EUROPEAN COMMISSION

DIRECTORATE-GENERAL
INFORMATICS

Information systems Directorate

European Commission

e-PRIOR User Activation Procedure

Date:	15/06/2011
Version:	0.7
Authors:	Olivier Derveau (Trasys)
	Maarten Daniëls (PricewaterhouseCoopers)
Revised by:	Didier Thunus (European Commission, DIGIT B4)
	Denis Janssen (Trasys)
Approved by:	---
Public:	
Reference Number:	

Document History

Version	Date	Comment	Modified Pages
0.007	15/06/2011	Updated SMT Configuration, Adding triggers for procedures Obtain a GLN number (customers)	§ 3 § 4.1.1, § 4.1.2 § 4.1.2.2
0.006	20/01/2011	Update supplier and customer activation paragraphs	§ 4.1.2 and § 4.1.1
0.005	31/03/2010	Add customer activation	§ 4
0.004	18/11/2009	Included communication of user details	All pages
0.003	16/10/2009	Implementation of comments by Didier Thunus	All Pages
0.002	13/10/2008	Included DIGIT/R approval and SMT configuration	All pages
0.001	28/09/2009	Initial version	All pages

Table of Contents

Table of figures

1. INTRODUCTION

This document describes the procedures to be followed when to configure Open E-Prior to add customers and suppliers and to configure the interchange agreement between them.

2. DATABASE CONFIGURATION

2.1. Procedures

2.1.1. USER AND SUPPLIER CREATION

This procedure is triggered after a Supplier has completed the CTP and has signed the ICA.

The following diagram details the steps to be executed by the e-PRIOR support team in order to register a supplier in the e-PRIOR database.

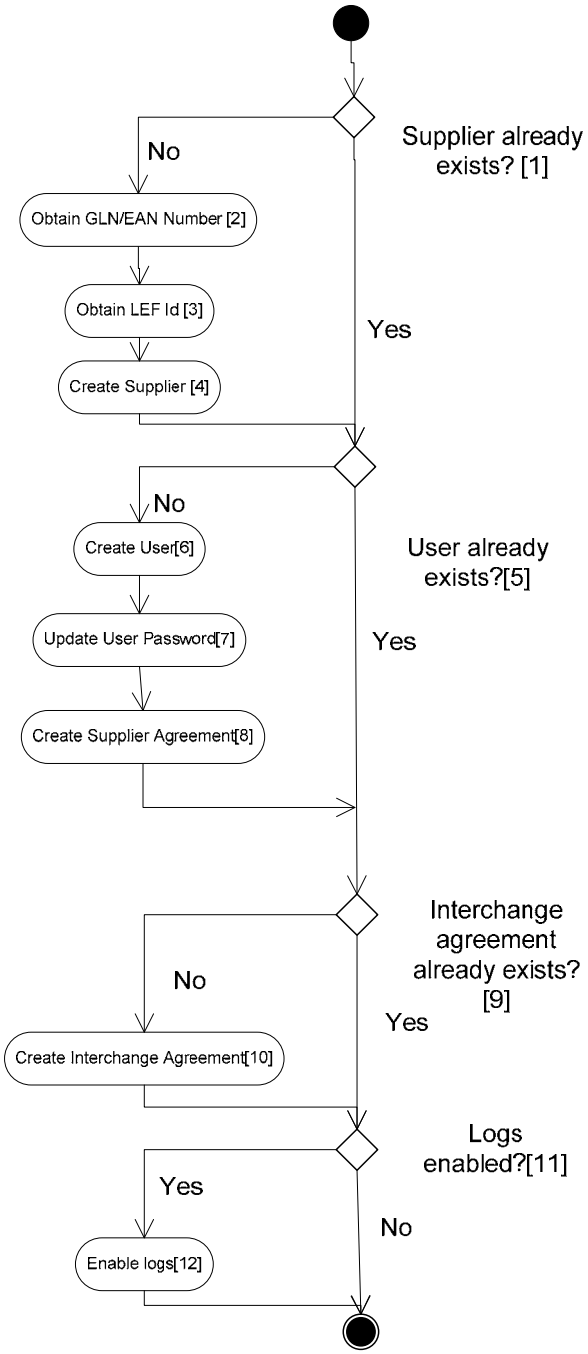

Figure 2 – User/Supplier Creation diagram

2.1.1.1. Check if the supplier already exists [1]

The Supplier can already exist and be linked to a different user in the context of another module of e-PRIOR (e.g. a Supplier uses different Users for e-Invoicing and e-Ordering).

The following SQL statement can be used.

```
select * from sup_tb_supplier where sup_name = 'SupplierXXX'
```

2.1.1.2. Obtain LEF ID Number [2]

Not relevant for Open e prior

2.1.1.3. Obtain GLN/EAN or VAT Number [2]

Suppliers in open e-Prior must be uniquely identified. We do recommend the use of GLN numbers (these can be obtained from GS1). Vat numbers can also be used to identify suppliers.

Currently the system does not support multiple identifiers for a single supplier. The same Supplier identified in the platform via it's GLN and Via it's VAT number will be considered as two different suppliers by the system.

2.1.1.4. Create supplier [4]

The following SQL statement can be used.

```
insert into sup_tb_supplier
    (SUP_NAME,SUP_EAN_CD)
values
('SupplierXXX','EANXXX')
```

2.1.1.5. Check if the user already exists [5]

The following SQL statement can be used.

```
select * from sup_tb_user where usr_name = 'UserXXX'
```

Attention (implicit rule): a new user name should not be already used by a back-office user! So the following query should return no result:

```
select * from epr_tb_backend where BCK_USER_NAME = 'UserXXX'
```

2.1.1.6. Create user [6]

The following SQL statement can be used.

```
insert into sup_tb_user
(USR_NAME,USR_EMAILADDRESS)
values
('UserXXX','mail@mail.mail')
```

Remark: It is also possible to update the e-mail address field afterwards by using the following statement:

```
update sup_tb_user
set USR_EMAILADDRESS = 'mail2@mail.mail'
where USR_NAME = 'UserXXX'
```

2.1.1.7. Update user password [7]

The following groovy script can be executed in the admin console. It provides original and encoded value for the password.

```
import java.security.MessageDigest;
import sun.misc.BASE64Encoder;

//Generate a random password
StringBuffer randomPassword = new StringBuffer();
for(int i=0 ; i<8; i++){
    double d = Math.random()*55;
    int j = (int)d;
    if (j<10)
          randomPassword.append((char)(48+j));
    else if (j < 36){
          randomPassword.append((char)(55+j));
    }else{
          randomPassword.append((char)(61+j));
    }
}

//Assign the random password value. If one wants a given value,
//Just replace randomPassword.toString() by "your_password_value"
String password = randomPassword.toString();

MessageDigest digest = MessageDigest.getInstance("SHA-1");
byte[] hash = digest.digest(password.getBytes());

String hashDB = "{SHA-1}"+ new BASE64Encoder().encode(hash);

//Display the original password value between <code></code> tags.
result.append("\nOriginal:&lt;code&gt;"+password+"&lt;/code&gt;\n");

//Display the encoded password value between <code></code> tags.
result.append("\nEncoded:&lt;code&gt;"+hashDB+"&lt;/code&gt;\n");
```

The following SQL statement can be used to update the password field in the database.

```
update sup_tb_user
set USR_PASSWORD = 'ENCODED_PASSWORD'
where USR_NAME = 'UserXXX'
```

2.1.1.8. Create Supplier Agreement [8]

The following SQL statements can be used.

1. Get user ID

```
select usr_id from sup_tb_user where usr_name = 'UserXXX'
```

2. Get supplier ID

```
select sup_id from sup_tb_supplier where sup_name = 'SupplierXXX'
```

3. Create supplier agreement

```
insert into sup_tb_agreement
(AGR_SUP_ID, AGR_USR_ID)
values
(SUPPLIERXXID,USERXXID)
```

2.1.1.9. Check if interchange agreement already exists [9]

The following SQL statements can be used.

1. Identify the supplier agreement
    a. Find the user ID

```
select usr_id from sup_tb_user where usr_name = 'UserXXX'
```

    b. Find the supplier ID

```
select sup_id from sup_tb_supplier where sup_name = 'SupplierXXX'
```

    c. Check for existing agreement

```
select agr_id from sup_tb_agreement where AGR_SUP_ID = SUPXXXID and
AGR_USR_ID = USRXXID
```

2. Identify the internal department ID

```
select IDP_ID from EPR_TB_INTERNALDEPT IDP_ID where IDP_NAME = 'DIGIT
Finances'
```

3. Identify the customer agreement

```
select idb_id from EPR_TB_INTERNALDEPT_BACKEND join
EPR_TB_INTERNALDEPT
 on idb_idp_id = idp_id join EPR_TB_BACKEND on idb_bck_id = bck_id
 where idp_name ='DIGIT Finances' and BCK_NM = 'ABAC Assets Invoice'
```

To obtain the relevant back-office name (second parameter) the following query can be used:

```
select bck_nm from epr_tb_backend
```

4. Determine the profile for that interchange agreement.

    The following statement gives the list of the supported profiles:

```
select * from EPR_TB_PROFILE
```

5. Check for existing interchange agreement for a given profile

```
select * from EPR_TB_INTERCHANGEAGR where ICA_AGR_ID =
SUPPLIER_AGREEMNT_ID and ICA_IDP_ID = INTERNAL_DPT_ID and ICA_PRF_ID
= PROFILE_ID and ICA_IDB_ID = IDB_ID
```

(Red values are parameters identified in point 1 to 4)

2.1.1.10. Create interchange agreement [10]

The following SQL statement can be used.

```
insert into EPR_TB_INTERCHANGEAGR
(ICA_AGR_ID,ICA_IDP_ID,ICA_PRF_ID, ICA_IDB_ID)
values
(SUPPLIER_AGREEMNT_ID, INTERNAL_DPT_ID, PROFILE_ID, IDB_ID)
```

Attention (implicit rules): the following statements should provide no result (before and after the creation of the interchange agreement!):

```
select count(*),agr_sup_id,agr_usr_id, idb_idp_id, ICA_PRF_ID from
EPR_TB_INTERCHANGEAGR join
EPR_TB_INTERNALDEPT_BACKEND on ica_idb_id = idb_id join
sup_tb_agreement on ica_agr_id = agr_id
having count(*) >1
group by agr_sup_id,agr_usr_id, idb_idp_id, ICA_PRF_ID

select count(*),agr_sup_id,idb_idp_id, idb_bck_id, ICA_PRF_ID from
EPR_TB_INTERCHANGEAGR join
EPR_TB_INTERNALDEPT_BACKEND on ica_idb_id = idb_id join
sup_tb_agreement on ica_agr_id = agr_id
having count(*) >1
group by agr_sup_id,idb_idp_id, idb_bck_id, ICA_PRF_ID
```

2.1.1.11. Check if logs are enabled [11]

The following SQL statements can be used.

1. Find the relevant endpoint identifier

```
select EDP_ID, PHR_DEFAULT_VALUE from EPR_TB_ENDPOINT join
EPR_TB_SERVICE on EDP_SRV_ID = SRV_ID join UTL_TB_PHRASE on
SRV_PHR_ID = PHR_ID
```

2. Find the log level identifier

```
select EVL_ID, PHR_DEFAULT_VALUE from REF_TB_EVENTLEVEL join
UTL_TB_PHRASE on EVL_PHR_ID = PHR_ID
```

3. Find the log configuration for given supplier agreement ID and internal department ID

```
select * from epr_tb_loglevel LGL_AGR_ID = SUPPLIER_AGREEMNT_ID and
LGL_IDP_ID = INTERNAL_DPT_ID
```

2.1.1.12. Enable logs [12]

The following SQL statement can be used.

```
insert into epr_tb_loglevel
(LGL_AGR_ID, LGL_IDP_ID, LGL_EDP_ID,LGL_EVL_ID)
values
(SUPPLIER_AGREEMNT_ID, INTERNAL_DPT_ID, ENDPOINT_ID, LOG_LEVEL)
```

2.1.2. BACK OFFICE AND CUSTOMER CREATION

.

The following diagram details the steps to be executed by the e-PRIOR support team in order to register a customer in the e-PRIOR database. The procedure is described in § 4.1.2.

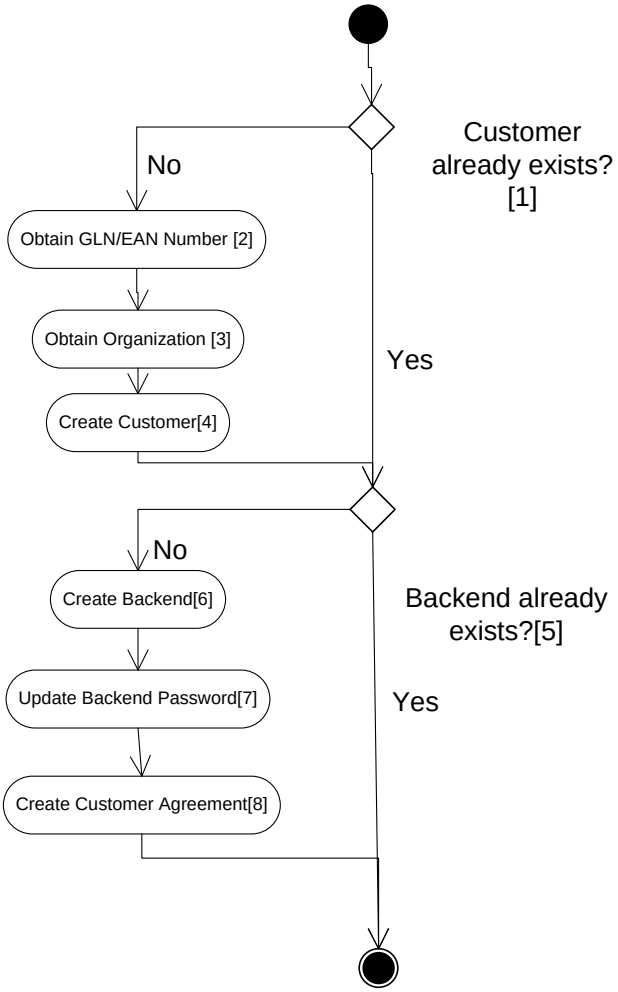

Figure 3 – Back Office/Customer Creation diagram

2.1.2.1. Check if the customer already exists [1]

The customer can already exist and be linked to a different back office in the context of another module of e-PRIOR (e.g. a Customer uses different back offices for e-Invoicing and e-Ordering).

The following SQL statement can be used.

```
select * from epr_tb_internaldept where IDP_NAME = 'CustomerXX'
```

2.1.2.2. Obtain GLN/EAN or VAT Number [2]

Customers in open e-Prior must be uniquely identified. We do recommend the use of GLN numbers (these can be obtained from GS1). Vat numbers can also be used to identify customers.

Currently the system does not support multiple identifiers for a single customer. The same customer identified in the platform via it's GLN and Via it's VAT number will be considered as two different customers by the system

2.1.2.3. Obtain Organization ID [3]

Not relevant for open e prior.

2.1.2.4. Create customer [4]

The following SQL statement can be used.

```
insert into epr_tb_internaldept
    (IDP_NAME ,IDP_EAN_CD)
values
('SupplierXXX','EANXXX')
```

2.1.2.5. Check if the back office already exists [5]

The following SQL statement can be used.

```
select * from epr_tb_backend where BCK_USER_NAME = 'UserXXX'
```

Attention (implicit rule): a new user name should not be already used by a supplier user! So the following query should return no result:

```
select * from sup_tb_user where usr_name = 'UserXXX'
```

2.1.2.6. Create back office [6]

We first need to identify the functionality to use:

```
select * from EPR_TB_FUNCTIONALITY
```

```
(INV for invoicing, ORD for ordering, RQS for sourcing (erequest))
```

The following SQL statement can be used to create the back office.

```
insert into epr_tb_backend
(BCK_NM,BCK_FCT_CD,BCK_USER_NAME)
values
('NAME', 'FUNCTIONALIY_CODE', 'USER')
```

2.1.2.7. Update back-office password [7]

The following groovy script can be executed in the admin console. It provides original and encoded value for the password.

```
import java.security.MessageDigest;
import sun.misc.BASE64Encoder;

//Generate a random password
StringBuffer randomPassword = new StringBuffer();
for(int i=0 ; i<8; i++){
    double d = Math.random()*55;
```

```
        int j = (int)d;
        if (j<10)
              randomPassword.append((char)(48+j));
        else if (j < 36){
              randomPassword.append((char)(55+j));
        }else{
              randomPassword.append((char)(61+j));
        }
    }

    //Assign the random password value. If one wants a given value,
    //Just replace randomPassword.toString() by "your_password_value"
    String password = randomPassword.toString();

    MessageDigest digest = MessageDigest.getInstance("SHA-1");
    byte[] hash = digest.digest(password.getBytes());

    String hashDB = "{SHA-1}"+ new BASE64Encoder().encode(hash);

    //Display the original password value between <code></code> tags.
    result.append("\nOriginal:&lt;code&gt;"+password+"&lt;/code&gt;\n");

    //Display the encoded password value between <code></code> tags.
    result.append("\nEncoded:&lt;code&gt;"+hashDB+"&lt;/code&gt;\n");
```

The following SQL statement can be used to update the password field in the database.

```
update epr_tb_backend
set BCK_USER_PASSWORD = 'ENCODED_PASSWORD'
where BCK_USER_NAME = 'UserXXX'
```

2.1.2.8. Create Customer Agreement [8]

The following SQL statements can be used.

1. Get back-office ID

```
select BCK_ID from epr_tb_backend where BCK_USER_NAME = 'UserXXX'
```

2. Get customer ID

```
select IDP_ID from epr_tb_internaldept where IDP_NAME = 'CustomerXX'
```

3. Create customer agreement

```
insert into EPR_TB_INTERNALDEPT_BACKEND
(IDB_IDP_ID , IDB_BCK_ID )
values
(CUSTOMERXXID,BACKENDXXID)
```

2.2. Procedure for modifications

2.2.1. UPDATE SUPPLIER EAN CODE

The following SQL statement can be used.

```
update sup_tb_supplier
```

```
set
    SUP_EAN_CD = 'EANYYY'
where
    sup_name = 'SupplierXXX'
```

2.2.2. UPDATE CUSTOMER EAN CODE

The following SQL statement can be used.

```
update EPR_TB_INTERNALDEPT
set
    IDP_EAN_CD = 'EANYYY'
where
    IDP_NAME = 'CustomerXXX'
```

3. COMMUNICATION OF USER DETAILS

3.1. User password

The user password will be provided to the Supplier contact by phone or e-mail.