Table des matières

1 Description de l'option EBICS de TBT/400

1.1 Concepts

1.1.1 Sécurité

Le protocole **EBICS** permet à un site client de se connecter aux banques pour y envoyer et/ou récupérer des fichiers métier (en remplacement des protocoles **ETEBAC3 et5**) via une connexion **TCP/IP**.

La nature non sécurisée de ce type de connexion rend obligatoire le chiffrement des données sensibles et leur signature électronique pour pouvoir être appliqué au secteur bancaire.

Pour ce faire **EBICS** nécessite l'utilisation de plusieurs certificats :

- un certificat de signature électronique (**A005**),
- un certificat d'authentification (**X002**),
- un certificat de cryptage (**E002**),
- un certificat pour la communication **SSL**.

Le standard **EBICS** (Version 2.4.2 du 16 février 2010) et la documentation officielle du CFONB (EBICS - Guide de mise en œuvre en France - Version 2.0) définissent le format et l'utilisation des trois premiers certificats et notamment:

- Clé RSA de 2048 bits,
- Algorithme de signature : RSA-SHA256,
- Validité : 5 ans si auto signé,
- Etc.

Le certificat SSL ne fait pas partie du standard.

TBT/400 permet la création de ce jeu de certificats qui sont, de ce fait, appelés « certificats auto signés ».

1.1.2 Les deux profils EBICS

Il existe deux profils de transmission en protocole EBICS :

- le profil T :
  - Succède au protocole ETEBAC 3,
  - Une unique signature de transport par transfert,
  - Utilisation de certificats de signature auto-signés autorisée,
  - La signature n'a pas pouvoir d'exécution:l'ordre de transfert doit être confirmé par un autre canal de communication avant d'être exécuté (le fax par exemple),
- le profil TS :
  - Succède au protocole ETEBAC 5,
  - Une signature de transport associée à une ou deux signatures TS par transfert,
  - Utilisation de certificats de signature sur support physique (Token USB par exemple),
  - La signature a pouvoir d'exécution:l'ordre de transfert est directement exécuté lors de sa réception par la banque.

Attention à bien prendre la mesure du choix de profil :

- Profil T : le certificat de signature n'est pas sécurisé, **la banque est responsable du flux transmis,**
- Profil TS : le certificat de signature est sécurisé, **la responsabilité du flux transmis est transférée au client, y compris en cas de maladresse, de doublons, de malveillance, etc.**

1.1.3 Identifiants

En EBICS l'identification banque/client se fait par les champs:

- HOSTID (identifie une banque),
- PARTNERID (identifie un numéro de contrat/abonnement),
- USERID (identifie un nom de l'utilisateur ou du service).

Le CFONB précise que chaque USERID doit être associé à un jeu de trois certificats (rien n'est précisé pour le certificat SSL).

2 Installation spécifique EBICS

2.1 Pré-requis

- Niveau d'OS :
  - Profil T : l'OS doit être au niveau V5R3M0 ou supérieur,
  - Profil Ts : l'OS doit être au niveau V5R4M0 ou supérieur,
- Les produits suivants doivent être installés
  - IBM HTTP Server for i5/OS DG1 base
  - Digital Certificate Manager SS1 option 34
  - CCA Cryptographic Service Provider SS1 option 35
  - Crypto Access Provider 128-bit AC3 (pour version OS/400 V5R3M0)

2.2 Pré-requis en profil TS

Les pré-requis en profil TS sont les même que ceux du profil T auxqules il faut ajouter :

- L'environnement WebTBT doit être entièrement opérationnel (se reporter à la documentation de ce dernier le cas échéant),
- Le produit Java SE 6 ou supérieur doit être installé sur le serveur IBM,
- Un Token USB (ou tout autre support matériel de certificat) doit être disponible à chaque utilisation,
- Un poste client (PC ou tablette) qui devra :
  - être équipé des dernières mises à jours recommandées par le fabricant matériel et par l'éditeur du système d'exploitation (Windows 7 au minimum pour les utilisateurs Microsoft Windows),
  - être équipé de la dernière version du logiciel Java,
  - être équipé des logiciels et drivers nécessaires à la bonne utilisation du Token USB selon les recommandation du fournisseur de ce dernier,
  - être autorisé à exécuter des Applets Java (signées par le certificat de l'éditeur de TBT/400) depuis un navigateur Internet Explorer ou Mozilla Firefox,
  - être autorisé à communiquer avec le serveur IBM sur les ports 10028, 10068 (TBT400/EBICS TS) et 10091 (WebTBT), ces valeurs pouvant être modifiées par paramétrage. Pour des raisons évidentes de sécurité il est conseillé de restreindre cet accès à votre réseau interne.

2.3 Digital Certificate Manager (DCM)

L'installation du **D**igital **C**ertificate **M**anager (DCM) est **impérative** puisqu'il permet de gérer tout ce qui concerne les couches SSL (obligatoire pour le protocole EBICS).

Après avoir vérifié les pré-requis précédents, lancez la commande `STRTCPSVR SERVER(*HTTP) HTTPSVR(*ADMIN)` pour démarrer le serveur d'administration web utilisé par le DCM.

La commande WRKACTJOB SBS(QHTTPSVR) permet de visualiser les jobs actifs spécifiques aux différents

serveurs web.

Ceux du serveur d'administration se nomment ADMIN, ADMIN1, ADMIN2, etc.

Si aucun job nommé ainsi n'existe dans le sous-système QHTTPSVR, il est très probable que le serveur d'administration web ne soit pas actif.

Dans ce cas, **il est impossible de poursuivre le paramétrage EBICS** (le SSL ne pourra pas être paramétré).

La commande `WRKJOB JOB(ADMIN)` peut vous aider à diagnostiquer un éventuel problème de configuration (il aussi probable que le problème soit lié à un défaut de PTFs).

Si au contraire les jobs sont actifs, essayez de vous connecter au serveur admin depuis un navigateur web à l'adresse :

```
http://adresse_iseries:2001
```

Après avoir saisi un nom d'utilisateur et un mot de passe cliquez sur « Gestionnaire de certificat numérique » ou «Digital Certificate Manager ».

Si la page s'affiche correctement, c'est que votre DCM est accessible et devrait vous permettre de paramétrer le SSL sur votre I5/OS.

Dans le cas contraire, vérifiez les messages d'erreurs et **corriger les problèmes avant de continuer le paramétrage EBICS**.

2.4 Création initiale des Certificats

Un premier jeu de certificats doit être créé. A cet effet utiliser la commande suivante :

```
IPLSP/IPSCRTEBIC  COMNAME('Client')
                  LOCALITY('Paris')
                  STATE('Ile de France')
                  COUNTRY('FR')
```

PS : Les autres champs de cette commande ne sont pas à modifier pour une installation standard (et pour un premier certificat).

Une fois la commande exécutée correctement **TBT/400** vous propose une « vue » de l'IFS où sont stockés les certificats (répertoire **/IFSTBTIPSC**).

- IPSTBTSUBS_A_APP.p12 Certificat de signature
- IPSTBTSUBS_E_APP.p12 Certificat de cryptage
- IPSTBTSUBS_X_APP.p12 Certificat d'authentification
- IPSTBTSUBS_APP.p12 Certificat SSL

2.4.1 En V5R4 et au delà

- Dans le DCM importer le certificat système sous le nom **IPSTBTSUBS_APP** (le mot de passe est 'PASSWORD').[1]
- Si le certificat existe déjà dans le DCM, ne pas le remplacer,
- Supprimer ce fichier de l'IFS (/IFSTBTIPSC).

[1] Le **DCM** est l'outil IBM de gestion des certificats ; après l'avoir démarré (**`STRTCPSVR SERVER(*HTTP) HTTPSVR(*ADMIN)),`** il est accessible en http su le port 2001**.**

2.4.2 En V5R3M0

- Renommer le certificat IPSTBTSUBS_APP.p12 en IPS530SUBS_APP.p12[2],
- Générer par Operation Navigator un certificat CA,
- Générer par Operation Navigator un certificat Système IPSTBTSUBS_APP,
- Ne pas supprimer le fichier de l'IFS.

[2] Le **DCM** 530 ne supporte pas l'importation de certificats auto signés ; de plus il n'est pas possible dans cette version de crypter/décrypter par application.

2.5 Certificats en profil TS

Dans ce mode de fonctionnement, TBT/400 n'a pas accès à la clé privée contenue dans le Token USB.

Il faut donc importer manuellement le certificat publique grâce à la commande suivante :

```
IPLSP/IPSCERTIFS CRTFNC(*INTREP)
                 NOMLOG(LOCCORP72219934)
                 CRTCTX(*LOCSIG)
                 IFSOBJ('/home/CORP72219934.cer')
```

Avec :

- CRTFNC : Code de fonction, ici « *INTREP » (remplacement d'un certificat local),
- NOMLOG : Correspondant local représentant le signataire (commence obligatoirement par LOC),
- CRTCTX : Contexte d'utilisation du certificat, ici « *LOCSIG » (certificat local de signature),
- IFSOBJ : Chemin d'accès du certificat à importer.

Cette manipulation est à faire pour tous les correspondants de signature EBICS TS.

2.6 Assignation du certificat SSL/TLS

La procédure d'installation a créé quatre applications (au sens DCM) ; il s'agit d'associer l'application au certificat précédemment importé (ou généré) :

- Associer IPSTBTSUBS_APP à IPSTBTSUBS_CLI(TBT/400)SslCli,
- Associer IPSTBTSUBS_APP à IPSTBTSUBS_SRV(TBT/400)SslSrv,
- Associer IPSTBTSUBS_APP à IPSTBTSUBS_AUT(TBT/400)SslAut,
- Associer IPSTBTSUBS_APP à IPSTBTSUBS_APP(TBT/400).

2.7 Table des Hosts

Il est nécessaire d'avoir accès à un serveur DNS, les serveurs diffusant des noms de host.

Cependant, peu de serveurs disposent d'une résolution inverse correcte. Pour assurer un suivi correct, et parfois améliorer les performances, il est souhaitable de définir les Serveurs en table des Hosts.

TBT/400 propose un programme pour le faire :

CALL IPLSP/IPSEBIHOST

Le source est disponible dans de fichier IPSSAMPLES de la bibliothèque IPLSP.

2.8 Sauvegardes des Certificats

Deux répertoires doivent être sauvegardés :

- **/QIBM/USERDATA/ICSS** qui contient tout le paramétrage SSL OS/400
- **/IfstbtIPSC** qui contient les certificats privées EBICS, ainsi que les certificats publics des partenaires

3 Initialisation d'une connexion avec une banque

Pour initialiser une nouvelle connexion :

- Effectuer les formalités d'abonnement EBICS,
- Paramétrer la banque dans l'annuaire **TBT/400,**
- Envoyer une première requête **INI** (envoi du certificat de signature _a = signature fichier),
- Envoyer une seconde requête **HIA** (envoi des certificats de cryptage _e et d'authentification _x = signature trames),
- Envoyer les lettres d'initialisation,
- Envoyer une troisième requête **HPB** (réception des certificats de cryptage et d'authentification de la banque).

A noter : En profil TS les requêtes INI et HIA sont spécifiques : se reporter au chapitre « Émission des requêtes techniques » pour plus d'information.

3.1 Considération TCP/IP pour le protocole EBICS

Le protocole EBICS utilise la pile de protocoles TCP/IP et, de ce fait, le paramétrage TCT/IP de votre i5/OS doit être correct et en particulier en ce qui concerne le client DNS.

En effet, les banques seront dans 99% des cas connues par ce que l'on appelle leur « Nom d'hôte » ou « Hostname ».

Il est donc fortement recommandé de paramétrer le client DNS de votre i5/OS de façon à ce qu'il soit capable de résoudre chacun de ces noms.

Pour vérifier ce paramétrage, depuis une ligne de commande saisissez :

- go tcpadm (appel du menu « TCP/IP Administration »),
- « 1. Configure TCP/IP »,
- « 12. Change TCP/IP domain information »,
- Vérifiez le champ INTNETADR qui devrait être renseigné en fonction des adresses correspondantes à vos serveurs DNS ou, à défaut, à celles de votre fournisseur d'accès à Internet.

Avant même tout paramétrage de TBT/400 vous devriez pouvoir réaliser la commande suivante :

- ping 'NOM DE HOST DE LA BANQUE'

Le nom d'hôte étant à récupérer depuis l'URL de la banque (sur votre contrat EBICS), par exemple :

- URL : ebics.bnpparibas.com => HOSTNAME: ebics.bnpparibas.com,
- URL : www.ebics.socgen.com/ebics/EbicsServlet => HOSTNAME: www.ebics.socgen.com,
- URL : pebics.cm-cic.com/ebicsweb/ebicsweb => HOSTNAME: pebics.cm-cic.com.

La commande PING doit renvoyer :

- Verifying connection to xxx at address 111.222.333.444.

Xxx étant le nom d'hôte de la banque et 111.222.333.444 son adresse résolue par l'un des serveurs DNS.

Il est à noter que la commande PING peut ne pas être concluante et s'achever avec le message :

- Connection verification statistics: 0 of 5 successful (0 %)

Ce n'est pas nécessairement une erreur et peut simplement vouloir dire que le serveur de la banque ne

« répond pas » à cette commande.

Pour notre test, il semble que le seul message d'erreur problématique soit :

- Unkonw host (où « Hôte inconnu » indiquant l'incapacité du client DNS à résoudre le nom d'hôte).

3.1.1 Cas particulier du profil TS

En profil TS, la signature électronique est générée par l'Applet Java IPSEBICSTS, cette dernière échangeant des informations avec TBT/400 au moyen d'une connexion HTTP/SSL.

Dans ce mode de fonctionnement TBT/400 est donc serveur et répond à des requêtes spéciales (format propriétaire) sur les ports 10028 et 10068 de votre serveur IBM (valeurs standards, modifiables dans le paramétrage TCP/IP).

Il est donc impératif que tous les postes clients susceptibles d'utiliser le module EBICS TS de TBT/400 soient capables de se connecter sur ces ports.

3.2 Correspondants modèles

3.2.1 IPLS$$$$PROFIL (protocole EBICS)

Ce correspondant n'est pas un correspondant réel ; il fournit des valeurs par défaut à l'ensemble des correspondants EBICS.

En particulier :

- Le nom des certificats utilisés
- Le profil par défaut d'émission (avec CR/LF)
- Le profil de réception

3.2.2 IPLS$$$EBICNF (protocole HTTP)

Ce correspondant n'est pas un correspondant réel ; il n'est pas modifiable et est utilisé par TBT/400 pour configurer l'Applet Java IPSEBICSTS .

3.2.3 IPLS$$$EBICTS (protocole HTTP)

Ce correspondant n'est pas un correspondant réel ; il n'est pas modifiable et est utilisé par TBT/400 pour sa communication avec l'Applet Java IPSEBICSTS.

3.3 Création d'un correspondant EBICS

TBT/400 dispose dans son annuaire d'une entrée standard pour les banques les plus connues.

Il est fortement recommandé des les utiliser directement plutôt que de les dupliquer et ce, pour éviter de compliquer inutilement la gestion des certificats des différents serveurs.

Si l'on prend l'exemple d'un accès à la BNP, il faut procéder ainsi :

- Entrez dans **TBT/400** : IPLSP/IPS
- « 4. Gestion de l'annuaire »,
- « 1. Définition des correspondants »,
- F10 sur le correspondant BNP.

3.3.1 Détail d'un correspondant

```
DANG   9941 Devt              Détail d'un correspondant     IPLS08    IPLSC
Type d'annuaire . . . $$$$EBICS  F4               Portée . . . . *GLOBAL
Nom du correspondant  . BNP                       Type réseau  . $$$$EBICS  F4
Libellé correspondant . Acces BNP
Commentaire utilisateur

Auteur  . . . . . .
Objet . . . . . .



                                      Suffixe N O,N   Trace        O,N
A l'attention de  . . .                               Impre.      O,N,C,B
Référence du message  .                               Scrut.      O,N
Emission mode puits . .     O,N           Messages demandés  .        O,N,C,B
Accusé demandé  . . . .     O,N,C         Avis  ====> Distri O Lectur Applic N
Mode transparent  . . .     O,N         Ajout caractères CR/LF      O,N
Suppression des blancs      O,N,L           Transfert ASCII . . .  Ccsi
Priorité réseau . . . .     N,U,H             Enreg. par segment  .  0 - 255
R. txt Lr     Tr   Ty   C      Ec   R.        bin Lr     Tr   Ty   C      Ec
Identifiant réseau  . .                                  Ha 2 C 3 S 1 Cm 6
```

Les champs importants dans cet écran sont :

- Nom du correspondant :
  - Dans le cas d'un correspondant local, ce nom doit commencer par LOC,
- Scrut (scrutation implicite des PSR lors de chaque émissions/réceptions – oui par défaut) =**SCRDEM**,
- Avis Distri (demande les PSR – oui par défaut) = **AVIDIS**,
- Transfert ASCII (émission ASCII/EBCDIC) = **ASCDEM (*associé au CCSID*)**,
- Les champs de la ligne 20 permettent de préciser un mode de réception spécifique à une banque (longueur d'enregistrements, gestion CR/LF, etc.) comme cela à toujours été le cas dans tous les protocoles de **TBT/40**0.

Après avoir validé les données appuyer sur F20 pour éditer l'écran suivant

3.3.2 Détail d'un correspondant EBICS (distant)

Disponible en profil T et TS.

```
DEBI   9931 Devt      Détail d'un correspondant EBICS       IPLS08     IPLSD
Type d'annuaire . . . . $$$$EBICS                  Portée . . . . *GLOBAL
Nom du correspondant  . BNP                        Type réseau  . $$$$EBICS
Libellé correspondant . TBT400*Acces BNP
Type d'entrée T T,S
Hstid émis  . BNPAFRPPXXX                        Cli
Prtid émis  . 30004BNPP
Usrid émis  . 123456_(carte_param_13_a_18)
Hstid attendu BNP                                Srv
Prtid attendu UNUSED
Usrid attendu BNP
Filetype  . . camt.xxx.cfonb$L$.dri
Test  . . . . N                    F4
Profil  . . . T                    F4
Corresp Sig1                       F4
UserId  . . .
Corresp Sig2                       F4
UserId  . . .
Nbr Sig . . .
Sélection d'application   A,C           Application par défaut           F4
F1=Hlp F3=Exi F6=Imp F7=Avn F8=Apr F9=Cmd F13=Hau F19=Gau F20=Dro F21=Def
F24=Bas                              Copyright Informatique Pour Les Sociétés
```

Renseigner les champs suivants :

- Type d'entrée :
  - T : pour un correspondant de type transport,
  - S : pour un correspondant de type signature – profile TS uniquement,
- Hstid émis (HOSTID de la banque indiqué sur le contrat),
- Prtid émis (PARTNERID de votre compte indiqué sur le contrat),
- Usrid émis (USERID de votre compte indiqué sur le contrat),
- Hstid attendu (nom de la banque utilisé pour documenter les lettres d'initialisation),
- Usrid attendu (nom de la banque utilisé pour stocker les certificats du serveur),
- Test (annonce le passage en mode test ou production),
- Profil :
  - T pour le profil EBICS T
  - S pour le profil EBICS TS
- Corresp Sig1 : Nom du correspondant TBT/400 associé à la première signature – profil TS uniquement,
- UserId : USERID associé au champ précédent – profil TS uniquement,
- Corresp Sig1 : Nom du correspondant TBT/400 associé à la deuxième signature – profil TS uniquement,
- UserId : USERID associé au champ précédent – profil TS uniquement,
- Nbr Sig : Nombre de signatures attendues pour ce correspondant (contrôler par TBT/400 à l'émission de tous fichiers EBICS en profil TS) – profil TS uniquement.

Le filetype constitue une valeur par défaut (si non renseigné dans la demande d'émission, celui de l'annuaire sera utilisé).

Après avoir validé les données appuyer sur F20 pour éditer l'écran suivant

3.3.3 Détail d'un correspondant EBICS (local)

Disponible en profil TS uniquement.

```
DEBI   9931 Devt      Détail d'un correspondant EBICS       IPLS08     IPLSD
Type d'annuaire . . . . $$$$EBICS                  Portée . . . . *GLOBAL
Nom du correspondant  . LOCCORP72219934            Type réseau  . $$$$EBICS
Libellé correspondant . Prof. TS - Signataire 01
Type d'entrée S T,S
Hstid émis  .                                     Cli
Prtid émis  .
Usrid émis  .
Hstid attendu                                     Srv
Prtid attendu
Usrid attendu
Filetype  . .
Test  . . . .                        F4
Profil  . . .                        F4
Corresp Sig1                         F4
UserId  . . .
Corresp Sig2                         F4
UserId  . . .
Nbr Sig . . .
Sélection d'application   A,C           Application par défaut           F4
F1=Hlp F3=Exi F6=Imp F7=Avn F8=Apr F9=Cmd F13=Hau F19=Gau F20=Dro F21=Def
F24=Bas                                Copyright Informatique Pour Les Sociétés
```

Dans le cas d'un correspondant de type signature, seul le champ « Type d'entrée » est renseigné.

L'intérêt d'un tel correspondant est de pouvoir retrouver son certificat pour la connexion SSL (voir l'applet Java).

Après avoir validé les données appuyer sur F20 pour éditer l'écran suivant

3.3.4 Détail d'un correspondant HTTP.

Dans cet écran vous pouvez spécifier l'url http permettant d'accéder au serveur EBICS de la banque.

```
DHTP   9994 Devt       Détail d'un correspondant Http     IPLS08 IPLSC
Type d'annuaire . . . . $$$$EBICS                  Portée . . . . *GLOBAL
Nom du correspondant  . BNP                        Type réseau  . $$$$EBICS
Libellé correspondant . Acces BNP

HTTP User . .
HTTP Password
HTTP Post URL

HTTP Ans  URL
```

Si rien n'est spécifié **TBT/400** sur connectera à la racine du serveur.

Post URL : URL à joindre

User+Password : ne sert à priori pas ; user et mot de passe du serveur http distant

Après avoir validé les données appuyer sur F20 pour éditer l'écran

3.3.5 Détail des paramètres TCP/IP

```
DTCP   9947 Devt        Détail des paramètres TCP/IP   IPLS08    IPLSC
Type d'annuaire . . . . $$$$EBICS                  Portée . . . . *GLOBAL
Nom du correspondant  . BNP                        Type réseau  . $$$$EBICS
Libellé correspondant . Acces BNP

Hostname IP distant . . ebics.bnpparibas.com

Adresse IP distant  . . 159.50.179.82

Port    IP distant  . .   443

Hostname IP local . . .

Adresse IP locale . . .

Usage adresse . . . . .

Utilisation Ssl . . . . O Protocole       Cipher       Lng  . T

Buffer Emission . . . .
```

En EBICS l'utilisation du mode SSL est obligatoire (mais son implémentation ne fait pas partie du standard).

Utilisation SSL : N, O,V,A ou F.

TBT/400 permet d'utiliser SSL en mode :

- **V**alidation du certificat (TBT/400 vérifie que le certificat n'a pas changé d'une session à l'autre),
- **A**uthentification (la couche SSL de l'OS vérifie la validité du l'autorité de certification),
- **F**ull (combinaison des deux options précédentes).

Attention l'authentification n'est pas possible avec les certificats auto signés.

Après avoir validé les données appuyer sur F20 pour éditer l'écran suivant

3.3.6 Détail des certificats de transport

Dans cet écran vous pouvez paramétrer les certificats à utiliser pour cette connexion.

Appuyer sur F21 pour afficher les valeurs par défaut.

```
DCRT   9973 Devt       Détail des certificats          IPLS08  IPLSC
Type d'annuaire . . . . $$$$EBICS                   Portée . . . . *GLOBAL
Nom du correspondant  . BNP                         Type réseau  . $$$$EBICS
Libellé correspondant . Acces BNP

Certificat local  Ssl . IPSTBTSUBS           K IPSTBTSUBS_CLI
                                               IPSTBTSUBS_SRV
                                               IPSTBTSUBS_AUT
Certificat remote Ssl . BNP                  K BNP
Certificat local  Aut . IPSTBTSUBS_X         K IPSTBTSUBS_X_APP
Certificat remote Aut . BNP_X                K BNP_X
Certificat local  Sig . IPSTBTSUBS_A         K IPSTBTSUBS_A_APP
Certificat remote Sig . BNP_A                K BNP_A
Certificat local  Cry . IPSTBTSUBS_E         K IPSTBTSUBS_E_APP
Certificat remote Cry . BNP_E                K BNP_E
Certificat local  Avd .                      K IPSTBTSUBS_A_APP
Certificat remote Avd .                      K BNP_A
```

Comme indiqué plus haut les champs « Certificat remote XXX » sont définies par le champ « Usrid attendu » de l'écran « Détail d'un correspondant EBICS ».

Lors du premier paramétrage il n'y a aucun certificat dans l'IFS de **TBT/400**, il est donc impossible d'en visualiser depuis cet écran (en rythme de croisière F10 sur un de ces champs vous donne accès au contenu du certificat et des lettres d'initialisation).

L'utilisation de la procédure de création initiale des certificats, les certificats locaux IPSTBTSUBS_X_APP IPSTBTSUBS_A_APP IPSTBTSUBS_E_APP ont été créés.

3.3.7 Détail des certificats de signature

Dans cet écran vous pouvez paramétrer les certificats à utiliser pour la connexion SSL entre TBT/400 et l'applet Java ainsi que pour la génération des lettres d'initialisation..

```
DCRT   9973 Devt           Détail des certificats           IPLS08     IPLSD
Type d'annuaire . . . . $$$$EBICS                  Portée . . . . *GLOBAL
Nom du correspondant  . LOCCORP72219934            Type réseau  . $$$$EBICS
Libellé correspondant . Prof. TS - Signataire 01

Certificat local  Ssl .                       K


Certificat remote Ssl .                       K
Certificat local  Aut .                       K
Certificat remote Aut .                       K
Certificat local  Sig . LOCCORP72219934       K
Certificat remote Sig . LOCCORP72219934       K
Certificat local  Cry .                       K
Certificat remote Cry .                       K
Certificat local  Avd .                       K
Certificat remote Avd .                       K



F1=Hlp F3=Exi F6=Imp F7=Avn F8=Apr F9=Cmd F10=Cer F13=Hau F19=Gau F20=Dro
F21=Def F24=Bas                        Copyright Informatique Pour Les Sociétés
```

Les valeurs sont imposées et reprennent les champ « Nom du correspondant ».

3.3.8 Lettres d'initialisation et visualisation des certificats

TBT/400 vous permet de contrôler les certificats utilisés ; vous pouvez utiliser une des méthodes suivantes

- Accès par le système de Menu comme décrit précédemment (F10)
- Utiliser la commande **IPLSP/IPSCERTIFS NOMLOG(BNP) CRTCTX(*LOCAUT)** pour voir le certificat local utilisé en authentification
- Utiliser la commande **IPLSP/IPSCERTIFS NOMLOG(BNP) CRTCTX(*ALLLOCEBI)** pour voir tous les certificats locaux utilisés en EBICS avec ce partenaire.

Chacune de ces méthodes donnent accès à un fichier que **TBT/400** génère dans le répertoire « /tmp/XXX », XXX représentant l'utilisateur courant.

Ce fichier peut être utilisé comme lettres d'initialisation pour un certificat :

- La première partie du fichier donne des informations sur le correspondant et le certificat utilisé.
- La deuxième partie (« Lettre d'initialisation du certificat XXXXX »), est à imprimer (pour les trois certificats), à faire signer par une personne habilitée et à envoyer à chaque banque (par courrier, fax, e-mail, etc.).

Voici un exemple de lettre d'initialisation (en gras les éléments validés par la banque) :

```
Lettre d'initialisation du certificat d'authentification
     ------------------------------------------------------

Date          : 01.01.2011
Heure         : 12:00:00
Host Id       : HHHHHHH
Banque        : BBBBBBB
User-ID       : UUUUUUU
Partner-ID    : PPPPPPP
Version       : Authentification X002

                Certificat d'authentification
                -----------------------------

Type          : X002

-----BEGIN CERTIFICATE-----




(Certificat en Base64)




-----END CERTIFICATE-----

       Hash du certificat d'authentification (SHA-256):
       ------------------------------------------------

(Hash SHA-256 du certificat)


Date :                      Signature :
```

En profil T les lettres d'initialisation doivent être imprimées pour :

- Le certificat de signature,
- Le certificat d'authentification,
- Le certificat de cryptage.

En profil TS les lettres d'initialisation doivent être imprimées pour :

- Les 3 certificats du correspondant de transport (les mêmes que pour le profil T),
- Le certificat de signature du premier signataire accompagné du certificat d'authentification et de cryptage du correspondant de transport,
- Le certificat de signature du deuxième signataire accompagné du certificat d'authentification et de cryptage du correspondant de transport.

Une connexion EBICS TS en mode double signature implique donc la génération de 9 lettres d'initialisation.

Les champs HOSTID et PARTNERID des lettres des correspondants de signature sont identiques à ceux de la lettre des correspondants de transport.

Le champs USERID des lettres des correspondants de signature sont à copier depuis les champs « UserId Sig1 » ou « UserId Sig2 » du correspondants de transport.

Attention : les correspondants locaux sont partagés avec l'ensemble des correspondants distants, lorsque les lettres d'initialisation sont générées les champs HOSTID, PARTNERID et USERID ne peuvent donc pas être renseignés automatiquement. Ils sont proposés avec des valeurs par défaut qu'il convient de modifier avant impression.

3.3.9 Un correspondant = Un couple compte/banque

Une entrée dans l'annuaire représente un compte dans une banque.

La « dérogation » classique (souvent constatée) est la multiplication des entrées *quasi identiques*, seul le filetype étant modifié.

Il est recommandé dans ce cas d'utiliser **une seule entrée** , en faisant varier le filetype lors de la demande de transfert.

Si malgré tout l'utilisateur persiste à vouloir multiplier le nombre d'entrées (fortement déconseillé), il est impératif que le champ « Userid attendu » utilisé soit identique à chaque fois.

En effet, ce dernier sert à composer le nom du certificat (de la banque) dans l'IFS.

3.3.10 Cas de plusieurs comptes dans une banque

La banque demande à avoir un jeu de trois certificats par compte (rien n'est spécifié quant au SSL)

Se référer au « chapitre 4 - Utilisation de plusieurs comptes » pour plus de détail sur ce point.

3.4 Émission des requêtes techniques

Pour envoyer les requêtes techniques(INI, HIA, HPB, etc. dites requêtes, RT dans la supervision du trafic) :

- Entrez dans **TBT/400** : IPLSP/IPS
- « Émission d'un fichier »,
- Dans le menu « Émission EBICS » renseignez les champs Bibliothèque, Fichier et Membre avec la valeur spéciale *DUMMY (= fichier fictif))
- Choisissez le type d'annuaire $$$$EBICS et le correspondant de votre choix puis « ENTER »
- Dans le menu « Émission EBICS » sur le champ « Filetype », appuyez sur F4 puis sélectionnez la requête à envoyer :
  - INI : envoi du certificat de signature du correspondant en cours,
  - $IN1$: correspond à la requête INI du premier correspondant de signature (« Corresp Sig1 » du menu « Détail d'un correspondant EBICS distant »),
  - $IN2$: correspond à la requête INI du deuxième correspondant de signature (« Corresp Sig2 » du menu « Détail d'un correspondant EBICS distant »),
  - HIA : envoi des certificats authentification et chiffrement du correspondant en cours,
  - $HI1$: correspond à la requête HIA du premier correspondant de signature (« Corresp Sig1 » du menu « Détail d'un correspondant EBICS distant »),
  - $HI2$: correspond à la requête HIA du deuxième correspondant de signature (« Corresp Sig2 » du menu « Détail d'un correspondant EBICS distant »),
- Appuyez sur F5 pour rafraîchir l'écran afin d'avoir un aperçu sur l'action à effectuer
- Si tout est correct appuyez sur F11 pour émettre la requête
- Le message « Message inséré dans la file d'attente » confirme que tout s'est bien passé


```
GMFF   0005 Devt          Emission d'un fichier           IPLS08     IPLSP

Bibliothèque  . . . . . *DUMMY                            *LIBL, *CURLIB, F4
Fichier . . . . . . . . *DUMMY                            F4 pour liste
Membre  . . . . . . . . *DUMMY                            F4 pour liste
Clé utilisateur . . . .                                Protocole  . . T
Type d'annuaire . . . . $$$$EBICS                   F4 Portée . . . . *GLOBAL
Nom du correspondant  . BNP                         F4 Type réseau  . $$$$EBICS
Libellé correspondant . Acces BNP
Suppression demandée  . N                                 O,N,C,H
Duplication demandée  . N                                 O,N,I
Date d'envoi différé  . 20110407 Heure 10165248
Date limite d'envoi . . 20110414 Heure 10165248
Ligne TBT/400 . . . . . *TCP                              F4 pour liste
Identifiant réseau  . . BNP                               F4 pour liste
Application émettrice . $$$$$TBT                          F4 pour liste
Application destinat. . $EXTERNB                          F4 pour liste
Environnement . . . . . *NO
Hash  . . . . . 2                Cryp  . . . . . 3
Sign  . . . . . 1                Comp  . . . . . 6
```

```
EEBI   9932 Devt               Emission EBICS               IPLS08     IPLSP
Type d'annuaire . . . . $$$$EBICS                      Portée  . . . *GLOBAL
Nom du correspondant  . BNP                            Type réseau . $$$$EBICS
Libellé correspondant . Acces BNP                      Environnement *NO

Hstid émis  . BNPAFRPPXXX
Prtid émis  . 30004BNPP
Usrid émis  . 123456_(carte_param_13_a_18)
Hstid attendu
Prtid attendu
Usrid attendu
Filetype  . . $INI$
Ordertype . . INI
Orderatt  . . DZNNN
Dat val . . .
Ebcdic  . . . O
Test  . . . . N
Profil  . . . T
```

Après traitement, le message doit avoir « EBICS_OK » comme libellé d'acheminement, signifiant que la banque à bien reçu le(s) certificat(s).

4 Utilisation de plusieurs comptes

4.1 Définition du besoin

Le même jeu de certificats peut être utilisé pour se connecter à plusieurs banques tant que vous ne disposez que d'un seul compte dans chacune d'elle.

Dans le cas contraire il vous faudra créer autant de jeu de certificats que nécessaire pour respecter le standard :

- **Un USERID = un jeu de certificat**

Voici deux exemples illustrant cette règle :

4.1.1 Exemple mono compte

La société X communique avec les banques BANQUE1, BANQUE2 et BANQUE3 dans lesquelles elle n'a qu'un seul compte :

- CPTXB1 : Compte (unique) de la société X dans BANQUE1,
- CPTXB2 : Compte (unique) de la société X dans BANQUE2,
- CPTXB3 : Compte (unique) de la société X dans BANQUE3.

Dans ce cas de figure un seul jeu de certificat pourra être utilisé pour se connecter aux trois banques.

4.1.2 Exemple multi comptes

La société Y communique avec les banques BANQUE1, BANQUE2 et BANQUE3.

A la différence du premier exemple, Y dispose d'un seul compte dans BANQUE1et BANQUE2 et de deux comptes dans BANQUE3 :

- CPTYB1 : Compte (unique) de la société Y dans BANQUE1,
- CPTYB2 : Compte (unique) de la société Y dans BANQUE2,
- CPT1YB3 : Compte N°1 de la société Y dans BANQUE3,
- CPT2YB3 : Compte N°2 de la société Y dans BANQUE3.

Dans ce cas de figure il faudra crée un deuxième jeu de certificats car la BANQUE3 exigera un jeu pour CPT1YB3 et un autre pour CPT2YB3.

Le premier pourra être utilisé pour les comptes CPTYB1, CPTYB2 et CPT1YB3 et le deuxième pour CPT2YB3.

4.2 Création d'un jeu de certificats supplémentaire

Utilisez la commande suivante :

```
IPLSP/IPSCRTEBIC COMNAME('COMPTE2')
                 LOCALITY('Paris')
                 STATE('Ile de France')
                 COUNTRY('FR')
                 CERTNAME(*CLI)
```

où « COMPTE2» représente le nom du deuxième jeu de certificats.

Une fois la commande exécutée correctement **TBT/400** vous propose une « vue » de l'IFS où sont stockés les certificats (répertoire **/IFSTBTIPSC**) :

- COMPTE2_A_APP.p12 Certificat de signature
- COMPTE2_E_APP.p12 Certificat de cryptage
- COMPTE2_X_APP.p12 Certificat d'authentification
- COMPTE2_APP.p12 Certificat SSL

4.2.1 V5R4 et au delà

- Dans le DCM importer le certificat système sous le nom **COMPTE2_APP** (le mot de passe est 'PASSWORD'),[3]
- Si le certificat existe déjà dans le DCM, ne pas le remplacer,
- Supprimer ce fichier de l'IFS (/IFSTBTIPSC).

4.2.2 V5R3M0

- **Renommer le certificat COMPTE2_APP.p12 en COMPTE2530_APP.p12**[4],
- Générer par Operation Navigator un certificat CA,
- Générer par Operation Navigator un certificat Système COMPTE2_APP,
- Ne pas supprimer le fichier de l'IFS.

---

3 Le **DCM** est l'outil IBM de gestion des certificats ; après l'avoir démarré (**`STRTCPSVR SERVER(*HTTP) HTTPSVR(*ADMIN)),`** il est accessible en http su le port 2001**.**

4 Le **DCM** 530 ne supporte pas l'importation de certificats auto signés ; de plus il n'est pas possible dans cette version de crypter/décrypter par application.

4.2.3 Gestion des application supplémentaires

Le standard EBICS ne précise rien quant au certificat SSL et il est donc **tout à fait légitime d'utiliser le certificat SSL par défaut** (jeu de certificat par défaut) pour touts les transferts EBICS.

Cependant, il est possible de paramétrer **TBT/400** pour qu'il utilise un certificat SSL plutôt qu'un autre grâce à la notion d'application.

La procédure d'installation ne créé que les quatre applications par défaut, celles qui seront utilisées pour les nouveaux jeux de certificat devront être crées manuellement (seul l'application de type « Client » est à créer pour les jeux de certificats supplémentaires) :

- Dans le DCM, sélectionnez le « Certificate store *SYSTEM » puis,
- Manage Applications,
- Add Application,
- Client - Add a client application,

Dans le menu « Add Application » renseignez les champs suivants (les autres peuvent conserver leur valeur par défaut) :

- Application ID : COMPTE2_CLI,
- Define the CA trust list : NO,
- Certificate revocation processing : YES,
- Application description : COMPTE2_CLI (TBT/400)SslCl.

Il ne reste qu'à associer le certificat COMPTE2_APP à l'application COMPTE2_CLI(TBT/400)SslCli.

4.2.4 Création des entrées dans l'annuaire et affectation des certificats

Une fois l'application créée et le certificat assigné vous pouvez éditer le correspondant et renseigner les champs suivants :

- Certificat local Ssl :COMPTE2
- Certificat local Aut :COMPTE2_X
- Certificat local Sig :COMPTE2_A
- Certificat local Cry :COMPTE2_E

afin d'obtenir l'écran suivant :

```
DCRT   9973 Devt          Détail des certificats          IPLS08     IPLSC
Type d'annuaire . . . . $$$$EBICS                 Portée . . . . *GLOBAL
Nom du correspondant  . BNPCOMPTE2                Type réseau  . $$$$EBICS
Libellé correspondant .

Certificat local  Ssl . COMPTE2             K


Certificat remote Ssl .                     K
Certificat local  Aut . COMPTE2_X           K
Certificat remote Aut .                     K
Certificat local  Sig . COMPTE2_A           K
Certificat remote Sig .                     K
Certificat local  Cry . COMPTE2_E           K
Certificat remote Cry .                     K
Certificat local  Avd .                     K
Certificat remote Avd .                     K
```

En appuyant sur F10 sur chacun des certificats locaux, vous pouvez vérifier la sélection des certificats prévus pour ce correspondant.

5 EBICS profil TS

5.1 Généralité

En profil TS les ordres de virements sont obligatoirement signés par un certificat électronique stocké sur un support matériel sécurisé (exp : un Token USB).

Ce support exige traditionnellement une connexion de type USB bien que n'importe quel autre moyen peut très bien être utilisé.

Cependant, ce type de connexion est incompatible avec l'utilisation courante des solutions sur serveurs IBM : il est en effet impensable de proposer une solution logiciel imposant à l'utilisateur de se déplacer en salle machine (dont l'accès est en général limité pour des raisons de sécurité) à chaque fois qu'il doit signer un fichier.

La solution retenue est donc d'utiliser une applet Java exécutée sur un poste client (PC ou tablette) et communiquant avec TBT/400 en protocole HTTP/SSL.

L'applet est distribuée par le module WebTBT. Son utilisation est donc conditionnée au strict respect des pré-requis présentés dans un précédent chapitre.

5.2 Paramétrage

Le paramétrage global de l'applet est fait au niveau de WebTBT. Il convient donc de se référer à la documentation de ce dernier pour ce qui concerne le paramétrage de l'environnement Java, la gestion du timeout, etc.

Les port utilisés par TBT/400 pour communiquer avec l'applet Java sont les ports HTTP SSL et AUT. Ils sont à paramétrer dans le menu TCP/IP :

- Configuration du système,
- Paramètres généraux,
- TCP/IP,
- TCP/IP – Serveurs.

```
PTBS   1081 Devt              TCP/IP - Serveurs                IPLS08     IPLSD

Hostname IP local  . . AS400D.ipls.local

Adresse IP locale  . . 10:2:3:134                                      Prot N
Usage adresse  . . . . O      Ssl O     Aut N     Buf émi  131072 Por Srv 10000
Port PeSIT . . . . . . 10040  Ssl 10060 Aut 10020 <----- Paramètres SSL ------>
Port Odette  . . . . .  3305  Ssl  6619 Aut 10021 Option 2 Exit N Protocole
Port FTP . . . . . . . 10042  Ssl 10062 Aut 10022 Ssh 10049 <-----Ciphers----->
Port TBT . . . . . . . 10043  Ssl 10063 Aut 10023 C024 C02C C028 C030 009D 003D
Port AS2 . . . . . . . 10044  Ssl 10064 Aut 10024 0035 C023 C02B C027 C02F 003C
Port RxCP  . . . . . . 10045  Int 10085           009C 002F C008 C012 000A C007
Port X400  . . . . . .   102  Ssl 10066 Aut 10026 C011 0005 0004 0009 0003 0006
Port EBICS . . . . . . 10047  Ssl 10067 Aut 10027 C006 C010 003B 0002 0001
Port HTTP  . . . . . . 10048  Ssl 10068 Aut 10028 Crt Cli O Ipinv . O  Ip V6  O
Nombre de Jobs maximum    30  Rso    15 Rsi    15 Mxo    15 Mxi    15 Nmx
Nombre de préstartés .     15           Lig LIGNEXOT       Xot O      Po  1998
Keyring filename . . .  /QIBM/USERDATA/ICSS/CERT/SERVER/DEFAULT.KDB
Keyring password . . .                                    MasterKey used  . 4
Application ID . . . .  *TBT                              Mode asynchrone . F
F1=Hlp F3=Exi F6=Imp F9=Cmd
                                  Copyright Informatique Pour Les Sociétés
```

Le Token USB utilise des données de configuration dépendantes du système d'exploitation sur lequel il est installé.

Exemple de configuration :

```
name = eToken slot = 0
library = c:/WINDOWS/system32/eTPKCS11.dll
```

De plus, l'applet Java doit pouvoir accéder au driver du Token USB.

Pour fournir ces informations de paramétrage indispensables au fonctionnement de l'applet, TBT/400 dispose d'un fichier de configuration éditable depuis le menu suivant :

- Configuration du système,
- Paramétrage des serveurs,
- Paramétrage FTP,SMTP,HTTP,EBICS,AS2,SFTP,
- Paramétrage Ebics,
- F10 (édition du fichier).

Exemple de configuration pour un poste Windows :

```
 Edit File: IPLS510C/IPSEBICONF(IPSEBICONF)
 Record :       1   of       3 by  10                      Column :   13    240 by 126
 Control :

CMD ..
+....2....+....3....+....4....+....5....+....6....+....7....+....8....+....9....+....0
....+....1....+....2.
     ************Beginning of data**************
    DLL: c:/WINDOWS/system32/eTPKCS11.dll :IPS: name = eToken slot = 0 library =
c:/WINDOWS/system32/eTPKCS11.dll
    XXX: XXXXXXXXXXXXXXXXXXXXXXXXXXXX
    YYY: YYYYYYYYYYYYYYYYYYYYYYYYYYYY
```

Avec :

- DLL : chemin d'accès au driver tel que défini sur le poste client,
- IPS : données de configuration nécessaires à l'accès au Token USB.

Une ligne correspond à une configuration. Il est possible d'ajouter autant de ligne que nécessaire pour tenir compte des besoins réels.

5.3 L'applet Java IPSEBICSTS

L'applet IPSEBICSTS est accessible depuis un navigateur web (Internet Explorer ou Mozilla Firefox) en se connectant au site suivant :

- https://monserveuribm:10091/ebicsts/IpsEbicsTS ou « monserveuribm » représente le nom d'hôte ou l'adresse IP du serveur exécutant TBT/400.

5.3.1 Sécurité

L'applet IPSEBICSTS permet à un utilisateur de signer des fichiers EBICS TS grâce à l'utilisation de certificats électronique stockés sur un Token USB.

Ce dernier est protégé par un mot de passe et doit être inséré dans un des ports USB prévus à cet effet.

Le mot de passe d'accès ainsi que le Token USB en lui-même doivent être protégés avec le plus grand soin : l'utilisateur est en effet responsable de flux EBICS TS signés par ce certificat, la signature électronique est dite « non répudiable ».

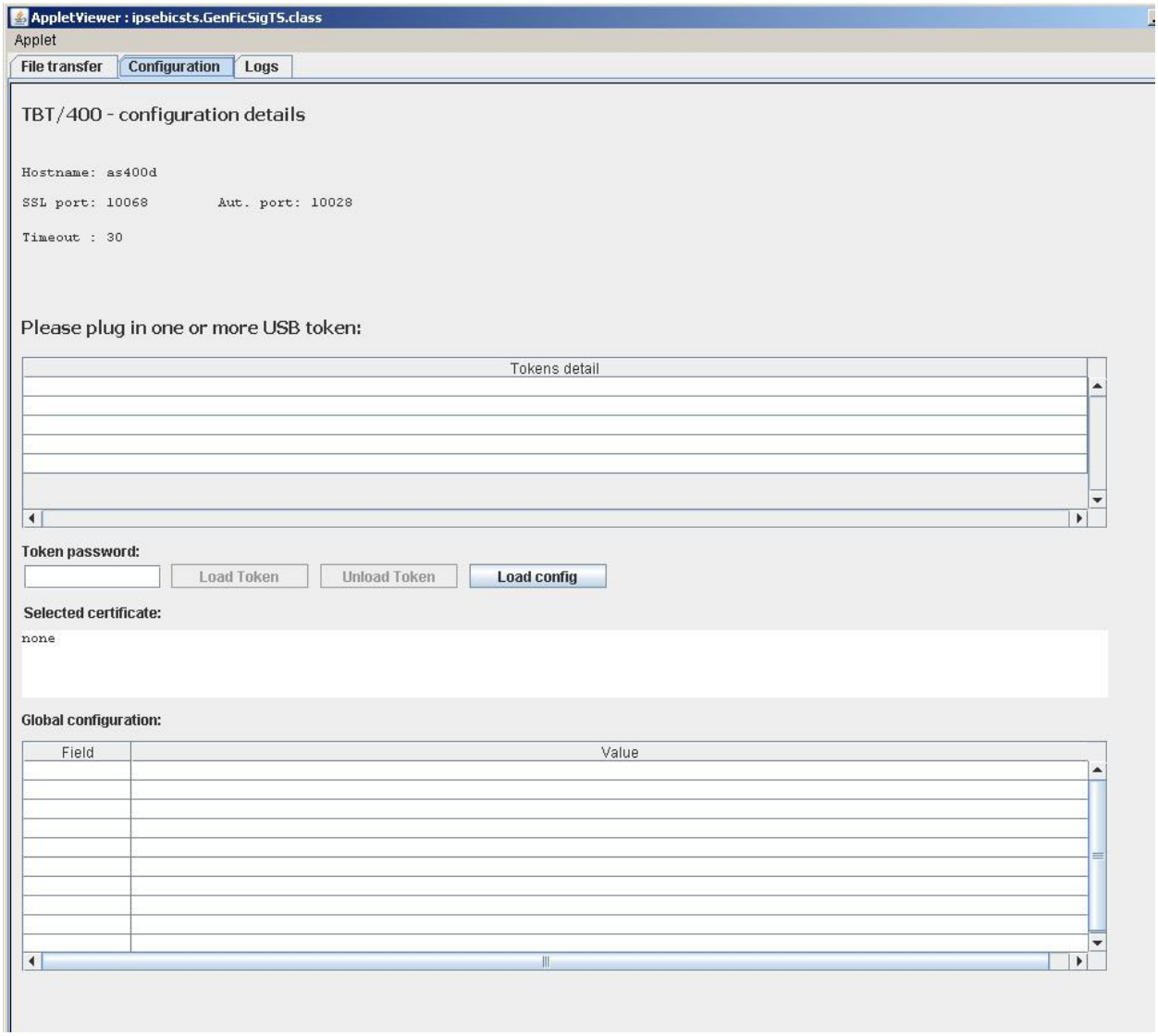
L'utilisateur doit donc veiller à ne jamais laisser son Token USB sans surveillance et à ne jamais divulguer

son mot de passe d'accès.

Par mesure de sécurité, l'applet IPSEBICSTS décharge automatiquement le Token USB après un temps correspondant au paramètre « Timeout » (voir ci-après).

5.3.2 Menu « Configuration »

Cet écran permet de retrouver la configuration générale de l'applet et de charger le Token USB avec lequel l'utilisateur souhaite travailler.



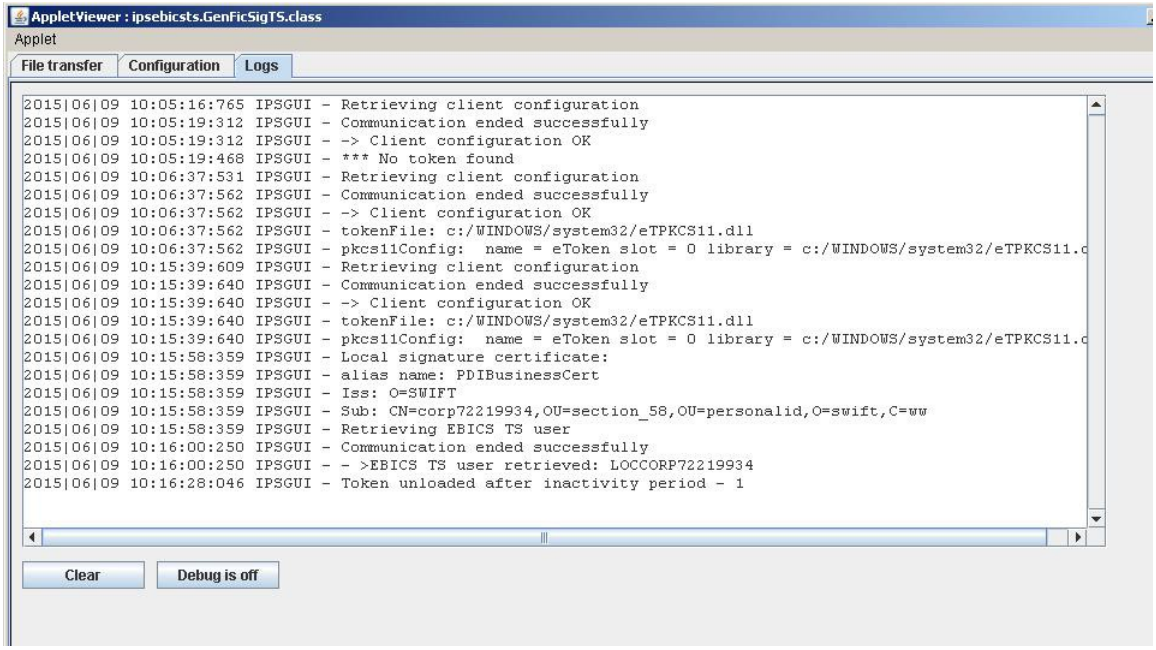
Description de l'écran :

- En haut à gauche sont affichées les informations relatives à la connexion avec TBT/400 :
  - Hostname : nom d'hôte du serveur hébergeant l'applet,
  - SSL port : Port de connexion en mode SSL non authentifié,
  - Aut port : Port de connexion en mode SSL authentifié,
  - Timeout : Timeout de la connexion TCP/IP ainsi que de l'applet elle même,
- Token detai : liste les Tokens USB connectés à un des ports USB,
- Token password : mot de passe d'accès au Token USB,
- Load Token : permet de charger le Token USB en fonction du mot de passe saisi (mot de passe vide interdit),
- Unload Token : permet de décharger le Token USB lorsque l'utilisateur à terminer d'effectuer ses transfers,
- Load config : permet de charger la configuration de l'applet depuis TBT/400,
- Selected certificate : permet de visualiser les informations contenues dans le certificat stocké dans le Token USB (une fois chargé),
- Global configuration : permet de visualiser la configuration de l'applet retournée par TBT/400.

Attention : le Token USB est traditionnellement protégé par son fournisseur contre les erreurs de mot de passe. Trois mots de passe erronés suffisent la plupart du temps à le rendre inutilisable. L'utilisateur est alors obligé de révoquer le certificat stocké avant de le recréer. **En protocole EBICS cela impose de passer par une étape de réinitialisation dans toutes les banques concernées par le USERID** (envoie des requêtes INI, HIA puis des lettres d'initialisation signées).

5.3.3 Menu « Logs »

Cet écran permet de visualiser la log de l'applet.

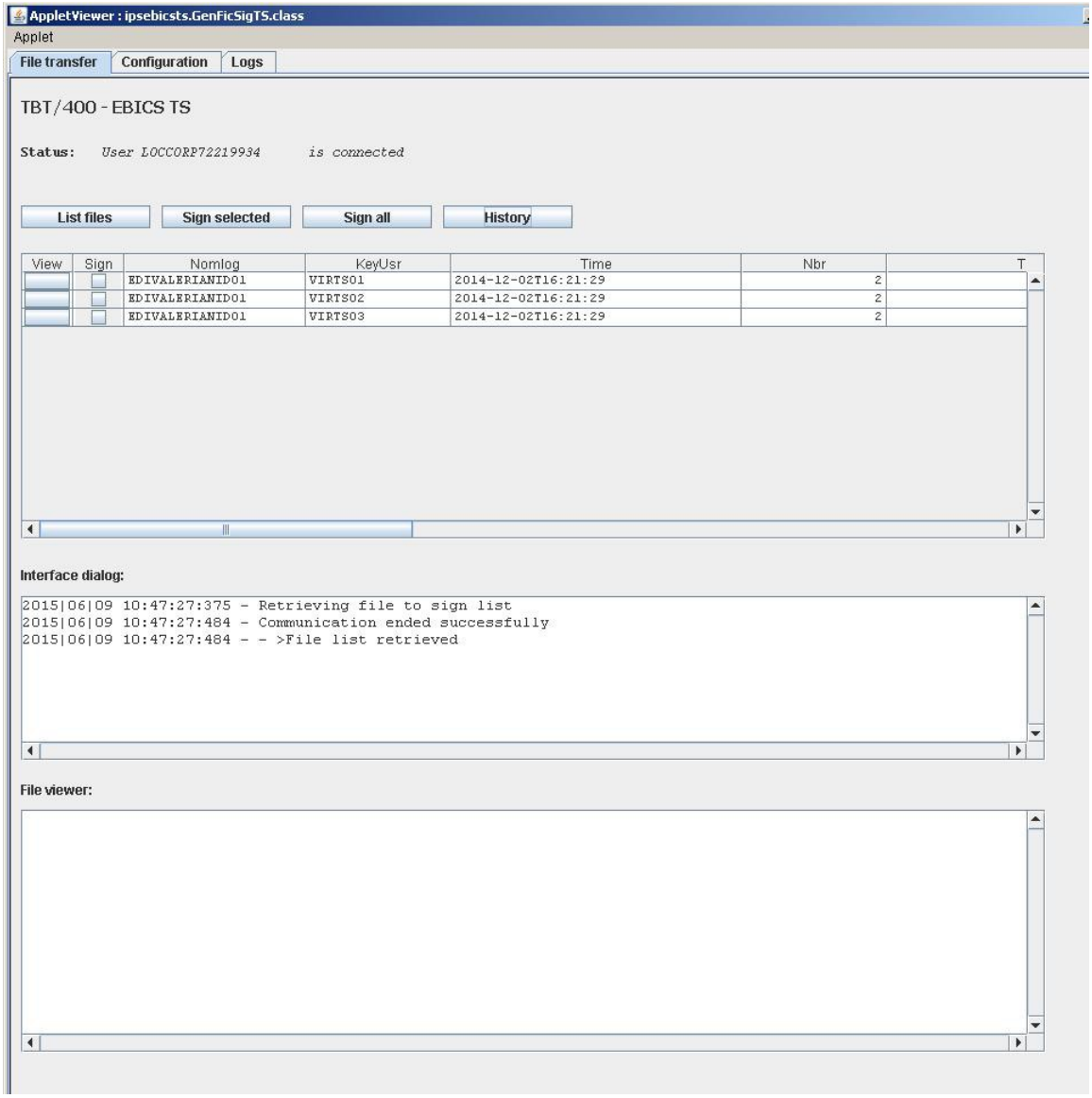


Description de l'écran :

- Le bouton « Clear » permet de vider la fenêtre de log,
- Le bouton « Degug is off » permet d'activer ou de désactiver le mode debug.

5.3.4 Menu « File transfer »

C'est l'écran principal de l'applet, il permet d'effectuer les signatures EBICS TS des fichiers en attentes dans TBT/400.



Description de l'écran :

- Status : représente le statut de connexion de l'applet :
  - « Connected », le nom du correspondant local EBICS de TBT/400 est précisé
  - « Disonnected »,
- List files : permet de lister les fichiers en attentes de signature EBICS TS dans TBT/400. Seuls les fichiers à signés par l'utilisateur connecté sont affichés,
- Sign selected : permet de signer le fichier sélectionné par la case à cochée correspondante. La sélection de plusieurs fichiers est possible en maintenant le bouton « Control » pendant la sélection,
- Sign all : permet de signer tous les fichiers listés,
- History : permet de lister les fichiers déjà traités par l'utilisateur connecté. Cette liste est liée à l'historique de TBT/400, un message supprimé dans TBT/400 ne sera plus disponible dans l'applet,
- Dans la liste des fichiers à signer :
  - View : permet de visualiser le fichier correspondant,
  - Sign : permet de sélectionner le fichier correspondant,
  - Les champs Nomlog, KeyTbt,RefMsg, ComUsr, Author, Object,Attent correspondent aux champs de même nom du fichier (à disposition de l'utilisateur),
  - Time : Date de création (extraite du fichier à signer si disponible),
  - Nbr : Nombre de transactions (extrait du fichier à signer si disponible),
  - Total : Somme totale des transactions (extrait du fichier à signer si disponible),
- Interface dialog : Information de diagnostique relative aux dernière actions sur l'interface,
- File viewer : permet d'afficher les informations contenues dans le fichier en cours de visualisation (limité à 1Mo de données).

Il est à noter que l'applet IPSEBICSTS n'est pas logiciel de gestion : les champs Time, Nbr et Total sont extraits du fichier à émettre lorsque cela est possible.

L'intérêt d'afficher ces informations est de permettre à l'utilisateur d'identifier rapidement un ou plusieurs fichiers à signer.

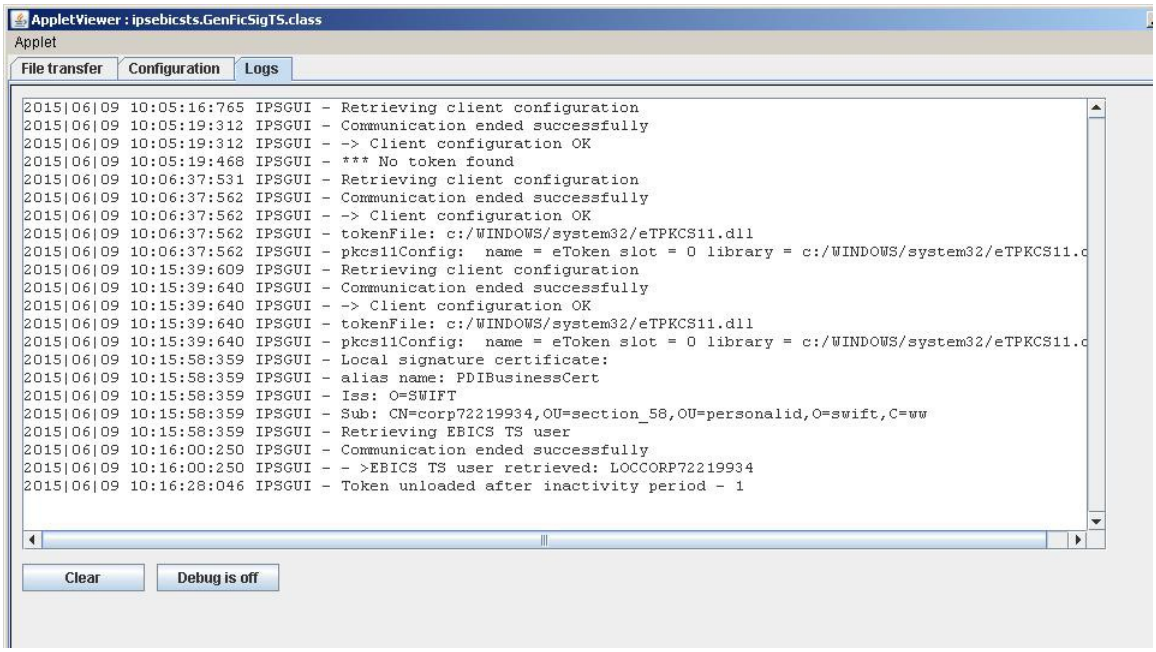
Il est conseillé de se référer à l'application ayant crée le fichier à signer pour tout ce qui concerne les informations métier (l'applet IPSEBICSTS permet cependant de visualiser le contenu des fichiers).

Les champs KeyTbt,RefMsg, ComUsr, Author, Object et Attent sont à la disposition de l'utilisateur : ils pourront être utiliser pour faire circuler une ou plusieurs références métier connues, par exemple, de l'application émettrice.

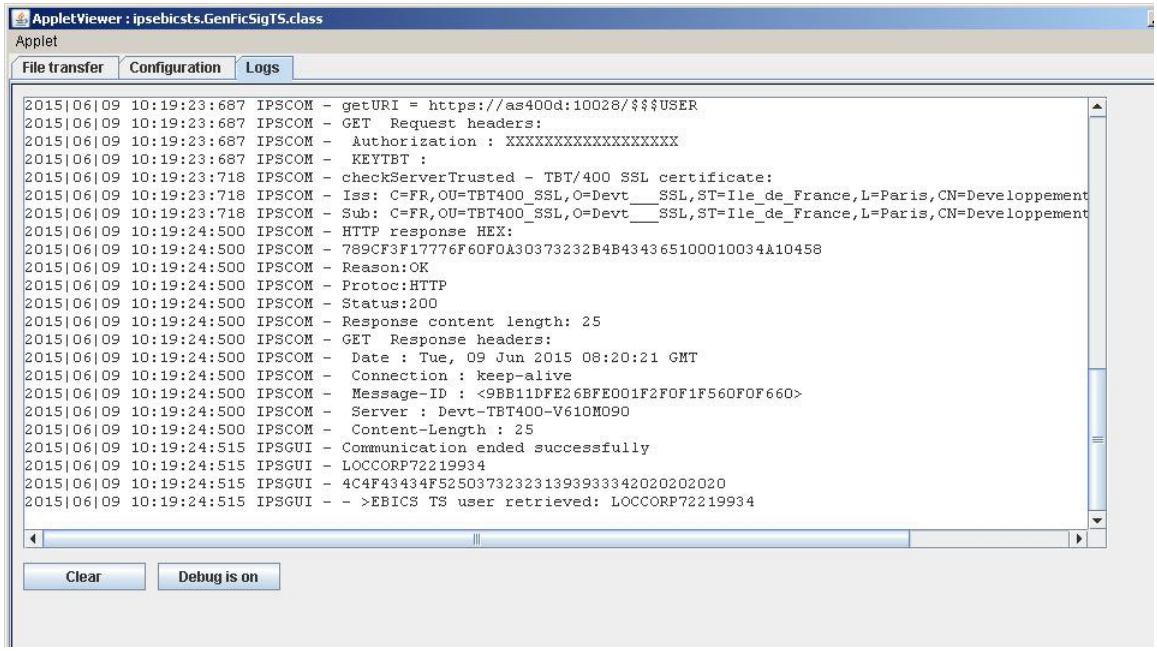
5.3.5 Mode DEBUG

Le menu « Logs » permet à l'utilisateur d'obtenir des informations relatives à l'interface et à la communication avec TBT/400.

Le mode debug n'est pas activé par défaut, seules les informations de diagnostique et d'erreurs bloquantes sont affichées (préfixe IPSGUI) :



Lorsque le mode debug est activé, les informations détaillées de l'applet et de la communication avec TBT/400 (préfixe IPSCOM) sont également affichées :



6 Envoi d'un fichier

6.1 Utilisation

Il est possible d'envoyer un fichier :

- Par menu,
- Par commande.

En EBICS, un envoi de fichier s'effectue via un ordertype FUL (géré automatiquement par **TBT/400**)

Le mode classique dans **TBT/400** est l'utilisation de la commande d'émission IPSNDEBICS :

Plusieurs champs sont à initialiser pour l'usage de cette commande :

- Les qualifiants du fichier à envoyer
  - OBJLIB,OBJFIL,OBJMBR pour un fichier natif OS/400
  - OBJFIL(*IFS), IFSDIR, IFSOBJ pour un fichier IFS
- La banque cible de l'opération (NOMLOG)
- Le filetype EBICS (= nom de fichier défini par le CFONB)
- La gestion de l'acquittement applicatif (ACKDEM, APPEME)

Le protocole EBICS permet de distinguer le mode TEST du mode PRODUCTION, dans **TBT/400**, il s'agit du champ EBITST :

- 'O' : mode TEST,
- 'N' : mode PRODUCTION.

Exemple d'une émission de fichier EBICS :


```
IPLSP/IPSNDEBICS  NOMLOG(BNP)
                  OBJFIL(FIC) OBJLIB(LIB) OBJMBR(MBR)      +
                  EBIFTY('pain.xxx.cfonb160.dco')                    +
                  APPEME(TEST) ACKDEM('O')                           +
                  KEYUSR('keyusr') COMUSR('comusr')
```

Cette commande peut être utilisée pour émettre un fichier à la banque BNP en utilisant les références métier « BNPSEPA001 » et « Releve_compte_BNP001 ».

6.2 Émission en profil TS

6.2.1 Vue TBT/400

En profil TS, un fichier émis par commande ou depuis le menu d'émission est automatiquement bloqué et en attente de signature :le champ MSGHLD = « O » .

Le nombre de signatures en attentes est précisé dans le paramétrage du correspondant.

Le fichier ne sera libéré que :

- lorsqu'il sera purgé (automatiquement, selon sa date de péremption ou manuellement),
- lorsque le nombre de signatures attendues correspondra au nombre de signatures reçues, dans ce cas, le fichier sera émis vers la banque.

Exemple :

```
SUM0   0022 Nrcu          Supervision des messages          IPLS08     IPLSD
File d'attente            Bib            Type réseau  .             Type obj   M
Application émet          Fic            Profil groupe             Type msg   M
Application dest          Mbr            Profil User  .             Accusé  .
Date de dépôt . . . . . .           Heure de dépôt  .             Protocol
Clé utilisateur . . . . .                Corresp.  .
F Date et Heure              Adresse réduite              Clé utilisateur  Ak
O  d'insertion               du destinataire
  150608 142913 IE  EBI EDIVALERIANID01 pain.001.001.02.sct VIRTS01
  150608 142919 IE  EBI EDIVALERIANID01 pain.001.001.02.sct VIRTS02
  150608 142923 IE  EBI EDIVALERIANID01 pain.001.001.02.sct VIRTS03
F1=Hlp F2=Trk F3=Exi F6=Imp F7=Avn F8=Apr F9=Cmd F10=Zoo F11=Cur F13=Hau
F14=Acr F15=Spr F16=Ace F17=Spe F18=Tra F19=Gau F20=Dro F21=Dsp F22=Pdm F23=Obj
```



```
ZEBI   9933 Devt          Détail d'un message EBICS          IPLS08     IPLSD
Fa M$EXTERNB Em $$$$$TBT De $EXTERNB Bi /IfsTBTIPZ Fi Ou20150609 Mb YXIHSI35QJ
Annu $$$$EBICS  *GLOBAL    EDIVALERIANID01      Rés $$$$EBICS  Util IPLS08
Typ M M Cl TBT 000B3A0E003B553CF3F4F4F7F5F70001 Cl Uti VIRTS01          Ack
--------------------------------------------------------------------------------
Hstid émis  . EBIXQUAL                                              Sign att. 2
Prtid émis  . PARTNERTBT                                            Sign rec. 1
Usrid émis  . USERTBT
Hstid attendu
Prtid attendu
Usrid attendu
Filetype  . . camt.xxx.cfonb000.dri
OrderId . . .      OrderAtt  . . OZHNN                Authentifié .   Distri O
OrderType . . FUL  Nonce . . . . . .
Dat val . . .                                      Timestamp
Ebcdic  . . . N    Code retour technique         Code retour business
Test  . . . . N    Trans Id  . . . . . .
Profil  . . . S    Date Distribution(PSR)
Signature 01  LOCCORP72219934      Usr USERTBTTS01                          Ok O
Signature 02  LOCCORP91935990      Usr USERTBTTS02                          Ok N
F1=Hlp F2=Trk F3=Exi F6=Imp F7=Avn F8=Apr F9=Cmd F10=Txt F11=Edt F13=Hau
F14=Acr F15=Spr F16=Ace F17=Spe F18=Tra F19=Gau F20=Dro F21=Dsp F22=Pdm F23=Obj
```

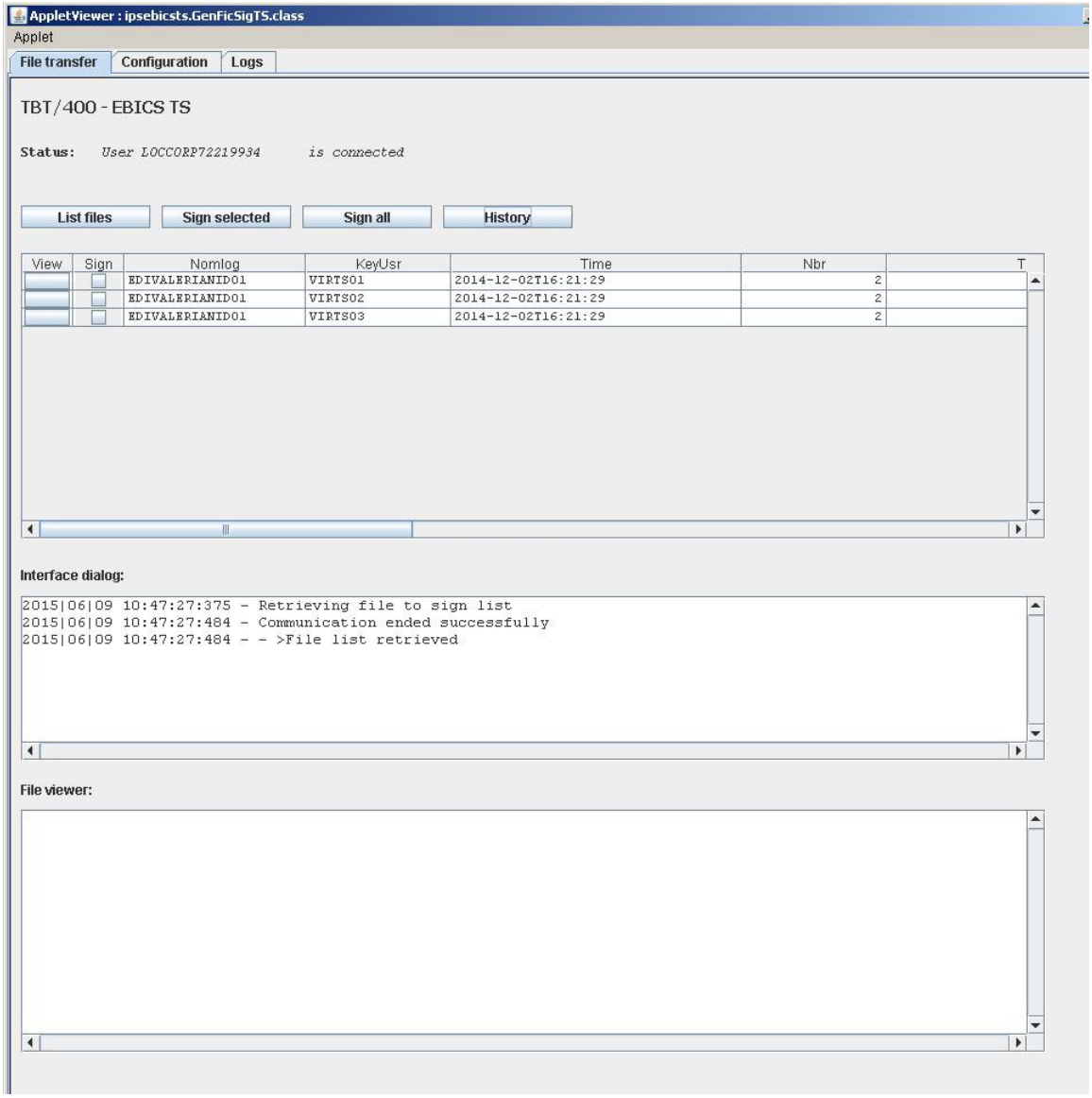
Ici, le fichier est en attente de deux signatures à effectuer depuis l'Applet Java IPSEBICSTS par les correspondants identifier par LOCCORP72219934 et LOCCORP91935990 dans l'annuaire de TBT/400.

Le correspondant LOCCORP72219934 a déjà envoyé sa signature (OK=O). Le fichier ne sera envoyé à la banque que lorsque « Sign rec. » sera égal à « Sign att ».

6.2.2 Vue Applet IPSEBICSTS

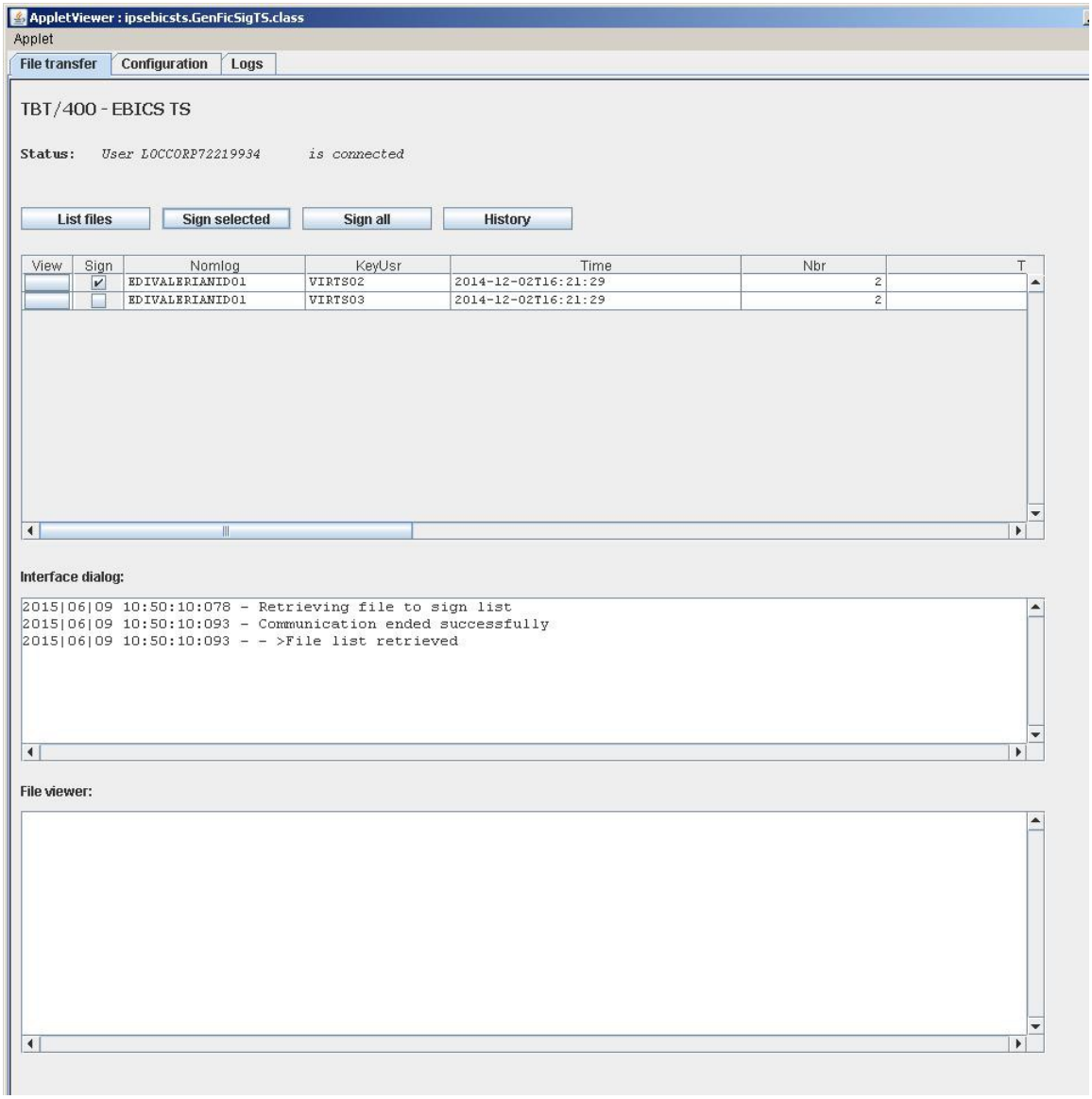
Pour signer un fichier :

- Sélectionner le fichier en cochant la case correspondante,
- Cliquer sur « Sign selected ».

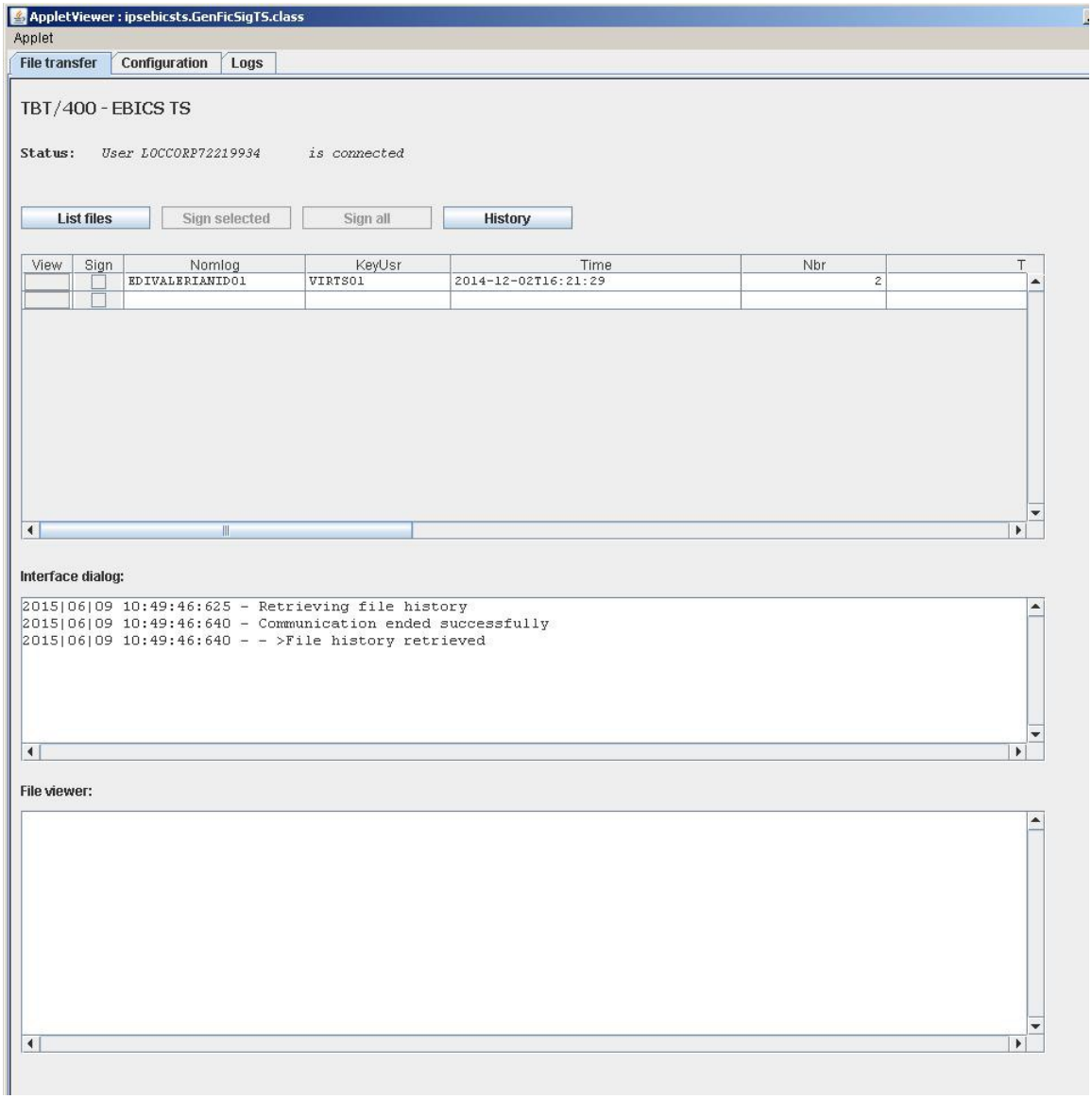


Les fichiers à signer sont identifiés par les champs « Nomlog », « KeyUsr », « Time », etc.

Après signature, la liste est rechargée :



Le fichier signé est visible dans l'historique :



6.3 Gestion du PSR

6.3.1 Définition

Le PSR (**P**ayement **S**tatus **R**eport) permet de s'assurer du bon déroulement d'une émission EBICS.

Il s'agit d'un fichier XML contenant, entre autre, la référence du message émis et un code retour.

Le PSR est obligatoirement mis à disposition par le serveur bancaire en cas d'erreur mais toutes les banques n'ont pas choisi de l'implémenter de façon informatique.

Certaines d'entre elles vont, par exemple, se contenter de l'envoyer par e-mail et dans ce cas de figure **TBT/400** devra être paramétré pour ne pas « chercher » à récupérer le PSR.

6.3.2 AVIDIS='O'

La banque est supposée gérer les **PSR**.

TBT/400, suite à un transfert infructueux, remontera à l'application émettrice un code acquittement 'KO' ,'CR', …, (tout code différent de 'OK' ,' ' (Blanc),'PC').

TBT/400, suite à un transfert réussi, remontera à l'application émettrice, le code acquittement 'PC',

signifiant que le transfert s'est bien passé, et que la banque remontera un statut (PSR) ultérieurement (voir paragraphe traitement des **PSR**)

Lors de la réception du PSR, **TBT/400** remontera à l'application émettrice :

- Soit un code acquittement ' ' (Blanc). Signifiant que la banque a validé (au sens réseau) le transfert ;
- Soit un code acquittement 'ED' (Erreur de distribution) signifiant un problème quant au fichier reçu ; le libellé d'acheminement LIBTBT précise le problème (Erreur de décryptage, Erreur de Signature …par exemple).

A noter que dans ce cas, le programme de traitement d'acquittements peut être appelé deux fois.

6.3.3 AVISDI='N'

La banque est supposée ne pas gérer les **PSR**.

TBT/400, suite à un transfert infructueux, remontera à l'application émettrice le code acquittement en erreur (idem cas précédent AVIDIS='N')

TBT/400, suite à un transfert réussi, remontera directement à l'application émettrice un code acquittement final (' ' Blanc). Si un PSR est reçu ultérieurement, il sera ignoré.

6.4 Champs principaux disponibles dans le programme de traitement d'acquittement

- KEYUSR Clé utilisateur (valorisé par le programme d'émission)
- COMUSR Commentaire (valorisé par le programme d'émission)
- ACKTBT Code acquittement
- LIBTBT Libellé d'acheminement
- EBIFTY Filetype EBICS

7 Réception d'un fichier

7.1 Utilisation

Dans TBT/400, il est possible d'envoyer un fichier :

- Par menu
- Par commande

En EBICS, une réception de fichier s'effectue via un ordertype FDL (géré automatiquement par **TBT/400**)

Le mode classique dans **TBT/400** est l'utilisation de la commande d'émission IPSNDEBICS :

Plusieurs champs sont à initialiser pour l'usage de cette commande :

- Les qualifiants du fichier doivent être alimentés avec *DUMMY (= fichier fictif),
- La banque cible de l'opération (NOMLOG),
- Le filetype EBICS (= nom de fichier défini par le CFONB),
- La gestion de l'acquittement applicatif (ACKDEM, APPEME),
- EBIDAD et EBIDAF (critère de sélection date debut et date de fin),
- EBIDAV permet de fixer des valeurs spéciales pour EBIDAD et EBIDAF.

TBT/400 créera dynamiquement le fichier de réception selon les spécifications de la file d'attente des messages entrants de APPEME **; TBT/400** déposera un acquittement dans la file de traitement des acquittements (si demandé), et un message dans la file d'attente des messages entrants.

```
IPLSP/IPSNDEBICS  NOMLOG(BNP)
                  OBJFIL(*DUMMY) OBJLIB(*DUMMY) OBJMBR(*DUMMY) +
                  EBIFTY('camt.xxx.cfonb120.stm')
                  APPEME(TEST)
                  ACKDEM(O)
```

Dans l'exemple ci-dessus,**TBT/400** déposera un acquittement dans la file d'attente d'acquittement de l'application TEST, et un fichier entrant dans la file d'attente de traitement des fichiers entrants de l'application TEST.(si la récupération s'est réalisée correctement).

7.2 Reprise de la logique ETEBAC3

Pour reprendre une solution **ETEBAC**, le plus simple semble être de confondre file d'attente d'acquittements et de traitement de messages entrants, et de demander un acquittement conditionnel.

Par ailleurs, une possibilité de la commande IPLSP/IPSNDETB3R consistait à préciser le nom de fichier utilisé pour la réception.

Cette possibilité, **pourtant déconseillée dans la plupart des cas**, s'est retrouvée largement utilisée en clientèle la rendant, de ce fait, impossible à supprimer en ETEBAC.

Cette option n'a volontairement pas été implémentée en protocole EBICS.

Pour reprendre à l'identique cette fonctionnalité en EBICS il suffit de copier le fichier reçu par **TBT/400** vers le fichier anciennement passé en paramètre à la commande IPLSP/IPSNDETB3R, par exemple :

7.2.1 Ancienne méthode (réception ETEBAC3)

En une seule phase : Soumission + réception dans le même programme. :

```
DCL        VAR(&DSTLIB)   TYPE(*CHAR) LEN(10)
DCL        VAR(&DSTFIL)   TYPE(*CHAR) LEN(10)
DCL        VAR(&DSTMBR)   TYPE(*CHAR) LEN(10)

CHGVAR     VAR(&DSTLIB) VALUE('BIBUTIL')
CHGVAR     VAR(&DSTFIL) VALUE('FICUTIL')
CHGVAR     VAR(&MBRLIB) VALUE('MBRUTIL')

IPLSP/IPSNDETB3R NOMLOG(BANQ1)
                 OBJFIL(&DSTFIL)
                 OBJLIB(&DSTLIB)
                 OBJMBR(&DSTMBR)
                 CRDETB(TYPERECEPTION)

/*                                                                  */
/*                                                                  */
/* Gestion des erreurs puis                                         */
/* traitement du fichier représenté par &DSTLIB, &DSTFIL et &DSTMBR */
/*                                                                  */
```

Dans ce cas le flux est **directement** reçu dans le fichier représenté par &DSTLIB, &DSTFIL et &DSTMBR, ce qui sous entend que **ce dernier soit entièrement géré par l'utilisateur**, et c'est précisément pour cette raison que cette méthode a toujours été déconseillée.

7.2.2 Réception EBICS

En 2 phases distinctes : Soumission de la requête (PGM n°1) + traitement du fichier reçu (PGM n°2 appelé par une application **TBT/400**).

```
/* Programmes N°1 : Soumission de la réception EBICS */

DCL   ...
DCL         ...

IPLSP/IPSNDEBICS NOMLOG(BANQ1)
                 OBJFIL(*DUMMY)
                 OBJLIB(*DUMMY)
                 EBIFTY(TYPERECEPTION)
/*                                                                  */
/*                                                                  */
/* Gestion des erreurs.                                          */
/*                                                                  */

/* Programmes N°2 : Traitement du fichier reçu */

PGM         PARM(&OBJLIB &OBJFIL &OBJMBR)       /* PGM appelé par TBT/400  */

DCL         VAR(&OBJLIB)   TYPE(*CHAR) LEN(10)  /* Fichier TBT/400         */
DCL         VAR(&OBJFIL)   TYPE(*CHAR) LEN(10)
DCL         VAR(&OBJMBR)   TYPE(*CHAR) LEN(10)
DCL         VAR(&DSTLIB)   TYPE(*CHAR) LEN(10)  /* Fichier utilisateur     */
DCL         VAR(&DSTFIL)   TYPE(*CHAR) LEN(10)
DCL         VAR(&DSTMBR)   TYPE(*CHAR) LEN(10)

CHGVAR      VAR(&DSTLIB) VALUE('BIBUTIL')
CHGVAR      VAR(&DSTFIL) VALUE('FICUTIL')
CHGVAR      VAR(&DSTMBR) VALUE('MBRUTIL')

CPYF FROMFILE(&OBJLIB/&OBJFIL)
     TOFILE(&DSTLIB/&DSTFIL)
     FROMMBR(&OBJMBR)
     TOMBR(&DSTMBR)


/*                                                                  */
/*                                                                  */
/* Gestion des erreurs puis                                         */
/* traitement du fichier représenté par & DSTLIB, &DSTFIL et &DSTMBR */
/*                                                                  */
```

Dans ce cas le flux est reçu dans un fichier entièrement géré par TBT/400 puis copié vers le fichier utilisateur représenté par &DSTLIB et &DSTFIL, ici BIBUTIL/FICUTIL(MBRUTIL).

Cette méthode de travail, certes plus contraignante, possède plusieurs avantages :

- Séparation (isolation) de la partie **TBT/400** (réception du flux réseau) de la partie utilisateur (traitement du fichier passé en paramètre),
- Le fichier reçu par **TBT/400** est entièrement géré par ce dernier,
- Cette méthode est LA méthode standard de **TBT/400** et est, de ce fait, entièrement compatible avec les évolutions futures de ce dernier.

7.3 Critères de sélection EBICS

Les champs EBIDAD et EBIDAF permettent de sélectionner une date de début et de fin mais certaines implémentations éxigent d'avoir ces deux champs à blanc pour certaines opérations, par exemple :

- Sélection du dernier relevé de compte disponible (pas encore récupéré aujourd'hui) :
  - EBIDAV : *SPACE,
  - EBIDAD et EBIDAF à blanc (automatiquement).
- Sélection du relevé de compte de la semaine dernière (déjà récupéré mais la comptabilité exige de le re-télécharger) :
  - EBIDAD : 201104120000,
  - EBIDAF : 201104140000,
  - EBIDAV non renseigné.

La valeur spéciale *SPACE du champ EBIDAV « blanchi » les champs EBIDAD et EBIDAF.

Les quatres derniers chiffres correspondent au décallage horaire et ne sont pas à confondre avec l'heure.

7.4 Champs principaux disponibles dans le programme de traitement du fichier reçu

- KEYUSR Clé utilisateur (valorisé par le programme ayant réalisé la demande)
- COMUSR Commentaire (valorisé par le programme ayant réalisé la demande)
- EBIFTY Filetype EBICS
- OBJLIB Qualifiants du fichier si réception dans fichier OS/400
- OBJFIL
- OBJMBR
- IFSOBJ Nom d'Objet si réception IFS

8 Renouvellement des certificats

Attention : La sauvegarde des certificats actuels avant leur renouvellement est impérative : tant que cette opération n'aura pas été entièrement effectuée, ces derniers seront toujours référencés sur chacun des serveurs bancaires ayant reçus les lettres d'initialisation originales.

Attention : les transferts EBICS doivent impérativement être bloqués pendant cette phase au risque d'être rejetés par la banque.

8.1 Méthode semi-automatique

Cette méthode à l'avantage de pouvoir être effectuée sans avoir à renvoyer les lettres d'initialisation.

Elle demande, en revanche, une manipulation des fichiers IFS qu'il convient de réaliser avec la plus grande précaution :

- **Sauvegarder impérativement le répertoire IFS contenant les certificats de TBT/400,**
- Créer les trois nouveaux certificats EBICS en utilisant une des commandes suivantes
  - jusqu'à la version 89 inclue de TBT400 :

```
IPLSP/IPSCRTCERT  CERUSAG(*EB)
                  CERNAME(CERT_ACTUEL)
                  CERSUFF(*NEW)
```

  - A partir de la version 90 de TBT400 :

```
IPLSP/IPSCRTEBIC  CERNAME(CERT_ACTUEL)
                  CERSUFF(*NEW)
```

- Une fois les trois certificats crées, **envoyer une requête PUB vers chacune des banques concernées par ce certificat de signature à mettre à jour :** les certificats actuels seront utilisés pour envoyer le nouveau certificat de signature.
- Renommer manuellement le certificat de signature :
  - CERT_ACTUEL_A_APP.P12 devient CERT_ACTUEL_OLD_A_APP.P12,
  - CERT_ACTUEL_A_APP_NEW.P12 devient CERT_ACTUEL_A_APP.P12.
- Arrêter puis relancer le sous-système de tbt400 pour forcer le rafraîchissement du cache
- **Envoyer une requête HCA vers chacune des banques concernées par ces certificats d'authentification et de cryptage à mettre à jour** :
  - Le *certificat d'authentification actuel* sera utilisé pour envoyer le nouveau certificat d'authentification et de cryptage,
  - Le *nouveau certificat de signature* sera utilisé pour signer le fichier contenant les certificats.
- Renommer manuellement les certificats d'authentification et de cryptage :
  - CERT_ACTUEL_E_APP.P12 devient CERT_ACTUEL_OLD_E_APP.P12,
  - CERT_ACTUEL_E_APP_NEW.P12 devient CERT_ACTUEL_E_APP.P12,
  - CERT_ACTUEL_X_APP.P12 devient CERT_ACTUEL_OLD_X_APP.P12,
  - CERT_ACTUEL_X_APP_NEW.P12 devient CERT_ACTUEL_X_APP.P12.
- Arrêter puis relancer le sous-système de tbt400 pour forcer le rafraîchissement du cache

A partir de ce moment les nouveaux certificats sont en place, il ne reste plus qu'a valider l'opération par quelques tests avant de pouvoir reprendre les transferts EBICS.

8.2 Réinitialisation du USERID

Une autre méthode de renouvellement des certificats EBICS consiste tout simplement à utiliser la méthode décrite au chapitre « 2.3. Création initiale des Certificats ».

Cette méthode comporte deux inconvénients :

- Le USERID correspondant aux certificats à renouveler doit obligatoirement être réinitialiser par chaque banque impactées par le renouvellement,
- Les lettres d'initialisation doivent à nouveaux être éditées, signées puis envoyées à chaque banque impactées par le renouvellement.

9 Commandes spécifiques

9.1 Commande d'envoi

```
                        Emission EBICS (IPSNDEBICS)

Nom logique du correspondant . . NOMLOG
Clé utilisateur  . . . . . . . . KEYUSR         ' '
Objet à traiter: Fichier . . . . OBJFIL
Objet à traiter: Bibliothèque  . OBJLIB         *LIBL
Objet à traiter: Membre  . . . . OBJMBR         *FIRST
Nom du spool à envoyer . . . . . SPLNAM         TBT400
Travail ayant créé le Spool  . . SPLJOB         *
  Utilisateur  . . . . . . . . .
  Numéro . . . . . . . . . . . .
Numéro du spool à envoyer  . . . SPLNUM         0
Suppression du spoolfile . . . . SPLSUP         ' '
IFS - Répertoire . . . . . . . . IFSDIR         *CURDIR
IFS - Objet  . . . . . . . . . . IFSOBJ
Auteur du courrier . . . . . . . AUTHOR
Objet  du courrier . . . . . . . OBJECT
A l'attention de . . . . . . . . ATTENT
Référence du courrier  . . . . . REFMSG
Filetype . . . . . . . . . . . . EBIFTY
Value Date . . . . . . . . . . . EBIDAV
Begin Date . . . . . . . . . . . EBIDAD
End   Date . . . . . . . . . . . EBIDAF
Fonction demandée  . . . . . . . FNCDEM         S
Fonction début     demandée  . . DEBDEM         O
Fonction fin       demandée  . . FINDEM         O
Fonction exception demandée  . . EXCDEM         O
Fonction trace     demandée  . . TRADEM         0
Fichier dupliqué   demandé . . . DUPDEM         ' '
Application émettrice  . . . . . APPEME         $INTERNA
Application destinatrice . . . . APPDES         $EXTERNA
Date  d'envoi différé  . . . . . DATDIF         '        '
Heure d'envoi différé  . . . . . HORDIF         '        '
Date limite d'envoi  . . . . . . DATPER         '        '
Heure limite d'envoi . . . . . . HORPER         '        '
Accusé demandé . . . . . . . . . ACKDEM         ' '
Suppression fichier demandée . . SUPDEM         ' '
Emission mode puit . . . . . . . PUIDEM         ' '
Impression demandée  . . . . . . IMPDEM         ' '
Break message demandé  . . . . . BRKDEM         ' '
Scrutation implicite . . . . . . SCRDEM         ' '
Commentaire utilisateur  . . . . COMUSR
Hauteur de page pour télécopie   HAUPAG         0
Environnement demandé  . . . . . SETENV         *TBT
Code utilisateur souhaité  . . . USRDEM         *CURRENT
Mode synchrone . . . . . . . . . SYNDEM         ' '
Format fichier . . . . . . . . . FMTFIC         *NONE
Taille fichier . . . . . . . . . SIZFIC         0
Ajout de CR/LF . . . . . . . . . CRLDEM         ' '
Ajout de CR/LF fin . . . . . . . CRLFIN         ' '
Suppression blancs . . . . . . . SPADEM         ' '
Traduction ASCII . . . . . . . . ASCDEM         ' '
Ccsid demandé  . . . . . . . . . CCSID          0
```


9.2 Commande de Statut

```
                          Statut EBICS (IPSSTEBICS)

Client Date  . . . . . . . . . . . IPHTDC
Server Date  . . . . . . . . . . . IPHTDS
User agent . . . . . . . . . . . . IPHTUA
Server . . . . . . . . . . . . . . IPHTSR
Local Host . . . . . . . . . . . . EBIHOS
Local Partner  . . . . . . . . . . EBIPAR
Local User . . . . . . . . . . . . EBIUSR
Filetype . . . . . . . . . . . . . EBIFTY
Order Id . . . . . . . . . . . . . EBIORD
Order Type . . . . . . . . . . . . EBIORT
Order Att  . . . . . . . . . . . . EBIORA
Remote Host  . . . . . . . . . . . EBIHOSR
Remote Partner . . . . . . . . . . EBIPARR
Remote User  . . . . . . . . . . . EBIUSRR
Value Date . . . . . . . . . . . . EBIDAV
Begin Date . . . . . . . . . . . . EBIDAD
Fonction fin       demandée  . . FINDEM         N
Fonction exception demandée  . . EXCDEM         O
Fonction trace     demandée  . . TRADEM         0
Code retour (Num. étendu)  . . . RTNCDP
Libellé du compte rendu  . . . . MSGTXT
```

10 Filetypes 'Techniques'

TBT/400 reconnaît un certain nombre de filetypes 'techniques' ; le fichier à envoyer doit être dans ce cas *DUMMY :

- HCA Envoi Idem HIA en renouvellement
- HIA Envoi certificats authentification et cryptage -x -e
- HPB Réception certifs authentification et cryptage -x -e
- HPD Réception paramètres de la banque
- HTD Download Subscriber's Customer and Subscriber
- INI Envoi du certificat de signature -a
- PUB Envoi certificat signature -a (Renouvellement)
- SPR Envoi suspension accès

```
IPLSP/IPSNDEBICS  NOMLOG(BNP)
                  OBJFIL(*DUMMY) OBJLIB(*DUMMY) OBJMBR(*DUMMY)
                  EBIFTY($HIA$)
```

11 Traitement des statuts (PSR)

Le protocole EBICS permet aux banques, lors d'un envoi de fichier (sens client vers banque), d'envoyer un code retour synchrone, suivi d'un code retour asynchrone (par exemple j'ai reçu le fichier -synchrone- , mais je n'ai pas réussi à le décrypter –asynchrone-).
Les codes retour synchrones sont renvoyés lors de la transmission, le code retour asynchrone est en réalité un fichier mis à disposition par la banque et à récupérer. Le filetype en est standard « pain.002.001.02.ack ».

Toutes les banques ne mettent pas à disposition ce fichier. D'où l'usage du paramètre AVIDIS dans l'annuaire :

- AVIDIS='O' La banque 'joue le jeu' ; lorsqu'un fichier est transmis avec succès, le code acquittement **TBT/400** ACKTBT passe à 'PC' (Le seul code qui attende une suite)
- AVIDIS='N' La banque n'envoie pas les 'PSR' ; **TBT/40**0 utilise le code acquittement ' ' (statut final) pour signaler un transfert réussi (faute d'en savoir plus).

Il est possible de récupérer ces statuts en réalisant une récupération des fichiers pain.002.001.02.ack (à noter que le traitement du fichier reçu est alors un traitement imposé **TBT/400**)

De plus **TBT/400**, si le paramètre SCRDEM est à 'O', fait une récupération implicite de statuts sur tout transfert FUL ou FDL ;

12 Exemple d'implémentation de la commande IPSNDEBICS

12.1 Mise en situation

Il s'agit d'implémenter la commande IPSNDEBICS pour réaliser deux types d'opérations :

- Envoi d'un fichier vers le serveur bancaire (Remise de virement),
- Réception d'un fichier depuis le serveur bancaire (Relevé de compte).

Nous faisons ici le choix de séparer les programmes d'émission et de réception mais ce n'est que pour en faciliter les explications.

12.2 Terminologie

Comme à son habitude **TBT/400** utilise les notions d'application, de files d'attente et de programmes de consommation.

- EBICSAPP : Application (au sens **TBT/400**),
- AEBICSAPP : File d'attente des acquittements,
- EBIPCACK : Programme de consommation des acquittements reçus (lie **TBT/400** à EBITRTACK),
- EBITRTACK : Nom du CL de traitements des acquittements reçus (programme utilisateur),
- MEBICSAPP : File d'attente des messages,
- EBIPCMSG : Programme de consommation des messages reçus (lie **TBT/400** à EBITRTMSG),
- EBITRTMSG : Nom du CL de traitements des messages reçus.

12.3 Emission

Déroulement du processus d'émission :

- Phase n°1 : Le programme utilisateur EBISEND appelle **TBT/400**,
  - Soumission de la commande «Remise de virement » **utilisant l'application émettrice « EBICSAPP »**,
  - Traitement des codes retour (de la soumission),
  - Fin du programme EBISEND avec messages d'erreurs si besoin.
- Phase n°2 : **TBT/400** appelle le programme EBITRTACK (traitement acquittements **TBT/400**) pour chaque soumission effectuée,
  - Test des codes retours,
  - Traitement des erreurs éventuelles,
  - Fin du programme EBITRTACK avec messages d'erreurs si besoin.

12.4 Réception

Déroulement du processus de réception :

- Phase n°1 : Le programme utilisateur EBIRECV appelle **TBT/400**
  - Soumission de la commande «Relevé de compte » **utilisant l'application émettrice « EBICSAPP »**,
  - Traitement des codes retour (de la soumission),
  - Fin du programme EBIRECV avec messages d'erreurs si besoin.
- Phase n°2 : **TBT/400** appelle le programme EBITRTACK (traitement acquittements **TBT/400**) pour chaque soumission effectuée
  - Test des codes retours,
  - Traitement des erreurs éventuelles,
  - Fin du programme EBITRTACK avec messages d'erreurs si besoin.
- Phase n°3 : **TBT/400** appelle le programme EBITRTMSG pour chaque fichier entrant
  - Traitement du fichier reçu dans l'applicatif client,
  - Alimentation des codes retour **TBT/400** permettant de mettre à jour l'historique,
  - Fin du programme EBITRTMSG avec messages d'erreurs si besoin.

12.5 Paramétrage des applications et files d'attente

Pour créer l'application EBICSAPP :

- Entrez dans **TBT/400** : IPLSP/IPS,
- «1. Configuration du système »,
- «3. Définition des applications »,
- Positionnez le curseur sur l'application EBICS (créée par **TBT/400**),
- Saisissez « EBICSAPP » puis ENTER, ce qui aura pour effet de créer une nouvelle application sur le modèle de EBICS (cette dernière ne sera pas modifiée).

Les valeurs par défaut sont suffisantes pour notre exemple.

Pour créer les files d'attentes AEBICSAPP et MEBICSAPP :

- Entrez dans **TBT/400** : IPLSP/IPS,
- «1. Configuration du système »,
- «4. Définition des files d'attente »,
- Positionnez le curseur sur la file d'attente MEBICS (créée par **TBT/400**),
- Saisissez « MEBICSAPP » puis ENTER, ce qui aura pour effet de créer une nouvelle file d'attente sur le modèle de MEBICS (cette dernière ne sera pas modifiée),
- Positionnez maintenant le curseur sur la file d'attente MEBICSAPP et modifiez les champs suivants puis « ENTER » :
- Nom du programme : EBIPCMSG (programme de consommation des messages),
- Nom de la biblio pgm : Votre bibliothèque utilisateur.

Idem pour la file d'attente AEBICSAPP en utilisant cette fois le modèle MEBICSAPP et le programme EBIPCACK.

Là aussi, les valeurs par défaut sont suffisantes.

Le paramétrage application + files d'attente est terminé et, dés ce moment, chaque fois que **TBT/400** recevra un message utilisant l'application MEBICSAPP le programme de consommation EBIPCMSG sera appelé avec toutes les variables nécessaire au traitement applicatif.

Il en va de même pour les acquittements.

12.6 Sources des programmes CL

12.6.1 EBISEND

```
/*-----------------------------------------------------------------*/
/*                                                                 */
/*  Deroulement du CL:                                             */
/*  - Appel IPSNDEBICS en mode émission,                           */
/*  - Gestion du code retour de la commande                        */
/*                                                                 */
/*-----------------------------------------------------------------*/

            PGM        PARM(&OBJLIB &OBJFIL &OBJMBR)

            DCL        VAR(&OBJLIB) TYPE(*CHAR) LEN(10)
            DCL        VAR(&OBJFIL) TYPE(*CHAR) LEN(10)
            DCL        VAR(&OBJMBR) TYPE(*CHAR) LEN(10)
            DCL        VAR(&KEYTBT)   TYPE(*CHAR) LEN(16)
            DCL        VAR(&KEYUSR)   TYPE(*CHAR) LEN(16)
            DCL        VAR(&RTNCDP) TYPE(*DEC) LEN(11) VALUE(8) /* +
                         8 = KO (par défaut) */
            DCL        VAR(&MSGTXT)   TYPE(*CHAR) LEN(256)


/*        La commande IPSNDEBICS soumet la demande d'émission   de   */
/*        fichiers et IPSRCVTBT permet d'en contrôler l'exécution.   */
/*        Une autre solution consiste à positionner EXCDEM à OUI     */
/*        (IPSNDEBICS) pour demander la génération d'une exception   */
/*        en cas d'erreur et utiliser un MONMSG IPS0000 (à la place  */
/*        de IPSRCVTBT + test code retour).                          */
/*                                                                   */
/*        FINDEM(N)       : Ne pas clôturer l'environnement          */
/*        EXCDEM(N)       : Pas d'exception en cas d'erreurs         */
/*        APPEME(EBICSAPP): Les accusés et messages reçus seront     */
/*                          envoyés à l'application EBICSAPP         */
/*        ACKDEM(O)       : Surveillance des accusés active          */
/*                          (= appel l'application EBICSAPP à la     */
/*                           reception d'un accusé)                  */
/*                                                                   */
/*        OBJLIB, OBJFIL et OBJMBR sont alimentés par l'applicatif   */
/*        appellant TBT/400.                                         */
/*                                                                   */
/*                                                                   */
            IPLSP/IPSNDEBICS NOMLOG(TSTSVREBICS) OBJFIL(&OBJFIL) +
                        OBJLIB(&OBJLIB) OBJMBR(&OBJMBR) +
                        EBIFTY('camt.xxx.cfonb080.dri') FINDEM(N) +
                        EXCDEM(N) APPEME(EBICSAPP) ACKDEM('O')
            MONMSG     MSGID(CPF0000)

/* Ici  : IPSRCVTBT interroge le statut de la dernière commande afin */
/*        d'en récupérer le code retour &RTNCDP                      */
/*                                                                   */
/*        FNCDEM(L)       : L pour Last (dernière commande émise)    */
/*        DEBDEM(N)       : Ne pas démarrer l'environnement          */
/*                        : (déja fait par IPSNDEBICS)               */
/*        FINDEM(O)       : Clôturer l'environnement                 */
/*                        : (si dernière commande TBT du programme)  */
/*        EXCDEM(N)       : Pas d'exception en cas d'erreurs         */
/*                                                                   */
            IPLSP/IPSRCVTBT FNCDEM(L) DEBDEM(N) FINDEM(O) EXCDEM(N) +
                        RTNCDP(&RTNCDP) KEYTBT(&KEYTBT) +
                        KEYUSR(&KEYUSR) MSGTXT(&MSGTXT)
            MONMSG     MSGID(CPF0000)
```



```
            IF         COND(&RTNCDP *NE 0) THEN(GOTO CMDLBL(ERREUR))


/* Autres traitements... */



/* Pas d'erreur... */
            SNDPGMMSG  MSG('Appel TBT/400 pour émission OK')
            GOTO       CMDLBL(FIN)



 ERREUR:    SNDPGMMSG  MSG('Erreur de traitement...')
            SNDPGMMSG  MSG('MSGTXT: ' *CAT &MSGTXT)
            GOTO       CMDLBL(FIN)


/* Autres traitements d'erreurs... */


 FIN:       ENDPGM
```

12.6.2 EBIRECV

```
/*-------------------------------------------------------------------*/
/*                                                                   */
/*  Deroulement du CL:                                               */
/*  - Appel IPSNDEBICS en mode scrutation (récuperation),            */
/*  - Gestion du code retour de la commande                          */
/*                                                                   */
/*-------------------------------------------------------------------*/

            PGM

            DCL        VAR(&KEYTBT)   TYPE(*CHAR) LEN(16)
            DCL        VAR(&KEYUSR)   TYPE(*CHAR) LEN(16)
            DCL        VAR(&RTNCDP) TYPE(*DEC) LEN(11) VALUE(8) /* +
                         8 = KO (par défaut) */
            DCL        VAR(&MSGTXT)   TYPE(*CHAR) LEN(256)


/*        La commande IPSNDEBICS soumet la demande de réception de   */
/*        fichiers et IPSRCVTBT permet d'en contrôler l'exécution.   */
/*        Une autre solution consiste à positionner EXCDEM à OUI     */
/*        (IPSNDEBICS) pour demander la génération d'une exception   */
/*        en cas d'erreur et utiliser un MONMSG IPS0000 (à la place  */
/*        de IPSRCVTBT + test code retour).                          */
/*                                                                   */
/*        FINDEM(N)       : Ne pas clôturer l'environnement          */
/*        EXCDEM(N)       : Pas d'exception en cas d'erreurs         */
/*        APPEME(EBICSAPP): Les accusés et messages reçus seront     */
/*                          envoyés à l'application EBICSAPP         */
/*        ACKDEM(O)       : Surveillance des accusés active          */
/*                          (= appel l'application EBICSAPP à la     */
/*                           reception d'un accusé)                  */
/*                                                                   */
/*        OBJLIB, OBJFIL et OBJMBR valent *DUMMY dans le cas d'une   */
/*        scrutation (*DUMMY = fichier "fictif").                    */
/*                                                                   */
/*                                                                   */
            IPLSP/IPSNDEBICS NOMLOG(TSTSVREBICS) OBJFIL(*DUMMY) +
                        OBJLIB(*DUMMY) OBJMBR(*DUMMY) +
                        EBIFTY('camt.xxx.cfonb080.dri') +
                        EBIDAD(201104120000) EBIDAF(201104122359) +
                        FINDEM(N) EXCDEM(N) APPEME(EBICSAPP) +
                        ACKDEM('O')
            MONMSG     MSGID(CPF0000)

/* Ici  : IPSRCVTBT interroge le statut de la dernière commande afin */
/*        d'en récupérer le code retour &RTNCDP                      */
/*                                                                   */
/*        FNCDEM(L)       : L pour Last (dernière commande émise)    */
/*        DEBDEM(N)       : Ne pas démarrer l'environnement          */
/*                        : (déja fait par IPSNDEBICS)               */
/*        FINDEM(O)       : Clôturer l'environnement                 */
/*                        : (si dernière commande TBT du programme)  */
/*        EXCDEM(N)       : Pas d'exception en cas d'erreurs         */
/*                                                                   */
            IPLSP/IPSRCVTBT FNCDEM(L) DEBDEM(N) FINDEM(O) EXCDEM(N) +
                        RTNCDP(&RTNCDP) KEYTBT(&KEYTBT) +
```



```
                            KEYUSR(&KEYUSR) MSGTXT(&MSGTXT)
             MONMSG     MSGID(CPF0000)

             IF         COND(&RTNCDP *NE 0) THEN(GOTO CMDLBL(ERREUR))

/* Pas d'erreur... */
             SNDPGMMSG  MSG('Appel TBT/400 pour réception OK')
             GOTO       CMDLBL(FIN)




 ERREUR:     SNDPGMMSG  MSG('Erreur de traitement...')
             SNDPGMMSG  MSG('MSGTXT: ' *CAT &MSGTXT)
             GOTO       CMDLBL(FIN)



/* Autres traitements d'erreurs... */



 FIN:        ENDPGM
```

12.6.3 EBIPCACK

```
/*****************************************************************/
/* Ceci est le source du programme "dummy" de consommation       */
/* d'une file d'attente. Il est destiné à servir de modèle.      */
/* Ne remplacez pas le programme IPSPADUMMY dans la bibliothèque */
/* du progiciel (IPLSP). Une version plus complete est fournie   */
/* sous le nom IPSPADUMMC.                                       */
/*****************************************************************/
/* This is the source of the "IPSPADUMMY" program. It must be    */
/* used as a skeleton program and duplicated in customer library */
/* for modifications.                                            */
/*****************************************************************/
             PGM
             DCL        VAR(&DEBDEM) TYPE(*CHAR) LEN(1) VALUE(O)
             DCL        VAR(&RTNCDP)   TYPE(*DEC) LEN(11)
             DCL        VAR(&KEYTBT)   TYPE(*CHAR) LEN(16)
             DCL        VAR(&KEYUSR)   TYPE(*CHAR) LEN(16)
             DCL        VAR(&SUPDEM)   TYPE(*CHAR) LEN(1)
             DCL        VAR(&COMUSR)   TYPE(*CHAR) LEN(128)
             DCL        VAR(&OBJLIB)   TYPE(*CHAR) LEN(10)
             DCL        VAR(&OBJFIL)   TYPE(*CHAR) LEN(10)
             DCL        VAR(&OBJMBR)   TYPE(*CHAR) LEN(10)
             DCL        VAR(&DATFPC)   TYPE(*CHAR) LEN(8)
             DCL        VAR(&HORFPC)   TYPE(*CHAR) LEN(8)
             DCL        VAR(&DATFTR)   TYPE(*CHAR) LEN(8)
             DCL        VAR(&HORFTR)   TYPE(*CHAR) LEN(8)
             DCL        VAR(&DATRPC)   TYPE(*CHAR) LEN(8)
             DCL        VAR(&HORRPC)   TYPE(*CHAR) LEN(8)
             DCL        VAR(&DATRTR)   TYPE(*CHAR) LEN(8)
             DCL        VAR(&HORRTR)   TYPE(*CHAR) LEN(8)
             DCL        VAR(&ACKTBT)   TYPE(*CHAR) LEN(2)
             DCL        VAR(&LIBTBT)   TYPE(*CHAR) LEN(128)
             DCL        VAR(&NOMLOG)   TYPE(*CHAR) LEN(20)
             DCL        VAR(&KEYEXT)   TYPE(*CHAR) LEN(32)
             DCL        VAR(&USRPRF)   TYPE(*CHAR) LEN(16)


             DCL        VAR(&JOB) TYPE(*CHAR) LEN(10)
             DCL        VAR(&USER) TYPE(*CHAR) LEN(10)
             DCL        VAR(&NBR) TYPE(*CHAR) LEN(6)
             DCL        VAR(&MSGCMD) TYPE(*CHAR) LEN(64)
             DCL        VAR(&MSGACK) TYPE(*CHAR) LEN(256)

             MONMSG     MSGID(CPF0000) EXEC(GOTO CMDLBL(CPF0000))

             RTVJOBA    JOB(&JOB) USER(&USER) NBR(&NBR)
             CHGVAR     VAR(&MSGCMD) VALUE('WRKJOB JOB(' *TCAT &NBR +
                          *TCAT '/' *TCAT &USER *TCAT '/' *TCAT +
                          &JOB *TCAT ')')

ITER:
/*************************************************************/
/* APPEL DE LA COMMANDE DE RECEPTION                         */
/*************************************************************/
/* CALL RECEIVE COMMAND                                      */
/*************************************************************/
             IPSRCVTBT  FNCDEM(R) DEBDEM(&DEBDEM) FINDEM(C) +
```

```
                          EXCDEM(N) TRADEM(0) RTNCDP(&RTNCDP) +
                          KEYTBT(&KEYTBT) KEYUSR(&KEYUSR) +
                          ACKTBT(&ACKTBT) LIBTBT(&LIBTBT) +
                          OBJLIB(&OBJLIB) OBJFIL(&OBJFIL) +
                          OBJMBR(&OBJMBR) USRPRF(&USRPRF) +
                          DATFPC(&DATFPC) HORFPC(&HORFPC) +
                          DATFTR(&DATFTR) HORFTR(&HORFTR) +
                          DATRPC(&DATRPC) HORRPC(&HORRPC) +
                          DATRTR(&DATRTR) HORRTR(&HORRTR) +
                          SUPDEM(&SUPDEM) COMUSR(&COMUSR) +
                          NOMLOG(&NOMLOG) KEYEXT(&KEYEXT) /* Appel +
                          des API de TBT/400 via la Command +
                          IPSRCVTBT */

             IF           COND(&RTNCDP *NE 0) THEN(GOTO +
                            CMDLBL(ENDPGM)) /* Plus rien dans la file +
                            d'attente */

             CHGVAR       VAR(&DEBDEM)   VALUE('N')

             SNDPGMMSG    MSG('KEYTBT=' *CAT &KEYTBT)
             SNDPGMMSG    MSG('KEYUSR=' *CAT &KEYUSR)
             SNDPGMMSG    MSG('DATFPC=' *CAT &DATFPC)
             SNDPGMMSG    MSG('HORFPC=' *CAT &HORFPC)
             SNDPGMMSG    MSG('DATFTR=' *CAT &DATFTR)
             SNDPGMMSG    MSG('HORFTR=' *CAT &HORFTR)
             SNDPGMMSG    MSG('DATRPC=' *CAT &DATRPC)
             SNDPGMMSG    MSG('HORRPC=' *CAT &HORRPC)
             SNDPGMMSG    MSG('DATRTR=' *CAT &DATRTR)
             SNDPGMMSG    MSG('HORRTR=' *CAT &HORRTR)
             SNDPGMMSG    MSG('SUPDEM=' *CAT &SUPDEM)
             SNDPGMMSG    MSG('COMUSR=' *CAT &COMUSR)
             SNDPGMMSG    MSG('ACKTBT=' *CAT &ACKTBT)
             SNDPGMMSG    MSG('LIBTBT=' *CAT &LIBTBT)
             SNDPGMMSG    MSG('OBJLIB=' *CAT &OBJLIB)
             SNDPGMMSG    MSG('OBJFIL=' *CAT &OBJFIL)
             SNDPGMMSG    MSG('OBJMBR=' *CAT &OBJMBR)
             SNDPGMMSG    MSG('USRPRF=' *CAT &USRPRF)
             SNDPGMMSG    MSG('NOMLOG=' *CAT &NOMLOG)
             SNDPGMMSG    MSG('KEYEXT=' *CAT &KEYEXT)


/*************************************************************/
/* INSERER L'APPEL DE VOS TRAITEMENTS ICI                    */
/* Brancher obligatoirement en MESOK si OK                   */
/* Brancher obligatoirement en MESKO si erreur               */
/* Brancher obligatoirement en MESPC si statut inconnu       */
/*                                                           */
/* R E M A R Q U E :  Ce programme de consommation est une   */
/* maquette commune pour le traitement :                     */
/*   - des fichiers en entrée                                */
/*   - des acquittements de transmision reçus.               */
/* Cependant, DANS LE CAS DES ACQUITTEMENTS, il n'est pas    */
/* nécessaire de brancher la suite du traitement sur les     */
/* étiquettes MESOK et MESKO car la valorisation des champs  */
/* KEYUSR, ACKTBT, LIBTBT est sans conséquence sur le menu   */
/* "Supervision de l'historique".                            */
/*                                                           */
/* CALL USERBIB(USERPGM)                                     */
```

```
/* MONMSG     MSGID(CPF0000) EXEC(GOTO CMDLBL(MESKO)          */
/*                                                            */
/************************************************************/

/************************************************************/
/* INSERT APPLICATION PROCESS HERE                          */
/* Mandatory GOTO label        MESOK si OK                  */
/* Mandatory GOTO label        MESKO si Error               */
/* Mandatory GOTO label        MESPS si Unknown state       */
/*                                                          */
/* CALL YOURLIB(YOURPROGRAM)                                */
/* MONMSG     MSGID(CPF0000) EXEC(GOTO CMDLBL(MESKO)          */
/************************************************************/
             CALL       PGM(BM/EBITRTACK) PARM(&ACKTBT &LIBTBT &NOMLOG)
             MONMSG     MSGID(CPF0000) EXEC(GOTO CMDLBL(MESKO))

 MESOK:      CHGVAR     VAR(&KEYUSR) VALUE('Userkey')
             CHGVAR     VAR(&COMUSR) VALUE('Commentaire envoyé par +
                          le programme d''application')
             CHGVAR     VAR(&ACKTBT) VALUE('OK')
             CHGVAR     VAR(&LIBTBT) VALUE('Message consommé avec +
                          succès')
/*           CHGVAR     VAR(&SUPDEM) VALUE('N')    Override valeur +
                          initiale */
             GOTO       CMDLBL(MESFIN)

 MESPC:      CHGVAR     VAR(&KEYUSR) VALUE('Userkey')
             CHGVAR     VAR(&COMUSR) VALUE('Commentaire envoyé par +
                          le programme d''application')
             CHGVAR     VAR(&ACKTBT) VALUE('PC')
             CHGVAR     VAR(&LIBTBT) VALUE('Message pris en compte')
/*           CHGVAR     VAR(&SUPDEM) VALUE('N')    Override valeur +
                          initiale */
             GOTO       CMDLBL(MESFIN)

 MESKO:      CHGVAR     VAR(&KEYUSR) VALUE('Userkey')
             CHGVAR     VAR(&COMUSR) VALUE('Commentaire envoye par +
                          le programme d''application')
             CHGVAR     VAR(&ACKTBT) VALUE('KO')
             CHGVAR     VAR(&LIBTBT) VALUE('Message en erreur')
/*           CHGVAR     VAR(&SUPDEM) VALUE('N')    Override valeur +
                          initiale */
             GOTO       CMDLBL(MESFIN)

 MESFIN:     CHGVAR     VAR(&MSGACK) VALUE('TBT/400 - Interface +
                          applicative - Code retour=' *CAT &ACKTBT +
                          *CAT ' :' *BCAT &LIBTBT *BCAT '- Pour +
                          visualiser le job utiliser la commande : +
                          ' *BCAT &MSGCMD)
             SNDMSG     MSG(&MSGACK) TOUSR(&USRPRF)
             MONMSG     MSGID(CPF0000)
             SNDMSG     MSG(&MSGACK) TOUSR(*SYSOPR)
             MONMSG     MSGID(CPF0000)
             IPSRCVTBT  FNCDEM(P) DEBDEM(N) FINDEM(C) EXCDEM(O) +
                          TRADEM(0) RTNCDP(&RTNCDP) KEYTBT(&KEYTBT) +
                          KEYUSR(&KEYUSR) ACKTBT(&ACKTBT) +
                          LIBTBT(&LIBTBT) SUPDEM(&SUPDEM) +
                          COMUSR(&COMUSR)
             GOTO       CMDLBL(ITER)
```



```
/*************************************************************/
/* INCIDENT HORS ITERATION                                   */
/*************************************************************/
/* ERROR    OUT OF LOOP                                      */
/*************************************************************/
 CPF0000:    CHGVAR     VAR(&ACKTBT) VALUE('AB')
             CHGVAR     VAR(&LIBTBT) VALUE('Exception rencontrée +
                          dans le programme')
             CHGVAR     VAR(&MSGACK) VALUE('TBT/400 - Interface +
                          applicative - Code retour=' *CAT &ACKTBT +
                          *CAT ' :' *BCAT &LIBTBT *BCAT '- Pour +
                          visualiser le job utiliser la commande : +
                          ' *BCAT &MSGCMD)
             SNDMSG     MSG(&MSGACK) TOUSR(&USRPRF)
             MONMSG     MSGID(CPF0000)
             SNDMSG     MSG(&MSGACK) TOUSR(*SYSOPR)
             MONMSG     MSGID(CPF0000)
             SNDPGMMSG  MSGID(CPF9898) MSGF(QSYS/QCPFMSG) +
                          MSGDTA(&MSGACK) MSGTYPE(*ESCAPE)
             MONMSG     MSGID(CPF0000)
ENDPGM:      ENDPGM
```

12.6.4 EBITRTACK

```
/*-------------------------------------------------------------------*/
/*           TRAITEMENT DES ACQUITTEMENTS EBICS                      */
/*-------------------------------------------------------------------*/

             PGM        PARM(&ACKTBT &LIBTBT &NOMLOG)

             DCL        VAR(&ACKTBT) TYPE(*CHAR) LEN(2)
             DCL        VAR(&LIBTBT) TYPE(*CHAR) LEN(128)
             DCL        VAR(&NOMLOG)   TYPE(*CHAR) LEN(20)


/* ----------------------------------------------------------------- */
/* Valeur du champ ACKTBT considérée comme positive:                 */
/* - &ACKTBT = '  ' => OK             (OK et PSR reçu ou non demandé) */
/* - &ACKTBT = 'PC' => Pris en compte (OK mais en attente du PSR)     */
/* - &ACKTBT = 'BV' => Boite vide     (OK mais rien à recevoir)       */
/* ----------------------------------------------------------------- */

             IF         COND(&ACKTBT *EQ '  ') THEN(GOTO CMDLBL(FINOK))
             IF         COND(&ACKTBT *EQ 'PC') THEN(GOTO CMDLBL(FINOK))
             IF         COND(&ACKTBT *EQ 'BV') THEN(GOTO CMDLBL(FINOK))



/* Autres tests... */



/* Sinon: Traitement en erreur... */
             GOTO       CMDLBL(FINKO)



 FINOK:      SNDPGMMSG  MSG('Traitement EBICS OK') TOUSR(QSYSOPR)
             SNDPGMMSG  MSG('NOMLOG: ' *CAT &NOMLOG) TOUSR(QSYSOPR)

             GOTO       CMDLBL(FIN)

 FINKO:      SNDPGMMSG  MSG('Traitement EBICS en erreur...') TOUSR(QSYSOPR)
             SNDPGMMSG  MSG('NOMLOG: ' *CAT &NOMLOG) TOUSR(QSYSOPR)
             SNDPGMMSG  MSG('ACKTBT: ' *CAT &ACKTBT) TOUSR(QSYSOPR)
             SNDPGMMSG  MSG('LIBTBT: ' *CAT &LIBTBT) TOUSR(QSYSOPR)

             GOTO       CMDLBL(FIN)

/* Autres traitements d'erreurs... */

 FIN:        ENDPGM
```


12.6.5 EBIPCMSG

```
/*****************************************************************/
/* Ceci est le source du programme "dummy" de consommation       */
/* d'une file d'attente. Il est destiné à servir de modèle.      */
/* Ne remplacez pas le programme IPSPADUMMY dans la bibliothèque */
/* du progiciel (IPLSP). Une version plus complete est fournie   */
/* sous le nom IPSPADUMMC.                                       */
/*****************************************************************/
/* This is the source of the "IPSPADUMMY" program. It must be    */
/* used as a skeleton program and duplicated in customer library */
/* for modifications.                                            */
/*****************************************************************/
            PGM
            DCL        VAR(&DEBDEM) TYPE(*CHAR) LEN(1) VALUE(O)
            DCL        VAR(&RTNCDP)   TYPE(*DEC) LEN(11)
            DCL        VAR(&KEYTBT)   TYPE(*CHAR) LEN(16)
            DCL        VAR(&KEYUSR)   TYPE(*CHAR) LEN(16)
            DCL        VAR(&SUPDEM)   TYPE(*CHAR) LEN(1)
            DCL        VAR(&COMUSR)   TYPE(*CHAR) LEN(128)
            DCL        VAR(&OBJLIB)   TYPE(*CHAR) LEN(10)
            DCL        VAR(&OBJFIL)   TYPE(*CHAR) LEN(10)
            DCL        VAR(&OBJMBR)   TYPE(*CHAR) LEN(10)
            DCL        VAR(&DATFPC)   TYPE(*CHAR) LEN(8)
            DCL        VAR(&HORFPC)   TYPE(*CHAR) LEN(8)
            DCL        VAR(&DATFTR)   TYPE(*CHAR) LEN(8)
            DCL        VAR(&HORFTR)   TYPE(*CHAR) LEN(8)
            DCL        VAR(&DATRPC)   TYPE(*CHAR) LEN(8)
            DCL        VAR(&HORRPC)   TYPE(*CHAR) LEN(8)
            DCL        VAR(&DATRTR)   TYPE(*CHAR) LEN(8)
            DCL        VAR(&HORRTR)   TYPE(*CHAR) LEN(8)
            DCL        VAR(&ACKTBT)   TYPE(*CHAR) LEN(2)
            DCL        VAR(&LIBTBT)   TYPE(*CHAR) LEN(128)
            DCL        VAR(&NOMLOG)   TYPE(*CHAR) LEN(20)
            DCL        VAR(&KEYEXT)   TYPE(*CHAR) LEN(32)
            DCL        VAR(&USRPRF)   TYPE(*CHAR) LEN(16)


            DCL        VAR(&JOB) TYPE(*CHAR) LEN(10)
            DCL        VAR(&USER) TYPE(*CHAR) LEN(10)
            DCL        VAR(&NBR) TYPE(*CHAR) LEN(6)
            DCL        VAR(&MSGCMD) TYPE(*CHAR) LEN(64)
            DCL        VAR(&MSGACK) TYPE(*CHAR) LEN(256)

            MONMSG     MSGID(CPF0000) EXEC(GOTO CMDLBL(CPF0000))

            RTVJOBA    JOB(&JOB) USER(&USER) NBR(&NBR)
            CHGVAR     VAR(&MSGCMD) VALUE('WRKJOB JOB(' *TCAT &NBR +
                         *TCAT '/' *TCAT &USER *TCAT '/' *TCAT +
                         &JOB *TCAT ')')

ITER:
/*************************************************************/
/* APPEL DE LA COMMANDE DE RECEPTION                         */
/*************************************************************/
/* CALL RECEIVE COMMAND                                      */
/*************************************************************/
            IPSRCVTBT  FNCDEM(R) DEBDEM(&DEBDEM) FINDEM(C) +
```



```
                          EXCDEM(N) TRADEM(0) RTNCDP(&RTNCDP) +
                          KEYTBT(&KEYTBT) KEYUSR(&KEYUSR) +
                          ACKTBT(&ACKTBT) LIBTBT(&LIBTBT) +
                          OBJLIB(&OBJLIB) OBJFIL(&OBJFIL) +
                          OBJMBR(&OBJMBR) USRPRF(&USRPRF) +
                          DATFPC(&DATFPC) HORFPC(&HORFPC) +
                          DATFTR(&DATFTR) HORFTR(&HORFTR) +
                          DATRPC(&DATRPC) HORRPC(&HORRPC) +
                          DATRTR(&DATRTR) HORRTR(&HORRTR) +
                          SUPDEM(&SUPDEM) COMUSR(&COMUSR) +
                          NOMLOG(&NOMLOG) KEYEXT(&KEYEXT) /* Appel +
                          des API de TBT/400 via la Command +
                          IPSRCVTBT */

            IF         COND(&RTNCDP *NE 0) THEN(GOTO +
                          CMDLBL(ENDPGM)) /* Plus rien dans la file +
                          d'attente */

            CHGVAR     VAR(&DEBDEM)   VALUE('N')

            SNDPGMMSG  MSG('KEYTBT=' *CAT &KEYTBT)
            SNDPGMMSG  MSG('KEYUSR=' *CAT &KEYUSR)
            SNDPGMMSG  MSG('DATFPC=' *CAT &DATFPC)
            SNDPGMMSG  MSG('HORFPC=' *CAT &HORFPC)
            SNDPGMMSG  MSG('DATFTR=' *CAT &DATFTR)
            SNDPGMMSG  MSG('HORFTR=' *CAT &HORFTR)
            SNDPGMMSG  MSG('DATRPC=' *CAT &DATRPC)
            SNDPGMMSG  MSG('HORRPC=' *CAT &HORRPC)
            SNDPGMMSG  MSG('DATRTR=' *CAT &DATRTR)
            SNDPGMMSG  MSG('HORRTR=' *CAT &HORRTR)
            SNDPGMMSG  MSG('SUPDEM=' *CAT &SUPDEM)
            SNDPGMMSG  MSG('COMUSR=' *CAT &COMUSR)
            SNDPGMMSG  MSG('ACKTBT=' *CAT &ACKTBT)
            SNDPGMMSG  MSG('LIBTBT=' *CAT &LIBTBT)
            SNDPGMMSG  MSG('OBJLIB=' *CAT &OBJLIB)
            SNDPGMMSG  MSG('OBJFIL=' *CAT &OBJFIL)
            SNDPGMMSG  MSG('OBJMBR=' *CAT &OBJMBR)
            SNDPGMMSG  MSG('USRPRF=' *CAT &USRPRF)
            SNDPGMMSG  MSG('NOMLOG=' *CAT &NOMLOG)
            SNDPGMMSG  MSG('KEYEXT=' *CAT &KEYEXT)


/*************************************************************/
/* INSERER L'APPEL DE VOS TRAITEMENTS ICI                    */
/* Brancher obligatoirement en MESOK si OK                   */
/* Brancher obligatoirement en MESKO si erreur               */
/* Brancher obligatoirement en MESPC si statut inconnu       */
/*                                                           */
/* R E M A R Q U E :  Ce programme de consommation est une   */
/* maquette commune pour le traitement :                     */
/*   - des fichiers en entrée                                */
/*   - des acquittements de transmision reçus.               */
/* Cependant, DANS LE CAS DES ACQUITTEMENTS, il n'est pas    */
/* nécessaire de brancher la suite du traitement sur les     */
/* étiquettes MESOK et MESKO car la valorisation des champs  */
/* KEYUSR, ACKTBT, LIBTBT est sans conséquence sur le menu   */
/* "Supervision de l'historique".                            */
/*                                                           */
/* CALL USERBIB(USERPGM)                                     */
```

```
/* MONMSG     MSGID(CPF0000) EXEC(GOTO CMDLBL(MESKO)          */
/*                                                            */
/************************************************************/

/************************************************************/
/* INSERT APPLICATION PROCESS HERE                          */
/* Mandatory GOTO label        MESOK si OK                  */
/* Mandatory GOTO label        MESKO si Error               */
/* Mandatory GOTO label        MESPS si Unknown state       */
/*                                                          */
/* CALL YOURLIB(YOURPROGRAM)                                */
/* MONMSG     MSGID(CPF0000) EXEC(GOTO CMDLBL(MESKO)        */
/************************************************************/
             CALL       PGM(BM/EBITRTMSG) PARM(&ACKTBT &LIBTBT +
                          &OBJLIB &OBJFIL &OBJMBR &MSGCMD)
             MONMSG     MSGID(CPF0000) EXEC(GOTO CMDLBL(MESKO))

             IF         COND(&ACKTBT *NE '  ') THEN(GOTO CMDLBL(MESKO))

 MESOK:      CHGVAR     VAR(&KEYUSR) VALUE('Userkey')
             CHGVAR     VAR(&COMUSR) VALUE('Commentaire envoyé par +
                          le programme d''application')
/*           CHGVAR     VAR(&ACKTBT) VALUE('OK')                  */
/*           CHGVAR     VAR(&LIBTBT) VALUE('Message consommé avec +  */
/*                        succès')                                */
/*           CHGVAR     VAR(&SUPDEM) VALUE('N')    Override valeur +
                          initiale */
             GOTO       CMDLBL(MESFIN)

 MESPC:      CHGVAR     VAR(&KEYUSR) VALUE('Userkey')
             CHGVAR     VAR(&COMUSR) VALUE('Commentaire envoyé par +
                          le programme d''application')
             CHGVAR     VAR(&ACKTBT) VALUE('PC')
             CHGVAR     VAR(&LIBTBT) VALUE('Message pris en compte')
/*           CHGVAR     VAR(&SUPDEM) VALUE('N')    Override valeur +
                          initiale */
             GOTO       CMDLBL(MESFIN)

 MESKO:      CHGVAR     VAR(&KEYUSR) VALUE('Userkey')
             CHGVAR     VAR(&COMUSR) VALUE('Commentaire envoye par +
                          le programme d''application')
/*           CHGVAR     VAR(&ACKTBT) VALUE('KO')                  */
/*           CHGVAR     VAR(&LIBTBT) VALUE('Message en erreur')   */
/*           CHGVAR     VAR(&SUPDEM) VALUE('N')    Override valeur +
                          initiale */
             GOTO       CMDLBL(MESFIN)

 MESFIN:     CHGVAR     VAR(&MSGACK) VALUE('TBT/400 - Interface +
                          applicative - Code retour=' *CAT &ACKTBT +
                          *CAT ' :' *BCAT &LIBTBT *BCAT '- Pour +
                          visualiser le job utiliser la commande : +
                          ' *BCAT &MSGCMD)
             SNDMSG     MSG(&MSGACK) TOUSR(&USRPRF)
             MONMSG     MSGID(CPF0000)
             SNDMSG     MSG(&MSGACK) TOUSR(*SYSOPR)
             MONMSG     MSGID(CPF0000)
             IPSRCVTBT  FNCDEM(P) DEBDEM(N) FINDEM(C) EXCDEM(O) +
                          TRADEM(0) RTNCDP(&RTNCDP) KEYTBT(&KEYTBT) +
                          KEYUSR(&KEYUSR) ACKTBT(&ACKTBT) +
```



```
                          LIBTBT(&LIBTBT) SUPDEM(&SUPDEM) +
                          COMUSR(&COMUSR)
             GOTO         CMDLBL(ITER)


/*************************************************************/
/* INCIDENT HORS ITERATION                                   */
/*************************************************************/
/* ERROR    OUT OF LOOP                                      */
/*************************************************************/
 CPF0000:    CHGVAR       VAR(&ACKTBT) VALUE('AB')
             CHGVAR       VAR(&LIBTBT) VALUE('Exception rencontrée +
                            dans le programme')
             CHGVAR       VAR(&MSGACK) VALUE('TBT/400 - Interface +
                            applicative - Code retour=' *CAT &ACKTBT +
                            *CAT ' :' *BCAT &LIBTBT *BCAT '- Pour +
                            visualiser le job utiliser la commande : +
                            ' *BCAT &MSGCMD)
             SNDMSG       MSG(&MSGACK) TOUSR(&USRPRF)
             MONMSG       MSGID(CPF0000)
             SNDMSG       MSG(&MSGACK) TOUSR(*SYSOPR)
             MONMSG       MSGID(CPF0000)
             SNDPGMMSG    MSGID(CPF9898) MSGF(QSYS/QCPFMSG) +
                            MSGDTA(&MSGACK) MSGTYPE(*ESCAPE)
             MONMSG       MSGID(CPF0000)
ENDPGM:      ENDPGM
```


12.6.6 EBITRTMSG

```
/*-----------------------------------------------------------------*/
/*                                                                 */
/*  Appellé automatiquement par TBT/400 pour chaque réception      */
/*  (via la notion d'application)                                  */
/*                                                                 */
/*  Les variables &ACKTBT et &LIBTBT sont des zones de retour qui  */
/*  seront affichées dans l'historique de TBT/400.                 */
/*                                                                 */
/*  &ACKTBT='  ' => OK     + ligne en vert  dans l'historique      */
/*  &ACKTBT='KO' => Erreur + ligne en rouge dans l'historique      */
/*                                                                 */
/*-----------------------------------------------------------------*/

             PGM        PARM(&ACKTBT &LIBTBT &OBJLIB &OBJFIL &OBJMBR &MSGCMD)

             DCL        VAR(&OBJLIB)   TYPE(*CHAR) LEN(10)
             DCL        VAR(&OBJFIL)   TYPE(*CHAR) LEN(10)
             DCL        VAR(&OBJMBR)   TYPE(*CHAR) LEN(10)
             DCL        VAR(&ACKTBT) TYPE(*CHAR) LEN(2) VALUE('KO') +
                          /* Code retour - KO par défaut (zone de +
                          retour) */
             DCL        VAR(&LIBTBT) TYPE(*CHAR) LEN(128) /* +
                          Libéllé d'acheminement (zone de retour) */
             DCL        VAR(&MSGCMD) TYPE(*CHAR) LEN(64)

/* Le fichier reçu par TBT/400 est identifié par les champs:
*/
/* - &OBJLIB: Bibliotèque de reception,
*/
/* - &OBJFIL: Fichier     de reception,
*/
/* - &OBJMBR: Membre      de reception.
*/
/*
*/
/* Exemple de copie du fichier reçu par TBT/400 vers un fichier utilisé par
*/
/* l'applicatif final (copie sans contrôle - *NOCHK - et avec remplacement du
*/
/* membre exitant - *REPLACE).
*/
/*
*/
             CPYF       FROMFILE(&OBJLIB/&OBJFIL) +
                          TOFILE(QTEMP/&OBJFIL) MBROPT(*REPLACE) +
                          CRTFILE(*YES) FMTOPT(*NOCHK)
             MONMSG     MSGID(CPF0000) EXEC(GOTO CMDLBL(ERREUR))

/*
*/
/* Cet exemple se contente d'imprimer le fichier copié dans QTEMP au lieu
*/
/* d'appeller un programme utilisateur...
*/
/*
*/
/*           CALL       PGM(PGMCOMPTA) PARM(QTEMP &OBJFIL)          */
```

```
/*
*/
             CPYF       FROMFILE(QTEMP/&OBJFIL) TOFILE(*PRINT)
             MONMSG     MSGID(CPF0000) EXEC(GOTO CMDLBL(ERREUR))


 /* Autres traitements... */



/*                                                                  */
/* Si tout s'est passé correctement:                                */
/* - &ACKTBT = '  '                                                 */
/* - &LIBTBT = Libéllé d'acheminement positif                       */
/*                                                                  */
/* Si &ACKTBT n'est pas à '  ' à la fin de ce programme, ce dernier  */
/* sera considéré comme étant en erreur par TBT/400 (ligne en rouge  */
/* dans l'historique).                                              */
/*                                                                  */
             CHGVAR     VAR(&ACKTBT) VALUE('  ')
             CHGVAR     VAR(&LIBTBT) VALUE('Traitement réalisée +
                          correctement')
             GOTO       CMDLBL(FIN)

/* En cas d'erreur de traitement:                                   */
/* - &ACKTBT = 'KO' (=> ligne en rouge dans l'historique TBT/400)   */
/* - &LIBTBT = Libéllé d'acheminement négatif                       */
/*                                                                  */
 ERREUR:     CHGVAR     VAR(&ACKTBT) VALUE('KO')
             CHGVAR     VAR(&LIBTBT) VALUE('Traitement en erreur - +
                          voir : ' *BCAT &MSGCMD)
             GOTO       CMDLBL(FIN)

 FIN:        ENDPGM
```

