ORIGINATOR: DG TAXUD	ISSUE DATE: 27/05/2011	VERSION: 7.20-EN

SUBJECT:

TARIC3

Test Plan Introduction

DOCUMENT HISTORY

EDI.	REV.	DATE	DESCRIPTION	ACTION	PAGES
0	01	08/01/2009	Initial version	Insert	All
0	10	19/02/2009	Implementation of internal review comments. Sent for Review.	Update	As requested
1	00	27/03/2009	Implementation of review comments. Sent for Acceptance.	Update	As required
1	01	06/05/2009	- Added Output Bridge section; - Minor corrections.	Update	Section 4 and 7
1	10	04/06/2009	Added section 8. Sent for review	Update	Section 8
2	00	15/07/2009	Implementation of review comments. Sent for Acceptance.	Update	As required
2	01	12/08/2009	Corrected section 4.1.	Update	Section 4.1
2	02	14/10/2009	Added section on Reports and Mass Updates.	Update	Section 9 and 10
2	03	05/11/2009	Updated section on Output Bridge.	Update, Insert	Section 4.4 and 4.5
2	04	12/11/2009	Updated section on Output Bridge test setup.	Update	Section 7.2.1

EDI.	REV.	DATE	DESCRIPTION	ACTION	PAGES
2	05	14/12/2009	Added section on Publication test setup.	Update	Section 4.6 and 11
2	06	19/01/2010	Added section on Input Bridge test setup.	Update	Section 4.7 and 12
2	07	26/01/2010	Added user for Input Bridge STP	Update	Section 4.7
2	08	30/03/2010	Updated section on Output Bridge test setup.	Update	Section 7.2.1.2
2	10	16/04/2010	Sent for review	-	-
3	00	18/05/2010	Sent for Acceptance	Update	As requested
3	10	08/06/2010	Sent for Review	-	-
4	00	23/06/2010	Sent for Acceptance	-	-
4	01	28/06/2010	Updated for QTM135.	Insert, Update	Section 6.2.1, 6.2.5, 7.2.1.4 and 7.2.5
4	10	29/06/2010	Implementation of internal review comments. Sent for Review	Update	As required
5	00	22/07/2010	Sent for Acceptance	-	-
5	10	23/07/2010	Removed section on QUOTA2 for CDCO test plan Sent for Review	Update, Delete	Section 4.1 and 6.2
6	00	10/08/2010	Implementation of review comments. Sent for Acceptance	Update	As required
6	01	26/10/2010	Updated for QTM 156 Internal Review	Update	Section 5.1
6	10	27/10/2010	Sent for Review	-	-
7	00	15/11/2010	Sent for Acceptance	Update	As requested
7	01	25/11/2010	Add of the “Test mode” section in the User Interface chapter.	Insert	5.2.1.1

EDI.	REV.	DATE	DESCRIPTION	ACTION	PAGES
7	02	29/03/2011	Updated Section 7.2.1.4 and 7.2.5 for defect 8380	Update	Section 7.2.1.4, 7.2.5
7	03	07/04/2011	Updated Section 9.2.1.1 for defect 8386	Update	Section 9.2.1.1
7	10	18/05/2011	Submitted for Review to Taxation and Customs Union DG.	Update	Section 9.2.1.3, 11.2.1.1 and 11.2.1.3
7	20	27/05/2011	Implemented comments from DG TAXUD and QAC. Submitted for Acceptance to Taxation and Customs Union DG.	Update	As Required

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1. INTRODUCTION

1.1. Objective of this document

This document explains how to execute the TARIC3 CDCO test plans.

1.2. Structure of this document

The document contains five chapters after chapter 1:

- Chapter 2 explains the content of the test deliverables;
- Chapter 3 introduces the TARIC3 CDCO system architecture to put TARIC3 CDCO system tests into context;
- Chapter 4 describes the security requirements;
- Chapter 5 introduces the user interface test plan;
- Chapter 6 introduces the system test plan.

1.3. Intended audience

This document is intended for people responsible for testing the TARIC3 CDCO system.

1.4. Abbreviations and acronyms

ANT	Another Neat Tool
CDCO	Centrally Developed Centrally Operated
HTTP	HyperText Transfer Protocol
ISO	International Organization for Standardization
JDBC	Java DataBase Connectivity
MS	Member State
SQL	Structured Query Language
STP	System Test Plan
T3	WebLogic T3 protocol for Java-to-Java connections
TAR	Tape ARchive
TARIC	TARiff Intégré Communautaire
TNS	Transparent Network Substrate
UM	User Management
URL	Uniform Resource Locator

XML	eXtensible Markup Language

Table 1: Abbreviations and acronyms

1.5. Reference documents

UM-INS-001	"UM Installation and Administration Guide", v1.00
TARIC3-IPM	"TARIC3 Installation Procedure Manual", 4.00
UITT-IPR	"UITT-IPR – UITT Installation Procedure Manual", v3.10
UITT-UM	"UITT-UM – UITT User Manual", v1.10

Table 2: Reference documents

1.6. Applicable documents

No applicable documents.

2. CONTENT OF THE DELIVERY

The test tar file, TARIC3_N.N.N_`test_suite.tar` contains:

- the server side test plan (srv_tst.tar);
- the user interface test plan (ui_tst.tar);
- the performance test plan (prf_tst.tar);
- the test DB (testDB.tar).

N.N.N will be replaced by the relevant version number of the delivery (e.g.: 1.0.0).

2.1. Server side test plan

The *srv_tst.tar* file contains the scripts and XML files used for the system test plan.

Upon extraction, this TAR file will create the following directory tree:

Directory	Content
/expected_output	XML files that are be used as references for the result of test cases
/input	XML files that will be sent to the server
/output	will contain the XML files produced by the test execution
/tools/sql	SQL scripts
/tools/security	User management scripts

2.2. User interface test plan

The *ui_tst.tar* file contains the scripts and XML files used for the user interface test plan.

Upon extraction, this TAR file will create the following directory tree:

Directory	Content
/expected_output	XML files that are be used as references for the setup XML messages required by the test plan.
/input	XML files that can be used as reference for data sent to the server by the UI.
/output	will contain the XML files produced by the test execution
/tools/sql	SQL scripts
/tools/security	User management scripts

2.3. Performance test plan

The *prf_tst.tar* file contains the scripts and XML files used for the performance test plan.

Upon extraction, this TAR file will create the following directory tree:

Directory	Content
/expected_output	XML files that are be used as references for the setup XML messages required by the test plan.
/input	XML files that can be used as reference for data sent to the server by the UI.
/output	will contain the XML files produced by the test execution
/tools/sql	SQL scripts
/tools/security	User management scripts

2.4. Output Bridge test plan

The *ob_tst.tar* file contains the scripts and XML files used for the Output Bridge test plan.

Upon extraction, this TAR file will create the following directory tree:

Directory	Content
srv_tst/expected_output	XML files that are be used as references for the setup XML messages required by the test plan.
srv_tst/input	XML files that can be used as reference for data sent to the server by the UI.
srv_tst/output	will contain the XML files produced by the test execution
srv_tst/tools/sql	SQL scripts
srv_tst/tools/security	User management scripts
ui_tst/expected_output	XML files that are be used as references for the setup XML messages required by the test plan.
ui_tst/input	XML files that can be used as reference for data sent to the server by the UI.
ui_tst/output	will contain the XML files produced by the test execution
ui_tst/tools/sql	SQL scripts
ui_tst/tools/security	User management scripts

2.5. Test database

The test database is required for the execution of the different test plans.

3. INTRODUCTION TO THE TARIC3 CDCO ARCHITECTURE

The TARIC3 CDCO system is an n-tier application that may be used synchronously or asynchronously. The user of the synchronous interface interacts with the user interface to access the client user interface using the HTTP protocol. The user of the asynchronous interface exchanges messages with the asynchronous layer via an input queue and an output queue. Both the client interface and the asynchronous layer act as client to the business logic of the TARIC3 CDCO system via exchanges of XML messages.

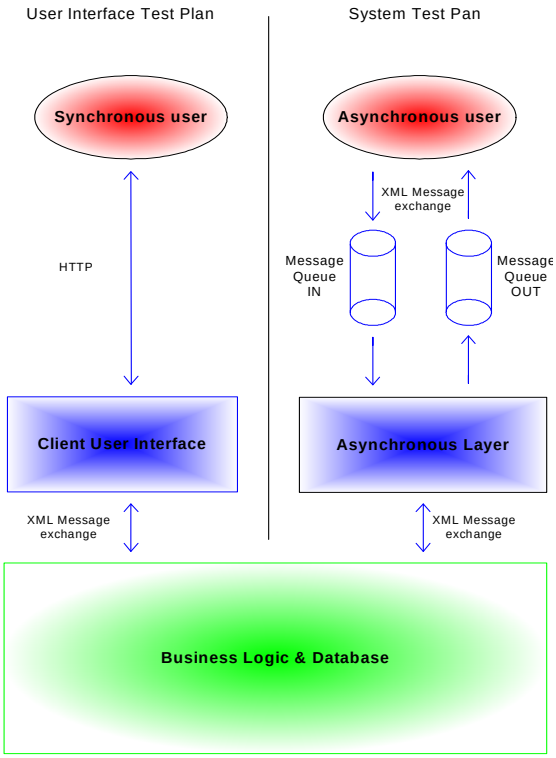

Figure 1: TARIC3 CDCO Architecture

The "TARIC3-STP-System Test Plan" aims to test the right side of Figure 1 (asynchronous). The TARIC3-TST User Interface Test Plan aims to test the left side of Figure 1 (synchronous). Later on, a specific test plan will be provided for performance testing.

4. SECURITY REQUIREMENTS

4.1. System test plan

The test cases of the system test plan assume the existence of the following users and group assignments.

User	Password	Groups
[T3-U1]	[T3-U1]	TAXUD, taric3-management, taric3-consultation,
[T3-U2]	[T3-U2]	TAXUD, taric3-consultation, taric3-batch
[T3-U3]	[T3-U3]	TAXUD

Table 3: System test plan security requirements

4.2. User interface test plan

The test cases of the user interface test plan assume the existence of the following users and group assignments.

User	Password	Groups
[T3-U1]	[T3-U1]	TAXUD, taric3-management, taric3-consultation, taric3-batch
[T3-U2]	[T3-U2]	TAXUD, taric3-consultation

Table 4: User interface test plan security requirements

4.3. Performance test plan

The test cases of the performance test plan assume the existence of the following users and group assignments.

User	Password	Groups
T3-U1	T3-U1	TAXUD, taric3-management, taric3-consultation, taric3-batch
T3-U2	T3-U2	TAXUD, taric3-management, taric3-consultation, taric3-batch
T3-U3	T3-U3	TAXUD, taric3-management, taric3-consultation, taric3-batch
T3-U4	T3-U4	TAXUD, taric3-management, taric3-consultation, taric3-batch
T3-U5	T3-U5	TAXUD, taric3-management, taric3-consultation, taric3-batch

Table 5: Performance test plan security requirements

4.4. Output Bridge test plan

The test cases of the Output Bridge test plan assume the existence of the following users and group assignments.

User	Password	Groups
[T3-U1]	[T3-U1]	TAXUD, taric3-management, taric3-consultation, taric3-batch
[T3-U2]	[T3-U2]	TAXUD, taric3-consultation
Q2-U1	Q2-U1	TAXUD, quota2-on-management, quota2-on-consultation, quota2-quota-management, quota2-quota-consultation, quota2-drawing-management, quota2-drawing-consultation, quota2-return-management, quota2-return-consultation, quota2-event-consultation, quota2-alloc-management, quota2-alloc-consultation, quota2-msafile-management, quota2-msafile-consultation, quota2-ref-management, quota2-ref-consultation, quota2-batch

Table 6: Output Bridge test plan security requirements

4.5. Reporting test plan

The test cases of the Reporting test plan assume the existence of the following user and group assignments.

User	Password	Groups
[T3-U1]	[T3-U1]	TAXUD, taric3-management, taric3-consultation, taric3-batch

Table 7: Reporting test plan security requirements

4.6. Publication test plan

The test cases of the Publication test plan assume the existence of the following user and group assignments.

User	Password	Groups
[T3-U1]	[T3-U1]	TAXUD, taric3-management, taric3-consultation, taric3-batch

Table 8: Publication test plan security requirements

4.7. Input Bridge test plan

The test cases of the Input Bridge test plan assume the existence of the following user and group assignments.

User	Password	Groups
[T3-U1]	[T3-U1]	TAXUD, taric3-management, taric3-consultation, taric3-batch
[T3-U3]	[T3-U3]	TAXUD, taric3-management, taric3-consultation, taric3-batch

Table 9: Input Bridge test plan security requirements

5. INTRODUCTION TO THE USER INTERFACE TEST PLAN

5.1. Approach

The User Interface Test Plan checks the correct creation of XML messages that are sent to the server. Different test scenarios will be executed.

It also tests the navigation paths in the application. A state-transition matrix is provided to specify the expected behaviour of the interface. Finally, it checks the availability of functions depending on the users' profile.

The User Interface Test Plan will be executed using the user interface. Most UI test cases are automated except for the reporting UI. The CDCO's and satellite systems' respective UITPs describe which test cases are automated.

5.2. Testing method

5.2.1. Test setup

The following operations must be performed before being able to execute the User Interface Test Plan.

5.2.1.1. Test mode

In the JNLP file of the TARIC3 Swing GUI application, located in .../s.taric3.ui/jws/launch.jnlp, configure the following property to false.

FORBID_DATA_MODIFICATION_FROM_DATA_INQUIRY

5.2.1.2. Dump of the XML messages

In the user interface, data are introduced in the screen for testing and submission.

The server must be configured to dump the XML files it receives from its client. Set the *DumpAppMessageOnReception* environment entry of the *TARIC3-Configuration* EJB[1] to *true*.

5.2.1.3. Database instance

The nominal database setup for the production environment is to have all TARIC3 database objects hosted inside the already existing TARIC3 schema.

As executing the user interface test plan requires a specific TARIC3 data set, a dedicated TARIC3 test schema is required, which is created and populated as described below:

- A dedicated TARIC3 test schema must be created **in an Oracle 10.2 database** as specified in the section 3.3 of the [TARIC3-IPM].

[1] This bean is in the g_demco_ejb jar component of the TARIC3 application ear file.

- The TARIC3 database objects must be created as specified in section 4.7 of the [TARIC3-IPM].
- The specific TARIC3 data set must be imported into the TARIC3 test schema as described below:

```
imp <user>/<psw>@<sid> file=<testDbDir>/taric3_testdb.dmp fromuser=taric3 touser=<user>
```

<user>, <psw> and <sid> are the credentials of the dedicated TARIC3 test schema. <TestDbDir> is the directory of the test suite which contains the Oracle 10.2 TARIC3 import file.

- All TARIC3 WebLogic datasource connection pools must be configured to point to this dedicated TARIC3 test schema.

After having applied all those changes, redeploy the application TARIC3.

5.2.2. Test initialisation

The correct execution of tests depends on the initial state of the database. An SQL script is provided to revert to the initial state of the test DB. del_all.sql (present in the directory /tools/sql) deletes all TARIC3 data and all reference data, created by the test plan, from the system.

The command to execute the SQL script:

```
sqlplus username/password@oracle_name @del_all.sql
```

5.2.3. Customisation

The tester should first customise the `ui_tst.properties` script and provide the correct values for the following variables:

connection.user: The oracle user name to connect to the TARIC3 schema;

connection.password: The associated password;

connection.url: The JDBC connection URL pointing to the above schema. (i.e. *[Removed]*);

oracle.name: The TNS alias to connect to the TARIC3 instance on the database server[2];

application.url: The T3 URL of the TARIC3 application (i.e. *[Removed]*);

um.url: The T3 URL of the UM application (i.e. *[Removed]*);

TARIC3_RUNTIME: The full path of the directory that contains the TARIC3 runtime hierarchy (i.e.

[2] The tester should be able to connect to the TARIC3 schema with *[Removed]*. The cleandb script may be run as a test for those parameters. This script is called repeatedly during the execution of the different parts of the test plan. The script should complete successfully.

/export/home/wlsapp/astaxud/groups/group10/app/taric3);

UM_RUNTIME: The full path of the directory that contains the User Management runtime hierarchy (i.e. */export/home/wlsapp/astaxud/groups/group10/app/um*);

load.security: This property specifies if the test plan users must be loaded before the test plan execution. By default, the value is *true*.

5.2.4. Creation of the test plan users

The test plan users and their group membership can be loaded using the UM application.

The User Management application must be properly configured, deployed and running before executing the following step. Ensure that the following ISO country codes are defined in the ejb-jar.xml deployment descriptor of the g_demco_management module (entry validIsoCodes) of the User Management application:

EU, BE, FR

By Default, the security environment is created but, the tester can also re-load the test plan users by executing the following command:

```
$ ./start.sh load.security
```

If the test users do not exist when running the test plan for the first time, the tester will get a warning message that he can ignore. This warning is produced because the program tries to delete the (un-existing) users before re-creating them.

To remove all test plan users after the execution of the test plan, the tester may execute the following command:

```
$ ./start.sh unload.security
```

Note: The test plan assumes the existence of the WebLogic user "*[WL-UM1]*" member of the groups TAXUD, security-manager, and security-all (see [UM-INS-001]. It is assumed that this user has as password "*[WL-UM1]*"; if it is not the case, the tester may change the password of that user in the file ./TARIC3_UI_TEST.xml (ANT target name "load.security").

5.2.5. Standard reference data loading

The reference data are automatically loaded during the execution of the test plan.

Nevertheless, if the tester must load the reference data manually, he can send the appropriate input message (eg : ./input/*init_ref_setup_01.xml*) to the system. This can be done using the sendmessage tool.

The following command can be used to empty the database before loading the reference data.

```
$ ./start.sh clean.db
```

The following command can be used to insert standard reference data in the system in order for the tests to execute correctly. If the database was just cleaned, no errors should be reported in the console.

```
$ ./start.sh init.db
```

The tester can execute all tasks (clean database, load security, reference data load and report data load) by executing:

```
$ ./start.sh
```

5.3. Result analysis

The tester compares visually the result displayed on the screen with the expected result (including error message). The structure of the XML messages must reflect the data entered in the interface by the tester.

If bugs or deviations are found, they must be tested under the other supported browsers.

6. INTRODUCTION TO THE SYSTEM TEST PLAN

6.1. Approach

The system test plan tests the business logic of the TARIC3 CDCO system using the system-to-system interface. The tester sends messages on an input queue, fetches the server responses from an output queue and compares the responses with expected response files.

6.2. Testing method

6.2.1. Test setup

The following operations must be performed before being able to execute the System Test Plan.

6.2.1.1. Dump of the XML messages

In the user interface, data are introduced in the screen for testing and submission.

The server must be configured to dump the XML files it receives from its client. Set the *DumpAppMessageOnReception* environment entry of the *TARIC3-Configuration* EJB[3] to *true*.

6.2.1.2. Database instance

The nominal database setup for the production environment is to have all TARIC3 database objects hosted inside the already existing TARIC3 schema.

As executing the system test plan requires a specific TARIC3 data set, a dedicated TARIC3 test schema is required. This schema is created and populated as described below:

- A dedicated TARIC3 test schema must be created **in an Oracle 10.2 database** as specified in the section 3.3 of the [TARIC3-IPM].
- The TARIC3 database objects must be created as specified in the section 4.7 of the [TARIC3-IPM].
- The specific TARIC3 data set must be imported into the TARIC3 test schema as described below:

```
imp <user>/<psw>@<sid> file=<testDbDir>/taric3_testdb.dmp fromuser=taric3 touser=<user>
```

<user>, <psw> and <sid> are the credentials of the dedicated TARIC3 test schema. <TestDbDir> is the directory of the test suite which contains the Oracle 10.2 TARIC3 import file.

- All TARIC3 WebLogic datasource connection pools must be configured to point to this dedicated TARIC3 test schema.

[3] This bean is in the g_demco_ejb jar component of the TARIC3 application ear file.

Please note that there is no need to import again the specific TARIC3 data set in case the System Test Plan or the User Interface Test Plan must be re-executed.

After having applied all those changes, redeploy the application TARIC3.

6.2.2. Test initialisation

The correct execution of tests depends on the initial state of the database. An SQL script is provided to revert to the initial state of the test DB. del_all.sql (present in the directory /tools/sql) deletes all TARIC3 data and all reference data, created by the test plan, from the system.

The command to execute the SQL script is:

```
sqlplus username/password@oracle_name @del_all.sql
```

6.2.3. Standard reference data loading

The reference data are automatically loaded during the execution of the test plan.

Nevertheless, if the tester must load the reference data manually, he can send the appropriate input message (eg: ./input/*init_ref_setup_01.xml*) to the system. This can be done using the sendmessage tool.

6.2.4. Test files

All XML messages and batch files to execute the test plan are provided in the archive *srv_tst.tar* included in the delivery. It can be extracted in any directory on the computer hosting the TARIC3 application hierarchy. Make sure the user running the test plan has sufficient access rights to those files (read and execute permission on all files).

6.2.5. Customisation

The operator must first customise the `srv_tst.properties` configuration file and provide the correct values for the following variables:

BEA_HOME: The directory in which the WebLogic software is installed (e.g. /opt/bea10);

JAVA_HOME: The absolute path of the JDK hierarchy;

WL_HOME: The absolute path of the directory in which the BEA WebLogic Server is installed (e.g. /opt/bea10/weblogic10.3/server);

connection.user: The Oracle user name to connect to the TARIC3 schema;

connection.password: The associated password;

connection.url: The JDBC connection URL pointing to the above schema. (i.e *[Removed]*);

oracle.name: The TNS alias to connect to the TARIC3 instance on the database server[4];

application.url: The T3 URL of the TARIC3 application (i.e. *[Removed]*);

um.url: The T3 URL of the UM application (i.e. *[Removed]*);

TARIC3_RUNTIME: The full path of the directory that contains the TARIC3 runtime hierarchy (i.e. */home/wlsapp/astaxud/groups/group10/app/taric3*);

UM_RUNTIME: The full path of the directory that contains the UM runtime hierarchy (i.e. */home/wlsapp/astaxud/groups/group10/app/um*);

load.security: This property specifies if the test plan users must be loaded before the test plan execution. By default, the value is *true*.

6.2.6. Creation of the test plan users

The test plan users and their group membership can be loaded using the UM application.

The UM application must be properly configured, deployed and running before executing the following operations. Ensure that the following ISO country codes are defined in the *ejb-jar.xml* deployment descriptor of the *g_demco_management* module (entry *validIsoCodes*) of the UM application: "*EU, AT, BE, BG, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, NL, PT, RO, SE, SI, CZ, HU, SK, PL, CY, LT, LV, MT, EE*".

By default, the security environment is created during the execution of the test plan; but, the operator can also re-load the test plan users by executing the following command:

```
$ ./start.sh load.security
```

If the test users do not exist when running the first time the test plan, the tester will get a warning message that he can ignore. This warning is produced because the program tries to delete the (non-existing) users before re-creating them.

[4] The operator must be able to connect to the database schema of the TARIC3 CDCO system with *[Removed]*. The *cleandb* script may be run as a test for those parameters. This script is called repeatedly during the execution of the different parts of the test plan. The script should complete successfully.

To remove all test plan users after the execution of the test plan, the operator may execute the following command:

```
$ ./start.sh unload.security
```

Note: The test plan assumes the existence of the WebLogic user "*[WL-UM1]*" member of the groups TAXUD, security-manager, and security-all (see [UM-INS-001]. It is assumed that this user has as password "*[WL-UM1]*"; if it is not the case the tester may change the password of that user in the file *./TARIC3_SRV_TST.xml* (ANT target name *load.security*).

6.2.7. General procedure

Before running any part of the test plan, make sure the application server in which the TARIC3 CDCO system is deployed is running.

The execution of the test plan is started from the shell script *start.sh* that puts the proper messages on the input queue and waits for the associated response messages, renames them and compares them to a prototype expected result messages.

The *start.sh* script can be started and the output captured in a log file using, for instance, the command:

```
$ ./start.sh 2>&1 | tee LOG
```

The difference between the received XML message and the expected result will be shown on the standard output and saved in the *LOG* file.

6.3. Result analysis

The comparison status of each message is displayed:

- Failure: If some differences are found.
- OK: If the output and expected output messages are identical.

At the end, the script displays the number of failures, in the following manner:

```
Failures: number of failures / number of messages sent
```

7. INTRODUCTION TO OUTPUT BRIDGE TEST PLAN

7.1. Approach

The output bridge test plan tests the business logic and user interface of the TARIC3 Output Bridge.

The system test plan tests the business logic of the TARIC3 Output Bridge system using the system-to-system interface. The tester sends messages on an input queue, fetches the server responses from an output queue and compares the responses with expected response files.

The User Interface Test Plan checks the correct creation of XML messages that are sent to the server. Different test scenarios will be executed. It also tests the navigation paths in the application. A state-transition matrix is provided to specify the expected behaviour of the interface. Finally, it checks the availability of functions depending on the users' profile.

The User Interface Test Plan will be executed using the user interface.

7.2. Testing method

7.2.1. Test setup

The following operations must be performed before being able to execute the Output Bridge System Test Plan.

7.2.1.1. Dump of the XML messages

In the user interface, data are introduced in the screen for testing and submission.

The server must be configured to dump the XML files it receives from its client. Set the *DumpAppMessageOnReception* environment entry of the *TARIC3-Configuration* EJB[5] to *true*.

7.2.1.2. Extract batch jobs

The following values must be modified in the TARIC3_job_agent_config.xml[6] file.

Full extraction:

- the 'OUB.default.full.extraction.directory' parameter: set it to the directory where the extract files must be generated;
- the 'debug.mode' parameter: set it to 'true';
- the 'OUB.AuditTimestamp.Cutoff.Margin' must be set to '2'.

[5] This bean is in the g_demco_ejb jar component of the TARIC3 application ear file.

[6] This file is in the g_demco_resources jar component of the TARIC3 application ear file.

```
<j2ee.job.definition>
  <path.name>/TARIC3/Output Bridge/Full extract</path.name>
  <path.name.as.text>
    <key>Full extract</key>
    <text>/TARIC3/Output Bridge/Full extract</text>
  </path.name.as.text>
  <classname>s.taric.batch.oub.TaricExport</classname>
  <transaction>true</transaction>
  <load>LOW</load>
  <property>
    <name>OUB.default.full.extraction.directory</name>
    <value>/tmp/oub/</value>
  </property>
  <property>
    <name>debug.mode</name>
    <value>true</value>
  </property>
  <property>
    <name>OUB.AuditTimestamp.Cutoff.Margin</name>
    <value>2</value>
  </property>
```

Incremental extraction:

- the ‘OUB.default.incremental.extraction.directory’ parameter: set it to the directory where the extract files must be generated;
- the ‘debug.mode’ parameter: set it to ‘true’;
- the ‘OUB.Extraction.Period.Margin’: set it to ‘0’;
- the ‘jms.output.queue’ parameter: set it to ‘s/taric3/jms/batch/ResultQueue’.

```
<j2ee.job.definition>
  <path.name>/TARIC3/Output Bridge/Incremental Export
Job</path.name>
  <path.name.as.text>
    <key>Incremental Export Job</key>
    <text>/TARIC3/Output Bridge/Incremental Export Job</text>
  </path.name.as.text>

<classname>s.taric.batch.oub.incremental.IncrementalExportJob</classnam
e>
  <transaction>true</transaction>
  <load>LOW</load>
  <property>
```

```
  <name>debug.mode</name>
  <value>true</value>
</property>
<property>
  <name>jms.output.queue</name>
  <value>s/taric3/jms/batch/ResultQueue</value>
</property>
<property>
  <name>OUB.Extraction.Period.Margin</name>
  <value>0</value>
 </property>
<property>
  <name>OUB.default.incremental.extraction.directory</name>
  <value>/tmp/oub/</value>
</property>
```

7.2.1.3. Database instance

The nominal database setup for the production environment is to have all TARIC3 database objects hosted inside the already existing TARIC3 schema.

As executing the system test plan requires a specific TARIC3 data set, a dedicated TARIC3 test schema is required. This schema is created and populated as described below:

- A dedicated TARIC3 test schema must be created **in an Oracle 10.2 database** as specified in the section 3.3 of the [TARIC3-IPM].
- The TARIC3 database objects must be created as specified in the section 4.7 of the [TARIC3-IPM].
- The specific TARIC3 data set must be imported into the TARIC3 test schema as described below:

```
imp <user>/<psw>@<sid> file=<testDbDir>/taric3_testdb.dmp fromuser=taric3 touser=<user>
```

  <user>, <psw> and <sid> are the credentials of the dedicated TARIC3 test schema. <TestDbDir> is the directory of the test suite which contains the Oracle 10.2 TARIC3 import file.
- All TARIC3 WebLogic datasource connection pools must be configured to point to this dedicated TARIC3 test schema.

Please note that there is no need to import again the specific TARIC3 data set in case the System Test Plan or the User Interface Test Plan must be re-executed.

After having applied all those changes, redeploy the application TARIC3.

7.2.1.4. QUOTA2 installation

In order to execute the test plan, a deployment of QUOTA2 (refer to Release notes for the latest version of QUOTA2) must be installed. This QUOTA2 deployment must point to the same DB schema as the TARIC3 deployment.

7.2.2. Test initialisation

The correct execution of tests depends on the initial state of the database. An SQL script is provided to revert to the initial state of the test DB. del_all.sql (present in the directory /tools/sql) deletes all TARIC3 data and all reference data, created by the test plan, from the system.

The command to execute the SQL script is:

```
sqlplus username/password@oracle_name @del_all.sql
```

7.2.3. Standard reference data loading

The reference data are automatically loaded during the execution of the test plan.

Nevertheless, if the tester must load the reference data manually, he can send the appropriate input message (eg: ./input/*init_ref_setup_01.xml*) to the system. This can be done using the sendmessage tool.

7.2.4. Test files

All XML messages and batch files to execute the test plan are provided in the archive *srv_tst.tar* included in the delivery. It can be extracted in any directory on the computer hosting the TARIC3 application hierarchy. Make sure the user running the test plan has sufficient access rights to those files (read and execute permission on all files).

7.2.5. Customisation

The operator must first customise the `srv_tst.properties` configuration file and provide the correct values for the following variables:

BEA_HOME: The directory in which the WebLogic software is installed (e.g. /opt/bea10);

JAVA_HOME: The absolute path of the JDK hierarchy;

WL_HOME: The absolute path of the directory in which the BEA WebLogic Server is installed (e.g. /opt/bea10/weblogic10.3/server);

connection.user: The Oracle user name to connect to the TARIC3 schema;

connection.password: The associated password;

connection.url: The JDBC connection URL pointing to the above schema. (i.e. *[Removed]*);

oracle.name: The TNS alias to connect to the TARIC3 instance on the database server[7];

application.url: The T3 URL of the TARIC3 application (i.e. *[Removed]*);

quota2.application.url The T3 URL of the QUOTA2 (refer to Release notes for the latest version of QUOTA2) application (i.e. *[Removed]*);

um.url: The T3 URL of the UM application (i.e. *[Removed]*);

TARIC3_RUNTIME: The full path of the directory that contains the TARIC3 runtime hierarchy (i.e. */home/wlsapp/astaxud/groups/group10/app/taric3*);

UM_RUNTIME: The full path of the directory that contains the UM runtime hierarchy (i.e. */home/wlsapp/astaxud/groups/group10/app/um*);

load.security: This property specifies if the test plan users must be loaded before the test plan execution. By default, the value is *true*.

7.2.6. Creation of the test plan users

The test plan users and their group membership can be loaded using the UM application.

The UM application must be properly configured, deployed and running before executing the following operations. Ensure that the following ISO country codes are defined in the *ejb-jar.xml* deployment descriptor of the *g_demco_management* module (entry *validIsoCodes*) of the UM application: "*EU, AT, BE, BG, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, NL, PT, RO, SE, SI, CZ, HU, SK, PL, CY, LT, LV, MT, EE*".

By default, the security environment is created during the execution of the test plan; but, the operator can also re-load the test plan users by executing the following command:

```
$ ./start.sh load.security
```

If the test users do not exist when running the first time the test plan, the tester will get a warning message that he can ignore. This warning is produced because the program tries to delete the (non-existing) users before re-creating them.

[7] The operator must be able to connect to the database schema of the TARIC3 CDCO system with *[Removed]*. The *cleandb* script may be run as a test for those parameters. This script is called repeatedly during the execution of the different parts of the test plan. The script should complete successfully.

To remove all test plan users after the execution of the test plan, the operator may execute the following command:

```
$ ./start.sh unload.security
```

Note: The test plan assumes the existence of the WebLogic user "*[WL-UM1]*" member of the groups TAXUD, security-manager, and security-all (see [UM-INS-001]. It is assumed that this user has as password "*[WL-UM1]*"; if it is not the case the tester may change the password of that user in the file *./TARIC3_SRV_TST.xml* (ANT target name *load.security*).

7.2.7. General procedure

Before running any part of the test plan, make sure the application server in which the TARIC3 CDCO system is deployed is running.

The execution of the test plan is started from the shell script *start.sh* that puts the proper messages on the input queue and waits for the associated response messages, renames them and compares them to a prototype expected result messages.

The *start.sh* script can be started and the output captured in a log file using, for instance, the command:

```
$ ./start.sh 2>&1 | tee LOG
```

The difference between the received XML message and the expected result will be shown on the standard output and saved in the *LOG* file.

7.3. Result analysis

Every fetched message must be compared to the message from the *expected_output* directory that is specified in the test plan document.

The comparison status of each message is displayed:

- Failure: If some differences are found.
- OK: If the output and expected output messages are identical.

At the end, the process displays the number of failures, in the following manner:

```
Failures: number of failures / number of messages sent
```

8. INTRODUCTION TO CDCO PERFORMANCE TEST PLAN

8.1. Approach

The performance test plan tests the responsiveness of the TARIC3 CDCO application by submitting data through the system-to-system and user interface.

8.2. Testing method

8.2.1. Test setup

The following operations must be performed before being able to execute the CDCO Performance Test Plan.

8.2.1.1. Dump of the XML messages

In the user interface, data are introduced in the screen for testing and submission.

The server must be configured to dump the XML files it receives from its client. Set the *DumpAppMessageOnReception* environment entry of the *TARIC3-Configuration* EJB[8] to *true*.

8.2.1.2. Database instance

The nominal database setup for the production environment is to have all TARIC3 database objects hosted inside the already existing TARIC3 schema.

As executing the system test plan requires a specific TARIC3 data set, a dedicated TARIC3 test schema is required. This schema is created and populated as described below:

- A dedicated TARIC3 test schema must be created **in an Oracle 10 database** as specified in the section 3.3 of the [TARIC3-IPM].
- The TARIC3 database objects must be created as specified in the section 4.7 of the [TARIC3-IPM].
- The specific TARIC3 data set (production dump) must be imported into the TARIC3 test schema as described below:

```
imp <user>/<psw>@<sid> file=<perfDbDir>/taric3_perfdb.dmp fromuser=taric3 touser=<user>
```

  <user>, <psw> and <sid> are the credentials of the dedicated TARIC3 test schema. <perfDbDir> is the directory of the test suite which contains the Oracle 10.2 TARIC3 import file.
- All TARIC3 WebLogic datasource connection pools must be configured to point to this dedicated TARIC3 test schema.

After having applied all those changes, redeploy the application TARIC3.

[8] This bean is in the g_demco_ejb jar component of the TARIC3 application ear file.

8.2.2. Standard reference data loading

The reference data are automatically loaded during the execution of the test plan.

Nevertheless, if the tester must load the reference data manually, he can send the appropriate input message (eg: ./input/*init_ref_setup_01.xml*) to the system. This can be done using the sendmessage tool.

8.2.3. Test files

All XML messages and batch files to execute the test plan are provided in the archive *prf_tst.tar* included in the delivery. It can be extracted in any directory on the computer hosting the TARIC3 application hierarchy. Make sure the user running the test plan has sufficient access rights to those files (read and execute permission on all files).

8.2.4. Customisation

The sting of the maximum message size supported by the WebLogic T3 protocol must be increased from 10 000 000 to 2 000 000 000.

This is done by accessing the WebLogic administration console, and following the navigation path below:

Home > Summary of Servers > Taric3Server > Protocols

Where *Taric3Server* is the name of the server hosting the TARIC3 application.

Change the value of the ‘Maximum Message Size’ parameter.

9. INTRODUCTION TO REPORTING TEST PLAN

9.1. Approach

The reporting test plan tests the business logic and user interface of the TARIC3 Reports.

The system test plan tests the business logic of the TARIC3 Reports using the system-to-system interface. The tester sends messages on an input queue, fetches the server responses from an output queue and compares the responses with expected response files.

The User Interface Test Plan checks the correct creation of XML messages that are sent to the server. Different test scenarios will be executed. It also tests the navigation paths in the application. A state-transition matrix is provided to specify the expected behaviour of the interface. Finally, it checks the availability of functions depending on the users' profile.

The User Interface Test Plan will be executed using the user interface.

9.2. Testing method

9.2.1. Test setup

The following operations must be performed before being able to execute the System Test Plan.

9.2.1.1. Preliminary steps

Before unfolding the test plan, the tester must deploy the provided Report test database. All the test cases must be run on the given test database which contain TARIC2 production data migrated to TARIC3. This test database allows creating "realistic" reports.

The provided test database has an approximate size, once deployed, of 10 GB.

The following "fake" user must be created through the weblogic console: "*[FK-US]*".

The deployment descriptor g_demco_ejb does require the following environment entries to be set in order to ease up the retrival of generated reports:

- "DebugReport" needs to be set to true in order to enable the dumping of reports
- "g.demco.batch.tmpdir" needs to be set accordingly. It represents a path on a disk where the batch files will be stored. This directory must exists and have the access rights allowing to the reports to be saved in it.

The batch temporary directory must be empty. The path to this directory is defined in the s.quota2/resources/descriptors/g_demco_ejb/ejb-jar.xml file (ejb: TARIC3-Configuration, entry: /tmp/taric3/batch).

9.2.1.2. Dump of the XML messages

In the user interface, data are introduced in the screen for testing and submission.

The server must be configured to dump the XML files it receives from its client. Set the *DumpAppMessageOnReception* environment entry of the *TARIC3-Configuration* EJB[9] to *true*.

9.2.1.3. Database instance

The nominal database setup for the production environment is to have all TARIC3 database objects hosted inside the already existing TARIC3 schema.

As executing the system test plan requires a specific TARIC3 Report data set, a dedicated TARIC3 Report test schema is required. This schema is populated as described below:

- A dedicated TARIC3 Report test schema must be created **in an Oracle 10 database** as specified in the section 3.3 of the [TARIC3-IPM].
- The TARIC3 database objects must be created as specified in the section 4.7 of the [TARIC3-IPM].
- The specific TARIC3 Report data set (available in TARIC3REP.dmp.gz) must be imported into the TARIC3 test schema as described below:

```
imp <user>/<psw>@<sid> file=TARIC3REP.dmp fromuser=taric3rep touser=<user>
```

  <user>, <psw> and <sid> are the credentials of the dedicated TARIC3 Report test schema. TARIC3REP.dmp is the database dump that has relevant data required for generation of realistic reports.
- All TARIC3 WebLogic datasource connection pools must be configured to point to this dedicated TARIC3 test schema.

Please note that there is no need to import again the specific TARIC3 data set in case the System Test Plan or the User Interface Test Plan must be re-executed.

After having applied all those changes, redeploy the application TARIC3.

9.2.2. Test initialisation

The correct execution of tests depends on the initial state of the database. An SQL script is provided to revert to the initial state of the test DB. del_all.sql (present in the directory /tools/sql) deletes all TARIC3 data and all reference data, created by the test plan, from the system.

The command to execute the SQL script is:

```
sqlplus username/password@oracle_name @del_all.sql
```

[9] This bean is in the g_demco_ejb jar component of the TARIC3 application ear file.

9.2.3. Standard reference data loading

The reference data are automatically loaded during the execution of the test plan.

Nevertheless, if the tester must load the reference data manually, he can send the appropriate input message (eg: ./input/*init_ref_setup_01.xml*) to the system. This can be done using the sendmessage tool.

9.2.4. Test files

All XML messages and batch files to execute the test plan are provided in the archive *srv_tst.tar* included in the delivery. It can be extracted in any directory on the computer hosting the TARIC3 application hierarchy. Make sure the user running the test plan has sufficient access rights to those files (read and execute permission on all files).

9.2.5. Customisation

The operator must first customise the `srv_tst.properties` configuration file and provide the correct values for the following variables:

BEA_HOME: The directory in which the WebLogic software is installed (e.g. /opt/bea10);

JAVA_HOME: The absolute path of the JDK hierarchy;

WL_HOME: The absolute path of the directory in which the BEA WebLogic Server is installed (e.g. /opt/bea10/weblogic10.3/server);

connection.user: The Oracle user name to connect to the TARIC3 schema;

connection.password: The associated password;

connection.url: The JDBC connection URL pointing to the above schema. (i.e. *[Removed]*);

oracle.name: The TNS alias to connect to the TARIC3 instance on the database server[10];

application.url: The T3 URL of the TARIC3 application (i.e. *[Removed]*);

um.url: The T3 URL of the UM application (i.e. *[Removed]*);

[10] The operator must be able to connect to the database schema of the TARIC3 CDCO system with *[Removed]*. The *cleandb* script may be run as a test for those parameters. This script is called repeatedly during the execution of the different parts of the test plan. The script should complete successfully.

TARIC3_RUNTIME: The full path of the directory that contains the TARIC3 runtime hierarchy (i.e. */home/wlsapp/astaxud/groups/group10/app/taric3*);

UM_RUNTIME: The full path of the directory that contains the UM runtime hierarchy (i.e. */home/wlsapp/astaxud/groups/group10/app/um*);

load.security: This property specifies if the test plan users must be loaded before the test plan execution. By default, the value is *true*.

9.2.6. Creation of the test plan users

The test plan users and their group membership can be loaded using the UM application.

The UM application must be properly configured, deployed and running before executing the following operations. Ensure that the following ISO country codes are defined in the *ejb-jar.xml* deployment descriptor of the *g_demco_management* module (entry *validIsoCodes*) of the UM application: "*EU, AT, BE, BG, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, NL, PT, RO, SE, SI, CZ, HU, SK, PL, CY, LT, LV, MT, EE*".

By default, the security environment is created during the execution of the test plan; but, the operator can also re-load the test plan users by executing the following command:

```
$ ./start.sh load.security
```

If the test users do not exist when running the first time the test plan, the tester will get a warning message that he can ignore. This warning is produced because the program tries to delete the (non-existing) users before re-creating them.

To remove all test plan users after the execution of the test plan, the operator may execute the following command:

```
$ ./start.sh unload.security
```

Note: The test plan assumes the existence of the WebLogic user "*[WL-UM1]*" member of the groups TAXUD, security-manager, and security-all (see [UM-INS-001]. It is assumed that this user has as password "*[WL-UM1]*"; if it is not the case the tester may change the password of that user in the file *./TARIC3_SRV_TST.xml* (ANT target name *load.security*).

9.2.7. General procedure

Before running any part of the test plan, make sure the application server in which the TARIC3 CDCO system is deployed is running.

The execution of the test plan is started from the shell script *start.sh* that puts the proper messages on the input queue and waits for the associated response messages, renames them and compares them to a prototype expected result messages.

The *start.sh* script can be started and the output captured in a log file using, for instance, the command:

```
$ ./start.sh 2>&1 | tee LOG
```

The difference between the received XML message and the expected result will be shown on the standard output and saved in the *LOG* file.

9.3. Result analysis

Every fetched message must be compared to the message from the *expected_output* directory that is specified in the test plan document.

The comparison status of each message is displayed:

- Failure: If some differences are found.
- OK: If the output and expected output messages are identical.

At the end, the process displays the number of failures, in the following manner:

```
Failures: number of failures / number of messages sent
```

10. INTRODUCTION TO MASS UPDATES TEST PLAN

10.1. Approach

The Mass Updates test plan tests the business logic and user interface of the TARIC3 Mass Updates.

The system test plan tests the business logic of the TARIC3 Mass Updates using the system-to-system interface. The tester sends messages on an input queue, fetches the server responses from an output queue and compares the responses with expected response files.

The User Interface Test Plan checks the correct creation of XML messages that are sent to the server. Different test scenarios will be executed. It also tests the navigation paths in the application. A state-transition matrix is provided to specify the expected behaviour of the interface. Finally, it checks the availability of functions depending on the users' profile.

The User Interface Test Plan will be executed using the user interface.

10.2. Testing method

10.2.1. Test setup

The following operations must be performed before being able to execute the System Test Plan.

10.2.1.1. Dump of the XML messages

In the user interface, data are introduced in the screen for testing and submission.

The server must be configured to dump the XML files it receives from its client. Set the *DumpAppMessageOnReception* environment entry of the *TARIC3-Configuration* EJB[11] to *true*.

10.2.1.2. Database instance

The nominal database setup for the production environment is to have all TARIC3 database objects hosted inside the already existing TARIC3 schema.

As executing the system test plan requires a specific TARIC3 data set, a dedicated TARIC3 test schema is required. This schema is created and populated as described below:

- A dedicated TARIC3 test schema must be created **in an Oracle 10 database** as specified in the section 3.3 of the [TARIC3-IPM].
- The TARIC3 database objects must be created as specified in the section 4.7 of the [TARIC3-IPM].

[11] This bean is in the g_demco_ejb jar component of the TARIC3 application ear file.

- The specific TARIC3 data set must be imported into the TARIC3 test schema as described below:

```
imp <user>/<psw>@<sid> file=<testDbDir>/taric3_testdb.dmp fromuser=taric3 touser=<user>
```

<user>, <psw> and <sid> are the credentials of the dedicated TARIC3 test schema. <TestDbDir> is the directory of the test suite which contains the Oracle 10.2 TARIC3 import file.

- All TARIC3 WebLogic datasource connection pools must be configured to point to this dedicated TARIC3 test schema.

Please note that there is no need to import again the specific TARIC3 data set in case the System Test Plan or the User Interface Test Plan must be re-executed.

After having applied all those changes, redeploy the application TARIC3.

10.2.2. Test initialisation

The correct execution of tests depends on the initial state of the database. An SQL script is provided to revert to the initial state of the test DB. del_all.sql (present in the directory /tools/sql) deletes all TARIC3 data and all reference data, created by the test plan, from the system.

The command to execute the SQL script is:

```
sqlplus username/password@oracle_name @del_all.sql
```

10.2.3. Standard reference data loading

The reference data are automatically loaded during the execution of the test plan.

Nevertheless, if the tester must load the reference data manually, he can send the appropriate input message (eg: ./input/*init_ref_setup_01.xml*) to the system. This can be done using the sendmessage tool.

10.2.4. Test files

All XML messages and batch files to execute the test plan are provided in the archive *srv_tst.tar* included in the delivery. It can be extracted in any directory on the computer hosting the TARIC3 application hierarchy. Make sure the user running the test plan has sufficient access rights to those files (read and execute permission on all files).

10.2.5. Customisation

The operator must first customise the `srv_tst.properties` configuration file and provide the correct values for the following variables:

BEA_HOME: The directory in which the WebLogic software is installed (e.g. /opt/bea10);

JAVA_HOME: The absolute path of the JDK hierarchy;

WL_HOME: The absolute path of the directory in which the BEA WebLogic Server is installed (e.g. /opt/bea10/weblogic10.3/server);

connection.user: The Oracle user name to connect to the TARIC3 schema;

connection.password: The associated password;

connection.url: The JDBC connection URL pointing to the above schema. (i.e. *[Removed]*);

oracle.name: The TNS alias to connect to the TARIC3 instance on the database server[12];

application.url: The T3 URL of the TARIC3 application (i.e. *[Removed]*);

um.url: The T3 URL of the UM application (i.e. *[Removed]*);

TARIC3_RUNTIME: The full path of the directory that contains the TARIC3 runtime hierarchy (i.e. */home/wlsapp/astaxud/groups/group10/app/taric3*);

UM_RUNTIME: The full path of the directory that contains the UM runtime hierarchy (i.e. */home/wlsapp/astaxud/groups/group10/app/um*);

load.security: This property specifies if the test plan users must be loaded before the test plan execution. By default, the value is *true*.

10.2.6. Creation of the test plan users

The test plan users and their group membership can be loaded using the UM application.

The UM application must be properly configured, deployed and running before executing the following operations. Ensure that the following ISO country codes are defined in the *ejb-jar.xml* deployment descriptor of the *g_demco_management* module (entry *validIsoCodes*) of the UM application: "*EU, AT, BE, BG, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, NL, PT, RO, SE, SI, CZ, HU, SK, PL, CY, LT, LV, MT, EE*".

[12] The operator must be able to connect to the database schema of the TARIC3 CDCO system with *[Removed]*. The *cleandb* script may be run as a test for those parameters. This script is called repeatedly during the execution of the different parts of the test plan. The script should complete successfully.

By default, the security environment is created during the execution of the test plan; but, the operator can also re-load the test plan users by executing the following command:

```
$ ./start.sh load.security
```

If the test users do not exist when running the first time the test plan, the tester will get a warning message that he can ignore. This warning is produced because the program tries to delete the (non-existing) users before re-creating them.

To remove all test plan users after the execution of the test plan, the operator may execute the following command:

```
$ ./start.sh unload.security
```

Note: The test plan assumes the existence of the WebLogic user "*[WL-UM1]*" member of the groups TAXUD, security-manager, and security-all (see [UM-INS-001]. It is assumed that this user has as password "*[WL-UM1]*"; if it is not the case the tester may change the password of that user in the file *./TARIC3_SRV_TST.xml* (ANT target name *load.security*).

10.2.7. General procedure

Before running any part of the test plan, make sure the application server in which the TARIC3 CDCO system is deployed is running.

The execution of the test plan is started from the shell script *start.sh* that puts the proper messages on the input queue and waits for the associated response messages, renames them and compares them to a prototype expected result messages.

The *start.sh* script can be started and the output captured in a log file using, for instance, the command:

```
$ ./start.sh 2>&1 | tee LOG
```

The difference between the received XML message and the expected result will be shown on the standard output and saved in the *LOG* file.

10.3. Result analysis

Every fetched message must be compared to the message from the *expected_output* directory that is specified in the test plan document.

The comparison status of each message is displayed:

- Failure: If some differences are found.
- OK: If the output and expected output messages are identical.

At the end, the process displays the number of failures, in the following manner:

```
Failures: number of failures / number of messages sent
```

11. INTRODUCTION TO PUBLICATION TEST PLAN

11.1. Approach

The publication test plan tests the business logic and user interface of the TARIC3 Publication.

The system test plan tests the business logic of the TARIC3 Publication using the system-to-system interface. The tester sends messages on an input queue, fetches the server responses from an output queue and compares the responses with expected response files.

The User Interface Test Plan checks the correct creation of XML messages that are sent to the server. Different test scenarios will be executed. It also tests the navigation paths in the application. A state-transition matrix is provided to specify the expected behaviour of the interface. Finally, it checks the availability of functions depending on the users' profile.

The User Interface Test Plan will be executed using the user interface.

11.2. Testing method

11.2.1. Test setup

The following operations must be performed before being able to execute the System Test Plan.

11.2.1.1. Preliminary steps

Before unfolding the test plan, the tester must deploy the provided TARIC3 Publication test database. All the test cases must be run on the given Publication test database which contains TARIC2 production data migrated to TARIC3. This test database allows creating "realistic" publications.

The provided test database has an approximate size, once deployed, of 10 GB.

The following "fake" user must be created through the weblogic console: "*[FK-US]*".

The deployment descriptor g_demco_ejb does require the following environment entries to be set in order to ease up the retrival of generated reports:

- "DebugReport" needs to be set to true in order to enable the dumping of reports
- "g.demco.batch.tmpdir" needs to be set accordingly. It represents a path on a disk where the batch files will be stored. This directory must exists and have the access rights allowing to the reports to be saved in it.

The batch temporary directory must be empty. The path to this directory is defined in the s.quota2/resources/descriptors/g_demco_ejb/ejb-jar.xml file (ejb: TARIC3-Configuration, entry: /tmp/taric3/batch).

11.2.1.2. Dump of the XML messages

In the user interface, data are introduced in the screen for testing and submission.

The server must be configured to dump the XML files it receives from its client. Set the *DumpAppMessageOnReception* environment entry of the *TARIC3-Configuration* EJB[13] to *true*.

11.2.1.3. Database instance

The nominal database setup for the production environment is to have all TARIC3 database objects hosted inside the already existing TARIC3 schema.

As executing the system test plan requires a specific TARIC3 Publication data set, a dedicated TARIC3 Publication test schema is required. This schema is created and populated as described below:

- A dedicated TARIC3 Publication test schema must be created **in an Oracle 10 database** as specified in the section 3.3 of the [TARIC3-IPM].
- The TARIC3 database objects must be created as specified in the section 4.7 of the [TARIC3-IPM].
- The specific TARIC3 data set must be imported into the TARIC3 test schema as described below:

```
imp <user>/<psw>@<sid> file=TARIC3PUB.dmp fromuser=taric3pub touser=<user>
```

<user>, <psw> and <sid> are the credentials of the dedicated TARIC3 Publication test schema. TARIC3PUB.dmp is the TARIC3 Publication test data dump.

- All TARIC3 WebLogic datasource connection pools must be configured to point to this dedicated TARIC3 test schema.

Please note that there is no need to import again the specific TARIC3 data set in case the System Test Plan or the User Interface Test Plan must be re-executed.

After having applied all those changes, redeploy the application TARIC3.

11.2.1.4. Publication output files folder

The following values must be modified in the TARIC3_job_agent_config.xml[14] file.

Main duty table:

- the 'PUB.default.publication.directory' parameter: set it to the directory where the publication files must be generated.

<j2ee.job.definition>

[13] This bean is in the g_demco_ejb jar component of the TARIC3 application ear file.

[14] This file is in the g_demco_resources jar component of the TARIC3 application ear file.

```
    <path.name>/TARIC3/Publication/Publication Main Duty
Table</path.name>
    <path.name.as.text>
            <key>menu-title-publication-main-duty-table</key>
            <text>/TARIC3/Publication/MainDutyTable</text>
    </path.name.as.text>
    <classname>s.taric.batch.publication.service.MainPubSrvImpl</classnam
e>
    <transaction>false</transaction>
    <load>HIGH</load>
    <property>
            <name>PUB.default.publication.directory</name>
            <value>/tmp/taric3/publication/</value>
    </property>
    <principals>
            <name>taric3-batch</name>
    </principals>
</j2ee.job.definition>
```

Annex 1:

- the 'PUB.default.publication.directory' parameter: set it to the directory where the publication files must be generated.

```
<j2ee.job.definition>
    <path.name>/TARIC3/Publication/Publication Annex1</path.name>
            <path.name.as.text>
                    <key>menu-title-publication-annex1</key>
                    <text>/TARIC3/Publication/Annex1</text>
            </path.name.as.text>

    <classname>s.taric.batch.publication.service.Anx1PubSrvImpl</classnam
e>
    <transaction>false</transaction>
    <load>HIGH</load>
    <property>
            <name>PUB.default.publication.directory</name>
            <value>/tmp/taric3/publication/</value>
    </property>
    <principals>
            <name>taric3-batch</name>
    </principals>
</j2ee.job.definition>
```

Annex 5:

- the 'PUB.default.publication.directory' parameter: set it to the directory where the publication files must be generated.

```
<j2ee.job.definition>
   <path.name>/TARIC3/Publication/Publication Annex5</path.name>
   <path.name.as.text>
            <key>menu-title-publication-annex5</key>
            <text>/TARIC3/Publication/Annex5</text>
   </path.name.as.text>
   <classname>s.taric.batch.publication.service.Anx5PubSrvImpl</classnam
e>
   <transaction>false</transaction>
   <load>HIGH</load>
   <property>
            <name>PUB.default.publication.directory</name>
            <value>/tmp/taric3/publication/</value>
   </property>
   <principals>
            <name>taric3-batch</name>
   </principals>
</j2ee.job.definition>
```

11.2.2. Test initialisation

The correct execution of tests depends on the initial state of the database. The database can be reverted back to its initial state by importing again the aforementioned specific TARIC3 data set.

11.2.3. Standard reference data loading

The reference data are automatically loaded during the execution of the test plan.

Nevertheless, if the tester must load the reference data manually, he can send the appropriate input message (eg: ./input/*init_ref_setup_01.xml*) to the system. This can be done using the sendmessage tool.

11.2.4. Test files

All XML messages and batch files to execute the test plan are provided in the archive *srv_tst.tar* included in the delivery. It can be extracted in any directory on the computer hosting the TARIC3 application hierarchy. Make sure the user running the test plan has sufficient access rights to those files (read and execute permission on all files).

11.2.5. Customisation

The operator must first customise the `srv_tst.properties` configuration file and provide the correct values for the following variables:

BEA_HOME: The directory in which the WebLogic software is installed (e.g. /opt/bea10);

JAVA_HOME: The absolute path of the JDK hierarchy;

WL_HOME: The absolute path of the directory in which the BEA WebLogic Server is installed (e.g. /opt/bea10/weblogic10.3/server);

connection.user: The Oracle user name to connect to the TARIC3 schema;

connection.password: The associated password;

connection.url: The JDBC connection URL pointing to the above schema. (i.e. *[Removed]*);

oracle.name: The TNS alias to connect to the TARIC3 instance on the database server[15];

application.url: The T3 URL of the TARIC3 application (i.e. *[Removed]*);

um.url: The T3 URL of the UM application (i.e. *[Removed]*);

TARIC3_RUNTIME: The full path of the directory that contains the TARIC3 runtime hierarchy (i.e. */home/wlsapp/astaxud/groups/group10/app/taric3*);

UM_RUNTIME: The full path of the directory that contains the UM runtime hierarchy (i.e. */home/wlsapp/astaxud/groups/group10/app/um*);

load.security: This property specifies if the test plan users must be loaded before the test plan execution. By default, the value is *true*.

11.2.6. Creation of the test plan users

The test plan users and their group membership can be loaded using the UM application.

The UM application must be properly configured, deployed and running before executing the following operations. Ensure that the following ISO country codes are defined in the *ejb-jar.xml* deployment descriptor of the *g_demco_management* module (entry *validIsoCodes*) of the UM application: "*EU, AT, BE, BG, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, NL, PT, RO, SE, SI, CZ, HU, SK, PL, CY, LT, LV, MT, EE*".

[15] The operator must be able to connect to the database schema of the TARIC3 CDCO system with *[Removed]*. The *cleandb* script may be run as a test for those parameters. This script is called repeatedly during the execution of the different parts of the test plan. The script should complete successfully.

By default, the security environment is created during the execution of the test plan; but, the operator can also re-load the test plan users by executing the following command[16]:

```
$ ./start.sh load.security
```

If the test users do not exist when running the first time the test plan, the tester will get a warning message that he can ignore. This warning is produced because the program tries to delete the (non-existing) users before re-creating them.

To remove all test plan users after the execution of the test plan, the operator may execute the following command:

```
$ ./start.sh unload.security
```

Note: The test plan assumes the existence of the WebLogic user "*[WL-UM1]*" member of the groups TAXUD, security-manager, and security-all (see [UM-INS-001]. It is assumed that this user has as password "*[WL-UM1]*"; if it is not the case the tester may change the password of that user in the file *./TARIC3_SRV_TST.xml* (ANT target name *load.security*).

11.2.7. General procedure

Before running any part of the test plan, make sure the application server in which the TARIC3 CDCO system is deployed is running.

The execution of the test plan is started from the shell script *start.sh* that puts the proper messages on the input queue and waits for the associated response messages, renames them and compares them to a prototype expected result messages.

The *start.sh* script can be started and the output captured in a log file using, for instance, the command:

```
$ ./start.sh 2>&1 | tee LOG
```

The difference between the received XML message and the expected result will be shown on the standard output and saved in the *LOG* file.

11.3. Result analysis

Every fetched message must be compared to the message from the *expected_output* directory that is specified in the test plan document.

The comparison status of each message is displayed:

- Failure: If some differences are found.
- OK: If the output and expected output messages are identical.

[16] There is no separate start.sh for the user interface test plan. The users can be created using the start.sh of the server test plan.

At the end, the process displays the number of failures, in the following manner:

```
Failures: number of failures / number of messages sent
```

12. INTRODUCTION TO INPUT BRIDGE TEST PLAN

12.1. Approach

The input bridge test plan tests the business logic and user interface of the TARIC3 Input Bridge.

The system test plan tests the business logic of the TARIC3 Input Bridge using the system-to-system interface. The tester sends messages on an input queue, fetches the server responses from an output queue and compares the responses with expected response files.

The User Interface Test Plan checks the correct creation of XML messages that are sent to the server. Different test scenarios will be executed. It also tests the navigation paths in the application. A state-transition matrix is provided to specify the expected behaviour of the interface. Finally, it checks the availability of functions depending on the users' profile.

The User Interface Test Plan will be executed using the user interface.

12.2. Testing method

12.2.1. Test setup

The following operations must be performed before being able to execute the System Test Plan.

12.2.1.1. Dump of the XML messages

In the user interface, data are introduced in the screen for testing and submission.

The server must be configured to dump the XML files it receives from its client. Set the *DumpAppMessageOnReception* environment entry of the *TARIC3-Configuration* EJB[17] to *true*.

12.2.1.2. Database instance

The nominal database setup for the production environment is to have all TARIC3 database objects hosted inside the already existing TARIC3 schema.

As executing the system test plan requires a specific TARIC3 data set, a dedicated TARIC3 test schema is required. This schema is created and populated as described below:

- A dedicated TARIC3 test schema must be created **in an Oracle 10 database** as specified in the section 3.3 of the [TARIC3-IPM].
- The TARIC3 database objects must be created as specified in the section 4.7 of the [TARIC3-IPM].

[17] This bean is in the g_demco_ejb jar component of the TARIC3 application ear file.

- The specific TARIC3 data set must be imported into the TARIC3 test schema as described below:

```
imp <user>/<psw>@<sid> file=<testDbDir>/taric3_testdb.dmp fromuser=taric3 touser=<user>
```

<user>, <psw> and <sid> are the credentials of the dedicated TARIC3 test schema. <TestDbDir> is the directory of the test suite which contains the Oracle 10.2 TARIC3 import file.

- All TARIC3 WebLogic datasource connection pools must be configured to point to this dedicated TARIC3 test schema.

Please note that there is no need to import again the specific TARIC3 data set in case the System Test Plan or the User Interface Test Plan must be re-executed.

After having applied all those changes, redeploy the application TARIC3.

12.2.2. Test initialisation

The correct execution of tests depends on the initial state of the database. The database can be reverted back to its initial state by importing again the aforementioned specific TARIC3 data set.

12.2.3. Standard reference data loading

The reference data are automatically loaded during the execution of the test plan.

Nevertheless, if the tester must load the reference data manually, he can send the appropriate input message (eg: ./input/*init_ref_setup_01.xml*) to the system. This can be done using the sendmessage tool.

12.2.4. Test files

All XML messages and batch files to execute the test plan are provided in the archive *srv_tst.tar* included in the delivery. It can be extracted in any directory on the computer hosting the TARIC3 application hierarchy. Make sure the user running the test plan has sufficient access rights to those files (read and execute permission on all files).

12.2.5. Customisation

The operator must first customise the `srv_tst.properties` configuration file and provide the correct values for the following variables:

BEA_HOME: The directory in which the WebLogic software is installed (e.g. /opt/bea10);

JAVA_HOME: The absolute path of the JDK hierarchy;

WL_HOME: The absolute path of the directory in which the BEA WebLogic Server is installed (e.g. /opt/bea10/weblogic10.3/server);

connection.user: The Oracle user name to connect to the TARIC3 schema;

connection.password: The associated password;

connection.url: The JDBC connection URL pointing to the above schema. (i.e *[Removed]*);

oracle.name: The TNS alias to connect to the TARIC3 instance on the database server[18];

application.url: The T3 URL of the TARIC3 application (i.e. *[Removed]*);

um.url: The T3 URL of the UM application (i.e. *[Removed]*);

TARIC3_RUNTIME: The full path of the directory that contains the TARIC3 runtime hierarchy (i.e. */home/wlsapp/astaxud/groups/group10/app/taric3*);

UM_RUNTIME: The full path of the directory that contains the UM runtime hierarchy (i.e. */home/wlsapp/astaxud/groups/group10/app/um*);

load.security: This property specifies if the test plan users must be loaded before the test plan execution. By default, the value is *true*.

12.2.6. Creation of the test plan users

The test plan users and their group membership can be loaded using the UM application.

The UM application must be properly configured, deployed and running before executing the following operations. Ensure that the following ISO country codes are defined in the *ejb-jar.xml* deployment descriptor of the *g_demco_management* module (entry *validIsoCodes*) of the UM application: "*EU, AT, BE, BG, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, NL, PT, RO, SE, SI, CZ, HU, SK, PL, CY, LT, LV, MT, EE*".

By default, the security environment is created during the execution of the test plan; but, the operator can also re-load the test plan users by executing the following command[19]:

```
$ ./start.sh load.security
```

[18] The operator must be able to connect to the database schema of the TARIC3 CDCO system with *[Removed]*. The *cleandb* script may be run as a test for those parameters. This script is called repeatedly during the execution of the different parts of the test plan. The script should complete successfully.

[19] There is no separate start.sh for the user interface test plan. The users can be created using the start.sh of the server test plan.

If the test users do not exist when running the first time the test plan, the tester will get a warning message that he can ignore. This warning is produced because the program tries to delete the (non-existing) users before re-creating them.

To remove all test plan users after the execution of the test plan, the operator may execute the following command:

```
$ ./start.sh unload.security
```

Note: The test plan assumes the existence of the WebLogic user "*[WL-UM1]*" member of the groups TAXUD, security-manager, and security-all (see [UM-INS-001]. It is assumed that this user has as password "*[WL-UM1]*"; if it is not the case the tester may change the password of that user in the file *./TARIC3_SRV_TST.xml* (ANT target name *load.security*).

12.2.7. General procedure

Before running any part of the test plan, make sure the application server in which the TARIC3 CDCO system is deployed is running.

The execution of the test plan is started from the shell script *start.sh* that puts the proper messages on the input queue and waits for the associated response messages, renames them and compares them to a prototype expected result messages.

The *start.sh* script can be started and the output captured in a log file using, for instance, the command:

```
$ ./start.sh 2>&1 | tee LOG
```

The difference between the received XML message and the expected result will be shown on the standard output and saved in the *LOG* file.

12.3. Result analysis

Every fetched message must be compared to the message from the *expected_output* directory that is specified in the test plan document.

The comparison status of each message is displayed:

- Failure: If some differences are found.
- OK: If the output and expected output messages are identical.

At the end, the process displays the number of failures, in the following manner:

```
Failures: number of failures / number of messages sent
```