Oracle FLEXCUBE Universal Banking ® 12.87.05.0.0 Development Workbench - Notifications

June 2018



Contents

1	Pref	ace3
1	1.1	Audience3
2	Intro	oduction3
ł	How to	o use this Guide
3	Noti	fication – Getting started4
	3.1	What is Notification
	3.2	Notification Trigger4
4	Noti	fication Development4
2	4.1	Pre-request for Notification development and testing4
2	4.2	Notification specification5
2	4.3	Notification XML development5
2	4.4	Notification Process
2	4.5	Development process in Development Workbench6
2	4.6	Notification Trigger
2	4.7	Notifications
5	Dep	loy Notification19
5	5.1	Notification - Workbench related deployment19
5	5.2	Notification Trigger deployment
6	Test	Notification
6	5.1	Notification flow
6	5.2	Scheduler based notification
6	5.3	MDB based notification flow21
6	5.4	Triggering notification and testing23

1 Preface

This document describes the steps to develop the notification XML and notification trigger using Oracle FLEXCUBE Development Workbench for Universal Banking.

1.1 Audience

The Development Workbench Notification Development book is intended for the FLEXCUBE Application Developers who perform the following tasks:

Develop new Notification

To Use this manual, you need conceptual and working knowledge of the below:

Proficiency	Resources
FLEXCUBE UBS Development	FCUBS-FD01-01-01-Development
overview	Overview Guide
Interface Getting started	FCUBS-FD04-01-01-Interface Getting started
FLEXCUBE Development Workbench for Universal Banking Reference	User manuals
Web service development to have query web service in place	FCUBS-FD02-03-01-RAD Web Service Development

2 Introduction

How to use this Guide

The information in this guide includes:

- <u>Chapter 3, "Introduction"</u>
- <u>Chapter 4, "Notification Getting started"</u>
- <u>Chapter 5, "Notification Development"</u>
- <u>Chapter 6, "Deploy Notification"</u>
- <u>Chapter 7, "Test Notification"</u>

3 Notification – Getting started

3.1 What is Notification

Notification framework in FLEXCUBE UBS is used to communicate the business event happened in FLEXCUBE UBS to external systems. Depending upon the event, the XML message is pushed to external system's asynchronous Queues for their consumption.

3.2 Notification Trigger

Notification Triggers is developed to recognize the event and then invoke the notification process. This trigger is developed using Development Workbench.

4 Notification Development

4.1 Pre-request for Notification development and testing

Following are pre-request for notification development:

- Target FLEXCUBE Environment with Notification framework installed
- Development Workbench link mapped to the FLEXCUBE environment
- Required Query Web services developed and tested

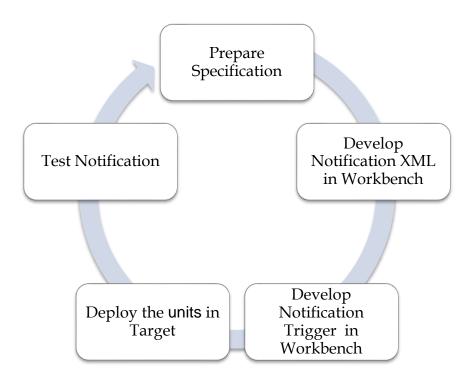


Fig 5.1.1: Development of Notifications

4.2 Notification specification

Identify the notification requirement as below

- What is the Notification function ID name for RAD XML (Third character should be N)?
- What is the Notification code?
- What is the Base table in FLEXCUBE UBS that triggers the notification?
 - What operation at base table triggers (insert/update/delete)?
 - What is the where clause for filter?
- What is the query Web service to be used?
 - What is the operation?
 - What are the tags required?

Example;

- Notification function ID name STNCUMOD
- Notification code NOTIF_CA_CUSTACC_MOD
- Base table STTM_CUST_ACCOUNT
 - Operation DELETE
 - \circ Filter Account class type in (S, U)
- Web service to be used *FCUBSAccService*
 - Operation *QueryCustAcc*
 - Request node Cust-Account-IO

4.3 Notification XML development

Notification RAD XML development creates the following files:

- RAD XML
- SPC
- SQL
- Static Data

4.4 Notification Process

There will be one trigger for the base table of notification and in case of multiple notifications sharing the same base table, there will be no new triggers created. Instead the same trigger created on the base table will be reused. This approach reduces the number of triggers being used for notifications.

4.5 Development process in Development Workbench

The notification development process in Workbench is split into two steps:

- 1. Notification Triggers
- 2. Notification Filter Procedure

The first step is to create notification triggers for base tables. The trigger generated from Workbench will be inserting key details into a static notification log table. The following details will be captured:

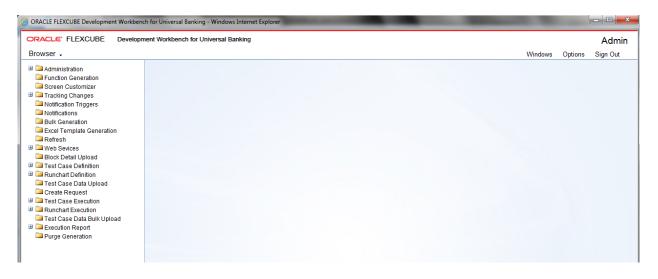
Trigger code: A unique value to for a notification trigger. **Base Table:** The base table on which, the trigger is built. **When Clause:** A simple when clause for the notification trigger.

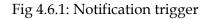
The second step is to capture details of notifications and generate the notification filter procedure. The following details are captured:

Notification code: A unique value to identify a notification. **Description:** Meaningful description of the notification. **Gateway Service:**

4.6 Notification Trigger

After successful login to Development Workbench click on Notification Trigger option in the tree as shown below:





Notification Trigger					_ ×
		X	1 17	Ъ ∣₹	Q +
Trigger Code • Description Firing Time Each Record Selected Columns Trigger When Clause	PK Cols * Before * Yes * Data Types Notification Codes				
Trigger Logic(Set SNC	ΣΠΕΥ ΤΟ ΥΝΙ)				

Fig 4.6.2: Notification trigger options

Notification Trigger we have two options - Add a new Trigger or Modify Existing one.

New:

Notification Trigger						_ ×
		×	I [I	1	V	9 0
Description Firing Time Each Record Selected Columns	TRG_CUSTADDR Base Table • MSTM_CUST_ADDRESS Trigger for Customer Address Maintenand PK Cols • CUSTOMER_NO-LOCATION-MEDIA After • PK Types • VARCHAR2-VARCHAR2 Yes • Data Types Interval Notification Codes					
Trigger When Clause	(new auth_stat=A)					
I_Oper ELSE	Id once_auth, N) ↔ Y THEN ation := TVPDATE; ation := UPDATE;					

Fig 4.6.3: Notification trigger: New option

Trigger Code: A unique value to for a notification trigger. Follow the naming conversion it should start with **TRG_XXXX**. This is mandatory field. This attribute signifies the trigger code created as part of trigger creation step in OTD. Each notification will be linked to a trigger code.

Description : Information field. Meaningful description of Trigger is to be given.

Firing Time : Specify when trigger needs to fired. We can create only BEFORE and AFTER triggers for tables. (INSTEAD OF triggers are only available for views; typically they are used to implement view updates.) (After/Before).

Each Record: specify for each row required or not. If FOR EACH ROW option is specified, the trigger is row-level; otherwise, the trigger is statement-level. **(Yes/No)**

Base Table: The base table on which, the trigger is built. This is mandatory field. Select a valid table from available LOV next to the field.

Pk Cols: Enter Primary key fields of table in tilde (~) separated format. This is mandatory field.

Pk Types: Enter Primary key type of the corresponding primary key field. This is mandatory field.

Selected Columns and Data Types: Defunct

Trigger When Clause: A simple when clause for the notification trigger. A trigger restriction can be specified in the WHEN clause, enclosed by parentheses. The trigger restriction is a SQL condition that must be satisfied in order for Oracle to fire the trigger. This condition cannot contain sub queries. Without the WHEN clause, the trigger is fired for each row.

Notification Codes: If the trigger is associated with a specific notification code, then the particular notification code has to be provided in the field. If the trigger is shared across many Notifications, field can be left empty

erote Rad Filos							
Front-On	d Files System Packages	Hook Paokages	м	leta Data		Others	^
] RadXML] Screen Xml] System JS	Main Package Spec Main Package Body Notification Triggers Upload Package Spec Upload Package Body	Kernel Package Spac Kernel Package Body Clusler Package Spac Clusler Package Body Custom Package Spac Custom Package Body	Menu Delalis Datasource Delalis D/V Detaits Eliock Delalis Screen Detaits Amendable Detaits Call form Delalis Summary Details	Gales	: PK Columns Ion Call Forms way Delaits cation Delaits Ion Parameters	Xsds Xsd With Annotations Screen Himi Upload Table Trager Upload Tables Definitio Archive Table Definitio	
SLNo		File Name		File	Туре	Status	~
	GWTR#_TRG_CUSTADDR Irg		TRO			Generaled -	
	GWTM_NOTIFICATION_TRIGGERSTF	RG_OUSTADOR INC	INC			Generalad *	
	TRG_CUSTADDR_RAD ml		RAD	XML		Generaled -	
nformation	Error Description	Error Code					
. Re	equest successfully Processed	RD-SAVE-007					
						Genera	te Exit

Fig 4.6.4: Notification trigger: Generation

On successful save Notification Trigger will generate two files (gwtr#_<trg-code>.trg and GWTM_NOTIFICATION_TRIGGERS__<trg-code>.INC) user needs to compile them in FLEXCUBE schema.

3 6 110	
Modify	•
IVIUUII V	٠

Notification Trigger							-	×
		×	x	17	6	Ŧ	6	
Trigger Code *	Base Table *							
Description	PK Cols *							
Firing Time	Before 👻 PK Types *							
Each Record	Yes 💌 Data Types							
Selected Columns	Notification Codes							
Trigger When Clause								
Tripper Lands (Oct 2010								
Trigger Logic(Set \$NC								
	Ψ							

Fig 4.6.5: Notification trigger: Modification

Notification Trigger						×	T. 15	31	77 Q	* ×
			Trigger Code		×					
Trigger Code *		×5 77	Trigger Code							
Description Firling Time Each Record	Belore *									
Selected Columns Trigger When Clause				Search Reset						
				<< < 1of5 > >>						
Trigger Logic (Sel \$NC	DTIFY TO Y/N)		Trigger Code	^						
			CUST_AC_BRN_TFR LOAN_BRN_TRFR							
			TD_AC_BRN_TFR TRG_APP_DETAIL							
			TRG_BLKDELMSTR							
			TRG_BRTMMSTR							
			TRG_CATDET TRG_CFRAMSTR							
			TRG_CHBK							
			TRG_CLAC							
			TRG_CLTMPRD							
			TRG_CONT TRG_CSTBCTRT							
			TRG_CSTMPRD							
			TRG_CUST							
				-						

Fig 4.6.6: Notification trigger: Modification-Selecting Trigger name

Notification Trigger					-
			×	I 17	1 F 🌒
				E	xecute Query
				_	
Trigger Code * TRG_CUSTADDR	Base Table '	:			
Description	PK Cols *				
Firing Time Before 👻	PK Types '				
Each Record Yes *	Data Types				
Selected Columns	Notification Codes				
Trigger When Clause					
Trigger Logic(Set \$NOTIFY To Y/N)					
		<u>^</u>			
		-			

Fig 4.6.7: Notification trigger: Modification- Entering values

Notification Trigger		_ ×
		🗎 🗶 🖬 📴 🎦 🐬 🍥 🤿
Trigger Code * TRG_CUSTADDR	Base Table * MSTM_CUST_ADDRESS	
Description Trigger for Customer Address Maintenand 🖓	PK Cols * CUSTOMER_NO~LOCATION~MEDIA	
Firing Time	PK Types * VARCHAR2~VARCHAR2~VARCHAR2	
Each Record Yes *	Data Types	
Selected Columns	Notification Codes	
Trigger When Clause (new.auth_stat='A')		
Trigger Logic(Set \$NOTIFY To Y/N) IF NVL(:old.once_auth, 'N') <> 'Y' THEN		
I_Operation := "INSERT";		
ELSE I_Operation := "UPDATE";		
END IF;		
	Ŧ	

Fig 4.6.8: Notification trigger: Modification- Entering values

erote Rad Files						
Front-En	f Filen System Packagen	Hook Paokages	N	leta Dala	Others	
] RadXML] Screen Xml] System JS	Main Package Spec Main Package Body Nobliccation Tinggers Upload Package Spec	Kernel Package Spec Kernel Package Body Oluster Package Spec Oluster Package Body Oustom Package Spec Custom Package Body	Menu Details Datasource Details LOV Details Block Details Screen Details Amendable Details Call form Details Summary Details	Label Details Block PK Columns Function Call Forms Catleway Details Notification Details Function Parameters Purge Details	XSds Xad With Annatalions Screen Himti Upload Table Trigger Upload Tables Ordinition Archive Table Definition	
SLNo		File Name		File Type	Status	^
	GWTR#_TRG_CUSTADDR Irg		TRO	G	Generaled +	
	GWTM_NOTIFICATION_TRIGGERSTRG	CUSTADOR INC	INC	;	Generaled -	
	TRG_CUSTADDR_RAD xml		RAD	DXML	Generaled -	
	Error Description equest successfully Processed	Error Code ARD-SAVE-007				

Fig 4.6.9: Notification trigger: Modification-Successful Generation

4.7 Notifications

Notifications Screen will be used to create new notification or modify existing notification; here we capture notification information for notification codes. We save notification details into xml.

Notification Maintenance							- 7 g	×
No	tification Function		Action None -	Save XM	L Path			
Notification Code * Description Notification Xsd Firing Time Filter Type Gateway Service Gateway Operation Gateway IO Request Type XSD Name Filter Logic(Set \$NOTIFY To Y	▼) ▼ N & Refer Current Record as	\$CURRENT_RECORD)	Module Des Bas F	Module				
Web Service Tags					+ -			
Order	Xsd Field	Table Field	Data Type	Maximum Length	~			

Fig 4.7.1: Notification Screen

Action: We can choose either new or Load action. New to create a new notification and Load is used to modify the existing one.

Save Xml Path: Specify the path to save notification xml. This would be considered only if the Save Mode is Client and Work Directory is specified as \$CURRENT_DIRECTORY

Notification Function: Specify the notification function-id name.

Conventions:

Maximum 8 chars. 3^{*rd*} *letter must be 'N'. Example: FTNCONON*

Notification Code: Enter the notification code to which we need to capture values. This is Mandatory field.

Recommended Convention for Notification Codes: NOTIF_<Module Code>_<Description> Example: NOTIF_LD_CONTRACT This is the notification indicating that a LD contract has been created/modified **Description:** Information field. Meaningful description of the Notification has to be provided in the field

Module: This attribute signifies the module on which the notification is based.

Module Description: Information field. Module Description which would be defaulted from Module LOV

Notification XSD: Notification XSD name will have to be provided in the corresponding Field. Naming convention to be followed while naming Notification XSD is as follows

[Module Name] – [Notification Description] – Notif.xsd Example: FT-Contract-Notif.xsd

Notification XSD has to be provided only if no Gateway Web Service Query Operation is configured to the Notification

Base Table: Select the base table on which trigger needs to be applied.

Firing Time: Indicates the Operation on the base Table for which Notifications has to be sent. Options available are Insert, Update or Both

Filter Type: This attribute can take the following values.

- 1. Where clause
- 2. Plsql block

Pk Cols: Enter Primary key columns of the Base Table.

Pk Types: Enter Primary key field Data Types.

Provide details of Gateway Service, Operation, Type XSD Name and Full Screen Reply if a Query Web Service has to be mapped to the Notification

Gateway Operation: The gateway operation name to execute query for the mentioned Service.

Gateway Service: The gateway service to be used to get the full screen response.

Gateway IO Request: The gateway IO request node to be used in querying operation.

Type XSD Name: This field has to be entered if Notification is mapped to a Service and Request. Name of the Master Type XSD for the service and operation has to be provided here. This can be found in include portion of the Request Msg XSD of particular Service-Operation

Example: LC-Contract-Types.xsd

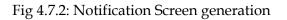
Full screen Reply: This attribute decides whether full screen or primary key notification response to be sent. This is applicable only if gateway Service details are provided

HO only: This attribute is used to send notification only from head office.

Filter Logic: The filter logic which decides whether the notification needs to be sent or not. This can be a simple where-clause on base table or a complex pl/sql block.

Web service Tags: The columns selected from base table as part of web service tags, will be used to send the full screen notification response. These tags defines the elements of Notification Xml when no Query service is mapped to it .

Front-End Files	System Packages	Hook Packages		Meta Data	Others	
RadXML	Main Package Spec	Kernel Package Spec	Menu Details	Label Details	Xsds	
Screen Xml	Main Package Body	Kernel Package Body	Datasource Details	Block PK Columns	Xsd With Annotations	
System JS	Notification Triggers	Cluster Package Spec	LOV Details	Function Call Forms	Screen Html	
	Upload Package Spec	Cluster Package Body	Block Details	Gateway Details	Upload Table Trigger	
	Upload Package Body	Custom Package Spec	Screen Details	Notification Details	Upload Tables Definition	
		Custom Package Body	Amendable Details	Function Parameters	Archive Table Definition	
			Call form Details	Purge Details		
			Summary Details			
SI.No		File Name	File Type		Status	
SI.NO		lie Name	File Type			
					Generate	Exit
						Exit



Front-End I	Files System Pack	ages Hook Packages	М	leta Dala	Others
2 RadXML 3 Screen Xml 3 System JS	Main Package Spe Main Package Bod Notification Trogger Upload Package S Upload Package B	y 📝 Kernel Package Body s 👘 Cluster Package Spec pec 👘 Cluster Package Body	Menu Delalis Datasource Delalis LOV Details Block Delalis Screen Details Amendable Details Call form Delalis Summary Details	Label Defails Block PK Columns Function Call Forms Galeway Defails Molification Defails Function Parameters Purge Defails	Xsds Xsd With Annolations Goreen Himi Upload Table Trigger Upload Tables Definition Archive Table Definition
SLNo		File Name		File Type	Status
	fipks_finconon_main.spc		SPC	>	Generated 👻
	fipks_itnconon_kernel.spc		SPC	>	Generated 👻
	fipks_itnconon_main.sql In	formation		×	Generated 👻
4	fipks_finconon_kernel.sql				Generated 👻
	GWTM_NOTIFICATIONS_MAST	Error Description	Erro	or Code	Generated 👻
	GWTM_NOTIFICATION_TAG_N	Request successfully Processed	RD-	SAVE-007	Generated 👻
•	FTNCONON_RAD xml				Generated -
				-	-
				Ok	Generate

Fig 4.7.3: Notification Screen Generation Successful

Modifying an Existing Notification RADXML

The process of modifying an existing Notification RADXML is illustrated in the images below

Notification Maintenance			" ×
Nolification Function	Action Load 👻	Load Screen Xml	BROWSE
Nolification Code *	Module		
Upload X	Choose File to Upload		—X —
	✓ ✓ ✓ ✓ MAIN ► FT ► RADXML	✓ Search RADXML	٩
Upload File Browse	Organize 👻 New folder		
	Downloads Name	Date modified	Туре 🔺
	CSCFTDUP_RAD.xml	12/23/2012 9:48 AM 12/23/2012 9:48 AM	
RECORD	Desktop	12/23/2012 9:48 AM	
	Libraries FTDCAUTH_RAD.xml FTDCAUTH_RAD.xml	12/23/2012 9:48 AM	
	ETDCONAU_RAD.xml	12/23/2012 9:48 AM 12/23/2012 9:48 AM	
	System (C:)	12/23/2012 9:48 AM	
	PTDDSHBD_KAD.xml	12/23/2012 9:48 AM	
	FTDMCKCH_RAD.xml	12/23/2012 9:48 AM 12/23/2012 9:48 AM	
	Xperia L + + +		•
	File name:		
		Open Ca	ancel
Web Service Tags			
Order Xsd Field Table Field	Data Type Maximum I	Length	
		*	

Fig 4.7.4: Notification Screen Loading

									[]	×	V	
			Notification Function F1	INCONON	Actio	n Load 👻	Save Xml F	Path FTNCONON_RAD.xr	BROWSE			
I	Notific	fication Code	* NOTIF_FT_CONTRACT			Module	FT	×1				Î
		Description				Module Description	Funds Transfer	2				
	Notif	tification Xsd				Base Table	* CSTB_CONTRACT					
	ſ	Firing Time	Insert 👻			PK Cols	* CONTRACT_REF_NO					
		Filter Type	PIsql Block 🔹			PK Types	* VARCHAR2					
	Gatev	eway Service	FCUBSFTService				Full Screen Reply					
		ay Operation		×=			HO Only					
Gate	ateway	y IO Request	Contract-Details-IO									
		e XSD Name										
				Record as SCURRENT_RECORD		END IF; RETURN TRUE;	*					
ilter Lo	IF	IF \$CURREN	NT_RECORD.module_cod			END IF; RETURN TRUE;	A 					
ilter Lo	IF		NT_RECORD.module_cod			END IF; RETURN TRUE;	*	*-				
V	Web S	IF SCURREN Service Ta	NT_RECORD.module_cod	ie = 'FT THEN \$NOTIFY := 'Y'; ELS Table Field	E \$NOTIFY := 'N';	Data Type	A Maximum Length	*-				
V IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII	Web S Orr	Service Ta	Igs Xsd Field SOURCEREFNO	le = FT THEN \$NOTIFY := Y; ELS Table Field EXTERNAL_REF_NO	E \$NOTIFY := 'N';	Data Type VARCHAR2	Maximum Length	•				
ilter Lo	Web S Orr	Service Ta	NT_RECORD.module_cod	ie = 'FT THEN \$NOTIFY := 'Y'; ELS Table Field	E \$NOTIFY := 'N';	Data Type	Maximum Length					

Fig 4.7.5: Notification Screen Loaded

Notification Function FTNCONON Action Load Save Xml Path FTNCONON_RAD.xml BRO Notification Code + NOTIF_FT_CONTRACT Module FT ## Description This is the notification indicating that a FT Module Secription Funds Transfer ## Notification Xsd Inset Image: The notification indicating that a FT Module Secription Funds Transfer ## Firing Time Inset Image: The Notification Xsd
Notification Code • NOTE_FT_CONTRACT Module FT ## Description This is the notification indicating that a FT Module Description Funds Transfer Notification Xsd
Description This is the notification indicating that a FT Notification Xsd Firing Time Insert • Filter Type Plsg Block • Filter Type Cateway Operation QueryContract Gateway IO Request Contract-Details-IO Type XSD Name Iter Logict Set Stronter Type Number • Iter Logict Set Stronter Type Number • Web Service Tags Web Service Tags Order Xsd Field Table Field Deta Type Maximum Length • 1 SOURCEREFNO ExtERNAL_REF_NO Filter Type Maximum Length •
Description This is the notification indicating that a FT Notification Xsd Fining Time Insert • Pisql Block • Filter Type Pisql Block • Pisql Pisql Block • Pisql Pisql Block • Pisql
Firing Time Insert PK Cols CONTRACT_REF_NO Filter Type Plisgi Block PK Types VARCHAR2 Gateway Service COUSERTService PI Full Screen Reply Gateway Operation QueryContract PI HO Only Gateway IO Request Contrad-Details-IO HO Only Gateway IO Request Contrad-Details-IO HO Only Type XSD Name Iter Logic (Set ShOTIFY to YIN & Refer Current Record as \$CURRENT_RECORD) If \$CURRENT_RECORD.module_code = FT THEN \$NOTIFY := Y; ELSE \$NOTIFY := N; END IF; RETURN TRUE; Image: Contract C
Filter Type Plsq Block Gateway Service FOUBSFTService Gateway Operation QueryContract Gateway Operation QueryContract Type XSD Name HO Only Gateway Io Request Contract-Details-IO Type XSD Name If SCURRENT_RECORD IN & Refer Current Record as \$CURRENT_RECORD I Iff \$CURRENT_RECORD.module_code = 'FT' THEN \$NOTIFY := 'Y', ELSE \$NOTIFY := 'N'; END IF; RETURN TRUE; Veb Service Tags Image: Contract Type State Short Type Order Xsd Field Table Field Data Type 1 SOURCEREFNO ExtERNAL_REF_NO VarCHAR2 64
Gateway Service FOUBSFTService Fill Gateway Operation QueryContract Fill Screen Reply Gateway IO Request Only Type XSD Name HO Only Itter Logici Set SNOTIFY To YN & Refer Current Record as \$CURRENT_RECORD I If \$CURRENT_RECORD.module_code = 'FT THEN \$NOTIFY := Y', ELSE \$NOTIFY := N', END IF, RETURN TRUE; If \$CURRENT_RECORD.module_code = 'FT THEN \$NOTIFY := Y', ELSE \$NOTIFY := N', END IF, RETURN TRUE; Veb Service Tags Order Xsd Field Table Field Data Type Maximum Length 1
Gateway Operation QueryContract HO Only Gateway IO Request Contract-Details-IO Type XSD Name Type XSD Name Itter Logic(Set SNOTIFY To VN & Refer Current Record as \$CURRENT_RECORD) F \$CURRENT_RECORD.module_code = FT THEN \$NOTIFY := Y; ELSE \$NOTIFY := N; END IF; RETURN TRUE; If \$CURRENT_RECORD.module_code = FT THEN \$NOTIFY := Y; ELSE \$NOTIFY := N; END IF; RETURN TRUE; Image: Contract-Details - ID Web Service Tags Image: Contract-Details - ID Image: Contract-Details - ID Order Xad Field Table Field Data Type Maximum Length Image: Contract-Details - ID Image: Contract-Details - ID 1 SOURCEREFNO ExtERNAL_REF_NO Image: Contract-Details - ID
Gateway ID Request Contract-Details-IO Type XSD Name Iter Logic(Set SNOTEY To YN & Refer Current Record as SCURRENT_RECORD) If \$CURRENT_RECORD.module_code = 'FT' THEN \$NOTIFY := 'N'; END IF; RETURN TRUE; Web Service Tags Veb Service Tags Corder Xsd Field Table Field Data Type Maximum Length 1 SOURCEREFNO EXTERNAL_REF_NO #2 VARCHAR2 64
Type XSD Name Iter Logic(Set SNOTEY To VN & Refer Current Record as SCURRENT_RECORD.) IF \$CURRENT_RECORD.module_code = FT THEN \$NOTIFY := Y; ELSE \$NOTIFY := 'N; END IF; RETURN TRUE; Web Service Tags Veb Service Tags Crider Xsd Field Table Field Data Type Maximum Length 1 SOURCEREFNO EXTERNAL_REF_NO # VARCHAR2 64
If SCURRENT_RECORD.module_code = FT THEN \$NOTIFY := Y; ELSE \$NOTIFY := N; END IF; RETURN TRUE; Web Service Tags Order Xsd Field Table Field Data Type Maximum Length 1 SOURCERENO
Web Service Tags + - Order Xsd Field Table Field Data Type Maximum Length 1 SOURCEREFNO EXTERNAL_REF_NO FE VARCHAR2 64
Order Xsd Field Table Field Data Type Maximum Length 1 SOURCEREFNO EXTERNAL_REF_NO # VARCHAR2 64
1 SOURCEREFNO EXTERNAL_REF_NO PE VARCHAR2 64
2 CONTREFNO CONTRACT_REF_NO
3 BOOKDT BOOK_DATE FE DATE 64

Fig 4.7.6: Notification Screen Loaded and Modified

5 Deploy Notification

5.1 Notification - Workbench related deployment

Compile the following files in Target FLEXCUBE UBS Database schema

- Notification Main Package generated from ODT
- Hook Packages
- GWTM_NOTIFICATION_TAG_MAP___<Notification Function
- ID>_.INC
 - GWTM_NOTIFICATIONS_MASTER___<Notification Function ID>_.INC

5.2 Notification Trigger deployment

Compile the following files in Target FLEXCUBE UBS Database schema

- GWTM_NOTIFICATION_TRIGGERS__TRIG_CONTRACT.INC
- GWTR#_TRIG_CONTRACT.TRG

6 Test Notification

This section explains the run time notification flow and testing steps.

6.1 Notification flow

The notification process occurs as two parts:

- 1. Oracle JOBs created using FCJ Scheduler framework that sends data required for notification to an internal JMS queue.
- 2. Gateway MBD that lists on internal JMS queue, that picks the notification XMLs and prepare full web service response and send to external system queues.

6.2 Scheduler based notification

The Notification Process in FLEXCUBE can be done using the jobs scheduler as follows:

The trigger generated from Workbench will be inserting key details into a static notification log (STTB_NOTIFICATION)

Once Job is triggered, a request is sent to EJB layer from job execution class and the notification log table will be polled for unprocessed records.

Each unprocessed record is locked.

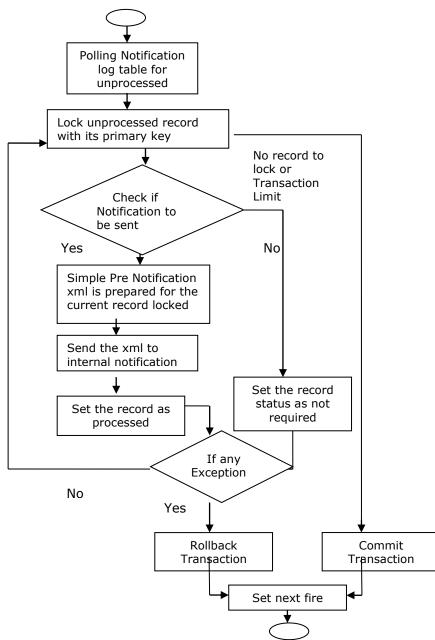
The record is verified against the notification maintenance and checked whether notification is to be sent or not.

If notification is to be sent, pre notification message xml is built and it is sent to internal NOTIFY_QUEUE(JMS queue) configured in Gateway layer.

The job is then rescheduled to fire next time based on the previous execution.

Refer Gateway Installation documents on how to setup the Queues.

Flow Chart for Notification Flow in Scheduler



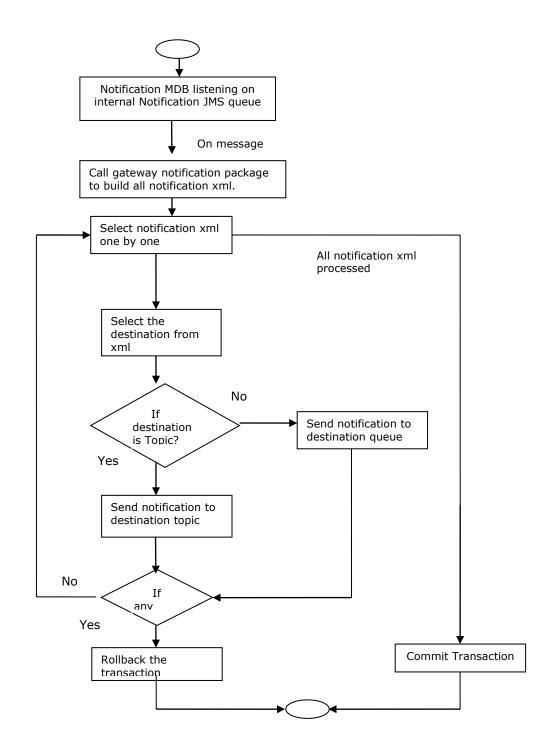
6.3 MDB based notification flow

Notification processes in MDB are as follows:

- 1. Notification MDB listens on the internal NOTIFY_QUEUE(JMS queue)
- 2. On any message received, the MDB identifies which schema to connect using the JNDI name being present as part of the message xml.
- 3. Gateway notification processing package is called from MDB to build notifications.

- 4. In MDB, the notifications built is processed and sent to the destination specified in corresponding notification.
- 5. In case of exception the transaction is rolled back.
- 6. If all notifications are successfully processed, transaction is committed.

Flow Chart for Notification Flow in MDB



6.4 Triggering notification and testing

Follow the below steps to test notification

- Simulate a case where base table under goes data change.
- Check record populated at STTB_NOTIFICATION table
- Check Notification message
 GWTBS_NOTIFICATIONS_LOG.NOTIFICATION_MESSAGE



Development Workbench - Notifications June 2018

Oracle Corporation World Headquarters 500 Oracle Parkway Redwood Shores, CA 94065 U.S.A.

Worldwide Inquiries: Phone: +1.650.506.7000 Fax: +1.650.506.7200 www.oracle.com/ financial_services/

Copyright © 2017-2018 Oracle Financial Services Software Limited. All rights reserved.

No part of this work may be reproduced, stored in a retrieval system, adopted or transmitted in any form or by any means, electronic, mechanical, photographic, graphic, optic recording or otherwise, translated in any language or computer language, without the prior written permission of Oracle Financial Services Software Limited.

Due care has been taken to make this *Development Workbench-Notifications* and accompanying software package as accurate as possible. However, Oracle Financial Services Software Limited makes no representation or warranties with respect to the contents hereof and shall not be responsible for any loss or damage caused to the user by the direct or indirect use of this *Development Workbench-Notifications* and the accompanying Software System. Furthermore, Oracle Financial Services Software Limited reserves the right to alter, modify or otherwise change in any manner the content hereof, without obligation of Oracle Financial Services Software Limited to notify any person of such revision or changes.

All company and product names are trademarks of the respective companies with which they are associated.