JTLS-2008-10000 Database Repository

Ellen F. Roland, Steve Tang, Jane Wu, John Ruck, Zafer Aktan, Robert Montgomery

1.0 Summary of Model Change Request

Traditionally, JTLS provides a set of standard databases, both as demonstrative representations of real world scenarios and as baselines from which users can build their own databases. These two purposes can be conflicting, because while for demonstration purposes, the data in the database should be concise, as baselines, the data should be robust.

Furthermore, user-developed scenarios that are derivatives of the baselines from previous releases are often not adapted to a new release's data updates. Specifically, the upgrade procedure provided with a new release can only upgrade the format of the old scenarios to the latest format. Any new data fields introduced in the new release are given default values that are designed to only mimic the functionalities of the old release during the upgrade. Should users wish to utilize new model functionalities, their database developers have to review these new fields, copy the meaningful values from the new baseline, or derive other suitable values.

Lastly, current and future releases of JTLS-GO are capable of supporting the modeling of more than one conflict regions in an exercise scenario. To build such a scenario, database developers should be able to merge data from existing scenarios that model individual conflicts from different areas into a single scenario.

To address these issues and requirements, a Data Repository System (DRS) is introduced as an added enhancement to the Database Development System (DDS). The DRS includes a tool that can migrate data from one database (Source) to another (Destination), thus enable database builders to, with ease, put data from various scenarios (Sources) into a single repository (Destination); and then later, can grab pertinent data from the repository (Source) and copy it to other exercise databases (Destinations). With this migration tool, users are able to maintain their own repositories as large databases that can either be demonstrative gaming scenarios containing data representing multiple conflict regions, or used only as a mean to hold extensive data, thus alleviate the need to use standard databases as baselines to build other exercise scenarios.

However, merely migrating data from one database to another is not quite enough. For example, the repository can potentially grow unnecessarily large with redundant data; or updated data not migrated due to there is existing record in the destination database with the same identification. In order to address these issues, the DRS should also have the capability to synchronize the data during or after the migration. The synchronization procedure is an important aspect of DRS in order to maintain a comprehensible repository, and truly support the updating of data in databases derived from previous release to the latest data.

2.0 Design Summary

This design will explain what the Data Repository System (DRS) is about, and how the components within the DRS are intended to work. It lays out the blueprint of how a repository can be built, expanded, and utilized. Database developers can get an idea on how to apply the capabilities of the DRS to their own specific needs, whether to maintain a robust repository to suit their database development, to update the data in their old databases with the most current data, or to combine data from different scenarios to create a multi-conflict representation scenario.

2.1 Basic Concept

The Data Repository System (DRS) is an add-on to the Database Development System (DDS). Users should be able to configure the DDS to enable the Data Migration Tool. Then the tool can copy any user-selected data and its related referential data from the source database to a destination database.

This data migration capability allows users to build or extend their data repository when they set the repository as the destination database; or use the repository to build their exercise scenario when the repository is used as the source database.

When migrate a data record, if the data does not exist in the destination database, it is inserted; however, if a similar record exists in the destination, that record is ignored even though its data maybe updated. To update the existing data in the destination database, a data synchronization capability is needed to update the data according to users' needs.

2.2 Background

In order to move data between databases, a Data Migration tool was created to be the first part of the Data Repository System (DRS). At this stage, the migration tool only migrate Order of Battle (ORBAT) and certain referential data from source to destination database. The migration tool is available under the Tools menu of a DDS client (DDSC) when that DDSC is configured properly using the DDS Configuration Program (DCP). To see how to use DCP to configure data migration, please refer to the *JTLS-GO DDS User Guide*, Chapter 3.

The Data Migration tool has a graphical user interface (GUI) that displays the source and destination databases' command hierarchy trees side by side. The tool lets user select desired Order of Battle (ORBAT) data from the source command tree, then drag-and-drop it at a specific position in the destination's command hierarchy. The GUI also has a status message area below the two command hierarchy trees that displays migration status.

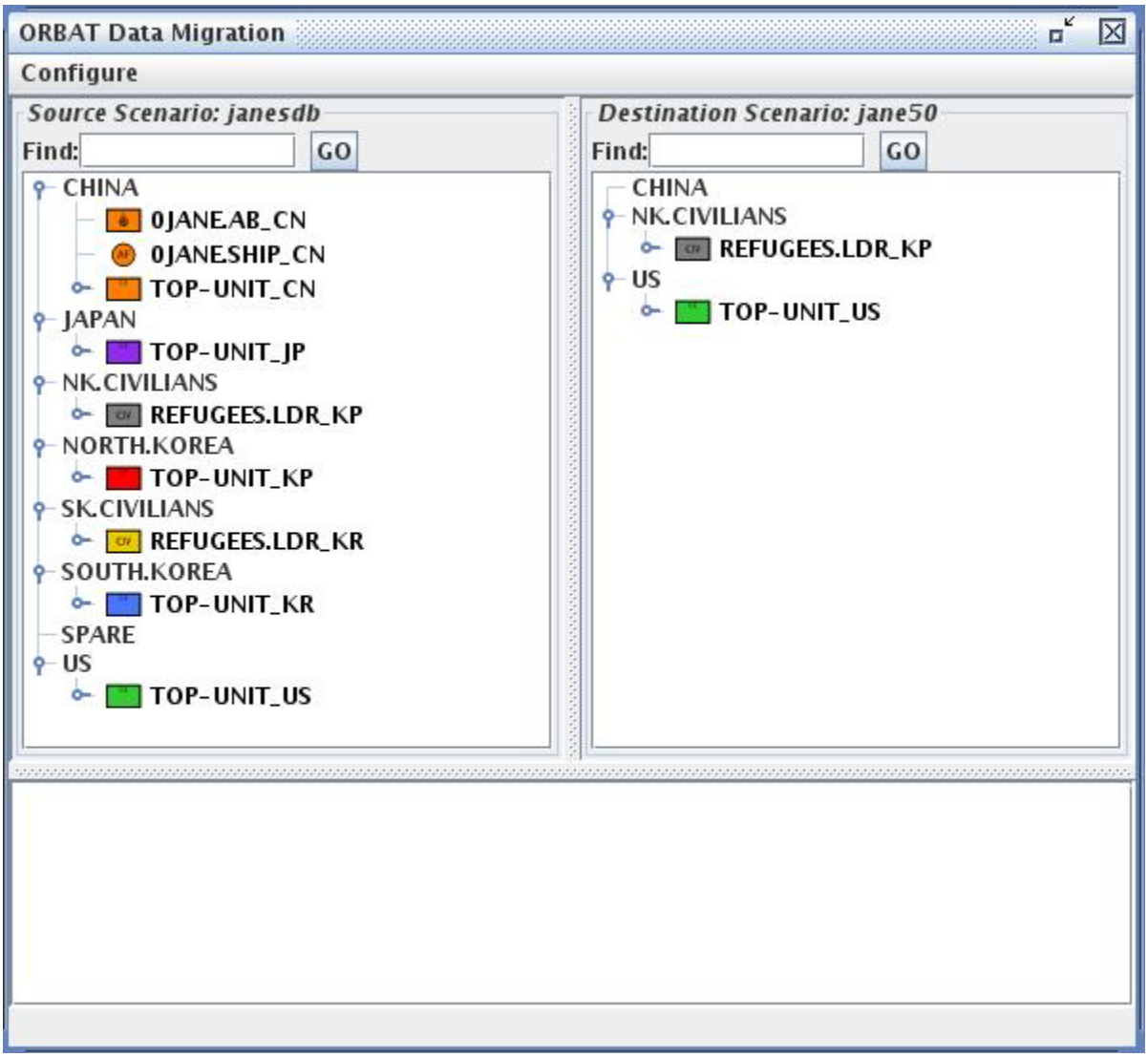


FIGURE 1. ORBAT Data Migration Tool GUI

2.3 Remaining Issues

2.3.1 Data Migration Incomplete

Currently, when ORBAT data is being migrated, not only selected units and targets data is migrated, any missing directly referenced data such as FACTION_COUNTRY, FORCE_SIDE, TUP, SUP, HUP, CALIBER, etc., and a few indirectly referenced data such as TUP_POT, SUP_POT, TUP_CS, SUP_CS, TUP_SC, SUP_SC, and TUP_ASP is migrated as well.

However, most of the indirectly referenced parametric data including any lethality data, etc. is ignored during the migration. This, although doesn't violate the database integrity from Oracle's standpoint, would result in many errors and warnings generated from the Scenario Verification Program (SVP) after the data is migrated. The plan is to batch process the migration of needed parametric data post the ORBAT data migration.

2.3.2 General Migration

So far, data migration can only start from the ORBAT data; but sometimes, it is desirable to migrate certain prototype or parametric data without the units or targets data. For example, users might want to copy a HUP data without copying a HRU that uses that HUP. So in addition to let users select data from source database's command hierarchy tree, there should be user interfaces where users can select specific rows from individual tables in the source database for migration.

2.3.3 Update or Synchronize Data

Currently during ORBAT data migration, only the missing referenced data such as TUPs, SUPs, etc., is migrated. However if the prototypes already exist (i.e., identified by the same names or keys) in the destination, the records are not copied even though their fields might be updated in the source database. To address this issue, we want to add a synchronization capability to the data migration tool where user can visually compare the similar records between the source and destination database and decide whether to skip or update the records accordingly in the destination database.

3.0 Detailed Design

The Data Repository System (DRS) mainly consists three parts:

1. a tool that configures the data migration,
2. a tool that migrates the data, and
3. the repository database.

The DDS Configuration Program (DCP) is used to let user decide how data can be migrated. The actual tool that migrates the data is in two parts: the DDS client that let user select which data to migrate, and where to, and the DDS GlassFish server that copies the relevant data from the source database and insert it in the designated destination database.

Lastly, the repository database is a JTLS database that contains comprehensive data from which user can further build upon or derive their own exercise scenario's database from.

3.1 DCP

3.1.1 Existing Capability

The DDS Configuration Program (DCP) is extended so that user can add destination databases to a source database through the "Setup Destination Databases..." option under DCP's Tools menu. When setting up a destination database, the GlassFish server of the source database "learns" the domain information of the configured destination database. Thus when the source database server starts, it can obtain "privileges" to query from and insert data in the destination database.

After the destination database(s) are set up in the DCP, user also must assign those databases to specific DDS clients that are made available for the source database, thus assigning specific migration "roles" to individual DDS clients. The Data Migration tool is only available in a DDS client if that client is assigned a migration role. A DDS client can have multiple roles to migrate data to different destinations; and consequently the same destination can be assigned to more than one DDS clients. Again, please refer to *DDS User's Guide*, Chapter 3 for detailed instruction.

3.1.2 Future Capability

Additionally, we will add further configuration of the destination database so that user can customize migration rules for certain database tables and columns. In order to perform migration, a set of special rules are defined for the GlassFish server to suit our specific needs. Please refer to Table 1 for detailed description of the rules. These rules may be expanded as needed when we work on the batch post processing of the parametric data migration. A graphic user interface (GUI) should be added to DCP's destination database's setup so that user can customize the migration rules for a specific destination database. For each destination user customizes, the DCP shall generate a <dest_scenario>_dmrules.xml data file under source database's GlassFish domain. Please refer to Figure 5 for an example of such file for details.

3.2 DDS Client

3.2.1 Existing Capability

A DDS Client (DDSC), when assigned with a migration role by the DCP, would have the "Data Migration" option available under the DDSC's Tools menu. The option, when selected, cascades to sub menu item(s) listing the specific destination database(s) assigned to the DDSC. Once user selects a particular destination, a login window similar to regular DDSC's login window would appear to ask user login to the database to download the ORBAT data from the destination. After the destination data is downloaded, an ORBAT Migration window would appear, listing the source and destination databases' command hierarchy trees side-by-side. From here, user can select ORBAT data from the source command tree and drag-and-drop it to a place in the destination command tree. When that happens, a migration request is sent from DDS client to its GlashFish server to perform the migration. The request would contain migration data encapsulated in XML format.

3.2.2 Future Capability

The ORBAT Migration tool interface should be expanded (and renamed) to allow user not only select ORBAT data from the command hierarchy tree, but also from any individual database tables for migration. That way, user can migrate prototype or parametric data without migrating the ORBAT data if so choose.

Furthermore, the migration tool should be expanded to support the user interface to synchronize the data between source and destination database for specific tables. The interface should also allow user to configure the synchronization such that user can specify certain data fields to ignore (or compare) for each table, and possibly allow user to specify a list of old and new names/keys pairs for the data records that are renamed so that they can be synchronized as well. Below is a generic GUI design for such configuration (shown only the main portion):

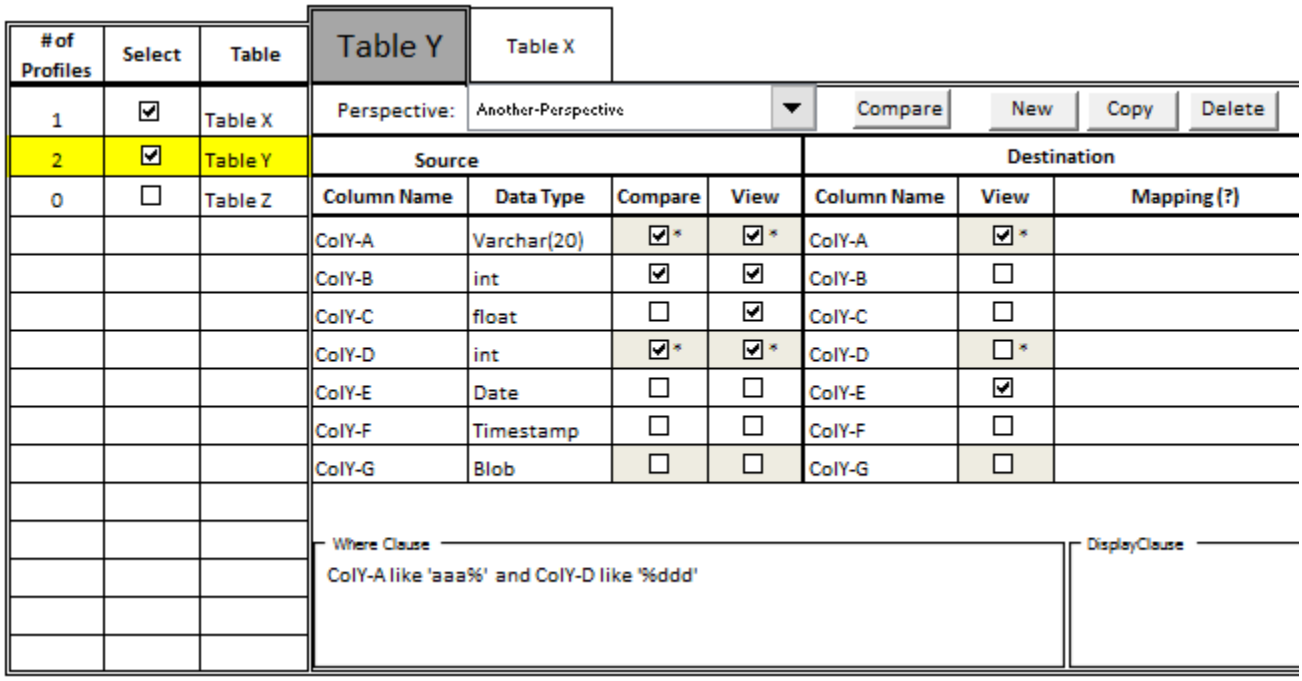


FIGURE 2. Synchronization Configuration Design

The synchronization configuration is designed as such:

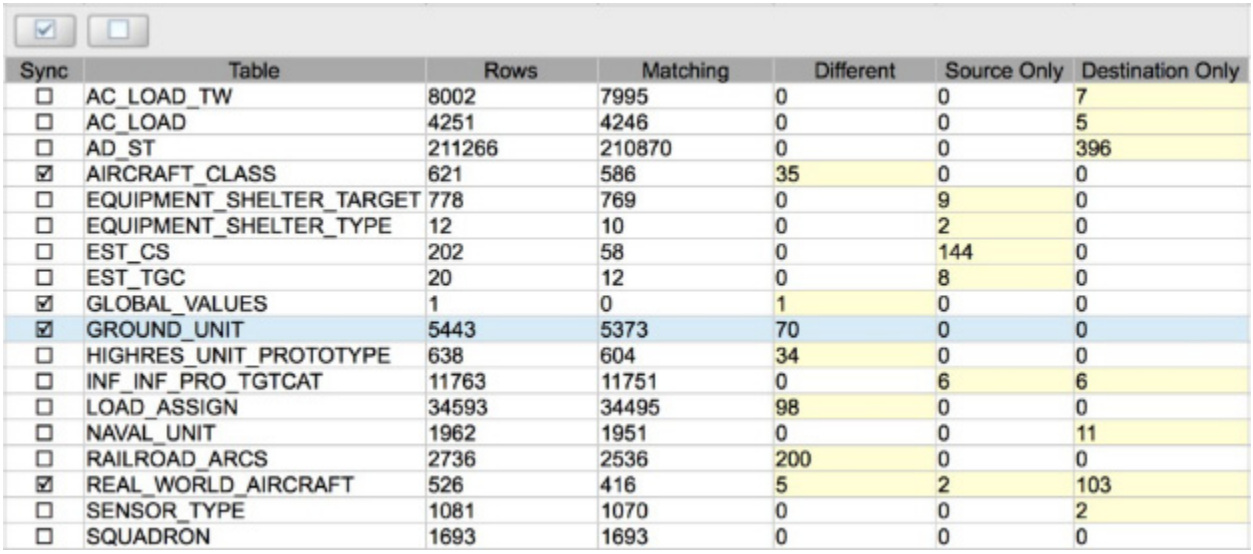
All the tables (X, Y, Z) are listed here, only the selected table(s) between the source and destination databases are compared. For each table, there can be multiple comparison profiles/ perspectives assigned, but only the selected perspective is used for comparison. When user selects a table from the list on the left, a detailed panel describing that table will be displayed on

the right in a tabbed pane. There can be multiple tabs for previously selected tables, but only the currently selected table is the active one, i.e., Table Y.

A comparison perspective is such that user can 1) select and exclude specific column(s) of the table from comparison; 2) specify a Where-Clause to only look a subset of data matching the criterion if applicable; and lastly, for certain tables, can 3) specify a list of old/new values mapping. This last one (mapping) can be useful when comparing possibly renamed records, but will be implemented as a "Next Step" enhancement. Also note that the shaded cells here indicate that those checkboxes' selections are not editable, and the ones with "*" indicate that those are primary key columns.

User can create a new (with default setting) perspective, copy an existing perspective, or delete an existing perspective for a specific table. User can also click the "Compare" button to compare the specific table only between the source and destination databases.

There should be an action button "Compare Selected" below the configuration panel to run the data comparison. When the client receives the result from the server, it will display an overview if multiple tables are being compared:



Sync	Table	Rows	Matching	Different	Source Only	Destination Only
☐	AC_LOAD_TW	8002	7995	0	0	7
☐	AC_LOAD	4251	4246	0	0	5
☐	AD_ST	211266	210870	0	0	396
☑	AIRCRAFT_CLASS	621	586	35	0	0
☐	EQUIPMENT_SHELTER_TARGET	778	769	0	9	0
☐	EQUIPMENT_SHELTER_TYPE	12	10	0	2	0
☐	EST_CS	202	58	0	144	0
☐	EST_TGC	20	12	0	8	0
☑	GLOBAL_VALUES	1	0	1	0	0
☑	GROUND_UNIT	5443	5373	70	0	0
☐	HIGHRES_UNIT_PROTOTYPE	638	604	34	0	0
☐	INF_INF_PRO_TGTCAT	11763	11751	0	6	6
☐	LOAD_ASSIGN	34593	34495	98	0	0
☐	NAVAL_UNIT	1962	1951	0	0	11
☐	RAILROAD_ARCS	2736	2536	200	0	0
☑	REAL_WORLD_AIRCRAFT	526	416	5	2	103
☐	SENSOR_TYPE	1081	1070	0	0	2
☐	SQUADRON	1693	1693	0	0	0

FIGURE 3. Synchronization Overview Design

The overview lists all the tables configured for comparison and the comparison result. Anything that does not match between the source and destination databases are shaded (i.e. with a light yellow background). There is a leading column with checkboxes for each table. If checked, those table(s) will be synchronized as such: update the different record(s); insert the source only record(s) in the destination; or both, depending on user selected action. The "Check" button

above the table when pressed, will select all the tables listed, and the "Uncheck" button will deselect all.

When a specific table from the overview is selected (highlighted in blue), user can elect to view the detailed synchronization result of the table. See example below:

Sync	GU_SHORT_NAME	GU_LONG_NAME	GU_PROTOTYPE
☑	3-509.ABN.INF.BN_US	3-509 AIRBORNE INF RGT 4BCT 25ID US	IBCT.INF.BN_US
	3-509.ABN.INF.BN_US	3-509 AIRBORNE INF RGT 4BCT 25ID US	ABABNINFBNAD_US
☑	3-7.FA.BN.HQ_US	3-7TH FIELD ARTILLERY BN HQ 3BCT 2ID US	SBCT.HHB.FA.BN
	3-7.FA.BN.HQ_US	3-7TH FIELD ARTILLERY BN HQ 3BCT 2ID US	SBCT.FA.BNHQ_US
☑	3.BCT.HQ.25ID_US	3RD BRIGADE COMBAT TEAM HQ 25ID US	IBCT.HHC_US
	3.BCT.HQ.25ID_US	3RD BRIGADE COMBAT TM HQ 25ID US	SBCT.HQ_US
☑	3.BCT.HQ.2ID_US	3RD STRYKER BRIGADE COMBAT TM HQ 2ID US	SBCT.HHC_US
	3.BCT.HQ.2ID_US	3RD STRYKER BRIGADE COMBAT TM HQ 2ID US	SBCT.HQ_US
☑	A.BTRY.1-15.FA_US	A BTRY 1-15 ARTY BN (M109A6) 1BCT 2ID US	HBCT.FA.BTRY_US
	A.BTRY.1-15.FA_US	A BTRY 1-15 ARTY BN (M109A6) 1BDE 2ID US	FA.BTRY.HBCT_US

Different Source Only Destination Only

FIGURE 4. Synchronized Detailed Table Design

Note also that this window can be presented directly when user elect to compare a single table from the synchronization configuration earlier.

The interface shows a tabbed pane that allows user to see records of a table (i.e., GROUND_UNIT) that are Different, Source Only, and Destination Only separately. When the "Different" tab is selected, user sees only the records that are different between the source and destination databases. The different values are highlighted (i.e. in light yellow). The odd rows are data from source database, and the even rows are those from destination. Again the "Check" button above the table will select all source/destination pairs, and the "Uncheck" button will unselect all. Updates are only made for the destination database if the record is checked here.

There is also a "Filter" button next to the "Uncheck" button that, when pressed, only the fields/ columns that contain different data are shown in the table.

When the "Source Only" tab is selected, user will see a list of records that are in the source database, but not in the destination. There should also be a "Sync" column that allow user to select or unselect the record. Only the selected record will be inserted in the destination when user elect to "Sync".

When the "Destination Only" tab is selected, user will see a list of records that are in the destination database, but not in the source. This is for information only because our "synchronization" is one-direction only.

3.3 DDS Server

3.3.1 Existing Capability

The DDS GlassFish server for a source database handles the actual data migration, and later on, also data synchronization. A database becomes a source database when destination database(s) are setup for it by the DCP. When the server receives migration request from its client, it performs the migration as such: if the data record contains field/column that is referenced to data from another table (parent), unless a migration rule is specified for that field/ column, the server will "drill down" and migrate the data in the parent table recursively before migrating the specified data record. The following table is the detailed description of the migration rules established for the server so far:

Table 1. Description of Migration Rules

KEY	DESCRIPTION
D1	No drill down, unconditionally set to null, e.g. FACTION_COUNTRY.FC_LEADER
D2	No drill down, if object does not exist in destination,unconditionally set to null
D3	No drill down, if object cannot be found in source, unconditionally skip insert the row
D4	No drill down, if object cannot be found in destination, unconditionally skip insert the row
D5	Drill down as the leading column of a composite reference, e.g. TUP_POT_SUBCATEGORY
D7	Drill down unless it refers back to itself, in which case no drill down, e.g. COMMAND_LEVEL.CL_LOWER_LEVEL
D2PU	Special Rule for PARENT_UNIT of Units, No drill down, if referenced parent unit does not exist in destination, unattached it.
H1	Special process for HHQ or owning unit.
H2	HHQ or owning unit, if not found skip insert the record
P1	Standard drill down w/ POST process P1, set column value per standard drill down, after row successfully processed, check through the list of specified columns.
P2	Standard drill down w/ POST process P2, set column value per standard drill down, after row successfully processed, do whatever.

Table 1. Description of Migration Rules

KEY	DESCRIPTION
PD1	D1 with POST process PD1, Set column value per Rule D1 for now just a placeholder to illustrate how rules work
PD2	D2 with POST process PD2, Set column value per Rule D2 for now just a placeholder to illustrate how rules work
SKIP	Skip the column, this can result in default value kicked in if not otherwise set, e.g. TUP_POT_CATEGORY
POST	After row inserted, process through the list of specified columns per the POST rules defined on the. The rule is set for table level.

The source database server, upon start, will load any customized <dest_scenario>_dmrules.xml files generated by its DCP if those are available. When the server receives the first migration request to a particular destination from its client, it will generate an XML stream capturing the characteristics of the database schema and any migration rules specified for the tables and columns. An example of such data is shown in Section 4.0 Data Changes.

3.3.2 Future Capability

The server will be expanded to handle batch migration of the parametric data according to user defined rules. To do that, additional migration rule(s) will be added to the current existing set as needed, and the relevant tables shall be marked for batch post processing in the <dest_scenario>_dmrules.xml by default, or by user customization if applicable.

Moreover, the server will also handle data update/synchronization request sent from the client in the future. The client will send its request querying for data comparison of a specific table. The request may specify certain fields/columns to overlook (or specifically compare if more suitable); and the request may provide alternative values in place of the actual primary keys for certain tables to compare because some records may have been renamed. The server shall send the comparison result back to the client to display visually. And given user's choice, the server may receive request from the client to update the records in the destination database. Notice that the direction of the synchronization is one-way only, meaning that copying values from source to destination database.

4.0 Data Changes

There is no specific database schema change planned for the data migration purpose. However, there are additional user specified and application generated data files and streams in support of the migration.

User will be able to customize the set of migration rules for a particular destination in the DCP. The DCP will generate the migration rules in an XML data file for the source database. The data file follows the naming convention of <dest_scenario>_dmrules.xml, and the following is an example of the data and its format:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ddsDMRule description="add your description 5.0">
        ...
        <table name="AIRBASE">
                <column name="AB_PARENT_UNIT" process="D2PU"/>
                <column name="AB_ALTERNATE_BASE" process="D2"/>
                <column name="AB_PORT_UNIT" process="D2"/>
                <column name="AB_REG_SUP_UNIT" process="D2"/>
                <column name="AB_HIGHER_HQ" process="H1"/>
                <column name="AB_INT_SUP_UNIT" process="D2"/>
        </table>
        ...
        <table name="FACTION_COUNTRY">
                <column name="FC_LEADER" process="D1"/>
        </table>
        ...
        <table name="SHIP_UNIT_PROTO" process="POST">
                <refColumn seqid="1">SHIP_UNIT_PROTO.SUP_NAME</refColumn>
                <column name="SUP_NAME" process="P1">
                        <refColumn seqid="1">SUP_POT.SUP_POT_NAME</refColumn>
                        <refColumn seqid="1">SUP_CS.SUP_NAME</refColumn>
                        <refColumn seqid="1">SUP_SC.SUP_NAME</refColumn>
                        <refColumn seqid="1">SUP_SMALL_BOAT.SUP_NAME</refColumn>
                </column>
                <column name="SUP_GRAPHICS_SYMBOL" process="D2"/>
        </table>
        ...
        <table name="SUP_POT">
                <column name="SUP_POT_SUBCATEGORY" process="D5"/>
                <column name="SUP_POT_CATEGORY" process="SKIP"/>
        </table>
</ddsDMRule>
```

FIGURE 5. Customized DM Rules Data File Example

Lastly, as part of the Data Repository System (DRS), a comprehensive database is going to be built that contains extensive data from which users can extend and derive their own exercise data.

5.0 Order Changes

No order changes are required for this ECP.

6.0 JODA Changes

No JODA Data System parameter, structure, or protocol changes are required to implement this design.

7.0 Test Plan

To be provided later.

7.1 Test 1 Title

Purpose: [*Describe the specific feature, function, or behavior to be tested or measured.*]

Step 1: Text

Step 2: Text

Expected Results: [*Describe the specific model behavior to be observed.*]

7.2 Test 2 Title

Purpose: [*Describe the specific feature, function, or behavior to be tested or measured.*]

Step 1: Text

Step 2: Text

[*Describe the specific model behavior to be observed.*]