Upgrade Tool 3.1 for Agile e6.0.2

Upgrade Manual

Agile Software GmbH, Ruschgraben 133, D-76139 Karlsruhe
Phone: +49 / 721 / 6291 0, Fax: +49 / 721 / 6291 88
Training-europe@agile.com
Upgrade Tool 3.1 for Agile e6.0.2
Revision: 26.09.2006

CONTENTS

1. Introduction

This guide is intended as a manual for upgrading earlier versions of the Agile e-series products to Agile e6.0.2 with the Agile Upgrade Tool.

The Upgrade tool allows a direct upgrade to Agile e6.0.2 from one of the following product versions

- *CADIM/EDB 2.3.2 or higher*
- *axalant 2000 SPx*
- *Eigner PLM 5.x*
- *Agile e6.x*

2. Prerequisites

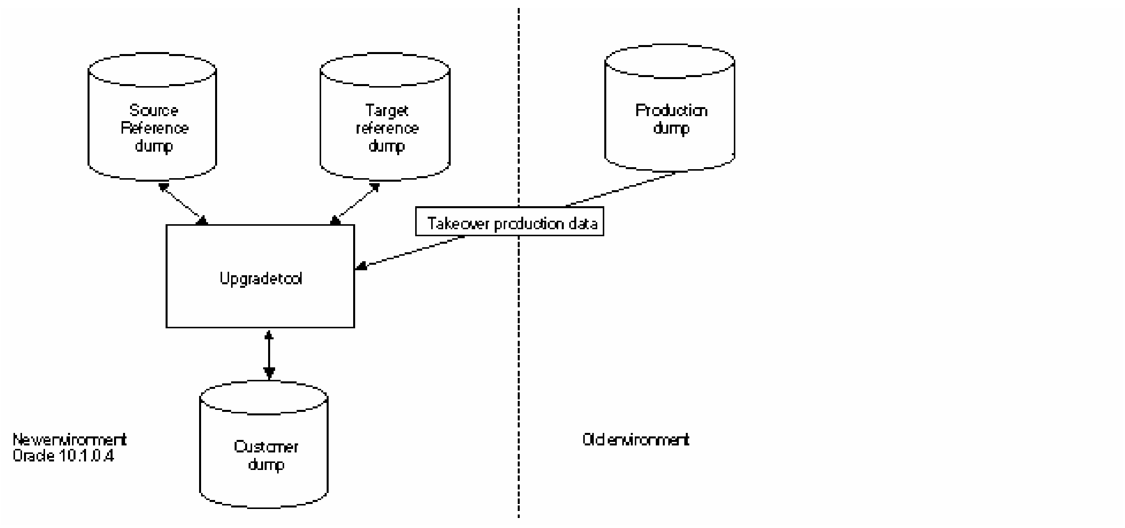
- *Do not use the software in any situation where significant damage to property or business could occur from a software error. In no event will AGILE or any other party who has been involved in the creation, production or delivery of the software be liable for special, direct, indirect, incidental, punitive or consequential damages, including loss of profits, revenue, business, goodwill, data or computer programs or inability to use the software, however caused and regardless of the theory of liability even if AGILE or such other party has been advised of the possibility of such damages.*
- *The Upgrade Tool addresses experienced project engineers and PLM administrators with customizing and database experience. Do not use the tool without the necessary knowledge. Read the complete manual in order to get all necessary information.*
- *Do not attach to or even change the production system. Always work on a copy of the production database dump. Avoid working on production computers to exclude any influence on the system. Never insert database connections of production database users in any configuration file or script except for exporting the dump or a source for copying tables (Production Database).*
- *You should always be able to restart the old PLM version (CADIM/EDB, axalant or Eigner PLM 5.x) as a fallback strategy.*
- *The best performance is reached when installing the upgrade tool on your database server. It is also possible to work on any machine in the LAN.*
- *Dump upgrade takes up to 8 Hours runtime per each environment - please have patience.*

- ❑ *upgrade tool is running on Windows 2003 Server and all unix platforms officially supported for Agile e6 series.*
- ❑ *At least JRE 1.4.2_08 should be installed on the machine and configured in the environment (JAVA_HOME must be set – this can be done in upg_env.cmd resp. upg_env.sh script).*
- ❑ *SQLCMD.EXE should be installed*
- ❑ *Database versions supported in this upgrade tool are Oracle 10.1.0.4 / Oracle 10.2.0.2 and Microsoft SQL Server 2005 SP1 (English/German)*
- ❑ *For production database link (step Takeover production data) Oracle 8.1.7 / 9.2.0.4 and Microsoft SQL Server 2000 SP4 can be used*
- ❑ *ORACLE_HOME variable must be set for "system V" operating system – Unix/Linux etc. This can be done in upg_env.cmd (upg_env.sh) script*
- ❑ *SQL*PLUS must be installed on the machine.*
- ❑ *At least 250 MB hard disk space must be available for the software and generated log and data files on the local hard disk where the tool is installed*
- ❑ *at least 512 Mb free RAM must be available to run a dump upgrade*
- ❑ *Sufficient disc space must be available to store copies of your production database and reference dumps on the machine / within the database.*

For additional information and most up-to-date Upgrade information, check the Agile Support page at **http://eignersupport.agilesoft.com/index.asp** (you will need a password to enter the support website).

3. How it works

The upgrade tool is implemented in Java. The tool accesses the databases directly using a JDBC connection. The configuration of all upgrade steps is stored in a set of xml control files. In addition SQL scripts are used for special steps.



The upgrade process is organized in following phases:

- *Pre-actions on the original production environment*
- *Customizing Upgrade*
  In this step the customization and configuration stored in the database is updated to Agile e6. The minimum passing time will be 4-5 hours (depending on the system, main parameter is memory!) Make a copy of your production database dump. Do not attach your production system. Always work on a copy of your data.
- *Customizing / Test phase*
  The Upgrade Tool will create a dump on which you can run Agile e6. All functionalities must be tested to run correctly, including the whole customer-specific functionality. This dump is not error free. You have to check all functionalities and clear out the errors caused by setting up the upgrade tool.
- *Take over production data*
  All tables containing production data like document and item master data are copied from the production system to your new Agile e6 dump. They will be adapted so that you can work on this data within Agile e6.

Log files are created and stored in the *log/* directory. After the dump upgrade is done, these files have to be reviewed, especially *errors.log* file to make sure there are no errors, otherwise manual dump changes have to be made. Further “synchronize step” log files placed in data/sync have to be reviewed. Please control log files after each upgrade step and correct occurred errors manually before proceeding with upgrade.

4. Installation and Configuration

4.1. ApplicationParameter.xml template

The Upgrade tool is preconfigured for an upgrade of an oracle database dump. Before proceeding with upgrade on Microsoft SQL Server, please copy the file *conf/template/ApplicationParameter.xml* to conf/ApplicationParameter.xml.

4.2. Database settings in Oracle

The Upgrade Tool needs a well-configured database to provide a good performance. The Oracle standard database settings are not sufficient to run the program within the stated time.

4.2.1. Oracle parameters

Check the Oracle Parameters and verify that at least the following minimum values are set in your database instance:

- ❑ *db_cache_size >= 200 000 000 (200MB)*
- ❑ *shared_pool_size >= 100 000 000 100Mbytes*
- ❑ *log_buffer >= 163840 3*64 Kbytes*

If database memory consumption is too small, adapt the values.

If you use the server parameter file *spfile* (like in the Agile e6 standard installation), execute the following commands to change the values of the initialization parameters.

- ❑ *Login into **Sql*Plus** as user **sys***

```
C:\> Sqlplus /nolog
SQL>CONNECT <sys>@<db_service> as sysdba
SQL>ALTER SYSTEM SET <parameter name>=<Value>  SCOPE=BOTH
```

- ❑ *Note: Do not change the values of production systems. Make a copy of the initialization file and adapt the values.*
- ❑ *Read the Oracle online manuals and the Oracle10.1 installation manual from Agile in addition.*

Oracle needs physical memory. If the system starts swapping or paging, the Oracle performance degrades or causes errors. Examine your free physical memory and prevent the OS from swapping.

Some Unix systems have maximum values for shared memory. Refer to the installation instructions before changing any value.

4.2.2. SQL Net configuration.

The network domain is part of different oracle settings. Please check if the domain is consistently used for the following settings:

- ❑ *Global Database*
- ❑ *Service name*
- ❑ *Listener.ora*
- ❑ *Default domain name*

4.2.3. Global Database name

Login into **Sql*Plus** as user **sys** and check the global database name. The name should contain the network domain. Here is an example:

```
sqlplus  <system>/<db_password>@<db_service>
SQL>select * from global_name;

GLOBAL_NAME -----------------------
PLM.WORLD
```

The example uses the default network domain in world. Also possible are values like agile.agilesoft.com.

Change the global database name login to **Sql*Plus** and execute the following commands:

```
SQL>alter database global name plm.agile.agilesoft.com
```

4.2.4. Service name

The service name in the SQL net configuration file *tnsnames.ora* in the directory *$ORACLE_HOME/network/admin* must include the network domain.

- ❑ *Change to the directory* $ORACLE_HOME/network/admin
- ❑ *Open the file* tnsnames.ora *and check if the service name is fully defined. That means the name contains the same network domain as the global database name.*

```
PLM.WORLD =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = localhost)(PORT = 1521))
    )
    (CONNECT_DATA =
      (SERVICE_NAME = PLM.WORLD)
    )
  )
```

4.2.5. listener.ora

Check if the global database name in the section SID_List of the listener configuration file contains also the same fully qualified global database name.

```
SID_LIST_LISTENER_PLM =
  (SID_LIST =
    (SID_DESC =
      (GLOBAL_DBNAME = PLM.WORLD)
      (SID_NAME = plm)
    )
  )
LISTENER_PLM =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = localhost)(PORT = 1521))
  )
```

4.2.6. Service name

The default setting for the network domain in the *sqlnet.ora* file should be the same. Change to the directory *$ORACLE_HOME/network/admin* and open the file *sqlnet.ora* and check the default domain settings.

```
names.default_domain = world
```

4.3. Database settings in Microsoft SQL Server

4.3.1. Memory settings

The memory consumption of an SQL Server is dynamic. Control the actual memory settings on the new database server.

- ❑ *Start the Enterprise Manager.*
- ❑ *Select the server name and choose SQL-Server Properties in the Extra Menu.*
- ❑ *Select folder "memory" in the new window and adjust the values*

4.4. Tablespaces / file groups

Check if the following table spaces (Oracle) or file groups (SQL Server) exist in your database. If one of them does not exist they have to be created.

- ❑ *edb_tmp*
- ❑ *edb_tmpidx*
- ❑ *edb_lob*
- ❑ *edb_tmp*
- ❑ *edb_tmpidx*

4.4.1. Oracle

- ❑ *Change to the directory* upgrade/ora/sql
- ❑ *Adapt file names, paths and file size in the script* cre_plm_tbs.sql
- ❑ *Login as user* **system** *to* **Sql*Plus** *and execute* cre_axa_tbs.sql

```
sqlplus system/<password>@PLM60
SQL> @cre_plm_tbs.sql
```

4.4.2. SQL Server

Please ensure following file groups are existing in your database:
edb, edb_idx, edb_lob, edb_tmp, edb_tmpidx

1. **Call the script *missing_f.cmd* for the customer dump to create the missing filegroups**

   Example how to call *missing_f.cmd* in the command line:
   missing_f.cmd[1] *sa*[2] *password*[3] *ceqell\axa*[4] *edbprod*[5] *d:\mssql\data*[6] *d:\mssql\data*[7] *d:\mssql\data*[8]

Parameters:

1	script name to be executed
2	database administrator login
3	database administrator password
4	instance name (e.g <hostname>\axa)
5	database name
6	path where edb_lob filegroup will be placed
7	path where edb_tmp filegroup will be placed
8	path where edb_tmpidx filegroup will be placed

4.5. Pre-action activities on the original production environment

Since Eigner PLM 5.0 UIC <= 1000 and GIC <= 1000 are reserved for the standard development, existing users or groups using such C_IC / C_GIC must be migrated to an higher value.

This action must be executed only on production dump with PLM version older than Eigner PLM 5.0. This migration must be executed before you start any other upgrade activity.

It can be very time consuming! In a big customer dump it takes 1h/6 users. To solve the time conflict, break down the update into different subsets and try to run it in parallel.

To do this adapt the following statement in the SQL script:

```
INSERT INTO TEMP_U (OLD_U) SELECT C_IC FROM T_USER a
WHERE
C_IC > 200 AND C_IC < 1000 AND C_NAME NOT LIKE 'EDB%' AND
C_NAME NOT LIKE 'DEMOEP%';
```

To execute the UIC/GIC Migration run following commands:

```
sqlplus <user>/<password>@PLM

SQL> @update_customers_UIC.sql
```

On Microsoft SQL Server run following:

```
cmd> sqlcmd.exe -x -S <server> -U <user> -P <password>
        -i ..\mssql\sql\update_customers_UIC.sql
        -o ..\log\update_customers_UIC.log
```

5. Import Dumps

As next 3 Dumps have to be imported into the new database environment:

- ❑ *Source reference dump*
- ❑ *Target reference dump*
- ❑ *Customer dump*

For importing the dumps, do not change the tablespace names because the created table statements on tables containing a blob clause will fail if the original table spaces like EDB, EDB_IDX and EDB_LOB do not exist

To import dumps into the database you have to be familiar with your database environment and Oracle / SQL Server import utilities as well. Alternatively a batch script called *imp_dmp.cmd* can be used. This script uses dumps located in the directory *upgrade/dumps*.

5.1. Import source reference dump

Download appropriate source reference dump from **http://eignersupport.agilesoft.com/index.asp** and import it into the new database environment.

Example: import *plm50upgref.dmp* into a user named PLM50UPGREF

5.2. Import target reference dump

Download the newest target reference dump from **http://eignersupport.agilesoft.com/index.asp** and import it into the new database environment. Example: import *plm601upgref.dmp* into a user named PLM601UPGREF.

Note: Agile upgrade tool supports the latest ePLM version only.

5.3. Import customer dump

Export the current production environment into a dump file and import it in the new environment. Always work on this customer dump. Do not attach production dump during the upgrade process except for take over step described below.

This dump will be the new production dump after the upgrade is completed. Therefore give the new environment an expressive name

Note: On SQL server set recovery model to “simple” for customer database, which prevents an unlimited growth of database log files. Run following statement to set the recovery model:

```
ALTER DATABASE <database> SET RECOVERY SIMPLE WITH NO_WAIT
GO
ALTER DATABASE <database> set READ_COMMITTED_SNAPSHOT OFF
GO
```

5.4. Create statistics for all involved database schemas

- ❑ *Check Language settings*
  Because of an Oracle bug the setting for the environment variable NLS_LANG must be AMERICAN_AMERICA.WE8ISO8859P15. Otherwise statistics will not be computed correctly.
- ❑ *Login in **Sql*Plus** as user **sys** with **sysdba** privilege and perform analyzing.*

```
SQL> EXECUTE
DBMS_STATS.GATHER_SCHEMA_STATS(<DBUSER>,CASCADE =>true);
```

6. Running Upgrade Tool

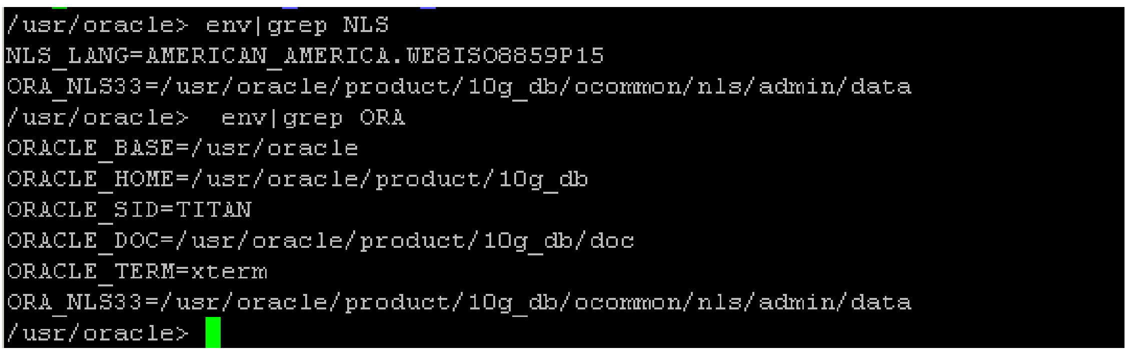
- ❑ Set a DISPLAY variable on UNIX
- ❑ Control the NLS_LANG setting in *upg_env.sh* (*upg_env.cmd* on windows) which specifies the client character set. “AMERICAN_AMERICA.WE8ISO8859P15” is a default setting for upgrade reference dumps.
- ❑ Adapt the following environment definitions in *upg_env.sh* on UNIX:

  JAVA_HOME - at least JRE 1.4.2 is required by upgrade tool. In the standard configuration of the file the *JRE of the Agile e6 installation is used for that.*

  ORACLE_HOME - make sure that Oracle 10.1.4 environment is set before proceeding with upgrade. *To check the environment for UNIX execute the following commands:*

```
user@host:~> env|grep ORA
```

The output should look like this:



```
/usr/oracle> env|grep NLS
NLS_LANG=AMERICAN_AMERICA.WE8ISO8859P15
ORA_NLS33=/usr/oracle/product/10g_db/ocommon/nls/admin/data
/usr/oracle>  env|grep ORA
ORACLE_BASE=/usr/oracle
ORACLE_HOME=/usr/oracle/product/10g_db
ORACLE_SID=TITAN
ORACLE_DOC=/usr/oracle/product/10g_db/doc
ORACLE_TERM=xterm
ORA_NLS33=/usr/oracle/product/10g_db/ocommon/nls/admin/data
/usr/oracle>
```

7. Configuring Upgrade Tool

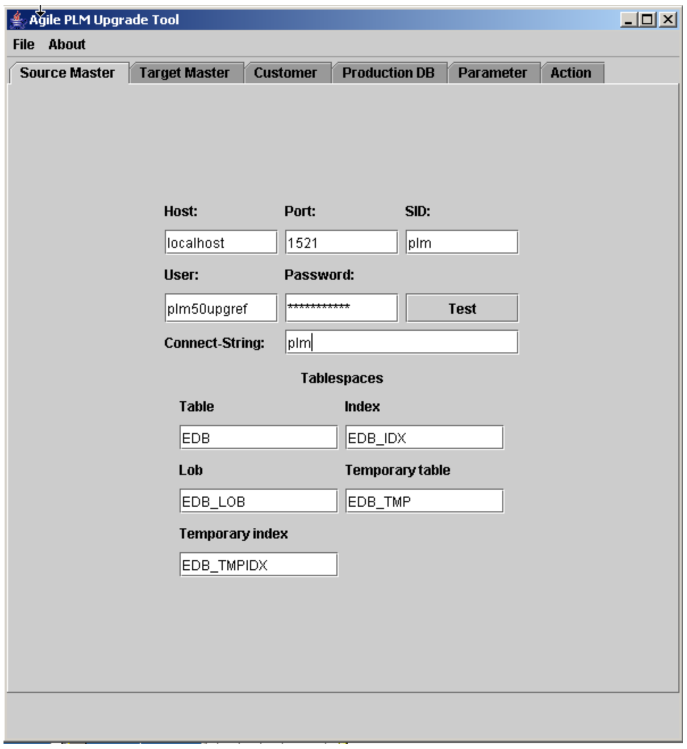
7.1. Configuring new database environment connections

Run start_upg.cmd (start_upg.sh on UNIX) and enter the following information for dumps in your new database environment. Each database connection can be tested with a "TEST" button on an appropriate tab.

Note: Press return after every change in a field. The color turns back from red to black. Otherwise the changes will be lost.

Note: Use file group names in Tablespace fields for Microsoft SQL Server. These are case sensitive, you have to use lower case therefore.

Note: If you want to connect to a Microsoft SQL Server named instance, please specify a correct *servername\instancename* in the field "*server*" as well as the corresponding TCP port number. You can find the appropriate port configuration within the Configuration Manager.



Host: localhost
Port: 1521
SID: plm
User: plm50upgref
Password: ************
Connect-String: plm

Tablespaces
Table: EDB
Index: EDB_IDX
Lob: EDB_LOB
Temporary table: EDB_TMP
Temporary index: EDB_TMPIDX

Parameter	Description
Host	Host name of database server
Port	Port number of Oracle listener (default 1521) or SQL Server port number (default 1433)
SID	Oracle_SID (uppercase) or database name for SQL Server (lowercase)
User	Database user name
Password	Password of database user
Connection String	Service name, which is used to run SQL*PLUS commands on the machine the upgrade tool is installed on. Use fully qualified name including the network domain, i.e. plm60.agile.agilesoft.com Note: the service name cannot be tested in the current version of upgrade tool

Please adapt tablespace names for each database connection since they are used, i.e for creation of new database objects or running SQL scripts:

Table	Default EDB*
Index	Default EDB_IDX*
LOB	Default EDB_LOB*
Temporary table	Default EDB_TMP* (edb on SQL Server until axalant 2000 SP3)
Temporary index	EDB_TMPIDX* (edb_idx on SQL Server until axalant 2000 Sp3)

Lower case for SQL Server, upper case for Oracle.

7.2. Configuring production dump connection

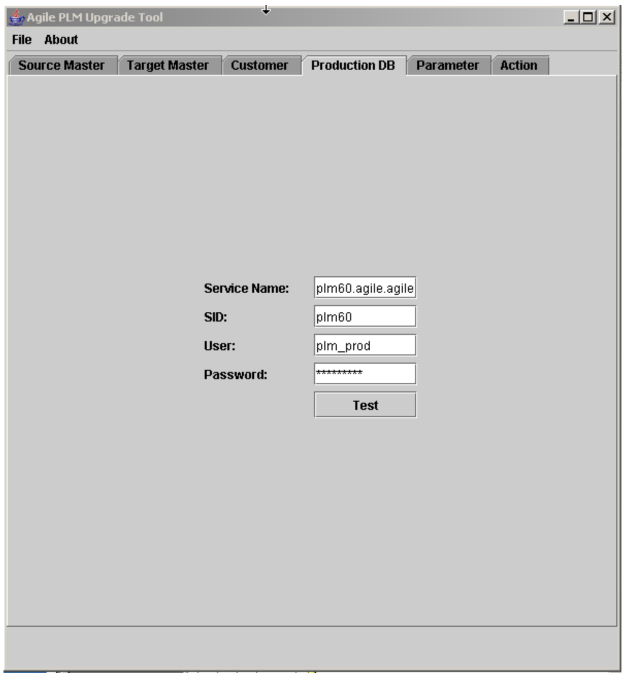
Enter a database connection for the current production dump. This connection will be used at the end of the upgrade process in the “Take over” phase. The definition of this connection is different, because it’s implemented as a database link, which is temporary created in your new customer dump. Creation of the database link can be tested with the “TEST” button.
This connection won’t be used until the Takeover phase of the upgrade process.

Note: Press return after every change in a field. The color turns back from red to black. Otherwise the changes will be lost.

Note: A named instance can also be used for a production environment connection. Please enter “*servername\instancename*” in the *Server* field in this case.

Note: For testing this connection, grant access to currently production environment for the cutomer database user running these statements as Microsoft SQL Server Administrator like SA.
<login_name> is the customer database login.

```
GRANT ALTER ANY LOGIN TO <login_name>
GO
GRANT ALTER ANY LINKED SERVER TO <login_name>
GO
```



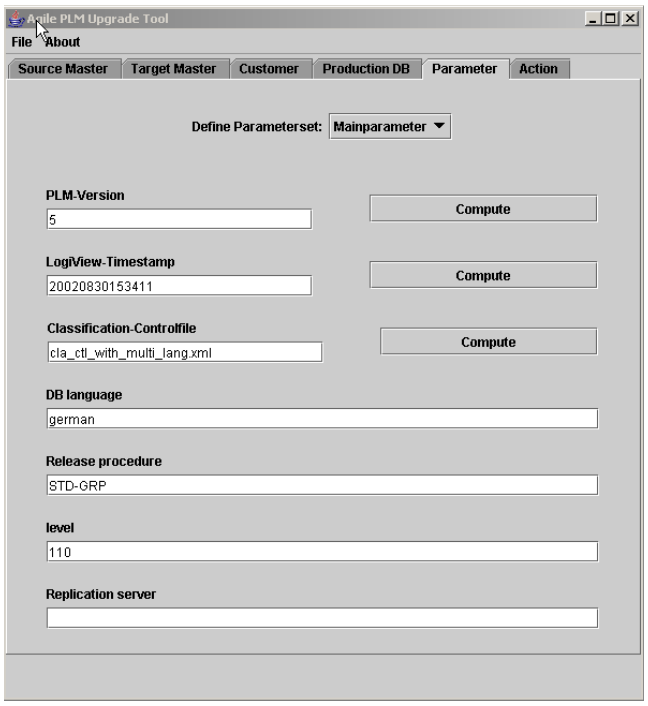
Service Name: plm60.agile.agile
SID: plm60
User: plm_prod
Password: *********

Parameter	Description
Service Name	Oracle service name including network domain i.e. AGILE.AGILESOFT.COM. Service name must be defined in tnsnames.ora
SID	Oracle_SID (uppercase) or database name for SQL Server (lowercase)
User	Database user name
Password	Password of database user

7.3. Setting up upgrade tool parameters

Review and correct the entries if necessary and check the following table for valid entries. With the “Compute” button, the right values can be determined. Always check the computed values.

Note: Press return after every change in a field. The color turns back from red to black. Otherwise the changes will be lost.



Define Parameterset: Mainparameter

PLM-Version: 5

LogiView-Timestamp: 20020830153411

Classification-Controlfile: cla_ctl_with_multi_lang.xml

DB language: german

Release procedure: STD-GRP

level: 110

Replication server:

PLM-Version	The customer dump version (before e6 upgrade process) Following values are valid: 1 = CADIB/EDB 2.3.x 2 = AXALANT SP1 3 = AXALANT SP2 4 = AXALANT SP3 5 = PLM 5.0 6 = e6.0 LA 7 = e6.0 GA 8 = e6.0.1 9 = e6.0.2

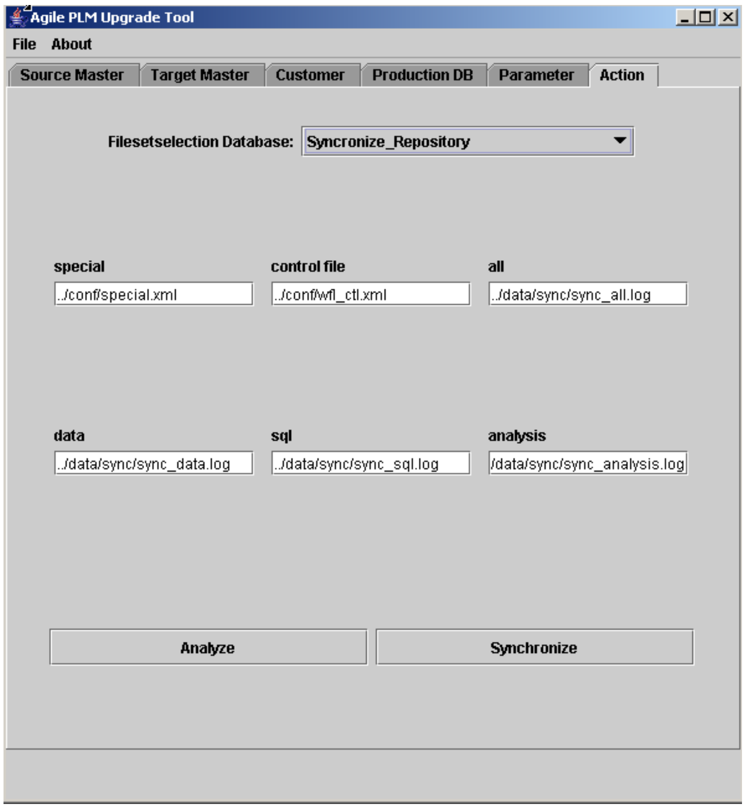
LogiView Timestamp	All LogiView items with a change date after this time point will be deleted. You can adapt this value manually. Following values are possible: CADIM/EDB 2.3.2 – 19990329094555 CADIM/EDB 2.3#3 – 19990707174038 CADIM/EDB 2.3#4 – 19990707174038 CADIM/EDB 2.3#5 – 20000329161725 axalant2000 SP1 – 20001109140557 axalant2000 SP2 – 20010723102350 axalant2000 SP3 – 20011113092600 axa2000 SP3 PA1 – 20020808110309 Eigner PLM 5.0 - 20020830153411 Agile e6.0 LA - 20050414160530 Agile e6.0 GA - 20050615170000 Agile e6.0.1 - 20051111135800 Agile e6.0.2 - 20060630200000
Classification – Control file	A file name of the control file for the customer dump in the present case Valid entries are: cla_ctl.xml cla_ctl_with_multi_lang.xml cla_ctl_with_multi_lang_repl.xml cla_ctl_with_repl.xml
Database Language	Language for the database dump. This influences the migration of the classification date. Values: German, English Default: German
Level	Status that is set during classification upgrade for records in the tables T_CLA_DAT (pool attributes), T_GROUP_DAT (classes)
Replication server	The valid name of the database server should in case of an implemented database replication to the environment be migrated.

7.4. Saving configuration

Save the current configuration by selecting File / Save parameter.

8. Performing upgrade in interactive mode

This chapter describes how the Customizing Upgrade is executed interactively. There are different steps available for selection on the **ACTION** tab, which should be executed in the respective order.



Filesetselection Database: Syncronize_Repository

special: ../conf/special.xml
control file: ../conf/wfl_ctl.xml
all: ../data/sync/sync_all.log
data: ../data/sync/sync_data.log
sql: ../data/sync/sync_sql.log
analysis: /data/sync/sync_analysis.log

8.1. Step "run preaction-scripts"

The Command script upg03_preaction_sql.cmd will be executed. This script executes a couple of SQL scripts (depending on the customer dump version). Log files created in this step in the upgrade/log directory have to be reviewed manually before proceeding with upgrade.

A complete list of SQL / LOG files created in this step can be found in the annex.

8.2. Step “DTV-upgrade"

This step upgrades the Dataview repository and is generally split into two steps:

- *Comparison of the data sets from the different dumps and storing the changes in an XML file for the three possible operations: delete, insert and update:*
  dtv_del.xml, dtv_ins.xml, dtv_upd.xml
  The Upgrade Tool selects each row from the Source and the Target reference dumps, compares the data sets from both dumps to identify the differences and checks if the customer has modified these data. The upgrade action (Insert, Update, Delete) is determined for each record and the information is stored in a set of XML files. The migration rules are listed in the Annex.
  This step takes around 10 min - 4 hours.
- *Performing operations. The Upgrade Tool reads the XML files created during the previous operation and performs the corresponding SQL-statements. This step takes about 15 - 40 min. During performing a error log file* data/dtv/dtv_err.xml *is created and should be checked for possible errors.*
- *A* dtv_customizing.log *is created in the directory* data/dtv/ *also. It contains conflicts, which were occurred during the DTV-upgrade. This log file must be reviewed.*

Check log files in the directory *upgrade\data\dtv*. If an error occurs, check the error description in *upgrade\log\erorrdetails.log*.

Note: The Upgrade Tool is only able to compare tables with the same table structure. Therefore the Dataview tables in the reference dumps (edb234upgref …) have Agile e6 structure.

Note: For a better readability, it is possible to create HTML files from the XML log files (see Convert XML files to HTML).

8.3. Step “run before-sync-scripts”

The command script *upg07_sync_update.cmd / upg07_sync_update.sh* will be executed. This script executes a SQL script (depending on the customer dump version). Log files created in this step in the upgrade/log directory have to be reviewed manually before proceeding with upgrade.

A complete list of SQL / LOG files created in this step can be found in the annex.

8.4. Step "Synchronize_Repository"

During this step the table definition in Dataview within the customer dump is compared to the physical table structure in the database. SQL statements to create and alter database objects are generated and executed automatically. The adaptation of the physical data structure will consist of the following steps:

- ❑ *Analyze phase*
  In this step you can see which statements the program will perform. It creates also a control file named conf/special.xml *for field values and special tasks. This step takes about 1-6 min.*
- ❑ *Review and edit the control file* conf/special.xml
- ❑ *Synchronize phase*
  In this step, database structure is converted according to the new Dataview repository. This step takes about 1 min – 4 std.

The data and log files are placed in the *data\sync* directory. A detailed description of the errors can be found in *log\errordetail.log*.

Note: If the program terminates the process and releases the connection because of a server error, drop the *special.xml* file in *conf/* directory and copy the preconfigured template from *conf/template* into the *conf/* directory. Restart the process.

8.4.1. Analyze

In this step, a proposal *special.xml* file will be written to *conf/* directory.

8.4.2. Configure special.xml for synchronizing repository

The delivery package contains a preconfigured *special.xml,* which defines standard setting for all expected cases. Very often the customer dump contains inconsistencies, so that in the analyze mode the tool will add entries to the special.xml file. In this case you need to review and adapt following configuration subsets:

Note: Do not change or delete the default settings.

- *Static default values for columns changed from null to not null*
  In the following example the column T_TRE_DAT.CUR_FLAG is set to 'n'.

```
<FieldDefault>
   <FieldName>T_TRE_DAT.CUR_FLAG</FieldName>
   <FieldType>S</FieldType>
   <FieldSize>1</FieldSize>
   <DefaultValue>
      <Value>n</Value>
   </DefaultValue>
</FieldDefault>
```

- *Dynamic default values*
  The field values can be computed dynamically based on a Java function / SQL Statement. Preconfigured function to use the number server to set values are available. Here an example for both SQL Statement / JAVA generated field default:

```
<FieldDefault>
   <FieldName>T_CTX_DAT.EDB_SEQ</FieldName>
   <FieldType>I</FieldType>
   <FieldSize>4</FieldSize>
   <DefaultValue>
      <Select>
         DISTINCT (SELECT COUNT(*) FROM T_CTX_DAT T WHERE T.C_ID &lt;=  thisRec.C_ID)*10
      </Select>
      <Where>C_ID &gt; 0</Where>
   </DefaultValue>
</FieldDefault>

<FieldDefault>
   <FieldName>T_MASTER_DOC.EDB_ID</FieldName>
   <FieldType>I</FieldType>
   <FieldSize>10</FieldSize>
   <DefaultValue>
      <Function>GetNewEDBID(EDBEDBID)</Function>
   </DefaultValue>
</FieldDefault>
```

- ❑ *Rename tables*
  If you have used DFM already, the following tables must be renamed (for upgrade from CADIM to Agile 6 only).

  T_EER_SIT
  T_EER_SIT_STR
  T_EER_SIT_MED

```
<RenameTable>
   <TableName>T_EER_SIT</TableName>
   <NewTableName>T_DDM_SIT</NewTableName>
</RenameTable>
<RenameTable>
   <TableName>T_EER_SIT_STR</TableName>
   <NewTableName>T_DDM_SIT_STR</NewTableName>
</RenameTable>
```

- ❑ *Move fields*
  This option allows to move a column of a table inclusive stored values to a new location. To move a field you have to specify:
  1. Source field (<table_name>.<column_name>)
  2. Target field (<table_name>.<column_name>)and
  3. Path (join condition between old and new table)

  The following sample configuration files show 3 different possibilities to move field values to a new location.

```
<!-- Example transfer from typetable to entitytable. -->
<MoveField>
     <SourceField>T_DOC_DRW.CRE_USER</SourceField>
     <Path>T_DOC_DRW.C_ID_2</Path>
     <Path>T_DOC_DAT.C_ID</Path>
     <DestField>T_DOC_DAT.CAX_CRE_SYSTEM</DestField>
</MoveField>

<!-- Example transfer from entitytable to entitytable via releationtable. -->
<MoveField>
     <SourceField>T_MASTER_DAT.PART_ID</SourceField>
     <Path>T_MASTER_DAT.C_ID</Path>
     <Path>T_MASTER_DOC.C_ID_1</Path>
     <Path>T_MASTER_DOC.C_ID_2</Path>
     <Path>T_DOC_DAT.C_ID</Path>
     <DestField>T_DOC_DAT.CAX_CRE_SYSTEM</DestField>
</MoveField>

<!-- Example transfer in table. -->
<MoveField>
     <SourceField>T_MASTER_DAT.PART_ID</SourceField>
     <DestField>T_MASTER_DAT.EDB_ICON</DestField>
</MoveField>
```

If you have a CAX interface installed already, please check if one of the columns used to store CAX specific information is defined as a document-type-table-column. These columns are now part of the standard data-model and included in the document master table T_DOC_DAT (migration from CADIM to Agile e6).

An example configuration file special_move.xml containing definition of moved fields are stored in the template directory conf/template/

- ❑ *Change data type of a field*
  *The upgrade tool allows changing the type definition of columns like integer to string. If the value of a column for all records is null then also incompatible data type changes can be executed, for example string to integer. Cutting a string field is only possible if no record contains a longer value. Please check max length of stored values directly within Sql*Plus.*

  You have to replace "false" by "true" to confirm such critical changes. The type definition "oldType" comes from the database; "newType" is the Dataview definition (stored in T_FIELD. C_FORMAT).

```
<FieldChange>
 <FieldName>T_DOC_DAT.FOO</FieldName>
 <ConfirmChange oldType="S80.0" newType="S40.0">false</ConfirmChange>
</FieldChange>
```

8.4.3. Synchronize

In this step, dump structure will be adapted to Dataview repository using the *special.xml* entries to fill NOT NULL fields.

8.5. Step "run after-sync-script"

The command script *upg08_postactions* will be executed. This script executes a set of SQL scripts. Check the results in the log files named *log/08_*.log*. For a complete list of log files please refer the Annex.

During this upgrade step a valid reference to T_STA_LUT is established with values in EDB_STA_REF for each entry in T_CHK_STA. Additionally EDB_LEVEL within the table T_STA_LUT is filled.

Note: The values of the table T_STA_LUT must be reviewed after the customizing upgrade.

Additionally BVB_ARTIKEL clean up step is executed. For more information please refer to "13.2 Cleanup BVB_ARTIKEL".

Note: PLM5 and older Favorites are migrated in this upgrade step as well. For more information please refer to "14.5 Favorites upgrade".

Note: Favorites migration can be performed after the first takeover execution only because it needs standard browser entries, which are inserted later within the step "BRW-upgrade".

And finally T_PRC_DAT.EDB_PRO_REF column will be filled in within this upgrade step. For more information please refer to "13.4 Project references in processes".

8.6. Step “run before-common-scripts”

The command script *upg09_common_get* will be executed. This script starts the SQL script to save and delete standard LogiView Models. Check log files named *log\09%.log*

8.7. Step “EDB-upgrade” (Configuration)

During this step the content of the common Agile configuration tables will be upgraded. The appropriate xml files are stored in *data/edb/* directory.

- ❑ *Click on the button "create files". XML files* edb_ins.xml, edb_del.xml *and* edb_upd.xml *will be created. They can be reviewed before performing these actions on the database dump.*
- ❑ *Click on the button "Perform delete,insert,update". XML files created in the previous step will be read and executed. Errors occurred during these actions will be stored in edb_err.xml – please control this file.*

If an error occurs, check the error description in *upgrade\log\erorrdetails.log*.

8.8. Step BRW-upgrade (Browser)

During this step the content of the browser configuration tables will be upgraded. The appropriate xml files are stored in *data/brw/* directory.

- ❑ *Click on the button "create files". XML files* brw_ins.xml, brw_del.xml *and brw_*upd.xml *will be created. They can be reviewed before performing these actions on the database dump.*

Click on the button "Perform delete,insert,update". XML files created in the previous step will be read and executed. Errors occurred during these actions will be stored in brw_err.xml – please control this file.

If an error occurs, check the error description in *upgrade\log\erorrdetails.log*.

8.9. Step DODE-upgrade (Print Studio)

During this step the content of the Print Studio configuration tables will be upgraded. The appropriate xml files are stored in *data/dode/* directory.

- ❑ *Click on the button "create files". XML files* dode_ins.xml, dode_del.xml *and* dode_upd.xml *will be created. They can be reviewed before performing these actions on the database dump.*
- ❑ *Click on the button "Perform delete,insert,update". XML files created in the previous step will be read and executed. Errors occurred during these actions will be stored in* dode*_err.xml – please control this file.*

If an error occurs, check the error description in *upgrade\log\erorrdetails.log*.

8.10. Step LGV-upgrade (LogiView)

Several standard LogiView procedures are changed in each new Agile PLM software release. All LogiView logic models will be deleted – if they are identical to new ones respectively saved if they were changed. As next all new logical model will be re-inserted. After the upgrade all LogiView changes made by customer must be made again.

Additionally standard LogiView variables, constants and system variables will be upgraded in the usual manner.

The appropriate xml files are stored in *data/lgv/* directory.

- ❑ *Click on the button "create files". XML files* lgv_ins.xml*,* lgv_del.xml *and* lgv_upd.xml *will be created. They can be reviewed before performing these actions on the database dump.*
- ❑ *Click on the button "Perform delete,insert,update". XML files created in the previous step will be read and executed. Errors occurred during these actions will be stored in* lgv*_err.xml – please control this file.*

If an error occurs, check the error description in *upgrade\log\erorrdetails.log*.

8.11. Step WFL-upgrade (Workflow)

During this step the content of the workflow configuration tables will be upgraded. The appropriate xml files are stored in *data/wfl/* directory.

- ❑ *Click on the button "create files". XML files* wfl_ins.xml*,* wfl_del.xml *and wfl*_upd.xml *will be created. They can be reviewed before performing these actions on the database dump.*
- ❑ *Click on the button "Perform delete,insert,update". XML files created in the previous step will be read and executed. Errors occurred during these actions will be stored in wfl_err.xml – please control this file.*

If an error occurs, check the error description in *upgrade\log\erorrdetails.log*.

8.12. Step CHG-upgrade (Change Management)

During this step the content of the change management configuration tables will be upgraded. The appropriate xml files are stored in *data/chg/* directory.

- ❑ *Click on the button "create files". XML files* chg_ins.xml*,* chg_del.xml *and chg*_upd.xml *will be created. They can be reviewed before performing these actions on the database dump.*
- ❑ *Click on the button "Perform delete,insert,update". XML files created in the previous step will be read and executed. Errors occurred during these actions will be stored in chg_err.xml – please control this file.*

If an error occurs, check the error description in *upgrade\log\erorrdetails.log*.

8.13. Step GTM-upgrade (Classification)

During this step the content of the classification configuration tables will be upgraded. The appropriate xml files are stored in *data/gtm/* directory.

- ❑ *Click on the button "create files". XML files* gtm_ins.xml, gtm_del.xml *and gtm_*upd.xml *will be created. They can be reviewed before performing these actions on the database dump.*
- ❑ *Click on the button "Perform delete,insert,update". XML files created in the previous step will be read and executed. Errors occurred during these actions will be stored in gtm_err.xml – please control this file.*

If an error occurs, check the error description in *upgrade\log\erorrdetails.log*.

8.14. Step GDM-upgrade (Office Suite)

During this step the content of the Office Suite configuration tables will be upgraded. The appropriate xml files are stored in *data/gtm/* directory.

- ❑ *Click on the button "create files". XML files* gdm_ins.xml, gdm_del.xml *and gdm_*upd.xml *will be created. They can be reviewed before performing these actions on the database dump.*
- ❑ *Click on the button "Perform delete,insert,update". XML files created in the previous step will be read and executed. Errors occurred during these actions will be stored in gdm_err.xml – please control this file.*

If an error occurs, check the error description in *upgrade\log\erorrdetails.log*.

8.15. Step RMT-upgrade (Requirement Management Traceability)

During this step the content of the RMT-module configuration tables will be upgraded. The appropriate xml files are stored in *data/rmt/* directory.

- ❑ *Click on the button "create files". XML files* rmt_ins.xml, rmt_del.xml *and rmt_*upd.xml *will be created. They can be reviewed before performing these actions on the database dump.*
- ❑ *Click on the button "Perform delete,insert,update". XML files created in the previous step will be read and executed. Errors occurred during these actions will be stored in gdm_err.xml – please control this file.*

If an error occurs, check the error description in *upgrade\log\erorrdetails.log*.

8.16. Step run after-common-scripts

The command script *upg10_common_update* will be executed. This script executes a SQL script to migrate non-standard browser entries to the new Agile e6 browser based on new database tables having a prefix T_EXP_*. Check the results in the log files named *log/10_*.log*. For a complete list of log files please refer the Annex.

This step can be skipped if you are migrating from Agile e6.0 or newer.

8.17. Classification

In Eigner PLM 5.0 a new concept – Attribute pool was introduced. This step converts attributes within the customer dump to this new concept. This step must be skipped, if you are migrating from PLM5.x version or newer.

- ❑ *Click on "Determine merge condition" – a special xml control file* data/cla/cla_merge_ctl.xml *will be created. It describes, how attributes are moved into a global attribute pool and contains so-called merge groups. All attributes mapped onto the same pool attribute are in the same merge group. Please review it and adapt if changes are necessary. This mapping file is used at the end of the Takeover phase to migrate current production data to existing attribute pool*
- ❑ *Click on "Perform mapping" – this will build attribute pool and perform mapping to item, documents etc.*

Please check log files located in the *data/cla/* directory.

For more information please refer to "13.7

Classification”.

8.18. Step “Classification attribute inheritance“

Since Agile e6.0 a new attribute inheritance concept was introduced. To convert existing class attributes, this upgrade step has to be executed. It’s not necessary if you are migrating from Agile e6.0 or newer.

8.19. Special replace

This optional upgrade step allows a string replacement within tables content.

This configuration file conf/specialreplace.xml contains an example definition for migrating from CADIM 2.3.x of substrings, which should be replace by another string.

Example: Update the strings ‘T_EER_SIT’ with ‘T_DDM_SIT’ in LogiView procedures

```
<?xml version="1.0" encoding="UTF-8"?>
<special>
    <replace>
            <table>LV_DT_PRC</table>
            <field>MAIN</field>
            <example>T_EER_SIT</example>
            <replacewith>T_DDM_SIT</replacewith>
    </replace>
</special>
```

8.20. Specific dump changes

Recreate customer specific Indexes, views, packages, procedures and triggers, database constraints like field defaults etc.

9. Performing upgrade in batch mode

After the upgrade is tested completely in the interactive mode, it can be used in batch mode. UNIX scripts for batch mode execution are available too. These have the *.sh* extension (instead of *.cmd* for Windows).

Note: Batch mode is intended for experienced upgrade users. Many upgrade steps are compressed in few batch scripts, so that they possibly need to be adapted manually.

The batch mode can be used in following manner:

- *Do an upgrade in interactive mode*
- *Re-import customer dump*
- *Start* start_upg.cmd *and control the configuration*
- *Clean the log directory by executing* upg01_pre_cleanlog.cmd
- *Run following scripts:*
  upg03_preaction_sql.cmd
  upg05_dtv_update.cmd
  upg07_sync_update.cmd
  upg08_postaction.cmd
  upg10_common_update.cmd
  upg11_cla.cmd
- *Control log files*
- *Take over production data by executing* upg13_prod1_takeover.cmd

Run post-action scripts by executing *upg14_prod2_rep_update.cmd.*

Alternatively a whole upgrade process may be performed in the batch mode:

- ❑ *Import customer dump*
- ❑ *Start* start_upg.cmd *and configure the upgrade tool.*
- ❑ *Run following scripts:*
  upg03_preaction_sql.cmd
  upg04_dtv_get.cmd
  upg05_dtv_update.cmd
  upg06_sync_get.cmd
- ❑ *Adapt* special.xml
- ❑ *Run following scripts:*
  upg07_sync_update.cmd
  upg08_postaction.cmd
  upg09_common_get.cmd
  upg10_common_update.cmd
  upg11_cla.cmd
- ❑ *Control log files*
- ❑ *Take over production data by executing* upg13_prod1_takeover.cmd
- ❑ *Run post-action scripts by executing* upg14_prod2_rep_update.cmd

10. Importing Dataview repository with DTV.BLD

Though the Dataview internal repository entries are handled by the upgrade tool, it is sometimes useful to import it from a DTV.BLD file. This is possible with the script *imp_dtv.cmd* / *imp_dtv.sh*.

However some parameters respective the current application server configuration must be adapted in the script before proceeding with the import.

Note: Since the *dtv.bld* file has to be located on the application server machine and there is no *dtv.bld* file available in the standard installation, this script can be executed on this machine only.

11. Convert XML files to HTML

You can convert the XML files to HTML and view them with a browser. The batch file xml2html.bat creates HTML files for insert, update and delete. This function is available on Windows platforms only.

Execute *xml2html.cmd* with one of the following parameter values, which specify for module the HTML file will be created:

- ❑ *all – HTML files for all modules will be created*
- ❑ *Module name - HTML files for specified module only will be created, possible values are:*

  dtv edb brw dode lgv wfl chg gdm rmt gtm

Note: You need memory for approximately sixth times the XML file size!
The batch job will run several hours. The Java Runtime Environment allocates 512MB of memory. You can adjust the memory allocation by editing the file *Upgrade\cmd\upg_env.cmd*

12. Takeover production data

The next step is to the transfer reference data from the production system. This phase is necessary, because during the upgrade and customizing of the new environment, new reference data (all tables except customized tables) is available in the old production dump.

Takeover phase is separated in several tasks:

- ❑ *Generate a relevant table list using a database connection to current production dump. This database connection is used as source for the tables copied into the new environment. No changes are made in the production database.*
- ❑ *Edit the list of tables, which should be copied during the takeover step*
- ❑ *Takeover: drop tables in the new dump and copy them from the current production dump*
- ❑ *Take over the newest number server values*
- ❑ *Make the dump up-to-date again by executing the Step "Synchronize_repository" and appropriate post-actions.*
- ❑ *Make test in the new environment. Let the user test all functionalities, maybe during training. If errors occur, remove them via customizing. Not everything might be done automatically. During the test period a lot of new data is created in the production system. Therefore the last two steps can be repeated till the new environment is error-free.*
- ❑ *If testing does not raise any error you can switch the production database to the new one Take over your data from the production system (and the files!) again and the new environment is your production system. Shut down your old system.*

12.1. Preaction on Microsoft SQL Server

Grant access to the current production environment for the cutomer database user running these statements as Microsoft SQL Server Administrator (e.g. SA).
<login_name> is the customer database login.

```
GRANT ALTER ANY LOGIN TO <login_name>
GO
GRANT ALTER ANY LINKED SERVER TO <login_name>
GO
```

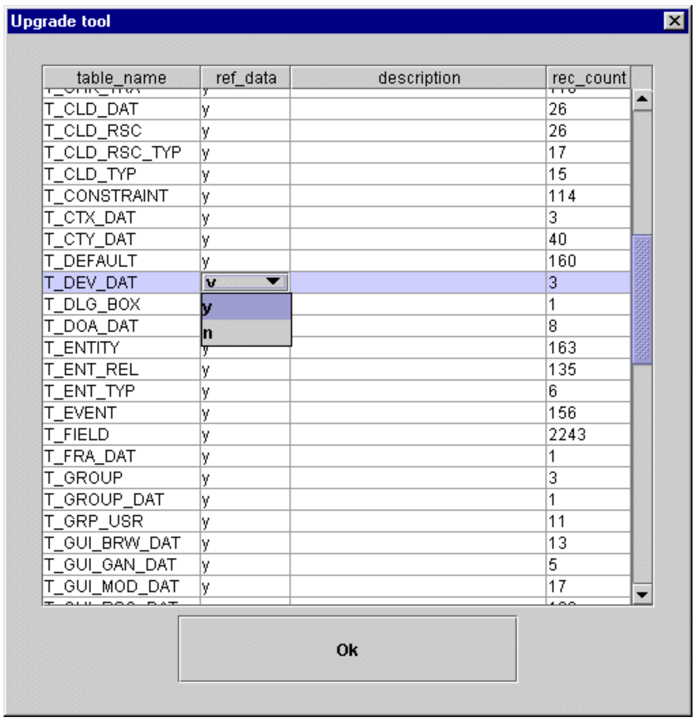
12.2. Define reference tables

Select folder **`Takeover`** in the Upgrade Tool and press the button **`Create Ref File.`** The Upgrade Tool will connect to the production database, identify all tables and synchronize the information with the predefined list (*ref_tables.xml*). Only tables with data will be written to the file.

12.3. Editing production table list

To adapt the list, press **`Edit ref. File.`** For each table you have to define, if it is a reference table. (ref_data=y.) Such reference tables will be dropped in the customer dump and copied in from the production system using the connection to the production DB.

Select OK to save the XML file (*conf/ref_tables.xml*).


Upgrade tool

table_name	ref_data	description	rec_count
T_CLD_DAT	y		26
T_CLD_RSC	y		26
T_CLD_RSC_TYP	y		17
T_CLD_TYP	y		15
T_CONSTRAINT	y		114
T_CTX_DAT	y		3
T_CTY_DAT	y		40
T_DEFAULT	y		160
T_DEV_DAT	y		3
T_DLG_BOX	y		1
T_DOA_DAT	n		8
T_ENTITY	y		163
T_ENT_REL	y		135
T_ENT_TYP	y		6
T_EVENT	y		156
T_FIELD	y		2243
T_FRA_DAT	y		1
T_GROUP	y		3
T_GROUP_DAT	y		1
T_GRP_USR	y		11
T_GUI_BRW_DAT	y		13
T_GUI_GAN_DAT	y		5
T_GUI_MOD_DAT	y		17

Note: Because of a special BVB_ARTMEH upgrade functionality `BVB_ARTIKEL, BVB_ARTMEH and BVB_ARTMEHUFK` tables have to be specified as production tables. Other BVB tables are treated as configuration tables. Otherwise an error can occur during execution of SQL scripts artmeh_1.sql / artmeh_2.sql.

Note: If you created new users and or groups since the date when the dump was exported form production system the following Dataview tables must be migrated. Attention: new user like EDB-WFL, EDB-DFM, EDB-DDM, EDB-GDM, EDB-EER, DODEKERNEL will be lost. Export these users first with the binary loader and reload them after the upgrade.

- *T_USER*
- *T_GROUP*
- *T_GRP_USR*
- *T_PROFILE*
- *Related tables of the PLM – person management*

Note: Table T_DEFAULT should be migrated by the loader (import/overload) otherwise new defaults will be missing.

12.4. Perform transfer

Backup your customer Agile e6 dump and Press the Takeover button. The tables containing non-repository information are dropped in your customers dump. The tables will be copied from the defined production environment into you customer dump.

Note: Check the log file *log/errordetails.log*

Note: After copying production database tables, a special upgrade step will be automatically executed – Takeover Workflow Masks. This step is described later in this document.

12.5. Takeover number server values

Since number server is used during the step "Synchronize_repository", the newest values have to be transferred to the new environment. This is done by executing the step "AFTER TAKEOVER: takeover number server values". Please control log files *14_01_get_numvalue.log* and *14_02_set_numvalue.log*.

12.6. Post-action scripts

Tables just copied from the current production environment have an old table structure and must be upgraded. This is done with the following steps:

- ❑ *Step "AFTER TAKEOVER: run before-sync-scripts"*
  The command script upg07_sync_update.cmd will be executed.
  Check log files upg07%.log in the upgrade\log directory
- ❑ *Step "AFTER TAKEOVER: Synchronize_repository"*
  execute "Synchronize" and control log/errors.log
  control log files in data\sync
- ❑ *Step "AFTER TAKEOVER > Run after-sync scripts"*
  The command script upg15_prod3_postaction.cmd will be executed.
  Check log files log/upg15%.log. *The complete list of sql scripts is described in the annex*

12.7. Classification (Stage 2)

It is required that the attributes are already migrated and so that only the classification lists will be updated. Click Perform mapping.

Note: If the customer has created new classes and attributes in the production system after the customization upgrade also classes, attributes and domain values must be copied and migrated.

12.8. Manual dump changes

12.8.1. LogiView

After the upgrade all LogiView changes made by customer in the standard must be made again. The previous changed standard procedures can be found within the S<timestamp> LogiView models.

12.8.2. STEP

After the upgrade or productive takeover you have to check the defaults "EDB-STP-DEF-NO-REF" and "EDB-STP-DEF-ORG-REF". During the upgrade to Agile e6 the values of these defaults were changed from "EP" to "NN". If the values of these defaults in your productive environment are already "NN", you have nothing to do.

These defaults are used as field defaults in several fields (<TABLE>.STEP_NO_REF, <TABLE>.STEP_ORG_REF).

This upgrade tool provides a default scripts (ora\sql\upg_org_ref_default.sql, ora\mssql\upg_org_ref_default.sql), which changes all values of these fields in the standard tables from "EP" to "NN".

Please review the script according to your customizing. If you didn't made any change to your customizing regarding STEP_ORG_REF,STEP_NO_REF you can run the scripts without changes.

If you don't run the script it is possible to create e.g. duplicate PART_ID's in T_MASTER_DAT, because these two fields are used in the uniqe key constraint of T_MASTER_DAT.

12.8.3. Customer specific Changes

Recreate customer specific views, packages, procedures and triggers, database constraints like field defaults etc.

12.9. Postactions on Microsoft SQL Server

Before switching the production system to the new database, some changes must be performed on a Microsoft SQL Server database:

Set recovery model and isolation level for customer database. These statements must be executed from an administrative account - like SA

```
ALTER DATABASE <database> SET RECOVERY FULL WITH NO_WAIT
GO
ALTER DATABASE <database> set READ_COMMITTED_SNAPSHOT ON
GO
```

And finally revoke access rights to old production database for the customer database user. These statements must be executed from an administrative account – e.g. as user SA
<login_name> is the customer database login.

```
REVOKE ALTER ANY LOGIN TO <login_name>
GO
REVOKE ALTER ANY LINKED SERVER TO <login_name>
GO
```

13. General Information

Following special handling modules are integrated into the upgrade tool. Therefore no additional actions are required to migrate the involved content. The steps described below are already integrated in the upgrade process and will be executed automatically.

13.1. Browser upgrade

With Agile e6.0 release a new browser was introduced. It's based on new database tables having a prefix T_EXP_*

During the upgrade standard content of this new tables is inserted. In Addition non-standard entries from the "old" Browser will be migrated.

13.2. Cleanup BVB_ARTIKEL

Some inconsistencies in the existing Item data structure will be eliminated during this specific upgrade step. Here a list of executed activities:

- ❑ *Superfluous BVB_ARTIKEL entries are dropped*
- ❑ *All BVB_ARTIKEL records without corresponding T_MASTER_DAT entries are deleted*
- ❑ *Missing BVB_ARTIKEL records are inserted*
- ❑ *Missing values of T_MASTER_DAT.UNIT are inserted into BVB_MASSEH*

13.3. STA_LUT

For each entry in T_CHK_STA a valid reference to T_STA_LUT is established with values in T_CHK_STA.EDB_STA_REF. Additionally EDB_LEVEL within the table T_STA_LUT is filled.

Note: The values of the table T_STA_LUT must be reviewed after the customizing upgrade.

13.4. Project references in processes

There is a known problem existing within the projects workflow in Agile e6.0. In case of creation of a relation between process and project, which has several versions several entries will be shown in the list because the PROJ_ID (kind of project name shortcut) is not unique over the stored versions. For this reason a new DB field were added: T_PRC_DAT.EDB_PRO_REF.

If necessary this field will be filled with values during the upgrade step upg08_postaction.

13.5. BVB_ARTMEH upgrade

During this step an upgrade of BVB_ARTMEH* tables is performed. SQL script artmeh_1.sql and artmeh_2.sql will be executed within the upg08_postaction step.

13.6. Favorites upgrade

The upgrade tool inserts several data into browser tables to make favorites and stored queries visible in the Agile e6 client browser window.

13.7. Classification

Overview Attribute concept (Old)

- *Attributes are defined class specific in the ATT concept*
- *Domain values for an attribute are defined in static menus*
- *No release procedures and status management for classes and attributes*

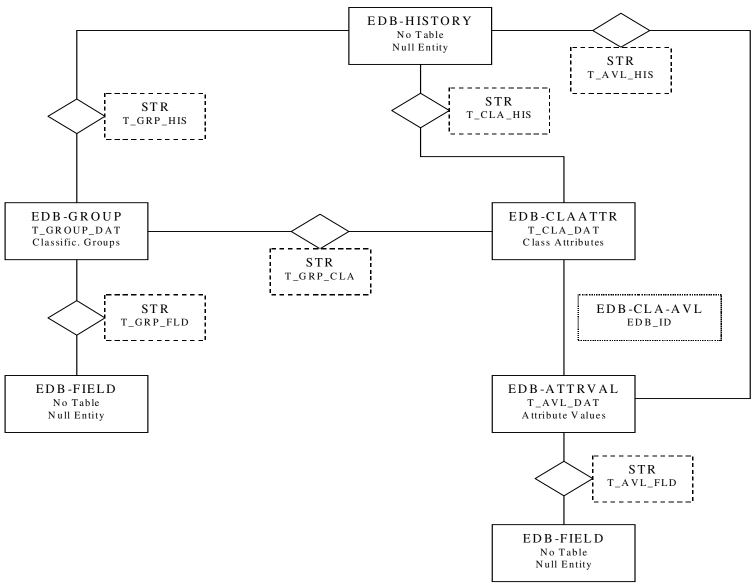
Overview new Pool concept

- *Attributes can be defined class independent*
- *Pool attributes can be assigned to more then one class*
- *Domain values for a pool attribute can be stored in special domain table*
- *For every class can be specified which domain value is valid*

The migration includes

- *Merge attribute definitions. Attributes are considered as identical if the following values are identical*
    1. C_LETTER
    2. C_TITLE
    3. C_TYPE

    If you have defined C-LETTER and C_TITLE as multi-language fields (standard since axalant 2000), then you have to define with the Parameter “DB language” which language is used as basic for the merge.
- *Initial Load of the attribute value pool including the activation of the attributes for special classes*
- *Update classification lists*
- *Update used field name*
- *Set of attribute ATT_VAL_REF in the classification Lists*
- *Update field and Mask definition (if you have defined own forms for classes).*



Note: If possible do not create or modify basic definition of classes and attributes between customization upgrade and take over data from production system. This influences the migration steps which must be executed after the takeover process:

- ❑ *No new classes and attributes are created. Only classification list tables must be defined as reference table.*
  1. T_GRP_ART
  2. T_GRP_DOC
  3. T_GRP_ORD
  4. T_GRP_PRO
- ❑ *Customers have created new classes and attributes in the production system after the customization upgrade. In addition to the classification list tables classes, attributes and domain values must be copied and migrated.*
  1. T_GROUP_DAT
  2. T_GROUP_STR
  3. T_GRP_FLD

14. Workflow Takeover

Since Upgrade Tool version 3.1.11 (upgrade to Agile e6.0.2) special generic workflow masks will be copied during the takeover phase.

After taking over of the common tables, workflow masks will be deleted, which are currently presented in the customer dump. As next generic workflow masks will be copied from the production database. Following tables are involved in this procedure:

- T_MASK
- T_MAS_FLD
- T_FIELD
- T_MENU
- T_MEN_SEL

Workflow masks have a special naming convention: EDB-WFL-<CID_OF_ACTIVITY>
Entries within T_FIELD follow naming rule EDB-DEC<CID_OF_ACTIVITY>
T_MEN_SEL entries contain names like EDB-SEL-<CID> and EDB-NSEL-<CID>
T_MENU entries of C_BUT_EDT / C_BUT_SEL / C_BUT_NOS only are considered for this action.

During the takeover phase, all statements will be generated and executed within the upgrade tool and additionally written to full.log file.

15. Appendix A: CMD scripts

Here is a short description of all shell scripts included in the upgrade download package.

Shell script	Description	Called SQL scripts
chown_mssql.cmd	(Re-) create a Login/User for an Microsoft SQL Server database	-
convert_it.cmd	Convert XML data file for a single table to HTML files. This script is called from xml2html.cmd.	--
exp_dmp.cmd	This script helps you to export oracle database to a dump file.	--
imp_dmp.cmd	This script helps you to create database users and import oracle dump files.	--
imp_dtv.cmd	Imports the Dataview repository. Please adapt parameters within before running this script	DEL_DTV.SQL > DEL_DTV.LOG
missing_f.cmd	Create missing file groups in MS SQL Server.	--
preaction_template .cmd	This file is for upgrade internal use only! It is used as a template for creating the file preaction.cmd, which is needed if all upgrade steps are executed in batch mode.	--
start_upg.cmd	Start upgrade user interface.	--
upg01_pre_cleanlo g.cmd	Cleanup all log files within current upgrade project.	--
upg03_preaction_s ql.cmd	Run several SQL scripts depending on source, target and customer product versions. Called SQL scripts are saved in the file 03_preaction_sql.log	source: CRE_REP_EDB.SQL > 03_01_CRE_REP_EDB.LOG source: TRUNC_LVTABS.SQL > 03_TRUNC_LVTABS.LOG source: GRANT_SELECT_T_CONSTRAINT.SQL > 03_16_GRANT_SELECT_T_CONSTRAINT.L OG target: GRANT_SELECT_T_CONSTRAINT.SQL > 03_15_GRANT_SELECT_T_CONSTRAINT.L OG

Shell script	Description	Called SQL scripts
		CRE_REP_EDB.SQL > 03_15_CRE_REP_EDB.LOG CLEANUP_C_ID_NULL.SQL > 03_03_CLEANUP_C_ID_NULL.LOG ANA_LV.SQL > 03_04_ANA_LV.LOG <=CADIM: ORA3-4.SQL > 03_05_ORA3-4.LOG <=AXA SP 1: PST10P2TOP3.SQL > 03_06_PST10P2TOP3.LOG <=AXA SP 1: ORA403-404.SQL > 03_07_ORA403-404.LOG <=AXA SP 1: AXASP1_TO_SP2.SQL > 03_08_AXASP1_TO_SP2.LOG <=AXA SP 3: DTV405-406.SQL > 03_10_DTV405-406.LOG <=AXA SP 3: UPD_T_SELECTION.SQL > 03_11_UPD_T_SELECTION.LOG <=e6 LA: DTV406-407.SQL > 03_12_DTV406-407.LOG <=e6 GA: DTV407-430.SQL > 03_13_DTV407-430.LOG <= e6.0.1: DTV430-431.SQL > 03_14_DTV430-431.LOG CUSTOMER_DATABASE_TASKS.SQL > 03_17_CUSTOMER_DATABASE_TASKS.LOG
upg04_dtv_get.cmd	Get XML files for the step “DTV-Upgrade” in shell mode.	--
upg05_dtv_update.cmd	Proceed XML files for the step “DTV-Upgrade” in shell mode.	--
upg06_sync_get.cmd	Run the step “Analyze repository” in shell mode.	
upg07_sync_update.cmd	Run the step “Synchronize repository” in shell mode. Called SQL scripts are saved in the file 07_sync_update.log	`<= PLM5.x: before_sync.sql > 07_01_before_sync.log`

Shell script	Description	Called SQL scripts

upg08_postaction.cmd	Run some SQL scripts, which are necessary after performing “Synchronize repository” upgrade step. Called SQL scripts are saved in the file 08_postaction.log	`cre_rep_edb.sql > 08_01_cre_rep_edb.log`
		`cleanup.sql > 08_02_cleanup.log`
		`db_defaults.sql > 08_03_db_defaults.log`
		`artmeh_1.sql > 08_04_artmeh_1.sql`
		`update_defartmehr.sql > 08_05_update_defartmehr.log`
		`artmeh_2.sql > 08_06_artmeh_2.sql`
		`special602.sql > 08_07_special602.log`
		`get_compile_all.sql > compile_all.sql`
		`compile_all.sql > 08_08_compile_all.log`
		`invalid_objects.sql > 08_09_invalid_objects.log`

upg09_common_get.cmd	Generate XML files for several upgrade steps, one for each common PLM module. Called SQL scripts are saved in the file 09_common_get.log	`del_and_save_lvmodel.sql > 09_01_del_and_save_lvmodel.log`

upg10_common_update.cmd	Proceed common PLM modules XML files. Called SQL scripts are saved in the file 10_common_update.log	`compare_lgv.sql > 10_01_compare_lgv.log`
		
		`<=PLM5.x: edb_explorer.sql > 10_02_edb_explorer.log`

upg11_cla.cmd	Upgrade PLM Classification.	--
upg13_prod1_takeover.cmd	Start a user interface to proceed with “Takeover production data”. Scripts upg14_prod2_rep_update and upg15_prod3_postaction have to be executed after that.	--

upg14_prod2_rep_update.cmd	Synchronize repository (this script includes all necessary pre-action and post-action calls).	`SQL Files of upg07_sync_update.cmd and upg08_postaction.cmd are called again`
		
		`get_numvalue.sql > 14_01_get_numvalue.log`
		`set_numvalue.sql > 14_02_set_numvalue.log`
		`cre_rep_edb.sql > 14_03_cre_rep_edb.log`

upg_env.cmd	Common upgrade settings, like Java, JRE, Path, etc.	--
xml2drop.cmd	Generates ora/ ref_data_tab.par and ora/ref_data_tab_drop.sql files for manually import/export of production tables. You have to configure which tables are relevant for the takeover step before.	--

xml2html.cmd	Converts generated XML files to HTML format for a module. Example: “xml2html.cmd dtv” Or “xml2html.cmd all”	`Possible module names ('all' for all modules):`
		
		`dtv edb brw dode lgv wfl chg gdm rmt gtm`

16. Appendix B: Configuration files in /conf/

- ***ApplicationParameter.xml*** *- Global application configuration file*
- ***brwDD.xml*** *- configuration file for upgrade module BRW (Explorer)*
- ***chgDD.xml*** *- configuration file for upgrade module CHG (Change Management)*
- ***wflDD.xml*** *- configuration file for upgrade module WFL (Workflow)*
- ***dodeDD.xml*** *- configuration file for upgrade module DODE (Print Studio)*
- ***dtvDD.xml*** *- configuration file for upgrade module DTV (Dataview Repository)*
- ***edbDD.xml*** *- configuration file for upgrade module EDB (Agile PLM configuration)*
- ***gdmDD.xml*** *- configuration file for upgrade module GDM (Office integration)*
- ***gtmDD.xml*** *- configuration file for upgrade module GTM*
- ***lgvDD.xml*** *- configuration file for upgrade module LGV (LogiView)*
- ***rmtDD.xml*** *- configuration file for upgrade module RMT*
- ***special.xml*** *- Configuration file for the step "Synchronize Repository"*
- ***specialreplace.xml*** *- A sample configuration file for special replace cases*
- ***ref_tables.xml*** *- Configuration file for the upgrade step "Takeover production data"*
- ***wfl_ctl.xml*** *- Configuration file for Workflow mapping*
- ***cla_ctl.xml*** *- Configuration file for Classification Upgrade (PLM5.x to Agile e6)*
- ***cla_post_ctl.xml*** *- Configuration file for Classification Upgrade (PLM5.x to Agile e6)*
- ***insert.xsl, delete.xsl, update.xsl, upgrade.xsl*** *- XSL-stylesheet for converting XML control files to HTML, used by xml2html.cmd script*
- ***ref_data_tab_drop.xsl*** *- XSL-stylesheet for generating SQL script, which drops production data tables in the customer dump. This stylesheet is used by xml2drop.cmd*
- ***ref_data_tab_par.xsl*** *- XSL-stylesheet for generating table list clause for oracle EXP command, which can be use alternatively to transfer production data tables from production database. This stylesheet is used by xml2drop.cmd*
- ***cla_stl.xsl*** *- XSL-stylesheet for Configuration file for Classification Upgrade (Axalant 2000 to PLM5.x), which generates a HTML output of performed mapping operations*
- ***dtv_dd.dtd*** *- Document type definition file for module control files*

17. Appendix C: Contents of the folder “upgrade/conf/template”

- ***ApplicationParameterORACLE.xml -*** *Upgrade tool settings file with standard values for an ORACLE database. Copy this file to upgrade/conf to reset the application settings.*
- ***ApplicationParameterMSSQL.xml -*** *Upgrade tool settings file with standard values for a Microsoft SQL Server database. Copy this file to upgrade/conf to reset the application settings.*
- ***cla_ctl_with_multi_lang.xml -*** *Example for the classification control file (with multi-language definition of the attributes).*
- ***cla_ctl_with_multi_lang_repl.xml -*** *Example for the classification control file (with multi language definition of the attributes and additional fields for database replication).*
- ***cla_ctl_with_repl.xml -*** *Example for the classification control file (with additional fields for database replication).*
- ***ref_tables.xml –*** *This configuration file is used during the takeover phase for read only reasons. Based on settings in this file, a ref_table.xml file is create in the CONF directory during the “Create ref. File” step.*
- ***special_move.xml -*** *Examples for the special case with table field moving.*
- ***special_rename.xml -*** *Examples for the special case with table field renaming.*
- ***Special.xml -*** *Default template.*
- ***specialreplace.xml -*** *Examples for special replace cases.*

18. Appendix D: SQL scripts

Here is a short description of SQL scripts delivered with the Agile Upgrade tool. All Script executions creates a log file in the *log/* directory which are named like the script itself with a prefix like 08.

SQL Script	Description
mssql/sql/after_restore.sql	This script is used by chown_mssql.cmd to setup the user/schema and default options within a just restored mssql database
ana_lv.sql	Analyze LogiView Content in the customer dump. Please control the log file of this script as described in the manual.
artmeh_1.sql	Convertion of BVB_ARTMEH* tables, for PLM version <= Eigner PLM 5.x (Part 1)
artmeh_2.sql	Convertion of BVB_ARTMEH* tables, for PLM version <= Eigner PLM 5.x (Part 2)
axasp1_to_sp2.sql	Upgrade axalant sp1 to axalant sp2.
before_sync.sql	This script has to be executed before running the step “Synchronize Repository”. It is done by default with the standard upgrade configuration. It prepares the Table T_STA_LUT and drops triggers, because otherwise it’s impossible to insert rows in the involved tables.
cleanup.sql	This script cleans up some dump content and is executed automatically after the step “Synchronize Repository”.
cleanup_c_id_null.sql	This script cleans up some inconsistencies in the customer dump (like rows with negative C_ID values). It must be executed before DTV-upgrade.
Compare_lgv.sql	This script compares logivew procedures in reference dump / customer dump. It is executed after the logiview upgrade.
customer_database_tasks.sql	This script executes some cleanup statements to get rid of common dump inconsistencies
cre_plm_tbs.sql	Creates missing oracle tablespaces.
cre_plm_usr.sql	Create a database user. This script need 2 Parameters: username and password
cre_rep_edb.sql	Create all schema object (tables, views, indexes, Packages, Triggers, Sequences etc.) and insert number server rows, the most of them already exist, so a lot of errors will be logged after executing this script.
db_defaults.sql	This script overwrites default constraints on the database

SQL Script	Description
	level, since they are different to Dataview default definitions. It's executed automatically after the step "Synchronize_repository"
del_and_save_lvmodel.sql	Delete standard LogiView content and save customized models with a prefix "SAVE-".
del_dtv.sql	Truncate all Dataview internal entries in DTV-tables.
mssql/sql/ dropall.sql	This script can be used to drop all objects within an existing mssql database
dtv405-406.sql	Pre-action script, has to be executed before DTV-upgrade for customer dump version <= axalant SP3.
dtv406-407.sql	Pre-action script, has to be executed before DTV-upgrade for customer dump version <= Agile e6.0 LA.
dtv407-430.sql	Pre-action script, has to be executed before DTV-upgrade for customer dump version <= Agile e6.0 GA.
Dtv430-431.sql	Pre-action script, has to be executed before DTV-upgrade for customer dump version <= Agile e6.0.1.
edb_explorer.sql	Converts DTV explorer to Agile e6 EDB-explorer. This step is executed once after common modules upgrade.
getoradrop.sql	Get script "dropall.sql" which cleans up a complete database schema.
get_compile_all.sql	Generates a script to recompile all db objects.
get_numvalue.sql	This script is executed in the production database and generates a file named " set_numvalue.sql " after takeover step. This file updates number server values in the customer database.
get_rebuildidx.sql	This script generates a file named " rebuildidx.sql " to rebuild all indexes in a right tablespace of a schema. It has 5 parameters for tablespaces: EDB EDB_IDX EDB_LOB EDB_TMP EDB_TMPIDX
grant_select_t_constraint.sql	This script grants a select on table T_CONSTRAINT for the customer Database. This permission is needed for constraint conversion.
Invalid_objects.sql	Lists all objects still invalid in the dump
levind_in_stalut.sql	This script is called automatically after "Synchronize Repository" and converts records in the table T_STA_LUT. It is needed only for upgrades form <= Eigner PLM to >= Agile e6.

SQL Script	Description
ora3-4.sql	This pre-action-script is called automatically if the customer dump is a CADIM dump.
ora403-404.sql	Pre-action script, has to be executed before DTV-upgrade for customer dump version <= axalant SP1.
Pst10P2ToP3.sql	Pre-action script, has to be executed before DTV-upgrade for customer dump version <= axalant SP1.
mssql/sql/ show_default_constraints.sql	This script can be used to examine default values within a mssql database
special602.sql	Special data modifications for upgrade to 6.0.2
trunc_lvtabs.sql	Truncate all LogiView tables. This script is executed on reference dumps only!
update_customers_UIC.sql	This script has to be executed on the customer dump before proceeding with upgrade
upd_t_selection.sql	This script is a workaround for incompatible changes for Table T_SELECTION in Eigner PLM5.0 This script is already executed on all reference dumps delivered with Agile Upgrade Tool. It will be automatically executed id necessary in the step “preaction-scripts”
update_defartmehr.sql	Fills DEFARTMEHR.BVB_ARTIKEL field during the CLEANUP_BVB phase of the step “Postaction”
mssql/sql/user_indexes.sql	creates a USER_INDEXES view (ORACLE-like) in a mssql database. This script is executed during the preaction phase of an upgrade
upg_org_ref_default.sql	A sample update script for existing STEP_NO_REF and STEP_ORG_REF values

19. Appendix E: Directories

Directory	Description
cmd	Windows 2000 shell scripts of the upgrade tool
conf	Configuration xml files
conf\template	Some templates of xml configuration files. The upgrade tool does not use these files. The only Exception is the file ref_tables.xml. It will be read by the tool to recreate /conf/ref_tables.xml
data	This directory contains several subdirectories, each for a module – like BRW, EDB etc. For each module delete, insert and update xml files are created. After performing of these operations on the customer database, error xml file will be written. Additionally html files generated for a module will be saved here. A file customizing.log in this directory contains conflicts caused by customizing of the original dump
data\dtv	Dataview Upgrade files as described above are stored here. Please read carefully the file customizing.log because it contains user-exit conflicts.
data\sync	log files of the synchronize repository upgrade step are stored in this directory
data\cla	log files of the classification upgrade step are stored in this directory
doc	Upgrade tool documentation
dumps	Database dumps can be stored here. Dumps, which are imported / exported by shell scripts *imp_dmp.cmd* and *exp_dmp.cmd* have to be stored in this directory
lib	Upgrade tool java executables
log	Log files of all sql scripts and common application log files
mssql\sql	MS SQL Server SQL-scripts
ora\sql	Oracle SQL-scripts
scripts	Unix / Linux shell scripts of the upgrade tool

20. Appendix F: Migration rules

Standard rules are available for insert, update und delete and these rules are verified during the comparison of the table contents. They can be overwritten by special definitions.

20.1. Standard Rules for Delete

Data records deleted in the standard are also deleted in the customer dump.

Customer-specific dump	Source master	Target master	Action
+	+	-	Delete

20.2. Standard Rules for Update

Data records existing in source master dump that were deleted in the customer dump are not re-created. Existing data records are updated. The standard changes overwrite the customer changes. Special rules apply on field level to protect customer-specific changes.

Customer-specific dump	Source master	Target master	Action
+	+	+	Update

20.3. Standard Rules for Insert

Data records not existing in source master dump or in the customer dump are added.

Customer-specific dump	Source master	Target master	Action
-	-	+	Insert

20.4. Special Rules

Customer changes that will not be overwritten by standard changes are:

- ❑ *Field defaults and check strings*
- ❑ *Customizing hints for fields containing user exits*
- ❑ *Special handling for mask components*
- ❑ *Replacements of strings ... (-> specialreplace.xml)*