

Oracle® Revenue Management and Billing

Version 2.7.0.0.0

File Upload Interface (FUI) - Quick Reference Guide

Revision 1.0

E98483-01

July, 2018

Oracle Revenue Management and Billing File Upload Interface (FUI) - Quick Reference Guide

E98483-01

Copyright Notice

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Trademark Notice

Oracle and Java are registered trademarks of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

License Restrictions Warranty/Consequential Damages Disclaimer

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure, and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or de-compilation of this software, unless required by law for interoperability, is prohibited.

Warranty Disclaimer

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

Restricted Rights Notice

If this software or related documentation is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS

Oracle programs, including any operating system, integrated software, any programs installed on the hardware and/or documentation delivered to U.S. Government end users are “commercial computer software” pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware and/or documentation shall be subject to license terms and restrictions applicable to the programs. No other rights are granted to the U.S. Government.

Hazardous Applications Notice

This software is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure the safe use of this software. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software in dangerous applications.

Third Party Content, Products, and Services Disclaimer

This software and documentation may provide access to or information on content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third party content, products, or services.

Preface

About This Document

This document provides a detail explanation of ORMB approach for Data Conversion and integration. It describes parameters related to File Upload Interface Master Configuration and also explains how to perform important tasks using File Upload Interface. This Reference Guide supplements the information provided in *File Upload Interface User Guide* and *File Upload Interface Version 2.6.0.1.0 Batch Execution Guide*.

Intended Audience

This document is intended for the following audience:

- End-users
- Implementation Team
- Consulting Team
- Development Team

Organization of the Document

The information in this document is organized into the following chapters:

Section No.	Section Name	Description
Section 1	ORMB Approach for Data Conversion on Premise	Explains the ORMB Approach for Data Conversion on premise. It also provides the steps to be followed for converting data using ORMB.
Section 2	File Upload Interface	Provides an overview of File Upload Interface.
Section 3	ORMB Approach for Data Conversion using File Upload Configuration	Explains the ORMB Approach for Data Conversion using File Upload Interface. It also provides the steps to be followed for converting data using File Upload Interface.
Section 4	File Upload Interface Master Configuration	Describes important parameters related to File Upload Interface Master Configuration.
Section 5	Creating File Request Type	Lists the steps to create file request type.
Section 6	Working with File Request Type	Explains different fields in File Request Type zone and tasks which you can perform using File Upload Interface.
Section 7	Updating records marked with 'Error' or 'Pending' status	Lists steps to update record status using File Upload Dashboard.
Section 8	Transforming Data	Identifies important concepts related to transforming data.

Related Documents

You can refer to the following documents for more information:

Document	Description
<i>Oracle Revenue Management and Billing Banking User Guide</i>	Lists and describes various banking features in Oracle Revenue Management and Billing. It also describes all screens related to these features and explains how to perform various tasks in the application.
<i>File Upload Interface User Guide</i>	Helps you configure File Upload Interface.
<i>File Upload Interface Version 2.6.0.1.0 Batch Execution Guide</i>	Explains the batches to be executed while performing various tasks in File Upload Interface.

Contents

1.	ORMB Approach for Data Conversion on Premise	1
1.1	Prerequisites.....	1
1.2	Steps for ORMB Conversion	1
1.3	ORMB Conversion Flow Diagram	2
1.4	Disadvantages.....	2
1.5	Cloud Guideline	2
2.	File Upload Interface	3
3.	ORMB Approach for Data Conversion and Integration using File Upload Interface	4
3.1	Prerequisites.....	4
3.2	Approaches for ORMB Conversion and Integration Process.....	4
3.2.1	ORMB Conversion and Integration without File Transformation	4
3.2.2	ORMB Conversion and Integration with File Transformation	5
4.	File Upload Interface Master Configuration	6
4.1	Parameters related to File Upload Interface Master Configuration	7
4.1.1	Validate Checksum.....	7
4.1.2	Validate Duplicate File Name.....	8
4.1.3	Audit Log.....	8
4.1.4	Archive File.....	8
4.1.5	Archive File Location	8
4.1.6	Archive Error File Location	9
4.1.7	File Encryption	10
4.1.8	File Decryption Algorithm	10
4.1.9	Cipher Type	10
4.1.10	Decryption Key.....	11
5.	Creating File Request Type	12
5.1	Creating File Request Type Without File Transformation	12
5.2	Creating File Request Type with File Transformation	14
6.	Working with File Request Type	16
6.1	Uploading and Processing Records using Single Batch Execution.....	16
6.2	File Extensions used for Reading Uploaded Files on SFTP server.....	17
6.3	Rollback all Processed Records in case of Single Record Failure.....	18
6.4	Validating Processing Details.....	19
6.5	Header and Footer Details in a File	20
6.6	Ignoring or Skipping Duplicate Records in a File	22
6.7	Marking the Default Status of Failed Records to “Retry” for Reprocessing.....	24
6.8	Overriding the Service Level Operation at Runtime.....	25

6.9	Skipping Service Execution for a Particular Record	26
6.10	Executing Dependent Service	27
6.11	Deferring Completion of Processed Request	28
6.12	Supporting Client Defined Date Format	29
6.13	Viewing To Do Entries for Failed Files	30
6.14	Post Processing Algorithm Support for File Request.....	32
7.	Updating Records marked with 'Error' or 'Pending' status	33
7.1	File Upload Dashboard	33
7.2	File Request Status Update Batch	34
8.	Transforming Data	35
8.1	Transforming File Record in Client Supported Format into ORMB Conform XML	35
8.2	Configuring a Default Value for a Particular Field	36
8.2.1	Applying Default Values Set in File Validation Algorithm to a Field	37
8.3	Mapping Data Line or Record in a File.....	38
8.4	Mapping File Sequence	41
8.5	Using Map Field XPath to Transform Records.....	42
8.6	Link Parent Service Output with Child Service Input.....	43
8.7	Validating Input Values.....	44

1. ORMB Approach for Data Conversion on Premise

Enterprise information systems comprise of a variety of data storage systems, which vary in complexity and in the ways they access internal data.

- **Shared Database** - All applications that you are integrating read data directly from the same database.
- **Maintain Data Copies** - Maintain copies of the application's database so that other applications can read the data (and potentially update it).
- **File Transfer** - Make the data available by transporting a file that is an extract from the application's database so that other applications can load the data from the files.
- **Service Integration** - Real time integration using SOAP/REST services.

ORMB uses file transfer approach for data conversion and data integration.

1.1 Prerequisites

To convert data using ORMB on premise, you should have:

1. Conversion tool kit.
2. Pre-staging, staging and production schema.
3. Stored procedures to map (transform) and transfer data from pre-staging stage to staging stage and staging stage to production stage.

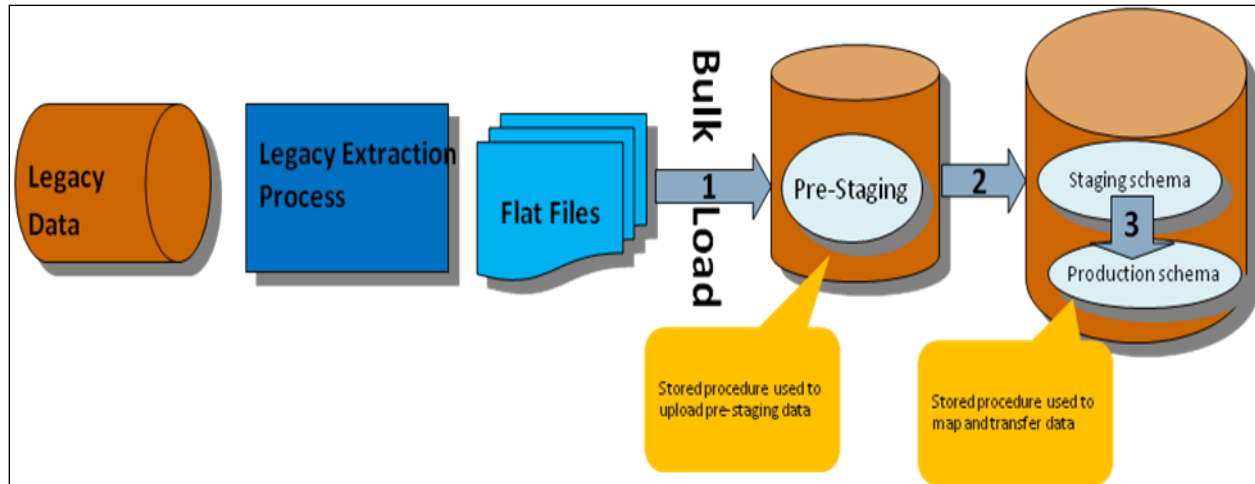
1.2 Steps for ORMB Conversion

To convert data, you need to follow the below steps:

1. The legacy data has to be manually relocated to a database for ConversionMapper to access the data before converting into ORMB table structure.
2. Create Pre-Staging schema.
3. Define tables to refer to a legacy tables external to ORMB. These tables which contain legacy data are defined as Input Tables in ConversionMapper.
4. A Bulk Load utility such as SQL*Loader in Oracle is typically used to copy the legacy files into the ORMB Pre-Staging schema.
5. Create Staging schema.
6. Instead of using the flat files directly, the Oracle tables (Input Tables) that represent the flat files are used to map to ORMB.
7. The legacy tables or "Input Tables" in ConversionMapper **MUST** exist on a database (staging schema), so that ConversionMapper can use SQL statements to access the tables.
8. The legacy tables can be defined as tables on the staging schema (STGADM) or as views (also on STGADM) of tables on another schema or database.
9. Stored Procedures are used to map, validate and transfer the data to staging schema.

1.3 ORMB Conversion Flow Diagram

The flow of data between the two systems is illustrated below:



1.4 Disadvantages

- Data transformation and loading is done using Stored procedures thereby resulting in duplication of data validation and business rules.
- Risk of missing data validation and business rules.

1.5 Cloud Guideline

Before deciding to migrate to cloud based frameworks, note that:

- Legacy systems do not have access to ORMB database.
- Stored procedures cannot be deployed on cloud database.

2. File Upload Interface

ORMB File Upload Interface will be used for data conversion and data integration on cloud. It provides ability to upload files to staging. It will provide way to map file records to ORMB services. ORMB file upload interface will invoke ORMB services for each of the record.

The ORMB file upload interface provides following benefits:

- Conversion tool kit not required.
- No Pre-staging schema required.
- No ORMB DB access required for any legacy system.
- Data load in ORMB system is done using existing ORMB service schemas thereby no duplication of data validation and business rules.
- No risk of missing data validation and business rules.
- Supports transformation for files in XML, JSON, Fixed Position, CSV, PSV and TSV formats.

Online system can be used to view the uploaded files and their corresponding processed details.

3. ORMB Approach for Data Conversion and Integration using File Upload Interface

This section lists the steps to be followed before using File Upload Interface. It also describes approaches you can take when working with ORMB File Upload Interface.

3.1 Prerequisites

To convert data using ORMB using File Upload Interface, you should:

1. Define File Upload Interface master configuration. For more information on how to define File Upload Interface master configuration, refer *File Upload Interface User Guide*.
2. Configure File Request Type with required mapping of existing ORMB Services Business Objects or Business Services or Service Script specific to Transaction Stage Upload service. You need to add date format in "FILE_UPLOAD_DATE_FORMATS" lookup to support client specific date inputs in Transaction Header.

3.2 Approaches for ORMB Conversion and Integration Process

To convert or integrate files in ORMB file upload interface, you can follow any of the below two approaches:

1. ORMB conversion and integration process without file transformation
2. ORMB conversion and integration process with file transformation

3.2.1 ORMB Conversion and Integration without File Transformation

To convert or integrate files without file transformation, you need to follow below steps:

1. Ensure that the files are in XML format and comply with ORMB service schema.

Note: File upload supports files in XML format only.

2. Publish service XMLs conforming to ORMB service schemas that are to be used for conversion.
3. If validations for Header, Footer or Checksum are required, then you need to write an algorithm deriving "FileValidationAlgorithmSpot".
4. If any preprocessing is required before invoking Business Object or Business Service or Service Script service, then you need to implement an algorithm deriving "FileRequestPreProcessingAlgorithmSpot".
5. Execute File Transform and Upload batch (C1-FTRAN). This will read flat file records and upload in file upload staging. For more information on C1-FTRAN batch, refer *File Upload Interface Version 2.6.0.1.0 Batch Execution Guide*.
6. Execute File Request Processing batch (C1-FREQP). This will read the records from staging and process those using Service configured in file request type. For more information on C1- FREQP batch, refer *File Upload Interface Version 2.6.0.1.0 Batch Execution Guide*.
7. File upload and processing steps can also be done by executing only C1-FTRAN batch. You are required to set "File Upload and Process" flag as true in File Request Type configuration.

8. Perform required fixes either in File Request Type Configuration or file details.

Note: During conversion process, conversion cycles can be run on UAT environment while the system is being tested on the production schema.

3.2.2 ORMB Conversion and Integration with File Transformation

To convert or integrate files with file transformation:

1. File upload supports files in XML, JSON, CSV, PSV, TSV or Fixed Position formats.
2. File record transformation algorithm can be implemented by deriving "FileRequestTransformationAlgorithmSpot".
3. File transformation will be done using Transformation Detail configuration in its file request type.
4. If validations for Header, Footer or Checksum are required, then implement an algorithm deriving "FileValidationAlgorithmSpot".
5. If any preprocessing is required before invoking Business Object or Business Service or Service Script service, then implement an algorithm deriving "FileRequestPreProcessingAlgorithmSpot".
6. Execute File Transform and Upload batch (C1-FTRAN). This will read the flat file records and transform those using "File Record Transformation" algorithm and upload in file upload staging.
7. Execute File Request Processing batch (C1-FREQP). This will read the records from staging and process those using Service configured in File Request Type.
8. File transform, upload and processing steps can also be done by executing only C1-FTRAN batch. You are required to set "File Upload and Process" flag as true in File Request Type configuration.
9. Perform required fixes either in File Request Type Configuration or file details.

Note: During conversion process, conversion cycles can be run on UAT environment while the system is being tested on the production schema.

4. File Upload Interface Master Configuration

This configuration is referred in file upload for file decryption, archival of file, and audit logging. To view and edit file upload configuration, you need to do the following:

1. From the **Admin** menu, select **M** and then click **Master Configuration**. The **Master Configuration window** appears.
2. Click the **Edit** (✎) icon corresponding to the **File Upload Interface Configuration**. The **Master Configuration window** appears.

The screenshot shows a web-based configuration window titled "Main". It contains a list of configuration items for "Master Configuration". The items are as follows:

Configuration Item	Value
Master Configuration	C1-FileUploadInterfaceConfig
Validate Checksum	<input type="checkbox"/>
Validate Duplicate File Name	<input type="checkbox"/>
Audit Log Required	<input type="checkbox"/>
Archive File	<input checked="" type="checkbox"/>
Archive File Location	INSTALL_DIR/UploadedFile
Archive Error File Location	INSTALL_DIR/ErrorFile
File Encryption Required	<input checked="" type="checkbox"/>
File Decryption Algorithm	C1-FRDA File Request Decryption Algorithm
Cipher Type	Advanced Encryption Standards
Decryption Key	DFGDFGDFGDFGDFGDFGD

Figure 1: File Upload Interface Configuration - Master Configuration

4.1 Parameters related to File Upload Interface Master Configuration

This section lists and describes following important parameters related to File Upload Interface Master Configuration.

1. Validate Checksum
2. Validate Duplicate File Name
3. Audit Log
4. Archive File
5. Archive File Location
6. Archive Error File Location
7. File Encryption
8. File Decryption Algorithm
9. Cipher Type
10. Decryption Key

4.1.1 Validate Checksum

This flag will decide whether to check the integrity of the file before staging file contents in ORMB system. The Validate Checksum parameter can be described as:

1. Field Type: Check box
2. Valid Values: True or False
3. Description:
 - If **True**, checksum validation will be always performed for all the uploaded files.
 - For every uploaded file, it is required to have the corresponding `<FILE_NAME>.checksum` file on SFTP server.
 - It is required to have **File validation algorithm** mapped with every **File Request Type**.
 - **File validation algorithm** is derived from **"FileHeaderValidationAlgorithmSpot"**.
 - Checksum validation logic is implemented in **File validation algorithm**. This algorithm is used for Header and Footer validations.
 - File Validation algorithm is derived from **"FileHeaderValidationAlgorithmSpot"**.
 - Sample **"FileHeaderValidationAlgorithm_Impl"** algorithm is provided with:
 - Header validation – Number of records are checked.
 - Checksum validation – Done using **"MD5"** algorithm type.
 - For checksum validation, if you want to use the default implementation in any specific **File validation algorithm**, then it can be done by invoking **"calculateChecksum(String fileString, String algoName)"** function in **"FileRequestProcessBusinessComponent_Impl"** business component.
 - If **False**, this flag skips this checksum validation.

Note: For uploaded file, it is not required to have corresponding `<FILE_NAME>.checksum` file.

4.1.2 Validate Duplicate File Name

This flag will be used to decide the required validation of duplicate file name before uploading a file. The Validate Duplicate File Name parameter can be described as:

1. Field Type: Check box
2. Valid Values: True or False
3. Description:
 - If **True**, file with same name which is already existing in staging, will not be uploaded.
 - If **False**, file with same name will be uploaded.

4.1.3 Audit Log

This flag will decide whether to log corresponding file request status transition after processing an individual file request. The Audit Log parameter can be described as:

1. Field Type: Check box
2. Valid Values: True or False
3. Description:
 - If **True**, status transitions for all the file request data will be maintained in ORMB system.
 - Logging will be done in “**CI_FILE_REQUEST_DTL_MSG**” table with “**LOG_ENTRY_TYPE_FLG**” value as “**F1ST**”.
 - “**CI_FILE_REQUEST_DTL_MSG**” table is also used for error logging with “**LOG_ENTRY_TYPE_FLG**” value as “**F1EX**”.
 - If **False**, status transitions for the file requests will not be performed.

4.1.4 Archive File

This flag decides whether to relocate the file to another location after processing on SFTP server. Here, location refers to the path mentioned in “Archive File Location” or “Archive Error File Location”. The Archive File parameter can be described as:

1. Field Type: Check box
2. Values: True or False
3. Description:
 - If **True**, files will be moved to another location. There are two scenarios:
 - If file is uploaded successfully, the files will be moved to defined archive file location.
 - If file upload fails, the files will be moved to defined archive error file location.
 - If **False**, the files will remain at the same location.

4.1.5 Archive File Location

It is used to specify the file path used for archiving the file. The successfully processed file will be moved to this location. The Archive File Location parameter can be described as:

1. Field Type: Input field
2. Values: It is free text

3. Description:

- If **True**, successfully uploaded files will be moved to defined archive file location.
- If **False**, the files will remain at the same location.
- File Location should always be a combination of logical path and relative path and prefixed with either **"INSTALL_DIR"** or **"SHARED_DIR"**. For example, if you want to define file location as **"/scratch/rmbbuild/sftpFile"**, then it will be **"INSTALL_DIR/ sftpFile"** where **"INSTALL_DIR"** value is defined as **"/scratch/rmbbuild"**
- **"INSTALL_DIR"** variable value is defined against **"spl.runtime.envIRON.SPLeBASE"** in **"spl.properties"** file.
- **"SHARED_DIR"** variable has a static value. It is a shared storage path mounted in cloud environment.
- The defined location will be always appended by the corresponding File Request Type.
 - **Archive Error File Location** - **INSTALL_DIR/FilesUploaded**
 - **INSTALL_DIR path** - **/scratch/rmbbuild**
 - If batch is executed for **"ADD_PERSON"** file request type, then files will be moved to **/scratch/rmbbuild/FilesUploaded/ADD_PERSON/**

4.1.6 Archive Error File Location

It is used to specify the file path used for archiving the error files. The files with errors will be moved to this location. The Archive Error File Location parameter can be described as:

1. Field Type: Input field

2. Values: It is free text

3. Description:

- If **True**, files that failed to upload will be moved to this defined error file location.
- If **False**, files will remain at the same location.
- Error File Location should always be a combination of logical path and relative path and prefixed with either **"INSTALL_DIR"** or **"SHARED_DIR"**. For example, if you want to define file location as **"/scratch/rmbbuild/sftpFile"**, then it will be **"INSTALL_DIR/ sftpFile"** where **"INSTALL_DIR"** value is defined as **"/scratch/rmbbuild"**
- **"INSTALL_DIR"** variable value is defined against **"spl.runtime.envIRON.SPLeBASE"** in **"spl.properties"** file.
- **"SHARED_DIR"** variable has a static value. It is a shared storage path mounted in cloud environment.
- The defined location will be always appended by the corresponding File Request Type.
 - **Archive Error File Location** - **INSTALL_DIR/ErrorFiles**
 - **INSTALL_DIR path** - **/scratch/rmbbuild**
 - If batch is executed for **"ADD_PERSON"** file request type then files will be moved to **/scratch/rmbbuild/ErrorFiles/ADD_PERSON/**

4.1.7 File Encryption

This flag will decide whether to first decrypt and then extract the files on SFTP server. The File Encryption parameter can be described as:

1. Field Type: Check box
2. Values: True or False
3. Description:
 - If **True**, files on SFTP server will be decrypted using File Decryption Algorithm, then extracted and processed to upload the file data in ORMB staging.
 - It is **MUST** to have **File Decryption Algorithm** for using the file encryption parameter. The algorithm must be derived from "**FileRequestDecryptionAlgorithmSpot**". This algorithm will have the implementation commands for getting the required decrypted file.
 - A sample "**FileRequestDecryptionAlgorithm_Impl**" algorithm is provided with ORMB application. The algorithm,
 - i. Gets the decryption key for defined "**com.oracle.ouaf.system.keystore.file**" alias in "**ouaf_keystore**" file.
 - ii. Gets the decrypted file using decryption key.

4.1.8 File Decryption Algorithm

It defines the algorithm to be used for decryption of a file. The File Decryption Algorithm parameter can be described as:

1. Field Type: Search field
2. Values: Algorithms deriving "**FileRequestDecryptionAlgorithmSpot**".
3. Description:
 - This algorithm must have an implementation for decrypting the input File.
 - It gets the secret key stored against "**com.oracle.ouaf.system.keystore.passwordFileName**" alias and uses secret key to access "**ouaf_keystore**" file.
 - It gets the decryption key stored against "**com.oracle.ouaf.system.keystore.file**" alias in "**ouaf_keystore**" file.
 - The decryption key decrypts and returns the file string.

4.1.9 Cipher Type

This is an algorithm type used to get the decryption key. The Cipher Type parameter can be described as:

1. Field Type: Dropdown
2. Values: The valid values are:
 - Advanced Encryption Standards
 - Data Encryption Standard
 - Password Based Encryption
 - RAS
3. Description: This lists decryption algorithm types.

4.1.10 Decryption Key

This key will be used to decrypt the uploaded encrypted files on SFTP server. The Decryption Key parameter can be described as:

1. Field Type: Input
2. Values: Free Text
3. Description:
 - This holds a decryption key which is further used to get the decrypted file data.
 - This field value is not stored in database.
 - It will be stored against “com.oracle.ouaf.system.keystore.file” alias in “ouaf_keystore” file.

5. Creating File Request Type

File request type is a configuration which will allow you to upload files in any format and transform the files in ORMB compliant format.

When creating a file request type, you have two options:

1. Create File Request Type without file transformation
2. Create File Request Type with file transformation

5.1 Creating File Request Type Without File Transformation

To create file request type without file transformation:

1. From the **Admin** menu, select **F** and then click **File Request Type**. A sub-menu appears.
2. Click **Add** option from the **File Request Type** sub-menu. The **File Request Type** window appears. This window has following sections:
 - Main
 - Services
 - Messages
 - Transformation Details
3. Enter name of file request type in **File Request Type** field.
4. Verify if the value for **File Format** field is **Extensible Markup Language**.
 - If the value is Extensible Markup Language, go to Step 5.

Note: Default value for File Format field is Extensible Markup Language.

- If **File Format** has any other value,
 - i. Select **File Transformation Required** check box. The File Format field gets enabled.
 - ii. Select Extensible Markup Language from the **File Format** drop-down list.
 - iii. Deselect **File Transformation Required** flag and go to Step 5.

Figure 2: File Transformation

5. Configure at least one service within Service section. This service will be used to process flat file records.

Sequence	Service Type	Service Name	FK Reference	Pre-Processing Algorithm	Postprocessing Algorithm	Operation	Dependent Service Name	Defer Completion
10	Business Object	C1_PERSON_BO	C1-P6 Person			Add		<input type="checkbox"/>

Figure 3: Configuring Service

6. Multiple relational or non-relational services can be configured under single File Request Type. For multiple services, service execution sequence of each record is decided on the basis of given Sequence number.





Services									
	Sequence	Service Type	Service Name	FK Reference	Pre-Processing Algorithm	Postprocessing Algorithm	Operation	Dependent Service Name	Defer Completion
	10	Business Object	C1_PERSON_BO	C1-PERSON			Add		<input type="checkbox"/>
				Person					
	20	Business Object	C1-AccountBO	ACCT	C1-ACCT-ID		Add	C1_PERSON_BO	<input type="checkbox"/>
				Account Relationship Type	Account id derivation algorithm				

Figure 4: Configuring Multiple Services without File Transformation

7. Click **Save**.

5.2 Creating File Request Type with File Transformation

To create file request type with file transformation:

1. From the **Admin** menu, select **F** and then click **File Request Type**. A sub-menu appears.
2. Click **Add** option from the **File Request Type** sub-menu. The **File Request Type** window appears.
3. Enter name of file request type in **File Request Type** field.
4. Select **File Transformation Required** check box. The File Format field gets enabled.

The screenshot shows the 'Main' window for creating a File Request Type. The 'File Request Type' field is set to 'PERSONADDXML'. The 'File Transformation Required' checkbox is checked. The 'File Format' dropdown is set to 'Extensible Markup Language'. Other fields include 'Description' (Person add), 'File Atomicity' (unchecked), and 'File Extension' (xml).

Figure 5: File Transformation

5. Select the required file format from the **File Format** drop-down list. The valid values are:
 - Comma Separated Values (CSV)
 - Extensible Markup Language (XML)
 - Fixed Position
 - JavaScript Object Notation (JSON)
 - Pipe Separated Values (PSV)
 - Tilde Separated Values (TSV)

The screenshot shows the 'Main' window for creating a File Request Type. The 'File Request Type' field is set to 'PSV_PERSON_ADD_DV'. The 'File Transformation Required' checkbox is checked. The 'File Format' dropdown is open, showing options: 'Comma Separated Values', 'Extensible Markup Language' (selected), 'Fixed Position', 'JavaScript Object Notation', 'Pipe Separated Values', and 'Tilde Separated Values'. Other fields include 'File Upload and Process', 'File Header Required', 'File Footer Required', 'Service Log Required', 'File Validation Algorithm' (C1-FRHVA), 'Error Record Maximum Retry' (3), 'Display Profile ID', and 'External System'.

Figure 6: File Format

6. Configure at least one service within Service section. This service which will be used to process flat file records.

Sequence	Service Type	Service Name	FK Reference	Pre-Processing Algorithm	Postprocessing Algorithm	Operation	Dependent Service Name	Defer Completion
10	Business Object	C1-AccountBO	C1-Account			Add		<input type="checkbox"/>

Figure 7: Service Sequence

7. Multiple relational or non-relational services can be configured under single File Request Type. For multiple services, service execution sequence of each record is decided on the basis of given Sequence number.

Services								
	Sequence	Service Type	Service Name	FK Reference	Pre-Processing Algorithm	Postprocessing Algorithm	Operation	Dependent Service Name
+ -	10	Business Object	C1_PERSON_BO	C1-Person			Add	
+ -	20	Business Object	C1-AccountBO	ACCT-Account	C1-ACTXMLGEN Sample File Request pre-processing algorithm		Add	C1_PERSON_BO

Figure 8: Configuring Multiple Services with File Transformation

8. If the file is in CSV or PSV or TSV or Fixed Position format then, Transformation Details need to be configured with respect to its corresponding flat file data line. For more information on file data line, refer section [Mapping Data Line or Record in a File](#).

Transformation Details							
	File Segment Type	Sequence	Field Name	Map Field XPath	Required	Record Identifier	Default Value
+ -	Field Detail	0	EXECUTEBATCH	C1-BILLWRAPSVC/executeBatch	<input type="checkbox"/>	<input type="checkbox"/>	Y
+ -	Field Detail	0	NECESSARYFIELDVALIDATION	C1-BILLWRAPSVC/necessaryFieldValidation	<input type="checkbox"/>	<input type="checkbox"/>	Y
+ -	Field Detail	0	KEYFIELDVALIDATION	C1-BILLWRAPSVC/keyFieldValidation	<input type="checkbox"/>	<input type="checkbox"/>	Y
+ -	Field Detail	0	MANDATORYFIELDVALIDATION	C1-BILLWRAPSVC/mandatoryFieldValidation	<input type="checkbox"/>	<input type="checkbox"/>	Y
+ -	Field Detail	1	ISELIGIBLEFORARCHIVING	C1-BILLWRAPSVC/isEligibleForArchiving	<input type="checkbox"/>	<input type="checkbox"/>	
+ -	Field Detail	2	ISADHOCBILL	C1-BILLWRAPSVC/isAdhocBill	<input type="checkbox"/>	<input type="checkbox"/>	

Figure 9: Configuring Transformation Details

9. Click **Save**.

6. Working with File Request Type

6.1 Uploading and Processing Records using Single Batch Execution

To upload and process records using single batch execution:

1. Define a new file request type or search an existing file request type. Select **File Upload and Process** check box present in **Main** section of **File Request Type** window. This will set the flag as True.

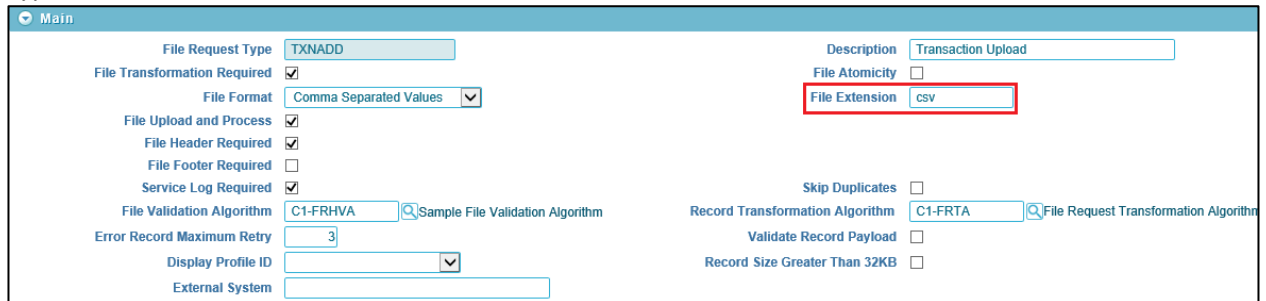
The screenshot shows the 'Main' section of the 'File Request Type' window. The 'File Request Type' is set to 'TXNADD'. The 'File Format' is 'Comma Separated Values'. The 'File Upload and Process' checkbox is checked and highlighted with a red box. Other settings include 'File Transformation Required' (checked), 'File Header Required' (checked), 'File Footer Required' (unchecked), 'Service Log Required' (checked), 'File Validation Algorithm' (C1-FRHVA), 'Error Record Maximum Retry' (3), 'Description' (transaction upload staging csv), 'File Atomicity' (unchecked), 'File Extension' (csv), 'Skip Duplicates' (unchecked), 'Record Transformation Algorithm' (C1-FRTA), and 'Validate Record Payload' (unchecked).

Figure 10: Upload and Process Records

2. Execute C1-FTRAN batch. This batch will upload the file and start processing of all records in the file.

6.2 File Extensions used for Reading Uploaded Files on SFTP server

1. File Upload interface uses flat files to upload data in ORMB system. These flat files can have any extensions like '.txt' or '.csv' or '.dat' or '.xml', etc.
2. The legacy system locates these files on SFTP server to upload data from it. It is possible that legacy system locates different files with different extensions at the same location.
3. The C1-FTRAN batch reads only those files which matches extensions configured in File Request Type.



The screenshot displays the 'Main' configuration window for a File Request Type. The 'File Request Type' is set to 'TXNADD'. The 'Description' is 'Transaction Upload'. The 'File Format' is 'Comma Separated Values'. The 'File Extension' field is highlighted with a red rectangle and contains the value 'csv'. Other fields include 'File Transformation Required' (checked), 'File Upload and Process' (checked), 'File Header Required' (checked), 'File Footer Required' (unchecked), 'Service Log Required' (checked), 'File Validation Algorithm' (C1-FRHVA), 'Error Record Maximum Retry' (3), 'Display Profile ID' (dropdown), 'External System' (text field), 'File Atomicity' (unchecked), 'Skip Duplicates' (unchecked), 'Record Transformation Algorithm' (C1-FRTA), 'Validate Record Payload' (unchecked), and 'Record Size Greater Than 32KB' (unchecked).

Figure 11: File Extension

6.3 Rollback all Processed Records in case of Single Record Failure

1. You can rollback all those processed records in case of single record failure in a file by executing “C1-FTRAN” batch.
2. To rollback processed records, select “**File Atomicity**” check box in **Main** section of **File Request Type** window. It will auto set **File Upload and Process** parameter value to ‘true’ and disable the File Upload and Process.
3. The C1-FTRANbatch will execute both upload and process in a single execution.

Figure 12: Rollback Processed Records

Note:

Since, batch will be executed in single thread, it will have a performance cost. Hence, File Atomicity should be preferred only in case of low data volume.

The “C1-FTRAN” batch will be executed using Single Transaction Strategy.

6.4 Validating Processing Details

1. “Service Log Required” attribute will validate if processing details need to be captured for individual records. If you require service log, select **Service Log Required** check box present in **Main** section of **File Request Type** window.

The screenshot shows the 'Main' configuration window for a file request type. The 'File Request Type' is set to 'TXNADD'. The 'Description' is 'Transaction Upload'. The 'File Format' is 'Comma Separated Values'. The 'File Upload and Process' checkbox is checked. The 'File Header Required' checkbox is checked. The 'File Footer Required' checkbox is unchecked. The 'Service Log Required' checkbox is checked and highlighted with a red box. The 'File Validation Algorithm' is 'C1-FRHVA'. The 'Record Transformation Algorithm' is 'C1-FRTA'. The 'Error Record Maximum Retry' is set to 3. The 'Display Profile ID' is set to a dropdown menu. The 'External System' is set to a text field. The 'File Atomicsity' checkbox is checked. The 'File Extension' is 'csv'. The 'Skip Duplicates' checkbox is unchecked. The 'Validate Record Payload' checkbox is unchecked. The 'Record Size Greater Than 32KB' checkbox is unchecked.

Figure 13: Service Log

2. Record details can have primary key with its service name. Primary key for that record will be stamped only if foreign Key reference is configured in File Request Type for that invoked service.

Sequence	Service Type	Service Name	FK Reference	Pre-Processing Algorithm	Postprocessing Algorithm	Operation	Dependent Service Name	Defer Completion
10	Business Service	C1-BILLWRAPSVC	BILL Header	C1-ACCT-ID Account id derivation algorithm	C1-SAMPLEALG Sample file upload service post processing algorithm	Add		<input checked="" type="checkbox"/>

Figure 14: Service Log_ Foreign Key Reference

3. These details in combination with foreign key reference will be used to navigate to its respective entity like 'Person' or 'Account' or 'Contract', etc.

Note: This feature is made optional to optimize the performance of batch.

You can view the record details which have primary key in File Upload Dashboard.

6.5 Header and Footer Details in a File

1. You can have both header and footer details in a file that is to be uploaded.
2. Header and footer details are optional in file.
3. If you want to upload a file with header, footer details, you need to configure Header and Footer details in corresponding File Request Type.

Note: Header and footer details will be used only for file validations.

4. These details will not be captured in file upload staging.
5. It is mandatory to implement File Validation Algorithm if the provided file has either of header or footer details.
6. If “File Transformation Required” is marked ‘True’ and “File Format” is ‘XML’, then
 - a. “Root XML Tag” is mandatory. This means service payload will have root XML tag as ‘<request>.....</request>’

Figure 15: Root XML Tag

- b. If “File Header Required” is marked as ‘true’ then, “Header XML Tag” is mandatory. This means service payload will have header XML tag as ‘<request><header>.....</header>.....</request>’

Figure 16: Header XML Tag

- c. If “File Footer Required” is marked as ‘true’ then, “Footer XML Tag” is mandatory. This means service payload will have header XML tag as ‘<request>.....<footer>.....</footer></request>’

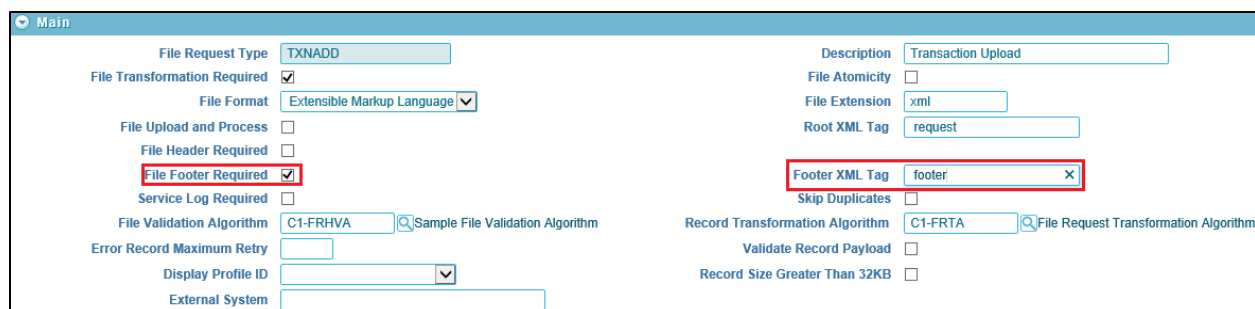
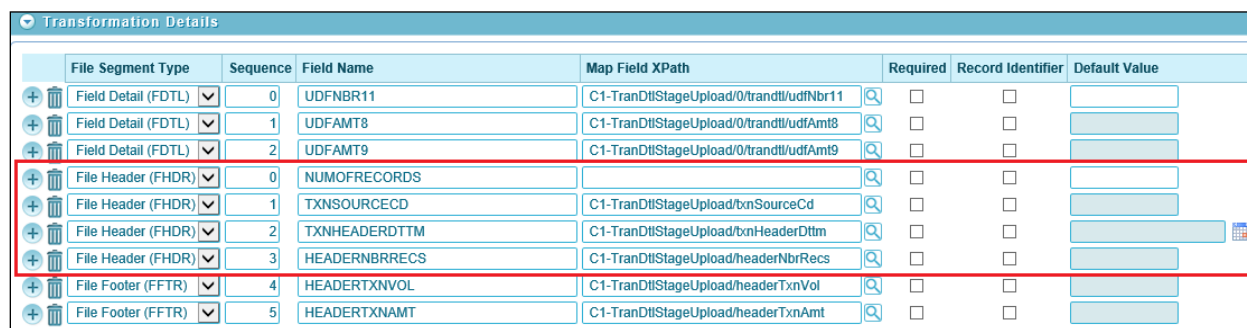


Figure 17: Footer XML Tag

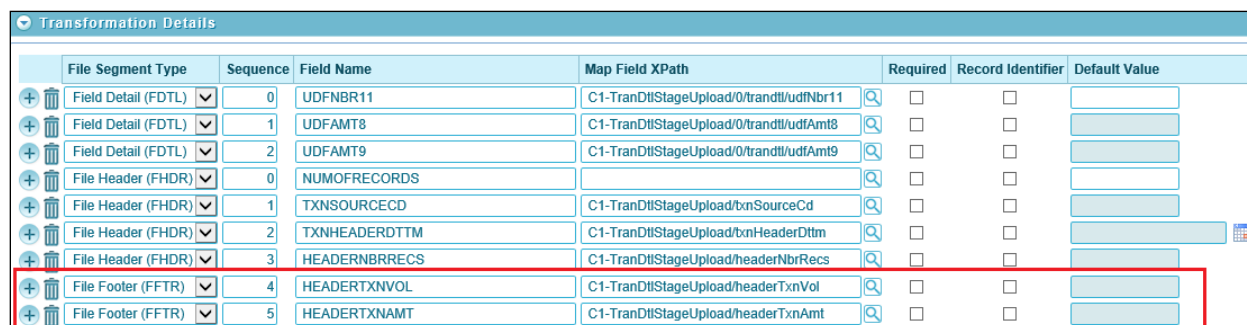
7. If “File Transformation Required” is marked ‘true’ and “File Format” is other than ‘XML’ and ‘JSON’,
 - a. If “File Header Required” is marked as ‘true’ then, “Transformation Details” are required to be configured for Header.



	File Segment Type	Sequence	Field Name	Map Field XPath	Required	Record Identifier	Default Value
+	Field Detail (FDTL)	0	UDFNBR11	C1-TranDtlStageUpload/0/trandtl/udfNbr11	<input type="checkbox"/>	<input type="checkbox"/>	
+	Field Detail (FDTL)	1	UDFAMT8	C1-TranDtlStageUpload/0/trandtl/udfAmt8	<input type="checkbox"/>	<input type="checkbox"/>	
+	Field Detail (FDTL)	2	UDFAMT9	C1-TranDtlStageUpload/0/trandtl/udfAmt9	<input type="checkbox"/>	<input type="checkbox"/>	
+	File Header (FHDR)	0	NUMOFRECORDS		<input type="checkbox"/>	<input type="checkbox"/>	
+	File Header (FHDR)	1	TXNSOURCECD	C1-TranDtlStageUpload/bxnSourceCd	<input type="checkbox"/>	<input type="checkbox"/>	
+	File Header (FHDR)	2	TXNHEADERDTM	C1-TranDtlStageUpload/bxnHeaderDtm	<input type="checkbox"/>	<input type="checkbox"/>	
+	File Header (FHDR)	3	HEADERNBRRECS	C1-TranDtlStageUpload/headerNbrRecs	<input type="checkbox"/>	<input type="checkbox"/>	
+	File Footer (FFTR)	4	HEADERTXNVOL	C1-TranDtlStageUpload/headerTxnVol	<input type="checkbox"/>	<input type="checkbox"/>	
+	File Footer (FFTR)	5	HEADERTXNAMT	C1-TranDtlStageUpload/headerTxnAmt	<input type="checkbox"/>	<input type="checkbox"/>	

Figure 18: File Transformation - File Header

- b. If “File Footer Required” is marked as ‘true’ then, “Transformation Details” are required to be configured for Footer.



	File Segment Type	Sequence	Field Name	Map Field XPath	Required	Record Identifier	Default Value
+	Field Detail (FDTL)	0	UDFNBR11	C1-TranDtlStageUpload/0/trandtl/udfNbr11	<input type="checkbox"/>	<input type="checkbox"/>	
+	Field Detail (FDTL)	1	UDFAMT8	C1-TranDtlStageUpload/0/trandtl/udfAmt8	<input type="checkbox"/>	<input type="checkbox"/>	
+	Field Detail (FDTL)	2	UDFAMT9	C1-TranDtlStageUpload/0/trandtl/udfAmt9	<input type="checkbox"/>	<input type="checkbox"/>	
+	File Header (FHDR)	0	NUMOFRECORDS		<input type="checkbox"/>	<input type="checkbox"/>	
+	File Header (FHDR)	1	TXNSOURCECD	C1-TranDtlStageUpload/bxnSourceCd	<input type="checkbox"/>	<input type="checkbox"/>	
+	File Header (FHDR)	2	TXNHEADERDTM	C1-TranDtlStageUpload/bxnHeaderDtm	<input type="checkbox"/>	<input type="checkbox"/>	
+	File Header (FHDR)	3	HEADERNBRRECS	C1-TranDtlStageUpload/headerNbrRecs	<input type="checkbox"/>	<input type="checkbox"/>	
+	File Footer (FFTR)	4	HEADERTXNVOL	C1-TranDtlStageUpload/headerTxnVol	<input type="checkbox"/>	<input type="checkbox"/>	
+	File Footer (FFTR)	5	HEADERTXNAMT	C1-TranDtlStageUpload/headerTxnAmt	<input type="checkbox"/>	<input type="checkbox"/>	

Figure 19: File Transformation – File Footer

You can also view file header and footer details for the files with complete or pending status. This is possible using the file upload dashboard.

For information on how to view the file header and footer details, refer to Viewing File Header and Footer Details section in *File Upload Interface User Guide*.

6.6 Ignoring or Skipping Duplicate Records in a File

1. To ignore or skip duplicate records in a file select **Skip Duplicates** check box present in **Main** section of **File Request Type** window. This will set the flag as True.

Figure 20: Ignore or Skip Duplicate Records

2. Duplicate records can be categorized in two ways:
 - a. Duplicate record within a same file. A sample file with duplicate records is represented below:

```
TRS|SETTLEMENT ACTIVITY|MF_FD128|2018-01-02-00.00.00|||1|0|USD|+|SUP|00081|0|0|01|N|PROCESSING|N/A|AA17L5
TRS|SETTLEMENT ACTIVITY|MF_FD128|2018-01-02-00.00.00|||1|0|USD|+|SUP|00081|0|0|01|N|PROCESSING|N/A|AA17L4
TRS|SETTLEMENT ACTIVITY|MF_FD128|2018-01-02-00.00.00|||1|0|USD|+|SUP|00081|0|0|01|N|PROCESSING|N/A|AA17L4
```

- The duplicate records will be uploaded with Ignore status and marked “IGN” and will not be processed.

Note: This duplicity is identified on the basis of configured “Record Identifier” in Transformation detail section of File Request Type.

Transformation Details							
	File Segment Type	Sequence	Field Name	Map Field XPath	Required	Record Identifier	Default Value
+	Field Detail (FDTL)	0	UDFNBR11	C1-TranDtStageUpload/0/trandtl/udfNbr11	<input type="checkbox"/>	<input type="checkbox"/>	
+	Field Detail (FDTL)	1	UDFAMT8	C1-TranDtStageUpload/0/trandtl/udfAmt8	<input type="checkbox"/>	<input checked="" type="checkbox"/>	
+	Field Detail (FDTL)	2	UDFAMT9	C1-TranDtStageUpload/0/trandtl/udfAmt9	<input type="checkbox"/>	<input type="checkbox"/>	
+	File Header (FHDR)	0	NUMOFRECORDS		<input type="checkbox"/>	<input type="checkbox"/>	
+	File Header (FHDR)	1	TXNSOURCECD	C1-TranDtStageUpload/txnSourceCd	<input type="checkbox"/>	<input type="checkbox"/>	
+	File Header (FHDR)	2	TXNHEADERDTM	C1-TranDtStageUpload/txnHeaderDtm	<input type="checkbox"/>	<input type="checkbox"/>	
+	File Header (FHDR)	3	HEADERNBRECS	C1-TranDtStageUpload/headerNbrRecs	<input type="checkbox"/>	<input type="checkbox"/>	
+	File Footer (FFTR)	4	HEADERTXNVOL	C1-TranDtStageUpload/headerTxnVol	<input type="checkbox"/>	<input type="checkbox"/>	
+	File Footer (FFTR)	5	HEADERTXNAMT	C1-TranDtStageUpload/headerTxnAmt	<input type="checkbox"/>	<input type="checkbox"/>	

Figure 21: Record Identifier

- b. Records that have been already processed in previous file upload batch execution. There are two possible scenarios:
 - i. A text file has three records. This file will be processed using file upload batch and the records will be uploaded in ORMB system. A sample file is represented below:

```
TRS|SETTLEMENT ACTIVITY|MF_FD128|2018-01-02-00.00.00|||1|0|USD|+|SUP|00081|0|0|01|N|PROCESSING|N/A|AA17L1
TRS|SETTLEMENT ACTIVITY|MF_FD128|2018-01-02-00.00.00|||1|0|USD|+|SUP|00081|0|0|01|N|PROCESSING|N/A|AA17L2
TRS|SETTLEMENT ACTIVITY|MF_FD128|2018-01-02-00.00.00|||1|0|USD|+|SUP|00081|0|0|01|N|PROCESSING|N/A|AA17L3
```

SampleFile_1.txt

- ii. A text file has five records, out of which first three records have been already uploaded and processed using ‘SampleFile_1.txt’. The duplicate records will be uploaded with “SKIP” status and will be skipped while processing. A sample file is represented below:

TRS	SETTLEMENT	ACIVITY	MF_FD128	2018-01-02-00.00.00				1	0	USD	+	SUP	00081	0	0	01	N	PROCESSING	N/A	AA17L1
TRS	SETTLEMENT	ACIVITY	MF_FD128	2018-01-02-00.00.00				1	0	USD	+	SUP	00081	0	0	01	N	PROCESSING	N/A	AA17L2
TRS	SETTLEMENT	ACIVITY	MF_FD128	2018-01-02-00.00.00				1	0	USD	+	SUP	00081	0	0	01	N	PROCESSING	N/A	AA17L3
TRS	SETTLEMENT	ACIVITY	MF_FD128	2018-01-02-00.00.00				1	0	USD	+	SUP	00081	0	0	01	N	PROCESSING	N/A	AA17L4
TRS	SETTLEMENT	ACIVITY	MF_FD128	2018-01-02-00.00.00				1	0	USD	+	SUP	00081	0	0	01	N	PROCESSING	N/A	AA17L5

SampleFile_2.txt

Note: These duplicity is identified on the basis of record existence in “CI_FILE_REQUEST_DETAIL” table using “hash key”.

6.7 Marking the Default Status of Failed Records to “Retry” for Reprocessing

1. You need to configure a value for **Error Record Maximum Retry** parameter in **Main** section of **File Request Type** window. The ‘Error Record Maximum Retry’ field defines the number of attempts allowed to reprocess the failed record. The value should be greater than zero.

The screenshot shows the 'Main' configuration window for a File Request Type. The 'File Request Type' is set to 'TXNFILEVALID'. The 'Description' is 'transaction upload staging csv'. The 'File Format' is 'Comma Separated Values (CSV)'. The 'File Extension' is 'csv'. The 'File Validation Algorithm' is 'C1-FRTA'. The 'Record Transformation Algorithm' is 'C1-FRTDTA'. The 'Error Record Maximum Retry' field is highlighted with a red box and contains the value 3.

Figure 22: Update Failed Record

2. To configure all those error messages for which the failed records should be marked with **Retry** status, select **Retry** from **Record Status** drop-down list in the **Messages** section.

The screenshot shows the 'Messages' section of the configuration window. It displays a table with columns: Message Category, Message Number, Message Text, and Record Status. The first row shows '3' for Message Category, '253' for Message Number, and '%1 field missing' for Message Text. The 'Record Status' dropdown menu is highlighted with a red box and shows 'Retry' selected.

Figure 23: Update Failed Record Error Message

3. All failed records which are not configured in Messages section of File Request Type will be marked with “Error” status.

6.8 Overriding the Service Level Operation at Runtime

1. You can override service level operations using pre-processing algorithm.

Services									
	Sequence	Service Type	Service Name	FK Reference	Pre-Processing Algorithm	Postprocessing Algorithm	Operation	Dependent Service Name	Defer Completion
	10	Business Service	C1-TranDtlStageUpload	C1TX Transaction Detail Information String	C1-ACTXMLGEN Sample File Request pre- processing algorithm		Add		<input type="checkbox"/>

Figure 24: Override Service Level

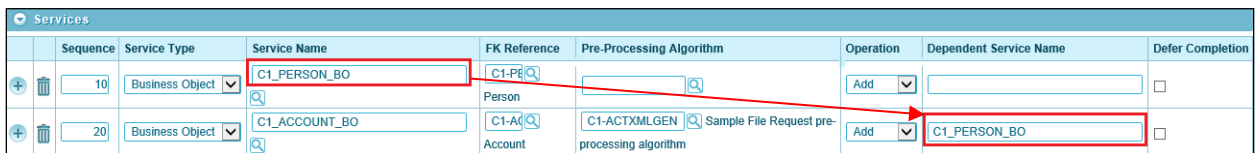
2. The pre-processing algorithm implements the '**setOperation(String operation)**' method to override service level operations.
3. Select **Add** from the **Operation** drop-down list to override the service level operation.

6.9 Skipping Service Execution for a Particular Record

1. You can skip service execution for a particular record using Pre-Processing Algorithm.
2. The algorithm implements the '**isSkipServiceExecution()**' method.

6.10 Executing Dependent Service

1. The **Dependent Service Name** field allows you to address the Payload nesting level and dependent service execution for two independent entities defined in ORMB structure.
2. This signifies parent-child relationship. For example, the ORMB system has 2 independent entities, 'Person' & 'Account' with each entity having their own Maintenance Object (MO). These entities will further have 'Business Object (BO)' derived from the corresponding MO.
3. The Business Object has its own XML schema which is specific to the entity.
4. The legacy system allows you to generate a file with record XML having 'Account' nested in 'Person'. To configure the service, you need to perform following steps:
 - i. Click **Add** (+) icon to add a new sequence.
 - ii. Specify sequence ID in **Sequence** field.
 - iii. Select **Service Type** as Business Object from the drop-down list.
 - iv. Enter **Service Name** for which you need to configure dependent service name. For example, C1_ACCOUNT_BO.
 - v. Specify FK Reference and Pre-Processing Algorithm.
 - vi. Enter **Dependent Service Name**. For example, C1_PERSON_BO.



Sequence	Service Type	Service Name	FK Reference	Pre-Processing Algorithm	Operation	Dependent Service Name	Defer Completion
10	Business Object	C1_PERSON_BO	C1-Person	C1-ACTXMLGEN	Add		<input type="checkbox"/>
20	Business Object	C1_ACCOUNT_BO	C1-Account	Sample File Request pre-processing algorithm	Add	C1_PERSON_BO	<input type="checkbox"/>

Figure 25: Dependent Service

5. Dependent service name configuration helps to get the parent's (Person) service 'Primary key' as an input to its child's (Account) service pre-processing algorithm.

6.11 Deferring Completion of Processed Request

1. It is a flag used to defer the completion of successfully processed request. The final status will be updated once corresponding Business Object Life cycle is completed.
2. It can be used for those Business Objects which have a predefined lifecycle. If service has 'Defer Completion' marked as 'true', then all the successfully processed records for this service will be updated with 'In-Progress' status.

Note: 'Defer Completion' is only used for record status update to legacy system.

3. With this status information, a legacy system can decide an action on the other dependent data upload. For example, if 'Person' Business Object has a lifecycle then, no corresponding 'Account' details will be uploaded until there is an update of 'Person' Business Object lifecycle completion.









Services								
	Sequence	Service Type	Service Name	FK Reference	Pre-Processing Algorithm	Operation	Dependent Service Name	Defer Completion
	 10	Business Object 	C1_PERSON_BO 	C1-PE  Person		Add 		<input checked="" type="checkbox"/>

Figure 26: Defer Completion

6.12 Supporting Client Defined Date Format

1. File Upload Interface now supports client defined dates while uploading and transforming file records.
2. This can be done by giving reference to display profile ID at the time of defining or editing or making a copy of a file request type.

Prerequisites

To link display profile ID with file request type, you should have:

- Display Profile defined in the application.

Procedure

To link display profile id with file request type, you need to:

1. Enter File Request Type name.
2. Select required fields.
3. Select value from the **Display Profile ID** drop-down list.

Note:

The file request should have the same date or time format as specified while defining a display profile.

Each individual File Request Type can support only a single date format.

The screenshot shows the 'Main' configuration page for a File Request Type. The 'File Request Type' field is set to 'TXNADD'. The 'File Format' is set to 'Extensible Markup Language'. The 'Display Profile ID' dropdown is open, showing the following options: Europe, File uploading processing, North America, and UK. Other fields include 'Description', 'File Atomicity', 'File Extension' (set to 'xml'), 'File Upload and Process', 'File Header Required', 'File Footer Required', 'Service Log Required', 'File Validation Algorithm', 'Error Record Maximum Retry', 'Skip Duplicates', 'Validate Record Payload', and 'Record Size Greater Than 32KB'.

Figure 27: Display Profile ID

6.13 Viewing To Do Entries for Failed Files

1. If an exception occurs while executing the File Upload and Transformation batch, you can notify the user about such exception by generating a 'To Do'. You can view the exception details within **To Do Entry**.
2. You can view the total number of To Do Entries that are open or being worked on or completed for each To Do Type respective to file transformation and upload process.
3. The To Do will be created using its corresponding To Do Type.

Note: When defining a To Do Type, it is mandatory to set up drill key of a file ID.

To Do Type				
Main Roles Sort Keys Drill Keys Message Overrides To Do Characteristics Algorithms				
To Do Type <input type="text" value="CM_FTAUB"/>				
Navigation Option File Upload Dashboard				
	Sequence	Table	Field	Owner
	<input type="text" value="1"/>	<input type="text" value="CI_FILE_REQUEST"/> File Request Table	<input type="text" value="FILE_ID"/> File ID	Customer Modification

Figure 28: To Do Type - Drill Keys

4. Once you create a To Do for capturing File Transformation and Upload Batch Error, then on submitting the respective file request, the file transformation and upload process creates a To Do for the user to review the file request. You can perform various actions or modify details about a To Do entry.
5. To view to do entries:
 - i. From the **Admin** menu, select **To Do** and then click **To Do Entry**. A sub-menu appears.
 - ii. Click **Search** option from the **To Do Entry** sub-menu. The **To Do Entry Search** window appears.
 - iii. Enter name of **To Do Type** in **To Do Type** field and click **Search**.
 - iv. The grid that follows contains the To Do entries that match your criteria.

To Do ID	Create Date/Time	Description	Status
08927982401111	05-30-2018 11:13PM	File Transformation and Upload Batch Error	Open
78561928357415	05-30-2018 10:58PM	File Transformation and Upload Batch Error	Open
35420528053979	05-24-2018 10:23PM	File Transformation and Upload Batch Error	Open
25731140516051	05-24-2018 10:21PM	File Transformation and Upload Batch Error	Open
50787708801753	05-24-2018 09:47PM	File Transformation and Upload Batch Error	Open
03491748084312	05-24-2018 09:44PM	File Transformation and Upload Batch Error	Open
26215427228348	05-24-2018 09:43PM	File Transformation and Upload Batch Error	Open
66070966201802	05-24-2018 09:40PM	File Transformation and Upload Batch Error	Open
55645017678216	05-24-2018 09:39PM	File Transformation and Upload Batch Error	Open
14854975504651	05-24-2018 09:39PM	File Transformation and Upload Batch Error	Open
54002274019979	05-24-2018 09:38PM	File Transformation and Upload Batch Error	Open
96996069369668	05-24-2018 09:35PM	File Transformation and Upload Batch Error	Open
23057439667899	05-24-2018 09:30PM	File Transformation and Upload Batch Error	Open
16660042017557	05-24-2018 06:00AM	File Transformation and Upload Batch Error	Open
73913080971364	05-24-2018 05:57AM	File Transformation and Upload Batch Error	Open
11059065053773	05-24-2018 05:54AM	File Transformation and Upload Batch Error	Open
25811460340405	05-24-2018 05:54AM	File Transformation and Upload Batch Error	Open
43741780307733	05-24-2018 05:53AM	File Transformation and Upload Batch Error	Open

Figure 29: To Do Entry Search Grid

The following information appears:

Column Name	Description
To Do ID	Used to indicate the unique identifier of the To Do entry.
Create Date/Time	Used to indicate the date and time the To Do entry was created by the system.
Description	Used to indicate the description of the To Do Type.
Status	Used to indicate the current status of the To Do entry. The valid values are: <ul style="list-style-type: none"> Open Being Worked On Complete

- v. Click on text portion in any of the columns to view the respective To Do Details.

6.14 Post Processing Algorithm Support for File Request

1. You can to define a post processing algorithm which is used to undertake some post-processing activities after successful processing of a record.
2. To attach a post-processing algorithm to file request type, specify the post-processing algorithm code in Postprocessing Algorithm field within Services section.

7. Updating Records marked with 'Error' or 'Pending' status

When updating records marked with 'Error' or 'Pending' status, you have two possible modes:

1. Update records marked with 'Error' status to 'Retry' status
2. Update records marked with 'Pending' status to 'Error' status

ORMB provides you two options to update records.

1. File Upload Dashboard
2. File Request Status Update Batch

7.1 File Upload Dashboard

This is used to update status for less number of records. To update records with "Error" to "Retry" status, you need to follow below steps:

1. Click the **Menu** link in the Application toolbar. A list appears.
2. Select **Tools** from the list. A sub-menu appears.
3. Click **File Upload Dashboard** option. The **Search File Detail** window appears.
4. Enter either **File Name** or select **File Request Type** from the drop-down list. Click **Search**.
5. The file details that meet the search criteria appear in the Search Results section in a tabular format.

File Upload Dashboard

BookmarkRefresh

Main

Search File Detail

File Name

File Request Type

File Upload From Date

Bill Segment Add with bill pricing without

File Status

External System

File Upload To Date

Complete

6 Results. Page 1 of 1 (6 records).

PreviousNext

File ID	File Name	File Request Type	Pending	Processed	Error	Retry	Ignore	Skipped	In Progress	Retry Limit Exceed	Total Records	File Upload Date
0000001841	FILE_BSEG_1_26032018.csv	Bill Segment Add with bill pricing without pre-processing	0	0	1	0	0	0	0	0	1	03-26-2018 05
0000001836	FILE_BSEG_1_23022018.csv	Bill Segment Add with bill pricing without pre-processing	0	0	1	0	0	0	0	0	1	03-26-2018 01
0000001848	FILE_BSEG_1_26032018.csv	Bill Segment Add with bill pricing without pre-processing	0	1	0	0	0	0	0	0	1	03-26-2018 08
0000001847	FILE_BSEG_1_23022018.csv	Bill Segment Add with bill pricing without pre-processing	0	1	0	0	0	0	0	0	1	03-26-2018 08
0000001837	FILE_BSEG_1_23022018.csv	Bill Segment Add with bill pricing without pre-processing	0	1	0	0	0	0	0	0	1	03-26-2018 01
0000001481	FILE_BSEG_1_16022018.csv	Bill Segment Add with bill pricing without pre-processing	0	1	0	0	0	0	0	0	1	02-16-2018 12

Figure 30: File Upload Dashboard

6. The counts in respective Status columns represent the number of records. The numbers are linked to Search File Record Detail form.
7. Click the number in **Error** column to view the record details.

File ID	File Name	File Request Type	Pending	Processed	Error	Retry	Ignore	Skipped	In Progress	Retry Limit Exceed	Total Records	File Upload Date	
0000001841	FILE_BSEG_1_26032018.csv	Bill Segment Add with bill pricing without pre-processing	0	0	1	0	0	0	0	0	1	03-26-2018 05	
0000001836	FILE_BSEG_1_23022018.csv	Bill Segment Add with bill pricing without pre-processing	0	1	0	0	0	0	0	0	1	03-26-2018 01	
0000001848	FILE_BSEG_1_26032018.csv	Bill Segment Add with bill pricing without pre-processing	0	1	0	0	0	0	0	0	1	03-26-2018 08	
0000001847	FILE_BSEG_1_23022018.csv	Bill Segment Add with bill pricing without pre-processing	0	1	0	0	0	0	0	0	1	03-26-2018 08	
0000001837	FILE_BSEG_1_23022018.csv	Bill Segment Add with bill pricing without pre-processing	0	1	0	0	0	0	0	0	1	03-26-2018 01	
0000001481	FILE_BSEG_1_16022018.csv	Bill Segment Add with bill pricing without pre-processing	0	1	0	0	0	0	0	0	1	02-16-2018 12	

Figure 31: Error Records

8. **Search File Record Detail** window appears. Select records for which "Retry" status is to be updated.

Search File Record Detail										
2 Results. Page 1 of 1 (2 records).										
Update Record Status										
<input type="checkbox"/>	Request Id	Primary Key1	Primary Key2	Primary Key3	Primary Key4	Primary Key5	Error Message	Retry Count	Record Payload	Transformed Record Payload
<input checked="" type="checkbox"/>	54205840000000000000						Transformation field details mismatch in MESSAGE_PARM1 file at record no. MESSAGE_PARM2	1		
<input type="checkbox"/>	64205840000000000000						Transformation field details mismatch in MESSAGE_PARM1 file at record no. MESSAGE_PARM2	1		

Figure 32: Search File Record Detail Screen

9. Click **Update Record Status**. The **File Request Detail Update Reason** dialog box appears.

File Request Detail Update Reason

Records with 'Error' and 'Retry Limit Exceed' status will be updated to 'Retry'. Records with 'Pending' status will be updated to 'Error'.

Reason

OK **Cancel**

Figure 33: File Request Detail Update Reason

10. Specify the reason and click **OK**.

Similarly, you need to follow above mentioned steps to update records with “Pending” to “Error” status.

Note: These is an online process, for bulk update it will have a performance impact.

7.2 File Request Status Update Batch

This batch can be used for bulk record status update from “Error” to “Retry” status or “Pending” to “Error” status.

8. Transforming Data

8.1 Transforming File Record in Client Supported Format into ORMB Conform XML

1. You can transform files in Comma Separated Values (CSV) or Extensible Markup Language (XML) or Fixed Position or JavaScript Object Notation (JSON) or Pipe Separated Values (PSV) or Tilde Separated Values (TSV) formats to ORMB conforming XML using “Record Transformation Algorithm”.
2. “Record Transformation Algorithm” implements “FileRequestTransformationAlgorithmSpot”.
3. To transform file record, specify File Request Transformation algorithm in Record Transformation Algorithm field present in Main section. You can also use the **Search** (🔍) icon corresponding to Record Transformation Algorithm field.

Note: Ensure that File Transformation Required field is selected before specifying the algorithm.


The screenshot shows the 'Main' configuration page for File Request Transformation. The 'Record Transformation Algorithm' field is highlighted with a red box and contains the value 'C1-FRTA'. Other fields include 'File Request Type' (TXNADD), 'File Format' (Comma Separated Values (CSV)), 'File Upload and Process' (checked), 'File Header Required' (checked), 'File Footer Required' (unchecked), 'Service Log Required' (checked), 'File Validation Algorithm' (C1-FRHVA), 'Error Record Maximum Retry' (3), 'Description' (transaction upload staging csv), 'File Atomicity' (unchecked), 'File Extension' (csv), 'Skip Duplicates' (unchecked), and 'Validate Record Payload' (unchecked).

Figure 34: Record Transformation Algorithm

8.2 Configuring a Default Value for a Particular Field

1. You can configure default value corresponding to a field. The default value for Sequence should be zero.

Note: Sequence zero is reserved for default value configuration.

2. To set default value for fields with 'Date' as datatype, you can use **Date Picker** () or any of the predefined values listed below.







Transformation Details						
	File Segment Type	Sequence	Field Name	Map Field XPath	Record Identifier	Default Value
	Field Detail	0	KEYFIELDVALIDATION	C1-TranDtlStageUpload/keyFieldValidation	<input type="checkbox"/>	Y
	Field Detail	0	HOW_TO_USE_TXN_FLG	C1-TranDtlStageUpload/0/trandtl/howToUseT	<input type="checkbox"/>	+
	Field Detail	0	TXNDTTM	C1-TranDtlStageUpload/0/trandtl/bnDttm	<input type="checkbox"/>	:SYSDATE
	Field Detail	0	BO_STATUS_CD	C1-TranDtlStageUpload/0/trandtl/status2	<input type="checkbox"/>	UPLD
	Field Detail	1	EXECUTEBATCH	C1-TranDtlStageUpload/executeBatch	<input type="checkbox"/>	
	Field Detail	2	TXNSOURCECD	C1-TranDtlStageUpload/0/trandtl/txnSourceCd	<input type="checkbox"/>	

Figure 35: Transformation Details - Default Value

- :BUS_DATE for Business Date time - relates to process date time
 - :SYSDATE for System date time - relates to System Date Time
 - :STD_DATE for Standard date time - relates to LOCALE date time
3. An error message appears. If you are using predefined values, click **OK** to ignore the warning message.


Invalid Date. Date format is MM-dd-yyyy.

Please re-enter date according to date format specification.

If you need support please supply the following info to system support:

Message number: 11001, 1402

OK

4. To update File Request Type detail, click **Edit** () icon. This enables the 'Default Value' field and fetches the datatype of that corresponding field. The updated value is then used to set the default value.

8.2.1 Applying Default Values Set in File Validation Algorithm to a Field

1. File Upload Interface provides you with set of five parameters which you can use while transforming records. These parameters are defined using setter methods for default1/2/3/4/5 in file validation algorithm.

Prerequisite

To refer default parameter values, you should have

- File Validation Algorithm attached in Main section.

Procedure

1. To refer these default parameters, you can use either of the following constants in **Default Value** field within **Transformation Details** section.
 - ':DEFAULT1'
 - ':DEFAULT2'
 - ':DEFAULT3'
 - ':DEFAULT4'
 - ':DEFAULT5'
2. When you use any of the above constants, the value corresponding to the defined parameter is set as default value to the respective field name.

Transformation Details							
	File Segment Type	Sequence	Field Name	Map Field XPath	Required	Record Identifier	Default Value
+	Field Detail	0	ADDRESS1	C1_PERSON_BO/address1		<input type="checkbox"/>	.DEFAULT2
+	Field Detail	0	PERSONID	C1_PERSON_BO/personId		<input checked="" type="checkbox"/>	.DEFAULT1
+	Field Detail	2	PERSONORBUSINESS	C1_PERSON_BO/personOrBusiness		<input type="checkbox"/>	
+	Field Detail	3	LIFESUPPORTSENSITIVELOAD	C1_PERSON_BO/lifeSupportSensitiveLoad		<input type="checkbox"/>	
+	Field Detail	4	LIFESUPPORTSENSITIVELOADDESC	C1_PERSON_BO/lifeSupportSensitiveLoadD		<input type="checkbox"/>	
+	Field Detail	5	EMAILADDRESS	C1_PERSON_BO/emailAddress		<input type="checkbox"/>	
+	Field Detail	6	OVRDMAILNAME1	C1_PERSON_BO/overrideMailingName1		<input type="checkbox"/>	
+	Field Detail	8	ADDRESS2	C1_PERSON_BO/address2		<input type="checkbox"/>	
+	Field Detail	9	ADDRESS3	C1_PERSON_BO/address3		<input type="checkbox"/>	
+	Field Detail	10	ADDRESS4	C1_PERSON_BO/address4		<input type="checkbox"/>	
+	Field Detail	11	CITY	C1_PERSON_BO/city		<input type="checkbox"/>	

Figure 36: Applying Default Values Set in File Validation Algorithm to a Field

8.3 Mapping Data Line or Record in a File

1. You can map record in a file using Transformation Details section. Using Transformation Detail configuration, mapping can be done only for files in CSV or XML or Fixed Position or JavaScript Object Notation (JSON) or Pipe Separated Values (PSV) or Tilde Separated Values (TSV) formats.

A sample file in PSV format is represented below. The entries are separated by '|' (pipe).

```

TRS|2018-01-02-00.00.00|2|300|1000| Header
N|TRS|SETTLEMENT ACTIVITY|MF_FD128|2018-01-02-00.00.00|||1|0|USD|+|SUP|00081|0|0|01|N|PROCESSING|N/A|AA17L5
2|TRS|SETTLEMENT ACTIVITY|MF_FD128|2018-01-02-00.00.00|||1|0|USD|+|SUP|00081|0|0|01|N|PROCESSING|N/A|AA17L3
Footer
  
```

Header
Records
Footer

Sample_1.dat

2. File is divided into three segments:
 - File Header - it is the first row in the file.
 - File records
 - File Footer - it is the last row in the file.

The File Segment Type is a lookup field with values:

- Field Detail
- File Footer
- File Header

Transformation Details							
	File Segment Type	Sequence	Field Name	Map Field XPath	Record Identifier	Default Value	
+ [X]	Field Detail	0	TXNDTTM	C1-TranDtlStageUpload/0/trandtl/bxnDttm	<input type="checkbox"/>	:SYSDATE	
+ [X]	Field Detail	0	KEYFIELDVALIDATION	C1-TranDtlStageUpload/keyFieldValidation	<input type="checkbox"/>	true	
+ [X]	Field Detail	1	EXECUTEBATCH	C1-TranDtlStageUpload/executeBatch	<input type="checkbox"/>		
+ [X]	Field Detail	2	TXNSOURCECD	C1-TranDtlStageUpload/0/trandtl/bxnSourceCd	<input type="checkbox"/>		
+ [X]	Field Detail	3	TXNRECTYPECD	C1-TranDtlStageUpload/0/trandtl/bxnRecType	<input type="checkbox"/>		
+ [X]	Field Detail	4	DIVISION	C1-TranDtlStageUpload/0/trandtl/division	<input type="checkbox"/>		
+ [X]	Field Detail	5	ACCT_NBR_TYPE_CD	C1-TranDtlStageUpload/0/trandtl/accountIden	<input type="checkbox"/>		
+ [X]	File Footer	1	NUMOFRECORDS		<input type="checkbox"/>		
+ [X]	File Header	2	TXNSOURCECD	C1-TranDtlStageUpload/bxnSourceCd	<input type="checkbox"/>		
+ [X]	File Header	3	BUSINESSDATE		<input type="checkbox"/>		
+ [X]	File Header	4	TXNHEADERDTTM	C1-TranDtlStageUpload/bxnHeaderDttm	<input type="checkbox"/>		
+ [X]	File Header	5	HEADERNBRRECS	C1-TranDtlStageUpload/headerNbrRecs	<input type="checkbox"/>		
+ [X]	File Header	6	HEADERTXNVOL	C1-TranDtlStageUpload/headerTxnVol	<input type="checkbox"/>		
+ [X]	File Header	7	HEADERTXNAMT	C1-TranDtlStageUpload/headerTxnAmt	<input type="checkbox"/>		

Figure 37: Mapping data line

3. All these three segments in transformation detail configuration are independent of each other. Default values can be configured for any of these segments.
4. Multiple number of file records should have same number of fields. For example, if a file has 10 records with seven fields, then all those 10 records should have seven fields with or without values.
5. There are two different ways to configure data line mapping:
 - a. For files in CSV or PSV or TSV format,

- i. Mapping is done by defining “Sequence”.

A sample file is represented below:

```

2-TXNSOURCECD
N | TRS | SETTLEMENT ACTIVITY
N | TRS | SETTLEMENT ACTIVITY
1-EXECUTEBATCH 3-TXNRECTYPECD
  
```

Sample File.txt

Transformation Details							
	File Segment Type	Sequence	Field Name	Map Field XPath	Record Identifier	Default Value	
+	Field Detail	0	MANDATORYFIELDVALIDATION	C1-TranDtlStageUpload/mandatoryFieldValid	<input type="checkbox"/>	N	
+	Field Detail	0	TXNDTTM	C1-TranDtlStageUpload/0/trandtl/bxnDttm	<input type="checkbox"/>	BUS_DATE	
+	Field Detail	1	EXECUTEBATCH	C1-TranDtlStageUpload/executeBatch	<input type="checkbox"/>		
+	Field Detail	2	TXNSOURCECD	C1-TranDtlStageUpload/0/trandtl/bxnSourceC	<input type="checkbox"/>		
+	Field Detail	3	TXNRECTYPECD	C1-TranDtlStageUpload/0/trandtl/bxnRecType	<input type="checkbox"/>		

Figure 38: Sample File Mapping

- ii. For file data mapping, ‘Sequence’ index starts with 1.
 iii. Sequence with Zero index is reserved for mapping default value.

Transformation Details							
	File Segment Type	Sequence	Field Name	Map Field XPath	Record Identifier	Default Value	
+	Field Detail	0	MANDATORYFIELDVALIDATION	C1-TranDtlStageUpload/mandatoryFieldValid	<input type="checkbox"/>	N	
+	Field Detail	0	TXNDTTM	C1-TranDtlStageUpload/0/trandtl/bxnDttm	<input type="checkbox"/>	BUS_DATE	
+	Field Detail	1	EXECUTEBATCH	C1-TranDtlStageUpload/executeBatch	<input type="checkbox"/>		
+	Field Detail	2	TXNSOURCECD	C1-TranDtlStageUpload/0/trandtl/bxnSourceC	<input type="checkbox"/>		

Figure 39: Sample File Mapping Default Value

- iv. Same ‘Sequence’ can be mapped for multiple fields.
 b. For files in ‘Fixed Position’ format,
 i. Mapping is done using ‘Start Position’ and ‘End Position’.

Transformation Details							
	File Segment Type	Sequence	Field Name	Map Field XPath	Start Position	End Position	Record Identifier
+	Field Detail	1	DIVISION	C1_SA/division	1	5	<input type="checkbox"/>
+	Field Detail	2	SAID	C1_SA/serviceAgreement	6	15	<input type="checkbox"/>
+	Field Detail	3	SATYPE	C1_SA/saType	16	21	<input type="checkbox"/>
+	Field Detail	4	STATUS	C1_SA/status	22	23	<input type="checkbox"/>
+	Field Detail	5	ACCTID	C1_SA/accountId	24	33	<input type="checkbox"/>

Figure 40: File Mapping – Fixed Position

- ii. ‘Start Position’ index starts with 1.
 iii. Sequence with ‘Zero’ index can be used for mapping default value.

Note that it is not required to define ‘Start Position’ and ‘End Position’.

Transformation Details							
	File Segment Type	Sequence	Field Name	Map Field XPath	Start Position	End Position	Record Identifier
+	Field Detail	0	DIV	C1_PERSON_BO/division			<input type="checkbox"/>
+	Field Detail	1	PERSONID	C1_PERSON_BO/personid	6	15	<input type="checkbox"/>
+	Field Detail	2	PERSONORBUSINESS	C1_PERSON_BO/personOrBusiness	16	21	<input type="checkbox"/>

Figure 41: File Mapping – Sequence Index

- iv. Same positions can be mapped for multiple fields.

8.4 Mapping File Sequence

1. This is a unique user defined field, which will have the mapped file sequence field value. It will be used as column identifier of a file record.
2. It is free text and can have any user defined value.
3. It has to be uniquely defined within its respective segment. Here, segment refers to File Header or File Detail or File Footer.








Transformation Details				
	File Segment Type	Sequence	Field Name	Map Field XPath
 	Field Detail ▼	0	DIV	C1_PERSON_BO/division 
 	Field Detail ▼	1	PERSONID	C1_PERSON_BO/personid 
 	Field Detail ▼	2	PERSONORBUSINESS	C1_PERSON_BO/personOrBusiness 
 	Field Detail ▼	3	LIFESUPPORTSENSITIVELOAD	C1_PERSON_BO/lifeSupportSensitiveLoad 

Figure 42: Transformation Details – Field Name

8.5 Using Map Field XPath to Transform Records

1. The Map Field XPath uses ORMB application provided sample Record Transformation Algorithm to transform file record into ORMB conform XML. If you want to use this algorithm, you must provide “Map Field XPath” for every configured field.

Transformation Details										
	File Segment Type	Sequence	Field Name	Map Field XPath		Start Position	End Position	Required	Record Identifier	Default Value
+ [X]	Field Detail	1	ACCUALCYCLE	C1-AccountBO/accrualCycle		1	10	<input type="checkbox"/>	<input checked="" type="checkbox"/>	
+ [X]	Field Detail	2	EXCLACRSW	C1-AccountBO/exclAccrSw		11	12	<input type="checkbox"/>	<input type="checkbox"/>	
+ [X]	Field Detail	3	ACCOUNTID	C1-AccountBO/accountId		13	14	<input type="checkbox"/>	<input type="checkbox"/>	
+ [X]	Field Detail	4	BILLCYCLE	C1-AccountBO/billCycle		15	35	<input type="checkbox"/>	<input type="checkbox"/>	
+ [X]	Field Detail	5	SETUPDATE	C1-AccountBO/setUpDate		36	38	<input type="checkbox"/>	<input type="checkbox"/>	
+ [X]	Field Detail	6	CURRENCY	C1-AccountBO/currency		39	41	<input type="checkbox"/>	<input type="checkbox"/>	

Figure 43: Transformation Details - Map Field XPath

Note: The File Segment Type value must be Field Detail.

2. These will have an 'XPath' value with reference to its configured service schema.

Services									
	Sequence	Service Type	Service Name	FK Reference	Pre-Processing Algorithm	Postprocessing Algorithm	Operation	Dependent Service Name	Defer Completion
+ [X]	10	Business Object	C1_PERSON_BO	C1-Person			Add		<input type="checkbox"/>

Figure 44: Service Name – Xpath Value

Note: You can search 'xpath' values using the Search (🔍) icon or enter a free text.

3. Index should be used to configure child entity element xpath in a service schema. To configure child entity element xpath, you need to follow below steps:
 - i. Click **Search** (🔍) icon corresponding to Map Field XPath. File request transform map field zone appears.
 - ii. Select Service Type from the drop-down list.
 - iii. Enter Service Name.

Note: You can select only the child element xpath with index value zero. If you require more than one child elements for a single entity, then you need to select child element xpath with zero' index and edit the required index. For example, '1','2', etc.

Transformation Details										
	File Segment Type	Sequence	Field Name	Map Field XPath		Start Position	End Position	Record Identifier	Default Value	
+ [X]	Field Detail	0	DIV	C1_PERSON_BO/division				<input type="checkbox"/>	CA	
+ [X]	Field Detail	1	PERSONID	C1_PERSON_BO/personid		1	10	<input type="checkbox"/>		
+ [X]	Field Detail	2	PERSONORBUSINESS	C1_PERSON_BO/personOrBusiness		11	12	<input type="checkbox"/>		
+ [X]	Field Detail	3	LIFESUPPORTSENSITIVELOAD	C1_PERSON_BO/lifeSupportSensitiveLoad		13	14	<input type="checkbox"/>		
+ [X]	Field Detail	4	EMAILADDRESS	C1_PERSON_BO/emailAddress		15	35	<input type="checkbox"/>		
+ [X]	Field Detail	5	LANGUAGE	C1_PERSON_BO/language		36	38	<input type="checkbox"/>		
+ [X]	Field Detail	6	DIV	C1_PERSON_BO/division		39	40	<input checked="" type="checkbox"/>		
+ [X]	Field Detail	7	ACCESSGRP	C1_PERSON_BO/accessGroup		41	43	<input checked="" type="checkbox"/>		
+ [X]	Field Detail	8	PERSONID5	C1_PERSON_BO/0/personName/personid		44	53	<input type="checkbox"/>		
+ [X]	Field Detail	9	ENTITYNAME	C1_PERSON_BO/0/personName/entityName		54	74	<input checked="" type="checkbox"/>		
+ [X]	Field Detail	10	NAMETYPE	C1_PERSON_BO/0/personName/nameType		75	78	<input type="checkbox"/>		
+ [X]	Field Detail	11	ISPRIMNAME	C1_PERSON_BO/0/personName/isPrimaryName		79	82	<input type="checkbox"/>		

Figure 45: Map Field Xpath

8.6 Link Parent Service Output with Child Service Input

1. File Upload Interface provides you the flexibility to link parent service output (Primary) key values to child service input field values while transforming data.
2. The supported mapping format is 'PK1/2/3/4/5[CHILD_SERVICE_NAME=PARENT_SERVICE_NAME]'
3. For example, there are three business objects which form a hierarchy:
 - Person
 - Account
 - Service Agreement
4. To map 'Per_Id' within 'Account' business object with primary key value of 'Person' business object (Per_Id), enter the value :PK1[C1-AccountBO=C1_PERSON_BO] in **Default Value** field corresponding to Person ID field.

Services								
Sequence	Service Type	Service Name	FK Reference	Pre-Processing Algorithm	Postprocessing Algorithm	Operation	Dependent Service Name	Defer Completion
10	Business Object	C1_PERSON_BO	Person			Add		<input type="checkbox"/>
20	Business Object	C1-AccountBO	Account			Add	C1_PERSON_BO	<input type="checkbox"/>
30	Business Object	C1_SA	Service Agreement			Add	C1-AccountBO	<input type="checkbox"/>

Messages			
Message Category	Message Number	Message Text	Record Status
CIS_Customer Information	253	%1 field missing	RET

Transformation Details								
File Segment Type	Sequence	Field Name	Map Field XPath	Start Position	End Position	Required	Record Identifier	Default Value
Field Detail	0	APERSONID	C1_PERSON_BO/0/C1-AccountBO/0/accountPerson/personid			<input type="checkbox"/>	<input type="checkbox"/>	:PK1[C1-AccountBO=C1_PERSON_BO]

Figure 46: Mapping Primary Key Value of 'Person' within 'Account' Business Object

5. To map 'C1_SA' within 'Service Agreement' business object with primary key value of 'Account' business object, enter the value :PK1[C1_SA=C1-AccountBO] in **Default Value** field corresponding to Account ID field.

Services								
Sequence	Service Type	Service Name	FK Reference	Pre-Processing Algorithm	Postprocessing Algorithm	Operation	Dependent Service Name	Defer Completion
10	Business Object	C1_PERSON_BO	Person			Add		<input type="checkbox"/>
20	Business Object	C1-AccountBO	Account			Add	C1_PERSON_BO	<input type="checkbox"/>
30	Business Object	C1_SA	Service Agreement			Add	C1-AccountBO	<input type="checkbox"/>

Messages			
Message Category	Message Number	Message Text	Record Status
CIS_Customer Information	253	%1 field missing	RET

Transformation Details								
File Segment Type	Sequence	Field Name	Map Field XPath	Start Position	End Position	Required	Record Identifier	Default Value
Field Detail	0	APERSONID	C1_PERSON_BO/0/C1-AccountBO/0/accountPerson/personid			<input type="checkbox"/>	<input type="checkbox"/>	:PK1[C1-AccountBO=C1_PERSON_BO]
Field Detail	0	CACCTID	C1_PERSON_BO/0/C1-AccountBO/0/C1_SA/accountid			<input type="checkbox"/>	<input type="checkbox"/>	:PK1[C1_SA=C1-AccountBO]
Field Detail	1	PERSONID	C1_PERSON_BO/personid			<input type="checkbox"/>	<input checked="" type="checkbox"/>	

Figure 47: Mapping Primary Key Value of 'Service Agreement' within 'Account' Business Object

6. Linking parent service output with child service input helps to avoid use of pre-processing algorithm.

8.7 Validating Input Values

1. You can perform mandatory field level validations for every record while uploading data using File Transform and Upload (C1-FTRAN) Batch.
2. To perform validations, select **Required** check box corresponding to respective **File Segment Type** in **Transformation Details** section.

Transformation Details								
File Segment Type	Sequence	Field Name	Map Field XPath	Start Position	End Position	Required	Record Identifier	Default Value
Field Detail	0	EXECUTEBATCH	C1-BILLWRAPSV/executeBatch			<input type="checkbox"/>	<input type="checkbox"/>	N
Field Detail	0	NECESSARYFIELDVALIDATION	C1-BILLWRAPSV/necessaryFieldValidation			<input checked="" type="checkbox"/>	<input type="checkbox"/>	Y
Field Detail	0	KEYFIELDVALIDATION	C1-BILLWRAPSV/keyFieldValidation			<input checked="" type="checkbox"/>	<input type="checkbox"/>	Y
Field Detail	0	MANDATORYFIELDVALIDATION	C1-BILLWRAPSV/mandatoryFieldValidation			<input checked="" type="checkbox"/>	<input type="checkbox"/>	Y

Figure 48: Validating Input Values