



PATIENT ADVOCATE TRACKING SYSTEM (PATS)

Systems Management Guide

April 2013

This page is left blank intentionally

Revision History

The following table displays the revision history for this document.

Date	Revision	Description	Author
04/11/07	Initial		Susan Bunker PATS Developers: Patrick Brady, Daryl Krauter, Adam Nisenbaum, Tami Winn Don Morgan, PM
6-2-09	1.1	Updated information pertaining to KAAJEE Upgrade	Padma Subbaraman Developer
10-30-09	1.2	Updated the formatting of the document. Changed the logo. Added the boilerplate ROC resolution text clarification.	Arsen Mikhailutsa, Developer
10-07-2010	1.3	Updated information on SDS, VistALink version and HSI name change	Arsen Mikhailutsa, Developer
08-12-2010	1.4	Removed link and exemplary connection information	Arsen Mikhailutsa, Developer
06-08-2011	1.5	Corrected the misspelling in the database job, added information regarding the root context.	Arsen Mikhailutsa, Developer
11-21-11	1.6	Updated formatting of the document, Updated Oracle version information	Arsen Mikhailutsa, Developer
04-01-2013	1.7	Added PAD servlet information detailing automatic ROC generation	Joshua Faulkner, Developer
04-04-2013	1.8	Updated document with ESE – HIS group rename.	Arsen Mikhailutsa, Developer

This page is left blank intentionally

Table of Contents

Introduction.....	1
1.0 PATS Application.....	3
1.1 Enterprise Archive	3
1.1.1 Application.xml.....	3
1.1.2 Weblogic-application.xml.....	3
1.1.3 APP-INF	3
1.1.4 Web Application.....	6
1.1.5 Ejb-jar Archive.....	9
1.2 HealtheVet Configuration Files	9
1.3 PATS Loggers.....	13
1.4 Exceptions	15
1.5 Service Imports.....	15
1.6 Requirement for Sending Notifications.....	17
1.7 Integration Agreements for use of Java Components.....	17
1.8 PAD Servlet ROC Creation	18
2.0 Java Enterprise Developer Workstation.....	20
2.1 Development Platform	20
2.2 Development Projects.....	20
2.3 Development Tools	21
2.3.1 ANT.....	21
2.3.2 XDoclet.....	21
2.3.3 Log4j.....	21
2.3.4 Libraries.....	21
2.3.5 Junit and Cactus	23
2.3.6 Rational XDE.....	23
2.3.7 Maven.....	23
2.3.8 PAD Tools	23
2.4 Services	25
3.0 Business Rules Implementation	26
3.1 Web Security	26
3.2 Ejb Method Permission Security.....	26
3.3 Transactions.....	26
3.4 ROC	26
3.5 Notification.....	26
3.6 Table Maintenance.....	26
3.7 Concurrency.....	26
3.8 Application Design Parameters.....	27
3.8.1 Timeout Parameters.....	27
3.8.2 Date Parameters.....	27
3.8.3 Report Instance Limits.....	27
3.8.4 Text Field Length	27
4.0 M VistA.....	27
4.1 Namespace.....	28
4.2 Routine Descriptions.....	28
4.3 Temporary Globals.....	29

4.4 Options	29
4.5 Remote Procedure Calls	30
4.6 External Relations	31
4.7 VistA Integration Agreements	33
4.7.1 Supported	33
4.7.2 Controlled Subscription	34
4.8 Online Documentation	35
4.9 Check Sum Values for Routines	35
4.10 Security and Keys	35
5.0 Database - Oracle	37
5.1 Database	37
5.2 Schemas	37
5.3 Users	37
5.4 Roles	38
5.5 Tablespaces	38
5.6 Tables	38
5.6.1 PATS Schema	38
5.6.2 PATSRPTS Schema	42
5.7 Field Information	44
5.8 Report Data Tables in PATSRPTS Schema	45
5.9 Procedures	45
5.9.1 PATS Schema	45
5.9.2 PATSRPTS Schema	51
5.10 Techniques used in PATS Procedures	54
5.11 Scheduled Jobs	55
5.12 Developer Workstation Setup	55
5.13 External Relations	57
5.14 Database Integration Agreements	57
6.0 Business Objects XI (BOEXI)	57
6.1 Repository and Central Management Console	58
6.1.1 Report Folder Structure	58
6.1.2 Standard Report Database Source	60
6.1.3 Users	60
6.1.4 Groups	61
6.1.5 Instance Limits	61
6.1.6 Security	61
6.1.7 Universe Connections	64
6.1.8 Universes	64
6.1.9 Licensing	65
6.2 WebIntelligence	66
6.2.1 PATS/WebLogic – WebIntelligence/Tomcat integration notes	66
6.2.2 Universe Connection	66
6.2.3 Universe	67
6.2.4 Universe Data Filtering	68
6.3 Business Objects Software	69
6.3.1 Crystal Reports XI	69
6.3.2 WebI Universe Designer XI	69
6.3.3 Central Management Console	70
6.3.4 InfoView	70
6.3.5 Web Intelligence	70

6.4 Multiple PATS Instances Sharing the Same BOEXI Repository	70
6.4.1 Standard Reports	71
6.4.2 Ad Hoc Reports	71
7.0 Rollup PATS Data to VSSC	73
7.1 Scheduled Job to Build Rollup Data	73
7.2 VSSC Fetches Rollup Data.....	73
7.3 Format of Rollup Data Records	73
7.3.1 ROC Main Data Records	73
7.3.2 Patient Data Records	74
7.3.3 Issue Multiple Records	75
7.3.4 Contacting Entity Records	75
7.3.5 Method of Contact Records	76
7.3.6 Patient Race Records	76
8.0 Troubleshooting PATS Reports	77

This page is left blank intentionally

Introduction

The Patient Advocate Tracking System (PATS) provides a way of documenting, tracking and communicating patient-related issues.

The *PATS Systems Management Guide* gives a technical description of PATS for supporting and maintaining the application. The intended audience of this guide is IRM, HSI, Product Support, and PIMS.

PATS is a web based application. Both the PATS application and the database are maintained at the HSI Data Center in Martinsburg.

Chapter Descriptions and Roles

The guide is divided into the following sections:

Chapter	Chapter Name	Intended Audience
1	PATS Application	Java developers and System Admin
2	Java Enterprise Developer Workstation	Java developers and System Admin
3	Business Rules	Business layer developer at HSI, Maintenance and Product Support
4	Database – M VistA	IRM staff, Maintenance, and Product Support
5	Database – Oracle	DBA at the HSI, Maintenance, and Product Support
6	Business Objects XI	HSI, Maintenance, and Product Support
7	Troubleshooting PATS Reports	HSI and Product Support

This page is left blank intentionally

1.0 PATS Application

PATS is designed to run on a J2EE 1.3 compliant application server running on a 1.4 JVM.

1.1 Enterprise Archive

PATS deploys as a Java enterprise archive named `PATS-APP-major.minor.revision.buildnumber.ear` containing the following elements.

1.1.1 Application.xml

This file exists in the META-INF directory per the Java enterprise 1.3 specifications. The contents are the modules and logical security roles of the PATS application.

Context-root `'/PATS'` in the META-INF/application.xml is the default context root specifier for the PATS application. This entry in the deployment descriptor can be updated appropriately based on the server context root requirements

1.1.2 Weblogic-application.xml

No special settings used in this deployment descriptor.

1.1.3 APP-INF

The classes directory contains configuration files updated at development time (see section 1.1.4.3 for a description):

- application.properties
- ApplicationResources.properties
- buildnumber.properties
- caipConfig.xml
- commands.properties
- pagenumber.properties
- patsweb.properties
- PersonLookupResources.properties
- URL.properties

The lib directory contains jar files for the PATS enterprise application:

Directory Name	Description
boconfig.jar	BOEXI
CAIP.jar	CAIP archive
cecore.jar	BOEXI

celib.jar	BOEXI
ceplugins.jar	BOEXI
Directory Name	Description
cereports.jar	BOEXI
cHSISSION.jar	BOEXI
ceutils.jar	BOEXI
cexsd.jar	BOEXI
commons-beanutils.jar	Apache Jakarta Commons
commons-collections-3.0.jar	Apache Jakarta Commons
commons-digester.jar	Apache Jakarta Commons
commons-discovery.jar	Apache Jakarta Commons
commons-httpclient-2.0.1.jar	Apache Jakarta Commons
commons-logging-1.0.3.jar	Apache Jakarta Commons
commons-resources.jar	Apache Jakarta Commons
commons-validator.jar	Apache Jakarta Commons
CorbaIDL.jar	BOEXI
ebus405.jar	BOEXI
gov.va.med.pats.common.jar	PATS
gov.va.med.pats.dao.jar	PATS
gov.va.med.pats.delegate.jar	PATS
gov.va.med.pats.report.jar	PATS
hivemind-1.1.1.jar	Apache Jakarta HiveMind
hivemind-jmx-1.1.1.jar	Apache Jakarta HiveMind

hivemind-lib-1.1.1.jar	Apache Jakarta HiveMind
jakarta-oro-2.0.8.jar	Apache Jakarta ORO
Directory Name	Description
javassist-3.0.jar	JBoss
jrecom.jar	BOEXI
jtools.jar	BOEXI
NamingDirectoryService-client.jar	NDS
PatientServiceR2.jar	PSC
rasapp.jar	BOEXI
rascore.jar	BOEXI
ReportPrinter.jar	BOEXI
ReportTemplate.jar	BOEXI
rpoifs.jar	BOEXI
Serialization.jar	BOEXI
u211java.jar	BOEXI
URIUtil.jar	Apache Jakarta Commons HttpClient
vha-stddata-basic-18.0.jar	SDS
vha-stddata-client-18.0.jar	SDS
webi.jar	BOEXI
webreporting.jar	BOEXI
webreporting-core.jar	BOEXI

1.1.4 Web Application

The PATS enterprise archive contains the PATS web application. The root contents are virtual directories and the WEB-INF directory.

1.1.4.1 Virtual Directories

PATS has the following virtual directories underneath the web context root:

Directory Name	Description
crystalreportviewers10	contains the style sheets and images for viewing crystal reports
help	contains the help files for the SIT, SRCU and VU roles
javascript	contains common Javascript files used by the jsp pages
jsp	root virtual directory for the core PATS application
login	contains KAAJEE sub-component
META-INF	Web Application descriptor per the J2EE specifications
npohelp	NPO-user help files
plu	PSL sub-component
stylesheets	PSL stylesheets
WEB-INF	contains web application configuration per the J2EE specifications

1.1.4.2 Web.xml and Weblogic.xml

Web.xml is the main configuration file for the PATS web application. Web.xml contains:

- context parameters
- servlets
- servlet mappings
- filters
- filter mappings
- ejb local references

- security roles
- security constraints
- login configuration
- taglib declarations
- welcome files
- error pages
- mime mappings
- listeners

Weblogic.xml contains the mapping of the logical Java enterprise role to the physical WebLogic security realm.

1.1.4.3 Other Configuration Files

WEB-INF contains the KAAJEE configuration file kaajeeConfig.xml as well as PSL's struts-config-plu.xml.

PATS uses an internal MVC framework. The files for administering the PATS MVC framework are contained in the APP-INF\classes folder of the enterprise application. These files are:

- commands.properties – contains the mapping of URL actions to PATS action classes in the package gov.va.med.pats.ui.command.
- pagenumber.properties – each JSP page has a page number property specified in this file.
- patsweb.properties – contains the user interface version number, plu security setting, and return row count property for the number of rows to display in list screens, and page sensitive help settings. Also contains the list of authorized IP addresses for the PAD SOAP web service, and switch for the PAD development tools.
- URL.properties – contains the mapping of PATS events to physical JSP files.
- **Note:** Other property files in the APP-INF\classes folder are used by the service dependencies.

1.1.4.4 Stylesheets

The main PATS stylesheet is the file named style.css in the virtual directory /jsp/css. This stylesheet provides a consistent look and feel throughout PATS.

The files located in the /jsp/css/aqua virtual directory are stylesheets used by the pop-up calendar to aid in date selection.

1.1.4.5 Libraries

The PATS web application contains the following libraries in the WEB-INF/lib directory:

File Name	Description
-----------	-------------

commons-httpclient-2.0.1.jar	Apache Jakarta Commons
File Name	Description
jakarta-regexp-1.3.jar	Jakarta
kaajee-1.0.1.003.jar	KAAJEE
keycodeDecoder.jar	BOEXI
LongList.jar	PSL
MetafileRenderer.jar	BOEXI
pslWeb_4.0.4.3.jar	PSL
struts.jar	Apache Jakarta Struts

1.1.4.6 Classes

The class files for the PATS web application are located in the WEB-INF/classes directory. The class packages are described in the following table.

Package Name	Description
gov.va.med.pats.ui.bean	Used for storing data during posts to the server and subsequent responses from the server
gov.va.med.pats.ui.bean.helper	Called by filter classes to populate a form bean before passing it to the requested page
gov.va.med.pats.ui.command	Contains the <i>ControllerServlet</i> which is the central class where all server requests made by the client are routed to the responding <i>Commands</i>
gov.va.med.pats.ui.command.maintenance	Contains table maintenance functionality
gov.va.med.pats.ui.command.report	Contains reports functionality
gov.va.med.pats.ui.command.roc	Contains ROC functionality
gov.va.med.pats.ui.command.roc.notification	Contains notifications functionality
gov.va.med.pats.ui.command.user	Contains user maintenance functionality
gov.va.med.pats.ui.common	Contains searching and sorting functionality
gov.va.med.pats.ui.filter	Filters to Verify a User, Add/Edit ROCs, and View/Update Notifications
gov.va.med.pats.ui.filter.exception	Used when a user attempts to view a notification not intended for the
gov.va.med.pats.ui.taglib.maintenance	JSP tags to generate dynamic content on the table maintenance pages
gov.va.med.pats.ui.taglib.roc	JSP tags to generate dynamic content on the Add/Edit ROC pages
gov.va.med.pats.ui.taglib.support	JSP tags to generate dynamic content in

	general areas within the application
Package Name	Description
gov.va.med.pats.ui.util	General utilities for the user interface
jsp_servlet	WebLogic adds compiled JSP pages under this directory tree
gov.va.med.pats.pad	Contains PAD web service automatic ROC creation functionality
gov.va.med.pats.pad.tools	Contains mock SOAP implementations to use in local development/testing for PAD SOAP web service

1.1.5 Ejb-jar Archive

PATS business layer is packaged as an ejb jar with deployment descriptors located in the META-INF directory and classes in the EJB JAR file.

PATS deploys the following local and remote ejbs:

- NationalMaintenance –business subsystem for NPO users
- NotificationFacade - notifications business subsystem
- PATSFacade –core PATS business subsystem
- VISNMaintenance –business subsystem for VISN-level users

The Ejb-jar archive contains the following descriptors under the /META-INF directory:

- Ejb-jar.xml contains the following elements:
 - Session beans: All PATS ejbs are stateless session beans. EJB is primarily used for declarative method security.
 - Security roles
 - Method permissions
 - Container transactions: supported
- Weblogic-ejb-jar.xml contains:
 - Weblogic enterprise beans: Important for the JNDI address of each bean
 - Weblogic security roles: The mapping of logical Java enterprise roles to physical WebLogic security roles.
- Hivemodule.xml contains service points for PATS data access objects.

1.2 HealthVet Configuration Files

PATS depends on configuration settings that must be updated by the deployer. Because of recent security updates some default port number and server information has been removed. Contact HealthVet Maintenance team for the values.

These files and settings are:

- gov.va.med.pats.businessobjects.enterprise.properties

Property Name	Example
CmsAuthType	secEnterprise
CmsName	Vhaisfxxxx.vha.med.va.gov:%CMSPORT%
CmsAdminUsername	Administrator
CmsAdminPassword	%password%
Property Name	Example
PatsCmsFolderName	PATS reports
LicenseType	1
PasswordNeverExpires	false
MustChangePassword	false
CannotChangePassword	false
UserPassword	%password%
Title	PATS reports user
Description	Programmatically added
UNIVERSE_NAME	PATS Universe
INFOVIEW_CONTEXT	http://vhaisfxxxx.vha.med.va.gov:%TOMCAT_PORT% /businessobjects/enterprise11 Note: The server name, vhaisfxxx, must match the CmsName.
PublicFolderName	Public Folders
PatsCmsFolderName	PATS Reports
AdHocFolderName	PATS Universe

- gov.va.med.pats.notification.properties (IN = information notification, ARN = action request notification)

Property Name	Example
activedirectory.ICFACTORY	com.sun.jndi.ldap.LdapCtxFactory
activedirectory.DOMAIN	Vhamaster
activedirectory.SERVER	ldap://vhaisfxxxx.vha.med.va.gov:%ADSERVE

Patient Advocate Tracking System (PATS)

	R_PORT%
activedirectory.PRINCIPAL	username
activedirectory.PRINCIPAL.CREDENTIAL	%password%
activedirectory.SEARCH.RESULTSIZE	500
activedirectory.SEARCH.CONTEXT	DC=va, DC=gov
mail.smtp.timeout	2000
Property Name	Example
mail.RELAY	Vhaishxxx.vha.med.va.gov
notification.FROMADDRESS	PATS@donotreply.va.gov
notification.FROMPERSONAL	PATS Notification
notification.URL.HOST	http://Vhaishxxx.vha.med.va.gov:portnumber
notification.URL.CONTEXT	/PATS/
notification.FOOTER1	At the login page select
notification.FOOTER2	From the institution drop down list
notification.FOOTER3	Note: If you have trouble accessing the notification, return to this email and click on the link again.
Notification.FOOTER4	If the problem continues, contact the Help Desk at 1.888.596.4357.
notification.IN.SUBJECT	PATS Notification – Informational
notification.IN.BODY_TEXT1	A Patient Advocate has logged an Informational Notification in PATS.
Notification.IN.BODY_TEXT2	For more information, click the following link:
notification.IN.ACTION	jsp/notifications/viewINRcpt.jsp?qs=
notification.ARN.RCPT.SUBJECT	PATS Notification – Action Request

notification.ARN.RCPT.BODY_TEXT1	A Patient Advocate has logged an Action Request Notification.
Notification.ARN.RCPT.BODY_TEXT2	For more information and to reply to the notification, click the following link:
notification.ARN.RCPT.UPDATED.SUBJECT	PATS Notification – Updated Action Request
notification.ARN.RCPT.UPDATED.BODY_TEXT1	The Patient Advocate has updated an Action Request Notification.
Property Name	Example
notification.ARN.RCPT.UPDATED.BODY_TEXT2	For more information and to reply to the notification, click the following link:
notification.ARN.RCPT.ACTION	jsp/notifications/viewARNRcpt.jsp?qs=
notification.ARN.SRCU.SUBJECT	PATS Notification - Action Request
notification.ARN.SRCU.BODY_TEXT1	An Employee has replied to an Action Request Notification.
notification.ARN.SRCU.BODY_TEXT2	For more information or to reply to the notification, click the following link:
notification.ARN.SRCU.ACTION	jsp/notifications/viewARNAdv.jsp?qs=

- gov.va.med.pats.migration.db.oracle.properties

Property Name	Example
db.driver	thin
db.port	%ORACLE_PORT%
db.sid	an_SID
db.server	vhaishxxxx.vha.med.va.gov
db.user	patshost
db.password	%password%
db.protocol	tcp

Patient Advocate Tracking System (PATS)

db.maxlimit	15
db.stmtcachesize	10

- gov.va.med.pats.application.properties

Property Name	Example
ROC.business.rule.year	2008

PATS also depends on the following service dependency configuration files that may need updating during deployment:

Configuration File	VA Service
application.properties	SDS
ApplicationResource.properties	PSL
PatientLookup.properties	PSL
PersonLookupResources.properties	PSL
PatSvcPkg.properties	PSC
KaajeeDatabase.properties	KAAJEE
gov.va.med.vistalink.connectorConfig.xml	VLJ

1.3 PATS Loggers

Description	Logger Name		Log Levels
	Package	Class	
User Interface utility logging class	gov.va.med.pats.common	Log	All
Data access loggers	gov.va.med.pats.dao	CompDAO	Error Uses HiveMind logging interceptor for debug information.
		CongressionalContactDAO	
		ContactingEntityDAO	
		EmployeeDAO	
		FacilityServiceOrSectionDAO	
		HospitalLocationDAO	
		IssueCategoryDAO	
IssueCodeDAO			

Patient Advocate Tracking System (PATS)

		MethodOfContactDAO	
		NotificationDAO	
		PatientDAO	
		ResolutionTextBoilerplateDAO	
		RocDAO	
		TreatmentStatusDAO	
		UserDAO	
Data access exception handler	gov.va.med.pats.dao.exception	DaoExceptionRule	Error

1.4 Exceptions

PATS internal checked exceptions are:

- gov.va.med.pats.dao.exception.DuplicateDataException
- gov.va.med.pats.dao.exception.NoDataFoundException
- gov.va.med.pats.dao.exception.OptimisticLockException

PATS internal runtime exceptions are:

- gov.va.med.pats.dao.exception.DataAccessException
- gov.va.med.pats.dao.exception.DatabaseNotFoundException
- gov.va.med.pats.delegate.exception.ServiceLocatorException

Data access exceptions are managed by the class gov.va.med.pats.dao.DaoExceptionRule.

Access Local Exception: You may see an error message similar to the following in the log file:

```
javax.ejb.AccessLocalException: [EJB:010160]Security
Violation: User: '<anonymous>' has insufficient permission to
access EJB: type=<ejb>, application=PATS-APP-xxxx,
module=PATSEJB.jar, ejb=PATSFacade, method=create,
methodInterface=LocalHome, signature={}
```

It probably means the user timed out of the PATS application. After 20 minutes of inactivity. After the application times out, if the user attempts to perform some action in PATS, they see a message telling them they must log on again, and an error is recorded in the log. However, if you see many such messages in a row, it is possible that someone is attempting to illegally access the application.

1.5 Service Imports

PATS imports the following classes from VA services:

Standard Data Service (SDS)

- gov.va.med.term.access.Ethnicity
- gov.va.med.term.access.FacilityType
- gov.va.med.term.access.Institution
- gov.va.med.term.access.Race
- gov.va.med.term.access.criteria.Criteria

KAAJEE

- gov.va.med.authentication.kernel.LoginUserInfoVO
- gov.va.med.authentication.kernel.VistaDivisionVO

Patient Service Lookup (PSL)

- gov.va.med.person.lookup.common.IPLUConstants
- gov.va.med.person.lookup.patient.transfer.PatientLookupBean
- gov.va.med.person.lookup.patient.transfer.UserBean
- gov.va.med.person.lookup.ui.web.common.PLSessionMgr

VistALink for Java (VLJ)

- gov.va.med.vistalink.adapter.cci.VistaLinkConnection
- gov.va.med.vistalink.adapter.cci.VistaLinkConnectionFactory
- gov.va.med.vistalink.adapter.cci.VistaLinkDuzConnectionSpec
- gov.va.med.vistalink.institution.InstitutionMapNotInitializedException
- gov.va.med.vistalink.institution.InstitutionMappingDelegate
- gov.va.med.vistalink.institution.InstitutionMappingNotFoundException
- gov.va.med.vistalink.rpc.RpcRequest
- gov.va.med.vistalink.rpc.RpcRequestFactory
- gov.va.med.vistalink.rpc.RpcResponse
- gov.va.med.xml.XmlUtilities

Patient Service Construct (PSC)

- gov.va.med.patientadmin.common.Ethnicity
- gov.va.med.patientadmin.common.IIdentifier
- gov.va.med.patientadmin.common.IName
- gov.va.med.patientadmin.common.IVistaDate
- gov.va.med.patientadmin.common.PatientId
- gov.va.med.patientadmin.common.RequestType
- gov.va.med.patientadmin.common.RequestedServices
- gov.va.med.patientadmin.delegate.IPatientServiceDelegate
- gov.va.med.patientadmin.delegate.IPatientServiceRequest
- gov.va.med.patientadmin.delegate.PatientServiceDelegate
- gov.va.med.patientadmin.delegate.PatientServiceRequest
- gov.va.med.patientadmin.exception.PatientServiceException
- gov.va.med.patientadmin.transfer.IEnrollmentEligibilityTO
- gov.va.med.patientadmin.transfer.IPatientServiceTO
- gov.va.med.patientadmin.transfer.IPrimaryDemographicsTO
- gov.va.med.patientadmin.transfer.ISecondaryDemographicsTO

- gov.va.med.patientadmin.transfer.ITertiaryDemographicsTO
- gov.va.med.patientadmin.transfer.PatientServiceTO

1.6 Requirement for Sending Notifications

PATS uses Microsoft Outlook email for sending Informational Notifications (INs) and Action Request Notifications (ARNs). One component of the email system is the Lightweight Directory Access Protocol (LDAP). In order for PATS to use LDAP for retrieving user information, an LDAP account was set up with the VA Enterprise Group. The user currently configured for PATS use is named vhaishpats.

This account is used during application configuration. This setting resides in the gov.va.med.pats.notification.properties file:

```
activedirectory.PRINCIPAL=vhaishpats
```

LDAP fail-over and load-balancing is also setup during configuration. This setting resides in the gov.va.med.pats.notification.properties file:

```
activedirectory.SERVER=ldap://bigkahunas.vha.med.va.gov:%ADSERVER_PORT%
```

1.7 Integration Agreements for use of Java Components

Integration Agreement	Description
4851 – KAAJEE (Supported)	KAAJEE addresses the Authentication and Authorization needs of Health_Vet-VistA web-based applications in the J2EE environment.
4148 - VistALink J2M (v1.5) Java APIs (Supported)	Lists all Java APIs for the VistALink J2M-software package. A summary of all VistALink J2M Java APIs by package is listed first, and the components of each package are presented in detail thereafter.
4338 - Person Service Patient Construct Java APIs - used by PATS (Private)	Lists supported Java APIs for the Patient Service Construct (PSC) software package that are used by PATS.
4368 - Person Service Patient Lookup Java APIs - used by PATS (Private)	Lists supported Java APIs for the Patient Service Lookup (PSL) software package that are used by PATS.
4369 - Standard Data Service Java APIs - used by PATS (Private)	Lists supported Java APIs for the Standard Data Service (SDS) software package that are used by PATS.

1.8 PAD Servlet ROC Creation

The PAD servlet is designed to receive XML messages sent over HTTP and automatically generate and store a new ROC. This service was envisioned to work with the Inquiry Routing and Information System (IRIS) but will be compatible with any SOAP web service consumer as long as the IP address of the caller is authorized in the patsweb.properties file.

The PAD functionality is encapsulated in the gov.va.med.pats.pad package in the patsweb project and is implemented in the following two classes:

- PadServlet.java
 - Main entry point for receiving the XML request
 - Determines the authorized IP address, creates the SOAP object from the request and passes the main ROC XML node to the Roc Parser
 - Validates the resulting ROC object and stores the ROC in the PATS database
 - Processes and sends the successful response message (or error message) back to the service caller
- RocParser.java
 - Receives the populated XML object from Pad Servlet
 - Transverses each XML node, parses and validates the data and stores in the ROC object
 - Searches for a matching patient in the PATS database and stores in the ROC object if found
 - Flags any error messages for the Pad Servlet to return to the caller

The PAD functionality utilizes the existing PATS classes and delegate framework for patient search, ROC validation, ROC storage, and ROC update. PAD requires a data source connection only to the PATS DB schema supplied by the hosting WebLogic Server.

The PAD servlet is invoked by sending an XML SOAP request to the pad servlet URL: <pats_production_url>/pad.servlet. See the PAD ICD at the PAD TSPR documentation page for details on the exact content of the XML request

<http://tspr.vista.med.va.gov/warboard/anotebk.asp?proj=1425>

1.8.1 Troubleshooting PAD

Successful execution of the PAD servlet will always result in an XML response to the caller with an associated ROC number in this format:

```
<rocNumber><![CDATA[442.201300001]]></rocNumber>
```

Any response from the PAD servlet that does not contain a <rocNumber> node in the XML response is an indication that the interface failed to execute properly. Error messages returned from PAD may include:

Your system is not authorized for this function XML error response returned to the caller. This is the message PAD sends back to the service caller if the IP address of the request is not in the authorized 'iris.ips' list in patsweb.properties. Validate the IP of the caller and add it to the list.

Error 500--Internal Server Error error message returned to the caller. This message indicates PAD was properly invoked but encountered an unchecked exception during parsing of the XML request; i.e. the thread crashed. Validate the request XML for

validity and format. Examine the server log file and associated stack trace for indications of the error.

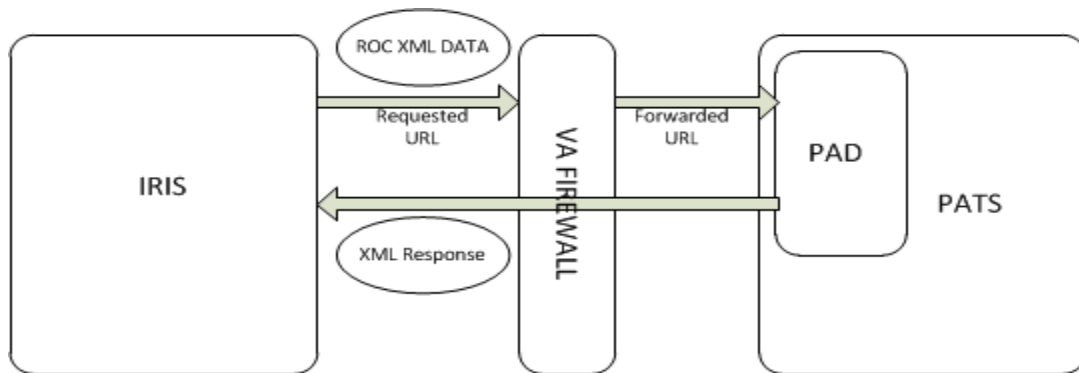
The <fieldname> is required error message returned to the caller. This message indicates the request XML contained a null value for a required field. Validate the request XML according the PAD ICD and re-send the request.

Invalid <fieldname> <description> error message returned to the caller. This message indicates the request XML contained data for an expected field but the supplied data failed to pass the validation rules; i.e. length was too long or too short. Validate the request XML for the failed data field and re-send the request.

Curl error: Cannot resolve host error message returned to the caller. This message indicates the SOAP request could not find the URL destination for the PAD servlet; most likely due to firewall blocking. Validate the URL and port the caller is using to invoke the PAD servlet.

The VA firewall must be configured to allow incoming requests to the PAD servlet URL to be passed through to the PATS server. Coordinate with the PATS system administrator and the VA NSOC to ensure existing FQDN forwarding rules and server certificates are still valid.

IRIS-PAD SOAP Communication Exchange



1. The IRIS system is provided the external URL to send messages to the PAD Servlet **<pats_public_fqdn>/PATS/pad.servlet, i.e. www.pats.va.gov/PATS pad.servlet**
2. The VA firewall receives the external request for the URL and forwards to the internal PATS production/staging/dev URL as appropriate to invoke the PAD servlet.
3. PAD processes the ROC and returns the Success message to the caller.

2.0 Java Enterprise Developer Workstation

PATS Java enterprise developer workstations are dependent on remote and local services. All VA Java service dependencies are deployed locally while Business Objects Enterprise and Oracle were utilized remotely.

2.1 Development Platform

The following infrastructure developed PATS on Windows workstations:

Application	Description
WebLogic Server (WLS) 8.1 sp 4 (or higher)	All service dependencies are required to be setup on a local WebLogic server.
Eclipse 3.2.2 and MyEclipse 5.1.1	Integration of Eclipse and WebLogic 8.1 allowed hot deployment and hotswap debugging.
Dreamweaver MX	HTML and JSP files were aligned and cleaned in Dreamweaver.
Microsoft Visual Source Safe	Source control management
Oracle 11g and TOAD	TOAD was used to develop Oracle objects (tables, procedures, etc.). Oracle 11g is where the data is stored.
Business Objects Enterprise XI sp1	For developing and deploying standard and ad hoc reports.

2.2 Development Projects

PATS' development structure is comprised of multiple dependent projects in the eclipse interactive development environment. These projects are:

Project	Description
etc	Base project containing master build scripts, runtime libraries, development libraries, deployment descriptors and configuration files
patscommon	PATS domain object archive and generic utilities, depends on etc
patsdao	PATS data access object archive, depends on patscommon
patsejb	PATS ejb and notification archive, depends on patsdao
patsreport	PATS reporting archive, depends on etc
patsdelegate	PATS delegate archive, depends on patsejb
patsweb	PATS web user interface archive and PAD servlet, depends on patsejb and patsreport
patstest	PATS cactus tests, depends on patsdelegate
patstestclient	Junit helper classes for running cactus tests from a remote client, depends on patstest

patreporttest	Junit tests for PATS reporting, depends on patreport
---------------	------------------------------------------------------

2.3 Development Tools

During development, deployment of PATS is done using the ANT build file in etc (for building PATS without unit tests) or patstest (for building PATS with unit tests). PATS can be deployed in two basic ways on Weblogic:

- as an uncompiled EAR (see the PATS-WLS-EAR target in etc/build.xml)
- or
- a WebLogic application compiled exploded directory structure (see the zip-compiled target in etc/build.xml.)

2.3.1 ANT

ANT 1.7 is the build tool for PATS. Each project contains a build file in the root directory named build.xml and where applicable an xdoclet-build.xml.

2.3.2 XDoclet

All Java enterprise deployment descriptors are generated by XDoclet 1.2. The following Xdoclet files are merged together with XDoclet Annotations from Servlets, taglibs, filters and EJBs. Xdoclet merge files are contained in the etc/merge directory for a given project.

2.3.3 Log4j

Apache Log4j 1.2 is PATS logging framework.

2.3.4 Libraries

Additional libraries (located in the Java project etc's lib directory) necessary for development and/or building of PATS are:

File Name	Description/Source Project
Aspectjrt-1.2.1.jar	Aspectj
Boconfig.jar	BOEXI
cactus-1.7.jar	Apache Jakarta Cactus
cactus-ant-1.7.jar	Apache Jakarta Cactus
cdzlet.jar	BOEXI
cecore.jar	BOEXI
celib.jar	BOEXI
ceplugins.jar	BOEXI
cereports.jar	BOEXI
cesession.jar	BOEXI
ceutils.jar	BOEXI
cexsd.jar	BOEXI

classes12dms.jar	Oracle
commons-beanutils.jar	Apache Jakarta Commons
File Name	Description/Source Project
commons-collections-3.0.jar	Apache Jakarta Commons
commons-collections.jar	Apache Jakarta Commons
commons-digester.jar	Apache Jakarta Commons
commons-discovery.jar	Apache Jakarta Commons
commons-httpclient-2.0.1.jar	Apache Jakarta Commons
commons-httpclient-2.0.2.jar	Apache Jakarta Commons
commons-logging-1.0.4.jar	Apache Jakarta Commons
commons-logging.jar	Apache Jakarta Commons
commons-resources.jar	Apache Jakarta Commons
commons-validator.jar	Apache Jakarta Commons
CorbaIDL.jar	BOEXI
ebus405.jar	BOEXI
Httpunit-1.6.jar	BOEXI
j2ee-1.3.jar	Sun
jasper-compiler-4.1.30.jar	Apache Jakarta
jasper-runtime-4.1.30.jar	Apache Jakarta
jrцерom.jar	BOEXI
junit-3.8.1.jar	Junit
kaajee_1.0.1.003.jar	KAJEE
keycodeDecoder.jar	BOEXI
log4j-1.2.8.jar	Apache Logging
MetafileRenderer.jar	BOEXI
nekohtml-0.9.1.jar	Apache
nls_charset12.jar	Oracle
org.mortbay.jetty-4.2.17.jar	Mortbay
rasapp.jar	BOEXI
rascore.jar	BOEXI
ReportPrinter.jar	BOEXI
ReportTemplate.jar	BOEXI
rpoifs.jar	BOEXI
Serialization.jar	BOEXI
servletapi-2.3.jar	Sun
u211java.jar	BOEXI

vljConnector-1.5.1.002.jar	VLJ
vljFoundationsLib-1.5.1.002.jar	VLJ
File Name	Description/Source Project
webi.jar	BOEXI
webreporting-advanced.jar	BOEXI
webreporting-core.jar	BOEXI
webreporting-jsf.jar	BOEXI
webreporting.jar	BOEXI
WebReportWizard.jar	BOEXI
wilog.jar	BOEXI
xdoclet-1.2.jar	Xdoclet
xdoclet-bea-module-1.2.jar	Xdoclet
xdoclet-ejb-module-1.2.jar	Xdoclet
xdoclet-Java-module-1.2.jar	Xdoclet
xdoclet-web-module-1.2.jar	Xdoclet
xdoclet-xdoclet-module-1.2.jar	Xdoclet
xercesImpl-2.7.1.jar	Apache
xml-apis-2.7.1.jar	Apache

2.3.5 Junit and Cactus

Junit 3.8 and Cactus 1.7 are PATS unit testing tools.

2.3.6 Rational XDE

All Rational XDE model files are stored in the root of each PATS project.

2.3.7 Maven

Maven can be used for running Junit tests in PATS and to generate site reports. A project.xml is located in the root of each Java project.

2.3.8 PAD Tools

The PAD servlet API is driven through SOAP XML requests sent from an external system. To facilitate local development and testing, the following classes and jsp pages were developed to simulate the external service consumer since both service endpoints cannot be reasonably expected to be available in all development environments.

- /public_html/jsp/pad/PadServletTool.jsp
 - Utilizes gov.va.med.pats.pad.tools.PadServletTestTool.java
 - This page and backing java class act as the service consumer and allows the developer to input direct XML messages to send to the PAD servlet.
 - This page also will collect the response XML data sent back from the PAD servlet after ROC creation and mimic the behavior of the

consuming system upon receiving the response.

- To access this tool, deploy PATS to your local weblogic instance and navigate to <http://localhost:7001/PATS/pad.servletTool>
- [/public_html/jsp/pad/PadTool.jsp](#)
 - Utilizes `gov.va.med.pats.pad.tools.PadTestToolServlet.java`
 - This page and backing java class acts as a wrapper around the PAD interface in order to directly invoke the ROC parser. The interface will display actual XML received by the PAD servlet and actual XML response data sent back after ROC creation.
 - To access this tool, deploy PATS to your local weblogic instance and navigate to <http://localhost:7001/PATS/pad.padTool>
- To enable the PAD development tools, set 'enable.pad.tools=true' in the `patsweb.properties` file.

In lieu of using the built in servlet test tools, developers may also utilize external SOAP testing tools such as SOAPUI to send XML to a local PAD instance.

2.4 Services

The following VA services are deployed locally in a development WebLogic server:

Dependency	Version Used
Standard Data Services (SDS)	Standard Data Services 18
Vista Link	Vista Link Java 1.5.2
Kernel Authentication Authorization Java Enterprise (KAAJEE)	Kernel Authentication Authorization Java Enterprise Environment 1.0
Person Service Lookup (PSL)	Person Service Lookup 4.0
Patient Service Construct (PSC)	Patient Service Construct R2

3.0 Business Rules Implementation

3.1 Web Security

The security constraints placed on the URLs that a user may access are located in web.xml. Programmatic security is used for hiding invalid options.

3.2 Ejb Method Permission Security

EJB declarative security in ejb-jar.xml is used to define method permissions in the business layer. Programmatic security is not used in the business layer.

3.3 Transactions

PATS does not use Java transactions. PATS uses Oracle transactions in stored procedures.

3.4 ROC

The business rules for reports of contact are in the class gov.va.med.pats.ejb.policy.RocRule.

3.5 Notification

The business rules for sending notifications are in gov.va.med.pats.ejb.policy.NotificationRule.

3.6 Table Maintenance

The table maintenance subsystem of PATS is isolated in two EJBs: NationalMaintenance and VISNMaintenance.

3.7 Concurrency

PATS uses optimistic concurrency. In the Java tier of PATS this means all value objects extend the abstract class gov.va.med.pats.value.ValueObject. The field rowVersion is passed to each update stored procedure transaction.

3.8 Application Design Parameters

3.8.1 Timeout Parameters

- PATS times out after 20 minutes of inactivity.
- When users select the Ad hoc reporting option, PATS launches a separate application, BOEXI Web Intelligence. It appears as if users are still in the PATS application though they're actually in Web Intelligence. Users may stay in the ad hoc reports application designing reports for longer than 20 minutes. For that reason, when PATS launches Web Intelligence, the timeout for the PATS application running in the background is temporarily changed to 60 minutes. The timeout is temporarily changed so that PATS won't timeout in the background while the user is in ad hoc reporting. When the user returns to the PATS application, the timeout reverts to 20 minutes.

3.8.2 Date Parameters

- PATS currently maintains all data migrated from Patient Rep. The age of the oldest data can vary from site to site, depending on how long a site used the Patient Rep application before migrating to PATS, and whether they ever archived and purged their Patient Rep data.
- For standard and ad hoc reports PATS provides data only for the current fiscal year and two previous fiscal years.

3.8.3 Report Instance Limits

The number of report instances that can be stored is limited as described in the Instance Limits section of the Business Objects chapter. If the limit is exceeded, old report instances are automatically purged.

3.8.4 Text Field Length

- The length of the Issue Text field is limited to 4000 characters.
- The length of the Resolution Text is limited to 8000 characters. This is stored in two 4000-character fields in the Oracle database but appears to the end user to be a continuous 8000-character text string.

4.0 M VistA

Prior to using the PATS application at a given site, the site must install a KIDS build. The KIDS build brings in routines, options, remote procedure calls (RPCs), and security keys. No new data files are brought in with this installation, and the existing Patient Representative files, routines, options and security keys are not changed by this installation. See the *Patient Advocate Tracking System (PATS) Installation Guide for IRM Staff* for instructions on the installation and setup.

Legacy data from the Patient Representative system must be cleaned then migrated into the PATS Oracle database prior to using the PATS application. See the *Patient Advocate Tracking System (PATS) Data Migration Guide* for instructions.

The PATS application retrieves some patient and employee demographic data from the VistA system where the current user is logged on. This data is stored for use in reporting, but cannot be edited from within the PATS system. When a patient is selected for inclusion on a Report of Contact, PATS uses the Patient Service Lookup and Patient Service Construct services to find and retrieve patient information. This data currently comes from VistA. In addition, PATS retrieves person demographics from the VistA NEW PERSON file. This occurs when a patient advocate is selected as either the information taker or entered by person on a ROC, when an employee is selected as the employee involved in a ROC, or when an employee is sent a notification from PATS.

4.1 Namespace

Routines, options, and remote procedures beginning with **QACI** are used to migrate legacy data into PATS. Routines, remote procedures and security keys beginning with **QACV** are used during normal operation of the PATS application.

4.2 Routine Descriptions

Routine Name	Description
QACI0	Entry point to generate the Pre-Migration error reports used to assist the user in cleansing the data. Calls QACI2.
QACI1 QACI1A	Auto-close open ROCs prior to the beginning of the previous quarter.
QACI2 QACI20 QACI2A QACI2B QACI2C QACI2D QACI2E	Checks data for errors and generates the migration error report, entry point to move legacy data into staging area in preparation for migration.
QACI3	Called by the Remote Procedure QACI LOAD REFERENCE TABLES from the Data Migration application (PATSDM). Pulls reference table data from the staging area into the ^TMP global during migration of legacy data to PATS.

Routine Name	Description
QACI4	Called by the Remote Procedure QACI LOAD ROC from the Data Migration application (PATSDM). Pulls ROC data from the staging area into the ^TMP global during the migration of legacy data to PATS.
QACI5	Inactivate or reactivate the Patient Representative options that allow editing of Patient Representative data.
QACVEMPX	Called by the Remote Procedure QACV PERSON LOOKUP VLH. This routine returns a list of entries from the NEW PERSON file that match a given partial last and first name. The routine takes 'last value'

	name parameters and a number of entries to return so that it can return the next 'n' entries following the last value (i.e., Value list Handler). This is called from the PATS application after the user has selected an employee involved or employee notified on a ROC.
QACVKHLD	Called by the Remote Procedure QACV KEY HOLDERS VLH. This routine returns a list of entries from the NEW PERSON file that have been assigned one or more security keys. The routine takes 'last value' name parameters and a number of entries to return so that it can return the next 'n' entries following the last value (i.e., Value list Handler). This is called to return a list of potential 'info takers' for a ROC.

4.3 Temporary Globals

The ^XTMP("QACMIGR", \$J) global is used to store information during the data migration process. See the *PATS Data Migration Guide* for details on the use of this global.

4.4 Options

New options are brought in to support migration of the legacy Patient Representative data into PATS. In addition, the Broker Type option QACV PATS RPC ACCESS is brought in to support the active PATS application.

Options	Description
QACI MAIN DATA CLEANUP MENU	Main Menu for Pre-Migration Data Cleanup
QACI PREMIGRATION ERROR REPORT	Check for Data Errors Prior to Migration
QACI AUTO CLOSE ROCS	Auto-Close Old Open ROCs
QACI REPORT MENU	Data Migration Reports
QACI REPORT AUTO-CLOSED ROCS	List All ROCs that were Auto-Closed
QACI REPRINT ERRORS REPORT	Reprint Migration/Pre-Migration Data Errors Report

Options	Description
QACI REPORT ROC CHANGES	System Generated Changes to ROCs for Migration
QACI REPORT MIGR COUNTS	Count of Errors, Ready to Migrate, Migrated
QACI REPORT NOTIFICATIONS	Pending Notifications Report
QACI ROC MIGRATION STATUS	Display Migration Status for a ROC
QACI MAIN PATS MIGRATION MENU	Main Menu for Patient Rep Migration Steps
QACI MIGRATION DATA BUILD	Move Data to Staging Area for Migration
QACI REPORT MENU	Data Migration Reports (same as above)
QACI UTILITY MENU	Activate/Inactivate Options or tasks
QACI INACTIVATE PAT REP	Activate/Inactivate Patient Rep Options
QACI RESCHEDULE ROLLUP TASK	Reschedule Rollup to Austin
QACI KILL ROLLUP TASK	Stop (Kill) Rollup to Austin
QACI PATS RPC ACCESS	Broker type option used to register PATS Remote Procedure Calls used during Data Migration
QACV PATS RPC ACCESS	Broker type option used to register PATS Remote Procedure Calls, which retrieve Employee data for the application.

4.5 Remote Procedure Calls

Name	Description
QACI DELETE ALL LISTS	Used in data migration (PATSDM), this is called when user selects the option to delete all of the data from the Oracle side to completely restart the data migration process. This deletes all lists of Patient Rep items that have been migrated into PATS from the ^XTMP global.
QACI LOAD REFERENCE TABLES	Used in data migration (PATSDM), this loads reference table data from the ^XTMP global into the ^TMP global to send back to the calling program.
QACI LOAD ROC	Used in data migration (PATSDM), this loads ROC data from the ^XTMP global into the ^TMP global to send back to the calling program.

Name	Description
QACI NATL INSTITUTION LIST	Used in application to download list of national divisions (PATSDV). Retrieves a list of station numbers from the Standard Data Services institution table std_institution. The list contains all station numbers that begin with the 3-character computing facility station number (default institution). The list is stored in the ^XTMP array and is used in the pre-migration data cleanup option to make sure that any data references only valid station numbers.
QACV KEY HOLDERS VLH	Used when running PATS, this returns lists of employees who hold one or more security keys. This takes as input an array of security key names, a number 'n' indicating how many rows to return, and optionally a 'previous name returned' and 'previous IEN returned'. If the 'last name' and 'last IEN' are input, they are used to find the starting point for the search. The RPC returns the next 'n' entries from the NEW PERSON file that holds one or more of the keys in the input list. The output list includes the person name, Title and Mail Code. The RPC also returns a flag indicating whether there are more entries.
QACV PERSON LOOKUP VLH	Used when running PATS, this returns lists of employees, matching on first and last name. This takes as input a name in the format 'last,first', a number 'n' indicating how many rows to return, and optionally a 'previous name returned' and 'previous IEN returned'. If the 'last name' and 'last IEN' are input, they are used to find the starting point for the search. The RPC returns the next 'n' entries from the NEW PERSON file whose name matches the input parameter. The output list includes the person name, Title and Mail Code. The RPC also returns a flag indicating whether there are more entries.

4.6 External Relations

KAAJEE: PATS uses the Kernel Authentication Authorization Java Enterprise Environment, a Security service located on VistA for use by reengineered web applications, for authenticating users during sign on, and for allowing them access to various PATS options based on roles. KAAJEE has a VistA component that must be installed in order to run PATS. The roles-based access is controlled by the use of new security keys created for PATS. See 'Security Key' section below for details. The list of accessible divisions' data is retrieved by KAAJEE from the DIVISION multiple on the VistA NEW PERSON file.

VistALink: PATS uses VistALink, a communications bridge between VistA and J2EE application Servers, to retrieve data. VistALink has a VistA component that must be installed in order to run PATS.

Person Service Lookup: PATS uses Person Service Lookup service when the user needs to add a patient to a ROC. The service includes a User Interface (UI) that is incorporated within PATS. The UI prompts for patient information and displays a list of matching patients. When a patient is selected, any patient related warnings and additional information are displayed. The identifier for the patient is then returned to the PATS application and stored in a local PATS table. Person Service Lookup has a VistA component that must be installed in order to run PATS.

Patient Service Construct: After a patient has been selected for a ROC, PATS calls Patient Service Construct service to get patient demographics information for the patient. This information is stored on a local PATS table to support reporting and rollup to Austin. Patient Service Construct has a VistA component that must be installed in order to run PATS.

VistA Data used in PATS: During data migration each site will use new VistA options to check for errors in their legacy data and a web-based application to migrate the clean data from VistA into the PATS Oracle tables. This includes some Patient demographics and data from the NEW PERSON file (#200). HOSPITAL LOCATION file (#44) entries referenced by ROCs in VistA are migrated to the hospital_location table within PATS. From that point forward, the table is maintained entirely by the PATS users and has no remaining ties to the VistA HOSPITAL LOCATION table.

Patient and Person demographic data cannot be altered by the PATS application. Patient data is refreshed from VistA when a patient is selected on a ROC using the Patient Service Construct methods. Remote Procedure Calls written for the PATS application are used to retrieve some demographic data from the VistA NEW PERSON file when an Employee Involved is added to a ROC, or when a Notification is sent to an Employee.

Note: None of the new VistA routines, options or remote procedure calls brought in with the PATS installation should be called by any application outside of PATS.

4.7 VistA Integration Agreements

4.7.1 Supported

Integration Agreement	Description
2171 – Routine XUAF4	LKUP^XUAF4 is used to retrieve the station IEN from file 4, from the station number portion of the ROC number. PARENT^XUAF4 is used to retrieve the VISN name and STA^XUAF4 to return the parent station number, based on the station in the QAC SITE PARAMTERS file 740. (Used in routines ^QACI1* and ^QACI2*, data migration from Patient Rep into PATS. Also Used in ^QACVEMPX, called by the RPC "QACV KEY HOLDERS VLH". In both cases, used to find the station number for the VistA server in order to build the KAAJEE user key for the pats_user table).
2701 – Routine MPIF001	\$\$GETICN^MPIF001 is used to retrieve the Integration Control Number of patients. This is used when migrating legacy data into PATS (routine QACI2B).
2918 – \$\$PRIORITY^DGENA	See 2462 in Section 4.7.2 for details.
3065 – Routine XLFNAME	\$\$HLNAME^XLFNAME is used to return the name components from the PATIENT file during data migration to load data into the pats_patient table (routine ^QACI2B). It is also used to retrieve the name components from the NEW PERSON file while entering or editing the Employee Involved or Information Taker data on a Report of Contact (routine ^QACVEMPX called by both the RPC "QACV PERSON LOOKUP VLH" and "QACV KEY HOLDERS VLH") and during data migration to load data into the pats_user table (routine ^QACI2B).
3799 – Routine DGUTL4	\$\$INACTIVE is used to determine whether a Patient's Race or Ethnicity code are currently inactive. We use \$\$PTR2CODE to use the pointer value for a patient's race to retrieve the HL7 VALUE field, and the pointer value for a patient's ethnicity to retrieve the ABBREVIATION field. A Standard Data Services programmer suggested that these two fields should be used to match a race or ethnicity in the VistA files against the std_race and std_ethnicity tables that are part of the Standard Data Service. (Used in routine QACVDEM that returns patient demographics for data migration from Patient Rep to PATS.)

Integration Agreement	Description
10060 – Access fields in NEW PERSON file (200)	LIST^DIC and GETS^DIQ is used on file 200 to retrieve field .01 NAME, 8 TITLE and 28 MAIL CODE. This is done when retrieving Employee Involved data or Information Taker data while entering or editing a Report of contact (routine ^QACVEMPX called by both the RPC "QACV PERSON LOOKUP VLH" and "QACV KEY HOLDERS VLH", and during data migration to load data into the pats_user table (routine ^QACI2B). We also use FIND^DIC with the "AB" index (on the KEYS multiple) to return a list of users who own one or more keys (Used in ^QACVKHLD, which is called by ^QACVEMPX, called by the RPC "QACV KEY HOLDERS VLH").
10061 – Routine VADPT	Calls DEM^VADPT and ELIG^VADPT are used to retrieve patient demographics and eligibility data. This is done during data migration to load data into the pats_patient table (routine ^QACI2B).

4.7.2 Controlled Subscription

Integration Agreement	Description
1518 – KERNEL SITE PARAMETERS file (8989.3) DEFAULT INSTITUTION field (217)	\$\$GET1^DIQ is used to return the pointer to the INSTITUTION file from the DEFAULT INSTITUTION file. PATS (Patient Rep) was added to the list of subscribers on 12/05/2003. (This is used in ^QACVEMPX, called by the RPC "QACV KEY HOLDERS VLH", and in data migration code ^QACI2. In both cases, it is used to find the station number for the VistA server in order to build the KAAJEE user key for the pats_user table).
2462 – Access to PATIENT ENROLLMENT file 27.11	(See also 2918 in the list of supported calls above) The supported call \$\$PRIORITY^DGENA is used to retrieve the internal value of the users Enrollment Priority (a set of codes) based on the patient's current enrollment. Then, we use a FileMan call \$\$EXTERNAL^DILFD to find the external value of the Enrollment Priority from the DD (File 27.11, field .07). (Used from routine ^QACVDEM). We are displaying this to users of the PATS system along with other patient data related to a patient complaint or compliment. This call is used during data migration of legacy Patient Rep data into PATS.
2689 – Reference to ALERT file 8992	We look in ^XTV(8992,"AXQAN" to find all entries that begin with "QAC-". From these entries, we extract the "sent to" and "date sent" from the cross-reference, then we use the 0 node of the ALERT DATE/TIME multiple, based on the date sent, in order to get the "sent from" person. For both the "sent to" and "sent from" persons, we're getting the .01 field from the NEW PERSON file. PATS (Patient Rep) was added as a subscriber in April, 2004. (Used during data migration to create a report of outstanding notifications, routine ^QACI5).

4.8 Online Documentation

The PATS options used for data migration use standard FileMan conventions for online help.

4.9 Check Sum Values for Routines

Using CHECK1^XTSUMBLD. There are 17 routines:

Routine	Value
QACI0	1984508
QACI1	23422676
QACI1A	6784447
QACI2	15898410
QACI20	56932032
QACI2A	18512192
QACI2B	36187753
QACI2C	34183381
QACI2D	68859293
QACI2E	22678389
QACI3	8458772
QACI4	3186113
QACI5	43902138
QACIENV	2063143
QACVDEM	35412405
QACVEMPX	18632980
QACVKHLD	5032982

4.10 Security and Keys

PATS uses the Kernel Authentication Authorization Java Enterprise Environment (KAAJEE), a Security service located on VistA for use by reengineered web applications, for authenticating users during sign on, and for allowing them access to various PATS options based on roles. The same access and verify codes used for VistA are also used to sign on to PATS.

Security Keys stored in the VistA system are used to control access to various options within the PATS system.

Keys	Access Level
------	--------------

QACV_ADUSH	Assistant Deputy Undersecretary for Health (later changed in PATS documentation to Central Office user). Has access only to the National Patient Report.
QACV_DMGR	Data Migration Manager –The person assigned the VistA option to move legacy Patient Representative data into the staging area prior to migration to PATS is given this key. This person is also responsible for using the PATS Data Migration application to do the migration of the legacy data into PATS.
QACV_NPO	National Program Office – The person given this key has access to maintain national tables within the PATS system. This includes the Issue Category, Issue Code, Contacting Entity, Method of Contact, Treatment Status, and National PATS Parameters. Also has access to the National Patient Report.
QACV_SIT	Site Information Taker – This is an administrative person at a medical center. The person given this key is allowed to enter or update ROCs in the PATS system, but is not allowed to perform maintenance tasks, such as entering boilerplate resolution information or close the ROCs.
QACV_SRCU	Site Record Control User – This is a Patient Advocate. The person given this key is allowed to enter, edit, resolve and close Reports of Contact. This person also maintains the locally controlled reference tables: Hospital Location, Congressional Contacts, Boilerplate Resolution Text and Comps.
QACV_VU	VISN User – The person given this key maintains the Facility Service or Section table in the PATS system for their VISN.

5.0 Database - Oracle

The PATS persistent data is stored in Oracle tables maintained on a central server. The version at the time of PATS initial installation is Oracle 11g.

5.1 Database

The name of the production database at the HSI's Data Center is VPFSPRD.med.vha.va.gov.

5.2 Schemas

Schema	Description
PATS	Owns all of the tables and procedures for the main PATS application.
PATSRPTS	Owns all of the tables and procedures that will be used to support both standard and ad hoc reporting.

5.3 Users

User	Description
PATS	Owns all of the tables and procedures for the main PATS application. No external database connections are made directly as the PATS user.
PATSRPTS	Owns all of the tables and procedures that will be used to support both standard and ad hoc reporting. The only external database connection made as PATSRPTS is when making a connection to the database in order to run standard reports (Crystal Reports) or to access the PATS Universe for creating ad hoc reports.
PATSUSER	Owns no Oracle objects. All database connections from the PATS application user interface, with the exception of reporting, are made as PATSUSER. The patsuser_role is assigned to this user and controls access to Oracle objects.
PATSHOST	Owns no Oracle objects. All database connections from the PATS Data Migration application are made as PATSHOST. The patshost_role is assigned to this user and controls access to Oracle objects.
PATSROLLUP	The VSSC will connect as this user when fetching the weekly rollup data for the VSSC reports. This user has SELECT access ONLY to the pats_rollup_natl_data table.

5.4 Roles

Role	Description
PATSUSER_ROLE	Controls access to objects used by the PATS application. This is granted to the PATSUSER user. This role is granted EXECUTE privileges on all of the packages in the PATS schema except for those used during migration of legacy data. This role is also granted EXECUTE privileges on the pkg_list (used in standard reports) and set_ad_hoc_user (used in ad hoc reporting) packages in the PATSRPTS schema.
PATSHOST_ROLE	Controls access to objects in the PATS schema used for Data Migration. This is granted to the PATSHOST user. This role is granted EXECUTE privileges on all of the packages and functions used to migrate legacy data into PATS. It is also granted select, insert, update and delete access on the DMLOG table used to track data migration errors.

5.5 Tablespaces

Tablespace	Description
PATS_DATA	Used to store tables and data in the PATS schema
PATS_NDX	Used to store indices on tables in the PATS schema
PATSRPTS_DATA	Used to store tables and data in the PATSRPTS schema
PATSRPTS_NDX	Used to store indices on tables in the PATSRPTS schema

5.6 Tables

5.6.1 PATS Schema

All of the data tables used in the PATS application are in the PATS schema. Refer to the data model in the *PATS Software Architecture Document*, located in the TSPR.

Table Name	Description
BOILERPLATE_RESOLUTION_TEXT	Contains standard boilerplate text that can be used to describe the resolution for a given issue code. Foreign keys reference the sdsadm.std_institution table, and the issue_code table. The institution_fk field references the division for which the entry was created; an entry can only be selected or edited when a user is logged on under that division.

Table Name	Description
COMPS	Contains a list of comps that can be given to a patient as part of the resolution of a ROC. A foreign key references the sdsadm.std_institution table. The institution_fk field references the division for which the entry was created; an entry can only be selected or edited when a user is logged on under that division.
CONGRESSIONAL_CONTACT	Contains a list of names of congressional persons or offices that can be selected when the contacting entity on a ROC is set to <i>congressional</i> . A foreign key references the sdsadm.std_institution table. The institution_fk field references the computing facility; an entry entered for one division can be selected or edited by all divisions within the facility.
CONTACTING_ENTITY	Contains a nationally managed list of contacting entities used to categorize the entity that contacted the patient advocate and caused a ROC to be created. Examples are: patient, relative, friend.
DMLOG	Used during migration of legacy data to track errors.
FACILITY_SERVICE_OR_SECTION	Contains a list of facility service or sections defined at the VISN level. This is a way of tracking standard organizational entities within every medical center in a division. An FSOS is entered as part of the issue code cluster for a ROC. A foreign key references the sdsadm.std_institution table for the VISN that owns the entry; an entry entered for one division can be selected or edited by a VISN-level user logging on to any division within the same VISN.
HOSPITAL_LOCATION	Contains a list of physical locations within a medical center. A hospital location can be entered as part of the issue code cluster for a ROC. A foreign key references the sdsadm.std_institution table. The institution_fk field references the division for which the entry was created; an entry can only be selected or edited when a user is logged on under that division. This table is not related to the HOSPITAL LOCATION file in VistA.
ISSUE_CATEGORY	Contains a nationally maintained list of codes used to group types of issues for reporting. Each issue code falls within a single issue category.

Table Name	Description
ISSUE_CODE	Contains a nationally maintained list of codes used to categorize the types of complaints or compliments on ROCs for reporting. An issue code is entered as part of the issue code cluster for a ROC. A foreign key references the issue_category table.
METHOD_OF_CONTACT	Contains a nationally managed list of ways a person can report a complaint or compliment that causes a ROC to be created. Examples are: phone, letter, visit.
NATIONAL_PATS_PARAMETERS	This nationally maintained table contains just one record. It holds the value of national parameters that control the behavior of PATS.
NOTIFICATION_DETAIL	This contains dialogue between the patient advocate and an employee when an Action Request Notification (ARN) is sent to the employee. Foreign keys reference the associated notification_master record, and the pats_user table entries for both the sender and recipient of each message.
NOTIFICATION_MASTER	This contains information about a single Action Request Notification (ARN) or Information Notification (IN) sent from a patient advocate to an employee. Foreign keys reference the associated ROC in the report_of_contact table, and the pats_user table for both the originating PA and the employee who received the notification.
PATS_PATIENT	This contains a list of patients who are involved in at least one ROC. It contains minimal demographics information about each patient used for reporting and identification. An entry on this table can be uniquely identified either by a combination of the institution and the VistA IEN (internal entry number), or by the Patient ICN. PATS users are not allowed to edit this table. This table is populated using data returned by Person Service methods. Foreign keys reference the sdsadm.std_institution table and the sdsadm.std_ethnicity table. The institution_fk field references the default institution.

Table Name	Description
PATS_PATIENT_RACE	This contains a list of the race values associated with a patient. This table is refreshed along with the pats_patient table above. Foreign keys reference the pats_patient table, and the sdsadm.std_race table.
PATS_ROLLUP_NATL_DATA	This table contains data that is ready to be rolled up to the VSSC system in Austin for national reporting. Every week a scheduled job runs. The old data in the table is deleted and new data is inserted, from the report_of_contact table and other data, for every ROC that has been entered or edited since the previous week. The selection of which ROCs to rollup is based on the rollup_to_natl_reports_status flag on the report_of_contact table entries.
PATS_USER	This contains a list of all users who have signed on to the PATS application, persons who are employees involved in a ROC, or persons who have been sent a notification. The table has an id field that is the primary key, but there is also a field user_identifier that contains the KAAJEE user identifier. This identifier contains the station number and the VistA IEN from the NEW PERSON file. There is a foreign key referencing the sdsadm.std_institution table. The parent_institution_fk field references the default institution.
REPORT_OF_CONTACT	This is the table that contains the main record of a complaint or compliment and its resolution. It contains foreign keys referencing the pats_user table, for both the information taker for the ROC and the person who entered the ROC. Foreign keys reference the congressional_contact, treatment_status, pats_patient and sdsadm.std_institution tables. The institution_fk field references the division for which the entry was created; an entry can only be selected or edited when a user is logged on under that division.
ROC_CONTACTING_ENTITY	This table contains the contacting entity(s) associated with a ROC. Foreign keys reference the report_of_contact and contacting_entity tables.

Table Name	Description
ROC_ISSUE	This table contains the issue cluster(s) associated with a ROC. Foreign keys reference the report_of_contact, issue_code, hospital_location, facility_service_or_section and pats_user (employee involved in ROC) tables.
ROC_METHOD_OF_CONTACT	This table contains the method(s) of contact associated with a ROC. Foreign keys reference the report_of_contact and method_of_contact tables.
ROC_PHONE_FAX	This table contains phone/fax information associated with a ROC. A foreign key references the report_of_contact table.
TREATMENT_STATUS	Contains a nationally managed list of values describing the treatment status of a patient. Examples are: inpatient, outpatient, long term care.
PAD_LOCATION	This table is a static lookup reference table used by the PAD servlet to map incoming location names to PATS institution/station names.

5.6.2 PATSRPTS Schema

Separate tables are created and populated to support both standard and ad hoc reporting in PATS. The data in the tables is refreshed from the production tables once daily using Oracle scheduled jobs.

Table Name	Description
CONTACTING_ENTITY_VIEW	Supports standard reports. Contains a list of all Contacting Entities. Created so that all contacting entities can be reported on spreadsheet reports, not just those that are referenced by ROCs.
ISSUE_CATEGORY_VIEW	Supports standard reports. Contains a list of all Issue Categories. Created so that all Issue Categories can be reported on the spreadsheet reports, not just those that are referenced by ROCs.
ISSUE_CODE_VIEW	Supports standard reports. Contains a list of all Issue Codes. Created so that all Issue Codes can be reported on the spreadsheet reports, not just those that are referenced by ROCs.
METHOD_OF_CONTACT_VIEW	Supports standard reports. Contains a list of all Methods of Contact. Created so that all Methods of Contact can be reported on the spreadsheet reports, not just those that are referenced by ROCs.

Table Name	Description
ROC_COMBO_DATA_VIEW	Supports ad hoc reporting. Contains the data used to support the WEBI Universe used for Ad Hoc reporting.
ROC_CONTACTING_ENTITY_VIEW	Supports standard reports. Contains all contacting entity data associated with ROCs in the ROC_MAIN_DATA_VIEW.
ROC_ISSUE_DATA_VIEW	Supports standard reports. Contains all issue cluster data associated with ROCs in the ROC_MAIN_DATA_VIEW.
ROC_MAIN_DATA_VIEW	Supports standard reports. Contains all ROC data for ROCs with dates of contact in the current or prior two fiscal years.
ROC_METHOD_OF_CONTACT_VIEW	Supports standard reports. Contains all method of contact data associated with ROCs in the ROC_MAIN_DATA_VIEW.
ROC_NOTIFICATION_VIEW	Supports standard reports. Contains all notification data associated with ROCs in the ROC_MAIN_DATA_VIEW.
ROC_PHONE_DATA_VIEW	Supports standard reports. Contains all phone/fax data associated with ROCs in the ROC_MAIN_DATA_VIEW.
TREATMENT_STATUS_VIEW	Supports standard reports. Contains a list of all Treatment Statuses. Created so that all Treatment Statuses can be reported on the spreadsheet reports, not just those that are referenced by ROCs.
USER_STATION_LIST_FOR_AD_HOC	Supports ad hoc reporting. Contains a list of station numbers and users within that station. When a user signs on to the PATS application, a procedure is called that replaces all current entries for that user matching on KAAJEE user identifier, with a current list of accessible station number(s) for that user. This is used to filter the data during ad hoc reporting so that the user can access only data for their station(s).

5.7 Field Information

Field	Description
Inactivation Date	Entries on tables that are referenced by foreign keys cannot be deleted. Instead, there is a procedure to inactivate (or reactivate) an entry. This sets (or deletes) an inactivation date on the entry. Inactivated entries do not show up in selection lists in the UI.
VER	The VER field on a table is used to maintain optimistic concurrency (described below in the section <i>Techniques used in PATS Procedures – Optimistic Concurrency</i>).
Sort Order	Some table data requires a standard order for displaying lists of data in the UI. This is maintained by using a numeric sort_order field. Arrays returned from the list procedures for these tables are sorted using this field.
Upper Case Name	Some tables have upper-case values stored for key search fields. These are automatically triggered when the key field is created or updated. These fields are used to allow searches to be non-case sensitive.
Sort Code	The issue_code table has a field called sort_code. This is triggered whenever an issue_code is created or updated. All of the current issue codes are 4 characters long, but the PATS user group wanted to allow for 5 characters. In the case of an issue code like SC99, the sort_code version is stored as SC099, so that in the future, SC100 will sort after SC99.
Standard Name	The pats_patient file and the pats_user file both have std_name_for_lookup and std_first_name fields, used to support lookup by first and last name. These are triggered when parts of the persons name are created or updated. Standardizing a name part puts the name into upper case and removes most punctuation, thus allowing, for example, a user input of O'Brien or OBRIEN to find the same name.
ID	Most of the tables have a sequential integer field named ID as the primary key. This field is automatically assigned during insert by a BEFORE INSERT trigger. The trigger takes the next value from an oracle SEQUENCE assigned to that table, by selecting sequence_name.NEXTVAL into :new.id.
Field names ending in FK	If the name of a field ends with _FK , it indicates that there is a foreign key constraint on the field.
SDS	Fields in some tables have foreign key constraints that reference tables in the SDSADM schema. These tables are maintained by Standard Data Services. There are references to STD_INSTITUTION, STD_RACE and STD_ETHNICITY.

Field	Description
-------	-------------

Resolution Text	Oracle limits the maximum size of a VARCHAR2 field to 4000 characters. To accommodate the resolution text, two fields were created on the REPORT_OF_CONTACT table. RESOLUTION_TEXT1 holds the first 4000 characters of the resolution text, RESOLUTION_TEXT2 holds the remaining 4000 characters, if any.
-----------------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

5.8 Report Data Tables in PATSRPTS Schema

Data is automatically extracted from the live PATS data into separate tables for use in standard and ad hoc reports. The data is refreshed once daily for ad hoc reports at 3:00 am and standard reports at 3:30 am based on the time zone of the server where the Oracle database resides. It uses Oracle scheduled jobs described in section 5.11. ROC data is extracted only for ROCs whose date of contact is within the current fiscal year, or the two previous fiscal years, based on the date the data is extracted. (The time the scheduled job runs can be adjusted using the Oracle OEM.)

The use of these tables separates the data used for reporting from the live data used for running the PATS application. Most foreign key values in the PATS data are resolved to their actual data values in the report tables, in order to simplify report design and to make reports run more efficiently by eliminating most of the joins.

The reporting tables are described above in the **PATSRPTS Schema, Tables** section.

5.9 Procedures

Detailed documentation for all procedures used in the PATS application can be found in the Oracle stored procedures document. This document acted as a contract between the business layer developer and the database developer. It describes each stored procedure, including the input and output parameters. See the Oracle stored procedures document in PATS TSPR (Background/Supporting Documents section) at <http://tspr.vista.med.va.gov/warboard/anotebk.asp?proj=604>.

5.9.1 PATS Schema

The PATS application uses functions and procedures owned by the PATS schema, for all database access and manipulation.

Functions: Functions defined within the PATS schema are used within procedures. They are:

Function	Description
Abbr_issue_text	Returns the first 80 characters of the issue_text field from the report_of_contact table.
Calculate_roc_overdue	Returns the date a ROC will become overdue, based on the date_of_contact for the ROC, and the number of days to process a ROC from the National PATS Parameter table.
Function	Description

Calendar_to_fiscal_year	Takes an incoming date and returns the fiscal year for that date as an integer.
Displayable_name	Takes the standard name parts (last, first, etc.) and depending on other input variables, returns a displayable name truncated to a requested length, in several formats.
Displayable_time_zone	Takes a timestamp as input and calculates the time zone for display (i.e., EDT, EST).
New_roc_number	Takes a station number and date of contact, and returns the next available ROC number for that station and fiscal year.
Return_id_table	Takes a string containing a comma delimited list of integers, and returns an array.
Standard_name	Takes a name input by a user, puts it in upper case and removes punctuation and spaces to assist in searches by name.

Packages and Procedures: Most Packages in the PATS schema are used to maintain a single PATS entity. Most of them contain standard procedures used for adding a single entry (add), updating a single entry (upd), retrieving a single entry (get), or retrieving a list of entities (list). If the table has an inactivation date field, the list procedure allows the caller to specify whether to return only active entries, inactive entries, or all entries. The list procedures generally return either all records, or they use the Value List Handler procedure to return the next ‘n’ records where ‘n’ is provided as an input parameter. This Value List Handler procedure is described below in the *Techniques used in PATS Procedures – Value List Handler Pattern* section.

Below are lists of the packages and procedures in PATS schema, with notes about procedures that differ from the standard set mentioned above.

Packages/Procedures called by PATS application

Packages/Procedures	Description
pkg_boilerplate_text	<ul style="list-style-type: none"> ▪ Maintains the boilerplate_resolution_text table. ▪ Contains a procedure to delete entries. ▪ The list procedure will return either all entries, or just those entries related to issue codes within a given issue category.
Pkg_comp	Maintains the comps table.
Pkg_congressional_contact	Maintains the congressional_contact table.

Packages/Procedures	Description
Pkg_contacting_entity	<ul style="list-style-type: none"> ▪ Maintains the contacting_entity table. ▪ The procedure upd_sortorder_list takes a delimited list of id value/sort order pairs, and updates the sort_order field for each entry referenced by the id value, with the new sort order. This sort order field is used to control the order in which the entries are displayed within the user interface.
Pkg_employee	<ul style="list-style-type: none"> ▪ Lists Employees Involved in a ROC. ▪ This package contains only a single procedure list_empinv_by_name. The procedure lists all employees involved in a ROC, whose last name and first name are partial matches to the input parameters. The data for employees involved in a ROC are stored in the pats_user table, so other table maintenance for employees is contained in the pkg_pats_user package.
Pkg_facility_serv_sect	Maintains the facility_service_or_section table.
Pkg_hospital_location	Maintains the hospital_location table.
Pkg_issue_category	<ul style="list-style-type: none"> ▪ Maintains the issue_category table. ▪ This package contains a procedure upd_sortorder_list similar to the one described above for pkg_contacting_entity. ▪ The list procedure sorts the entries first by the is_customer_service_standard field, then within it by the sort_order field.
pkg_issue_code	<ul style="list-style-type: none"> ▪ Maintains the issue_code table. ▪ The list procedure allows the caller to send an issue category, in order to limit the output to just those issue codes that reference that issue category.
Pkg_method_of_contact	<ul style="list-style-type: none"> ▪ Maintains the method_of_contact table. ▪ This package contains a procedure upd_sortorder_list similar to the one described above for pkg_contacting_entity.

Packages/Procedures	Description
Pkg_natl_pats_parameters	<ul style="list-style-type: none"> ▪ Maintains the National PATS Parameters table. ▪ At the time of release, there was just one parameter, the number of days within which a ROC must be processed. ▪ There are only two procedures. ▪ Upd_param updates the number of days within which a ROC must be processed. ▪ Get_days_to_process_roc returns the current value of the number.
Pkg_notification	<ul style="list-style-type: none"> ▪ Maintains the notification_master and notification_detail tables. ▪ Add_notif adds a new entry to the notification_master file and the first entry in the notification_detail when a new IN or ARN is created. ▪ Add_detail is called to add additional dialogue (i.e., a new entry in the notification_detail table) to an existing ARN.
Pkg_pats_patient	<ul style="list-style-type: none"> ▪ Maintains the pats_patient table. ▪ Rather than a separate add and update procedure, this has a <i>set</i> procedure that either adds a new entry, or refreshes all data in the existing entry, matching on parent station and the internal entry number from the VistA Patient file. ▪ List_patient_by_name returns all patients whose last name and first name are partial matches to input parameter values. ▪ List_patient_by_ssn_nssn returns all patients whose ssn or nssn match the input parameter. (Nssn is the first initial of the last name, followed by the last 4 digits of the ssn).

Packages/Procedures	Description
pkg_pats_user	<ul style="list-style-type: none"> ▪ Maintains the pats_user table. ▪ Users with access to the PATS application, employees sent a notification, and employees involved in a ROC are all kept in this table. ▪ Rather than a separate add and update procedure, this has a <i>set</i> procedure that uses the KAAJEE user identifier to see whether the user already exists, then either adds a new entry, or refreshes all data in the existing entry. ▪ There is no list procedure in this package. Person lookups are currently done using a remote procedure call that searches for the user on the VistA NEW PERSON file.
Pkg_report_of_contact	<ul style="list-style-type: none"> ▪ Maintains the report_of_contact table, and tables that reference the report of contact (roc_contacting_entity, roc_issue, roc_method_of_contact, roc_phone_fax). ▪ List_roc_vlh returns a list of open, closed or all ROCs for a single station, for either a single information taker, matches to a partial ROC number, all ROCs for a single employee is involved in the complaint, or ROCs within a given date range. ▪ List_roc_no_vlh returns a list of open, closed or all ROCs for a single station, for either a single patient, or only ROCs that are overdue ▪ Open_close_roc sets the status of a ROC to either open or closed. ▪ Delete_roc allows user to physically delete a ROC from the database. All the table entries referencing the ROC are also deleted.
Pkg_report_of_contact	<p>Count_overdue returns a count of open ROCs that are overdue, based on their date of contact, the current date and the number of days within which a ROC must be processed, according to the National PATS parameter.</p>
Pkg_treatment_status	<ul style="list-style-type: none"> ▪ Maintains the treatment_status table. ▪ This package contains a procedure upd_sortorder_list similar to the one described above for pkg_contacting_entity.

Package used to roll up data to the VSSC

Rollup Data Package/ Procedures	Description
Pkg_rollup_natl_data.rollup	<p>This procedure is scheduled within Oracle to run once weekly. The procedure deletes all data from the pats_rollup_natl_data table, then reads through every entry on the report_of_contact file. For each entry with the rollup_to_natl_reports_status field set to 1, and reference by at least one roc_issue table entry, the ROC data is moved into the pats_rollup_natl_data table. The VSSC group in Austin runs a job once weekly that collects all of the data from the pats_rollup_natl_data table to support national reporting.</p>

Packages/Procedures used for Data Migration

Data Migration Package/ Procedures	Description
pkg_load_legacy_data	<p>This package contains procedures called by the Java Data Migration Application, while it is moving legacy data from the VistA Patient Rep system into the Oracle PATS tables.</p> <ul style="list-style-type: none"> ▪ The package adds entries to the congressional_contact, hospital_location, pats_patient, pats_user, facility_service_or_section and report_of_contact tables. ▪ Data to be updated is input in a delimited array. Key values for the entries on the Oracle tables are passed back to the calling routine in an output array. Procedures to populate each table, check key values from the VistA data to see whether an entry is already on the table. If it is, the key value from the Oracle table is returned in the output array. If it is not, a new entry is added, and the key value from the Oracle table is returned in the output array. ▪ The output array is passed by the Java application back to VistA to keep track of which entries have been successfully migrated.
Delete_all_data	<p>This procedure is called by the Java Data Migration Application to delete all of the data from the PATS Oracle tables for a list of sites. The delete_all procedure deletes all of the data from the congressional_contact, hospital_location, pats_patient, pats_user and report_of_contact, for the sites whose values are passed in input parameters.</p>

Build PATS National Table Data

Build PATS National Table Data	Description
pkg_build_pats_static_data	<p>This package is used to populate the national reference tables with initial lists of values. It is called during initialization of PATS, when the PATS Oracle database is deployed. It populates the following tables.</p> <ul style="list-style-type: none"> ▪ national_pats_parameter ▪ contacting_entity ▪ method_of_contact ▪ treatment_status ▪ issue_category ▪ issue_code

5.9.2 PATSRPTS Schema

Functions and Procedures used for database access and manipulation to generate report data for the PATS application are owned by the PATSRPTS schema.

Functions: Functions defined within the PATSRPTS schema are used within procedures.

Function	Description
Abbr_issue_text	Returns the first 80 characters of the issue_text field from the report_of_contact table.
Displayable_boolean	Takes an integer valued at 0 or 1 and returns the text Yes or No .
Displayable_ssn	Takes the internal value of the SSN and formats it with hyphens like 000-00-0000.
Display_roc_status	Takes the internal value of the ROC status (O or C) and returns the text Open or Closed .
Old_fy_end	The tables that support reporting contain data for the current fiscal year and the entire two previous fiscal years. This function takes a date as input and returns a date that is the end of the fiscal year three years prior to the current fiscal year. The code that builds report data uses this function to select ROCs whose date of contact falls within the current or previous two fiscal years.
Return_inst_id_table	This function is not currently in use.
Return_stationno_table	Takes a string containing a comma delimited list of substrings (station numbers), and returns an array.

Procedures to support standard reporting: Each of these procedures generates the output data that is the input to a Crystal Report.

Procedure Name	Used to build output data for:
Contacts_no_patient	Contacts with No Patient Identified report
Cust_serv_std_detail	Customer Service Standard Detail report
Cust_serv_std_summary	Customer Service Standard Summary report
Daily_list_patient_contacts	Daily List of Patient Contacts report
Employee_contact_totals	Employee Contact Totals report
Fsos_contact_totals	Facility Service or Section Contact Totals report
Fsos_issue_totals	Facility Service or Section issue Totals report
Issue_totals_by_gender	Issue Totals by Gender (All, Male or Female) report
List_of_open_cases	List of Open Cases report
Location_issue_totals	Location Issue Totals report
Natl_rocs_by_patient	National Patient Report
Notifications_no_response	Employee Notifications with No Response report
Patient_brief_data	Patient Name with Brief Data report
Percent_of_rocs_with_comps	Per Cent of ROCs with Comps report
Report_by_employee	Report by Employee Involved report
Report_of_contact1:5	Report of Contact report. There is a main report and four sub-reports used to create the Report of Contact. Each of these is associated with a separate stored procedure. The data for the main report is generated by the procedure report_of_contact1, the subreport data is generated by procedures report_of_contact2 through report_of_contact5.
Rocs_with_comps_by_issue_code	ROCS with Comps by Issue Code report
Ss_count_issues_by_fsos	Spreadsheet – Issue Count by Facility Service or Section report
Ss_count_issues_by_institution	Spreadsheet – Issue Count by Division report
Ss_count_issues_by_issue_cat	Spreadsheet – Issue Count by Issue Category report
Ss_count_issues_by_location	Spreadsheet – Issue Count by Location report
Ss_count_issues_by_sex	Spreadsheet – Issue Count by Patient Gender report
Ss_count_issues_by_ts	Spreadsheet – Issue Count by Treatment Status report

Procedure Name	Used to build output data for:
Ss_count_rocs_by_cont_entity	Spreadsheet – Contact Count by Contacting Entity report
Ss_count_rocs_by_institution	Spreadsheet – Contact Count by Division report
Ss_count_rocs_by_issue_code	Spreadsheet – Contact Count by Issue report
Ss_count_rocs_by_moc	Spreadsheet – Contact Count by Method of Contact report
Ss_count_rocs_by_ts	Spreadsheet – Contact Count by Treatment Status report

Procedures to support ad hoc reporting

Procedure Name	Description
Set_ad_hoc_user	This procedure is called by pats.pkg_pats_user.set_person whenever a user logs on to the PATS application. It adds entries to the user_station_list_for_ad_hoc for each of the station numbers to which this user has access. The table is used to filter data during ad hoc report creation, so that the user can see only ROCs for stations to which they have access.
Build_roc_adhoc_data	This procedure is called by the scheduled job build_roc_adhoc_data_job. The procedure first deletes all data from the roc_combo_data_view table that is used for ad hoc reporting. It then reads through the entire pats.report_of_contact table. For each ROC in the current or previous two fiscal years, the procedure builds new data into the roc_combo_data_view table. Data from the pats.report_of_contact table, and all associated tables is joined in the roc_combo_data_view. This data is used to support the WEBI universe used for ad hoc reporting

Packages to support standard reports

Package Name	Description
Pkg_list	This package contains procedures used to do lookups of employees and patients on the PATS tables for the user interface to standard reports.
Pkg_report_local	This package contains only a Package Specification defining an output cursor used by all of the standard reports.

Package Name	Description
Pkg_bld_std_report_data	Entry point bldall is called by the scheduled job bld_std_report_data_job. The procedure first deletes all data from all the tables that are used to support standard reporting. It then reads through the entire pats.report_of_contact table. For each ROC in the current or previous two fiscal years, the procedure builds new data into the tables.
Util_Batch_Job	This package contains procedures for disabling/enabling constraints and truncating report tables from the calling code. It is used mainly to get rid of the old data without generating archive logs.

5.10 Techniques used in PATS Procedures

Technique	Description
Packages	Packages allow for the definition of package level output cursors that can be referenced as result sets (arrays) in Java Code. Within a package, a developer can define private procedures that can be called from within the package, but can't be referenced from outside the package. Packages allow code used to maintain a single table or related tables to be consolidated.
Result Sets	REF Cursor datatypes are used to return Result Sets (arrays of data records). These REF Cursors are always defined within the Package Specification rather than within the Package Body, so that they are public and can be referenced by the Java Code. Any package that contains either list or get procedures contains code in the Package Specification that looks something like: TYPE t_cursor IS REF CURSOR.
Optimistic Concurrency	In order to maintain optimistic concurrency, any table whose entries might be edited by more than one user simultaneously, has a date field named VER. When a record is selected for update, the VER field is converted to VARCHAR2 and returned with the other data. When calling the update procedure, the calling routine must pass the VER field, along with the data to be updated. VER is converted back to a date, then is compared against the VER field in the record to be updated. If VER has not changed, no other user has updated the record since it was selected, so the update is done. Otherwise, the update does not occur and an error is displayed: <i>"This table_name was updated by another user. Please make your changes again."</i>
Value List Handler Pattern	The value list handler pattern is used for any of the procedures that return lists of data that might be large. This pattern allows the user to specify that they want to see the next n records. The input parameters always include the number of records to return n and which set of

	records to return. Output parameters help the caller determine how many sets of n records there are on the table.
--	--------------------------------------------------------------------------------------------------------------------------

5.11 Scheduled Jobs

Job	Description
PATS.BUILD_ROLLUP_DATA_JOB	Scheduled to run once weekly. This job builds the rollup data which the Austin VSSC fetches once weekly in order to create the national VSSC reports. This job calls the pats.pkg_rollup_natl_data.rollup procedure described above. More details can be found in <i>Chapter 7, Rollup PATS Data to VSSC</i> .
PATSRPTS.BUILD_ROC_ADHOC_DATA_JOB	Scheduled to run once daily at 3:00 am, based on the time zone of the server where the Oracle database resides (EHS can adjust the time). This job builds the data that underlies the WEBI Universe used for ad hoc reporting. The job calls the patsrpts.build_roc_adhoc_data procedure described above.
PATSRPTS.BLD_STD_REPORT_DATA_JOB	Scheduled to run once daily at 3:30 am, based on the time zone of the server where the Oracle database resides (HSI can adjust the time). This job builds the data that underlies standard reporting. The job calls the package entry point patsrpts.Pkg_bld_std_report_data.bldall described above.

5.12 Developer Workstation Setup

TOAD Professional version 8.5.3.2, with DBA Module and Knowledge Expert – This is from Quest Software. Used for table and other Oracle entity maintenance, to maintain stored procedures and to execute SQL code. The DBA module is used to build the scripts to deploy the Oracle part of the PATS application. You can also do debugging with this tool (set breakpoints and watchpoints, look at the value of local variables, etc.) TOAD can be used by a DBA to manage the Oracle database.

The Quest Knowledge Expert gives help on PL/SQL syntax, and on interpreting and correcting Oracle database errors.

The database developer for PATS has found the Quest tools to be intuitive and easy to use. Quest offers classes and their support is excellent!

Patient Advocate Tracking System (PATS)

Oracle Enterprise Manager Console – This comes with the Oracle Installation. This is mainly a tool for the DBA. The database developer for PATS used it for managing tablespaces, adding new users and assigning them privileges.

5.13 External Relations

Standard Data Services (SDS): PATS has subscribed to the SDS group to load their standard data tables into the SDSADM schema on the PATS Oracle database. The data will automatically be refreshed by the SDS application. PATS uses supported SDS APIs to extract data from the tables as needed. For reporting and weekly rollup to the VSSC system in Austin, PATS creates local tables and uses joins with the SDS tables in order to extract data into the tables. Because PATS is accessing the SDS tables directly rather than using the supported calls to retrieve the data in this case, integration agreements have been requested and granted for the reporting and rollup needs. The tables used are std_institution, std_race and std_ethnicity.

5.14 Database Integration Agreements

PATS has the following Private integration agreements with SDS (Custodial package **term**).

Integration Agreement	Description
4887 – Use of STD_INSTITUTION table in PATS application	Describes PATS foreign key constraints referencing the std_institution table and granting of REFERENCES and SELECT privileges to PATS users. Also describes how PATS uses the data in the procedure for rolling data up to Austin to support the VSSC reports and in building data to support local reporting. Columns used are listed and described.
4888 – Use of STD_INSTITUTION table in PATS Data Migration	Describes how PATS uses data migrated from the VistA system to look up values in the std_institution table and populate foreign key values in the PATS tables. Describes how the contingency option to delete all data for an institution uses std_institution data. Columns used are listed and described.
4889 – Use of STD_RACE table in PATS application	Describes PATS foreign key constraint in the pats_patient_race table, which references the std_race table. Also describes how PATS uses the data in the procedure for rolling data up to Austin to support the VSSC reports. Columns used are listed and described.
4890 – Use of STD_ETHNICITY table in PATS application	Describes PATS foreign key constraint in the pats_patient table, which references the std_ethnicity table. Also describes how PATS uses the data in the procedure for rolling data up to Austin to support the VSSC reports. Columns used are listed and described.
4891 – Use of STD_RACE table in PATS Data Migration	Describes how PATS uses the HL7 VALUE field from the VistA system to look up values in the std_race table and populate foreign key values in the pats_patient_race table. Columns used are listed and described.
4892 – Use of STD_ETHNICITY table in PATS Data Migration	Describes how PATS uses the ABBREVIATION field from the VistA system to look up values in the std_ethnicity table and populate foreign key values in the pats_patient table. Columns used are listed and described.

6.0 Business Objects XI (BOEXI)

Standard and ad hoc report structures are built and managed with Business Objects' software. PATS was tested using Business Objects Enterprise XI edition, with Service Pack 1.

6.1 Repository and Central Management Console

The repository is populated during the Import of the Repository Items as described in the *Patient Advocate Tracking (PATS) Installation Guide for HSI Staff*. This section describes the structure of the repository after the import is complete.

After the BOEXI objects are imported onto your server, the database connection information for each of the standard reports must be edited to point to your database (see *Section 6.1.2, Standard Report Database Source*). Additional configuration is described in the *PATS Installation Guide for HSI Staff*, Chapter 6, Setting up Business Objects Enterprise Repository.

The repository contains all of the objects needed to manage both standard and ad hoc reports. It includes standard report templates, the universe and the universe connection to support ad hoc reporting, folder structure to organize reports, users and groups to control access to reports, and other objects. These objects are managed using the BOEXI Central Management Console (CMC).

6.1.1 Report Folder Structure

Standard Reports

The tables in this section show the report structure for standard reports.

Folder PATS – Top Level Public Reports Folder
<i>Subfolder Ad Hoc – Holds all Public Ad Hoc (Site Created) Reports</i>
<i>Subfolder PATS Reports – Holds all Standard Reports</i>

<i>Subfolder Name: Employee</i>	
Report File Name	Description
Emp_Notifications_no_Response.rpt	Lists all Action Request Notifications (ARNs) where the employee has not responded
Employee_Contact_Totals.rpt	Lists ROCs by Employee
Report_by_Employee_Involved.rpt	Lists ROCs by Employee Involved

<i>Subfolder Name: Facility Service Section</i>	
Report File Name	Description
Facility_Serv_Sect_Contact_Totals.rpt	Lists ROCs with issue codes and total ROC count by Facility Service or Section
Facility_Serv_Sect_Issue_Totals.rpt	Lists Issue categories and codes and total Issue count by Facility Service or Section

<i>Subfolder Name: Issue Code</i>	
Report File Name	Description
Customer_Serv_Std_Detail.rpt	Lists the count of ROC issues at the Facilities Service or Section level
Customer_Serv_Std_Summary.rpt	Lists the count of ROC issues at the Division level
Issue_Totals_by_Gender.rpt	Lists Issue Codes and Issue counts for All, Male, or Female patients
Location_Issue_Totals.rpt	Lists Issue categories and codes and total Issue count by Hospital Location
SS_Count_Issues_by_FSOS.rpt	Display Issue Code counts for each Facility Service or Section
SS_Count_Issues_by_Institution.rpt	Display Issue Code counts for each Division
SS_Count_Issues_by_Issue_Cat.rpt	Displays Issue Code counts for each Issue Category
SS_Count_Issues_by_Location.rpt	Displays Issue Code count for each Hospital Location
SS_Count_Issues_by_Sex.rpt	Displays Issue Code count for all Males and all Females
SS_Count_Issues_by_TS.rpt	Displays Issue Code count for each Treatment Status

<i>Subfolder Name: National</i>	
Report File Name	Description
Natl_Patient_Report.rpt	Lists all ROCs at all Divisions for a given Patient
Percent_of_ROCs_with_comps.rpt	National Report Lists Counts of Total ROCs and ROCs with Comps, and Percent of ROCs with Comps by Facility Service or Section

<i>Subfolder Name: Patient</i>	
Report File Name	Description
Daily_List_of_Patient_Contacts.rpt	Lists the daily ROCs with patient data
Patient_Name_Brief_Data.rpt	Lists All ROCs for a single Patient

<i>Subfolder Name: Report of Contact</i>	
Report File Name	Description
Contacts_No_Patient_Identified.rpt	Lists ROCs that have no patient involved
List_of_Open_Cases.rpt	Lists All Open ROCs by Information Taker
Report_of_Contact.rpt	Displays detailed information for a single Report of Contact
Rocs_w_Comps_by_Issue_code.rpt	Lists All ROCs with Comps by Issue Code
SS_Count_ROCs_by_Cont_Entity.rpt	Displays ROC count for each Contacting Entity
SS_Count_ROCs_by_Institution.rpt	Displays ROC count for each Division
SS_Count_ROCs_by_Issue_Code.rpt	Displays ROC counts for each Issue Code
SS_Count_ROCs_by_MOC.rpt	Displays ROC count for each Method of Contact
SS_Count_ROCs_by_TS.rpt	Displays ROC count for each Treatment Status

Ad Hoc Report Structure

Private Folders

Users are allowed to create ad hoc reports and store them in their private folders. Within the BOEXI Central Management Console, these private folders are stored within User Folders. Each user's private folder is named the same as the user's Account Name. When accessed from PATS or from the BOEXI InfoView portal, the users' private folder is called their **Favorites** folder. A user's private folder is automatically created when a user is added to the BOEXI repository. When a user logs on to PATS for the first time, the user is automatically added to the BOEXI repository by PATS (see *Section 6.1.3, Users*).

Public Folder

Users are allowed to create their own Ad Hoc reports from within the PATS application. After creating a report, if users wish to make the report public (available for all other site users), they can move the report into the public AD HOC folder described in the Report Folder Structure table above.

6.1.2 Standard Report Database Source

The database source for each of the standard reports is set up in the same way. Starting from the Folders, navigate to the report object. You can use the Folders structure documented in section *6.1.1 Report Folder Structure* above to help you navigate to the standard reports. Then select the *Process* tab, *Database* option.

1. Click the **Use custom database logon information specified here** button.
2. **Database Type:** Use the default—*Select a database driver*. From the drop-down list, select *Oracle*. We use the Oracle Native Driver.
3. **Server:** Enter the name of the server where the Oracle tables reside.
4. **Database:** Enter the SID or service name for your database.
5. **User:** Enter **PATSRPTS**, the name of the schema where the reports data resides.
6. **Password:** Enter the password that you assigned to the PATSRPTS user.
7. **Table Prefix:** Select *PATSRPTS* from the drop-down list. Select the *Use default table prefix* option.
8. Verify that the *Use same database logon as when report is run* box is checked.

6.1.3 Users

When a user logs on to the PATS application, PATS automatically adds the user to the BOEXI repository. Their Account Name is set to the KAAJEE User Identifier. The password for every user is set to the value retrieved from the `gov.va.med.pats.businessobjects.enterprise.properties` file described in section 1.2 of this guide. This allows the PATS application to log a user on to the BOEXI software invisibly.

In addition, the PATS application automatically enrolls each user in the `PATS_REPORTS_GROUP` group. Members of this group automatically inherit certain rights that are described in detail in the Security section below.

NOTE: The PATS application also adds each user to the Oracle table `user_station_list_for_ad_hoc` in the PATSRPTS schema of the PATS Oracle database. This table is used by the WEBI Universe created for PATS ad hoc reporting. The table is used to control data filtering so that users are allowed to see only station data that they are allowed to access (See Universes below for details).

6.1.4 Groups

The Import creates a single User Group in the BOEXI repository. This group is used to control access to both standard and ad hoc reporting. The group is named `PATS_REPORTS_GROUP`. Users are automatically added to the BOEXI repository and are enrolled as members of this group when they log on to the PATS application. The Security section below describes details of the rights given to members of this group.

6.1.5 Instance Limits

Instance limits on standard reports

The instance limit for the PATS REPORT folder is set to 500. That means that, for example, there can be up to 500 instances of the Employee Contact Totals report stored in the PATS REPORTS folder at any one time. If any user creates a 501st instance of that report, then the oldest instance of that report in the folder will be automatically deleted, regardless of who generated it.

Within the PATS REPORTS folder, the `PATS_REPORTS_GROUP` has an instance limit of 3. This means that any user who belongs to the group, which is all users of PATS, can have a maximum of 3 instances of any single report within the PATS REPORTS folder. So if a user already has 3 instances of the Employee Contact Totals report, and they generate a 4th instance, their oldest instance is deleted.

Instance limits on ad hoc reports

The PATS system does not support multiple instances of ad hoc reports. For that reason, there are no instance limits on the Ad Hoc report folder, or on the individual user's folders.

6.1.6 Security

Along with bringing the folders and the user group, the import brings in the rights assigned to users who are members of the `PATS_REPORTS_GROUP` group as described in this section.

Folder and Report Access Assigned to the `PATS_REPORTS_GROUP`

Standard Reports Folders

Advanced Rights are assigned to the group `PATS_REPORTS_GROUP` for the PATS Folder. The PATS Reports subfolder and all the subfolders within it, inherit all rights from the PATS Folder. This controls user access to the standard reports. The rights that the group `PATS_REPORTS_GROUP` has for the folder PATS are set up as shown on the next page.

Home > Folders > PATS > Advanced Rights

Choose which rights group "PATS_REPORTS_GROUP" has for the object "PATS":

Click Apply to see the updated Inherited rights. Rights that are neither granted nor denied (through inheritance or access level) are treated as denied.

"PATS_REPORTS_GROUP" will inherit rights from its parent groups

"PATS" will inherit rights from its parent folders

Inherited	Explicitly Granted	Explicitly Denied	Not Specified	The Right To:	OK	Cancel	Apply
General							
<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	Add objects to the folder			
<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	View objects			
<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	Edit objects			
<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	Modify the rights users have to objects			
<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	Schedule the document to run			
<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	Delete objects			
<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	Define server groups to process jobs			
<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	Delete instances			
<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	Copy objects to another folder			
<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	Schedule to destinations			
<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	View document instances			
<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	Pause and Resume document instances			
<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	Securely modify rights users have to objects.			
<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	Reschedule instances			
<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	Schedule on behalf of other users			
<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	View objects that the user owns			
<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	Edit objects that the user owns			
<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	Modify the rights users have to objects that the user owns			
<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	Delete objects that the user owns			
<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	Delete instances that the user owns			
<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	View document instances that the user owns			
<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	Pause and Resume document instances that the user owns			
<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	Securely modify rights users have to objects that the user owns.			
<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	Reschedule instances that the user owns			
Report							
<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	Print the report's data			
<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	Refresh the report's data			
<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	Export the report's data			
<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	Download files associated with the report			
Text							
<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	Allow discussion threads			
Web Intelligence Document							
<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	Refresh the report's data			

Patient Advocate Tracking System (PATS)

<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	Edit Query
<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	Refresh List of Values
<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	Use Lists of Values
<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	View SQL
<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	Export the report's data
<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	Download files associated with the object

Ad Hoc Reports Folder

Advanced Rights are assigned to the group PATS_REPORTS_GROUP for the AD HOC subfolder. Though this subfolder descends from the PATS folder, these advanced rights override the rights given to the PATS folder. This controls user access to ad hoc reporting. The rights that the group PATS_REPORTS_GROUP has for the folder AD HOC are set up as shown:

Home > Folders > PATS > Ad Hoc >

Advanced Rights

Choose which rights group "PATS_REPORTS_GROUP" has for the object "Ad Hoc":

Click Apply to see the updated Inherited rights. Rights that are neither granted nor denied (through inheritance or access level) are treated as denied.

"PATS_REPORTS_GROUP" will inherit rights from its parent groups

"Ad Hoc" will inherit rights from its parent folders

Inherited	Explicitly Granted	Explicitly Denied	Not Specified	The Right To:
<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	Add objects to the folder
<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	View objects
<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	Edit objects
<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	Modify the rights users have to objects
<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	Schedule the document to run
<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	Delete objects
<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	Define server groups to process jobs
<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	Delete instances
<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	Copy objects to another folder
<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	Schedule to destinations
<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	View document instances
<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	Pause and Resume document instances

<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	Securely modify rights users have to objects.
<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	Reschedule instances
<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	Schedule on behalf of other users
<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	View objects that the user owns
<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	Edit objects that the user owns
<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	Modify the rights users have to objects that the user owns
<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	Delete objects that the user owns
<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	Delete instances that the user owns
<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	View document instances that the user owns
<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	Pause and Resume document instances that the user owns
<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	Securely modify rights users have to objects that the user owns.
<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	Reschedule instances that the user owns
Report				
<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	Print the report's data
<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	Refresh the report's data
<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	Export the report's data
<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	Download files associated with the report
Text				
<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	Allow discussion threads
Web Intelligence Document				
<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	Refresh the report's data
<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	Edit Query
<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	Refresh List of Values
<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	Use Lists of Values
<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	View SQL
<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	Export the report's data
<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	Download files associated with the object
				OK Cancel Apply

User Access to Ad Hoc Data

Users belonging to the PATS_REPORTS_GROUP group are granted **View on Demand** access to the Web Intelligence universe named **PATS Universe**, built to support Ad Hoc reporting, and to the universe connection named **PATS Oracle 10g** used to connect to the PATS data in the underlying Oracle database.

Data in the Ad Hoc Universe is filtered by user. Each user is allowed to access only data for the sites to which that user has access. Details are described in the Universe Data Filtering section within the WebIntelligence section below.

6.1.7 Universe Connections

The universe connection, **PATS Oracle 10g**, is used to make the database connection between the Web Intelligence Universe used for Ad Hoc reporting, and the underlying PATS data in the Oracle 10g database. The Import installs this universe connection in the BOEXI repository. The universe connection is created and maintained by the PATS developer using the BOEXI Universe Designer rich client application. See the Universe Connection section 6.2.2 for details on how the **PATS Oracle 10g** connection was designed, as well as how to change it to reference a different database if necessary.

6.1.8 Universes

The Web Intelligence universe that supports ad hoc reporting is called the **PATS Universe**. The Import installs this universe in the BOEXI repository. A universe is a meta-data mapping that hides the complexity of the underlying PATS data used for ad hoc reporting. This structure provides end users with user-friendly field names that can be used to create ad hoc reports. The universe is created and maintained using the BOEXI Universe Designer rich client application. This tool is also used to connect the Universe with its Universe Connection to the database. There is no Object Level Security associated with the PATS Universe. The PATS_REPORTS_GROUP, to which all PATS users belong, is given **View on Demand** rights to the **PATS Universe**. See section 6.2.3 for details on how the developer designed the PATS Universe.

6.1.9 Licensing

Production will run with a CPU license for BOEXI. The CMC is used to add the license key. When the license key is entered, the associated items on the screen are automatically updated (i.e. Named Users, Concurrent Users, Processors and Expires). Note that the licensing information is not installed by the Import, but must be added manually by the server Administrator.

6.2 WebIntelligence

6.2.1 PATS/WebLogic – WebIntelligence/Tomcat integration notes

PATS hosts Web Intelligence via an IFrame to two web pages deployed in InfoView, openQueryPanelWithToken.jsp (an unsupported file distributed by Business Objects consultants for creating a new ad hoc report) and openDocument.jsp (opens existing ad hoc reports.)

The pros and cons of this implementation are:

Pros:

- Single sign on between two distributed web applications achieved
- Seamless integration fulfills the common look and feel that PATS and WebIntelligence (WEBI) look similar

Cons:

- Two stateful web applications must be managed with distinct objects. PATS holds a Business Objects IEnterpriseSession and so does InfoView (which runs WEBI) in Tomcat. Additionally each application has an HttpSession. These two sessions are not synchronized. Therefore, the PATS application can time out while a user is working in WEBI. To accommodate this, the timeout in PATS web pages containing WebIntelligence links is set to one hour while the timeout on the rest of the PATS web pages is set to 20 minutes. While a user is working in WEBI, the timeout on the PATS application running in the background is one hour, so PATS is less likely to time out.
- Additionally, the multitude of session instances (HTTP and Business Objects) can introduce scalability issues.

6.2.2 Universe Connection

The PATS Oracle 10g universe connection object is used to make the database connection between the PATS universe and the PATS tables that support ad hoc reporting. The BOEXI Designer application was used to create both the PATS Universe and the PATS Oracle 10g connection.

To access the Designer application:

1. Select **Start>All Programs>BusinessObjects 11>BusinessObjects Enterprise>Designer**.
2. Select **Tools>connections**.

When the PATS Oracle 10g database was originally designed, the following information was entered in the wizard prompts.

Wizard Prompt	Setting
Database Middleware	Navigate to Oracle>Oracle 10>Oracle Client
Type	Secured

Wizard Prompt	Setting
User name	PATSRPTS
Password	Password established for the PATSRPTS Oracle user
Keep the connection active during the whole session (local mode only)	Select this option
Array fetch size	100
Array bind size	50
Login timeout	600
Binary Slice size	32000
Hint value	null

To edit the PATS Oracle 10g Universe Connection (for example, to connect to a different database or to change the database password):

1. Select PATS Oracle 10g from the list.
2. Click Edit.

Result: A wizard is started that allows you to edit the parameters.

6.2.3 Universe

The universe used for ad hoc reporting, **PATS Universe**, was created by the PATS developer using the BOEXI Designer application. The Import installs this universe in the BOEXI repository

- The table ROC_COMBO_DATA_VIEW is a single table that combines data from joins of the Report_of_contact table with all of the tables that it references, and all of the tables that reference it. All of the columns that the user sees come from this table. This table is refreshed from the PATS application table data nightly.
- The table USER_STATION_LIST_FOR_AD_HOC contains a list of User/Station pairs, and is used to filter the data as described in the Security section below.
- There is a many-to-many join between the two tables on the STATION_NUMBER columns.
- All columns were transferred from the underlying ROC_COMBO_DATA_VIEW table to the universe.
- Several new dimensions were created from underlying columns to assist the user in designing reports. For example, the issue code and the issue code name have been concatenated into a new dimension since report designers often want to show the two fields together.

6.2.4 Universe Data Filtering

Data is filtered so that when a user does ad hoc reporting from the PATS Universe, they are allowed to see only data for stations to which they have access. This section describes the steps taken by the PATS developer to implement the data filtering.

When a user logs on to the PATS application, the KAAJEE log on service returns a list of accessible station numbers for that user. The PATS application uses that list to build entries consisting of KAAJEE User Identifier/Station Number pairs in the USER_STATION_LIST_FOR_AD_HOC table in the database.

Adding the User/Station table to the Universe: In the BOEXI Designer application, the USER_STATION_LIST_FOR_AD_HOC table was added as a table to the **PATS Universe**, with a many-to-many relationship to the ROC_COMBO_DATA_TABLE. The join is between the STATION_NUMBER fields on each table. The USER_STATION_LIST_FOR_AD_HOC table was then made into a class in the PATS Universe. This class is hidden from the end users.

Forcing connection between a ROC and users(s) who have access to the station associated with the ROC: An enforced join was established between every field in the main data class, and the User Station List For Ad Hoc class. Because of this, whenever a Web Intelligence query is made involving any field on the main data table ROC_COMBO_DATA_TABLE, a join on station number will automatically be made to the USER_STATION_LIST_FOR_AD_HOC table as well. This makes the connection between every ROC, and every user who has access to the station number to which that ROC belongs.

The enforced join was made as follows:

1. On the Properties for each field, click on the **Tables** button.
2. Highlight both the ROC_COMBO_DATA_VIEW table and the USER_STATION_LIST_FOR_AD_HOC table.

Security Restriction linking the current user to their accessible divisions list: The next step in creating the filter was to set up a Security Restriction based on the current user logged on to Web Intelligence. This security restriction will create an addition to the WHERE clause for any query against the USER_STATION_LIST_FOR_AD_HOC_USER table, that restricts the selection to rows where the USER_IDENTIFIER matches the BOUSER variable. BOUSER is a system variable set by WEBI to the user identifier of the user currently logged on to WEBI. Our security restriction was set up as follows:

1. Select **Tools>Edit Security Restrictions>New**.
2. Select the **ROWS** tab.
3. Enter the name of the security restriction **RestrictUsersByStation**.
4. Under Restricted Tables, press **Add**.
5. In the Table box, press **>>**.
6. Select the **USER_STATION_LIST_FOR_AD_HOC**.

7. In the Where clause box, press >>.
8. Create a Where clause that says:
**USER_STATION_LIST_FOR_AD_HOC.USER_IDENTIFIER =
@Variable('BOUSER')**

Note: The BOEXI developers warned that in future versions of the Business Objects Web Intelligence application, the maintenance developers for the PATS application may need to change this code to use a new variable **DBUSER** instead of **BOUSER**.

Apply the security restriction to all PATS Users: The final step to create the data filter by station was to apply the new Security Restriction to the PATS_REPORTS_GROUP. This was done as follows:

1. Select **Tools>Apply Security Restrictions**.
2. From the Manage Access Restrictions dialogue box click the Users or Groups icon.
3. From the Select Users and Groups dialogue box highlight the group PATS_REPORTS_GROUP, press **Select** then close the box.
4. The user group appears in the Manage Access Restrictions user list. Highlight the user group and select the restriction set **RestrictUsersByStation** from the Restriction Set drop-down list.
5. Click **Apply**.
6. Click **OK**.

6.3 Business Objects Software

6.3.1 Crystal Reports XI

All of the standard reports used by PATS run under Crystal Reports XI. Crystal Reports is a rich client application used to design reports. The reports all use the output from Oracle stored procedures in the PATSRPTS schema. Information about the stored procedures themselves, the associated report templates, the folder structure within which the reports are stored, and the configuration of the reports has been previously described in the Database – Oracle chapter, and the Repository and Central Management Console section of this chapter.

6.3.2 WebI Universe Designer XI

The **PATS Universe** used within the Web Intelligence product for all ad hoc reporting was designed using the Business Objects Enterprise XI Designer tool. The Designer is a rich client application used to define database connections, and to create universes, a meta-layer on data dictionary tables that can be viewed from the Web Intelligence reporting product.

6.3.3 Central Management Console

The Business Objects Enterprise XI Central Management Console is a web based application that is used to configure all of the objects in the BOEXI Repository. This includes all of the objects needed to store and support both standard and ad hoc reports.

6.3.4 InfoView

This is a web based portal that gives end users access to the folder structure and to existing reports in the repository to which they have access (both Crystal Reports and Web Intelligence Reports). It is part of the Business Objects suite. It also allows them to create new Web Intelligence reports, to save them locally, and if they have access, to move them to other locations. In order to maintain single sign-on for end users, users of the PATS application will not use the InfoView product directly. Instead, Web Intelligence and portions of InfoView have been incorporated into PATS and can be launched directly from PATS.

6.3.5 Web Intelligence

Web Intelligence is a web based ad hoc query builder/report generation tool that is part of the Business Objects suite. This allows users to connect to the PATS database and to create queries and simple reports. PATS incorporates access to this tool from within the PATS application.

6.4 Multiple PATS Instances Sharing the Same BOEXI Repository

The HSI has multiple instances of the PATS application sharing the same BOEXI repository. This section describes the setup of the Hines server BOEXI repository at the present time. The future plan for production will be somewhat different. Current plans indicate that the BOEXI repository at Falling Waters will host the production and support data and the repository at Hines will host the training and failover production data.

BOEXI Folder Structure at the HSI

Folder	Sub-Folder
PATS	Ad Hoc PATS Reports Employee Facility Service Section Issue Code National Patient Report of Contact

Folder	Sub-Folder
PATS_Staging	Ad Hoc – Staging PATS Reports – Staging Employee Facility Service Section Issue Code National Patient Report of Contact
PATS_Training	Ad Hoc – Training PATS Reports – Training Employee Facility Service Section Issue Code National Patient Report of Contact

6.4.1 Standard Reports

1. The PATS folder contains all of the sub-folders and objects for the PATS production instance. Two additional folders were created at the same level as PATS—one for the Staging instance of PATS (**PATS_Staging**) and one for the Training instance (**PATS_Training**).
2. In each of these new top level folders, a new folder was created to hold the standard reports—**PATS Reports – Staging** and **PATS Reports – Training** (see diagram above).
3. The folder structure and objects in the **PATS Reports** folder were copied into the two new folders (see diagram above).
4. The database connection information for all standard reports in the new staging and training sub-folders was changed using the CMC to connect to the staging (or training) database.
5. The gov.va.med.pats.businessobjects.enterprise.properties files for both the staging and training instances of the application were edited. For the training instance:
 - PatsPublicFolderName was changed from **PATS** to **PATS_Training**
 - PatsCmsFolderName was changed from **PATS Reports** to **PATS Reports – Training**

6.4.2 Ad Hoc Reports

1. In each of the new top level folders described in step 1 above, **PATS_Staging** and **PATS_Training**, a new folder was created to hold the public ad hoc reports, **Ad Hoc – Staging** and **Ad Hoc – Training**. (See the diagram above.)

2. The **PATS Universe** was cloned two times under new names for the staging and training accounts (**PATS Universe – Training** and **PATS Universe – Staging**).
3. The **PATS Oracle 10G** universe connection was cloned two times under new names for the staging and training accounts (**PATS Oracle 10G – Training** and **PATS Oracle 10G – Staging**).
4. The new universes for staging and training were edited using the BOEXI Designer tool to use the new universe connections.
5. The database connection information for the new universe connection objects was changed using the BOEXI Designer tool, to connect to the staging (or training) database.
6. The gov.va.med.pats.businessobjects.enterprise.properties files for both the staging and training instances of the application were edited. For the training instance: UNIVERSE_NAME was changed from **PATS Universe** to **PATS Universe – Training**.

7.0 Rollup PATS Data to VSSC

7.1 Scheduled Job to Build Rollup Data

The installation of the PATS application will set up an Oracle scheduled job named BUILD_ROLLUP_DATA_JOB in the PATS schema of the Oracle database. The job is initially disabled when PATS is installed. The HSI and the VSSC staff (in Austin) will determine the schedule for running this Oracle job to build the data and for running the job at the VSSC to fetch the data from the PATS Oracle database. The job will be run once a week.

This job executes the ROLLUP method from the Package PATS.PKG_ROLLUP_NATL_DATA. This method:

- Deletes all data from the PATS.PATS_ROLLUP_NATL_DATA table.
- Selects all entries on the REPORT_OF_CONTACT table that have the ROLLUP_TO_NATL_REPORTS_STATUS flag set to 1, and that have at least one entry from the ROC_ISSUE table referencing the ROC.
- For each ROC that meets the criteria, creates records on the PATS_ROLLUP_NATL_DATA table.

7.2 VSSC Fetches Rollup Data

Once a week, at a time agreed upon between the HSI and the VSSC (in Austin), a job initiated by the VSSC connects to the PATS database as the PATSROLLUP user. This user has only SELECT access to the PATS_ROLLUP_NATL_DATA table. The job reads the data and transfers it into the VSSC system to support national reporting. If the password for the PATSROLLUP user is changed, you will need to notify the VSSC.

7.3 Format of Rollup Data Records

The records for a single ROC will be sequenced in the order shown below. There will be at least one of each type of record for every ROC, even if that record contains no data. All fields in each record are separated by a single up-arrow ^, the final field in each record is followed by a single up-arrow ^.

7.3.1 ROC Main Data Records

There will be just one **ROC** record per Report of Contact.

Fields Transferred	Format Information	Data Type
ROC Number	New: Station_Number.YYYYNNNNNN	String (13:16)
ROC	Literal Value	String (3)
Date of Contact	MMDDYYYY	Integer (8 digits)
Status	O or C (open or closed)	Char (1)
Resolution Date	MMDDYYYY (or null)	Integer (8 digits)

Fields Transferred	Format Information	Data Type
Treatment Status	O (Outpatient), I (Inpatient), E (Long Term Care) or null	Char (1)
Station Number	From SDSADM.STD_INSTITUTION table	String (3:7)
Days to Resolution	Calculated (date_of_contact – date_closed)	Integer (between 1-999)
VISN Number	From SDSADM.STD_INSTITUTION table	Integer (between 1-23)
Is Clinical Appeal	1=true, 0=false	Integer (0 or 1)
Info Taken By Name	Last,First Middle Suffix	String (3:60)
Congressional Contact		String (1:30)
Was Comp Given	1=true, 0=false	Integer (1 digit)
Comp Name		String (1:30)

7.3.2 Patient Data Records

There will be just one **PT** record per report of contact.

Fields Transferred	Format Information	Data Type
ROC Number	New: Station_Number.YYYYNNNNN	String (13:16)
PT	Literal Value	String (2)
SSN	9 numeric, plus 'P' if pseudo	String (9:10)
Sex	M or F	Char (1)
DOB	MMDDYYYY	Integer (8 digits)
Period of Service		String (1:25)
Eligibility Status	Set to UNK if null	String (1:30)
Patient ICN	Internal Control Number (person identifier)	String (1:29)
Category	Current Means Test when ROC entered	String (1:30)
Ethnicity		String (1:30)

7.3.3 Issue Multiple Records

There may be more than one **ISSC** record per report of contact.

Fields Transferred	Format Information	Data Type
ROC Number	New: Station_Number.YYYYNNNNN	String (13:16)
ISSC	Literal Value	String (4)
Issue Code	Same as in Patient Rep	String (4:5)
Issue Category Code	First 2 characters of Issue Code	String (2)
Is Customer Service Standard	1=true, 0=false	Integer (1)
Hospital Location		String (1:30)
Facility Service or Section	External Value from PATS	String (0:50)

7.3.4 Contacting Entity Records

(In Patient Rep this was Contact Made By)

There may be more than one **CE** record per report of contact.

Fields Transferred	Format Information	Data Type
ROC Number	New: Station_Number.YYYYNNNNN	String (13:16)
CE	Literal Value	String (2)
Contacting Entity	RE=Relative FR=Friend CO=Congressional VI=VISN HQ=Central Office VO=Veterans Service Organization AT=Attorney/Legal Guardian/Conservator/Trustee DI=Director's Office ST=Staff – VAMC OT=Other VH=VISN/HQ (inactive) PA=Patient	String (2)

7.3.5 Method of Contact Records (In Patient Rep this was Sources of Contact)

There may be more than one **MOC** record per report of contact.

Fields Transferred	Format Information	Data Type
ROC Number	New: Station_Number.YYYYNNNNN	String (13:16)
MOC	Literal Value	String (3)
Method of Contact	P (phone), V (Visit), I (Internet), L (letter), S (survey)	Char (1)

7.3.6 Patient Race Records

There may be more than one **RACE** record per report of contact.

Fields Transferred	Format Information	Data Type
ROC Number	New: Station_Number.YYYYNNNNN	String (13:16)
RACE	Literal Value	String (4)
Race		String (1:30)


8.0 Troubleshooting PATS Reports

This section covers the following situations:

1. Reports Server fails to generate a report for no apparent reason, or the report server is behaving erratically.
2. A user logs into PATS and is informed that the Reports Server is not available yet it appears to be running; the Windows services and the servers in the console are enabled.
3. A previous instance of a report is missing.
4. After deploying a new report to the Crystal Enterprise repository, verify that the report can be run successfully.
5. The user does not have access to reports in PATS. There are no errors regarding the availability of reports; just the Reports option is not available.
6. The user runs a report and gets results that do not meet the parameter values entered
7. User can generate a report, but most recent data is not showing on the report.

Scenario 1: Reports Server fails to generate a report for no apparent reason or the report server is behaving erratically.

Check

- Select **Start>Control Panel>Administrative tools>Services** to verify that all Crystal Reports and Web Intelligence services on the server have a status of started.
- Log into Central Management Console (CMC) and select the Servers option. Verify that all listed servers have the green arrow () indicating that the server is enabled.
- In Administrative tools>Services try stopping and restarting the Crystal and Web Intelligence services.
- In the Central Management Console select the Servers option. Then try disabling and enabling the servers. Select a server by clicking in the **Selected** box, then click **Disable** then **Enable** buttons in the top right part of the screen.

Scenario 2: A user logs into PATS and is informed that the Reports Server is not available yet it appears to be running and the Windows services and the servers in the console are enabled.

Check

- Log into the Central Management Console. Select the Users option and check that the user appears in the list of users, and that the user is a member of the PATS REPORTS GROUP group.
- If not, have the user log off the PATS application, then log back on.

Scenario 3: A previous instance of a report is missing. Instance Limits have been set up as described in the Business Objects XI chapter above. If the instance limits are exceeded, the oldest instances of reports are purged automatically. The user may have to re-run the report.

Check

- You may choose to increase the instance limits in the Central Management Console
- Or, inform the user that if they want to be sure of saving old copies of reports, they should export them to their own PC.

Scenario 4: After deploying a new report to the Crystal Enterprise repository, verify that the report can be run successfully.

Complete the following tasks:

1. In the Central Management Console, select the newly deployed report.
2. Click the **Properties** tab and select the **Preview** option.
3. For each parameter listed in the parameters table set a value for which you know data exists.
4. Click **OK** to execute the report.

Scenario 5: The user does not have access to reports in PATS. There are no errors regarding the availability of reports, but the reports options do not appear in the main menu of PATS.

Check

- In order to perform reporting PATS the user must fall into one of three roles: SRCU, NPO or ADUSH.
- Verify in VistA that the user has one of the roles QACV_SRCU, QACV_NPO or QACV_ADUSH.
- Check in the CMC to make sure the user exists and has been assigned to the reports group as in Scenario 2.

Scenario 6: The user runs a report and gets results that do not meet the parameter values entered.

Check

- If default values have been set for the report in the CMC, and no value is selected for a parameter in PATS (for example, running the Employee Contact Totals report leaving the Employee parameter null to get all employees), the default values set in the CMC will be used to generate the report.
- From the CMC, select the report. From the Process tab, select Parameters. Make sure all of the parameters are set to null (empty). The 508C flag can be set to 0 instead of null.

Scenario 7: User can generate a report, but the data they expected is not showing on the report.

Check

- Did the user enter the data today? The report data comes from tables whose data is refreshed daily at a time determined by the HSI, so data entered one day will not appear on reports until the following day.
- Did the user select a Date of Contact date range that was too old? The reports data tables contain data only for *the current and previous two fiscal years*.
- Check to see whether the data is in the production tables in the PATS schema, then check to see whether it is in the tables used for reporting in the PATSRPTS schema. If the data is in the reporting tables, first check the user's input parameters, then see previous scenario.
- If data is in the production tables but not in the reporting tables, check the scheduled job `bld_std_report_data_job` to make sure it is still running nightly to refresh the data in the tables that support standard reporting.
- If data is in the production tables but not in the reporting tables, check the procedure `pkg_bld_std_report_data` to see whether it needs to be recompiled. If the underlying production data table structures change, this procedure may need to be altered.
- Check the tables in the PATSRPTS schema. If the structure of the underlying production tables in the PATS schema changes, it is possible that the structure in these tables must also be changed to reflect the new data format.
- After correcting the problem, you can refresh the data by directly calling the procedure `patsrpts.pkg_bld_std_report_data.bldall` to refresh the data manually.
- Monitor the scheduled job for a few days to make sure that the refresh is now working.