

Oracle® Revenue Management and Billing Cloud Services

Release 8

Configuration Migration Assistant User Guide

Revision 3.0

F31150-01

May, 2020

Oracle Revenue Management and Billing Cloud Services Configuration Migration Assistant User Guide
F31150-01

Copyright Notice

Copyright © 2020, 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, documentation, and/or technical data delivered to U.S. Government end users are “commercial computer software” or “commercial technical data” 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, documentation, and/or technical data shall be subject to license terms and restrictions as mentioned in Oracle License Agreement, and to the extent applicable, the additional rights set forth in FAR 52.227-19, Commercial Computer Software-Restricted Rights (June 1987). 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 introduces Configuration Migration Assistant (CMA), its workflow, execution process. It will help you to understand the important topics with respect to CMA, describes screens related to the features and explains how to perform various tasks in the application.

Intended Audience

This document is intended for the following audience:

- End-Users
- Administrators
- Consulting Team
- Implementation Team

Organization of the Document

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

Section No.	Section Name	Description
Section 1	Introduction	Provides an introduction of Configuration Migration Assistant.
Section 2	CMA Process Flow	Provides an overview of CMA process.
Section 3	CMA Requirements	Lists the migration assumptions, restrictions and recommendations which considered while working with Configuration Migration Assistant (CMA).
Section 4	Master Configuration - Migration Assistant	Defines Master Configuration and related tasks required for CMA configuration. It also covers defining / searching Migration Plans and Migration Requests.
Section 5	CMA Execution Process	Defines Configuration Migration Assistant execution process. It also describes detailed export / import processes.
Section 6	Import Process Summary	Summarizes the steps required to complete the import process. It highlights what steps are manual and what steps are performed by a batch monitor process.

Section No.	Section Name	Description
Appendix A	List of Configurations Bundled With the Application – Admin Menu	Lists the migration plans and migration requests shipped with the application configured under Admin Menu.
Appendix B	List of Configurations Bundled With the Application – Main Menu	Lists the migration plans and migration requests shipped with the application configured under Main Menu.

Related Documents

You can refer to the following document for more information:

Document	Description
<i>Oracle Utilities Application Framework Administrative User Guide</i>	Describes how to administer the Oracle Utilities Application Framework.

Contents

1.	Introduction	1
2.	CMA Process Flow	2
3.	CMA Requirements.....	5
3.1	Type of Data Migrated.....	5
3.2	Data with System Generated Primary Keys.....	5
3.2.1	MOs with a Mixture of Administration and Non-Administration Data	6
3.3	No Record Deletion	7
3.4	File Transfer Considerations.....	7
3.5	Multi-Language Environment Considerations.....	7
4.	Master Configuration - Migration Assistant	9
4.1	Migration Plans.....	9
4.1.1	Defining a Migration Plan	9
4.1.2	Searching Migration Plan.....	13
4.2	Understanding the BO Filtering Process.....	16
4.3	Migration Plans for Objects with XML-Embedded Links	17
4.4	Migration Request	18
4.4.1	Defining a Migration Request.....	18
4.4.2	Searching Migration Request	23
4.5	Wholesale and Targeted Migrations	26
4.5.1	Wholesale Migrations.....	26
4.5.2	Targeted Migrations	27
4.6	Identifying Tables to Exclude From Migrations	28
4.7	Configuring Custom Objects for Migration	29
4.7.1	Physical Business Object.....	29
4.7.2	Review MO Option Configuration	29
4.7.3	Characteristic Type Configuration	29
4.7.4	Standard CMA Configuration.....	30
5.	CMA Execution Process	31
5.1	Migration Data Set Export	31
5.1.1	Creating Migration Data Set Export.....	31
5.1.2	Searching Existing Migration Data Set Export	33
5.2	Export Lifecycle.....	34
5.3	Importing and Applying a Migration	35
5.3.1	Import Step.....	36

5.3.2	Compare Step	37
5.3.3	Approval Step	41
5.3.4	Apply Step	41
5.3.5	Adjusting Data Prior to Comparing	47
5.3.6	Cancelling a Data Set	48
5.3.7	Additional Note Regarding Imports	48
5.3.8	Caching Considerations	49
5.4	Maintaining Import Data	49
5.4.1	Migration Data Set Import	49
5.4.2	Migration Transaction Portal	53
5.4.3	Migration Object Portal	53
5.5	Running Batch Jobs	53
5.6	CMA Reference	55
5.6.1	Framework-Provided Migration Configuration	55
6.	Import Process Summary	58
Appendix A: List of Configuration Objects Bundled With the Application – Admin Menu		65
Appendix B: List of Configuration Objects Bundled With the Application – Main Menu		70

1. Introduction

Configuration Migration Assistant is a facility to enable the export of customer-owned configuration data from one environment to another.

The Configuration Migration Assistant (CMA) provides implementers with a flexible, extensible facility for migrating configuration data from one environment to another (e.g., from a development environment to a production environment). Data is exported from the source system to a file. The file can then be checked in to a version control system for reuse, or can be immediately imported into the target system and applied.

How does CMA works?

- If you want to transfer some configuration data from production to test or vice versa CMA can be used.
- For CMA to work first of all both environments should have the same product version installed.
- Then from source environment using CMA extract configuration data to file (File will be saved in file system where source environment is).
- Then import this configuration file to target environment to migrate configuration data.
- After file import approve the data to be migrated.
- And finally apply the changes to target environment.

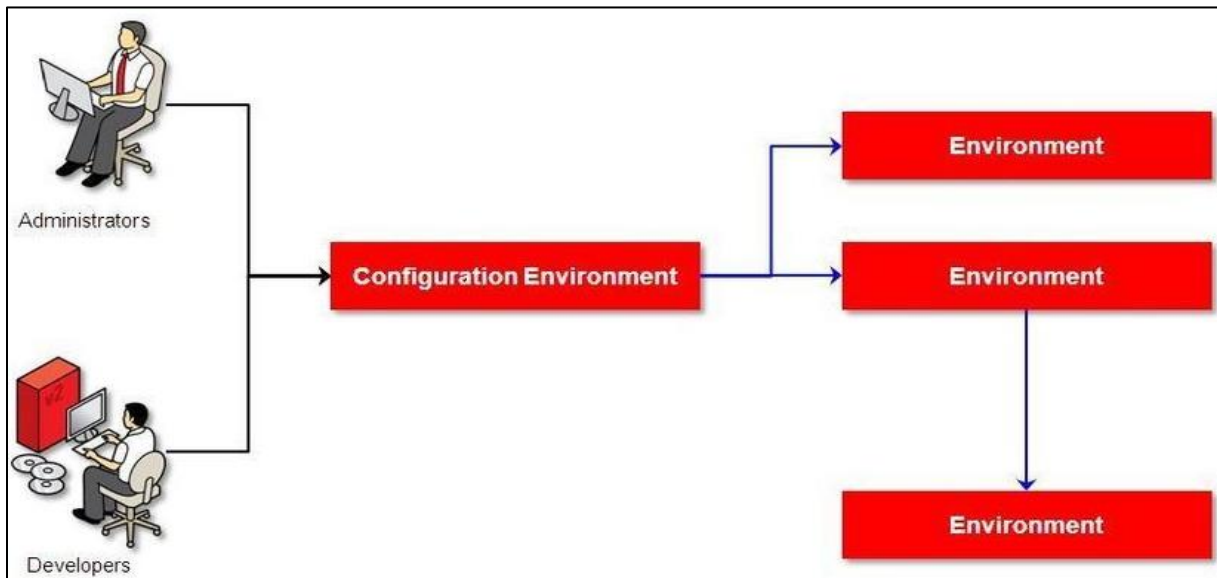


Figure 1: Understanding CMA

Note: As used in this chapter, *source* systems are those on which export-related activities are conducted and *target* systems are those on which migration updates are to occur. The two system preparation tasks described in [Migration Assistant Configuration](#) must be performed on both source and target systems.

2. CMA Process Flow

The high-level CMA process comprises of different tasks and flows, each being described in detail later in this document.

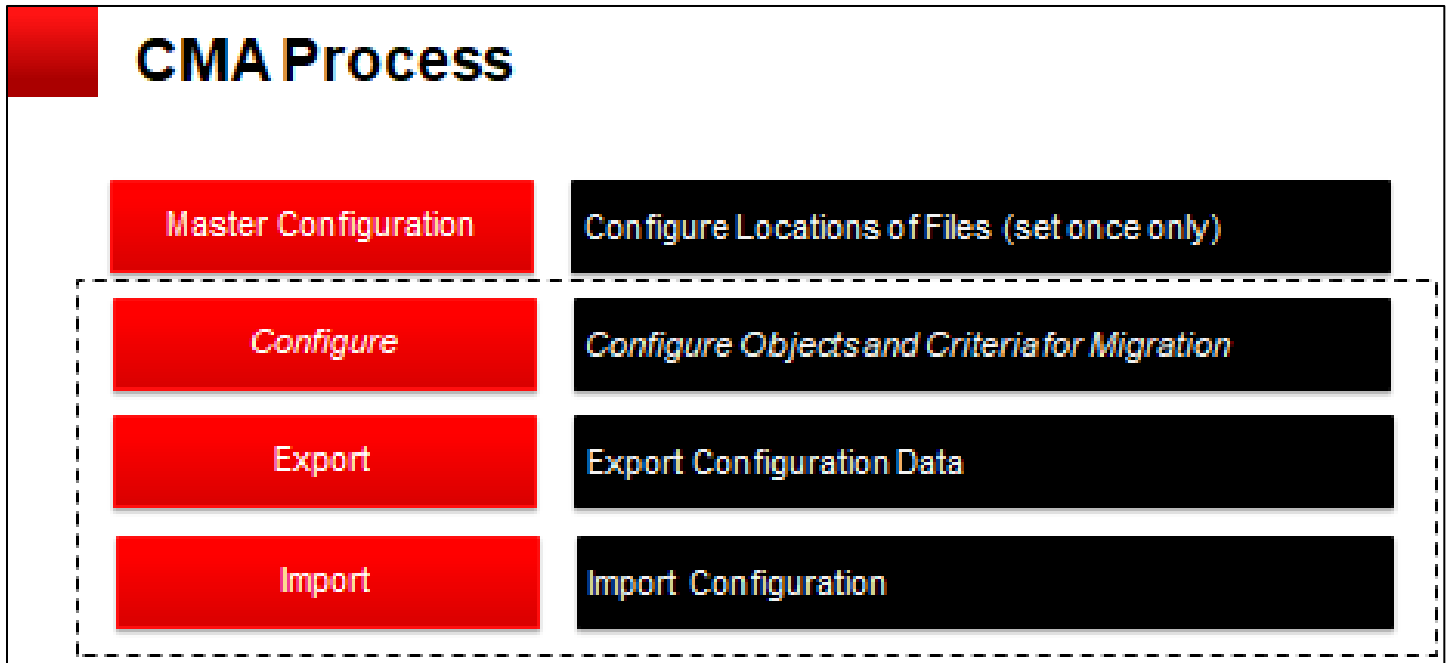


Figure 2: CMA Process Flow

- **Configuration** steps are used to define the data to migrate. This task is performed on the source system and may be defined at any time. Note that the products provide base delivered configuration that may be used as is or used as a template for building more specific configuration for a given implementation.
 - Specify the source and target paths and a file extension for import and export data files in the **Migration Assistant Configuration** master configuration record.
 - Each type of record that may be copied requires a **Migration Plan**. The migration plan is used to identify the maintenance object (MO) for the record (using a business object) and allows for instructions to specify related records that may be included in the migration. Multiple migration plans may exist for a given maintenance object if there are different requirements for migrating records in that MO under different circumstances.
 - The primary instruction in the migration plan could define the physical BO of the record. This signals to CMA that all records for the MO are eligible to be migrated.
 - The primary instruction may define a specific BO, in which case only records that reference that BO in its parental hierarchy are considered.

Note: For more information, refer to the [Understanding the BO Filtering Process](#) section.

- Subsequent instructions may include references to related records. There may be some circumstances where a migration should include subsequent records and some circumstances where they shouldn't be based on requirements.

- A **Migration Request** is used to define the data to include in a given migration. There are several types of migration requests:
 - **Criteria-based** - This type of migration request defines a set of migration plans to be logically included together as part of a migration. For each migration plan, selection criteria may be defined to limit the migration to a subset of data for each MO. Selection may be defined using SQL, an algorithm or explicit primary key values.
 - **Entity List** - This type of migration request allows the user to choose explicit MO / prime keys. It is meant for a targeted migration of specific objects. Note that although the user views and maintains MO / PK values when configuring this type of migration request, a migration plan is still used internally for the CMA migration processing.
 - **Group** - This type of migration request points to other migration requests. This allows you to define separate migration requests that represent logical groupings of migration plan instructions for ease of maintenance, but to combine all the separate migration requests into a single "grouped" migration request for streamlined export / import purposes.
- **Export** process includes all the steps needed to select records to be exported from the source environment and create the export file.
 - A **Migration Data Set Export** object is created for a specific migration request.
 - The lifecycle of the Migration Data Set Export business object includes algorithms that select the appropriate records according to the migration request, determine dependencies between records to build groupings of related objects and create the export file.
 - Because there may be a large volume of data in the export, many of the steps in the lifecycle of the migration data set export are executed via the Migration Data Set Export Monitor.

For more information, refer to the [Exporting a Migration](#) section.

- The file created by the export, which is a BINARY file, needs to be transferred from the export directory to the import directory. The transfer needs to be done in such a way as to preserve the file structure. For more information, refer to the [Migration Assumptions, Restrictions and Recommendations](#) section.
- **Import** processes include all the steps needed to read an imported file, compare the data in the file to the data in the target, review the proposed changes and apply the updates. The following is a high level overview of the process.
 - A **Migration Data Set Import** object is created for a given file to import.
 - The next step reads the import file and creates **Migration Transactions** and **Migration Objects** based on the information in the import file. A migration object is created for every maintenance object record to potentially be created or updated. The migration transaction is a grouping record that groups together related migration objects.
 - The next step is for the objects to **Compare** the data being imported against the data for that record in the target environment. If it is found that the migration object is

new or represents a change to the existing record in the target environment, appropriate SQLs are generated. If no changes are found, the object is marked “unchanged” and doesn’t progress.

- Once all the objects are compared, the user may review the objects for acceptance or rejection.
- When the migration objects are all accepted or rejected, the next step is to apply the objects and update the target environment.

For more information, refer to the [Importing and Applying a Migration](#) section.

3. CMA Requirements

This section describes migration assumptions, restrictions and recommendations which considered while working with Configuration Migration Assistant (CMA).

3.1 Type of Data Migrated

CMA is designed to migrate configuration data.

The comparison step of the import process will generate appropriate insert or update SQL statements for the following data found in the export:

- Configuration data in a maintenance object with no owner flag. This is purely implementation data.
- Configuration data in a maintenance object with owner flag, where the owner is **Customer Modification**. For example, implementer-specific business objects.
- Configuration data in a maintenance object with owner flag, where the main record is owned by the product but where a child record exists with an owner of **Customer Modification**. For example, implementer-specific algorithms added to a base owned business object.
- Customizable fields in a record that is owned by the product. For example, the priority of a based owned To Do type.

3.2 Data with System Generated Primary Keys

The tool provides support for administrative data with system-generated primary keys. The logic relies on the maintenance object to use a method that looks at other attributes of the record (considered a “logical key”) to detect whether the record being migrated already exists in the target region or not. The examples in this section will use the Attachment maintenance object as an example. Common attachments are considered administrative data. The attachment MO uses the file name and the creation date as the “logical key”.

Imagine a common attachment for the "standard rate codes" file exists in a source region with the key 123456789. The table below highlights possible situations at the target region and actions supported in CMA.

Scenario	Target Situation	Action	Comments
1	No matching record	Record can be added with key 123456789.	
2	Record exists with key 123456789 and logic confirms that it is also the "standard rate codes" attachment.	Record can be updated.	

3	Record exists with key 123456789, but logic detects that it is not the "standard rate codes" attachment.	Record is not updated. An error is issued.	The system cannot update this record because it's not the right attachment record.
4	The system detects that another attachment record exists for the "standard rate codes" attachment with a different ID.	Record is not updated. An error is issued.	Assumption is that the record was created directly in the target or was copied from a different source.

The use cases described in scenarios 3 and 4 above would require key mapping to keep track of the id from the source to the id in the target so that any other records from the source that reference this key as a foreign key would be updated as part of the migration. This functionality is not supported.

Scenarios 1 and 2 above are supported for maintenance objects that use the method to detect the logical key.

Note: If a maintenance object with a system generated key does not supply a method to detect the logical key, CMA will update an existing record with the same ID. For maintenance objects in the framework that provide this method, refer to the [Framework Provided Migration Configuration](#) section. For your specific edge application, refer to the CMA addendum for information about support for admin data with system generated keys.

The product recommends that an implementation establishes a migration strategy such that administrative records with system generated keys are always created in the same region and always follow a standard migration path for promoting the data from this source region to other regions. Following this strategy, you would minimize or eliminate the possibility that a record for the same logical key is created in multiple places such that different IDs would be generated as described by scenario 4 above.

3.2.1 MOs with a Mixture of Administration and Non-Administration Data

There are some MOs that contain a mixture of master or transaction data and administrative data. The Attachment is an example of this. The product supports common attachments and owned attachments. Owned attachments are records that are specific to its owner. The owner could be master or transaction data and its attachments are therefore considered master or transaction data. Owned attachments are not candidates for migration using CMA. Common attachments on the other hand are considered administrative data and may be candidates for migration using CMA. For these use cases, an implementation may follow the suggested strategy of only creating the administrative data in one region so that IDs for common attachments are not reused. However, it is reasonable and expected that owned attachments are being created in the target region and may receive a system generated key that matches the key of a common attachment from the source region.

To try to minimize this issue, the system includes special logic to be used by any MO that may contain administrative data mixed in with master or transaction data. This special logic generates the key of an administrative record with a zero (0) in the middle of the key and ensures that the keys for master and transaction data do not include a zero in this spot. For maintenance objects in the framework that use this method, refer to the [Framework Provided Migration Configuration](#) section. For your specific edge application, refer to the CMA addendum for information about additional maintenance objects that may be in this category.

3.3 No Record Deletion

The CMA process allows users to define records to copy from a source environment to a target environment. In that way, the import process for the migrated records is able to identify objects to add and objects to change. There is no mechanism for indicating that records in the target environment should be deleted. The absence of those records in the import is not enough because the migration may be only importing a subset to add or update. If data on the target system must be deleted, users must delete the records in the target accordingly.

Note that CMA does support the deletion of child rows of an object as a result of a comparison. This is only applicable to child records that are owned by the implementation.

3.4 File Transfer Considerations

When moving the export file between systems, use the binary transfer option of whatever tool you use to move the file so that line-end characters are not converted from Linux-style to Windows-style or vice versa.

It is recommended to avoid using 'txt' for the export file's extension (defined in the [master configuration](#)). That file extension by default implies a non-binary file and tools that perform file transfer may treat this as a non-binary file unless explicitly stated. The recommendation is to define 'cma' as the extension. This is not a recognized file extension and most file transfer tools will transfer the file as is.

Note that if the file gets converted, there are two likely outcomes - either a numeric conversion error, or a buffer under-run error may be received when attempting to import the file.

3.5 Multi-Language Environment Considerations

If your implementation uses a language other than English, it means that migrated objects may have multiple language rows (because English is always enabled). There are some important points with respect to multiple languages and CMA:

- As described in User Language, there are steps to follow when supporting an additional language. The steps outlined in that topic highlight that for system data, translation of the strings may be provided via a language pack provided by the product or may be the responsibility of your implementation. In either case, this effort is non-trivial and will have its own established plan. The

expectation is that the translation of the system data is applied for each region at your implementation site. CMA should not be used to create a new language in a target region.

- For administrative / control data that your implementation develops as part of your project, the expectation is that descriptions for your supported language are entered in the region that is considered the source region used to promote changes to regions in the “chain”. For example, control data is entered in a development region and promoted to a test with the supported language enabled in both regions.
- What if you export data from a region with more languages enabled than your target? This scenario is perhaps a case where the source region is a type of test or playpen region where the additional language is enabled for other purposes. In this case, if the language code does exist at all in the target region, the import will produce an error given that the code is invalid. If the language code exists but is not enabled, this will cause the extra language rows to be inserted in the target region, but will not cause any issues. They are simply ignored.
- What if you export data from a region with fewer languages enabled than your target? In this situation, the import process will only create language rows for the languages that were copied from the source. It will not automatically create language rows in the target as part of the import. For this situation, the recommendation is to run the **New Language** batch program (F1–LANG) that creates any missing language entries.

4. Master Configuration - Migration Assistant

In both the source environment and the target environment, the system needs to know the location of the export directory and the import directory along with the expected file suffix. Implementations may define the information explicitly for each region using the **Migration Assistant Configuration** master configuration record.

Note: This record can be updated at any time to change details. The new configuration takes effect on all subsequent exports and imports.

Implementations may also rely on the system defaults. If no Migration Assistant Configuration record is found, the system assumes that there is an entry defined in the system's substitution variable list for F1_CMA_FILES. Further for some of the parameters the default values are set as follows:

- Export directory is the value for this variable plus "**\export**".
- Import directory is the value of this variable plus "**\import**".
- File suffix is set to **cma**

4.1 Migration Plans

A migration plan defines one or more types of objects that are eligible for migration. It is essentially a set of instructions describing how the data to be exported is structured, allowing objects to be migrated together as a logical unit to ensure consistency and completeness.

4.1.1 Defining a Migration Plan

1. To **define** a new migration plan, navigate using **Admin > M > Migration Plan > Add**.

Instruction Sequence	Instruction Type	Parent Instruction Sequence	Description	Business Object	Traversal Criteria Type	Traversal Criteria	Next Migration Plan
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

Figure 3: Migration Plan

The **Add** interface contains following sections:

- **Main** – Allows you to specify the basic details of the migration plan.
- **Instructions** – Allows you to define Instructions for a migration plan.

The **Main** section contains the following fields:

Field	Description
Migration Plan	Used to specify the name for the migration plan.
Description	Used to specify description for the migration plan.
Detailed Description	Used to specify detailed description for the migration plan.

2. In **Migration Plan**, type a name.
3. In **Description**, enter a description.
4. In **Detailed Description**, enter detailed description.

The **Instructions** section contains the following fields:

Field	Description
Instruction Sequence	Used to identify unique instructions. It is recommended to use increments of 10 to allow insertion of other instructions in the future.
Instruction Type	Used to define instruction type.
Parent Instruction Sequence	Used to identify all subsequent instructions.
Description	Used to specify description for the migration instruction.
Business Object	Used to define the type of object from which data will be derived.
Traversal Criteria Type	Used to define the relationship between each of the objects in a migration plan. It is further divided in 3 sub-types.
Traversal Criteria	Used to describe the respective traversal criteria type.
Next Migration Plan	Used to indicate that in addition to copying the object defined in the instruction, any additional instructions included in that referenced migration plan will also be included in an export.
Algorithms	Used to define algorithms associated with each instruction.

5. In **Instruction Sequence**, enter a value.
6. From the **Instruction Type** drop-down list, select **Primary** to add first Instruction Type. All migration plans must contain one and only one primary instruction. All subsequent instructions require a **Subordinate** instruction type. In this case, you must enter Parent Instruction Sequence.
7. In **Description**, enter instruction description.

8. In **Business Object**, enter business object name. You can also use the **Search** (🔍) icon which appears corresponding to this field. Click the **Search** icon and **Business Object Search** window appears. Enter details in any of the search fields:

- Business Object – Enter business object name. You can also use wildcard character ‘%’ to search for business object. Press **Enter**.
- Description
- Maintenance Object associated with the Business Object.

Figure 4: Business Object Search

From ‘Maintenance Object’ drop-down list, select a Maintenance Object. Click Search. When you click Search, the corresponding Business Object is displayed in main window within the Business Object field along with the Maintenance Object.

Business Object	Traversal Criteria Type
<input type="text" value="C1-ACCBALCNT"/> 🔍 Account Balance Counter	<input type="text"/> ▼

Figure 5: Business Object

Note: Though BOs are specified in each instruction, it's important to understand that each BO is used only for filtering purposes. The migrated data set comprises the complete contents of the *maintenance object* that the business object structure is defined against. For a more detailed explanation of this, refer to the [Understanding the BO Filtering Process](#) section.

Note: For more information about defining child tables to always exclude from a migration, refer to the [Identifying Tables to Exclude From Migrations](#) section.


9. Traversal Criteria Type

Traversal Criteria Type	Traversal Criteria	
<input type="text" value="Constraint"/> SQL XPath	Constraint ID	Referring Constraint Owner
	<input type="text"/> 🔍	<input type="text"/> ▼

Figure 6: Traversal Criteria Type

The following table lists the Traversal Criteria Types supported by the Configuration Migration Assistant:

Traversal Criteria Type	Type Description
Constraint	<p>A constraint is a foreign key constraint defined in metadata. If Constraint is selected, the following additional fields are enabled:</p> <ul style="list-style-type: none"> • Constraint ID is a unique identifier for the constraint. The search will show the valid table constraints for the MO of the instruction's BO and the MO of the parent instruction's BO. • Constraint Owner is used to define the owner of the constraint. This is populated automatically when selecting a constraint from the search.
SQL	<p>You can specify SQL join criteria between the parent instruction's object and the child object in the SQL Traversal Criteria. The syntax of the traversal criteria is a WHERE clause (without including the word WHERE). When referring to a field on the parent instruction's object, use the syntax #PARENT.TABLE_NAME.FIELD_NAME. When referring to a field on the current instruction's object, use the syntax #THIS.TABLE_NAME.FIELD_NAME. For example, the following statement is used on a migration plan for Business Object, where the parent instruction is the BO and the subordinate instruction is used to reference the UI Map that is referred to as a BO option with the option type "F1DU":#PARENT.F1_BUS_OBJ_OPT.BUS_OBJ_OPT_FLG = 'F1DU' AND @trim(#THIS.F1_MAP.MAP_CD) = @trim(#PARENT.F1_BUS_OBJ_OPT.BUS_OBJ_OPT_VAL).</p>
XPath	<p>You can apply syntax in an XPath expression referencing elements in the instructions' referenced business objects. This is entered in the XPath Traversal Criteria. For example, the display map collection statement in the SQL example noted above would be written as follows in XPath: #this/mapCd = #parent/businessObjectOption/businessObjectOptionValue AND #parent/businessObjectOption/businessObjectOptionType = 'F1DU'. This technique allows foreign key references that are mapped inside an XML column to be referenced.</p>

10. In **Next Migration Plan**, enter migration plan name. You can also use the **Search**  icon which appears corresponding to this field. Click the **Search** icon and **Migration Plan Search** window appears. Enter details in any of the search fields. (Refer [Searching Migration Plan](#))

- Migration Plan
- Description
- Primary Instruction Business Object
- Primary Instruction Maintenance Object

Figure 7: Migration Plan Search

11. Define Algorithm which needs to be associated with each instruction. You must define the following for each algorithm:
 - Specify **System Event** with which the algorithm is associated. The table below states description of all possible events.

System Event	Optional / Required	Event Description
Import	Optional	Algorithms of this type are no longer supported.
Pre-Compare	Optional	1) Algorithms of this type may be used to adjust the data after it is moved to the target system. These may only be defined on the primary instruction. 2) Refer to Adjusting Imported Data for more information.

- Specify the **Sequence** and **Algorithm** for each system event. You can set the **Sequence** to 10 unless you have a **System Event** that has multiple **Algorithms**. In this case, you need to tell the system the Sequence in which they should execute.

12. Click **Save** to save the migration plan details.

4.1.2 Searching Migration Plan

To **Search** a migration plan, navigate using **Admin > M > Migration Plan > Search**

The **Migration Plan Query** window appears.

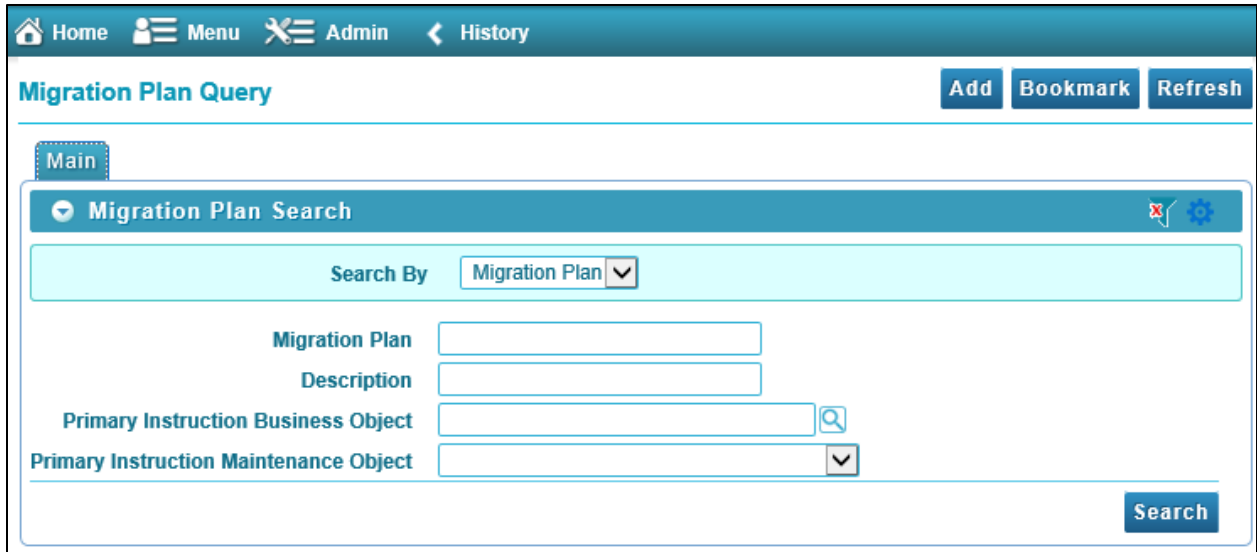


Figure 8: Migration Plan Query

It has 4 search options:

1. **Migration Plan** - Enter the migration plan name. You can also use wildcard character ‘%’ to search for migration plans. Press **Enter**. The Migration Plans appear in Filter section.

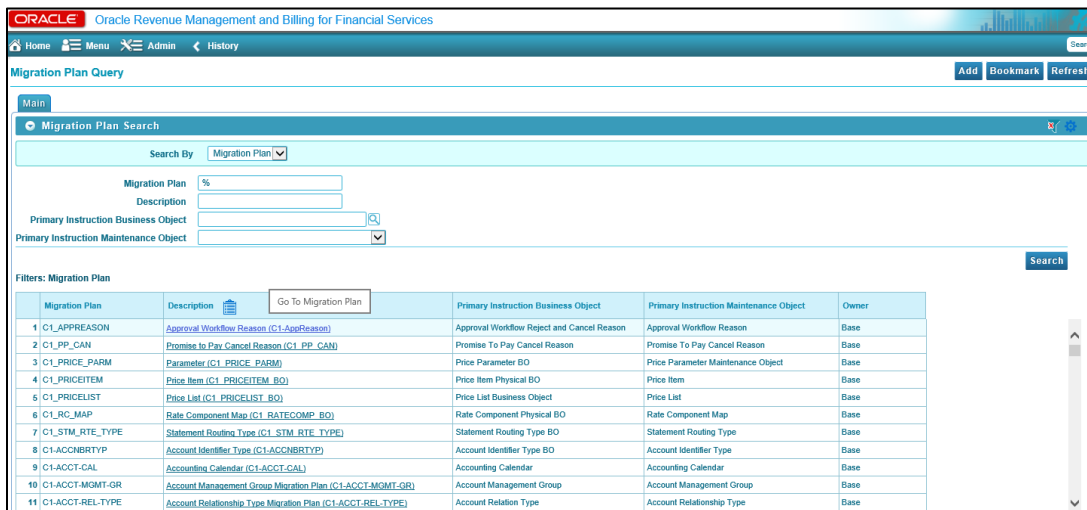


Figure 9: Migration Plan Results

In the Filter section, click the link in the Description column corresponding to the Migration Plan to go to respective migration plan.

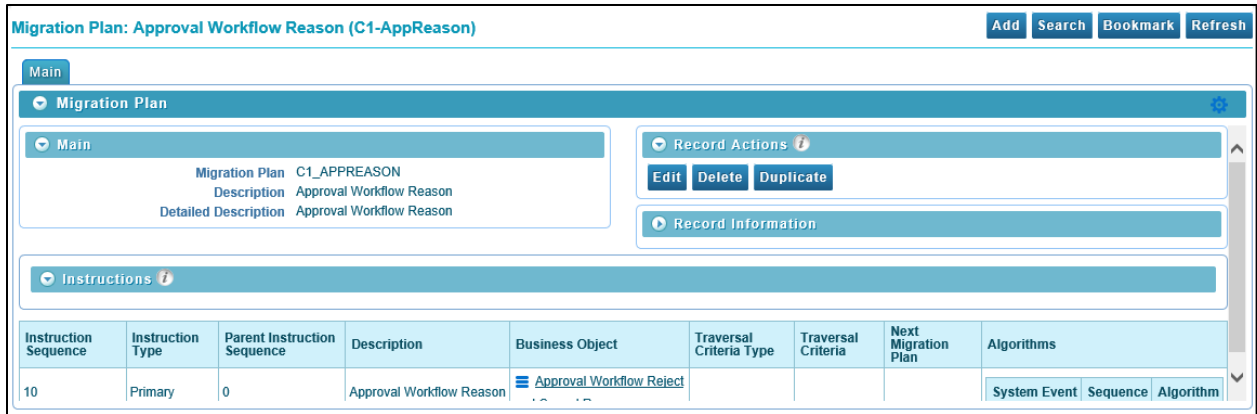


Figure 10: Migration Plan Parameters

2. **Description** - You can enter full text or a portion of the text in the Description field.
3. **Primary Instruction Business Object** – Enter the Business Object name. You can also use the **Search** (🔍) icon which appears corresponding to this field. Click the **Search** icon and **Business Object Search** window appears. Enter details in any of the search fields:
 - Business Object
 - Description
 - Maintenance Object associated with the Business Object – these values are represented as drop-down list.

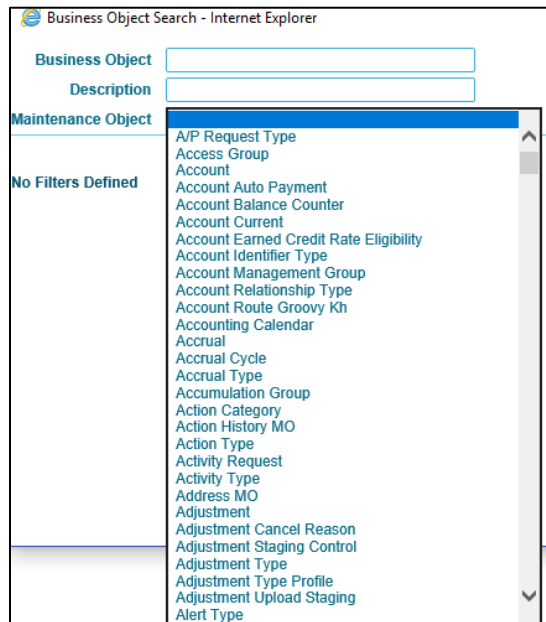




Figure 11: Maintenance Object Values

Select a value from the list and click Search. When you click Search, the corresponding Primary Instruction Business Object is displayed in the text box along with Maintenance Object.

Figure 12: Primary Instruction Business Object

4. **Primary Instruction Maintenance Object** - select a value from the drop-down list.
5. Click **Search**. The Maintenance Object associated with respective migration plan appears in the Filter section.

By default, the Search zone is visible. You can hide the Search zone and view only the Filters by clicking the **Filters** () icon in the upper right corner.

Click **Add** () icon in the upper right corner of the Search zone to define new Migration Plan.

4.2 Understanding the BO Filtering Process

Migration plan instructions require the definition of a business object to provide CMA with information about the record related to the instruction.

If the business object is the physical business object for the maintenance object, then CMA assumes that the instruction applies to all records that satisfy the traversal criteria. CMA recognizes the physical BO by comparing the BO to the value defined in the maintenance object option. If the business object defined is not the physical BO, then CMA will limit the records in the instruction to those that explicitly reference this BO or reference a child of this BO as its identifying BO value. (In other words, this BO must be in the parentage hierarchy of the records to be included in the instruction.)

Note: Unlike Bundling, CMA does not use the BO schema to drive what data is copied for a given record. The BO is only used as a filtering mechanism for selecting records. For more information about how to ensure a child table is not included in a migration, refer to the [Identifying Tables to Exclude from Migration](#) section.

For example, if you define a migration plan for Master Configuration and use the physical business object for the instruction (**F1-MstCfgPhysicalBO**) then all master configuration records are considered for the instruction. If instead the business object you define is Migration Assistant Configuration (**F1-MigrationAssistantConfig**) then only the record related to this business object is included in the instruction.

4.3 Migration Plans for Objects with XML-Embedded Links

When migrating objects where foreign key references are captured in the object's XML based field, subordinate instructions are needed to define the foreign key references in order for CMA to understand the relationships. This is in contrast to direct foreign keys where CMA can determine the relationships using constraints. The instructions provide two purposes. For wholesale migrations, where all data (or a large amount of data) is being migrated, the instructions allow CMA to group related objects into transactions. This helps in the apply process at import time to ensure that related objects are grouped together. However, the apply process includes iterative steps to try to overcome dependencies like this so defining the instructions is not critical for this type of migration. For targeted migrations, defining instructions ensures that the related objects are included in the migration, if appropriate.

The following are options for creating migration plans with XML-embedded links:

- One option is to use the specific logical (business) BO in the primary instruction to define the object you are copying. With this option, the subordinate instructions may use XPath criteria to define the related foreign key. When this approach is used, a separate Migration Plan must be created for each logical BO. (For more information, refer to the [Understanding the BO Filtering Process](#) section.) This option would only be used in isolated cases.
- Another option is to create a migration plan that uses the Physical BO as the primary instruction, and then include a subordinate instruction for the real logical BO, using SQL Traversal to join the object to itself by its primary key. Note that with this technique, the records that reference the logical BO will still only be included in the export file once. At this point further subordinate instructions may use XPath notation to define the foreign key data. Using the physical BO as the primary instruction ensures that all records in the MO are considered. The subordinate instructions with the logical BO and XPath notations will only apply to the records that are applicable to that BO. This option is useful for MOs that have a small number of logical business objects with disparate foreign keys.
- Another option is to use the physical BO in the primary instruction and use raw SQLs in the subordinate instruction's traversal criteria to identify the foreign keys using substring commands. A separate Subordinate Instruction is needed for each SQL corresponding to each element occurrence. Using this technique has the same advantages of the previous in that all records for the MO are included in the migration. However, this technique may be useful for maintenance objects with a larger number of business objects expected where each has one or more foreign keys. It's especially useful if many business objects reference the same foreign key. Then only one instruction is required for that foreign key. Note that a single migration plan may use this technique and the XPath technique for different elements.

A migration request may have multiple migration plans for the same maintenance object. That allows for some flexibility and long term maintainability in that the above techniques may be used in multiple migration plans. Consider the following example:

- A product provides base business objects with foreign keys defined in the XML field and provides the appropriate migration plan with instructions. An implementation extends this business object

or perhaps creates their own business object for the same maintenance object and includes different additional foreign keys in the XML. Rather than duplicating the base migration plan and adding additional instructions for the additional foreign keys, the implementation can create a second migration plan for the MO with the additional foreign keys defined. A migration request should be defined to include both migration plans. In this case if the implementation has only one custom BO, they can choose to use the custom BO as the primary instruction as described above in the first option.

4.4 Migration Request

Once you have created Migration Plans, you can use the same in Migration Requests. Migration Requests are used to define the data to be included in a migration.

4.4.1 Defining a Migration Request

To **define** a new migration request, navigate using **Admin > M > Migration Request > Add. Select Business Object** window appears.

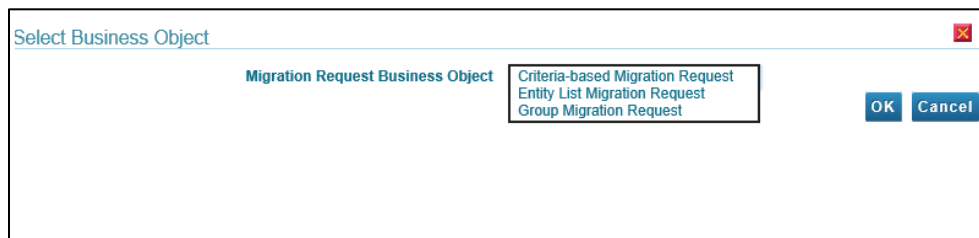


Figure 13: Business Object

There are three types or classes of migration request. The system provides a base business object for each along with a migration request class, which matches the business object. The subsequent sections provide more information about each class of migration request.

Note that all migration requests support defining a Category, which allows implementers to categorize the migration request.

In addition, all classes of migration request include the following zones:

- **Migration Request** This zone contains display-only information about the selected record. Please see the zone's help text for information about this zone's fields.
- **Referencing Migration Requests** This zone is only visible if the displayed migration request is included in a Group migration request. It lists each group migration request that includes it.

4.4.1.1 Criteria-based Migration Request

This type of migration request defines a set of migration plans to be logically included together as part of a migration.

Figure 14: Criteria-based Migration Request

The **Criteria-based Migration Request** window has following sections:

- **Main** – Allows you to specify the basic details of the migration request.
- **Migration Plans** – Allows you to define the group of business objects to migrate as a set of related objects to migrate together as a logical unit.

The **Main** section contains the following fields:

Field	Description
Migration Request	Used to specify the name for the migration request.
Description	Used to specify description for the migration request.
Detailed Description	Used to specify detailed description for the migration request.
Category	Used to categorize the migration request.

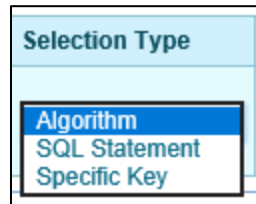
To define Criteria-based Migration Request:

1. In **Migration Request**, type a name.
2. In **Description**, type a description.
3. In **Detailed Description**, type detailed description.
4. Select Category.

The **Migration Plans** section contains the following fields:

Field	Description
Migration Plan	Used to specify the name for the migration plan.
Selection Type	Used to specify the selection criteria to subset the Migration Request for use in exports.
Key Selection	Used to specify a specific instance of an object.

5. In **Migration Plan**, type Migration Plan name. You can also use the **Search** (🔍) icon which appears corresponding to this field. Click the **Search** icon and **Migration Plan Search** window appears. Enter details in any of the search fields. (Refer [Searching Migration Plan](#))
 - Migration Plan
 - Description
 - Primary Instruction Business Object
 - Primary Instruction Maintenance Object
6. For each migration plan, select '**Selection Type**' from the drop-down list. Selection criteria is defined to indicate which records for each Maintenance Objects should be included. Selection may be defined using an algorithm, SQL statement, or explicit primary key values.



- **Algorithm** - For complex selection processes, an Algorithm can be used to return the primary key values of the records to select to migrate. This is ideal for migrations where criterion is determined by values across multiple objects that cannot be expressed with an SQL.

For this Selection Type, the Algorithm code should be selected from the list as the Key Selection. The Algorithm can be written in any algorithm supported language and is defined in metadata as a Migration – Select algorithm.

- **SQL Statement Based Selection Type** - When specifying selection criteria against objects in a Migration Request a common technique is to specify an SQL WHERE clause. This makes sense in that the fundamental objects in Migration Requests are tables via Migration Plans, Business Objects and Maintenance Objects.

In this Selection Type you need to specify the SQL WHERE clause for the primary table in the primary instruction's maintenance object. For example, if your Migration Plan's primary instruction is for a user record, the SQL WHERE clause will be selecting USERID from SC_USER table. You can add any valid WHERE clauses that refers to that table. There are a number of guidelines to optimize this clause:

- Do not include the WHERE SQL reserved word in the criteria as it is implied unless part of a join or subselect. For example, OWNER_FLG = 'CM' is a valid SQL criteria.
- As with most SQL based criteria there is a table alias, which is THIS. This is handy for joins.
- To include criteria from other related tables, use EXISTS to perform subselects. For example: EXISTS (SELECT 1 from F1_BUS_OBJ OPTION WHERE

BUS_OBJ_OPT = 'F1DU' and OPTION.BUS_OBJ_CD = THIS.BUS_OBJ_CD).

- Avoid using SELECT * in subselects for performance reasons. Use SELECT 1 instead.
- Use 1=1 to select all records.
- **Selecting Specific Keys** - One of the common requirements for migrating data is to migrate a specific instance of an object. For example, if you want to migrate a particular user or a particular rate, choose the Specific Key Selection Type. This allows for a specific key value to be specified including composite keys of up to five (5) fields in length. You can specify as many specific primary keys as you want for a single migration plan. In this case you add multiple Specific Key Selection Types.

Note: If for any reason, the table or object has a key of more than five elements then use the SQL Selection Type to cover this requirement.

7. Click **Save** to save the details.

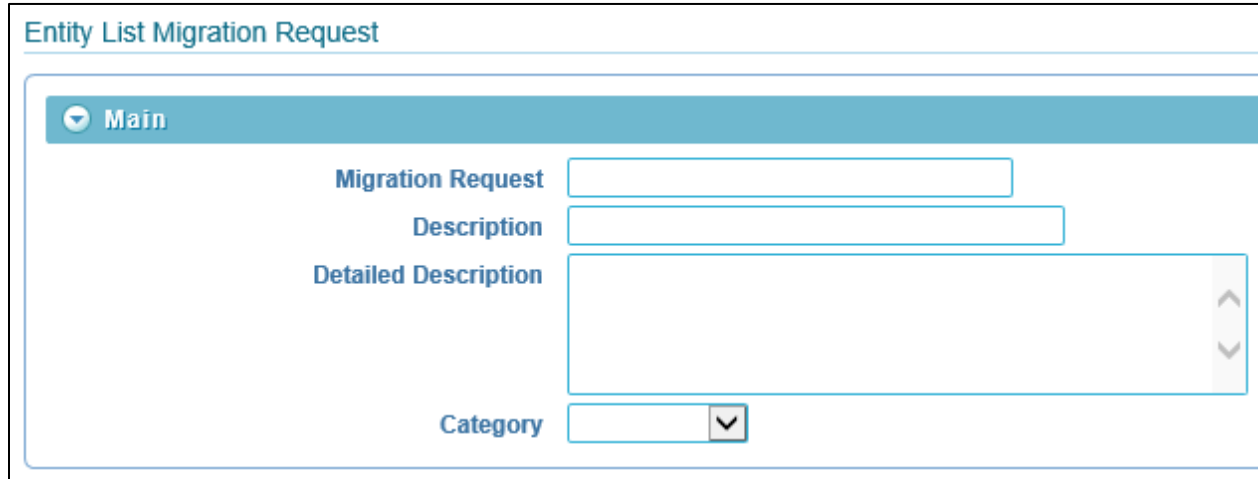
4.4.1.2 Entity List

This type of migration request allows you to choose explicit MO / prime keys. The MOs that are eligible are those that are configured with a **Default Migration Plan** option. Although the user is managing MO / PKs, the migration instructions are still defined with a migration plan. The system maps the migration instructions in a similar way to a **Criteria-based** migration requests that use a Specific Key selection type.

Note: However, it will create a separate migration instruction for each MO / PK combination. It does not try to group all PKs for the same MO / migration under one migration instruction.

For this type of migration request, you need to add a migration request record with its description and other main information. Then, a special zone **Add Entities** is provided to find and select records based on a selected maintenance object and add to the migration request. You are prompted to provide a reference and comments, if desired. If the category selected is one that requires a reference, then this will be validated.

When maintaining a migration request with existing entities, they are visible in the zone **Migration Request Entities**. This zone allows you to remove the entity from the list.



The screenshot shows a web form titled "Entity List Migration Request". At the top, there is a navigation bar with a "Main" tab. Below the navigation bar, the form contains four input fields:

- Migration Request**: A single-line text input field.
- Description**: A single-line text input field.
- Detailed Description**: A multi-line text area with vertical scrollbars on the right side.
- Category**: A dropdown menu with a downward-pointing arrow.

Figure 15: Entity List Migration Request

To define Entity List Migration Request:

1. In **Migration Request**, type a name.
2. In **Description**, type a description.
3. In **Detailed Description**, type detailed description.
4. Select **Category**.
5. Click **Save** to save the details.

4.4.1.3 Group Migration Request

This type of migration request points to other migration requests. This allows you to define separate migration requests that represent logical groupings of migration plan instructions for ease of maintenance, but to combine all the separate migration requests into a single "grouped" migration request for streamlined export / import purposes.

The CMA export process will build an extract that includes the union of all the objects that qualify for the export and group them together based on their relationships.

The screenshot shows a web form titled "Group Migration Request". It features a "Main" section with the following fields: "Migration Request" (text input), "Description" (text input), "Detailed Description" (text area), and "Category" (dropdown menu). Below the "Main" section is an "Included Migration Requests" section, which contains a table with a header "Migration Request" and a search icon. At the bottom right of the form are "Save" and "Cancel" buttons.

Figure 16: Group Migration Request

To define Group Migration Request:

1. In **Migration Request**, type a name.
2. In **Description**, type a description.
3. In **Detailed Description**, type detailed description.
4. Select **Category**.
5. Click **Save** to save the details.

4.4.2 Searching Migration Request

To Search a migration request, navigate using **Admin > M > Migration Request > Search**

The **Migration Plan Search** window appears. There are four modes of operation present as drop-down list next to 'Search By'.

The screenshot shows a search window titled "Migration Request Query". It has buttons for "Add", "Bookmark", and "Refresh". The main section is "Migration Request Search". A "Search By" dropdown menu is open, showing options: "Code / Description", "Migration Plan", "Included Migration Request", and "Entity". Below the dropdown are fields for "Migration Request", "Description", "Class", and "Category", along with a "Search" button.

Figure 17: Migration Request Search

Default value is set to **Code/Description**.

Code/ Description

1. In **Migration Request**, type a name. You can also use wildcard character '%' to search for migration requests. Press **Enter** and the Migration Requests are displayed as Filters. Click the Description name to go to respective migration request.

The screenshot shows a web application window titled "Migration Request Search". At the top, there is a "Search By" dropdown menu set to "Code / Description". Below this, there are input fields for "Migration Request" (containing "%"), "Description", "Class" (a dropdown menu), and "Category" (a dropdown menu). A "Search" button is located to the right of these fields. Below the search area, a section titled "Filters: Migration Request" contains a table with the following data:

	Migration Request	Description	Class	Category	Owner
1	C1_APPREASON	Approval Workflow Reason, Criteria-based	Criteria-based		Base
2	C1_PP_CAN	Promise to Pay Cancel Reason, Criteria-based	Criteria-based		Base
3	C1_PRICE_PARM	Price Parameter, Criteria-based	Criteria-based		Base
4	C1_PRICEITEM	Price Item, Criteria-based	Criteria-based		Base


Figure 18: Migration Request Search_ Code / Description

2. In **Description**, type a description.
3. Select **Class** name from the drop-down list. Select any of the options:
 - Criteria-based
 - Entity List
 - Group
4. Select **Category**.

Migration Plan

1. Migration Plan Search (Refer [Searching Migration Plan](#))
2. In **Migration Request**, type a name. You can also use wildcard character '%' to search for migration requests. Press **Enter** and the Migration Requests are displayed as Filters. Click the Description name to go to respective migration request.
3. In **Description**, type a description.
4. Select **Class** name from the drop-down list. You can select any of the options:
 - Criteria-based
 - Entity List
5. Select **Category**.


Included Migration Request


1. In **Included Migration Request**, enter migration request name and click Search.
2. You can also use the Search () icon which appears corresponding to this field. Click the Search icon and Code/Description window appears. Refer to the [Code/ Description](#) section.

Entity

To search by Entity, enter details in any of the search fields:

1. Select maintenance object name from the drop-down list.
2. Enter Primary Key 1.
3. Enter Primary Key 2.
4. Enter Primary Key 3.
5. Enter Primary Key 4.
6. Enter Primary Key 5.
7. Enter Reference Information.
8. Select **Class** name from the drop-down list. You can select any of the options:
 - Criteria-based
 - Entity List
9. Select Category.
10. Click **Search**. The Migration Requests appear in the Filter section. In the Filter section, click the link in the Description column corresponding to the Migration Request to go to respective migration request.

By default, the Search zone is visible. You can hide the Search zone and view only the Filters by clicking the Filters () icon in the upper right corner.

Click Add () icon in the upper right corner of the Search zone to define new Migration Request.

4.5 Wholesale and Targeted Migrations

The Configuration Migration Assistant is used for two general types of migrations:

- Wholesale Migrations
- Targeted Migrations

The following sections provide some additional information about these concepts.

4.5.1 Wholesale Migrations

They are used when migrating all the configuration and/or administrative data from one environment to another. For example, a wholesale migration might be used when migrating administrative data from a development or test environment to a production environment.

A wholesale migration may be comprised of one or more migration requests that in total include all the administrative data to move. With the ability to group migration requests, the expectation is that implementations follow these guidelines:

- Multiple migration requests using the Criteria-based or Entity List migration request classes are used to group information logically together to allow for more reuse.
- A Group migration request is used for the export. This allows for one data set to export and one data set on the import side, simplifying the process. Note that depending on the amount of data, this may be a large import set to process. An implementation may find it easier to create multiple migration requests that break down the process into several steps.

You should consider that the framework product provides base migration requests and your specific edge product may provide base migration requests as well that may or may not include framework migration plans. Using the product provided migration requests is beneficial with respect to maintenance. As features are added to the product (including new administrative maintenance objects), any impact on CMA should be included in the product owned migration requests. If your implementation introduces new custom administrative maintenance objects that should be included in CMA, then custom migration plans and a custom migration request should be added. Your implementation can build a Group migration request that includes the base migration request and your custom migration requests to have a consolidated export.

Migration plans used in wholesale migrations may be designed to omit subordinate instructions related to explicit foreign keys that are identified through constraints as they are not needed, assuming that the data they are referring to will also be included in the migration.

Note: For more information about migration requests provided in the framework product, refer to the [Framework Provided Migration Objects](#) section. Refer to your specific product's documentation for information about addition base provided migration requests.

4.5.2 Targeted Migrations

A targeted migration refers to migrating a specific subset of data from one environment to another. Examples of targeted migrations include:

- Migration of a new Business objects with its options and algorithms.
- Migration of a new maintenance portal, its zones, and its application service.
- Migration of all outbound message types.

It is expected that the Entity List migration request is used for these types of migrations.

4.6 Identifying Tables to Exclude From Migrations

Some maintenance objects that are eligible to be migrated may include child tables that should not be included in the migration. For example, if an object includes log tables, the entries in the log should reflect the actions on the object in that system, and will be different between the source system and the target system. If you have a custom Maintenance Object that includes tables you don't wish to migrate (such as a log table), use the **Non-Migrated Table** option on the MO to specify this table. All child records for this table will also be ignored during migration.

Another use case to consider is a child "many-to-many" table that connects two administrative objects and exists in the maintenance object of both tables. The child table may be in both MOs for convenience sake, but it may be that one MO is considered more of a "driver" object and the other more of a subordinate. If you are doing a targeted migration where you want to copy a subset of objects, you may want to only copy the driver object and its children and their data but not their children. For example, in Oracle Public Sector Revenue Management, a Form Type includes a collection of valid Form Change Reasons and in turn the Form Change Reason refers to its Form Types. If an implementation wants to do a targeted migration of a form type and include all its related information, including form rules and form change reasons, it does not want the migration of a form change reason to in turn copy all its form types (and their data).

Note: The MO option must be set in both the Source and Target systems for a given MO.

4.7 Configuring Custom Objects for Migration

During the implementation and extension of the product, new custom administrative maintenance objects may be introduced. If your implementation would like to migrate records in those maintenance objects using the Configuration Migration Assistant, additional steps must be performed, which are highlighted in the following sections.

4.7.1 Physical Business Object

As described in [Understanding the BO Filtering Process](#), the migration plan requires a business object for its instruction. The business object is used to identify the records eligible for inclusion in the migration. Assuming your custom tables use one or more “logical” business objects for their processing, your implementation must decide if these business objects are appropriate for use by the migration plans, or if a physical BO is warranted. If so, create an appropriate physical BO.

4.7.2 Review MO Option Configuration

The following points highlight maintenance object (MO) configuration that should be reviewed or updated to support CMA:

- If a physical BO was created (above), link it to the MO as an option using the appropriate option type.
- Be sure that your MO defines an appropriate FK Reference and includes an Option on the MO that identifies the FK Reference. This is used by various portals and zones for CMA when showing detail about records being imported into the target region. Also be sure that this FK reference defines an Information program.
- As described in [Identifying Tables to Exclude From Migrations](#), an MO option is used to identify child tables for an MO that should never be included in a migration. If your custom maintenance object includes a standard Log table, then the recommendation is to list that table as an excluded table. Depending on the specific design of the maintenance object, there may be other child tables to define.

4.7.3 Characteristic Type Configuration

The CMA import process will attempt to create a log record for any administrative object that includes a log table. If your implementation has introduced any custom administrative tables that you plan to include in a migration request and it includes a log table, you must, to ensure that the log creation is successful, add your log table as a valid characteristic entity to the characteristic type **F1-MGO** (Migration Object).

Navigate to Characteristic Type and select the characteristic type **F1-MGO**. Navigate to the **Characteristic Entity** tab and add a row to include the characteristic entity for your custom maintenance object's log table.

4.7.4 Standard CMA Configuration

Create one or more migration plans for the new object, depending on the type of data in the maintenance object and the types of migrations you envision:

- If you have implemented only one "logical" business object used to define the data in the MO, then a single migration plan that references the this BO (or the maybe the MO's physical BO) is appropriate.
- If you have implemented more than one "logical" business object, would the data for multiple business objects get copied together? Then perhaps a single migration plan that references the MO's physical BO is appropriate.
- Are there additional foreign keys defined using mapXML in the business object(s) for the MO? If so, then it is recommended to include sub-instructions to define the links. At this point, if multiple "logical" BOs exist, your implementation may choose to define all the additional elements in the same migration plan or choose to define separate migration plans for each logical BO.
- Your implementation may decide to define more than one migration plan for the same type of record based on the types of migrations you plan to include. For example, you may decide to include a migration plan that copies only the records in this maintenance object. You may decide to define another migration plan that copies the records in this MO along with related records in another MO (for a special type of migration). Having said that, be sure to design the migration plan with reuse in mind to minimize maintenance efforts.

In order to support **Entity List** migration request, a default migration plan must be defined as an option on the maintenance object. This should be a single migration plan that supports all types of business objects for the MO.

If your implementation has a template migration request to use for targeted or wholesale migrations, include the new migration plan(s) as appropriate.

Caution: Important! New migration plans and migration requests should follow naming conventions. Refer to System Data Naming Convention for more information.

5. CMA Execution Process

The execution process includes two processes which are further classified into sub processes. These are:

1. Defining Migration Export
 - Export Data
2. Defining Migration Import
 - Import Data

The sub processes are described in more detail in the following sections. The migration export process begins in the source environment by defining a **Migration Data Set Export**, which specifies a defined **Migration Request** and provides a file name and description for the export file. After the data set is defined and saved, the **Migration Data Set Export Monitor** batch job can be submitted generate the export file.

5.1 Migration Data Set Export

To migrate data from one region to another, define a **Migration Data Set Export**. This establishes the export file name and identifies the migration request.

To define a new migration data set export or view an existing migration data set export, navigate using **Admin > M > Migration Data Set Export**.

5.1.1 Creating Migration Data Set Export

To define a Migration Data Set Export on the source environment, navigate using **Admin > M > Migration Data Set Export > Add**

Figure 19: Migration Data Set Export

The **Migration Data Set Export** window appears. Enter the field values:

1. In Migration Request, type a name. You can search for Migration Request name. (Refer [Searching Migration Request](#))

2. In **File Name**, type a unique name. Do not use spaces in the name, and do not enter the file extension or a path.
3. In **Export Description**, enter the description text to provide information about the purpose of the export. Note that this field is not language enabled.
4. **Export Directory** - is the variable of the directory to which migration export files will be written. This path should be recognizable by the batch job processor, and that process must have write access to this directory. The variable must be predefined in the system and the field value will be validated for a presence of the valid variable enclosed in char '@'. The default variable for this process is @F1_CMA_FILES@. Your implementation may define a different variable.
5. **File Suffix** - defines the file suffix to be appended to the end of your migration file names, for example 'cma'. The period should not be included. It will be added when the file is created. The import process also uses this suffix to find files at import time.

Note: The output location and file extension of the intended export file, which should appear in the Export Directory and File Suffix labels, are defined as described in the topic [Migration Assistant Configuration](#).

6. **Source Environment Reference** - is for information purposes. It should be populated with text that provides a meaningful description of the source environment. The default value is the URL of the source environment.
7. Click **Save**. The Migration data set export request is generated and **Migration Data Set Export Details** window appears. An export ID appears in top pane of the window.

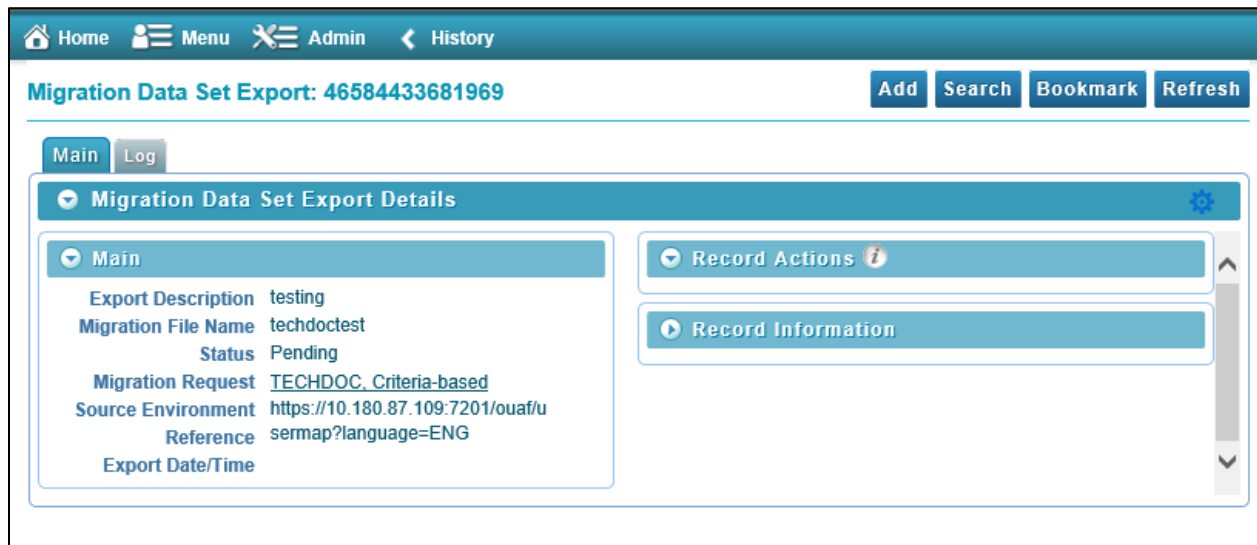



Figure 20: Migration Data Set Export Details

There are 3 different sections:

1. **Main** – this section has following fields:
 - Export Description
 - Migration File Name
 - Status - defines the state of the process. This status is classified as:

- Canceled – you may choose to temporarily cancel a pending data set to prevent it from being processed. You can later return it to the Pending state when desired.
 - Error - If an error is detected, the process stops and the record transitions to Error.
 - Exported - When the process is marked as Exported, the export file can be imported into the target system.
 - Pending - The record remains in Pending until its monitor batch job is submitted
- Migration Request
 - Source Environment
 - Reference
 - Export Date/Time
2. **Record Actions** - shows the valid actions that can be performed on this object in its current state.
 3. **Record Information** – contains following details
 - Migration Data Set id
 - created by
 - business object name
 - status date/time
 - create date/time

You can view export logs in Log section. Click  icon to view the logs.

5.1.2 Searching Existing Migration Data Set Export

To search existing Migration Data Set Export on the source environment, navigate using **Admin > M > Migration Data Set Export > Search**. The Migration Data Set Export Query window appears.

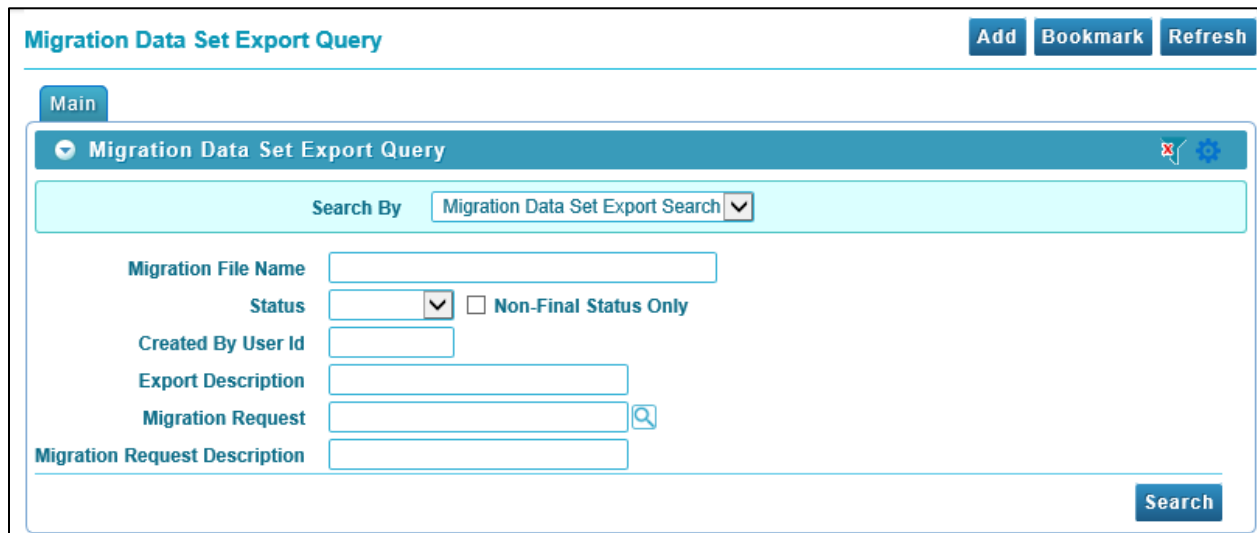




Figure 21: Migration Data Set Export Query

You can search by any of the following options:

1. **Migration File Name** - Type a file name. You can also use wildcard character ‘%’ to search for migration file name. Press **Enter**.
2. **Status** – is classified as:
 - **Canceled** – you may choose to temporarily cancel a pending data set to prevent it from being processed. You can later return it to the Pending state when desired.
 - **Error** - If an error is detected, the process stops and the record transitions to Error.
 - **Exported** - When the process is marked as Exported, the export file can be imported into the target system.
 - **Pending** - The record remains in Pending until its monitor batch job is submitted.
 - Non-Final Status Only.
3. **Created By User Id** – Enter id of the user who has created a Data Set.
4. **Export Description** - Enter description of the data set.
5. Migration Request Search (Refer [Searching Migration Request](#))
6. **Migration Request Description** - Enter description of migration request.
7. Click **Search**. The Migration Data Set Export list appears in the Filter section.

By default, the Search zone is visible. You can hide the Search zone and view only the Filters by clicking the **Filters** () icon in the upper right corner.

Click **Add** () icon in the upper right corner of the Search zone to define new Migration Data Set Export.

5.2 Export Lifecycle

The following diagram describes the lifecycle of a Migration Data Set Export (data set).

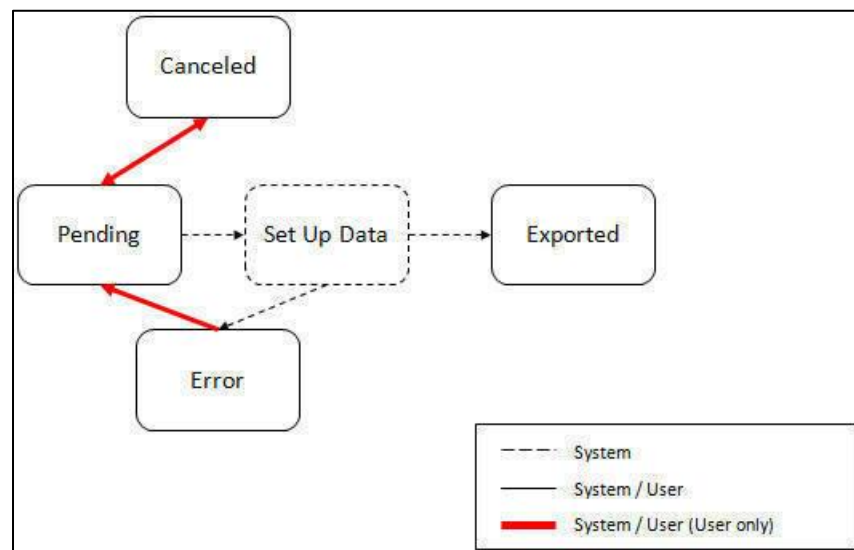


Figure 22: Export Lifecycle

The following points describe the lifecycle.

- The data set is created in **Pending** status.
- You may choose to temporarily **Cancel** a pending data set to prevent it from being processed. The user can later return it to the Pending state when desired.
- The record remains in Pending until its monitor batch job is submitted. The **Migration Data Set Export Monitor** (F1–MGDPR) selects pending records and transitions them to **Set up data**. For more information, refer to the [Running Batch Jobs](#) section.
- Set up data is a transitory state that includes the algorithm that does the work of determining the objects to include in the export and group related objects together into a transaction. If everything is successful, the export file is written to the appropriate file location and the record transitions to **Exported**. If an error is detected, the process stops and the record transitions to **Error**.
- If a record is in error and it is possible to correct the error, the record may be transitioned back to Pending to try again.
- When the process is marked as **Exported**, the export file can be imported into the target system.

Note: The export process creates a file, providing the benefits of having a standalone file. It can be stored in a version control system for audit purposes or provided to others for import purposes.

Caution: Under no circumstances should exported data files be edited manually. Doing so could cause data corruption when the file content is applied to the target environment.

Note: The export functionality is supported using the business object **Migration Data Set Export** (F1-MigrDataSetExport). The expectation is that implementations will use the delivered base business object and its logic and will have no reason to implement a custom business object for the CMA export process.

5.3 Importing and Applying a Migration

Once the data has been exported and placed in an export file it can be imported at any time, as required. The Migration Data Set Import is registered with the location and name of the file to import. At this time, the behaviour of the import process can be configured to decide how to treat changes to existing data and new data.

The import process involves four steps which can be stated as:

- Import - The first step covers importing the file and creating appropriate Migration Import records in the target environment to facilitate the subsequent steps.
- Compare – This step reviews each object that is in the import file and compares the object in the import against the equivalent record in the target environment. The comparison step results in noting which objects are unchanged, which are new (and the appropriate SQL to insert them) and which objects are changed (and the appropriate SQL to update them). Based on user configuration at import time, the objects that qualify for the import may be in a state that requires review or may be pre-approved.

- **Approve** - Once the comparison is complete, the user should review the results. There may be records marked for review. All of these records must be approved or rejected before the import can proceed. When the user is satisfied with the results of the comparison and has completed the Import Step
- **Apply** - This is the final step where the records in the target environment are added or updated. Because of potential high volumes of data and because of possible dependencies between records, this step supports two levels of attempting to apply the records. There is an apply step at the object level and an apply step at the transaction level.

5.3.1 Import Step

The import process starts with verifying the import directory configured as described in topic [Master Configuration](#) and ensuring that the exported file is located in that directory. Then, in the target environment, a **Migration Data Set Import** record should be created. The user indicates the file name.

In addition, the user decides what the default status should be for resulting objects.

- The **Default Status for Add** sets the default status for objects that are determined to be *new* during the import comparison process. The default is to automatically set new objects to **Approved** status. Other options are to set any new objects to **Rejected** or **Needs Review** status.
- The **Default Status for Change** sets the default status for objects that are determined to be changed during the import comparison process. As with new objects, the default for changed objects is **Approved**, with **Rejected** or **Needs Review** options available.

The user may also configure the **Automatically Apply** flag to **Yes**. This allows for use cases where the migration is repetitive and has been tested and the user feels that there is no need for manual approval. Note that when configuring this setting, neither of the Default Status values may be set to **Needs Review** and at least one must be set to **Approved**.

The file to import contains a list of all the objects included in the export. Any objects that the export step determined to be related have been grouped into “transactions”. Once the Migration Data Set Import is created, the next step is for the system to read in the file and create Migration Transactions and Migration Objects.

While executing Migration Data Set Import, there are 2 main status associated with the data set.

- The data set is created in **Pending** status.
- The record remains in Pending until its monitor batch job is submitted. The **Migration Data Set Import Monitor** (F1–MGDIM) selects pending records and transitions them to **Ready to Compare**. For more information, refer to the [Import Process Summary](#) section and [Running Batch Jobs](#) section.

The Ready to Compare state has an algorithm that is responsible for reading the related import file and creating the migration transactions and migration objects. The data set remains in this state until the comparison step is complete.

Note: You may choose to **Cancel** a data set. For more information, refer to the [Cancelling a Data Set](#) section.

The following diagram highlights the relationships of the resulting migration import records.

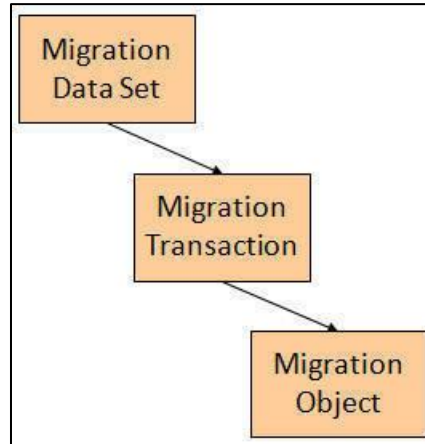


Figure 23: Migration Import Relationships

The migration transaction and migration object each have their own lifecycle that will help manage the subsequent compare and apply steps. At the end of the import step, the status values of the three types of records are as follows:

- Migration Data Set Import is in the **Ready to Compare** state.
- Migration Transaction is in the **Pending** state.
- Migration Object is in the **Pending** state.

Note: The import functionality is supported using business objects supplied by the base product. The expectation is that implementations will use the delivered base business objects and their logic and will have no reason to implement a custom business objects for the CMA import process. The base business objects are **Migration Data Set Import** (F1-MigrObjectImport), **Migration Transaction** (F1-MigrTransactionImport) and **Migration Object** (F1-MigrObjectImport).

5.3.2 Compare Step

The import step results in the creation of one or more migration objects, one for each record selected in the export based on the export's migration request and its configuration. The related objects are grouped together in migration transactions. The next step in the import process is the Comparison step. In this step, the data captured by the import file for each object is compared to the view of that object in the target environment.

To cater for a possible large volume of objects, the comparison is done via a batch monitor. To aide in performance of the process, the monitor is performed on the migration objects so that it can be run multi-threaded. Once the objects are finished with the comparison, the migration transactions and the migration data set should be updated with an appropriate overall status before continuing to the next step. As a result, the comparison actually requires three steps: Migration Object Comparison, Migration

Transaction Status Update and Migration Data Set Export Status Update. The steps are explained in detail in the following sections.

Note: For more information about streamlining the various steps in the process, refer to the [Running Batch Jobs](#) section.

1. Migration Object Compare

This is the main step of the comparison. The **Migration Object Monitor (F1-MGOPR)** selects pending migration object records and transitions them to **Comparing**. This is a transitory state that includes the algorithm that does the work of comparing. There are various possible outcomes that could occur based on the logic in the algorithm. The following diagram illustrates a portion of the migration object lifecycle that pertains to comparison.

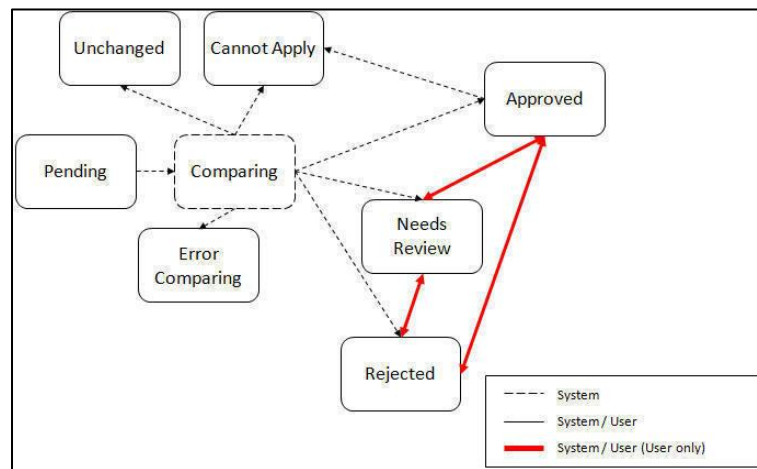


Figure 24: Migration Object Comparison

The following points describe the lifecycle.

- When **Pending** records are selected by the monitor batch job, it transitions to **Comparing**. If the migration object refers to one or more pre-compare algorithms, they are executed to [adjust the data prior to comparison](#). Then algorithm will determine the appropriate next state by comparing the source data to the target data.
- If the record in the migration object is found in the target environment and the data is exactly the same, the record transitions to **Unchanged** (with the object action value also set to **Unchanged**).
- If the record in the migration object is found in the target environment and the data is different, the algorithm sets the object action value to Change and generates the appropriate SQL to be used later in the Apply step to update the record. It then transitions to **Approved**, **Needs Review** or **Rejected** based on the Default Status For Change setting captured on the Data Set.
- If the record in the migration object is not found in the target environment, the algorithm sets the object action value to **Add** and generates the appropriate SQL to be used later in the Apply step to insert the record. It then transitions to **Approved**, **Needs Review** or **Rejected** based on the Default Status For Add setting captured on the Data Set.

- If there is any issue with attempting to parse the object data from the import, the record transitions to **Error Comparing**.
- If there is any reason that the imported object is not valid for import, the record transitions to **Cannot Apply**. The log will be updated with the error that caused the record to transition to this state. An example is that perhaps the record was exported in a different version of the product and has additional elements that are not recognized in this version.

Note: For information about cancelling a data set and its impact on its related objects, refer to the [Cancelling a Data Set](#) section.

2. Migration Transaction Status Update

After the import step, the migration transaction remains in the Pending state until all its objects have completed the comparison step. At that point, the status of the transactions should be updated based on the results of their objects. The **Migration Transaction Monitor** (F1-MGTPR) selects pending migration transaction records and runs its monitor algorithms. There are various possible outcomes that could occur based on the logic in the algorithms. The following diagram illustrates a portion of the migration transaction lifecycle that pertains to comparison.

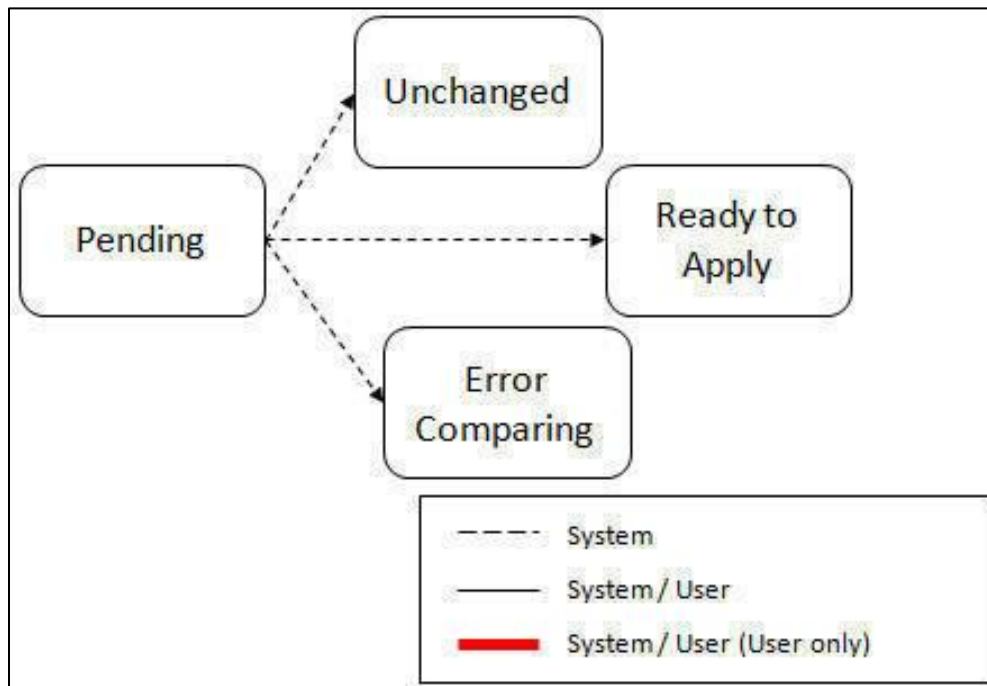


Figure 25: Migration Transaction Status Update

The following points describe the lifecycle possible next states after Pending.

- If any related migration object is in the Error Comparing state, the transaction transitions to **Error Comparing**.
- If all related migration objects are in the Unchanged state, the transaction transitions to **Unchanged**.

- Otherwise, the transaction transitions to **Ready to Apply**. This means that at least one object is in an “apply-able” state.

The transaction remains in the **Ready to Apply** state until a user has approved the data set to move to the Apply step and the transaction’s related objects have attempted to apply themselves. This is described in more detail below.

Note: For information about cancelling a data set and its impact on its related objects, refer to the [Cancelling a Data Set](#) section.

3. Migration Data Set Import Status Update

Once all the objects and all transactions have been updated via the previous two steps, the migration data set export must be updated based on the results of their transactions. The **Migration Data Set Import Monitor** (F1-MGDIM) selects Ready to Compare data sets and runs its monitor algorithms. Note that this is the same monitor process that is used to select Pending data sets. There are various possible outcomes that could occur based on the logic in the algorithms. The following diagram illustrates the portion of the migration transaction lifecycle that pertains to comparison.

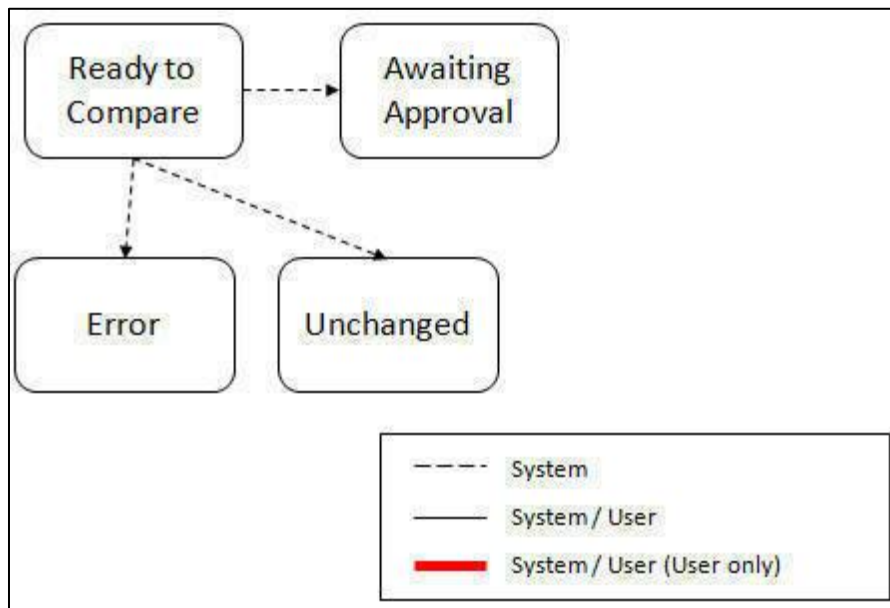


Figure 26: Migration Data Set Import Status Update

The following points describe the lifecycle possible next states after Ready to Compare.

- If any related migration transactions is in the Error Comparing state, the data set transitions to **Error**.
- If all related migration transactions are in the Unchanged state, the data set transitions to **Unchanged**.
- Otherwise, the transaction transitions to **Awaiting Approval**. This means that there are no errors and at least one object is in an “apply-able” state.

The data set remains in the **Awaiting Approval** state until a user decides that the data set and all its records are ready to progress to the Apply step.

Note: You can choose to cancel a data set at any time while it is in progress. For more information, refer to the [Cancelling a Data Set](#) section.

5.3.3 Approval Step

Once the comparison is complete and the data set transitions to the Awaiting Approval state, you need to progress the data set to **Apply Objects** to trigger the Apply step. The following points describe steps you may take during the approval step.

- If the data set configuration for the Default State for Add and Change was set to **Approved**, then any migration object that is determined to be eligible for the Apply step will be in the Approved state. In this situation, you may want to review the data set and its transactions and objects to see verify that the results make sense. At that time, the user is able to move an object to Needs Review or Rejected as appropriate.
- If the data set configuration for the Default State for Add and Change was set to Needs Review for either option, then each migration object in the **Needs Review** state must be reviewed and the user must either Reject or Approve each object before moving to the Apply step.
- If the data set configuration for the Default State for Add and Change was set to **Rejected** for either option, the assumption is that the rejected records don't need to be reviewed. But if a user finds a rejected record that shouldn't be rejected, it may be transitioned to Approved (or Needs Review) as appropriate.

Once the user is comfortable with the data set's results and no more objects are in the Needs Review state, the user should transition the record to **Apply Objects**. This will initiate the Apply step.

Alternatively, if the Automatically Apply flag was set to Yes when creating the import record, the import data set will progress from **Awaiting Approval** to **Apply Objects**. For more information, refer to the [Import Process Summary section](#).

Note: For details about the pages provided to help the user review a data set and its transactions and objects to help in the approval step, refer to the [Maintaining Import Data](#) section.

Note: You can choose to cancel a data set at any time while it is in progress.

5.3.4 Apply Step

The apply step is the step where records in the target environment are added or updated. Like the comparison step, the apply step is actually multiple steps to optimally handle high volume and dependencies between records as smoothly as possible.

Note: For more information about streamlining the various steps in the process, refer to the [Running Batch Jobs](#) section.

Before explaining the apply steps in detail, the following points highlight the type of data that may be included in a given data set.

1. Records that have no foreign keys and therefore no dependencies on other records. Examples: Message, Display Profile.
2. Records that have foreign keys that may already be in the target. Examples: Algorithms for existing algorithm types, To Do Roles for existing To Do Types.
3. Records that have foreign keys that are new records but also part of the migration. CMA detected the relationship at export time and grouped the objects in the same transaction. Example: Script-based Algorithm Type where the script is also in the migration.
4. Records that have foreign keys that are new records but also part of the migration. CMA did not detect the relationship. This may occur if the reference the foreign key is in a XML or parameter column and the migration plan did not include an instruction to explicitly define the relationship. Example, a Zone that references a visibility script.
5. Records that have circular references where both records are new and are part of the migration. CMA detected the relationship at export time and grouped the objects in the same transaction. Example: plug-in Script for a BO plug-in spot. The script references the BO and the BO references an algorithm for the script's algorithm type.

To handle high volume data, the first step in the apply process is to perform the apply logic at the migration object level via a multi-threaded batch job. This should result in all records in categories 1 and 2 above being applied successfully.

For records in categories 3 and 4 above, if a record with a foreign key is added or updated before its related record, the validation will fail. However, if the related record is added first and then the record referring to it is added, validation will pass. Because the migration objects are not ordered, the multi-threaded batch process may not process records in the desired order. To overcome this potential issue, the Apply step has special functionality, described in detail below.

For records in category 5 above, the circular reference will mean that the apply process at the object level will not successfully add or update these records. The apply process at the transaction level will cover these records. This is described in detail below.

1. Apply Objects

Once the Data Set is in the state of **Apply Objects**, the **Migration Object Monitor - Apply** process (F1-MGOAP) runs to attempt to apply the objects. The background process in conjunction with the Apply algorithm have special functionality to ensure records in categories 3 and 4 (above) successfully apply during this step:

- The **Migration Object Monitor - Apply** process is a special one that continually re-selects records in the **Approved** state until there are no more eligible records to process.
- When an error is received in the Apply Object algorithm, the algorithm increments an "iteration count" on the migration object record. If the iteration count does not exceed a maximum count (noted in the algorithm), the object remains in the **Approved** state and is eligible to be picked up for processing again. If the iteration count exceeds the maximum defined in the algorithm, the record transitions to the **Error Applying** state.

Note: When submitting this Apply batch job, be sure to set the number of threads to a number that does not exceed the number of threads supported by the thread pool worker. Doing this will cause the 'excess' threads to wait for the supported number of threads to finish.

The following diagram is the portion of the migration object lifecycle that pertains to the Apply step.

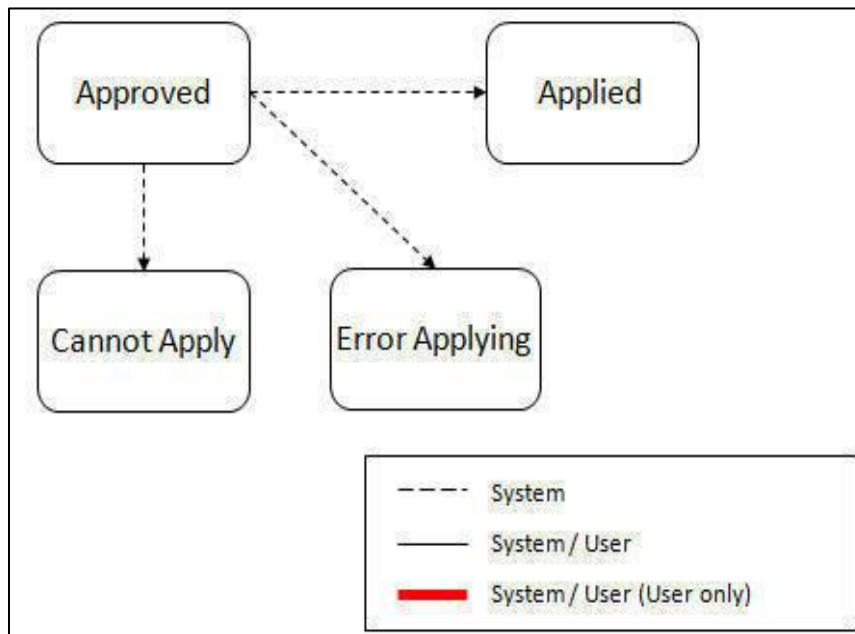


Figure 27: Apply Object

At the completion of the Apply monitor process, typically the objects will be in the **Applied** state or the **Error Applying** state. The records in the Error Applying state are in that state for one of two reasons.

- They are in category 5 described above where the records have a circular reference with another record. For this scenario, the Apply Transactions step described below should successfully apply the records.
- There is some other error that is unrelated to the records in the current migration. In this case, manual intervention may be required. For more information, refer to the [Resolving Errors](#) section.

As shown in the diagram, the Apply Objects algorithm may also detect a reason that the object cannot be applied. This may occur if the object in the target environment has been updated since the comparison

step, making the SQL captured at that point no longer applicable. If this occurs, after the current migration is fully applied, the original file may be imported again, and new comparisons can be generated and applied.

2. Apply Transactions

Ideally, after the Apply Objects step, all the objects are **Applied** or are in **Error Applying** due to the "circular reference" situation. The typical next step is to turn over responsibility to the transactions. The migration transactions can then attempt to apply their objects in bulk.

In order to ensure that multiple background processes are not trying to select migration objects to run the Apply step, the Transactions are only eligible to attempt to "apply my objects" if the Data Set is in the **Apply Transactions** state.

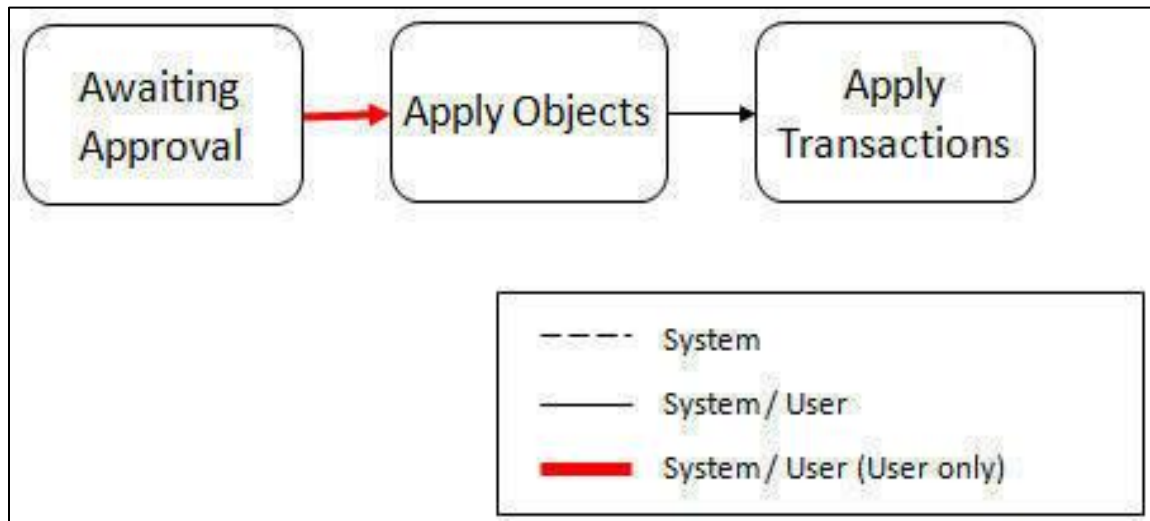


Figure 28: Apply Transactions

A monitor algorithm (executed by the data set monitor batch process) on the Apply Objects state checks to see if all migration objects are no longer **Approved** or the count of records in **Error Applying** does not exceed a configured limit. If so, it automatically transitions the record to the Apply Transactions state.

If the number of objects in **Error Applying** exceeds a configured limit, the monitor algorithm does not automatically transition the record. In that case, you must determine if the large number of errors can be resolved or manually transition to **Apply Transactions** (despite the large number of errors). The Resolving Errors section below describes alternative steps that the user may take if there are errors.

Once the Data Set is in the state of **Apply Transactions**, the **Migration Transaction Monitor - Apply** process (F1-MGTAP) runs. It attempts to apply the transaction's objects. If no migration objects are in error, the migration transaction simply transitions to **Applied**. If any of the migration objects are in **Error Applying**, the background process and the Apply algorithm have special functionality to try to overcome dependencies in migrated objects:

- The Apply algorithm selects all migration objects in error and performs all their SQL, then validates all the records. If there are objects in the transaction with circular references, they should pass validation at this point.

- Because there may still be some dependencies across transactions, similar error handling described in the Apply Objects step occurs here. When an error is received in the Apply Transaction's Object algorithm for any of the objects in the transaction, the algorithm increments an "iteration count" on the migration transaction record. If the iteration count does not exceed a maximum count (noted in the algorithm), the transaction remains in the **Ready to Apply** state and is eligible to be picked up for processing again. If the iteration count exceeds the maximum, the record transitions to the **Error Applying** state. Note that if any objects in the transaction are in error, none of the objects are applied. They all remain in error.
- The **Migration Transaction Monitor - Apply** process is a special one that continually re-selects records in the **Ready to Apply** state until there are no more eligible records to process.

Note: When submitting this Apply batch job, be sure to set the number of threads to a number that does not exceed the number of threads supported by the thread pool worker. Doing this will cause the 'excess' threads to wait for the supported number of threads to finish, erasing the benefit of the iteration processing.

The following diagram is the portion of the migration transaction lifecycle that pertains to the Apply step illustrating the points above.

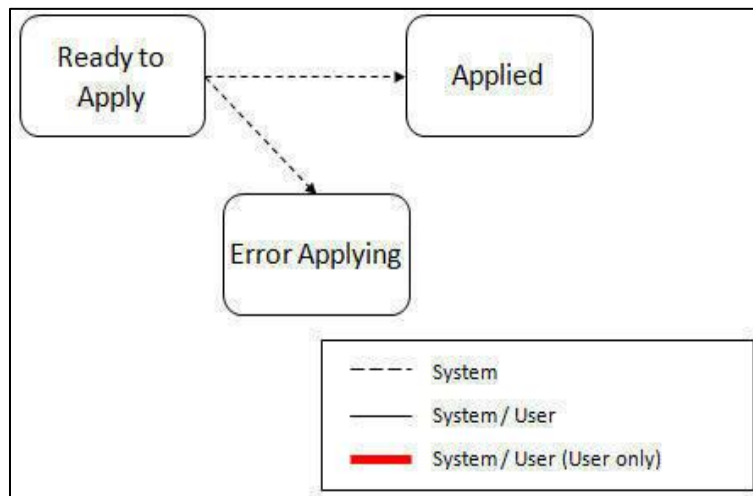


Figure 29: Apply Transactions

If at the end of the transaction level Apply process there are transactions in error (and therefore there are still objects in error), you must review the errors and determine how to fix them. For more information, refer to the [Resolving Errors](#) section.

3. Resolving Errors

As mentioned in the previous sections, errors may be received after the Apply Objects process runs. If the number of records in error is below a certain limit (and the data set monitor batch job is submitted to execute the monitor algorithms) the system will automatically transition the data set to the **Apply Transactions**. If the monitor batch job is not run or if the number of objects in error exceeds a certain

limit, you must make the decision after viewing the errors in the Objects in Error zone on the [Migration Data Set Import](#) portal.

- If the errors appear to be dependency related, the user can decide to let the "transactions apply their objects" and transition the data set to **Apply Transactions**, described above.
- If the errors appear to be related to an outside issue that can be manually resolved, the user may choose to fix the issue and redo the Apply Objects step.
- The user may also decide to reject one or more objects to remove them from the migration.

After the Apply Transactions step, if there are still errors, you must review the records and determine how to proceed. Errors are visible in the **Transactions in Error** zone on the [Migration Data Set Import](#) portal.

- The user may decide to reject one or more objects to remove them from the migration.
- The user may manually resolve an issue external to the migration and then decide to do one of the following:
 - Redo the **Apply Objects** step. This is recommended if there are a large number of Objects still in error and not a large number of dependencies expected. The benefits of running the Apply Objects multi-threaded will ensure that the process runs efficiently.
 - Redo the **Apply Transactions** step.

Because the objects and transactions are in Error Applying, in order to "retry" the Apply step after manually fixing an error, the system needs to move the records back to the state that allows them to be picked up by the appropriate Apply process. For migration objects, records need to be moved back to **Approved**. For migration transactions, records need to be moved back to **Ready to Apply**. The following points describe the Retry logic for migration objects.

- If a user decides to **Retry Objects** (using an action button on the Migration Data Set Import page), the data set transitions to the **Retry Objects** state. At this point the Migration Object monitor must be run.
- The monitor on the **Error Applying** state for the objects detects that the data set is in the state of **Retry Objects** and that triggers the transition back to **Approved**.
- The next step is to transition the data set from **Retry Objects** to **Apply Objects**. This may be done manually or by running the Migration Data Set Import monitor process.
- Now the objects are eligible to be picked up by the object level Apply process.

Analogous logic exists for the migration transactions.

- If a user decides to **Retry Transactions** (using an action button on the Migration Data Set Import page), the data set transitions to the **Retry Transactions** state. At this point the Migration Transaction monitor must be run.
- The monitor on the **Error Applying** state for the transactions detects that the data set is in the state of **Retry Transactions** and that triggers the transition back to **Ready to Apply**.

- The next step is to transition the data set from **Retry Transactions** to **Apply Transactions**. This may be done manually or by running the Migration Data Set Import monitor process.
- Now the transactions are eligible to be picked up by the transaction level Apply process.

The retry logic may also occur when transitioning between the Apply Objects and Apply Transactions depending on whether or not there are errors. The following scenario highlights this point.

- After the **Apply Objects** step there are objects in **Error Applying**. The data set transitions to **Apply Transactions** and the Apply step is done at the transaction level.
- After the Apply Transactions step there are transactions in **Error Applying**.
- User chooses to try to apply objects again (by clicking **Retry Objects**). The steps outlined above for retrying objects are followed at this point.
- After the apply objects, user may choose to retry objects again (after fixing errors if applicable).
- At some point the user will transition to **Apply Transactions** again. If there are transactions in **Error Applying**, the system will automatically transition the data set to **Retry Transactions** and the steps outlined above for retry transactions are followed.

4. Finalize Apply Step

Once all the migration objects for a migration transaction are in a final state (**Applied**, **Rejected** or **Cannot Apply**), the migration transaction transitions to the Applied state. Once all the migration transactions are in the **Applied** state, the Migration Data Set record transitions to the **Completed** and the import is complete.

Note: To review the full lifecycle for each record, refer to the Business Object - Summary tab in the application metadata for the base business objects **Migration Data Set Import** (F1-MigrObjectImport), **Migration Transaction** (F1-MigrTransactionImport) and **Migration Object** (F1-MigrObjectImport).

5.3.5 Adjusting Data Prior to Comparing

Some records may have data that is specific to the environment it is in and won't apply in the target environment. In such cases, an algorithm plugged into the migration plan primary instruction may be used to adjust the data when importing. This algorithm is executed by the comparison algorithm before any comparison is performed. Algorithms of this system event receive the view of the source record (being imported) and the view of the existing record in the target region, if it exists. The data is provided using the physical BO of the migration plan's maintenance object. The algorithm may make changes and pass a new view of the record that should be used for the comparison. This system event supports multiple algorithms that are executed in sequence. Each algorithm receives the original record's data, the target record's data (if applicable) and the 'new' view of the data (as populated by previous algorithms, if any). The final 'new' view of the data is used for the object comparison.

Some examples of records that may require import algorithms.

Batch Control references its next batch sequence number along with snapshot information like the last run date / time. This information is only relevant with respect to its environment. The instruction for a batch control can include an algorithm to not overwrite the batch sequence number when copying a batch control.

Some products include administrative objects that reference a master data object. Master data objects are not copied as part of CMA. An import algorithm may be used to adjust the referenced master data foreign key when importing, for example to reset it (or not overwrite when updating). If the algorithm knows how to find the appropriate master data record to link, that may also be included.

Note that it is possible to use the algorithm to "reset" the source data as a way of indicating that the record should not be imported. For these situations, the migration object comparison step will transition the record to **Unchanged** and will use an object action value of **Canceled**. (Note that object action is a simple lookup value. The record is not transitioned to the **Canceled** BO state as to reserve that status for user initiated cancellations of the object or one of its parent records). This technique not expected to be used often because ideally using appropriate selection criteria at export time should ensure that the only records exported are those that should be imported.

Note: Legacy 'Import' system event. The system originally provided an Import system event / plug-in spot. The purpose of algorithms for this plug-in spot were similar in that they were meant to adjust imported data prior to adding or updating. The algorithms were executed in the Apply step. The logic does not allow for easily interacting with the record using a BO. This makes it difficult to use a plug-in script as the plug-in type. In addition, it is difficult to update elements in an XML column. The support for the plug-in spot will be removed in a future release. Algorithms to adjust the data should be using the pre-compare system event.

5.3.6 Cancelling a Data Set

You may choose to **Cancel** a data set to prevent it from being processed at any point during the process.

If related migration transactions or migration objects have already been created, they will not be canceled as part of the data set getting canceled (due to possible high volumes of related records). They will be canceled the next time an appropriate monitor batch process runs. The child records checks to see if the data set has been canceled prior to any state transition.

5.3.7 Additional Note Regarding Imports

The following points describe miscellaneous comments related to Migration Import.

- CMA relies on the fact that database referential integrity constraints are not in place, and that the SQL statements can be run in any order within the transaction. Any archiving solution that requires referential integrity constraints (such as Information Lifecycle Management) would not be possible on this data. Given that CMA migrations comprise administrative data and not transactional data, this should be a reasonable exception.

- The validation that is performed is only via the **Page Validate** service. BO validation algorithms are not executed. Page validation does not include validation of the business object against the schema (for example, for required fields, field sizes, etc.).
- If multiple migration requests are exported at the same time, on the import side, you should consider importing, reviewing, and applying an entire file/data set before moving on to the next one. The reason is that if objects are included in more than one file, two sets of "inserts" will be generated, but only the first will succeed. The second will cause the object to transition to "Cannot Apply". If instead you wait until the first file is completed before importing the second, the second data set will not generate any SQL for the object, since it has already been inserted. It's a matter of efficiency: If you first import all files and then try to apply all, you'll have to identify the duplicated object as an error and then mark the object as rejected before applying the transaction. This may also be avoided by using a Group migration request to include all objects in one file rather than multiple files.

5.3.8 Caching Considerations

Because CMA updates administrative data that is usually read from a cache, after a successful migration, the target region now has new administrative data which needs to be part of various caches. It is recommended to flush the server cache (which will trigger a 'global' flush of the cache). If the thread pool workers in the target region are configured to refresh their caches when a global flush is requested, then this is the only step required. If not, then the **F1-FLUSH** batch job should also be submitted to refresh the caches used in batch processing.

5.4 Maintaining Import Data

This section describes the portals provided to add, view and maintain migration import data.

5.4.1 Migration Data Set Import

Use the Migration Data Set Import portal to view and maintain migration data set import records. For an overview of the import process, refer to the [Importing and Applying a Migration](#) section. This zone contains display-only information about the selected record.

5.4.1.1 Creating Migration Data Set Import

To define Migration Data Set Import, navigate using **Admin > M > Migration Data Set Import > Add**.

The Migration Data Set Import window appears.

Figure 30: Migration Data Set Import

The import process starts with verifying the import directory configured as described in the topic [Master Configuration](#) and ensuring that the exported file is located in that directory. Then, in the target environment, a Migration Data Set Import record should be created.

Enter the field values:

1. In File Name, type a unique name.
2. Set the Default Status For Add.
3. Set the Default Status for Change.
4. Click Save. The Migration Data Set Import status window appears.

5.4.1.2 Searching Migration Data Set Import

To search Migration Data Set Import, navigate using **Admin > M > Migration Data Set Import > Search**. The Migration Data Set Import Query window appears. It contains two modes of operation: Migration Data Set Import Search and Migration Data Set Objects in Error

Figure 31: Migration Data Set Import Query

Migration Data Set Import Search

Figure 32: Migration Data Set Import Search

To search using Migration Data Set Import Search


1. In Migration File Name, enter the file name.
2. Select status from Status drop-down list.
3. Enter User Id within Created By User Id.
4. In Export Description, enter description.
5. Enter Source Environment Reference.
6. Click **Search**. The Migration Data Set Import appears in the Filter section.


Migration Data Set Objects in Error

This zone is only visible if there are objects for this data set in a non-final status that have errors. It indicates the error for each object. You may use this zone to review errors after the monitor batch job to apply objects completes. Using the error information shown, you can choose to drill into the record to transition it to **Error Applying** or choose to manually fix the cause of the errors and click **Retry Objects**. You may also choose to select one or more records to **Reject**.

Figure 33: Migration Data Set in Error

Click **Search**. The Migration Data Set Imports appear in the Filter section.

By default, the Search zone is visible. You can hide the Search zone and view only the Filters by clicking the **Filters** () icon in the upper right corner.

Click **Add** () icon in the upper right corner of the Search zone to define new Migration Data Set Import.

Note: For more information about resolving errors, refer to the [Apply Step](#) section.

The following zones can also be used while working with Migration Data Set Import

- **Migration Data Set Transactions** - This zone is visible once the [Import Step](#) has occurred and lists all the transactions that are related to the data set. To see more information about a specific migration transaction, click the hypertext for its ID. This brings you to the [Migration Transaction](#) portal.
- **Migration Data Set Impacted Object Summary** - This zone is visible once the [Import Step](#) has occurred and lists the objects that are related to the data set. To see more information about a specific migration object, click the hypertext for its ID. This brings you to the [Migration Object](#) portal. You may choose to update the status of one or more records by checking the records and clicking **Approve**, **Reject** or **Needs Review** accordingly.
- **Migration Data Set Transactions in Error** - This zone is only visible if there are transactions for this data set in a non-final state that have errors. It indicates the error for each transaction. You may use this zone to review errors after the monitor batch job to apply transactions completes. The errors received when attempting to apply objects at the transaction level may differ from those received when attempting to apply objects at the object level. A transaction log is created for each object error received and these exceptions are shown in this zone.

Note: For more information about resolving errors, refer to the [Apply Step](#) section.

5.4.2 Migration Transaction Portal

This page appears after drilling into a specific migration transaction from the migration data set portal or from the migration object portal.

For more information on the import process, refer to the [Importing and Applying a Migration](#) section.

The following zones are visible on the main tab:

- **Migration Transaction** - This zone contains display-only information about the selected record. Please see the zone's help text for information about this zone's fields.
- **Migration Transaction Objects** - This zone lists the objects that are related to the data set. To see more information about a specific migration object, click the hypertext for its ID. This brings you to the Migration Object portal. You may choose to update the status of one or more records by checking the records and clicking **Approve**, **Reject** or **Needs Review** accordingly.

5.4.3 Migration Object Portal

This page appears after drilling into a specific migration object from the migration data set portal or from the migration transaction portal.

For an overview of the import process, refer to the [Importing and Applying a Migration](#) section.

The **Migration Object** zone contains display-only information about the selected record. Please see the zone's help text for information about this zone's fields.

5.5 Running Batch Jobs

There are several batch jobs that are part of the CMA process, especially the import step (which are highlighted in [Import Process Summary](#)). And in some cases, a single batch jobs may process multiple states in the same business object lifecycle. Implementations must decide the best way to manage the batch job submission depending on how they plan to work.

- **Batch scheduler** - If an implementation wishes to put these batch jobs in the batch scheduler, a given job may need to be included several times to manage progressing the records to completion.
- **Timed Batches** - The batch controls can be configured as timed batches so that they run every N minutes based on the setting. This allows for the batch jobs to run periodically and process whatever is ready. You do not have to manually submit a batch request. Navigate to the Batch Control page and select the appropriate batch controls. For each one, change the Batch Control Type to Timed. Fill in the additional information that appears for timed batches.
- **Event Driven** - The system provides BO enter plug-in algorithms and batch control post processing plug-in algorithms that automatically submit the appropriate next batch job for that step in the process. This allows for as much automation as possible for the steps that don't require user input. Note that configuration is required because the BOs / batch controls are not configured for this

scenario by default. The following table highlights the BO and status where an algorithm may be plugged in and the name of the algorithm to use.

Business Object	Status	Algorithm
Migration Data Set Export	Pending	F1-MGDPR-SJ (Submit Migration Data Set Export Monitor).
Migration Data Set Import	Pending	F1-MGDIM-SJ (Submit Migration Data Set Import Monitor).
	Ready To Compare, Retry Objects	F1-MGOPR-SJ (Submit Migration Object Monitor).
	Apply Objects	F1-MGOAP-SJ (Submit Migration Object Apply Monitor).
	Apply Transactions	F1-MGTAP-SJ (Submit Migration Transaction Apply Monitor).
	Retry Transactions	F1-MGTPR-SJ (Submit Migration Transaction Monitor).

The following table highlights the batch controls where an algorithm may be plugged in and the name of the algorithm to use.

Batch Control	Algorithm
F1-MGTPR (Migration Transaction Monitor)	F1-MGDIM-NJ (Submit Migration Data Set Import Monitor).
F1-MGTAP (Migration Transaction Monitor - Apply)	F1-MGDIM-NJ (Submit Migration Data Set Import Monitor).
F1-MGOAP (Migration Object Monitor - Apply)	F1-MGDIM-NJ (Submit Migration Data Set Import Monitor).
F1-MGOAP (Migration Object Monitor - Apply)	F1-MGTPR-NJ (Submit Migration Transaction Monitor).

- Manual submission. The user managing the CMA import process submits the appropriate batch jobs on demand when a particular step is ready. Navigate to Batch Job Submission, select the appropriate batch control and fill in the parameters as needed.

Note that after successfully applying the migrated data, the L2 cache of all thread pools must be refreshed. For more information, refer to the [Caching Considerations](#) section.

Caution: Be sure that the Thread Count set when submitting the batch job does not exceed the number supported by the thread pool. Otherwise the extra threads will wait until the supported number of threads are finished, possibly resulting in a large number of errors in the Apply steps.

5.6 CMA Reference

This section provides additional reference information.

5.6.1 Framework-Provided Migration Configuration

This topic describes special information relating to migration objects provided for use by CMA in the product. Additional objects may be provided by your specific product. Any special information for objects is provided separately in each product's documentation.

The following points highlight some information about Framework-provided migration requests. Navigate to the migration request page in the application to view the details of all provided objects.

- Several base migration requests are supplied to logically group system and administrative tables. For example, there is a migration request for Framework System Configuration **F1-SystemConfig** where most system configuration objects are included. There is another one provided for CMA related configuration objects.
- There are several different security related migration requests that include different combinations of migration plans to support multiple possible business requirements related to security migration.
- The system supplies a group migration request **F1-FrameworkConfig** (Framework Configuration), which includes several other migration requests. The expectation is that this migration request includes all the typical objects that are included in a wholesale migration. Your specific product may include this migration request into its own group migration request to support a wholesale migration of all the framework and product administrative tables. An implementation may choose to build a custom group migration request. In this case, review the various migration requests provided by base to see if any may be included as components for the custom migration request. Then any new migration plans added to the base migration request in future releases are automatically included in future migrations.

Note: Refer to your specific product's CMA documentation for its recommendation on which migration requests to use for a full migration of framework and product administrative tables.

The following points highlight some information about the Framework-provided migration plans. Navigate to the migration plan page in the application to view the details of all provided objects.

- Fields and characteristic types are not migrated with an object (like a business object or a data area) unless specifically indicated.
- The **Application Service** used by an object is migrated only if it is CM-owned.
- The **Batch Control** object optionally references a User. If this user does not exist on the target system, CMA cannot apply the requested changes. Also note that when running a batch job, snapshot information is captured on the batch control. Updates like this increment the version number. If a batch control record is part of the migration and the comparison step has detected

a change to the batch control, the Apply step will error out for this batch control if a batch job is submitted between the compare and apply step.

Note: CMA batch controls that are part of the import step are executing and as such, the system does not include these records in a migration. If your implementation changes default parameters for any of the batch controls, the recommendation is to manually make those changes to the target region.

- The base migration plans for MO and BO include instructions to copy option types that use foreign key references to refer to other objects. Note that the data stored in the options are not validated, so defining these instructions is not required when doing wholesale migrations. However, including subordinate instructions for foreign key references is useful for targeted migrations to ensure that the related data is included in the migration. If you add additional MO or BO option types that use foreign keys and you want to support targeted migrations, you must create custom migration plans and requests for MO and BO, respectively to include these referenced objects in the migration plan. Note that you do not need to duplicate the instructions in the base migration plans. You may define the additional migration plans to only have the additional custom option types. When submitting a migration request for MO or BO you must include both the base migration plans and the custom migration plans in the request.
- For scripts, schema-based objects and zones, the migration plans provided by the product migrate, through constraints, some of the typical associated data with them. However, data specified through alternate formats (such as through **Edit Data** steps in scripts, referenced in schemas for schema-based objects, or data from mnemonics in zone parameters, etc.) are not identified and combined in the same transaction. The iterative processing functionality of the import step should resolve any timing issues that may result in validation errors for these types of objects.
- There are two migration plans for **Scripts**. The migration plan **F1-ScriptOnly** migrates just the script and its **Application Service** (provided the Application Service is CM-owned). The migration plan **F1-Script** includes most related objects, but does not migrate any objects referenced in the edit data area steps. It does not move the **Function** maintenance object. It may be included in any appropriate custom targeted migration request where scripts and related data should be migrated.
- If your implementation includes a **Feature Configuration** setting for the **F1_DBCONINFO** entry that will be included in a migration request, be sure that the import user on the target region has the appropriate security rights to this entry (**Administrator** access mode for the Feature Configuration application service (**CILTWSDP**)).
- The common attachments in the Attachment maintenance object may be considered administrative data to include in a migration. Because this MO has a system generated key, as described in [Migration Assumptions, Restrictions and Recommendations](#), it uses a logical key of the file name and the creation date to determine if the record exists in the target environment. In addition, this MO contains admin data (common attachments) and non-admin data (owned attachments). To try to minimize the possibility of key “collision”, new common attachments

receive a generated key that includes a zero in the middle whereas owned attachments receive a generated key that does not have a zero in the middle.

- The Menu maintenance object has a user defined key, however, its menu lines and menu items have system generated keys. To avoid the possibility of overriding a menu line or menu item incorrectly, the menu MO will check the menu line's menu name in the source and target to be sure they match and will check the menu item's menu line in the source and target to be sure they match otherwise an error will be issued in the comparison step.
- For the system messages, the product provides three different migration plans.
 - Message Category and its Messages (F1-MessageCategory) - This migration plan is included in the F1-SystemConfig migration request.
 - Message Category (F1-MessageCategoryOnly) - This migration plan is provided to support a targeted migration where an implementation has created a custom message category and wants to move it but doesn't want to move all its messages.
 - Message (F1-Message) - This migration plan is provided to support a targeted migration where only specific messages within a message category should be migrated.
- For lookup values, the product provides two different migration plans.
 - Lookup Field and its Values (F1-Lookup) - This migration plan is included in the **F1-SystemConfig** migration request.
 - Lookup Value (F1-LookupValue) - This migration plan is provided to support a targeted migration where only specific lookup values within a lookup field should be migrated.
- There are some system data objects where no information in a base delivered record may be modified by an implementation. For these records, the base delivered migration requests include selection criteria to only select CM-owned records (because the base records will always exist in the target region assuming both regions have the same release). An example is Algorithm Type. The **F1-SystemConfig** migration request only includes CM-owned algorithm types. However, many system data objects support custom changes to one or more fields, for example the Zone object allows an implementation to override the zone text or certain parameters. Other system data objects support custom additions to a collection. For example, the Maintenance Object allows an implementation to add algorithms or options. For the migration plans related to these system data objects, all records are included in the base delivered migration requests to allow for any customized configuration to be migrated. It means that during the Import / Compare step many base delivered objects that are not customized will be marked **Unchanged**.
- Many of the integration related maintenance objects that include references to environment-specific data, such as Message Senders. This data should be migrated with extreme care. When appropriate, consider taking advantage of URI Substitution.

6. Import Process Summary

The following table summarizes the steps required to complete the import process from start to finish. Note that this section is only a summary and assumes that you are familiar with the details described in the previous sections. It highlights what steps are manual and what steps are performed by a batch monitor process. For each step, the table highlights the Next Action sequence that would occur. For the Apply steps, there are two parts where multiple next actions are possible based on whether there are errors and the user's decision on how to resolve the error. For more information, refer to the [Resolving Errors](#) section. The possible next actions have the same sequence with a letter following the sequence highlighting the action to take based on the results of the previous step.

Note: When running the Apply batch jobs, be sure to set the number of threads to a number that does not exceed the number of threads supported by the thread pool worker.

Also note that a sequence and action marked in bold is considered the "normal path".

Step	Seq	Action	Manual / Batch	Portal Action	Batch Control	Record Impacted - Resulting Status	Next Action Sequence
Import	10	Create record Import	Manual	Migration Data Set Import - Add		Migration Data Set Import - Pending	11
	11	Import file	Batch		F1-MGDIM - Migration Data Set Import Monitor	Migration Data Set Import - Ready to Compare Migration Transaction - Pending Migration Object - Pending	20
Compare	20	Migration Object Compare	Batch		F1-MGOPR - Migration Object Monitor	Migration Object - Approved, Needs Review, Rejected, Unchanged or Error Comparing	21
	21	Migration Transaction status update	Batch		F1-MGTPR - Migration Transaction Monitor	Migration Transaction - Ready to Apply, Unchanged or Error Comparing	22

Step	Seq	Action	Manual / Batch	Portal Action	Batch Control	Record Impacted - Resulting Status	Next Action Sequence
	22	Migration Data Set Import status update	Batch		F1-MGDIM - Migration Data Set Import Monitor	Migration Data Set Import - Awaiting Approval, Apply Objects (if configured for Automatic Apply), Unchanged or Error	30
Approval	30	Review comparison results, approve / reject as needed (This step is skipped if the data set is configured for Automatic Apply)	Manual	Migration Data Set Import, drill in to the Transactions / Objects as necessary.		Migration Object - Approved or Rejected (no records should be in Needs Review) Migration Data Set Import - Apply Objects	40
Apply	40	Apply Objects	Batch		F1-MGOAP - Migration Object Monitor - Apply	Migration Object - Applied or Error Applying	41 Appropriate next action is based on error review, if applicable.
	41a	Migration Data Set Import status update - auto transition to Apply Transactions. Only applicable if the number of migration objects in Error Applying is below a threshold	Batch		F1-MGDIM - Migration Data Set Import Monitor	Migration Data Set Import - Apply Transactions	42
	41b	Migration Data Set Import status update - manual transition to Apply Transactions.	Manual	Migration Data Set Import - click Apply Transactions			

Step	Seq	Action	Manual / Batch	Portal Action	Batch Control	Record Impacted - Resulting Status	Next Action Sequence
		Occurs if user reviews errors and determines that they may be resolved in the Apply Transaction step.					
		System detects that all the transactions are in the Ready to Apply state and proceeds to the Apply Transactions state.				Migration Data Set Import - Apply Transactions	42
		System detects that there are transactions in the Error Applying state and transitions instead to the Retry Transactions state.				Migration Data Set Import - Retry Transactions	45
	41c	Migration Data Set Import status update - manual transition to Retry Objects. Occurs if user reviews errors and decides to fix an external error and wants to try the batch level Apply again at the object level.	Manual	Migration Data Set Import - click Retry Objects		Migration Data Set Import - Retry Objects	44
	42	Apply Transactions	Batch		F1-MGTAP - Migration Transaction	Migration Transaction - Applied or Error Applying	43 Appropriate next action

Step	Seq	Action	Manual / Batch	Portal Action	Batch Control	Record Impacted - Resulting Status	Next Action Sequence
					Monitor - Apply	Migration Object - Applied or Error Applying	is based on error review, if applicable.
	43a	Migration Data Set Import status update - auto transition to Completed Applicable if all transactions are Applied	Batch		F1-MGDIM - Migration Data Set Import Monitor	Migration Data Set Import - Completed.	N/A
	43b	Migration Data Set Import status update - manual transition to Retry Objects. Occurs if user reviews errors and decides to fix an external error and wants to try the batch level Apply again at the Object level.	Manual	Migration Data Set Import - click Retry Objects		Migration Data Set Import - Retry Objects	44
	43c	Migration Data Set Import status update - manual transition to Retry Transactions. Occurs if user reviews errors and decides to fix an external error and wants to try the batch level Apply	Manual	Migration Data Set Import - click Retry Transactions		Migration Data Set Import - Retry Transactions	45

Step	Seq	Action	Manual / Batch	Portal Action	Batch Control	Record Impacted - Resulting Status	Next Action Sequence
		again at the Transaction level.					
	44	Migration Objects status update from Error Applying back to Approved. Occurs if user chose to Retry Objects.	Batch		F1-MGOPR - Migration Object Monitor	Migration Object - Approved	
		Migration Data Set Import status update from Retry Objects to Apply Objects	Batch		F1-MGDIM - Migration Data Set Import Monitor	Migration Data Set Import - Apply Objects	40
	45	Migration Transactions status update from Error Applying back to Ready to Apply. Occurs if user chose to Retry Transactions or if the user transitions to Apply Transactions and the system detects that there are Transactions in the Error Applying state.	Batch		F1-MGTPR - Migration Transaction Monitor	Migration Transaction - Ready to Apply	42
		Migration Data Set Import status update from Retry Objects to Apply Objects	Batch		F1-MGDIM - Migration Data Set Import Monitor	Migration Data Set Import - Apply Transactions	42

The following table summarizes the batch monitor jobs that are used in the import process. You can see that there are special monitor processes for the Apply step for both the Object and Transaction. However, for all other states that have monitor logic, the standard monitor process for that MO is used.

Batch Control	Description	Comments
F1-MGDIM	Migration Data Set Import Monitor	Processes data set records in the following states: <ul style="list-style-type: none"> • Pending • Ready to Compare • Apply Objects • Retry Objects • Apply Transactions • Retry Transactions
F1-MGTPR	Migration Transaction Monitor (Deferred)	Processes transaction records in the following states: <ul style="list-style-type: none"> • Pending • Error Applying
F1-MGTAP	Migration Transaction Monitor - Apply	Processes transaction records in the Ready to Apply state where the data set is in the Apply Transactions or Canceled state. <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p>Note: Be sure to set the number of threads to a number that does not exceed the number of threads supported by the thread pool worker.</p> </div>
F1-MGOPR	Migration Object Monitor	Processes object records in the following states: <ul style="list-style-type: none"> • Pending • Needs Review (check for Data Set cancellation) • Rejected (check for Data Set cancellation) • Error Applying
F1-MGOAP	Migration Object Monitor - Apply	Processes object records in the Approved state where the data set is in the Apply Objects or Canceled state. <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p>Note: Be sure to set the number of threads to a number that does not exceed the number of threads supported by the thread pool worker.</p> </div>

For more information about managing the batch jobs, including ways to automate the above steps, refer to the [Running Batch Jobs](#) section.

Appendix A: List of Configuration Objects Bundled With the Application – Admin Menu

The following table provides list of migration plans and migration requests which are part of the Configuration Migration Assistant which comes with Oracle Revenue Management and Billing application. These are configured under **Admin** menu.

Sr. No.	Description	Migration Plan	Migration Request
1.	A/P Request Type	CI-APREQ-TYPE	CI-APREQ-TYPE
2.	Account Identifier Type	C1-ACCNBRTYP	C1-ACCNBRTYP
3.	Account Management Group	C1-ACCT-MGMT-GR	C1-ACCT-MGMT-GR
4.	Account Relation Type	C1-ACCT-REL-TYPE	C1-ACCT-REL-TYPE
5.	Account Identifier Type	C1-ACCNBRTYP	C1-ACCNBRTYP
6.	Accounting Calendar	C1-ACCT-CAL	C1-ACCT-CAL
7.	Activity Type	C1-ACM-ACTTY	C1-ACM-ACTTY
8.	Adjustment Cancel Reason	C1-ADJ-CAN-RSN	C1-ADJ-CAN-RSN
9.	Adjustment Type	C1-AdjustmentType	C1-AdjustmentType
10.	Adjustment Type Profile	C1-ADJ-TYPE-PRF	C1-ADJ-TYPE-PRF
11.	Alert Type	C1-ALERT-TYPE	C1-ALERT-TYPE
12.	Approval Profile	C1-ADJ-APPR-PROF	C1-ADJ-APPR-PROF
13.	Approval Workflow Chain	C1-APP-CHAIN	C1-APP-CHAIN
14.	Approval workflow Criteria Type	C1-APP-CRIT	C1-APP-CRIT
15.	Approval Workflow Group	C1-APPGRP	C1-APPGRP
16.	Approval Workflow Group Chain Linkage	C1-APP-BOCHN	C1-APP-BOCHN
17.	Approval Workflow Reason	C1_APPREASON	C1_APPREASON
18.	Approval Workflow Setting	C1-APP-CH-EL	C1-APP-CH-EL
19.	Auto Pay Route Type	C1-APAY-RT-TYPE	C1-APAY-RT-TYPE
20.	Auto Pay Source Type	C1-AUTOPAY-SRC	C1-AUTOPAY-SRC
21.	Auto Payment Upload Reason	CI-APYUPLRSN	CI-APYUPLRSN
22.	Bank	C1-BANK	C1-BANK

23.	Bill Cancel Reason	C1-BILL-CAN-RSN	C1-BILL-CAN-RSN
24.	Bill Charge Line Type	C1-BCHG-UP-TYPE	C1-BCHG-UP-TYPE
25.	Bill Cycle	C1-BILL-CYCLE	C1-BILL-CYCLE
26.	Bill Message	C1-BILL-MESSAGE	C1-BILL-MESSAGE
27.	Bill Period	C1-BILL-PERIOD	C1-BILL-PERIOD
28.	Bill Route Type	C1-BILL-RT-TYPE	C1-BILL-RT-TYPE
29.	Bill Segment Type	C1-BILL-SEG-TYP	C1-BILL-SEG-TYP
30.	Billable Charge Template	C1-BILL-CH-TMPL	C1-BILL-CH-TMPL
31.	Case Type	C1-CASE-TYPE	C1-CASE-TYPE
32.	Charge Type	C1-CHGTYPE	C1-CHGTYPE
33.	Collection Agency	C1-COLL-AGENCY	C1-COLL-AGENCY
34.	Collection Class	C1-COLL-CLASS	C1-COLL-CLASS
35.	Collection Class Control	C1-COLL-CL-CNTL	C1-COLL-CL-CNTL
36.	Collection Class Overdue Rules	C1-OD-RULE	C1-OD-RULE
37.	Collection Event Type	C1-COLL-EVT-TYP	C1-COLL-EVT-TYP
38.	Collection Class Template	C1-COLL-PROC-TM	C1-COLL-PROC-TM
39.	Collection Type	C1-COLLECT	C1-COLLECT
40.	Contract Quantity Type	C1-CONT-QTY-TYP	C1-CONT-QTY-TYP
41.	Contract Relation Type	C1-SATY-SARELTY	C1-SATY-SARELTY
42.	Contract Type	C1-SA-TYPE	C1-SA-TYPE
43.	Contract Type - Charge Type mapping	C1-CTypeSALink	C1-CTypeSALink
44.	Contract Type - Pay Plan template mapping	C1-PayPlanTmplCompl	C1-PayPlanTmplCompl
45.	Credit Unit	C1-CREDIT-UNIT	C1-CREDIT-UNIT
46.	Customer Class	C1-CUST-CLASS	C1-CUST-CLASS
47.	Customer Contact Class	C1-CUST-CONT-CLASS	C1-CUST-CONT-CLASS
48.	Customer Contact Type	C1-CUST-CNT-TYP	C1-CUST-CNT-TYP
49.	Debt Class	C1-DEBT-CLASS	C1-DEBT-CLASS
50.	Deposit Class	C1-DEPOSIT-CL	C1-DEPOSIT-CL
51.	Distribution code	C1-DISTR-CODE	C1-DISTR-CODE

52.	Distribution Rule	C1-DST-RULE	C1-DST-RULE
53.	File Type	C1-UploadFileType	C1-UploadFileType
54.	Frequency	C1-FREQUENCY	C1-FREQUENCY
55.	Fund	FUND	FUND
56.	Funding Request Type	C1-FundingRequestType	C1-FundingRequestType
57.	General Division	C1-GL-DIVISION	C1-GL-DIVISION
58..	Hold Request Type	C1-HoldRequestType	C1-HoldRequestType
59.	Invoice Frequency	C1-PolicyInvoiceFrequency	C1-PolicyInvoiceFrequency
60.	Letter Template	C1-LETTER-TMPL	C1-LETTER-TMPL
61.	Match Event Cancel Reason	C1-MatchEvtCanRsn	C1-MatchEvtCanRsn
62.	Match Type	C1-MATCHTYPE	C1-MATCHTYPE
63.	Non Cash Deposit Type	C1-NCDTYPE	C1-NCDTYPE
64.	Offset Request Type	C1-OFFSTRQTY	C1-OFFSTRQTY
65.	Order Cancel Reason	C1-OrderCanReason	C1-OrderCanReason
66.	Order Hold Reason	C1-OrderHoldReason	C1-OrderHoldReason
67.	Overdue Event Cancel Reason	C1-OE-CNREAS	C1-OE-CNREAS
68.	Overdue Event Type	C1-OD-EVTYPE	C1-OD-EVTYPE
69.	Overdue Process Template	C1-ODPROC-TM	C1-ODPROC-TM
70.	Parameter	C1_PRICE_PARM	C1_PRICE_PARM
71.	Payment Cancel Reason	C1-PayCanRsn	C1-PayCanRsn
72.	Pay Plan Template	C1-PPTMPL	C1-PPTMPL
73.	Payment Request Type	C1-PAYRQTYPE	C1-PAYRQTYPE
74.	Payment Segment Type	C1-PaySegType	C1-PaySegType
75.	Payment Template	C1-NonCISPayTemplate	C1-NonCISPayTemplate
76.	Person Identifier Type	C1-IDType	C1-IDType
77.	Person Relationship Type	C1-PerRelType	C1-PerRelType
78.	Policy Cancelation Reason	C1-POLICYCAN	C1-POLICYCAN
79	Policy Type	CI_POLICYTYPE	CI_POLICYTYPE

80.	Profile	C1-Profile	C1-Profile
81.	Promise to Pay Cancel Reason	C1_PP_CAN	C1_PP_CAN
82.	Promise to Pay type	C1-PAYPLAN	C1-PAYPLAN
83.	Proposal Contract Decline Reason	C1-PropDclRsn	C1-PropDclRsn
84.	Quote Request type	C1-QteRteType	C1-QteRteType
85.	Reason code	C1-Variance	C1-Variance
86.	Reconciliation Object Line Status	C1-ReconDtlStat	C1-ReconDtlStat
87.	Refund / Write off Request Type	C1-RFWORQTY	C1-RFWORQTY
88.	Register Rule	C1-RegRule	C1-RegRule
89.	Revenue Class	C1-RevGL	C1-RevGL
90.	Rule	C1-Rule	C1-Rule
91.	Rule Type	C1-RULETYPE	C1-RULETYPE
92.	SC Membership Inactive reason	C1-ScmInctvRs	C1-ScmInctvRs
93.	Service Credit Event Type	C1-ScEvtType	C1-ScEvtType
94.	Service Credit Membership Type	C1-ScmType	C1-ScmType
95.	Service Quantity Identifier	C1-SQI	C1-SQI
96.	Service Quantity Rule	C1-SQRule	C1-SQRule
97.	Service Type	C1-SVCTYPE	C1-SVCTYPE
98.	SIC Code	C1-SICCD	C1-SICCD
99.	Statement Cycle	C1-STMCycle	C1-STMCycle
100.	Statement Route Type	C1_STM_RTE_TYPE	C1_STM_RTE_TYPE
101.	Tax Exempt Type	C1-TaxExType	C1-TaxExType
102.	Tender Source	C1-TSRCE	C1-TSRCE
103.	Tender Type	C1-CreditCardWithAuthorization	C1-CreditCardWithAuthorization
104.	Terms and Conditions	C1-TermNCond	C1-TermNCond
105.	Transaction Aggregation Rule	C1-TXNSQIFRAG	C1-TXNSQIFRAG

106.	Transaction Details	C1-TXNTFMDTL	C1-TXNTFMDTL
107.	Transaction Record Type	C1-TXNSOURCE	C1-TXNSOURCE
108.	Transaction Source	C1-TxnSource	C1-TxnSource
109.	UOM	C1-UOM	C1-UOM
110.	Upload Request Type	C1-UPLREQTYP	C1-UPLREQTYP
111.	Variance Parameter	C1-TOU	C1-TOU
112.	Workflow Event Type	C1-WFProc	C1-WFProc
113.	Workflow Process Profile	C1-WFProcProf	C1-WFProcProf
114.	Write off Control	C1-WOCntl	C1-WOCntl
115.	Write off Debt Class	C1-WODEbtClass	C1-WODEbtClass
116.	Write off Event Type	C1-WOEvtType	C1-WOEvtType
117.	Write off Process template	C1-WOProcTmpl	C1-WOProcTmpl

Appendix B: List of Configuration Objects Bundled With the Application – Main Menu

The following table provides list of migration plans and migration requests which are part of the Configuration Migration Assistant which comes with Oracle Revenue Management and Billing application. These are configured under **Main** menu.

Sr. No.	Description	Migration Plan	Migration Request
1.	Rate Schedule	C1-RateSchedule	C1-RateSchedule
2.	Manage Rate Definition	C1_RC_MAP	C1_RC_MAP
3.	Bill Factor	C1-BILLFACTOR	C1-BILLFACTOR
4.	Bill Factor Value	C1-BFVALUE	C1-BFVALUE
5.	Price Assignments	C1-PriceAssignments	C1-PriceAssignments
6.	Price Item	C1_PRICEITEM	C1_PRICEITEM
7.	Price List	C1_PRICELIST	C1_PRICELIST