Connect:Express® Unix for SWIFTNet

Guide d'Implémentation

Version 2.0

Revision Date: 28 Mars 2007

Sterling Commerce
An IBM Company

Connect:Express Unix For SWIFTNet
Guide d'Implémentation

Version 2.0
Deuxième édition

TABLE DES MATIERES

Table des Illustrations

Préface

Le Guide d'Implémentation de Connect:Express Unix for SWIFTNet décrit l'architecture d'un serveur de transferts de fichiers SWIFT construit autour de Connect:Express UNIX. Il fournit les informations nécessaires à la configuration et à l'utilisation de Connect:Express UNIX associé, grâce à son option SWIFTNet, à l'interface de transfert de fichiers (FTI) de la SWIFTAlliance Gateway (la SAG).

Ce guide est destiné aux gestionnaires des transferts sur le réseau SWIFTNet et aux administrateurs de Connect:Express et. Il suppose connus le produit Connect:Express Unix et les fonctionnalités de la SAG. Il suppose que les composants ont été préalablement installés.

L'installation de Connect :Express for SWIFTNet est décrite dans un document séparé (*Notes d'installation*). L'installation et la configuration de la SAG sont décrites dans la documentation SWIFT.

Composition de ce manuel

Les informations sont organisées dans ce manuel de la façon suivante :

Type d'information	**Section**
Connect:Express et SWIFTNet	Chapitre 1 : Généralités
Environnement de Connect:Express Unix for SWIFTNet	Chapitre 1 : Généralités
Configuration de Connect:Express Unix for SWIFTNet	Chapitre 2 : Configuration
Configuration de FTI et utilisation des paramètres de FTI	Chapitre 2 : Configuration
Utilisation du protocole PeSIT (Pi99)	Chapitre 3 : Personnalisation des fichiers de Contrôle
Personnalisation	Chapitre 3 : Personnalisation des fichiers de Contrôle
Mise en œuvre du routage de bout en bout	Chapitre 3 : Personnalisation des fichiers de Contrôle
Compression des données	Chapitre 3 : Personnalisation des fichiers de Contrôle
Contrôle d'accès	Chapitre 3 : Personnalisation des fichiers de Contrôle
Gestion multi Back Office	Chapitre 3 : Personnalisation des fichiers de Contrôle
Identification et acquittement de bout en bout	Chapitre 4 : Description des flux
Procédures et automatismes	Chapitre 4 : Description des flux
Gestion des incidents, fichiers d'acquittement et de logging	Chapitre 4 : Description des flux
Routage par défaut (Règle numéro 0)	Annexe A - Exemples de règles de routage
Implémentation multi Back Office	Annexe A - Exemples de règles de routage
Implémentation d'une WhiteList	Annexe A - Exemples de règles de routage
Connect:Express OS/390 sur le Back Office	Annexe B - Connect:Express sur le Back-Office
Connect:Express UNIX sur le Back Office	Annexe B - Connect:Express sur le Back-Office
Exemples de masques de sélection	Annexe C - Expressions Régulières

Chapitre 1
Généralités

Ce chapitre décrit l'ensemble des composants mis en œuvre sur le serveur de transferts de fichiers SWIFT :

- SWIFT: SWIFTAlliance Gateway
- FileAct et File Transfer Integration
- Connect:Express Unix for SWIFTNet.

Il définit aussi l'environnement du Back Office et il indique la marche à suivre pour configurer l'ensemble et mettre en oeuvre les mécanismes de routage.

SWIFT et SWIFTNet

SWIFT (Society for Worldwide Interbank Financial Telecommunication) fournit un service de messagerie sécurisée à plus de 7000 institutions financières dans 197 pays.

SWIFTNet est un réseau sur IP basé sur les technologies de l'internet. Avec SWIFTNet de nouvelles infrastructures de service sont proposées. Ces services, désignés sous le terme de FileAct, fournissent l'infrastructure nécessaire au transfert de gros volumes de données et de nombres importants de fichiers.

SWIFTAlliance Gateway

La '*SWIFT Alliance Gateway'* (SAG) est le serveur qui fournit l'accès à SWIFTNet. La SAG est l'un des trois moyens d'accès à SWIFTNet. En plus du traffic de message existant, La SAG supporte la fonctionnalité FileAct au travers d'un produit intégré : File Transfer Integrated (FTI). FTI fournit une interface de commande permettant d'envoyer et de recevoir des fichiers au travers de SWIFTNet. Les produits Connect:Express s'appuient sur cette interface pour assurer l'intégration des environnements de la SAG et du Back-Office.

La SAG assure aussi un service sécurisé, Secure Network Link (SNL).

FileAct et File Transfer Integrated (FTI)

FileAct est un produit SWIFT fonctionnant parallèlement à InterAct (messagerie en temps réel). FileAct est une interface de programmation (API) de type commande utilisant XML pour ses strutures de requêtes et d'acquittements. FTI fait parti du produit SWIFT Alliance Gateway. Il remplace Interactive File Transfer (IFT), une technologie basée sur le réseau X25 qui a été utilisée de façon limitée. FileAct propose trois modes d'accès :

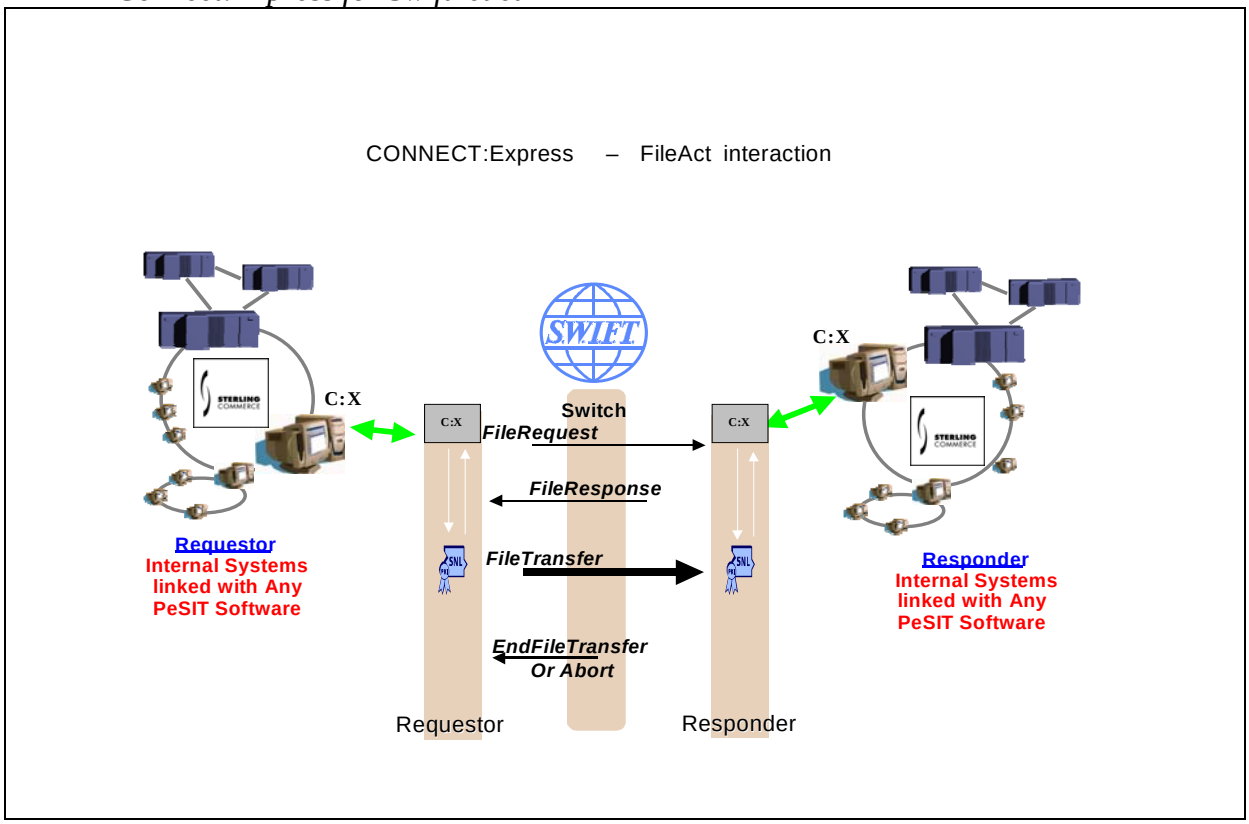
- File Transfer Integrated (FTI)
- File Transfer Agent (FTA)
- Interface Web

Connect:Express Unix for SWIFTNet et FTI

Connect:Express Unix permet d'intégrer les transferts de fichiers via SWIFTNet à un réseau existant basé sur des transferts de fichier PeSIT. Le produit standard Connect:Express s'appuie sur le mécanisme des commandes de fin de transfert et sa propre interface batch de requête de transfert pour assurer cette intégration. Ceci permet à l'utilisateur de mettre en place l'automatisation des transferts de fichiers SWIFTNet. L'interaction entre Connect:Express for SWIFNet et FileACT est basée sur les paramètres de transfert de FileACT.

Le schéma ci-dessous décrit l'interaction entre Connect:Express et FileAct lorsque une demande Get ou Put est soumise. Le Back Office est supposé équipé lui même de Connect:Express.

Connect:Express for Swiftnet et FTI



Un mécanisme de routage est mis en place à l'aide de programmes activés automatiquement par Connect:Express à la fin des transferts de fichiers effectués en PeSIT entre le Back-office et la SAG, dans les deux sens, et de programmes activés automatiquement par FTI sur toute demande en provenance du réseau SWIFTNet.

La version 2.0 de Connect:Express Unix for SWIFTNet prend en charge les flux suivants :

- Envoi d'un fichier depuis le Back-office vers le réseau Swift (Programme SWIFTPUT)
- Routage d'un fichier depuis le réseau Swift vers le Back-office (Programme SWIFTFWD)
- Demande de réception d'un fichier du Back-office au réseau Swift (Programme SWIFTGET)
- Mise à disposition d'un fichier depuis le Back-office vers le réseau Swift (Programme SWIFTSTO)
- Récupération par le réseau Swift d'un fichier du Back-office (Programme SWIFTINQ)

La version 2.0 de Connect:Express for SWIFTNet permet de personnaliser le contrôle des demandes en provenance du réseau SWIFT.

- Contrôle des demandes Put (Programme ConnectAcceptPutInit)
- Contrôle des demandes Get (Programme ConnectAccept, fichiers disponibles sur la SAG seulement)
- Contrôle des demandes Get (Programme ConnectAcceptInquiry, recherche sur le Back Office)

Les mécanismes de routage peuvent être personnalisés et diversifiés.

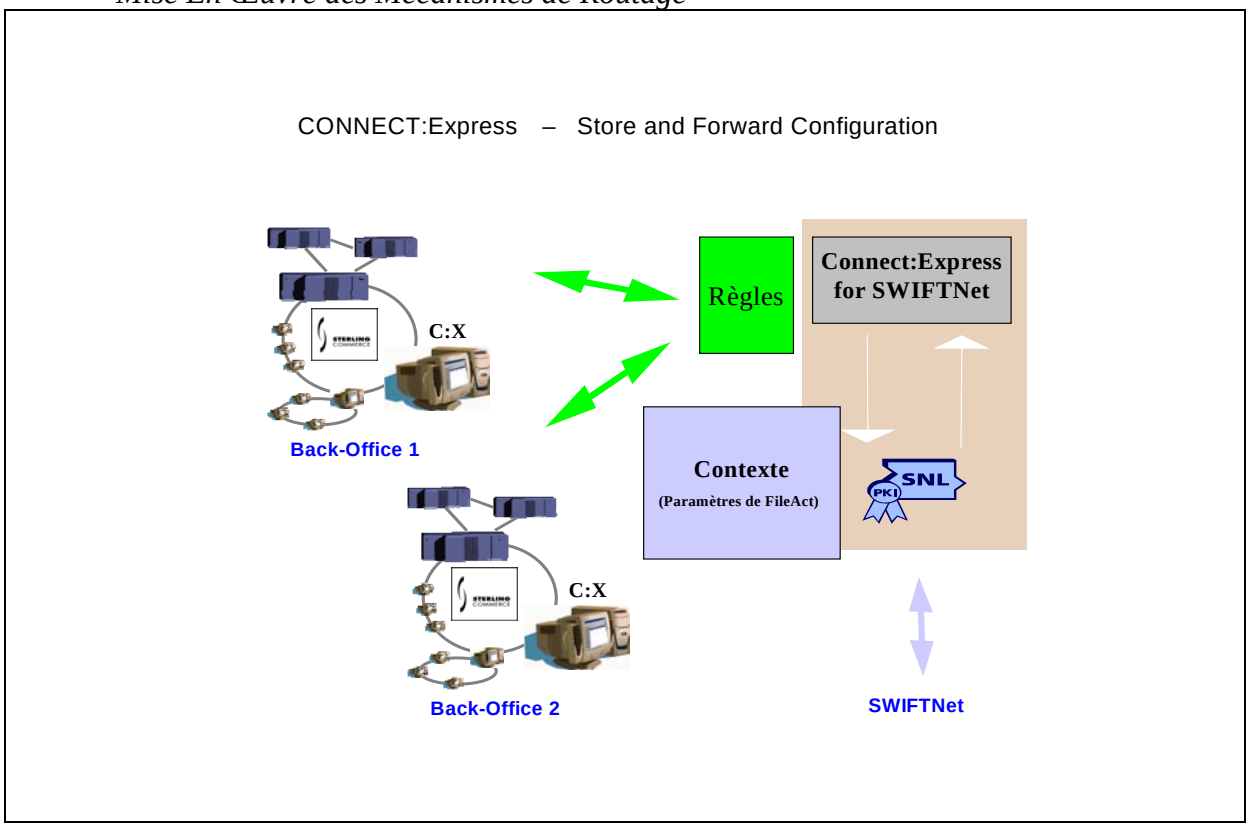
Connect:Express Unix for SWIFTNet et le Back-Office

La version 2.0 de Connect:Express for SWIFTNet permet de mettre en place des communications avec plusieurs Back-offices et de contrôler les accès depuis le réseau SWIFTNet.

Les procédures automatiques exécutent le routage et le contrôle d'accès en fonction de règles déterminées d'après le contexte courant (les paramètres de FileAct) de la demande de transfert. Chaque Back-Office peut bénéficier de règles de routage particulières.

Le schéma ci-dessous illustre les interactions entre FTI, Connect:Express for SWIFTNet et les Back Offices.

Mise En Œuvre des Mécanismes de Routage



Description des Environnements

Connect:Express Unix for SWIFTNet est destiné à s'exécuter sur la Gateway Swift (SAG) en relation avec l'interface de transfert Swift (FTI). Une fois l'interface FTI de la SAG et Connect:Express installés, sur le même compte utilisateur, la mise en œuvre peut se faire en deux phases : une phase de configuration, décrite au Chapitre 2, permet de mettre en œuvre les interactions entre les deux composants et les transferts PeSIT avec le Back Office. Une phase de personnalisation, décrite au Chapitre 3, permet de mettre en œuvre les mécanismes de routage.

Les informations données dans ce chapitre précisent les composants présents sur la SAG et la marche à suivre pour la mise en oeuvre de Connect:Express Unix for SWIFTNet. Elles concernent trois environnements. L'environnement Connect:Express sur la SAG, organisé sur la racine désignée par $TOM_DIR, l'environnement de FTI sur la SAG, organisé sur la racine désignée par $SAG_DIR, et l'environnement du Back-office.

Un fichier *profile* est fourni: il contient des variables d'environnement à définir et activer avant la mise en route de Connect:Express Unix for SWIFTNet.

Les installations de Connect:Express Unix for SWIFTNet et de FTI doivent être faites sur le même compte utilisateur pour que toutes les procédures automatiques de Connect:Express for SWIFTNet aient le droit de s'exécuter .

L'Environnement de Connect:Express Unix for SWIFTNet

L'environnement de Connect:Express Unix for SWIFTNet a été installé sur la racine désignée par $TOM_DIR . Cette section vous permet de vérifier que tous les composants ont bien été installés.

Connect:Express Unix for SWIFTNet fournit des programmes exécutables, des fichiers de configuration et des fichiers de contrôle.

Les Répertoires de Connect:Express Unix for SWIFTNet

$TOM_DIR/

Répertoire	Description
(Racine)	Fichier *profile* définissant les variables d'environnement
config/	Fichiers de configuration des transferts PeSIT (Sysin, Partenaires et Fichiers)
exit/	Les programmes de Connect:Express for SWIFTNet et les fichiers de contrôle du routage ainsi que le fichier swift.log de Connect:Express for SWIFTNet Pour activer une trace le fichier témoin swifttrace.flag doit y être créé.
exit/swlog/	Fichiers d'archivage journalier de swift.log, dont le nom est de la forme L'date' (Exemple: L20030810)
itom/	Emplacements des utilitaires standards
in/	Emplacement provisoire des fichiers de données reçus du Back-office par Connect:Express et des fichiers de paramètres associés
out/	Emplacement provisoire des fichiers de données reçus de SWIFTNet et des fichiers de paramètres associés

Le Chapitre 2 décrit la mise en œuvre des fichiers de configuration, le chapitre 3 décrit la personnalisation des fichiers de contrôle du routage. Le Chapitre 4 décrit les automatismes mis en œuvre par les programmes. Des exemples sont fournis en Annexe.

Le Fichier Profile Fourni dans le Répertoire Racine

Le Chapitre 2 présente un exemple de fichier profile.

Dans ce qui suit, et dans les programmes fournis, les variables d'environnement suivantes sont déclarées:

Variable	**Valeur**
$TOM_DIR	<Connect:Express install directory>
$SAG_DIR	<SAG install directory>
$SAG_XML	<SAG install directory>/FT/data/xmlparamfile/
$SAG_RCV	<SAG install directory>/FT/data/reception/
$SAG_DOWN	<SAG install directory>/FT/data/download/
$SAG_LOG	<SAG install directory>/FT/data/log/
$SAG_BIN	<SAG install directory>/bin
$CENTRAL Paramètre facultatif.	Nom symbolique de partenaire représentant le Back-office par défaut et défini dans le fichier des Partenaires de Connect:Express .
$TOM_INQUIRY_TIMER Paramètre facultatif	Temporisation utilisée pour la reherche de fichier sur le Back Office, valeur par défaut = 60 secondes.
$KEEP_SW_LOG Paramètre facultatif	Nombre maximum de fichiers de log journaliers archivés conservés, valeur par défaut = 7 jours

Note : Ces variables doivent être actives dans tous les environnements où peuvent s'exécuter les programmes. Le fichier *profile* livré doit donc être intégré au fichier *.profile* du compte utilisateur sur lequel s'exécutent les composants.
La SAG doit être arrêtée puis re-démarrée afin de prendre en compte tout changement des variables ci-dessus.

Les Fichiers de Configuration Fournis dans le Répertoire Config

$TOM_DIR/config/

Les fichiers de configuration des transferts PeSIT sont presque complètement pré-définis : la définition des partenaires est la principale information à compléter.

Le Chapitre 2 explique les principes d'utilisation de ces fichiers et donne des exemples de configuration.

Fichier	Description
sysin	Contient l'identité par défaut de Connect:Express vis à vis du ou des Back-offices
RPAR	Contient la définition du ou des Back-offices et leur configuration PeSIT
RFIC	Contient la définition des transferts de fichiers avec le ou les Back-offices
#CENTRAL	Contient la liste des Back-offices

Les Fichiers de Contrôle Fournis en Exemples dans le Répertoire Exit

$TOM_DIR/exit/

Des fichiers xml et leurs fichiers de contrôle de syntaxe xsd sont livrés dans le répertoire /exit.
La personnalisation des fichiers de contrôle du routage doit être précédée par une étude préalable des règles de routage à mettre en œuvre. Des fichiers de contrôle correspondant aux règles de routage appliquées dans la version 1 de Connect:Express for SWIFTNet présentée dans l'Annexe A sont fournis.

Le Chapitre 3 explique les principes d'utilisation de ces fichiers. L'Annexe A donne des exemples de personnalisation des fichiers de contrôle.

Fichier	Description
SwiftPut.xml	Squelette de fichier de paramètres FTI utilisés pour soumettre un Put
SwiftGet.xml	Squelette de fichier de paramètres FTI utilisés pour soumettre un Get
SwiftSto.xml	Squelette de fichier de paramètres FTI utilisés pour mettre à disposition un fichier
RuleList.xml	Description des régles de routage
RuleList.xsd	Contrôle syntaxique du fichier RuleList.xml
Findrule.xml	Détermination des règles de routage en fonction du contexte courant
FindRule.xsd	Contrôle syntaxique du fichier FindRule.xml
WhiteList.xml	Contrôle des utilisateurs autorisés
BlackList.xml	Contrôle des utilisateurs non autorisés
AccessList.xsd	Contrôle syntaxique des fichiers BlackList.xml et WhiteList.xml

Les Programmes Fournis dans le Répertoire Exit

$TOM_DIR/exit/

Le Chapitre 4 décrit l'exécution de ces programmes au travers de la description des flux.

1. Programmes activés en fin de transfert

Programme	**Description**
SWIFTPUT	Requête à FTI – ft-put
SWIFTGET	Requête à FTI – ft-get et transfert vers le Back-office du fichier reçu
SWIFTSTO	Mise à disposition sur la SAG d'un fichier en attente de get entrant
SWIFTINQ	Récupération d'un fichier sur le Back-office, sur get entrant
CLEARFWD	Fin de transfert d'un fichier de données routé vers le Back-office
CLEARACK	Fin d'aquittement positif
CLEARNAK	Fin d'aquittement négatif

2. Programmes activés par FTI

Programme	**Description**
ConnectPutInit	Contrôle d'accès suite à un Put entrant
ConnectAccept	Contrôle d'accès et contrôle de la présence du fichier sur la SAG suite à un Get entrant
ConnectAcceptInq	Contrôle d'accès et contrôle de la présence du fichier sur la SAG suite à un get entrant avec récupération sur le Back-office
SWIFTFWD	Requête à Connect:Express – transfert vers le Back-office suite à un put entrant

3. Utilitaires

Programme	**Description**
PURGESAG des	Exploitation - purge des objets périmés : fichiers de données, xml, ack, nak et log, et requêtes de transfert
ParseRuleList	Contrôle du fichier RuleList.xml à partir du fichier RuleList.xsd
ParseFindRule	Contrôle du fichier FindRule.xml à partir du fichier FindRule.xsd
ParseAccessList	Contrôle des fichier WhiteList.xml et BlackList à partir du fichier AccessList.xsd
testre	Utilitaire de test des expressions régulières (voir Annexe C)

L'Environnement de FTI

La configuration de FTI se fait par la console d'administration de la Sag. Le Chapitre 2 donne des exemples d'écrans de configuration de FTI .

L'interaction entre FTI et Connect:Express est contrôlée par les paramètres suivants :

Paramètre	Description
LTA-PutInit	Emplacement du programme de contrôle déclenché sur réception de put externe
LTA-GetInit	Emplacement du programme de contrôle déclenché sur réception de get externe
LTA-PutEnd	Emplacement du programme d'émission vers le Back-office
GenerateXMLparamfile	Option de génération des fichiers XML de paramètres FTI
XMLParamFilesDirectory	Emplacement des fichiers XML de paramètres FTI
Reception Directory	Emplacement des fichiers reçus par FTI
Download Directory	Emplacement des fichiers à émettre par FTI
Log Directory	Emplacement des fichiers log de FTI
Download Notification	Option de génération d'une notification par FTI
Download Notification…	Définition du paramètre Request type

Les programmes fournis avec Connect:Express Unix for SWIFTNet prennent en compte de façon dynamique les répertoires FTI grâce aux variables d'environnements définies dans le fichier profile. Le paramètre *LTA-PutInit* doit indiquer le programme *ConnectAccepPutInit* qui sera déclenché sur réception par FTI d'une demande d'émission externe (Put). Le paramètre *LTA-PutEnd* doit indiquer le programme *SWIFTFWD* qui sera déclenché en fin de réception par FTI d'un fichier à destination du Back-office. Le paramètre *LTA-Getinit* doit indiquer le programme *ConnectAccept* (ou *ConnectAcceptInq*) qui sera déclenché sur réception par FTI d'une demande de réception externe (download).

L'option de génération des fichiers XML doit être activée (*yes*) car les programmes *SWIFTFWD* et *ConnectAccept* traitent un fichier de paramètres XML en entrée.

L'option de génération des notifications peut être activée (*yes*).

L'Environnement du Back-Office

L'environnement du Back-Office comprend le moniteur de transfert de fichier en charge des communications avec Connect:Express Unix for SWIFTNet installé sur la SAG. Ce moniteur doit supporter le protocole PeSIT, connaitre le moniteur PeSIT de la SAG et prendre en compte les flux de données prévus entre la SAG et le Back-office. De plus toute requête de transfert vers Connect:Express Unix for SWIFTNet doit avoir un format spécifique utilisé pour passer des paramètres FTI associés au fichier transféré. Le mécanisme d'acquittement de bout en bout repose sur ces paramètres. Le moniteur sur le Back-office peut être Connect:Express.

Connect:Express sur le Back-Office

Si le moniteur PeSIT installé sur le Back-office est Connect:Express, l'environnement doit intégrer les définitions suivantes :

Répertoire des Partenaires : définition de Connect:Express sur la SAG, avec un alias égal au nom de partenaire défini dans le répertoire de Connect:Express Unix for SWIFTNet.

Répertoire des Fichiers : définition des fichiers symboliques correspondant aux fichiers symboliques définis dans le répertoire de Connect:Express Unix for SWIFTNet .

Requête de transfert : Utilisation du champ 'Pi99' de la requête de transfert, ou équivalent, pour passer les paramètres FTI.

L'annexe B donne des exemples de paramétrage de Connect:Express du côté du Back Office.

Etapes de la Mise en Oeuvre

Une fois les différents composants installés, il est nécessaire de configurer des paramètres généraux puis de personnaliser les mécanismes de routage entre Connect:Express for SWIFTNet et le ou les Back-Offices.

La configuration des paramètres généraux est décrite au Chapitre 2. Elle concerne la mise en œuvre des interactions entre les trois environnements :

1. Personnaliser le fichier $TOM_DIR/profile
2. S'assurer qu'à la connexion de l'utilisateur le fichier *profile* fourni s'exécute et vérifer que les variables d'environnements sont bien actives. Si la SAG est déjà démarrée, l'arrêter puis la re-démarrer.
3. Personnaliser le fichier sysin de Connect:Express Unix for SWIFTNet: *TCPORT*, *AUTH21*, *ALIASN*
4. Définir le ou les Back-offices sur le modèle du partenaire *CENTRAL* défini dans le fichier RPAR de Connect:Express Unix for SWIFTNet
5. Mettre à jour la liste des Back-offices #CENTRAL
6. Vérifier avec chaque Back-office la configuration du protocole PeSIT : version, taille de message, compression
7. Configurer les paramètres *LTA-PutInit, LTA-PutEnd* et *LTA-GetInit* de FTI
8. Activer l'option *GenerateXMLParamFile* de FTI
9. Activer l'option *DownloadNotification* de FTI
10. Définir et paramétrer le partenaire Connect:Express for SWIFTNet et les flux sur le Back-office

La personnalisation des règles de routage est décrite au Chapitre 3. Elle concerne la mise en œuvre du suivi de bout en bout des transferts de fichier SWIFTet du contrôle d'accès :

11. Définir les règles de routage .
12. Personnaliser les fichiers de contrôle xml dans le répertoire /exit de Connect:Express Unix for SWIFTNet. Pour tous ces fichiers le tag AccessListCfg doit impérativement indiquer le nom physique complet du fichier xsd de contrôle de syntaxe associé.
13. Appliquer impérativement les programmes de contrôle de syntaxe ParseRuleList, ParseFindRule puis ParseAccessList aux fichiers RuleList.xml, FindList.xml puis WhiteList.xml et BlackList.xml, respectivement.

Note : En phase de mise au point il est possible d'activer une trace dans le fichier swift.log. Pour celà, créer un fichier témoin swiftrace.flag dans le répertoir /exit. Pour désactiver dynamiquement la trrace, il suffit de supprimer ce fichier.

Chapitre 2

Configuration

Ce chapitre décrit les différentes étapes à exécuter après l'installation des composants et avant la personnalisation des mécanismes de routage.

Les paramètres de configuration de Connect:Express Unix for SWIFTNet et de l'interface FTI doivent être modifiés pour assurer l'interaction entre les composants. Le ou les Back-Offices doivent aussi être paramétrés pour s'intégrer à la chaine de bout en bout.

Ce chapitre suppose la SAG et Connect:Express préalablement installés, conformément aux manuels d'installation respectifs. Les deux produits doivent être installés sous le même compte utilisateur.

Configuration de Connect:Express Unix for SWIFTNet

Une fois Connect:Express installé, il est nécessaire de configurer certains paramètres. Les fichiers de configuration sont préparés et livrés avec un exemple complet de paramétrage, mais quelques valeurs doivent obligatoirement être adaptées. Il est possible aussi de définir un autre paramétrage, sur la base des exemples fournis : on peut par exemple décider d'utiliser des noms symboliques de partenaires différents, aussi bien pour le ou les Back-offices que pour Connect:Express Unix for SWIFTNet .

Les noms symboliques de fichiers, eux, sont fixés et obligatoires. Ils sont partagés par tous les Back-offices définis. Il est possible d'en ajouter pour des utilisations spécifiques liées à l'exploitation.

Le moniteur Connect:Express est identifié, par défaut, par le paramètre DPCSID=SWIFTNET du fichier SYSIN, le moniteur du Back-office est défini, pour l 'exemple, dans le fichier des partenaires (RPAR) sous l'identifiant CENTRAL et les transferts avec le partenaire CENTRAL sont définis dans le répertoire des fichiers (RFIC) sous sept noms symboliques commençant par la racine SWIFT et correspondant aux différents flux : PUT, GET, ACK, NAK, STO, FWD, INQ.

Toutes ces définitions symboliques doivent avoir leur correspondance sur le moniteur du Back-office : partenaire SWIFTNET et fichiers symboliques.

Activation des Variables d'Environnement

$TOM_DIR/profile

Les procédures livrées avec Connect:Express Unix for SWIFTNet ne nécessitent aucune personnalisation. Ceci est possible grâce aux variables d'environnements suivantes, qui doivent être personnalisées :

Exemple de Fichier Profile

```
#
TOM_DIR=/test/cexpress/x141

env_sys=`uname -s`
if [ $env_sys = "AIX" ]; then
  export LIBPATH=$TOM_DIR/exit:$TOM_DIR/strf:$LIBPATH
fi
if [ $env_sys = "SunOS" ]; then
  export LD_LIBRARY_PATH=$TOM_DIR/exit:$TOM_DIR/strf:$LD_LIBRARY_PATH
fi
if [ $env_sys = "HP-UX" ]; then
  export SHLIB_PATH=$TOM_DIR/exit:TOM_DIR/strf:$SHLIB_PATH
fi

compress_tom=$TOM_DIR"/config/compress.sh"
gtrf=$TOM_DIR/gtrf/tom_mon
sterm=$TOM_DIR/sterm/tom_opr
start_tom=$TOM_DIR"/gtrf/tom_mon"
stop_tom=$TOM_DIR/config/stop_tom
export compress_tom gtrf sterm stop_tom TOM_DIR start_tom

p1b8preq=$TOM_DIR/itom/p1b8preq
p1b8pcan=$TOM_DIR/itom/p1b8pcan
p1b8pren=$TOM_DIR/itom/p1b8pren
p1b8pret=$TOM_DIR/itom/p1b8pret
p1b8ppur=$TOM_DIR/itom/p1b8ppur
export p1b8pcan p1b8pren p1b8pret p1b8ppur p1b8preq

#
# Define the following timer value if transfer downloads from Back Office by using
# ConnectAcceptInq can last more than the default value of 60 seconds
# (see Documentation)
#export TOM_INQUIRY_TIMER=60
# Define the maximum number of archived swift.log files that should be kept in the /exit/swlog directory
(default=7)
#export KEEP_SW_LOG=7
#export CENTRAL=CENTRAL

export SAG_DIR=/SWIFTAlliance/Gateway
export SAG_XML=/SWIFTAlliance/Gateway/FT/data/xmlparamfile
export SAG_RCV=/SWIFTAlliance/Gateway/FT/data/reception
export SAG_DOWN=/SWIFTAlliance/Gateway/FT/data/download
export SAG_LOG=/SWIFTAlliance/Gateway/FT/data/log
export SAG_BIN=/SWIFTAlliance/Gateway/bin
```

Ce fichier profile contient les variables d'environnement liées au moniteur Connect:Express UNIX et les variables propres à Connect:Express for SWIFTNet. le tableau suivant définit les secondes.

Variable	Description
$TOM_DIR	Répertoire racine de Connect:Express
$SAG_DIR	Répertoire racine de la SAG
$SAG_XML	Stockage des fichiers paramètres de FTI
$SAG_RCV	Stockage des fichiers reçus par FTI
$SAG_DOWN	Stockage des fichiers mis à disposition par le Back Office
$SAG_LOG	Stockage des fichiers de logging de FTI
$SAG_BIN	Stockage des exécutables ft-put et ft-get
$CENTRAL Paramètre facultatif	Nom symbolique de partenaire représentant le Back-office par défaut et défini dans le fichier des Partenaires de Connect:Express.
$TOM_INQUIRY_TIMER Paramètre facultatif	Temporisation utilisée pour la reherche de fichier sur le Back Office. La valeur par défaut est 60 secondes.
$KEEP_SW_LOG Paramètre facultatif	Nombre maximum de fichiers de log journaliers archivés conservés, valeur par défaut = 7 jours

Fichier d'Initialisation de Connect:Express

$TOM_DIR/config/sysin

Un fichier sysin initialisé est fourni. Les valeurs marquées entre crochets (**< >**) doivent être obligatoirement personnalisées. Les autres valeurs peuvent être modifiées. L'identification locale par défaut de Connect:Express Unix for SWIFTNet est déclarée dans ce fichier.

Notes : Les paramètres DPCSID, DPCPSW et TCPORT doivent être échangés avec le Back-office.Le partenaire 'SWIFTNET' doit être déclaré sur le Back-office.

Fichier Sysin de Connect:Express For Swiftnet

Keyword	Length	Value	Description
DPCSID	1 to 8 alph. c.	**SWIFTNET**	Default symbolic partner name of Local Monitor.
DPCPSW	1 to 8 alph. c.	**SWIFTNET**	Password of Local Monitor.
STIMEV	2 fields of 2 numeric values; unit = 1 min.	**01,01**	1st field: Time between connection retries. 2nd field: Time to wait before initiating a transfer request again.
AUTH21	50 to 75 hexa.l c	**< >**	The authorization key to use CONNECT:Express when you first install the product. This key is provided by Sterling Commerce
ALIASN	Alphanum.	**< >**	Optional authorization alias name given by Sterling Commerce when you first install CONNECT:Express.
SIZLOG	4 numeric c.	**5000**	The number of records in the LOG file. If this value is changed, you must convert the old structure to a new one.
Keyword	**Length**	**Value**	**Description**
LAUNCH	1 c.	**H**	C=Cold start. With a cold start, the monitor ignores the RENC file and all unfinished transfers are considered as abnormally ended. H=Hot start. The monitor tries to restart transfers that were in Progress when the monitor terminated.
DEVDEF	*Paramétrage X25*	*Not used*	
TCPORT	4 numeric c. <= 6999	**< >**	TCPIP port to listen for incoming calls.
STRACE	1 numeric char.	**0**	This is a trace option flag. 0 means no trace. In this case, work files are not created. 1 means a minimal trace is active. Trace can be activated dynamically. You can use the command kill -USR1 to the monitor and change the STRACE flag.
SYSLOG	1 numeric char.	**0**	This is the logging option flag. 0 disables this option. 1 means syslog support is active. In this case, informational or error messages are sent to the syslog daemon.
FTPORT	4 numeric c. <= 6999	*Not used*	
DEFILE	1 to 8 alphanum. c.	*Not used*	

Définition du Back-office

$TOM_DIR/config/RPAR

Un fichier des Partenaires RPAR initialisé est fourni. La liaison avec le Back-office est représentée par le partenaire CENTRAL défini en PeSIT-E sur TCP/IP, de type 'Other'.

Note : Le Back-office doit se présenter et se reconnaître comme partenaire 'CENTRAL' .

Les valeurs entre crochets (**< >)** doivent être obligatoirement personnalisées. Les autres valeurs peuvent être modifiées.

Definition du Back Office

```
C:X for SWFTNet 141-1-V2.00 ------------ PARTNERS DIRECTORY ------------------------ TOM1
OPTION ===>
SYMBOLIC NAME :              CENTRAL

PASSWORD ........... :       CENTRAL    PASSWORD OF PARTNER
INITIALIZATION STATUS . :    E          E:ENABLE H:DISABLE
PARTNER TYPE ......... :     O          T/O
PROTOCOL NUMBER ...... :     3          1:ETEBAC 3, 2:FTP, 3:PESIT
SESSION TABLE NUMBER .. :    1          1->9 SESSION TABLES
X25 PORT ............. :                X25 DEVICE NAME
MAX. NO. CONNECTIONS .. :    10/10/10   01->64 TOT/IN/OUT
TYPE OF CONNECTION .... :    T          X, P, T OR M
X25 DIAL NUMBER ...... :     1-15 CHARACTERS
LOCAL DIAL NUMBER ..... :    1-15 CHARACTERS
EXTRA NETWORK FIELD ... :    'USER-DATA-FIELD'
FACILITIES ........... :
TCPIP HOST ........... :     <  >
TCPIP ADDRESS ........ :     <  >       PORT ......... : <  >
DPCSID ALIAS ......... :     DEF FTP FILE .. :
DPCPSW ALIAS ......... :
DO YOU WANT TO GO ON ?
OPTION : VIEW UPD : 98/08/04 10:41 root
-ENTER- NEXT FIELD -F3- CANCEL -F8- COMPLETION
```

Cette définition contitue un modèle. Il peut y avoir autant de définitions que de Back-offices possibles.

De plus l'utilisation des champs **ALIAS** permet de donner une identification locale différente de celle fournie dans le fichier SYSIN de Connect:Express Unix for SWIFTNet.

Modification de la Liste des Back-Offices

$TOM_DIR/config/#CENTRAL

Une liste des Back Offices # CENTRAL initialisé est fourni : il ne contient que le partenaire CENTRAL.
L'ajout d'un nouveau nom dans le fichier des Partenaires doit être accompagné de la mise à jour de cette liste :

Liste des Back Offices

```
vi '#CENTRAL'

CENTRAL
MYBANK1
MYBANK2
```

Paramétrage du PeSIT E
$TOM_DIR/config/Param

Le partenaire CENTRAL est configuré avec la table de session numéro 1. Cette table de session est livrée avec le numéro de version PeSIT D : modifier le champ Level, le passer à 2.

```
C:X for SWFTNet 141-1-V2.00 -------------- SESSION S1 ------------------------ ce01
OPTION ===>
U UPDATE
LINE MESSAGE SIZE ..... : 04096              BYTES < 65536
SYNCHRONIZATION ....... : 32                 K. BYTES
WINDOW ................ : 16                 0 - 16
LEVEL ................. : 2                  PROTOCOL VERSION NUMBER
NUMBER OF RETRY ....... : 5                  0 - 9
CRC ................... : N                  Y/N
MAJ : 97/09/16 10:55 pla
-ENTER- NEXT FIELD -F3- CANCEL -F8- COMPLETION
```

Visualisation des Transferts avec le Back-Office et du Paramétrage des Commandes de Fin de Transfert
$TOM_DIR/config/RFIC

Un fichier des Fichiers Symboliques RFIC initialisé est fourni. Il définit l'ensemble des profils de fichiers nécessaires à la gestion des échanges avec le ou les Back-offices. Ce fichier peut être utilisé tel quel.

Chaque définition de fichier décrit le fichier à transférer et les actions à déclencher en fin de transfert. Tous les fichiers sont considérés comme fichiers texte. Le tableau suivant présente la liste des définitions : chacune associe une commande de fin de transfert à un nom symbolique de fichier et est utilisée pour les fonctions décrites.

Nom Symbolique	Commande de fin	Fonctions	Description
SWIFTPUT	SWIFTPUT	put	Réception pour transmission du back-office vers swift
SWIFTGET	SWIFTGET	get	Réception pour download du back-office vers swift
SWIFTSTO	SWIFTSTO	get externe	Réception pour mise à disposition de swift
SWIFTINQ	SWIFTINQ	get externe	Réception pour récupération par swift
SWIFTFWD	CLEARFWD	put externe,get	Transmission des données de swift vers le back-office
SWIFTACK	CLEARACK	put, get	Transmission ack de swift vers le Back-office
SWIFTNAK	CLEARNAK	put, get	Transmission nak de swift vers le Back-office

D'autres profils peuvent être définis, avec des noms symboliques quelconques, mais ils doivent être paramétrés en fonction du flux concerné et sur le modèle des profils fournis.

Note : Les noms symboliques de fichiers définis ci-dessus sont fixes et doivent être connus par le Back-office.

SWIFTPUT

Le fichier **SWIFTPUT** est défini en réception, le Back-office est émetteur. Les champs *RECEIVING PARTNER* et *TRANSMITTING PARTNER* pointent sur la liste des Back-offices. Le champ *RECEPTION END COMMAND* est initialisé à SWIFTPUT. SWIFTPUT lance une command ft-put à FTI puis lance le transfert SWIFTACK ou le transfert SWIFTNAK. Le format d'enregistrement est fixe ou variable (*****) en fonction de l'information reçue du Back Office émetteur, la taille d'enregistrement sera indiquée aussi par le Back Office (**0000**).

Definition de Fichier Swiftput

```
C:X for SWFTNet 141-1-V2.00 -------------- FILES DIRECTORY ------------------------ TOM1
OPTION ===>
SYMBOLIC NAME :            SWIFTPUT
INITIALIZATION STATUS . :  E            E:ENABLE H:DISABLE
DIRECTION ............. :  R            T:TRANSMIT R:RECEIVE *:EITHER
RECEIVING PARTNER ..... :  #CENTRAL     'NAME',#LISTE, $$ALL$$
TRANSMITTING PARTNER .. :  #CENTRAL     'NAME',#LISTE, $$ALL$$
PRIORITY .............. :  0            0:URGENT 1:FAST 2:NORMAL
DEFINITION TYPE ....... :  D            D:DYNAMIC F:FIXED
PRESENTATION TABLE .... :  1            1 -> 9 PRESENTATION TABLE
PARAMETER CARDS FILE    :  N            Y/N
SPACE TO RESERVE ...... :  N            Y/N
ALLOCATION RULE ....... :  0            0:INDIF., 1:PREALL., 2:TO CREATE
PHYSICAL NAME ......... :  $TOM_DIR/in/put_&REQNUMB.tmp
RECORD FORMAT ......... :  T*           TF, TV, BF, BU, T*, B*, **
RECORD LENGTH ......... :  00000        1-5 NUMERIC CHARAC.
REMOTE DSN (FTP) ...... :
TYPE/STRUCTURE/MODE FTP :  E/A/I/*,F/R/*,B/S/*
STORE UNIQUE (FTP) .... :  Y/N              FA :   Y/N
OPTION : VIEW UPD : 98/03/13 14:59 pla
-ENTER- NEXT FIELD -F3- CANCEL -F8- COMPLETION
```

```
C:X for SWFTNet 141-1-V2.00 -------------- FILES DIRECTORY ------------------------ TOM1
OPTION ===>
SYMBOLIC NAME        :     SWIFTPUT     DEFINITION : D DIRECTION : R
TRANSMISSION :
START EXIT ............ : ............
START COMMAND ......... : ............
END EXIT .............. : ............
END COMMAND ........... : ............
RECEPTION :
START EXIT ............ : ............
START COMMAND ......... : ............
END EXIT .............. : ............
END COMMAND ........... : SWIFTPUT....
DO YOU WANT TO GO ON ? UPD : 19980722112010 C:E 140-1
-ENTER- NEXT FIELD -F3- CANCEL -F8- COMPLETION
```

SWIFTGET

Le fichier **SWIFTGET** est défini en réception, le Back-office est émetteur, le fichier émis peut être vide. Les champs *RECEIVING PARTNER* et *TRANSMITTING PARTNER* pointent sur la liste des Back-offices. Le champ *RECEPTION END COMMAND* est initialisé à SWIFTGET. SWIFTGET lance une command ft-get à FTI puis lance SWIFTFWD ou SWIFTNAK. Le format d'enregistrement est fixe ou variable (*****) en fonction de l'information reçue du Back Office émetteur, la taille d'enregistrement sera indiquée aussi par le Back Office (**00000**).

Definition de Fichier Swiftget

```
C:X for SWFTNet 141-1-V2.00 -------------- FILES DIRECTORY ------------------------ TOM1
OPTION ===>
SYMBOLIC NAME :              SWIFTGET
INITIALIZATION STATUS . :    E           E:ENABLE H:DISABLE
DIRECTION ..... :            R           T:TRANSMIT R:RECEIVE *:EITHER
RECEIVING PARTNER ..... :    #CENTRAL    'NAME',#LISTE, $$ALL$$
TRANSMITTING PARTNER .. :    #CENTRAL    'NAME',#LISTE, $$ALL$$
PRIORITY ..... :             0           0:URGENT 1:FAST 2:NORMAL
DEFINITION TYPE ..... :      D           D:DYNAMIC F:FIXED
PRESENTATION TABLE .... :1   5           1 -> 9 PRESENTATION TABLE
PARAMETER CARDS FILE :       N           Y/N
SPACE TO RESERVE ..... :     N           Y/N
ALLOCATION RULE ..... :      0           0:INDIF., 1:PREALL., 2:TO CREATE
PHYSICAL NAME ..... :        $TOM_DIR/in/dummy.tmp
RECORD FORMAT ..... :        T*          TF, TV, BF, BU, T*, B*, **
RECORD LENGTH ..... :        00000       1-5 NUMERIC CHARAC.
REMOTE DSN (FTP) ..... :
TYPE/STRUCTURE/MODE FTP :                E/A/I/*,F/R/*,B/S/*
STORE UNIQUE (FTP) .... :                Y/N         FA :   Y/N
OPTION : VIEW UPD : 98/03/13 14:59 pla
-ENTER- NEXT FIELD -F3- CANCEL -F8- COMPLETION
```

```
C:X for SWFTNet 141-1-V2.00 -------------- FILES DIRECTORY ------------------------ TOM1
OPTION ===>
SYMBOLIC NAME        :     SWIFTGET      DEFINITION : D DIRECTION : R
TRANSMISSION :
START EXIT ..... : .....
START COMMAND ..... : .....
END EXIT ..... : .....
END COMMAND ..... : .....
RECEPTION :
START EXIT ..... : .....
START COMMAND ..... : .....
END EXIT ..... : .....
END COMMAND ..... : SWIFTGET....
DO YOU WANT TO GO ON ? UPD : 19980722112010 C:E 140-1
-ENTER- NEXT FIELD -F3- CANCEL -F8- COMPLETION
```

SWIFTSTO

Le fichier **SWIFTSTO** est défini en réception, le Back-office est émetteur. Les champs *RECEIVING PARTNER* et *TRANSMITTING PARTNER* pointent sur la liste des Back-offices. Le champ *RECEPTION END COMMAND* est initialisé à SWIFTSTO. SWIFTSTO met le fichier à disposition dans le bon répertoire pour demande de réception entrante ultérieure. Le format d'enregistrement est fixe ou variable (*****) en fonction de l'information reçue du Back Office émetteur, la taille d'enregistrement sera indiquée aussi par le Back Office (**00000**).

Definition de Fichier Swiftsto

```
C:X for SWFTNet 141-1-V2.00 -------------- FILES DIRECTORY ------------------------ TOM1
OPTION ===>
SYMBOLIC NAME :              SWIFTSTO
INITIALIZATION STATUS . :    E           E:ENABLE H:DISABLE
DIRECTION ..... :            R           T:TRANSMIT R:RECEIVE *:EITHER
RECEIVING PARTNER ..... :    #CENTRAL    'NAME',#LISTE, $$ALL$$
TRANSMITTING PARTNER .. :    #CENTRAL    'NAME',#LISTE, $$ALL$$
PRIORITY ..... :             0           0:URGENT 1:FAST 2:NORMAL
DEFINITION TYPE ..... :      D           D:DYNAMIC F:FIXED
PRESENTATION TABLE .... :    1           1 -> 9 PRESENTATION TABLE
PARAMETER CARDS FILE :       N           Y/N
SPACE TO RESERVE ..... :     N           Y/N
ALLOCATION RULE ..... :      0           0:INDIF., 1:PREALL., 2:TO CREATE
PHYSICAL NAME ..... :        $TOM_DIR/in/sto_&REQNUMB.tmp
RECORD FORMAT ..... :        T*          TF, TV, BF, BU, T*, B*, **
RECORD LENGTH ..... :        00000       1-5 NUMERIC CHARAC.
REMOTE DSN (FTP) ..... :
TYPE/STRUCTURE/MODE FTP :                E/A/I/*,F/R/*,B/S/*
STORE UNIQUE (FTP) .... :                Y/N         FA :   Y/N
OPTION : VIEW UPD : 98/03/13 14:59 pla
-ENTER- NEXT FIELD -F3- CANCEL -F8- COMPLETION
```

```
C:X for SWFTNet 141-1-V2.00 -------------- FILES DIRECTORY ------------------------ TOM1
OPTION ===>
SYMBOLIC NAME        :    SWIFTSTO      DEFINITION : D DIRECTION : R
TRANSMISSION :
START EXIT ..... : .....
START COMMAND ..... : .....
END EXIT ..... : .....
END COMMAND ..... : .....
RECEPTION :
START EXIT ..... : .....
START COMMAND ..... : .....
END EXIT ..... : .....
END COMMAND ..... : SWIFTSTO....
DO YOU WANT TO GO ON ? UPD : 19980722112010 C:E 140-1
-ENTER- NEXT FIELD -F3- CANCEL -F8- COMPLETION
```

SWIFTINQ

Le fichier SWIFTINQ est défini en réception, le Back Office est émetteur. Les champs RECEIVING PARTNER et TRANSMITTING PARTNER pointent sur la liste des Back Offices. Le champ RECEPTION END COMMAND est initialisé à SWIFTINQ. La commande SWIFTINQ place le fichier dans le bon répertoire et signale la bonne fin du transfert. Le format d'enregistrement est fixe ou variable (*) en fonction de l'information reçue du Back Office émetteur, la taille d'enregistrement sera indiquée aussi par le Back Office (00000).

Definition de Fichier Swiftinq

```
C:X for SWFTNet 141-1-V2.00 -------------- FILES DIRECTORY ------------------------- TOM1
OPTION ===>
SYMBOLIC NAME :              SWIFTINQ
INITIALIZATION STATUS . :    E           E:ENABLE H:DISABLE
DIRECTION ..... :            R           T:TRANSMIT R:RECEIVE *:EITHER
RECEIVING PARTNER ..... :    #CENTRAL    'NAME',#LISTE, $$ALL$$
TRANSMITTING PARTNER .. :    #CENTRAL    'NAME',#LISTE, $$ALL$$
PRIORITY ..... :             0           0:URGENT 1:FAST 2:NORMAL
DEFINITION TYPE ..... :      D           D:DYNAMIC F:FIXED
PRESENTATION TABLE .... :    1           1 -> 9 PRESENTATION TABLE
PARAMETER CARDS FILE :       N           Y/N
SPACE TO RESERVE ..... :     N           Y/N
ALLOCATION RULE ..... :      0           0:INDIF., 1:PREALL., 2:TO CREATE
PHYSICAL NAME ..... :        $TOM_DIR/in/inq_&REQNUMB.tmp
RECORD FORMAT ..... :        T*          TF, TV, BF, BU, T*, B*, **
RECORD LENGTH ..... :        00000       1-5 NUMERIC CHARAC.
REMOTE DSN (FTP) ..... :
TYPE/STRUCTURE/MODE FTP :                E/A/I/*,F/R/*,B/S/*
STORE UNIQUE (FTP) .... :                Y/N         FA :   Y/N
OPTION : VIEW UPD : 98/03/13 14:59 pla
-ENTER- NEXT FIELD -F3- CANCEL -F8- COMPLETION
```

```
C:X for SWFTNet 141-1-V2.00 -------------- FILES DIRECTORY ------------------------- TOM1
OPTION ===>
SYMBOLIC NAME        :     SWIFTINQ      DEFINITION : D DIRECTION : R
TRANSMISSION :
START EXIT ..... : .....
START COMMAND ..... : .....
END EXIT ..... : .....
END COMMAND ..... : .....
RECEPTION :
START EXIT ..... : .....
START COMMAND ..... : .....
END EXIT ..... : .....
END COMMAND ..... : SWIFTINQ....
DO YOU WANT TO GO ON ? UPD : 19980722112010 C:E 140-1
-ENTER- NEXT FIELD -F3- CANCEL -F8- COMPLETION
```

SWIFTFWD

Le fichier **SWIFTFWD** est défini en émission, le Back-office est récepteur. Les champs *RECEIVING PARTNER* et *TRANSMITTING PARTNER* pointent sur la liste des Back-offices. Le champ *TRANSMISSION END COMMAND* est initialisé à CLEARFWD. La commande CLEARFWD détruit les objets temporaires devenus inutiles. Le format d'enregistrement est variable (**V**), la taille d'enregistrement de 8192 permet de supporter les tailles habituelles.

Definition de Fichier Swiftfwd

```
C:X for SWFTNet 141-1-V2.00 -------------- FILES DIRECTORY ------------------------ TOM1
OPTION ===>
SYMBOLIC NAME :              SWIFTFWD
INITIALIZATION STATUS . :    E           E:ENABLE H:DISABLE
DIRECTION ..... :            T           T:TRANSMIT R:RECEIVE *:EITHER
RECEIVING PARTNER ..... :    #CENTRAL    'NAME',#LISTE, $$ALL$$
TRANSMITTING PARTNER .. :    #CENTRAL    'NAME',#LISTE, $$ALL$$
PRIORITY ..... :             0           0:URGENT 1:FAST 2:NORMAL
DEFINITION TYPE ..... :      D           D:DYNAMIC F:FIXED
PRESENTATION TABLE .... :    1           1 -> 9 PRESENTATION TABLE
PARAMETER CARDS FILE :       N           Y/N
SPACE TO RESERVE ..... :     N           Y/N
ALLOCATION RULE ..... :      0           0:INDIF., 1:PREALL., 2:TO CREATE
PHYSICAL NAME ..... :        $SAG_RCV
RECORD FORMAT ..... :        TV          TF, TV, BF, BU, T*, B*, **
RECORD LENGTH ..... :        08192       1-5 NUMERIC CHARAC.
REMOTE DSN (FTP) ..... :
TYPE/STRUCTURE/MODE FTP :                Y/N         FA :   Y/N
STORE UNIQUE (FTP) .... :
OPTION : VIEW UPD : 98/03/13 14:59 pla
-ENTER- NEXT FIELD -F3- CANCEL -F8- COMPLETION
```

```
C:X for SWFTNet 141-1-V2.00 -------------- FILES DIRECTORY ------------------------ TOM1
OPTION ===>
SYMBOLIC NAME        :     SWIFTFWD     DEFINITION : D DIRECTION : R
TRANSMISSION :
START EXIT ..... : .....
START COMMAND ..... : .....
END EXIT ..... : .....
END COMMAND ..... : CLEARFWD....
RECEPTION :
START EXIT ..... : .....
START COMMAND ..... : .....
END EXIT ..... : .....
END COMMAND ..... : .....
DO YOU WANT TO GO ON ? UPD : 19980722112010 C:E 140-1
-ENTER- NEXT FIELD -F3- CANCEL -F8- COMPLETION
```

SWIFTACK

Le fichier **SWIFTACK** est défini en émission, le Back-office est récepteur. Les champs *RECEIVING PARTNER* et *TRANSMITTING PARTNER* pointent sur la liste des Back-offices. Le champ *TRANSMISSION END COMMAND* est initialisé à CLEARACK. La commande CLEARACK détruit les objets temporaires devenus inutiles. Le format d'enregistrement est variable (**V**) en fonction de l'information fournie avec la requête de transfert construite par le programme SWIFTACK, la taille d'enregistrement de 8192 permet de supporter les tailles habituelles.

Definition de Fichier Swiftack

```
C:X for SWFTNet 141-1-V2.00 -------------- FILES DIRECTORY ------------------------ TOM1
OPTION ===>
SYMBOLIC NAME :              SWIFTACK
INITIALIZATION STATUS . :    E           E:ENABLE H:DISABLE
DIRECTION ..... :            T           T:TRANSMIT R:RECEIVE *:EITHER
RECEIVING PARTNER ..... :    #CENTRAL    'NAME',#LISTE, $$ALL$$
TRANSMITTING PARTNER .. :    #CENTRAL    'NAME',#LISTE, $$ALL$$
PRIORITY ..... :             0           0:URGENT 1:FAST 2:NORMAL
DEFINITION TYPE ..... :      D           D:DYNAMIC F:FIXED
PRESENTATION TABLE .... :    1           1 -> 9 PRESENTATION TABLE
PARAMETER CARDS FILE :       N           Y/N
SPACE TO RESERVE ..... :     N           Y/N
ALLOCATION RULE ..... :      0           0:INDIF., 1:PREALL., 2:TO CREATE
PHYSICAL NAME ..... :        $SAG_XML

RECORD FORMAT ..... :         TV         TF, TV, BF, BU, T*, B*, **
RECORD LENGTH ..... :        08192       1-5 NUMERIC CHARAC.
REMOTE DSN (FTP) ..... :
TYPE/STRUCTURE/MODE FTP :                E/A/I/*,F/R/*,B/S/*
STORE UNIQUE (FTP) .... :                Y/N         FA :   Y/N
OPTION : VIEW UPD : 98/03/13 14:59 pla
-ENTER- NEXT FIELD -F3- CANCEL -F8- COMPLETION
```

```
C:X for SWFTNet 141-1-V2.00 -------------- FILES DIRECTORY ------------------------ TOM1
OPTION ===>
SYMBOLIC NAME        :     SWIFTACK      DEFINITION : D DIRECTION : R
TRANSMISSION :
START EXIT ..... : .....
START COMMAND ..... : .....
END EXIT ..... : .....
END COMMAND ..... : CLEARACK....
RECEPTION :
START EXIT ..... : .....
START COMMAND ..... : .....
END EXIT ..... : .....
END COMMAND ..... : .....
DO YOU WANT TO GO ON ? UPD : 19980722112010 C:E 140-1
-ENTER- NEXT FIELD -F3- CANCEL -F8- COMPLETION
```

SWIFTNAK

Le fichier **SWIFTNAK** est défini en émission, le Back-office est récepteur. Les champs *RECEIVING PARTNER* et *TRANSMITTING PARTNER* pointent sur la liste des Back-offices. Le champ *TRANSMISSION END COMMAND* est initialisé à CLEARNAK. La commande CLEARNAK détruit les objets temporaires devenus inutiles. Le format d'enregistrement est variable (**V**) en fonction de l'information fournie avec la requête de transfert construite par le programme SWIFTACK, la taille d'enregistrement de 8192 permet de supporter les tailles habituelles.

Definition de Fichier Swiftnak

```
C:X for SWFTNet 141-1-V2.00 -------------- FILES DIRECTORY ------------------------ TOM1
OPTION ===>
SYMBOLIC NAME :              SWIFTNAK
INITIALIZATION STATUS . :    E           E:ENABLE H:DISABLE
DIRECTION ..... :            T           T:TRANSMIT R:RECEIVE *:EITHER
RECEIVING PARTNER ..... :    #CENTRAL    'NAME',#LISTE, $$ALL$$
TRANSMITTING PARTNER .. :    #CENTRAL    'NAME',#LISTE, $$ALL$$
PRIORITY ..... :             0           0:URGENT 1:FAST 2:NORMAL
DEFINITION TYPE ..... :      D           D:DYNAMIC F:FIXED
PRESENTATION TABLE .... :    1           1 -> 9 PRESENTATION TABLE
PARAMETER CARDS FILE :       N           Y/N
SPACE TO RESERVE ..... :     N           Y/N
ALLOCATION RULE ..... :      0           0:INDIF., 1:PREALL., 2:TO CREATE
PHYSICAL NAME ..... :        $SAG_LOG

RECORD FORMAT ..... :         TV         TF, TV, BF, BU, T*, B*, **
RECORD LENGTH ..... :        08192       1-5 NUMERIC CHARAC.
REMOTE DSN (FTP) ..... :
TYPE/STRUCTURE/MODE FTP :                E/A/I/*,F/R/*,B/S/*
STORE UNIQUE (FTP) .... :                Y/N         FA :   Y/N
OPTION : VIEW UPD : 98/03/13 14:59 pla
-ENTER- NEXT FIELD -F3- CANCEL -F8- COMPLETION
```

```
C:X for SWFTNet 141-1-V2.00 -------------- FILES DIRECTORY ------------------------ TOM1
OPTION ===>
SYMBOLIC NAME        :     SWIFTNAK      DEFINITION : D DIRECTION : R
TRANSMISSION :
START EXIT ..... : .....
START COMMAND ..... : .....
END EXIT ..... : .....
END COMMAND ..... : CLEARNAK....
RECEPTION :
START EXIT ..... : .....
START COMMAND ..... : .....
END EXIT ..... : .....
END COMMAND ..... : .....
DO YOU WANT TO GO ON ? UPD : 19980722112010 C:E 140-1
-ENTER- NEXT FIELD -F3- CANCEL -F8- COMPLETION
```

Utilisation de l'Interface FTI

L'interface FTI doit être configurée pour déclencher l'un des programmes de Connect:Express Unix for SWIFTNet dans le cas où une demande externe est reçue : contrôle d'accès (ConnectAccept), recherche d'un fichier disponible sur la SAG (ConnectAccept) ou déclenchement d'un transfert de fichier de la SAG vers le Back-office (SWIFTFWD) .

Les programmes de Connect:Express for SWIFNet activés par FTI reçoivent en paramètre un fichier xml qui dépend du sens de transfert. Les programmes de Connect:Express for SWIFTNet préparent le même type de fichier de paramètre avant de lancer une commande à FTI.

Configuration de FTI

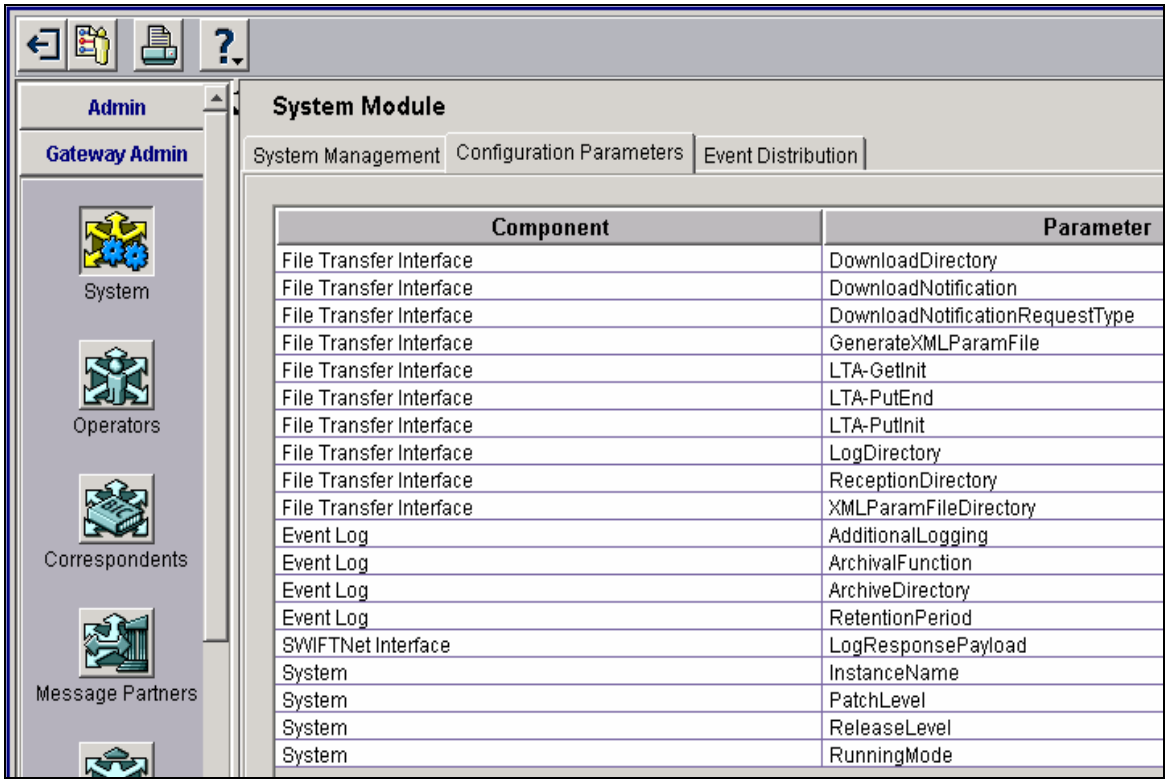
Parmi les paramètres de configuration du service de transfert de fichiers de la SAG Représentés dans le tableau ci-dessous, seuls les paramètres concernés sont explicités.

Paramètre	Valeur	Description
ScanPeriod Directory		
RetryCount		
RetryDelay		
LTA-PutInit	ConnectAcceptPutInit	Contrôle si la demande peut être exécutée
LTA-GetInit	ConnectAccept	Contrôle si si la demande peut être exécutée
LTA-PutEnd	SWIFTFWD	Transfère le fichier reçu vers le Back-office
GenerateXMLParamfile	YES	Demande l'utilisation de fichiers paramètres XML
XMLParamFile	…*FT/data/xmlparamfile/*	
Directory		
ReceptionDirectory	…*FT/data/reception/*	Répertoire de stockage des fichiers reçus par FTI
DownloadDirectory	…*FT/data/download/*	Répertoire de stockage des fichiers à dispositi on
SuccessDirectory		
ErrorDirectory		
UnknownDirectory		
LogDirectory	…*FT/data/log/*	Répertoire de stockage des fichiers de logging de FTI
ArchiveDirectory		
Paramètre	**Valeur**	**Description**
RetentionPeriod		
ArchivalFunction		
DownloadNotification	YES	Demande la génération de Notifications pour les récupérations de fichier .
DownloadNotification RequestType		Indique le type de requête dans la cas où la notification est demandée

Des exemples d'écrans de configuration de FTI sont Représentés dans ce qui suit.

L'interface d'administration de FTI proposée par la Station Web de la SAG permet d'accéder aux différents paramètres cités plus haut.

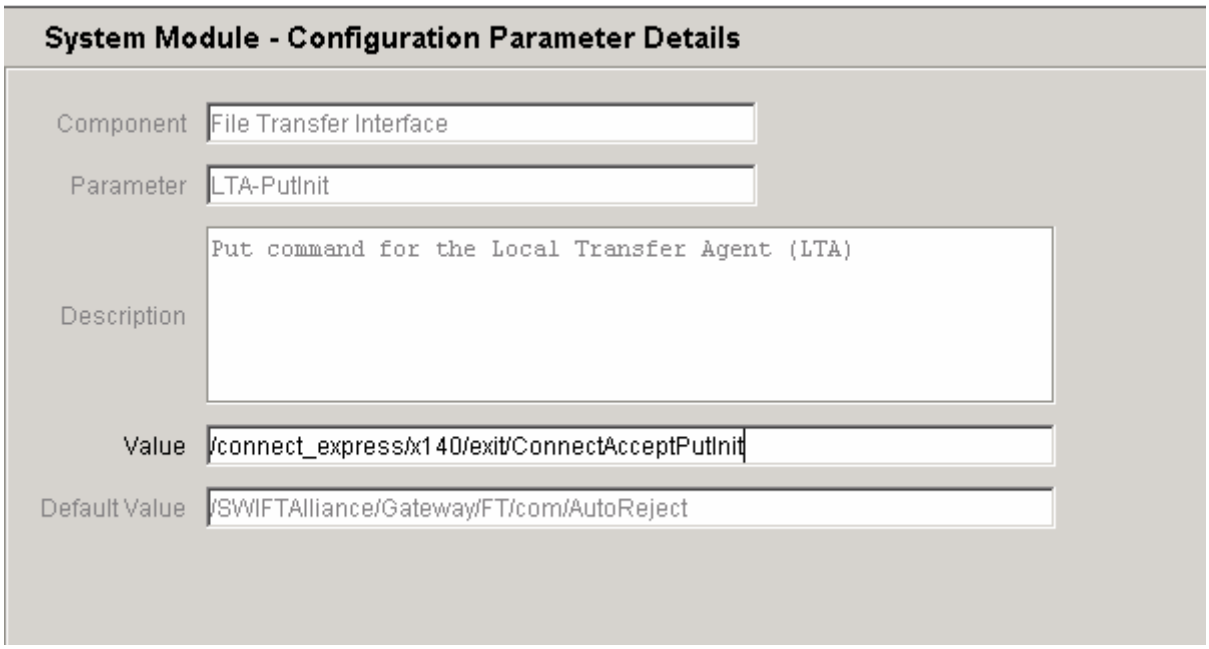
Interface d'Administration de la SAG



Paramétrage du Traitement du Put Entrant : Contrôle d'Accès.

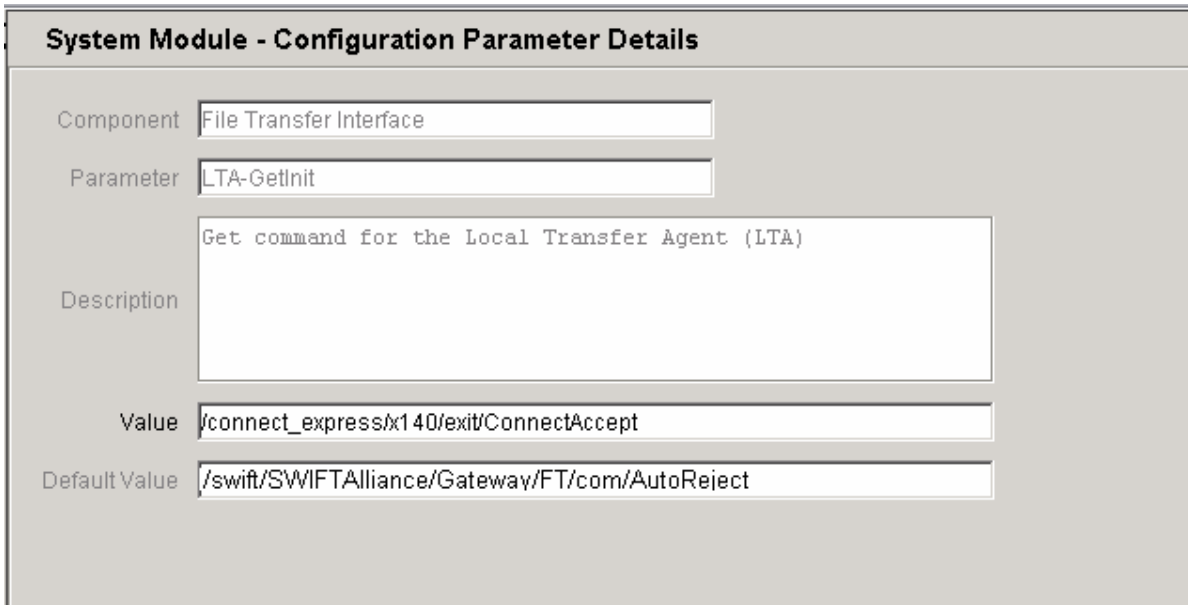
Le paramètre LTA-PutInit peut prendre trois valeurs :

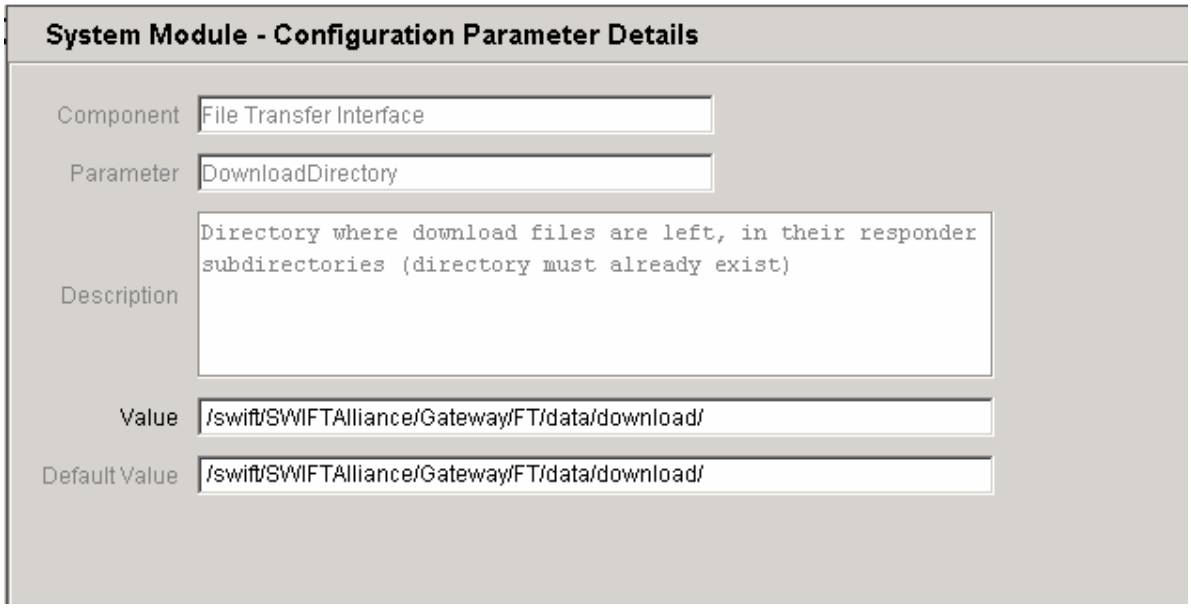
- Autoaccept : tous les appels sont acceptés.
- AutoReject : tous les appels sont rejetés.
- ConnectAcceptPutInit : ConnectAcceptPutInit effectue le contrôle d'accès .



Paramétrage du Traitement du Get Entrant : Contrôle d'Accès et Recherche du Fichier.

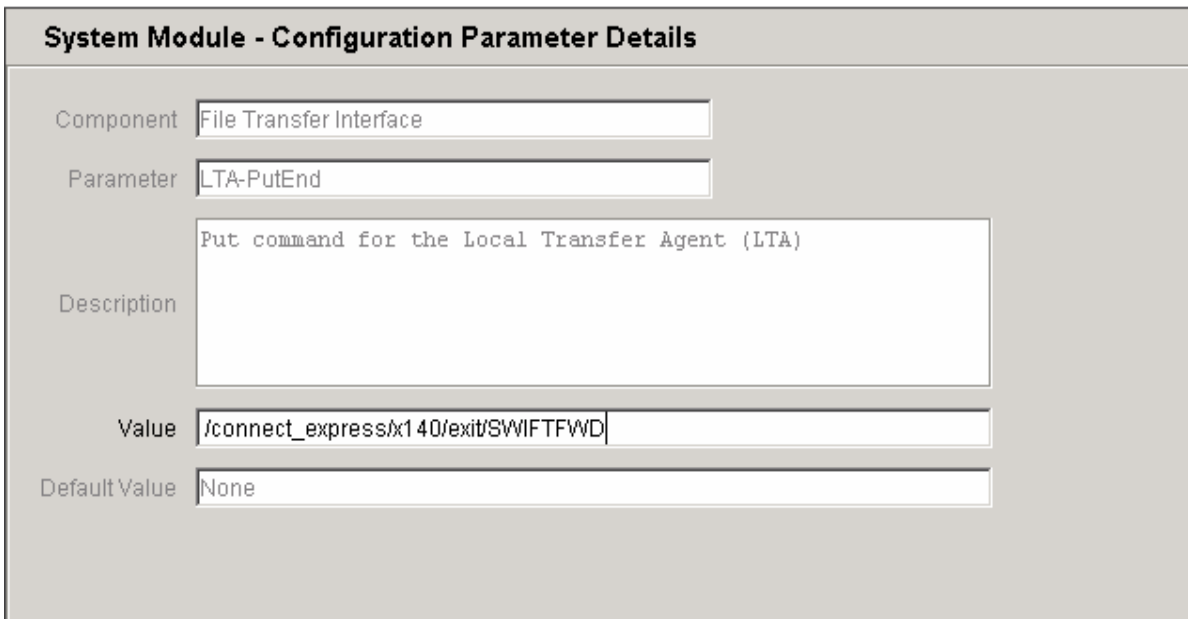
Le paramètre LTA-GetInit peut avoir la valeur ConnectAccept ou ConnectAcceptInquiry si la demande peut être routée vers le Back Office. ConnectAccept effectue le contrôle d'accès, puis recherche le fichier dans les répertoires de download de la SAG. Si le fichier n'est pas présent, la demande est rejetée. La valeur AutoReject du paramètre LTA-GetInit signifie que tous les appels sont rejetés.


System Module - Configuration Parameter Details
Component
File Transfer Interface
Parameter
LTA-GetInit
Description
Get command for the Local Transfer Agent (LTA)
Value
/connect_express/x140/exit/ConnectAccept
Default Value
/swift/SWIFTAlliance/Gateway/FT/com/AutoReject


System Module - Configuration Parameter Details
Component
File Transfer Interface
Parameter
DownloadDirectory
Description
Directory where download files are left, in their responder subdirectories (directory must already exist)
Value
/swift/SWIFTAlliance/Gateway/FT/data/download/
Default Value
/swift/SWIFTAlliance/Gateway/FT/data/download/

Paramétrage du Routage vers le Back Office

Le paramètre LTA-PutEnd doit avoir la valeur SWIFTFWD pour qu'un fichier reçu par FTI soit remis à Connect:Express for SWIFTNet et pour que le routage soit activé selon la règle prévue.



Les Paramètres FTI

Le mécanisme de routage est géré par Connect:Express Unix for SWIFTNet au travers des fichiers de paramètres FTI. Pour cette raison le paramètre *GenerateXMLParamFile* de FTI doit être positionné à YES. Les commandes ft-put et ft-get sont passées vers FTI avec les paramètres nécessaires stockés dans un fichier xml $SAG_XML/*Filename.TransferRef.par* construit à partir du squelette fourni $TOM_DIR/exit/SwiftPut.xml ou SwiftGet.xml. Ces deux squelettes doivent être configurés comme indiqué au paragraphe '*Personnalisation des squelettes xml*'.

Les programmes lancés par FTI reçoivent en entrée le nom du fichier de paramètre xml créé par FTI comme dans les deux exemples ci-dessous :

SWIFTFWD –a17 $SAG_XML/Filename.TransferRef.par

ConnectAccept –a17 $SAG_XML/Filename.TransferRef.par

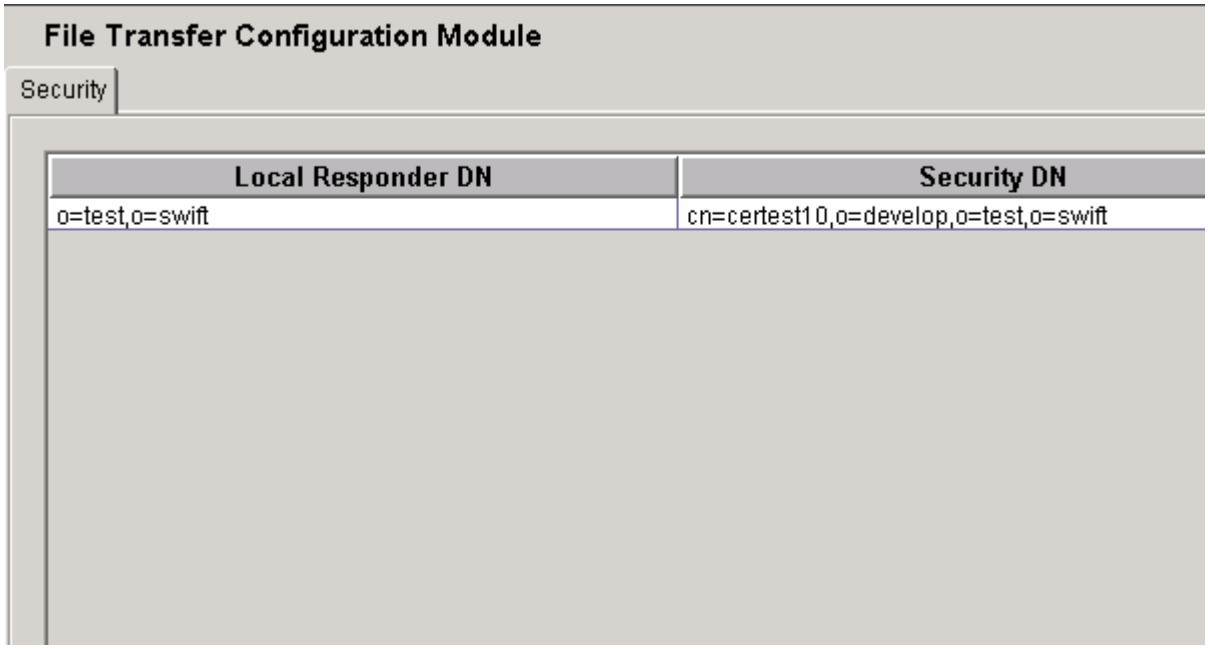
Voir la description du fichier au paragraphe '*Fichier de paramètres FTI*'' du Chapitre 3.

Définition des Profils de Sécurité

Une fois la configuration terminée et avant de commencer les premiers tests, il est nécessaire de vérifier que le profil de sécurité qui sera utilisé par le Local Responder DN est bien associé à un Security DN dans la rubrique <Transfer Module>

Cliquer sur l'icône :

Définir la correspondance comme le montre l'exemple ci-dessous :



Chapitre 3

Personnalisation des Fichiers de Contrôle

Ce chapitre explique le rôle et l'utilisation des fichiers de contrôle et propose une méthode pour leur personnalisation.

Généralités

Le mécanisme de routage est effectué sous le contrôle de fichiers personnalisés. Il permet le contrôle d'accès, la définitions de différentes règles de routage, et le suivi de bout en bout des flux. Les fichiers de contrôle sont des fichiers xml qui peuvent être modifiés et dont la syntaxe peut être validée par des utilitaires fournis. Toute erreur de syntaxe détectée au moment de l'exécution provoque l'arrêt des opérations.

Des fichiers de contrôle par défaut sont fournis (voir Annexe A) et correspondent à peu de chose prêt au mécanisme de routage assuré par la version 1 de Connect:Express for SWIFTNet.

Les fichiers de contrôle permettent de définir des règles de routage adaptées et de déterminer des critères de sélection utilisés pour contrôler les demandes et traiter la règle correspondant au contexte courant. Les règles de routage doivent être préalablement établies au cours d'une étude faite à partir des contraintes que l'utilisateur aura identifiées. Connect:Express for SWIFTNet prendra en charge les flux selon les souhaits de l'utilisateur en s'adaptant au plus prêt aux contraintes applicatives.

Pour pouvoir définir les règles il est nécessaire de considérer en premier lieu le traitement des paramètres de transfert FTI :

- Connect:Express for SWIFNet utilise le champ Pi99 du protocole PeSIT pour véhiculer les paramètres FTI entre le Back Office et la SAG, dans les deux sens.
- L'interaction entre Connect:Express for SWIFTNet et FTI est basée sur les paramètres FTI et sur la structure des fichiers dans lesquels ils sont stockés.
- Les critères de sélection utilisés pour le contrôle d'accès et la recherche de la règle de routage sont basés sur les paramètres FTI.

La connaissance plus précise des fichiers de contrôle et des mécanismes de sélection permettra d'affiner l'étude.

L'étude devra déterminer d'une part les informations dont l'application a besoin pour maîtriser le contrôle des flux et, d'autre part, les informations à fournir à Connect:Express for SWIFTNet, dans ses fichiers de contrôle, pour la gestion des flux.

Fichier de Paramètres FTI

Chaque transfert est associé à un fichier xml de paramètres FTI, stocké dans le répertoire /exit de Connect:Express for SWIFTNet et détruit systématiquement à la fin des opérations. Certains paramètres font l'objet d'un traitement automatique par Connect:Express for SWIFTNet. Tout autre traitement est déterminé par la personnalisation des squelettes et du Pi99, ou de la règle de routage (voir le paragraphe '*Utilisation du protocole PeSIT* ') et par la personnalisation du contrôle d'accès et de la détermination de la règle de routage (voir le paragraphe '*Utilisation des Paramètres FTI* ').

Le tableau ci-dessous répertorie la liste des paramètres FTI et décrit les traitements effectués par Connect:Express for SWIFTNet. La première colonne indique le numéro de paramètre selon les spécifications de FTI, la seconde colonne indique le nom du paramètre selon les spécifications de FTI, la troisième colonne définit le paramètre et la dernière colonne indique si Connect:Express traite le paramètre automatiquement ou d'après les fichiers de contrôle personnalisés:

P	Name	Description	Traitement
-a1	RequestorDN	Identification locale Exemple : o=*Localname*,o=*swift*	Selon personnalisation *Dépôt et recherche du fichier de données sur la SAG dans le cas d'une demande de download*
-a2	ResponderDN	Identification du distant Exemple : o=test,o=swift	Selon personnalisation *Dépôt et recherche du fichier de données sur la SAG dans le cas d'une demande de download*
-a3	ServiceName	Service supporté par le Responder Exemple: swift.sft!x	Selon personnalisation
-a4	RequestType	Request type supporté par le Responder	Selon personnalisation
-a5	SecurityDN	Authoriser DN and Signer DN. Exemple: cn=chantal,o=bredfrpp,o=swift	Selon personnalisation
-a6	TransferDescription	Informations utilisateur	Selon personnalisation
-a7	TransferInfo	Compression (Get) et autres informations.	Selon personnalisation *Exécution de la dé-compression (Get sortant)*
-a8	UserRef	Informations utilisateur	Selon personnalisation
-a9	FileLocation	Nom physique de fichier	Selon personnalisation *Accès au fichier de données sur la SAG*
-a10	FileName	Nom du fichier de données Exemple: releve	Selon personnalisation *Dépôt et recherche du fichier de données sur la SAG dans le cas d'une demande de download*
-a11	FileDescription	Informations utilisateur	Selon personnalisation
-a12	FileInfo	Type de données , Compression (Put) et autres informations. Exemple: UInfo,SwCompression=ZIP	Selon personnalisation *Exécution de la compression (Put,Store)*
-a13	NonRepudiation	Exemple : FALSE	Selon personnalisation
-a14	PossibleDuplicate	Exemple : FALSE	Selon personnalisation
-a16	FileSize	Taille maximum de fichier acceptée. Exemple : Maxfile size	Selon personnalisation
-a18	DeliveryNotification	Exemple : FALSE	Selon personnalisation

P	Name	Description	Traitement
-a24	FileToken		Selon personnalisation
-a25	DeliveryNotificationRequestType		Selon personnalisation
-a26	DeliveryNotification ReceiverDN		Selon personnalisation

Utilisation du protocole PeSIT

Le protocole PeSIT fournit un champ utilisateur de 254 caractères, le Pi99, qui permet d'associer des informations applicatives à la demande de transfert de fichier.

Dans le sens Back Office vers Connect:Express for SWIFTNet ce champ est utilisé pour passer les paramètres nécessaires à l'exécution de la demande locale. Le Pi99 est construit par l'application au moment de constituer la requête de transfert sur le Back Office. L'application doit utiliser les conventions de construction conformes aux règles de syntaxe et à ses propres spécifications.

Dans le sens Connect:Express for SWIFTNet vers le Back Office, ce champ est utilisé pour rendre compte de l'exécution des demandes locales et distantes. Le Pi99 est construit par Connect:Express for SWIFTNet au moment de constituer la requête de transfert vers le Back Office. Connect:Express for SWIFTNet trouve la description de la structure à utiliser dans le fichier des règles de routage .

Pi99 Emis par le Back Office

L'application doit associer un Pi99 à chaque demande de transfert vers SWIFTNet.

Le Pi99 émis par le Back Office doit indiquer la règle de routage à utiliser et un certain nombre de valeurs qui seront dynamiquement remplacées dans les paramètres FTI de la demande. Supposons une demande d'émission envoyée par le Back Office avec les deux Pi99 équivalents représentés dans l'exemple ci-dessous :

Exemple de Pi99

Pi99	R=001**:**o=test**:**o=test2**:**filename**:**ZIP
Pi99	o=test**:**o=test2**:**filename**:**ZIP**:**R=001

Pour cette demande, le routage sera exécuté selon la règle identifiée par le numéro '001'. Les autres valeurs sont séparées par le caractère '**:**'. La règle de routage indiquera le squelette xml de

paramètres FTI à utiliser pour instancier les valeurs. Le paragraphe '*Personnalisation des Règles'* décrit dans le détail la syntaxe d'une règle de routage.

Chaque valeur est traitée en fonction de sa position et remplacera le jeton numéroté par cette position dans le squelette des paramètres FTI indiqué par la règle de routage.
Le paramètre représentant la règle n'est pas positionnel et n'est pas pris en compte dans le calcul des positions. Dans les deux exemples ci-dessus la chaîne de caractères 'o=test' est en position 1, la chaîne de caractères 'o=test2' est en position 2 et ainsi de suite.

Cette méthode permet de construire le Pi99 émis en dissociant l'ordre qui satisfait l'affichage, selon les besoins de l'application, de l'ordre des paramètres de transfert FTI.

Si le paramètre R=xxx est absent, la règle par défaut est la règle numéro 0 (R=000).

Dans l'exemple ci-dessous extrait du paragraphe '*Personnalisation des Règles'*, la règle 001 précise, par le tag <PutXmlFile>, le nom d'un fichier squelette xml utilisé pour les paramètres de la commande Put selon la syntaxe suivante.

```
<Rule Num="001" Status="Enabled" Name="REGLE001">
  <PutXmlFile>$TOM_DIR/exit/swiftput01.xml</PutXmlFile>
```

Le fichier de la commande Ft-Put passée à FTI sera construit à partir du squelette swiftput01.xml. Le jeton numéro <1> sera remplacé par la chaîne de caractères 'o=test' qui est en position 1, le jeton numéro <2> sera remplacé par la chaîne de caractères 'o=test2', en position 2, et ainsi de suite…. comme dans l'exemple ci-dessous :

Exemple de Squelette Xml

```
<Ft:PutFileRequest>....
<a1>o=bankbebb,o=swift</a1>
<a2><2>,o=swift</a2>
<a3>swift.sft</a3>
<a4></a4>
<a5>o=bankbebb,o=swift</a5>
<a6><1></a6>
<a7>priority=high</a7>
<a8><3></a8>
<a9></a9>
<a10>payments1</a10>
<a11>payments 017019 batch 1</a11>
<a12>SwCompression=<4>;coding=ebcdic</a12>
<a13>TRUE</a13>
<a14>FALSE</a14>
<a15></a15>
<a18>TRUE</a18>
<a24></a24>
<a25></a25>
<a26></a26>
</Ft:PutFileRequest>
```

Après remplacement des jetons, le fichier passé à la commande Ft-Put sera le suivant :

```
<Ft:PutFileRequest>....
<a1>o=bankbebb,o=swift</a1>
<a2>o=test2,o=swift</a2>
<a3>swift.sft</a3>
<a4></a4>
<a5>o=bankbebb,o=swift</a5>
<a6>o=test</a6>
<a7>priority=high</a7>
<a8>filename</a8>
<a9>/test/cexpress/x141/put_03100004.tmp</a9>
<a10>payments1</a10>
<a11>payments 017019 batch 1</a11>
<a12>SwCompression=ZIP;coding=ebcdic</a12>
<a13>TRUE</a13>
<a14>FALSE</a14>
<a15></a15>
<a18>TRUE</a18>
<a24></a24>
<a25></a25>
<a26></a26>
</Ft:PutFileRequest>
```

Pi99 Emis par Connect:Express for SWIFTnet

Connect:Express for SWIFTNet doit associer un Pi99 à chaque transfert vers le Back Office. Les transferts vers le Back Office ont lieu soit pour l'acquittement d'une demande locale, soit pour remonter une demande externe. Dans les deux cas une règle de routage a d'abord été selectionnée soit d'après le numéro de règle passé par le Pi99 envoyé par le Back Office (demande locale), soit selon une recherche décrite au paragraphe '*Détermination de la Règle de Routage'* (demande externe) .

Le Pi99 émis par Connect:Express for SWIFTNet a la structure indiquée par la règle de routage utilisée. L'exemple ci-dessous donne un aperçu de la syntaxe utilisée et des possibilités offertes pour envoyer sur le Back Office une structure de Pi99 qui répondent à des besoins particuliers d'affichage et de suivi de bout en bout. La syntaxe d'une règle de routage est décrite en détail au paragraphe '*Personnalisation des Règles*'.

Quatre structures de Pi99 peuvent être définies, en fonction du sens de transfert (Put ou Get) et du sens de la demande (In ou Out). Le Pi99 est construit à partir des paramètres FTI (tout ou partie) et de chaînes de caractères constantes.

```
<Rule Num="001" Status="Enabled" Name="REGLE001">
  ……
  <Pi99Delim>:</Pi99Delim>
    …
  <Pi99PutIn Delim=":">
  </Pi99PutIn>

  <Pi99GetIn Delim=":">
    …
  </Pi99GetIn>

  <Pi99PutOut Delim=":">
    <Field Name="Responder">
      <Param Delim="," SubNum="1" Length="12">a2</Param>
    </Field>
    <Field Name="Filename">
      <String>F=</String>
      <Param>a8</Param>
    </Field>
    <Field Name="TransferDescription">
      <Param Length="8">a6</Param>
    </Field>
  </Pi99PutOut>

  <Pi99GetOut Delim=":">
    ….
  </Pi99GetOut>
```

Le Pi99 envoyé au Back Office avec l'acquittement du transfert précédent sera construit de la façon suivante :

- Les champs seront séparés par le caractère ':'
- Le premier champ contiendra les douze premiers caractères du sous paramètre N° 1 du paramètre FTI <a2>
- Le second champ contiendra les caractères 'F=' et le paramètre FTI <a8> dans sa totalité
- Le troisième champ contiendra les 8 premiers caractères du paramètre FTI <a6>.

Dans l'exemple précédent, le fichier des paramètres de transfert était le suivant :

```
<Ft:PutFileRequest>....
<a1>o=bankbebb,o=swift</a1>
<a2>o=test2,o=swift</a2>
<a3>swift.sft</a3>
<a4></a4>
<a5>o=bankbebb,o=swift</a5>
<a6>o=test</a6>
<a7>priority=high</a7>
<a8>filename</a8>
<a9>/test/cexpress/x141/put_03100004.tmp</a9>
<a10>payments1</a10>
<a11>payments 017019 batch 1</a11>
<a12>SwCompression=ZIP;coding=ebcdic</a12>
<a13>TRUE</a13>
<a14>FALSE</a14>
<a15></a15>
<a18>TRUE</a18>
<a24></a24>
<a25></a25>
<a26></a26>
</Ft:PutFileRequest>
```

Le Pi99 envoyé pour rendre compte de l'émission du fichier aura la structure suivante :

Pi99	o=test2**:**F=filename**:**o=test

Utilisation des Paramètres FTI

Les paramètres FTI liés à un transfert servent à l'exécution des opérations et au suivi de bout en bout, comme on l'a vu précédemment avec la gestion du Pi99 PeSIT. Ils servent aussi, dans le cas de demandes externes, au contrôle d'accès et à la détermination de la règle de routage. L'utilisation des 'Expressions Régulières' est traitée à l' Annexe C.

Contrôle d'Accès

Lorsqu'une demande externe est reçue par la SAG, FTI remet cette demande à l'un des programmes ConnectAccept paramétrés dans sa configuration selon la nature de la demande. La première fonction de ces programmes est de traiter les fichiers de contrôle WhiteList.xml et BlackList.xml s'ils existent. Ces fichiers conditionnent l'acceptation de la demande et déterminent les droits associés à cette demande. Le fichier Whitelist.xml détermine les demandes autorisées et les droits d'accès qui leurs sont attribués, le fichier BlackList.xml détermine les demandes non autorisées ou limite leurs droits d'accès. Voir la description de la gestion des accès au paragraphe *'Personnalisation du Contrôle d'Accès'* .

L'exemple ci-dessous donne un aperçu de la syntaxe et des possibilités offertes par le fichier de contrôle WhiteList. La syntaxe est décrite en détail au paragraphe *'Personnalisation du Contrôle d'Accès'*.

Syntaxe d'une Access List

```
<AccessList>

<Filter Name="filter1" AccessMode="IP" Logic="And" Status="Enabled">
<Select Parser="UserProgram" FTIParam="None" Match="True"  CaseSensitive="True">selectionP</Select>
<Select Parser="Regex" FTIParam="RequestorDN" Match="True" CaseSensitive="False">^o=tes.*,.*</Select>
</Filter>

<Filter Name="filter2" AccessMode="IGP" Logic="And" Status="Enabled">
<Select Parser="Regex" FTIParam="ResponderDN" Match="True" CaseSensitive="False">.*,o=swift</Select>
<Select Parser="Regex" FTIParam="RequestorDN" Match="True" CaseSensitive="False">O=ban.*,.*</Select>
</Filter>

</AccessList>
```

Chaque 'Filtre' précise les droits d'accès (Put, Get), le type de logique booléenne utilisée, le programme de contrôle et le paramètre FTI à tester. Le premier filtre répondant à la demande est pris en compte.

Dans cet exemple le filtre 'filter1' autorise l'accès en écriture (AccessMode=P) et en lecture (AccessMode=I). Les critères de sélection doivent être tous satisfaits (Logic=and). Le premier critère de sélection est testé par un programme utilisateur (Parser=UserProgram) qui doit répondre positivement (Match=True)à la demande (il reçoit le nom du fichier contenant les paramètres FTI en entrée). Le second critère de sélection utilise les expressions régulières (Parser=Regex) et doit

vérifier positivement (Match=True) le masque '^o=tes.*,.*' dans le paramètre FTIparam= RequestorDN.

Le champ FTIparam= peut prendre toutes les valeurs existantes de nom de paramètre FTI. Les champs Logic= et Match= permettent de construire toutes les combinaisons booléennes possibles. Les expressions régulières sont utilisées dans les masques appliqués aux champs.

Détermination de la Règle de Routage

Lorsqu'une demande externe est reçue, si le contrôle d'accès est positif, il est nécessaire de déterminer les conditions de routage. C'est le rôle du programme ConnectAccept sélectionné, selon la nature de la demande, par FTI dans sa configuration. Il lance une recherche dans le fichier FindRule.xml à partir des paramètres FTI courants.
L'exemple ci-dessous donne un aperçu de la syntaxe et des possibilités offertes par le fichier de contrôle FindRule. La syntaxe est décrite en détail au paragraphe *'Personnalisation de la Recherche des Règles'*.

Détermination de la Règle de Routage

```
<FindRuleList>

<FindRule Num="000" Status="Enabled">
<Select Action="Include" FTIParam="ResponderDN">.*,o=swift$</Select>
<Select Action="Include" FTIParam="RequestorDN">^o=tes.*,.*</Select>
<Select Action="Exclude" FTIParam="RequestorDN">^o=.*,ex.*</Select>
</FindRule>
<FindRule Num="001" Status="Enabled">
<Select Action="Include" FTIParam="RequestorDN">^o=bank0.*,.*</Select>
</FindRule>

</FindRuleList>
```

Le premier numéro de règle correspondant au contexte courant déterminera la règle de routage à utiliser : les règles sont décrites dans le fichier RuleList.xml. La sélection du numéro de règle est faite à partir de critères de sélection appliqués aux paramètres FTI courants.

Dans cet exemple la règle numéro 000 est déterminée en fonction des paramètres ResponderDN et RequestorDN. Seuls les ResponderDN dont le second champ est égal à 'o=swift' sont retenus. Seuls les RequestorDN dont le premier champ commence par 'o=tes' sont retenus. Parmi les demandes retenues, celles dont le RequestorDN satisfait à la dernière condition sont exclues.

La règle numéro 001 est utilisée pour tous les ResquestorDn commençant par 'o=bank0'.

La sélection peut considérer chaque paramètre existant dans la liste des paramètres FTI. Les expressions régulières sont utilisées dans les masques appliqués aux champs.

Personnalisation des Squelettes XML

Le passage des paramètres entre Connect:Express et FTI se fait par des fichiers XML, dont la structure fait partie des spécifications de l'interface de FTI. Ces fichiers XML sont créés par Connect:Express ou par FTI selon que le transfert est initialisé par le Back-office ou par un partenaire distant.

Pour les transferts initialisés par le Back-office, Connect:Express crée un fichier XML par transfert, à partir d'un des squelettes fournis dans le répertoire /exit. Ce fichier est détruit automatiquement à la fin des opérations Trois squelettes sont fournis, l'un pour envoyer un fichier (put), le second pour recevoir un fichier (get) le dernier pour mettre un fichier à disposition (sto). Le paragraphe '*Pi99 Emis par le Back Office*' explique le principe d'utilisation de ces fichiers xml.

Les fichiers squelettes doivent être personnalisés. Ils peuvent contenir des valeurs constantes et des valeurs variables qui sont résolues par Connect:Express Unix for SWIFTNet avant de transmettre la demande de transfert à FTI. Les valeurs constantes peuvent être personnalisées et ne sont pas modifiées au cours des opérations. Les variables sont résolues à partir des informations procurées par le Back-office au moment de la requête de transfert vers la SAG (dans le Pi99, champ libre PeSIT). Connect:Express for SWIFTNet ajoute aussi des informations dans certains champs au cours des opérations. Les traitements appliqués au différents paramètres FTI sont décrits au paragraphe '*Fichiers de Paramètres FTI*'.

La personnalisation d'un squelette xml est associée à la personnalisation d'une structure de Pi99. Elle consiste à placer des jetons à la place adéquate afin de recevoir les informations du Pi99 associé.

Les fichiers squelettes fournis sont représentés ci-dessous . Ils correspondent au traitement effectué par la version 1 de Connect:Express for SWIFTNet et sont associés à la structure de Pi99 proposée dans cette version. La règle de routage est la règle par défaut (R=000).

$TOM_DIR/exit/SwiftSto.xml

Ce fichier n'est pas utilisé par FTI, il est réservé à Connect:Express for SWIFTNet.

Fichier SwiftSto.xml et Pi99 Associé

```
<Cx:StoreFile>
<RequestorDN>o=<1>,o=swift</RequestorDN>
<ResponderDN>o=<2>,o=swift</ResponderDN>
<FileName><3></FileName>
<FileInfo>SwCompression=<4></FileInfo>
</Cx:StoreFile>
```

Pi99 associé	R=000**:**o=test**:**o=test2**:**filename**:**Compression

$TOM_DIR/exit/SwiftPut.xml

Fichier SwiftPut.xml et Pi99 Associé

```
<Ft:PutFileRequest>
<a1>o=test,o=swift</a1>
<a2><1>,o=swift</a2>
<a3>TestService</a3>
<a4>RequestType</a4>
<a5>cn=certest10,o=develop,o=test,o=swift</a5>
<a6><3></a6>
<a7>TransferInfo</a7>
<a8>CENTRAL</a8>
<a9></a9>
<a10><2></a10>
<a11>File Description</a11>
<a12>SwCompression=<4>,coding=ascii</a12>
<a13>FALSE</a13>
<a14>FALSE</a14>
<a18>FALSE</a18>
</Ft:PutFileRequest>
```

Pi99 associé	R=000:o=test:filename:Reqnumb:Compression

$TOM_DIR/exit/SwiftgGet.xml

Fichier SwiftGet.xml et Pi99 Associé

```
<Ft:GetFileRequest>
<a1>o=test,o=swift</a1>
<a2><1>,o=swift</a2>
<a3>TestService</a3>
<a4>RequestType</a4>
<a5>cn=certest10,o=develop,o=test,o=swift</a5>
<a6><3></a6>
<a7>SwCompression=<4>,coding=ascii</a7>
<a8>CENTRAL</a8>
<a9></a9>
<a10><2></a10>
<a13>FALSE</a13>
<a16></a16>
<a24></a24>
</Ft:GetFileRequest>
```

Pi99 associé	R=000:o=test:filename:Reqnumb:Compression

Personnalisation du Contrôle d'Accès

Le contrôle d'accès est effectué sur les demandes externes, par les programmes ConnectAccept, ConnectAcceptPutInit et ConnectAcceptInquiry. Il traite les fichiers WhiteList.xml et BlackList.xml en séquence, s'ils existent.

Le traitement ne se limite pas au seul contrôle mais détermine aussi les droits associés à la demande : les droits concernent le sens de transfert et la recherche d'un fichier sur le Back Office dans le cas d'une récupération de fichier (voir le paragraphe '*Traitement du Download*'). Une demande acceptée après le contrôle dans la White Liste, peut être rejetée ou voir ses droits restreints après le contrôle dans la Black Liste.

La 'White liste' sans 'Black liste' détermine l'ensemble des demandes autorisées et précise les droits d'accès, en particulier le droit d'aller rechercher un fichier directement sur le Back Office. La 'Black liste' sans 'White liste' détermine l'ensemble des demandes interdites ou les restrictions à appliquer sur le sens de transfert, put ou get. Voir au paragraphe '*Gestion des droits d'accès*' comment on peut utiliser la combinaison des deux listes.

Une demande peut être non seulement caractérisée par le nom du demandeur, mais aussi par n'importe lequel des paramètres comme le nom de fichier par exemple. Toutes les combinaisons booléennes sont possibles. On peut définir un filtre comprenant un ensemble de conditions dont toutes doivent être respectées ou seulement une d'entre elles au moins. Les conditions peuvent être constituées d'une recherche sur le respect ou le non respect d'un critère.

Exemple : toute demande à destination des ResponderDN '.*,o=swift$' ou en provenance des RequestorDN '^O=tes.*,.*' est acceptée avec la seule possibilité de déposer des fichiers (Filtre filter1 dans l'exemple ci-dessous).

Les fichiers WhiteList.xml et BlackList ont la même structure représentée ci-dessous.

```
<?xml version="1.0" encoding="UTF-8"?>
<AccessListCfg xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation='/test/cexpress/x141/exit/AccessList.xsd'>

<AccessList>

<Filter Name="filter2" AccessMode="IP" Logic="And" Status="Enabled">
<Select Parser="UserProgram" FTIParam="None" Match="True"
                CaseSensitive="True">$TOM_DIR/exit/selectionProgram</Select>
<Select Parser="Regex" FTIParam="RequestorDN" Match="True"
CaseSensitive="False">^o=BANK.*,.*</Select>
</Filter>

<Filter Name="filter1" AccessMode="P" Logic="Or" Status="Enabled">
<Select Parser="Regex" FTIParam="ResponderDN" Match="True" CaseSensitive="False">.*,o=swift</Select>
<Select Parser="Regex" FTIParam="RequestorDN" Match="True" CaseSensitive="False">^O=tes.*,.*</Select>
</Filter>

</AccessList>
```

Les fichiers WhiteList.xml et BlackList.xml peuvent être personnalisés puis contrôlés avec l'utilitaire ParseAccessList qui vérifie la syntaxe sur la base du fichier AccessList.xsd. L'utilitaire se trouve

dans le répertoire /exit de Connect:Express for SWIFTNet. Le fichier xsd se trouve dans le même répertoire : c'est un fichier non modifiable. La première règle de syntaxe est que tous les tags et les paramètres sont obligatoires.

Le tableau ci-dessous définit les TAG de ce fichier :

TAG	**Description**	**Attributs**
AccessListCfg	Cette en-tête ne doit être modifiée qu'une fois à l'installation. Elle doit indiquer le nom physique complet du fichier .xsd à utiliser pour le contrôle de syntaxe.	Aucun
AccessList	Associé à son Tag de fin. Ce bloc contient l'ensemble des filtres valides	Aucun
Filter	Associé à son Tag de fin. Ce bloc contient l'ensemble des critères de sélection qui permettent d'accepter ou de rejeter la demande et la façon de les combiner. Les blocs sont traités les uns après les autres et le premier Bloc trouvé provoque l'arrêt diu traitement. La demande sera comparée aux droit d'accès fourni par le bloc sélectionné, par exemple : **P** : seule une demande de dépôt peut être acceptée. **G** : seule une demande de récupération peut être acceptée. Cette demande ne permet que de récupérer un fichier à disposition sur la SAG. **PI** : la demande peut être un dépôt ou une récupération, et, en cas de récupération, si le fichier n'est pas présent sur la SAG, il est possible d'aller le récupérer sur le Back Office **I** et **F**, en majuscules, indiquent que la requête de réception vers le Back Office est de type inquiry, **i** et **f**, en minuscules, indiquent que la requête de réception vers le Back Office est de type normal. Les valeurs I et F en majuscules sont obligatoires si le moniteur sur le Back Office n'est pas Connect:Express **Note** : En absence de 'White Liste' le droit d'accès par défaut est **PG**. En présence de 'White Liste' le droit d'accès par défaut est nul. Voir le paragraphe '*Gestion des droits d'accès*'	**Name**: valeur indicative du filtre **AccessMode**: Droits d'accès concernés. ***P*** = Dépôt d'un fichier (Put) ***G*** = Récupération sans accès au Back Office (Get) ***I*** ou ***i*** = Récupération avec possible recherche sur le Back Office ***F*** ou ***f*** = Récupération systématique sur le Back Office *Note: Les combinaisons possibles pour la 'White liste' sont P, PG, GP, PI, IP, PF, FP, G, I, F et les combinaisons valides pour la 'Black List'e sont P, G, PG et GP* **Logic**: relation booléenne entre les différents critères du bloc. ***And*** = toutes les conditions doivent être respectées ***Or*** = l'une des conditions doit être respectée au moins **Status**: état du filtre ***Enabled*** = le filtre est actif ***Disabled*** = le filtre n'est pas actif
Select	Associé à son Tag de fin. Ce bloc indique le traitement à appliquer : un programme utilisateur ou une expression régulière. Il définit une méthode de sélection qui porte sur un paramètre FTI ou sur la totalité du fichier des paramètres FTI. Cette méthode peut être un programme utilisateur ou le traitement d'une expression régulière. La condition recherchée peut être le rejet ou la vérification du critère.	**Parser** : méthode de sélection ***UserProgram*** : le programme utilisateur indiqué va vérifier la demande. Il reçoit le fichier des paramètres FTI en entrée. ***Regex*** : Le masque indiqué est une expression régulière à appliquer pour la sélection sur le paramètre indiqué par FTIparam. **FTIParam** : nom FTI du paramètre à traiter ***None*** : le traitement s'applique sur tout le fichier des paramètres FTI (seulement dans le cas où Parser="UserProgram") ***Nom de paramètre*** : paramètre sur lequel on applique le masque indiqué par ce Bloc **Match** : type de condition recherchée ***True***: la condition recherchée est un retour positif.

TAG	Description	Attributs
		False : la condition recherchée est un retour négatif. **CaseSensitive** : traitement des voyelles ou non ***True***: les caractères sont traités tels quels ***False***: les lettres sont toutes mises en majuscules pour le traitement.

Gestion des Droits d' Accès

Un système d'habilitation permet d'affiner le contrôle d'accès en ajoutant au droit de se connecter les droits de transférer et d'accéder directement au Back Office.

Les droits d'accès supportés sont les suivants :

- Le droit de se connecter
- Le droit de déposer un fichier (droit P)
- Le droit de récupérer un fichier sur la SAG (droit G)
- Le droit de récupérer un fichier sur le Back Office (droits I, F, i ou f)

Par défaut, c'est à dire sans 'White' ou 'Black Liste', les droits sont les suivants, pour tous les demandeurs :

- Le droit de se connecter
- Le droit de déposer un fichier (droit P)
- Le droit de récupérer un fichier sur la SAG (droit G)

La présence d'une 'White Liste' signifie que l'accès n'est autorisé qu'aux demandes définies dans cette liste. De la même façon seuls les droits d'accès définis dans la liste sont valides : le droit d'accès par défaut dans ce cas est nul.

Seule la 'White Liste' permet d'étendre l'accès à la recherche d'un fichier sur le Back Office,comme le montre l'exemple suivant : pour tous les demandeurs (RequestorDN) , si le fichier n'est pas présent sur la SAG, une recherche peut être lancée sur le Back Office (droit I).

La 'Black Liste' peut être utilisée pour restreindre les droits d'accès.

```
<?xml version="1.0" encoding="UTF-8"?>
<AccessListCfg xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation='/test/cexpress/x141/exit/AccessList.xsd'>

<AccessList>

<Filter Name="Access Back Office" AccessMode="IP" Logic="And" Status="Enabled">
<Select Parser="Regex" FTIParam="RequestorDN" Match="True" CaseSensitive="False">.*</Select>
</Filter>

</AccessList>
```

Les valeurs du paramètre AccessMode sont :

P = Dépôt d'un fichier (Put)
G = Récupération sans accès au Back Office (Get)
I ou **i** = Récupération avec possible recherche sur le Back Office
F ou **f** = Récupération systématique sur le Back Office

Note: Les combinaisons possibles pour la 'White liste' sont P, PG, GP, PI, IP, PF, FP, G, I, F et les combinaisons valides pour la 'Black Liste' sont P, G, PG et GP

Selon le choix d'organisation, on peut implémenter soit une White Liste pour définir toutes les demandes autorisées, soit une Black Liste pour définir toutes les demandes interdites. Mais, si l'accès au Back Office est prévu, une White Liste comme celle de l'exemple précédent est obligatoire.

La 'Black Liste' permet d'interdire l'accès pour le dépôt de fichier ou pour la récupération de fichier ou pour les deux. L'exemple suivant empêche l'accès en récupération de fichier pour tous et empêche l'accès en connexion au demandeur Bank1:

```
<?xml version="1.0" encoding="UTF-8"?>
<AccessListCfg xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation=' /test/cexpress/x141/exit/AccessList.xsd'>

<AccessList>

<Filter Name="Access Back Office 1" AccessMode="GP" Logic="And" Status="Enabled">
<Select Parser="Regex" FTIParam="RequestorDN" Match="True"
CaseSensitive="False">^o=Bank1,.*</Select>
</Filter>
<Filter Name="Access Back Office" AccessMode="G" Logic="And" Status="Enabled">
<Select Parser="Regex" FTIParam="RequestorDN" Match="True" CaseSensitive="False">.*</Select>
</Filter>

</AccessList>
```

Note : l'ordre des filtres est très important car le premier vérifié est sélectionné.

Traitement du Download

Les paragraphes précédents donnent la définition des droits d'accès dans le cas de demandes externes.
Ces droits d'accès portent sur le sens de transfert mais, dans le cas d'une récupération de fichier (Download), ils précisent si la récupération se limite aux fichiers disponibles sur la SAG ou si la récupération directe sur le Back Office est possible, voire obligatoire (accès à la dernière version d'un fichier).

Note : Si aucune liste d'accès n'est active, le droit d'accès par défaut est PG. Seule la 'White Liste permet d'ouvrir les accès I, i, F et f.

Ce paragraphe décrit dans son ensemble le mécanisme de recherche d'un fichier à disposition, y compris la mise à disposition sur la SAG par le Back Office.

Les étapes du processus sont donc les suivantes :

1. Mise à disposition du fichier sur la SAG (Facultatif)
2. Demande externe de récupération
3. Contrôle d'accès
4. Si la demande est autorisée, avec le droit de type f ou F : envoi d'une requête de réception, de type Normal ou Inquiry respectivement, vers le Back Office sur le fichier symbolique SWIFTINQ
5. Si la demande est autorisée, avec le droit de type I ou i : lancement de la recherche du fichier sur la SAG. S'il est présent réponse à la demande, s'il n'est pas disponible, envoi d'une requête de réception, de type Normal ou Inquiry respectivement, vers le Back Office sur le fichier symbolique SWIFTINQ
6. Si la demande est autorisée, avec le droit de type G : lancement de la recherche du fichier sur la SAG. S'il est présent, réponse à la demande, sinon rejet de la demande.

Mise à disposition d'un fichier sur la SAG

Le Back Office doit utiliser une requête de transfert sur le nom symbolique SWIFTSTO. Le fichier de données est stocké sur le répertoire $SAG_DOWN ou l'un de ses sous-répertoires.

Les répertoires de Download sont organisés en fonction du nom de demandeur (RequestorDN) et du nom de serveur (ResponderDN). Un fichier peut être mis à disposition d'un demandeur unique, de tous les demandeurs liés à un serveur ou de tous les demandeurs.

- Le répertoire $SAG_DOWN/Responder/Requestor sert le demandeur 'Requestor'
- Le répertoire $SAG_DOWN/Responder sert tous les demandeurs liés au serveur 'Responder'.
- Le répertoire $SAG_DOWN sert tous les demandeurs

Le dépôt considère les répertoires dans l'ordre de priorité ci dessus.

Cette hiérarchie doit être créée manuellement par l'utilisateur. En l'absence d'un sous-répertoire, le fichier est stocké dans le répertoire de niveau supérieur. Par exemple si un fichier est déposé pour le demandeur Requestor1 par le serveur Responder et si le répertoire $SAG_DOWN/Responder/Requestor1 n'existe pas, le fichier sera déposé dans le répertoire $SAG_DOWN/Responder. Le répertoire $SAG_DOWN existe à l'origine de l'installation de la SAG : donc le fichier pourra toujours être déposé.

Le dépôt du fichier est accompagné d'un Pi99 comme tout transfert en provenance du Back Office.

Ce Pi99, construit comme il est décrit au paragraphe '*Envoi du Pi99 par le Back Office'*, permet de préciser la règle de routage et les informations ResponderDN, RequestorDN et FileName nécessaires pour placer le fichier à disposition.

Rappel important : le système unix distingue majuscules et minuscules.

Considérons le squelette SwiftSto.xml et le Pi99 associé décrits au paragraphe '*Personnalisation des fichiers xml*' :

```
<Cx:StoreFile>
<RequestorDN>o=<1>,o=swift</RequestorDN>
<ResponderDN>o=<2>,o=swift</ResponderDN>
<FileName><3></FileName>
<FileInfo>SwCompression=<4></FileInfo>
</Cx:StoreFile>
```

```
Pi99 associé          =:o=test:o=test2:filename:Compression
```

La règle numéro 'xxx' indiquera le nom du squelette xml à traiter.
Il y a substitution des parametres <1> à <4> dans ce fichier squelette. Ce qui donne un fichier de paramètres:

```
<Cx:StoreFile>
<RequestorDN>o=test,o=swift</RequestorDN>
<ResponderDN>o=test2,o=swift</ResponderDN>
<FileName>filename</FileName>
<FileInfo>SwCompression=ZIP</FileInfo>
</Cx:StoreFile>
```

Le fichier à stocker, sera comprimé par InfoZip et placé dans le répertoire $SAG_DOWN/test2/test sous le nom 'filename'. Le back office peut ainsi déterminer le nom du fichier, le type de compression et le répertoire de téléchargement.

Les paramètres <1> et <2> du Pi99 et du fichier xml permettent de contrôler le répertoire de dépôt. En effet si l'on veut que le fichier soit placé dans $SAG_DOWN/Responder on peut ne pas mettre le tag <RequestorDN> dans le fichier SwiftSto.xml ou indiquer le pi99 sous la forme R=xxx::test2:filename:ZIP . De même si l'on veut que le fichier soit placé dans $SAG_DOWN directement ne mettre aucun des tags <RequestorDN> et <ResponderDN> dans SwiftSto.xml ou indiquer le pi99 sous la forme R=xxx:::filename :ZIP

Recherche d'un Fichier sur la SAG

Lorsque la demande externe de récupération est acceptée et si le droit d'accès suppose la recherche du fichier sur la SAG (droit d'accès G, I et i), l'algorithme suivant est traité :
En fonction des paramètres ResponderDN, RequestorDn et FilenName courants, le fichier 'filename' est recherché d'abord dans le répertoire $SAG_DOWN/ResponderDN/RequestorDN s'il existe, puis dans le répertoire $SAG_DOWN/ResponderDN s'il existe puis enfin dans le répertoire racine $SAG_DOWN. Si le fichier est trouvé, la demande est acceptée, s'il n'est pas trouvé, la demande est soit rejetée (droit d'accès 'G'), soit remontée par une requête de transfert en réception Inquiry (droit d'accès 'I' majuscule), ou Normale (droit d'accès 'i' minuscule).

Recherche d'un Fichier sur le Back office

Lorsque la demande externe de récupération est acceptée et si le droit d'accès permet la recherche du fichier sur la SAG, Connect:Express for SWIFTNet peut être amené à lancer une recherche sur le Back Office.

Le transfert SWIFTINQ est lancé avec un Pi99 associé dont la structure est définie par la règle de routage correspondant au contexte. le règle de routage est déterminée en fonction des paramètres FTI comme indiqué au paragraphe '*Détermination de la Règle de Routage*' . Le Pi99 est défini dans la règle sélectionnée, comme la figure suivante l'indique :

```
<Rule Num="001" Status="Enabled" Name="REGLE001">
  ……
  <Pi99Delim>:</Pi99Delim>
    …
  <Pi99PutIn Delim=":">
  </Pi99PutIn>

  <Pi99GetIn Delim=":">
    <Field Name="Requestor">
    <Param Delim="," SubNum="1" Length="12">RequestorDN</Param>
    </Field>
    <Field Name="Filename">
    <Param>FileName</Param>
    </Field>
  </Pi99GetIn>

  <Pi99PutOut Delim=":">
  </Pi99PutOut>

  <Pi99GetOut Delim=":">
    ….
  </Pi99GetOut>
```

Le tag <Pi99GetIn> de cet exemple fournira le Pi99 suivant :

Pi99 associé	Requestor**:** filename

Les informations contenues dans ce pi99 participerons à la recherche du fichier sur la Back Office. Le choix du contenu du Pi99 sera dicté par les contraintes applicatives.

Requête Normale ou Inquiry vers le Back office

Le type de requête de transfert en réception est une notion spécifique des moniteurs Connect:Express : le type 'Normal' indique que le nom physique de fichier doit être pris au répertoire des fichiers du moniteur, alors que la requête de type 'Inquiry' indique qu'on s'attend à trouver une requête de transfert en attente sur le serveur. Dans le cas où le back Office n'est pas servi par un moniteur Connect:Express, le type de requête doit être obligatoirement 'Inquiry'.

Traitement de la Compression

La compression peut être exécutée par l'application sur le Back Office ou par Connect:Express for SWIFTNet sur la SAG. Le mécanisme est différent selon la fonction : pour le Put local ou externe, le paramètre de compression est véhiculé par SWIFTNet dans le champ <a12> (FileInfo). il est donc possible de gérer dynamiquement l'option de compression. Pour le Get, le paramètre compression n'est pas véhiculé : la récupération d'un fichier chez un partenaire distant suppose une entente préalable au sujet de la compression.

Deux informations sont paramétrables pour la compression : l'algorithme utilisé et l'activation de la compression par Connect:Express for SWIFTNet.

Fonction Put et Compression

L'algorithme utilisé est déclaré dans le fichier de paramètres FTI comme l'exemple ci-dessous le montre :

Fonction Put:	<a12>SwCompression=**ZIP,**coding=ascii</a12>

Le paramètre FTI numéro <a12> est échangé entre les partenaires de transfert SWIFTNet. Pour les demandes locales le paramètre FTI numéro <a12> peut être fixé dans le squelette SwiftPut.xml, ou passé dynamiquement par le Back Office dans le Pi99 en remplacement d'un jeton dans le squelette. Pour les demandes externes le paramètre est positionné par le demandeur.

L'exécution de la compression sur la SAG est paramétrée dans la règle de routage par le tag ci-dessous :

<Compression>Y</Compression>

Si ce paramètre est positionné à 'Y', Connect:Express for SWIFTNet exécute la compression demandée. Cette méthode permet la gestion de la compression par le Back Office : dans ce cas il doit être positionné à 'N'. Connect:Express for SWIFTNet supporte trois compressions : InfoZip, gzip et compress. les valeurs correspondantes dans les paramètres FTI sont indiquées dans le tableau suivant :

Valeur du Paramètre	**Type de Compression**
SwCompression=ZIP ou INFOZIP	Compression par InfoZip
SwCompression=GZIP	Compression par gzip
SwCompression=COMPRESS	Compression par l'utilitaire Unix compress
SwCompression=NONE	Pas de compression

Fonction Get et Compression

Dans le cas du Get, le paramètre FTI <a12> n'est pas échangé entre les partenaires de transfert SWIFTNet. Pour les demandes Get locales, si la dé-compression doit être effectuée sur la SAG par Connect:Express for SWIFTNet, l'algorithme utilisé est déclaré dans le fichier de paramètres FTI comme l'exemple ci-dessous le montre :

Fonction Get locale: <a7>SwCompression=**ZIP**</a7>

Le paramètre FTI <a7> (TransferInfo) peut être fixé dans le squelette SwiftGet.xml, ou passé dynamiquement par le Back Office dans le Pi99 en remplacement d'un jeton dans le squelette. La compression est donc gérée localement en fonction d'une convention avec le partenaire distant.
L'activation de la dé-compression est indiquée par le paramètre de la règle de routage :

<Compression>Y</Compression>

Pour les demandes Get externes deux types de fonctionnement sont supportés : la mise à disposition sur la SAG ou l'accès direct au Back Office.
Après mise à disposition sur la SAG, le fichier est envoyé au partenaire distant tel qu'il a été mis à disposition par le Back Office sur la SAG. Si la compression doit être exécutée par Connect:Express for SWIFTNet sur la SAG au cours de la mise à disposition du fichier, l'algorithme de compression est fixé dans le champ <FileInfo> du squelette SwiftSto.xml ou passé dynamiquement par le Back Office dans le Pi99 en remplacement d'un jeton dans ce squelette:

Fonction Store locale: <FileInfo>SwCompression=**ZIP,**coding=ascii</FileInfo>

Pour les demandes Get externes avec accès direct sur le Back Office l'exécution de la compression sur la SAG est paramétrée dans la règle de routage d'après le tag ci-dessous:

<InquiryCompression>ZIP</InquiryCompression>

Connect:Express for SWIFTNet supporte trois compressions : InfoZip, gzip et compress.

Remarque : La compatibilité avec la version 1 de Connect:Express for SWIFTNet n'est que partiellement assurée. En effet le paramètre utilisé pour la compression en version 1 peut être ré-utilisé en version 2, mais un seul profil de routage étant disponible en version 1 cela suppose que la compression soit traitée de la même façon pour tous les transferts.

Personnalisation de la Recherche des Règles

La recherche de la règle de routage est effectuée sur les demandes externes, par les programmes ConnectAccept, ConnectAcceptPutInit et ConnectAcceptInquiry. Ils traitent le fichier FindRule.xml: l'absence de ce fichier provoque l'utilisation de la règle 000 dans tous les cas .

Les critères de sélection peuvent s'appliquer à tout paramètre de la demande en incluant ou en excluant le respect de la condition.

Le fichier FindRule.xml est représenté ci-dessous :

Personnalisation de la Recherche de la Règle de Routage

```
<?xml version="1.0" encoding="UTF-8"?>
<FindRuleCfg xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation='/test/cexpress/x141/exit/FindRule.xsd'>

<FindRuleList>

<FindRule Num="000" Status="Enabled">
<Select Action="Include" FTIParam="ResponderDN">.*,o=swift$</Select>
<Select Action="Include" FTIParam="RequestorDN">^o=tes.*,.*</Select>
</FindRule>
<FindRule Num="002" Status="Enabled">
<Select Action="Include" FTIParam="RequestorDN">^o=bank0.*,.*</Select>
</FindRule>

</FindRuleList>

</FindRuleCfg>
```

Le fichier FindRule.xml peut être personnalisé puis contrôlé avec l'utilitaire ParseFindRule qui vérifie la syntaxe sur la base du fichier FindRule.xsd. L'utilitaire se trouve dans le répertoire /exit de Connect:Express for SWIFTNet. Le fichier xsd se trouve dans le même répertoire : c'est un fichier non modifiable. La première règle de syntaxe est que tous les tags et les paramètres sont obligatoires.

Le tableau suivant définit les TAG de ce fichier :

TAG	Description	Attributs
FindRuleCfg	Cette en-tête ne doit être modifiée qu'une fois à l'installation. Elle doit indiquer le nom physique complet du fichier .xsd à utiliser pour le contrôle de syntaxe.	Aucun
FindRuleList	Associé à son Tag de fin. Ce bloc contient l'ensemble des filtres valides	Aucun
FindRule	Associé à son Tag de fin. Ce bloc indique le numéro de règle qui sera selectionné si l'ensemble des critères de sélection qu'il contient est satisfait. Les blocs sont traités les uns après les autres et le premier Bloc trouvé provoque l'arrêt du traitement.	**Num**: numéro de règle **Status**: état du filtre ***Enabled*** = le filtre est actif ***Disable****d* = le filtre n'est pas actif
Select	Associé à son Tag de fin. Ce bloc indique une expression régulière et le traitement à appliquer : inclure ou exclure en cas de résultat positif sur le paramètre FTI indiqué.	**Action** : méthode de traitement ***Include*** : si la condition est vérifiée la demande est conservée. ***Exclude***: si la condition est vérifiée la demande est rejetée. **FTIParam** : nom FTI du paramètre à traiter sur lequel on applique le masque indiqué par ce Bloc

Personnalisation des Règles

Tout transfert de fichier avec Connect:Express for SWIFTNet s'inscrit dans un mécanisme de routage et nécessite l'utilisation d'une règle. Une règle de routage est identifiée par un numéro et comprend les informations nécessaires pour assurer le routage de bout en bout conformément aux spécifications de l'utilisateur. Toutes les règles sont stockées dans le fichier RuleList.xml situé dans le répertoire /exit.

```
<?xml version="1.0" encoding="UTF-8"?>
<RuleListCfg xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation='/test/cexpress/x141/exit/RuleList.xsd'>

<RuleList>

<Rule Num="000" Status="Enabled" Name="DEFAULT">
<PutXmlFile>$TOM_DIR/exit/SwiftPut.xml</PutXmlFile>
 <GetXmlFile>$TOM_DIR/exit /SwiftGet.xml</GetXmlFile>
 <StoXmlFile>$TOM_DIR/exit/SwiftSto.xml</StoXmlFile>
<InquiryCompression>ZIP</InquiryCompression>
<BackOffice>CENTRAL</BackOffice>
 <Compression>Y</Compression>
 <Pi99Delim>:</Pi99Delim>

 <Pi99PutIn Delim=":">
  <Field Name="Requestor">
   <Param SubName="o" SubNum="1" Offset="0" Length="12">RequestorDN</Param>
  </Field>
  <Field Name="Filename">
   <Param>FileName</Param>
  </Field>
 </Pi99PutIn>

 <Pi99GetIn Delim=":">
  <Field Name="Requestor">
   <Param SubName="o" SubNum="1" Length="12">a1</Param>
  </Field>
  <Field Name="Filename">
   <Param>FileName</Param>
  </Field>
 </Pi99GetIn>

 <Pi99PutOut Delim=":">
  <Field Name="Responder">
   <Param Delim="," SubNum="1" Length="12">a2</Param>
  </Field>
  <Field Name="Filename">
   <Param>a10</Param>
  </Field>
  <Field Name="TransferDescription">
   <Param Length="8">a6</Param>
  </Field>
 </Pi99PutOut>

 <Pi99GetOut Delim=":">
  <Field Name="Responder">
   <Param Delim="," SubNum="1" Length="12">a2</Param>
  </Field>
  <Field Name="Filename">
   <Param>a10</Param>
  </Field>
 </Pi99GetOut>
 </Rule>

 </RuleList>

 </RuleListCfg>
```

Le tableau ci-dessous définit les TAG du fichier RuleList.xml, le tableau suivant décrit les TAG de la structure du Pi99:

TAG	Description	Attributs
RuleListCfg	Cette en-tête ne doit être modifiée qu'une fois à l'installation. Elle doit indiquer le nom physique complet du fichier .xsd à utiliser pour le contrôle de syntaxe.	Aucun
RuleList	Associé à son Tag de fin. Ce bloc contient l'ensemble des règles valides	Aucun
Rule	Associé à son Tag de fin. Ce bloc délimite l'ensemble des informations caractérisant la règle. Toutes les informations doivent être fournies.	**Num**: numéro de règle **Status**: état du filtre ***Enabled*** = le filtre est actif ***Disabled*** = le filtre n'est pas actif **Name** : Nom indicatif de la règle.
PutXmlFile	Associé à son Tag de fin. Ce champ indique le nom du fichier squelette xml utilisé pour passer la liste des paramètres de la commande Ft-Put à FTI.	Aucun
GetXmlFile	Associé à son Tag de fin. Ce champ indique le nom du fichier squelette xml utilisé pour passer la liste des paramètres de la commande Ft-Get à FTI.	Aucun
StoXmlFile	Associé à son Tag de fin. Ce champ indique le nom du fichier squelette xml utilisé pour mettre un fichier à disposition sur la SAG .	Aucun
BackOffice	Associé à son Tag de fin. Ce champ indique le nom symbolique du Bak Office vers lequel Connect:Express for SWIFTNet doit diriger les requêtes de transfert des fichiers SWIFTFWD, SWIFTINQ, SWIFTACK et SWIFTNAK par exemple.	Aucun
Compression	Associé à son Tag de fin. Ce champ indique si la compression doit être activée dans le cas d'une demande locale. **Y** : le paramètre FTI <a7> ou <a12> est traité. Par exemple <a7>SwCompression=ZIP </a7> (Get local) provoquera la décompression par InfoZip d'un fichier récupéré chez un partenaire. <a12>SwCompression=ZIP,coding=ascii</a12> provoquera la compression par InfoZip d'un fichier à émettre vers un partenaire. **N** : la compression n'est pas traitée sur la SAG. Ceci permet de laisser le Back Office traiter la compression. Les valeurs de SwCompression peuvent être ZIP, INFOZIP, GZIP, COMPRESS, NONE	Aucun
InquiryCompression	Associé à son Tag de fin. Ce champ indique quelle compression doit être activée dans le cas d'une mise à disposition. Exemple : ZIP, GZIP,COMPRESS,NONE	Aucun
Pi99Delim	Associé à son Tag de fin. Ce champ indique le caractère à rechercher comme délimiteur entre les valeurs incluses dans le Pi99 reçu.	Aucun
Pi99PutIn	Associé à son Tag de fin. Ce bloc délimite la description du Pi99 à envoyer vers le Back Office avec le transfert du fichier SWIFTFWD (demande externe de dépôt de fichier). Voir Voir dans le tableau suivant le détail du bloc.	**Delim** : précise le délimiteur utilisé dans le Pi99 émis.
Pi99PutOut	Associé à son Tag de fin. Ce bloc délimite la description du Pi99 à envoyer vers le Back Office avec le transfert du fichier	**Delim** : précise le délimiteur utilisé

TAG	Description	Attributs
	SWIFTACK ou SWIFTNAK (Acquittement d'une demande locale de dépôt de fichier). Voir dans le tableau suivant le détail du bloc.	dans le Pi99 émis.
Pi99GetIn	Associé à son Tag de fin. Ce bloc délimite la description du Pi99 à envoyer vers le Back Office avec le transfert du fichier SWIFTINQ (demande externe de récupération de fichier). Voir dans le tableau suivant le détail du bloc.	**Delim** : précise le délimiteur utilisé dans le Pi99 émis.
Pi99GetOut	Associé à son Tag de fin. Ce bloc délimite la description du Pi99 à envoyer vers le Back Office avec le transfert du fichier SWIFTACK ou SWIFTNAK (Acquittement d'une demande locale de récupération de fichier). Voir dans le tableau suivant le détail du bloc.	**Delim** : précise le délimiteur utilisé dans le Pi99 émis.

Définition de la Structure du Pi99 en Emission Vers le Back office.

Le tableau ci-dessous décrit précisément la syntaxe proposée :

TAG	Description	Attributs
Pi99PutIn Pi99PuOut Pi99GetIn Pi99GetOut	Chaque bloc décrit une structure qui peut comprendre des valeurs constantes et des valeurs extraites du fichier des paramètres FTI. La structure peut comprendre plusieurs champs séparés par le délimiteur défini plus haut. Chaque champ peut être composé d'une ou plusieurs valeurs.	Aucun
Field	Associé à son Tag de fin. Ce bloc délimite la description d'un champ compris entre deux délimiteurs. Cette description comprend un ensemble quelconque de tags param et string.	**Name** : nom indicatif du champ
String	Associé à son Tag de fin. Ce champ contient une chaîne de caractères quelconque .	Aucun
Param	Associé à son Tag de fin. Ce champ indique un paramètre FTI à traiter et l'information à en extraire. Le nom du paramètre est de la forme 'ai' dans le cas d'une demande locale, de la forme 'Nom de paramètre' dans le cas de demandes externes. On peut extraire le paramètre entier, ou les x premiers caractères du paramètre entier. On peut traiter un sous paramètre : certains paramètres FTI se composent de sous paramètres (par exemple le RequestorDN). Les sous paramètres sont séparés par un caractère (virgule, deux points …), peuvent être 'nommés' (exemple : o=.., cn=..). Un même paramètre peut contenir plusieurs sous paramètres de même 'nom'. On peut extraire le nième sous paramètre, le sous paramètre de nom ' x=', le nième sous paramètre de nom ' x='. Chaque extraction peut se faire sur une longueur maximum, ou jusqu'au délimiteur suivant. Remarque : l'extraction du nième sous paramètre comprend le 'nom' du paramètre (exemple : 'o=test'). L'extraction du paramètre de 'nom' 'o' exclus le 'nom' (exemple 'test').	**Delim** : indique le caractère utilisé comme délimiteur dans ce paramètre FTI composé (exemple ','). Ce paramètre est combiné avec les paramètres Subname et Subnum décrits ci après. **Subname** : indique le nom de sous paramètre FTI à rechercher (exemple 'o=') il se trouve au début ou après le délimiteur défini par le paramètre Delim. **Subnum** : indique le numéro de sous paramètre FTI à rechercher (exemple : le troisième 'o=') **Offset**: indique la position dans le paramètre ou sous-paramètre.. **Length** : indique la longueur maximum de la chaîne de caractères à extraire.

Note : le champ <param> </param> indique un nom de paramètre ou un numéro de paramètre selon que la demande est interne ou externe. Ceci provient du fait que Connect:Express for SWIFTNet traite le fichier xml des paramètres FTI associé au transfert pour en extraire les chaînes de caractères.

Dans le cas d'une demande interne le fichier xml est créé par Connect:Express pour SWIFTNet qui utilise la syntaxe suivante :

<a1>……. </a1>

Dans le cas d'une demande externe le fichier xml est créé par FTI qui utilise la syntaxe suivante :

<RequestorDN>……. </RequestorDN>

Chapitre 4

Description des flux

Ce chapitre présente au cas par cas le déroulement des opérations pour chaque flux et montre comment est assuré le suivi de bout en bout et la suppression des objets temporaires. Cette description utilise les noms symboliques standards et la règle de routage numéro 000 décrite à l'annexe A.

Généralités

Les flux décrits sont les suivants :

- Emission d'un fichier du Back-office vers SWIFTNet
- Transmission au Back-office d'un fichier reçu par SWIFTNet
- Demande de réception d'un fichier par le Back-office
- Mise à disposition d'un fichier sur la SAG par le Back-office
- Récupération d'un fichier du Back-office à partir de la SAG

Chaque flux est constitué d'un enchaînement de requêtes de transfert de fichiers, de transferts de fichiers, et d'exécution de programmes. Des objets temporaires sont créés – requêtes de transfert, fichiers de travail (données, comptes rendus et paramètres). Tous les objets temporaires sont systématiquement détruits une fois que l'opération demandée s'est terminée avec succès.

Afin d'assurer le suivi de bout en bout des échanges, chacun d'eux est identifié de façon unique.

Identification des Echanges

L'identification de bout en bout est construite par l'utilisateur à partir de différentes informations selon le sens de la demande. Elle est véhiculée par le Pi99 et dépend donc des choix faits par l'utilisateur. Avec la règle de routage 000, l'identification est traitée de la façon suivante :

Pour une demande locale : le Back-office est le 'Requestor' (ou paramètre FTI <a1>)
1. Le nom du fichier ('filename' ou paramètre FTI <a10>)
2. Le nom du destinataire ('Responder' ou paramètre FTI <a2>)
3. L'identifiant du transfert (sur le Back-office)

Pour une demande externe : le Back-office est le 'Responder' (ou paramètre FTI <ResponderDN>)
1. Le nom du demandeur ('Requestor' ou paramètre FTI <RequestorDN>)
2. Le nom du fichier ('filename' ou paramètre FTI <FileName>)

Ces informations sont véhiculées par Connect:Express dans le Pi99 des échanges PeSIT et conservées sur la SAG dans le fichier de paramètres FTI xml.

Lorsque le Back-office est à l'origine du transfert (il est donc 'Requestor') le nom du destinataire (le 'Responder'), le nom de fichier, l'identifiant de transfert (Numéro de requête dans le cas de Connect:Express sur le Back Office) et une option de compression sont passés par le Pi99 échangé au cours de la session PeSIT. Ces quatre informations sont stockées dans le fichier .xml et seront conservées jusqu'à réception confirmée du fichier chez le destinataire. Le Pi99 échangé au cours du transfert d'acquittement vers le Back-office reprend les trois premières informations (destinataire, nom de fichier, identifiant du transfert).

Note : La compression est indiquée dans le paramètre FTI FileInfo ou <a12>, l'identifiant de transfert est stocké dans le paramètre FTI TransferDescription ou <a6>.

Lorsque le fichier est reçu d'un partenaire extérieur, le nom du demandeur et le nom de fichier sont sauvegardés dans le fichier xml créé par créé par FTI et envoyés dans le Pi99 au cours de l'échange PeSIT avec le Back-office.

Note : L'utilisation du pi99 est supposée possible sur le Back-office. Dans le cas où le moniteur de transferts de fichiers sur le Back-office est Connect:Express OS/390, Unix ou Windows cette condition est vérifiée.

Compte-Rendus des Transferts

Connect:Express for SWIFTNet construit pour chaque transfert un fichier d'acquittement. Ce fichier est transféré sous le nom symbolique SWIFTACK ou SWIFTNAK selon le résultat. Son nom physique est de la forme :

$TOM_DIR/out/ Numéro de Requête.type.ack
ou
$TOM_DIR/out/Numéro de Requête.type.nak

La valeur 'Numéro de Requête' est égale au numéro de requête du transfert par Connect:Express for SWIFTNet. à l'origine de la demande. La valeur 'type' peut être égale a 'sto', 'put' ou 'get'.

Exemple : $TOM_DIR/out/03700001.put.ack

La structure du fichier d'acquittement est représentée page suivante. En cas d'anomalie, le fichier d'acquittement peut inclure les paramètres FTI utilisés et le fichier de logging de FTI lorsqu'il existe .

Structure d'un Fichier d'Acquittement

```
<?xml version='1.0' encoding='UTF-8'?>
<Cx:StatusAttributes>
   <Cx:Severity>Information</Cx:Severity>
   <Cx:Code>OK</Cx:Code>
```

```
<?xml version='1.0' encoding='UTF-8'?>
<Cx:StatusAttributes>
   <Cx:Severity>Information</Cx:Severity>
   <Cx:Code>OK</Cx:Code>
   <Cx:Text>Transfer complete</Cx:Text>
   <Cx:ProcessingState>Ended</Cx:ProcessingState>
   <Cx:Component>SWIFTPUT</Cx:Component>
   <Cx:ExpressRequestNumber>17800001</Cx:ExpressRequestNumber>
   <Cx:SNLTransferRef>SNL00152XX1056703503037416C</Cx:SNLTransferRef>
   <Cx:Details>
      <Cx:Processing>
         File stored in /SWIFTAlliance/Gateway/FT/data/download/test/test/filename
      </Cx:Processing>
      <Cx:FTISubmittedParameters>
         <Ft:PutFileRequest>
         <a1>o=test,o=swift</a1>
         . . . .
         <a13>FALSE</a13>
         <a14>FALSE</a14>
         <a18>FALSE</a18>
         </Ft:PutFileRequest>
      </Cx:FTISubmittedParameters>
   </Cx:Details>
</Cx:StatusAttributes>
```

Les champs d'un fichier d'acquittement sont définis dans le tableau ci-dessous :

Champ	**Description et valeurs**
Severity	Détermine le type d'action à prendre à la réception de ce fichier. Ce champ peut prendre les valeurs : Information , Error
Code	Détermine le type d'acquittement. Ce champ peut prendre les valeurs OK ou NOK.
Text	Exprime en clair le résultat des opération.
ProcessingState	Indique l'étape de traitement du programme (SWIFTPUT, SWIFTGET, SWIFTSTO) qui envoie les résultats.
Component	Nom du programme à l'origine du compte-rendu. Ce champ peut prendre les valeurs suivantes SWIFTPUT, SWFTSTO, SWIFTGET
ExpressRequestNumber	Numéro de requête du transfert de fichier effectué entre le Back office et la Sag et à l'origine de la demande.
SNLTransferRef	Numéro de requête de transfert attribué par FTI.
FTICommandReturned	Ce bloc fournit le code retour (<Code> </Code>) et le message d'erreur (<Text> </Text>) reçus de FTI .
Details	Ce bloc contient un complément d'information constitué de l'un ou plusieurs des champs décrits ci-dessous.
Processing	Fournit une information de traitement complémentaire.
FTISubmittedParameters	Ce bloc fournit la liste des paramètres FTI utilisés.
FTILogFile	Le fichier Log de FTI est inclus dans le bloc Details

Exemple de Fichier d'Acquittement Négatif :

```
<?xml version='1.0' encoding='UTF-8'?>
<Cx:StatusAttributes>
   <Cx:Severity>Error</Cx:Severity>
   <Cx:Code>KO</Cx:Code>
   <Cx:Text></Cx:Text>
   <Cx:FTICommandReturned>
      <Cx:Code>1</Cx:Code>
      <Cx:Text></Cx:Text>
   </Cx:FTICommandReturned>
   <Cx:processingState>Submitting the ft-get request to FTI</Cx::processingState>
   <Cx:Component>SWIFTGET</Cx:Component>
   <Cx:ExpressRequestNumber>03000039</Cx:ExpressRequestNumber>
   <Cx:SNLTransferRef></Cx:SNLTransferRef>
   <Cx:Details>
      <Cx:FTISubmittedParameters>
         <Ft:GetFileRequest>
            <a1>o=test,o=swift</a1>
            <a2>o=test,o=swift</a2>
            <a3>TestService</a3>
            <a4>RequestType</a4>
            <a5>cn=certest10,o=develop,o=test,o=swift</a5>
            <a6>idt</a6>
            <a7>SwCompression=none,coding=ascii</a7>
            <a8>reqnoCENTRAL</a8>
            <a9>/test/cexpress/x141/out/03000039.get</a9>
            <a10>filename</a10>
            <a13>FALSE</a13>
            <a16></a16>
            <a24></a24>
         </Ft:GetFileRequest>
      </Cx:FTISubmittedParameters>
      <Cx:FTILogFile>
         <Sw:ExchangeFileResponse>
         . . . . .
      </Cx:FTILogFile>
   </Cx:Details>
</Cx:StatusAttributes>
```

Gestion des Incidents

La gestion des incidents est assurée à différents niveaux :

- Transferts des données entre le back-office et la SAG
- Traitements de routage
- Transferts SWIFT

Connect:Express for SWIFTNet gère un fichier de logging dans lequel sont consignés les différents évènements. Ce fichier est journalier, il est archivé en début de journée et le nombre de versions est paramétrable. Par exemple :

Fichier log de Connect:Express for SWIFTNet	$TOM_DIR/exit/swift.log
Si le nombre de fichiers conservé est	#export KEEP_SW_LOG=7
Fichiers log archivés	$TOM_DIR/exit/swlog/L20030810
	$TOM_DIR/exit/swlog/L20030811
	$TOM_DIR/exit/swlog/L20030812
	$TOM_DIR/exit/swlog/L20030813
	$TOM_DIR/exit/swlog/L20030814
	$TOM_DIR/exit/swlog/L20030815
	$TOM_DIR/exit/swlog/L20030816

La figure page suivante montre un exemple de fichier swift.log : ce fichier regroupe les références, identifications et événements nécessaires à la recherche d'incidents.

Exemple de Fichier Swift.log

```
20030811040029 PUT CENTRAL  04000021           SWIFTPUT started
20030811040029 PUT CENTRAL  04000021           No rule number indicated in Pi99. Using default
20030811040029 PUT CENTRAL  04000021           RequestorDN : o=test,o=swift. ResponderDN :
o=test
20030811040029 PUT CENTRAL  04000021           Compression command : cexpress/x141/exit/gzip
20030811040030 API                             ConnectAcceptPutInit started
20030811040030 API SNL00152XX1060592429055949S RequestorDN : o=test,o=swift. ResponderDN :
o=test
20030811040030 API SNL00152XX1060592429055949S Controlling access. There is a white list
20030811040030 API SNL00152XX1060592429055949S Controlling access. There is no black list
20030811040030 API SNL00152XX1060592429055949S Access authorized by the white list
20030811040030 API SNL00152XX1060592429055949S Put file authorized
20030811040030 API SNL00152XX1060592429055949S ConnectAcceptPutInit terminated successfully
20030811040030 FWD                             SWIFTFWD started
20030811040030 FWD SNL00152XX1060592429055949S RequestorDN : o=test,o=swift. ResponderDN :
o=test
20030811040030 FWD SNL00152XX1060592429055949S Matching rule number 000 used
20030811040030 FWD SNL00152XX1060592429055949S Uncompression command :
cexpress/x141/exit/gunzip
20030811040030 FWD SNL00152XX1060592429055949S File filename successfully forwarded to the
back o
20030811040030 FWD SNL00152XX1060592429055949S SWIFTFWD terminated successfully
20030811040034 CLF CENTRAL  04000022           CLEARFWD started
20030811040034 CLF CENTRAL  04000022           CLEARFWD terminated successfully
20030811040044 PUT CENTRAL  04000021           Ft-put command is OK
20030811040044 PUT CENTRAL  04000021           Ft-Put OK. SNL Transfer Reference :
SNL00152XX1060
20030811040044 PUT CENTRAL  04000021           SWIFTPUT terminated successfully
20030811040047 CLA CENTRAL  04000023           CLEARACK started
20030811040048 CLA CENTRAL  04000023           CLEARACK terminated successfully
20030811040108 GET CENTRAL  04000024           SWIFTGET started
20030811040108 GET CENTRAL  04000024           No rule number indicated in Pi99. Using default
20030811040109 GET CENTRAL  04000024           RequestorDN : o=test,o=swift. ResponderDN :
o=test
```

```
20030811040111 CNI                              ConnectAcceptInq started
20030811040111 CNI SNL00152XX1060592470056736S RequestorDN : o=test,o=swift. ResponderDN :
o=test
20030811040111 CNI SNL00152XX1060592470056736S Controlling access. There is a white list
20030811040111 CNI SNL00152XX1060592470056736S Controlling access. There is no black list
20030811040111 CNI SNL00152XX1060592470056736S Access authorized by the white list
20030811040111 CNI SNL00152XX1060592470056736S Matching rule number 000 used
20030811040111 CNI SNL00152XX1060592470056736S fileName = filename. responderSubdir = test.
reque
20030811040111 CNI SNL00152XX1060592470056736S Download of file filename from the back office
20030811040114 INQ CENTRAL  04000025            SWIFTINQ started
20030811040114 INQ CENTRAL  04000025            SWIFTINQ terminated successfully
20030811040121 CNI SNL00152XX1060592470056736S File filename successfully transfered from
CENTRAL
20030811040121 CNI SNL00152XX1060592470056736S Compression command :
cexpress/x141/exit/InfoZip
20030811040121 CNI SNL00152XX1060592470056736S File /test/cexpress/x141/in/35118.inq moved to
20030811040121 CNI SNL00152XX1060592470056736S Download accepted
20030811040121 CNI SNL00152XX1060592470056736S ConnectAcceptInq terminated successfully
20030811040124 GET CENTRAL  04000024            Ft-get is OK
20030811040124 GET CENTRAL  04000024            Ft-get OK. SNL Transfer Reference :
SNL00152XX1060
20030811040124 GET CENTRAL  04000024            Uncompression command :
cexpress/x141/exit/InfoUnz
20030811040124 GET CENTRAL  04000024            File filename successfully forwarded to the
back o
20030811040124 GET CENTRAL  04000024            SWIFTGET terminated successfully
20030811040127 CLF CENTRAL  04000026            CLEARFWD started
20030811040127 CLF CENTRAL  04000026            CLEARFWD terminated successfully
20030811040145 STO CENTRAL  04000027            SWIFTSTO started
20030811040145 STO CENTRAL  04000027            No rule number indicated in Pi99. Using default
ru
20030811040145 STO CENTRAL  04000027            RequestorDN : o=test,o=swift. ResponderDN =
o=test
20030811040145 STO CENTRAL  04000027            FileName : filename. FileInfo :
SwCompression=ZIP
20030811040145 STO CENTRAL  04000027            Download directory /SWIFTAlliance/Gateway…
exists
20030811040145 STO CENTRAL  04000027            Compression command :
cexpress/x141/exit/InfoZip
20030811040145 STO CENTRAL  04000027            File filename stored as /SWIFTAllia
…/download/tes
20030811040145 STO CENTRAL  04000027            Ack file sent to the back office CENTRAL
20030811040145 STO CENTRAL  04000027            SWIFTSTO terminated successfully
20030811040148 CLA CENTRAL  04000028            CLEARACK started
20030811040148 CLA CENTRAL  04000028            CLEARACK terminated successfully
```

On peut faire des rapprochements entre les numéros de requêtes Connect:Express , les identifiants SNL…. de FTI, les noms symboliques de partenaires et de fichiers utilisées pour les transferts avec le Back Office, les paramètres FTI RequestorDN, ResponderDN et Filename utilisés pour identifier et contrôler les demandes SWIFTNet. La première colonne indique la date et l'heure de l'événement, la seconde colonne identifie la procédure en cours, le troisième colonne fourni l'identifiant Connect:Express (Numéro de requête et Back office) ou l'identifiant FTI. La derniére colonne indique en clair la nature de l'événement.

Incident de Transfert entre le Back-office et la SAG

Les mécanismes standard de compte-rendu et de reprise s'appliquent aux transferts de fichiers entre le moniteur PeSIT du Back-office et le moniteur Connect:Express for SWIFNet installé sur la SAG. Les procédures existantes au niveau du Back-office restent valables.

Connect:Express Unix for SWIFTNet fournit les outils standards de gestion d'incidents:

- Reprise automatique sur les requêtes de transfert locales
- Compte rendus inscrit dans le fichier LOG
- Journalisation et codes retours dans le fichier RENC
- Exit généralisé UEXERR : cette commande, déclenchée en cas d'erreur, peut être personnalisée pour remonter des alertes ou déclencher quelque traitement que ce soit sur détection d'une anomalie

Incident au Cours du Traitement de routage

Toute anomalie dans l'exécution des programmes automatiques est traitée comme une anomalie dans les transferts SWIFT traitée au paragraphe suivant.

Incident de Transfert sur SWIFTNet

Les anomalies détectées au cours des échanges SWIFT sont remontées vers le Back-office par le transfert d'un fichier d'acquittement, créé par Connect:Express for SWIFTNet pour chaque erreur, sous le nom symbolique SWIFTNAK.

Le programme (SWIFTPUT, SWIFTGET, SWIFTSTO) soumet une requête de transfert à Connect:Express Unix for SWIFTNet :

p1b8preq	**Paramètre**	**Valeur**
	Fichier	SWIFTNAK
	Partenaire	Back-office ID
	Fichier physique	$TOM_DIR/out/noderequête.type.ack
	Pi99 ou équivalent	Responder:FileName:TransferRef

Voir la description d'un fichier d'acquittement au paragraphe '*Compte-Rendus des Transferts*'.

Utilitaire d'exploitation

/Exit/PURGESAG

En fonctionnement normal aucun objet inutile n'est conservé sur la SAG. L'utilitaire PURGESAG n'est fourni que pour les cas exceptionnel ou une purge manuelle devrait être déclenchée.

La liste des objets obsolètes à purger est la suivante:

- Requêtes de transfert de Connect:Express : lancement de p1b8pppur
- Fichiers de données dans $TOM_DIR/OUT (*) et $SAG_DOWN (*)
- Fichiers de paramètres dans $TOM_DIR/IN (*.par) et $SAG_XML (*.par)
- Fichiers d'acquittement dans $TOM_DIR/out (*.ack) et $TOM_DIR/out (*.nak)
- Fichiers résultats dans $SAG_LOG (*.err , *.error et *.ok)

Emission d'un Fichier du Back-office Vers SWIFTNet

Dans ce qui suit &REQNUMB désigne le numéro de requête de transfert de Connect:Express (par exemple 03900007) et &PARTNID désigne le nom de partenaire (Par exemple CENTRAL).

Les procédures s'enchaînent de la façon suivante :

Etape 1 : Le Back-office envoie un fichier vers SWIFTNet

Requête de transfert	Paramètre	Valeur
	Fichier	SWIFTPUT
	Partenaire	SWIFTNET
	Alias	CENTRAL
	Mot de passe alias	CENTRAL
	Fichier physique	Données à transférer
	Pi99	Responder**:**FileName**:**TransferRef**:**Compression

Etape 2 : Réception des données sur la SAG

Connect:Express Unix for SWIFTNet déclenche la commande de fin de transfert :

SWIFTPUT	Paramètre	Valeur
	$1	Requête de transfert locale &REQNUMB
	$2	Partenaire (Back-office) &PARTNID
	$4	put_&REQNUMB.tmp fichier des données
	$9	Responder**:**FileName**:**TransferRef**:**Compression

SWIFTPUT déduit de l'absence de numéro de règle que la règle numéro 000 s'applique. Il utilise le nom du squelette xml indiqué pour l'opération de put et construit le fichier de paramètres xml : $SAG_XML/*&REQNUMB.put.*

SWIFTPUT exécute la compression si elle est demandée par la règle 000 (appel de infoZip, gzip ou compress).

Etape 3 : Transfert des données vers SWIFTNet

Connect:Express for SWIFTNet envoie la commande à FTI accompagnée du fichier de paramètres xml $SAG_XML/*&REQNUMB.put :*

```
ft-put –p $SAG_XML/&REQNUMB.put
```

Si SWIFTPUT reçoit un code nul de FTI, Connect:Express for SWIFTNet envoie un acquittement positif , SWIFTACK . Voir la description du fichier d'acquittement au paragraphe '*Compte-Rendus des Transferts*'. Le cas d'échec est traité plus bas.

SWIFTPUT Purge les objets temporaires à partir des informations $1 et $4.

SWIFTPUT utilise la règle courante numéro 000. Il utilise la structure de Pi99 sur Put sortant (Pi99PutOut) et le nom symbolique du Back Office.

Etape 4 : Envoi de l'acquittement positif vers le Back-office

SWIFTPUT envoie une requête de transfert à Connect:Express Unix for SWIFTNet :

p1b8preq	**Paramètre**	**Valeur**
	Fichier	SWIFTACK
	Partenaire	CENTRAL
	Fichier physique	$TOM_DIR/out/ xxxxxxxx.put.ack
	Pi99 ou équivalent	Responder**:**FileName**:**TransferRef

Etape 5 : Suppression des objets temporaires

Connect:Express Unix for SWIFTNet déclenche la commande de fin de transfert:

CLEARACK	**Paramètre**	**Valeur**
	$1	Requête de transfert locale &REQNUMB
	$2	Partenaire (Back-office) &PARTNID
	$4	$TOM_DIR/out/ xxxxxxxx.put.ack

CLEARACK Purge les objets temporaires à partir des informations $1 et $4.

Echec du transfert

Si SWIFTPUT reçoit un code non nul de FTI, Connect:Express for SWIFTNet envoie un acquittement négatif , SWIFTNAK . Voir la description du fichier d'acquittement au paragraphe '*Compte-Rendus des Transferts*'.

SWIFTPUT utilise la règle courante numéro 000. Il utilise la structure de Pi99 sur Put sortant (Pi99PutOut) et le nom symbolique du Back Office.

Etape 4 : Envoi de l'acquittement négatif vers le Back-office

SWIFTPUT envoie une requête de transfert à Connect:Express Unix for SWIFTNet :

p1b8preq	**Paramètre**	**Valeur**
	Fichier	SWIFTNAK
	Partenaire	CENTRAL
	Fichier physique	$TOM_DIR/out/ xxxxxxxx.put.nak
	Pi99 ou équivalent	Responder**:**FileName**:**TransferRef

Etape 5 : Suppression des objets temporaires

Connect:Express Unix for SWIFTNet déclenche la commande de fin de transfert :

CLEARNAK	**Paramètre**	**Valeur**
	$1	Requête de transfert locale &REQNUMB
	$2	Partenaire (Back-office) &PARTNID
	$4	$TOM_DIR/out/ xxxxxxxx.put.nak

CLEARNAK Purge les objets temporaires à partir des informations $1 et $4.

Transmission au Back-office d'un Fichier Reçu par SWIFTNet

Dans ce qui suit &REQNUMB désigne le numéro de requête de transfert de Connect:Express (par exemple 03900007) et &PARTNID désigne le nom de partenaire (Par exemple CENTRAL).

Les procédures s'enchaînent de la façon suivante :

Etape 1 : FTI reçoit la demande de transfert (Put)

FTI crée un fichier *$SAG_XML/Filename.TransferRef.par*.

Etape 2 : Activation du contrôle d'accès de Connect:Express Unix for SWIFTNet par FTI

Le paramètre FTI LTA-PutInit provoque le déclenchement de ConnectAcceptPutInit en lui passant en paramètre le nom du fichier des paramètres:

```
ConnectAcceptPutInit –a17 $SAG_XML/FileName.TransferRef.par
```

ConnectAcceptPutInit recherche le fichier WhiteList.xml : s'il le trouve, il vérifie que la demande est acceptée.
ConnectAcceptPutInit recherche le fichier BlackList.xml : s'il le trouve, il vérifie que la demande n'est pas rejetée.
Si la demande est rejetée, un code non nul est retourné à FTI. Si la demande est acceptée le fichier est reçu par FTI, et le processus continue :

FTI crée le fichier de données sous le répertoire $SAG_RCV avec le nom qu'il a indiqué dans le tag <SaveFileAs> du fichier xml des paramètres.

Etape 3 : Activation du programme de Connect:Express Unix for SWIFTNet par FTI

Le paramètre FTI LTA-PutEnd provoque le déclenchement de SWIFTFWD en lui passant en paramètre le nom du fichier des paramètres:

```
SWIFTFWD –a17 $SAG_XML/FileName.TransferRef.par
```

SWIFTFWD déduit du fichier FindRule.xml que la règle numéro 000 s'applique. Il utilise la structure de Pi99 sur Put entrant (Pi99PutIn) et le nom symbolique du Back Office.

SWIFTFWD exécute la dé-compression indiquée dans le paramètre <FileInfo>SwCompression=.
SWIFTFWD construit la requête de transfert à Connect:express Unix for SWIFTNet.
SWIFTFWD renomme le fichier des paramètres FTI pour destruction ultérieure et déplace le fichier de données à transférer dans son répertoire d'émission /out.

Etape 4 : Routage des données vers le Back-office

SWIFTFWD soumet une requête de transfert à Connect:Express Unix for SWIFTNet :

p1b8preq	**Paramètre**	**Valeur**
	Fichier	SWIFTFWD
	Partenaire	CENTRAL
	Fichier physique	Données à transférer
	Pi99 ou équivalent	Requestor :FileName

Etape 5 : Suppression des objets temporaires

Connect:Express Unix for SWIFTNet déclenche la commande de fin de transfert:

CLEARFWD	**Paramètre**	**Valeur**
	$1	Requête de transfert locale &REQNUMB
	$2	Partenaire (Back-office) &PARTNID
	$4	Fichier des données

CLEARFWD Purge les objets temporaires à partir des informations $1 et $4 et purge le fichier des paramètres FTI.

Demande de Réception d'un Fichier par le Back-office

Dans ce qui suit &REQNUMB désigne le numéro de requête de transfert de Connect:Express (par exemple 03900007) et &PARTNID désigne le nom de partenaire (Par exemple CENTRAL).

Les procédures s'enchaînent de la façon suivante :

Etape 1 : Le Back-office utilise le transfert de fichier pour envoyer la demande vers SWIFTNet

Requête de transfert	**Paramètre**	**Valeur**
	Fichier	SWIFTGET
	Partenaire	SWIFTNET
	Alias	CENTRAL
	Mot de passe alias	CENTRAL
	Fichier physique	Fichier vide
	Pi99 ou équivalent	Responder**:**FileName**:**TransferRef**:**Compression

Etape 2 : Réception de la demande sur la SAG

Connect:Express Unix for SWIFTNet déclenche la commande de fin de transfert:

SWIFTGET	**Paramètre**	**Valeur**
	$1	Requête de transfert locale &REQNUMB
	$2	Partenaire (Back-office) &PARTNID
	$9	Responder**:**FileName**:**TransferRef**:**Compression

SWIFTPUT déduit de l'absence de numéro de règle que la règle numéro 000 s'applique. Il utilise le nom du squelette xml indiqué pour l'opération de get et construit le fichier de paramètres xml : $SAG_XML/*&REQNUMB.get.*

Etape 3 : Envoi de la demande vers SWIFTNet

Connect:Express Unix for SWIFTNet envoie la commande à FTI.

```
ft-get –p $SAG_XML/&REQNUMB.get
```

Si SWIFTGET reçoit un code nul de FTI, il déclenche le transfert SWIFTFWD. Le cas d'échec est traité plus bas.

Etape 4 : Envoi des données vers le Back-office

SWIFTGET soumet une requête de transfert à Connect:Express Unix for SWIFTNet :

p1b8preq	**Paramètre**	**Valeur**
	Fichier	SWIFTFWD
	Partenaire	CENTRAL
	Fichier physique	Fichier de données
	Pi99 ou équivalent	Responder**:**FileName**:**TransferRef

Etape 5 : Suppression des objets temporaires

Connect:Express Unix for SWIFTNet déclenche la commande de fin de transfert :

CLEARFWD	**Paramètre**	**Valeur**
	$1	Requête de transfert locale &REQNUMB
	$2	Partenaire (Back-office) &PARTNID
	$4	Nom de fichier de données

CLEARFWD Purge les objets temporaires à partir des informations $1 et $4.

Echec du transfert

Si SWIFTGET reçoit un code non nul de FTI, il envoie un acquittement négatif , SWIFTNAK . Voir la description du fichier d'acquittement au paragraphe '*Compte-Rendus des Transferts*'.

SWIFTGET utilise la règle courante numéro 000. Il utilise la structure de Pi99 sur Get sortant (Pi99GetOut) et le nom symbolique du Back Office.

Etape 4 : Envoi de l'acquittement négatif vers le Back-office

SWIFTGET soumet une requête de transfert à Connect:Express Unix for SWIFTNet :

p1b8preq	**Paramètre**	**Valeur**
	Fichier	SWIFTNAK
	Partenaire	CENTRAL
	Fichier physique	$TOM_DIR/exit/xxxxxxxx.get.nak
	Pi99 ou équivalent	Responder:FileName:TransferRef

Etape 5 : Suppression des objets temporaires

Connect:Express Unix for SWIFTNet déclenche la commande de fin de transfert :

CLEARNAK	**Paramètre**	**Valeur**
	$1	Requête de transfert locale &REQNUMB
	$2	Partenaire (Back-office) &PARTNID
	$4	$TOM_DIR/exit/xxxxxxxx.get.nak

CLEARNAK Purge les objets temporaires à partir des informations $1 et $4.

Fichier Mis à Disposition sur la Sag par le Back-office

Dans ce qui suit &REQNUMB désigne le numéro de requête de transfert de Connect:Express (par exemple 03900007) et &PARTNID désigne le nom de partenaire (Par exemple CENTRAL).

Les procédures s'enchaînent de la façon suivante :

Etape 1 : Le Back-office met à disposition un fichier vers SWIFTNet

Requête de transfert	Paramètre	Valeur
	Fichier	SWIFTSTO
	Partenaire	SWIFTNET
	Alias	CENTRAL
	Mot de passe alias	CENTRAL
	Fichier physique	Données à transférer
	Pi99 ou équivalent	Responder:FileName:TransferRef:Compression

Etape 2 : Réception des données sur la SAG

Connect:Express Unix for SWIFTNet déclenche la commande de fin de transfert:

SWIFTSTO	Paramètre	Valeur
	$1	Requête de transfert locale &REQNUMB
	$2	Partenaire (Back-office) &PARTNID
	$4	put_&REQNUMB.tmp fichier des données
	$9	Responder:Requestor:FileName:Compression

SWIFTSTO déduit de l'absence de numéro de règle, que la règle numéro 000 s'applique. Il utilise le nom du squelette xml indiqué pour l'opération de store et en déduit les valeurs des paramètres ResponderDN, RequestorDN, FileName et FileInfo.

SWIFTSTO effectue la compression indiquée dans le paramètre <FileInfo>SwCompression= du fichier xml. SWIFTSTO met à disposition le fichier dans le répertoire adéquat, selon la règle suivante :

- Si le répertoire $SAG_DOWN/Responder/Requestor existe, le fichier y est stocké.
- Sinon, Si le répertoire $SAG_DOWN/Responder existe, le fichier y est stocké.
- Sinon, le fichier est stocké dans le répertoire $SAG_DOWN

Si le dépôt du fichier a échoué, SWIFTSTO renvoie au Back Office le compte-rendu négatif du transfert $TOM_DIR/out/xxxxxxxx.sto.nak. Voir la description du fichier d'acquittement au paragraphe '*Comptes-Rendus des Transferts*'.

Demande de Réception d'un Fichier par SWIFTNet

Dans ce qui suit &REQNUMB désigne le numéro de requête de transfert de Connect:Express (par exemple 03900007) et &PARTNID désigne le nom de partenaire (Par exemple CENTRAL).

Les procédures s'enchaînent de la façon suivante :

Etape 1 : FTI reçoit la demande de transfert (Get)

FTI crée un fichier $SAG_XML/Filename.TransferRef.par.
LTA-GetInit active le programme ConnectAccept et lui passe en paramètre le fichier xml des paramètres:

```
ConnectAccept –a17 $SAG_XML/FileName.TransferRef.par
```

ConnectAccept recherche le fichier WhiteList.xml : s'il le trouve, il vérifie que la demande est acceptée.
ConnectAccept recherche le fichier BlackList.xml : s'il le trouve, il vérifie que la demande n'est pas rejetée.
Si la demande est rejetée, un code non nul est retourné à FTI.
Si la demande est acceptée, le processus continue :

ConnectAccept déduit du fichier FindRule.xml que la règle numéro 000 s'applique. Il utilise la structure de Pi99 sur Get entrant (Pi99getIn) et le nom symbolique du Back Office.

ConnectAccept vérifie les droits d'accès, et si la recherche sur le Back Office n'est pas systématique (option F ou f), il recherche le fichier selon l'algorithme suivant :

- Dans le répertoire $SAG_DOWN/Responder/Requestor s'il existe … puis
- Dans le répertoire $SAG_DOWN/Responder s'il existe existe... puis
- Dans le répertoire $SAG_DOWN

Etape 2 : Réponse positive à FTI si le fichier a été trouvé

Récupération d'un Fichier du Back-office par SWIFTNet

Cette procédure est une alternative à la procédure précédente dans laquelle le fichier est préalablement mis à disposition sur la SAG.

Dans ce qui suit &REQNUMB désigne le numéro de requête de transfert de Connect:Express (par exemple 03900007) et &PARTNID désigne le nom de partenaire (Par exemple CENTRAL).

Les procédures s'enchaînent de la façon suivante :

FTI reçoit la demande de transfert (get) et crée un fichier *$SAG_XML/Filename.TransferRef.par*.

Etape 1 : Activation du programme de Connect:Express Unix for SWIFTNetpar FTI

LTA-GetInit active le programme ConnectAcceptInq et lui passe en paramètre le fichier xml des paramètres:

```
ConnectAcceptInquiry –a17 $SAG_XML/FileName.TransferRef.par
```

ConnectAcceptInq recherche le fichier WhiteList.xml : s'il le trouve, il vérifie que la demande est acceptée.
ConnectAcceptInq recherche le fichier BlackList.xml : s'il le trouve, il vérifie que la demande n'est pas rejetée.
Si la demande est rejetée, un code non nul est retourné à FTI.
Si la demande est acceptée, le processus continue.

ConnectAcceptInq déduit du fichier FindRule.xml que la règle numéro 000 s'applique. Il utilise la structure de Pi99 sur Get entrant (Pi99GetIn) et le nom symbolique du Back Office.

ConnectAcceptInq vérifie les droits d'accès, et si la recherche sur le Back Office n'est pas systématique (option F ou f), il recherche le fichier selon l'algorithme suivant :

- Dans le répertoire $SAG_DOWN/Responder/Requestor s'il existe … puis
- Dans le répertoire $SAG_DOWN/Responder s'il existe existe... puis
- Dans le répertoire $SAG_DOWN

Etape 2 : Si le fichier n'est pas présent sur la SAG, routage de la demande vers le Back-office

ConnectAccept vérifie les droits d'accès, et si la recherche sur le Back Office n'est pas interdite (option G), il soumet une requête de transfert à Connect:Express Unix for SWIFTNet :

p1b8preq	**Paramètre**	**Valeur**
	Fichier	SWIFTINQ
	Partenaire	CENTRAL
	Pi99 ou équivalent	Responder:Requestor:FileName
	Type de requête	Inquiry pour option F ou I, Normale pour option f ou i

Etape 3 : Réception des données sur la SAG

Connect:Express Unix for SWIFTNet déclenche la commande de fin de transfert :

SWIFTINQ	**Paramètre**	**Valeur**
	$1	Requête de transfert locale &REQNUMB
	$2	Partenaire (Back-office) &PARTNID
	$4	inq_&REQNUMB.tmp fichier des données
	$9	Responder**:**Requestor**:**FileName

Retour à ConnectAcceptInq.
ConnectAcceptInq effectue la compression demandée par le paramètre <InquiryCompression> de la règle de routage.

Etape 4 : Réponse positive à FTI si le fichier a été reçu, négative dans le cas contraire.

Annexe A

Exemples de Règles de Routage

Ce chapitre fournit des exemples de personnalisation afin d'illustrer les précédents chapitres. En particulier il décrit la règle de routage numéro zéro, fournie par défaut.

Routage par Défaut

La règle de routage numéro zéro correspond aux fonctionnalités de la première version de Connect:Express for SWIFTNet. Elle est basée sur le choix d'une structure de Pi99 et ne comporte aucun contrôle d'accès. La règle 000 s'applique à toutes les demandes internes et externes. Ce paragraphe décrit donc le Pi99, les squelettes xml, le fichier FindRule.xml et le fichier RuleList.xml.

Structure du Pi99 par Défaut

Le tableau ci-dessous indique la structure du Pi99 envoyé par le Back Office ou reçu par le Back Office selon les cas, Put ou Get, entrant ou sortant . Un Pi99 émis doit être placé dans les paramètres de la requête de transfert émise par le Back-office, un Pi99 reçu est construit par Connect:Express for SWIFTNet à partir de la règle de routage et peut être exploité par le Back-office en début ou fin de transfert de fichier.

Table des Pi99 par Défaut

Fonction	Sens	Transfert	Structure
Put sortant	Pi99 émis	SWIFTPUT	Responder:Filename:TransferIdent:compression
Get sortant	Pi99 émis	SWIFTGET	Responder:Filename:TransferIdent:compression
Store	Pi99 émis	SWIFTSTO	Responder:Requestor:Filename:compression
Acquittement	Pi99 reçu	SWIFTACK	Requestor:Filename:TransferIdent
Acquittement	Pi99 reçu	SWIFTNAK	Requestor:Filename:TransferIdent
Get sortant	Pi99 reçu	SWIFTFWD	Responder:Filename:TransferIdent
Put entrant	Pi99 reçu	SWIFTFWD	Requestor:Filename
Get entrant	Pi99 reçu	SWIFTINQ	Requestor:Filename

Exemple

Pi99 = ou=sftest2,o=swhqbebb:test001:reqnumb**:**ZIP

Le premier champ représente la partie variable du paramètre FTI numéro 2 (-a2 ResponderDN), le second champ correspond au paramètre FTI numéro 10 (-a10 FileName), le troisième champ doit être déterminé par le Back-office afin d'identifier le transfert. Ce champ n'est pas un paramètre FTI. Il est conservé et est utilisé pour l'acquittement de bout en bout. Le dernier champ indique le type de compression à positionner dans le fichier XML dans le champ SwCompression= du paramètre <a12> pour le put sortant, <a7> pour le get sortant et <FileInfo> pour le store. Ce paramètre peut prendre les valeurs ZIP, INFOZIP, GZIP, COMPRESS ou NONE.

Fichiers Squelettes de Paramètres xml

Les fichiers squelettes sont utilisés par Connect:Express for SWIFTNet pour traiter le Pi99 émis par le Back Office. Chaque squelette correspond à la structure de Pi99 correspondante du tableau des Pi99 ci dessus.

$TOM_DIR/exit/SwiftSto.xml

Store	Pi99 émis	SWIFTSTO	Responder:Requestor:Filename:compression

Squelette SwiftSto.xml par Défaut

```
<Cx:StoreFile>
<RequestorDN><1>,o=swift</RequestorDN>
<ResponderDN><2>,o=swift</ResponderDN>
<FileName><3></FileName>
<FileInfo>SwCompression=<4></FileInfo>
</Cx:StoreFile>
```

$TOM_DIR/exit/SwiftPut.xml

Put sortant	Pi99 émis	SWIFTPUT	Responder:Filename:TransferIdent:compression

Squelette SwiftPut.xml par Défaut

```
<Ft:PutFileRequest>
<a1>o=test,o=swift</a1>
<a2><1>,o=swift</a2>
<a3>TestService</a3>
<a4>RequestType</a4>
<a5>cn=certest10,o=develop,o=test,o=swift</a5>
<a6><3></a6>
<a7>TransferInfo</a7>
<a8>CENTRAL</a8>
<a9></a9>
<a10><2></a10>
<a11>File Description</a11>
<a12>SwCompression=<4>,coding=ascii</a12>
<a13>FALSE</a13>
<a14>FALSE</a14>
<a18>FALSE</a18>
</Ft:PutFileRequest>
```

$TOM_DIR/exit/SwiftGet.xml

Get sortant	Pi99 émis	SWIFTGET	Responder:Filename:TransferIdent:compression

Squelette SwiftGet.xml par Défaut

```
<Ft:GetFileRequest>
<a1>o=test,o=swift</a1>
<a2><1>,o=swift</a2>
<a3>TestService</a3>
<a4>RequestType</a4>
<a5>cn=certest10,o=develop,o=test,o=swift</a5>
<a6><3></a6>
<a7>SwCompression=<4> </a7>
<a8>CENTRAL</a8>
<a9></a9>
<a10><2></a10>
<a13>FALSE</a13>
<a16></a16>
<a24></a24>
</Ft:GetFileRequest>
```

Fichier FindRule – Sélection de la Règle 000

Le fichier FindRule.xml ci-dessous indique que toute demande est traitée avec la règle numéro 000.

Fichier FindRule.xml par Défaut

```
<?xml version="1.0" encoding="UTF-8"?>
<FindRuleCfg xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation='/test/cexpress/x141/exit/FindRule.xsd'>

<FindRuleList>

<FindRule Num="000" Status="Enabled">
<Select Action="Include" FTIParam="ResponderDN">.*</Select>
</FindRule>

</FindRuleList>

</FindRuleCfg>
```

Fichier RuleList – Description de la Règle 000

Le règle zéro présentée ci-dessous contient les valeurs par défaut : squelettes xml fournis, structures de Pi99 telles qu'elles sont décrites plus haut, nom de Back Office décrit dans le fichier des partenaires.

Elle est découpée en plusieurs parties afin de mettre en évidence les définitions correspondant aux différentes structures de Pi99 attendue par le Back office.

Règle numéro 000 par Défaut

```
<?xml version="1.0" encoding="UTF-8"?>
<RuleListCfg xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation='/test/cexpress/x141/exit/RuleList.xsd'>

<RuleList>

<Rule Num="000" Status="Enabled" Name="DEFAULT">
<PutXmlFile>$TOM_DIR/exit/SwiftPut.xml</PutXmlFile>
 <GetXmlFile>$TOM_DIR/exit /SwiftGet.xml</GetXmlFile>
 <StoXmlFile>$TOM_DIR/exit/SwiftSto.xml</StoXmlFile>
 <InquiryCompression>ZIP</InquiryCompression>
 <BackOffice>CENTRAL</BackOffice>
 <Compression>Y</Compression>
 <Pi99Delim>:</Pi99Delim>
```

Put entrant	Pi99 reçu	SWIFTFWD	Requestor:Filename

```
<Pi99PutIn Delim=":">
 <Field Name="Requestor">
  <Param SubName="o" SubNum="1" Offset="0" Length="12">RequestorDN</Param>
 </Field>
 <Field Name="Filename">
  <Param>FileName</Param>
 </Field>
</Pi99PutIn>
```

Get entrant	Pi99 reçu	SWIFTINQ	Requestor:Filename

```
 <Pi99GetIn Delim=":">
  <Field Name="Requestor">
   <Param SubName="o" SubNum="1" Length="12">RequestorDN</Param>
  </Field>
  <Field Name="Filename">
   <Param>FileName</Param>
  </Field>
 </Pi99GetIn>
```

Put sortant	Pi99 reçu	SWIFTACK	Responder:Filename:TransferIdent
Put sortant	Pi99 reçu	SWIFTNAK	Responder:Filename:TransferIdent

```
 <Pi99PutOut Delim=":">
  <Field Name="Responder">
   <Param Delim="," SubNum="1" Length="12">a2</Param>
  </Field>
  <Field Name="Filename">
   <Param>a10</Param>
  </Field>
  <Field Name="TransferDescription">
   <Param Length="8">a6</Param>
  </Field>
 </Pi99PutOut>
```

Get sortant	Pi99 reçu	SWIFTFWD	Responder:Filename:TransferIdent
Get sortant	Pi99 reçu	SWIFTNAK	Responder:Filename:TransferIdent

```
<Pi99GetOut Delim=":">
 <Field Name="Responder">
  <Param Delim="," SubNum="1" Length="12">a2</Param>
 </Field>
 <Field Name="Filename">
  <Param>a10 </Param>
 </Field>
 <Field Name="TransferDescription">
  <Param Length="8">a6</Param>
</Pi99GetOut>
```

```
</Rule>
</RuleList>
</RuleListCfg>
```

Choix du Back Office

L'exemple ci-dessous illustre l'utilisation de plusieurs règles pour communiquer avec plusieurs Back offices. La règle 'DEFAULT' permet de traiter avec le Back Office CE NTRAL0, la règle 'BACK OFFICE 2' permet de traiter avec le Back Office CENTRAL2. Chaque Back Office supporte des règles de routage différentes.

Le fichier FindRule.xml ci-dessous permet de diriger les demandes en fonction du paramètre ResponderDN vers l'un des deux Back office.

Fichier FindRule.xml pour Deux Back Offices

```
<?xml version="1.0" encoding="UTF-8"?>
<FindRuleCfg xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation='/test/cexpress/x141/exit/FindRule.xsd'>

<FindRuleList>

<FindRule Num="000" Status="Enabled">
<Select Action="Include" FTIParam="ResponderDN">^xxxxx,.*</Select>
</FindRule>

<FindRule Num="002" Status="Enabled">
<Select Action="Include" FTIParam="ResponderDN">^yyyyy,.*</Select>
</FindRule>

</FindRuleList>

</FindRuleCfg>
```

Pour les demandes internes, chaque Back Office indique le numéro de règle dans le Pi99 associé à la requête de transfert vers Connect:Express for SWIFTNet. Chaque Back Office gère sa propre structure de pi99 et donc un jeu de squelettes xml particuliers lui est associé.

Fichier RuleList.xml pour Deux Back Offices

```
<?xml version="1.0" encoding="UTF-8"?>
<RuleListCfg xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation='/test/cexpress/x141/exit/RuleList.xsd'>

<RuleList>



<Rule Num="000" Status="Enabled" Name="DEFAULT">

 <PutXmlFile>$TOM_DIR/exit/SwiftPut.xml</PutXmlFile>
 <GetXmlFile>$TOM_DIR/exit /SwiftGet.xml</GetXmlFile>
 <StoXmlFile>$TOM_DIR/exit/SwiftSto.xml</StoXmlFile>
 <InquiryCompression>ZIP</InquiryCompression>
 <BackOffice>CENTRAL0</BackOffice>
 <Compression>Y</Compression>
 <Pi99Delim>:</Pi99Delim>

 <Pi99PutIn Delim=":"> … </Pi99PutIn>

 <Pi99GetIn Delim=":">… </Pi99GetIn>

 <Pi99PutOut Delim=":">…</Pi99PutOut>

 <Pi99GetOut Delim=":">…</Pi99GetOut>
</Rule>



<Rule Num="002" Status="Enabled" Name="BACK OFFICE 2">

 <PutXmlFile>$TOM_DIR/exit/swiftput2.xml</PutXmlFile>
 <GetXmlFile>$TOM_DIR/exit /swiftget2.xml</GetXmlFile>
 <StoXmlFile>$TOM_DIR/exit/SwiftSto.xml</StoXmlFile>
 <InquiryCompression>ZIP</InquiryCompression>
 <BackOffice>CENTRAL2</BackOffice>
 <Compression>N</Compression>
 <Pi99Delim>=</Pi99Delim>

 <Pi99PutIn Delim="=> … </Pi99PutIn>

 <Pi99GetIn Delim="=">… </Pi99GetIn>

 <Pi99PutOut Delim="=">…</Pi99PutOut>

 <Pi99GetOut Delim="=">…</Pi99GetOut>
</Rule>


</RuleList>

</RuleListCfg>
```

White Liste

L'exemple ci-dessous illustre l'utilisation du contrôle d'accès pour définir des groupes de correspondants ayant des droits d'accès différents. Aucune 'Black Liste' n'étant définie le contrôle d'accès est complètement géré par la 'White Liste'.

Au cours de la recherche, le premier filtre vérifié est sélectionné : le filtre 'filter0 est donc l'accès par défaut.

'filter4' : Les demandes concernant les fichier 'CRI.*' à destination des serveurs .*,o=swift$ ne peuvent être que des get et provoquent systématiquement une recherche sur le Back Office.
'filter3' : Les demandes concernant les fichier 'depo.*' à destination des serveurs .*,o=swift$ ne peuvent être que des put.
'filter2' : Les demandes en provenance des partenaires ^o=ban.*,.* peuvent être des put et des Get et son autorisées à aller rechercher un fichier sur le Back Office.
'filter1' : Les demandes en provenance des partenaires ^o=bk.*,.* ou à destination du serveur .*,o=swift$ ne peuvent effectuer que des put.
'filter0' : toutes les autres demandes ne peuvent être que get, sans recherche de fichier sur le Back Office.

Exemple de 'White Liste'

```
<?xml version="1.0" encoding="UTF-8"?>
<AccessListCfg xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation=' /test/cexpress/x141/exit/AccessList.xsd'>

<AccessList>

<Filter Name="filter4" AccessMode="F" Logic="And" Status="Enabled">
<Select Parser="Regex" FTIParam="Filename" Match="True" CaseSensitive="False">^CRI.*</Select>
<Select Parser="Regex" FTIParam="ResponderDN" Match="True"
CaseSensitive="False">.*,o=swift$</Select>
</Filter>

<Filter Name="filter3" AccessMode="P" Logic="And" Status="Enabled">
<Select Parser="Regex" FTIParam="Filename" Match="True" CaseSensitive="False">^Depo.*</Select>
<Select Parser="Regex" FTIParam="ResponderDN" Match="True"
CaseSensitive="False">.*,o=swift$</Select>
</Filter>

<Filter Name="filter2" AccessMode="IP" Logic="And" Status="Enabled">
<Select Parser="Regex" FTIParam="RequestorDN" Match="True"
CaseSensitive="False">^o=ban.*,.*</Select>
</Filter>

<Filter Name="filter1" AccessMode="P" Logic="Or" Status="Enabled">
<Select Parser="Regex" FTIParam="ResponderDN" Match="True"
CaseSensitive="False">.*,o=swift$</Select>
<Select Parser="Regex" FTIParam="RequestorDN" Match="True" CaseSensitive="False">^o=bk.*,.*</Select>
</Filter>

<Filter Name="defaut" AccessMode="G" Logic="And" Status="Enabled">
<Select Parser="Regex" FTIParam="RequestorDN" Match="True" CaseSensitive="False">.*</Select>
</Filter>

</AccessList>
```

Annexe B

Connect:Express sur le Back Office

Ce chapitre décrit la mise en œuvre de transferts avec SWIFTNet sur un Back Office où le moniteur PeSIt est un Connect:Express. On se limite aux plates formes OS/390 et UNIX. Pour chacune d'elles des exemples sont donnés et les particularités sont précisées.

Configuration de Connect:Express pour les Transferts SWIFTNet

Quelle que soit la plate forme, le paramétrage concerne d'abord les partenaires et les fichiers. Les différences entre plates formes relèvent du détail.

Les définitions des partenaires et fichiers symboliques ne posent pas de problèmes particuliers. En supposant que les noms proposés dans ce document soient conservés, il faut donc ajouter les définitions suivantes dans les répertoires :

Partenaire SWIFTNET : impérativement de type 'other' car il faut traiter le Pi99 dans sa forme libre. L'alias doit correspondre à la définition 'CENTRAL' du côté de la SAG : nom = CENTRAL, password = CENTRAL .

Fichiers :

Sept fichiers doivent être définis, quatre en émission, 3 en réception. Le premier tableau ci-dessous indique le rôle de chacun des fichiers.

Nom Symbolique	Fonctions	Description
SWIFTPUT	put	Demande d'émission - sera suivi d'un transfert SWIFTACK ou SWIFTNAK
SWIFTGET	get	Demande de réception - sera suivi d'un transfert SWIFTFWD ou SWIFTNAK
SWIFTSTO	store	Emission pour mise à disposition sur la SAG, peut être suivi d'un transfert SWIFTNAK
SWIFTINQ	get externe	Emission sur demande externe
SWIFTFWD	put externe,get	Réception d'un fichier sur demande interne ou externe
SWIFTACK	put, get	Réception d'un acquittement positif
SWIFTNAK	put, get	Réception d'un acquittement négatif

Le tableau suivant indique le sens de transfert, quel composant dépose la requête, le demandeur du transfert et les informations liées à chacun des fichiers. Ces informations permettront éventuellement de mettre en place des procédures d'exploitation au travers des exits et commandes utilisateur ou même dans l'applicatif. Lorsque la SAG est indiquée dans le nom du demandeur il s'agit de Connect:Express for SWIFTNet. Sauf dans le cas de la requête en Hold, le demandeur est le composant qui a déposé la requête.

Nom Symbolique	Sens	Requête (Type)	Demandeur	Informations transportées
SWIFTPUT	Emission	Back Office	Back Office	Fichier de données, Pi99
SWIFTGET	Emission	Back Office	Back Office	Fichier de travail (vide) , Pi99
SWIFTSTO	Emission	Back Office	Back Office	Fichier de données, Pi99
SWIFTINQ	Emission	Back Office (HOLD)		Fichier de données
SWIFTINQ	Emission	SAG	SAG	Fichier de données, Pi99
SWIFTFWD	Réception	SAG	SAG	Fichier de données, Pi99
SWIFTACK	Réception	SAG	SAG	Fichier d'acquittement positif , Pi99
SWIFTNAK	Réception	SAG	SAG	Fichier d'acquittement négatif , Pi99

Lorsque le Pi99 est envoyé par le Back Office (cas où le Back Office dépose la requête), il doit être structuré par l'applicatif conformément aux règles de routage convenues.

Lorsque Le Pi99 est reçu de la SAG (cas où la SAG dépose la requête), il peut être traité avec les mots clés &Pi99 : ceci permet de traiter les informations SWIFTNet dans des procédures automatiques.

Les fichiers d'acquittement envoyés par la SAG peuvent être traités par l'applicatif car ils ont une structure constante et homogène.

Ces dernières informations permettent de mettre en place des procédures d'intégration qui ne seront pas traitées ici.

L'association d'un Pi99 à une requête de transfert peut se faire de différentes façons, les deux paragraphes suivants rappellent l'usage sur OS/390 et sur UNIX. Les trois premiers transferts du tableau précédent sont concernés.

Connect:Express OS/390

Ce paragraphe présente la définition du partenaire SWIFTNet, l'une des définitions de fichier symbolique SWIFT et précise l'usage du Pi99. Il indique en particulier comment les transferts SWIFT apparaissent dans le Journal.

Partenaire SWIFTNet

Partenaire SWIFTNET sur un Back Office Connect:Express OS/390

```
TOM4111     PARTENAIRE DE TOM4 A VISUALISER (2/4)
 OPTION ===>                          -ENTREE- : SUITE, -PF3- : ANNULER  X :
EXIT
 TYPE: COMPAT.,PESIT-E
 MOD: PSR0008  03/03/14 03:37:34 -
 NOM SYMBOLIQUE        : SWIFTNET           DPCSID ALIAS          : CENTRAL
 MOT DE PASSE TOM      : SWIFTNET           DPCPSW ALIAS          : CENTRAL
 ETAT INITIALISATION   : E                  CLASSE APM RECEPTION  : A
 UTILISATEUR RACF      : RACUSR             GROUPE RACF           : -

 NATURE PARTENAIRE     : O
 PROT.SESSION NUM.-T.  : 5    : 2           T-SECURITE RSA/DES    : -
 REESSAI AUTOMATIQUE   : NON

 TYPES DE LIAISON      : I    : -           PARTENAIRE ADJACENT    : -
 EFF. TOT.-ENT.-SOR.   : 020  : 010  : 010  T-REGULATION FLUX SLD  : -

 SNA: LUNAME  : -         LOGMODE    : -        LOGDATA   : -         DISC
: N
 X25: MCHMSC  : -     ADR.DIST.  : -               ADR.LOC.  : -
      GFA     : -     UDF        : -               TAXATION  : -
      SERVICES COMPLEMENTAIRES   : -
 IP : ADRES.  : 010.078.061.028   PASV   : -       PORT      : 6000
      HOTE    : -                                 PROFIL    : -
 NOTE  : CONNECT:EXPRESS WINDOWS FOR SWIFTNET
```

Fichier SWIFT

Fichier SWIFTPUT sur un Back Office Connect:Express OS/390

```
TOM4111---- REPERTOIRE DES FICHIERS (2/5) ------------------------------------
 OPTION ===>

 NOM SYMBOLIQUE             : SWIFTPUT   MODE: NORMAL

 ETAT INITIALISATION ... ===> E          E: EN-SERVICE H: HORS-SERVICE

 DIRECTION ............. ===> T          T:TRANSMETTRE R:RECEVOIR *:TRANS./REC.
 PARTENAIRE RECEPTEUR .. ===> SWIFTNET   'NOM', £LISTE, */$$ALL$$ OU $$API$$
 PARTENAIRE EMETTEUR ... ===> -          'NOM', £LISTE, */$$ALL$$ OU $$API$$

 PRIORITE .............. ===> 1          0:URGENT 1:RAPIDE 2:NORMAL 3:LENT
 TYPE DEFINITION DU DSN  ===> D          D:DYNAMIQUE F:FIXE
 REGLE ALLOCATION ...... ===> 1          0:CREER/REMPLACER 1:PREALL. 2:A CREER
                                         3:EXIT A:SERVEUR APPLICATIF
 TYPE FICHIER .......... ===> S          S/M/P/PU/V/VU/UU/SU
 PRESENTATION .......... ===> 05         COMPRESS.,T.DONNEES (01-24)
 MEMBRE CHARG./DECHARG.. ===> -          OPTIONNEL
 TABLE DE SECURITE ..... ===> -          OPTIONNEL

 OPTION   : MODIFIER                     MAJ   : 03/03/24 09:08 PSR0008
 -ENTREE- : ECRAN SUIVANT                -PF3- : ANNULATION
```

```
TOM4111---- REPERTOIRE DES FICHIERS (3/5) ------------------------------------
 OPTION ===>

 NOM SYMBOLIQUE             : SWIFTPUT   DEF.: D   ALL.: 1   TYPE: S   DIR.: T
                              SDB= Y
 DSN LOCAL ............. ===> -
 NUMERO DE GDG ......... ===> -           +XX OU -XX
 1 NOM SYMBOLIQUE UNITE  ===> -           UNITNAME
   NOM DES VOLUMES ..... ===> -          -         -         -         -
 2 SMS DATA-STOR-MGMT .. ===> -                    -                   -
 DISPOSITION ........... ===> OLD         SHR/OLD/NEW

 TYPE ALLOCATION ....... ===> -           CYL/TRK UB/KB/MB(?BYTES-SMSSDB)
 ALLOCATION PRIM./SEC. . ===> -     -     1-4 CAR. NUMERIQUES
 NBRE BLOCS "DIRECTORY"  ===> -           SI PARTITIONNE
 FORMAT ENREGISTREMENT . ===> -           F,FB,FBA,FBM,V,VB,VBA,VBM,VBS,VS,U
 LONGUEUR ENREG./BLOC .. ===> -     -     1-5 CAR. NUMERIQUES
 RETENTION (EXPDT/RETPD) ===> -           X'CCAAQQQ',E'AAQQQ'/R'NNNN'

 Dsn distant/Pi99 ...... ===> -                                          <
 TYPE/STRUCTURE/MODE FTP ===> -   -  -    EN/AN/I,F/R,B/C/S
 UNIQUE ............ FTP ===> NON         Oui/Non
 NOTE ===> -
```

```
TOM4111---- REPERTOIRE DES FICHIERS (4/5) -----------------------------------
 OPTION ===>

 NOM SYMBOLIQUE             : SWIFTPUT   DEF.: D   ALL.: 1   TYPE: S   DIR.: T
                              UPRFCT= Y
 --- S : VERIFICATION DES SYMBOLES
 V
 EMISSION :
   EXIT DE DEBUT ....... ===> -          NOM DU PROGRAMME DE CONTROLE
 _ COMMANDE DE DEBUT ... ===> -
   EXIT DE FIN ......... ===> -          NOM DU PROGRAMME DE CONTROLE
 _ COMMANDE DE FIN ..... ===> -

 RECEPTION :
   EXIT DE DEBUT ....... ===> -          NOM DU PROGRAMME DE CONTROLE
 _ COMMANDE DE DEBUT ... ===> -
   EXIT DE FIN ......... ===> -          NOM DU PROGRAMME DE CONTROLE
 _ COMMANDE DE FIN ..... ===> -


 OPTION   : MODIFIER                     MAJ   : 03/03/24 09:08 PSR0008
 -ENTREE- : ECRAN SUIVANT                -PF3- : ECRAN PRECEDENT
```

Les procédures d'exploitation en usage sur le Back Office peuvent prendre en charge les transferts SWIFT. Des procédures spéciales peuvent être mises en place sur les transferts qui reçoivent des informations SWIFT comme le Pi99 ou les fichiers d'acquittement : SWIFTFWD, SWIFTINQ, SWIFTACK et SWIFTNAK.

Transfert SWIFT

Les transferts SWIFTNet sont caractérisés par l'usage d'un Pi99. Sur OS/390 plusieurs méthodes sont possibles pour passer un Pi99 dans les paramètres de la requête de transfert.

En batch ou par l'interface opérateur, il existe un champ de taille courte (44 caractères) et un champ de taille longue (80 caractères). Dans le contexte des transferts SWIFNet les deux champs sont disponibles.

Les deux exemples ci-dessous présentent les deux possibilités, en utilisant chaque interface. Il faut noter que l'utilisation des minuscules est nécessaire.

La page suivante représente une requête de transfert SWIFT par l'interface opérateur avec utilisation du champ de 44 caractères 'Rdsn Distant/Pi99' : dans ce champ il est possible de mettre des minuscules et d'insérer des mots clés comme le mot clé &REQNUMB .

Requête SWIFTNET sur un Back Office Connect:Express OS/390

```
TOM4111     TRANSFERT NORMAL
 OPTION ===>                                                          CSGB

 SOUS-SYSTEME ===> TOM4
 FICHIER .... ===> SWIFTPUT
 DIRECTION .. ---> T         (T/R)
 PARTENAIRE . ---> SWIFTNET

 DSNAME ....  ---> TEST.SWIFT.PUT


 TYPE ....... ---> N         (N/H/I/K/U)
 MODE ....... ---> I         (I/D)
 LIAISON..... ---> I         ( /C/I/S/T/X)
 CLASSE ..... ---> A         (A-Z/*)
 PRIORITE ... ---> 1         (0-3)
 MEMBRE ..... --->           (BFX)
 VERIFICATION ---> OUI       (OUI/NON)
 EXTENSION .. ---> OUI       (OUI/NON) ALIAS/ORG/DST/API-ETB3/SEC/RGR.
 NOTE-> N
             X EXIT, -ENTREE- REQUETE, -PF1- AIDE TRC, -PF3- FIN
```

```
TOM4111     EXTENSION DE TRANSFERT                    NOMS INITIALISES     !
 OPTION ===>                                                          CSGB


 SOUS-SYSTEME  :   TOM4
 FICHIER ....  :   SWIFTPUT                                    EN-SERVICE
 DIRECTION ..  :   T         (T/R)     <- T
 PARTENAIRE .  :   SWIFTNET            <- SWIFTNET 52          EN-SERVICE
 DSN LOCAL ..  :   TEST.SWIFT.PUT                              DYNAMIQUE
               <- ?                                       ?    SEQUENTIEL
 Rdsn/Pi99 .. ---> O=TEST:ESSAI1:&REQNUMB:ZIP                     <  *1
               <-
 FTP T/S/M .. ---> '' ---> '  ---> '   <- -   -  -     STOU   ---> '    <- N
 GROUPE RACF. ---> ''''''''                            ('VALEUR'/'BLANC')
 Org/Dest ... --->          < --->          <          *1
 ET SEULEMENT SI TOM EST ACTIF :
 TABLE SECU.  ---> ''                                  ('VALEUR'/'BLANC')
 Alias id/psw --->          < --->          <          *1
 V----- S : DETAIL
 '' Api ....  --->
                     <                          *1:('VALEUR'/'valeur'/'BLANC')
             X EXIT, -ENTREE- CONFIRMATION, -PF1- AIDE TRC, -PF3- RETOUR
```

L'utilisation du mot clé &REQNUMB permettra un suivi de bout en bout à partir du numéro de requête d'origine.

Page suivante, l'écran présente l'utilisation du champ API sur 80 caractères.

```
TOM4111     EXTENSION DE TRANSFERT                        NOMS INITIALISES     !
 OPTION ===>                                                                CSGB


 SOUS-SYSTEME  :   TOM4
 FICHIER ....  :   SWIFTPUT                                          EN-SERVICE
 DIRECTION ..  :   T         (T/R)     <- T
 PARTENAIRE .  :   SWIFTNET            <- SWIFTNET 52                EN-SERVICE
 DSN LOCAL ..  :   TEST.SWIFT.PUT                                    DYNAMIQUE
               <- ?                                            ?     SEQUENTIEL
 Rdsn/Pi99 .. --->                                                    <  *1
               <-
 FTP T/S/M .. ---> '' ---> '  ---> '   <- -   -  -     STOU   ---> '    <- N
 GROUPE RACF. ---> ''''''''                            ('VALEUR'/'BLANC')
 Org/Dest ... --->          < --->          <          *1
 ET SEULEMENT SI TOM EST ACTIF :
 TABLE SECU.  ---> ''                                  ('VALEUR'/'BLANC')
 Alias id/psw --->          < --->          <          *1
 V----- S : DETAIL
 '' Api ....  ---> P:O=TEST:ESSAI1:&REQNUMB:ZIP
                     <                           *1:('VALEUR'/'valeur'/'BLANC')
             X EXIT, -ENTREE- CONFIRMATION, -PF1- AIDE TRC, -PF3- RETOUR
```

Le préfixe P : indique que la chaîne de caractères est à placer dans le Pi99.

Les deux exemples suivants présentent les deux options, en utilisant le programme batch P1B2PRQ2 :

//STEP01EXECPGM=P1B2PRQ2,PARM='SSN=TOM3,**CAPS=OFF**'

Il faut utiliser l'option CAPS=OFF qui permet l'usage des minuscules.

SEND	SFN=SWIFTPUT,	SYMBOLIC FILE NAME
	SPN=SWIFTNET,	SYMBOLIC PARTNER NAME
	TYP=N,	REQUEST TYPE
	CLS=A,	REQUEST CLASS
	PRT=1,	REQUEST PRTY
	RDS=O=TEST:ESSAI1:&REQNUMB:ZIP,	**44 caractères disponibles**
	DSN=TEST.SWIFT.PUT	
SEND	SFN=SWIFTPUT,	SYMBOLIC FILE NAME
	SPN=SWIFTNET,	SYMBOLIC PARTNER NAME
	TYP=N,	REQUEST TYPE
	CLS=A,	REQUEST CLASS
	PRT=1,	REQUEST PRTY
	A48='P:O=TEST:ESSAI1:&REQNUMB:ZIP',	**46 caractères …. +**
	A34='',	**…. 34 caractères disponibles**
	DSN=TEST.SWIFT.PUT	

La figure suivante indique comment utiliser l'interface applicative L0B2Z20 à partir d'un programme applicatif :

L'utilisation des minuscules est demandée en indicant la valeur hexadécimale X'20' dans le champ EXHPRTY de l'en-tête de la demande . Le Pi99 doit être placé dans l'un des deux autres paramètres.

```
          03 D0B2ZEX1-EXHENTRY.
…..
               05  D0B2ZEX1-EXHPRTY PIC X(1) .
…..
               05 D0B2ZEX1-EX1ENTRY PIC X(360).
…..
                    07  D0B2ZEX1-EX1SRDS  PIC X(44).
……
                    07  D0B2ZEX1-EX1SAPI  PIC X(82).
```

Connect:Express UNIX

Ce paragraphe présente la définition du partenaire SWIFTNet, l'une des définitions de fichier symbolique SWIFT et précise l'usage du Pi99. Il indique en particulier comment les transferts SWIFT apparaissent dans le Journal.

Partenaire SWIFTNet

Partenaire SWIFTNET sur un Back Office Connect:Express UNIX

```
C:E/UNIX 141-1 --------------- PARTNERS DIRECTORY ------------------------ ce01
OPTION ===>
SYMBOLIC NAME : SWIFTNET
PASSWORD ............... : SWIFTNET          PASSWORD OF PARTNER
INITIALIZATION STATUS . : E                 E:ENABLE H:DISABLE
PARTNER TYPE .......... : O                 T/O
PROTOCOL NUMBER ....... : 3                 1:ETEBAC 3, 2:FTP, 3:PESIT
SESSION TABLE NUMBER .. : 6                 1->9 SESSION TABLES
X25 PORT ............... : 0                 X25 DEVICE NAME
MAX. NO. CONNECTIONS .. : 10/10/10          01->64 TOT/IN/OUT
TYPE OF CONNECTION .... : T                 X, P, T OR M
X25 DIAL NUMBER ....... :                   1-15 CHARACTERS
LOCAL DIAL NUMBER ..... :                   1-15 CHARACTERS
EXTRA NETWORK FIELD ... :                   'USER-DATA-FIELD'
FACILITIES ............. :
TCPIP HOST ............. : swift
TCPIP ADDRESS ......... :                   PORT .......... : 7000
DPCSID ALIAS .......... : CENTRAL           DEF FTP FILE .. :
DPCPSW ALIAS .......... : CENTRAL
DO YOU WANT TO GO ON ?
OPTION : VIEW UPD : 98/08/04 10:41 root
-ENTER- NEXT FIELD -F3- CANCEL -F8- COMPLETION
```

Fichier SWIFT

Fichier SWIFTPUT sur un Back Office Connect:Express UNIX

```
C:E/UNIX 141-1 ---------------- FILES DIRECTORY -------------------------- ce01
OPTION ===>
SYMBOLIC NAME : SWIFTPUT
INITIALIZATION STATUS . : E              E:ENABLE H:DISABLE
DIRECTION ............. : T              T:TRANSMIT R:RECEIVE *:EITHER
RECEIVING PARTNER ..... : SWIFTNET       'NAME',#LISTE, $$ALL$$
TRANSMITTING PARTNER .. : SWIFTNET       'NAME',#LISTE, $$ALL$$
PRIORITY .............. : 0              0:URGENT 1:FAST 2:NORMAL
DEFINITION TYPE ....... : D              D:DYNAMIC F:FIXED
PRESENTATION TABLE .... : 5              1 -> 9 PRESENTATION TABLE
PARAMETER CARDS FILE    : Y              Y/N
SPACE TO RESERVE ...... : N              Y/N
ALLOCATION RULE ....... : 0              0:INDIF., 1:PREALL., 2:TO CREATE
PHYSICAL NAME ......... :
RECORD FORMAT ......... : TV             TF, TV, BF, BU, T*, B*, **
RECORD LENGTH ......... : 08192          1-5 NUMERIC CHARAC.
REMOTE DSN (FTP) ...... :
TYPE/STRUCTURE/MODE FTP :                E/A/I/*,F/R/*,B/S/*
STORE UNIQUE (FTP) .... :                Y/N FA : Y/N
OPTION : VIEW                UPD : 98/03/13 14:59 pla
-ENTER- NEXT FIELD -F3- CANCEL -F8- COMPLETION
```

```
C:E/UNIX 141-1 ---------------- FILES DIRECTORY -------------------------- ce01
OPTION ===>
SYMBOLIC NAME : SWIFTPUT     DEFINITION :       D DIRECTION : R
TRANSMISSION :
START EXIT ............ : ............
START COMMAND ......... : ............
END EXIT .............. : ............
END COMMAND ........... : ............
RECEPTION :
START EXIT ............ : ............
START COMMAND ......... : ............
END EXIT .............. : ............
END COMMAND ........... : ............
DO YOU WANT TO GO ON ?       UPD : 19980722112010 C:E 141-1
-ENTER- NEXT FIELD -F3- CANCEL -F8- COMPLETION
```

Les procédures d'exploitations en usage sur le Back Office peuvent prendre en charge les transferts SWIFT. Des procédures spéciales peuvent être mises en place sur les transferts qui reçoivent des informations SWIFT comme le Pi99 ou les fichiers d'acquittement : SWIFTFWD, SWIFTINQ, SWIFTACK et SWIFTNAK.

Transfert SWIFT

Les transferts SWIFTNet sont caractérisés par l'usage d'un Pi99. Sur UNIX plusieurs méthodes sont possibles pour passer un Pi99 dans les paramètres de la requête de transfert.

En batch ou par l'interface opérateur, il existe un champ de taille courte (44 caractères), le 'REMOTE PHYSICAL NAME'. En batch il existe aussi un second champ de 254 caractères.

Les exemples ci-dessous présentent les trois possibilités, en utilisant chaque interface.

Le premier exemple représente une requête de transfert SWIFT par l'interface opérateur avec utilisation du champ de 44 caractères 'REMOTE PHYSICAL NAME' : dans ce champ il est possible de mettre des minuscules et d'insérer des mots clés comme le mot clé &REQNUMB .

Requête SWIFTNET sur un Back Office Connect:Express UNIX

```
C:E/UNIX 141-1 -------------- TRANSFER REQUEST ----------------------- ce01
OPTION ===>
FILE ................. : SWIFTPUT
DIRECTION ............ : T               (T/R)
PARTNER .............. : SWIFTNET.
DPCSID ALIAS ......... :                 DPCPSW ALIAS .......... :  .
PHYSICAL NAME ........ : /usr/lib/libsock.a.........................
REMOTE PHYSICAL NAME .. : o=test:ESSAI1:&REQNUMB:ZIP.................
LABEL:.....................................................
RECORD FORMAT ........ : TV              BU TF, TV, BF, BU
RECORD LENGTH ........ : 08192
TYPE/STRUCTURE/MODE FTP :                E/A/I/*,F/R/*,B/S/*
STORE UNIQUE (FTP) .... :                Y/N FA : O/N
TYPE ................. :                (N/I/H)
TYPE OF CONNECTION .... :                (X/P/T)
PRIORITY ............. :                (0/1/2)
DATE ................. :                (YYYYMMDDHHMMSS)
API FIELD (ETEBAC3 : 80 CHARACTERS FOR CARD)
1...5....0....5....0....5....0....5....0....5....0....5....0....5....0....5....0
DO YOU WANT TO GO ON ?
-ENTER- NEXT FIELD -F3- CANCEL -F8- COMPLETION
```

L'utilisation due mot clé &REQNUMB permettra un suivi de bout en bout à partir du numéro de requête d'origine.

Les deux exemples suivants présentent les options de l'utilitaire P1B8PREQ : le champ /UDF peut contenir 44 caractères, le champ /P99 peut en recevoir 254.

```
$TOM_DIR/itom/p1b8preq "/SFN=SWIFTPUT/PRT=1/LNK=T/SPN=SWIFTNET"
                       "/DSN=/tmp/TOM.tmp" "/UDF=o=test:ESSAI1:&REQNUMB:ZIP"


$TOM_DIR/itom/p1b8preq "/SFN=SWIFTPUT/PRT=1/LNK=T/SPN=SWIFTNET"
                       "/DSN=/tmp/TOM.tmp" "/P99=o=test:ESSAI1:&REQNUMB:ZIP"
```

Annexe C

Expressions Régulières

Les filtres des listes de contrôle d'accès utilisent des expressions régulères étendues. Les appels standards regcomp et regexec sont utilisés pour tester si une chaîne satisfait ou non une expression régulière.

Consulter la page manuel de regexec ainsi que les documentations du système d'exploitation pour plus d'informations sur les expressions régulières.

Le tableau ci dessous rappelle quelques principes de base de la syntaxe des expressions régulières :

Syntaxes	**Signification et exemple**
^ (Chapeau)	Début de chaîne. Exemple : «**^mot.*»** signifie 'toute chaîne de caractères commençant par «mot», y compris « mot » '
$ (Dollar)	Fin de chaîne. Exemple : «**.*mot$»** signifie 'toute chaîne de caractères terminée par «mot», y compris « mot » '
. (Point)	N'importe quel caractère. Exemple : «.+ban,.+reg.+» représente 'toute chaîne de caractères de la forme «*xxx*ban,*yyy*reg*zzz*» dans laquelle les chaînes xxx, yyy et zzz sont composées d'un nombre non nul de caractères quelconques.
* + ?	De zéro à n fois : Voir exemples ci-dessus. De 1 à n fois Zéro ou 1 fois
[A-Z0-9] [^A-Z0-9]	Un caractère de A à Z ou de 0 à 9. Un caractères quelconque n'appartenant pas à cette famille. Exemple : «**^Début[A-Z0-9]?fin$»** représente toute chaîne de caractères de la forme «Début*X*fin» avec au plus un caractère numérique ou alphabétique majuscule entre «Début» et «fin» .
\	Permet d'utiliser un des caractères de la syntaxe (., +, *, $, ^). Exemple : «[\., ;=-]» désigne l'ensemble des caractères point, virgule, point-virgule, égal et tiret.

Un utilitaire $TOM_DIR/exit/testre est fourni afin d'aider à l'écriture des filtres de WhiteList.xml et BlackList.xml. Il permet de tester rapidement si une chaîne de caractères correspond à une expression régulière ou pas.

Pour lancer le programme de test taper :

```
./testre 'nom de fichier'
```

Le fichier passé en paramètre est un fichier éditable dont chaque ligne a le format suivant :

```
string=string to test,expression=regular expression

Exemple :  string=ABCD,expression=.*CD
```

Avec un fichier re.txt contenant les lignes suivantes:

```
string=ABCD,expression=.*CD
string=U,expression=.*CD
```

./testre re.txt affiche le résultat suivant:

```
String to test     = ABCD
Regular expression = .*CD
regexec MATCH.string=ABCD.expression=.*CD
String to test     = U
Regular expression = .*CD
regexec NOMATCH.string=U.pattern=.*CD
```

La figure suivante donne un aperçu des possibilités offertes :

```
String to test     = ABCD
Regular expression = .*CD
regexec MATCH.string=ABCD.expression=.*CD
String to test     = U
Regular expression = .*CD
regexec NOMATCH.string=U.pattern=.*CD
String to test     = ABCD
Regular expression = [A].*CD
regexec MATCH.string=ABCD.expression=[A].*CD
String to test     = UABCD
Regular expression = ^[A].*CD
regexec NOMATCH.string=UABCD.pattern=^[A].*CD
String to test     = 1.get.uuu.err
Regular expression = .*get[/.][^.]+[/.]err$
regexec MATCH.string=1.get.uuu.err.expression=.*get[/.][^.]+[/.]err$
String to test     = 1.getuuu.err
Regular expression = .*get[/.][^.]+[/.]err$
regexec NOMATCH.string=1.getuuu.err.pattern=.*get[/.][^.]+[/.]err$
String to test     = 11.get.uuu.err
Regular expression = ^.*get[/.][^.]+[/.]err$
regexec MATCH.string=11.get.uuu.err.expression=^.*get[/.][^.]+[/.]err$
String to test     = ou=abc01,o=bank01,o=swift
Regular expression = ^ou=abc[0-9]{2},o=bank[0-9]{2},o=swift$
regexec MATCH.string=ou=abc01,o=bank01,o=swift.expression=^ou=abc[0-9]{2},o=bank[0-
9]{2},o=swift$
String to test     = ou=abcXX,o=bankYY,o=swift
Regular expression = ^ou=abc[0-9]{2},o=bank[0-9]{2},o=swift$
regexec NOMATCH.string=ou=abcXX,o=bankYY,o=swift.pattern=^ou=abc[0-9]{2},o=bank[0-
9]{2},o=swift$
String to test     = A
Regular expression = [A]{2}
regexec NOMATCH.string=A.pattern=[A]{2}
String to test     = AA
Regular expression = [a-zA-Z]{2}
regexec MATCH.string=AA.expression=[a-zA-Z]{2}
String to test     = 111
Regular expression = [A]{3}
regexec NOMATCH.string=111.pattern=[A]{3}
String to test     = 5
Regular expression = [0-9]+
regexec MATCH.string=5.expression=[0-9]+
String to test     = :
Regular expression = [^0-9]+
regexec MATCH.string=:.expression=[^0-9]+
```