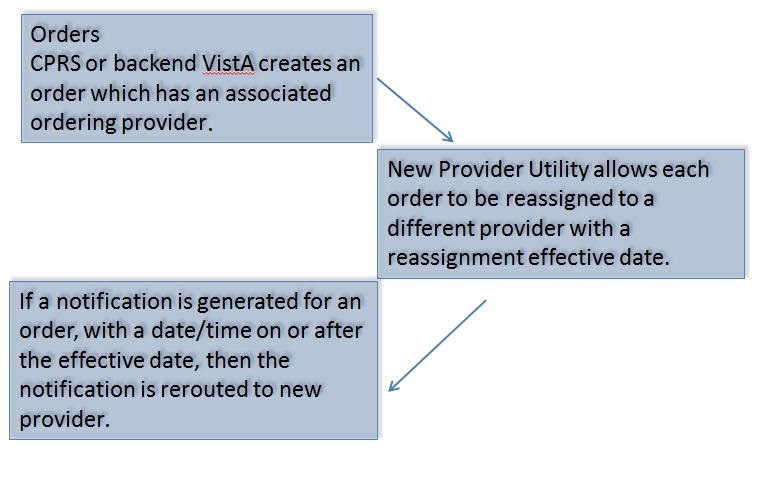
Table 8: Application Users

| Application Component | Location | User |
| --- | --- | --- |
| Provider Role Change  CPRS  VistA Server | Users for both CPRS and the VistA server are located not only locally at each VAMC, but can also be located at CBOCS or may be located at remote location and connect through CAG or VPN | Physician  Clinical Applications Coordinator |

## Conceptual Data Design

### Project Conceptual Data Model

Project Conceptual Data Model



### Database Information

Table 9: Database Inventory

| Database Name | Description | Type | Steward |
| --- | --- | --- | --- |
| VistA | M/Cache database containing clinical and non-clinical data for all VistA application | Modify | Overall, the VA is the steward of the database in its’ entirety.  Specifically, data is being modified in data that is owned by Pharmacy, CPRS, Kernel and Laboratory. |

### User Interface Data Mapping

#### Application Screen Interface

See section 8 for a full description of the Provider Utilites GUI hierarchy.

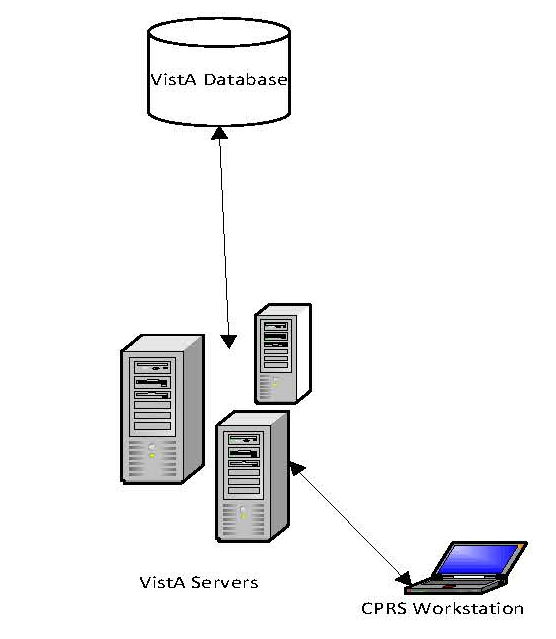
#### Application Report Interface

N/A.

#### Unmapped Data Element

Currently, there are no new data elements being added that are not mapped to a user interface.

## Conceptual Infrastructure Design



### System Criticality and High Availability

Provider Utilities is a CPRS adjunct application. CPRS is considered a mission critical system. However, without the VistA database and servers, CPRS cannot function. OIT has designed regional data processing centers and ‘hot backup’ sites for VistA instances. This project is not modifying those approaches. Any modification to those schemes will be done outside the consideration of these enhancements.

### Special Technology

N/A

### Technology Locations

Table 13: Technology Location Details

| Technology Component  Production 1 | Location | Usage |
| --- | --- | --- |
| Workstations | VAMCs  CBOCs  Remote (through CAG and VPN) | Clinicians and support staff |
| Special Hardware | None |  |
| Interface Processors | None | None |
| Legacy Mainframe | None |  |
| Legacy Application Server | None |  |
| Legacy Databases | Local  Regional | Houses VistA database. |
| Other | None |  |

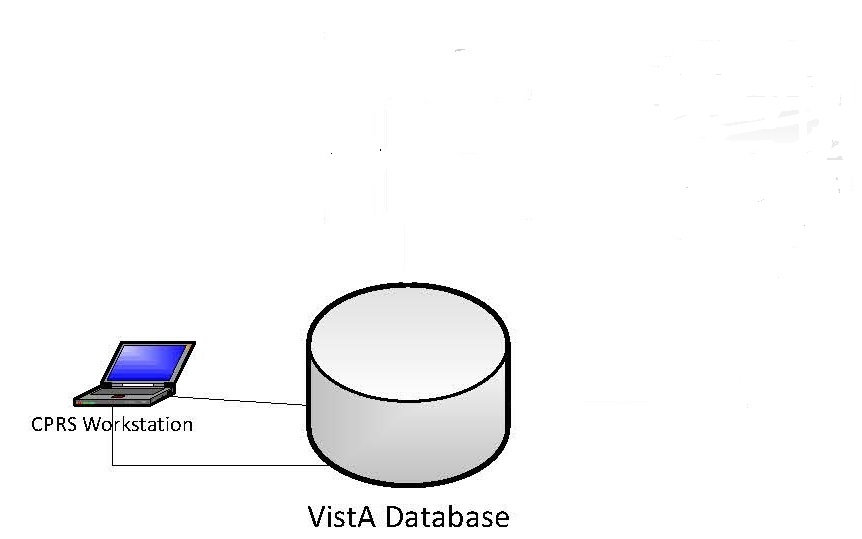
### Conceptual Infrastructure Diagram

#### Location of Environments and External Interfaces

Currently, the location of environments varies between VAMCs. Some VAMCs still have local hardware configurations. Many are converted to a regional data processing setup. There is a group responsible for determining the appropriate configuration and scheduling conversions for sites from local to regional deployment.

Provider Utilities and CPRS is designed to fit within whatever environment exists: either local or regional.

#### Conceptual Production String Diagram



Conceptual Production String Diagram

# System Architecture

## Hardware Architecture

Provider Utilities is a CPRS adjunct application. It follows and utilizes CPRS conventions, described below.

CPRS is a legacy GUI application that provides a GUI front-end to the VistA system and is primarily used by physicians, nurses and other clinicians responsible for providing patient care. As such, it utilizes the existing VistA hardware architecture.

Some VistA instances are local to the VAMC that use it. Others are installed in regional data processing centers. Most of these instances have hot backup sites located in another geographically separated location.

The primary architecture at this time is a cluster of Linux servers that act as the applications server. The client workstations are connected to this cluster. This cluster is connected to a cluster of VMS servers that act as the database server where the VistA database resides.

The client workstations may be local to the VAMC or they may be remote at CBOCs, other VAMCs (in the case of integrated sites) or may even be one-off remote workstations that connect via CAG or VPN.

While personnel outside of Product Development are responsible for determining the best configurations and ensuring adequate hardware and network connectivity, the CPRS v32 project considers additional space and potential performance impacts. The ultimate goal is to add no more than an additional 5% disk space or CPU requirements.

During the field testing phase, any additional files created are monitored to ensure this is not exceeded. Local and regional IT staff monitor the systems and notifie CPRS development is the performance exceeds expectations based on the development environment.



## Software Architecture

Provider Utilities is an independent Delphi application running under Windows. It communicates with the VistA server using existing CPRS mechanisms.

The Delphi GUI application is a single-threaded Windows executable with a main and ancillary windows. The window hierarchy is described in detail in section 8.

## Network Architecture

Provider Utilities is a CPRS adjunct application. It follows and utilizes CPRS conventions, described below.

CPRS is a legacy GUI application that provides a GUI front-end to the VistA system and is primarily used by physicians, nurses and other clinicians responsible for providing patient care. The executable portion of CPRS will continue communicate using the existing network architecture that supports the legacy VistA systems.

CPRS utilizes remote procedure calls (RPCs) over the local or wide area network to communicate between the client and the VistA instance. This communication utilizes Kernel’s broker package.

Reference the hardware architecture for a high-level overview of the communication pathways.

## Service Oriented Architecture / ESS

Provider Utilities is a CPRS adjunct application. It follows and utilizes CPRS conventions, described below.

CPRS is a legacy GUI application that provides a GUI front-end to the VistA system and is primarily used by physicians, nurses and other clinicians responsible for providing patient care. CPRS is enhancing the existing CPRS system.

Note: CPRS’s architecture does not supply new services or consume services.

## Enterprise Architecture

Provider Utilities is a CPRS adjunct application. It follows and utilizes CPRS conventions. Provider Role Change uses Delphi XE8 for the Delphi development.

The server side code is written using Cache/MUMPS, which is approved under the TRM.

# Data Design

## DBMS Files

CPRS Provider Role Change will utilize the existing FileMan database. The following updates have been made to Data Dictionaries within FileMan.

* Within the ORDER File (#100), a new “New Style” cross reference “EPRACDT” has been created which is keyed off of the PROVIDER (#100.008, 3) & DATE/TIME ORDERED (#100.008, .01) subfields of the ORDER ACTIONS Multiple so that Orders can speedily be retrieved for a specific Provider and Date/Time Ordered for possible Re-Assignment to one or more other Providers.
* Also, within the ORDER File (#100), a new ORDER TRANSFERS Multiple (#100.011) has been created with each subrecord holding a TRANSFER DATE/TIME (#100.011, .01) Date/Time type field, TRANSFERRED FROM (#100.011, .02) pointer field, TRANSFERRED TO (#100.011, .03) pointer field, & TRANSFER USER (#100.011, .04) pointer field. The latter three fields point to NEW PERSON File (#200).
* As regards this new ORDER TRANSFERS Multiple just described above, a new “New Style” cross reference “EPRTRDT” has been created which is keyed off of the TRANSFERRED TO (#100.011, .03) & TRANSFER DATE/TIME (#100.011, .01) subfields to handle the case where an Order was Re-assigned to a Provider, yet this Provider has to Re-Assign the same Order to a 3rd Provider.

Section 6.2.2 contains the specifics of the data dictionary updates.

The files will be accessed either by FileMan Database Server (DBS) calls or by direct global access, where appropriate and allowed.

## Non-DBMS Files

N/A

## Data View

There are no updates planned for the persistent data objects currently in the released version of CPRS. Since no Entity Relationship Diagram (ERD) can be located CPRS will construct a high level ERD to be included in the final version of this document.

# Detailed Design

## Hardware Detailed Design

Provider Utilities will use existing CPRS hardware infrastructure and will be designed to require no significant increase in data storage capacity or CPRS resources.

## Software Detailed Design

### Delphi GUI Code Overview

See Section 8 for a visual overview of the Delphi GUI windows, to include the functionality and specified behavior or each.

#### GUI Vista Remote Procedure Calls (RPC)

The following RPCs are used by the Delphi GUI layer. Some are shared with CPRS and are used unmodified. Others are exclusive to this application. See Section 6 for a detailed explanation of new and modified RPCs. Most RPC calls are made from the Delphi “Utilities.pas” unit.

##### ORWU DT [Now]

Existing CPRS utility used unchanged. Gets the server time. Used by the GUI’s date-time controls. See the Delphi “FMNow” method.

##### ORQOR DETAIL [OrderIEN, PatientIEN]

Existing CPRS utility used unchanged. Gets order details. Used by the GUI’s “Order Detail” window. See the Delphi “OrderDetails” method.

##### ORWU NEWPERS [Provider]

Existing CPRS utility used unchanged. Gets a list of providers for use in the GUI’s current and new provider list boxes. See the Delphi “SubSetOfProviders” method.

##### ORQ3 EN [ProviderIEN, StartDate, EndDate]

*This is a new utility created for this application*. Gets a list of qualifying orders issued by the provider during the specified date period. See the Delphi “GetProviderOrdersFromServer” method.

##### ORQ3 XFER [Order List]

*This is a new utility created for this application*. Transfers specified orders between old and new providers. See the Delphi “reassignOrders” method.

##### ORWU VERSRV

Existing CPRS utility used unchanged. However, this application passes a unique application context, “OR PU GUI”. The Vista server then restricts the RPCs this application is allowed to access. See the Delphi “ServerVersion” method.

This method serves a second purpose, as the application checks the returned server version to ensure it supports the required RPC functionality.

##### ORWU USERINFO

Existing CPRS utility used unchanged. This gets user information. This application only uses the user timeout option in the returned record (to support GUI timeout functionality). See the Delphi “GetUserInfo” method.

##### ORQ3 AUTHUSR

*This is a new utility created for this application*. It verifies the user is authorized to use Provider Utilities by checking the “OR PU ACCESS” key. See the Delphi “IsAuthUser” method.

##### ORQ3 CHKPATCH

*This is a new utility created for this application.* It verifies to CPRS that OR\*3.0\*453 has been installed on the M Server side. Other future patches in relation to Provider Role could update this call to include those patch(es) as well.

#### Delphi Units

* **ProviderUtilities.dpr**: The Delphi project file, which lists the units used by the application and contains program initialization code.
* **ProviderUtilities.dproj**: The Delphi project configuration files. This is an XML file containing various project settings and information.
* **ProviderUtilities.res**: The Delphi project resource file. Contains standard Windows resource information linked to the final executable.
* **fAbout**.**pas & .dfm**: The Delphi source code for the “About” dialog.
* **fBase508Dialog**.**pas & .dfm**: The Delphi source code for the ancestor to all popup dialog windows.
* **fBase508Form**.**pas & .dfm**: The Delphi source code for the ancestor to all application forms.
* **fReassignOrders.pas & .dfm**: The Delphi source for the main window.
* **fApplyChanges.pas & .dfm**: The Delphi source code for the “Review and Apply Changes” window.
* **ProviderEngine**.**pas**: The Delphi source code for the provider engine object. This object performs all non-visual work involved in reassigning orders.
* **fConnectToServer.pas & .dfm**: The Delphi source code for a connection banner frame that appears at the top of the main window during startup.
* **fSelectProvider.pas & .dfm**: The Delphi source code for the form to select a new or current provider (and optionally a qualified order date range).
* **fPatientOrders**.**pas & .dfm**: The Delphi source code for the orders display form.
* **fProgramInformation**.**pas & .dfm**: The Delphi souce code for the “About” form.
* **Utilities**.**pas**: The Delphi source code for some support utilities.
* **fGN\_RPCLog**.**pas & .dfm**: Borrowed from Group Notes application. A logging window for RPC debugging.
* **UInit**.**pas**: Manages timeout functionality. Borrowed from CPRS.

### Conceptual Design

#### Product Perspective

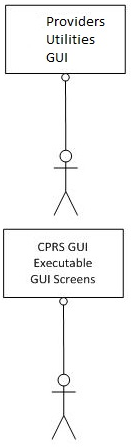
[Link to CPRS EP1 Requirements](http://vista.med.va.gov/pasdocs/traceability/20150609_CPRS%20Enhancements%20Phase%20I_RTM.xlsx)

[Link to (CPRS) v32 Requirements](http://tspr.vista.med.va.gov/warboard/ProjectDocs/CPRS_version_32/CPRSv32%20RSD%20v1.5%20(Feb%202015).docx)

##### User Interfaces

The following three block diagrams show, at its simplest form the interfaces in use for CPRS. There will be two GUI executables: CPRS and Provider Role Change. Both of these utilize GUI screens to communicate with users.

For all VistA (server side) changes, the user interface will be roll and scroll.



##### Hardware Interfaces

Provider Utilities is a CPRS adjunct application.

CPRS is a legacy GUI application that provides a GUI front-end to the VistA system and is primarily used by physicians, nurses and other clinicians responsible for providing patient care. Provider Utilities is enhancing the existing CPRS system. No modifications to the existing hardware interfaces are planned.

Currently, CPRS, utilizing the Kernel system will support any device that Kernel supports.

##### Software Interfaces

Provider Utilities is a CPRS adjunct application.

CPRS is a legacy GUI application that provides a GUI front-end to the VistA system and is primarily used by physicians, nurses and other clinicians responsible for providing patient care. Provider Utilities is enhancing the existing CPRS system.

CPRS communicates with all the VistA clinical applications and several of the financial applications. In addition, CPRS uses FileMan and Kernel. CPRS is written using the current nationally released version of each of these packages and plans to continue to utilize and support the currently released version.

##### Communications Interfaces

Provider Utilities is a CPRS adjunct application.

CPRS is a legacy GUI application that provides a GUI front-end to the VistA system and is primarily used by physicians, nurses and other clinicians responsible for providing patient care. Provider Utilities is enhancing the existing CPRS system. No modifications will be made to the existing communications interface which are under the control of Engineering.

##### Memory Constraints

N/A

##### Special Operations

N/A

#### Product Features

Please refer to Section 2.3 and Section 8.

#### User Characteristics

Provider Utilities is a CPRS adjunct application.

CPRS is used primarily by clinicians such as physicians, nurses, physician assistants, nurse practioners, pharmacists and other ancillary clinical users.

#### Dependencies and Constraints

Provider Utilities must utilize the existing hardware and network infrastructure. Therefore, the increase in network, memory, CPU and data storage can not be significant. Some remote installations, such as the CBOCs are particularly susceptible.

Provider Utilities is not 508 compliant.

### Specific Requirements

#### Database Repository

The data for CPRS Provider Role Change will reside in the existing VistA database.

Modifications (shown in Yellow) to Fields in ORDER ACTIONS Multiple of ORDER File (#100):

100.008,.01 DATE/TIME ORDERED 0;1 DATE (Required)

INPUT TRANSFORM: S %DT="ETXR" D ^%DT S X=Y K:Y<1 X

LAST EDITED: MAY 16, 2017

HELP-PROMPT: Enter the date/time this action was

Ordered.

DESCRIPTION: This is the date/time this action was

ordered.

CROSS-REFERENCE: 100^ACT3^MUMPS

1)= N ORDA S ORDA=DA(1) D ACT1^ORDD100A(ORDA,DA)

2)= N ORDA S ORDA=DA(1) D ACT2^ORDD100A(ORDA,DA)

^OR(100,"ACT",ORVP,9999999- ORLOG,ORDG,ORIFN,DA)

Allows retrieval of orders in

reverse-chronological order.

CROSS-REFERENCE:100^AC1^MUMPS

1)= N X1,X2 S X1=DA(1),X2=DA D SET^ORDD100(X1,X2)

2)= N X1,X2 S X1=DA(1),X2=DA D KIL^ORDD100(X1,X2)

^OR(100,"AC",ORVP,9999999-ORLOG,ORIFN,DA)

Allows retrieval of "active" orders; includes

orders with any kind of active status, as

well as those that have ended w/in the

Active Orders Context Hours.

CROSS-REFERENCE:100^AS1^MUMPS

1)= N ORDA S ORDA=DA(1) D S1^ORDD100(ORDA,DA,"",X)

2)= N ORDA S ORDA=DA(1) D S2^ORDD100(ORDA,DA,"",X)

^OR(100,"AS",ORVP,9999999-ORLOG,ORIFN,DA)

Allows retrieval of unsigned orders.

CROSS-REFERENCE:100^AF

1)= S ^OR(100,"AF",$E(X,1,30),DA(1),DA)=""

2)= K ^OR(100,"AF",$E(X,1,30),DA(1),DA)

RECORD INDEX: EPRACDT (#1454) REGULAR IR

LOOKUP & SORTING WHOLE FILE (#100)

Short Descr: Index on PROVIDER & ACTION DATE/TIME for

Activities

Description: This index will be used as part of the new

ORQ3 EN Remote Procedure Call to speedily

pick up all Orders for a specific Ordering

Provider and within a date range that has

been selected by the end user so that the

Orders that come up can be Re-Assigned

(Transferred) to one or more New Providers.

Set Logic: S ^OR(100,"EPRACDT",X(1),X(2),DA(1),DA)=""

Kill Logic: K ^OR(100,"EPRACDT",X(1),X(2),DA(1),DA)

Whole Kill: K ^OR(100,"EPRACDT")

X(1): PROVIDER (100.008,3) (Subscr 1) (forwards)

X(2): DATE/TIME ORDERED (100.008,.01) (Subscr 2)

(forwards)

100.008,3 PROVIDER 0;3 POINTER TO NEW PERSON FILE (#200)

LAST EDITED: MAY 16, 2017

HELP-PROMPT: Enter the name of the requesting clinician

for this order.

DESCRIPTION:

This is the requestor of this order.

RECORD INDEX: EPRACDT (#1454) REGULAR IR

LOOKUP & SORTING WHOLE FILE (#100)

Short Descr: Index on PROVIDER & ACTION DATE/TIME for

Activities

Description: This index will be used as part of the new

ORQ3 EN Remote Procedure Call to speedily

pick up all Orders for a specific Ordering

Provider and within a date range that has

been selected by the end user so that the

Orders that come up can be Re-Assigned

(Transferred) to one or more New Providers.

Set Logic: S ^OR(100,"EPRACDT",X(1),X(2),DA(1),DA)=""

Kill Logic: K ^OR(100,"EPRACDT",X(1),X(2),DA(1),DA)

Whole Kill: K ^OR(100,"EPRACDT")

X(1): PROVIDER (100.008,3) (Subscr 1) (forwards)

X(2): DATE/TIME ORDERED (100.008,.01) (Subscr 2)

(forwards)

New ORDER TRANSFERS Multiple (along with New Style Cross Reference) within ORDER File (#100):

100,70 ORDER TRANSFERS 11;0 DATE Multiple #100.011

DESCRIPTION: This is the list of Order Transfers that have

occurred transferring to a new Provider from

an old one.

100.011,.01 TRANSFER DATE/TIME 0;1 DATE (Required)

INPUT TRANSFORM: S %DT="ETXR" D ^%DT S X=Y K:Y<1 X

LAST EDITED: JUN 01, 2017

HELP-PROMPT: Enter the Date/Time of the Order Transfer.

DESCRIPTION:

This is the Date/Time of the Order

Transfer.

CROSS-REFERENCE: 100.011^B

1)= S

^OR(100,DA(1),11,"B",$E(X,1,30),DA)=""

2)= K ^OR(100,DA(1),11,"B",$E(X,1,30),DA)

RECORD INDEX: EPRTRDT (#1332) REGULAR IR

LOOKUP & SORTING WHOLE FILE (#100)

Short Descr: Index on TRANSFERRED TO Provider and

TRANSFER DATE/TIME

Description: This index will be used as part of the new

ORQ3 EN Remote Procedure Call to speedily

pick up all Orders which have already been

transferred from the original Ordering

Provider to one or more New Providers in

the case where a New Provider also happens

to need his or her Orders transferred to a

third New Provider, then this index can

then be conducive to this type of scenario.

It works similar to EPRACDT Index but will

utilize TRANSFERRED TO Provider & TRANSFER

DATE/TIME to quickly bring up Orders for

additional Re-Assignment (Transfer).

Set Logic: S ^OR(100,"EPRTRDT",X(1),X(2),DA(1),DA)=""

Kill Logic: K ^OR(100,"EPRTRDT",X(1),X(2),DA(1),DA)

Whole Kill: K ^OR(100,"EPRTRDT")

X(1): TRANSFERRED TO (100.011,.03) (Subscr 1)

(forwards)

X(2): TRANSFER DATE/TIME (100.011,.01) (Subscr

2) (forwards)

100.011,.02 TRANSFERRED FROM 0;2 POINTER TO NEW PERSON FILE (#200)

(Required)

LAST EDITED: MAY 04, 2017

HELP-PROMPT: Enter the Provider that this Order is being

Transferred From.

DESCRIPTION: This is the Provider that this Order is

being Transferred From.

100.011,.03 TRANSFERRED TO 0;3 POINTER TO NEW PERSON FILE (#200)

(Required)

LAST EDITED: JUN 01, 2017

HELP-PROMPT: Enter in the Provider that this Order is

being Transferred To.

DESCRIPTION: This is the Provider to which this Order is

being Transferred.

RECORD INDEX: EPRTRDT (#1332) REGULAR IR

LOOKUP & SORTING WHOLE FILE (#100)

Short Descr: Index on TRANSFERRED TO Provider and

TRANSFER DATE/TIME

Description: This index will be used as part of the new

ORQ3 EN Remote Procedure Call to speedily

pick up all Orders which have already been

transferred from the original Ordering

Provider to one or more New Providers in

the case where a New Provider also happens

to need his or her Orders transferred to a

third New Provider, then this index can

then be conducive to this type of scenario.

It works similar to EPRACDT Index but will

utilize TRANSFERRED TO Provider & TRANSFER

DATE/TIME to quickly bring up Orders for

additional Re-Assignment (Transfer).

Set Logic: S ^OR(100,"EPRTRDT",X(1),X(2),DA(1),DA)=""

Kill Logic: K ^OR(100,"EPRTRDT",X(1),X(2),DA(1),DA)

Whole Kill: K ^OR(100,"EPRTRDT")

X(1): TRANSFERRED TO (100.011,.03) (Subscr 1)

(forwards)

X(2): TRANSFER DATE/TIME (100.011,.01) (Subscr

2) (forwards)

100.011,.04 TRANSFER USER 0;4 POINTER TO NEW PERSON FILE (#200)

(Required)

LAST EDITED: MAY 04, 2017

HELP-PROMPT: Enter in the User that performed this Order

Transfer.

DESCRIPTION: This is the User who performed the Order

Transfer.

FILES POINTED TO FIELDS

NEW PERSON (#200) ORDER TRANSFERS:TRANSFERRED FROM (#.02)

TRANSFERRED TO (#.03)

TRANSFER USER (#.04)

#### System Features

Please refer to the RSD using this link:

<http://DNS.URL/pasdocs/analysis/20150609_CPRS%20Enhancements%20Phase%20I_BRD.zip>

##### NSR # 20130504 – Provider Role Change Enhancement

###### OR3P453

| Routines | Activities | | | |
| --- | --- | --- | --- | --- |
| **Routine Name** | OR3P453 | | | |
| **Enhancement Category** | New | Modify | Delete | No Change |
| **RTM** |  | | | |
| **Related Options** | N/A | | | |

| Related Routines | Routines “Called By” | | | Routines “Called” | | |
| --- | --- | --- | --- | --- | --- | --- |
|  | Called by Installation of OR\*3.0\*453 by Installing User | | | %ZTLOAD, DIK, DIR, XLFDT, XMD, XPDUTL | | |
| Routines | Activities | | | | | |
| **Data Dictionary (DD) References** | N/A | | | | | |
| **Related Protocols** | N/A | | | | | |
| **Related Integration Control Registrations (ICRs)** | N/A | | | | | |
| **Data Passing** | Input | Output Reference | Both | | Global Reference | Local |
| **Input Attribute Name and Definition** | Name:  Definition: | | | | | |
| **Output Attribute Name and Definition** | Name:  Definition: | | | | | |

| Current Logic |
| --- |
| N/A |

| Modified Logic (Changes are in bold) |
| --- |
| **OR3P453 ;SLC/RBD - Post Install 453 ;Aug 14, 2018@09:15**  **;;3.0;ORDER ENTRY/RESULTS REPORTING;\*\*453\*\*;Dec 17, 1997**  **;**  **ENV ; ensure that user understands what is about to happen with creation**  **; of New Style cross reference.**  **N DIR,DIRUT,DTOUT,DUOUT,Y**  **S XPDABORT=""**  **W !!,$C(7),"\*\*\*\*\*\* Creation of New Style 'EPRACDT' Cross Reference \*\*\*\*\*\*",!**  **W !,"This will loop through the ORDER ACTIONS Multiple of the entire ORDER"**  **W !,"File (#100) and create the 'EPRACDT' New Style Cross Reference based"**  **W !,"on the PROVIDER & DATE/TIME ORDERED sub-fields.",!**  **W !,"WARNING: Once you agree to create this, you should let it run until"**  **W !,"it has finished completely !!",!**  **S DIR("A")="Are you sure you want to do this"**  **S DIR("A",1)="You are about to create the index which could take quite awhile."**  **S DIR("B")="NO"**  **S DIR(0)="Y"**  **D ^DIR**  **I $D(DIRUT)!$D(DTOUT)!$D(DUOUT)!(Y'=1) W !!,"Ok, I am stopping the install." S XPDABORT=1 Q**  **E W !,"Ok, let's continue!",!**  **I XPDABORT="" K XPDABORT**  **Q**  **;**  **POST ; Post-Install for OR\*3.0\*453**  **; This will assist with EPRACDT cross reference creation along with**  **; Creation of new EPRACDT Indices**  **N ORMSG**  **S ORMSG(1)="This patch will create a new New Style cross reference"**  **S ORMSG(2)="called 'EPRACDT' which will be at the ORDER file level"**  **S ORMSG(3)="but on PROVIDER & DATE/TIME ORDERED sub-fields of the"**  **S ORMSG(4)="ORDER ACTIONS Multiple."**  **S ORMSG(5)=" "**  **S ORMSG(6)="Creation of 'EPRACDT' will now go forward in the"**  **S ORMSG(7)="Background."**  **S ORMSG(8)=" "**  **S ORMSG(9)="You will be given a TaskMan task # to check on or,"**  **S ORMSG(10)="alternately, you can check your mail on MailMan for a"**  **S ORMSG(11)="message expressing Completion of this Task with"**  **S ORMSG(12)="appropriate details."**  **S ORMSG(13)=" "**  **S ORMSG(14)="Note Install of this Patch cannot be considered"**  **S ORMSG(15)="Complete unless and until this Task is Completed."**  **S ORMSG(16)=" "**  **S ORMSG(17)="Note also that the Status of the 'EPRACDT' Creation"**  **S ORMSG(18)="can be checked by requesting IT to run 'D CHECK^OR3P453'"**  **S ORMSG(19)="at the Command Prompt."**  **S ORMSG(20)=" "**  **D BMES^XPDUTL(.ORMSG)**  **I $D(^XTMP("OR3P453","START")) D**  **. D MES^XPDUTL("Task to Create 'EPRACDT' Already Begun "\_$$HTE^XLFDT(^XTMP("OR3P453","START"))\_".")**  **. D MES^XPDUTL("")**  **I $D(^XTMP("OR3P453","FINISH")) D Q**  **. D MES^XPDUTL("...and Completed "\_$$HTE^XLFDT(^XTMP("OR3P453","FINISH"))\_".")**  **. D MES^XPDUTL("")**  **I $D(^XTMP("OR3P453","STOPPED")) D G SKPQUIT**  **. D MES^XPDUTL("...and Stopped "\_$$HTE^XLFDT(^XTMP("OR3P453","STOPPED"))\_".")**  **. D MES^XPDUTL(""),MES^XPDUTL("...Resuming 'EPRACDT' Creation."),MES^XPDUTL("")**  **Q:$D(^XTMP("OR3P453"))**  **SKPQUIT ;**  **S ZTRTN="SETXREF^OR3P453",ZTIO="",ZTDTH=$H**  **S ZTDESC="Creation of New Style X-Ref 'EPRACDT' in ORDER file" D ^%ZTLOAD**  **I $G(ZTSK) D MES^XPDUTL("Task #"\_ZTSK\_" queued to start "\_$$HTE^XLFDT($G(ZTSK("D")))) I 1**  **E D MES^XPDUTL("\*\*\*\*\* UNABLE TO QUEUE CREATION OF 'EPRACDT' ORDER FILE X-REF \*\*\*\*\*")**  **K ZTRTN,ZTIO,ZTDTH,ZTDESC,ZTSK**  **Q**  **;**  **SETXREF ; Set new EPRACDT New Style cross reference for old data**  **N CNT,CNT2,DA,DIK,LASTREC,ORIEN,STOP,XTMPCNT,XTMPMSG,ZTREQ**  **S U="^" S STOP=0 I $G(^XTMP("OR3P453","STOPPED"))]"" D G RESUME**  **. K ^XTMP("OR3P453","STOPPED") S XTMPCNT=$O(^XTMP("OR3P453"," "),-1)**  **K ^XTMP("OR3P453")**  **S ^XTMP("OR3P453",0)=$$FMADD^XLFDT($$DT^XLFDT(),90)**  **S ^XTMP("OR3P453","START")=$H**  **S ^XTMP("OR3P453","RECTOTAL")=$P($G(^OR(100,0)),U,4)**  **S XTMPCNT=0**  **S XTMPCNT=XTMPCNT+1**  **S XTMPMSG="Creation of 'EPRACDT' X-Ref for ORDER file Started "**  **S XTMPMSG=XTMPMSG\_$$HTE^XLFDT(^XTMP("OR3P453","START"))\_"."**  **S ^XTMP("OR3P453",XTMPCNT)=XTMPMSG**  **S XTMPCNT=XTMPCNT+1,^XTMP("OR3P453",XTMPCNT)=" "**  **RESUME ; Possibly resume here if previously stopped**  **S LASTREC=$G(^XTMP("OR3P453","LAST\_REC\_PROCESSED"))**  **S ORIEN=$P(LASTREC,U,1),CNT=$P(LASTREC,U,2)**  **I ORIEN="" S ORIEN=" ",CNT=0 K ^OR(100,"EPRACDT")**  **F S ORIEN=$O(^OR(100,ORIEN),-1) Q:'ORIEN D I STOP Q**  **. S DIK="^OR(100,"\_ORIEN\_",8,",DIK(1)=".01^EPRACDT",DA(1)=ORIEN D ENALL^DIK**  **. S CNT=CNT+1,^XTMP("OR3P453","LAST\_REC\_PROCESSED")=ORIEN\_U\_CNT**  **. I '(CNT#50000) D I STOP Q**  **.. F CNT2=1:1:3600 H 1 S STOP=$$REQ2STOP() I STOP Q**  **. S STOP=$$REQ2STOP() I STOP Q**  **I STOP Q**  **S XTMPMSG="Creation of 'EPRACDT' X-Ref Completed."**  **S XTMPCNT=XTMPCNT+1,^XTMP("OR3P453",XTMPCNT)=XTMPMSG**  **S XTMPCNT=XTMPCNT+1,^XTMP("OR3P453",XTMPCNT)=" "**  **S ^XTMP("OR3P453","FINISH")=$H**  **S XTMPMSG="Background Task Finished "**  **S XTMPMSG=XTMPMSG\_$$HTE^XLFDT(^XTMP("OR3P453","FINISH"))\_"."**  **S XTMPCNT=XTMPCNT+1,^XTMP("OR3P453",XTMPCNT)=XTMPMSG**  **;**  **; Send Mail to installer to notify of completion**  **S XMSUB="OR\*3.0\*453 post install has run to completion."**  **S XMDUZ="Patch OR\*3.0\*453"**  **S XTMPCNT=0**  **XRFLOOP S XTMPCNT=$O(^XTMP("OR3P453",XTMPCNT)) G:XTMPCNT'?1N.N FIN**  **S ^TMP($J,"OR3P453",XTMPCNT,0)=^XTMP("OR3P453",XTMPCNT)**  **G XRFLOOP**  **;**  **FIN S XMTEXT="^TMP($J,""OR3P453"","**  **S XMY(DUZ)="" D ^XMD K ^TMP($J,"OR3P453") S ZTREQ="@"**  **K XMDUZ,XMSUB,XMTEXT,XMY**  **Q**  **;**  **REQ2STOP() ;**  **; Check for task stop request**  **; Returns 1 if stop request made.**  **N STATUS,X**  **S STATUS=0**  **I '$D(ZTQUEUED) Q 0**  **S X=$$S^%ZTLOAD()**  **I X D ;**  **. S STATUS=1**  **. S X=$$S^%ZTLOAD("Received shutdown request")**  **;**  **I STATUS S ^XTMP("OR3P453","STOPPED")=$H**  **Q STATUS**  **;**  **CHECK ; Check the Status of the Task by looking at XTMP information**  **S U="^" N PCNT,RECTOTAL**  **I '$D(^XTMP("OR3P453")) D Q**  **. D MES^XPDUTL("Task to Create 'EPRACDT' has either not started or has")**  **. D MES^XPDUTL("automatically purged from the system.")**  **. D MES^XPDUTL("")**  **I $D(^XTMP("OR3P453","START")) D**  **. D MES^XPDUTL("Task to Create 'EPRACDT' Begun "\_$$HTE^XLFDT(^XTMP("OR3P453","START"))\_".")**  **. D MES^XPDUTL("")**  **I $D(^XTMP("OR3P453","FINISH")) D Q**  **. D MES^XPDUTL("...and Completed "\_$$HTE^XLFDT(^XTMP("OR3P453","FINISH"))\_".")**  **. D MES^XPDUTL("")**  **I $D(^XTMP("OR3P453","STOPPED")) D**  **. D MES^XPDUTL("...and Stopped "\_$$HTE^XLFDT(^XTMP("OR3P453","STOPPED"))\_".")**  **. D MES^XPDUTL("")**  **S CNT=$P($G(^XTMP("OR3P453","LAST\_REC\_PROCESSED")),U,2)**  **I +$G(CNT)'>0 Q**  **S RECTOTAL=$G(^XTMP("OR3P453","RECTOTAL"))**  **S PCNT=$P(((CNT/RECTOTAL)\*100),".",1)**  **D MES^XPDUTL("...Currently, "\_PCNT\_"% of Records have been Processed.")**  **D MES^XPDUTL("")**  **Q**  **;** |

###### ORB3 Routine

| Routines | Activities | | | |
| --- | --- | --- | --- | --- |
| **Routine Name** | ORB3 | | | |
| **Enhancement Category** | New | Modify | Delete | No Change |
| **RTM** |  | | | |
| **Related Options** | N/A | | | |

| Related Routines | Routines “Called By” | Routines “Called” |
| --- | --- | --- |
|  | Called by GMRCP, LR7ORB3, OCXOCMPG, OCXOZ0I, OCXOZ0J, OCXOZ0K, OCXOZ0L, OCXOZ0M, OCXOZ0N, OCXOZ0O, OCXOZ0P, OCXOZ0Q, OCXOZ0R, OCXOZ0S, OCXOZ0T, OCXOZ0Y, OCXOZ0Z, OCXOZ11, OCXOZ12, OCXOZ13, OLDCP, ORAREN, ORB3LAB, ORB3TIM1, ORB3TIM2, ORB3UTL, ORBSMART, ORKCHK, ORTSKLPS, ORWDAL32, ORX3, PSOPKIV1, PXRMNTFY, RAO7PC4, RAUTL00, RAUTL1, RAUTL19, WVSNOMED, WVTDALRT | DGCV, ORB31, ORB3FN, ORB3REG, ORB3SPEC, ORBSMART, ORQOR2, ORQPTQ1, SCAPMC, SCMCMHTC, SDUTL3, VADPT, XLFDT, XLFSTR, XPAR, XQALBUTL, XQALERT, |

| Routines | Activities | | | | |
| --- | --- | --- | --- | --- | --- |
| **Data Dictionary (DD) References** | ^OR(100, ^ORD(100.9 | | | | |
| **Related Protocols** | N/A | | | | |
| **Related Integration Control Registrations (ICRs)** | 1362 | | | | |
| **Data Passing** | Input | Output Reference | Both | Global Reference | Local |
| **Input Attribute Name and Definition** | Name: ORN  Definition: IEN into OE/RR NOTIFICATIONS file (#100.9)  Name: ORBDFN  Definition: IEN into PATIENT file (#2)  Name: ORNUM  Definition: IEN into ORDER file (#100)  Name: ORBADUZ  Definition: Array of Recipient DUZs.  Name: ORBPMSG  Definition: Message Text  Name: ORBPDATA  Definition: Identifies the package the Notification is based on. | | | | |
| **Output Attribute Name and Definition** | Name:  Definition: | | | | |

| Current Logic |
| --- |
| USER ;should USER (ORBDUZ) be a recip  D USER^ORB3USER(.XQA,ORBDUZ,ORN,.ORBU,.ORBUI,ORBDFN,+$G(ORNUM))  I $D(ORFORCE(ORBDUZ)) S XQA(ORBDUZ)=""  Q |

| Modified Logic (Changes are in bold) |
| --- |
| USER ;should USER (ORBDUZ) be a recip  **; RBD OR\*3.0\*453 Intercept User (Provider) to receive alert to see if it permanently routes to another User (Provider)**  **; Then check if that User can receive Alerts**  **N ORTRDAT,ORTRNUM,ORTRREC,ORTRREC2 I +$G(ORNUM)>0 D**  **. S ORTRDAT=$O(^OR(100,ORNUM,11,"B",$$NOW^XLFDT()),-1) I +ORTRDAT>0 D**  **.. S ORTRNUM=$O(^OR(100,ORNUM,11,"B",ORTRDAT,""),-1) I +ORTRNUM>0 D**  **... S ORTRREC2=$G(^OR(100,ORNUM,11,ORTRNUM,0)) I ORTRREC2]"" D**  **.... S:ORTRNUM=1 ORTRREC=ORTRREC2**  **.... S:ORTRNUM'=1 ORTRREC=$G(^OR(100,ORNUM,11,1,0))**  **.... I $P(ORTRREC,U,2)=ORBDUZ,$P(ORTRREC2,U,3) S ORBDUZ=$P(ORTRREC2,U,3)**  D USER^ORB3USER(.XQA,ORBDUZ,ORN,.ORBU,.ORBUI,ORBDFN,+$G(ORNUM))  I $D(ORFORCE(ORBDUZ)) S XQA(ORBDUZ)=""  Q |

###### ORCSAVE Routine

| Routines | Activities | | | |
| --- | --- | --- | --- | --- |
| **Routine Name** | ORCSAVE | | | |
| **Enhancement Category** | New | Modify | Delete | No Change |
| **RTM** |  | | | |
| **Related Options** | N/A | | | |

| Related Routines | Routines “Called By” | Routines “Called” |
| --- | --- | --- |
|  | OR3C100,  ORCACT2, ORCACT4,  ORCDFH, ORCDFHO,  ORCDFHTF, ORCDLG,  ORCMED, ORCONV3,  ORCSEND, ORCSEND2,  ORCSEND3, ORMBLDSD, ORMEVNT1,  ORMFH, ORMFH1,  ORMGMRC, ORMLR,  ORMLR1, ORMPS,  ORMPS2, ORMPS3,  ORMRA, ORMSD, ORMVBEC,  ORWD, ORWDFH,  ORWDX, ORWDXA,  ORWDXM, ORWDXR,  VEJDVGR1 | DIC, DIQ, HMPOR, ORCD, ORCSAVE1, ORMBLDOR, OROCAPI1, ORWDPS1, XLFDT |

| Routines | Activities | | | | |
| --- | --- | --- | --- | --- | --- |
| **Data Dictionary (DD) References** | ^DD(100, ^DIC(9.4, ^OR(100, ^ORD(100.02, ^ORD(100.05, ^ORD(100.7, ^ORD(101.41 | | | | |
| **Related Protocols** | N/A | | | | |
| **Related Integration Control Registrations (ICRs)** | N/A | | | | |
| **Data Passing** | Input | Output Reference | Both | Global Reference | Local |
| **Input Attribute Name and Definition** | Name: ORDIALOG  Definition: Array of Dialog Values  Name: ORDG  Definition: Display Group  Name: ORPKG  Definition: Package  Name: ORCAT  Definition: Patient Class  Name: OREVENT  Definition: OE/RR Patient Event  Name: ORDUZ  Definition: Who Entered Order  Name: ORLOG  Definition: When Order Entered’  Name: CODE  Definition: Action Code  Name: DA  Definition: IEN of ORDER file (#100)  Name: PROV  Definition: Action Provider  Name: REASON  Definition: Reason for Action  Name: WHEN  Definition: When Action Performed  Name: WHO  Definition: Who Performed Action  Name: DLG  Definition: Order Dialog | | | | |
| **Output Attribute Name and Definition** | Name: ORIFN  Definition: Order Number  Name: NEXT  Definition: Next Action Number  Name: ORPIFN  Definition: IFN of New Parent Order | | | | |

| Current Logic |
| --- |
| EN1, ACTION & SET Tags affected:  EN1 S ^OR(100,ORIFN,0)=ORIFN\_U\_ORVP\_U\_U\_$G(ORNP)\_U\_+ORDIALOG\_";ORD(101.41,^  "\_USR\_U\_LOG\_U\_U\_U\_LOC\_U\_DG\_U\_CATG\_U\_TRSPEC\_U\_PKG\_U\_U\_SIGNREQD\_U\_$G(OREVENT)\_U\_$G(ORAPPT)  S ^OR(100,ORIFN,3)=LOG\_"^90^"\_STS\_U\_$S($G(ORIT):ORIT\_";ORD(101.41,",1:"  ")\_U\_$G(ORDIALOG("PREV"))\_"^^1^^^^"\_TYPE  S ^OR(100,ORIFN,8,0)="^100.008DA^1^1",^OR(100,ORIFN,8,1,0)=LOG\_"^NW^"\_$  G(ORNP)\_U\_$S(SIGNREQD:2,1:3)\_"^^^^^^^^"\_NATR\_U\_USR\_"^1^"\_STS,^OR(100,ORIFN,8,"C"  ,"NW",1)=""  S ^OR(100,"AF",LOG,ORIFN,1)=""  ACTION(CODE,DA,PROV,REASON,WHEN,WHO) ; -- save new action  N NEXT,TOTAL,HDR,LAST,X,PAT,DGRP,SIG,NATR,TXT S DA=+DA  Q:'$D(^OR(100,DA,0)) 0 Q:$G(CODE)'?2U 0  S:'$G(WHEN) WHEN=+$E($$NOW^XLFDT,1,12) S:'$G(WHO) WHO=DUZ  S NATR=+$O(^ORD(100.02,"C","E",0)) ;assume Elec Entered until changed  S PAT=$P(^OR(100,DA,0),U,2),DGRP=$P(^(0),U,11),SIG=$P(^(0),U,16),X=+$P(  $G(^(3)),U,7),HDR=$G(^(8,0))  S:X'>0 X=1 S TXT=$P($G(^OR(100,DA,8,X,0)),U,14) ;current actn's txt ptr  S:HDR="" HDR="^100.008DA^^" S TOTAL=+$P(HDR,U,4)  S LAST=$O(^OR(100,DA,8,"C",CODE,"?"),-1) I LAST D  . S X=$G(^OR(100,DA,8,LAST,0)) Q:$P(X,U,15)'=11 Q:$P(X,U,4)'=2  . S NEXT=LAST I PAT,$P(X,U) D ; kill old xref entries  . . K:DGRP ^OR(100,"ACT",PAT,(9999999-$P(X,U)),DGRP,DA,NEXT)  . . K ^OR(100,"AC",PAT,(9999999-$P(X,U)),DA,NEXT),^OR(100,"AS",PAT,(999  9999-$P(X,U)),DA,NEXT),^OR(100,"AF",$P(X,U),DA,NEXT)  S:'$G(NEXT) NEXT=$O(^OR(100,DA,8,"?"),-1)+1,TOTAL=TOTAL+1  S ^OR(100,DA,8,NEXT,0)=WHEN\_U\_CODE\_U\_$G(PROV)\_U\_$S(SIG:2,1:3)\_"^^^^^^^^  "\_NATR\_U\_WHO\_U\_TXT\_"^11",^OR(100,DA,8,"C",CODE,NEXT)=""  S ^OR(100,"AF",WHEN,DA,NEXT)=""  SET(DLG) ; -- Create new parent for order set ORDIALOG  ; Returns ORPIFN = ifn of new parent order for set  ;  Q:'$G(ORVP) Q:'$G(DLG) N OR0,PKG,NOW,CATG,STS,ORLOC,TRSPEC,X  S OR0=$G(^ORD(101.41,DLG,0)) Q:OR0="" S ORPIFN=$$NEXTIFN Q:'ORPIFN  S PKG=$O(^DIC(9.4,"C","OR",0)),CATG=$S($$INPT^ORCD:"I",1:"O"),STS=$S($G  (OREVENT):10,1:11),NOW=$S($G(ORSLOG):ORSLOG,1:+$E($$NOW^XLFDT,1,12))  I $G(OREVENT) S ORLOC="",TRSPEC=""  S ^OR(100,ORPIFN,0)=ORPIFN\_U\_ORVP\_U\_U\_$G(ORNP)\_U\_DLG\_";ORD(101.41,^"\_DU  Z\_U\_NOW\_U\_U\_U\_ORLOC\_U\_U\_CATG\_U\_TRSPEC\_U\_PKG\_"^^^"\_$G(OREVENT),^(3)=NOW\_"^90^"\_STS\_U\_$S($G(ORIT):ORIT\_"ORD(101.41,",1:"")\_"^^^1^^^^0^^"\_+$P(OR0,U,6)  S ^OR(100,ORPIFN,8,0)="^100.008DA^1^1",^(1,0)=NOW\_"^NW^"\_$G(ORNP)\_"^^^^  ^^^^^^"\_DUZ\_"^^"\_STS,^OR(100,ORPIFN,8,"C","NW",1)="",^OR(100,"AF",NOW,ORPIFN,1)="" |

| Modified Logic (Changes are in bold) |
| --- |
| EN1, ACTION & SET Tags affected:  EN1 S ^OR(100,ORIFN,0)=ORIFN\_U\_ORVP\_U\_U\_$G(ORNP)\_U\_+ORDIALOG\_";ORD(101.41,^  "\_USR\_U\_LOG\_U\_U\_U\_LOC\_U\_DG\_U\_CATG\_U\_TRSPEC\_U\_PKG\_U\_U\_SIGNREQD\_U\_$G(OREVENT)\_U\_$G  (ORAPPT)  S ^OR(100,ORIFN,3)=LOG\_"^90^"\_STS\_U\_$S($G(ORIT):ORIT\_";ORD(101.41,",1:"  ")\_U\_$G(ORDIALOG("PREV"))\_"^^1^^^^"\_TYPE  S ^OR(100,ORIFN,8,0)="^100.008DA^1^1",^OR(100,ORIFN,8,1,0)=LOG\_"^NW^"\_$  G(ORNP)\_U\_$S(SIGNREQD:2,1:3)\_"^^^^^^^^"\_NATR\_U\_USR\_"^1^"\_STS,^OR(100,ORIFN,8,"C","NW",1)=""  S ^OR(100,"AF",LOG,ORIFN,1)=""  **; RBD OR\*3.0\*453 Add setting of EPRACDT index explicitly**  **I $G(ORNP)]"" S ^OR(100,"EPRACDT",ORNP,LOG,ORIFN,1)=""**  ACTION(CODE,DA,PROV,REASON,WHEN,WHO) ; -- save new action  N NEXT,TOTAL,HDR,LAST,X,PAT,DGRP,SIG,NATR,TXT S DA=+DA  Q:'$D(^OR(100,DA,0)) 0 Q:$G(CODE)'?2U 0  S:'$G(WHEN) WHEN=+$E($$NOW^XLFDT,1,12) S:'$G(WHO) WHO=DUZ  S NATR=+$O(^ORD(100.02,"C","E",0)) ;assume Elec Entered until changed  S PAT=$P(^OR(100,DA,0),U,2),DGRP=$P(^(0),U,11),SIG=$P(^(0),U,16),X=+$P($G(^(3)),U,7),HDR=$G(^(8,0))  S:X'>0 X=1 S TXT=$P($G(^OR(100,DA,8,X,0)),U,14) ;current actn's txt ptr  S:HDR="" HDR="^100.008DA^^" S TOTAL=+$P(HDR,U,4)  S LAST=$O(^OR(100,DA,8,"C",CODE,"?"),-1) I LAST D  . S X=$G(^OR(100,DA,8,LAST,0)) Q:$P(X,U,15)'=11 Q:$P(X,U,4)'=2  . S NEXT=LAST I PAT,$P(X,U) D ; kill old xref entries  . . K:DGRP ^OR(100,"ACT",PAT,(9999999-$P(X,U)),DGRP,DA,NEXT)  . . K ^OR(100,"AC",PAT,(9999999-$P(X,U)),DA,NEXT),^OR(100,"AS",PAT,(999  9999-$P(X,U)),DA,NEXT),^OR(100,"AF",$P(X,U),DA,NEXT)  **. . I $P(X,U,3) K ^OR(100,"EPRACDT",$P(X,U,3),$P(X,U),DA,NEXT) ; RBD**  **OR\*3.0\*453 Handle Kill of EPRACDT index as AF index is done**  S:'$G(NEXT) NEXT=$O(^OR(100,DA,8,"?"),-1)+1,TOTAL=TOTAL+1  S ^OR(100,DA,8,NEXT,0)=WHEN\_U\_CODE\_U\_$G(PROV)\_U\_$S(SIG:2,1:3)\_"^^^^^^^^  "\_NATR\_U\_WHO\_U\_TXT\_"^11",^OR(100,DA,8,"C",CODE,NEXT)=""  S ^OR(100,"AF",WHEN,DA,NEXT)="" **I $G(PROV)]"" S ^OR(100,"EPRACDT",PROV,WHEN,DA,NEXT)="" ; RBD OR\*3.0\*453 Handle Set of EPRACDT index**  SET(DLG) ; -- Create new parent for order set ORDIALOG  ; Returns ORPIFN = ifn of new parent order for set  ;  Q:'$G(ORVP) Q:'$G(DLG) N OR0,PKG,NOW,CATG,STS,ORLOC,TRSPEC,X  S OR0=$G(^ORD(101.41,DLG,0)) Q:OR0="" S ORPIFN=$$NEXTIFN Q:'ORPIFN  S PKG=$O(^DIC(9.4,"C","OR",0)),CATG=$S($$INPT^ORCD:"I",1:"O"),STS=$S($G  (OREVENT):10,1:11),NOW=$S($G(ORSLOG):ORSLOG,1:+$E($$NOW^XLFDT,1,12))  I $G(OREVENT) S ORLOC="",TRSPEC=""  S ^OR(100,ORPIFN,0)=ORPIFN\_U\_ORVP\_U\_U\_$G(ORNP)\_U\_DLG\_";ORD(101.41,^"\_DU  Z\_U\_NOW\_U\_U\_U\_ORLOC\_U\_U\_CATG\_U\_TRSPEC\_U\_PKG\_"^^^"\_$G(OREVENT),^(3)=NOW\_"^90^"\_ST  S\_U\_$S($G(ORIT):ORIT\_"ORD(101.41,",1:"")\_"^^^1^^^^0^^"\_+$P(OR0,U,6)  S ^OR(100,ORPIFN,8,0)="^100.008DA^1^1",^(1,0)=NOW\_"^NW^"\_$G(ORNP)\_"^^^^  ^^^^^^"\_DUZ\_"^^"\_STS,^OR(100,ORPIFN,8,"C","NW",1)="",^OR(100,"AF",NOW,ORPIFN,1)=""  **I $G(ORNP)]"" S ^OR(100,"EPRACDT",ORNP,NOW,ORPIFN,1)="" ; RBD OR\*3.0\***  **453 Set new EPRACDT index explicitly as AF index is done** |

###### ORELR5 Routine

| Routines | Activities | | | |
| --- | --- | --- | --- | --- |
| **Routine Name** | ORELR5 | | | |
| **Enhancement Category** | New | Modify | Delete | No Change |
| **RTM** |  | | | |
| **Related Options** | Check Lab orders from file 69 to 100 [ORE LAB ORDERS CHECK 69=>100] | | | |

| Related Routines | Routines “Called By” | Routines “Called” |
| --- | --- | --- |
|  | ORELR | DICN, ORCSAVE2, ORDD100, ORU, ORUTL1, ORX, XLFDT |

| Routines | Activities | | | | |
| --- | --- | --- | --- | --- | --- |
| **Data Dictionary (DD) References** | ^DD(100, ^LAB(60, ^LR, ^LRO(69, ^OR(100, | | | | |
| **Related Protocols** | N/A | | | | |
| **Related Integration Control Registrations (ICRs)** | N/A | | | | |
| **Data Passing** | Input | Output Reference | Both | Global Reference | Local |
| **Input Attribute Name and Definition** | Name: LRDFN  Definition: IEN into PATIENT file (#2)  Name: ORAFIX  Definition: Flag to Fix Database or not | | | | |
| **Output Attribute Name and Definition** | Name:  Definition: | | | | |

| Current Logic |
| --- |
| LOOP(LRDFN,ORAFIX) ;Loop on patient  I '$D(^LR(LRDFN,0)) D WRT(,,,"No entry in ^LR("\_LRDFN,ORAFIX) S DCNT=DC  NT+1,TOTCNT=TOTCNT+1 K:ORAFIX ^LRO(69,"D",LRDFN) Q  Q:$P(^LR(LRDFN,0),"^",2)'=2 ;Not in patient file.  S DFN=$P(^LR(LRDFN,0),"^",3)  Q:'$D(^LRO(69,"D",$G(LRDFN)))  N LRODT,LRSN,LRTI,LRTST,LRENT,X,X0,X3,ORX1,ORX2,ORIFN,X8O  S LRODT=0 F S LRODT=$O(^LRO(69,"D",LRDFN,LRODT)) Q:'LRODT S LRSN=0 F  S LRSN=$O(^LRO(69,"D",LRDFN,LRODT,LRSN)) Q:'LRSN D  . I '$D(^LRO(69,LRODT,1,LRSN,0)) D WRT(LRODT,LRSN,,"D X-ref invalid",ORAFIX) S DCNT=DCNT+1,TOTCNT=TOTCNT+1 K:ORAFIX ^LRO(69,"D",LRDFN,LRODT,LRSN) Q  . S X=^LRO(69,LRODT,1,LRSN,0),LRENT=$P(X,"^",5)  . S LRTI=0 F S LRTI=$O(^LRO(69,LRODT,1,LRSN,2,LRTI)) Q:LRTI<1 S X0=^(LRTI,0) D  .. S LRTST=+X0,ORIFN=$P(X0,"^",7)  .. I ORIFN D  ... I '$D(^OR(100,ORIFN)) D WRT(LRODT,LRSN,LRTI,"Broken pointer to 100:  "\_ORIFN,ORAFIX) S F100CNT=F100CNT+1,TOTCNT=TOTCNT+1 S:ORAFIX $P(^LRO(69,LRODT,1,LRSN,2,LRTI,0),"^",7)="P" Q ;P=purged  ... S X=^OR(100,ORIFN,0),X3=$G(^(3))  ... I DFN'=+$P(X,"^",2) D WRT(LRODT,LRSN,LRTI,"Patient mismatch:"\_ORIFN  \_"<"\_$P(X3,"^",3)\_">") S PTCNT=PTCNT+1,TOTCNT=TOTCNT+1 Q  ... D STATUS(LRODT,LRSN,LRTI,X0,ORAFIX)  ... I LRENT,$P(X,"^",7)>$S($P($P(X,"^",8),".",2):$P(X,"^",8),1:$P(X,"^",8)\_".2359") D  .... S ORX1=$$FMADD^XLFDT($P(X,"^",7),,,30),ORX2=$$FMADD^XLFDT($P(X,"^"  ,7),,,-30)  .... I LRENT<ORX2!(LRENT>ORX1) S ENTCNT=ENTCNT+1,TOTCNT=TOTCNT+1 I ORAFIX D  ..... S $P(^OR(100,ORIFN,0),"^",7)=LRENT  ..... I $P(X,"^",7)=+$G(^OR(100,ORIFN,8,1,0)) S X8O=$G(^(0)) D  ...... N DI,DIC,DIE,DA,DR,D0,DQ,DISYS  ...... I $P(X,"^",11) K ^OR(100,"ACT",$P(X,"^",2),9999999-+X8O,$P(X,"^",11),ORIFN,1)  ...... K ^OR(100,"AC",$P(X,"^",2),9999999-+X8O,ORIFN,1),^OR(100,"AF",+X  8O,ORIFN,1),^OR(100,"AS",$P(X,"^",2),9999999-(+X8O),ORIFN,1)  ...... I $P(X8O,"^",16)=+X8O K ^OR(100,"AR",$P(X,"^",2),9999999-(+X8O),  ORIFN,1) S ^OR(100,"AR",$P(X,"^",2),9999999-LRENT,ORIFN,1)="",$P(^OR(100,ORIFN,8  ,1,0),"^",16)=LRENT  ...... S $P(^OR(100,ORIFN,8,1,0),"^")=LRENT,^OR(100,"AF",LRENT,ORIFN,1)=""  ...... D S1^ORDD100(ORIFN,1,"",LRENT),SET^ORDD100(ORIFN,1),ACT1^ORDD100  A(ORIFN,1)  Q |

| Modified Logic (Changes are in bold) |
| --- |
| LOOP(LRDFN,ORAFIX) ;Loop on patient  I '$D(^LR(LRDFN,0)) D WRT(,,,"No entry in ^LR("\_LRDFN,ORAFIX) S DCNT=DCNT+1,TOTCNT=TOTCNT+1 K:ORAFIX ^LRO(69,"D",LRDFN) Q  Q:$P(^LR(LRDFN,0),"^",2)'=2 ;Not in patient file.  S DFN=$P(^LR(LRDFN,0),"^",3)  Q:'$D(^LRO(69,"D",$G(LRDFN)))  N LRODT,LRSN,LRTI,LRTST,LRENT,X,X0,X3,ORX1,ORX2,ORIFN,X8O  S LRODT=0 F S LRODT=$O(^LRO(69,"D",LRDFN,LRODT)) Q:'LRODT S LRSN=0 F  S LRSN=$O(^LRO(69,"D",LRDFN,LRODT,LRSN)) Q:'LRSN D  . I '$D(^LRO(69,LRODT,1,LRSN,0)) D WRT(LRODT,LRSN,,"D X-ref invalid",ORAFIX) S DCNT=DCNT+1,TOTCNT=TOTCNT+1 K:ORAFIX ^LRO(69,"D",LRDFN,LRODT,LRSN) Q  . S X=^LRO(69,LRODT,1,LRSN,0),LRENT=$P(X,"^",5)  . S LRTI=0 F S LRTI=$O(^LRO(69,LRODT,1,LRSN,2,LRTI)) Q:LRTI<1 S X0=^(LRTI,0) D  .. S LRTST=+X0,ORIFN=$P(X0,"^",7)  .. I ORIFN D  ... I '$D(^OR(100,ORIFN)) D WRT(LRODT,LRSN,LRTI,"Broken pointer to 100:"\_ORIFN,ORAFIX) S F100CNT=F100CNT+1,TOTCNT=TOTCNT+1 S:ORAFIX $P(^LRO(69,LRODT,1,LRSN,2,LRTI,0),"^",7)="P" Q ;P=purged  ... S X=^OR(100,ORIFN,0),X3=$G(^(3))  ... I DFN'=+$P(X,"^",2) D WRT(LRODT,LRSN,LRTI,"Patient mismatch:"\_ORIFN\_"<"\_$P(X3,"^",3)\_">") S PTCNT=PTCNT+1,TOTCNT=TOTCNT+1 Q  ... D STATUS(LRODT,LRSN,LRTI,X0,ORAFIX)  ... I LRENT,$P(X,"^",7)>$S($P($P(X,"^",8),".",2):$P(X,"^",8),1:$P(X,"^",8)\_".2359") D  .... S ORX1=$$FMADD^XLFDT($P(X,"^",7),,,30),ORX2=$$FMADD^XLFDT($P(X,"^",7),,,-30)  .... I LRENT<ORX2!(LRENT>ORX1) S ENTCNT=ENTCNT+1,TOTCNT=TOTCNT+1 I ORAFIX D  ..... S $P(^OR(100,ORIFN,0),"^",7)=LRENT  ..... I $P(X,"^",7)=+$G(^OR(100,ORIFN,8,1,0)) S X8O=$G(^(0)) D  ...... N DI,DIC,DIE,DA,DR,D0,DQ,DISYS  ...... I $P(X,"^",11) K ^OR(100,"ACT",$P(X,"^",2),9999999-+X8O,$P(X,"^",11),ORIFN,1)  ...... K ^OR(100,"AC",$P(X,"^",2),9999999-+X8O,ORIFN,1),^OR(100,"AF",+X  8O,ORIFN,1),^OR(100,"AS",$P(X,"^",2),9999999-(+X8O),ORIFN,1)  **...... I $P(X8O,"^",3) K ^OR(100,"EPRACDT",$P(X8O,"^",3),+X8O,ORIFN,1) ; RBD OR\*3.0\*453 Clean up new EPRACDT index also**  ...... I $P(X8O,"^",16)=+X8O K ^OR(100,"AR",$P(X,"^",2),9999999-(+X8O),ORIFN,1) S ^OR(100,"AR",$P(X,"^",2),9999999-LRENT,ORIFN,1)="",$P(^OR(100,ORIFN,8,1,0),"^",16)=LRENT  ...... S $P(^OR(100,ORIFN,8,1,0),"^")=LRENT,^OR(100,"AF",LRENT,ORIFN,1)=""  **...... S ^OR(100,"EPRACDT",$P(X8O,"^",3),LRENT,ORIFN,1)="" ; RBD OR\*3.0\*453 Reset EPRACDT index also**  ...... D S1^ORDD100(ORIFN,1,"",LRENT),SET^ORDD100(ORIFN,1),ACT1^ORDD100A(ORIFN,1)  Q |

###### ORQ2 Routine

| Routines | Activities | | | |
| --- | --- | --- | --- | --- |
| **Routine Name** | ORQ2 | | | |
| **Enhancement Category** | New | Modify | Delete | No Change |
| **RTM** |  | | | |
| **Related Options** | N/A | | | |

| Related Routines | Routines “Called By” | Routines “Called” |
| --- | --- | --- |
|  | ORCXPND2, ORRCOR, ORWOR, SAVEWOR | DIQ, DIWP, ICDEX, ORCD, ORCHECK, ORHLESC, ORQ12, ORQ20, ORQ21, ORX8, VADPT, XLFSTR, XPAR |

| Routines | Activities | | | | |
| --- | --- | --- | --- | --- | --- |
| **Data Dictionary (DD) References** | ^DIC(45.7, ^GMR(123, ^OR(100, ^ORD(100.01, ^ORD(101.41, ^SC | | | | |
| **Related Protocols** | N/A | | | | |
| **Related Integration Control Registrations (ICRs)** | 4203 | | | | |
| **Data Passing** | Input | Output Reference | Both | Global Reference | Local |
| **Input Attribute Name and Definition** | Name: ORIFN  Definition: IEN into ORDER file (#100) | | | | |
| **Output Attribute Name and Definition** | Name: ORY  Definition: Array of Order Detail information | | | | |

| Current Logic |
| --- |
| DETAIL(ORY,ORIFN) ; -- Returns details of order ORIFN in ORY(#)  N X,X2,I,CNT,ORDIALOG,OR0,OR3,OR6,SEQ,ITEM,PRMT,MULT,FIRST,TITLE,INST,D  IWL,DIWR,DIWF,ACTION,VAIN,ORIGVIEW,ORNMSP,ORYT,ORCRACT  ; p 471 added ORCACT to use action if passed in  S CNT=0,ORCRACT=+$P(ORIFN,";",2),ORIFN=+ORIFN,OR0=$G(^OR(100,ORIFN,0)),  OR3=$G(^(3)),OR6=$G(^(6))  S ORNMSP=$$NMSP^ORCD($P(OR0,U,14))  ; p 471 if no action, default to CURRENT ACTION..or if not $O(<The action subfile>, -1)  S:'ORCRACT ORCRACT=$S(+$P(OR3,U,7):+$P(OR3,U,7),1:+$O(^OR(100,ORIFN,8,"  :"),-1))  K @ORY,ORYT S ORIGVIEW=1 D TEXT^ORQ12(.ORYT,+ORIFN\_";"\_ORCRACT,80) ;CurrTx  M @ORY=ORYT ;Move text to global  S I=0 F CNT=1:1 S I=$O(ORYT(I)) Q:I'>0 D:$D(IORVON) SETVIDEO(I,1,$L(OR  YT(I)),IORVON,IORVOFF)  S CNT=CNT+1,@ORY@(CNT)=" " ;blank  D1 I $O(^OR(100,+ORIFN,2,0)) D  . S CNT=CNT+1,@ORY@(CNT)="Sub Orders:"  . D:$D(IOUON) SETVIDEO(CNT,1,11,IOUON,IOUOFF)  . N IFN S IFN=0  . F S IFN=+$O(^OR(100,+ORIFN,2,IFN)) Q:IFN<1 I $D(^OR(100,IFN,0)) D SUB(IFN)  . S CNT=CNT+1,@ORY@(CNT)=" " ;blank  I $P(OR3,U,9),$D(^OR(100,+$P(OR3,U,9),0)) D  . S CNT=CNT+1,@ORY@(CNT)="Parent Order:"  . D:$D(IOUON) SETVIDEO(CNT,1,12,IOUON,IOUOFF)  . D SUB(+$P(OR3,U,9))  . S CNT=CNT+1,@ORY@(CNT)=" " ;blank  I $P(OR3,U,11)=1,$P(OR3,U,5) D ;Changed - show previous order  . S CNT=CNT+1,@ORY@(CNT)="Previous Order:"  . D:$D(IOUON) SETVIDEO(CNT,1,15,IOUON,IOUOFF) ;prev order original text  . N ORZ,I,ORIGVIEW S ORIGVIEW=2 D TEXT^ORQ12(.ORZ,+$P(OR3,U,5),55)  . S CNT=CNT+1,@ORY@(CNT)=" Order Text: "\_$G(ORZ(1))  . S I=1 F S I=$O(ORZ(I)) Q:I'>0 S CNT=CNT+1,@ORY@(CNT)=$$REPEAT^XLFST  R(" ",24)\_$G(ORZ(I))  D2 S CNT=CNT+1,@ORY@(CNT)="Activity:"  D:$D(IOUON) SETVIDEO(CNT,1,9,IOUON,IOUOFF)  S DIWL=1,DIWR=64,DIWF="C64",ORI=0 K ^UTILITY($J,"W")  F S ORI=$O(^OR(100,ORIFN,8,ORI)) Q:ORI'>0 S ACTION=$G(^(ORI,0)) D ACT^ORQ20 |

| Modified Logic (Changes are in bold) |
| --- |
| DETAIL(ORY,ORIFN) ; -- Returns details of order ORIFN in ORY(#)  N X,X2,I,CNT,ORDIALOG,OR0,OR3,OR6,SEQ,ITEM,PRMT,MULT,FIRST,TITLE,INST,D  IWL,DIWR,DIWF,ACTION,VAIN,ORIGVIEW,ORNMSP,ORYT,ORCRACT  **N ORTRANS ; OR\*3\*453 rbd Var needed for ORDER TRANSFERS logic below**  ; p 471 added ORCACT to use action if passed in  S CNT=0,ORCRACT=+$P(ORIFN,";",2),ORIFN=+ORIFN,OR0=$G(^OR(100,ORIFN,0)),  OR3=$G(^(3)),OR6=$G(^(6))  S ORNMSP=$$NMSP^ORCD($P(OR0,U,14))  ; p 471 if no action, default to CURRENT ACTION..or if not $O(<The action subfile>, -1)  S:'ORCRACT ORCRACT=$S(+$P(OR3,U,7):+$P(OR3,U,7),1:+$O(^OR(100,ORIFN,8,":"),-1))  K @ORY,ORYT S ORIGVIEW=1 D TEXT^ORQ12(.ORYT,+ORIFN\_";"\_ORCRACT,80) ;CurrTx  M @ORY=ORYT ;Move text to global  S I=0 F CNT=1:1 S I=$O(ORYT(I)) Q:I'>0 D:$D(IORVON) SETVIDEO(I,1,$L(ORYT(I)),IORVON,IORVOFF)  S CNT=CNT+1,@ORY@(CNT)=" " ;blank  D1 I $O(^OR(100,+ORIFN,2,0)) D  . S CNT=CNT+1,@ORY@(CNT)="Sub Orders:"  . D:$D(IOUON) SETVIDEO(CNT,1,11,IOUON,IOUOFF)  . N IFN S IFN=0  . F S IFN=+$O(^OR(100,+ORIFN,2,IFN)) Q:IFN<1 I $D(^OR(100,IFN,0)) D SUB(IFN)  . S CNT=CNT+1,@ORY@(CNT)=" " ;blank  I $P(OR3,U,9),$D(^OR(100,+$P(OR3,U,9),0)) D  . S CNT=CNT+1,@ORY@(CNT)="Parent Order:"  . D:$D(IOUON) SETVIDEO(CNT,1,12,IOUON,IOUOFF)  . D SUB(+$P(OR3,U,9))  . S CNT=CNT+1,@ORY@(CNT)=" " ;blank  I $P(OR3,U,11)=1,$P(OR3,U,5) D ;Changed - show previous order  . S CNT=CNT+1,@ORY@(CNT)="Previous Order:"  . D:$D(IOUON) SETVIDEO(CNT,1,15,IOUON,IOUOFF) ;prev order original text  . N ORZ,I,ORIGVIEW S ORIGVIEW=2 D TEXT^ORQ12(.ORZ,+$P(OR3,U,5),55)  . S CNT=CNT+1,@ORY@(CNT)=" Order Text: "\_$G(ORZ(1))  . S I=1 F S I=$O(ORZ(I)) Q:I'>0 S CNT=CNT+1,@ORY@(CNT)=$$REPEAT^XLFSTR(" ",24)\_$G(ORZ(I))  D2 S CNT=CNT+1,@ORY@(CNT)="Activity:"  D:$D(IOUON) SETVIDEO(CNT,1,9,IOUON,IOUOFF)  S DIWL=1,DIWR=64,DIWF="C64",ORI=0 K ^UTILITY($J,"W")  F S ORI=$O(^OR(100,ORIFN,8,ORI)) Q:ORI'>0 S ACTION=$G(^(ORI,0)) D ACT^ORQ20  **; RBD OR\*3.0\*453 Add ORDER TRANSFERS information**  **S ORI=0 F S ORI=$O(^OR(100,ORIFN,11,ORI)) Q:ORI'>0 D**  **. I ORI=1 D**  **.. S CNT=CNT+1,@ORY@(CNT)=" ",CNT=CNT+1,@ORY@(CNT)="Reassignments (Transfers):"**  **. S ORTRANS=$G(^OR(100,ORIFN,11,ORI,0))**  **. S CNT=CNT+1,@ORY@(CNT)="Transferred from "\_$$USER^ORQ20($P(ORTRANS,U,2))\_" to "\_$$USER^ORQ20($P(ORTRANS,U,3))**  **. S CNT=CNT+1,@ORY@(CNT)=" by "\_$$USER^ORQ20($P(ORTRANS,U,4))\_" Effective From "\_$$DATE^ORQ20($P(ORTRANS,U))**  **I $D(^OR(100,ORIFN,11)) S CNT=CNT+1,@ORY@(CNT)=" "** |

###### ORQ3 Routine

| Routines | Activities | | | |
| --- | --- | --- | --- | --- |
| **Routine Name** | ORQ3 | | | |
| **Enhancement Category** | New | Modify | Delete | No Change |
| **RTM** |  | | | |
| **Related Options** | N/A | | | |

| Related Routines | Routines “Called By” | Routines “Called” |
| --- | --- | --- |
|  | Called by CPRS GUI via RPC calls ORQ3 EN, ORQ3 XFER, ORQ3 AUTHUSR & ORQ3 CHKPATCH. | DICN, ORQOR2, VADPT, XLFDT |

| Routines | Activities | | | | |
| --- | --- | --- | --- | --- | --- |
| **Data Dictionary (DD) References** | ^OR(100, ^VA(200 | | | | |
| **Related Protocols** | N/A | | | | |
| **Related Integration Control Registrations (ICRs)** | N/A | | | | |
| **Data Passing** | Input | Output Reference | Both | Global Reference | Local |
| **Input Attribute Name and Definition** | Name: ORPRIEN  Definition: IEN into NEW PERSON file (#200)  Name: ORDT1  Definition: Starting Date to Search From  Name: ORDT2  Definition: Ending Date to Search To  Name: LST  Definition: List of Order IENs, Transferred From Provider, Transferred To Providers, User Requesting Transfer, and Date/Times of Transfer  Name: ORPATCH  Definition: String containing “OR\*3.0\*453” to make sure this patch is installed. | | | | |
| **Output Attribute Name and Definition** | Name: ORY  Definition: Return Array of Patients & Orders information; also used to return a “1” or “0” to denote if a user can access Provider Role Change from within CPRS or not. Also used to return if OR\*3.0\*453 has been installed or not (may be used in future to check if other Provider Role Change patches are installed).  Name: RESULTS  Definition: Return Array of Orders that have Successfully and Unsuccessfully had Transfer/Re-Assignment attempted; if Unsuccessful then Error Message will also be returned. | | | | |

| Current Logic |
| --- |
| N/A |

| Modified Logic (Changes are in bold) |
| --- |
| **ORQ3 ;SLC/RBD - Provider Role Change APIs ;Mar 04, 2018@08:52**  **;;3.0;ORDER ENTRY/RESULTS REPORTING;\*\*453\*\*;Dec 17, 1997**  **;**  **Q**  **;**  **EN(ORY,ORPRIEN,ORDT1,ORDT2) ; Get Qualifying Orders for Provider**  **;**  **; This RPC allows retrieval of Patients and Orders for which the**  **; Provider is the Ordering Provider and the Orders are Signed**  **;**  **; Input:**  **; ORY is the name of the return array**  **; ORPRIEN identifies the IEN of the Provider to retrieve Orders for**  **; ORDT1 identifies the start date to start looking from**  **; ORDT2 identifies the end date to stop looking once reached**  **; Output:**  **; Global ^TMP("ORPATRTN",$J,n) where n represents each record found**  **; starting with record "1"**  **; Contains data for Patient Name ^ Patient IEN ^ Order IEN ^**  **; Order Status ^ Order Date**  **;**  **N CNT,DFN,OR0,ORIEN,OROBJ,ORPRXFRD,ORPTNM,ORSIGNED,ORTMPDT,ORXFERNM,VADM**  **S:'$D(U) U="^" K ^TMP("ORPTINFO",$J),^TMP("ORPATRTN",$J)**  **S ORY=$NA(^TMP("ORPATRTN",$J))**  **; Loop through Providers that are Ordering Providers for Orders that have been Signed**  **S ORTMPDT=ORDT1,ORDT2=ORDT2\_".2359"**  **F S ORDT1=$O(^OR(100,"EPRACDT",ORPRIEN,ORDT1)) Q:ORDT1>ORDT2 Q:ORDT1="" D**  **. S ORIEN="" F S ORIEN=$O(^OR(100,"EPRACDT",ORPRIEN,ORDT1,ORIEN)) Q:ORIEN="" D**  **.. S OR0=$G(^OR(100,ORIEN,0)) Q:ORIEN'["" Q:$$ORDERER^ORQOR2(ORIEN)'=ORPRIEN ; skip if not Ord. Prv.**  **.. S ORSIGNED=($P($G(^OR(100,+ORIEN,8,1,0)),U,4)'=2) Q:'ORSIGNED ; skip if Order not Signed**  **.. ; if Old Provider already Transferred, skip**  **.. S ORPRXFRD=0,ORXFERNM=0**  **.. F S ORXFERNM=$O(^OR(100,ORIEN,11,ORXFERNM)) Q:ORXFERNM="" Q:ORXFERNM'?1N.N D**  **... I $P($G(^OR(100,ORIEN,11,ORXFERNM,0)),U,2)=ORPRIEN D**  **.... S ORPRXFRD=1**  **.. Q:ORPRXFRD=1 S OROBJ=$P(OR0,U,2) Q:OROBJ'["DPT("**  **.. S DFN=+OROBJ D OERR^VADPT S ORPTNM=$G(VADM(1)) Q:ORPTNM=""**  **.. S ^TMP("ORPTINFO",$J,ORPTNM,DFN,ORIEN)=""**  **; Now order through Transferred To Provider index for situation where he/she has**  **; to Transfer Order to a third Provider (or fourth, etc.) ... only allow last entry**  **; from ORDER TRANSFERS multiple to be used though.**  **S ORDT1=ORTMPDT**  **F S ORDT1=$O(^OR(100,"EPRTRDT",ORPRIEN,ORDT1)) Q:ORDT1>ORDT2 Q:ORDT1="" D**  **. S ORIEN="" F S ORIEN=$O(^OR(100,"EPRTRDT",ORPRIEN,ORDT1,ORIEN)) Q:ORIEN="" D**  **.. S ORXFERNM=$O(^OR(100,ORIEN,11,"B"),-1) Q:$P($G(^OR(100,ORIEN,11,ORXFERNM,0)),U,3)’=**  **ORPRIEN**  **.. S OR0=$G(^OR(100,ORIEN,0)) Q:ORIEN'[""**  **.. S OROBJ=$P(OR0,U,2) Q:OROBJ'["DPT("**  **.. S DFN=+OROBJ D OERR^VADPT S ORPTNM=$G(VADM(1)) Q:ORPTNM=""**  **.. S ^TMP("ORPTINFO",$J,ORPTNM,DFN,ORIEN)=""**  **; Put in Patient Name, Patient IEN, & Order IEN order to return to GUI**  **S CNT=0,ORPTNM="" F S ORPTNM=$O(^TMP("ORPTINFO",$J,ORPTNM)) Q:ORPTNM="" D**  **. S DFN="" F S DFN=$O(^TMP("ORPTINFO",$J,ORPTNM,DFN)) Q:DFN="" D**  **.. S ORIEN="" F S ORIEN=$O(^TMP("ORPTINFO",$J,ORPTNM,DFN,ORIEN)) Q:ORIEN="" D**  **... S CNT=CNT+1**  **... S ^TMP("ORPATRTN",$J,CNT)=ORPTNM\_U\_DFN\_U\_ORIEN\_U\_$P($$STATUS^ORQOR2**  **(ORIEN),U,2)\_U\_$$FMTE^XLFDT($P($G(^OR(100,ORIEN,0)),U,7))**  **K ^TMP("ORPTINFO",$J)**  **Q**  **;**  **XFER(RESULTS,LST) ; Transfer Orders to New Providers**  **;**  **; This RPC allows the Transferring from one Provider to another**  **; which will create an entry into each order in a List into**  **; the ORDER TRANSFERS multiple**  **;**  **; Input:**  **; LST(1..n) where each entry contains:**  **; ORIEN identifies the IEN of the Order to create a**  **; Reassignment (Transfer) for**  **; ORPRIEN1 identifies the Transferring From Provider**  **; ORPRIEN2 identifies the Transferring To Provider**  **; ORUSER identifies the User requesting the Transfer**  **; ORDTTM identifies the Transfer Date/Time**  **; Output:**  **; Global ^TMP("ORORDRTN",$J,n) where n represents each record**  **; starting with record "1"**  **; Contains data for Order IEN ^ Success Flag**  **; ^ Error Message if Unsuccessful**  **; Where Success Flag = 0 if Unsuccessful or 1 if Successful**  **;**  **N CNT,DA,DIC,DIE,DR,ORDTTM,ORIEN,ORPRIEN1,ORPRIEN2,ORUSER,X,Y**  **S:'$D(U) U="^" K ^TMP("ORORDRTN",$J)**  **S RESULTS=$NA(^TMP("ORORDRTN",$J))**  **S CNT=0 F S CNT=$O(LST(CNT)) Q:CNT="" D**  **. S ORIEN=$P(LST(CNT),U,1),ORPRIEN1=$P(LST(CNT),U,2)**  **. S ORPRIEN2=$P(LST(CNT),U,3),ORUSER=$P(LST(CNT),U,4)**  **. S ORDTTM=$P(LST(CNT),U,5)**  **. I ORIEN']"" S ^TMP("ORORDRTN",$J,CNT)=ORIEN\_U\_0\_U\_"Order IEN Blank" Q**  **. I '$D(^OR(100,ORIEN)) D Q**  **.. S ^TMP("ORORDRTN",$J,CNT)=ORIEN\_U\_0\_U\_"Invalid Order IEN"**  **. I ORPRIEN1']"" D Q**  **.. S ^TMP("ORORDRTN",$J,CNT)=ORIEN\_U\_0\_U\_"From Provider IEN Blank"**  **. I '$D(^VA(200,ORPRIEN1)) D Q**  **.. S ^TMP("ORORDRTN",$J,CNT)=ORIEN\_U\_0\_U\_"From Provider IEN "\_ORPRIEN1\_" Invalid"**  **. I ORPRIEN2']"" D Q**  **.. S ^TMP("ORORDRTN",$J,CNT)=ORIEN\_U\_0\_U\_"To Provider IEN Blank"**  **. I '$D(^VA(200,ORPRIEN2)) D Q**  **.. S ^TMP("ORORDRTN",$J,CNT)=ORIEN\_U\_0\_U\_"To Provider IEN "\_ORPRIEN2\_" Invalid"**  **. I ORUSER']"" D Q**  **.. S ^TMP("ORORDRTN",$J,CNT)=ORIEN\_U\_0\_U\_"Transferring User IEN Blank"**  **. I '$D(^VA(200,ORUSER)) D Q**  **.. S ^TMP("ORORDRTN",$J,CNT)=ORIEN\_U\_0\_U\_"Transferring User IEN "\_ORUSER\_" Invalid"**  **. I ORDTTM'?7N1"."1.6N S ^TMP("ORORDRTN",$J,CNT)=ORIEN\_U\_0\_U\_"Invalid Transfer Date/Time" Q**  **. I ORDTTM<$$NOW^XLFDT() D Q**  **.. S ^TMP("ORORDRTN",$J,CNT)=ORIEN\_U\_0\_U\_"Transfer Date/Time Cannot be in the Past"**  **. L +^OR(100,ORIEN):0 I '$T D Q**  **.. S ^TMP("ORORDRTN",$J,CNT)=ORIEN\_U\_0\_U\_"Another user is editing this Order."**  **. S DIC="^OR(100,"\_ORIEN\_",11,",DA(1)=ORIEN,DIC(0)="L",X=ORDTTM**  **. S DIC("DR")=".02////"\_ORPRIEN1\_";.03////"\_ORPRIEN2**  **. S DIC("DR")=DIC("DR")\_";.04////"\_ORUSER D FILE^DICN**  **. I +Y<0 S ^TMP("ORORDRTN",$J,CNT)=ORIEN\_U\_0\_U\_"Save Unsuccessful" L -^OR(100,ORIEN) Q**  **. S ^TMP("ORORDRTN",$J,CNT)=ORIEN\_U\_1 L -^OR(100,ORIEN)**  **Q**  **;**  **AUTHUSR(ORY) ; Does user have permission to access Provider**  **; Utilities (Provider Role Change)**  **S ORY=0**  **I $D(^XUSEC("OR PU ACCESS",DUZ)) S ORY=1**  **Q**  **;**  **CHKPATCH(ORY,ORPATCH) ; Checks for KIDS Patch install**  **; RPC: ORQ3 CHKPATCH**  **; Input: ORPATCH - Patch Number (i.e. OR\*3.0\*453)**  **; Output: Returns "1^Patch Is Installed" on success;**  **; otherwise returns "0^Patch Is Not Installed"**  **N ORX**  **S ORX=$$PATCH^XPDUTL(ORPATCH)**  **S ORY=$S(ORX:"1^Patch Is Installed",1:"0^Patch Is Not Installed")**  **Q**  **;** |

##### Remote Procedure Call (RPC)

| RPCs | Activities | | |
| --- | --- | --- | --- |
| **Name** | 'ORQ3 AUTHUSR' | | |
| **TAG^RTN** | AUTHUSR^ORQ3 | | |
| **Input Parameters** | None – CPRS will already know the DUZ of the user logged in to it | | |
| **Results Array** | Single Value | Array | Word Processing |
| Global Array | Global Instance |  |
| **Description** | NSR20130504 – Provider Role Change  Returns “1” if user has the new SECURITY KEY called OR PU ACCESS, otherwise “0” | | |

| RPCs | Activities | | |
| --- | --- | --- | --- |
| **Name** | 'ORQ3 CHKPATCH' | | |
| **TAG^RTN** | CHKPATCH^ORQ3 | | |
| **Input Parameters** | ORPATCH – initially will hold value of “OR\*3.0\*453” to make sure this patch is installed but could have other future Provider Role Change patches as a parameter. | | |
| **Results Array** | Single Value | Array | Word Processing |
| Global Array | Global Instance |  |
| **Description** | NSR20130504 – Provider Role Change  Returns “1^Patch Is Installed” or “0^Patch Is Not Installed” | | |

| RPCs | Activities | | |
| --- | --- | --- | --- |
| **Name** | 'ORQ3 EN' | | |
| **TAG^RTN** | EN^ORQ3 | | |
| **Input Parameters** | ORPRIEN – IEN into NEW PERSON file (#200), ORDT1 – Start Date to begin Search for, ORDT2 – End Date to stop Search for | | |
| **Results Array** | Single Value | Array | Word Processing |
| Global Array | Global Instance |  |
| **Description** | NSR20130504 – Provider Role Change  Returns results in Patient Name, Patient IEN, Order IEN sort order with the following 5 fields:  Patient Name, IEN into PATIENT file (#2), IEN into ORDER file (#100), Status of the Order, e.g., ACTIVE, PENDING, etc., Date of Order | | |

| RPCs | Activities | | |
| --- | --- | --- | --- |
| **Name** | 'ORQ3 XFER' | | |
| **TAG^RTN** | XFER^ORQ3 | | |
| **Input Parameters** | LST(1..n) where each entry contains:  ORIEN – IEN of ORDER file (#100), ORPRIEN1 – IEN into NEW PERSON file (#200) representing Transferring From Provider, ORPRIEN2 – IEN into NEW PERSON file (#200) representing Transferring To Provider, ORUSER – IEN into NEW PERSON file (#200) representing User Requesting Transfer, & ORDTTM representing Date/Time of Transfer | | |
| **Results Array** | Single Value | Array | Word Processing |
| Global Array | Global Instance |  |
| **Description** | NSR20130504 – Provider Role Change  Returns results of ORDER TRANSFERS Multiple subrecord creation as IEN into ORDER file (#100), Success Flag “1” if Successful and “0” otherwise, & Error Message if Success Flag is “0” | | |

## Network Detailed Design

One of the design constraints for Provider Utilities is to utilize the existing network infrastructure. Therefore no procurement is expected.

## Security and Privacy

### Security

There is a new OR PU GUI VistA menu option of type Broker listed below with the RPC calls that it needs (including the ones newly created, ORQ3 AUTHUSR, ORQ3 EN, & ORQ3 XFER).

NAME: OR PU GUI MENU TEXT: CPRS Provider Utilities

TYPE: Broker (Client/Server) CREATOR: DAVIS,ROBERT B

DESCRIPTION: This option is required by the Kernel Broker to give access to

the RPCs used by the CPRS Provider Utilities.

TIMESTAMP OF PRIMARY MENU: 64509,48974

RPC: ORWU DT

RPC: ORQOR DETAIL

RPC: ORWU NEWPERS

RPC: ORQ3 EN

RPC: ORQ3 XFER

RPC: ORWU VERSRV

RPC: ORWU USERINFO

RPC: ORQ3 AUTHUSR

UPPERCASE MENU TEXT: CPRS PROVIDER UTILITIES

Also, a new SECURITY KEY has been added with its definition below.

NAME: OR PU ACCESS

DESCRIPTIVE NAME: Provider Utilities Access

DESCRIPTION: This security key will be used to control user access to the

Provider Utilities (Provider Role Change) off the Tools menu in CPRS.

Instructions for adding the ProviderUtilities.exe to the Tools menu within CPRS are outlined in the OR\*3.0\*453 patch description within FORUM. This is also part of its security.

### Privacy

This project will continue to use the existing VistA / CPRS privacy model.

## Service Oriented Architecture / ESS Detailed Design

CPRS is a legacy GUI application that provides a GUI front-end to the VistA system and is primarily used by physicians, nurses and other clinicians responsible for providing patient care. Provider Utilities is enhancing the existing CPRS system.

One note: CPRS’s architecture does not supply new services or consume services.

# External System Interface Design

This section documents the Provider Utilities Delphi GUI application.

## Interface Architecture

Provider Utilities uses existing CPRS architecture, to include the RPC broker and base 508 form.

### GUI Security

Access to this GUI application is restricted using multiple redundant mechanisms and checks.

* The application is not shown in the CPRS Tools menu unless a system admin or authorized entity has configured it to be present for a given user or user group.
* The application will not run without 2FA authentication.
* The application will not run unless a system admin or authorized entity has configured the user access key “OR PU GUI”.

## Interface Detailed Design

Provider Utilities uses existing CPRS conventions for communicating with the VistA server. See Section 6 for a full list of Vista RPCs used by the GUI application.

# Human-Machine Interface (GUI)

## Interface Design Rules

VA standards, Science Advisory & Coordinating Committee (SACC) and IBM Common User Interface (CUA). This GUI application is Section 508 compliant. Whenever possible, code and interface standards are shared (borrowed unchanged) from the CPRS application.

## Inputs

Mouse and keyboard, no special or novel input device is required. Drag & drop is supported in the primary window, but keyboard, menu and button alternatives are also available. Context menus are provided in most windows. Command keys can be displayed using the standard Windows mechanism of pressing the “ALT” key.

## Outputs

Provider Role Change (GUI) provides a single optional text report after order reassignment. The report containts patient sensitive information and must be handled appropriately by the user. The report is simple text that can be placed into the Windows clipboard for appropriate disposition by the user.

## Delphi Design Notes

### Compliance with GUI SAC, V1.0

The Delphi GUI application was designed to conform to the standards presented in the document “Graphical User Interface Standards And Conventions (GUI SAC)”, Version 1.0, February 2002. No hyperlink to this document could be located in the VA VDL.

### Windows Registry

This application does not directly utilize or modify the Windows Registry. It also does not create any local files.

### Action Lists

All actions available to end users are encapsulated in action list items, making them available for reuse from main menus, popup menus, buttons, shortcuts, etc. All enabling/disabling of GUI controls is controlled from tActionList.OnUpdate events.

### Section 508-Compliant Form Inheritance

All forms and dialogs using in this application are Section 508 compliant. All forms and dialogs inherit from a root ancestor of type TFormBase508Form or TFormBase508Dialog (borrowed from CPRS).

### Windows Help

A compiled HTML Windows help file is delivered with the application. Context-sensitive help is accessed from the Help menu or via standard keyboard shortcuts.

## Navigation Hierarchy

The following diagram illustrates the GUI navigation flow. The names shown are the window captions – not to be confused with the Delphi form names.

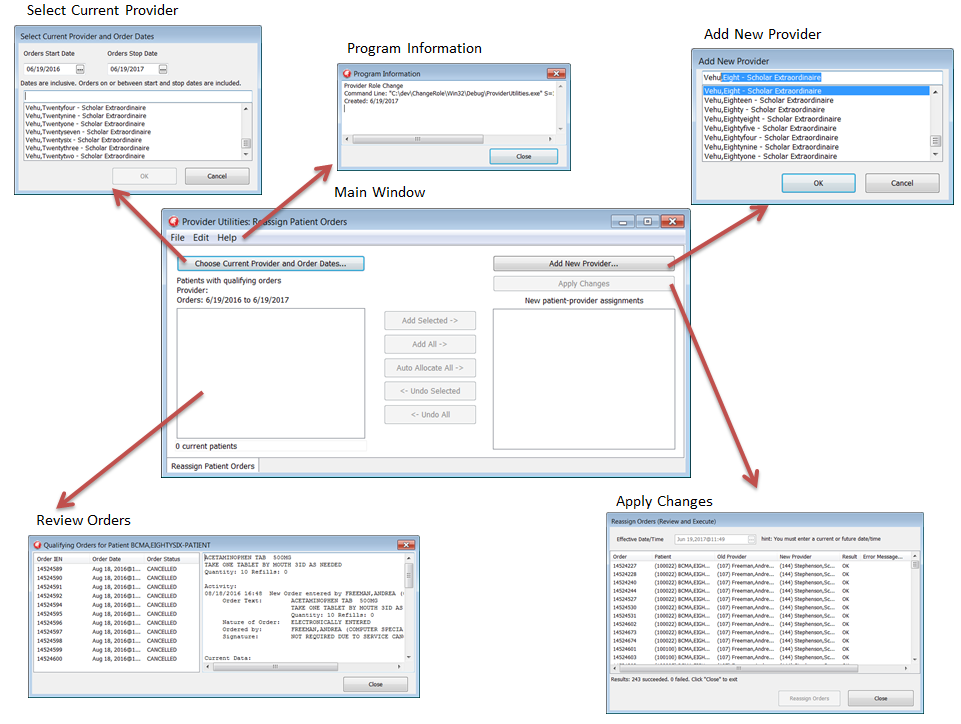


Figure : GUI Navigation Flow

### Main Form: “Provider Utilities: Reassign Patient Orders”

*This is the main window*. The user views and reassigns patients from current to new providers by dragging from left to right. This form utilizes a drag and drop GUI metaphor, supplemented by command buttons, to reassign patients from a current provider to one or more new providers.

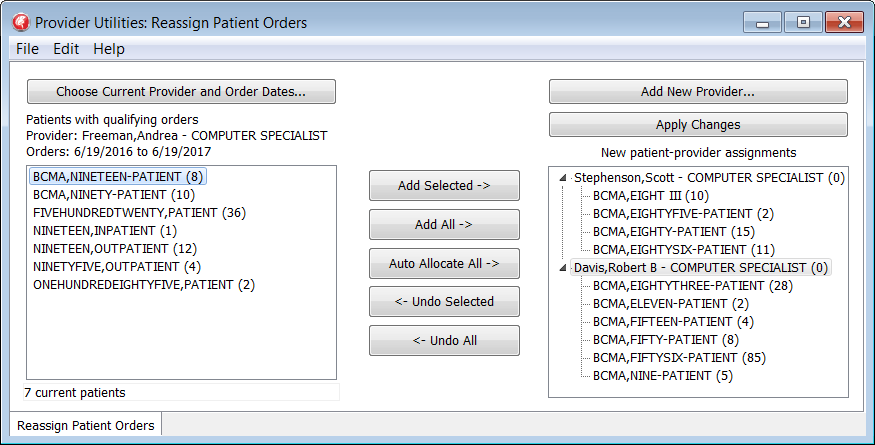


Figure : Main Form

#### Usage (optimal program flow)

The main window enables a 4-step process to reassign patients: select current provider, select new provider, move patients, review & execute.

* User opens the application and the main window appears
* User selects a current provider and order dates by clicking the button “Choose Current Provider and Order Dates”.
* User selects one or more new providers by clicking the button “Add New Provider”.
* User drags patients from the left list to the right list, dropping on the desired new provider. As an alternative to drag and drop, user can utilized the buttons at screen center to move patients.
* User clicks the button “Apply Changes” to review and execute the order reassignments.
* User exits the application or repeats this process for additional providers.

#### Specific functionality

The main window Delphi units are fReassignOrders.pas and fReassignOrders.dfm. All data used in the window is derived from the Delphi provider engine object.

The main window has this specific GUI functionality:

* **Selection**: The user can select one or more current patients at left using the mouse. The mouse supports Shift+click and Ctrl+ click to assist selecting multiple patients.
* **Selection**: The user can select on or more current patients at left using the keyboard. The user can select multiple patients by holding down the Shift key while using the down arrow key.
* **Selection**: A context (popup) menu will select or deselect all current patients.
* **Information:** A list of current patients is displayed at left.  
  Information: A list of new providers and assigned patients is displayed at right.
* **Information**: The main window always displays an updated patient count below the current patients list.
* **Information**: The main window always displays (above the current patient list) a caption for the currently selected provider, to include name and the order date range.
* **Information**: The order count for each current patient is shown in () for any patient in the left or right lists.
* **Information**: The user can “drill-down” into a patient’s orders by double–clicking a patient in the list at left. A context menu also provides this ability.
* **Action**: The button or edit menu item “*Choose Current Provider and Order Dates*” displays a modal dialog in which the user can enter the required data.
* **Action**: The button or edit menu item “*Add New Provider*” displays a modal dialog in which the user can select a new provider.
* **Action**: The button or edit menu item “*Apply Changes*” displays a modal dialog in which the user can review changes and optionally send them to the server (or cancel).
* **Action**: The button or edit menu item “*Add Selected*” assigns all selected current patients to the selected new provider. This action is enabled only if a patient is selected at left and a new provider is selected at right.
* **Action**: The button or edit menu item “*Add All*” assigns all patients at left to the selected new provider at right. This action is enabled only if a new provider is selected at right.
* **Action**: The button or edit menu item “*Auto Allocate All*” evenly distributes all patients at left between new providers at right.
* **Action**: The button or edit menu item “*Undo Selected*” will move all selected new patients at right back to the current patient list at left.
* **Action**: :The button or edit menu item “*Undo All*” will move all new patients at right to the current patient list at left.
* **Behavior**: Inappropriate actions (buttons, menu items) are disabled when appropriate.
* **Behavior**: Hot keys for all actions are automatically assigned. The user can view hot keys by pressing the “Alt” key.
* **Behavior**: If a patient is selected at right, the program will consider the new provider parent to be selected when enabling/disabling actions.

### Modal Form: “Select Current Provider”

In this modal form, the user selects a current provider, and also accepts or modifies the default order start date and stop date. The date range will be used to extract orders from the Vista database for the selected current provider. The dates are inclusive, meaning any order entered on or after the start date and before or on the end date will be processed.

The Delphi units for this form are fSelectProvider.pas and fSelectProvider.dfm. The default stop date is “now” as obtained from the Vista server. Provider names are taken from the server via the *ORWU NEWPERS* M command.

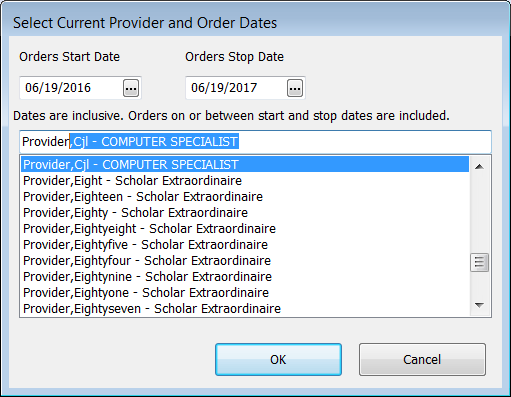


Figure : Select Current Provider

This form is reused to obtain the new provider. The only functional difference is that the date edit controls are hidden, as they are not relevant when selecting new providers.

### Modal Form: Qualifying Orders

When the user double-clicks a current patient, the application will display this modal form. The form displays a list of orders for the selected patient. In this form, the user can click any order at left and see an order detail at right. This form is informational only – there is no other functionality.

The Delphi units for this window are fPatientOrders.pas and fPatientOrders.dfm. Data for the right portion of this window is taken from the Vista M command *ORQOR DETAIL*. The list of orders at left is obtained from the Delphi ProviderEngine object.

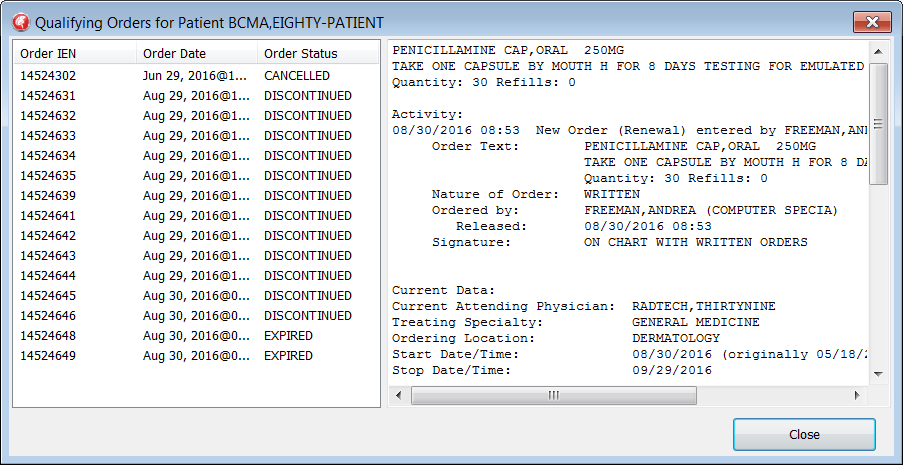


Figure : Qualifying Orders for Patient

### Modal Form: Reassign Orders (Review and Execute)

When the user clicks the button “Apply Changes” in the main form, this modal form appears. This enforces a “two click” design pattern that gives the user the opportunity to review the proposed changes and cancel if desired. Also, the user is required to modify or accept the default execution time, which is “now” as obtained from the Vista server. A future time can be entered – this is useful when a provider’s role change date is known in advance, and the order reassignment is prepared before that date. Not that a full date/time value is required.

The Delphi units for this window are fApplyChanges.pas and fApplyChanges.dfm. All data for this window is obtained from the ProviderEngine Delphi object.

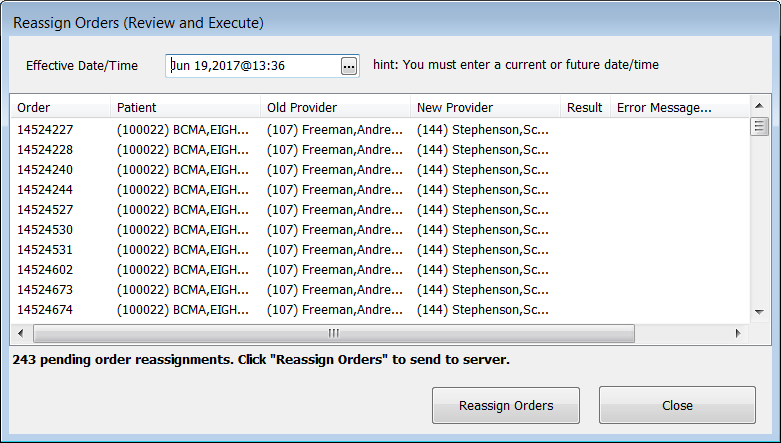


Figure : Reassign Orders

### Modal Form: About dialog

A standard CPRS-style About dialog is displayed from the Help menu.

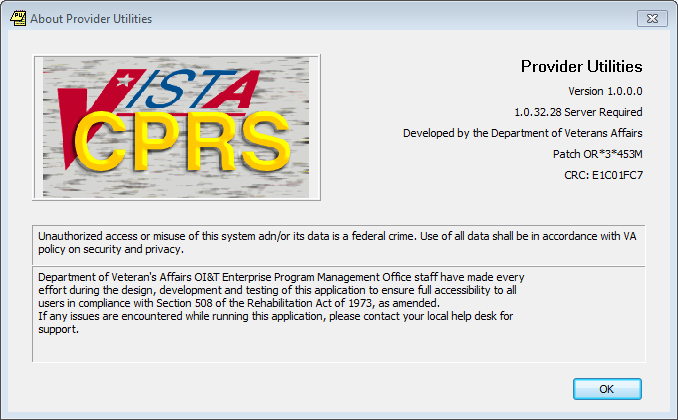


Figure : About Dialog

# Attachment A – Approval Signatures

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

The Business Sponsor and Project Manager are required to sign.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Signed: Date:

< Business Sponsor >

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Signed: Date:

< Project Manager >

1. Additional Information

Attach any addition information that supplements the design specification.

* 1. Identification of Technology and Standards

Identify the system and software which apply to the SDD, including: identification number(s), title(s), abbreviation(s), version number(s), and release number(s). Identify all standards (e.g., American National Standards Institute [ANSI], International Organization for Standardization [ISO], Institute of Electrical and Electronics Engineers [IEEE], etc.).

* 1. Constraining Policies, Directives and Procedures

Identify any constraints or requirements placed on this document by policies, directives, or procedures.

* 1. Requirements Traceability Matrix

Include an RTM that traces modules and data structures to the software requirements. A reference to the location of the RTM is also acceptable.

* 1. Packaging and Installation

Outline any special considerations for software packaging and installation.

* 1. Design Metrics

Describe all metrics to be used during the design activity.

Template Revision History

| Date | Version | Description | Author |
| --- | --- | --- | --- |
| June 2015 | 2.10 | Changed Heading 1 default setting to eliminate page break before | Process Management |
| May 2015 | 2.9 | Edited for Section 508 conformance and remediated with Common Look Office tool | Process Management |
| February 2015 | 2.8 | Incorporates revisions from PMAS Reform Lockdown; namely removing requirements for information that can be obtained from other PMAS authoritative sources. | Andrew Slawter, Office of Technology Strategies |
| September 2014 | 2.7 | Adds Enterprise Shared Services terms and requires AERB Compliance Certificate attachment. | Process Management |
| August 2014 | 2.6 | Signature block update authorized by AERB CR\_018934 | Process Management |
| March 2014 | 2.5 | Section 508 repairs to new version approved by AERB Chair approved | Process Management |
| August 2013 | 2.3 | Replaced the Service Architecture sub-section with new sub-sections for consumed and provided services. Also applied miscellaneous feedback from VA team. | ASD Enterprise Shared Services (ESS) Work Group |
| June 2013 | 1.3 | Upgraded to MS Office 2007-2010 format | Process Management |
| June 2013 | 1.2 | Address inconsistencies in Section 3, Conceptual Design, Correct headings | Process Management |
| March 2013 | 1.1 | Formatted to documentation standards and edited for Section 508 conformance | Process Management |
| January 2013 | 1.0 | Initial Document | PMAS Business Office |

See TOGAF® 9.1, Part III: ADM Guidelines & Techniques, Gap Analysis on TOGAF website at <http://pubs.opengroup.org/architecture/togaf9-doc/arch/chap27.html>