Table of Contents

1. Preface ........................................................................................................................................4
   1.1 Intended Audience ............................................................................................................. 4
   1.2 Documentation Accessibility .......................................................................................... 4
   1.3 Access to Oracle Support ............................................................................................... 4
   1.4 Structure ........................................................................................................................... 4
   1.5 Related Information Sources ........................................................................................... 4

2. Introduction .................................................................................................................................5
   2.1 Overview (OBAPIs -FCUBS Integration) .......................................................................... 5
   2.2 OBAPIs Integration Model ............................................................................................... 5
   2.3 OBAPIs Adapter Project Model ........................................................................................ 6

3. FCUBS Adapter Implementation ...............................................................................................7
   3.1 Guideline to Implement Adapter ..................................................................................... 7
   3.2 Inquiry Operation on FCUBS ............................................................................................. 7
   3.3 Transactional operation on FCUBS .................................................................................... 9
       3.3.1 Create FCUBS SOAP Client Instance ...................................................................... 9
       3.3.2 Get FCUBSHeader Values ....................................................................................... 9

4. OBAPIs -FCUBS Configuration/ Installation .............................................................................12
   4.1 Server Setup ...................................................................................................................... 13
       4.1.1 Create Data source with JNDI name as defined in config/jdbc.properties file. .... 13
       4.1.2 Server Arguments .................................................................................................... 13
   4.2 Deployment .......................................................................................................................... 13

5. Configuration for Integration of FCUBS Interaction with OBAPIs Mailbox .................................14
6. Configuration for Attachments in OBDX Mailbox or Interaction Module .................................15
1. Preface

1.1 Intended Audience
This document is intended for the following audience:
- Customers
- Partners

1.2 Documentation Accessibility
For information about Oracle’s commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

1.3 Access to Oracle Support
Oracle customers have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

1.4 Structure
This manual is organized into the following categories:
Preface gives information on the intended audience. It also describes the overall structure of the User Manual.
The subsequent chapters describes following details
- Introduction
- Adapter Implementation
- Configuration/ Installation

1.5 Related Information Sources
For more information on Oracle Banking APIs Release 19.1.0.0.0, refer to the following documents:
- Oracle Banking APIs Installation Manuals
2. **Introduction**

2.1 **Overview (OBAPIs - FCUBS Integration)**

This document is intended to outline the integration of OBAPIs (Oracle Banking APIs) with FCUBS (Oracle FLEXCUBE Universal Banking).

OBAPIs is the banking solution platform that enables single-view of a customer’s entire banking world.

It ensures that the program is closely aligned to the business strategies identified and provides accelerated path towards realizing business value.

It provides the prerequisite operations to be performed on the customer banking world, which is exposed in the form of REST web service API’s to decouple presentation and service layer. OBAPIs needs to be integrated with one of core banking systems to retrieve the customer banking details and execute the set of operations on the core banking system as per the service requirement.

OBAPIs has adapter layer to integrate with core banking operations. It provides a set of adapter specifications (Java Interfaces) which can be implemented for host specific service invocation. Adapter layer decouples the services from core banking operation. Any number of core banking system can be integrated with OBAPIs by implementing adapter classes, Adapter layer is responsible for mapping service request/response with host specific request and response.

This document exhibits the integration between core system & OBAPIs including the basic attributes involved in integration process.

2.2 **OBAPIs Integration Model**

![Diagram showing OBAPIs Banking, Integration Layer (Adapter), and Core Banking System (i.e. FCUBS / OBP)]
2.3 OBAPIs Adapter Project Model

Above Integration model depicts the OBAPIs adapter integration with core banking system. Integration can be accomplished by concrete adapter implementations.

Following model illustrates OBAPIs adapter specifications and its concrete adapter implementation relationship project model.

In above model, project com.ofss.digx.extxface contains all the adapter specifications (Java Interface) for external facing adapters. The concrete implementation classes of the adapter interfaces will reside in the host specific adapter implementation project.

Example: To integrate with FCUBS 12.4 core banking system, adapter interfaces should be implemented for host specific service invocation and concrete implementation adapter classes will reside in com.ofss.digx.extxface.ubs124.impl project.
3. **FCUBS Adapter Implementation**

To process the request with FCUBS core banking system, OBAPIs has set of adapter specifications which would be implemented for host specific services invocation. Concrete implementation adapter classes need to be created by implementing respective adapters interface and interface defining methods should be implemented.

Concrete adapter defining methods may have two types of operation:

- Inquiry Operation
- Transactional Operation

Inquiry operation is process to interchange data between OBAPIs and FCUBS core banking system without altering customer banking state. It queries on the host system to fetch needed information required for tracking, summarizing the details or processing the transactional operation.

Example: Fetch Customer detail, Accounts detail.

Transactional operation will request to host system for altering/creating end user banking states.

Example: Payments, Account Opening.

### 3.1 Guideline to Implement Adapter

1. Adapter implementation class must implement the respective interface and provide implementations for all methods defined in the interface.

2. Any exception while invoking core banking services must be translated into OBAPIs exception (com.ofss.digx.infra.exceptions.Exception) and host error code(s) should be mapped to an OBAPIs error code.

### 3.2 Inquiry Operation on FCUBS

To inquire data on core banking system, FCUBS provides a set of database view/synonyms object to inquire the required information. A connector schema is required which hosts the required views, synonyms, functions and procedures for querying data in FCUBS. OBAPIs relies on a middleware API packaged as "com.ofss.extsystem.ubs" which provides host adapters that interact with the connector schema to fetch the required information. To invoke these host adapters, the static Java method com.ofss.extsystem.ubs.business.extsystems.HostAdapterManager.processRequest(HostRequestDTO) needs to be invoked from within OBAPIs adapter implementation. The HostRequestDTO class must be provided with the required request information.

All the FCUBS middleware adapters are configured with unique a request id in the database table MSTHOSTINTERFACE. HostAdapterManager identifies the adapter class for given request id configured in MSTHOSTINTERFACE table and invoke the processRequest() method of respective adapter class.

Following steps are required to invoke the host adapter:

1. Create the Request DTO object and fill required fields.

2. Build host request by calling com.ofss.extsystem.ubs.business.extsystems.HostAdapterHelper.buildHostRequest(RequestDTO) method.

3. Invoke HostAdapterManager.processRequest(HostRequestDTO) with the HostRequestDTO created by invoking the method in step 2.
Reference:

```java
HostRequestDTO hostRequest = null;
HostResponseDTO hostResponse = null;

AccountDetailsRequest accountDetailsRequest = new AccountDetailsRequest();
    accountDetailsRequest.userContext = new UserContextDTO();
    accountDetailsRequest.userContext.idEntity = "B001";
    accountDetailsRequest.userContext.idTxn = "PAR";
    accountDetailsRequest.userContext.idRequest = "PARTYACCOUNTREL";
    accountDetailsRequest.userContext.serviceVersion = 0;
    accountDetailsRequest.userContext.refIdEntity = "B001";
    accountDetailsRequest.userContext.userType = "EN1";
    accountDetailsRequest.account = new AccountNoInputDTO();
    accountDetailsRequest.account.idCustomer = partyId;
    accountDetailsRequest.account.acctType = accountType;

hostRequest = HostAdapterHelper.buildHostRequest(accountDetailsRequest);

try {
    HostResponse = HostAdapterManager.processRequest(hostRequest);

} catch (java.lang.Exception e) {
    logger.log(Level.SEVERE, formatter.formatMessage("Exception"));
```
3.3 Transactional operation on FCUBS

For processing transactional operations, FCUBS core banking system exposes SOAP web services. To invoke the SOAP web services, SOAP client stubs need to be generated with the help of WSDL exposed by FCUBS. The thus generated client stubs should be included in classpath and configure database entries in `DIGX_FW_CONFIG_OUT_WS_CFG_B` table. Create instance of FCUBS Soap client service using `com.ofss.fc.infra.ws.JAXWSFactory.createServiceStub(String, String)` and invoke the respective methods with required request payload.

```
com.ofss.fc.infra.ws.JAXWSFactory.createServiceStub(String, String) method will instantiate the respective SOAP client services by passing the service and operation name. It uses the `DIGX_FW_CONFIG_OUT_WS_CFG_B` database table to retrieve the SOAP WSDL URL, End point service name and proxy services name for instantiating the client services, so all the relevant database entries should be configured in database table for each service operations.
```

Following step to invoke the soap service:

1. Generate SOAP client stubs from WSDL file.
2. Insert the database entry in `DIGX_FW_CONFIG_OUT_WS_CFG_B` table for the respective service and operation.
3. Get the SOAP client services instance by invoking `com.ofss.fc.infra.ws.JAXWSFactory.createServiceStub(String, String)` method.
4. Create request header instance and fill the mandatory fields.
5. Create request body instance and fill the essential detail.
6. Invoke the respective SOAP method by passing the request parameter containing request header and body.

### 3.3.1 Create FCUBS SOAP Client Instance

Create an instance of soap client services by invoking `JAXWSFactory.createServiceStub` method by passing service id and operation name arguments to identify the service.

**Important:** A database entry must be available for respective service and operation in `DIGX_FW_CONFIG_OUT_WS_CFG_B` database table.

```
(FCUBSAccServiceSEI)JAXWSFactory.createServiceStub(WebserviceConstants.FCUBS_ACCOUNT_SERVICE_SPI, WebserviceConstants.CREATE_TD_ACCOUNT);
```

### 3.3.2 Get FCUBSHeader Values

Each and every FCUBS soap request requires header object of class FCUBSHeaderType.
The class `com.ofss.digx.ubs.adapter.impl.RequestHeader`, provides all properties to set in FCUBS SOAP request header object.

A `RequestHeader` object should be instantiated by invoking `AbstractAdapterHelper.getRequestHeader` method and set all the required properties in FCUBS soap request header object.

```
com.ofss.digx.FCUBS.adapter.impl.AbstractAdapterHelper.getRequestHeader(String, String, String)
```

Create the Adapter helper instance and retrieve Request Header Object.

```
AbstractAdapterHelper helper = AbstractAdapterHelper.getInstance();
RequestHeader header = helper.getRequestHeader("IA", "FCUBSAccService", "ModifyIATDCustAcc");
```

Instantiate the request header and fill the essential details:

```
MODIFYTDCUSTACCSFSFSREQ createTDRequest = new MODIFYTDCUSTACCSFSFSREQ();

FCUBSHEA DTETYPE fcUBSHeader Type = new FCUBSHEADERType();
fcUBSHeader Type. setUSERID(header.getUserid());
fcUBSHeader Type. setCORRELID(header.getCorrelid());
fcUBSHeader Type. setMSGID(header.getMsgid());
fcUBSHeader Type. setDESTINATION(header.getDestination());
fcUBSHeader Type. setMSGSTAT(MsgStatType. fromValue(header.getMsgstat()));
fcUBSHeader Type. setSOURCE(header.getSource());
fcUBSHeader Type. setBRANCH(helper.getHostBranchId(termDeposit.getTermDepositAccountId().getValue()));
fcUBSHeader Type. setUBSCOMP(UBSCOMPType. fromValue(header.getUbscomp()));
fcUBSHeader Type. setMODULEID(header.getModuleid());
fcUBSHeader Type. setSERVICE(header.getService());
fcUBSHeader Type. setOPERATION(header.getOperation());
fcUBSHeader Type. setSOURCEOPERATION(header.getSourceoperation());
createTDRequest. setFCUBSHEADER(fcUBSHeader Type);
```
Invoke the FCUBS SOAP service

```java
FCUBSAccServiceSEI clientProcess = (FCUBSAccServiceSEI)
JAXWSFactory.createServiceStub(WebserviceConstants.FCUBS_ACCOUNT_SERVICE_SPI,
WebserviceConstants.CREATE_TD_ACCOUNT);

MODIFYTDCUSTACCFSFSRESPONSE =
clientProcess.modifyTDCustAccFS(createTDRequest);
```
4. **OBAPIs -FCUBS Configuration/ Installation**

1. To inquire the banking information, FCUBS provides a set of database views/synonyms. These are created on a connector schema which is created at the time of installation of the product.

A datasource must be created on the application server where the application has been deployed. The name of the datasource must be specified in the jdbc.properties file for the property name FCON.A1.JNDI.NAME, FCON.AP.JNDI.NAME and FCON.B1A1.JNDI.NAME.

Below properties should be define in config/jdbc.properties file

```
FCAT.WEBSERVER.ID=ZZ
FCAT.APPSERVER.ID=ZZ
FCAT.ROUTER.DAEMON.NAME=ROUTER
DISPLAY.MESSAGE.ID=N
FCAT.LDB.DATABASE.NAME=ORACLE

FCON.A1.LDB.DRIVER=oracle.jdbc.driver.OracleDriver
FCON.A1.LDB.URL=%%DB_CONNECT_STRING%%
FCON.A1.JNDI.NAME=B1A1
FNDI.A1.ABCD=

FCON.AP.LDB.DRIVER=oracle.jdbc.driver.OracleDriver
FCON.AP.LDB.URL=%%DB_CONNECT_STRING%%
FCON.AP.JNDI.NAME=B1A1
FNDI.AP.ABCD=

B001.A1=B1A1
FNDI.B1A1.ABCD=
FCON.B1A1.JNDI.NAME=B1A1
FCON.B1A1.LDB.DRIVER=oracle.jdbc.driver.OracleDriver
FCON.B1A1.LDB.URL=%%DB_CONNECT_STRING%%
```
4.1 Server Setup

4.1.1 Create Data source with JNDI name as defined in config/jdbc.properties file.

Please note that in case of multi entity scenario, the JNDI name should be in the format `<ENTITY_ID>_B1A1`. For example, for entity identifier as 'UBDX_BU', the JNDI name will be `OBADX_BU_B1A1`.

4.1.2 Server Arguments

-Dfcat.jvm.id=1

4.2 Deployment

We have FCUBS deployable ear containing all the jar files required to invoke the FCUBS adapter. Following deployable should be deployed on OBAPIs server:

`obapis.extsystem.domain.ear`

It contains all the related libraries required to process FCUBS system request.
5. Configuration for Integration of FCUBS Interaction with OBAPIs Mailbox

Out of the box installation OBAPIs provided with the mailbox to interact within OBAPIs i.e. Back Office user of the banks in the OBAPIs will be able to access the mail send by the customer.

If the OBAPIs Mails by the customer need to be integrated with the FCUBS Interaction module then the below steps should be taken care of.

Assumption:- Installation of OBAPIs has been completed and configuration of all the entity requires in the system has been completed.

Execute below scripts in the OBAPIs Admin Schema to configure entity specific integration of mailbox with interaction module in FCUBS.

Insert into DIGX_FW_CONFIG_ALL_O (PROP_ID,PREFERENCE_NAME,PROP_VALUE,DETERMINANT_VALUE,CREATED_BY,CREATION_DATE,LAST_UPDATED_BY,LAST_UPDATED_DATE) values
('MAILBOX_PROCESSOR','MailboxProcessor','com.ofss.digx.app.collaboration.service.mailbox.message.mail.processor.RemoteMailboxProcessor','OB_DX_BU1','superadmin',sysdate,'superadmin',sysdate);

Insert into DIGX_FW_CONFIG_ALL_O (PROP_ID,PREFERENCE_NAME,PROP_VALUE,DETERMINANT_VALUE,CREATED_BY,CREATION_DATE,LAST_UPDATED_BY,LAST_UPDATED_DATE) values
('MAIL_REPOSITORY_ADAPTER','RepositoryAdapterFactories','com.ofss.digx.domain.collaboration.entity.mailbox.message.mail.repository.adapter.RemoteMailRepositoryAdapter,com.ofss.digx.domain.collaboration.entity.mailbox.message.mail.repository.adapter.LocalMailRepositoryAdapter','OB_DX_BU1','superadmin',sysdate,'superadmin',sysdate);

Highlighted values can be varied based on the entity configuration require to be integrated with FCUBS interaction module.
6. **Configuration for Attachments in OBDX Mailbox or Interaction Module**

By default the mailbox attachment will be integrated to OIPM. If the Bank wants to change this then below are the configuration steps.

1) If bank want to use local database to save uploaded document i.e. other than OIPM then below script need to be executed on OBDX Admin schema. This will point content service to the local data base for mailbox attachment specifically.

   ```sql
   UPDATE DIGX_FW_CONFIG_ALL_B SET PROP_VALUE = 'com.ofss.digx.domain.content.entity.repository.adapter.LocalContentRepositoryAdapter' WHERE PROP_ID = 'IM_CONTENT_REPOSITORY_ADAPTER'
   
   2) If bank want to use OIPM server to manage uploaded document then below script need to be executed on OBDX Admin schema. This will point content service to the OIPM server for mailbox attachment specifically.

   ```sql
   UPDATE DIGX_FW_CONFIG_ALL_B SET PROP_VALUE = 'com.ofss.digx.domain.content.entity.repository.adapter.RemoteContentRepositoryAdapter' WHERE PROP_ID = 'IM_CONTENT_REPOSITORY_ADAPTER'
   ```