Development Workbench Service XML Development
Oracle FLEXCUBE Investor Servicing
Release 14.4.0.0.0
[July] [2020]

ORACLE

Table of Contents

1 Preface

This document describes the webservice development using Oracle FLEXCUBE Development Workbench for Universal Banking.

1.1 Audience

This document is intended for FLEXCUBE Application developers/users that use Development Workbench to develop various FLEXCUBE components.

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

Proficiency	*Resources*
FLEXCUBE Functional Architecture	Training programs from Oracle Financial Software Services.
FLEXCUBE Technical Architecture	Training programs from Oracle Financial Software Services.
FLEXCUBE Object Naming conventions	*Development Overview Guide*
Working knowledge of Web based applications	Self Acquired
Working knowledge of Oracle Database	Oracle Documentations
Working knowledge of PLSQL developer	Respective vendor documents
Working knowledge of PLSQL & SQL Language	Self Acquired
Working knowledge of XML files	Self Acquired

2 Introduction

This Document explains the steps to create/Modification of Service xml and generating webservice artifacts for building ear file using the Oracle FLEXCUBE Development Workbench for Universal Banking

3 Service XML

Oracle FLEXCUBE Development Workbench provides the developer with a user friendly console for defining a gateway service of FCUBS.
One Service XML corresponds to one Gateway Service. All the Function Ids which are part of the particular service would be captured in the Service XML along with the Operation details.

ODT assist developers in developing the webservice with the capability of generating the Following artifacts for building ear file.

Files	Description
<Service Name>Src*Impl.java	IMPL files for service
<Service Name>WSDL*.wsdl	WSDL files for service
<Service Name>Config*.xml	Config files
<Service Name>XSD*.xsd	Service specific xsd's
<Service Name>Common*.xsd's	Common XSD's (call forms) part of service
<Service Name>\<Service Name>\META-INF\application.xml <Service Name>\<Service Name>\META-INF\MANIFEST.MF	Config XML's for building the Web service
<Service Name>\<Service Name>\commons-codec-1.2.jar	Utility Jar for building the web service
<Service Name>\<Service Name>\wscommon.jar	Utility Jar for building the web service
Sample Ant file	For building service ear file

Note: Non-extensibility function Id's operations don't allow add/modify any existing service; it will allows delete operation only.

3.1 Process Steps

Login to the Oracle FLEXCUBE ODT using the credentials maintained (refer 02-ODT Administration.docx for creating users)

Map the session to the release and environment as required (Refer 03-ODT Getting Started.docx for detailed explanation)

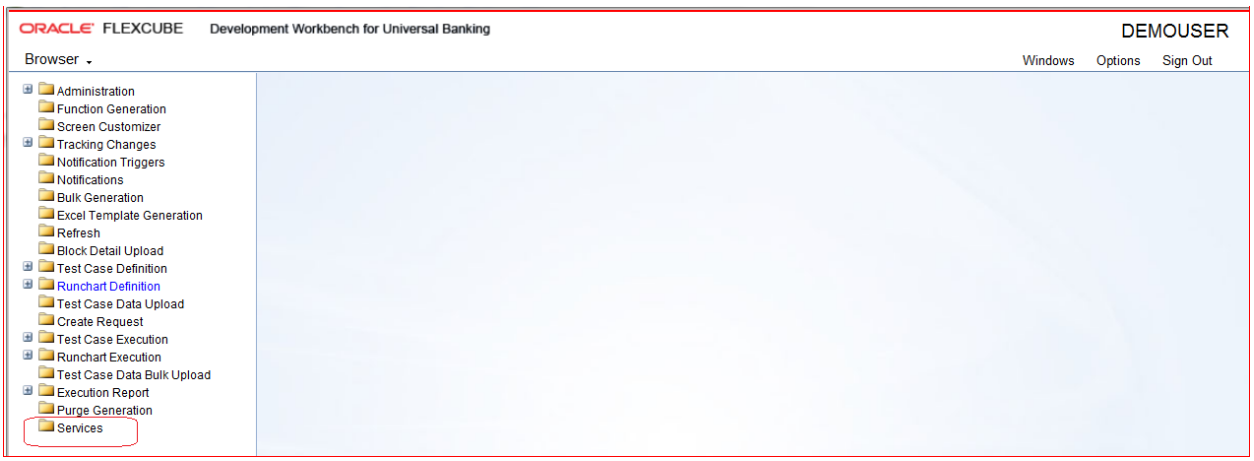

Fig 3.1: ODT Screen Showing Services Option

Click on Service node in the browser tree found in the Landing page of ODT.

Services window gets launched, while creating a new Service in ODT, below information needs to be provided in the Header section.

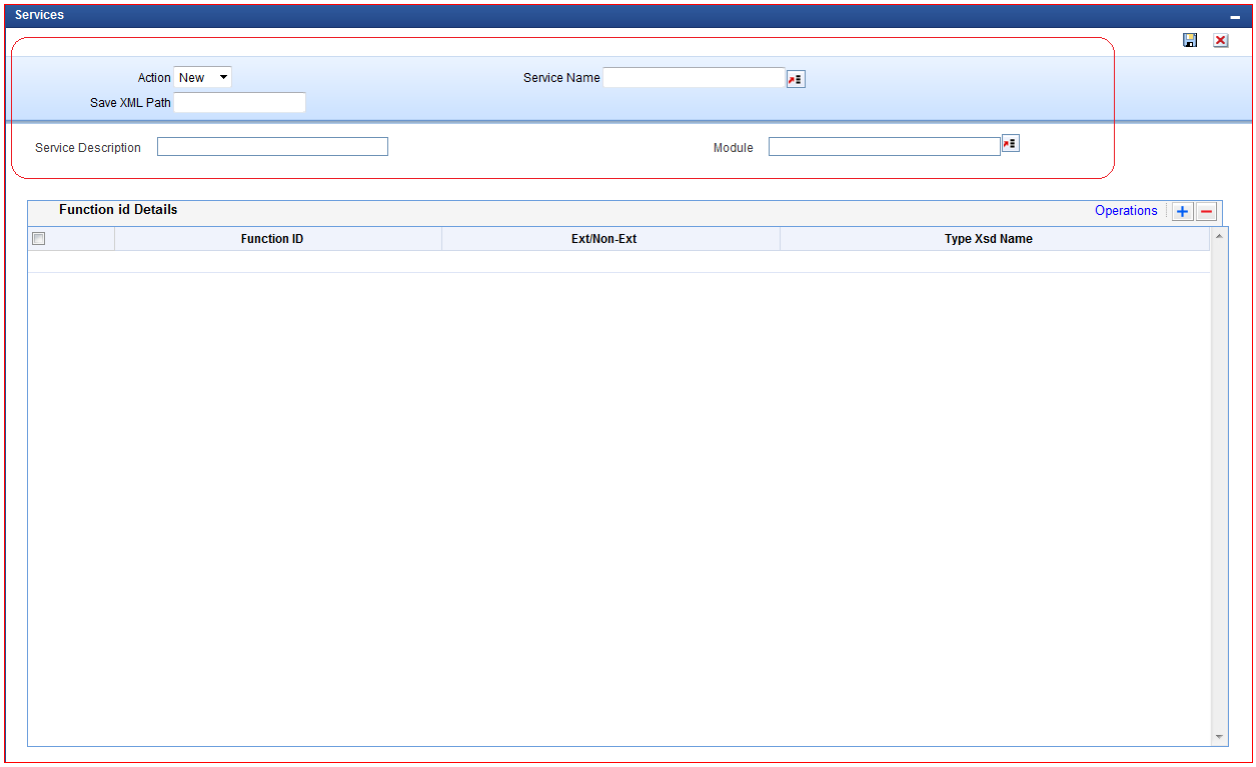

Fig 3.2: Service Screen

While loading an Existing Service in ODT

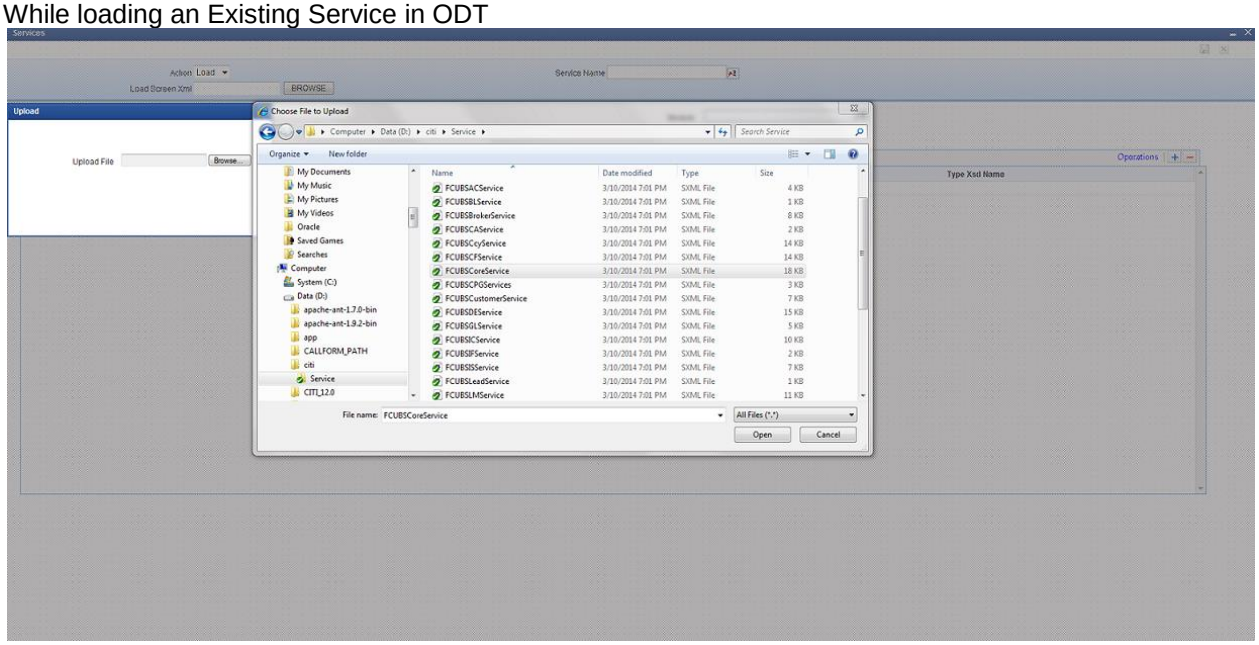

Fig 3.3: Loading Service XML

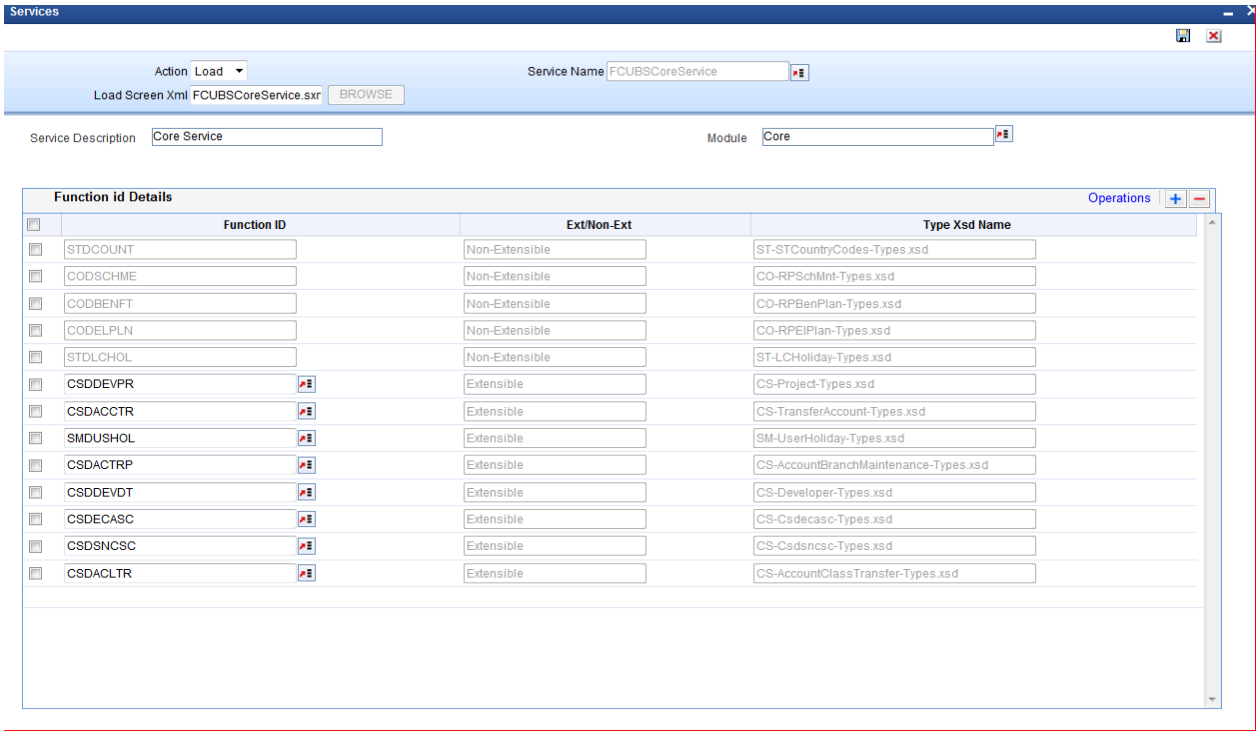

Fig 3.4: Screen after Loading Service XML.

The Header portion of the Function Generation screens consists of the following fields:

Action

New and Load options are provided for this field. For a new Service development, select the action as New

If the action is load then corresponding Service xml has to be loaded using browser option in Save Xml Path; all the header information will get populated.

Service Name

If the action is selected as new, Service name has to be selected from service LOV.(Service LOV will fetch values from GWTM_SERVICES_MASTER For new service, service name needs to be added in GWTM_SERVICES_MASTER of business schema)

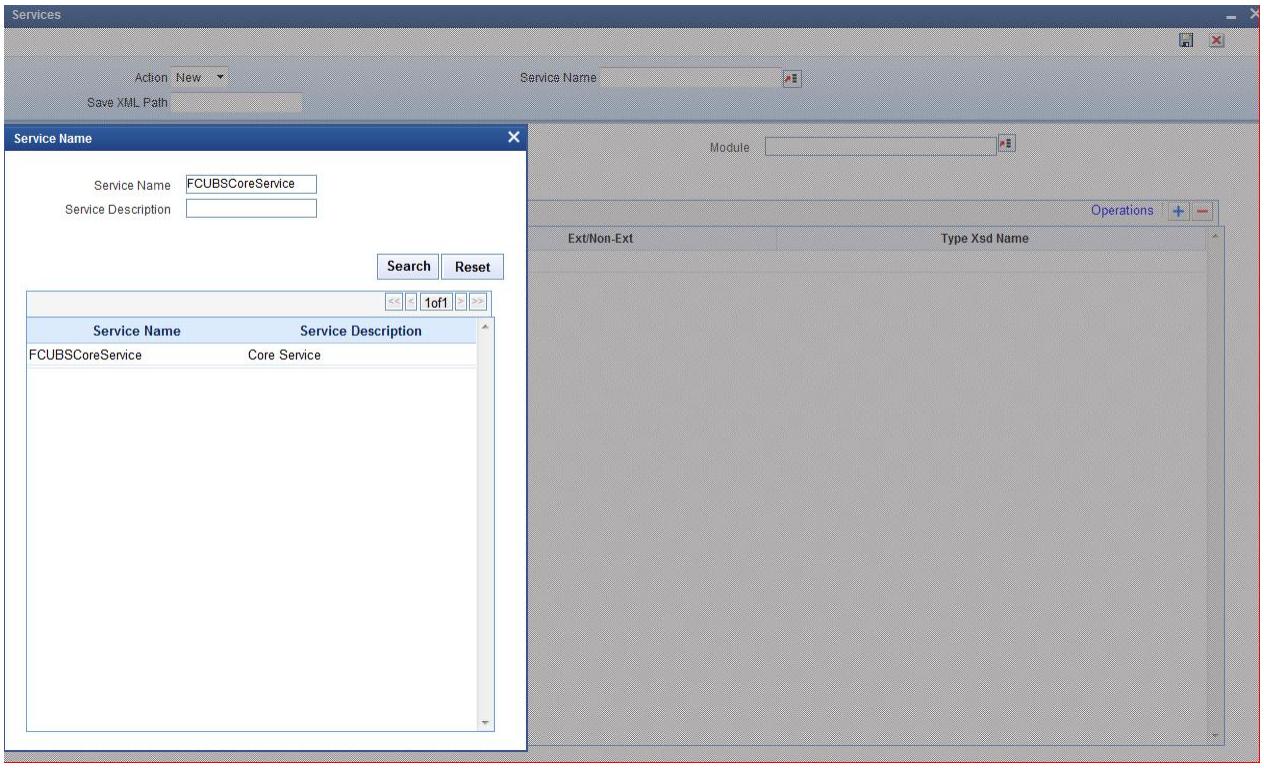

Fig 3.5: Lov to populate Service name.

Service Description

On Select of service name service description will be populated in service description field

Module

If the action is selected as New, Module has to be selected from Module LOV.(Module LOV will fetch values from SMTB_MODULES of Business schema).

Module Name need not always be from the LOV. Note that Artifacts would be generated based on the Module Name specified. Hence provide source Folder module names (Example: Core for CS) in this field

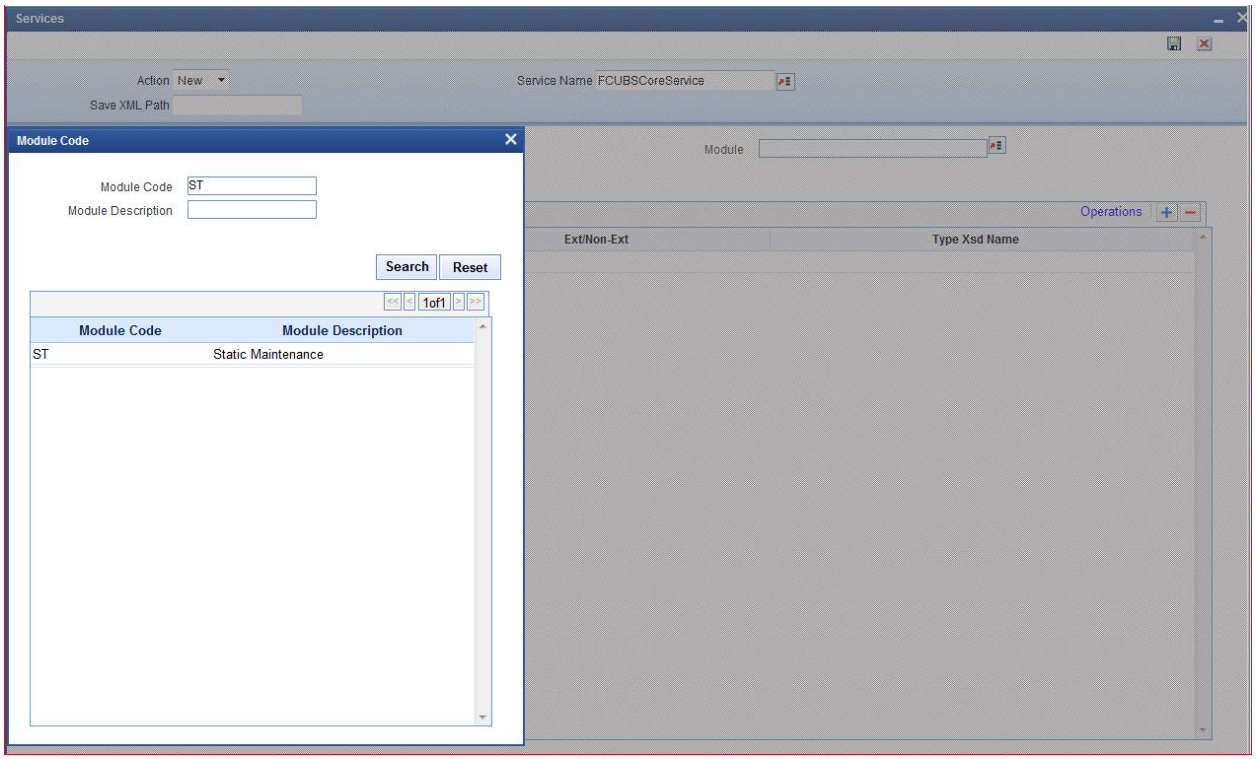

Fig 3.6: Lov to populate Module Code.

Save Xml Path

If the action is New, save xml path is optional. If provided, then the generated units will be saved in the path mentioned.
Note that the value in the Save Xml Path will be used only if the Save Format is Client Path and if the User has given “CURRENT_DIRECTORY” in the User Preferences Work Directory.

The label description of the field will change depending on the action .If the action is load, ODT attaches a Browse button to it so that user can browse the Service xml and load it.

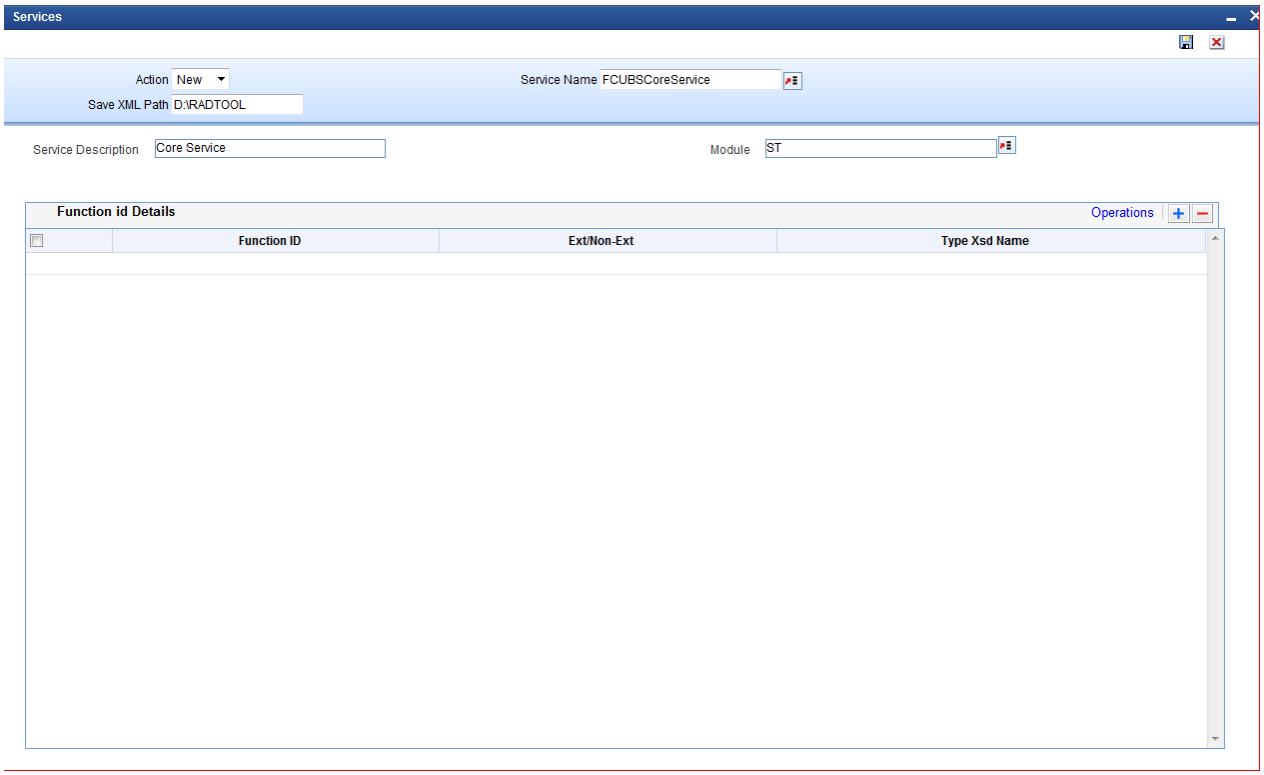

Fig 3.7: Screen to Show save Xml Path.

Function Id Details:

Developer can attach the Function IDs which are part of this service.
He can also remove the same from service if not required.

Function Id

Select the function id from function id LOV for adding the function id for that service
Function Id LOV will populate data from SMTB_MENU.
Make sure that FC_FUNCTION_ID values are selected for Function Id so that physical radxml file for the same function Id exists.
Example: Select STDCIF and not STGCIF

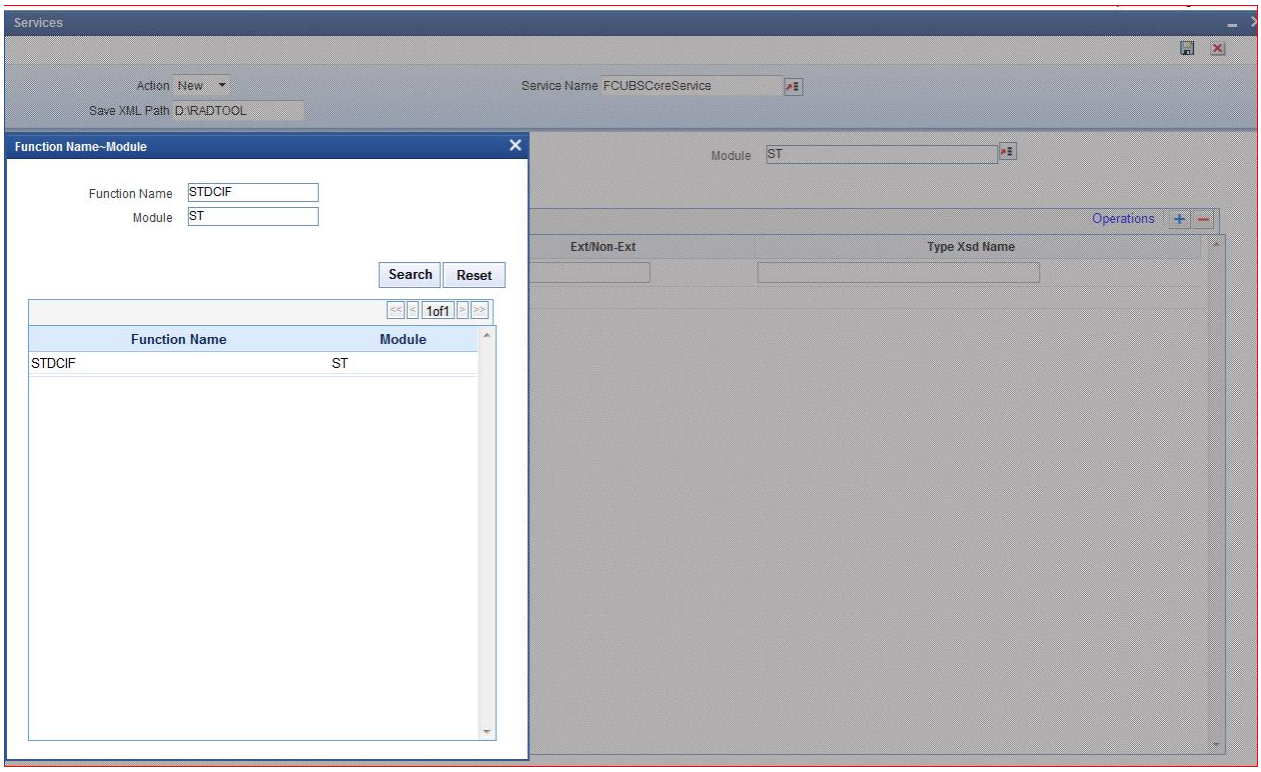

Fig 3.8: Lov to populate Function Id.

Extensible/Non-Extensible:

On Select of Function id, this field value would be populated

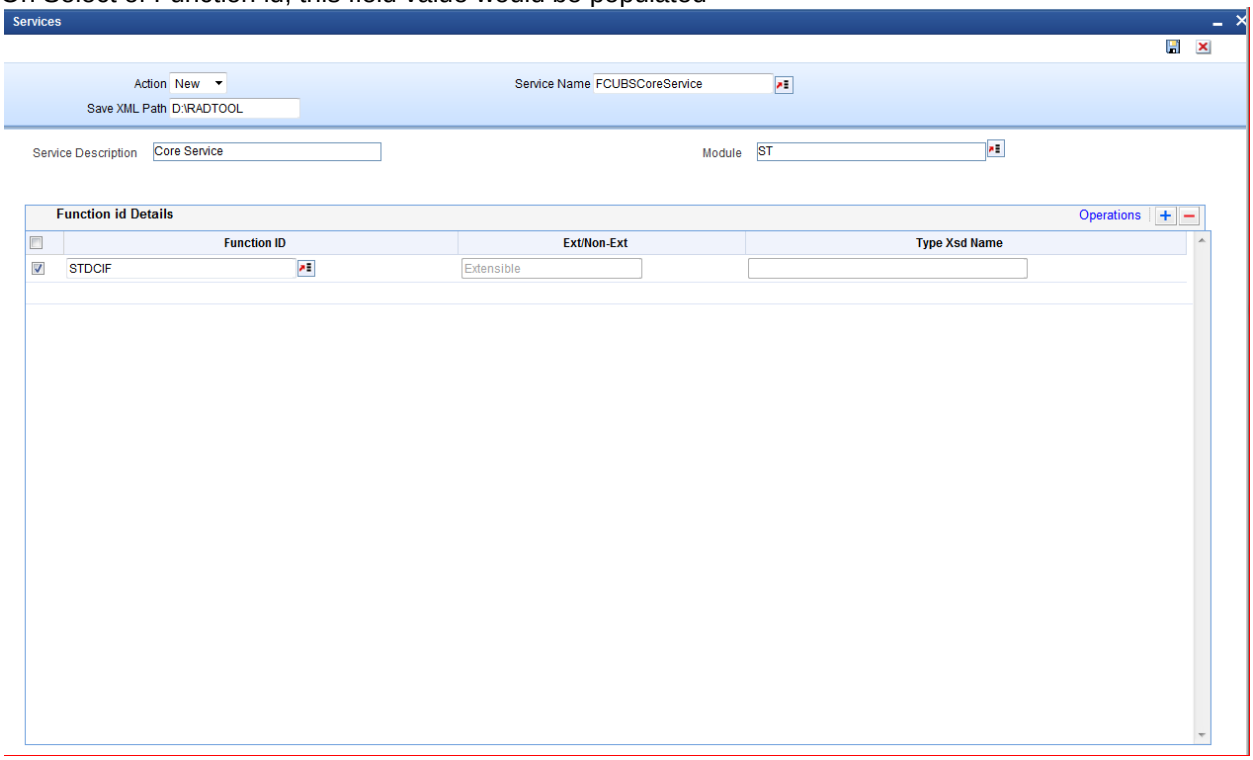

Fig 3.9: Screen to populate Function Id Value.

Type Xsd Name:

Type XSD name would be defaulted along with Operations for an Extensible Function Id.
For Non Extensible Function Id, Type XSD name has to be explicitly mentioned in the field

Operation:

Operation Codes would be defined in each radxml which has to be defaulted in the Service XML as well.
Select function id checkbox and click on Operations.
Operation details popup screen will be displayed.

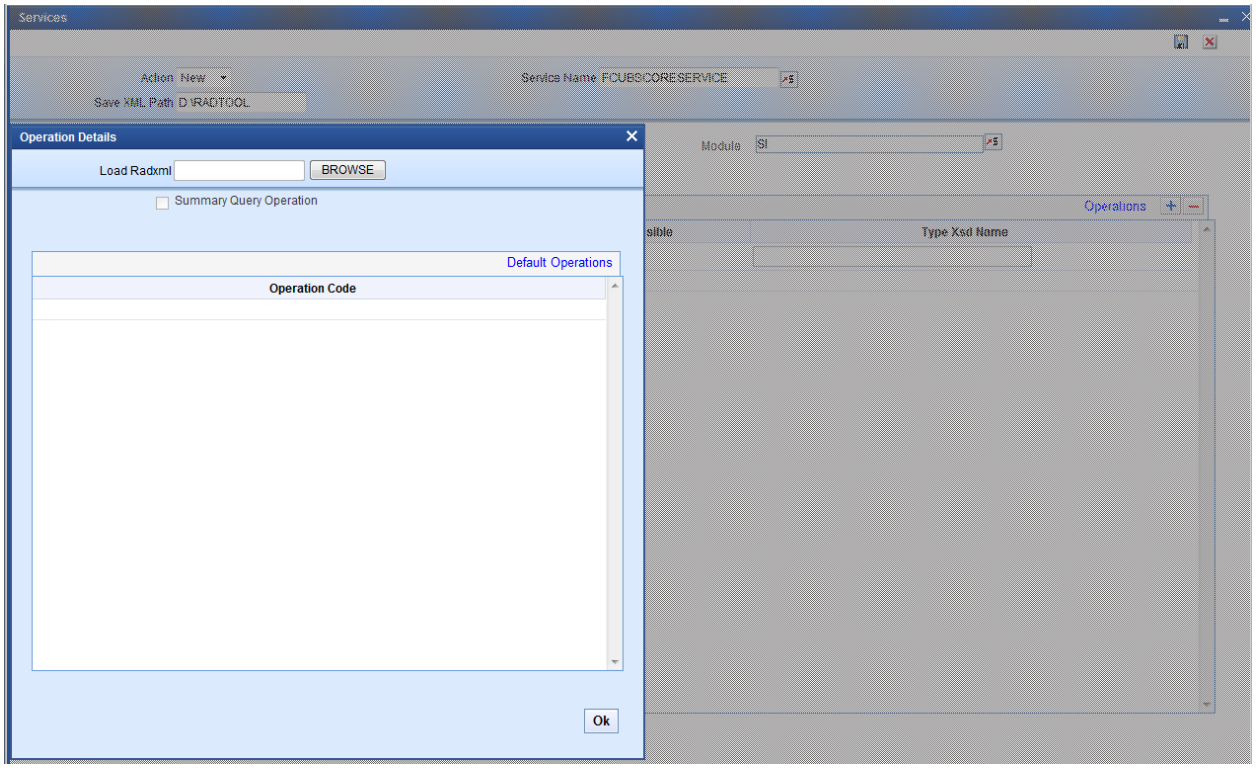

Fig 3.10: Screen to Show Operations.

Operation Details Screen:

Load Radxml

Operation details screen attaches a Browse button to it so that user can browse the Function id RAD XML and load it to populate operations

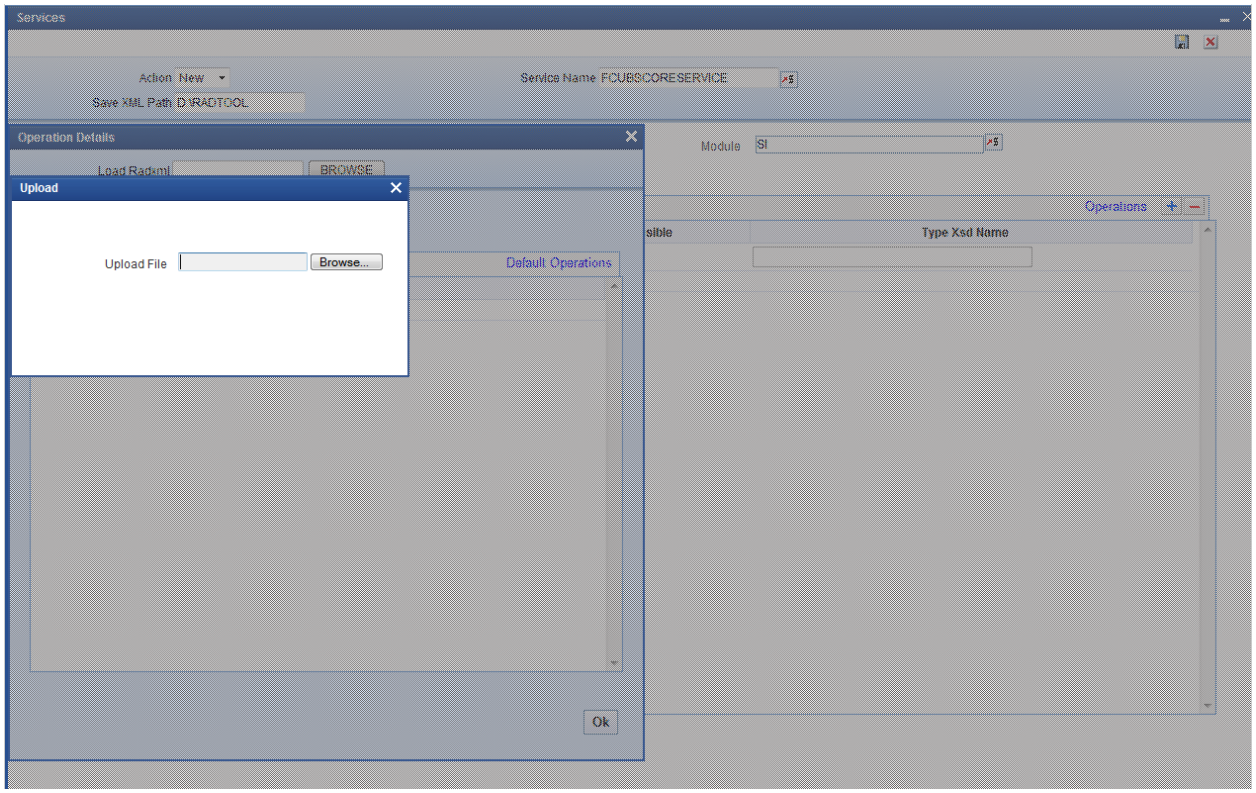

Fig 3.11: Screen to Load Radxml.

Default Operations:

Click on Default operation. Operation code and Type Xsd Names will be default from loaded RADXML.

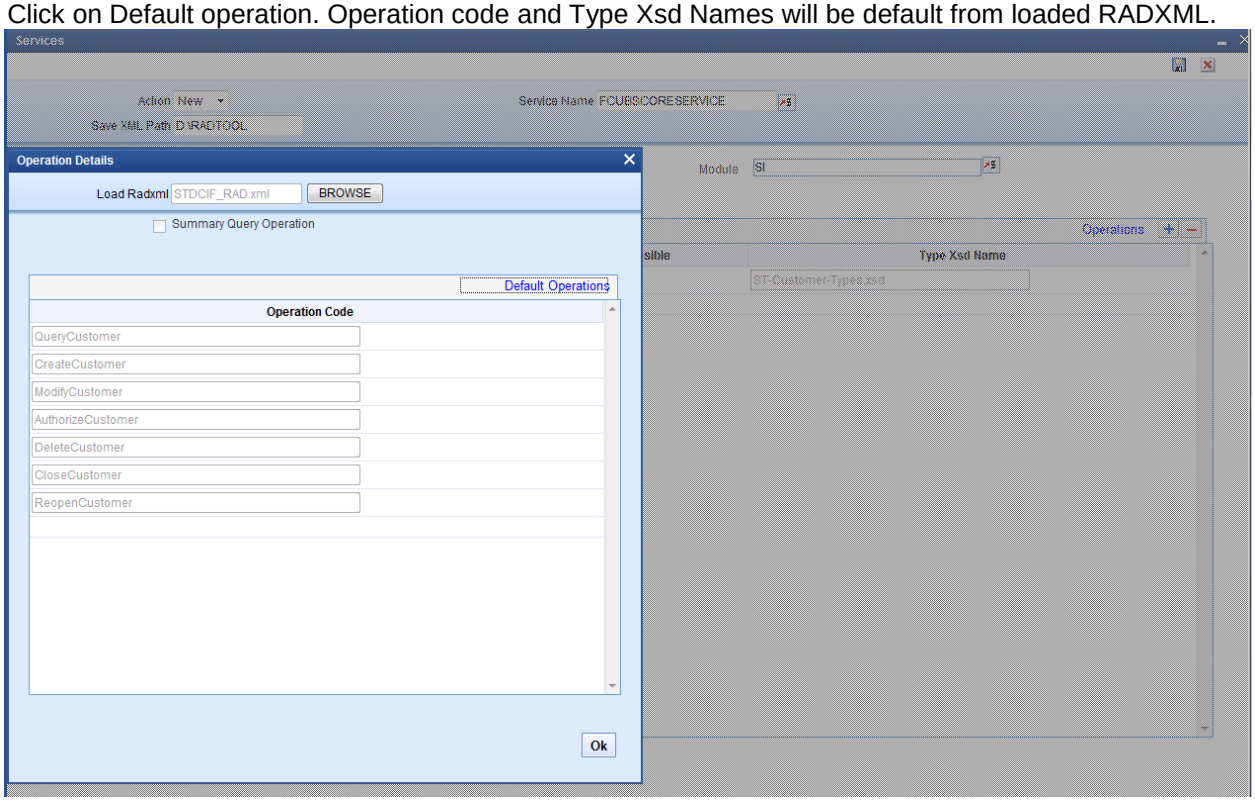

Fig 3.12: Screen to Default Operations..

Save: ServiceXML

ODT saves all the activities carried out by the developer in an xml file hereby referred to as SXML. Persistence of the WEBSERVICE is achieved through SXML and RADXML.
If some changes are required on the webservice in a future release, the same SXML can be loaded and changes can be done on this SXML. ODT can segregate the changes done on different releases and saves the SXML accordingly.
SXML will adhere to following naming convention
Service Name + .sxml
Example: FCUBSCoreService.sxml

4 ODT Silent Utility

The Following operations are supported in silent utility of ODT

1. LOGIN
2. SETRELEASE: Setting Release and Environment Details
3. BULKGENERATION: Bulk Generation of RADXML's units
4. REFRESH: Bulk refresh of RADXML's
5. SXML_REFRESH: Bulk refresh of Service XML's
6. SXML_UPDATER: Bulk Updater of service XML's based on the changes in RADXML's
7. SXML_BULKGENERATION: Bulk Generation of web service artifacts.

Execution of Operation will be as per the sequence maintained in OdtOperations.properties.

Example:

1. Operation = LOGIN

--

2. Operation= SETRELEASE

--

3. Operation=REFRESH

If sequence of operations is as above, then Login Operation, Set Release and Refresh Operations would be processed in respective sequence

*Note**: login and set release are mandatory operations to be performed.***

4.1 Prerequisites

- **JDK**

  License Information:
  JDK is distributed by Sun Microsystems, Inc under Java Development Kit Binary Code License agreement.
  Instructions:
  Installer requires JDK 1.7.xx_xx version to be downloaded in the system and the same Should be set as environmental variable

- **Apache Ant 1.7.1**

Instructions:
Installer requires ANT 1.7.1 version to be downloaded in the system and the same should be set as environmental variable

4.2 How to run utility on Windows/Unix

After copying the installer sources and library folder to your local system, make sure you uncheck the read only check box in source properties and apply the same to all the sub folders.
The screen shot below shows how the source folder in your local system should look like.

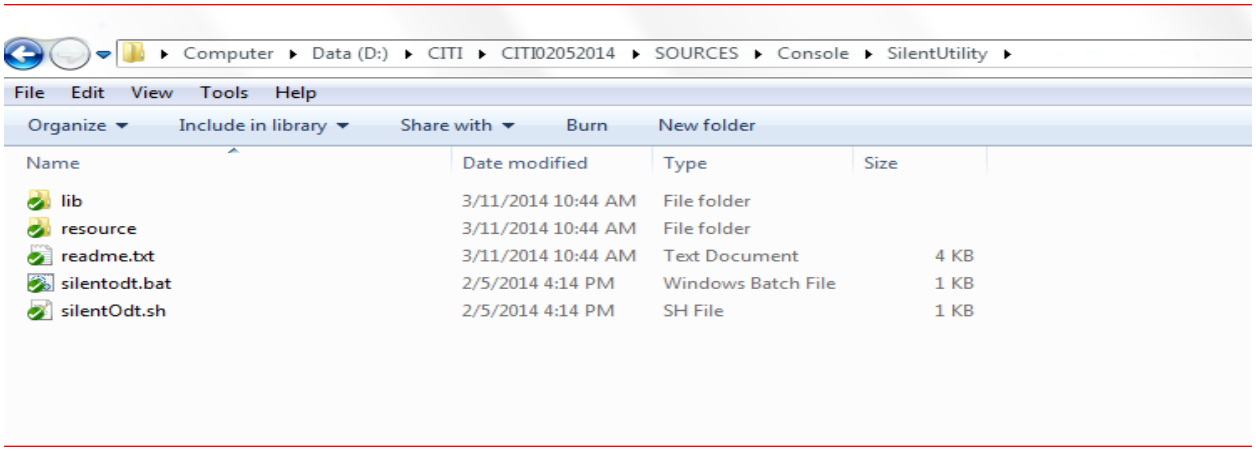

Fig 4.2.1: Source of SilentODTUtility.

4.3 Configuration of SilentODTUtility

All Configuration files can be found inside /resource folder of the utility.
Note: Please copy jaxb-xjc.jar to lib folder. This can be obtained from application server libraries. For example, in Weblogic 12c
<Oracle_Home>\oracle_common\modules\com.sun.xml.bind.jaxb-jxc.jar

Also make sure ojdbc6.jar is available under lib folder

4.3.1 SilentOdt.properties

The sample property file has been given below. Please refer the details mentioned for each property in the below table. Some of them are encrypted using ODTPassEncryption.bat (ODTPassEncryption.sh for unix). Use 16 characters length of symmetric key for encryption (Preferably Alphanumeric) which will be prompted for input from user when the encryption utility is lunched. The same symmetric key must be mentioned in the property file as well.

```
#####################################################################
##ODT Version  ==> 12.2/12.1/12.0.2/12.0.1/11.4 etc
##type         ==> FCUBS/FCIS/ELCM  ; specifies the product
##release      ==> Release of the product specified
#####################################################################
odtVersion=12.2
#FCUBS,FCIS,ELCM,PAYMENTS
type=FCUBS
release=FCUBS_12.2.0.0.0
productDesc=Oracle FLEXCUBE Universal Banking
ReleaseMonth=May
ReleaseYear=2016

###ODT DataSource Credentials
#########################################################
OdtJdbcUrl=O40fjmATqNKRRdNFP9UR3eeXUUMaPnZJ1gtHVtXkIyVEgM1qCkuNqcIs96vR4NFq
OdtDbUser=ODT121
OdtDbPassword=ITgBkLEJpGO6AuYE6jJkmg==
SymmetricKey=oraclefinancials

############ Logger Properties#############################
#logreqd Default set to N
#Default Path set to User Home directory, if not provided
#LEVEL ==>DEBUG/INFO/WARNING/SEVERE   ; default value is INFO
#########################################################
logreqd = Y
logpath = D:/DESTTEMPDIR/ODT/log.txt
level = DEBUG


##########System Properties#############################################################
##JAVA_HOME is mandatory
##WEBLOGIC and WebSphere Home would be required only if ANT scripts are being generated.
##Use Backward Slash(\) for File Separator
#######################################################################################
JAVA_HOME=C:\Program Files\Java\jdk1.8.0_73
WEBLOGIC_HOME=D:\Oracle\Middleware
WAS_HOME=D:\WAS
```

Fig 4.3.1: SilentODT Properties.

ODT Data Source Detail Credentials	
OdtJdbcUrl	Jdbc Url jdbc:oracle:thin:@10.184.xx.xx:1521:FCDEMO
OdtDbUser	DB User name
OdtDbPassword	DP Password
Logger Properties	
Logreqd	Y/N. Default set to N,
Logpath	Provide the path where the Logger files will be generated.
Level	Provide the Logger Level. This can be either DEBUG/INFO/WARNING/SEVERE. Provide as DEBUG for writing detailed log. Default value would be set to INFO
System Properties:	
JAVA_HOME	Maintain the Java installed location
WEBLOGIC_HOME	Maintain the oracle weblogic installed location

WAS_HOME	Maintain the IBM websphere installed location

Note: *WEBLOGIC_HOME and WAS_HOME are optional and would be used for generating template Ant scripts whereas JAVA_HOME is mandatory to run silentOdt utility*

4.3.2 ODTOperations.properties

Configure the Operations files as per Requirement.

The Following operations are supported in silent utility of ODT

8. LOGIN
9. SETRELEASE: Setting Release and Environment Details
10. BULKGENERATION: Bulk Generation of Radxml units
11. REFRESH: Bulk refresh of radxml
12. SXML_REFRESH: Bulk refresh of Service Xmls
13. SXML_UPDATER: Bulk Updater of service Xmls based on the changes in radxml's
14. SXML_BULKGENERATION: Bulk Generation of web service artifacts.

Execution of Operation will be as per the sequence maintained in OdtOperations.properties.

Example:

1. Operation = LOGIN

--

2. Operation= SETRELEASE

--

3. Operation=REFRESH

If sequence of operations is as above, then Login Operation , Set Release and Refresh Operations would be processed in respective sequence

*Note**: login and setrelease are mandatory operations to be performed.***

4.3.3.1 Login

Userid: Provide the ODT Userid which is created in the ODT Application

Password: Provide the ODT Password which is created in the ODT Application

```
1.operation = LOGIN
1.userId= RADTOOL
1.password= wS/PEjVOI5pdJ7aYvjLuNQ==
```

Fig 4.3.3.1: SilentODT Login Properties

Login should always be the first operation which to be configured as part of any execution

Login to Tool	
operation	Login
Userid	ODT Userid which is created in the ODT Application
password	ODT Password which is created in the ODT Application. Encrypted using ODTPassEncryption.bat. Refer section 4.3 for more details about encryption.

4.3.3.2 Set Release

This operation can be used for setting Release and Environment Preferences for SilentODTUtility

relcode: Provide the ODT Release Code which is created in the ODT Application
envCode: Provide the ODT Environment Code which is created in the ODT Application
langcode: Provide the Lang code for above mentioned release code

Connection to the FLEXCUBE schema would be established based on data maintained in ODT or through the data in env_config.xml as explained in earlier section

```
6  ##Set Release and Environment for the User
7  ## 2.operation= SETRELEASE
8  ## 2.relCode=MODEL_BANK
9  ## 2.envCode=MODEL_BANK_DEV_ENV
10 ## 2.langCode=ENG
11
```

Fig 4.3.3.2: SilentODT Set Release Properties

Set Release and Environment for User	
operation	SETRELEASE
relCode	ODT Release Code which is created in the ODT Application
envCode	ODT Environment Code which is created in the ODT Application
langCode	Lang code for above mentioned release code

4.3.3.3 Bulk Generation

For generating all radxml artifacts for release in bulk this feature can be used.

radxmlListFile: Prepare text file which contains absolute path of all radxml's. Provide same file path
srcPath: source Path Refers to the path where all radxml's are presented .List File would be generated by the Tool in this case.
Note that only either of radxml List File or srcPath should be present .If both is present, then radxmlListFile parameter would be considered for Bulk Generation
fileType *:*
EXTENSIBLE – artifacts generated only for extensible screens
NON_EXTENSIBLE - artifacts generated only for non-extensible screens
BOTH – artifacts for all files would be generated
destPath: Provide the path where the files will be generated.
gen: Provide the type of files to be generated
example : UIXML, SYS_JS, MAIN_SPC, MAIN_SQL,KERNEL_SPC, KERNEL_SQL

```
##Bulk Generation Utility
## 3.operation=BULKGENERATION
## 3.radxmlListFile=D:\ODT123\ABC.TXT     -- A File containing absolute path of all radxmls to be processed
## #3.srcPath=Z:\FCUBS12.0\MAIN            -- Source Path Refers to the path where all radxmls are presnt.List File would be generated by the Tool in this case.
##                                           Note that only either of radxmlListFile or srcPath should be present .
##                                           If both are present ,then radxmlListFile would be considered for Bulk Generation
## 3.fileType=EXTENSIBLE                  -- EXTENSIBLE/NON_EXTENSIBLE/BOTH
## 3.destpath=D:\RADTOOL                  -- destination Path
## 3.gen = UIXML,SYS_JS                   -- Files to be Generated seperated by coma. Possible entries are listed below
##
UIXML,SYS_JS,MAIN_SPC,MAIN_SQL,KERNEL_SPC,KERNEL_SQL,CLUSTER_SPC,CLUSTER_SQL,CUSTOM_SPC,CUSTOM_SQL,UPLOAD_SPC,UPLOAD_SQL,
##
UPLOAD_TRIGGER,UPLOAD_TABLE_DDL,XSD_FILES,MENU_DETAILS,LABEL_DETAILS,AMEND_DETAILS,SUMMARY_DETAILS,SCREEN_DETAILS,LOV_DETAILS,
##
BLOCK_PK_COLS,CALL_FORM_DETAILS,BLOCK_DETAILS,DATASCR_DETAILS,FUNCTION_CALL_FORMS,GATEWAY_DETAILS,NOTIFICATION_DETAILS,FUNCTION_PARAMETERS
##   NOTIFICATION_TRIGGER,PURGE_DETAILS,ARCHIVE_TBL_DEF
```

Fig 4.3.3.3: SilentODT Bulk Generation Properties

Bulk Generation Utility	
Input	**Output**
radxmlListFile srcPath fileType gen destpath	UIXML,SYS_JS,MAIN_SPC,MAIN_SQL,KERNEL_SPC,KERNEL_SQL, CLUSTER_SPC,CLUSTER_SQL,CUSTOM_SPC,CUSTOM_SQL, UPLOAD_SPC,UPLOAD_SQL,UPLOAD_TRIGGER,UPLOAD_TABLE_DDL, XSD_FILES,MENU_DETAILS,LABEL_DETAILS,AMEND_DETAILS, SUMMARY_DETAILS,SCREEN_DETAILS,LOV_DETAILS, BLOCK_PK_COLS,CALL_FORM_DETAILS,BLOCK_DETAILS, DATASCR_DETAILS,FUNCTION_CALL_FORMS,GATEWAY_DETAILS, NOTIFICATION_DETAILS,FUNCTION_PARAMETERS NOTIFICATION_TRIGGER,PURGE_DETAILS,ARCHIVE_TBL_DEF in destpath

4.3.3.4 Refresh

Refresh Functionality allows developers to upgrade the existing radxml to its later version keeping the sub version specific changes intact. Three kinds of refresh can done using the Tool.(Please refer the 09-Development_WorkBench_Source_Upgrade.docx)

1) Child Refresh
2) Screen Child Refresh
3) Source Refresh

Refresh Type: Provide the refresh Type
(CHILD_REFRESH/SCRCHILD_REFRESH/SOURCE_REFRESH)
srcFileList: A txt File containing the List of all Sources radxml's. I.e. radxml's which has to be refreshed

baseFileList: A txt File containing the List of all base radxml's.

srcRelType: Provide the release type of Source Radxmls list(KERNEL/CLUSTER/CUSTOM)

baseRelType: Provide the release type of base Radxmls list (KERNEL/CLUSTER/CUSTOM)

destpath: Provide the path where the files will be generated

```
##Refresh Utility
## 4.operation=REFRESH
## 4.refreshType=SOURCE_REFRESH         -- Either of CHILD_REFRESH/SCRCHILD_REFRESH/SOURCE_REFRESH
## 4.srcFileList=D:\\REFRESH\\src.txt     -- A txt File containing the List of all Sources radxmls. i.e radxmls which has to be refreshed
## 4.baseFileList=D:\\REFRESH\\base.txt   -- A txt File containing the List of all base radxmls.
## 4.srcRelType=CUSTOM                  -- Release Type of Source Radxmls; Either of KERNEL/CLUSTER/CUSTOM
## 4.baseRelType=KERNEL                 -- Release Type of Base Radxmls;Either of KERNEL/CLUSTER/CUSTOM
##                                        Note that base and Src Release Types should be the same for Child and screen Child Refresh
##                                        Base Release Type should be atleast one level below Src Release type for SOURCE Refresh.
## 4.destpath=D:\\RADTOOL
```

Fig 4.3.3.4: SilentODT Refresh Properties

Refresh Utility	
Input	**output**
refreshType	Refreshed Radxml's in destpath
srcFileList	
baseFileList	
srcRelType	
baseRelType	
destpath	

4.3.3.5 Service XML Bulk Generation

Web service artifacts can be generated through this operation

sxmlListFile: Prepare text file which contains absolute path of all Service xml .

radxmlListFile: Prepare text file which contains absolute path of all radxmls which are used for those services .

xsdListFile: Prepare text file which contains absolute path of all XSDs which are used for those services. Non-extensibility/Common XSDs are copied from this path

srcPath : provide source folder path which is option (Tool will create radxmlListFile and xsdListFile by itself from the srcPath
Note that if srcPath is provided, radxmlListFile and xsdListFile need not be provided

gen: Provide type of Files to be Generated (separated by coma) . Options are IMPL_FILE, CONFIG_FILES,WSDL_FILE,XSD_FILES,GW_WS_PROP_FILES,ANT_BUILD

nonExtServicesReqd: Y/N Specifies whether NonExtensible Operations has to included in the generated Components

destpath: Provide the path where the files will be generated.
validateXsds : Y/N. Default set to Y,
If value set to 'Y' all XSD will be validated by tool

```
40 ## Service XML Componenet Generator
41 ## 5.operation=SXML_BULKGENERATION
42 ## 5.sxmlListFile=D:\\ODT123\\TEST\\srcFile.TXT              -- List Of Absolute path of all sxml files in a text file
43 ## 5.radxmlListFile=                                        -- A txt File containing the List of all radxmls.
44 ## 5.xsdListFile=                                           -- A txt File containing the List of all xsds. This
parameter is required only if NonExt Operations are Required.
45 ##                                                           Nonextensile XSds are copied from this path
46 ## 5.srcPath=Z:\\EXEC\\FLEXCUBE_Kernel\\FCUBS_12.0.0\\MAIN   -- Src Path. Tool will create radxmlListFile and
xsdListFile by itself from the srcPath if provided
47 ##                                                              Note that if srcPath is provided,radxmlListFile and
xsdListFile need not be provided
48 ## 5.gen=IMPL_FILE,CONFIG_FILES,WSDL_FILE,XSD_FILES          -- Files to be generated. Possible entries are Listed
below
49 ##
IMPL_FILE,CONFIG_FILES,WSDL_FILE,XSD_FILES,GW_WS_PROP_FILES,ANT_BUILD
50 ## 5.nonExtServicesReqd=Y                                   -- Y/N Specifies whtherNonExtensible Operations has to
included in the Generated Components
51 ## 5.destpath=D:\RADTOOL                                    -- destination Path
52 ## 5.validateXsds=Y                                         -- validate the xsds Y/N
```

Fig 4.3.3.5: SilentODT Service XML Bulk Generation Properties

Service XML Component Generator	
Input	**output**
sxmlListFile	IMPL_FILE,CONFIG_FILES,WSDL_FILE,XSD_FILES,GW_WS_PROP_FILES,ANT_BUILD in destpath
radxmlListFile	
xsdListFile	
srcPath	
nonExtServicesReqd	
destpath	
validateXsds	

4.3.3.6 Service XML Updater

This feature can be used to update the Service XMLs with the latest data from Radxmls.
Following details will be updated.

1) Any **addition, deletion or modification of operation codes** in function Id would be updated in Service XML
2) If any **function Id is removed from the service (specified in radxml);** then the same would be removed from Service XML
   *Note that if any **new function Id** is attached to the service (in Radxml); then the same will **not be updated in the Service XML**. This has to added manually in the Service XML through ODT user interface*

.
Operation: SXML_UPDATE

sxmlListFile: Prepare text file which contains absolute path of all Service xml. Provide same file path .

radxmlListFile: Prepare text file which contains absolute path of all radxmls which are used for those services . Provide same file path

srcPath: Provide source folder path. This field is optional (Tool will create radxmlListFile and SxmlListFile by itself from the srcPath)
Note that if srcPath is provided, radxmlListFile and sxmlListFile need not be provided

destpath: Provide the path where the files will be generated.

confirmStage : SINGLE_STAGE_UPDATE(Default Value should not be modified by developer)

```
## Service XML Updater
## 6.operation=SXML_UPDATER
## 6.sxmlListFile=D:\\ODT123\\TEST\\srcFile.TXT              -- List Of Absolute path of all sxml files in a text file
## 6.radxmlListFile=                                        -- A txt File containing the List of all radxmls.
## 6.xsdListFile=                                           -- A txt File containing the List of all xsds. This parameter is required only if NonExt Operations are Required.
##
## 6.srcPath=Z:\\EXEC\\FLEXCUBE_Kernel\\FCUBS_12.0.0\\MAIN   -- Src Path. Tool will create radxmlListFile and xsdListFile by itself from the srcPath if provided
##                                                             Note that if srcPath is provided,radxmlListFile and xsdListFile need not be provided
## 6.destpath=D:\\RADTOOL                                    -- destination Path
## 6.confirmStage=SINGLE_STAGE_UPDATE                         --
SINGLE_STAGE_UPDATE/UPDATE_FROM_STAT_FILES/STAT_FILE_GEN
##                                                            SINGLE_STAGE_UPDATE : Updation of service Xmls  in one step process.
##                                                              If any New function id is found mapped to a Service, it will not be updated.
##                                                            STAT_FILE_GEN : First Stage if Updation carried in 2 Steps. generates Stat Files in destPath.
##                                                               User can decide whether to update new FunctionId's to Service Xmls
##                                                            UPDATE_FROM_STAT_FILES : Second Step . Utility will update the Service Xml based on the confirmation
##                                                               information provided by User in the Stat files generated in previous Stage.Stat Files has to be placed in the destPath
#
```

Fig 4.3.3.6: SilentODT Service XML Updater Properties

Service XML Updater	
Input	**output**
sxmlListFile	updated Service XML's in destpath
radxmlListFile	
xsdListFile	
srcPath	
destpath	
confirmStage	

4.3.3.7 Service XML Refresh

Refresh Functionality allows us to upgrade the existing service xml to its later version keeping the sub version specific changes intact.

srcFileList: Prepare text file which contains absolute path of all Service xml, same file should be provide i.e. service xml which has to be refreshed

baseFileList: Prepare text file which contains absolute path of all base service xmls. Provide same file path .

For instance, for a custom development team ; all the latest Kernel files has to mentioned in baseFileList while the custom Service XMLs to be refreshed has to be mentioned in the srcFileList

srcRelType: Provide the release type of Source Radxmls list(KERNEL/CLUSTER/CUSTOM)

baseRelType: Provide the release type of base Radxmls list (KERNEL/CLUSTER/CUSTOM)

baseRelType should be at least one level below srcRelType for SOURCE Refresh.

For Instance, for a custom development team, srcRelType would be CUSTOM and baseRelType can be either KERNEL/CLUSTER depending on the base source type

destpath: Provide the path where the files will be generated.

```
## Service XML Refresh
## 7.operation=SXML_REFRESH
## 7.srcFileList=D:\\REFRESH\\src.txt     -- A txt File containing the List of all Sources sxmls. i.e sxmls which has
to be refreshed
## 7.baseFileList=D:\\REFRESH\\base.txt   -- A txt File containing the List of all base sxmls.
## 7.srcRelType=CUSTOM                  -- Release Type of Source sxmls; Either of KERNEL/CLUSTER/CUSTOM
## 7.baseRelType=KERNEL                 -- Release Type of Base sxmls;Either of
KERNEL/CLUSTER/CUSTOM##
##                                        Base Release Type should be atleast one level below Src Release type for
SOURCE Refresh.
## 7.destpath=D:\\RADTOOL
```

Fig 4.3.3.7: SilentODT Service XML Refresh Properties

Service XML Refresh	
operation	Refreshed Service XML's in destpath
srcFileList	
baseFileList	
srcRelType	
baseRelType	
destpath	

4.3.3 GW_CONFIG.properties

This properties files parameters are used for generation gateway web service properties files. This configuration file is optional. Provide only if Gateway Web service property files (**GW_WS_PROP_FILES**) is being generated as part of Service XML Bulk generation operation

EJB_APP_NAME: Provide Name of the deployed EJB Application

EJB_APP_SERVER: Provide Application server name in which ear deployed

EJB_JNDI_NAME: *EJB JNDI Name* is the reference name of the ejb by which the ejb has

been deployed

EJB_SERVER_URL: Application server IP Address & port where the EJB application is deployed.

EJB_SERVER_USERNAME: User Name of the application server where the EJB application is deployed.

EJB_SERVER_PASSWORD: Password of the application server where the EJB application is deployed

GW_WS_LOGGER_PROP_FILE_PATH: Provide the Location of Logger Property File path in the server where web service is to be deployed. Provide path including the file name.
Example.: D:/Kernel11.1/GW_WS/config/gw_ws_logger.properties(Windows path)
/oraint1/kernel//Gateway/GWWS/config/gw_ws_logger.properties (Linux or UNIX)

GW_WS_LOGGER_FILE_PATH: Provide the location where debug files will be written
Example : D:/Kernel11.1/GW_WS/log/ (Windows)
/oraint1/kernel/FC120INS_DEBUG/Gateway/GWWS/log (Linux or Unix)

GW_WS_PROP_FILE_PATH: Path where property file is placed in the server. This will be referred in web.xml of web service property file
Example: D:/Kernel11.1/GW_WS/prop/ (Windows)
/oraint1/kernel/FC120INS_DEBUG/Gateway/GWWS/prop (Linux or UNIX)

```
######GW_EJB Properties######Used For Creation GW Property File ###############
#############################################################################
#EJB_APP_NAME - Name of the deployed EJB Application
EJB_APP_NAME=GWEJB
# EJB_APP_SERVER(WEBLOGIC,WEBSPHERE)
EJB_APP_SERVER=WEBLOGIC
#Reference name of the EJB by which it has been deployed. This should be '<EJB_APPLICATION_NAME>/ejb/GW_EJB_Bean'
EJB_JNDI_NAME=GWEJB/ejb/GW_EJB_Bean

EJB_SERVER_URL=http://localhost:7010
EJB_SERVER_USERNAME=weblogic
EJB_SERVER_PASSWORD=weblogic1

##Location of Logger Property File path in the server where Webservice is to be deployed.
##Provide Path Including File Name
GW_WS_LOGGER_PROP_FILE_PATH=/home/ora11gas/Gateway11.0/GW_WS/config/gw_ws_logger.properties
##Location where Debug Files will be written
GW_WS_LOGGER_FILE_PATH=/home/ora11gas/Gateway11.0/GW_WS/log
#Location of Gateway Property File. If not provided ; assumed to be same as of Logger property File Path
GW_WS_PROP_FILE_PATH=/home/ora11gas/Gateway11.0/GW_WS/prop

XSD_PATH=/home/ora11gas/Gateway11.0/GW_WS/XSD
```

Fig 4.3.4.1: SilentODT GW Config Properties

GW_CONFIG	
EJB_APP_NAME	Provide Name of the deployed EJB Application
EJB_APP_SERVER	Provide Name of the deployed EJB Application

EJB_JNDI_NAME	*EJB JNDI Name* is the reference name of the ejb by which the ejb has been deployed
EJB_SERVER_URL	Application server IP Address & port where the EJB application is deployed.
EJB_SERVER_USERNAME	User Name of the application server where the EJB application is deployed.
EJB_SERVER_PASSWORD	Password of the application server where the EJB application is deployed.
GW_WS_LOGGER_PROP_FILE_PATH	Provide the Location of Logger Property File path in the server where web service is to be deployed. Provide path including the file name.
GW_WS_LOGGER_FILE_PATH	Provide the location where debug files will be written
GW_WS_PROP_FILE_PATH	Path where property file is placed in the server. This will be referred in web.xml of web service property file.
XSD_PATH	XSD Path.

4.4 Generation of Web service Artifacts through SilentOdtUtility

To generate web service artifacts for a service, configure the property files of the utility as explained in the previous section.

Following operations needs to be configured in OdtOperations.properties in respective sequence

1) LOGIN
2) SETRELEASE
3) SXML_BULKGENERATION

Provide following values for gen parameter of SXML_BULKGENERATION
IMPL_FILE, CONFIG_FILES, WSDL_FILE, XSD_FILES, GW_WS_PROP_FILES, ANT_BUILD

Double click the batch file silentOdt.bat/sh present in the ODT source.

.

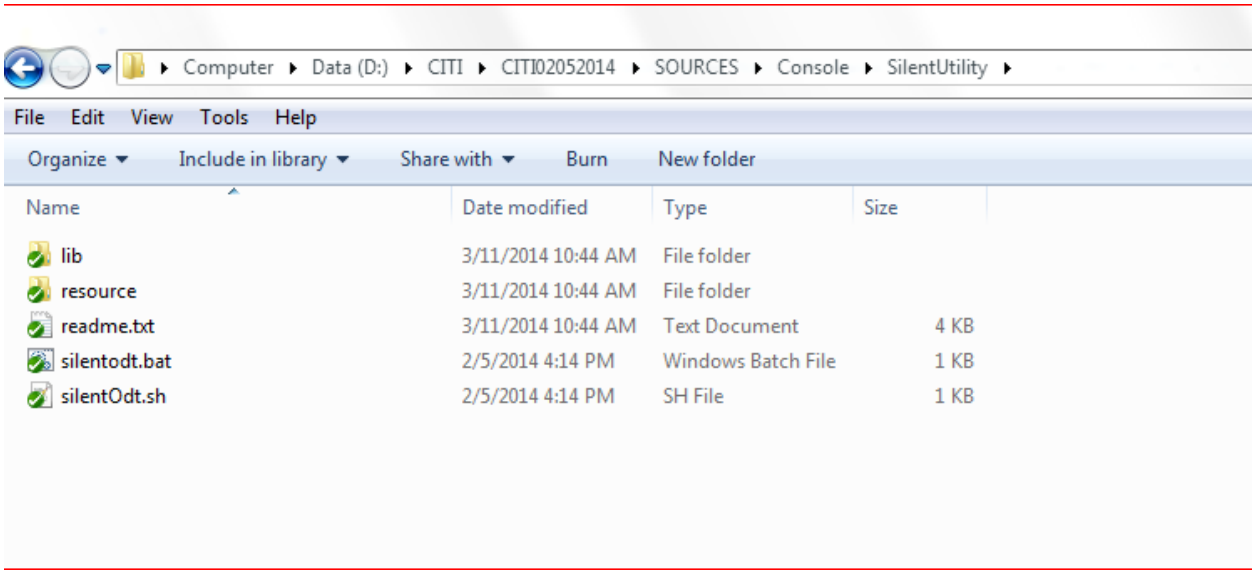

Fig 4.4.1: SilentODT Sources

This displays the screen as follows.

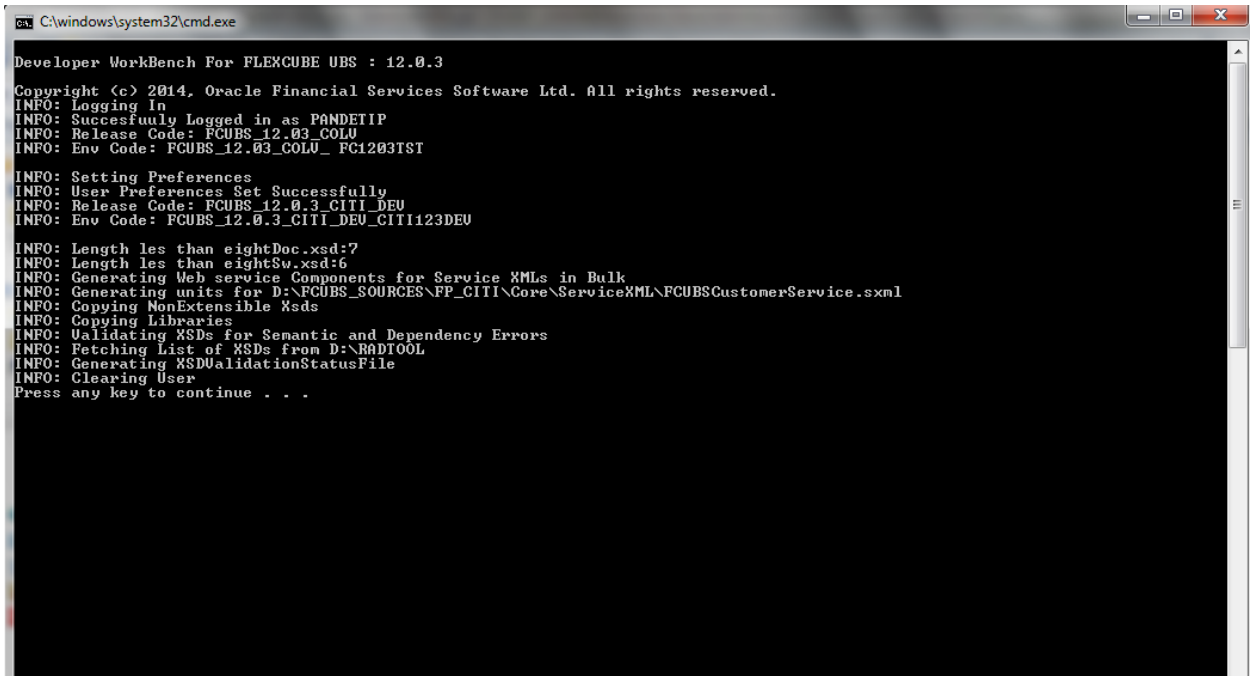

Fig 4.4.2: SilentODT Running in Command Prompt

After successful built operation, The Following Service artifacts files will be generated in the destination directory specified

Files	Description
<Service Name>Src*Impl.java	IMPL files for service
<Service Name>WSDL*.wsdl	WSDL files for service
<Service Name>Config*.xml	Config files
<Service Name>XSD*.xsd	Service specific xsd's
<Service Name>Common*.xsd's	Common XSD's (call forms) part of service
<Service Name>\<Service Name>\META-INF\application.xml <Service Name>\<Service Name>\META-INF\MANIFEST.MF	Config XML's for building the Web service
<Service Name>\<Service Name>\commons-codec-1.2.jar	Utility Jar for building the web service
<Service Name>\<Service Name>\wscommon.jar	Utility Jar for building the web service
Sample Ant file	For building service ear file (Can be modified by Dev team as per Folder structure)

Fig 4.4.3: SilentODT Generated Files Tree Structure

4.4.1 Log Files

- **ServiceGenerationStatus.csv**

  Generation status will be saved in above mentioned file.
  This will be generated in the destination path

- **XSDValidationErrors.csv**

  XSD Validation errors, if any, will be saved in above mentioned file.
  This will be generated in the destination path

- **Utility Log File**

  Log File of the utility would be generated in the path configured in SilentOdt.properties. This can be used in case of any troubleshooting

Proceed only if status is Success for all services in ServiceGenerationStatus.csv and XSDValidationErrors.csv is not generated

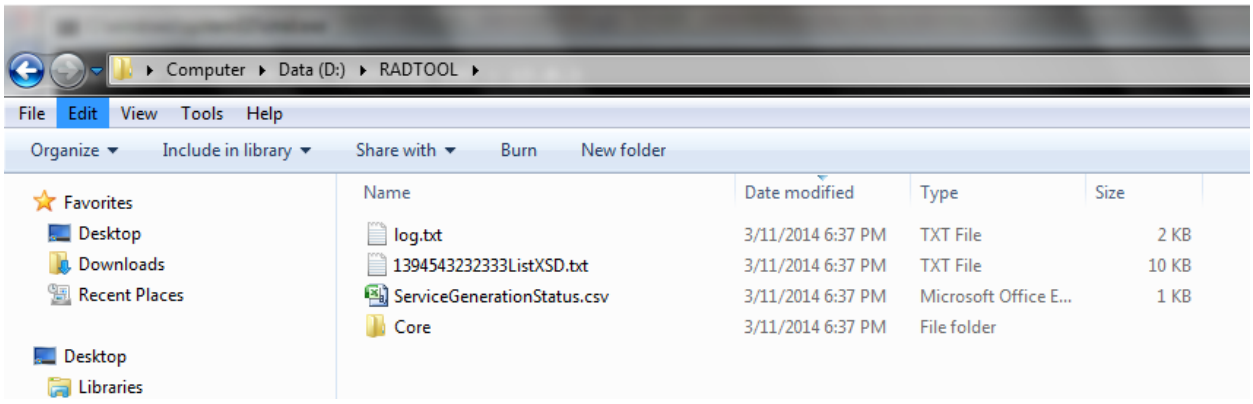

Fig 4.4.1.1: SilentODT Generated log Files.

4.4.2 Ant Build Scripts

Tool will generate the sample ant scripts for weblogic and web sphere application server.
Developer can write ant script based on sample ant script or same ant file can be used for building ear file .
WLANTBuild.xml – Ant script for web logic server
WASAntBuild.xml – Ant Script for web sphere server

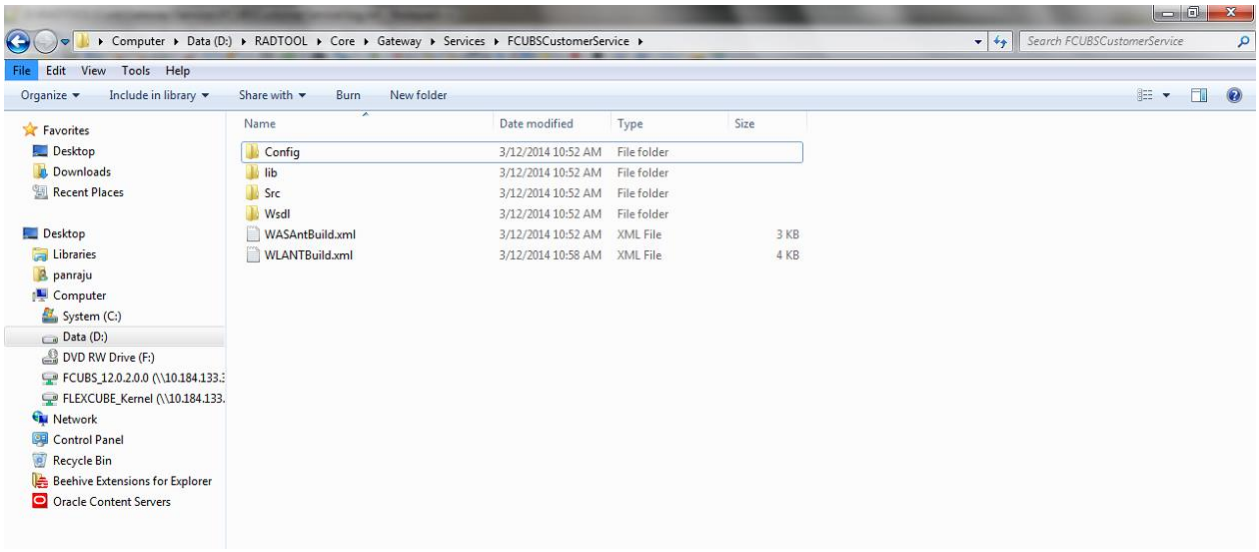

Fig 4.4.1.2: SilentODT Generated Files Ant Build Files.

4.4.3 Gateway Property Files

Gateway property files would be generated in GW_WS folder inside destination folder. .Following files would be generated.

- gw_ws_logger.properties
- GW_WS_Prop.properties
-

Development Workbench Service XML Development
[July] [2020]
Version 14.4.0.0.0

Oracle Financial Services Software Limited
Oracle Park
Off Western Express Highway
Goregaon (East)
Mumbai, Maharashtra 400 063
India

Worldwide Inquiries:
Phone: +91 22 6718 3000
Fax:+91 22 6718 3001
www.oracle.com/financialservices/