SIEMENS

SIMATIC IT Unilab 6.7

Upgrading Unilab

Upgrade Manual

Edition 11/2014

A5E00428231-09

Guidelines

This manual contains notes of varying importance that should be read with care; i.e.:

Important

It contains essential information regarding handling the product, the product itself or a specific part of the documentation.

Note

Provides supplementary information regarding handling the product, the product itself or a specific part of the documentation.

Trademarks

Disclaimer of Liability

Siemens AG
Industry Sector
Postfach 48 48
90026 NÜRNBERG
GERMANY

A5E00428231-09
11/2014

Copyright © Siemens AG 2014
Technical data subject to change

Preface

Where is this manual valid?

This manual is valid for release 6.7 of SIMATIC IT Unilab.

Basic knowledge required

This guide is intended for SIMATIC IT Unilab users who are responsible for system configuration, such as application managers and system integrators (consultants). To be able to understand the concepts and examples discussed in this guide, the reader should at least have taken the SIMATIC IT Unilab Basic Training.

Purpose

This Upgrade Manual explains how to upgrade a **V6.4** version (with or without HotFixes installed) of the Unilab server and client to SIMATIC IT Unilab 6.7.

Important

The upgrade procedure only applies for a Unilab server **V6.7**. Older server installations need to be upgraded to **V6.7** first.

Please refer to the **SIMATIC IT Unilab – Prerequisites** installation manual for information regarding supported platforms.

Related documentation

The following documents contain information related to the content of this Upgrade Manual.

- Unilab Installation Manual
- Prerequisites installation manual
- Multiple Time Zone manual

These documents are all available online from the SIMATIC IT Unilab Documentation Library.

Conventions

The table below describes the specific typographic conventions that are used throughout this manual:

Symbol/Convention	Indicates...
E.g.	Where examples are given.
Text in bold	The names of menus, commands, dialog boxes and toolbar buttons and, in general, all strings (e.g. **File** menu; **Save** command).

Symbol/Convention	**Indicates...**
KEY1+KEY2	Shortcut keys, which permit rapid access to commands (e.g. CTRL+C).
UPPERCASE	The names of keyboard keys (e.g. RETURN key).
Italics	Noun with special importance or significance for which emphasis is needed. The names of parameters that must be replaced with a specific name or value.
>	A succession of commands in which the command preceding the symbol must be selected before the command following it.
`Code example`	Code example.

SIMATIC IT Documentation Library

The SIMATIC IT Unilab Documentation Library provides you with a comprehensive and user-friendly interface to access the overall product documentation where manuals and helps online can be browsed by functionality or by component.

Readme

The installation includes a readme file, which contains information on upgrade procedures and compatibility with previous releases. This file is supplied both in standard text (**Readme.wri**) and in Acrobat PDF (**Readme.pdf**) format.

This file is available in folder \ReleaseNotes of the setup DVD and is available from the SIMATIC IT Unilab Documentation Library.

SIMATIC IT Training Center

Siemens IA AS MES offers a number of training courses to familiarize you with the SIMATIC IT product suite. To successfully achieve this goal, training consists of lessons in both theory and practice.

Courses are held year-round, according to a program that is published well in advance of the first scheduled session.

The material on the basis of which our courses are conducted reflects the result of years of experience in process, LIMS,quality control and production management.

All courses are held by expert personnel that is aware of the developments and innovations in the Siemens IA AS MES product suite.

Courses are held in English at the Siemens IA AS MES Training Centers.

Upon request, training courses can also be organized on the customer's premises.

For more information on the training course calendar, please visit our technical web site (http://www.siemens.com/simatic-it/training).

SIMATIC IT Service & Support

A comprehensive Software Maintenance program is available with SIMATIC IT products. Software Maintenance includes the following services:

- **Software Update Service** (SUS): automatic distribution of upgrades and service packs
- **Technical Support Service** (TSS): support on technical problems with SIMATIC IT software (standard support and other optional services)
- **Online Support**: a technical web site, providing information such as Frequently Asked Questions and technical documentation on SIMATIC IT products

Software Update Service (SUS)

This service provides automatic shipment of new versions and service packs when released. When a new version / service pack is available for shipping, it is typically shipped within one month.

One copy of the installation DVD is shipped for each Server covered by Software Maintenance.

Hot fixes (officially tested and released) are not shipped and must be downloaded from the Technical Support Service web site.

Technical Support Service (TSS)

Siemens provides a dedicated technical support team for SIMATIC IT products.

The following options are available:

Bronze support: 9 hours/day, 5 days/week

Silver support: 24 hours/day, 5 days/week

Gold support: 24 hours/day, 7 days/week

The principal language of the SIMATIC IT hotline is English.

SIMATIC IT partners and customers covered by the Software Maintenance program are entitled to direct access to the TSS.

Access to TSS

To be able to access TSS, the customer needs to register as a user on the Technical Support web site. Connect to http//www.siemens.com/mes-simaticit/ and follow the **Technical Support Service** link.

The registration form must be completed with:

- Personal data
- The required company and plant information
- The Contract Number provided by Siemens Back Office when the contract is agreed.

Online Support

A customer who is a registered TSS user, can access the Technical Support web site (http://www.siemens.com/mes-simaticit/tss), which contains technical information such as:

- Service conditions (Phone numbers, Working hours, Reaction times,…)
- SIMATIC IT knowledge base: a technical support database that includes practical service solutions from Technical Support or the SIMATIC IT community
- SIMATIC IT software (e.g. hot fixes, software examples) and release notes that can be downloaded
- SIMATIC IT cross-industry libraries that can be downloaded (limited access to SIMATIC IT certified partners)
- SIMATIC IT product documentation that can be downloaded
- Frequently Asked Questions and useful tips.

Table of Contents

1 Introduction

This document describes the steps to upgrade **SIMATIC IT Unilab V6.4** to **SIMATIC IT Unilab V6.7**.

For upgrading versions prior to V6.4, please refer to the upgrade procedure for V6.4.

SIMATIC IT Unilab V6.7 is a Major Version: important technological innovations have been introduced, as well as major bug fixes. Available API calls and DB structures have been maintained (and eventually extended).

Important

The Unilab 6.4 license key cannot be used on Unilab 6.7.
These licenses must be upgraded with a powerpack license distributed by Siemens. The procedure to follow is described in the chapter "License Upgrade".

Step	Action	Refer to
1	Make sure all prerequisites are met. • Refer to *Simatic IT Unilab Prerequisites* in the Product Library for a detailed overview.	---
2	Server Upgrade	Chapter 2
3	Client Upgrade	Chapter 3
4	License Upload	Chapter 4
5	Recompilation and redistribution of unicf.dll (if applicable)	No details provided

2 Server Upgrade

2.1 Important notes on preparing/evaluating the upgrade

2.2 Introduction

In the following paragraphs, the server upgrade procedures are described. The server upgrade procedures depend on the Operating System: Unix (Linux, Oracle Linux, IBM AIX) or Windows (2008R2, 2012, 2012R2). It also depends on the actual Unilab version that is installed on your server. Before starting the upgrade, you should know:

- The operating system: Windows or Unix?
- The installed version of the Unilab server?

Which steps should be followed to upgrade?

1. Carefully read the Prerequisites
2. Prepare the upgrade scripts
3. Execute the General Preparations (specific to your OS)
4. Execute the procedures

2.3 Prerequisites and Warnings

- The Unilab 6.4 license keys cannot be used on Unilab 6.7.
  Typically a powerpack license is delivered to upgrade your 6.4 license to a 6.7 license. This must be performed with all licenses installed on the hard drive (not in the database). All details of the step-by-step procedure are provided in the chapter “License Upgrade” at the end of this document.
- Upgrade is only supported on the Oracle **server** platform: 11.1, 11.2, and 12.1.
- This procedure only supports upgrades starting from any 6.4 version (with or without HotFixes). For earlier versions, follow the upgrade procedures applicable to your version to obtain a 6.4 compatible version prior to following this procedure.

2.4 Prepare the Upgrade Scripts

2.4.1 Locate the Scripts on the CD

Some common scripts are located under **\Server Setup\db**. Copy the files (not the sub-directories) contained in this directory to a local directory (in this manual, this directory is called **<<server_scripts>>**).

Below is the setting used during the upgrade.

```
server_scripts :=
```

Other upgrade scripts to be used depend on the current version to upgrade and are located under **\Server Setup\Db\Upgrade\** on the CD (in this manual, this directory is called **<<upgrade_scripts>>**).

Below is the setting used during the upgrade.

```
upgrade_scripts :=
```

2.4.2 Prepare the Database for the Upgrade

2.4.2.1 Large redo log files

On very large databases, the upgrade can be performed in a faster way by increasing the size of the redo-log files (which should be located already on separate physical disks – refer to Oracle documentation).
Working with only one group of log files instead of two can also increase the speed considerably, but be sure to have a backup of your database if you plan proceeding in this manner.

You can adapt and execute the script **newbiglogfiles.sql.**

The following query returns the location and sizes of your **redo** log files:

```
SELECT a.GROUP#, a.member, b.bytes, b.members, b.status
FROM v$logfile a, v$log b
WHERE a.GROUP# = b.GROUP#
ORDER BY 1
```

Remember to recreate **redo** log files with the appropriate size after the upgrade.

2.4.2.2 Disabling triggers

It might be interesting to disable some triggers installed for the project during the upgrade. This should be evaluated for each project separately, since triggers are typically used to maintain consistency between tables.

2.5 General Preparations

2.5.1 Prepare a UNIX Environment

1. Prepare Unix environment for Unilab server upgrade

Check the following prerequisites:

- A user **Unilab** must be defined on the system. This user must have the appropriate permissions to create/delete the files and directories involved in the installation. Also, this user must be able to administrate the Unilab database (having two users with complementary permissions is another possibility).
- A **local_unilab_server** directory should exist on the server, containing installation scripts of the old Unilab version. Make a backup of the old **local_unilab_server** directory and remove the old directory.

  Below is the setting used during the upgrade.

```
local_unilab_server :=
```

Warning

Make sure your backup is on a safe location and includes the customized PL/SQL package scripts.

2. Transfer the Upgrade Files to the local_unilab_server Directory

The **<<server_scripts>>** and **<<upgrade_scripts>>** directories should be transferred on the server.

Transferring files can be done using an FTP-client application as in the following example (note that if you have a graphical oriented ftp-client things will be easier than this example):

Example: In the example, it is assumed that the name of the Unix system is **unixhost**. The files you want to transfer are located at **D:\unisetup** and must be copied to **/unilab/server**. We assume that the destination directory **/unilab/server** exists and can be written to by all (see previous paragraph). The commands you need to type are in bold.

```
D:\unisetup> ftp unixhost
Connected to unixhost.mydomain.be.
220 unixhost FTP server (Version 1.7.212.1 Sat Feb  1
01:30:15 GMT 1997) ready.
User (unixhost.mydomain.be:(none)): Unilab
331 Password required for Unilab.
Password:
230 User Unilab logged in.
ftp> pwd
257 "/home/myuser" is current directory.
ftp> cd unisetup
250 CWD command successful.
ftp> prompt
Interactive mode Off.
ftp> ascii
200 Type set to A.
ftp> mput *.cmd *.txt *.h *.sql *.run *.plb *.par *.pack
*.bat *.ksh *. *.java
200 PORT command successful.
150 Opening ASCII mode data connection for …
…
… (all specified files are being copied)
…
226 Transfer complete.
ftp> quit
```

FTP Problems

The above example worked when using the ftp software provided on Windows. Some of our customers experienced problems while transferring the **.plb** files with commercial or shareware ftp software. Be careful with the configuration of such tools: it could be that no error is issued during the transfer of a plb file, but that the package doesn’t get compiled. For the details concerning this problem, see the problem description provided below:

Description:
UNAPIRA5.PLB package doesn't get compiled when file transferred through ftp on Unix server. Error message in sys.dba_errors: PLS-00801: internal error [56104]
Bug rejected:
This is not a bug in fact. This is an error in the configuration of ftp software used to transfer the file from the CD to the Unix server. I experienced the same problem when working with Reflection FTP.PLB files are transferred in ASCII mode. In ASCII mode, Reflection FTP has some default translating options (convert tabs into spaces - a checkbox in the translation tab of the site properties). That translation is corrupting the encrypted file (.plb file)
UNAPIRA5 was the only package that is containing some tabs. This explains why only that package is concerned by the problem.

3. Be sure that all script names are in lowercase

During upgrade on Unix, all Unilab **sql** and **h** files transferred to the Unix server must be in lowercase. A script is provided in **upgrade_scripts** to convert all files in a directory to lowercase.

This script must be run after transferring all necessary files to the server (cfr. step 2). Be sure that also the conversion script has been transferred.

Change the current working directory to the one to which all files have been transferred and type the following command

```
$ ksh tolower.ksh
(if tolower.ksh is itself in uppercase, execute it as
follows):
$ ksh TOLOWER.KSH
```

This procedure will change ALL files in the **current** directory and its subdirectories to lowercase. Note that, if you have the same file written in a different case, you will end up with only one file in lowercase (the last one overwrites the previous one)

4. Check that you are about to upgrade the correct database

Login as a user with appropriate settings for connecting to the Unilab database.

Type the following commands:

```
$ ORACLE_SID=<<ORACLE_SID>>
$ .  /usr/local/bin/oraenv   (that is: 'dot' 'space'
/usr/local/bin/oraenv )
          (on AIX this becomes : /usr/lbin/oraenv)
```

Verify that the Unilab database you are about to upgrade has the correct **ORACLE_SID**. Execute the following query to check this.

```
select * from v$database
```

The result should comply with the **ORACLE_SID**> you already entered.

2.5.2 Prepare a Windows Environment

1. Definitions

In the following steps, the string **SQLPLUS** should be replaced with the appropriate command to start the Oracle DB Administration tool on the corresponding server OS platform.

E.g. for Oracle 11.0.0.6, **SQLPLUS = SQLPLUS /NOLOG** with **SET DEFINE OFF** as first executed statement.

Important

Packages will not compile when not performing **SET DEFINE OFF** in **sql*plus**.

2. Prepare Windows environment for Unilab server upgrade

Check the following prerequisites:

- A user **Administrator** with administrator privileges must exist (usually 'Administrator')

  Below is the setting used during the upgrade.

  ```
  Administrator :=
  ```

- A **local_unilab_server** directory should exist on the server, containing installation scripts of the old Unilab version. Make a backup of the old **local_unilab_server** directory and remove the old directory.

  Below is the setting used during the upgrade.

  ```
  local_unilab_server :=
  ```

Warning

Make sure your backup is on a safe location and includes the customized PL/SQL package scripts e.g. **C:\UNAB\Server Custom Packages**.

3. Transfer the required installation files to the server

The **<<server_scripts>>** and **<<upgrade_scripts>>** directories should be transferred on to the server.

Make sure that all copied files are modifiable and you are logged on as **Administrator**.

Log on as a user with appropriate settings for connecting to the Unilab database and check the **ORACLE_SID** of the database you are about to upgrade.

Below is the setting used during the upgrade.

```
ORACLE_SID :=
```

4. Check whether you are going to upgrade the correct database

Log on as a user with appropriate settings for connecting to the Unilab database.

Verify that the Unilab database you are going to upgrade has the correct ORACLE_SID with the following query:

```
select * from v$database
```

The result should comply with the **ORACLE_SID** you already entered.

2.6 Upgrade Procedures

This procedure only supports upgrades starting from any 6.4 version (with or without HotFixes).

2.6.1 Preparation steps

Some files used during the upgrade must be prepared before starting the upgrade by itself. These files must be prepared by replacing some expressions written between @ in a .txt file to generate a .sql file.

- Manually generate the 4 files **dbuvcoXX.sql**, **dbuvopXX.sql**, **allowapi.sql** and **dburapi.sql** by replacing **@DBANAME@** and **@PASSWORD@** with your local Unilab DBA name and DBA password in the 4 files **dbuvcoXX.txt, dbuvopXX.txt, allowapi.txt** and **dburapi.txt**.

  XX is 16, 32, 64 or 128, depending on the number of data domains installed on your database.

  Add the 4 generated scripts to the <<**server_scripts>>** directory (do not forget to transfer these files if you are working under Unix).

- Manually generate the file **upddba.sql** by replacing **@DBANAME@** with your local Unilab DBA in the file **upddba.txt**.

  Add the generated scripts to the **<<upgrade_scripts>>** directory (do not forget to transfer these files if you are working under Unix).

2.6.2 Upgrade steps

1. If your database is currently running with a cost-based optimizer (the following query should not return RULE if you are running cost-based: SELECT value FROM v$parameter WHERE name='optimizer_mode') , modify the init<SID>.ora file (or your spfile) to set the parameter **optimizer_index_cost_adj** to the value **1**. Stop and start the db to make the change become effective.
2. Set your current directory to <<server_scripts>>. All scripts in SQL*plus will be called with a relative path based on that directory

```
cd <<server_scripts>>
```

3. Unlock the Unilab dba user when locked. Connect as SYS to unlock the Unilab dba user.

```
sqlplus /nolog

connect SYS/<<sys password>> AS SYSDBA

ALTER USER <<unilab dba>> ACCOUNT UNLOCK

EXIT
```

4. Connect as Unilab dba user in <<SQLPLUS>>

```
sqlplus /nolog
connect <<unilab dba>>/<<unilab dba password>>
SET DEFINE OFF
```

5. Turn DBMS_OUTPUT on and spool output to a file.

```
SET SERVEROUTPUT ON SIZE 999999
SPOOL u67upgradestop.log
```

6. Stop the event manager and any job using the Unilab packages

```
@stopevmr (or BEGIN cxapp.stopalldbjobs; END;)
SPOOL OFF
EXIT
```

7. Connect as SYS to grant missing roles to Unilab dba user.

```
sqlplus /nolog

connect SYS/<<sys password>> AS SYSDBA

@.\upgrade\upddba

EXIT
```

8. Connect as Unilab dba user in <<SQLPLUS>>. Turn DBMS_OUTPUT on and spool output to a file.

```
sqlplus /nolog

connect <<unilab dba>>/<<unilab dba password>>

SET DEFINE OFF

SET SERVEROUTPUT ON SIZE 999999

SPOOL u67upgrade.log
```

9. Install the new database structures and alter the existing ones.

```
  @.\upgrade\newtables

  @.\upgrade\altertables

  @.\upgrade\upddata

  @.\upgrade\updweeknr

  Regenerate the data-domain views. Replace XX with the
number of data domains installed (16, 32, 64, 128).

  @dbuvcoXX
  @dbuvopXX
  SET SERVEROUT ON SIZE 999999
  SET DEFINE OFF
```

10. Regenerate the correct synonyms for archiving

- If you are upgrading an operational database where no remote archive database is used

```
@crsynoar
```

- If you are upgrading the archive database which is used as a remote archive database

```
@crsynoar
```

- If you are upgrading an operational database where a remote archive database is used

```
@crsynoop
```

Disregard the errors: "ORA-01434: private synonym to be dropped   does not exist.", the script will perform its job.

11. Compile the new packages

```
@upd_pack
```

Note1: If you get the error message “ORA-04031: unable to allocate 4032 bytes of shared memory ("shared pool","unknown object","joxlod: in ehe","ioc_allocate_pal") while compiling unapifile.java, increase the java_pool_size in the init<SID>.ora to 12582912, stop and start the db, and then compile unapifile.java

12. Compile all packages that have been marked as invalid by Oracle after running the last compilation.

```
@compinv
```

You may get the error message 'ORA-01555: snapshot too old: rollback segment number % with name "UNILAB_%" too small' while running the compinv script. This is due to internal Oracle procedures. Just restart the script until no more error messages are displayed.

13. Modify the custom packages as follows.

Reanalysis behavior in Uniconnect

Following step is optional and might already be implemented as part of Fix release in 6.4.0

- The custom package for the UNICONNECT has been modified.

If a method is completed and a new result is received by Unilink:

- In case P_SET_REANALYSIS is enabled, a reanalysis is performed and the new result is filled in.
- In case P_SET_REANALYSIS is disabled, the result will not be accepted and the file will not be processed.

The package body uconnect3.sql has been modified.

If you need this new functionality, you can find the necessary modifications by making DIFF between the UCONNECT3.SQL 6.4 source and 6.7 source. The source is located on the CD, in the server\db folder.

Reset password

Following step is mandatory but might already be implemented as part of fix release in 6.4.0

- The custom package UNUSER (header and body) has been modified because of the next error: reset password doesn’t work on 21CFR11 environment.

A new Functional Access Right “RESET PASSWORD (FOR OTHER USER)” was added to “Define User Profile”.

Security doesn’t allow us to change the password for another user. Sending a password over the network can be considered as a security leak. In case of ChangePassword, a ResetPassword was introduced. Behaviour is similar as the default behaviour during CreateUser.

The ResetPassword functional access right can only be set on User level, and not on UserProfile level.

This new custom function, available as API in UNUSER. ResetPassword, will reset the password and send a temporary password to the user (same logic as CreateUser). If needed, this custom function can be changed, based on the needs of the project (no expiration, new default password).

Thus, the new ResetPassword function was defined in the CxApp package, and there is an example available for it on the server/db folder.

Dedicated EventManager

Following step is mandatory but might already be implemented as part of fix release in 6.4.0

- EventManagers and dedicated EventManagers will now have a unique id used to broadcast events.
Procedure cxapp.sql to start dedicated eventmanagers has been modified.

Modifying the custom package Cxapp.StartAllDbJobs / Cxapp.StopAllDbJobs needed in case dedicated manager is used.

If you need this new functionality, you can find the necessary modifications by making DIFF between CXAPP.SQL 6.4 source and 6.7 source. The source is located on the CD, in the server\db folder.

Appendix A describes the steps to Start and Stop the Dedicated Event Manager.

Method Group Key "InExecution" for improving performance

Following step is optional and might already be implemented as part of fix release in 6.4.0

An Equipment constant is used as the input_source of a cell in the Method Sheet.

Updating this constant will evaluate all open Method Sheets and update with the new value.

For increasing the performance a new solution was introduced and can be enabled by setting the Database Package variable UNAPIEQ. P_USE_MEGKINEXECUTION4EQCTE to TRUE. This requires a new method group key INEXECUTION. The new behavior is disabled by default. Follow the detailed installation instructions if you want to enable the feature.

Implement Method Group key "InExecution"

This installation procedure is only needed when you want to use the optimized query, used for updating Equipment Constants.

The way the query should be executed depends on the selectivity of the Equipment constant, the amount of open Methods, and other relations in the data. This makes it impossible to optimize/tune the query in a generic way.

To handle this issue, a new Method Group Key has been introduced: InExecution. This Group Key is used to retrieve all Open Methods in the most efficient and performant way. This restricted set of data is used to update all methods having the same equipment constant.

Requirements: The installation of the Server part must be completed.

Technical Details:

- A new Method Group Key will be created : **InExecution**

- The Group Key will be assigned/unassigned using an EventRule

- Assignment : After creation of the method cells

```
       Obj_tp : me

       ev_tp : MeDetailsCreated

       dbapi_name : CreateScMeDetails

       af : AssignGroupKey('megk', 'InExecution', '1')
```

- Unassignment : Once the method is completed

```
       Obj_tp : me

       Condition: MeExecuted (new condition)

       Af : DeAssignGroupKey('megk', 'InExecution')
```

These EventRules are needed to assign/unassign the Groupkey in any possible situation. Additional project specific optimization might be considered (more restricted rules).

Installation Procedure:

Summary

- Enable functionality by changing Package variable.

- Create the new method Group key: InExecution.

- Add the new condition “MeExecuted”

- Add the new EventRules.

- Assign the Group key based on the data.

Installation procedure on the database server

1. Enable the new behavior:

Modify the source-file **unapieq.h** and change the value of the package variable P_USE_MEGKINEXECUTION4EQCTE from FALSE to TRUE

This becomes :

```
P_USE_MEGKINEXECUTION4EQCTE BOOLEAN DEFAULT TRUE;
```

Recompile Package header to enable the behavior

```
@unapieq.h
```

2. Add the new function MeExecuted in the Package UNCONDITION.

Extend the Package with the Function MeExecuted. The required package function header and body is provided in the file *megk_step1_newcondition.txt* under the folder UPGRADE.

3. Insert the EventRules by executing the next script

```
@.\upgrade\megk_step2_adddata
```

The script is located under the UPGRADE folder.

4. Verify that all packages are valid. If needed, recompile dependant packages.

```
SELECT object_name, object_type
FROM user_objects
WHERE status <> 'VALID' ;
```

5. After the installation, activate the database jobs.

```
BEGIN
cxapp.startalldbjobs;
END;
/
```

6. Create the new Groupkey InExecution.

```
@setcon4install
@.\upgrade\megk_step3_creategk
```

7. Apply the new Group key to the operational methods.

```
@setcon4install
@.\upgrade\megk_step4_assigngk.sql
```

8. After the installation, stop database jobs.

```
BEGIN
cxapp.stopalldbjobs;
END;
/
```

Versioning on database packages

Following feature is new since 6.7.0

- A new revision for packages was introduced.

The new GetVersion function was introduced in each custom package.

The function call to be included in the header of each custom package is:

```
FUNCTION GetVersion
RETURN VARCHAR2;
```

The function body to be included in each custom package is:

```
FUNCTION GetVersion
    RETURN VARCHAR2
```

```
        IS
        BEGIN
             RETURN('06.07.00.00_13.00');
        EXCEPTION
             WHEN OTHERS THEN
             RETURN (NULL);
        END GetVersion;
```

14. Compile the procedures that are marked as INVALID.

```
alter procedure minilink compile;
```

15. Extend utcf with the functions that have been added to the custom packages (this can take several minutes).

```
@updcf8
```

16. Adapt the version

```
@version
```

17. Regenerate the right synonyms for archiving (in <<server_scripts>> directory)

If you are upgrading an operational database where no remote archive database is used

```
@crsynoar.sql
```

If you are upgrading the archive database which is used as a remote archive database

```
@crsynoar.sql
```

If you are upgrading an operational database where a remote archive database is used

```
@crsynoop.sql
```

Ignore explicitly the errors: "ORA-01434: private synonym to be dropped does not exist.", the script will perform its job.

18. Regenerate Unilab user schema objects (the scripts dburapi and allowapi should have been generated during the preparation steps)

```
SET SERVEROUTPUT ON SIZE 999999
@dburapi
```

19. Regenerate all users part 1 (this can take several minutes).

```
@allowapi
```

20. Regenerate all users part 2

```
@rcrusers
```

21. Compile the views that are marked as INVALID.

```
@compviews.sql
```

22. Compile the packages that are marked as INVALID.

```
@compinv.sql
```

23. Verify if all packages are valid. If necessary, recompile dependant packages.

```
SELECT object_name, object_type
   FROM user_objects
   WHERE status <> 'VALID';
```

24. Start the event manager

```
@strtevmr (or BEGIN cxapp.startalldbjobs; END;)
```

25. Regenerate the object structures for group-key tables that can be used for reporting purposes. (Please ignore the error that may be returned by the SetConnection call if you don't have the new license installed.)

```
@CreateUserStructsForAllGks.sql
```

26. Generate the shared schemas that permit support of the new Enterprise Users functionality.

```
@rcrsharedschemas.sql
```

27. Lock the Unilab dba user when necessary.

```
sqlplus /nolog
connect SYS/<<sys password>> AS SYSDBA
SET DEFINE OFF
ALTER USER <<unilab dba>> ACCOUNT LOCK
```

Upgrade completed

Execute the following statement in <<SQLPLUS>> to complete the upgrade:

```
SPOOL OFF
EXIT
```

Unix Note

If you need to change files to lowercase in Unix, use the script tolower.ksh. It will change ALL files in the current directory to lowercase. If the target file already exists, it will be overwritten. The script also gives read-access to all users. You must be the owner of the files to change them.

Usage: go to directory, ftp (ASCII) the file to your unix system and:

ksh tolower.ksh

3 Client Upgrade

Upgrading a client consists in installing the client applications.

Important notes to Client upgrade

- Before performing an upgrade, we strongly suggest that you make a backup of your custom unicf.dll and its source code, as upgrading will automatically replace this file with the standard unicf.dll, thus resulting in the loss of your customizations.
- Likewise, if you want to avoid losing certain customizations made in previous versions to your HKEY_CURRENT_USER and HKEY_LOCAL_MACHINE registry settings, we suggest that you export only those registry keys that you want to maintain and edit the names of any “version-sensitive” keys so that they contain the new version to which you are upgrading. Then import these keys to restore them. At this point, you can proceed to perform the upgrade. The contents of certain HKEY_LOCAL_MACHINE registry keys will be overwritten after the upgrade. Instead, editing the contents of certain HKEY_CURRENT_USER registry keys may be required in order for the system to point to the correct path.

Standard Unilab upgrade on a PC

Proceed as follows to upgrade standard Unilab on a PC:

1. Create a backup of your Unilab Client registry settings.
2. Uninstall older Unilab versions.
3. Put the Unilab CD in the CD-ROM drive.
4. Open **Windows Explorer** and select the CD-ROM drive icon.
5. Execute the client setup in accordance with the *Unilab Installation Manual*.
6. Merge the Unilab Client registry settings of which you created a backup with the new settings.

4 Upgrading the Database Custom Code

Upgrading from any 6.4 revision

Database custom code developed for any 6.4 version must be adapted in order to work for Unilab 6.7. All details in merit are provided in the Server Upgrade chapter of this manual.

5 Upgrading the Initial Database Parameters

Database startup parameters are regularly reviewed and tuned. These modifications are taken into account in our installation procedure. When upgrading, these parameters must be checked and if necessary set manually.

List of SUPPRESSED parameters when working with Oracle 11:

Parameter	**Why?**
background_dump_dest, core_dump_dest, user_dump_dest	Obsolete in Oracle11. replaced by diagnostic_dest
#log_archive_start = TRUE	Obsolete parameter in oracle11
sort_area_size, sort_area_retained_size, pga_aggregate_size, shared_pool_size, java_pool_size, db_cache_size	Deprecated in Oracle11.. Replaced by memory_target

List of MODIFIED/ADDED parameters when working with Oracle 11:

Parameter	**Value**	**Why?**
optimizer_mode	ALL_ROWS	Instead of RULE no more supported by Oracle
optimizer_index_cost_adj	1	Necessary to improve performance of the Unilab database. The cost-based optimizer will use our indexes in a better way (similar to the old rule-based optimizer).
diagnostic_dest	<<user specified directory>>	Oracle will create the necessary new subdirectories upon install. Refer to Oracle documentation for more details.
memory_target	<<calculated>>	More efficient memory management. Oracle strongly recommends to migrate to that memory management. See AMM (Automatic Memory Management in Oracle documentation for details.

shared_servers=0
CLUSTER_DATABASE='FALSE'
		Added to be sure that we are running in the

INSTANCE_TYPE='RDBMS'		right mode after a default install
SEC_CASE_SENSITIVE_LOGON	'TRUE'	**User passwords are case sensitive** by default in Oracle 11. Set eventually to False if you have backward compatibility problems.

Also modified in Oracle11, all parameters that are alphanumeric values must be written between single-quotes.

List of SUPPRESSED parameters when working with Oracle 12:

Parameter	**Why?**
SEC_CASE_SENSITIVE_LOGON	Deprecated in Oracle 12c – It is retained for backward compatibility only. Password is case-sensitive

Note:

1. Starting with Oracle Database 12c, Oracle Enterprise Manager Database Control is desupported and is no longer available. Oracle introduces Oracle Enterprise Manager Database Express (Oracle EM Express) as a replacement.
2. Oracle Restart is deprecated in Oracle Database 12c. Oracle Restart is currently restricted to manage single-instance Oracle databases and Oracle ASM instances only, and is subject to desupporting in future releases. Oracle continues to provide Oracle ASM as part of the Oracle Grid Infrastructure installation for Standalone and Cluster deployments.

List of MODIFIED/ADDED parameters when working with Oracle 11:

Parameter	**Value**	**Why?**
db_domain	‘’	There is no default value. You must set this parameter for every instance, and multiple instances must have the same value. This value cannot be `NULL`.
memory_target	<<calculated>>	`MEMORY_TARGET` specifies the Oracle systemwide usable memory. It should be set higher than or equal to the sum of the current sizes of the SGA and PGA. It doesn’t include memory consumed by server and user processes.

6 Web Server Upgrade

Some of the Unilab functionalities are also accessible from within a web browser. You will need to install the Unilab web site if you plan to install and use your own Unilab web site on your own web server. For details on how to install the web server, please consult the *Unilab Web Installation Guide* in the Unilab Product Library.

For what concerns upgrading the web server, keep in mind that the only files that may be customized (i.e. updated) are located in folders with a name beginning with "_my" (e.g. _my_data).

The upgrade procedure illustrated below refers to upgrading from V6.4 to V6.7.

Important Prior to performing an upgrade of the web server, remember to create a backup copy of the old structure.

The upgrade procedure should be performed as follows:

1. Copy all the folders that do not have prefix "_my" and the files contained in the root folder over the old folder structure.
2. Perform a merge between the old "_my" folders and the new "_my" folders, depending on what customizations have been made on the old version. This should be done on a project-basis.

7 License Upgrade

When a Unilab 6.7 license is installed, no upgrade of the license is required.

When a license for a version previous to 6.7 is installed, in this case, the license should be changed.

Performing a license upgrade is a rather simple task.

1. Transfer the old licenses from the database to a local hard disk with the ALM client application.
2. Insert the floppy in the drive, open the ALM, select the OLD license (V6.x, where x <> 7) that you want to upgrade and click on License Key > Upgrade. The ALM automatically checks if the drive contains an upgrade license that is suitable for upgrading the selected old license and transforms it into the new one. In effect, you must perform the same steps when you buy a powerpack, e.g. from basic to standard (in this case, remaining with the same version).
3. Transfer the upgraded licenses from the local hard disk to the database with the ALM client application.

Note

The Unilab database jobs should be stopped and then started again after the upgrade of server licenses. If you do not perform this operation, the behaviour of these jobs will be unpredictable. If you are performing a database upgrade and cannot restart the jobs after the stop, this is not a problem, as these jobs will be restarted at the end of the database upgrade procedure

Our ALM software experienced problems over the last months with some new elements of Microsoft Windows OS. We recommend installing the latest version of the ALM software on the PCs where the ALM software was installed. Also install the corresponding libraries (containing the plug-in for ALM) on the same PCs. These libraries are included in the client setup of Unilab.

---

8 Appendix A: How to Start/Stop the dedicated event manager

The way the dedicated eventmanager is started has been modified.

The custom package Cxapp.StartAllDbJobs / Cxapp.StopAllDbJobs, needed in case the dedicated eventmanager is used, has been modified. The new template of Cxapp package is provided in this package.

Start standard EventManagers

```
BEGIN
      cxapp.startalldbjobs;
END;
/
```

Stop standard EventManagers

```
BEGIN
      cxapp.stopalldbjobs;
END;
/
```

Start dedicated EventManager (name = DEDICEVMGR)

```
DECLARE
      l_ret INTEGER;
BEGIN
      l_ret := UNAPIEV.StartEventmgr('DEDICEVMGR', 2[1], 200[2]);
      IF l_ret <> 0 THEN
         DBMS_OUTPUT.PUT_LINE('StartDedMgrs returned '||l_ret);
      ELSE
         DBMS_OUTPUT.PUT_LINE('StartDedMgrs OK ');
      END IF;
END;
/
```

Stop dedicated EventManager (name = DEDICEVMGR)

```
DECLARE
      l_ret INTEGER;
BEGIN
      l_ret := UNAPIEV.StopEventmgr('DEDICEVMGR');
      IF l_ret <> 0 THEN
         DBMS_OUTPUT.PUT_LINE('StopDedMgrs returned '||l_ret);
      ELSE
         DBMS_OUTPUT.PUT_LINE('StopDedMgrs OK ');
      END IF;
END;
/
```

[1] Two dedicated eventmanagers will be started.

[2] 200 becomes the start sequence of the dedicated eventmanager. If different dedicated eventmanagers are used, ensure this argument remains unique.

Check for unique IDs[3]

```
SELECT TO_NUMBER(SUBSTR(module,9)) seq_nr, inst_id, module
  FROM sys.gv_$session
 WHERE module LIKE 'EvMgrJob%'
   AND module NOT LIKE 'EvMgrJob9__'
 ORDER BY seq_nr ;
```

[3] Ensure seq_nr is unique for each instance (inst_id)