Technical Design Specification



|  |  |  |
| --- | --- | --- |
| Author | : | Prithiviraj S |
| Creation Date | : | September 27, 2021 |
| Last Updated | : |  |
| Document Ref | : |  |
| Version | : | Draft |

1. **Title, Subject, Last Updated Date, Reference Number**, **and** **Version** are marked by a Word Bookmark so that they can be easily reproduced in the header and footer of documents. When you change any of these values, be careful not to accidentally delete the bookmark. **You can make bookmarks visible by selecting Tools->Options…View and checking the Bookmarks option in the Show region.**

**Approvals:**

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |

**BANK PAYMENT DOCUMENTATION**

**CONTENT:**

1. **ETEXT 1**

1.1 WHY ETEXT? 1

1.2 DEFINITION AND TYPES 1

1. **COMPLETE PAYMENT FLOW 1**

2.1 FUNCTIONAL SPECIFICATION DOCUMNET(FSD) 1

2.2 MAPPING SHEET 2

2.3 OUTPUT FLAT FILE(TEXT FILE) 2

2.4 XML 3

2.4.1 SYSTEM GENERATED XML 3

2.4.2 CUSTOM XML 3

1. **PAYMENT PROCESS PROFILE 3**
   1. PAYMENY PROCESS PROFILE CREATION 4
   2. TRANSMISSION SETUP CONFIGURATION 4

3.2.1 HOW TO SETUP TRANSMISSION CONFIGURATION? 5

* 1. PGP KEYS(Pretty Good Privacy) 5
  2. SIGNING KEYS 5
  3. STEPS TO UPLOAD PGP AND SIGNING KEYS IN UCM - INBOUND 6
  4. STEPS TO UPLOAD PGP AND SIGNING KEYS IN UCM - OUTBOUND 6
  5. STEPS TO UPLOAD BANK GIVEN PUBLIC KEY 7

3.7.1 DOWNLOAD FUSION GENERATED PUBLIC KEY 8

* 1. NAMING CONVENTIONS 8
  2. PAYMENT TYPES 8

3.9.1 INBOUND WITH KEYS FLOW 8

3.9.2 OUTBOUND WITH KEYS FLOW 9

3.10 ALGORTHIM 9

3.10.1 TYPES AND DEFINITIONS 9

3.10.2 CIPHER 10

3.10.3 AES-128 10

3.10.4 BLOWFISH 10

3.10.5 HASHING 10

3.10.6 SHA 10

1. **ETEXT TEMPLATE 10**
   1. MANDATORY FORMAT SETUP 10
   2. DELIMITER\_BASED 10
   3. FIXED\_POSITION\_BASED 10
   4. DIFFERENCE BETWEEN DELIMITER\_BASED & FIXED\_POSITION\_ BASED 11
   5. DIFFERENT SCENARIOS AND CONCEPTS IN ETEXT TEMPLATE 12
2. **EXTENSIBLE STYLESHEET LANGUAGE TEMPLATE(XSLT): 13**
   1. CONCEPTS AND USAGE OF XSLT 14
3. **GENERAL NOTES FOR ETEXT AND XSLT 14**
4. **ETEXT:**
   1. **WHY ETEXT?**

* Company/Organization want to pay multiple invoices to multiple suppliers.
* In order to make payment to supplier in easy way we go for payment automation through bank. Bank will made the payment to all suppliers using payment automation.
* For that company want to send the flat file with bank specific format, in order to generate that format(flat file) e-text is introduced.
* Using e-text template prepared and output file is send to bank.
  1. **DEFINITIONS AND TYPES:**

Electronic Text(ETEXT) is a type of RTF(Rich Text Format) based template used to generate the output flat file (TEXT FILE)with the bank specific format.

**TYPES OF ETEXT:**

1. **Electronic Fund Transfer(EFT)** - Fixed Position Based Format Text File.
2. **Electronic Data Interchange** - Delimiter Based Format Text File.

**Electronic Fund Transfer(EFT):**

Electronic transmission of financial and payment to bank.

**Electronic Data Interchange:**

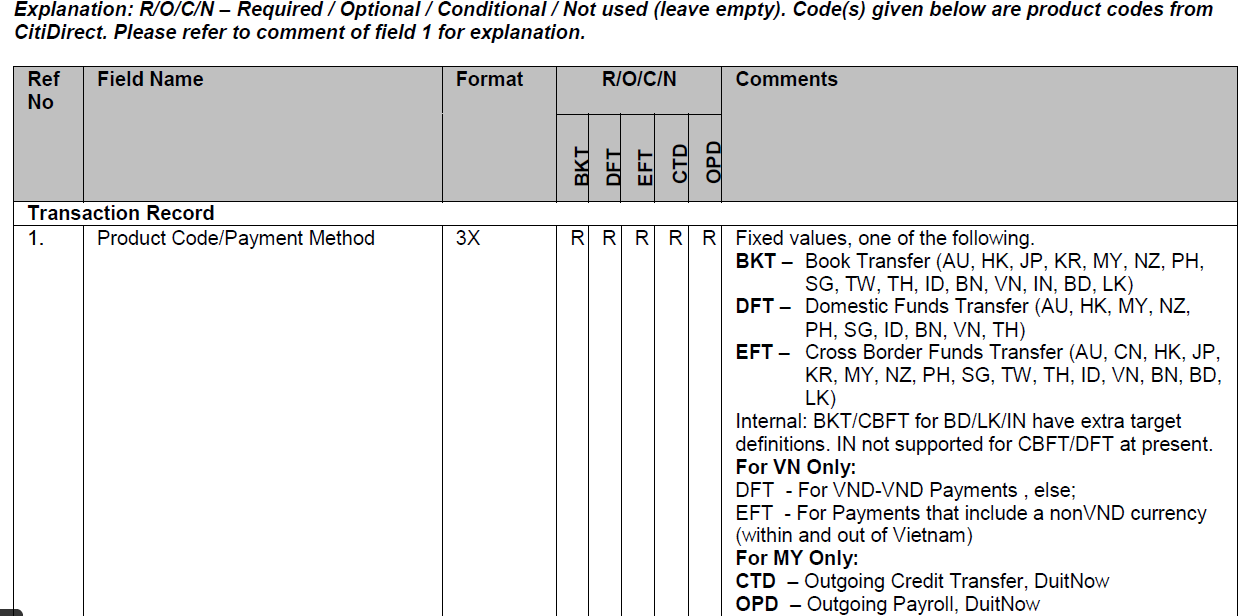
Electronic transmission of financial and payment to bank and interchange the business documents(eg: PO, Invoice etc..)

1. **PAYMENT FLOW:**
2. Bank(company’s bank account) give FSD to company to prepare output flat file according to that particular bank format.(financial team in bank side means they send mapping sheet along with spec).
3. Company financial team receive the FSD and prepare the mapping sheet according to that spec and give it to technical team.
4. Technical team check the mapping sheet with the spec after received it from financial team.
5. Technical team want to export the XML file by query the table IBY\_TRXN\_DOCUMENTS(table in which the payment hit in the oracle cloud(fusion) by financial team and they give the payment number) using that number we want to search that particular payment and copy it for XML file.(select document type 100).
6. Technical team start to prepare the e-text template and send that to bank.
7. Bank received that flat file and automat the payment to the bank(supplier’s contain bank account).

**2.1** **FUNCTIONAL SPECIFICATION DOCUMENT(FSD):**

* Company receive it from bank.
* It contains the necessary field name, datatype of that field, length and position of that field, required condition of that field.
* Using that, mapping sheet will prepare.
* Specification will vary according to bank and also slightly vary according to type -FIXED\_POSITION\_BASED AND DELIMITER BASED.

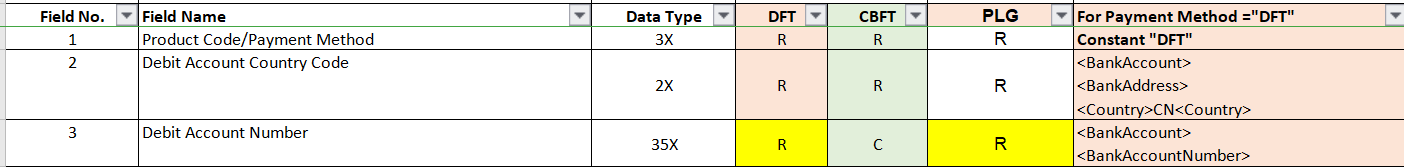
**SAMPLE SNAP:**



**2.2 MAPPING SHEET:**

* Functional team will prepare the mapping sheet with help of FSD.
* Mapping sheet contain field name, datatype, length, field condition, XML tag. It will slightly change according to the spec.
* XML tag can be taken from sample XML according to that field name.

**SAMPLE MAPPING SHEET:**



**2.3** **OUTPUT FLAT FILE(TEXT FILE):**

* Output flat file(bank required format file to automat payment) will send by company to bank.
* Using that file bank will automat the payment to supplier or multiple supplier.

**SAMPLE OUTPUT TEXT FILE:**

DFT@MY@111111111@MYR@16064.04@@20210712@10002@@@@@@@@@@@@SYNTECH CHEMICALS PTE LTD@2A SECOND CHIN BEE ROAD@UNIT NUMBER AND BUILDING NAME @ADDRESS LINE3 @@0020253704@@DBS BANK LTD@@@SG618781@C9@003@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@SYNTECH CHEMICALS PTE LTD@INT@@@@ENQUIRYCNPAYMENTS@THE-ASCOTT.COM@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

XML FILE:

* It consists of tags - Used in mapping sheet.

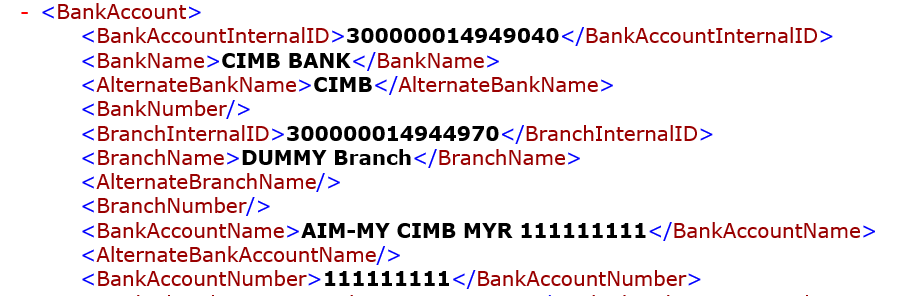
**SAMPLE TAG USAGE IN MAPPING SHEET.**



* XML used to fetch the data for that particular field.

**2.4 XML:**

**SAMPLE XML TAG WITH DATA:**



* In template creation it plays a vital role as a path(X-path used to navigate the XML file).

**SAMPLE USAGE OF X-PATH IN TEMPLATE:**



EG: BankAccount -> BankAccountNumber -> **111111111**

**2.4.1 SYSTEM GENERATED XML:**

Function team made the payment in oracle and give the payment number to technical. Technical team search the payment number in table IBY\_TRXN\_DOCUMENTS and copy the document type 100 record and paste in notepad and save as .XML extension(to get as a XML file).

**2.4.1 CUSTOM XML:**

When we can’t able to achieve the customer specific task using system generated xml we go for custom XML. In that we manually type the code and export the xml file to achieve the customer needs.

**NOTE:** So in order to communicate or transmission between company and bank. We have to connect cloud between company and bank using payment process profile transmission setup configuration.

1. **PAYMENT PROCESS PROFILE:**

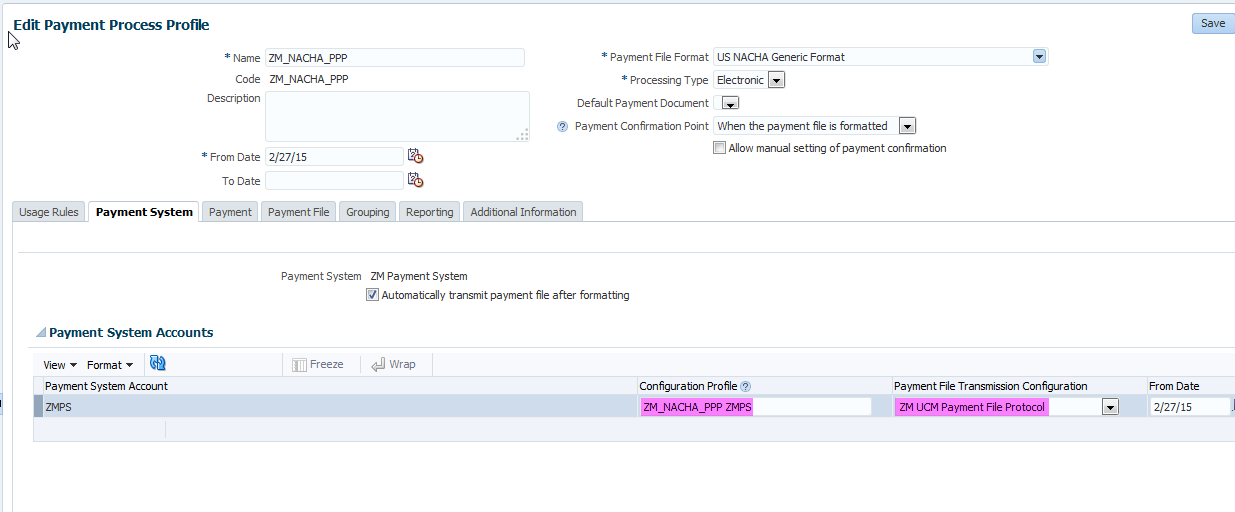
After the technical team prepared the e-text template:

* Functional team and technical team setup the transmission configuration to send the output file to bank.
* LOAD IN FUSION USING UCM:They setup and connect the technical prepared e-text template with XML and functional prepared payment file and they also setup if we have PGP keys or Signing keys process.
* After loaded in fusion using UCM, at the time of transmission using File Transfer Protocol(FTP) file got signed and flat file generated and encrypted using public key and that flat file send to bank.
  1. **PAYMENY PROCESS PROFILE CREATION :**

**Navigation:** Setup and Maintenance ---> Manage Payment Process Profile

Create a new PPP or update an existing one to add payment system and transmission details. Perform the following setups related to transmission:

* Under Payment System Tab, select and add the payment system created earlier. This will add a record for the payment system account associated with this payment system.
* Under Payment File Transmission Configuration, select the transmission configuration we created in previous(CREATION TRANSMISSION SETUP MENTIONED BELOW).
* If you wish to automatically trigger the payment file transmission after formatting phase, check the option "Automatically transmit payment file after formatting"



At this point, we can submit the payment process profile to select installments, process payments and create formatted payment file. This payment file is then transmitted to destination specified in the transmission configuration.

Payment files saved under UCM server can be accessed using Fusion Applications File Import and Export functionality.

**3.2** **TRANSMISSION SETUP CONFIGURATION:**

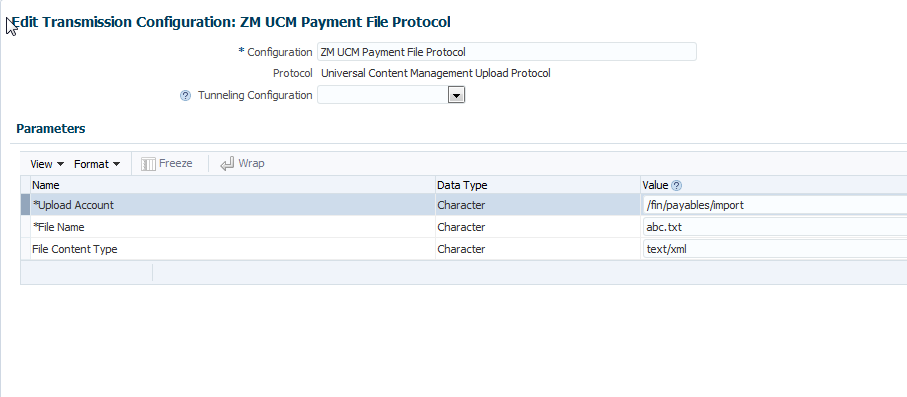
* This setup is mandatory to make transmission/communication between company and bank.
* Setup to made connection between own server cloud(host) and bank host.
* To make a transaction between own server cloud(host) and bank host through Oracle Web Center Content server , the Transmission Configuration must be set up . After that, a connection was made between the two for transmission.

**3.2.1 HOW TO SETUP TRANSMISSION CONFIGURATION?**

**Navigation:** Setup and Maintenance--->Manage Transmission Configurations

Create a new transmission configuration by selecting Universal Content Management Upload Protocol in select protocol LOV and click create. Enter following details:

* Upload Account: /fin/payables/import
* FileName:  Specify a static file name (abc.txt)
* File Content Type: text/xml



**3.3** **PGP KEYS(**Pretty Good Privacy)**:**

* In order to make the transmission more secure PGP keys are used.
* Two type of PGP Keys : Public PGP key and Private PGP key.
* Receiver only create the PGP keys and send the Public PGP key to sender for encryption.
* Receiver holds the Private PGP key to decrypt the fie send by the sender.
* So the file get transmission in a secure way.

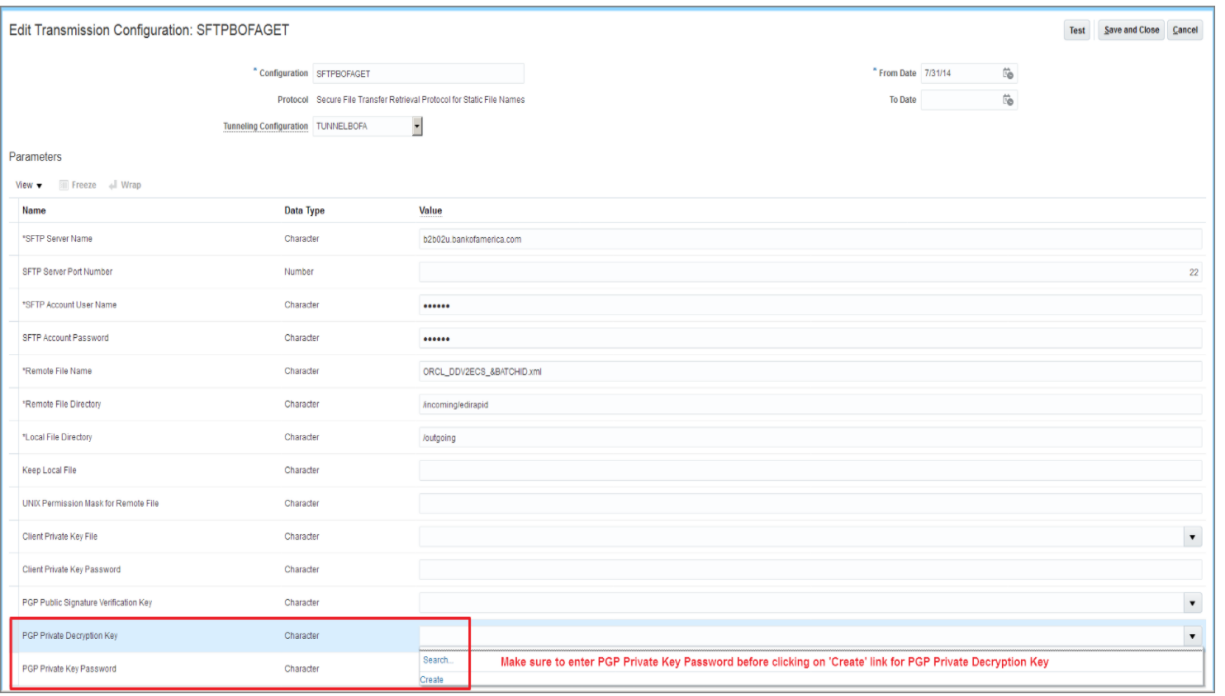
**3.4 SIGNING KEY:**

**WHY WE USE SIGNING KEY? USAGE?**

1. Integrity :  Detects whether that file or message has been modified.
2. Authentication : Verify Cryptographically the person who signed a given message.
3. Non - Repudiation : Prevent re-claim.

* Two types of signing keys Public Signing key and Private Signing key.
* Signing keys are generated by the sender.
* Sender will sign the file using Private Signing key and send the Public Signing key to the receiver to verify that file got modified or not as discussed above.

**3.5 STEPS TO UPLOAD PGP AND SIGNING KEYS IN UCM - INBOUND:**



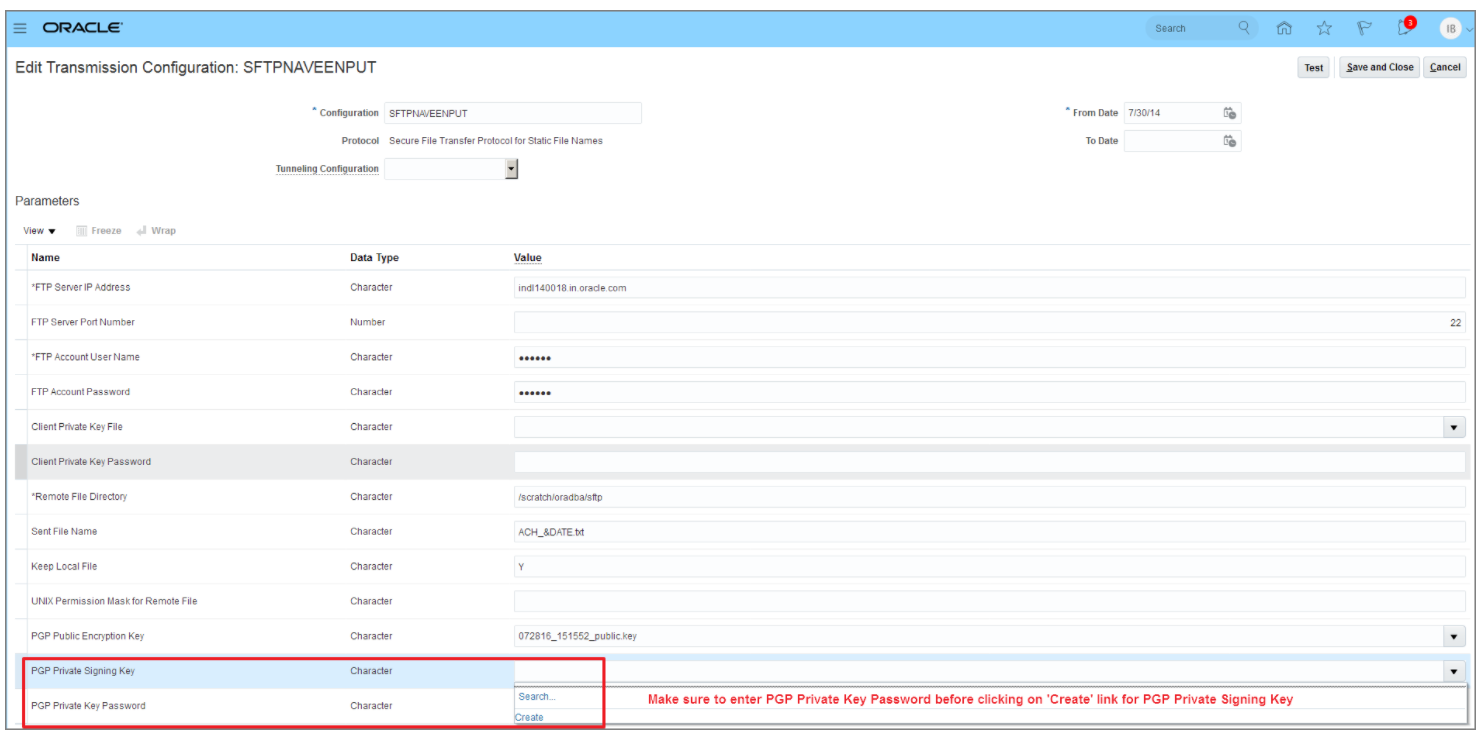
### **PGP Private Decryption Key**

* Key generated by you to decrypt the inbound encrypted file.
* You can select the **Create** option from the LOV which will automatically generate the key and link the private key with your transmission configuration.
* It will also generate a public key file which you can download from UCM (File Import and Export) and share with your bank.
* The bank will use your public key file to encrypt the acknowledgement / bank statement file.

### PGP Public Signature Verification Key

* Bank given key which will be used for validating the digital signature of inbound acknowledgement file or bank statement.
* Once you upload the bank provided signature verification public key via UCM, you can select the same here.
* Once you select the file here, it will be imported automatically and you do not need to manually submit the 'Import Security Credential Job'.

**3.6 STEPS TO UPLOAD PGP AND SIGNING KEYS IN UCM - OUTBOUND:**



### **PGP Public Encryption Key**

* Bank given key which will be used for encrypting the outbound payment file.
* Once you upload the bank provided encryption public key file via UCM, you can select the same here.
* Once you select the file, it will be imported automatically and you do not need to manually submit the 'Import Security Credential Job'.

### **PGP Private Signing Key**

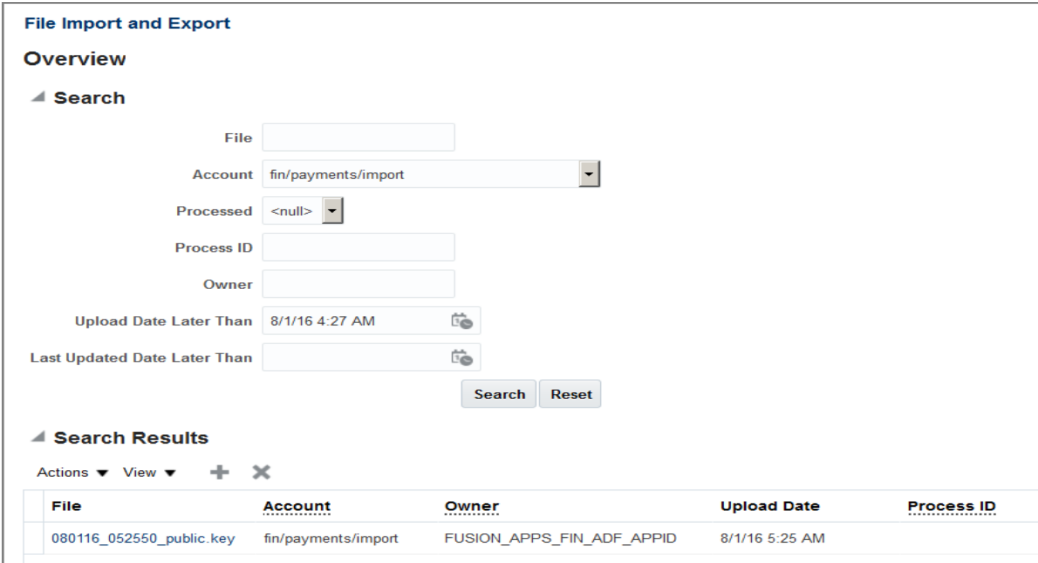
* Key generated by you to digitally sign the outbound payment file.
* You can select the **Create** option from the LOV which will automatically generate the key and link the private key with your transmission configuration.
* It will also generate a public key file which you can download from UCM (File Import and Export) and share with your bank.
* The bank will use your public key file to verify the digital signature for the payment files which you will be transmitting to bank.

### **3.7 Steps to Upload Bank Given Public Key File**

1. Rename the bank-provided key file by including ‘\_public.key’ as a suffix. Also, make sure the key file name doesn’t have any special characters other than an underscore (\_).
2. Navigate to File Import and Export.
3. Upload the bank given key file in account ‘fin/payments/import’.
4. Navigate to your desired transmission configuration in the transmission configuration page and select the uploaded key file from LOV in the related parameter. The key name in the LOV will be the same which you uploaded in UCM.
5. Once You select the key and save the configuration, the key will be automatically imported in Fusion.

**3.7.1 DOWNLOAD FUSION GENERATED PUBLIC KEY:**

* Select the **Create** option in the transmission configuration for the key related parameter.
* Navigate to File Import and Export.
* Search the generated key file using account ‘fin/payments/import’.
* Download the key file which will have similar name as private key file generated and attached in the transmission configuration.



**3.8 NAMING CONVENTIONS:**

* Sender will create the PGP keys at that time keys download in the name of:

**Extension When Download:**

* fusion-key\_pub.asc
* fusion-key\_priv.asc

**Extension When Upload to UCM:**

* fusion-key\_public.key
* fusion-key\_secret.key

**Key length** - 1024 or 2048

**3.9 PAYMENT TYPES:**

There are two types of payment type:

* INBOUND PAYMENT
* OUTBOUND PAYMENT

**3.9.1 INBOUND PAYMENT:**

BANK SEND THE STATEMENT FILE TO COMPANY.

**INBOUND FLOW WITH PGP AND SIGNING KEY:**

**Company** creates **PGP** keys --->  **Company** sends **public** key to **Bank** ---> ***Bank*** *creates signing keys --->* ***Bank*** *sends the public signing key to* ***Company*** *--->* ***Bank*** *sign the file using* ***private*** *signing key* ---> **Bank** encrypts file using the **public** key ---> **Bank** sends the encrypted file to **Company**  ---> **Company** decrypts the file using **private** key ---> ***Company*** *verify the file using* ***public*** *signing key.*

|  |  |
| --- | --- |
| **Parameter** | **Purpose** |
| PGP Public Signature Verification Key | Bank given key which will be used for validating the digital signature of inbound acknowledgement file or bank statement. Once you upload the bank provided signature verification public key via UCM, you can select the same here. Once you select the file here, it will be imported automatically and you do not need to manually submit the 'Import Security Credential Job'. |
| PGP Private Decryption Key | Key generated by you to decrypt the inbound encrypted file. You can select the **Create** option from the LOV which will automatically generate the key and link the private key with your transmission configuration. It will also generate a public key file which you can download from UCM (File Import and Export) and share with your bank. The bank will use your public key file to encrypt the acknowledgement / bank statement file. |

**3.9.2 OUTBOUND PAYMENT:**

COMPANY SEND THE FLAT FILE TO BANK.

**OUTBOUND FLOW WITH PGP AND SIGNING KEY:**

**Bank** creates **PGP** keys --->  **Bank** sends **public** key to **Company** ---> ***Company*** *creates signing keys --->* ***Company*** *sends the public signing key to* ***Bank*** *--->* ***Company*** *sign the file using* ***private*** *signing key* ---> **Company** encrypts file using the **public** key ---> **Company** sends the encrypted file to **Bank**  ---> **Bank** decrypts the file using **private** key ---> ***Bank*** *verify the file using* ***public*** *signing key.*

|  |  |
| --- | --- |
| **Parameter** | **Purpose** |
| PGP Public Encryption Key | Bank given key which will be used for encrypting the outbound payment file. Once you upload the bank provided encryption public key file via UCM, you can select the same here. Once you select the file, it will be imported automatically and you do not need to manually submit the 'Import Security Credential Job'. |
| PGP Private Signing Key | Key generated by you to digitally sign the outbound payment file. You can select the **Create** option from the LOV which will automatically generate the key and link the private key with your transmission configuration. It will also generate a public key file which you can download from UCM (File Import and Export) and share with your bank. The bank will use your public key file to verify the digital signature for the payment files which you will be transmitting to bank. |

**3.10 ALGORITHM :**

* Algorithm consists of bits. If we use larger bits algorithm the encryption process became stronger.
* Mainly algorithm is used to make the encryption and decryption process in stronger way.

**3.10.1 TYPES AND DEFINITIONS:**

1. RSA: Rivest-Shamir-Adleman
2. Cipher: AES-128, Blowfish, CAST5, 3DES
3. Compression: bzip2, zlib, .zip, uncompressed
4. Hash: SHA-1, SHA-256, SHA-224, SHA-512, MD5, SHA-384, RIPEMD-160

**3.10.2 CIPHER:**

* A cipher is the algorithm used to encrypt and decrypt data.
* Generally speaking, the more bits that a cipher uses during encryption, the stronger or more secure the encryption is.

**3.10.3 AES-128 :**

* AES128 uses the AES-128 cipher, which has a key size of 128 bits.

**3.10.4 BLOWFISH:**

* Blowfish encryption with a 64-bit block size and a variable-length key size from 32 bits to 128 bits.

**3.10.5 HASHING:**

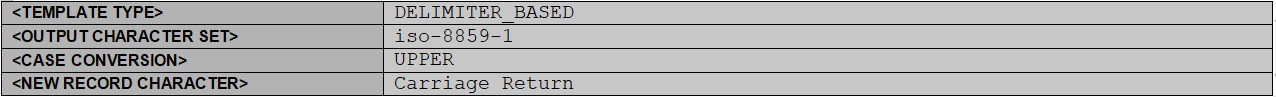
* It is a process to convert information to a shorter fixed value known as the key that is used to represent the original information.

**3.10.6 SHA - Secure Hashing Algorithm:**

* Cryptographic hash function used to produce a hash value from the input file or message.
* Family - SHA-1, SHA-256, SHA-224, SHA-512 etc.,.

1. **ETEXT TEMPLATE:**

**4.1 Template format setup:**

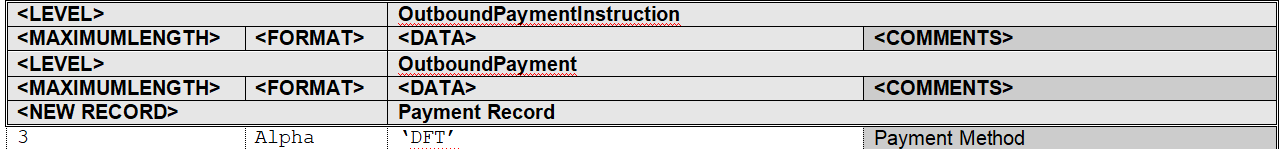


The above sample template format setup.

1. TEMPLATE TYPE set depends on the type of template either DELIMITER\_BASED or FIXED\_POSITION.
2. OUTPUT CHARACTER SET must be iso-8859-1.
3. CASE CONVERSION must be UPPER.
4. NEW RECORD CHARACTER must be Carriage Return.

**4.2 DELIMITER\_BASED:**

* Each field separated by any special character(delimiter).
* Each delimiter specify the ending of one field.

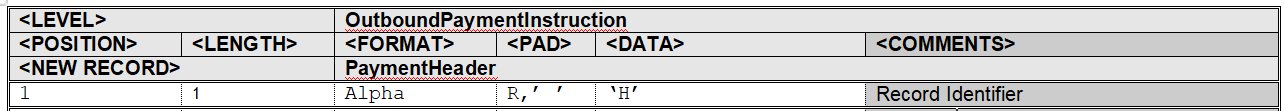


**4.3 FIXED\_POSITION\_BASED:**

* End of the each filed is identified depends upon the position of that particular field.
* In addition to the delimiter\_based template in fixed\_position\_based template we have:

**<POSITION>**  Specify the starting position of each field.

**<PAD> T**his is used to specify the alignment of field data in flat file.



**<LEVEL>**

We have to give the level name under which the fields are present in xml file.

HERE WE GIVE SOME OF IMPORTANT LEVELS USED IN TEMPLATE:

OutboundPaymentInstruction -> HEADER/MAIN LEVEL

OutboundPayment -> PAYMENT INFO LEVEL

DocumentPayable -> INVOICE DETAILS LEVEL

InstructionTotals -> AMOUNTPAY/TAXPAY LEVEL

**<MAXIMUMLENGTH>**

In this column we have to mention the maximum length of the specific field.

**<FORMAT>**

In this column we have to specify format of the specific field nothing but datatype of that field.

**<NEW RECORD>**

We use this set when we need a records in new line.

**<DATA>**

Here we have to mention the field specific data that we want in our output file.

Using path we can fetch the data from XML for that field.

**<COMMENTS>**

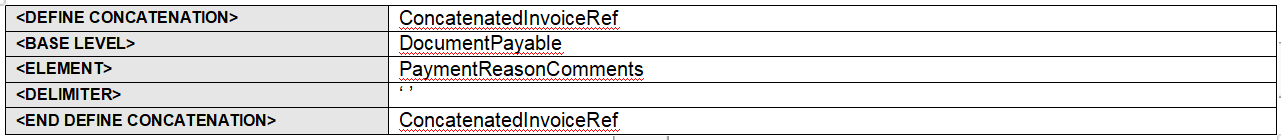
This column contains field name.

**4.4 DELIMITER\_BASED VS FIXED\_POSITION\_BASED:**

|  |  |
| --- | --- |
| **DELIMITER\_BASED** | **FIXED\_POSITION\_BASED** |
| <LEVEL>  <MAXIMUMLENGTH>  <FORMAT>  <NEW RECORD>  <DATA>  <COMMENTS> | <LEVEL>  <POSITION>  <LENGTH>  <FORMAT>  <PAD>  <DATA>  <NEW RECORD>  <COMMENTS> |
| End of each field is identified by delimiter. | End of each field is identified by the start position. |
| Bank will decide in which type they want.(purely bank suggestion.) | Bank will decide in which type they want.(purely bank suggestion.) |

**4.5 DIFFERENT SCENARIOS AND CONCEPTS IN ETEXT TEMPLATE:**

**SCENARIO:1**

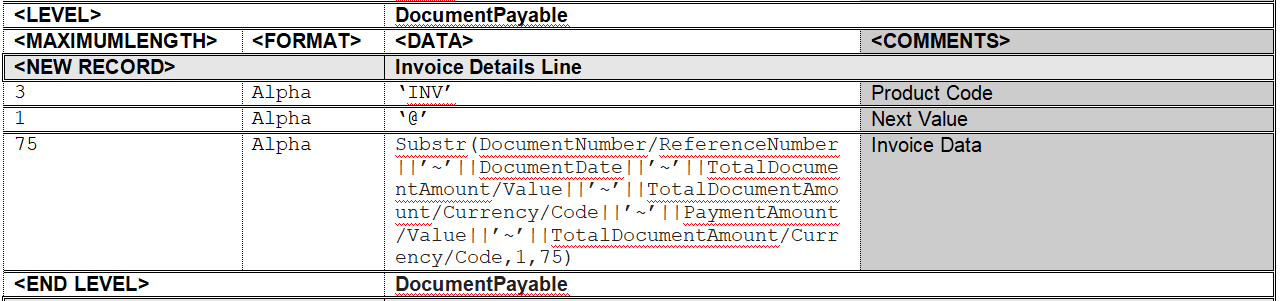


* In this scenario a file may have multiple DocumentPayable.
* Inside each DocumentPayable there is a element called paymentReasonComments inside this element only we have data for that field.
* If we mention only the path(DocumentPayable/PaymentReasonComments) in the template means it will display the data(if available) in flat file one time anyway it have multiple DocumentPayable.
* In order display all the data available in DocumentPayable/PaymentReasonComments we have to loop all the DocumentPayable/PaymentReasonComments available in that file.
* For that we have to used that format setup in top.
* **<DEFINE CONCATENATION> ConcatenatedInvoiceRef(give any name as we like to use)**
* **<BASE LEVEL> DocumentPayable**
* **<ELEMENT> PaymentReasonComments**
* **<DELIMITER> ‘ ’(in what way we have to seperate the data)**
* **<END DEFINE CONCATENATION> ConcatenatedInvoiceRef(end the definition)**

**SCENARIO 2:**

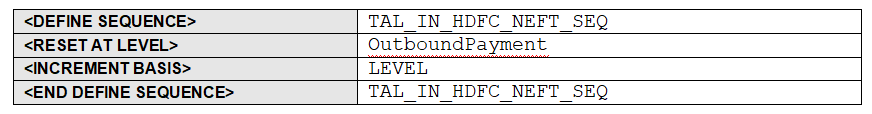
* DOCUMENT PAYABLE CONTAINS INVOICE DETAILS, TO DISPLAY ALL THE INVOICE DETAILS(LOOP) WITHIN THE FILE:

**SAMPLE SNAP:**



**SCENARIO 3:**

SAMPLE SNAP TO GENERATE SEQUENCE NUMBER:



* **<DEFINE SEQUENCE> TAL\_IN\_HDFC\_NEFT\_SEQ (ANY NAME AS WE WANT USER DEFINIED)**
* **<RESET AT LEVEL> OutboundPayment (WHEN THE LEVEL WANT TO RESET?AT EACH OUTBOUNDPAYMENT THE LEVEL WILL RESET)**
* **<INCREMENT BASIS> LEVEL (LEVEL WISE INCREAMENT)**
* **<END DEFINE SEQUENCE> TAL\_IN\_HDFC\_NEFT\_SEQ (CLOSING)**
* **If we want to sequentially increase the count for particular scenario in template we have to done the format setup at top.**
* **We have to use sequence\_number function to generate the sequence number using the setup.**
* **Eg: SEQUENCE\_NUMBER(TAL\_IN\_HDFC\_NEFT\_SEQ)**
* **SEQUENCE\_NUMBER - FUNCTION NAME**
* **(TAL\_IN\_HDFC\_NEFT\_SEQ) - SEQUENCE DEFINE NAME**

**SCENARIO 4: DISPLAY CONDITION**



* We have multiple INV\_GROUP in that we want WHT total tax calculation means we have to use the display condition as mentioned above.
* Fetching the data for particular record.

**NOTE**: After developed the e-text template run the template with xml using template viewer(e-text) to generate flat file for our verification.

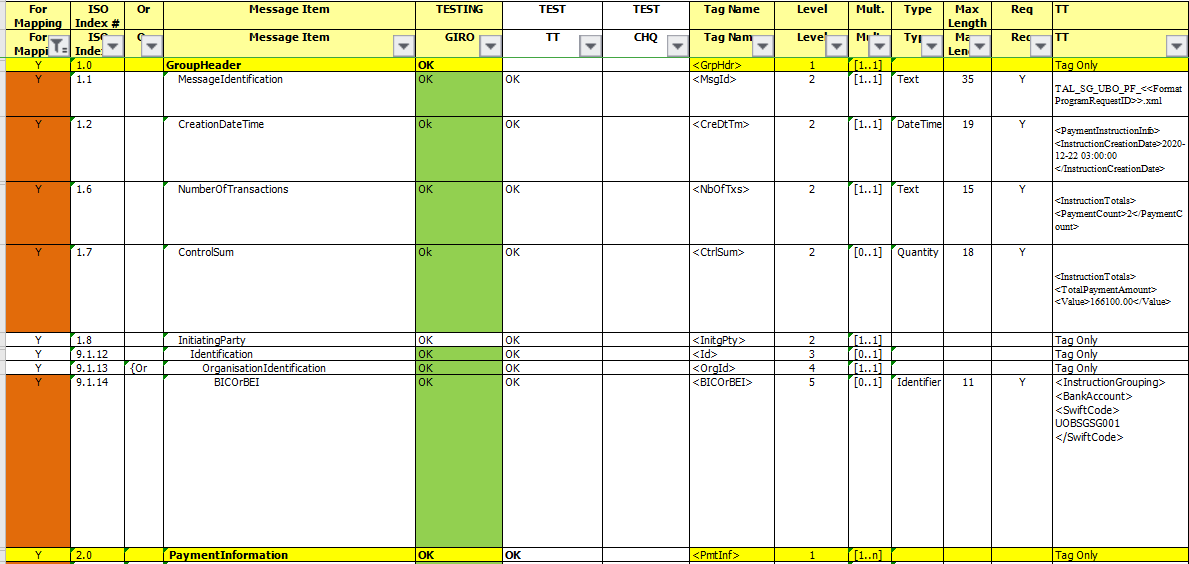
1. **Extensible Stylesheet Language Template(XSLT):**

XSLT, which stands for **Extensible Stylesheet Language Transformations**, is a declarative, XML-based language used for the transformation of XML documents into other XML documents. ... When the external requirements for an XML document do not achieve this match, XSLT is used.

AFTER DEVELOPED THE XSLT TEMPLATE RUN THE TEMPLATE WITH XML USING TEMPLATE VIEWER(RUN XSLT AND SAVE THE FILE AS .XML EXTENSION.

**5.1 CONCEPTS AND USAGE OF XSLT:**

**SAMPLE XSLT MAPPING SHEET:**



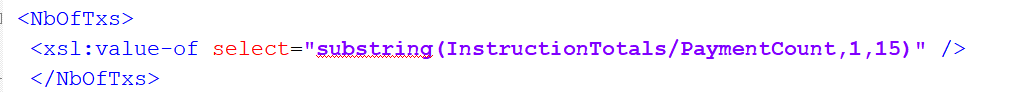
XSLT Tag name we have to use this tag in XSLT template inside this tag only we have provide the XML path.

This level is used where to open and close the XSLT tag. Level number indicate opening and closing position of each tag.

This is the path we have to mention inside the XSLT tag.

**XSLT Template:**

**SAMPLE:**



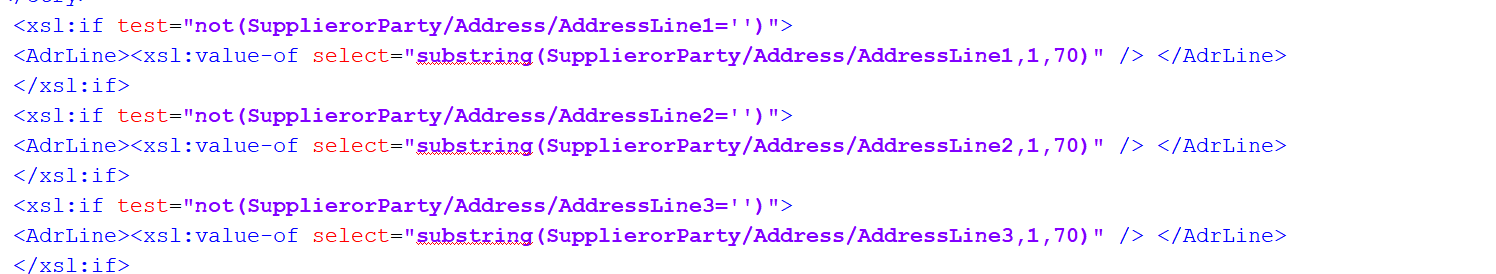
* <NbOfTxs> -> OPENING XSLT TAG.
* <XSL:VALUE-OF SELECT=”” /> -> WE HAVE TO USE THIS TAG TO MENTION THE XML PATH.
* </NbOfTxs> -> CLOSING XSLT TAG.

**SAMPLE USAGE OF FOR EACH IN XSLT TEMPLATE:**



* This is just like level we used in the e-text template(FOR LOOPING THE PROCESS).
* The file may have one or more payment so in order to loop all the payment for-each is used.
* Same scenario for document payable(invoice details).

**SAMPLE USAGE OF IF IN XSLT:**



* In this scenario if tag check for the availability of data if data is not null means it enters into main tag and fetch data.(IF usage)

1. **GENERAL NOTES FOR ETEXT AND XSLT:**

* If we use the outside level path inside the another level we have to mention ../ before the path.
* ../ nothing but pointing the outside level / parent level.
* Bank only decide whether we want to develop the e-text template or XSLT template based on the requirement.

============================**THANK YOU**===============================