Sips Office Server 2.04
Guide des composants

Version 1.01 – Octobre 2010

REACH YOUR TARGETS »

Contact

By phone: +33 (0)811 107 033
By fax: +33 (0)811 107 033
By email: sips@atosorigin.com

www.atosworldline.com

Sommaire

1. INTRODUCTION

L'objectif de ce document est de vous aider à mettre en œuvre les différents composants Sips Office Server. Pour ce faire, nous allons tout d'abord décrire le fonctionnement général du composant. Cette description va vous permettre de visualiser les différentes étapes d'une requête. Nous listerons ensuite les étapes de la mise en œuvre du composant avant de les décrire précisément.

Remarque : Ce document ne décrit pas comment vous interfacer avec votre système d'information ou votre base de données. Dans les exemples de requêtes XML fournis, les variables sont déjà renseignées, vous devrez programmer la lecture et la mise à jour des données de votre système d'information.

Pré-requis :

- Savoir programmer une connexion socket en protocole TCP/IP (des exemples en ASP, C, JAVA, PERL et PHP sont fournis avec l'API Sips Office Server).
- Le compilateur ou l'interpréteur associé au langage choisi.
- L'API Sips Office Server installé (cf. *LE GUIDE D'INSTALLATION*).
- Avoir des notions de XML.
- Avoir lu la *PRESENTATION FONCTIONNELLE*

Conventions d'écriture

Dans tout le document, les conventions d'écriture suivantes seront utilisées :

- Les renvois à d'autres documentations seront notés en majuscules et en italique.
  ex *: LE GUIDE D'INSTALLATION*
- Les champs du composant seront notés en **gras**.
  ex : **transaction_id**
- Les chemins et les noms de fichiers seront notés en *italique*.
  ex : *pathfile*

2. FONCTIONNEMENT GENERAL DES COMPOSANTS

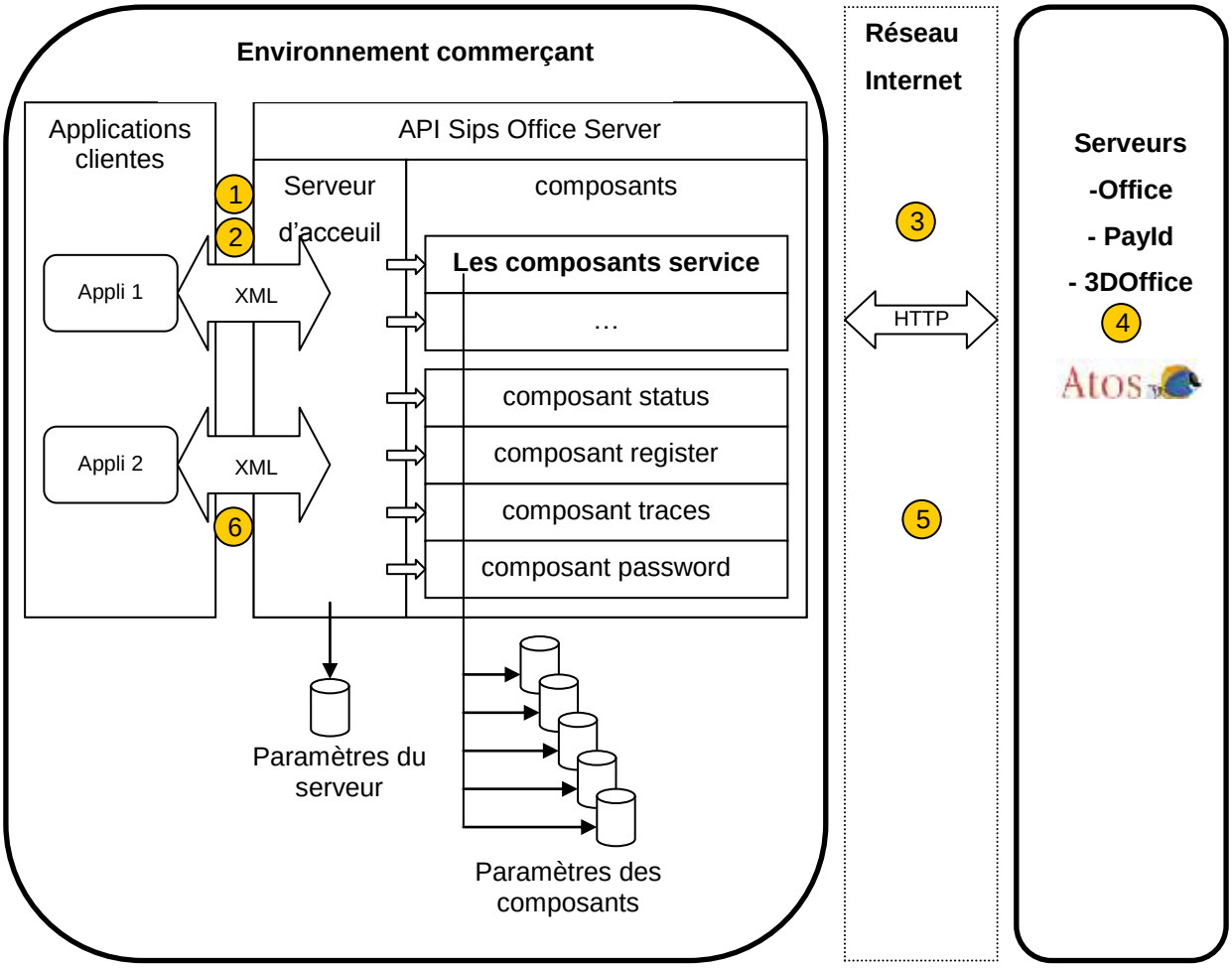

Figure 1: Shéma des connexions entre les applications clientes du commerçant et le serveur Office d'Atos Origin

Pour créer, modifier ou diagnostiquer une transaction, le commerçant doit envoyer une requête au composant qui se connectera ensuite au serveur Office de Atos Worldline. Ce processus peut se résumer en six étapes représentées sur la Figure 1 :

1/ connexion socket en protocole TCP/IP entre l'application cliente et l'API Sips Office Server

2/ envoi de la requête associée à l'opération souhaitée au format XML

3/ connexion socket en protocole TCP/IP et envoi d'une requête HTTP entre le composant et le serveur Office de Atos Worldline à l'adresse *office.sips-atos.com*

4/ accès à la base de données Atos Worldline, demande d'autorisation vers les services bancaires si nécessaire

5/ le composant reçoit la réponse du serveur Office de Atos Worldline et ferme la connexion socket

6/ l'application cliente reçoit la réponse du composant et ferme la connexion socket

Liste des composants de type service :

Office : permet d'effectuer des transactions et des opérations de gestion de caisse

Diag : permet de vérifier l'état d'une transaction

PayId : permet de gérer des abonnés

Checkout : permet d'effectuer des transactions 3D-Secure

CashManagement : permet d'effectuer des opérations sur des abonnés

Identification d'une transaction

Le diagnostic ou la modification d'une transaction via une opération nécessite de pouvoir identifier cette transaction de manière unique. Pour ce faire, les transactions dans la base de données Atos Worldline sont différenciées grâce à la clé formée des données suivantes :

- **merchant_country** : pays du commerçant
- **merchant_id** : numéro de commerçant
- **payment_date** : date de création.
- **transaction_id** : identifiant de transaction

Ces quatre champs sont donc toujours présents dans les requêtes de création, de modification ou de diagnostic d'une transaction.

3. MISE EN ŒUVRE D'UN COMPOSANT

La mise en place complète d'un composant se déroule en quatre phases:

- installation et paramétrage du composant
- premiers tests sur le serveur de démonstration Office avec les fichiers d'exemple fournis dans le répertoire *examples* du répertoire principal de l'API Sips Office Server et les exemples de requête XML présentés dans ce document
- le développement de vos applications clientes pour l'envoi de requête vers l'API Server
- les tests sur le serveur de production

Toutes ces étapes seront décrites plus précisément dans la suite de ce guide.

3.1 ENREGISTREMENT DU COMPOSANT

Vous devez enfin enregistrer votre composant pour qu'il soit connu de l'API Server. Pour enregistrer votre composant et lister les composants enregistrés, référez-vous respectivement au chapitre **la commande « ENREGISTRER UN NOUVEAU COMPOSANT »** et au chapitre **la commande « LISTER LES COMPOSANTS »** du *GUIDE D'ADMINISTRATION*.

3.2 PROCEDURE DE TEST D'UN COMPOSANT

3.2.1 Vérification de l'installation du composant

La première étape consiste à vérifier le paramétrage et l'enregistrement du composant. Vous devez, tout d'abord, créer une connexion socket en protocole TCP/IP entre votre application cliente et l'API Server. Pour ce faire, vous pouvez vous baser sur les fichiers d'exemples livrés avec l'API Server (répertoire *examples* du répertoire principal de l'API Server) et décrits dans le *GUIDE DU DEVELOPPEUR*. Vous devez ensuite envoyer une requête XML de vérification (checkoffice, checkdiag, …). Suite à cette requête XML l'API Server ne tente pas de se connecter au serveur Office d'Atos Origin, mais il vérifie la présence des fichiers *pathfile* et *certificat*, et renvoie une réponse à votre application cliente.

3.2.2 Test de connexion vers le serveur de démo

Cette étape va vous permettre de tester la connexion entre l'API Server et le serveur de démo Sips d'Atos Origin. Vous pouvez transmettre tous les types de requêtes existants (demande d'autorisation, annulation, validation, remboursement etc…). Les serveurs de démo et de production Office d'Atos Origin ne sont pas localisés sur les mêmes machines. Par conséquent, si vos requêtes passent par un proxy et/ou un firewall, vous aurez sans doute à le configurer pour laisser passer les requêtes de démo ET de production. Pour la démo l'url du serveur office est http://rcet-office.sips-atos.com:2001, en production l'url du serveur office est http://office.sips-atos.com:2001

Pour effectuer votre test de connexion, vous devez, tout d'abord, créer une connexion socket en protocole TCP/IP entre votre application cliente et l'API Server en vous basant sur les fichiers d'exemples livrés avec l'API Server (répertoire *examples* du répertoire principal de l'API Server) et décrits dans le *GUIDE DU DEVELOPPEUR*.

L'API server est livré avec un commerçant «011223344553333 » et un certificat de démonstration.

Pour envoyer une requêt XML vers le serveur de démonstration d'Atos origin, vous devrez paramétrer le **merchant_id** à « 011223344553333 » et le **merchant_country** à « fr ».

Le composant testé et le serveur utiliseront alors le certificat certif.fr.011223344553333 livré avec le composant.

Le mode de fonctionnement du serveur de démo est décrit au niveau de chaque fonction dans le paragraphe « règles de simulation du serveur de démo».

3.2.3 Comment passer sur le serveur de production ?

La dernière phase est le passage en mode « pré-production » (cf. paragraphe ***Comment en profiter*** de la *PRESENTATION FONCTIONNELLE*). A partir de ce moment les demandes d'autorisation ne sont plus simulées comme sur le serveur de démonstration, mais elles empruntent le circuit complet du réseau bancaire. Cette phase va permettre de contrôler la bonne inscription de votre contrat bancaire.

Le passage en « pré-production » se fait par l'activation du certificat du commerçant qui va remplacer le certificat de démonstration (certif.fr.011223344553333).

1. Copier le certificat de « production », qui vous a été transmis sur disquette ou par mail, dans le même répertoire que le certificat de « démonstration » et renommer ce certificat en certif.fr.<my_merchant_id>, où <my_merchant_id> est votre numéro de boutique.
2. Remplacer le numéro de la boutique de démonstration (011223344553333) par votre numéro de boutique (champ merchant_id) dans vos requêtes vers l'API Sips Office Server.
3. Vous devez faire au minimum un test de demande d'autorisation acceptée. Pour ce test, vous devez utiliser un numéro de carte réelle. Tant que vous êtes en phase de pré-production, les demandes d'autorisation ne sont pas débitées.
4. Pour le passage en « production », voir le document de *PRESENTATION FONCTIONNELLE* paragraphe ***Comment en profiter*.**

3.2.4 Tests en Pre-Production

La phase de pré-production sert à valider la bonne inscription de votre boutique.

Nous vous invitons à valider l'ouverture de votre contrat VAD en transmettant des requêtes de demande d'autorisation avec un vrai numéro de carte. Vous devez recevoir un accord (response_code = 00). Ces transactions ne sont pas enregistrées dans notre base de données et ne sont pas envoyées en banque.

Nous vous invitons à valider l'ouverture de vos droits sur les différentes opérations office. S'il vous manque un droit sur une opération, vous recevrez un code 40. Si le droit sur cette opération est ouvert vous recevrez un code 25 (transaction non trouvée) car les transactions de pré-production ne sont pas enregistrées dans notre base de données.

3.3 DESINSTALLATION DU COMPOSANT

Pour désinstaller le composant Office, vous devez :

1) Arrêter votre API Sips Office Server (cf. chapitre **la commande « ARRETER LE SERVEUR »** du *GUIDE D'ADMINISTRATION*)
2) Supprimer le répertoire *composant* du répertoire principal de l'API Sips Office Server
3) Supprimer le fichier *composant.jar* du répertoire *server/components/service* du répertoire principal de l'API Server.
4) Supprimer le fichier *config.xml* localisé dans le répertoire *server/config/* de l'API Server les lignes suivantes :

```
<pathfile id="composant" path="chemin_absolu_vers_le_fichier_pathfile" />
<alternateTrace id="composant" path="chemin_absolu_vers_le_répertoire_traces" prefix="office" />
```

5) Supprimer les fichiers de traces du composant
6) Supprimer les listes d'accès (cf. paragraphe **Configuration des listes d'accès** du *GUIDE D'INSTALLATION*) les adresses des machines qui ne doivent plus accéder à l'API Sips Office Server
7) Supprimer l'autorisation de connexion au serveur Office d'Atos Origin de votre firewall si aucun des composants restants ne l'utilisent.

Au cours du redémarrage de l'API Sips Office Server, la liste des composants restants sera remise à jour.

4. LE COMPOSANT OFFICE 3.17

4.1 FONCTIONNEMENT GENERAL DU COMPOSANT OFFICE

Le composant Office permet de créer des transactions et d'agir sur des transactions précédemment créées via le composant Office et les services Sips Payment Web, Sips Subscription ou Sips Office Extranet.

Plusieurs opérations peuvent être appliquées à une transaction pour la modifier : la validation, l'annulation, le forçage, le remboursement ou la duplication. Pour plus d'information sur la création des transactions et sur les opérations permettant de les modifier, référez-vous à la *PRESENTATION FONCTIONNELLE*.

4.2 INSTALLATION ET PARAMETRAGE DU COMPOSANT

4.2.1 Installation

4.2.1.1 Liste des objets livrés

Le composant Office est livré sous la forme d'un fichier *composant_office_317.tar* contenant les fichiers suivants :

Fichier *Version.txt*	Fichier précisant l'environnement dans lequel le composant a été compilé et testé.
Répertoire *office/param*	
certif.fr.011223344553333	Certificat de la boutique de démonstration
pathfile	Fichier des informations proxy et du chemin d'accès au certificat
Répertoire *server/components/service*	
office.jar	archive du composant Office

4.2.1.2 Copie des fichiers

Vous devez tout d'abord décompresser le fichier *composant_office_317.tar*.

Pour un OS de type Unix :

Utiliser la commande suivante : tar –xvf *composant_office_317.tar*

Pour Windows :

Utiliser un logiciel de décompression qui accepte les fichiers d'extension « .tar ».

Après avoir décompressé le fichier *composant_office_317.tar*, vous devez copier le répertoire *office* et tout ce qu'il contient dans le répertoire principal de l'API Sips Office Server. Le répertoire principal de l'API Sips Office Server contient également les répertoires associés aux autres composants que vous utilisez, ainsi que le répertoire *server*.

Vous devez également copier le fichier *office.jar* dans le répertoire *server/components/service* du répertoire principal de l'API Sips Office Server.

4.2.2 Configuration

4.2.2.1 Les paramètres du proxy

Si vous utilisez un proxy pour vous connecter à Internet, vous devez le paramétrer dans le fichier *pathfile* copié précédemment dans le répertoire *office/param*. Ce paramétrage permettra au composant Office de contacter le serveur Office d'Atos Origin via l'Internet (cf. étape 3 de la Figure 1).

Si vous n'utilisez pas de proxy pour vous connecter à Internet, vous ne devez rien indiquer dans les champs suivants :

PROXY_HOST : adresse IP ou nom du proxy.

PROXY_PORT : le numéro de port du proxy.

En cas de doute, votre administrateur réseau pourra vous indiquer les valeurs à renseigner.

Exemple :
Si vous passez par le proxy d'adresse IP 111.11.11.11 et de port 7878 vous devez paramétrer les champs :

PROXY_HOST!111.11.11.11!

PROXY_PORT!7878!

4.2.2.2 Configuration du chemin vers les fichiers certificats

Le chemin vers les fichiers certificat est localisé dans le fichier *pathfile* copié précédemment dans le répertoire *office/param*. Ce chemin doit être renseigné dans le champ F_CERTIFICATE et est unique pour tous les certificats du composant Office. Tous les fichiers certificats que vous utiliserez avec le composant Office devront donc être localisés dans un même répertoire.

Remarque : Les extensions du fichier certificat ne doivent en aucun cas être indiquées dans le champ F_CERTIFICATE. En effet, ces extensions correspondent aux paramètres **merchant_country** et **merchant_id** transmis dans les requêtes associées aux différentes opérations. Ils seront donc ajoutés par le composant au paramètre F_CERTIFICATE pour obtenir le chemin complet du certificat souhaité.

Exemple Windows :

Si le fichier *certif.fr.011223344553333* est localisé dans le répertoire *C:\API_Server\office\param*, vous devez renseigner le champ F_CERTIFICATE comme suit :

F_CERTIFICATE!C:\API_Server\office\param\certif!

Exemple autres OS :

Si le fichier *certif.fr.011223344553333* est localisé dans le répertoire */home/API_Server/office/param/*, vous devez renseigner le champ F_CERTIFICATE comme suit :

```
F_CERTIFICATE!/home/API_Server/office/param/certif!
```

4.2.2.3 Configuration du chemin vers le fichier pathfile

Vous devez paramétrer dans le fichier *config.xml* le chemin vers le fichier *pathfile* du composant Office. Le fichier *config.xml* est localisé dans le répertoire *server/config/* de l'API Server.

Vous trouverez ci-dessous le paragraphe concerné dans le fichier *config.xml* :

```
<components>
<pathfile id="example" path="chemin_du_fichier_pathfile" />
</components>
```

Vous devez ajouter la ligne suivante entre les balises <components> et </components> :

```
<pathfile id="office" path="chemin_absolu_vers_le_fichier_pathfile" />
```

avec :

id : l'identifiant du composant (ici office).

path : le chemin absolu vers le fichier *pathfile*.

Remarque

Le nom indiqué dans le champ « id » doit être identique au nom de l'archive sans l'extension « .jar » (cf. paragraphe 4.2.1.1)

Exemple Windows :

Si le fichier *pathfile* est localisé dans le répertoire *C:\API_Server\office\param*, vous devez renseigner la ligne comme suit :

```
<pathfile id="office" path="C:\API_Server\office\param\pathfile" />
```

Exemple autres OS :

Si le fichier *pathfile* est localisé dans le répertoire */home/API_Server/office/param*, vous devez renseigner la ligne comme suit :

```
<pathfile id="office" path="/home/API_Server/office/param/pathfile" />
```

4.2.2.4 Configuration des traces

Vous devez paramétrer dans le fichier *config.xml* le chemin des fichiers de traces du composant Office. Le fichier *config.xml* est localisé dans le répertoire *server/config/* de l'API Server.

Vous trouverez ci-dessous le paragraphe concerné dans le fichier *config.xml* :

```
<watchdog>
        <pollingTimer>6000</pollingTimer>
        <survPort>7183</survPort>
        <trace level="0" sizeLimit="1000" unit="Line" path="C:\chemin_racine\server\trace"
prefix="APIServer" />
<!--    <alternateTrace id="example" path="C:\chemin_racine\server\trace" prefix="example" />
        -->
        <alternateTrace id="" path="./" prefix="" />
</watchdog>
```

Vous devez ajouter la ligne suivante au-dessus du tag « </watchdog> » (la ligne avec l'identifiant example est en commentaire).

```
<alternateTrace id="office" path="chemin_absolu_vers_le_répertoire_traces" prefix="office" />
```

avec :

id : l'identifiant du composant (ici office).

path : le chemin absolu vers le répertoire des traces. Vous pouvez utiliser pour vos traces le répertoire *server/traces* du répertoire principal de l'application ou tout autre répertoire de votre choix.

prefix : préfixe du fichier de traces du composant. Dans l'exemple ci-dessus, le nom du composant est utilisé, mais tout autre préfixe est également valable.

Remarque

- le nom indiqué dans le champ « id » doit être identique au nom de l'archive sans l'extension « .jar » (cf. paragraphe 4.2.1.1).
- La ligne « <alternateTrace id="" path="./" prefix="" /> » est nécessaire au démarrage de l'API Server dans le cas ou aucun composant n'est installé, il est donc important de ne pas la modifier.

Exemple Windows :

Si le chemin du répertoire trace est *C:\API_Server\server\traces*, vous devez renseigner la ligne comme suit :

```
<alternateTrace id="office" path="C:\API_Server\server\traces" prefix="office" />
```

Exemple autres OS :

Si le chemin du répertoire trace est */home/API_Server/server/traces*, vous devez renseigner la ligne comme suit :

```
<alternateTrace id="office" path="/home/API_Server/server/traces" prefix="office" />
```

4.2.2.5 Adresse IP pour la configuration de votre firewall

Si vous utilisez un firewall pour vous connecter au réseau Internet, vous devez paramétrer au niveau de celui-ci l'adresse IP et le port du serveur Office d'Atos Origin. Pour communiquer avec le serveur Office d'Atos Origin votre firewall doit vous permettre une connexion socket en protocole TCP/IP vers l'adresse IP 193.56.46.110 port 2001.

4.3 DESCRIPTION DETAILLEE DU COMPOSANT

Ce chapitre décrit les opérations disponibles avec le composant Office. Pour chacune de ces opérations, des exemples de requête et de réponse au format XML sont fournis, ainsi que les listes des paramètres de la requête et de la réponse avec des liens vers le dictionnaire des données.

Avant de décrire les fonctions disponibles, nous allons préciser les aspects de la sécurisation du transfert des informations entre le composant Office et le serveur Office d'Atos Origin.

Remarques

- Afin d'alléger le code des requêtes, les entêtes standard XML ne sont pas nécessaires, mais sont ajoutées par l'API Server pour vérification lors du parsing XML (utilisation d'un DTD). Par contre, lors des réponses de l'API Server, vers l'application cliente, aucune référence à un DTD n'est incluse dans le message. Il est donc important de désactiver la validation lors du parsing, afin de ne pas déclencher des erreurs durant cette phase.
- Pour des raisons de lisibilité, toutes les requêtes décrites dans ce chapitre sont présentées sur plusieurs lignes. Toutefois, lors de l'appel à l'API Server, elles doivent être codées sans retours chariot.

4.3.1 Requête de vérification vers le composant Office : checkOffice

4.3.1.1 Objet

Cette fonction permet de tester :

- la communication entre l'application cliente que vous développez et l'API Server
- le bon fonctionnement de l'API Server
- la configuration du composant Office
  - l'accès au fichier *pathfile*
  - l'accès au fichier certificat

4.3.1.2 Paramètres de la requête et de la réponse

Dans le tableau ci-dessous sont décrits tous les champs de la requête de vérification.

nom du champ	Facultatif/ Obligatoire	Utilisation
component	O	nom du composant appelé
name	O	nom de la fonction appelée
merchant_country	O	code pays du commerçant
merchant_id	O	identifiant du commerçant

Dans le tableau ci-dessous sont décrits tous les champs de la réponse de la vérification.

nom du champ	valeur
pathfile	renseigné à OK si le fichier *pathfile* a bien été trouvé
certificate	renseigné à OK si le fichier *certif.<merchant_country>.<merchant_id>* a bien été trouvé
message	renseigné si le fichier *pathfile* ou *certificat* n'a pu être lu

4.3.1.3 Exemple de requête et de réponse XML

Ci-dessous est présenté un exemple de requête XML de vérification. La signification des champs **merchant_country** et **merchant_id** est fournie dans le *DICTIONNAIRE DES DONNEES*. Ces champs correspondent respectivement à la première et la deuxième extension du fichier certificat (exemple : le fichier *certif.fr.011223344553333* correspond à **merchant_country**=fr et **merchant_id**=011223344553333).

```
<service component="office" name="checkoffice">
<checkoffice merchant_id="011223344553333" merchant_country="fr"/>
</service>
```

Lors de la vérification, le composant Office va contrôler l'accès en lecture au fichier *pathfile* paramétré dans le fichier *config.xml*, puis l'accès en lecture au fichier certificat (*certif.<merchant_country>.<merchant_id>*) dont le chemin générique est indiqué dans le fichier *pathfile*. Si ces accès sont corrects, la réponse suivante sera envoyée :

```
<response pathfile="OK" certificate="OK" />
```

Si ces accès ne sont pas corrects, la réponse suivante sera envoyée :

```
<Error message="message d'erreur"/>
```

La liste des messages d'erreur est précisée dans le tableau ci-dessous.

Messages d'erreur	Cause	Solution
C:\API_Server\office\pam\pathfile (Le chemin d'accès spécifié est introuvable)	le fichier *pathfile* configuré dans le fichier *config.xml* n'est pas accessible en lecture	Vérifier les droits d'accès du fichier *pathfile* et corriger le chemin du fichier *pathfile* si nécessaire (cf. paragraphe 4.2.2.3).
Cannot open certificat. (C:\API_Server\office\pam\certif.fr.011223344553333)	le fichier certif.fr.01122334455222 n'est pas accessible en lecture	Vérifier les droits d'accès du fichier certificat *certif.fr.011223344553333* et corriger le chemin génériques des fichiers certificats si nécessaire (cf. paragraphe 4.2.2.2).

4.3.2 Requête de demande d'autorisation carte bancaire : author

4.3.2.1 Objet

Cette fonction permet au commerçant disposant des coordonnées bancaires de ses clients d'effectuer lui-même les demandes d'autorisation et ainsi de créer des transactions dans Sips (cf. *PRESENTATION FONCTIONNELLE*).

Pour les cartes VISA, MASTERCARD et CB, le cryptogramme visuel peut être renseigné (cf. Annexe K du *DICTIONNAIRE DES DONNEES*).

4.3.2.2 Paramètres de la requête et de la réponse

Dans le tableau ci-dessous sont décrits tous les champs de la requête de demande d'autorisation.

Remarque : Si une carte ne possède pas de date de validité, le champ **card_validity** doit être présent dans la requête et non renseigné.

Bien que les champs **cvv_key** et **cvv_flag** soient tous les deux facultatifs il doit y avoir une cohérence entre eux. Si le champ **cvv_key** est renseigné, le champ **cvv_flag** doit obligatoirement contenir la valeur 1 (cf. Annexe K du *DICTIONNAIRE DES DONNEES*).

Remarque : Les champs mpi_cavv, mpi_cavv_alogorithm, mpi_eci, mpi_tx_status, mpi_xid et security_indicator sont utilisés dans le cadre d'une transaction 3D Secure. Bien que ces champs soient facultatifs il doit y avoir une cohérence entre eux (cf. Annexe Q du *DICTIONNAIRE DES DONNEES*).

Nom du champ	Facultatif/ Obligatoire	Utilisation
component	**O**	nom du composant appelé
name	**O**	nom de la fonction appelée
amount	**O**	montant de la transaction
capture_day	**F**	délai d'envoi en banque
capture_mode	**F**	mode d'envoi en banque
card_number	**O**	numéro de la carte
card_type	**O**	type de carte utilisée
card_validity	**O**	date de validité de la carte
currency_code	**O**	code de la monnaie utilisée
customer_ip_address	**F**	adresse IP de l'internaute
cvv_flag	**F**	indique la présence ou non du cryptogramme visuel
cvv_key	**F**	cryptogramme visuel
data	**F**	paramétrage particulier
merchant_country	**O**	code pays du commerçant
merchant_id	**O**	identifiant du commerçant
order_id	**F**	numéro de commande du commerçant
mpi_cavv	**F**	valeur de vérification de l'authentification du porteur
mpi_cavv_algorithm	**F**	algorithme de génération du CAVV
mpi_eci	**F**	indicateur de commerce électronique
mpi_tx_status	**F**	Résultat de l'authentification

mpi_xid		identifiant de la transaction 3D Secure
order_validity	**F**	non utilisé
order_channel	**F**	canal de paiement
origin	**F**	permet d'identifier le programme à l'origine de la demande d'autorisation
payment_pattern	**F**	type de paiement utilisé
return_context	**F**	contexte de la transaction
security_indicator	**F**	indicateur de sécurité de la transaction
transaction_id	**O**	numéro de la nouvelle transaction
transmission_date	**O***	date de la requête
version	**O***	version du composant

** renseigné par le composant*

Dans le tableau ci-dessous sont décrits tous les champs de la réponse de la demande d'autorisation.

nom du champ	origine de la valeur
authorisation_id	renseigné par le serveur bancaire (si transaction autorisée)
bank_response_code	renseigné par le serveur bancaire
complementary_code	renseigné par le serveur Office
complementary_info	renseigné par le serveur Office
credit_amount	non renseigné
currency_code	identique à la requête
cvv_response_code	renseigné par le serveur bancaire (si demande d'autorisation avec cryptogramme effectuée)
avs_response_code	renseigné par le serveur Office (si demande d'autorisation avec vérification d'adresse effectuée)
data	identique à la requête
new_amount	renseigné par le serveur Office
new_status	renseigné par le serveur Office
response_code	renseigné par le serveur Office
transaction_date	renseigné par le serveur Office
transaction_time	renseigné par le serveur Office
transaction_certificate	renseigné par le serveur Office (si transaction acceptée)

4.3.2.3 Exemple de requête et de réponse XML

Ci-dessous est présenté un exemple de requête XML de demande d'autorisation.

```
<service component="office" name="author">
<author origin="Batch"
merchant_id="011223344553333"
merchant_country="fr"
transaction_id="26"
amount="12300"
currency_code="978"
card_number="498465123123123200"
cvv_flag="1"
cvv_key="400"
card_validity="200411"
card_type="VISA"
return_context="context"
order_id="OI_131100_8744"
capture_mode="VALIDATION"
capture_day="2"
data=""
order_validity=""
mpi_cavv="Q0FWVkNBVlZDQVZWQ0FWVkNBVlY="
mpi_cavv_algorithm="2"
mpi_eci="05"
mpi_tx_status="Y"
mpi_xid ="MjAwNTA1MjAxMDE3NDY4OTEyNzg="
security_indicator="20" />
</service>
```

Pour connaître la signification de ces différents champs référez-vous au *DICTIONNAIRE DES DONNEES.*

Ci-dessous est présentée la réponse XML de la demande d'autorisation précédente.

```
<response component="office" name="author">
<author response_code="00"
transaction_time="114147"
transaction_date="20030801"
transaction_certificate="1059730907"
authorisation_id="1059"
new_status="TO_VALIDATE"
new_amount="12300"
credit_amount=""
currency_code="978"
data=""
cvv_response_code="4D"
bank_response_code="00"
complementary_code=""
complementary_info="" />
</response>
```

Pour connaître la signification de ces différents champs référez-vous au *DICTIONNAIRE DES DONNEES.*

4.3.2.4 Règles de simulation du serveur de démo

Le processus d'autorisation vers l'acquéreur est simulé, Il est donc possible de renseigner n'importe quel numéro de carte sans aucune conséquence.

Le code réponse (**response_code**) de la transaction simulée est donné par les deux derniers chiffres du numéro de la carte bancaire.

Exemple :	numéro de carte	code réponse	
	4974934125497800	00	(acceptée)
	4972187615205	05	(refusée)

Les cryptogrammes se terminant par 00 ou 40 conduisent à une acceptation.

4.3.3 Requête de demande d'autorisation elv : elvauthor

4.3.3.1 Objet

Cette fonction permet à un commerçant disposant des coordonnées compte de ses clients d'effectuer lui-même les demandes d'autorisation pour des paiements « ELV » (paiement par débit direct possible uniquement pour un acheteur allemand chez un commerçant allemand).

4.3.3.2 Paramètres de la requête et de la réponse

Dans le tableau ci-dessous sont décrits tous les champs de la requête de demande d'autorisation ELV.

nom du champ	Facultatif/ Obligatoire	Utilisation
component	O	nom du composant appelé
name	O	nom de la fonction appelée
account_id	O	numéro de compte de l'internaute
amount	O	montant de la transaction
bank_number	O	numéro de la banque de l'internaute
capture_day	F	délai d'envoi en banque
capture_mode	F	mode d'envoi en banque
currency_code	O	code de la monnaie utilisée
customer_ip_address	F	adresse IP de l'internaute
data	F	paramétrage particulier
merchant_country	O	code pays du commerçant
merchant_id	O	identifiant du commerçant
order_id	F	numéro de commande du commerçant
order_validity	F	non utilisé
origin	F	permet d'identifier le programme à l'origine de la demande d'autorisation
return_context	F	contexte de la transaction
transaction_id	O	numéro de la nouvelle transaction
transmission_date	O*	date de la requête
version	O*	version du composant

** renseigné par le composant*

Dans le tableau ci-dessous sont décrits tous les champs de la réponse de la demande d'autorisation ELV.

nom du champ	origine de la valeur
authorisation_id	renseigné par le serveur bancaire (si transaction autorisée)
bank_response_code	renseigné par le serveur bancaire
complementary_code	renseigné par le serveur Office
complementary_info	renseigné par le serveur Office
credit_amount	non renseigné
currency_code	identique à la requête
data	identique à la requête
new_amount	renseigné par le serveur Office
new_status	renseigné par le serveur Office

response_code	renseigné par le serveur Office
transaction_date	renseigné par le serveur Office
transaction_time	renseigné par le serveur Office
transaction_certificate	renseigné par le serveur Office (si transaction acceptée)

4.3.3.3 Exemple de requête et de réponse XML

Ci-dessous est présenté un exemple de requête XML de demande d'autorisation ELV.

```
<service component="office" name=" elvauthor">
<elvauthor origin="API SERVER"
merchant_id="011223344553333"
merchant_country="fr"
bank_number="12345678"
account_id="9876543200"
transaction_id="174649"
data=""
amount="3200"
return_context="context"
currency_code="978"
order_id="OI_131100_8744"
capture_mode="VALIDATION"
capture_day="6"/>
</service>
```

Pour connaître la signification de ces différents champs référez-vous au *DICTIONNAIRE DES DONNEES.*

Ci-dessous est présentée la réponse XML de la demande d'autorisation ELV précédente.

```
<response component="office" name="elvauthor">
<elvauthor response_code="00"
transaction_time="084916"
transaction_date="20060329"
transaction_certificate="14c4a16d2bdc"
authorisation_id="084916"
new_status="TO_VALIDATE"
currency_code="978"
data=""
bank_response_code="00"
complementary_code=""
complementary_info="" />
</response>
```

Pour connaître la signification de ces différents champs référez-vous au *DICTIONNAIRE DES DONNEES.*

4.3.3.4 Règles de simulation du serveur de démo

Le processus d'autorisation est simulé. Il est donc possible de renseigner n'importe quel numéro de compte sans aucune conséquence.

Le code réponse (**response_code**) de la transaction simulée est donné par les deux derniers chiffres du numéro de compte.

Exemple :

numéro de compte	code réponse	
1234567800	00	(acceptée)
8765432105	05	(refusée)

4.3.4 Requête de validation d'une transaction : validate

4.3.4.1 Objet

Cette fonction permet de déclencher l'envoi en banque d'une transaction (cf. *PRESENTATION FONCTIONNELLE*).

4.3.4.2 Paramètres de la requête et de la réponse

Dans le tableau ci-dessous sont décrits tous les champs de la requête de validation d'une transaction.

Nom du champ	Facultatif/ Obligatoire	Utilisation
amount	O	montant à valider
currency_code	O	code de la monnaie utilisée
merchant_country	O	code pays du commerçant
merchant_id	O	identifiant du commerçant
origin	F	permet d'identifier le programme à l'origine de la validation
payment_date	O	date de création de la transaction à valider
transaction_id	O	numéro de la transaction à valider
transmission_date	O*	date de la requête
version	O*	version du composant

* *renseigné par le composant*

Dans le tableau ci-dessous sont décrits tous les champs de la réponse de la validation d'une transaction.

Nom du champ	Origine de la valeur
authorisation_id	renseigné par le serveur bancaire (si transaction autorisée)
bank_response_code	renseigné par le serveur bancaire
complementary_code	renseigné par le serveur Office
complementary_info	renseigné par le serveur Office
credit_amount	non renseigné
currency_code	identique à la requête
cvv_response_code	renseigné par le serveur Office (si demande d'autorisation avec cryptogramme effectuée)
data	identique à la requête
new_amount	renseigné par le serveur Office
new_status	renseigné par le serveur Office
response_code	renseigné par le serveur Office
transaction_date	renseigné par le serveur Office
transaction_time	renseigné par le serveur Office
transaction_certificate	renseigné par le serveur Office (si opération acceptée)

4.3.4.3 Exemple de requête et de réponse XML

Ci-dessous est présenté un exemple de requête XML de validation.

```
<service component="office" name="validate">
<validate origin="Batch"
merchant_id="011223344553333"
merchant_country="fr"
transaction_id="26"
payment_date="20030801"
amount="8300"
currency_code="978"/>
</service>
```

Pour connaître la signification de ces différents champs référez-vous au *DICTIONNAIRE DES DONNEES.*

Ci-dessous est présentée la réponse XML de la validation précédente.

```
<response component="office" name="validate">
<validate response_code="00"
transaction_time="145204"
transaction_date="20030801"
transaction_certificate="b37437b40293"
authorisation_id=""
new_status="TO_CAPTURE"
new_amount="8300"
credit_amount=""
currency_code="978"
data=""
cvv_response_code=""
bank_response_code=""
complementary_code=""
complementary_info="" />
</response>
```

Pour connaître la signification de ces différents champs référez-vous au *DICTIONNAIRE DES DONNEES.*

4.3.4.1 Règles de simulation du serveur de démo

Pas de règle de simuation, pour valider une transaction, il faut au prélable créer la transaction en mode VALIDATION (capture_mode = VALIDATION)

4.3.5 Requête d'annulation d'une transaction : cancel

4.3.5.1 Objet

Cette fonction permet de modifier le montant d'une transaction à envoyer en banque (cf. *PRESENTATION FONCTIONNELLE*).

Attention

Toute opération de caisse CANCEL envoyée entre 22h et 03h (heure locale française) retournera systématiquement un code réponse 24, et l'opération ne sera pas traitée.

4.3.5.2 Paramètres de la requête et de la réponse

Dans le tableau ci-dessous sont décrits tous les champs de la requête d'annulation partielle ou totale d'une transaction.

Nom du champ	Facultatif/ Obligatoire	Utilisation
amount	O	montant à annuler
currency_code	O	code de la monnaie utilisée
merchant_country	O	code pays du commerçant
merchant_id	O	identifiant du commerçant
origin	F	permet d'identifier le programme à l'origine de la modification du montant
payment_date	O	date de création de la transaction à modifier
transaction_id	O	numéro de la transaction à modifier
transmission_date	O*	date de la requête
version	O*	version du composant

* renseigné par le composant

Dans le tableau ci-dessous sont décrits tous les champs de la réponse de l'annulation partielle ou totale d'une transaction.

Nom du champ	Origine de la valeur
authorisation_id	non renseigné
bank_response_code	non renseigné
complementary_code	non renseigné
complementary_info	non renseigné
credit_amount	non renseigné
currency_code	identique à la requête
cvv_response_code	non renseigné
data	identique à la requête
new_amount	renseigné par le serveur Office
new_status	renseigné par le serveur Office
response_code	renseigné par le serveur Office
transaction_date	renseigné par le serveur Office
transaction_time	renseigné par le serveur Office
transaction_certificate	renseigné par le serveur Office (si opération acceptée)

4.3.5.3 Exemple de requête et de réponse XML

Ci-dessous est présenté un exemple de requête XML d'annulation.

```
<service component="office" name="cancel">
<cancel origin="Batch"
merchant_id="011223344553333"
merchant_country="fr"
transaction_id="26"
payment_date="20030801"
amount="1300"
currency_code="978"/>
</service>
```

Pour connaître la signification de ces différents champs référez-vous au *DICTIONNAIRE DES DONNEES.*

Ci-dessous est présentée la réponse XML de l'annulation précédente.

```
<response component="office" name="cancel">
<cancel response_code="00"
transaction_time="145727"
transaction_date="20030801"
transaction_certificate="692dbf19761d"
authorisation_id=""
new_status="TO_CAPTURE"
new_amount="7000"
credit_amount=""
currency_code="978"
data=""
cvv_response_code=""
bank_response_code=""
complementary_code=""
complementary_info="" />
</response>
```

Pour connaître la signification de ces différents champs référez-vous au *DICTIONNAIRE DES DONNEES.*

4.3.5.4 Règles de simulation du serveur de démo

Pas de règle de simulation, pour annuler une transaction, il faut au prélable créer la transaction en mode capture automatique (capture_mode = AUTHOR_CAPTURE)

4.3.6 Requête de remboursement d'une transaction : credit

4.3.6.1 Objet

Cette fonction permet de recréditer le compte de l'acheteur (cf. *PRESENTATION FONCTIONNELLE*).

Attention

Toute opération de caisse CREDIT envoyée entre 22h et 03h (heure locale française) retournera systématiquement un code réponse 24, et l'opération ne sera pas traitée.

4.3.6.2 Paramètres de la requête et de la réponse

Dans le tableau ci-dessous sont décrits tous les champs de la requête d'un remboursement partiel ou total d'une transaction.

Nom du champ	Facultatif/ Obligatoire	Utilisation
amount	O	montant à rembourser
currency_code	O	code de la monnaie utilisée
merchant_country	O	code pays du commerçant
merchant_id	O	identifiant du commerçant
origin	F	permet d'identifier le programme à l'origine du remboursement
payment_date	O	date de création de la transaction à rembourser
transaction_id	O	numéro de la transaction à rembourser
transmission_date	O*	date de la requête
version	O*	version du composant

* renseigné par le composant

Dans le tableau ci-dessous sont décrits tous les champs de la réponse d'un remboursement partiel ou total d'une transaction.

Nom du champ	Origine de la valeur
authorisation_id	cartes AURORE et PASS : renseigné par le serveur bancaire autres cartes : non renseigné
bank_response_code	cartes AURORE et PASS : renseigné par le serveur bancaire autres cartes : non renseigné
complementary_code	non renseigné
complementary_info	non renseigné
credit_amount	renseigné par le serveur Office
currency_code	identique à la requête
cvv_response_code	non renseigné
data	identique à la requête
new_amount	renseigné par le serveur Office
new_status	renseigné par le serveur Office
response_code	renseigné par le serveur Office
transaction_date	renseigné par le serveur Office
transaction_time	renseigné par le serveur Office
transaction_certificate	renseigné par le serveur Office (si opération acceptée)

4.3.6.3 Exemple de requête et de réponse XML

Ci-dessous est présenté un exemple de requête XML de remboursement.

```
<service component="office" name="credit">
<credit origin="Batch"
merchant_id="011223344553333"
merchant_country="fr"
transaction_id="27"
payment_date="20030801"
amount="2300"
currency_code="978"/>
</service>
```

Pour connaître la signification de ces différents champs référez-vous au *DICTIONNAIRE DES DONNEES.*

Ci-dessous est présentée la réponse XML du remboursement précédent.

```
<response component="office" name="credit">
<credit response_code="00"
transaction_time="150125"
transaction_date="20030801"
transaction_certificate="3d037d6d39cb"
authorisation_id=""
new_status="CAPTURED"
new_amount="10000"
credit_amount="2300"
currency_code="978"
data=""
cvv_response_code=""
bank_response_code=""
complementary_code=""
complementary_info="" />
</response>
```

Pour connaître la signification de ces différents champs référez-vous au *DICTIONNAIRE DES DONNEES.*

4.3.6.1 Règles de simulation du serveur de démo

Pas possible de tester le remboursement en démo car il n'y a pas de process Back office pour simuler la remise en banque.

4.3.7 Requête de forçage d'une transaction : advice

4.3.7.1 Objet

Cette fonction permet de forcer une transaction en « referral » (cf. le paragraphe **forçage** de la *PRESENTATION FONCTIONNELLE*).

4.3.7.2 Paramètres de la requête et de la réponse

Dans le tableau ci-dessous sont décrits tous les champs de la requête du forçage d'une transaction.

Nom du champ	Facultatif/ Obligatoire	Utilisation
authorisation_id	O	numéro d'autorisation
merchant_country	O	code pays du commerçant
merchant_id	O	identifiant du commerçant
origin	F	permet d'identifier le programme à l'origine du forçage
payment_date	O	date de création de la transaction à forcer
transaction_id	O	numéro de la transaction à forcer
transmission_date	O*	date de la requête
version	O*	version du composant

* renseigné par le composant

Dans le tableau ci-dessous sont décrits tous les champs de la réponse du forçage d'une transaction.

Nom du champ	Origine de la valeur
authorisation_id	non renseigné
bank_response_code	non renseigné
complementary_code	non renseigné
complementary_info	non renseigné
credit_amount	non renseigné
currency_code	renseigné par le serveur Office
cvv_response_code	non renseigné
data	identique à la requête
new_amount	renseigné par le serveur Office
new_status	renseigné par le serveur Office
response_code	renseigné par le serveur Office
transaction_date	renseigné par le serveur Office
transaction_time	renseigné par le serveur Office
transaction_certificate	renseigné par le serveur Office (si opération acceptée)

4.3.7.3 Exemple de requête et de réponse XML

Ci-dessous est présenté un exemple de requête XML de forçage.

```
<service component="office" name="advice">
<advice origin="Batch"
merchant_id="011223344553333"
merchant_country="fr"
transaction_id="28"
payment_date="20030801"
authorisation_id="111100"/>
</service>
```

Pour connaître la signification de ces différents champs référez-vous au *DICTIONNAIRE DES DONNEES.*

Ci-dessous est présentée la réponse XML du forçage précédent.

```
<response component="office" name="advice">
<advice response_code="00"
transaction_time="150750"
transaction_date="20030801"
transaction_certificate="3db6baf3f3ad"
authorisation_id=""
new_status="TO_CAPTURE"
new_amount="12300"
credit_amount=""
currency_code="978"
data=""
cvv_response_code=""
bank_response_code=""
complementary_code=""
complementary_info="" />
</response>
```

Pour connaître la signification de ces différents champs référez-vous au *DICTIONNAIRE DES DONNEES.*

4.3.7.4 Règles de simulation du serveur de démo

Pour forcer une transaction, il faut au préalable la créer avec un numéro de carte se terminant par 02 (réponse appel phonie)

4.3.8 Requête de duplication d'une transaction : duplicate

4.3.8.1 Objet

Cette fonction permet de créer une nouvelle transaction en se basant sur les coordonnées bancaires d'une transactionexistante.

4.3.8.2 Paramètres de la requête et de la réponse

Dans le tableau ci-dessous sont décrits tous les champs de la requête de la duplication d'une transaction.

Nom du champ	Facultatif/ Obligatoire	Utilisation
amount	O	montant de la transaction
capture_day	F	délai d'envoi en banque
capture_mode	F	mode d'envoi en banque
currency_code	O	code de la monnaie utilisée
data	F	paramétrage particulier
from_transaction_id	O	numéro de la transaction à dupliquer
from_payment_date	O	date de création de la transaction à dupliquer
merchant_country	O	code pays du commerçant
merchant_id	O	identifiant du commerçant
order_id	F	numéro de commande du commerçant
order_validity	F	non utilisé
origin	F	permet d'identifier le programme à l'origine de la duplication
transaction_id	O	numéro de la nouvelle transaction
return_context	F	contexte de la nouvelle transaction
transmission_date	O*	date de la requête
version	O*	version du composant

* renseigné par le composant

Dans le tableau ci-dessous sont décrits tous les champs de la réponse de la duplication d'une transaction.

nom du champ	origine de la valeur
authorisation_id	renseigné par le serveur bancaire (si transaction autorisée)
bank_response_code	renseigné par le serveur bancaire
complementary_code	renseigné par le serveur Office
complementary_info	renseigné par le serveur Office
credit_amount	non renseigné
currency_code	identique à la requête
cvv_response_code	renseigné par le serveur Office (si demande d'autorisation avec cryptogramme effectuée)
data	identique à la requête
new_amount	renseigné par le serveur Office
new_status	renseigné par le serveur Office
response_code	renseigné par le serveur Office
transaction_date	renseigné par le serveur Office
transaction_time	renseigné par le serveur Office
transaction_certificate	renseigné par le serveur Office (si opération acceptée)

4.3.8.3 Exemple de requête et de réponse XML

Ci-dessous est présenté un exemple de requête XML de duplication.

```
<service component="office" name="duplicate">
<duplicate origin="Batch"
merchant_id="011223344553333"
merchant_country="fr"
transaction_id="101"
amount="12300"
currency_code="978"
from_transaction_id="26"
from_payment_date="20030801"
return_context="context"
order_id="OI_131100_8744"
capture_mode=""
capture_day=""
data=""
order_validity=""/>
</service>
```

Pour connaître la signification de ces différents champs référez-vous au *DICTIONNAIRE DES DONNEES.*

Ci-dessous est présentée la réponse XML de duplication précédente.

```
<response component="office" name="duplicate">
<duplicate response_code="00"
transaction_time="153607"
transaction_date="20030801"
transaction_certificate="1059744967"
authorisation_id="1059"
new_status="TO_CAPTURE"
new_amount="12300"
credit_amount=""
currency_code="978"
data=""
cvv_response_code="4D"
bank_response_code="00"
complementary_code=""
complementary_info=""  />
</response>
```

Pour connaître la signification de ces différents champs référez-vous au *DICTIONNAIRE DES DONNEES.*

4.3.8.4 Règles de simulation du serveur de démo

Pas de règle de simulation, pour dupliquer une transaction il faut au prélable la créer.

4.3.9 Requête de credit porteur : creditholder

4.3.9.1 Objet

Cette fonction permet au commerçant disposant des coordonnées bancaires de ses clients d'effectuer un crédit sur leur compte, sans aucune transaction au préalable.

Warning sur l'usage de cette fonctionnalité

La mise à disposition de cette fonction est soumise à validation explicite de votre banque car elle implique que le commerçant dispose des coordonnées cartes de son client.

4.3.9.2 Paramètres de la requête et de la réponse

Dans le tableau ci-dessous sont décrits tous les champs de la requête de type crédit porteur.

Nom du champ	Facultatif/ Obligatoire	Utilisation
component	O	nom du composant appelé
name	O	nom de la fonction appelée
amount	O	montant de la transaction
card_number	O	numéro de la carte
card_type	O	type de carte utilisée
card_validity	O	date de validité de la carte
currency_code	O	code de la monnaie utilisée
merchant_country	O	code pays du commerçant
merchant_id	O	identifiant du commerçant
order_id	F	numéro de commande du commerçant
return_context	F	contexte de la transaction
transaction_id	O	numéro de la nouvelle transaction
transmission_date	O*	date de la requête
version	O*	version du composant

** renseigné par le composant*

Dans le tableau ci-dessous sont décrits tous les champs de la réponse de la demande de crédit porteur.

Nom du champ	Origine de la valeur
authorisation_id	renseigné par le serveur Office
avs_response_code	renseigné par le serveur Office
bank_response_code	renseigné par le serveur Office
complementary_code	renseigné par le serveur Office
complementary_info	renseigné par le serveur Office
credit_amount	renseigné par le serveur Office
currency_code	identique à la requête
cvv_response_code	renseigné par le serveur Office
data	renseigné par le serveur Office
new_amount	renseigné par le serveur Office
new_status	renseigné par le serveur Office
response_code	renseigné par le serveur Office
transaction_date	renseigné par le serveur Office
transaction_time	renseigné par le serveur Office
transaction_certificate	renseigné par le serveur Office (si transaction acceptée)

4.3.9.3 Exemple de requête et de réponse XML

Ci-dessous est présenté un exemple de requête XML de demande de crédit porteur.

```
<service component="office" name="creditholder">
<creditholder merchant_id="011223344553333"
merchant_country="fr"
transaction_id="26"
amount="12300"
currency_code="978"
card_number="498465123123123200"
card_validity="200411"
card_type="VISA"
return_context="context"
order_id="OI_131100_8744" />
</service>
```

Pour connaître la signification de ces différents champs référez-vous au *DICTIONNAIRE DES DONNEES.*

Ci-dessous est présentée la réponse XML de la demande de crédit porteur précédente.

```
<response component="office" name="creditholder">
<creditholder response_code="00"
transaction_time="114147"
transaction_date="20030801"
transaction_certificate="1059730907"
authorisation_id=""
new_status="TO_CREDIT"
new_amount="12300"
credit_amount=""
currency_code="978"
data=""
cvv_response_code=""
bank_response_code=""
complementary_code="00"
complementary_info="CARD_COUNTRY=FRA"  />
</response>
```

Pour connaître la signification de ces différents champs référez-vous au *DICTIONNAIRE DES DONNEES.*

4.3.9.4 Règles de simulation du serveur de démo

Le code réponse (**response_code**) de la transaction credit_holder est donné par les deux derniers chiffres du numéro de la carte bancaire.

Exemple :	numéro de carte	code réponse	
	4974934125497800	00	(acceptée)
	4972187615205	05	(refusée)

4.3.10 Absence de réponse et messages d'erreur

Si vous ne recevez pas de réponse, vous devez vérifier les points suivants :

- L'API Server est-il bien démarré ? (cf. chapitre **Lancement du serveur** du *GUIDE D'INSTALLATION*)
- N'y a-t-il pas un firewall entre le client et l'API Server qui pourrait empêcher l'accès au serveur ?
- Est-ce que l'adresse de la machine cliente est paramétrée dans les listes d'accès de l'API Server ? (cf. paragraphe **Configuration des listes d'accès** du GUIDE D'INSTALLATION)
- L'adresse IP de l'API Server que vous utilisez pour votre connexion socket est-elle correcte ?
- Le port de l'API Server que vous utilisez pour votre connexion socket est-il identique à celui configuré dans le fichier config.xml (cf. paragraphe **Configuration des paramètres de service** du GUIDE D'INSTALLATION)
- Avez-vous enregistré le composant Office (cf. paragraphe **La commande « ENREGISTRER UN NOUVEAU COMPOSANT »** du GUIDE D'ADMINISTRATION)

Si vous obtenez un message d'erreur tel que celui présenté ci-dessous :

```
<Error message=" message d'erreur "/>
```

Cela signifie que la requête a bien été reçue par l'API Server, mais elle comporte une erreur. Vous trouverez dans le tableau ci-dessous les principaux messages renvoyés par le composant ainsi que leur origine.

Exemple de messages	**Cause**	**solution**
(Le chemin d'accès spécifié est introuvable)	il y a une erreur au niveau du paramètre « id=″office″ » du sous-attribut pathfile dans le fichier *config.xml*	Corriger la configuration en vous référant au paragraphe 4.2.2.3.
Element "service" does not allow "toto" here.	toto n'est pas un nom d'opération correct dans la requête XML.	Corriger le nom de l'opération en vous référant aux exemples des paragraphes 4.3.2 à 4.3.9
Attribute "toto" is not declared for element "checkoffice".	Vous utilisez un nom d'attribut inconnu dans la requête XML.	Vérifier le nom des attributs en vous référant aux exemples des paragraphes 4.3.2 à 4.3.9
Root element type is "service", but was declared to be "command".	Vous utilisez le port de commande de l'API Server au lieu du port de service	Utiliser le port de service configuré dans le fichier *config.xml* pour votre connexion socket à l'API Server (cf. (cf. GUIDE D'ADMINISTRATION).
Autres messages		Contacter le Centre d'Assistance Technique

4.4 PARAMETRAGE DE LA MODALITE D'ENVOI EN BANQUE

L'envoi en banque d'une transaction, également appelé capture ou remise d'une transaction, peut être défini à l'aide de deux paramètres : **capture_mode** et **capture_day**.

Le champ **capture_mode** précise le mode d'envoi en banque, tandis que le champ **capture_day** indique le délai avant l'envoi en banque.

Pour connaître les règles de gestion exactes de la capture différée, veuillez vous référer au document de *PRESENTATION FONCTIONNELLE* et à l'Annexe W du dictionnaire des données

.

4.5 PROPOSITION POUR RESOUDRE VOS CONTRAINTES METIERS

Ce chapitre propose des marches à suivre pour faire face aux principaux problèmes fonctionnels que peuvent rencontrer les commerçants.

4.5.1 Débit à la livraison

Certains commerçants souhaitent débiter leurs clients à la livraison afin d'éviter le remboursement des produits non disponibles. Pour ce faire le commerçant doit, lors de la demande d'autorisation (cf. paragraphe 4.3.2), renseigner à VALIDATION le champ **capture_mode** et indiquer dans le champ **capture_day** un nombre de jours supérieur au délai maximum de livraison (cf. chapitre 0précédent).

Le jour de la livraison il ne restera plus qu'à valider (cf. paragraphe 4.3.4) la transaction avec le montant des marchandises réellement livrées.

4.5.2 Traitement des transactions expirées

Une transaction qui a expiré (statut EXPIRED cf. *PRESENTATION FONCTIONNELLE*) ne sera jamais remisée, le commerçant ne sera donc jamais crédité.

Si le commerçant a livré la marchandise et souhaite donc débiter son client, il peut dupliquer la transaction expirée (cf. paragraphe 4.3.8). Cette opération lui permet de créer une nouvelle transaction avec les coordonnées bancaires de la transaction précédente.

4.5.3 Problème de connexion avec la banque lors d'une demande d'autorisation

Lorsque le serveur Office d'Atos Origin ou les serveurs bancaires rencontrent un problème, cela conduit au renvoi d'un code 90 ou 99 dans le champ **response_code**. Le commerçant risque alors de refuser une vente alors qu'un essai ultérieur peut conduire à une acceptation de la transaction. Dans ce cas, il est préférable que le commerçant indique à son client qu'il sera contacté ultérieurement pour la suite de sa commande. Le commerçant peut ensuite tenter une nouvelle demande d'autorisation, soit en utilisant les coordonnées bancaires de son acheteur si il les a conservées, soit en dupliquant la transaction précédemment refusée (cf. paragraphe 4.3.8).

4.5.4 Time out suite à une opération

Si lors d'une opération sur une transaction ou lors de la création d'une transaction, la connexion vers le serveur Office d'Atos Origin est coupée pour cause de time out avant que la réponse ait été reçue par le composant Office, vous ne savez pas si votre opération ou votre transaction a été prise en compte. Pour savoir si elle a été prise en compte et quel en est le résultat, vous pouvez utiliser le composant Diag.

Ce composant permet de récupérer, pour une transaction particulière, les informations stockées dans la base de données d'Atos Origin. Pour plus d'information sur ce composant vous pouvez vous référer au *GUIDE DU COMPOSANT DIAG* ou contacter le Centre d'Assistance Technique si vous ne disposez pas de cette documentation.

4.6 DESCRIPTION DES TRACES

Suite à la réception d'une requête, le composant Office écrit dans les fichiers de traces configurés au paragraphe 4.2.2.4.

4.6.1 Si aucune erreur n'est survenue

Le composant inscrit dans les traces le nom de la fonction, la clé définissant la transaction créée ou modifiée (**merchant_id**, **merchant_country**, **transaction_id** et **payment_date**), le code réponse de l'opération et la réponse XML renvoyée au client de l'API Server.

Un exemple de traces pour une demande d'autorisation est présenté ci-dessous.

```
238:10:40:58-0-(5)-OfficeWrapper : fonction : author
238:10:40:58-0-(5)-OfficeWrapper : merchant_id=011223344553333, merchant_country=fr
238:10:41:01-0-(5)-OfficeWrapper : transaction_id=151515, payment_date=20030826
238:10:41:01-0-(5)-OfficeWrapper : response_code=00
238:10:41:01-0-(5)-OfficeWrapper : response : <response component="office" name="author"><author response_code="00" transaction_time="103658" transaction_date="20030826" transaction_certificate="1061887018" authorisation_id="1061" new_status="TO_VALIDATE" new_amount="12300" credit_amount="" currency_code="978" data="" cvv_response_code="4D" bank_response_code="00" complementary_code="" /></response>
238:10:41:01-0-(5)-OfficeWrapper : --------------------------------------------------------------------------------
```

Dans le cas de la fonction de vérification checkoffice, les champs **transaction_id** et **payment_date** contiennent respectivement l'heure et la date de la requête de vérification.

Un exemple de traces de vérification est présenté ci-dessous.

```
238:10:38:04-0-(1)-OfficeWrapper : fonction : checkoffice
238:10:38:04-0-(1)-OfficeWrapper : merchant_id=011223344553333, merchant_country=fr
238:10:38:04-0-(1)-OfficeWrapper : transaction_id=103804, payment_date=20030826
238:10:38:04-0-(1)-OfficeWrapper : response : <response  pathfile="OK"  certificate="OK" />
238:10:38:04-0-(1)-OfficeWrapper : --------------------------------------------------------------------------------
```

4.6.2 En cas d'erreur

Le composant inscrit dans les traces le nom de la fonction appelée suivi du message d'erreur complet.

Un exemple de traces en cas d'erreur lors d'une demande d'autorisation est présenté ci-dessous.

```
238:10:43:09-2-(8)-OfficeWrapper : fonction : author
238:10:43:09-2-(8)-OfficeWrapper : Error message: C:\sips_office\API\Server\office\param\pathfile (Le fichier spécifié est introuvable)
238:10:43:09-2-(8)-OfficeWrapper : --------------------------------------------------------------------------------
```

5. LE COMPOSANT DIAG 3.06

5.1 FONCTIONNEMENT GENERAL DU COMPOSANT DIAG

Le composant Diag permet de connaitre l'état des transactions précédemment créées via l'une des interface Sips (le composant Office , Sips Payment Web, Sips Subscription ou Sips Office Extranet….)

5.2 INSTALLATION ET PARAMETRAGE DU COMPOSANT

5.2.1 Installation

5.2.1.1 Liste des objets livrés

Le composant Diag est livré sous la forme d'un fichier composant_diag_306.tar contenant les fichiers suivants :

Fichier *Version.txt*	Fichier précisant l'environnement dans lequel le composant a été compilé et testé.
Répertoire *diag/param*	
certif.fr.011223344553333	Certificat de la boutique de démonstration
pathfile	Fichier des informations proxy et du chemin d'accès au certificat
Répertoire *server/components/service*	
diag.jar	archive du composant Diag

5.2.1.2 Copie des fichiers

Vous devez tout d'abord décompresser le fichier *composant_diag_306.tar*.

Pour un OS de type Unix :
Utiliser la commande suivante : tar –xvf *composant_diag_306.tar*

Pour Windows :
Utiliser un logiciel de décompression qui accepte les fichiers d'extension « .tar ».

Après avoir décompressé le fichier *composant_diag_306.tar*, vous devez copier le répertoire *diag* et tout ce qu'il contient dans le répertoire principal de l'API Sips Office Server. Le répertoire principal de l'API Sips Office Server contient également les répertoires associés aux autres composants que vous utilisez, ainsi que le répertoire *server*.
Vous devez également copier le fichier *diag.jar* dans le répertoire *server/components/service* du répertoire principal de l'API Sips Office Server.

5.2.2 Configuration

5.2.2.1 Les paramètres du proxy

Si vous utilisez un proxy pour vous connecter à Internet, vous devez le paramétrer dans le fichier *pathfile* copié précédemment dans le répertoire *diag/param*. Ce paramétrage permettra au composant Diag de contacter le serveur Office d'Atos Origin via l'Internet (cf. étape 3 de la Figure 1).

Si vous n'utilisez pas de proxy pour vous connecter à Internet, vous ne devez rien indiquer dans les champs suivants :

```
PROXY_HOST : adresse IP ou nom du proxy.
PROXY_PORT : le numéro de port du proxy.
```

En cas de doute, votre administrateur réseau pourra vous indiquer les valeurs à renseigner ici.

Exemple :
Si vous passez par le proxy d'adresse IP 111.11.11.11 et de port 7878 vous devez paramétrer les champs :

```
PROXY_HOST!111.11.11.11!
PROXY_PORT!7878!
```

5.2.2.2 Configuration du chemin vers les fichiers certificats

Le chemin vers les fichiers certificat est localisé dans le fichier *pathfile* copié précédemment dans le répertoire *diag/param*. Ce chemin doit être renseigné dans le champ F_CERTIFICATE et est unique pour tous les certificats du composant Diag. Tous les fichiers certificats que vous utiliserez avec le composant Diag devront donc être localisés dans un même répertoire.

Remarque
Les extensions du fichier certificat ne doivent en aucun cas être indiquées dans le champ F_CERTIFICATE. En effet, ces extensions correspondent aux paramètres **merchant_country** et **merchant_id** de la requête de vérification et de diagnostic. Ils seront donc ajoutés par le composant au paramètre F_CERTIFICATE pour obtenir le chemin complet du certificat souhaité.

Exemple Windows :
Si le fichier *certif.fr.011223344553333* est localisé dans le répertoire *C:\API_Server\diag\param*, vous devez renseigner le champ F_CERTIFICATE comme suit :

```
F_CERTIFICATE!C:\API_Server\diag\param\certif!
```

Exemple autres OS :
Si le fichier *certif.fr.011223344553333* est localisé dans le répertoire */home/API_Server/diag/param/*, vous devez renseigner le champ F_CERTIFICATE comme suit :

```
F_CERTIFICATE!/home/API_Server /diag/param/certif!
```

5.2.2.3 Configuration du chemin vers le fichier pathfile

Vous devez paramétrer dans le fichier *config.xml* le chemin vers le fichier *pathfile* du composant Diag. Le fichier *config.xml* est localisé dans le répertoire *server/config/* de l'API Server.

Vous trouverez ci-dessous le paragraphe concerné dans le fichier config.xml :

```
<components>
<pathfile id="example" path="chemin_du_fichier_pathfile" />
</components>
```

Vous devez ajouter la ligne suivante entre les balises <components> et </components>.

```
<pathfile id="diag" path="chemin_absolu_vers_le_fichier_pathfile" />
```

avec :
id : l'identifiant du composant (ici diag).
path : le chemin absolu vers le fichier pathfile.

Remarque
Le nom indiqué dans le champ « id » doit être identique au nom de l'archive sans l'extension « .jar » (cf. paragraphe 5.2.1.1)

Exemple Windows :
Si le fichier *pathfile* est localisé dans le répertoire *C:\API_Server\diag\param*, vous devez renseigner la ligne comme suit :

```
<pathfile id="diag" path="C:\API_Server\diag\param\pathfile" />
```

Exemple autres OS :
Si le fichier *pathfile* est localisé dans le répertoire */home/API_Server/diag/param*, vous devez renseigner la ligne comme suit :

```
<pathfile id="diag" path="/home/API_Server/diag/param/pathfile" />
```

5.2.2.4 Configuration des traces

Vous devez paramétrer dans le fichier *config.xml* le chemin des fichiers de traces du composant Diag. Le fichier *config.xml* est localisé dans le répertoire *server/config/* de l'API Server.

Vous trouverez ci-dessous le paragraphe concerné dans le fichier *config.xml* :

```
<watchdog>
        <pollingTimer>6000</pollingTimer>
        <survPort>7183</survPort>
        <trace level="0" sizeLimit="1000" unit="Line" path="C:\chemin_racine\server\trace" prefix="APIServer" />
<!--    <alternateTrace id="example" path="C:\chemin_racine\server\trace" prefix="example" />
        -->
        <alternateTrace id="" path="./" prefix="" />
</watchdog>
```

Vous devez ajouter la ligne suivante au-dessus du tag </watchdog> (la ligne avec l'identifiant example est en commentaire).

```
<alternateTrace id="diag" path="chemin_absolu_vers_le_répertoire_traces" prefix="diag" />
```

avec :
id : l'identifiant du composant (ici diag).
path : le chemin absolu vers le répertoire des traces. Vous pouvez utiliser pour vos traces le répertoire *server/traces* du répertoire principal de l'application ou tout autre répertoire de votre choix.
prefix : préfixe du fichier de traces du composant. Dans l'exemple ci-dessus, le nom du composant est utilisé, mais tout autre préfixe est également valable.

Remarque

- le nom indiqué dans le champ « id » doit être identique au nom de l'archive sans l'extension « .jar » (cf. paragraphe 5.2.1.1).
- La ligne « <alternateTrace id="" path="./" prefix="" />" est nécessaire au démarrage de l'API Server dans le cas ou aucun composant n'est installé, il est donc important de ne pas la modifier.

Exemple Windows :
Si le chemin du répertoire trace est *C:\API_Server\server\traces*, vous devez renseigner la ligne comme suit :

```
<alternateTrace id="diag" path="C:\API_Server\server\traces" prefix="diag" />
```

Exemple autres OS :
Si le chemin du répertoire trace est */home/API_Server/server/traces*, vous devez renseigner la ligne comme suit :

```
<alternateTrace id="diag" path="/home/API_Server/server/traces" prefix="diag" />
```

5.2.2.5 Adresse IP pour la configuration de votre firewall

Si vous utilisez un firewall pour vous connecter au réseau Internet, vous devez paramétrer au niveau de celui-ci l'adresse IP et le port du serveur Office d'Atos Origin. Pour communiquer avec le serveur Office d'Atos Origin votre firewall doit vous permettre une connexion socket en protocole TCP/IP vers l'adresse IP 193.56.46.110 port 2001.

5.3 DESCRIPTION DU COMPOSANT

5.3.1 Introduction

Ce chapitre décrit les opérations disponibles avec le composant Diag. Pour chacune de ces opérations, des exemples de requête et de réponse au format XML sont fournis, ainsi que les listes des paramètres de la requête et de la réponse avec des liens vers le dictionnaire des données.

Avant de passer en revue les opérations disponibles, nous allons préciser les aspects de la sécurisation du transfert des informations entre le composant Diag et le serveur Office d'Atos Origin.

Remarque

- Afin d'alléger le code, les entêtes standard XML ne sont pas nécessaires, mais sont ajoutées par l'API Server pour vérification lors du parsing XML (utilisation d'un DTD). Par contre, lors des réponses de l'API Server, vers l'application cliente, aucune référence à un DTD n'est incluse dans le message. Il est donc important de désactiver la validation lors du parsing, afin de ne pas déclencher des erreurs durant cette phase.
- Pour des raisons de lisibilité, toutes les requêtes décrites dans ce chapitre sont présentées sur plusieurs lignes. Toutefois, lors de l'appel à l'API Server, elles doivent être codées sans retours chariot.

5.3.2 Requête de vérification vers le composant Diag : checkdiag

5.3.2.1 Objet

Cette fonction permet de tester :

- la communication entre le client que vous développez et l'API Server
- le bon fonctionnement de l'API Server
- la configuration du composant Diag
  - l'accès au fichier pathfile
  - l'accès au fichier certificat

5.3.2.2 Paramètres de la requête et de la réponse

Dans le tableau ci-dessous sont décrits tous les champs de la requête de vérification.

nom du champ	Facultatif/ Obligatoire	Utilisation
component	O	nom du composant appelé
name	O	nom de la fonction appelée
merchant_country	O	code pays du commerçant
merchant_id	O	identifiant du commerçant

Dans le tableau ci-dessous sont décrits tous les champs de la réponse de la vérification.

nom du champ	valeur
pathfile	renseigné à OK si le fichier pathfile a bien été trouvé
certificate	renseigné à OK si le fichier certif.<merchant_country>.<merchant_id> a bien été trouvé
message	renseigné si le fichier pathfile ou certificat n'a pu être lu

5.3.2.3 Exemple de requête et de réponse XML

Ci-dessous, vous trouverez un exemple de requête de vérification. La signification des champs **merchant_country** et **merchant_id** est fournie au *DICTIONNAIRE DES DONNEES.* Ces champs correspondent respectivement à la première et la deuxième extension du fichier certificat (exemple : le fichier *certif.fr.011223344553333* correspond à **merchant_country**=fr et **merchant_id**=011223344553333).

```
<service component="diag" name="checkdiag">
<checkdiag merchant_id="011223344553333" merchant_country="fr"/>
</service>
```

Lors de la vérification, le composant Diag va contrôler l'accès en lecture au fichier *pathfile* paramétré dans le fichier *config.xml*, puis l'accès en lecture au fichier certificat (*certif.<merchant_country>.<merchant_id>*) dont le chemin générique est indiqué dans le fichier *pathfile*. Si ces accès sont corrects, la réponse suivante sera envoyée :

```
<response pathfile="OK" certificate="OK" />
```

Si ces accès ne sont pas corrects, la réponse suivante sera envoyée :

```
<Error message=" message d'erreur "/>
```

La liste des messages d'erreur est précisée dans le tableau ci-dessous.

Messages d'erreur	Cause	solution
C:\API_Server\diag\pam\pathfile (Le chemin d'accès spécifié est introuvable)	le fichier *pathfile* configuré dans le fichier *config.xml* n'est pas accessible en lecture	Vérifier les droits d'accès du fichier *pathfile* et corriger le chemin du fichier *pathfile* si nécessaire (cf. paragraphe 5.2.2.3).
Cannot open certificat. (C:\API_Server\diag\pam\certif.fr. 011223344553333)	le fichier certif.fr.011223344553333 n'est pas accessible en lecture	Vérifier les droits d'accès du fichier certificat *certif.fr.011223344553333* et corriger le chemin génériques des fichiers certificats si nécessaire (cf. paragraphe 5.2.2.2).

5.3.3 Requête de diagnostic d'une transaction : diagnosis

5.3.3.1 Objet

Cette fonction permet de récupérer, pour une transaction particulière, des informations stockées dans la base de données d'Atos Origin.

5.3.3.2 Paramètres de la requête et de la réponse

Dans le tableau ci-dessous sont décrits tous les champs de la requête de diagnostic.

nom du champ	Facultatif/ Obligatoire	Utilisation
component	O	nom du composant appelé
name	O	nom de la fonction appelée
merchant_country	O	pays du commerçant qui a créé la transaction à diagnostiquer
merchant_id	O	identifiant du commerçant qui a créé la transaction à diagnostiquer
transaction_id	O	numéro de la transaction à diagnostiquer
payment_date	O	date de création de la transaction à diagnostiquer
transmission_date	O*	date de la requête
version	O*	version du composant

** renseigné par le composant*

Dans le tableau ci-dessous sont décrits tous les champs de la réponse de demande d'autorisation.

nom du champ	origine de la valeur
automatic_response_status	base de données Atos Origin
authorisation_id	base de données Atos Origin
capture_date	base de données Atos Origin
card_type	base de données Atos Origin
currency_code	base de données Atos Origin
current_amount	base de données Atos Origin
customer_ip_address	base de données Atos Origin
diagnostic_code	le serveur Office Atos Origin
diagnostic_date	le serveur Office Atos Origin
diagnostic_time	le serveur Office Atos Origin
diagnostic_certificate	le serveur Office Atos Origin si le diagnostic a été réalisé
last_operation_date	base de données Atos Origin
last_operation_code	base de données Atos Origin
merchant_country	identique à la requête
merchant_id	identique à la requête
order_channel	canal de paiement
order_id	base de données Atos Origin
origin_amount	base de données Atos Origin
payment_certificate	base de données Atos Origin
payment_date	identique à la requête
payment_means	base de données Atos Origin
payment_pattern	type de paiement
private_diag_data	base de données Atos Origin
response_code	base de données Atos Origin
transaction_id	identique à la requête
transaction_status	base de données Atos Origin

5.3.3.3 Exemple de requête et de réponse XML

Ci-dessous est présenté un exemple de requête XML de diagnostic.

```
<service component="diag" name="diagnosis">
<diagnosti merchant_id="011223344553333"
merchant_country="fr"
transaction_id="22"
payment_date="20030901"/>
</service>
```

Pour connaître la signification de ces différents champs référez-vous au *DICTIONNAIRE DES DONNEES.*

Ci-dessous est présenté un exemple de réponse XML de diagnostic.

```
<response component="diag" name="diagnosis">
<diagnosis diagnostic_code="00"
diagnostic_time="120917"
diagnostic_date="20030901"
diagnostic_certificate="485a89b17029"
merchant_id="011223344553333"
merchant_country="fr"
transaction_id="22"
payment_date="20030901"
order_id="OI_131100_8744"
origin_amount="12300"
current_amount="12300"
currency_code="978"
transaction_status="TO_VALIDATE"
capture_date="20030903"
response_code="00"
authorisation_id="1062"
payment_certificate="1062410937"
payment_means="CARD"
card_type="VISA"
customer_ip_address=""
automatic_response_status="UNDEFINED"
last_operation_date="20030901120859"
last_operation_code=""
private_diag_data="" />
</response>
```

Pour connaître la signification de ces différents champs référez-vous au *DICTIONNAIRE DES DONNEES.*

5.3.3.4 Règles de simulation su serveur de démo

Pas de règle de simulation, pour diagostiquer une transaction transaction, il faut la créer au préalable.

Absence de réponse et messages d'erreur

Si vous ne recevez pas de réponse, vous devez vérifier les points suivants :

- L'API Server est-elle bien démarré ? (cf. chapitre **Lancement du serveur** *GUIDE D'ADMINISTRATION*)
- N'y a-t-il pas un firewall entre le client et l'API Server qui pourrait empêcher l'accès au serveur ?
- Est-ce que l'adresse de la machine cliente est paramétrée dans les listes d'accès de l'API Server ? (cf. paragraphe **Configuration des listes d'accès** du GUIDE D'INSTALLATION)
- L'adresse IP de l'API Server que vous utilisez pour votre connexion socket est-elle correcte ?
- Le port de l'API Server que vous utilisez pour votre connexion socket est-il identique à celui configuré dans le fichier config.xml (cf. paragraphe **Configuration des paramètres de service** du GUIDE D'INSTALLATION)
- Avez-vous enregistré le composant Diag (cf. paragraphe **La commande « ENREGISTRER UN NOUVEAU COMPOSANT »** du GUIDE D'ADMINISTRATION)

Si vous obtenez un message d'erreur tel que celui présenté ci-dessous :

<Error message="*message d'erreur*"/>

cela signifie que la requête a bien été reçue par l'API Server, mais elle comporte une erreur. Vous trouverez dans le tableau ci-dessous les principaux messages renvoyés par le composant ainsi que leur origine.

Exemple de messages	Cause	solution
(Le chemin d'accès spécifié est introuvable)	il y a une erreur au niveau du paramètre « id=″diag″ » du sous-attribut *pathfile* dans le fichier *config.xml*	Corriger la configuration en vous référant au paragraphe 5.2.2.3.
Element "service" does not allow "toto" here.	toto n'est pas un nom d'opération correct dans la requête XML.	Corriger le nom de l'opération en vous référent aux exemples du paragraphe 5.3.3.3
Attribute "toto" is not declared for element "checkDiag".	Vous utilisez un nom d'attribut inconnu dans la requête XML.	Vérifier le nom des attributs en vous référant aux exemples du paragraphe 5.3.3.
Root element type is "service", but was declared to be "command".	Vous utilisez le port de commande de l'API Server au lieu du port de service	Utiliser le port de service configuré dans le fichier *config.xml* pour votre connexion socket à l'API Server (cf. GUIDE D'ADMINISTRATION).
Autres messages		Contacter le Centre d'Assistance Technique

5.4 DESCRIPTION DES TRACES

Suite à la réception d'une requête, le composant Diag écrit dans les fichiers de traces configurés au paragraphe 5.2.2.4.

5.4.1 Si aucune erreur n'est survenue

Le composant inscrit dans les traces le nom de la fonction, la clé définissant la transaction diagnostiquée (**merchant_id**, **merchant_country**, **transaction_id** et **payment_date**), le code réponse du diagnostic et la réponse XML renvoyée au client de l'API Server.

Un exemple de traces de diagnostic est présenté ci-dessous.

```
245:11:15:25-0-(2)-DiagWrapper : fonction : diagnostic
245:11:15:25-0-(2)-DiagWrapper : merchant_id=011223344553333, merchant_country=fr
245:11:15:25-0-(2)-DiagWrapper : transaction_id=22, payment_date=20030901
245:11:15:30-0-(2)-DiagWrapper : response_code=00
245:11:15:30-0-(2)-DiagWrapper : response : <response component="diag"
name="diagnostic"><diagnostic diagnostic_code="00" diagnostic_time="111123"
diagnostic_date="20030902" diagnostic_certificate="9c82905c3813"
merchant_id="011223344553333" merchant_country="fr" transaction_id="22"
payment_date="20030901" order_id="OI_131100_8744" origin_amount="12300"
current_amount="12300" currency_code="978" transaction_status="TO_VALIDATE"
capture_date="20030903" response_code="00" authorisation_id="1062"
payment_certificate="1062410937" payment_means="CARD" card_type="VISA"
customer_ip_address="" automatic_response_status="UNDEFINED"
last_operation_date="20030901120859" last_operation_code="" private_diag_data="" /></response>
245:11:15:30-0-(2)-DiagWrapper : --------------------------------------------------------------------------------
```

Dans le cas de la fonction de vérification checkdiag, les champs **transaction_id** et **payment_date** contiennent respectivement l'heure et la date de la requête de vérification.

Un exemple de traces de vérification est présenté ci-dessous.

```
245:11:13:45-0-(1)-DiagWrapper : fonction : checkdiag
245:11:13:45-0-(1)-DiagWrapper : merchant_id=011223344553333, merchant_country=fr
245:11:13:45-0-(1)-DiagWrapper : transaction_id=111345, payment_date=20030902
245:11:13:45-0-(1)-DiagWrapper : response : <response  pathfile="OK"  certificate="OK" />
245:11:13:45-0-(1)-DiagWrapper : --------------------------------------------------------------------------------
```

5.4.2 En cas d'erreur

Le composant inscrit dans les traces le nom de la fonction appelée suivi du message d'erreur complet.

Un exemple de traces en cas d'erreur est présenté ci-dessous.

```
245:11:16:58-2-(3)-DiagWrapper : fonction : diagnostic
245:11:16:58-2-(3)-DiagWrapper : Error message: C:\sips_office\API\Server\diag\param\pathfile (Le fichier spécifié est introuvable)
245:11:16:58-2-(3)-DiagWrapper : --------------------------------------------------------------------------------
```

6. LE COMPOSANT CHECKOUT 1.02

6.1 FONCTIONNEMENT GENERAL DU COMPOSANT CHECKOUT

3-D Secure est un protocole de paiement sécurisé sur Internet. Il a été développé par Visa pour augmenter le niveau de sécurité des transactions, et il a été adopté par Mastercard.

Le concept de base de ce protocole est de lier le processus d'autorisation bancaire avec une authentification en ligne. Cette authentification est basée sur un modèle comportant 3 domaines (d'où le nom *3D*) qui sont:

- Le commerçant
- La banque
- Le système de carte bancaire

Lors de son authentification en ligne, l'internaute doit saisir une information personnelle sur l'ACS de sa banque (serveur de contrôle d'authentification piloté par la banque du porteur).

Dans le processus de la norme 3D-Secure, une transaction se déroule en trois étapes :

- Vérification d'enrôlement afin de savoir si la carte est enrôlée 3D-Secure ou non
- Authentification du porteur sur l'ACS de sa banque seulement dans le cas où il est enrôlé 3D-Secure
- Demande d'autorisation 3D-Secure

L'enchaînement de ces trois étapes forme un dispositif visant à réduire le risque d'impayé émis pour contestation du porteur.

Ce dispositif appelé **"liability shift"** ou **"transfert de responsabilité"** a pour principe de faire supporter le risque d'impayé émis pour contestation du porteur à la banque de celui-ci et non plus au commerçant. Si le porteur a validé son paiement en renseignant les données 3D Secure et que le commerçant a respecté les mesures de sécurité énoncées dans les conditions générales de vente de son contrat de commerce électronique, le commerçant ne subira pas la conséquence des impayés.

Le composant Checkout permet de créer des transactions 3D ou simplement contrôler si le porteur s'est bien authentifié 3D sur l'ACS de son acquéreur.

Pour plus d'informations sur la création des transactions, référez-vous à la *PRESENTATION FONCTIONNELLE.*

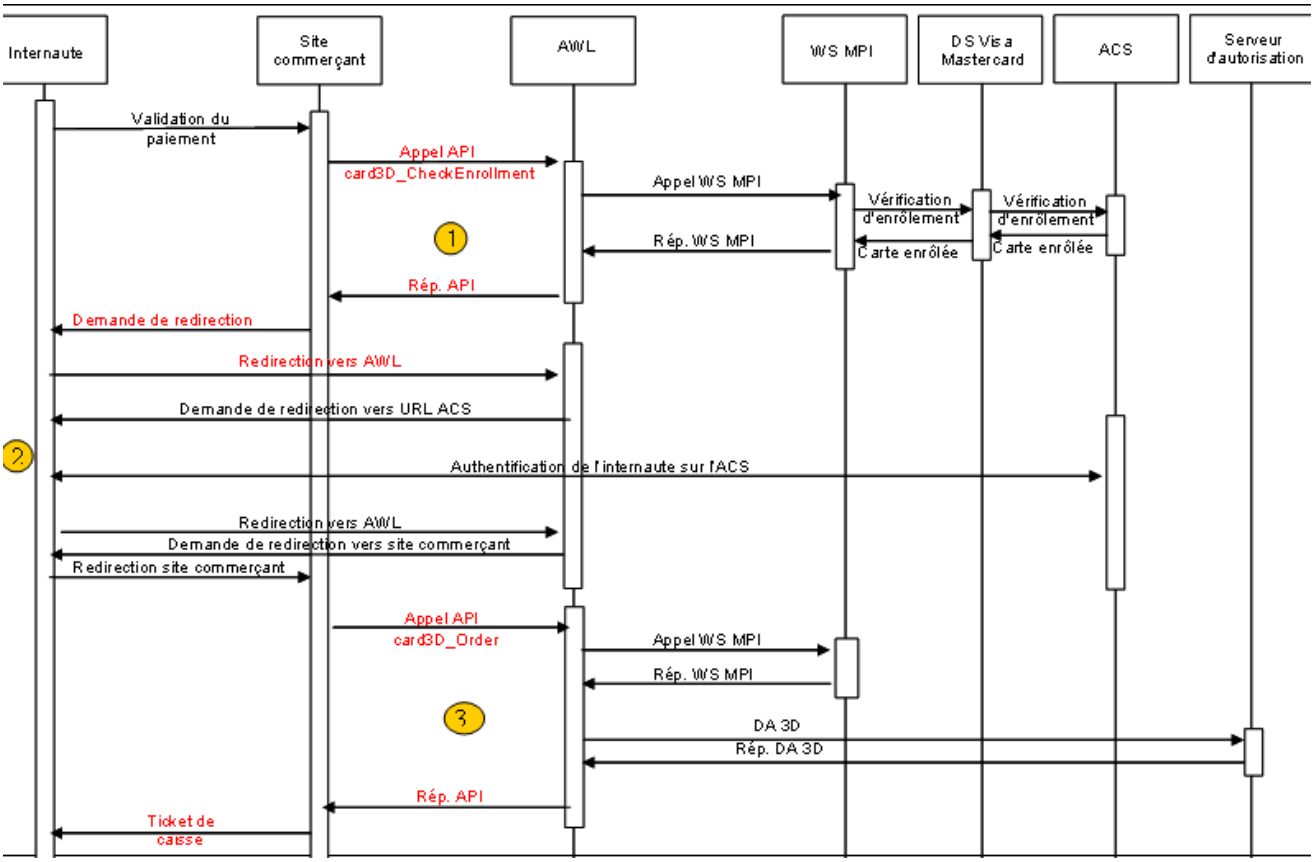

Figure 2: Shéma de la cinématique d'une transaction 3D-Secure (avec demande d'autorisation)

La création d'une transaction 3D peut se résumer en 3 étapes représentées sur la Figure 2 :

1/ **VERIFICATION D'ENROLEMENT** : Après validation du paiement par l'internaute, le site commerçant envoie une requête card3D_CheckEnrollment qui va permettre de vérifier l'enrôlement 3D du porteur. Lors de cette requête, AWL contacte le Web Service MPI (Merchant Plug-In). Ce dernier consultera les DS Visa/Mastercard (Directory Server) et l'ACS (Access Control Server) afin d'obtenir cette information.

2/ **AUTHENTIFICATION DU PORTEUR SUR L'ACS** : En retour de la requête card3D_CheckEnrollment, le site commerçant redirigera le flux vers AWL à l'aide d'une URL obtenue en réponse de la requête. Cela va provoquer la redirection de l'internaute sur son ACS afin qu'il puisse s'authentifier.

3/ **DEMANDE D'AUTORISATION 3D** : Après la phase d'authentification, le site commerçant envoie une requête card3D_Order qui va permettre de vérifier si l'internaute s'est bien authentifié sur son ACS (par le biais d'un appel au Web Service MPI) puis d'effectuer une demande d'autorisation 3D.

Ou

3'/ **VERIFICATION D'AUTHENTIFICATION 3D** : Après la phase d'authentification, le site commerçant envoie une requête card3D_Authenticate qui va SEULEMENT permettre de vérifier si l'internaute s'est bien authentifié sur son ACS. Il est à noter qu'aucune demande d'autorisation ne sera effectuée avec cette requête.

6.2 INSTALLATION ET PARAMETRAGE DU COMPOSANT

6.2.1 Installation

6.2.1.1 Liste des objets livrés

Le composant Checkout est livré sous la forme d'un fichier *composant_checkout_102.tar* contenant les fichiers suivants :

Fichier *Version.txt*	Fichier précisant l'environnement dans lequel le composant a été compilé et testé.
Répertoire checkout/*param*	
certif.fr.011223344553334	Certificat de la boutique de démonstration
pathfile	Fichier des informations proxy et du chemin d'accès au certificat
Répertoire *server/components/service*	
checkout*.jar*	archive du composant Checkout

6.2.1.2 Copie des fichiers

Vous devez tout d'abord décompresser le fichier *composant_checkout_102.tar*.

Pour un OS de type Unix :

Utiliser la commande suivante : tar –xvf *composant_checkout_102.tar*

Pour Windows :

Utiliser un logiciel de décompression qui accepte les fichiers d'extension « .tar ».

Après avoir décompressé le fichier *composant_checkout_102.tar*, vous devez copier le répertoire *checkout* et tout ce qu'il contient dans le répertoire principal de l'API Server. Le répertoire principal de l'API Server contient également les répertoires associés aux autres composants que vous utilisez, ainsi que le répertoire *server*.

Vous devez également copier le fichier *checkout.jar* dans le répertoire *server/components/service* du répertoire principal de l'API Server.

6.2.2 Configuration

6.2.2.1 Les paramètres du proxy

Si vous utilisez un proxy pour vous connecter à Internet, vous devez le paramétrer dans le fichier *pathfile* copié précédemment dans le répertoire *checkout/param*. Ce paramétrage permettra au composant Checkout de contacter le serveur Office d'Atos Origin via l'Internet (cf. étape 3 de la Figure 1).

Si vous n'utilisez pas de proxy pour vous connecter à Internet, vous ne devez rien indiquer dans les champs suivants :

PROXY_HOST : adresse IP ou nom du proxy.

PROXY_PORT : le numéro de port du proxy.

En cas de doute, votre administrateur réseau pourra vous indiquer les valeurs à renseigner.

Exemple :
Si vous passez par le proxy d'adresse IP 111.11.11.11 et de port 7878 vous devez paramétrer les champs :

```
PROXY_HOST!111.11.11.11!
PROXY_PORT!7878!
```

6.2.2.2 Configuration du chemin vers les fichiers certificats

Le chemin vers les fichiers certificat est localisé dans le fichier *pathfile* copié précédemment dans le répertoire *checkout/param*. Ce chemin doit être renseigné dans le champ F_CERTIFICATE et est unique pour tous les certificats du composant Checkout. Tous les fichiers certificats que vous utiliserez avec le composant Checkout devront donc être localisés dans un même répertoire.

Remarque

Les extensions du fichier certificat ne doivent en aucun cas être indiquées dans le champ F_CERTIFICATE. En effet, ces extensions correspondent aux paramètres **merchant_country** et **merchant_id** transmis dans les requêtes associées aux différentes opérations. Ils seront donc ajoutés par le composant au paramètre F_CERTIFICATE pour obtenir le chemin complet du certificat souhaité.

Exemple Windows :

Si le fichier *certif.fr.011223344553334* est localisé dans le répertoire *C:\API_Server\checkout\param*, vous devez renseigner le champ F_CERTIFICATE comme suit :

```
F_CERTIFICATE!C:\API_Server\checkout\param\certif!
```

Exemple autres OS :

Si le fichier *certif.fr.011223344553334* est localisé dans le répertoire */home/API_Server/checkout/param/*, vous devez renseigner le champ F_CERTIFICATE comme suit :

```
F_CERTIFICATE!/home/API_Server/checkout/param/certif!
```

6.2.2.3 Configuration du chemin vers le fichier pathfile

Vous devez paramétrer dans le fichier *config.xml* le chemin vers le fichier *pathfile* du composant Checkout. Le fichier *config.xml* est localisé dans le répertoire *server/config/* de l'API Server.

Vous trouverez ci-dessous le paragraphe concerné dans le fichier *config.xml* :

```
<components>
<pathfile id="example" path="chemin_du_fichier_pathfile" />
</components>
```

Vous devez ajouter la ligne suivante entre les balises <components> et </components> :

```
<pathfile id="checkout" path="chemin_absolu_vers_le_fichier_pathfile" />
```

avec :

id : l'identifiant du composant (ici checkout).

path : le chemin absolu vers le fichier *pathfile*.

Remarque

Le nom indiqué dans le champ « id » doit être identique au nom de l'archive sans l'extension « .jar » (cf. paragraphe 6.2.1.1)

Exemple Windows :

Si le fichier *pathfile* est localisé dans le répertoire *C:\API_Server\checkout\param*, vous devez renseigner la ligne comme suit :

```
<pathfile id="checkout" path="C:\API_Server\checkout\param\pathfile" />
```

Exemple autres OS :

Si le fichier *pathfile* est localisé dans le répertoire */home/API_Server/checkout/param*, vous devez renseigner la ligne comme suit :

```
<pathfile id="checkout" path="/home/API_Server/checkout/param/pathfile" />
```

6.2.2.4 Configuration des traces

Vous devez paramétrer dans le fichier *config.xml* le chemin des fichiers de traces du composant Checkout. Le fichier *config.xml* est localisé dans le répertoire *server/config/* de l'API Server.

Vous trouverez ci-dessous le paragraphe concerné dans le fichier *config.xml* :

```
<watchdog>
        <pollingTimer>6000</pollingTimer>
        <survPort>7183</survPort>
        <trace level="0" sizeLimit="1000" unit="Line" path="C:\chemin_racine\server\trace"
prefix="APIServer" />
<!--    <alternateTrace id="example" path="C:\chemin_racine\server\trace" prefix="example" />
        -->
        <alternateTrace id="" path="./" prefix="" />
</watchdog>
```

Vous devez ajouter la ligne suivante au-dessus du tag « </watchdog> » (la ligne avec l'identifiant example est en commentaire).

```
<alternateTrace id="checkout" path="chemin_absolu_vers_le_répertoire_traces" prefix="checkout" />
```

avec :

id : l'identifiant du composant (ici checkout).

path : le chemin absolu vers le répertoire des traces. Vous pouvez utiliser pour vos traces le répertoire *server/traces* du répertoire principal de l'application ou tout autre répertoire de votre choix.

prefix : préfixe du fichier de traces du composant. Dans l'exemple ci-dessus, le nom du composant est utilisé, mais tout autre préfixe est également valable.

Remarque

- le nom indiqué dans le champ « id » doit être identique au nom de l'archive sans l'extension « .jar » (cf. paragraphe 6.2.1.1).
- La ligne « <alternateTrace id="" path="./" prefix="" />" est nécessaire au démarrage de l'API Server dans le cas ou aucun composant n'est installé, il est donc important de ne pas la modifier.

Exemple Windows :

Si le chemin du répertoire trace est *C:\API_Server\server\traces*, vous devez renseigner la ligne comme suit :

```
<alternateTrace id="checkout" path="C:\API_Server\server\traces" prefix="checkout" />
```

Exemple autres OS :

Si le chemin du répertoire trace est */home/API_Server/server/traces*, vous devez renseigner la ligne comme suit :

```
<alternateTrace id="checkout" path="/home/API_Server/server/traces" prefix="checkout" />
```

6.2.2.5 Adresse IP pour la configuration de votre firewall

Si vous utilisez un firewall pour vous connecter au réseau Internet, vous devez paramétrer au niveau de celui-ci l'adresse IP et le port du serveur Office d'Atos Origin. Pour communiquer avec le serveur Office d'Atos Origin votre firewall doit vous permettre une connexion socket en protocole TCP/IP vers l'adresse IP 193.56.46.110 port 2001.

6.3 DESCRIPTION DETAILLEE DU COMPOSANT

Ce chapitre décrit les opérations disponibles avec le composant Checkout. Pour chacune de ces opérations, des exemples de requête et de réponse au format XML sont fournis, ainsi que les listes des paramètres de la requête et de la réponse avec des liens vers le dictionnaire des données.

Avant de décrire les fonctions disponibles, nous allons préciser les aspects de la sécurisation du transfert des informations entre le composant Checkout et le serveur Office d'Atos Origin.

Remarques

- Afin d'alléger le code des requêtes, les entêtes standard XML ne sont pas nécessaires, mais sont ajoutées par l'API Server pour vérification lors du parsing XML (utilisation d'un DTD). Par contre, lors des réponses de l'API Server, vers l'application cliente, aucune référence à un DTD n'est incluse dans le message. Il est donc important de désactiver la validation lors du parsing, afin de ne pas déclencher des erreurs durant cette phase.
- Pour des raisons de lisibilité, toutes les requêtes décrites dans ce chapitre sont présentées sur plusieurs lignes. Toutefois, lors de l'appel à l'API Server, elles doivent être codées sans retours chariot.

6.3.1 Sécurisation du transfert des informations

Voir chapitre **La sécurité des paiements en ligne avec SIPS Office Server** de la *PRESENTATION FONCTIONNELLE*

6.3.2 Requête de vérification vers le composant Checkout: checkCheckout

6.3.2.1 Objet

Cette fonction permet de tester :

- la communication entre l'application cliente que vous développez et l'API Server
- le bon fonctionnement de l'API Server
- la configuration du composant Checkout
  - l'accès au fichier *pathfile*
  - l'accès au fichier certificat

6.3.2.2 Paramètres de la requête et de la réponse

Dans le tableau ci-dessous sont décrits tous les champs de la requête de vérification.

nom du champ	Facultatif/ Obligatoire	Utilisation
component	O	nom du composant appelé
name	O	nom de la fonction appelée
merchant_country	O	code pays du commerçant
merchant_id	O	identifiant du commerçant

Dans le tableau ci-dessous sont décrits tous les champs de la réponse de la vérification.

nom du champ	valeur
pathfile	renseigné à OK si le fichier *pathfile* a bien été trouvé
certificate	renseigné à OK si le fichier *certif.<merchant_country>.<merchant_id>* a bien été trouvé
message	renseigné si le fichier *pathfile* ou *certificat* n'a pu être lu

6.3.2.3 Exemple de requête et de réponse XML

Ci-dessous est présenté un exemple de requête XML de vérification. La signification des champs **merchant_country** et **merchant_id** est fournie dans le *DICTIONNAIRE DES DONNEES*. Ces champs correspondent respectivement à la première et la deuxième extension du fichier certificat (exemple : le fichier *certif.fr.011223344553334* correspond à **merchant_country**=fr et **merchant_id**=011223344553334).

```
<service component="checkout" name="checkCheckout">
<checkCheckout merchant_id="011223344553334" merchant_country="fr"/>
</service>
```

Lors de la vérification, le composant Checkout va contrôler l'accès en lecture au fichier *pathfile* paramétré dans le fichier *config.xml*, puis l'accès en lecture au fichier certificat (*certif.<merchant_country>.<merchant_id>*) dont le chemin générique est indiqué dans le fichier *pathfile*. Si ces accès sont corrects, la réponse suivante sera envoyée :

```
<response pathfile="OK" certificate="OK" />
```

Si ces accès ne sont pas corrects, la réponse suivante sera envoyée :

```
<Error message="message d'erreur"/>
```

La liste des messages d'erreur est précisée dans le tableau ci-dessous.

Messages d'erreur	Cause	solution
C:\API_Server\checkout\pam\pathfile (Le chemin d'accès spécifié est introuvable)	le fichier *pathfile* configuré dans le fichier *config.xml* n'est pas accessible en lecture	Vérifier les droits d'accès du fichier *pathfile* et corriger le chemin du fichier *pathfile* si nécessaire (cf. paragraphe 6.2.2.3).
Cannot open certificat. (C:\API_Server\checkout\pam\certif.fr.011223344553334)	le fichier certif.fr.011223344553334 n'est pas accessible en lecture	Vérifier les droits d'accès du fichier certificat *certif.fr.011223344553334* et corriger le chemin génériques des fichiers certificats si nécessaire (cf. paragraphe 6.2.2.2).

6.3.3 Requête de vérification d'enrôlement 3D : card3D_CheckEnrollment

6.3.3.1 Objet

Cette fonction permet au commerçant de vérifier l'enrôlement 3D d'un porteur.

6.3.3.2 Paramètres de la requête et de la réponse

Dans le tableau ci-dessous sont décrits tous les champs de la requête de la vérification d'enrôlement 3D.

nom du champ	Facultatif/ Obligatoire	Utilisation
component	O	nom du composant appelé
name	O	nom de la fonction appelée
amount	O	montant de la transaction
card_number	O	numéro de la carte
card_type	O	type de carte utilisée
card_validity	O	date de validité de la carte
cvv_key	F	cryptogramme visuel
cvv_flag	F	indique la présence ou non du cryptogramme visuel
currency_code	O	code de la monnaie utilisée
merchant_country	O	code pays du commerçant
merchant_id	O	identifiant du commerçant
origin	F	permet d'identifier le programme à l'origine de la demande d'autorisation
transaction_id	O	numéro de la nouvelle transaction
merchant_name	O	nom du commerçant affiché sur l'ACS
merchant_url	O	URL du commerçant affichée sur l'ACS
merchant_url_return	O	URL de retour vers le site commerçant après l'authentification du porteur sur son ACS
transmission_date	O*	date de la requête
version	O*	version du composant

** renseigné par le composant*

Dans le tableau ci-dessous sont décrits tous les champs de la réponse de la vérification d'enrôlement.

nom du champ	origine de la valeur	description
o3d_response_code	renseigné par le serveur 3D Office	Code réponse du serveur 3D Office
o3d_office_url_acs	renseigné par le serveur 3D Office	URL de redirection vers Atos
o3d_session_id	renseigné par le serveur 3D Office	Identifiant de session de la transaction 3D

6.3.3.3 Exemple de requête et de réponse XML

Ci-dessous est présenté un exemple de requête XML de vérification d'enrôlement 3D.

```
<service component="checkout" name="card3D_CheckEnrollment">
<card3D_CheckEnrollment origin="Batch"
merchant_id="011223344553334"
merchant_country="fr"
transaction_id="130685"
amount="12300"
currency_code="978"
card_number="4970651231011232"
card_validity="201211"
```

```
card_type="VISA"
merchant_name=”Commercant_test”
merchant_url=”http://www.commercant_test.com"
merchant_url_return=”http://www.retourACS.com"
cvv_key=”600”
cvv_flag=”1”
/></service>
```

Pour connaître la signification de ces différents champs référez-vous au *DICTIONNAIRE DES DONNEE.*

Ci-dessous est présentée la réponse XML de la vérification d’enrôlement 3D précédente.

```
<response component="checkout" name=" card3D_CheckEnrollment">
<card3D_CheckEnrollment
o3d_response_code="00"
o3d_office_url_acs= “https://vre16:29567/3doffice/prod/call_acs;jsessionid=ERF56GD..."
o3d_session_id="3D8B9C023BBDD10EA45294AEC9C2C922.jvm1" />
</response>
```

Pour connaître la signification de ces différents champs référez-vous au *DICTIONNAIRE DES DONNEES.*

6.3.4 Redirection vers l'ACS et authentification 3DS

Dans le cas où le porteur est enrôlé 3D, il est nécessaire d'effectuer la redirection vers l'ACS afin que celui-ci puisse s'authentifier.

Pour cela, il faut extraire le champ **o3d_office_url_acs** de la réponse à card3D_CheckEnrollment et rediriger le navigateur de l'internaute vers cette URL. Pour tous les cas (erreur ou porteur non enrôlé), ce champ sera vide.

Le navigateur de l'internaute sera ensuite redirigé vers l'ACS de sa banque afin qu'il puisse effectuer son authentification 3D.

En phase de test (demo), l'ACS correspondra au screenshot suivant :

Cette page permet seulement de simuler le passage par un ACS. La saisie d'un mot de passe n'est pas obligatoire et n'aura aucune conséquence sur la suite de la cinématique.

Une fois cette page validée, le navigateur sera alors redirigé vers l'URL initialement présente dans le champ **merchant_url_return** de la requête card3D_CheckEnrollment.

Il est à noter qu'aucun paramètre n'est à récupérer lors de cette redirection chez le commerçant. Cela permet simplement au commerçant de reprendre la main afin d'effectuer une nouvelle requête.

Le site commerçant enverra enfin la dernière requête choisie :

- card3D_Order pour effectuer la vérification de l'authentification 3D et la demande d'autorisation 3D
- card3D_Authenticate pour seulement effectuer la vérification de l'authentification 3D.

6.3.4.1 Règles de simulation du serveur de démo

Le processus de vérification d'enrôlement 3D est simulé sur le serveur de démonstration, il est donc possible de renseigner n'importe quel numéro de carte sans aucune conséquence.

Le code réponse (**3d_response_code**) de la transaction simulée est donné par les 11 et 12ème chiffres du numéro de carte bancaire .

Exemple :

Numéros de carte	Decription du test
4970 xxxx xx00 xxxx	Porteur enrôlé 3D-Secure
4970 xxxx xx01 xxxx	Porteur non enrôlé 3D-Secure
4970 xxxx xx10 xxxx	Impossible de déterminer si le porteur est enrôlé 3D ou non
4970 xxxx xx81 xxxx	Erreur interne sur le MPI
4970 xxxx xx85 xxxx	MPI injoignable
4970 xxxx xx94 xxxx	Erreur technique au cours de l'authentification sur le Directory Server
4970 xxxx xx12 xxxx	Paramètres transmis au MPI invalides
4970 xxxx xx98 xxxx	Problème réseau lors de la tentative d'accès aux Directory Server

6.3.5 Requête de demande d'autorisation 3D : card3D_Order

6.3.5.1 Objet

Cette fonction permet au commerçant disposant des coordonnées bancaires de ses clients d'effectuer lui-même les demandes d'autorisation 3D (cf. *PRESENTATION FONCTIONNELLE)*.

6.3.5.2 Paramètres de la requête et de la réponse

Dans le tableau ci-dessous sont décrits tous les champs de la requête de demande d'autorisation 3D.

Remarque

Si une carte ne possède pas de date de validité, le champ **card_validity** doit être présent dans la requête et non renseigné.

Bien que les champs **cvv_key** et **cvv_flag** soient tous les deux facultatifs il doit y avoir une cohérence entre eux. Si le champ **cvv_key** est renseigné, le champ **cvv_flag** doit obligatoirement contenir la valeur 1 (cf. Annexe K du *DICTIONNAIRE DES DONNEES*).

nom du champ	Facultatif/ Obligatoire	Utilisation
component	O	nom du composant appelé
name	O	nom de la fonction appelée
amount	O	montant de la transaction
capture_day	F	délai d'envoi en banque
capture_mode	F	mode d'envoi en banque
customer_ip_address	F	adresse IP de l'internaute
data	F	paramétrage particulier
merchant_country	O	code pays du commerçant
merchant_id	O	identifiant du commerçant
order_id	F	numéro de commande du commerçant
order_validity	F	non utilisé
origin	F	permet d'identifier le programme à l'origine de la demande d'autorisation
return_context	F	contexte de la transaction
transaction_id	O	numéro de la nouvelle transaction
o3d_session_id	O	Identifiant de session reçu en retour de l'appel à card3D_CheckEnrollment
transmission_date	O*	date de la requête
version	O*	version du composant

** renseigné par le composant*

Dans le tableau ci-dessous sont décrits tous les champs de la réponse d'une demande d'autorisation 3D.

nom du champ	origine de la valeur	description
o3d_response_code	Renseigné par le serveur 3D Office	Code réponse de l'authentification 3D
authorisation_id	renseigné par le serveur bancaire (si transaction autorisée)	Identifiant d'autorisation retourné par la banque
bank_response_code	renseigné par le serveur bancaire	Code réponse du serveur d'autorisation bancaire ou privatif
complementary_code	renseigné par le serveur Office	Code réponse complémentaire du serveur
complementary_info	renseigné par le serveur Office	Information sur le code réponse complémentaire
currency_code	identique à la requête	Code de la devise
cvv_response_code	renseigné par le serveur bancaire (si demande d'autorisation avec cryptogramme effectuée)	Champ renvoyé dans le cas d'une demande d'autorisation avec cryptogramme visuel
avs_response_code	renseigné par le serveur Office (si demande d'autorisation avec vérification d'adresse effectuée)	Champ renvoyé dans le cas d'une demande d'autorisation d'un porteur britannique
data	identique à la requête	Champ privé
status	renseigné par le serveur Office	l'état de la transaction après une opération
response_code	renseigné par le serveur Office ou le serveur 3DOffice	Code réponse de la demande d'autorisation
transaction_date	renseigné par le serveur Office	date et heure GMT
transaction_time	renseigné par le serveur Office	heure locale du serveur Office
transaction_certificate	renseigné par le serveur Office (si transaction acceptée)	Champ certifiant que la transaction a été traitée par le serveur Office

Remarque

Le **transaction_id** et l'**amount**, déjà transmis lors de la requête card3D_CheckEnrollment, sont une nouvelle fois des champs obligatoires lors de la requête card3D_Order ou card3D_Authenticate. Il est obligatoirede transmettre un **transaction_id** et un **amount** identiques que ceux envoyés dans la première requête.

Le **transaction_id** de la requête card3D_Order sera inscrit en base de données alors que celui de la requête card3D_CheckEnrollment est utilisé pour le cryptage des messages échangés.Quant à l'**amount**, celui de la requête card3D_Order est inscrit en base puis remisé en banque alors que celui de la requête card3D_CheckEnrollment est seulement affiché sur l'ACS.

Remarque

Lorsque la carte de paiement n'est pas enrôlé 3D-Secure, la demande de paiement s'effectue tout de même avec la requête card3D_Order du composant Checkout et non plus avec la fonction author() du composant Office.

6.3.5.3 Exemple de requête et de réponse XML

Ci-dessous est présenté un exemple de requête XML de demande d'autorisation 3D.

```
<service component="checkout" name="card3D_Order">
<card3D_Order origin="Batch"
merchant_id="011223344553334"
merchant_country="fr"
transaction_id="130685"
amount="12300"
return_context="context"
order_id="OI_131100_8744"
capture_mode="VALIDATION"
capture_day="2"
data=""
order_validity=""
o3d_session_id=" 71B78471DAC1B849421690AF2C418931.sips_3doffice-1"
/></service>
```

Pour connaître la signification de ces différents champs référez-vous au *DICTIONNAIRE DES DONNEES.*

Ci-dessous est présentée la réponse XML de la demande d'autorisation 3D précédente.

```
<response component="checkout" name="card3D_Order">
<card3D_Order response_code="00"
o3d_response_code="00"
transaction_time="114147"
transaction_date="20030801"
transaction_certificate="1059730907"
authorisation_id="1059"
status="TO_VALIDATE"
currency_code="978"
data=""
avs_response_code=""
cvv_response_code="4D"
bank_response_code="00"
complementary_code=""
complementary_info="" />
</response>
```

Pour connaître la signification de ces différents champs référez-vous au *DICTIONNAIRE DES DONNEES.*

6.3.5.4 Règles de simulation du serveur de démo

Le processus d'autorisation 3D est simulé sur le serveur démonstration, il est donc possible de renseigner n'importe quel numéro de carte sans aucune conséquence.

Le code réponse (**o3d_response_code**) de l'authentification du porteur sur l'ACS est donné par les 13 et 14ème chiffres du numéro de carte bancaire.

Le code réponse (**response_code**) de la transaction simulée est donné par les deux derniers chiffres du numéro de la carte bancaire.

Exemple :

Numéros de carte	Decription du test
4970 xxxx xxxx 0000	Porteur authentifié et autorisation acceptée
4970 xxxx xxxx 0005	Porteur authentifié et refus de la demande d'autorisation
4970 xxxx xxxx 5575	Porteur non authentifié et refus de la demande d'autorisation pour suspicion de fraude
4970 xxxx xxxx 6200	By-pass du porteur sur l'ACS et autorisation acceptée

Pour les cartes VISA, MASTERCARD et CB, le cryptogramme visuel peut être renseigné (cf. Annexe K du *DICTIONNAIRE DES DONNEES*). Les cryptogrammes se terminant par 00 ou 40 sont considérés valides, les autres sont considérés invalides.

6.3.6 Requête de demande d'authentification 3D : card3D_Authenticate

6.3.6.1 Objet

Cette fonction permet au commerçant de contrôler l'authentification 3D d'un porteur sans effectuer de demande d'autorisation.

6.3.6.2 Paramètres de la requête et de la réponse

Dans le tableau ci-dessous sont décrits tous les champs de la requête de demande d'authentification 3D.

nom du champ	Facultatif/ Obligatoire	Utilisation
component	O	nom du composant appelé
name	O	nom de la fonction appelée
amount	O	montant de la transaction
merchant_country	O	code pays du commerçant
merchant_id	O	identifiant du commerçant
origin	F	permet d'identifier le programme à l'origine de la demande d'autorisation
transaction_id	O	numéro de la nouvelle transaction
o3d_session_id	O	Identifiant de session reçu en retour de l'appel à card3D_CheckEnrollment
data	F	Paramètre particulier
transmission_date	O*	date de la requête
version	O*	version du composant

** renseigné par le composant*

Dans le tableau ci-dessous sont décrits tous les champs de la réponse de la demande d'authentification 3D.

nom du champ	origine de la valeur	description
o3d_response_code	renseigné par le serveur 3D Office	Code réponse de l'authentification 3D

6.3.6.3 Exemple de requête et de réponse XML

Ci-dessous est présenté un exemple de requête XML de demande d'authentification 3D.

```
<service component="checkout" name="card3D_Authenticate">
<card3D_Authenticate origin="API SERVER"
merchant_id="011223344553334"
merchant_country="fr"
transaction_id="130685"
data=""
amount="3200"
o3d_session_id=”3D8B9C023BBDD10EA45294AEC9C2C922.jvm1”
</service>
```

Pour connaître la signification de ces différents champs référez-vous au *DICTIONNAIRE DES DONNEES.*

Ci-dessous est présentée la réponse XML de la demande d'authentification 3D précédente.

```
<response component="checkout" name="card3D_Authenticate">
< card3D_Authenticate o3d_response_code="00"
/></response>
```

Pour connaître la signification de ces différents champs référez-vous au *DICTIONNAIRE DES DONNEES.*

6.3.6.1 Règles de simulation du serveur de démo

Le processus d'authentification 3D est simulé sur le serveur de démonstration, il est donc possible de renseigner n'importe quel numéro de carte sans aucune conséquence.

Le code réponse (**o3d_response_code**) de l'authentification du porteur sur l'ACS est donné par les 13 et 14ème chiffres du numéro de carte bancaire.

Exemple :

Numéros de carte	**Decription du test**
4970 xxxx xxxx 00xx	Porteur authentifié
4970 xxxx xxxx 55xx	Porteur non authentifié
4970 xxxx xxxx 62xx	By-pass du porteur sur l'ACS

6.3.7 Absence de reponse et messages d'erreur

Si vous ne recevez pas de réponse, vous devez vérifier les points suivants :

- L'API Server est-il bien démarré ? (cf. chapitre **Lancement du serveur** du *GUIDE D'INSTALLATION*)
- N'y a-t-il pas un firewall entre le client et l'API Server qui pourrait empêcher l'accès au serveur ?
- Est-ce que l'adresse de la machine cliente est paramétrée dans les listes d'accès de l'API Server ? (cf. paragraphe **Configuration des listes d'accès** du GUIDE D'INSTALLATION)
- L'adresse IP de l'API Server que vous utilisez pour votre connexion socket est-elle correcte ?
- Le port de l'API Server que vous utilisez pour votre connexion socket est-il identique à celui configuré dans le fichier config.xml (cf. paragraphe **Configuration des paramètres de service** du GUIDE D'INSTALLATION)
- Avez-vous enregistré le composant Checkout (cf. paragraphe **La commande « ENREGISTRER UN NOUVEAU COMPOSANT »** du GUIDE D'ADMINISTRATION)

Si vous obtenez un message d'erreur tel que celui présenté ci-dessous :

```
<Error message=" message d'erreur "/>
```

Cela signifie que la requête a bien été reçue par l'API Server, mais elle comporte une erreur. Vous trouverez dans le tableau ci-dessous les principaux messages renvoyés par le composant ainsi que leur origine.

Exemple de messages	Cause	solution
(Le chemin d'accès spécifié est introuvable)	il y a une erreur au niveau du paramètre « id=″checkout″ » du sous-attribut pathfile dans le fichier *config.xml*	Corriger la configuration en vous référant au paragraphe 6.2.2.3.
Element "service" does not allow "toto" here.	toto n'est pas un nom d'opération correct dans la requête XML.	Corriger le nom de l'opération en vous référant aux exemples des paragraphes 6.3.2 à 6.3.6
Attribute "toto" is not declared for element "checkCheckout".	Vous utilisez un nom d'attribut inconnu dans la requête XML.	Vérifier le nom des attributs en vous référant aux exemples des paragraphes 6.3.2 à 6.3.6
Root element type is "service", but was declared to be "command".	Vous utilisez le port de commande de l'API Server au lieu du port de service	Utiliser le port de service configuré dans le fichier *config.xml* pour votre connexion socket à l'API Server (cf. (cf. GUIDE D'ADMINISTRATION).
Autres messages		Contacter le Centre d'Assistance Technique

6.4 DESCRIPTION DES TRACES

Suite à la réception d'une requête, le composant Checkout écrit dans les fichiers de traces configurés au paragraphe 6.2.2.4.

6.4.1 Si aucune erreur n'est survenue

Le composant inscrit dans les traces le nom de la fonction, la clé définissant la transaction créée ou modifiée (**merchant_id**, **merchant_country**, **transaction_id** et **payment_date**), le code réponse de l'opération et la réponse XML renvoyée au client de l'API Server.

Un exemple de traces pour une vérification d'enrôlement est présenté ci-dessous.

```
132:18:54:09-0-(3)-CheckoutWrapper : fonction : card3D_CheckEnrollment
132:18:54:09-0-(3)-CheckoutWrapper : merchant_id=011223344552222, merchant_country=fr
132:18:54:10-0-(3)-CheckoutWrapper : response_code=00
132:18:54:10-0-(3)-CheckoutWrapper : response : <response component="checkout"
name="card3D_CheckEnrollment"><card3D_CheckEnrollment merchant_country="fr"
merchant_id="011223344552222" 3d_response_code="00"
3d_checkenrolled_resp="00&VISA&eJxVUdtugkAQ/RXCc2UvgKIZ1lC1qQ9eYvG5IbBBjIAuUPTvO4u
oLS875zBzZuYMTK/5yfiRqsrKwjeZRU1DFnGZZEXqm/vwY+CZUwHhQUk5/5Jxo6SAlayqKJVGlvhmU
JfVYKOyNCu+GXfYkHvMdobcMQVsg528COjVBYpbHMgDooyKD1FRC4jiy/tyLZzxiFIKpIeQS7WcC8o
Y57btOK7L8QNyp6GIcilmZY4wRhWjllUNpGMhLpuiVjfBXZR7AGjUSbRta+lMKy5zIJoB8hpk2+ioQoVrl
ojVMWg3YcBW4bJdH/cUX7YOU74OFz4QnQFJVEvBKR1Tl3GDeRPXmTCcseMhynVrwbj9Rqmx2O9
wtzsFZ90peP7Xa/+lAJ1WeIibGI88XOGBQF7PZSExA5s8YyCvwWef2s+4Rovuj/azg7o2Qx8YOtoVaw
BEF5D+VKQ/LUb/Tv4LM+yztw==&20090512185412&01=01&https://rcet-3dsecure.sips-
atos.com/acs/simu/ACSServlet&26@011223344552222&www.test.com&9vGqYkS4wJc6ymwrMDJdG
VCqF+U=
" correct_card_type="VISA" 3d_office_url_acs="https://vre16:29567/3doffice/prod/call_acs"
3d_session_id="F783C00697C899CFE6E85713BD532619.sips_3doffice-1"/></response>
132:18:54:10-0-(3)-CheckoutWrapper : ----------
```

Dans le cas de la fonction de vérification checkCheckout, les champs **transaction_id** et **payment_date** contiennent respectivement la référence de la transaction et la date de la requête de vérification.

Un exemple de traces de vérification est présenté ci-dessous.

```
132:18:53:41-0-(2)-CheckoutWrapper : fonction : checkCheckout
132:18:53:41-0-(2)-CheckoutWrapper : merchant_id=011223344552222, merchant_country=fr
132:18:53:41-0-(2)-CheckoutWrapper : ----------
132:18:53:41-0-(2)-CheckoutWrapper : response : <response pathfile="OK" certificate="OK" />
132:18:53:41-0-(2)-CheckoutWrapper : ----------
```

6.4.2 En cas d'erreur

Le composant inscrit dans les traces le nom de la fonction appelée suivi du message d'erreur complet.

Un exemple de traces en cas d'erreur lors d'une demande d'autorisation est présenté ci-dessous.

```
238:10:43:09-2-(8)-CheckoutWrapper : fonction : checkCheckout
238:10:43:09-2-(8)-CheckoutWrapper : Error message:
C:\sips_office\API\Server\checkout\param\pathfile (Le fichier spécifié est introuvable)
238:10:43:09-2-(8)-CheckoutWrapper : ----------
```

7. LE COMPOSANT PAYID 1.03

7.1 FONCTIONNEMENT GENERAL DU COMPOSANT PAYID

Le composant PayId permet de gérer les paiements réguliers et/ou récurrent (comme lesabonnements) , en stockant de manière sécurisée l'association entre un identifiant d'abonné et des informations cartes. Au travers de l'abonnement, il propose aussi des fonctionnalités de paiement, en s'appuyant complètement sur la solution Office.

Pour plus d'information sur la création des transactions et sur les opérations permettant de les modifier, référez-vous à la partie composant Office de ce guide et à la *PRESENTATION FONCTIONNELLE.*

7.2 INSTALLATION ET PARAMETRAGE DU COMPOSANT

7.2.1 Installation

7.2.1.1 Liste des objets livrés

Le composant PayId est livré sous la forme d'un fichier *composant_payid_103.tar* contenant les fichiers suivants :

Fichier *Version.txt*	Fichier précisant l'environnement dans lequel le composant a été compilé et testé.
Répertoire *payid/param*	
certif.fr.011223344553333	Certificat de la boutique de démonstration
pathfile	Fichier des informations proxy et du chemin d'accès au certificat
Répertoire *server/components/service*	
payid*.jar*	archive du composant PayId

7.2.1.2 Copie des fichiers

Vous devez tout d'abord décompresser le fichier *composant_payid_103.tar*.

Pour un OS de type Unix :

Utiliser la commande suivante : tar –xvf *composant_payid_103.tar*

Pour Windows :

Utiliser un logiciel de décompression qui accepte les fichiers d'extension « .tar ».

Après avoir décompressé le fichier *composant_payid_103.tar*, vous devez copier le répertoire *payid* et tout ce qu'il contient dans le répertoire principal de l'API Server. Le répertoire principal de l'API Server contient également les répertoires associés aux autres composants que vous utilisez, ainsi que le répertoire *server*.

Vous devez également copier le fichier *payid.jar* dans le répertoire *server/components/service* du répertoire principal de l'API Server.

7.2.2 Configuration

7.2.2.1 Les paramètres du proxy

Si vous utilisez un proxy pour vous connecter à Internet, vous devez le paramétrer dans le fichier *pathfile* copié précédemment dans le répertoire *payid/param*. Ce paramétrage permettra au composant PayId de contacter le serveur PayId d'Atos Origin via l'Internet (cf. étape 3 de la Figure 1).

Si vous n'utilisez pas de proxy pour vous connecter à Internet, vous ne devez rien indiquer dans les champs suivants :

PROXY_HOST : adresse IP ou nom du proxy. PROXY_PORT : le numéro de port du proxy.

En cas de doute, votre administrateur réseau pourra vous indiquer les valeurs à renseigner.

Exemple :
Si vous passez par le proxy d'adresse IP 111.11.11.11 et de port 7878 vous devez paramétrer les champs :

PROXY_HOST!111.11.11.11! PROXY_PORT!7878!

7.2.2.2 Configuration du chemin vers les fichiers certificats

Le chemin vers les fichiers certificat est localisé dans le fichier *pathfile* copié précédemment dans le répertoire *payid/param*. Ce chemin doit être renseigné dans le champ F_CERTIFICATE et est unique pour tous les certificats du composant PayId. Tous les fichiers certificats que vous utiliserez avec le composant PayId devront donc être localisés dans un même répertoire.

Remarque

Les extensions du fichier certificat ne doivent en aucun cas être indiquées dans le champ F_CERTIFICATE. En effet, ces extensions correspondent aux paramètres merchant_country et merchant_id transmis dans les requêtes associées aux différentes opérations. Ils seront donc ajoutés par le composant au paramètre F_CERTIFICATE pour obtenir le chemin complet du certificat souhaité.

Exemple Windows :

Si le fichier *certif.fr.011223344553333* est localisé dans le répertoire *C:\API_Server\payid\param*\, vous devez renseigner le champ F_CERTIFICATE comme suit :

F_CERTIFICATE!C:\API_Server\payid\param\certif!

Exemple autres OS :

Si le fichier *certif.fr.011223344553333* est localisé dans le répertoire */home/API_Server/payid/param/*, vous devez renseigner le champ F_CERTIFICATE comme suit :

F_CERTIFICATE!/home/API_Server/payid/param/certif!

7.2.2.3 Configuration du chemin vers le fichier pathfile

Vous devez paramétrer dans le fichier *config.xml* le chemin vers le fichier *pathfile* du composant PayId. Le fichier *config.xml* est localisé dans le répertoire *server/config/* de l'API Server.

Vous trouverez ci-dessous le paragraphe concerné dans le fichier *config.xml* :

```
<components>
<pathfile id="example" path="chemin_du_fichier_pathfile" />
</components>
```

Vous devez ajouter la ligne suivante entre les balises <components> et </components> :

```
<pathfile id="payid" path="chemin_absolu_vers_le_fichier_pathfile" />
```

avec :

id : l'identifiant du composant (ici payid).

path : le chemin absolu vers le fichier *pathfile*.

Remarque

Le nom indiqué dans le champ « id » doit être identique au nom de l'archive sans l'extension « .jar » (cf. paragraphe 7.2.1.1).

Exemple Windows :

Si le fichier *pathfile* est localisé dans le répertoire *C:\API_Server\payid\param*\, vous devez renseigner la ligne comme suit :

```
<pathfile id="payid" path="C:\API_Server\payid\param\pathfile" />
```

Exemple autres OS :

Si le fichier *pathfile* est localisé dans le répertoire */home/API_Server/payid/param*, vous devez renseigner la ligne comme suit :

```
<pathfile id="payid" path="/home/API_Server/payid/param/pathfile" />
```

7.2.2.4 Configuration des traces

Vous devez paramétrer dans le fichier *config.xml* le chemin des fichiers de traces du composant PayId. Le fichier *config.xml* est localisé dans le répertoire *server/config/* de l'API Server.

Vous trouverez ci-dessous le paragraphe concerné dans le fichier *config.xml* :

```
<watchdog>
        <pollingTimer>6000</pollingTimer>
        <survPort>7183</survPort>
        <trace level="0" sizeLimit="1000" unit="Line" path="C:\chemin_racine\server\trace"
prefix="APIServer" />
```

```
<!--    <alternateTrace id="example" path="C:\chemin_racine\server\trace" prefix="example" />
        -->
        <alternateTrace id="" path="./" prefix="" />
</watchdog>
```

Vous devez ajouter la ligne suivante au-dessus du tag « </watchdog> » (la ligne avec l'identifiant example est en commentaire).

```
<alternateTrace id="payid" path="chemin_absolu_vers_le_répertoire_traces" prefix="payid" />
```

avec :

id : l'identifiant du composant (ici payid).

path : le chemin absolu vers le répertoire des traces. Vous pouvez utiliser pour vos traces le répertoire *server/traces* du répertoire principal de l'application ou tout autre répertoire de votre choix.

prefix : préfixe du fichier de traces du composant. Dans l'exemple ci-dessus, le nom du composant est utilisé, mais tout autre préfixe est également valable.

Remarque

le nom indiqué dans le champ « id » doit être identique au nom de l'archive sans l'extension « .jar » (cf. paragraphe 7.2.1.1).

La ligne « <alternateTrace id="" path="./" prefix="" />" est nécessaire au démarrage de l'API Server dans le cas ou aucun composant n'est installé, il est donc important de ne pas la modifier.

Exemple Windows :

Si le chemin du répertoire trace est *C:\API_Server\server\traces*, vous devez renseigner la ligne comme suit :

<alternateTrace id="payid" path="C:\API_Server\server\traces" prefix="payid" />

Exemple autres OS :

Si le chemin du répertoire trace est */home/API_Server/server/traces*, vous devez renseigner la ligne comme suit :

<alternateTrace id="payid" path="/home/API_Server/server/traces" prefix="payid" />

7.2.2.5 Adresse IP pour la configuration de votre firewall

Si vous utilisez un firewall pour vous connecter au réseau Internet, vous devez paramétrer au niveau de celui-ci l'adresse IP et le port du serveur PayId d'Atos Origin. Pour communiquer avec le serveur Office d'Atos Origin votre firewall doit vous permettre une connexion socket en protocole TCP/IP vers l'adresse IP 193.56.46.110 port 2001.

7.3 DESCRIPTION DETAILLEE DU COMPOSANT

Ce chapitre décrit les opérations disponibles avec le composant PayId. Pour chacune de ces opérations, des exemples de requête et de réponse au format XML sont fournis, ainsi que les listes des paramètres de la requête et de la réponse avec des liens vers le dictionnaire des données.

Avant de décrire les fonctions disponibles, nous allons préciser les aspects de la sécurisation du transfert des informations entre le composant PayId et le serveur Office d'Atos Origin.

Remarques

- Afin d'alléger le code des requêtes, les entêtes standard XML ne sont pas nécessaires, mais sont ajoutées par l'API Server pour vérification lors du parsing XML (utilisation d'un DTD). Par contre, lors des réponses de l'API Server, vers l'application cliente, aucune référence à un DTD n'est incluse dans le message. Il est donc important de désactiver la validation lors du parsing, afin de ne pas déclencher des erreurs durant cette phase.
- Pour des raisons de lisibilité, toutes les requêtes décrites dans ce chapitre sont présentées sur plusieurs lignes. Toutefois, lors de l'appel à l'API Server, elles doivent être codées sans retours chariot.

7.3.1 Sécurisation du transfert des informations

Voir chapitre **La sécurité des paiements en ligne avec SIPS Office Server** de la *PRESENTATION FONCTIONNELLE.*

7.3.2 Résumé des fonctions disponibles

Le composant PayId propose les fonctions :

- **checkpayid** : vérification locale de l'accès aux fichiers pathfile et certificat
- **check** : contrôle d'existence d'un abonné
- **sign_in** : création d'un abonné (*)
- **modify** : modification d'un abonné (*)
- **sign_off** : suppression d'un abonné
- **author** : réalisation d'un paiement

documentées dans les paragraphes suivants. Les fonctions marquées d'un (*) permettent de réaliser un paiement dans la foulée.

7.3.3 Contraintes de présence des champs

Les champs faisant partie du langage du composant sont listés dans le *DICTIONNAIRE DES DONNEES*. Avant de passer à la description unitaire de chaque fonction, voici deux tableaux synoptiques. Ils précisent, pour chaque champ du dictionnaire et pour chaque fonction disponible du composant PayId, les contraintes de présence s'appliquant à ce champ dans la requête et la réponse de cette fonction.

7.3.3.1 Requêtes

Ce tableau indique les contraintes de présence s'appliquant aux champs des requêtes.

Champs requêtes	check	sign off	sign in	modify	author	Utilisation
merchant_country	O	O	O	O	O	code pays du commerçant
merchant_id	O	O	O	O	O	identifiant du commerçant
customer_id	O	O	F	O	O	numéro d'abonné
customer_pwd	-	-	-	C1	C1	mot de passe à valider
customer_newpwd	-	-	F	C6	-	établissement d'un nouveau mot de passe
payid_type	-	-	F	F	-	"0" prise d'empreinte seule, "1" paiement
card_type	-	-	O	C5	-	type de carte (CB, VISA, MASTERCARD…)
card_number	-	-	O	C5	-	numéro de la carte
card_number_format	F	-	-	-	-	format du numéro crypté
invalidation_date	-	-	F	-	-	Date d'invalidation de l'abonné
cvv_flag	-	-	O	O	O	présence d'un CVV
cvv_key	-	-	C2	C2	C2	CVV
card_validity	-	-	C3	C5	-	date de validité de la carte
amount	-	-	C4	C4	O	montant à valider
currency_code	-	-	O	O	O	code de la monnaie utilisée
transaction_id	-	-	O	O	O	numéro de la transaction à valider
capture_mode	-	-	C4	C4	C4	mode d'envoi en banque
capture_day	-	-	C4	C4	C4	délai d'envoi en banque
transmission_date	O1	O1	O1	O1	O1	date de la requête
origin	-	-	F	F	F	programme à l'origine de la requête
order_id	-	-	F	F	F	numéro de commande du commerçant
data	-	-	F	F	F	paramétrage particulier
return_context	F	F	F	F	F	contexte de la nouvelle transaction
version	O1	O1	O1	O1	O1	version du composant

O : champ obligatoire
O1 : champ obligatoire renseigné par le composant
F : champ facultatif
C1 : champ conditionnel => obligatoire si mot de passe dans la base, non requis sinon
C2 : champ conditionnel => obligatoire en fonction du cvv_flag
C3 : champ conditionnel => voir spécificités des cartes ou demandes spécifiques
C4 : champ conditionnel => obligatoire si payid_type = 1
C5 : champ conditionnel => obligatoire si customer_newpwd non renseigné
C6 : champ conditionnel => obligatoire si card_type, card_number et card_validity non renseigné
- : champ non présent

7.3.3.2 Réponses

Ce tableau indique les contraintes de présence s'appliquant aux champs des réponses.

Champs requêtes	check	sign off	sign in	modify	author	Utilisation
merchant_country	O	O	O	O	O	code pays du commerçant
merchant_id	O	O	O	O	O	Identifiant du commerçant
sub_response_code	O	O	O	O	O	code réponse du module payId renseigné par le serveur Atos
response_code	-	-	C1	C1	C1	code réponse de la demande d'autorisation renseigné par le serveur Atos
bank_response_code	-	-	C1	C1	C1	code réponse de l'acquéreur renseigné par le serveur bancaire en cas d'autorisation
customer_id	-	-	C3	-	-	identifiant du client
card_type	C1	-	-	-	-	type de carte (CB, VISA, MASTERCARD…)
card_number	C1	-	-	-	-	numéro de la carte
card_validity	F	-	-	-	-	date de validité de la carte
authorisation_id	-	-	C2	C2	C2	renseigné par le serveur bancaire en cas d'autorisation
complementary_code	-	-	C2	C2	C2	renseigné par le serveur Atos en cas d'autorisation
currency_code	-	-	C2	C2	C2	identique à la requête
cvv_response_code	-	-	C2	C2	C2	non renseigné
data	-	-	C2	C2	C2	paramétrage particulier
status	-	-	C2	C2	C2	renseigné par le serveur Atos
transaction_date	-	-	C1	C1	O	renseigné par le serveur Atos
transaction_time	-	-	C1	C1	O	renseigné par le serveur Atos
transaction_certificate	-	-	C2	C2	C2	renseigné par le serveur Atos (si opération acceptée)
return_context	C4	C4	C4	C4	C4	identique à la requête
complementary_info	-	-	C2	C2	C2	renseigné par le serveur Atos en cas d'autorisation

O : champ obligatoirement restitué
C1 : champ conditionnel => obligatoire si payid_response_code = "00"
C2 : champ conditionnel => obligatoire si transaction acceptée
C3 : champ conditionnel => obligatoire si généré
C4 : champ conditionnel => si présent dans la requête
F : champ restitué selon le cas
- : champ non présent

Le service checkpayid étant très particulier dans sa réponse, se référer au paragraphe détaillé le concernant.

7.3.4 Requête de vérification du composant : checkpayid

7.3.4.1 Objet

Cette fonction permet de tester :

- la communication entre l'application cliente que vous développez et l'API Server
- le bon fonctionnement de l'API Server
- la configuration du composant PayId
  - l'accès au fichier *pathfile*
  - l'accès au fichier certificat

7.3.4.2 Exemple de requête et de réponse XML checkpayid

Ci-dessous est présenté un exemple de requête XML de vérification. La signification des champs **merchant_country** et **merchant_id** est fournie dans le *DICTIONNAIRE DES DONNEES*. Ces champs correspondent respectivement à la première et la deuxième extension du fichier certificat (exemple : le fichier *certif.fr.011223344553333* correspond à **merchant_country**=fr et **merchant_id**=011223344553333).

```
<service component="payid" name="checkpayid">
<checkpayid merchant_id="011223344553333" merchant_country="fr"/>
</service>
```

Lors de la vérification, le composant PayId va contrôler l'accès en lecture au fichier *pathfile* paramétré dans le fichier *config.xml*, puis l'accès en lecture au fichier certificat (*certif.<merchant_country>.<merchant_id>*) dont le chemin générique est indiqué dans le fichier *pathfile*. Si ces accès sont corrects, la réponse suivante sera envoyée :

```
<response pathfile="OK" certificate="OK" />
```

Si ces accès ne sont pas corrects, la réponse suivante sera envoyée :

```
<Error message="message d'erreur"/>
```

La liste des messages d'erreur est précisée dans le tableau ci-dessous.

Messages d'erreur	Cause	solution
C:\API_Server\payid\param\pathfile (Le chemin d'accès spécifié est introuvable)	le fichier *pathfile* configuré dans le fichier *config.xml* n'est pas accessible en lecture	Vérifier les droits d'accès du fichier *pathfile* et corriger le chemin du fichier *pathfile* si nécessaire (cf. paragraphe 7.2.2.3).
Cannot open certificat. (C:\API_Server\payid\param\certif.fr.011223344553333)	le fichier certif.fr.01122334455222 n'est pas accessible en lecture	Vérifier les droits d'accès du fichier certificat *certif.fr.011223344553333* et corriger le chemin génériques des fichiers certificats si nécessaire (cf. paragraphe 7.2.2.2).

7.3.5 Requête de contrôle d'un abonné : check

7.3.5.1 Objet

Cette fonction permet de tester l'existence d'un abonné dans la base de données SIPS. Du point de vue d'un commerçant, identifié par son pays et son numéro, l'identification de l'abonné porte sur un numéro d'abonné. Du point de vue de SIPS, la fourniture des trois champs (pays du commerçant, numéro commerçant, numéro d'abonné) est donc requise en entrée de la fonction.

Si l'abonné n'existe pas, la réponse véhicule un code réponse 01.

Si l'abonné existe, la réponse véhicule le code réponse 00 ainsi qu'un lot d'informations concernant cet abonné :

- le type de carte
- le numéro de carte, partiellement masqué (seuls les 4 premiers et 2 derniers chiffres sont en clair)
- la date de validité

7.3.5.2 Exemple de requête et de réponse XML check

Ci-dessous est présenté un exemple de requête XML:

```
<service component="PayId" name="check">
	<check
		merchant_id="011223344553333"
		merchant_country="fr"
		customer_id="xxxxxx"
	/>
</service>
```

Et deux exemples de réponse:

```
<response component="PayId" name="check">
	<check
		payid_response_code ="00"
		card_type="VISA"
		card_number="4907####91"
		card_validity="200710"
	/>
</response>
```

```
<response component="PayId" name="check">
	<check
		payid_response_code ="01"
	/>
</response>
```

7.3.5.1 Règles de simulation du serveur de démo

Voici la règle qui permet de calculer le code retour en démonstration.

Si le customer_id = 00 ou 01, le payid_response_code = customer_id

sinon le payid_response_code = 99

7.3.6 Requête de création d'un abonné : sign_in

7.3.6.1 Objet

Cette requête est émise par le commerçant lorsque ce dernier veut initier l'association entre un numéro de carte et un numéro d'abonné et, éventuellement, dans la foulée, effectuer un paiement ; le commerçant utilise la fonction "sign_in".

Le numéro d'abonné fourni doit être inexistant.

Le choix du mode, avec paiement / sans paiement, s'effectue au moyen du champ **payid_type** :

0 (valeur par défaut) : sans paiement

1 : avec paiement

Dans le mode avec paiement

- **transaction_id** doit être renseigné
- **amount** doit être renseigné
- **currency_code** doit être renseigné
- **capture_mode** peut être renseigné (valeur par défaut : AUTHOR_CAPTURE)
- **capture_day** peut être renseigné (valeur par défaut : 0)

Dans le mode sans paiement

- **transaction_id** doit quand même[1] être renseigné
- **amount** ne doit pas être renseigné
- **currency_code** doit quand même être renseigné
- **capture_mode** ne doit pas être renseigné
- **capture_day** ne doit pas être renseigné

L'autorisation ou prise d'empreinte effectuée permet au serveur de s'assurer de la validité des coordonnées cartes fournies. Si ces coordonnées sont valides, l'abonné est créé et les coordonnées cartes sont stockées en base.

Si le **customer_newpwd** est renseigné, le mot de passe correspondant est stocké ; la validation du mot de passe (champ **customer_pwd**) sera demandée pour toute opération autre que l'invalidation.

Le code réponse **payid_response_code** vaut

- 00 en cas de succès,
- 02 en cas d'erreur : abonné déjà existant
- 03 en cas d'erreur : demande d'autorisation ou prise d'empreinte refusée
- 99 en cas d'erreur indéterminée

Cf. tableau synoptique des codes retours, paragraphe 7.3.10.

[1] Dans ce mode, la requête donne en effet naissance à une prise d'empreinte (demande d'autorisation de 2 euros) dont le but est d'éprouver la validité des coordonnées cartes fournies.

7.3.6.2 Exemple de requête et de réponse XML sign_in

Exemple de requête:

```
<service component="PayId" name="sign_in">
        <sign_in
                origin="MYSHOP"
                merchant_id="011223344553333"
                merchant_country="fr"
                customer_id="xxxxxx"
                cvv_flag="1"
                cvv_key="100"
                card_validity="200804"
                card_type="VISA"
                card_number="4907000000000600"
                payid_type="1"
                amount="19835"
                currency_code="978"
                transaction_id="155529"
        />
</service>
```

Exemple de réponse:

```
<response component="PayId" name="sign_in">
        <sign_in
                payid_response_code ="00"
                response_code="00"
                transaction_time="155543"
                transaction_date="20070425"
                transaction_certificate="9337ae54ad34"
                authorisation_id="502841"
                currency_code="978"
                data=""
                cvv_response_code=""
                bank_response_code="00"
                complementary_code=""
                complementary_info=""
        />
</response>
```

7.3.6.3 Règles de simulation du serveur de démo

En démonstration, le code réponse suit la règle suivante :

Si customer_id = 02, payid_response_code = 02

Si customer_id = 00,

si 2 "avant derniers numéros de customer_id" = 00, payid_response_code = 00

sinon payid_response_code = 03 (+ response_code retourné par Office)

sinon payid_response_code = 99

Pour une description du fonctionnement du serveur de démo concernant le champ response_code est détaillé au paragraphe 4.3.2.4.

7.3.7 Requête de modification d'un abonné : modify

7.3.7.1 Objet

Cette requête est émise par le commerçant lorsque ce dernier veut modifier les données carte d'un abonné ou le mot de passe d'un abonné et, éventuellement, dans la foulée, faire un paiement; le commerçant utilise la fonction "modify"

Le numéro d'abonné fourni doit correspondre à un abonné existant et non invalidé.

Les règles de gestion du mode avec ou sans paiement sont essentiellement les mêmes que pour la fonction **sign_in** vue au paragraphe précédent. (Utilisation du champ **payid_type**, existence de contraintes de présence sur les champs liés au paiement). Le **transaction_id** n'est cependant pas obligatoire dans un cas : lorsqu'il n'y a ni paiement dans la foulée et ni modification des données carte.

Le mot de passe est contrôlé s'il existe dans la base. Si le **customer_newpwd** est renseigné, le mot de passe de l'abonné est stocké (en remplacement éventuel du précédent). La validation du mot de passe sera demandée pour toute opération autre que l'invalidation.

Le code réponse **payid_response_code** vaut

- 00 en cas de succès,
- 01 en cas d'erreur : abonné inexistant
- 03 en cas d'erreur : demande d'autorisation ou prise d'empreinte refusée
- 04 en cas d'erreur : mot de passe incorrect
- 99 en cas d'erreur indeterminée

Cf. tableau synoptique des codes retours, paragraphe 7.3.10.

7.3.7.2 Exemple de requête et de réponse XML modify

Exemple de requête :

```
<service component="PayId" name="modify">
	<modify
		origin="MYSHOP"
		merchant_id="011223344553333"
		merchant_country="fr"
		customer_id="xxxxxx"
		cvv_flag="1"
		cvv_key="100"
		card_validity="200804"
		card_type="VISA"
		card_number="4907000000000600"
		currency_code="978"
		transaction_id="187182"
	/>
</service>
```

Exemple de réponse

```
<response component="PayId" name="modify">
	<modify
		payid_response_code ="00"
		response_code="00"
		transaction_time="155543"
		transaction_date="20070425"
		transaction_certificate="9337ae54ad34"
		authorisation_id="502841"
		currency_code="978"
		data=""
		cvv_response_code=""
		bank_response_code="00"
		complementary_code=""
		complementary_info=""
	/>
</response>
```

7.3.7.3 Règles de simulation du serveur de démo

En démonstration, le code réponse suit la règle suivante :

Si customer_id = 01, payid_response_code = 01

Si customer_id = 00

Si customer_pwd != "pwd", payid_response_code = 04

sinon

si 2 "avant derniers numéros de customer_id" = 00, payid_response_code = 00

sinon payid_response_code = 03 (+ response_code retourné par Office)

sinon payid_response_code = 99

Pour une description du fonctionnement du serveur de démo concernant le champ response_code est détaillé au paragraphe 4.3.2.4.

7.3.8 Requête de suppression d'un abonné : sign_off

7.3.8.1 Objet

Cette requête est émise par le commerçant lorsque ce dernier veut invalider définitivement un abonné ; le commerçant utilise la fonction "sign_off"

Le numéro d'abonné fourni doit être existant dans la base.

L'abonné est signalé comme invalide. Aucun paiement ne pourra plus être effectué sur cet abonné. Aucune modification ultérieure ne pourra lui être apportée.

Le code réponse **payid_response_code** vaut

- 00 en cas de succès,
- 01 en cas d'erreur : abonné inexistant
- 99 en cas d'erreur indéterminée

Cf. tableau synoptique des codes retours, paragraphe 7.3.10.

7.3.8.2 Exemple de requête et de réponse XML sign_off

Exemple de requête:

```
<service component="PayId" name="sign_off">
        <sign_off
                merchant_id="000000000005555"
                merchant_country="fr"
                customer_id="xxxxxx"
        />
</service>
```

Exemple de réponse:

```
<response component="PayId" name="sign_off">
        <sign_off
                payid_response_code ="00"
        />
</response>
```

7.3.8.3 Règles de simulation du serveur de démo

En démonstration, le code réponse suit la règle suivante :

Si customer_id = 00 ou 01, payid_response_code = customer_id

sinon payid_response_code = 99

7.3.9 Requête de paiement par abonnement : author

7.3.9.1 Objet

Cette requête est émise par le commerçant lorsque ce dernier initie un paiement en utilisant un numéro d'abonné ; le commerçant utilise la fonction "author"

Le numéro d'abonné fourni doit correspondre à un abonné existant et non invalidé. Le mot de passe est contrôlé s'il existe dans la base.

La requête donne naissance à une demande d'autorisation.

Si le champ card_validity est fourni en entrée, sa valeur est transmise sans modification dans la demande d'autorisation ; sinon, la valeur de ce champ est lue dans la base de données SIPS pour former la demande d'autorisation.

Le code réponse payid_response_code vaut

- 00 en cas de succès,
- 01 en cas d'erreur : abonné inexistant
- 04 en cas d'erreur : mot de passe incorrect
- 99 en cas d'erreur indéterminée

Cf. tableau synoptique des codes retours, paragraphe 7.3.10.

7.3.9.2 Exemple de requête et de réponse XML author

Exemple de requête:

```
<service component="PayId" name="author">
	<author
		origin="MYSHOP"
		merchant_id="000000000005555"
		merchant_country="fr"
		transaction_id="155529"
		amount="200"
		currency_code="978"
		customer_id="xxxxxx"
		cvv_flag="1"
		cvv_key="100"
		return_context=""
		order_id="OI_131100_8744"
		capture_mode="AUTHOR_CAPTURE"
		capture_day="6"
		data=""
	/>
</service>
```

Exemple de réponse :

```
<response component="PayId" name="author">
	<author
		payid_response_code ="00"
		response_code="00"
		transaction_time="155543"
		transaction_date="20070425"
		transaction_certificate="9337ae54ad34"
		authorisation_id="502841"
		status="TO_VALIDATE"
		currency_code="978"
		data=""
		cvv_response_code=""
		bank_response_code="00"
		complementary_code=""
		complementary_info=""
	/>
</response>
```

7.3.9.3 Règles de simulation du serveur de démo

En démonstration, le code réponse suit la règle suivante :

Si customer_id en 01, payid_response_code = 01

Si customer_id en 00,

si customer_pwd != "pwd", payid_response_code = 04

sinon payid_response_code = 00

sinon payid_response_code = 99

Pour une description du fonctionnement du serveur de démo concernant le champ response_code est détaillé au paragraphe 4.3.2.4.

7.3.10 Codes retour

L'interprétation d'une réponse s'appuie sur l'analyse de plusieurs champs code-retour.

payid_response_code indique si l'opération relative à la gestion de l'abonné a pu être effectuée.

	check	**sign_in**	**modify**	**sign_off**	**author**
00	abonné existant	abonné correctement créé	abonné correctement modifié	abonné correctement invalidé	abonné existant
01	abonné inexistant	-	abonné inexistant	abonné inexistant	abonné inexistant
02	-	abonné déjà existant	-	-	-
03	-	abonné non créé car paiement refusé	abonné non modifié car paiement refusé	-	-
04	-	-	mot de passe erroné	-	mot de passe erroné
99	erreur indéterminée	erreur indéterminée	erreur indéterminée	erreur indéterminée	erreur indéterminée

response_code indique si l'éventuelle opération d'autorisation ou de paiement, que PayId a déléguée au serveur Office, a pu être effectuée. ***response_code*** et ses valeurs possibles sont documentés dans le *DICTIONNAIRE DES DONNEES,* de même que le code retour encore sous-jacent, issu de la banque acquéreur, le ***bank_response_code***.

Combinaisons possibles de codes retours PayId et Office :

	payid_response_code	**response_code**
check	présent	absent.
sign_in	présent	présent quelle que soit la valeur de payid_response_code car toute création d'abonné fait l'objet d'une demande d'autorisation. La création est conditionnée par l'acceptation (response_code 00) de cette demande d'autorisation.
modify	présent	présent si une requête vers Office a été déclenchée, c'est-à-dire si : une modification des données cartes a été opérée ou (non exclusif) une demande de paiement dans la foulée a été effectuée.
sign_off	présent	absent.
author	présent	présent quelle que soit la valeur de payid_response_code. N.B : même si payid_response_code vaut 00, le paiement peut avoir été refusé.
modify	présent	présent si une requête vers Office a été déclenchée, c'est-à-dire si : une modification des données cartes a été opérée ou (non exclusif) une demande de paiement dans la foulée a été effectuée.
sign_off	présent	absent.
author	présent	présent quelle que soit la valeur de payid_response_code. N.B : même si payid_response_code vaut 00, le paiement peut avoir été refusé.

8. LE COMPOSANT CASHMANAGEMENT 1.00

8.1 FONCTIONNEMENT GENERAL DU COMPOSANT CASHMANAGEMENT

Le composant cashManagement permet au commerçant d'effectuer des opérations Back office, à terme ce composant contiendra les opérations de caisse aujourd'hui intégrée au composant office.

Il a été initialisé avec la mise en place du crédit porteur (carte) à partir d'un identifiant d'abonné.

Pour plus d'information sur la création des transactions et sur les opérations permettant de les modifier, référez-vous à la partie composant Office de ce guide et à la *PRESENTATION FONCTIONNELLE.*

8.2 INSTALLATION ET PARAMETRAGE DU COMPOSANT

8.2.1 Installation

8.2.1.1 Liste des objets livrés

Le composant CashManagement est livré sous la forme d'un fichier *composant_cashmanagement_100.tar* contenant les fichiers suivants :

Fichier *Version.txt*	Fichier précisant l'environnement dans lequel le composant a été compilé et testé.
Répertoire *cashmanagement/param*	
certif.fr.011223344553333	Certificat de la boutique de démonstration
pathfile	Fichier des informations proxy et du chemin d'accès au certificat
Répertoire *server/components/service*	
cashmanagement.jar	archive du composant CashManagement

8.2.1.2 Copie des fichiers

Vous devez tout d'abord décompresser le fichier *composant_cashmanagement_100.tar*.

Pour un OS de type Unix :

Utiliser la commande suivante : tar –xvf *composant_cashmanagement_100.tar*

Pour Windows :

Utiliser un logiciel de décompression qui accepte les fichiers d'extension « .tar ».

Après avoir décompressé le fichier *composant_cashmanagement_100.tar*, vous devez copier le répertoire *cashmanagement* et tout ce qu'il contient dans le répertoire principal de l'API Sips Office Server. Le répertoire principal de l'API Sips Office Server contient également les répertoires associés aux autres composants que vous utilisez, ainsi que le répertoire *server*.

Vous devez également copier le fichier *cashmanagement.jar* dans le répertoire *server/components/service* du répertoire principal de l'API Sips Office Server.

8.2.2 Configuration

8.2.2.1 Les paramètres du proxy

Si vous utilisez un proxy pour vous connecter à Internet, vous devez le paramétrer dans le fichier *pathfile* copié précédemment dans le répertoire *cashmanagement/param*. Ce paramétrage permettra au composant CashManagement de contacter le serveur CashManagement d'Atos Origin via l'Internet (cf. étape 3 de la Figure 1).

Si vous n'utilisez pas de proxy pour vous connecter à Internet, vous ne devez rien indiquer dans les champs suivants :

PROXY_HOST : adresse IP ou nom du proxy.
PROXY_PORT : le numéro de port du proxy.

En cas de doute, votre administrateur réseau pourra vous indiquer les valeurs à renseigner.

Exemple :
Si vous passez par le proxy d'adresse IP 111.11.11.11 et de port 7878 vous devez paramétrer les champs :

PROXY_HOST!111.11.11.11!
PROXY_PORT!7878!

8.2.2.2 Configuration du chemin vers les fichiers certificats

Le chemin vers les fichiers certificat est localisé dans le fichier *pathfile* copié précédemment dans le répertoire *cashmanagement/param*. Ce chemin doit être renseigné dans le champ F_CERTIFICATE et est unique pour tous les certificats du composant CashManagement. Tous les fichiers certificats que vous utiliserez avec le composant CashManagement devront donc être localisés dans un même répertoire.

Remarque

Les extensions du fichier certificat ne doivent en aucun cas être indiquées dans le champ F_CERTIFICATE. En effet, ces extensions correspondent aux paramètres merchant_country et merchant_id transmis dans les requêtes associées aux différentes opérations. Ils seront donc ajoutés par le composant au paramètre F_CERTIFICATE pour obtenir le chemin complet du certificat souhaité.

Exemple Windows :

Si le fichier *certif.fr.011223344553333* est localisé dans le répertoire *C:\API_Server\cashmanagement\param*, vous devez renseigner le champ F_CERTIFICATE comme suit :

F_CERTIFICATE!C:\API_Server\cashmanagement\param\certif!

Exemple autres OS :

Si le fichier *certif.fr.011223344553333* est localisé dans le répertoire */home/API_Server/cashmanagement/param/*, vous devez renseigner le champ F_CERTIFICATE comme suit :

```
F_CERTIFICATE!/home/API_Server/cashmanagement/param/certif!
```

8.2.2.3 Configuration du chemin vers le fichier pathfile

Vous devez paramétrer dans le fichier *config.xml* le chemin vers le fichier *pathfile* du composant CashManagement. Le fichier *config.xml* est localisé dans le répertoire *server/config/* de l'API Server.

Vous trouverez ci-dessous le paragraphe concerné dans le fichier *config.xml* :

```
<components>
<pathfile id="example" path="chemin_du_fichier_pathfile" />
</components>
```

Vous devez ajouter la ligne suivante entre les balises <components> et </components> :

```
<pathfile id="cashmanagement" path="chemin_absolu_vers_le_fichier_pathfile" />
```

avec :

id : l'identifiant du composant (ici cashmanagement).

path : le chemin absolu vers le fichier *pathfile*.

Remarque

Le nom indiqué dans le champ « id » doit être identique au nom de l'archive sans l'extension « .jar » (cf. paragraphe 8.2.1.1).

Exemple Windows :

Si le fichier *pathfile* est localisé dans le répertoire *C:\API_Server\cashmanagement\param*, vous devez renseigner la ligne comme suit :

```
<pathfile id="cashmanagement" path="C:\API_Server\cashmanagement\param\pathfile" />
```

Exemple autres OS :

Si le fichier *pathfile* est localisé dans le répertoire */home/API_Server/cashmanagement/param*, vous devez renseigner la ligne comme suit :

```
<pathfile id="cashmanagement" path="/home/API_Server/cashmanagement/param/pathfile" />
```

8.2.2.4 Configuration des traces

Vous devez paramétrer dans le fichier *config.xml* le chemin des fichiers de traces du composant CashManagement. Le fichier *config.xml* est localisé dans le répertoire *server/config/* de l'API Server.

Vous trouverez ci-dessous le paragraphe concerné dans le fichier *config.xml* :

```
<watchdog>
        <pollingTimer>6000</pollingTimer>
        <survPort>7183</survPort>
        <trace level="0" sizeLimit="1000" unit="Line" path="C:\chemin_racine\server\trace"
prefix="APIServer" />
<!--    <alternateTrace id="example" path="C:\chemin_racine\server\trace" prefix="example" />
        -->
        <alternateTrace id="" path="./" prefix="" />
</watchdog>
```

Vous devez ajouter la ligne suivante au-dessus du tag « </watchdog> » (la ligne avec l'identifiant example est en commentaire).

```
<alternateTrace id="cashmanagement" path="chemin_absolu_vers_le_répertoire_traces"
prefix="cashmanagement" />
```

avec :

id : l'identifiant du composant (ici cashmanagement).

path : le chemin absolu vers le répertoire des traces. Vous pouvez utiliser pour vos traces le répertoire *server/traces* du répertoire principal de l'application ou tout autre répertoire de votre choix.

prefix : préfixe du fichier de traces du composant. Dans l'exemple ci-dessus, le nom du composant est utilisé, mais tout autre préfixe est également valable.

Remarque

le nom indiqué dans le champ « id » doit être identique au nom de l'archive sans l'extension « .jar » (cf. paragraphe 8.2.1.1).

La ligne « <alternateTrace id="" path="./" prefix="" />" est nécessaire au démarrage de l'API Server dans le cas ou aucun composant n'est installé, il est donc important de ne pas la modifier.

Exemple Windows :

Si le chemin du répertoire trace est *C:\API_Server\server\traces*, vous devez renseigner la ligne comme suit :

```
<alternateTrace id="cashmanagement" path="C:\API_Server\server\traces" prefix="cashmanagement"
/>
```

Exemple autres OS :

Si le chemin du répertoire trace est */home/API_Server/server/traces*, vous devez renseigner la ligne comme suit :

```
<alternateTrace id="cashmanagement" path="/home/API_Server/server/traces"
prefix="cashmanagement" />
```

8.2.2.5 Adresse IP pour la configuration de votre firewall

Si vous utilisez un firewall pour vous connecter au réseau Internet, vous devez paramétrer au niveau de celui-ci l'adresse IP et le port du serveur Office d'Atos Origin. Pour communiquer avec le serveur Office d'Atos Origin votre firewall doit vous permettre une connexion socket en protocole TCP/IP vers l'adresse IP 193.56.46.110 port 2001.

8.3 DESCRIPTION DETAILLEE DU COMPOSANT

Ce chapitre décrit les opérations disponibles avec le composant CashManagement. Pour chacune de ces opérations, des exemples de requête et de réponse au format XML sont fournis, ainsi que les listes des paramètres de la requête et de la réponse avec des liens vers le dictionnaire des données.

Avant de décrire les fonctions disponibles, nous allons préciser les aspects de la sécurisation du transfert des informations entre le composant CashManagement et le serveur Office d'Atos Origin.

Remarques

- Afin d'alléger le code des requêtes, les entêtes standard XML ne sont pas nécessaires, mais sont ajoutées par l'API Server pour vérification lors du parsing XML (utilisation d'un DTD). Par contre, lors des réponses de l'API Server, vers l'application cliente, aucune référence à un DTD n'est incluse dans le message. Il est donc important de désactiver la validation lors du parsing, afin de ne pas déclencher des erreurs durant cette phase.
- Pour des raisons de lisibilité, toutes les requêtes décrites dans ce chapitre sont présentées sur plusieurs lignes. Toutefois, lors de l'appel à l'API Server, elles doivent être codées sans retours chariot.

8.3.1 Sécurisation du transfert des informations

Voir chapitre **La sécurité des paiements en ligne avec Sips Office Server** de la *PRESENTATION FONCTIONNELLE.*

8.3.2 Résumé des fonctions disponibles

Le composant CashManagement propose les fonctions :

- **checkcashmanagement** : vérification locale de l'accès aux fichiers pathfile et certificat
- **merchantwallet_credit** : demande de crédit porteur d'abonné

documentées dans les paragraphes suivants.

8.3.3 Requête de vérification du composant : checkcashmanagement

8.3.3.1 Objet

CheckCashManagement permet de tester les éléments suivants :

- communication entre l'application cliente et l'API Server ;
- bon fonctionnement de l'API Server ;
- configuration du composant CashManagement :
- accès au fichier pathfile ;
- accès au fichier certificat.

8.3.3.2 Exemple de requête et de réponse XML checkcashmanagement

Ci-dessous est présenté un exemple de requête XML de vérification. La signification des champs **merchant_country** et **merchant_id** est fournie dans le *DICTIONNAIRE DES DONNEES*. Ces champs correspondent respectivement à la première et la deuxième extension du fichier certificat (exemple : le fichier *certif.fr.011223344553333* correspond à **merchant_country**=fr et **merchant_id**=011223344553333).

```
<service component = "cashmanagement" name = "checkcashmanagement">
<checkcashmanagement merchant_id = "011223344550000"
merchant_country = "fr"
/>
</service>
```

Lors de la vérification, le composant CashManagement va contrôler l'accès en lecture au fichier *pathfile* paramétré dans le fichier *config.xml*, puis l'accès en lecture au fichier certificat (*certif.<merchant_country>.<merchant_id>*) dont le chemin générique est indiqué dans le fichier *pathfile*. Si ces accès sont corrects, la réponse suivante sera envoyée :

```
<response pathfile="OK" certificate="OK" />
```

Si ces accès ne sont pas corrects, la réponse suivante sera envoyée :

```
<Error message="message d'erreur"/>
```

La liste des messages d'erreur est précisée dans le tableau ci-dessous.

Messages d'erreur	Cause	solution
C:\API_Server\cashmanagement\ param\pathfile (Le chemin d'accès spécifié est introuvable)	le fichier *pathfile* configuré dans le fichier *config.xml* n'est pas accessible en lecture	Vérifier les droits d'accès du fichier *pathfile* et corriger le chemin du fichier *pathfile* si nécessaire (cf. paragraphe 8.2.2.3).
Cannot open certificat. (C:\API_Server\cashmanagement\ param\certif.fr.011223344553333)	le fichier certif.fr.01122334455222 n'est pas accessible en lecture	Vérifier les droits d'accès du fichier certificat *certif.fr.011223344553333* et corriger le chemin génériques des fichiers certificats si nécessaire (cf. paragraphe 8.2.2.2).

8.3.4 Requête de demande de crédit porteur d'abonné : merchantwallet_credit

8.3.4.1 Objet

MerchantWallet_Credit est le service qui permet de demander un crédit porteur d'abonné.

8.3.4.2 Exemple de requête et de réponse XML merchantwallet_credit

Ci-dessous est présenté un exemple de requête XML:

```
<service component = "cashmanagement" name = "merchantwalletcredit">
<merchantwalletcredit merchant_id = "011223344550000"
        merchant_country = "fr"
        merchantwallet_id = "554"
        merchantwallet_password = "P4ssw*rd"
        transaction_id = "26"
        amount = "12300"
        currency_code = "978"
        order_id = "OI_150_AM"
        return_context = "context" />
</service>
```

La description des différents champs est fournie dans le *DICTIONNAIRE DES DONNEES.*

Exemple de réponse :

```
<response component="cashmanagement" name="merchantwalletcredit">
<merchantwalletcredit merchant_id = "011223344550000"
        merchant_country = "fr"
        merchantwallet_id = "554"
        transaction_id = "26"
        transaction_time = "112505"
        transaction_date = "20090624"
        response_code = "00"
        new_status = "TO_CREDIT"
        new_amount = "12300"
        currency_code = "978"
        bank_response_code = "00"
        transaction_certificate = "1058584851"
        order_id = "OI_150_AM"
        return_context = "context" />
</response>
```

La description des différents champs est fournie dans le *DICTIONNAIRE DES DONNEES.*

8.3.4.3 Règles de simulation du serveur de démo

Il n'y a pas de règles de simulation particulières mises en place sur le serveur de démo pour ce composant.

9. LE COMPOSANT CHEQUE 3.00

9.1 FONCTIONNEMENT GENERAL DU COMPOSANT CHEQUE

Le composant Cheque permet, à partir d'un numéro de chèque, d'effectuer, via Chèque service, des transactions d'interrogation FNCI (Fichier National des Chèques Irréguliers) ou de garantie chèque.

Chaque transaction chèque doit être identifiée de manière unique. Pour ce faire, les transactions dans la base de données d'Atos Origin sont différenciées grâce à la clé formée des données suivantes :

- **merchant_country** : pays du commerçant
- **merchant_id** : numéro de commerçant
- **cheque_payment_date** : date de création.
- **cheque_transaction_id** : identifiant de la transaction cheque

Ces quatre champs sont donc toujours présents dans les requêtes d'interrogation FNCI et de garantie chèque.

9.2 INSTALLATION ET PARAMETRAGE DU COMPOSANT

9.2.1 Installation

9.2.1.1 Liste des objets livrés

Le composant Cheque est livré sous la forme d'un fichier *composant_cheque_300.tar* contenant les fichiers suivants :

Fichier *Version.txt*	Fichier précisant l'environnement dans lequel le composant a été compilé et testé.
Répertoire *cheque/param*	
certif.fr.011223344553333	Certificat de la boutique de démonstration
pathfile	Fichier des informations proxy et du chemin d'accès au certificat
Répertoire *server/components/service*	
cheque.jar	archive du composant Cheque

9.2.1.2 Copie des fichiers

Vous devez tout d'abord décompresser le fichier *composant_cheque_300.tar*.

Pour un OS de type Unix :

Utiliser la commande suivante : tar –xvf *composant_cheque_300.tar*

Pour Windows :

Utiliser un logiciel de décompression qui accepte les fichiers d'extension « .tar ».

Après avoir décompressé le fichier *composant_cheque_300.tar*, vous devez copier le répertoire *cheque* et tout ce qu'il contient dans le répertoire principal de l'API Server. Le répertoire principal de l'API Server contient également les répertoires associés aux autres composants que vous utilisez, ainsi que le répertoire *server*.

Vous devez également copier le fichier *cheque.jar* dans le répertoire *server/components/service* du répertoire principal de l'API Server.

9.2.2 Configuration

9.2.2.1 Les paramètres du proxy

Si vous utilisez un proxy pour vous connecter à Internet, vous devez le paramétrer dans le fichier *pathfile* copié précédemment dans le répertoire *cheque/param*. Ce paramétrage permettra au composant Cheque de contacter le serveur Office d'Atos Origin via l'Internet (cf. étape 3 de la Figure 1).

Si vous n'utilisez pas de proxy pour vous connecter à Internet, vous ne devez rien indiquer dans les champs suivants :

PROXY_HOST : adresse IP ou nom du proxy. PROXY_PORT : le numéro de port du proxy.

En cas de doute, votre administrateur réseau pourra vous indiquer les valeurs à renseigner.

Exemple :
Si vous passez par le proxy d'adresse IP 111.11.11.11 et de port 7878 vous devez paramétrer les champs :

PROXY_HOST!111.11.11.11! PROXY_PORT!7878!

9.2.2.2 Configuration du chemin vers les fichiers certificats

Le chemin vers les fichiers certificats est localisé dans le fichier *pathfile* copié précédemment dans le répertoire *cheque/param*. Ce chemin doit être renseigné dans le champ F_CERTIFICATE et est unique pour tous les certificats du composant Cheque. Tous les fichiers certificats que vous utiliserez avec le composant Cheque devront donc être localisés dans un même répertoire.

Remarque

Les extensions du fichier certificat ne doivent en aucun cas être indiquées dans le champ F_CERTIFICATE. En effet, ces extensions correspondent aux paramètres **merchant_country** et **merchant_id** transmis dans les requêtes associées aux transactions chèque. Ils seront donc ajoutés par le composant au paramètre F_CERTIFICATE pour obtenir le chemin complet du certificat souhaité.

Exemple Windows :

Si le fichier *certif.fr.011223344553333* est localisé dans le répertoire *C:\API_Server\cheque\param*, vous devez renseigner le champ F_CERTIFICATE comme suit :

```
F_CERTIFICATE!C:\API_Server\cheque\param\certif!
```

Exemple autres OS :

Si le fichier *certif.fr.011223344553333* est localisé dans le répertoire */home/API_Server/cheque/param/*, vous devez renseigner le champ F_CERTIFICATE comme suit :

```
F_CERTIFICATE!/home/API_Server/cheque/param/certif!
```

9.2.2.3 Configuration du chemin vers le fichier pathfile

Vous devez paramétrer dans le fichier *config.xml* le chemin vers le fichier *pathfile* du composant Cheque. Le fichier *config.xml* est localisé dans le répertoire *server/config/* de l'API Server.

Vous trouverez ci-dessous le paragraphe concerné dans le fichier *config.xml* :

```
<components>
<pathfile id="example" path="chemin_du_fichier_pathfile" />
</components>
```

Vous devez ajouter la ligne suivante entre les balises <components> et </components> :

```
<pathfile id="cheque" path="chemin_absolu_vers_le_fichier_pathfile" />
```

avec :

id : l'identifiant du composant (ici cheque).

path : le chemin absolu vers le fichier *pathfile*.

Remarque

Le nom indiqué dans le champ « id » doit être identique au nom de l'archive sans l'extension « .jar » (cf. paragraphe 9.2.1.1)

Exemple Windows :

Si le fichier *pathfile* est localisé dans le répertoire *C:\API_Server\cheque\param*, vous devez renseigner la ligne comme suit :

```
<pathfile id="cheque" path="C:\API_Server\cheque\param\pathfile" />
```

Exemple autres OS :

Si le fichier *pathfile* est localisé dans le répertoire */home/API_Server/cheque/param*, vous devez renseigner la ligne comme suit :

```
<pathfile id="cheque" path="/home/API_Server/cheque/param/pathfile" />
```

9.2.2.4 Configuration des traces

Vous devez paramétrer dans le fichier *config.xml* le chemin des fichiers de traces du composant Cheque. Le fichier *config.xml* est localisé dans le répertoire *server/config/* de l'API Server.

Vous trouverez ci-dessous le paragraphe concerné dans le fichier *config.xml* :

```
<watchdog>
        <pollingTimer>6000</pollingTimer>
        <survPort>7183</survPort>
        <trace level="0" sizeLimit="1000" unit="Line" path="C:\chemin_racine\server\trace"
prefix="APIServer" />
<!--    <alternateTrace id="example" path="C:\chemin_racine\server\trace" prefix="example" />
        -->
        <alternateTrace id="" path="./" prefix="" />
</watchdog>
```

Vous devez ajouter la ligne suivante au-dessus du tag « </watchdog> » (la ligne avec l'identifiant example est en commentaire).

```
<alternateTrace id="cheque" path="chemin_absolu_vers_le_répertoire_traces" prefix="cheque" />
```

avec :

id : l'identifiant du composant (ici cheque).

path : le chemin absolu vers le répertoire des traces. Vous pouvez utiliser pour vos traces le répertoire *server/traces* du répertoire principal de l'application ou tout autre répertoire de votre choix.

prefix : préfixe du fichier de traces du composant. Dans l'exemple ci-dessus, le nom du composant est utilisé, mais tout autre préfixe est également valable.

Remarque

- le nom indiqué dans le champ « id » doit être identique au nom de l'archive sans l'extension « .jar » (cf. paragraphe 9.2.1.1).
- La ligne « <alternateTrace id="" path="./" prefix="" /> » est nécessaire au démarrage de l'API Server dans le cas ou aucun composant n'est installé. Il est donc recommandé de dupliquer cette ligne avant de la modifier.

Exemple Windows :

Si le chemin du répertoire trace est *C:\API_Server\server\traces*, vous devez renseigner la ligne comme suit :

```
<alternateTrace id="cheque" path="C:\API_Server\server\traces" prefix="cheque" />
```

Exemple autres OS :

Si le chemin du répertoire trace est */home/API_Server/server/traces*, vous devez renseigner la ligne comme suit :

```
<alternateTrace id="cheque" path="/home/API_Server/server/traces" prefix="cheque" />
```

9.2.2.5 Adresse IP pour la configuration de votre firewall

Si vous utilisez un firewall pour vous connecter au réseau Internet, vous devez paramétrer au niveau de celui-ci l'adresse IP et le port du serveur Office d'Atos Origin. Pour communiquer avec le serveur Office d'Atos Origin votre firewall doit vous permettre une connexion socket en protocole TCP/IP vers l'adresse IP 193.56.46.110 port 2001.

9.3 DESCRIPTION DETAILLEE DU COMPOSANT

Ce chapitre décrit les opérations disponibles avec le composant Cheque. Pour chacune de ces opérations, des exemples de requête et de réponse au format XML sont fournis, ainsi que les listes des paramètres de la requête et de la réponse avec des liens vers le dictionnaire des données.

Avant de décrire les fonctions disponibles, nous allons préciser les aspects de la sécurisation du transfert des informations entre le composant Cheque et le serveur Office d'Atos Origin.

Remarques

- Afin d'alléger le code des requêtes, les entêtes standard XML ne sont pas nécessaires, mais sont ajoutées par l'API Server pour vérification lors du parsing XML (utilisation d'une DTD). Par contre, lors des réponses de l'API Server, vers l'application cliente, aucune référence à un DTD n'est incluse dans le message. Il est donc important de désactiver la validation lors du parsing, afin de ne pas déclencher des erreurs durant cette phase.
- Pour des raisons de lisibilité, toutes les requêtes décrites dans ce chapitre sont présentées sur plusieurs lignes. Toutefois, lors de l'appel à l'API Server, ces requêtes ne doivent comporter qu'un seul retour chariot en fin de requête.

9.3.1 Sécurisation du transfert des informations

Voir chapitre **La sécurité des paiements en ligne avec SIPS Office Server** de la *PRESENTATION FONCTIONNELLE*.

9.3.2 Requête de vérification vers le composant Cheque : checkcheque

9.3.2.1 Objet

Cette fonction permet de tester :

- la communication entre l'application cliente que vous développez et l'API Server
- le bon fonctionnement de l'API Server
- la configuration du composant Cheque
  - l'accès au fichier *pathfile*
  - l'accès au fichier certificat

9.3.2.2 Paramètres de la requête et de la réponse

Dans le tableau ci-dessous sont décrits tous les champs de la requête de vérification.

nom du champ	Facultatif/ Obligatoire	Utilisation
component	O	nom du composant appelé
name	O	nom de la fonction appelée
merchant_country	O	code pays du commerçant
merchant_id	O	identifiant du commerçant

Dans le tableau ci-dessous sont décrits tous les champs de la réponse de la vérification.

nom du champ	valeur
pathfile	renseigné à OK si le fichier *pathfile* a bien été trouvé
certificate	renseigné à OK si le fichier *certif.<merchant_country>.<merchant_id>* a bien été trouvé
message	renseigné si le fichier *pathfile* ou *certificat* n'a pu être lu

9.3.2.3 Exemple de requête et de réponse XML

Ci-dessous est présenté un exemple de requête XML de vérification. La signification des champs **merchant_country** et **merchant_id** est fournie dans le *DICTIONNAIRE DES DONNEES*. Ces champs correspondent respectivement à la première et la deuxième extension du fichier certificat (exemple : le fichier *certif.fr.011223344553333* correspond à **merchant_country**=fr et **merchant_id**=011223344553333).

```
<service component="cheque" name="checkcheque">
<checkcheque merchant_id="011223344553333" merchant_country="fr" />
</service>
```

Lors de la vérification, le composant Cheque va contrôler l'accès en lecture au fichier *pathfile* paramétré dans le fichier *config.xml*, puis l'accès en lecture au fichier certificat (*certif.<merchant_country>.<merchant_id>*) dont le chemin générique est indiqué dans le fichier *pathfile*. Si ces accès sont corrects, la réponse suivante sera envoyée :

```
<response pathfile="OK" certificate="OK" />
```

Si ces accès ne sont pas corrects, la réponse suivante sera envoyée :

```
<Error message="message d'erreur"/>
```

La liste des messages d'erreur est précisée dans le tableau ci-dessous.

Messages d'erreur	Cause	solution
C:\API_Server\cheque\pam\pathfile (Le chemin d'accès spécifié est introuvable)	le fichier *pathfile* configuré dans le fichier *config.xml* n'est pas accessible en lecture	Vérifier les droits d'accès du fichier *pathfile* et corriger le chemin du fichier *pathfile* si nécessaire (cf. paragraphe 9.2.2.3).
Cannot open certificat. (C:\API_Server\cheque\pam\certif.fr.011223344553333)	le fichier certif.fr.01122334455222 n'est pas accessible en lecture	Vérifier les droits d'accès du fichier certificat *certif.fr.011223344553333* et corriger le chemin génériques des fichiers certificats si nécessaire (cf. paragraphe 9.2.2.2).

9.3.3 Requête d'interrogation cheque : interrogcheque

9.3.3.1 Objet

Cette fonction permet au commerçant disposant des informations chèque de ses clients d'effectuer une requête d'interrogation chèque FNCI (cf. *PRESENTATION FONCTIONNELLE*).

En démonstration (avec le certificat certif.fr.011223344553333), le processus d'interrogation cheque est simulé. Il est donc possible de renseigner n'importe quel numéro de chèque sans aucune conséquence. Pour plus d'information sur le mode simulation référez-vous à l'annexe A.

9.3.3.2 Paramètres de la requête et de la réponse

Dans le tableau ci-dessous sont décrits tous les champs de la requête d'interrogation chèque.

nom du champ	Facultatif/ Obligatoire	Utilisation
component	O	nom du composant appelé
name	O	nom de la fonction appelée
cheque_amount	O	montant du chèque
cheque_cmc7_bank_zone	O	zone bancaire dans la piste CMC7
cheque_cmc7_cheque_number	O	numéro du chèque dans la piste CMC7
cheque_cmc7_internal_zone	O	zone interne dans la piste CMC7
cheque_currency_code	O	code de la monnaie utilisée
cheque_payment_date	O*	date de la requête
cheque_payment_time	O*	heure de la requête
cheque_track_type	O	type de la piste du chèque
cheque_transaction_id	O	numéro de la nouvelle transaction
merchant_country	O	code pays du commerçant
merchant_id	O	identifiant du commerçant
order_id	F	numéro de commande du commerçant
origin	F	permet d'identifier le programme à l'origine de la requête d'interrogation chèque
transmission_date	O*	date de la requête
version	O*	version du composant

** renseigné par le composant*

Dans le tableau ci-dessous sont décrits tous les champs de la réponse de l'interrogation chèque.

nom du champ	origine de la valeur
cheque_fnci_msg	renseigné par Chèque Service
cheque_fnci_response	renseigné par Chèque Service
cheque_guarantee_msg	non renseigné
cheque_guarantee_number	non renseigné
cheque_guarantee_response	non renseigné
response_code	renseigné par le serveur Office
service_type	renseigné par le serveur Office
transaction_certificate	renseigné par le serveur Office (si transaction acceptée)
transaction_date	renseigné par le serveur Office
transaction_time	renseigné par le serveur Office

9.3.3.3 Exemple de requête et de réponse XML

Ci-dessous est présenté un exemple de requête XML d'une interrogation chèque.

```
<service component="cheque" name="interrogcheque">
<interrogcheque origin="Batch"
order_id="A1111"
merchant_id="011223344553333"
merchant_country="fr"
cheque_transaction_id="000004"
cheque_amount="2200"
cheque_currency_code="978"
cheque_track_type="cmc7"
cheque_cmc7_cheque_number="9999999"
cheque_cmc7_internal_zone="999999999999"
cheque_cmc7_bank_zone="999999999918" />
</service>
```

Pour connaître la signification de ces différents champs référez-vous au *DICTIONNAIRE DES DONNEES*.

Ci-dessous est présentée la réponse XML de l'interrogation chèque précédente.

```
<response component="cheque" name="interrogcheque">
<interrogcheque response_code="00"
transaction_time="133127"
transaction_date="20040903"
transaction_certificate="4c0858d53a91"
cheque_service_type="9310"
cheque_fnci_response="00"
cheque_fnci_msg="    VERT    SIGN"
cheque_guarantee_response=""
cheque_guarantee_msg=""
cheque_guarantee_number="" />
</response>
```

Pour connaître la signification de ces différents champs référez-vous au *DICTIONNAIRE DES DONNEES*.

9.3.3.4 Règles de simulation du serveur de démo

Les processus d'interrogation et de garantie cheque sont simulés sur le serveur de démo, . Il est donc possible de renseigner n'importe quelle piste CMC7 sans aucune conséquence.

La seule restriction concernant la piste CMC7 se situe au niveau du dixième caractère du champ **cheque_cmc7_internal_zone**. En effet, ce caractère doit être le chiffre 9 pour une transaction en Euros.

Le code réponse (**cheque_fnci_response**) de la transaction simulée est donné par les deux derniers chiffres de la piste CMC7 (champ **cheque_cmc7_bank_zone**).

Dans le tableau ci-dessous, est rassemblé l'ensemble des codes réponse obtenus en fonction de la piste CMC7 renseignée. Toute autre valeur des deux derniers chiffres conduira à une acceptation (identique à une piste CMC7 se terminant par 18)

Piste CMC7	cheque_fnci_response	Signification
xxxxxxx xxxxxxxxx9xx xxxxxxxxxx10	99	Pas d'interrogation FNCI
xxxxxxx xxxxxxxxx9xx xxxxxxxxxx11	04	Abonné inconnu
xxxxxxx xxxxxxxxx9xx xxxxxxxxxx12	06	Refus. Erreur piste CMC7
xxxxxxx xxxxxxxxx9xx xxxxxxxxxx13	99	Refus. Erreur montant
xxxxxxx xxxxxxxxx9xx xxxxxxxxxx14	99	Problème de paramétrage
xxxxxxx xxxxxxxxx9xx xxxxxxxxxx15	01	Réponse FNCI Orange
xxxxxxx xxxxxxxxx9xx xxxxxxxxxx16	02	Réponse FNCI Rouge
xxxxxxx xxxxxxxxx9xx xxxxxxxxxx17	03	Réponse FNCI Blanc
xxxxxxx xxxxxxxxx9xx xxxxxxxxxx18	00	Réponse FNCI Vert
xxxxxxx xxxxxxxxx9xx xxxxxxxxxx19	99	Service indisponible

cheque_fnci_response renvoyé en fonction du numéro de chèque

9.3.4 Requête de garantie cheque : guaranteecheque

9.3.4.1 Objet

Cette fonction permet au commerçant disposant des informations chèque de ses clients d'effectuer une requête de garantie chèque (cf. *PRESENTATION FONCTIONNELLE*).

En démonstration (avec le certificat certif.fr.011223344553333), le processus de garantie cheque est simulé. Il est donc possible de renseigner n'importe quel numéro de chèque sans aucune conséquence. Pour plus d'information sur le mode simulation référez-vous à l'annexe A.

9.3.4.2 Paramètres de la requête et de la réponse

Dans le tableau ci-dessous sont décrits tous les champs de la requête de garantie chèque.

nom du champ	Facultatif/ Obligatoire	Utilisation
component	O	nom du composant appelé
name	O	nom de la fonction appelée
cheque_amount	O	montant du chèque
cheque_cmc7_bank_zone	O	zone bancaire dans la piste CMC7
cheque_cmc7_cheque_number	O	numéro du chèque dans la piste CMC7
cheque_cmc7_internal_zone	O	zone interne dans la piste CMC7
cheque_currency_code	O	code de la monnaie utilisée
cheque_payment_date	O*	date de la requête
cheque_payment_time	O*	heure de la requête
cheque_track_type	O	type de la piste du chèque
cheque_transaction_id	O	numéro de la nouvelle transaction
merchant_country	O	code pays du commerçant
merchant_id	O	identifiant du commerçant
order_id	F	numéro de commande du commerçant
origin	F	permet d'identifier le programme à l'origine de la requête de garantie chèque
transmission_date	O*	date de la requête
version	O*	version du composant

** renseigné par le composant*

Dans le tableau ci-dessous sont décrits tous les champs de la réponse de la demande de garantie chèque.

nom du champ	origine de la valeur
cheque_fnci_msg	renseigné par Chèque Service
cheque_fnci_response	renseigné par Chèque Service
cheque_guarantee_msg	renseigné par Chèque Service
cheque_guarantee_number	renseigné par Chèque Service
cheque_guarantee_response	renseigné par Chèque Service
response_code	renseigné par le serveur Office
service_type	renseigné par le serveur Office
transaction_certificate	renseigné par le serveur Office (si transaction acceptée)
transaction_date	renseigné par le serveur Office
transaction_time	renseigné par le serveur Office

9.3.4.3 Exemple de requête et de réponse XML

Ci-dessous est présenté un exemple de requête XML d'une demande de garantie chèque.

```
<service component="cheque" name=\"guaranteecheque\">
<guaranteecheque origin="Batch"
order_id="A1111"
merchant_id="011223344553333"
merchant_country="fr"
cheque_transaction_id="000004"
cheque_amount="2200"
cheque_currency_code="978"
cheque_track_type="cmc7"
cheque_cmc7_cheque_number="9999999"
cheque_cmc7_internal_zone="999999999999"
cheque_cmc7_bank_zone="999999999918" />
</service>
```

Pour connaître la signification de ces différents champs référez-vous au *DICTIONNAIRE DES DONNEES*.

Ci-dessous est présentée la réponse XML de la demande de garantie chèque précédente.

```
<response component="cheque" name="guaranteecheque">
<guaranteecheque response_code="00"
transaction_time="133055"
transaction_date="20040906"
transaction_certificate="294ffb0ea950"
cheque_service_type="9110"
cheque_fnci_response="00"
cheque_fnci_msg="              SIGN"
cheque_guarantee_response="000"
cheque_guarantee_msg="REP. FNCI:VERT"
cheque_guarantee_number="000000" />
</response>
```

Pour connaître la signification de ces différents champs référez-vous au *DICTIONNAIRE DES DONNEES*.

9.3.4.4 Règles de simulation du serveur de démo

Mêmes règles que pour la fonction interrogcheque.

9.3.5 Absence de réponse et messages d'erreur

Si vous ne recevez pas de réponse, vous devez vérifier les points suivants :

- L'API Server est-il bien démarré ? (cf. chapitre **Lancement du serveur** du *GUIDE D'INSTALLATION*)
- Est-ce qu'un firewall est-présent entre le client et l'API Server ? Peut-il empêcher l'accès au serveur ?
- Est-ce que l'adresse de la machine cliente est paramétrée dans les listes d'accès de l'API Server ? (cf. paragraphe **Configuration des listes d'accès** du GUIDE D'INSTALLATION)
- L'adresse IP de l'API Server que vous utilisez pour votre connexion socket est-elle correcte ?
- Le port de l'API Server que vous utilisez pour votre connexion socket est-il identique à celui configuré dans le fichier config.xml (cf. paragraphe **Configuration des paramètres de service** du GUIDE D'INSTALLATION)
- Avez-vous enregistré le composant Cheque (cf. paragraphe **La commande « ENREGISTRER UN NOUVEAU COMPOSANT »** du GUIDE D'ADMINISTRATION)

Si vous obtenez un message d'erreur tel que celui présenté ci-dessous :

```
<Error message=" message d'erreur "/>
```

Cela signifie que la requête a bien été reçue par l'API Server, mais elle comporte une erreur. Vous trouverez dans le tableau ci-dessous les principaux messages renvoyés par le composant ainsi que leur origine.

Exemple de messages	Cause	solution
(Le chemin d'accès spécifié est introuvable)	il y a une erreur au niveau du paramètre « id=″cheque″ » du sous-attribut pathfile dans le fichier *config.xml*	Corriger la configuration en vous référant au paragraphe 9.2.2.3.
Element "service" does not allow "toto" here.	toto n'est pas un nom d'opération correct dans la requête XML.	Corriger le nom de l'opération en vous référant aux exemples des paragraphes 9.3.2 à 9.3.4.
Attribute "toto" is not declared for element "checkcheque".	Vous utilisez un nom d'attribut inconnu dans la requête XML.	Vérifier le nom des attributs en vous référant aux exemples des paragraphes 9.3.2 à 9.3.4.
Root element type is "service", but was declared to be "command".	Vous utilisez le port de commande de l'API Server au lieu du port de service	Utiliser le port de service configuré dans le fichier *config.xml* pour votre connexion socket à l'API Server (cf. (cf. GUIDE D'ADMINISTRATION).
Autres messages		Contacter le Centre d'Assistance Technique

9.4 PROPOSITION POUR RESOUDRE VOS CONTRAINTES METIER

Ce chapitre propose des solutions fonctionnelles aux principaux problèmes rencontrés par les commerçants lors de l'exploitation en production du composant Chèque.

9.4.1 Problème de connexion avec Sips Office Server ou Chèque Service

En cas de problème temporaire sur le serveur Office d'Atos Origin le champ **response_code** comportera un code 90 tandis que le champ **cheque_fnci_response** ne sera pas renseigné. De même, si le serveur Chèque Service rencontre temporairement un problème, le champ **response_code** comportera un code 00 tandis que le champ **cheque_fnci_response** sera renseigné avec la valeur 99.

Le commerçant risque alors de refuser une vente alors qu'un essai ultérieur peut conduire à une acceptation de la transaction. Dans ce cas, il est préférable que le commerçant indique à son client qu'il sera contacté ultérieurement pour la suite de sa commande. Le commerçant peut ensuite tenter une nouvelle interrogation ou garantie en utilisant la piste cmc7 fournie par son acheteur.

9.5 DESCRIPTION DES TRACES

Suite à la réception d'une requête, le composant Cheque écrit dans les fichiers de traces configurés au paragraphe 9.2.2.4.

9.5.1 Si aucune erreur n'est survenue

Le composant inscrit dans les traces le nom de la fonction, la clé définissant la transaction créée (**merchant_id**, **merchant_country**, **cheque_transaction_id** et **cheque_payment_date**), le code réponse du serveur Office (**response_code**) et la réponse XML renvoyée au client de l'API Server.

Un exemple de traces pour une interrogation chèque est présenté ci-dessous.

```
252:13:30:34-0-(2)-ChequeWrapper : fonction : interrogcheque
252:13:30:34-0-(2)-ChequeWrapper : merchant_id=011223344553333, merchant_country=fr
252:13:30:35-0-(2)-ChequeWrapper : cheque_transaction_id=000004, cheque_payment_date=20040730
252:13:30:35-0-(2)-ChequeWrapper : response_code=00
252:13:30:35-0-(2)-ChequeWrapper : response : <response component="cheque" name="interrogcheque"><interrogcheque response_code="00" transaction_time="132443" transaction_date="20040908" transaction_certificate="25b601f5b7b2" cheque_service_type="9310" cheque_fnci_response="00" cheque_fnci_msg=" VERT SIGN" cheque_guarantee_response="" cheque_guarantee_msg="" cheque_guarantee_number="" /></response>
252:13:30:35-0-(2)-ChequeWrapper : ----------
```

Dans le cas de la fonction de vérification checkcheque, les champs **cheque_transaction_id** et **cheque_payment_date** contiennent respectivement l'heure et la date de la requête de vérification.

Un exemple de traces de vérification est présenté ci-dessous.

```
252:13:32:11-0-(3)-ChequeWrapper : fonction : checkcheque
252:13:32:11-0-(3)-ChequeWrapper : merchant_id=011223344553333, merchant_country=fr
252:13:32:11-0-(3)-ChequeWrapper : cheque_transaction_id=133211, cheque_payment_date=20040908
252:13:32:11-0-(3)-ChequeWrapper : response : <response  pathfile="OK"  certificate="OK" />
252:13:32:11-0-(3)-ChequeWrapper : ----------
```

9.5.2 En cas d'erreur

Le composant inscrit dans les traces le nom de la fonction appelée suivi du message d'erreur complet.

Un exemple de traces en cas d'erreur lors d'une interrogation chèque est présenté ci-dessous.

```
252:13:34:51-2-(4)-ChequeWrapper : fonction : interrogcheque
252:13:34:51-2-(4)-ChequeWrapper : Error message: C:\apiserver\cheque\param\pathfile (Le fichier spécifié est introuvable)
252:13:34:51-2-(4)-ChequeWrapper : ----------
```