ARIS RISK & COMPLIANCE MANAGER

DATA MIGRATION GUIDE FOR CUSTOMIZED VERSIONS

VERSION 10.0 - SERVICE RELEASE 12

April 2020

This document applies to ARIS Risk & Compliance Manager Version 10.0 and to all subsequent releases.

Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

Contents

1 Text conventions

Menu items, file names, etc. are indicated in texts as follows:

- Menu items, keyboard shortcuts, dialogs, file names, entries, etc. are shown in **bold**.
- Content input that you specify is shown in as **<bold text in angle brackets>**.
- Example texts that are too long to fit on a single line, such as a long directory path, are wrapped to the next line by using ↵ at the end of the line.
- File extracts are shown in the following font:

  `This paragraph contains a file extract.`
- Warnings have a colored background:

Warning

This paragraph contains a warning.

2 Introduction

As of version 4.0 of ARIS Risk & Compliance Manager the server has a migration framework for the incremental migration of data from previous versions. This document provides you with an introduction into the handling, operation and extension possibilities of this framework. It is oriented towards all developers who adapt ARIS Risk & Compliance Manager to specific customer requirements and are responsible for data migration.

DATA MIGRATION FOR STANDARD VERSIONS

This guide describes the data migration procedure for a customer-specific version of ARIS Risk & Compliance Manager. For detailed information on data migration for a standard version, refer to the **ARCM Upgrade Guide** chapter **ARIS Risk & Compliance Manager database migration**.

3 The migration framework of ARIS Risk & Compliance Manager

The server has a mechanism for the incremental migration of data from previous versions. With it, data can be migrated in all database systems (Oracle, MSSQL and PostgreSQL) supported by ARIS Risk & Compliance Manager. In addition, a portfolio of high-level API functions is available. These functions can be used within migration steps that you can write or change yourself to add tables and fields and to adapt data to new requirements.

The framework can be extended with Java classes. These must be available in text form. A compilation is not required. The internal migration logic determines the steps required to adapt data structures and data on the currently started server using an individually customizable XML migration plan.

Warning

In order to prevent data loss and irreversible changes to your data, you are recommended that you perform a full backup of all your data. To do so, use the administrative tool from your database system.

3.1 What the framework cannot do

The internal migration framework only processes data and its structures. This is sufficient for migrating the standard version of ARIS Risk & Compliance Manager. However, it cannot adapt internal logic such as rules or workflows. This means for customer-specific adapted versions that in addition to the data migration mentioned here, the logic may have to be adapted.

4 Start the migration

From version 10.0.6, it is checked at server start whether the schema version in the connected database coincides with the current schema version of the ARCM server. If, during this check, the server detects that the server version does not coincide with the database version it tries to generate an appropriate migration strategy using the migration plan (Page 5). For detailed information about data migration with different database management systems, refer to the **ARCM – Upgrade Guide**, chapter **Data migration (version 10.0 or newer")**.

Now a productive environment can be generated via a database export and import. To do so, create a new schema in the target database and generate the required tables by starting the ARIS Risk & Compliance Manager Server. A database import can also be imported into another DBMS (database management system). It is possible, for example, to go live on an MSSQL server with a database migrated under Oracle.

5 The migration plan

The migration plan is an XML file with the name **migrationPlan.xml**. Here a version transition is assigned to a directory in which the corresponding migration logic is located.

5.1 Format

A version transition with the following attributes is defined in the **migration** tag:

- **Name**: contains the name of the version transition. This name is entered into the **A_SCHEMAPROPERTY_TBL** table when the version transition is carried out. Later, the migration history of the database can be tracked with this information.
- **Source**: contains the start version located in the **A_SCHEMAPROPERTY_TBL** table of the connected database system. If the start version is not specified, the **start** value can be entered here. In this case, if an appropriate source to a schema is not found, the migration is started with the version transition marked as **start** in the plan.
- **Target**: contains the target version to be reached after the current version transition.
- **Approach**: Outputs whether the current version transition refers to a specific approach. Up to version 10.0.10.0 there was a distinction of the risk-based approach (rba) and the control-based approach (cba).

  From version 10.0.12.0 only a unified approach is supported in ARIS Risk & Compliance Manager. To ensure downward compatibility, the **approach** attribute must be set to **unified**, which is the only valid attribute value. See the XML example below.
- **Implementation**: the folder that contains the migration logic for this version transition. The path to this folder is composed of two components. The first part is the general **SourceFolder [installation Directory]/jsp/WEB-INF/config/migration**. The second part is composed of the Java packages that create a path to the basis package **com.idsscheer.webapps.arcm.dl.datamigration**. The **implementation** folder is now saved under this basis package. This path lies outside the ARIS Risk & Compliance Manager library and can therefore be extended with your own classes and resources that do not need to be compiled.
- **Fix**: if this optional attribute is set to the value **true**, the specified version transition is executed as a hotfix on an existing version. The database version remains unchanged. Here the attributes **source** and **target** must contain the same version.

Example

```
<?xml version="1.0" encoding="UTF-8" ?>
<migrationPlan xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
           xsi:noNamespaceSchemaLocation="./xsd/migrationPlan.xsd">
   <!-- up to version 10.0.10.0 -->
      <migration name="10.0.10.0_FixUserSubstituteAssignment_RBA"
source="10.0.10.0_rba_standard" target="10.0.10.0_rba_standard" approach="rba"
implementation="migFix10060_UserSubstitute" fix="true"/>
   <migration name="10.0.10.0_FixUserSubstituteAssignment_CBA"
source="10.0.10.0_cba_standard" target="10.0.10.0_cba_standard" approach="cba"
implementation="migFix10060_UserSubstitute" fix="true"/>

      <!-- 10.0.10.0 => 10.0.12.0 - from this version on there exists no
separate approach anymore -->
   <migration name="10.0.10.0_RBA_to_10.0.12.0" source="10.0.10.0_rba_standard"
target="10.0.12.0_standard" approach="unified"
implementation="mig100100To100120" />
   <migration name="10.0.10.0_CBA_to_10.0.12.0" source="10.0.10.0_cba_standard"
target="10.0.12.0_standard" approach="unified"
implementation="mig100100To100120" />

      <!-- 10.0.12.0 or later -->
      <migration name="10.0.12.0_to_10.0.14.0" source="10.0.12.0_standard"
target="10.0.14.0_standard" approach="unified"
implementation="mig100120To100140" />
</migrationPlan>
```

5.2 The XML schema of the migration plan

The documented schema for the **migrationPlan.xml** file is integrated as a resource in the **arcm_datalayer_migration_migsteps.jar** library and is located in the **com.idsscheer.webapps.arcm.dl.datamigration.xsd** package. The schema allows you to use the context-sensitive auto-complete, syntax check, and help if you edit the **migrationPlan.xml** file in a development environment such as **IDEA** or **ECLIPSE**. This file cannot be changed. A modified schema cannot be processed by the migration framework.

5.3 The location of the migration plan

The migration plan is integrated in the **migrationPlan.xml** folder as a resource in the **arcm_datalayer_migration_migsteps.jar** library and is located in the **com.idsscheer.webapps.arcm.dl.datamigration** package. Changes and extensions are possible in the file **migrationPlan.xml**, as long as you generate a new library **arcm_datalayer_migration_migsteps.jar** with the modified class.

In CTK you need the sources of the standard migration in order to carry out custom adjustments. Subsequent to this, a library is created from these modified sources, which replaces the standard library.

For detailed information on how to adjust migration of customer-specific databases in the CTK, refer to CTK user guide chapter **Implementing custom database migration steps**. You can download the user guide together with the CTK from the ARIS Risk & Compliance Manager area in Partner Business Portal (http://partner.softwareag.com/portfolio/ARIS/GRC.html).

6 The architecture

Java classes are saved in the source folder as migration steps (Page 5) that implement the **IMigrationStep** interface and extend the abstract **BaseMigrationStep** class. The method **::execute(…)** from **IMigrationStep** receives an instance from IMapping from the migration framework. To carry out the migration, various HighLevel functions from IMapping and IMigrationStep can now be used in the Execute method. The methods of the IMapping and **IMigrationStep** interfaces are stable and have Java documentation helps.

Figure 1: Mapping interface

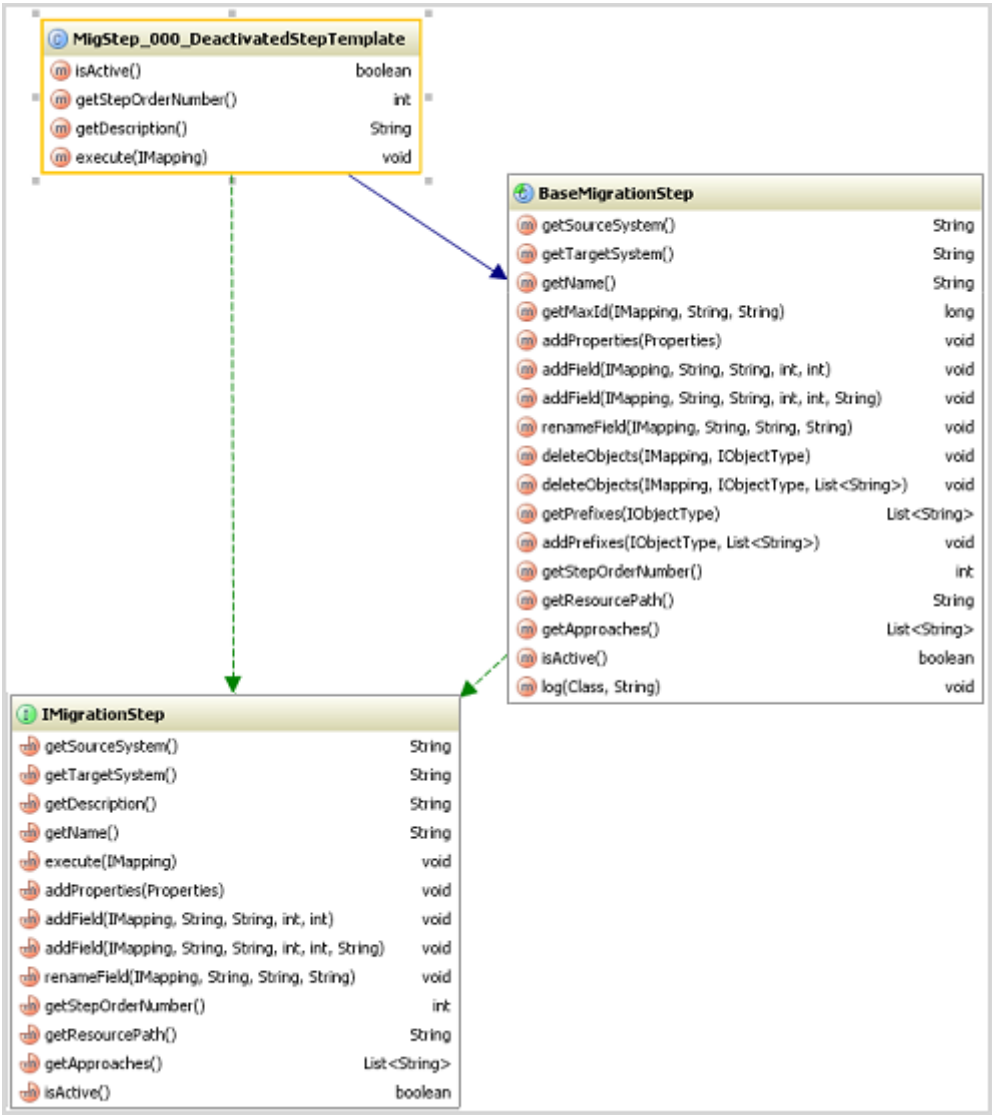

Figure 2: Migration interface

6.1 The construction set

All public methods from the **IMigrationStep** and **IMapping** interfaces can be combined just like in a construction set to achieve the desired results.

6.1.1 IMigrationStep

The **migSteps** subfolder is located in the **implementation** (Page 5) folder that contains all of the logic and data regarding a version transition. This folder contains all migration steps required for version transition, which implement the **IMigrationStep** interface. The **IMigrationStep** interface provides help functions that package database-specific dependencies and that are not dependent on database-specific requirements. The following methods are not implemented by the abstract superior class **BaseMigrationStep** and must be implemented in the specific migration step.

- **IMigrationStep::getDescription() String**

  Provides the description of the step as a string.

- **IMigrationStep::execute(IMapping)**

  Carries out the step. All public methods of **IMapping** can be used.

The following methods can be overwritten in the specific migration step.

- **IMigrationStep::getStepOrderNumber()**

  Determines the order (priority) of the steps to be performed. The default implementation always returns the order number **0**. Overwrite this function and enter a number larger than **0**, according to the position at which this step should be carried out. The lower the number the higher the priority.

- **IMigrationStep::isActive() Boolean**

  Specifies whether this step should be carried out (**return true**) or not (**return false**). The default implementation returns true. The step is thus active and is carried out. Overwrite this function if you want to temporarily turn off a step during development. Some steps, such as the generation of database indices, should be deactivated as soon as a follow-up version is available. Indices should then be generated in the last version transition only.

The interface provides other useful functions for processing data and data structures. These are documented in the Java documentation.

6.1.2 Step template

The deactivated step template is located in the standard migration folder and provides a template with basic examples for processing the schema and the data. If you want to create a new step, copy this template and adjust the class name, the name of the constructor and the **package** entry accordingly.

Also, make sure that you adjust all literals to the current requirements. Do not edit the template because it is used as a template for further steps.

Warning

Never set the template to **true** in the **isActive()** function. The step would then be carried out at the start of a migration and your data could be damaged.

6.1.3 IMapping

The **IMapping** interface provides help functions that package data-specific properties. With these functions you can for example, add tables or fields and process data. A few of these functions perform the relevant operations immediately and then close the required database resources automatically. A few functions also provide database resources directly. In this case, you must be sure to close these resources yourself in a Finally-block.

Close database resources such as **Connection** or **ResultSet**, which you receive from the **IMapping** interface, when they are no longer required. All methods in the **IMapping** interface affect the database connection specified in the **runtimeconfig.xml** configuration file in the **Datalayer** section. The interface provides other useful functions for processing data and data structures. These are documented in the Java documentation.

7 MigrationObject

The MigrationObject is a help structure with which to write data consistently in schema tables for ARIS Risk & Compliance Manager. Generate and write the objects sequentially in order to prevent conflicts with the internal ID management. This help structure does not generate the tables, but rather fills in the data semantically correct.

It is possible to generate a MigrationObject with the operator **new**.

```
MigrationObject migObject = new MigrationObject("POLICYREVIEWTASK", mapping,
this, UUID.randomUUID().toString(),
OVIDFactory.getOVID(SystemGUID.INTERNAL_SYSTEM_USER.getObjID()));
```

The newly generated object provides an API that can be used to maintain attributes with their values.

```
migObject.setAttribute("reviewRelevant", IMapping.TYPE_NUMBER, "0");
```

Relations can also be maintained like this:

```
migObject.setRelationAttribute("POLICYREVIEWTASK", "owner_group",
IMapping.TYPE_RELATION_1_1, ownerGroupID, 5520, 0, null);
```

With the **::write()** function the object can be written in the database.

```
migObject.write();
```

In order to receive a complete overview of the API from this class, check the **javadoc** of the file **MigrationObject.java**.

8 Automatic update of the schema version

In older versions of the migration framework it was necessary to write your own MigrationStep to maintain the **currentSchemaId** field in the **A_SCHEMAPROPERTY_TBL** table. This is no longer necessary with the current version of the framework. The corresponding field is now maintained automatically by the framework during the data migration.

9 Partly automatic cleanup

Each migration includes the automatic migration step **MigStep_CleanupMigration**. This migration step compares the metadata of the database with the metadata of ARIS Risk & Compliance Manager, corrects the database if possible, and if this is impossible, outputs an error message. This results in a timely warning about inconsistencies concerning the database structure. The following individual steps are performed:

- Create index, foreign and primary keys, as well as unique constraints.
- Remove dummy fields that were created during the creation of new objects.
- If required, enlarge field lengths, pre-decimal and decimal places, and output this information in the log file. However, you are recommended always specifying the field lengths properly. Data is not adjusted.
- If objects, attributes and 1:1 relationship attributes are missing an error is output.

10 Data migration from version 3.1.4 to 9.x

For migrations based on ARIS Risk & Compliance Manager version 3.1.4, special requirements apply.

10.1 Modeling approach

Like the current version, ARIS Risk & Compliance Manager version 3.1.4 saves version information in the **A_SCHEMAPROPERTY_TBL** database table. However, this information does not include the required modeling approach. (Up to version 10.0.10.0 there was a distinction of the risk-based approach (rba) and the control-based approach (cba).) Therefore, there is no synchronization of the modeling approach between the database and the current server configuration when starting the migration on an ARIS Risk & Compliance Manager version 3.1.4 database. It is assumed that the modeling approach configured matches that specified in the ARIS Risk & Compliance Manager version 3.1.4 database. Therefore, before starting the migration, you must check if the database matches the current configuration.

From version 10.0.12.0 only a unified approach is supported in ARIS Risk & Compliance Manager.

10.2 Schema ID

The schema information in the **A_SCHEMAPROPERTY_TBL** table for ARIS Risk & Compliance Manager version 3.1.4 was used for ensuring consistency of the database for the current version. However, this information might have varied within one version of ARIS Risk & Compliance Manager. In the current version, a migration path is automatically determined based on the data saved. Therefore, you must check if the value **PROPERTYVALUE** starting with **arcm_3** exists for the **PROPERTYKEY = "currentSchemaId"** key in the **A_SCHEMAPROPERTY_TBL** table.

10.3 Migration sandbox

The database schema was completely renewed in ARIS Risk & Compliance Manager version 4.0 compared to version 3.1.4. A data migration from ARIS Risk & Compliance Manager version 3.1.4 ends with marking the database as a migration sandbox. If a database is thus marked, the server starts only if the **runtimeconfig.xml** parameter in the configuration file is set to **automigration=true**. If this parameter is set, an ARIS Risk & Compliance Manager database cannot be used productively. To be able to generate a consistent productive database after migration, the following steps must be performed manually.

Prerequisite

- The migration was successfully performed.
- You have the **System administrator** role.

Procedure

1. Start ARIS Risk & Compliance Manager.
2. Export the database of ARIS Risk & Compliance Manager. For detailed information, refer to the ARIS Risk & Compliance Manager online help.
3. Stop the ARIS Risk & Compliance Manager Server.
4. Generate a new schema.
5. Set the relevant parameters in the **runtimeconfig.xml** configuration file.
6. In the **runtimeconfig.xml** configuration file, set the **automigration=false** parameter.
7. Restart the ARIS Risk & Compliance Manager Server.
8. In ARIS Risk & Compliance Manager Administration, import the backup as **system** user.

You have generated a consistent productive database.

11 Adjustments to the data migration in CTK

For detailed information on how to adjust migration of customer-specific databases in the CTK, refer to CTK user guide chapter **Implementing custom database migration steps**. You can download the user guide together with the CTK from the ARIS Risk & Compliance Manager area in Partner Business Portal (http://partner.softwareag.com/portfolio/ARIS/GRC.html).

12 Logging

Logging during migration takes place according to the settings in **log4j2.xml**. Set the **arcm** and **dl.framework** packages to **debug** before migration.

```
<Logger name=" com.idsscheer.webapps.arcm" level=" DEBUG "/>
<Logger name=" com.idsscheer.webapps.arcm.dl.framework" level=" DEBUG "/>
```

The migration output is displayed on the console and in the output file that are set in the **log4j2.xml** configuration file.

To prevent poor performance do not forget to undo these changes in the productive system.

Check the resulting log file carefully for error messages before you export the file and then import it into the productive system.

13 Legal information

13.1 Documentation scope

The information provided describes the settings and features as they were at the time of publishing. Since documentation and software are subject to different production cycles, the description of settings and features may differ from actual settings and features. Information about discrepancies is provided in the Release Notes that accompany the product. Please read the Release Notes and take the information into account when installing, setting up, and using the product.

If you want to install technical and/or business system functions without using the consulting services provided by Software AG, you require extensive knowledge of the system to be installed, its intended purpose, the target systems, and their various dependencies. Due to the number of platforms and interdependent hardware and software configurations, we can describe only specific installations. It is not possible to document all settings and dependencies.

When you combine various technologies, please observe the manufacturers' instructions, particularly announcements concerning releases on their Internet pages. We cannot guarantee proper functioning and installation of approved third-party systems and do not support them. Always follow the instructions provided in the installation manuals of the relevant manufacturers. If you experience difficulties, please contact the relevant manufacturer.

If you need help installing third-party systems, contact your local Software AG sales organization. Please note that this type of manufacturer-specific or customer-specific customization is not covered by the standard Software AG software maintenance agreement and can be performed only on special request and agreement.

If a description refers to a specific ARIS product, the product is named. If this is not the case, names for ARIS products are used as follows:

Name	Includes
ARIS products	Refers to all products to which the license regulations of Software AG standard software apply.
ARIS Clients	Refers to all programs that access shared databases via ARIS Server, such as ARIS Architect or ARIS Designer.
ARIS Download clients	Refers to ARIS clients that can be accessed using a browser.

13.2 Data protection

Software AG products provide functionality with respect to processing of personal data according to the EU General Data Protection Regulation (GDPR).

Where applicable, appropriate steps are documented in the respective administration documentation.

13.3 Restrictions

ARIS products are intended and developed for use by people. Automatic processes such as generation of content and import of objects/artefacts using interfaces can lead to a huge data volume, processing of which may exceed the available processing capacity and physical limits. Physical limits can be exceeded if the available memory is not sufficient for execution of the operations or storage of the data.

Effective operation of ARIS Risk & Compliance Manager requires a reliable and fast network connection. A network with an insufficient response time reduces system performance and can lead to timeouts.

If ARIS products are used in a virtual environment, sufficient resources must be available to avoid the risk of overbooking.

The system has been tested in the **Internal control system** scenario with 400 users logged in simultaneously. It contains 2,000,000 objects. To guarantee adequate performance, we recommend operating with not more than 500 users logged in simultaneously. Customer-specific adaptations, particularly in lists and filters, have a negative impact on performance.