# Document Attributes and Revision History

| Attribute | Value |
| --- | --- |
| Owner | Arun Kumar Madas |
| Owner Contact Information | Am6489@att.com |
| *Other Attribute* | Rally Project Id : 274829, Iteration 1   * US255258 : (1)EPIC --- Provide a GUI for users to display accounts for IRU audit * US255314 : (9) EPIC --- Profiles |
| *Release* | 1410 Release |

## Revision History

| Author | Date | Version # | PMT #ID  Rally # ID | Applications | Revision Description |
| --- | --- | --- | --- | --- | --- |
| Arun Madas (am6489) | 07/14/2014 | 0.1 | PMT#430664  Rally # 274829 | OPUS/OM | Initial Draft |

Table of Contents

[Document Attributes and Revision History 1](#_Toc385928016)

[Revision History 1](#_Toc385928017)

[Introduction 3](#_Toc385928018)

[Component List 3](#_Toc385928019)

[Detail Design 3](#_Toc385928020)

[Functional Overview 4](#_Toc385928021)

[Middleware Design - Presentation Tier 5](#_Toc385928022)

[Middleware Design – Business Tier 17](#_Toc385928023)

[Middleware Design – Externalized Configuration 23](#_Toc385928024)

[Database Tier Design 25](#_Toc385928025)

[Other Information and References 29](#_Toc385928026)

[Acceptance & Approvals 30](#_Toc385928027)

[Overview 30](#_Toc385928028)

[Approvers 30](#_Toc385928029)

# Introduction

The Design Package describes the components of the solution that will be created or modified and how they will be created or modified in order to meet the requirements.

## Component List

This section lists each component that will be created, modified, or removed.

* Opus mobile Urgent message

## Detail Design

This section describes the changes for each listed component sufficiently to meet the needs of the people identified to develop, test, deploy, and support the solution.

|  |  |  |
| --- | --- | --- |
| **Req. ID** | **Design Component** | **Trace-To** |
| 274829.OPUS\_OM.AD.01 | OPUS-OM Authentication of users   * Uses Existing opus login mechanism (BAU) – Users will be using the Global Login and myLogins will be used to add the new users to the OPUS system for IRU (roles/permissions) * 3 New User Roles are created   + IRU Audit Tool – Auditor   + IRU Audit Tool – Manager   + IRU Audit Tool – Super User * After user login, check new retail location type = IRU, and redirect to IRU functionality * No ON/OFF Flag is present * Logged in user Roles are identified and based on appropriate permissions users are presented UI appropriately * New user roles will have permissions from existing OPUS DB. We will be creating new permissions for each of the new tabs and assign to the new IRU roles. | US261562  US255314 |
| 274829.OPUS\_OM.AD.02 | OPUS-OM Populating in-store and FAX queues  In-Store Queue   * In-Store Queue is available in opus database IRU\_FAN\_POE table * After the record is assigned first time from IRU\_FAN\_POE table, we will add another entry into IRU\_AUDIT\_RECORD\_HISTORY & IRU\_AUDIT\_RECORD\_LATEST\_VERSION with validation status = Audit In Progress. * If Auditor puts the record back in pending Queue, then we have a new pool of records in IRU\_AUDIT\_RECORD\_LATEST\_VERSION with status = Pending Audit in addition to the records in IRU\_FAN\_POE table (the combination forms the in-store que)   Fax Queue   * Fax Queue need to be built (not part of iteration 1)   + Logic to build Fax Que   + 🡺 Customer Sends FAX   + 🡺 Recieves FAX in a network Share location   + 🡺 Cron job runs to pick up the network share location documents and push to database table and removes it from network share   + 🡺 Records in the new table iru\_audit\_cron\_fax\_drive2db form the fax que (in iteration1, we assume that records exist in the db) | US270433 |
| 274829.OPUS\_OM.AD.03 | OPUS-OM systemic assignment of records for audit  **In-Store Queue Assignment:**   * Create a new enterprise\_config IRU\_AUDIT\_INSTORE\_ASSIGNMENT\_TIMELAG\_MINS * Default value = 12 hrs \* 60 mins = 720 mins * Record gets assigned when user selects radio button from in-store queue with delay of configurable 720 mins lag from in-store record created date time. OLDEST record gets assigned from the queue as Vendor has 48 hr SLA to meet.   **Fax Queue Assignment LAG ?**   * Process of Record Assignment and DB entries for in-store * Process of Record Assignment and DB Entries for FAX Record * Oldest record gets assigned first. Time when Record was synched by Cron2DB. | US262880 |
| 274829.OPUS\_OM.AD.4 | OPUS-OM IRU audit GUI - perform audit options  OPUS-OM IRU audit GUI for managers - audit an in-store record  OPUS-OM IRU audit GUI for super-user - audit an in-store record  OPUS-OM IRU audit GUI for auditors - audit an in-store record  **Perform Audit – In-Store Record**   * FAST FAN Lookup * Pull Information from Telegence * Show Thumbnail image of POE * Click to open enlarged image of POE Image * Validation Status Change & Submit   + Pass / Pending Audit – Does not require ReasonCode & Notes   + Fail – Mandates Reason Code & Notes, Optional Contact Process & Contact Updates | US273277  US270428  US272364  US270436 |
| 274829.OPUS\_OM.AD.5 | OPUS-OM IRU audit GUI for managers - audit a FAX record  OPUS-OM IRU audit GUI for super-user - audit a FAX record  OPUS-OM IRU audit GUI for auditors - audit a FAX record  **Perform Audit – Replacement FAX POE**   * FAST FAN Lookup * Pull Information from Telegence * Show Thumbnail image of POE * Click to open enlarged image of POE Image * Discard – In-Activate Record for FAX Queue * Fetch – Map AUDIT\_RECORD\_ID * Validation Status Change & Submit   + Pass / Pending Audit – Does not require ReasonCode & Notes   + Fail – Mandates Reason Code & Notes, Optional Contact Process & Contact Updates   **Perform Audit – Fresh FAX Application**   * FAST FAN Lookup * Pull Information from Telegence * Show Thumbnail image of POE * Click to open enlarged image of POE Image * Discard – In-Activate Record for FAX Queue * Capture POE\_TYPE, COMPANY\_NAME, FAN, BAN * Validation Status Change & Submit   + Pass / Pending Audit – Does not require ReasonCode & Notes   + Fail – Mandates Reason Code & Notes, Optional Contact Process & Contact Updates | US270946  US272365  US270947 |
| 274829.OPUS\_OM.AD.6 | OPUS-OM IRU audit GUI for managers - re-assign  OPUS-OM IRU audit GUI for super-user - re-assign   * Get all records from latest\_version with status=Audit in progress with (sysdate-assigned\_date) > REASSIGN\_LAG\_MINS * Select a record and display all details * Select record and reassign to single auditor * Hit Move all to pending button | US266615  US272370 |
| 274829.OPUS\_OM.AD.7 | OPUS-OM IRU audit GUI for managers - BAN Tracker  OPUS-OM IRU audit GUI for super-user - BAN Tracker  OPUS-OM IRU audit GUI for auditors - BAN Tracker | US270430  US272366  US262857 |
| 274829.OPUS\_OM.AD.8 | OPUS-OM IRU audit GUI for super-user - Validation Status Reconciliation/True-up  OPUS-OM IRU audit GUI for auditors - Validation Status Reconciliation/True-up | US272373  US264441 |
| 274829.OPUS\_OM.AD.9 | OPUS-OM IRU Audit GUI for super-user - stop customer contact follow-ups  OPUS-OM IRU Audit GUI for auditors - stop customer contact follow-ups | US272375  US267657 |

### Functional Overview

*Provide a link to HLD for a functional description of the module or set of modules covered in this design. This will help set the functional context before delving into the technical implementation outlined in this detailed design document.*

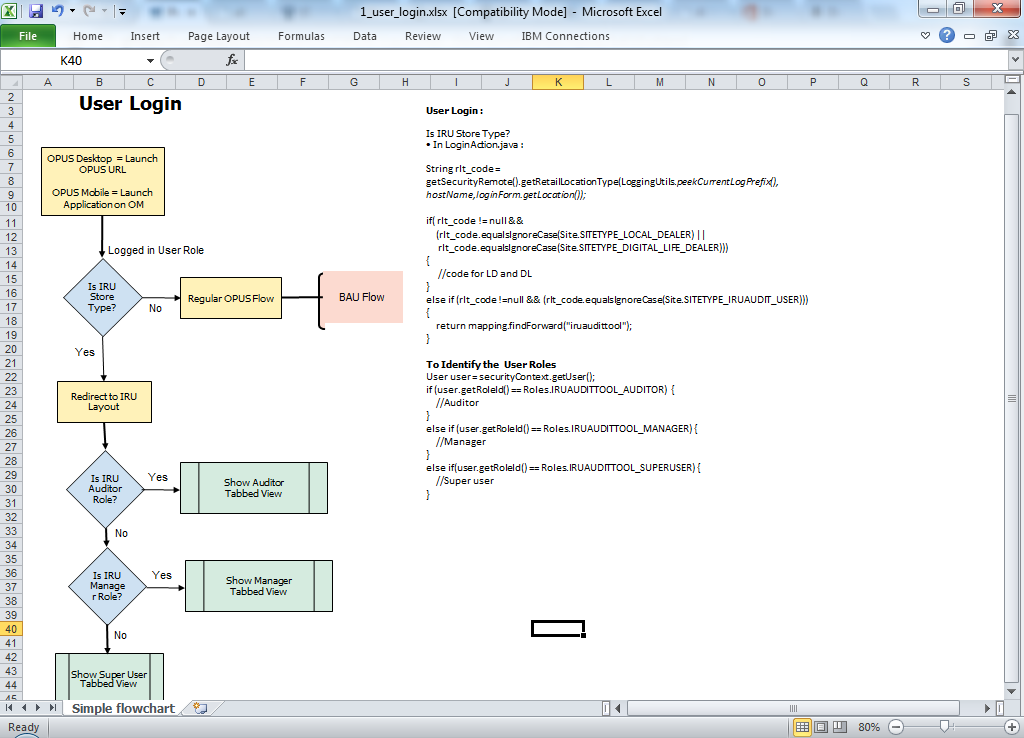
|  |  |  |
| --- | --- | --- |
| **Module Name** | **New/Existing** | **Functional description** |
| IRU Audit Tool | New | When the rep logs in, OPUS and OPUS Mobile shall redirect the user based on the user role (IRU Audit Tool – Auditor, IRU Audit Tool – Manager, IRU Audit Tool – Super User) |
|  |  |  |
|  |  |  |

### Middleware Design - Presentation Tier

*The Middleware Design section in the HLD is decomposed into Presentation Tier Middleware and Business Tier Middleware design in the AD, so that the level of detail is better organized by physical tiers in the implementation architecture.*

*This section captures the detailed design for the presentation tier middleware (frontend components running on the server side). If there is a physically separated business tier (ex. with a classic three-tier architecture), the design for business services (ex. implementation of EJBs) should be elaborated in the Business Tier section. In that scenario, the Presentation Tier design will refer to the interfaces for Business Tier services, but the implementation details of those services will be defined in the Business Tier section of the design****.***

**274829.OPUS\_OM.AD.01 :** OPUS-OM Authentication of users

******

***Database SQL Required for creation of new Retail Location:***

*insert into retail\_location(RL\_NUMBER, RL\_NAME, RL\_IS\_VALID, RLC\_CODE, RL\_DIRECTIONS, RLT\_CODE, RL\_DOMAIN, RL\_TAX\_PERCENT, RL\_JDE\_BANK\_CODE, RL\_OWNER, RL\_ADJUSTMENT\_LIMIT, RL\_ADDRESS1, RL\_ADDRESS2, RL\_CITY, RL\_STATE, RL\_ZIP, RL\_ZIP4, RL\_HOME\_PHONE, RL\_WORK\_PHONE, RL\_FAX\_PHONE, RL\_OASYS\_AGENT, RL\_OASYS\_LOCATION, CS1\_ID, RL\_CC\_DIVISION\_ID, RL\_DC\_MERCHANT\_ID, RL\_GENERATING\_DEPOSIT, ORA\_COMPANY\_CODE, ORA\_FUNC\_CODE, ORA\_DEPT\_CODE, BM\_CODE, RL\_BANK\_NAME, RL\_BANK\_ACCOUNT, RL\_CREATION\_DATE, REGION\_ID, COMPANY\_ID)*

*select new\_rl\_in RL\_NUMBER, RL\_NAME, RL\_IS\_VALID, RLC\_CODE, RL\_DIRECTIONS, RLT\_CODE, RL\_DOMAIN, RL\_TAX\_PERCENT, RL\_JDE\_BANK\_CODE, RL\_OWNER, RL\_ADJUSTMENT\_LIMIT, RL\_ADDRESS1, RL\_ADDRESS2, RL\_CITY, RL\_STATE, RL\_ZIP, RL\_ZIP4, RL\_HOME\_PHONE, RL\_WORK\_PHONE, RL\_FAX\_PHONE, RL\_OASYS\_AGENT, RL\_OASYS\_LOCATION, CS1\_ID, RL\_CC\_DIVISION\_ID, RL\_DC\_MERCHANT\_ID, RL\_GENERATING\_DEPOSIT, ORA\_COMPANY\_CODE, ORA\_FUNC\_CODE, ORA\_DEPT\_CODE, BM\_CODE, RL\_BANK\_NAME, RL\_BANK\_ACCOUNT, RL\_CREATION\_DATE, REGION\_ID, COMPANY\_ID*

*from retail\_location where rl\_number = source\_rl\_in;*

*--insert into RL\_ACTIVATION\_SETTINGS(RL\_NUMBER,BM\_CODE, RL\_AS\_ID, RL\_AS\_VALUE )*

*--select RL\_NUMBER,BM\_CODE, RL\_AS\_ID, RL\_AS\_VALUE from RL\_ACTIVATION\_SETTINGS where rl\_number = source\_rl\_in;*

*--insert into store\_profile(RL\_NUMBER,SP\_OPEN\_TIME, SP\_CLOSE\_TIME, SP\_REGISTER\_OPEN\_AMOUNT, SP\_DEFAULT\_LOCATION1, SP\_DEFAULT\_LOCATION2, SP\_DEFAULT\_LOCATION3, SP\_RECEIPT\_FOOTER, SP\_DEPT\_PHONE\_RANK, SP\_DEPT\_ACC\_RANK, SP\_SELLING\_FEET\_RANK, SP\_TOTAL\_FEET\_RANK, SP\_ALLOW\_CR\_OPEN, JDE\_ADDRESS\_BOOK#, JDE\_SHIP\_TO#, SP\_CASH\_BOUNDARY, SP\_CASH\_WARN\_BOUNDARY, SP\_DISCOUNT\_BOUNDARY, JDE\_TAX\_AREA, JDE\_TAX\_EXPL\_CODE, SP\_NO\_OF\_CONTRACTS, SP\_BWS\_MONTHS, SP\_NO\_OF\_RECEIPTS, SP\_COMPASS\_MARKET\_ID, SP\_IS\_COMPASS\_LOCATION, SP\_INV\_ID, SP\_COMPASS\_CONVERSION\_DATE, TZ\_OFFSET, SP\_IS\_TELEGENCE\_LOCATION, DEALER\_CODE, LOCATION\_CODE, SP\_IS\_CASH\_CHECK\_SPLIT, SP\_STORE\_MANAGER, VALIDATE\_CHECK\_TENDER, SP\_ACCESSORY\_SALE\_ENABLED, SP\_FEATURE\_ADDITION\_ENABLED, SP\_STORE\_SECURITY\_ENABLED, SP\_UPGRADE\_CHECK\_ENABLED, SP\_SEARCH\_CVD, SP\_IS\_CREDITDEBIT\_ENABLED, SP\_IS\_REDATH\_MARKET\_ENABLED, SP\_IS\_FIZ\_EOD\_REPORT\_ENABLED, SP\_REPRINT\_FEE, SP\_DISCOUNT\_NOTE\_REQ)*

*--select RL\_NUMBER,SP\_OPEN\_TIME, SP\_CLOSE\_TIME, SP\_REGISTER\_OPEN\_AMOUNT, SP\_DEFAULT\_LOCATION1, SP\_DEFAULT\_LOCATION2, SP\_DEFAULT\_LOCATION3, SP\_RECEIPT\_FOOTER, SP\_DEPT\_PHONE\_RANK, SP\_DEPT\_ACC\_RANK, SP\_SELLING\_FEET\_RANK, SP\_TOTAL\_FEET\_RANK, SP\_ALLOW\_CR\_OPEN, JDE\_ADDRESS\_BOOK#, JDE\_SHIP\_TO#, SP\_CASH\_BOUNDARY, SP\_CASH\_WARN\_BOUNDARY, SP\_DISCOUNT\_BOUNDARY, JDE\_TAX\_AREA, JDE\_TAX\_EXPL\_CODE, SP\_NO\_OF\_CONTRACTS, SP\_BWS\_MONTHS, SP\_NO\_OF\_RECEIPTS, SP\_COMPASS\_MARKET\_ID, SP\_IS\_COMPASS\_LOCATION, SP\_INV\_ID, SP\_COMPASS\_CONVERSION\_DATE, TZ\_OFFSET, SP\_IS\_TELEGENCE\_LOCATION, DEALER\_CODE, LOCATION\_CODE, SP\_IS\_CASH\_CHECK\_SPLIT, SP\_STORE\_MANAGER, VALIDATE\_CHECK\_TENDER, SP\_ACCESSORY\_SALE\_ENABLED, SP\_FEATURE\_ADDITION\_ENABLED, SP\_STORE\_SECURITY\_ENABLED, SP\_UPGRADE\_CHECK\_ENABLED, SP\_SEARCH\_CVD, SP\_IS\_CREDITDEBIT\_ENABLED, SP\_IS\_REDATH\_MARKET\_ENABLED, SP\_IS\_FIZ\_EOD\_REPORT\_ENABLED, SP\_REPRINT\_FEE, SP\_DISCOUNT\_NOTE\_REQ*

*--from STORE\_PROFILE where rl\_number = source\_rl\_in;*

*insert into USERS\_RL(RL\_NUMBER,USER\_ID, IS\_ACTIVE, LAST\_LOGIN\_DATE)*

*select RL\_NUMBER,USER\_ID, IS\_ACTIVE, LAST\_LOGIN\_DATE from USERS\_RL where rl\_number = source\_rl\_in;*

*insert into USER\_ROLES (RL\_NUMBER,USER\_ID, RVL\_CODE, UR\_DATE\_OF\_CREATION )*

*select RL\_NUMBER,USER\_ID, RVL\_CODE, UR\_DATE\_OF\_CREATION from USER\_ROLES where rl\_number = source\_rl\_in;*

*insert into COLLATERAL\_MARKETS(RL\_NUMBER,COLLATERAL\_MARKET, CSERVICE\_COLLATERAL, AUTO\_PRINT)*

*select RL\_NUMBER,COLLATERAL\_MARKET, CSERVICE\_COLLATERAL, AUTO\_PRINT from COLLATERAL\_MARKETS where rl\_number = source\_rl\_in;*

*insert into STORE\_CONFIGURABLES(RL\_NUMBER,CONFIG\_NAME, CONFIG\_VALUE)*

*select RL\_NUMBER,CONFIG\_NAME, CONFIG\_VALUE from STORE\_CONFIGURABLES where rl\_number = source\_rl\_in;*

*insert into TAX\_BREAKDOWN(RL\_NUMBER,TB\_STATE\_JUR, TB\_COUNTY\_JUR, TB\_CITY\_JUR, TB\_SEC\_CITY\_JUR, TB\_FED\_PCT, TB\_STATE\_PCT, TB\_SEC\_STATE\_PCT, TB\_COUNTY\_PCT, TB\_SEC\_COUNTY\_PCT, TB\_CITY\_PCT, TB\_SEC\_CITY\_PCT, TB\_FED\_TYPE, TB\_STATE\_TYPE, TB\_SEC\_STATE\_TYPE, TB\_COUNTY\_TYPE, TB\_SEC\_COUNTY\_TYPE, TB\_CITY\_TYPE, TB\_SEC\_CITY\_TYPE, TB\_PRODUCT\_CODE)*

*select RL\_NUMBER,TB\_STATE\_JUR, TB\_COUNTY\_JUR, TB\_CITY\_JUR, TB\_SEC\_CITY\_JUR, TB\_FED\_PCT, TB\_STATE\_PCT, TB\_SEC\_STATE\_PCT, TB\_COUNTY\_PCT, TB\_SEC\_COUNTY\_PCT, TB\_CITY\_PCT, TB\_SEC\_CITY\_PCT, TB\_FED\_TYPE, TB\_STATE\_TYPE, TB\_SEC\_STATE\_TYPE, TB\_COUNTY\_TYPE, TB\_SEC\_COUNTY\_TYPE, TB\_CITY\_TYPE, TB\_SEC\_CITY\_TYPE, TB\_PRODUCT\_CODE from*

*TAX\_BREAKDOWN where rl\_number = source\_rl\_in;*

*insert into OFFLINE\_TAX\_BREAKDOWN (STORE\_LOCATION\_ID, ORDER\_TAX\_AREA\_ID, TAX\_DATE, PRODUCT\_CLASS, JUR\_CODE, JUR\_LEVEL, JUR\_NAME, MEMO\_GL, PRINT\_NAME, SKU\_SPECIFIC\_IND, TAXABLE\_IND, TAX\_CODE, TAX\_LINE\_ID, TAX\_TYPE\_IND, TAX\_FIXED\_AMOUNT, TAX\_RATE)*

*select STORE\_LOCATION\_ID, ORDER\_TAX\_AREA\_ID, TAX\_DATE, PRODUCT\_CLASS, JUR\_CODE, JUR\_LEVEL, JUR\_NAME, MEMO\_GL, PRINT\_NAME, SKU\_SPECIFIC\_IND, TAXABLE\_IND, TAX\_CODE, TAX\_LINE\_ID, TAX\_TYPE\_IND, TAX\_FIXED\_AMOUNT, TAX\_RATE from OFFLINE\_TAX\_BREAKDOWN where STORE\_LOCATION\_ID = source\_rl\_in;*

*commit;*

***Database SQL Required for creation of new user IRU roles:***

--Replace the ID’s with correct ID’s

select max(rvl\_code) from role\_value\_list

insert into role\_value\_list values('2004','IRUAuditToolAuditor')

insert into role\_value\_list values('2005','IRUAuditToolManager')

insert into role\_value\_list values('2006','IRUAuditToolSuperuser')

commit

insert into users\_rl values ('am6489', 'IRUAuditStore', 'Y',sysdate);

insert into user\_roles values ( 'am6489', 2005, sysdate, ' IRUAuditStore');

insert into user\_login\_history values ('am6489', sysdate, ' IRUAuditStore');

insert into external\_access values('am6489','IRUAuditStore',1,'Not Set','22CB7',null,null);

insert into external\_access values('am6489','IRUAuditStore',2,'Not Set','002034',null,null);

insert into external\_access values('am6489','IRUAuditStore',3,'Not Set','CBTEST02',null,null);

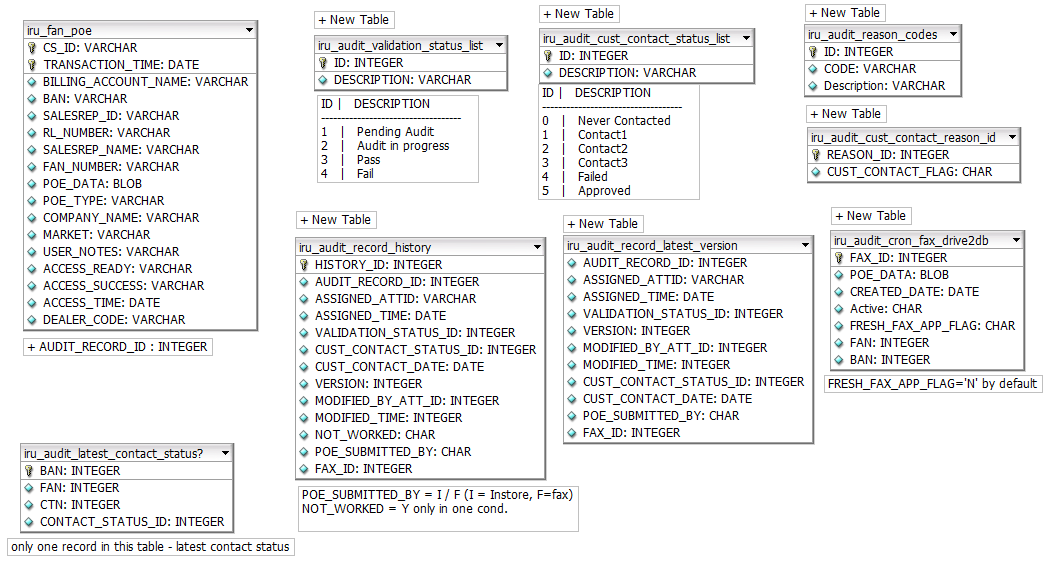
insert into password\_history values('am6489',null,null,null,null,null,null,sysdate);

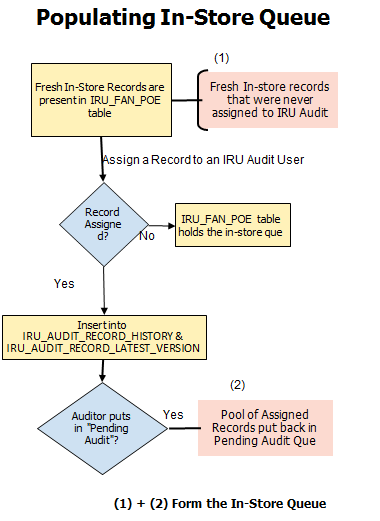
insert into user\_billing\_settings values('am6489','A0000', 'A0000', 'A0000','IRUAuditStore','DLS');

insert into cash\_register values ( '835', 'IRUAuditStore', 'OPUS Register', 'N', 1, 'TXCDTL01AM6489','001', 'Y', '12345', '2000', '8000', '81', '1', '1', '1', '', '', 'L', '','','');

**274829.OPUS\_OM.AD.02 :**

**Overall IRU Audit DB Design:**



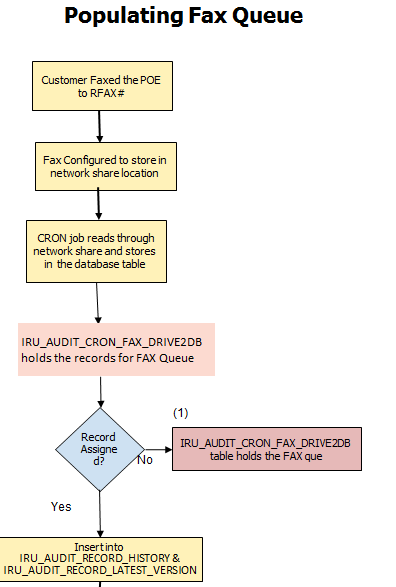


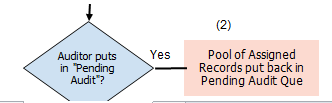
***In-Store Queue*** =

Records in IRU\_FAN\_POE table (never assigned) +

Records in IRU\_AUDIT\_RECORD\_LATEST\_VERSION (with validation\_status = Pending Audit and POE\_SUBMITTED\_BY = I)

***FAX Queue :***

****

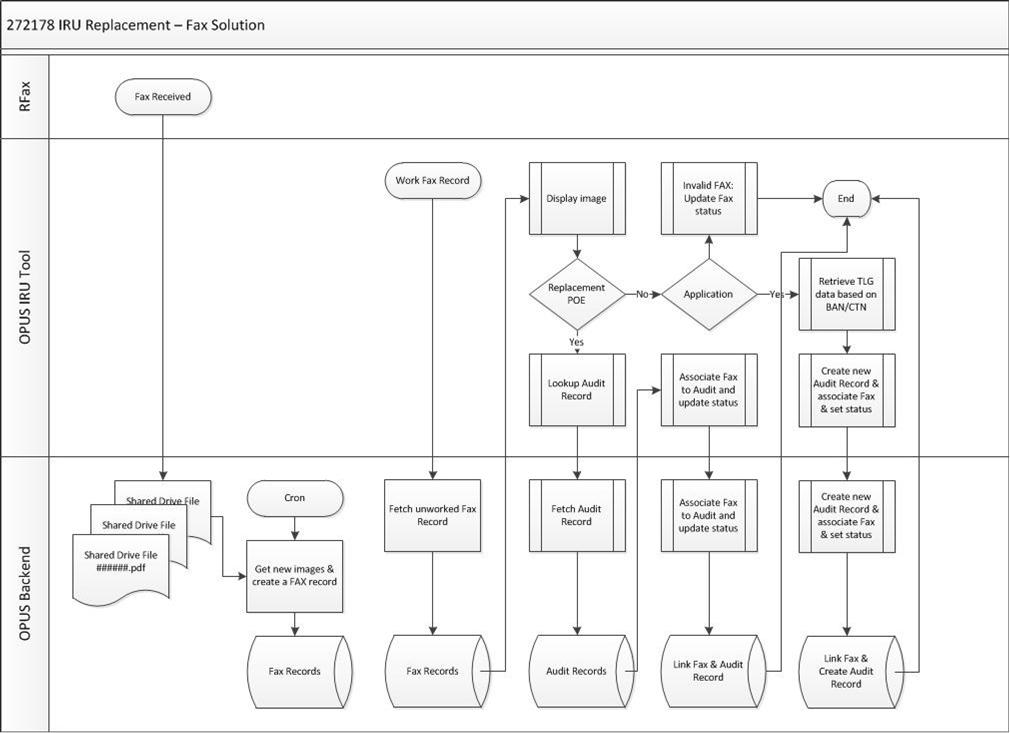
****

1. **+ (2) – Form the FAX Queue for assignment**

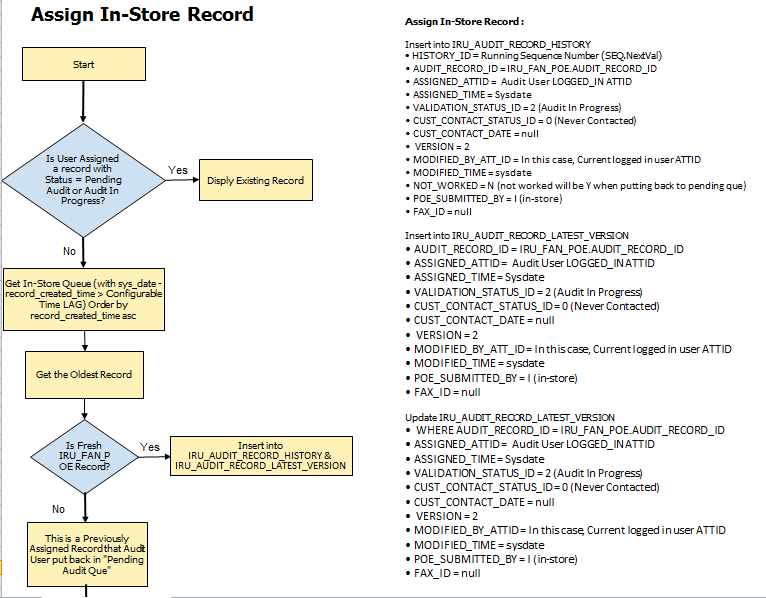
***FAX Queue*** =

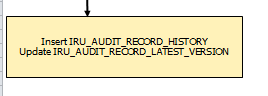
Records in IRU\_AUDIT\_CRON\_FAX\_DRIVE2DB table (never assigned once) +   
Records in IRU\_AUDIT\_RECORD\_LATEST\_VERSION (with validation\_status = Pending Audit and POE\_SUBMITTED\_BY = F)

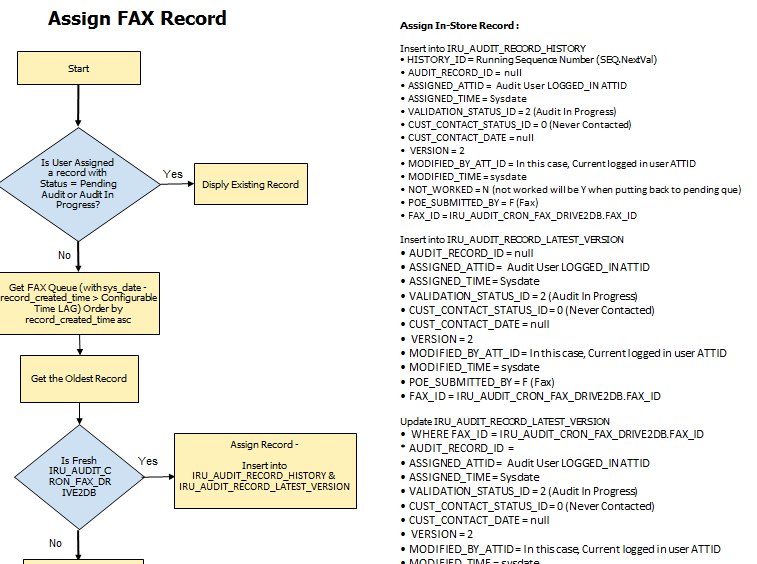
**FAX Complete solution:**

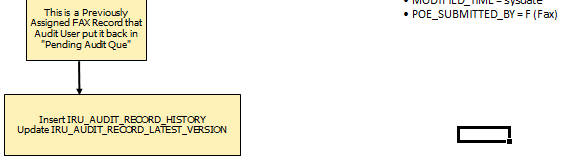


**274829.OPUS\_OM.AD.03 :**



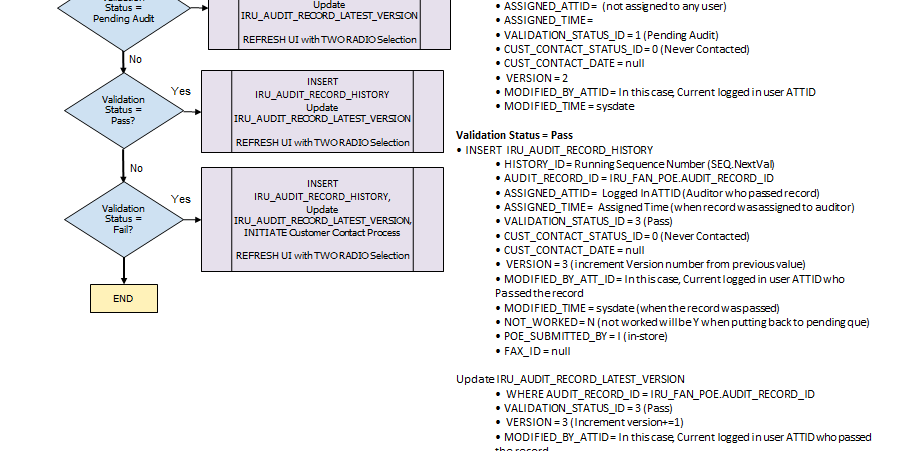


******

******

**274829.OPUS\_OM.AD.4 : Perform Audit In-Store**

#### 



**Validation Status = Fail**

• INSERT IRU\_AUDIT\_RECORD\_HISTORY

• HISTORY\_ID = Running Sequence Number (SEQ.NextVal)

• AUDIT\_RECORD\_ID = IRU\_FAN\_POE.AUDIT\_RECORD\_ID

• ASSIGNED\_ATTID = Logged In ATTID (Auditor who failed the record)

• ASSIGNED\_TIME = Assigned Time (when record was assigned to auditor)

• VALIDATION\_STATUS\_ID = 4 (FAIL)

• CUST\_CONTACT\_STATUS\_ID = 1 (Contact1)

• CUST\_CONTACT\_DATE = sysdate (when contact was done)

• VERSION = 3 (increment Version number from previous value)

• MODIFIED\_BY\_ATT\_ID = In this case, Current logged in user ATTID who Failed the record

• MODIFIED\_TIME = sysdate (when the record was failed)

• NOT\_WORKED = N (not worked will be Y when putting back to pending que)

• POE\_SUBMITTED\_BY = I (in-store)

• FAX\_ID = null

Update IRU\_AUDIT\_RECORD\_LATEST\_VERSION

• WHERE AUDIT\_RECORD\_ID = IRU\_FAN\_POE.AUDIT\_RECORD\_ID

• VALIDATION\_STATUS\_ID = 4 (Fail)

• CUST\_CONTACT\_STATUS\_ID = 1 (Contact1)

• CUST\_CONTACT\_DATE = sysdate (when contact was done)

• VERSION = 3 (Increment version+=1)

• MODIFIED\_BY\_ATTID = In this case, Current logged in user ATTID who failed the record

• MODIFIED\_TIME = sysdate (when the record was failed)

**POE Image display:**

• POE Thumbnail Image =

POE Image = IRU\_FAN\_POE.POE\_DATA

byte[] ba = getRSADecryptedPOEImageBytes();

request.getSession().setAttribute("IMAGE\_BYTE\_ARRAY", ba);

      <div style="margin:20px 0px 10px 20px;border:1px solid black; width:225px; height:225px;">

             <img src="<%=contextPath%>/servlet/SIGNATURE?action=SHOWTENDERSIGNATURE"  width="225" height="225" >

      </div>

**FAST FAN Lookup =**

function handleECPV(destination) {

timeoutRequestReset();

if(destination == "ecpv")

{

var url = '${pageContext.request.contextPath}/customerService/LaunchECPV.do?dispatch=launchecpv';

window.open(url,"","location=0,menubar=0,resizable=1,toolbar=0,scrollbars=1");

}

if(destination == "fastfan")

{

//1404 - Quick Links - FaSt new URL for LD

var isLocalDealer = ${mysession.pdcRetailLocationType};

var url = "";

if(isLocalDealer){

url = "${customerSummaryForm.fasturl}"+"${customerSummaryForm.fanInfo.fan}";

} else{

url = "${customerSummaryForm.fasturl}"+"/profile/"+"${customerSummaryForm.fanInfo.fan}";

}

window.open(url,"","location=0,menubar=0,resizable=1,toolbar=0,scrollbars=1");

//1404 - Quick Links - FaSt new URL for LD ends

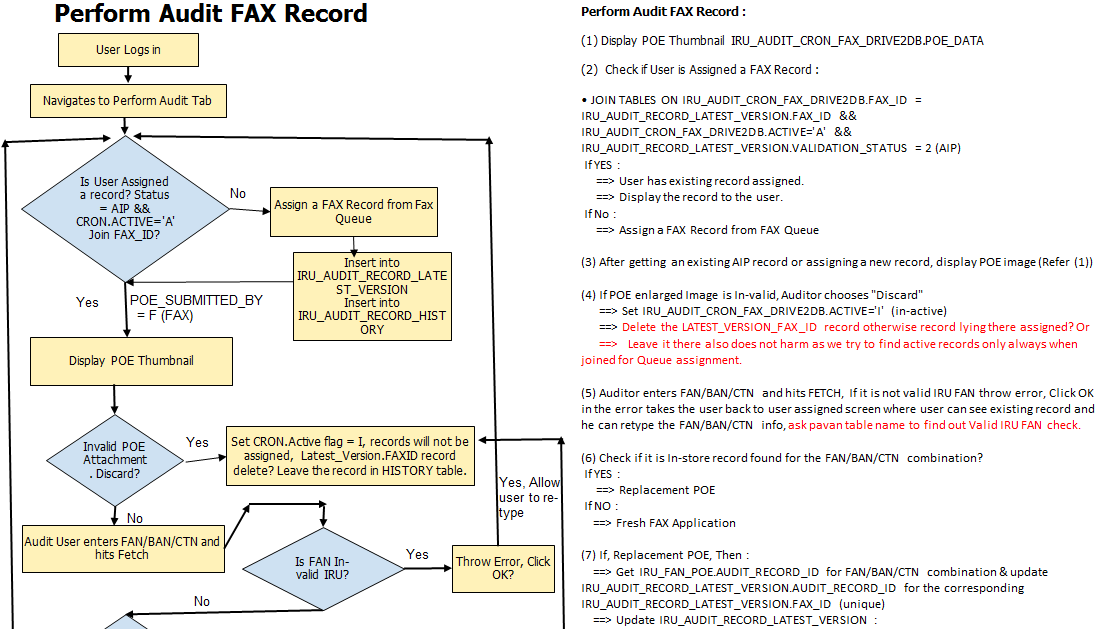
}

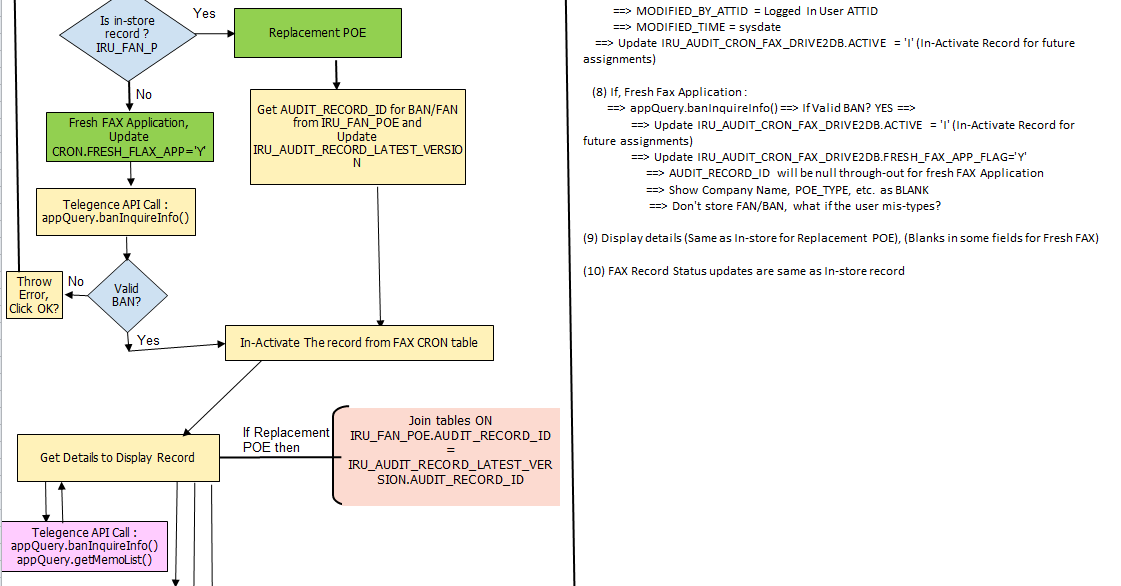
}

select \* from store\_configurable\_list where config\_name like '%FAST\_EXT\_LNK\_ENABLED%' and rl\_number = 'D112'

select \* from store\_configurables where config\_name like '%OM\_FAST\_EXT\_LNK\_ENABLED%' and rl\_number = 'D112'

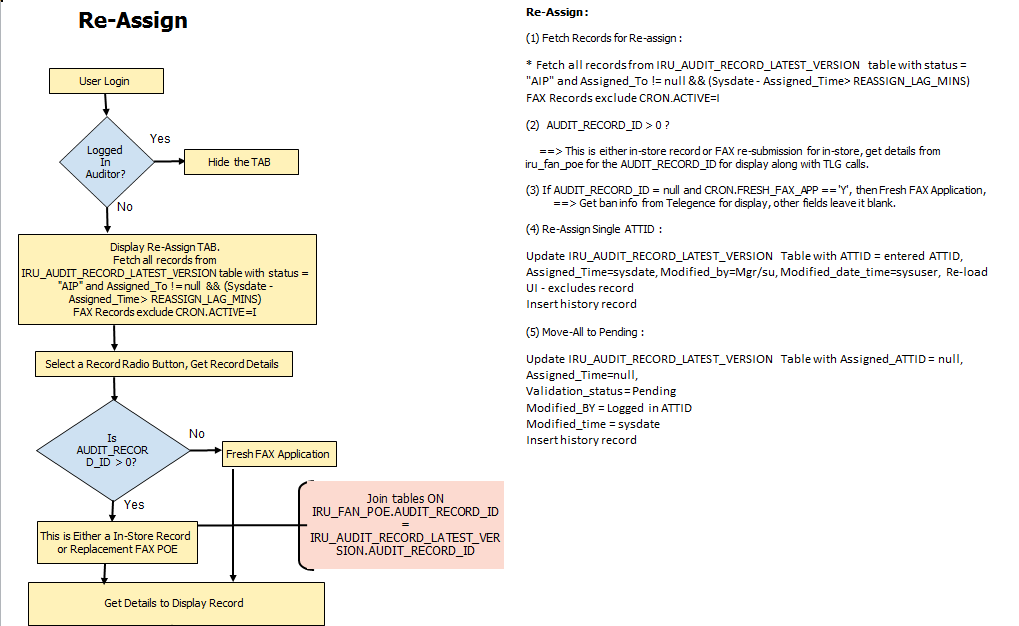
select \* from store\_configurables where config\_name like '%FAST\_EXT\_LNK\_ENABLED%' and rl\_number = 'D112'

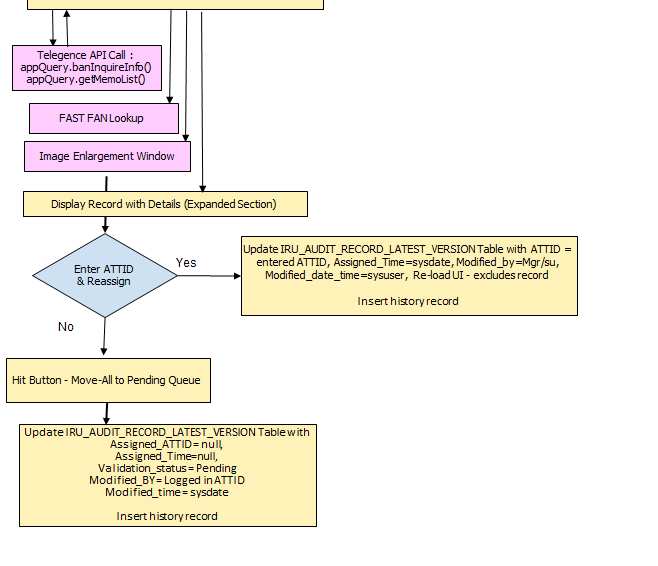




#### 

#### 274829.OPUS\_OM.AD.6

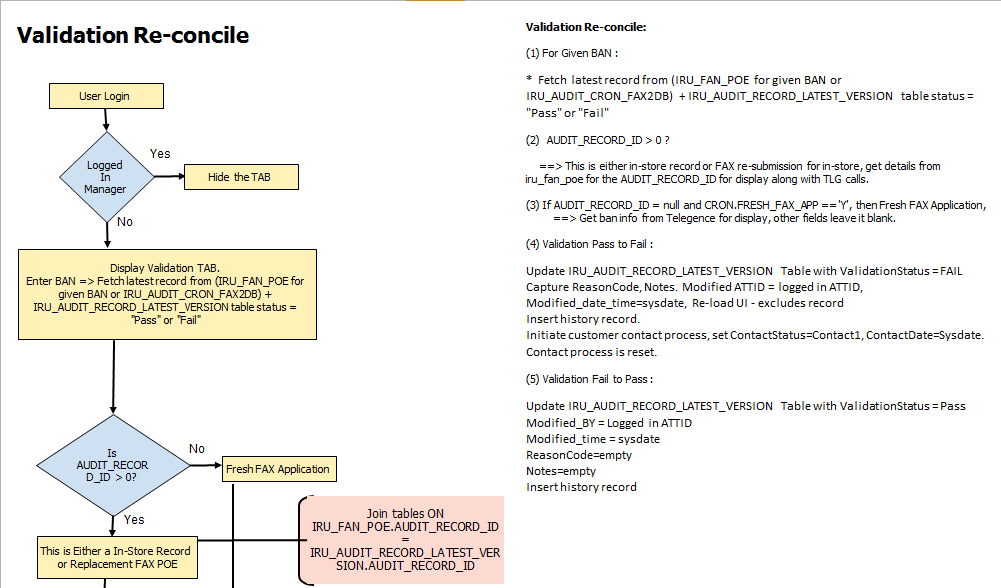
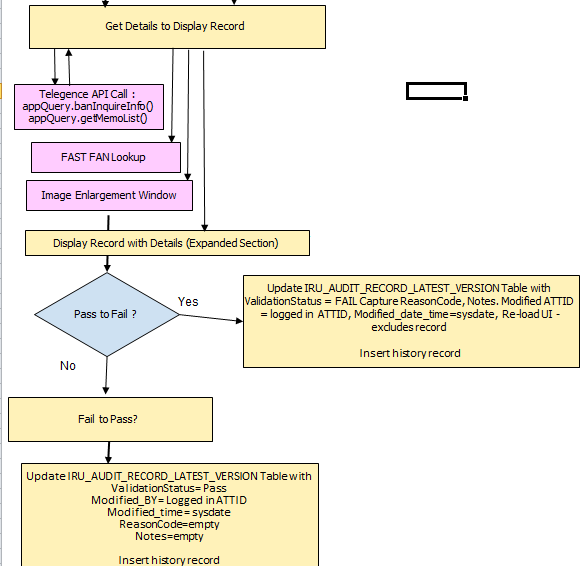




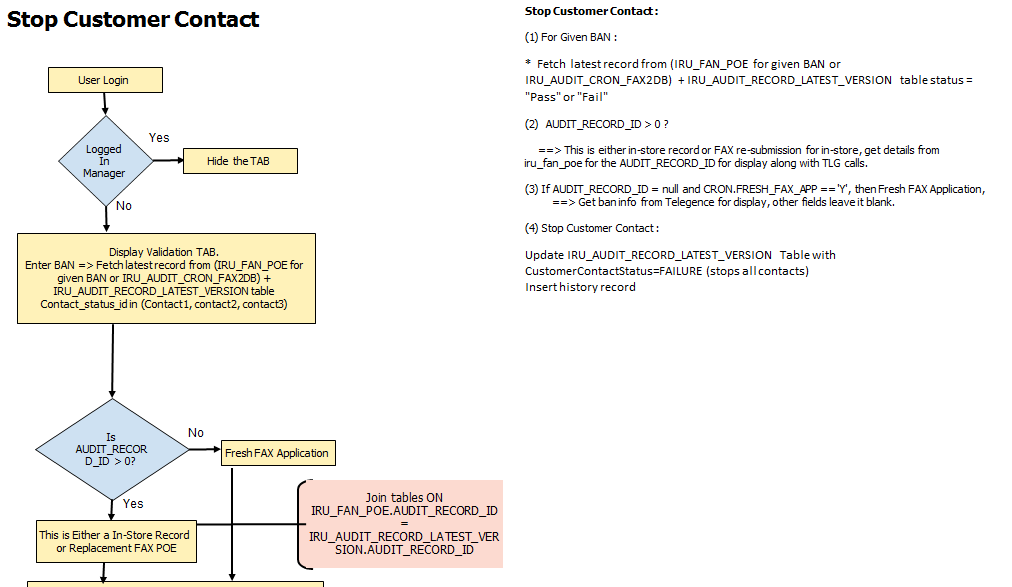
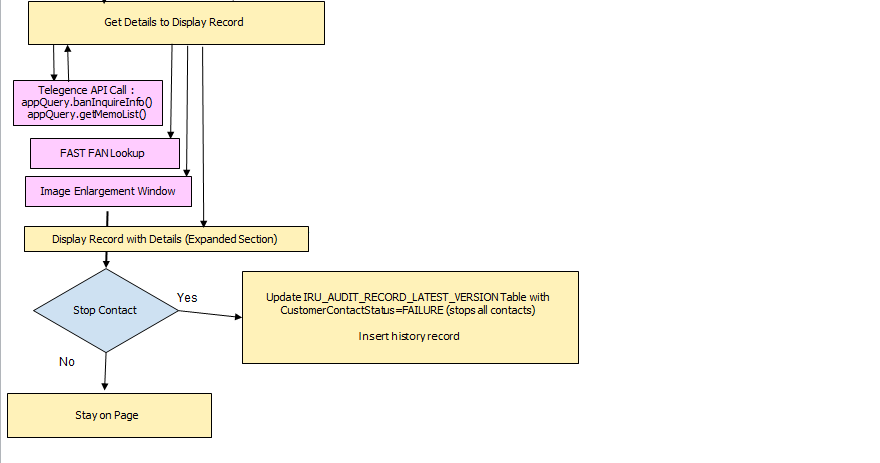
**274829.OPUS\_OM.AD.7**

* Display of Search Criteria fields only in-scope for Iteration1

**274829.OPUS\_OM.AD.8**

**274829.OPUS\_OM.AD.9**

#### Class Diagram

*This section captures the structural aspects of the module being designed which includes one of more class diagram(s) depending on complexity. The Business Tier interfaces will be depicted as a dependency in the class diagrams but their descriptions and implementation details will be elaborated in the Business Tier Design section.*

*Depict all classes and interfaces along with associations between them. Each class should have the correct data type identified for each attribute along with operation signatures. Sample Presentation Tier class diagram for a module being designed is depicted below. Use the component diagram from the HLD as the starting point, identifying classes and interfaces needed to realize each of the impacted(new or modified) sub-components running in the presentation tier.*

*Typical classes in the presentation tier design include:*

* *Action/UI Controller Classes*
* *Form Beans,*
* *JSPs*
* *Helper/Utility Classes (ex watermarking)*
* *Depict dependency on Business Tier interfaces (ex. Action class dependent on Remote EJB Interface)*

*Add a change summary table below if this design is for enhancements to a module for a project (as opposed to new module development).*

|  |  |  |
| --- | --- | --- |
| **Change Summary** | | |
| **Class/JSP Name** | **Change type** | **Change Description** |
| LoginAction.java | Modify | Modify to check Role Id and forward mapping to new layout based on IRU Roles. |
| IRUAuditToolLayout.jsp | New | New layout jsp |
|  |  |  |
|  |  |  |

#### Class Description

*Add a description for each class/interface impacted (new or modified) in the module. The description should clearly state the purpose of the class/interface (focusing on what the class/interface does), and each operation contained. For data wrapper classes (Javabeans with setters and getters), there should be a description for each attribute (as opposed to repetitive description of what getter and setter methods do). Note – This information can also be captured in the class diagram if the UML tool supports documentation, and then exported to a word document that can be embedded in the design doc. Sample class description is depicted below.*

*Data recognized as SPI/PCI is expected to be handled as below*

* *The request/response with SPI data should be transmitted on the secured network (SSL or digital signature).*
* *The SPI/PCI data elements should not be logged.*
* *Retrieval of SPI/PCI data from local DB storage should ensure voltage decryption before use.*
* *Storage of SPI/PCI data to local DB should ensure the voltage encryption before persistence.*

|  |  |
| --- | --- |
| **Class/Interface :** | |
| **Description** | *High level description/purpose of the class and encapsulated responsibility/behavior. If this is a helper/utility class, indicate if it is reusable and associated considerations for implementation (ex. packaging).* |
| **Change Summary** | *If this an existing class modified to support an enhancement project design, summarize the changes :*   * *Identify new attributes and operations.* * *Identify updated attributes and operations.* * *Identify deleted attributes and operations.* * *In rest of the table below show only added, updated or deleted attributes and operations (strike out the deleted ones).* |
| **Operation/Attributes** | **Description** |
| *Provide the name of each attribute and operation expected in the class, along with data type for attributes and signature for operations.* | *For each attribute, describe the purpose and data stored. For each operation, describe what it does and high-level implementation steps. Pseudo code could be used for specifying complex private logic that cannot be effectively depicted in sequence diagrams (where the focus is on interactions across classes), but use of java code is strongly discouraged.* |

#### UI Design

*This section captures the User Interface design, including look and feel, rendering, validations, and event processing logic for every user action on the screen. The sub-sections below should be repeated for each impacted screen in the module.*

##### Wireframe

[*http://alhoov1uvinda01:8600/OPUS/OPUS\_Launcher\_DEV/pdfs/OPUS/OPUS\_274829\_IRU\_Replacement\_Tool\_AuditInStoreRecord.pdf*](http://alhoov1uvinda01:8600/OPUS/OPUS_Launcher_DEV/pdfs/OPUS/OPUS_274829_IRU_Replacement_Tool_AuditInStoreRecord.pdf)

[*http://alhoov1uvinda01:8600/OPUS/OPUS\_Launcher\_DEV/pdfs/OPUS/OPUS\_274829\_IRU\_Replacement\_Tool\_AuditFaxRecord.pdf*](http://alhoov1uvinda01:8600/OPUS/OPUS_Launcher_DEV/pdfs/OPUS/OPUS_274829_IRU_Replacement_Tool_AuditFaxRecord.pdf)

[*http://alhoov1uvinda01:8600/OPUS/OPUS\_Launcher\_DEV/pdfs/OPUS/OPUS\_274829\_IRU\_Replacement\_Tool\_Reassign.pdf*](http://alhoov1uvinda01:8600/OPUS/OPUS_Launcher_DEV/pdfs/OPUS/OPUS_274829_IRU_Replacement_Tool_Reassign.pdf)

[*http://alhoov1uvinda01:8600/OPUS/OPUS\_Launcher\_DEV/pdfs/OPUS/OPUS\_274829\_IRU\_Replacement\_Tool\_BanTracker.pdf*](http://alhoov1uvinda01:8600/OPUS/OPUS_Launcher_DEV/pdfs/OPUS/OPUS_274829_IRU_Replacement_Tool_BanTracker.pdf)

[*http://alhoov1uvinda01:8600/OPUS/OPUS\_Launcher\_DEV/pdfs/OPUS/OPUS\_274829\_IRU\_Replacement\_Tool\_ValidationStatusReconcile.pdf*](http://alhoov1uvinda01:8600/OPUS/OPUS_Launcher_DEV/pdfs/OPUS/OPUS_274829_IRU_Replacement_Tool_ValidationStatusReconcile.pdf)

[*http://alhoov1uvinda01:8600/OPUS/OPUS\_Launcher\_DEV/pdfs/OPUS/OPUS\_274829\_IRU\_Replacement\_Tool\_StopCustomerContact.pdf*](http://alhoov1uvinda01:8600/OPUS/OPUS_Launcher_DEV/pdfs/OPUS/OPUS_274829_IRU_Replacement_Tool_StopCustomerContact.pdf)

[*http://alhoov1uvinda01:8600/OPUS/OPUS\_Launcher\_DEV/pdfs/OPUS/OPUS\_274829\_IRU\_Replacement\_Tool\_NavigationTabs.pdf*](http://alhoov1uvinda01:8600/OPUS/OPUS_Launcher_DEV/pdfs/OPUS/OPUS_274829_IRU_Replacement_Tool_NavigationTabs.pdf)

##### Screen Element Definition

*Extend the Screen Element Definition in HLD by specifying form bean data mapping for each screen element impacted (added or modified or deleted) .Use the table created in the HLD as the starting point, and update the “source/target binding” column with form bean/session attributes as applicable (replacing the database/backend service attributes in the HLD source/target). The mapping of form bean to business tier objects, and from business tier objects to backend services/database is captured in subsequent data mapping sections of the AD.*

##### Screen CAR Definition

*CAR (Command Action Response) definition for a screen captures all user actions on a screen (Ex. button click), processing entry point (ex. Struts Action class/method) for each action, and the resulting screen display from that action. Sample CAR definition is depicted below.*

##### Input Validations

*Create input data validation table capturing validation rules for all input data on the screen. Sample validation table is depicted below.*

|  |  |  |
| --- | --- | --- |
| **Screen :** *Provide a logical screen name along with JSP name.* | | |
| **Change Summary** | | *If this is for an existing screen with modified validations to support an enhancement project design, summarize the changes:*   * *Identify new validations.* * *Identify updated validations.* * *Identify deleted validations.* * *In rest of the table below show only added, updated or deleted validations (strike out the deleted ones).* |
| **Element Name** | **Field Validation Rules** | **Message displayed** |
|  |  |  |
|  |  |  |
|  |  |  |

##### Conditional Rendering

*Create a conditional rendering table capturing display rules for sections of the screen, including any enable/disable rules for UI controls (such as buttons). Sample conditional rendering table shown below.*

##### **UI Action Processing - Sequence Diagram**

*Create a sequence diagram for each UI action on the screen (typically button/hyperlink clicks, but includes any asynchronous events fired via AJAX, or UI control events such as onChange that require server or client side processing). The sequence diagram should include both client-side (ex. Javascript) and server side (ex. controller/action logic in Struts) processing. The invocation of physically separated Business Tier components (ex. EJBs) should be shown as a black box in these sequence diagrams. The sequence diagrams in the Business Tier design section will elaborate on the implementation of the middle tier services (ex. EJBs). Sample Sequence Diagram shown below. Use the high-level sequence diagram from the HLD as the starting point, and increase the level of detail needed from an implementation perspective (including logic for caching, security and other non-functional aspects).*

*Sequence diagrams are not effective in capturing logic that involves complex in-memory computations/algorithms and data transformations etc. Factor out that logic in private methods (or helper classes), depicting those methods as black boxes in the sequence diagram (using interaction fragment notation in sequence diagrams). Depict that kind of logic as pseudo code (or UML activity diagram) in the class/interface description covered earlier in section labeled Class Description.*

*If this design is for enhancements to module(s) (as opposed to new module development) that results in changes to logic depicted in the sequence diagram, use callouts (depicted below as an example) to highlight the change.*

*Important: Be sure to depict the watermarking utility method call if applicable.*

##### **UI Action Processing - Data Mapping**

*Create input and output data mapping table for each sequence diagram (corresponding to the UI event/user action).*

*Input data mapping table depicts the flow of data (left to right) for each method call in the sequence diagram. Sample Input Data Mapping table shown below.*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input Data Mapping for Sequence Diagram :** *Provide a logical name for the sequence diagram this mapping is for* | | | | |
| **Change Summary** | | *If this is for an existing sequence diagram that is being updated, summarize the data mapping changes in depicted method calls in the sequence diagram:*   * *Identify new data elements passed in method calls.* * *Identify updates to data elements passed in method calls (ex. data format change or an added condition around data transfer).* * *Identify deleted data elements in the mapping.* * *In rest of the table below show only added, updated or deleted data elements (strike out deleted ones).* | | |
| **Arrow#** | **Target method call** | **Condition** | **Source data element** | **Target data element** |
|  |  |  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

*Output data mapping table depicts population of response data returned by each method call in the sequence diagram, particularly for those methods that populate response data from multiple sources or the data requires some kind of transformation logic (e. format). Sample Output Data Mapping table shown below.*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Output Data Mapping for Sequence Diagram :** *Provide a logical name for the sequence diagram this mapping is for* | | | | |
| **Change Summary** | | *If this is for an existing sequence diagram that is being updated, summarize the data mapping changes in depicted method calls in the sequence diagram:*   * *Identify new data elements populated in the response.* * *Identify updates to data elements populated in the response (ex. data format change or an added condition around data transfer).* * *Identify deleted data elements in the response.* * *In rest of the table below show only added, updated or deleted data elements (strike out deleted ones).* | | |
| **Response Arrow#** | **Method call response** | **Condition** | **Source data element** | **Target data element** |
|  |  |  |  |  |

### Middleware Design – Business Tier

*This section captures the design of the physically separated middle tier/business services (ex. EJBs) backing the presentation tier for the module being designed.*

#### Class Diagram

*This section captures the structural aspects of the middle tier services supporting the module being designed, which includes one of more class diagram(s) depending on complexity. Depict all classes and interfaces along with associations between them. Each class should have the correct data type identified for each attribute along with operation signatures. Sample class diagram for Middle Tier services supporting the module is depicted below. Use the component diagram from the HLD as the starting point, identifying classes and interfaces needed to realize each of the impacted(new or modified) sub-components running in the business/backend tier.*

*Typical classes in business tier design include:*

* *Business Service Interfaces for services implemented as a part of the module being designed (EJB Remote/Local interface for Stateless Session Bean based architecture)*
* *Business Service Implementation classes for services implemented as a part of the module being designed (ex. EJB Home, Enterprise Bean class for Stateless Session Bean based architecture)*
* *Business Objects (input/output data for service interface methods)*
* *DAO interface and implementation classes*
* *Data Objects (input/output data for DAO interfaces, representing relation tables or views)*
* *Persistence framework specific classes (ex. for Entity Beans based architecture : EJB Home, Enterprise Bean classes)*
* *Business Service Interface classes for outbound service calls (calls to external Web Services or Stateless Session beans )*
* *Business Objects for outbound service calls (input/output data for above business service interface methods)*
* *Helper/Utility Classes*

*Add a change summary table below if this design is for enhancements to module(s) for a project (as opposed to new module development).*

|  |  |  |
| --- | --- | --- |
| **Change Summary** | | |
| **Class Name** | **Change type** | **Change Description** |
|  |  |  |
|  |  |  |
|  |  |  |

#### Class Description

*Add a description for each impacted (new or modified) class/interface in the module(s). The description should clearly state the purpose of the class/interface (focusing on what the class/interface does), and each operation contained. For data wrapper classes (Javabeans with setters and getters), there should be a description for each attribute (as opposed to repetitive description of what getter and setter methods do). Note – This information can also be captured in the class diagram if the UML tool supports documentation, and then exported to a word document that can be embedded in the design doc. Sample class diagram is depicted below.*

*Data recognized as SPI/PCI is expected to be handled as below*

* *The request/response with SPI data should be transmitted on the secured network (SSL or digital signature).*
* *The SPI/PCI data elements should not be logged.*
* *Retrieval of SPI/PCI data from local DB storage should ensure voltage decryption before use.*
* *Storage of SPI/PCI data to local DB should ensure the voltage encryption before persistence.*

|  |  |
| --- | --- |
| **Class/Interface :** *Name of the class or interface* | |
| **Description** | *High level description/purpose of the class and encapsulated responsibility/behavior. If this is a helper/utility class, indicate if it is reusable and associated considerations for implementation (ex. packaging).* |
| **Change Summary** | *If this an existing class modified to support an enhancement project design, summarize the changes :*   * *Identify new attributes and operations.* * *Identify updated attributes and operations.* * *Identify deleted attributes and operations.* * *In rest of the table below show only added, updated or deleted attributes and operations (strike out the deleted ones).* |
| **Operation/Attribute** | **Description** |
| *Provide the name of each attribute and operation expected in the class, along with data type for attributes and signature for operations.* | *For each attribute, describe the purpose and data stored. For each operation, describe what it does and high-level implementation steps. Pseudo code could be used for specifying complex private logic that cannot be effectively depicted in sequence diagrams (where the focus is on interactions across classes), but use of java code is strongly discouraged.* |

|  |  |
| --- | --- |
| **Class :** | |
| **Description** |  |
| **Change Summary** | *If this an existing class modified to support an enhancement project design, summarize the changes :*   * *Identify new attributes and operations.* * *Identify updated attributes and operations.* * *Identify deleted attributes and operations.* * *In rest of the table below show only added, updated or deleted attributes and operations (strike out the deleted ones).* |
| **Operation/Attributes** | **Description** |
|  |  |

#### Business Method - Sequence Diagram

*Create a sequence diagram for each business method in the middle tier service (ex. EJB). The sequence diagram should depict the implementation details of the business method, including any calls to data access component (DAO) for database interaction and/or external Web Services. If the DAO implementation calls a database stored procedure, the stored procedure itself should be depicted as the black box in the sequence diagram (with stored proc design specs elaborated later in the database tier design). Sample sequence diagram is shown below. Use the high-level sequence diagram from the HLD as the starting point, and increase the level of detail needed from an implementation perspective (including logic for caching, security and other non-functional aspects).*

*If this design is for enhancements to module(s) (as opposed to new module development) that results in changes to logic depicted in the sequence diagram, use callouts (depicted below as an example) to highlight the change.*

*Important: Be sure to depict the watermarking utility method call if applicable.*

#### Business Method - Data Mapping

*Create input and output data mapping tables for each business method, depicting flow of data to and from all outbound calls originating from the business method implementation. Each sequence diagram (corresponding to the business method of a middle tier service) should have a corresponding input/output data mapping.*

*Input data mapping table depicts the flow of data for each method call in the sequence diagram. Sample Input Data Mapping table shown below.*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input Data Mapping for Sequence Diagram :** | | | | |
| **Change Summary** | | *If this is for an existing sequence diagram that is being updated, summarize the data mapping changes in depicted method calls in the sequence diagram:*   * *Identify new data elements passed in method calls.* * *Identify updates to data elements passed in method calls (ex. data format change or an added condition around data transfer).* * *Identify deleted data elements in the mapping.* * *In rest of the table below show only added, updated or deleted data elements (strike out deleted ones).* | | |
| **Arrow#** | **Target method call** | **Condition** | **Source data element** | **Target data element** |
|  |  |  |  |  |

*Output data mapping table depicts population of response data returned by each method call in the sequence diagram, particularly for those methods that populate response data from multiple sources or the data requires some kind of transformation logic (e. format). Sample Output Data Mapping table shown below.*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Output Data Mapping for Sequence Diagram :** | | | | |
| **Change Summary** | | *If this is for an existing sequence diagram that is being updated, summarize the data mapping changes in depicted method calls in the sequence diagram:*   * *Identify new data elements populated in the response.* * *Identify updates to data elements populated in the response (ex. data format change or an added condition around data transfer).* * *Identify deleted data elements in the response.* * *In rest of the table below show only added, updated or deleted data elements (strike out deleted ones).* | | |
| **Response Arrow#** | **Method call response** | **Condition** | **Source data element** | **Target data element** |
|  |  |  |  |  |

#### Asynchronous/Batch Processing - Sequence Diagram

*If there is any asynchronous or batch processing involved in the design, create a sequence diagram for each asynchronous event or batch process (ex. processing a JMS message on a queue or a topic using Message Driven Beans, timer events such as the ones used to refresh cached data, batch process with business logic that goes beyond just raw ETL). The sequence diagram should depict the implementation details of asynchronous/batch processing, including any calls to stateless session EJB, external Web Services etc. Sample sequence diagram is shown below. Use the high-level sequence diagram from the HLD as the starting point, and increase the level of detail needed from an implementation perspective (including logic for caching, security and other non-functional aspects).*

#### Asynchronous/Batch Processing - Data Mapping

*If there is any asynchronous or batch processing involved in the design, create input data mapping table for each asynchronous event or batch processing, depicting flow of data to all outbound calls originating from the entry point method. Each sequence diagram (corresponding to an asynchronous event or batch process) should have a corresponding input data mapping.*

*Input data mapping table depicts the flow of data for each method call in the sequence diagram. Sample Input Data Mapping table shown below.*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input Data Mapping for Sequence Diagram :** *Name of the sequence diagram* | | | | |
| **Change Summary** | | *If this is for an existing sequence diagram that is being updated, summarize the data mapping changes in depicted method calls in the sequence diagram:*   * *Identify new data elements passed in method calls.* * *Identify updates to data elements passed in method calls (ex. data format change or an added condition around data transfer).* * *Identify deleted data elements in the mapping.* * *In rest of the table below show only added, updated or deleted data elements (strike out deleted ones).* | | |
| **Arrow#** | **Target method call** | **Condition** | **Source data element** | **Target data element** |
|  |  |  |  |  |

### Middleware Design – Externalized Configuration

*For middleware components (presentation and business tier), specify all application and technology configuration parameters that are externalized. Use the inventory of parameters from the HLD as the starting point, and add details such as storage location, default values, and environment sensitivity.*

*Capture all custom (application specific) configuration parameters in the table below. The first shaded row in the table below has column definitions and subsequent rows are representative examples.*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Application Configuration Parameters** | | | | |
| **Parameter Scope** | **Parameter name** | **Location** | **Description** | **Default Value** |
| *Enterprise Level Configuration*  *OR*  *Store Level configuration* | *Name of the parameter.* | *For parameters stored in a table, specify schema and table name.*  *For parameters stored in a properties file, specify the file name and whether it is in frontend or backend app server.* | *A brief description of the parameter (from the HLD).* | *Parameter value in both non-production and production environments (if different).* |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

*Capture technology framework specific configuration parameters in the table below. Focus on parameters that are significant from architecture/design standpoint. As an example, the transaction attribute for a Stateless Session EJB is architecturally significant because it has major impact on runtime behavior. On the other hand, struts navigation configuration (content of struts-config.xml) that can be derived by a developer from the Command Action Response (CAR) section of the design is not required in this section. The first shaded row in the table below has column definitions and subsequent rows are representative examples.*

|  |  |  |  |
| --- | --- | --- | --- |
| **Technology/Framework Configuration Parameters** | | | |
| **Technology framework** | **Location** | **Description** | **Configuration name/value pairs** |
| *Identify the framework this configuration is for (ex. EJB, JDBC, JMS etc.).* | *Identify the file or place where the parameter is configured.* | *A brief description of the parameter.* | *Provide name/value pairs of parameters in this group, both for non-production and production environments if different.* |
|  |  |  |  |
|  |  |  |  |

*If there are any new batch processes (scheduled outside the database engine) included in this design, capture the scheduling parameters.*

|  |  |  |  |
| --- | --- | --- | --- |
| **Batch Configuration Parameters** | | | |
| **Scheduler** | **Script Name** | **Description** | **Scheduling Frequency** |
| *Identify the scheduler that will kick off the batch process (ex. Cron, Control-M etc.)* | *Name of shell script that will launch the batch program.* | *A brief description of the batch process (whose logic specifications must be covered in Asynchronous/Batch Processing section).* | *The frequency of execution of the process (hourly, daily, weekly, monthly etc. with specific times).* |

### Database Tier Design

*This section captures the design for the database components supporting the module(s) being designed.*

#### ER Diagram

*Using the logical data model from the HLD as the starting point, create an ER (Entity Relationship) diagram depicting physical database design for the functional unit/module(s) impacted. For enhancement projects, add a change summary table below expressing deltas. If there are no database changes, this section is not required.*

|  |  |  |
| --- | --- | --- |
| **Change Summary** | | |
| **Table Name** | **Change type** | **Change Description** |
|  | *New/Modified/Deleted* |  |
|  |  |  |
|  |  |  |

*Sample diagram shown below.*

#### Table Definitions/Data Dictionary

*Describe each table in the solution (for an enhancement project, only new and modified tables). If there are no database changes, this section is not required.*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Table :** *Name of the table* | | | | | |
| Description | *Provide a brief description of the table (purpose, type of data stored etc.).* | | | | |
| Expected data volume/growth | *Maximum size of each row, number of rows generated per day/month etc.* | | | | |
| Archival approach | *How long the data will be retained for before archival and the associated approach (name of the stored procedure used of archiving etc.)?* | | | | |
| RIM (Records Information Management) retention classification category | *Provide RIM classification for the data being generated. This may impact the retention period. As an example, if the data stored is used for marketing (RIM classification), it will need to be retained for a different period than other types of data.* | | | | |
| Change Summary | *If this is for an existing table that is being updated, summarize the new column, data type and constraint.*  *If the column holds SPI/PCI functionality then those needs to explicitly Y and the data populated in the column should be voltage encrypted.* | | | | |
| **COLUMN** | **DataType** | **Description** | **NULLABLE** | **Constraint** | **SPI/PCI** |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

#### Type and Sequence Definitions

*If there are any custom PL/SQL data types (used by the stored procedures/functions) and database sequences (often used for generating surrogate primary keys), describe them in this section.*

|  |  |  |  |
| --- | --- | --- | --- |
| Type: *Name of the custom type* | | | |
| ***Description*** | | *Describe the data entity represented by this custom type along with purpose.* | |
| ***ChangeSummary*** | | *If this is for an existing type being updated, summarize the new/updated columns.* | |
|
|
| **COLUMNS** | | **DataType** | **Description** |
| Sequence: *Name of the sequence* | | | |
| ***Description*** | *Describe the purpose of the sequence and provide a reference to the table/field this sequence will be used for.* | | |
| ***Change Summary*** | *If this is for an existing sequence being modified, summarize the update.* | | |
| ***Specification*** | *Provide the parameter values (START WITH, INCREMENT BY, MINVALUE, MAXVALUE etc.) for the sequence.* | | |
|

#### Database Scripts

*If there are any database impacts (DDL, and DML for seed and configuration data if any), attach all the scripts or provide a reference to the script in version control (SVN).*

#### Database Replication Design

*If there are any replication impacts, attach the replication requirement specification using the template below for reference.*

#### Database Stored Procedure/Trigger Design

*If there are any stored procedure/trigger impacts (new or modified), repeat the following for each database stored procedure/trigger added or updated for the module(s). These stored procedures are typically used by the DAOs or async/batch processes depicted in the sequence diagrams for business tier. Include any custom stored procedures used for archiving or similar batch processing.*

##### Interface and Implementation Specifications

*Create stored procedure interface specification, describing all inputs and outputs for the stored procedure. Designer should design and develop stored procedures/triggers and get them reviewed with ADBA.*

|  |  |  |  |
| --- | --- | --- | --- |
| **Package** : *Name of the package* | | | |
| ***Description*** | | *Description of the package.* | |
| ***Change Summary*** | | *If this is for an existing package being updated  • Summarize the methods introduced in the package if any. • Summarize the changes to the existing methods in the package.* | |
|
|
| **Procedure/Function** | |  | |
| **Procedure/Function/Type Description** | |  | |
| **Interface specification** | | | |
| **Parameters** | **TYPE** | **Data type** | **Description** |
|  |  |  |  |

#### Database Batch Process Design

*If there are any batch processes scheduled using the database engine’s scheduler (ex. DBMS\_SCHEDULER in Oracle), provide the associated scheduling parameters. The design of the actual program (typically a stored procedure) being scheduled should be covered in the previous section.*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Database Batch Configuration Parameters** | | | | |
| **Process Name** | **Scheduler** | **Script/Procedure Name** | **Description** | **Scheduling Frequency** |
| *Name of the batch process (ex. click2store order archival)* | *Identify the scheduler that will kick off the batch process (ex. Oracle DBMS\_SCHEDULER)* | *Name of stored procedure or shell script (that will launch the stored procedure).* | *A brief description of the batch process (purpose and responsibilities).* | *The frequency of execution of the process (hourly, daily, weekly, monthly etc. with specific times).* |

## Other Information and References

*This section includes conditional information (for example PCI-related considerations), as well as topics deemed important by those developing this package.*

*If no additional information is deemed necessary, enter N/A. Provide any additional information believed valuable in meeting the purpose of this document. For additional guidance and support refer to* your [Local Support Team (LST)](http://itup.it.att.com/ittools/itmap/resources/cfm/itup/1_ProcessElement.cfm?xPEname=LST%20List)

# Acceptance & Approvals

## Overview

Use this section to capture approvals in the event that electronic approvals via the PRISM Project Workflow Module will not be used.

The Approvers of this work product agree that this document is acceptable and complete to the best of their knowledge and will be used by the project team as an official deliverable for the project. It is further agreed that this document can now be baselined and any changes to these sections from this point forward must follow the Managing Change in the IT UP. Embed evidence of approval in the review table below.

Embed evidence of approval in the review table below, or use the PRISM Approval Functionality in the Project Workflow Module Workflow Template View.

## Approvers

|  |  |  |  |
| --- | --- | --- | --- |
| **ATTUID and Name** | **Role** | **Group/Application** | **Version Approved, Approval Date and Approval Evidence** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |