

TrustyServer



Manuel de développement

Référence : CSSI/HLS/TRUSTY/FR/11/0004

Etat APPROUVE

Version 3.10 du 3/11/2017



la confiance sur le long terme

CS SYSTEMES D'INFORMATION

Sommaire



1. Généralités	5
1.1 Objet	5
1.2 Documents de référence	5
1.3 Terminologie	7
1.3.1 Définitions	7
1.3.2 Abréviations	7
2. Présentation générale	8
2.1 TrustyServer	8
2.2 Déploiement de TrustyServer	9
2.3 Interface	10
3. Interface « Web Services »	11
3.1 Format des requêtes	11
3.1.1 « TransactionKey » de type « TransactionKey »	12
3.1.2 Data	12
3.1.3 ReturnX509Info	13
3.2 Format des réponses	13
3.2.1 « Result » de type « Result »	14
3.2.2 « X509Info » de type « X509Info »	15
3.2.3 FinalContent	16
3.3 Fonction « Sign »	16
3.3.1 Requête SignatureRequest	16
3.3.1.1 “Options” de type “SignatureOptions”	17
3.3.2 Réponse SignatureResponse	17
3.3.2.1 “Informations” de type “SignerInfo”	19
3.3.2.2 “SignerIdentifier”	20
3.3.2.3 TimeStamp	21
3.4 Fonction « Verify »	22
3.4.1 Requête VerifyRequest	22
3.4.1.1 Options de type “VerifyOptions”	23
3.4.1.2 « SignedObjects » de type « SignatureData »	24
3.4.2 Réponse « VerifyResponse »	24
3.4.2.1 “Reports” de type “Report”	25

3.4.2.2	« VerificationDetails » de type « VerificationDetails »	26
3.4.2.3	« EnhancementDetails » de type « EnhancementDetails »	27
3.5	Fonction « reVerify »	28
3.5.1	Requête ReVerifyRequest.....	28
3.5.2	Réponse « VerifyResponse »	30
3.6	Fonction « Enhance »	31
3.6.1	Requête « EnhanceRequest »	31
3.6.1.1	« Options » de type « EnhanceOptions »	32
3.6.2	Réponse « EnhanceResponse »	32
4.	Exemples.....	34
4.1	Création de signature	34
4.1.1	Requête	34
4.1.2	Réponse	34
4.2	Vérification de signature	35
4.2.1	Requête	35
4.2.2	Réponse	36
5.	Mise en œuvre de SoapUI	37
5.1	Créer le projet.....	37
5.2	Ajouter la définition du .wsdl	37
5.3	Déclarer une suite de tests.....	39
5.4	Déclarer un cas de test	39
5.5	Déclarer l'appel HTTPS	39
5.6	Signature.....	41
5.6.1	Déclarer une étape de test	41
5.6.2	Lancer la requête.....	43
5.6.3	Récupérer la réponse	43
5.7	Vérification de signature	44
5.7.1	Déclarer une étape de test	44
5.7.2	Lancer la requête.....	45
5.7.3	Récupérer la réponse	46
6.	Codes d'erreur.....	47

Table des illustrations



Liste des figures

Figure 1 - Infrastructure de sécurité	9
Figure 2 – Données de la requête	11
Figure 3 - Clé de transaction	12
Figure 4 - Données de la requête	12
Figure 5 – Retour informations sur les certificats	13
Figure 6 - Réponse de base	13
Figure 7 - Informations sur le statut de la transaction	14
Figure 8 - Description du certificat	15
Figure 9- Le document final après traitement	16
Figure 10 – Requête de signature	17
Figure 11 – Options de la requête de signature	17
Figure 12 – Réponse aux requêtes de signature	18
Figure 13 - Informations sur le signataire	19
Figure 14 - Identification du signataire	20
Figure 15 - Jeton d'horodatage	21
Figure 16 – Requête de vérification	22
Figure 17 - Options de vérification	23
Figure 18 – Options de vérification d'une signature détachée	24
Figure 19 - Réponse à une requête de vérification	25
Figure 20 – Rapport du traitement de chaque vérification de signature	25
Figure 21 - Détails de la vérification d'une signature	26
Figure 22 - Détails de l'enrichissement d'une signature	27
Figure 23 – Vérification différée	28
Figure 24 – Requête de vérification	29
Figure 25 – Requête d'enrichissement de signature	31
Figure 26 – Options d'enrichissement de signature	32
Figure 27 - Réponse à une requête d'enrichissement	33

1. GENERALITES

1.1 Objet

Ce document constitue le manuel de développement et d'utilisation de l'interface Web Service **TrustyServer**.

1.2 Documents de référence

[TrustyBox]	Manuel d'Administration TrustyBox CSSI/SECU/TRUSTY/140003
[TrustyServer]	Manuel d'administration TrustyServer CSSI/HLS/TRUSTY/FR/8/0104
[HTTP]	RFC2068 : Hypertext Transfer Protocol -- HTTP/1.1 T. Berners-Lee (MIT/LCS), R. Fielding (UC Irvine), H. Frystyk (MIT/LCS), J. Gettys, J. Mogul(DEC) January 1997
[SOAP]	Simple Object Access protocol Version 1.1 Don Box (DevelopMentor), David Ehnebuske (IBM), Gopal Kakyvaya, Andrew Layman, Henrik Frystyk nielsen, Satish Thatte (Microsoft), Noah Mendelsohn (Lotus Development Corp.), Dave Winer (UserLand Software, Inc.) W3C Note 08 May 2000
[WSDL]	Web Service Description Language Version 2.0 SOAP 1.1 Binding A. S. Vedamuthu (WebMethods) W3C Working Draft 10 May 2005
[XML-DSIG]	XML-Signature Syntax and Processing RFC 3275 D. Eastlake 3rd (Motorola), J. Reagle (W3C), D. Solo (Citigroup) mars 2002
[XML-SCHEMA]	XML Schema Part 0 : Primer Second Edition XML Schema Part 1 : Structures Second Edition XML Schema Part 2 : Datatypes Second Edition W3C Recommendation 28 October 2004 Part 0 : D. C. Fallside (IBM), P. Walmsley Part 1 : H.S Thompson (University of Edinbrough), D. Beech (Oracle Corporation), M. Maloney (Commerce One), N. Mendelsohn (Lotus Development Corporation)

	Part 2 : P. V. Biron (Kaiser permanente for Health Level Seven), A. Malhotra (Microsoft formerly IBM) 28 October 2004
[1999/93/EC]	Directive 1999/93/EC du Parlement Européen et du Conseil sur le cadre communautaire des signatures électroniques 13 décembre 1999
[PKI-TSP]	Internet X.509 Public Key Infrastructure Time-Stamp Protocol (TSP) RFC3161 août 2001
[OCSP]	Online Certificate Status Protocol RFC 2560 juin 1999
[XAdES-1]	ETSI TS 101 733 Formats de signatures électroniques Décembre 2003
[XAdES-2]	DGME_SDAE_XAdESV1.0_v0.97 Format de signature électronique XAdES de l'Administration Octobre 2006

1.3 Terminologie

1.3.1 Définitions

Terme	Définition
Politique	<p>De manière générale, une politique de signature est un document permettant de décrire le contexte et les processus de réalisation et de validation d'une signature électronique.</p> <p>Dans TrustyServer, une politique de signature est un fichier de configuration comprenant un ensemble de paramètres techniques et métier, permettant d'implémenter le document de politique de signature de l'organisation.</p>

1.3.2 Abréviations

Acronyme	Signification
DN	Distinguished Name, identifiant d'une personne ou d'une entité
HTTP	HyperText Transport Protocol, protocole d'échange de données utilisé par Internet
IHM	Interface Homme Machine
LCR/CRL	Liste de Certificats Révoqués, liste de révocation contenant les numéros de série de certificats d'utilisateurs révoqués
LDAP	Lightweight Directory Access Protocol, protocole d'accès à des annuaires publiques
MTC	Most Trusted Certificate, certificat de confiance (n'est pas vérifié car considéré fiable)
RFC	Request For Comments, ensemble de standards publiques
SOAP	Simple Object Access Protocol
TCP	Transmission Control Protocol, protocole réseau de niveau transport utilisé sur Internet
WSDL	Web Service Description Language
X.509	Format standard de certificat de sécurité
XML	eXtended Markup Language
XSD	XML Schema Definition

2. PRESENTATION GENERALE

2.1 TrustyServer

TrustyServer est une solution centralisée de confiance offrant à des applications clientes les services de :

- ✓ signature électronique de documents
- ✓ validation de signature électronique de documents

TrustyServer est capable de s'adapter à de nombreux contextes de déploiement grâce aux possibilités étendues de configuration de chacun des services. L'administrateur peut configurer finement les règles à appliquer lors de la signature et de la vérification de signature.

TrustyServer se déploie aisément grâce à son intégration dans la solution **TrustyBox** de **CS**. **TrustyServer** dispose ainsi d'une interface graphique d'administration en client léger, puissante et ergonomique. L'administrateur dispose d'une vision globale et précise du système et peut entièrement configurer les paramètres techniques et métiers de la solution. L'interface d'administration permet de piloter la génération, la mise en production et la destruction des clés cryptographiques. Elle gère automatiquement la réplication des configurations métier et des clés lors de l'ajout de serveurs supplémentaires pour augmenter les capacités du système.

Les politiques de signature gérées par **TrustyServer** sont des ensembles de règles liées à la création et la validation de la signature électronique. Cet ensemble de règles définit par exemple :

- ✓ règles de création comprenant les algorithmes et le format de signature à créer,
- ✓ règles d'utilisation et d'accès aux magasins de certificats de confiance et de listes de révocation
- ✓ règles de validation de la signature comprenant les champs à vérifier ou enrichir.

TrustyServer supporte la signature et la vérification de tout type de document :

- ✓ Signatures XML-DSIG et XAdES
- ✓ Signature PDF et PAdES de documents PDF
- ✓ Signature CML et CAdES

Au-delà des simples signatures et vérification de signature, **TrustyServer** peut, sur demande du client, horodater les signatures présentes via **TrustyTime** (solution d'horodatage de **CS**) à l'issue du processus de signature et de validation de signature.

TrustyServer propose également des fonctions de ré-horodatage via **TrustyTime** dans le cadre de la préservation de l'intégrité des données et la maintenance de leur caractère « non répudiable » sur le long terme. La signature peut être enrichie avec les chaînes de certification et les listes de révocation actuelles, en vue d'une conservation du document sur le long terme.

2.2 Déploiement de TrustyServer

TrustyServer est déployé dans le cadre **TrustyBox** (se référer au guide d'administration et de déploiement de TrustyBox [TrustyBox]).

La figure suivante montre l'intégration de **TrustyServer** dans l'infrastructure de sécurité cible :

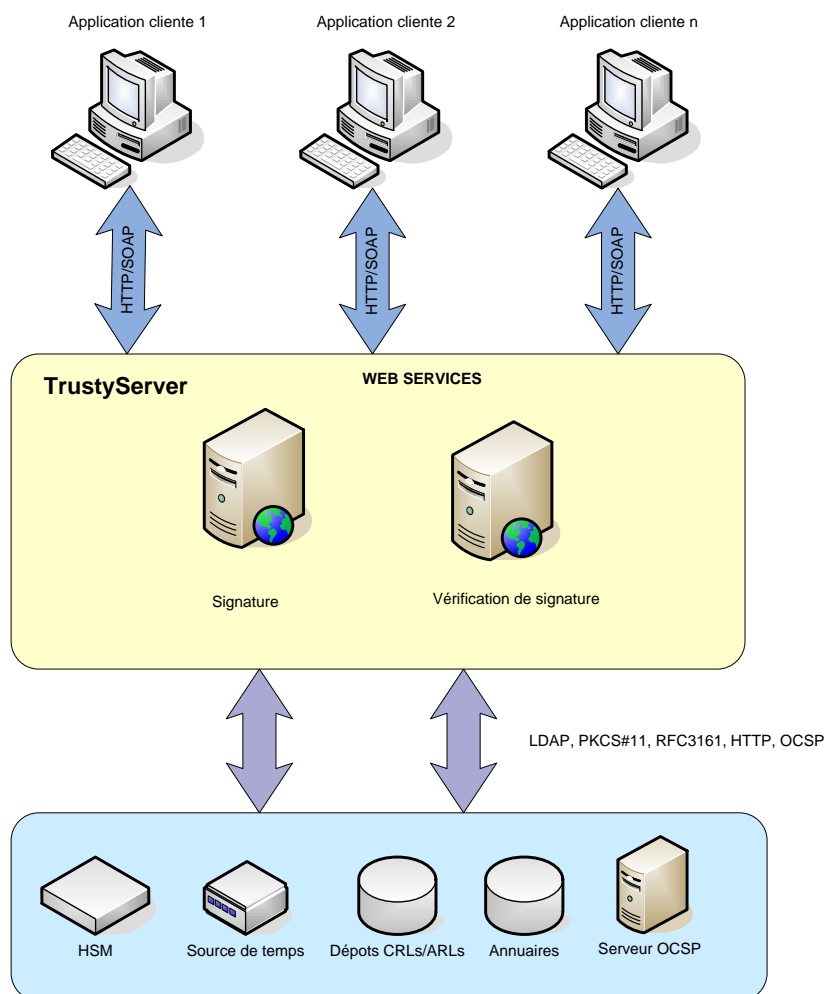


Figure 1 - Infrastructure de sécurité

L'administration de **TrustyServer** se fait au travers de **TrustyBox**. Les opérations d'administration permettent de définir en particulier :

- ✓ Les politiques de signatures applicables par TrustyServer (voir [TrustyServer])
- ✓ Les bi-clés de signature de TrustyServer
- ✓ Les applications clientes autorisées à envoyer des requêtes à TrustyServer

Le déploiement et la configuration correcte de TrustyServer sont des préalables obligatoires pour débiter l'envoi de requêtes à TrustyServer. Ces opérations sont décrites dans le guide d'administration de TrustyServer [TrustyServer].

Selon le service souhaité l'application cliente formate une requête et la transmet au serveur **TrustyServer**. Ce dernier est en mesure, en fonction de l'identification de la politique indiquée dans la requête de procéder aux traitements demandés.

2.3 Interface

Le composant « Web Services » met à disposition d'application cliente des fonctions via une interface « Web Services » se basant sur le protocole HTTPS/SOAP en mode message.

Les services proposés par **TrustyServer** sont :

- ✓ **Sign** : elle permet de signer un document dans l'un des formats suivants : CMS, CAdES, XML-DSIG, XAdES, PDF et PAdES. Son fonctionnement est détaillé au § 3.3
- ✓ **Verify** : elle permet de valider un document signé dans l'un des formats suivants : CMS, CAdES, XML-DSIG, XAdES, PDF et PAdES. Son fonctionnement est détaillé au § 3.4
- ✓ **Enhance** : elle permet de valider un document signé au format XAdES et de transformer les signatures vérifiées en un format XAdES-T ou XAdES-A. Son fonctionnement est détaillé au § 0

Ces fonctions s'appuient sur l'ensemble des composants de **TrustyServer** pour rendre le service souhaité. Elles sont décrites en WSDL (WEB Services Description Language), et leurs paramètres dans des schémas XSD.

TrustyServer vérifie le certificat d'authentification de l'application cliente qui doit être déclarée et configurée au niveau de **TrustyBox**.

3. INTERFACE « WEB SERVICES »

Les fonctions proposées par **TrustyServer** sont accessibles en Web Services. Ces Web Services sont :

- ✓ Décrits en WSDL ;
- ✓ Disponibles via le protocole SOAP en mode message par l'intermédiaire du protocole HTTPS ;
- ✓ Disponibles via le protocole SOAP en mode message avec attachement par l'intermédiaire du protocole HTTPS.

3.1 Format des requêtes

Toutes les requêtes étendent `RequestBaseType`, cet objet regroupe les éléments communs à tout type de requête.

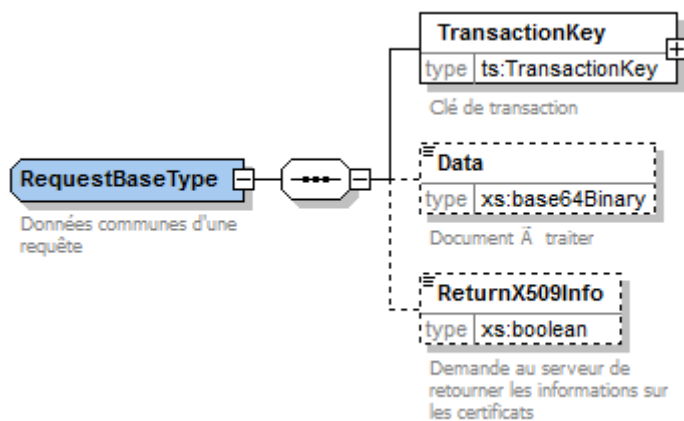


Figure 2 – Données de la requête

3.1.1 « TransactionKey » de type « TransactionKey »

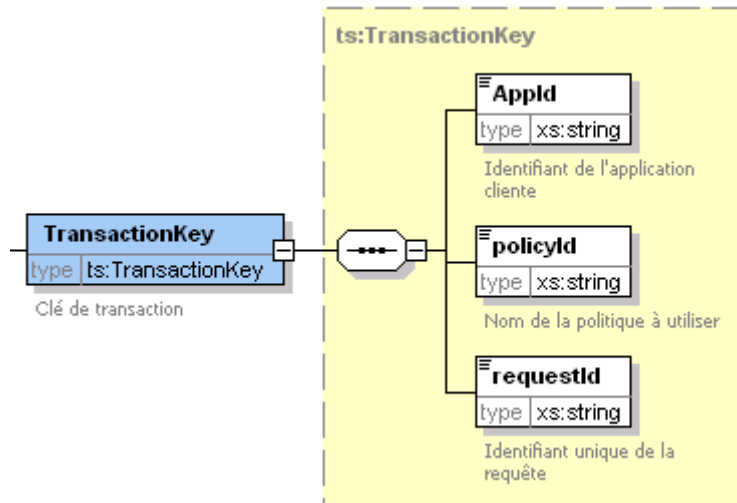


Figure 3 - Clé de transaction

- ✓ AppId : l'identifiant de l'application cliente (nom du client au niveau TrustyBox)
- ✓ policyId : le nom de la politique demandée (défini au niveau TrustyBox)
- ✓ requestId : Identifiant unique de la requête (généré par l'utilisateur du service, le système n'effectue aucun contrôle sur cette valeur qui est retournée dans la réponse)

La donnée « policyId » est l'élément discriminant permettant de déterminer la politique à mettre en œuvre pour traiter la requête. Si la politique demandée n'est pas déclarée dans le serveur ou si l'application cliente ne dispose pas des droits nécessaires pour l'invoquer, la requête ne sera pas traitée.

Une application cliente peut être autorisée pour plusieurs politiques.

3.1.2 Data

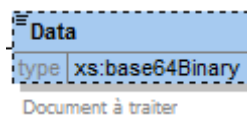


Figure 4 - Données de la requête

- ✓ Data : document à traiter (format binaire)

3.1.3 ReturnX509Info

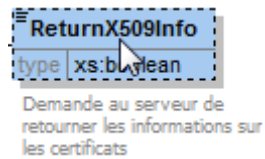


Figure 5 – Retour informations sur les certificats

- ✓ ReturnX509 : indique si le serveur doit retourner les informations sur les certificats (Vrai ou faux) (uniquement pour le format XAdES)

3.2 Format des réponses

Toutes les réponses étendent `ResponseBaseType`, cet objet regroupe les éléments communs à tout type de réponse.

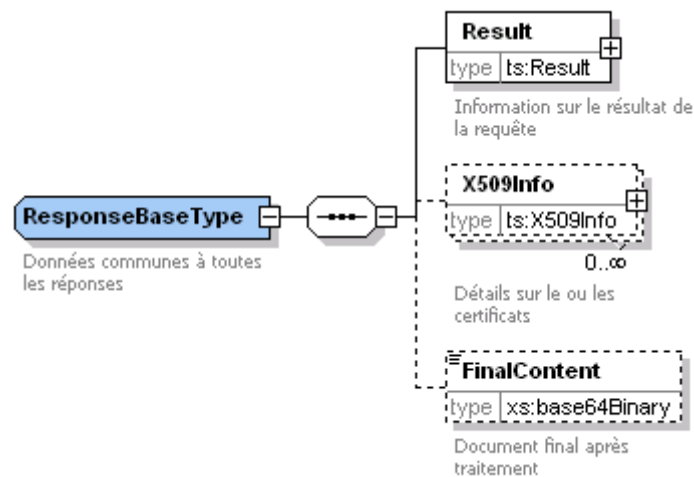


Figure 6 - Réponse de base

3.2.1 « Result » de type « Result »

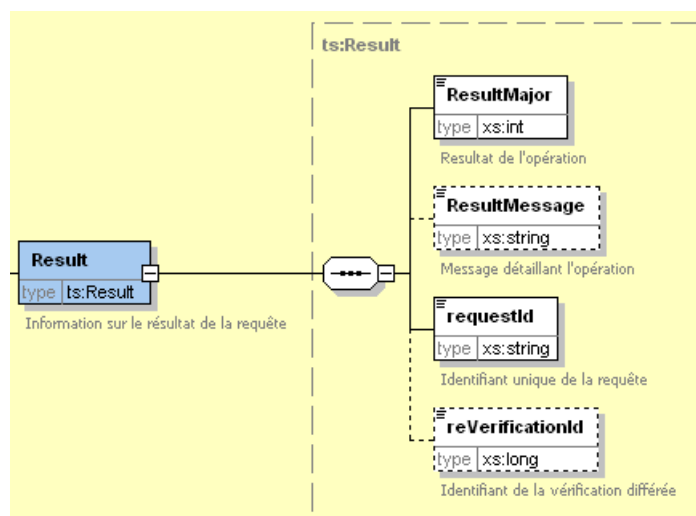


Figure 7 - Informations sur le statut de la transaction

- ✓ **ResultMajor** : entier indiquant le résultat de l'opération, les valeurs possibles sont :
 - 0 : Traitement de la requête OK
 - 1 : Traitement de la requête KO
 - 2 : Traitement de la requête non autorisée
 - 3 : Ressources manquantes
 - 4 : Erreur interne
- ✓ **ResultMessage** : Message détaillant le résultat de l'opération (se référer au § 4. pour le détail des erreurs)
- ✓ **RequestId** : Identifiant unique de la requête (champ présent dans la requête)
- ✓ **reVerificationId** : Ce champ est utilisé dans le cas d'un appel à la fonction `verify`. Dans le cas où la politique de vérification de signature précise qu'une vérification différée est mise en œuvre, cet identifiant est retourné par le système pour être utilisé lors d'un appel à la fonction `reVerify`.

Dans ce cas TrustyServer Sign est configuré au niveau de la politique pour effectuer un second contrôle après un délai d'attente défini par l'administrateur de la solution. Une fois ce délai d'attente respecté, la fonction `reVerify` peut être appelée avec en paramètre la valeur de `reVerificationId` qui a été conservé.

3.2.2 « X509Info » de type « X509Info »

Les détails sur le ou les certificats de signature. Il n'est présent que si le champ ReturnX509Info de la requête est positionné à vrai (true).

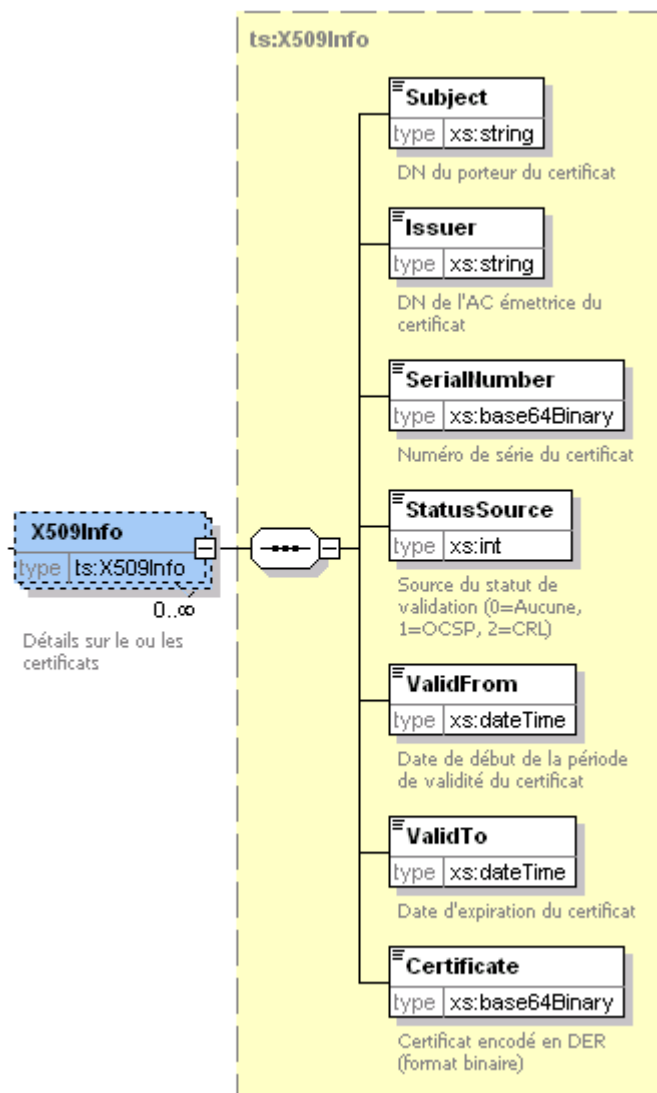


Figure 8 - Description du certificat

La description du ou des certificats de l'application cliente compte les champs suivants :

- ✓ **Subject** : le DN du porteur du certificat
- ✓ **Issuer** : le DN de l'Autorité de Certification émettrice du certificat
- ✓ **SerialNumber** : le numéro de série du certificat
- ✓ **StatusSource** : la source permettant de valider le certificat. Il peut être validé à partir d'une liste CRL ou d'une requête OCSP.
- ✓ **ValidFrom** : date de début de validité du certificat
- ✓ **ValidTo** : date de fin de validité du certificat

- ✓ Certificate : le certificat encodé en DER

3.2.3 FinalContent

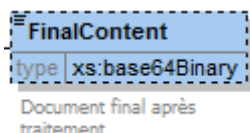


Figure 9- Le document final après traitement

- ✓ FinalContent : le document final, résultat de la requête

3.3 Fonction « Sign »

La fonction `sign` permet la signature d'une donnée par TrustyServer. La clé privée impliquée dans une opération de signature est stockée au niveau du magasin sécurisé (HSM/magasin logiciel).

Cette signature s'effectue selon la politique de signature fournie par l'application cliente, permettant par exemple la validation du certificat signataire et/ou pour les formats CMS et XADES d'effectuer une requête pour récupérer un jeton d'horodatage.

3.3.1 Requête SignatureRequest

Le format de la requête de signature est une variation du format générique (Cf.3.1). L'élément `Options` est spécifique à la requête.

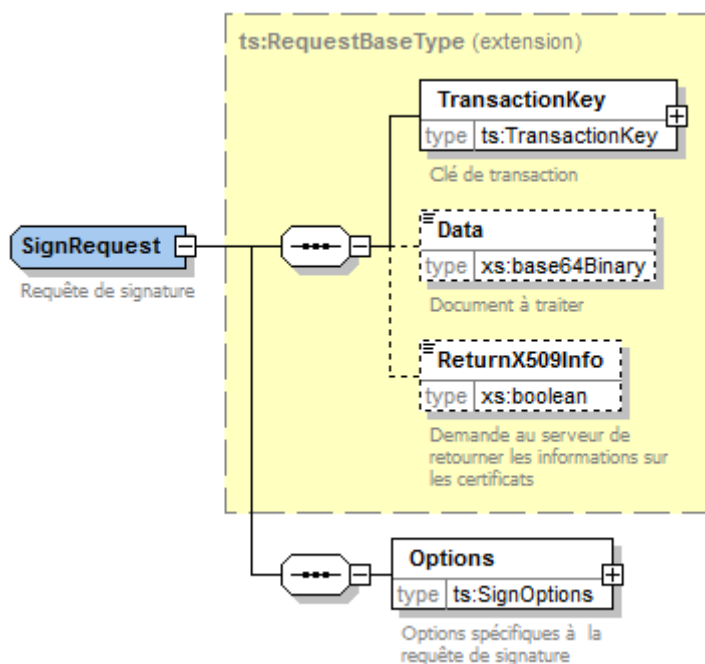


Figure 10 – Requête de signature

3.3.1.1 “Options” de type “SignatureOptions”

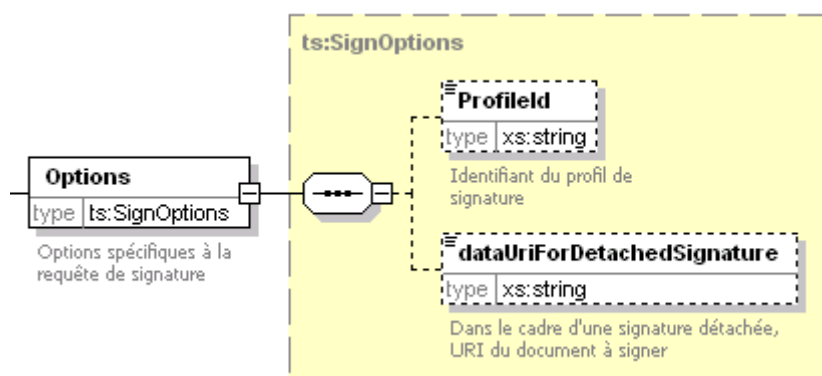


Figure 11 – Options de la requête de signature

L'élément SignatureOptions contient l'élément suivant :

- ✓ **ProfileId** : profil du signataire. Il s'agit d'une information qui est utilisée par TrustyServer pour identifier les traitements à réaliser par rapport à la politique demandée. Ce champ est optionnel, un profil par défaut est défini au niveau de la politique.
- ✓ **dataUriForDetachedSignature** : Cet identifiant, permettant d'identifier les données signées, est ajouté à la signature dans le cas d'une signature détachée. Ce champ est optionnel.

3.3.2 Réponse SignatureResponse

Le format de la réponse à une requête de signature est une extension du format générique (Cf. 3.2). L'élément SignatureInfo est ajouté à la réponse.

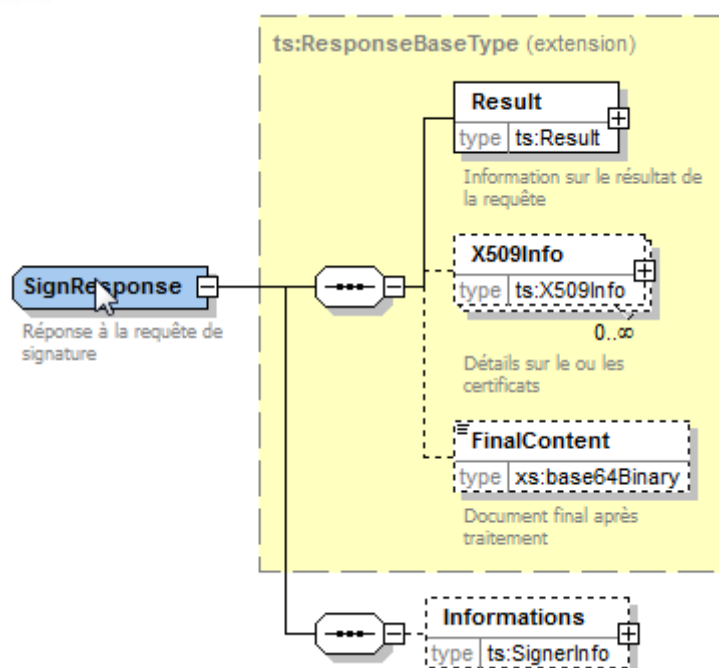


Figure 12 – Réponse aux requêtes de signature

3.3.2.1 “Informations” de type “SignerInfo”

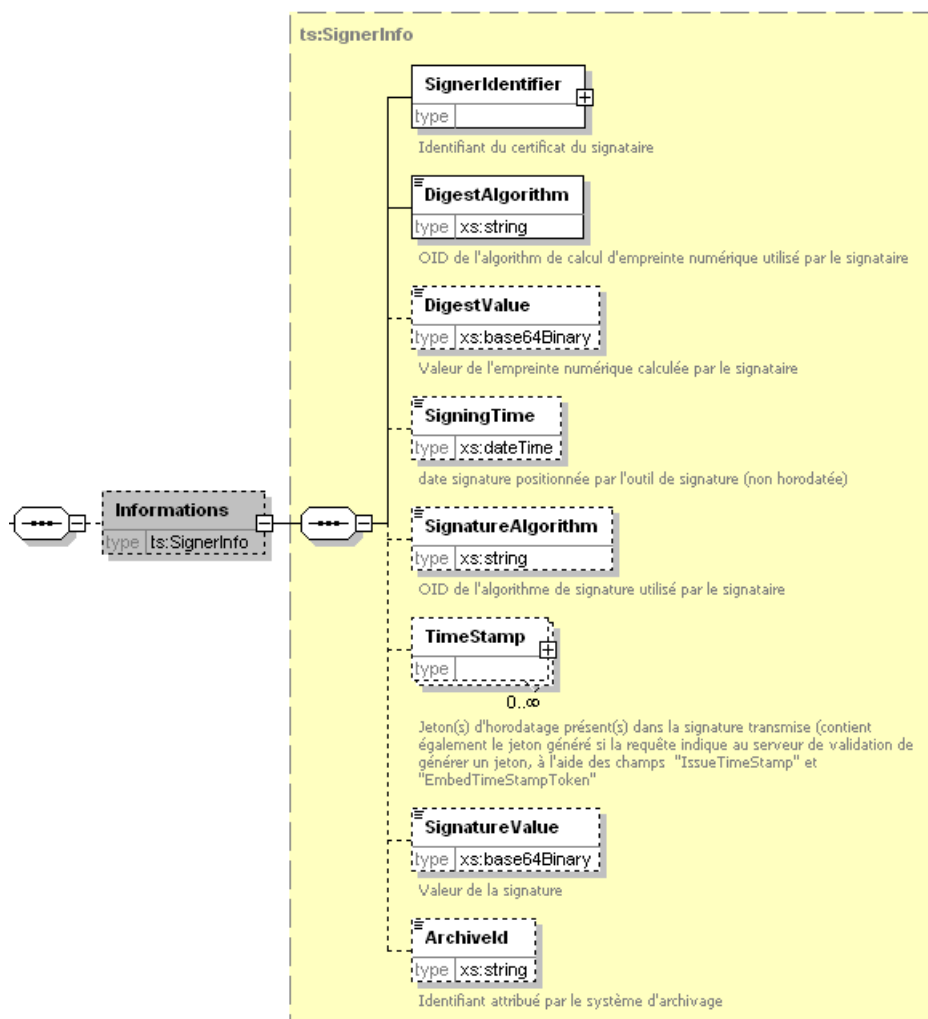


Figure 13 - Informations sur le signataire

L'élément « Informations » de type « SignerInfo » contient les détails sur le signataire.

Il se compose de :

- ✓ SignerIdentifier: permet d'identifier le certificat du signataire
- ✓ DigestAlgorithm: identification (OID) de l'algorithme de hachage utilisé par le signataire pour calculer l'empreinte numérique.
 - Valeurs possibles :
 - 1.3.14.3.2.26 (SHA-1)
 - 2.16.840.1.101.3.4.2.4 (SHA-224)
 - 2.16.840.1.101.3.4.2.1 (SHA-256)
 - 2.16.840.1.101.3.4.2.2 (SHA-384)
 - 2.16.840.1.101.3.4.2.3 (SHA-512)
- ✓ DigestValue: empreinte du document sur lequel porte la signature, calculée par le signataire.

- ✓ **SigningTime** : date de signature. Cette date est présente dans les attributs signés du document signé. Ce n'est pas l'heure d'horodatage mais celle fournie par l'outil de signature. Cette information est présente pour des signatures de type CAdES, XAdES et PDF.
- ✓ **TimeStamp** : informations sur les jetons d'horodatage présents dans la signature
- ✓ **SignatureAlgorithm** : identification (OID) de l'algorithme de signature utilisé par le signataire.
- ✓ **SignatureValue** : valeur de la signature.
- ✓ **ArchiveId** : Identifiant attribué par le service d'archivage TrustyArchive lorsque le document signé est archivé. L'archivage est activé par l'administrateur, au niveau de la politique de signature.

3.3.2.2 "SignerIdentifier"

Le champ « SignerIdentifier » permet à TrustyServer d'identifier le certificat signataire.

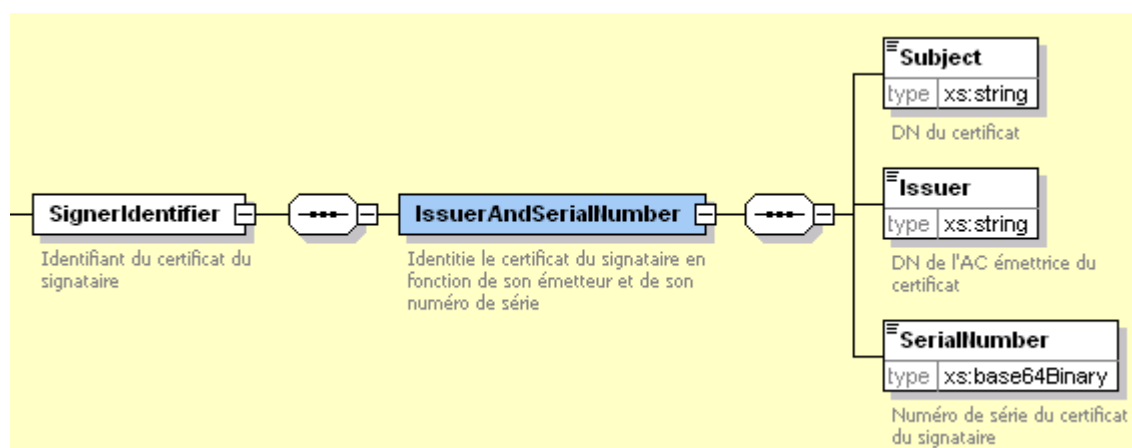


Figure 14 - Identification du signataire

Il se compose de :

- ✓ **IssuerAndSerialNumber** : les identifiants du certificat auprès de l'AC émettrice
 - **Subject** : DN du certificat
 - **Issuer** : DN de l'AC
 - **SerialNumber** : numéro de série du certificat du signataire

3.3.2.3 TimeStamp

Une signature peut éventuellement contenir des jetons d'horodatage.

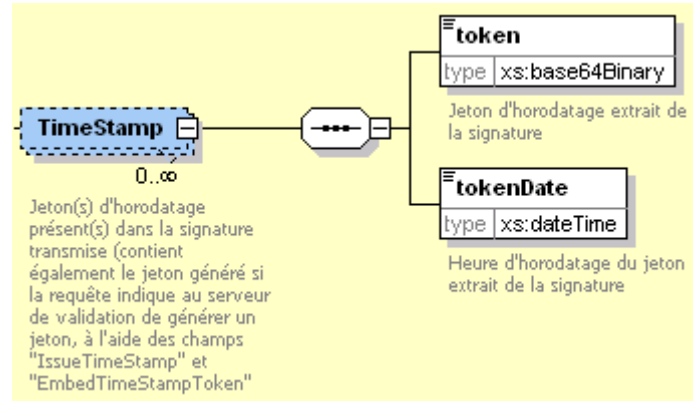


Figure 15 - Jeton d'horodatage

- ✓ token : jeton d'horodatage extrait de la signature
- ✓ tokenDate : heure du jeton d'horodatage extrait

Un jeton est stocké sous forme du jeton en lui-même, tel qu'extrait de la signature, et de son heure d'horodatage

3.4 Fonction « Verify »

La fonction `verify` permet de :

- ✓ Valider selon une politique donnée une signature électronique d'un document (pouvant en comporter plusieurs dans le cas des signatures XML enveloppées, une seule dans le cas des signatures CMS et PDF) ;
- ✓ Valider un document selon une politique en vérifiant toutes les signatures du document ;
- ✓ Fournir les informations sur les signataires vérifiés.

3.4.1 Requête VerifyRequest

Le format de la requête de validation est une extension du format générique (Cf.3.1). L'élément `Options` est spécifique à la requête.

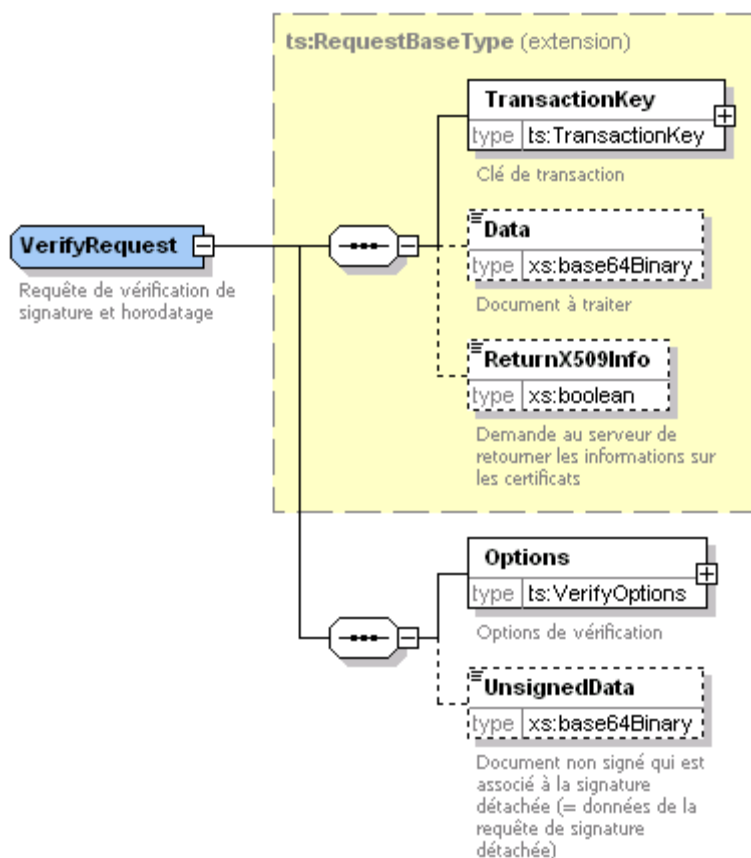


Figure 16 – Requête de vérification

- ✓ UnsignedData : Ce champ permet fournir le document non signé utilisé pour générer la signature détachée afin de pouvoir vérifier la signature.

3.4.1.1 Options de type "VerifyOptions"

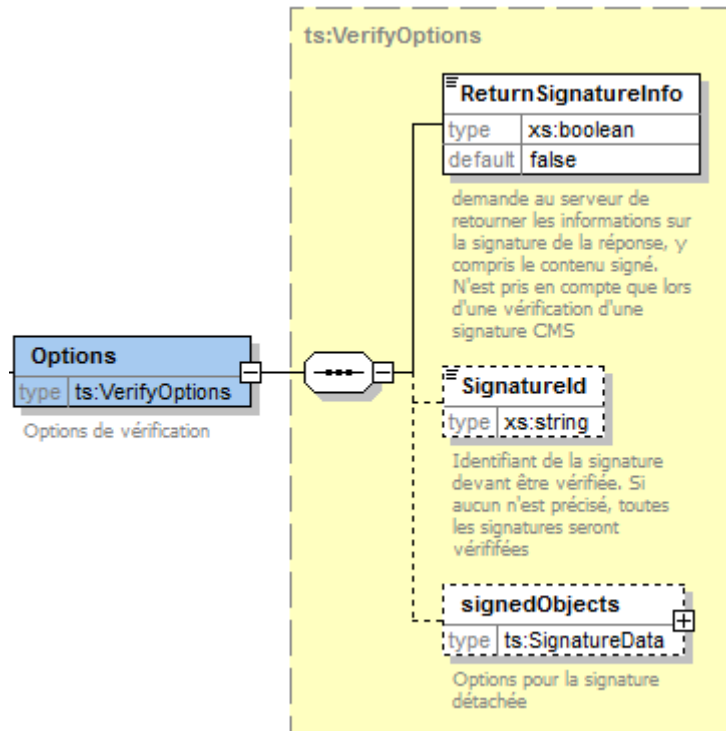


Figure 17 - Options de vérification

L'élément *VerifyOptions* représente les options de vérification.

- ✓ **ReturnSignatureInfo** : indique si la réponse doit contenir des informations détaillées sur la signature (Cf.0) Le champ n'est pris en compte que dans le cas de la vérification d'une signature CMS.
- ✓ **SignatureId** : identifiant de la signature au sein du document devant être vérifiée. Si l'identifiant est non présent toutes les signatures seront vérifiées
- ✓ **signedObjects** : Options de vérification d'une signature détachée. Ces paramètres sont obsolètes et seront supprimés dans les versions ultérieures.

3.4.1.2 « SignedObjects » de type « SignatureData »

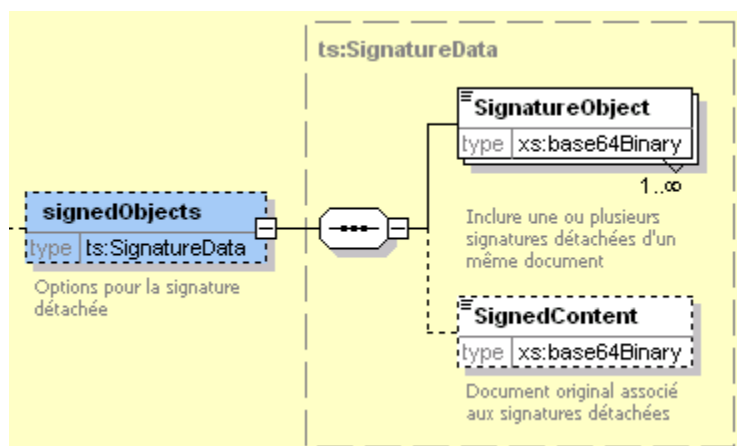


Figure 18 – Options de vérification d'une signature détachée

L'élément SignedObjects permet de renseigner les options de vérification d'une signature détachée :

- ✓ SignatureObject : Une ou plusieurs signatures détachées correspondant au même document.
- ✓ SignedContent : Document original associé aux signatures détachées.

3.4.2 Réponse « VerifyResponse »

Le format de la réponse à une requête de validation est une extension du format générique (Cf.3.2). L'élément Report est spécifique à la réponse.

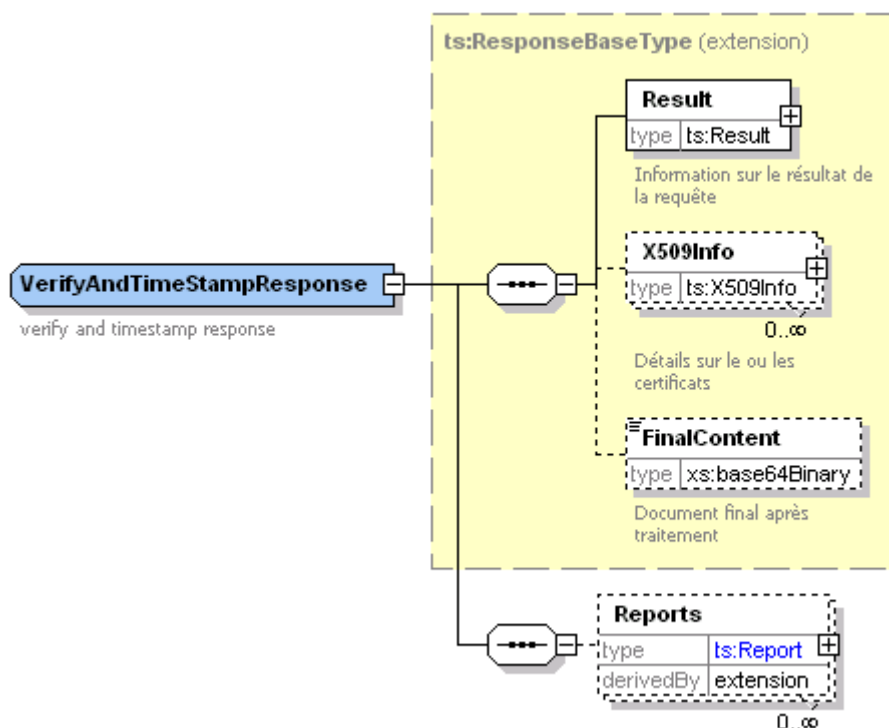


Figure 19 - Réponse à une requête de vérification

3.4.2.1 "Reports" de type "Report"

Le champ « Reports » est composé d'un élément rapport générique à toutes les requêtes de vérification et d'un élément décrivant le signataire (cf. 3.3.2.1).

Ce rapport porte sur une signature dont l'identifiant est précisé par le champ « signatureId ».

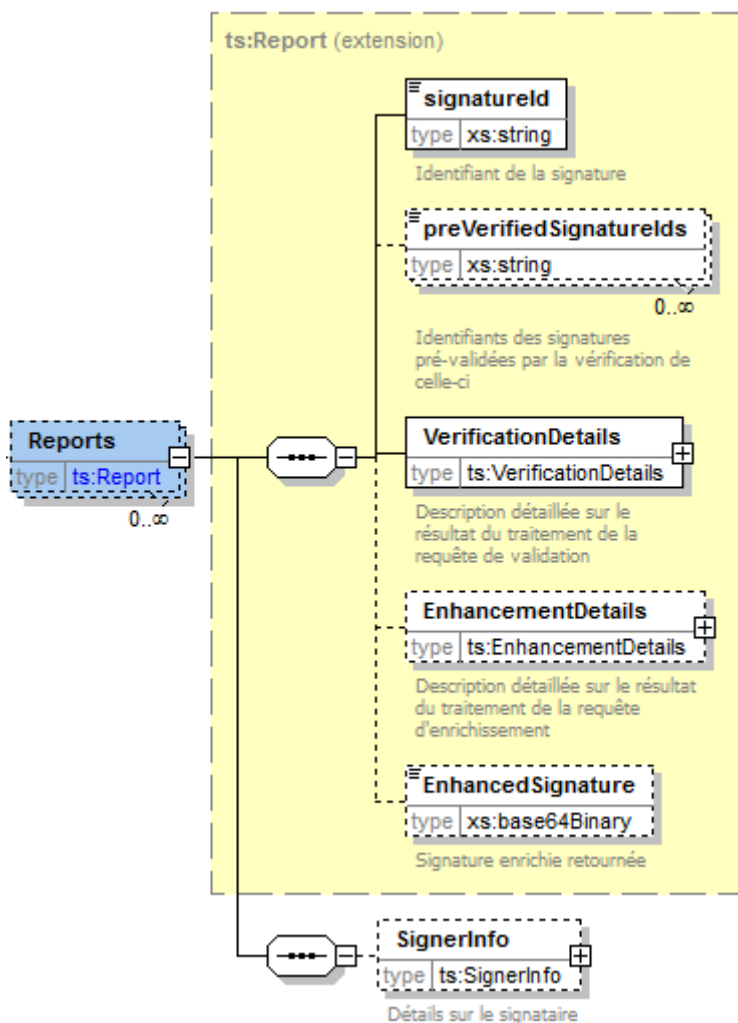


Figure 20 – Rapport du traitement de chaque vérification de signature

- ✓ signatureId : l'identifiant de la signature
- ✓ preVerifiedSignatureIds : la liste des identifiants des signatures pré-vérifiées pour celle-ci
- ✓ VerificationDetails : les détails de la vérification de la signature
- ✓ EnhancementDetails : les détails de l'enrichissement de la signature
- ✓ EnhancedSignature : Signature enrichie

3.4.2.2 « VerificationDetails » de type « VerificationDetails »

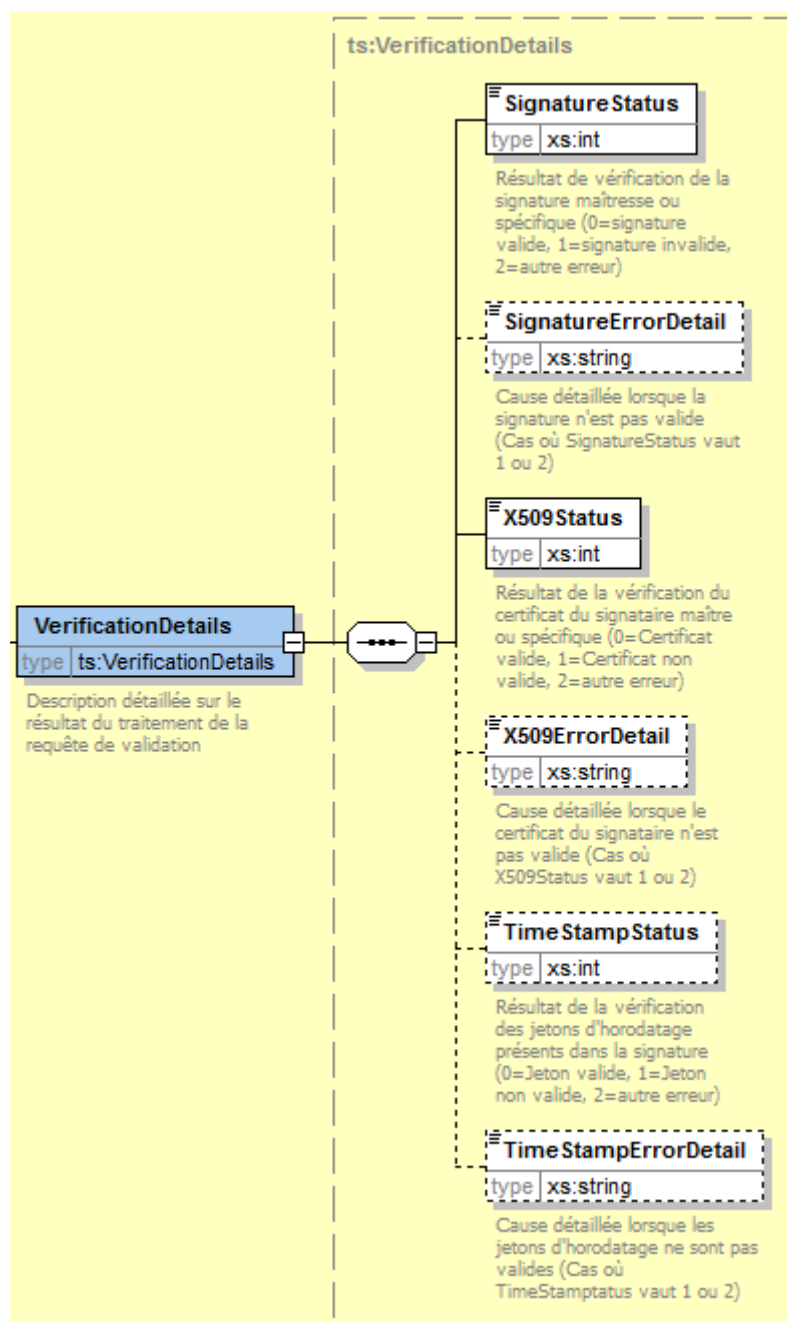


Figure 21 - Détails de la vérification d'une signature

- ✓ SignatureStatus : le statut global de la signature
- ✓ SignatureErrorDetail : le détail de l'erreur si la signature n'est pas valide (hors erreurs décrites ci-après)
- ✓ X509Status : le statut du certificat du signataire
- ✓ X509ErrorDetail : le détail de l'erreur si le certificat n'est pas valide
- ✓ TimeStampStatus : le statut des jetons contenus dans la signature
- ✓ TimeStampErrorDetail : le détail de l'erreur si les jetons ne sont pas valides

3.4.2.3 « EnhancementDetails » de type « EnhancementDetails »

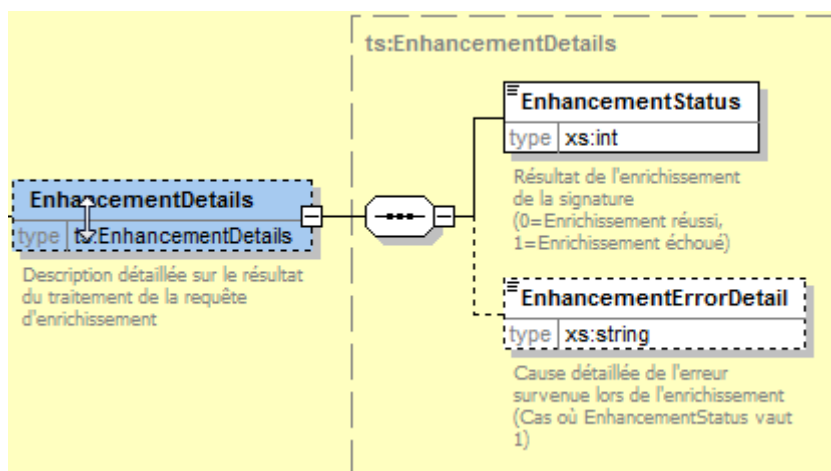


Figure 22 - Détails de l'enrichissement d'une signature

- ✓ EnhancementStatus : le statut de l'enrichissement
- ✓ SignatureErrorDetail : le détail de l'erreur si l'enrichissement a échoué

3.5 Fonction « reVerify »

La fonction `reVerify` permet de :

- ✓ Valider selon une politique donnée une signature électronique d'un document (pouvant en comporter plusieurs dans le cas des signatures XML enveloppées, une seule dans le cas des signatures CMS et PDF) ;
- ✓ Valider un document selon une politique en vérifiant toutes les signatures du document ;
- ✓ Fournir les informations sur les signataires vérifiés.

La fonction `reVerify` est utilisée pour faire suite à un appel à la fonction `verify` dans les conditions d'utilisation décrites ci-dessous :

1. Le client appelle la fonction de vérification de signature `verify()`, la politique référencée dans la requête est configurée pour fonctionner en vérification différée
2. TrustyServer Sign effectue un premier contrôle de vérification et conserve les données d'entrée
3. TrustyServer Sign retourne le résultat de ce premier contrôle, calcule et y intègre l'attribut `reVerificationId`
4. TrustyServer Sign met en place un délai d'attente défini par l'administrateur de la solution avec la politique et exprimé en jours. Ce délai peut donner lieu à une mise à jour des CRLs
5. TrustyServer Sign effectue un second contrôle des données
6. Le client appelle la fonction de vérification de signature `reVerify()` en utilisant l'identifiant `reVerificationId` récupérée à l'étape 3/
7. TrustyServer Sign retrouve le résultat du second contrôle à partir de l'identifiant passé dans la requête
8. TrustyServer Sign retourne le résultat de ce second contrôle

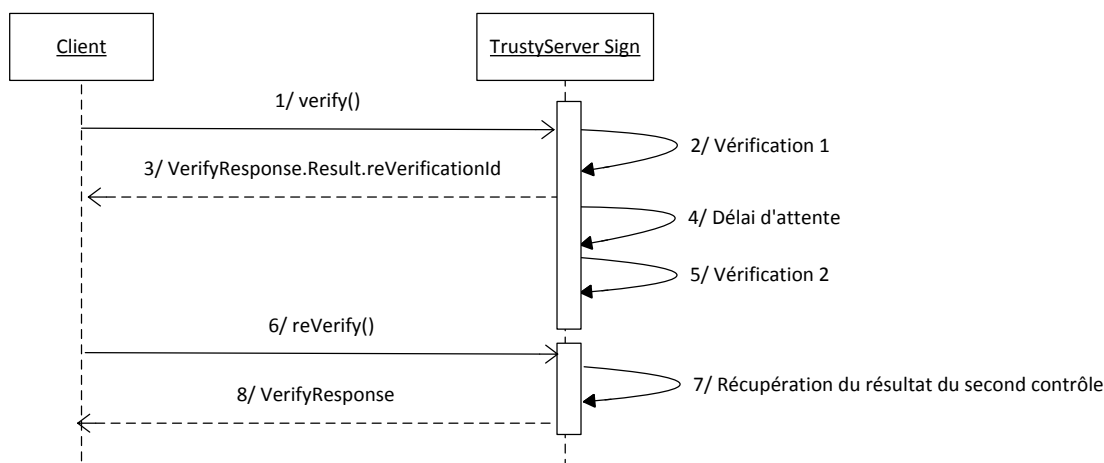


Figure 23 – Vérification différée

3.5.1 Requête ReVerifyRequest

Le format de la requête de validation est une extension du format générique (Cf.3.1). L'élément `Options` est spécifique à la requête.

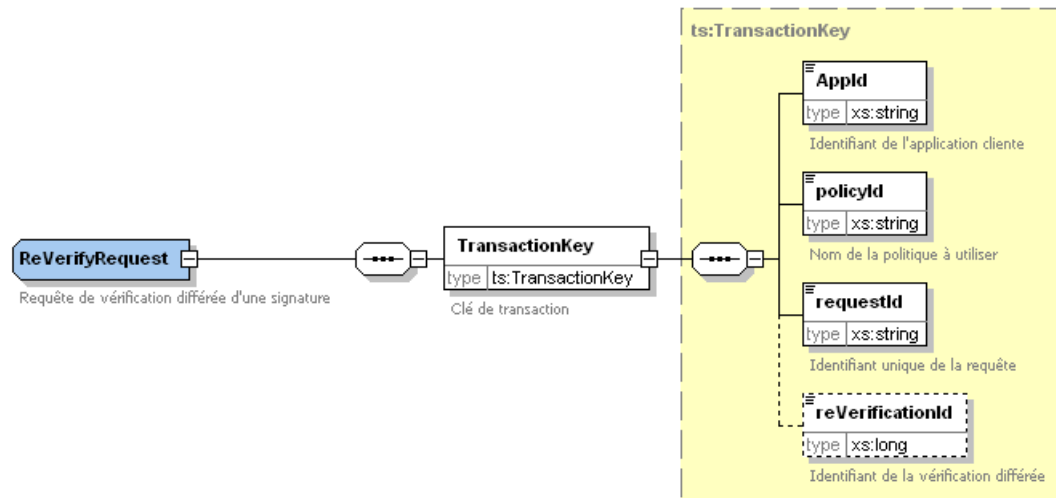


Figure 24 – Requête de vérification

- ✓ AppId : l'identifiant de l'application cliente (nom du client au niveau TrustyBox)
- ✓ policyId : le nom de la politique demandée (défini au niveau TrustyBox)
- ✓ requestId : Identifiant unique de la requête (généré par l'utilisateur du service, le système n'effectue aucun contrôle sur cette valeur qui est retournée dans la réponse)
- ✓ reVerificationId : Identifiant retourné par la fonction `verify` permettant de retrouver le résultat du contrôle différé.

La donnée « `policyId` » est l'élément discriminant permettant de déterminer la politique à mettre en œuvre pour traiter la requête. Si la politique demandée n'est pas déclarée dans le serveur ou si l'application cliente ne dispose pas des droits nécessaires pour l'invoquer, la requête ne sera pas traitée.

Une application cliente peut être autorisée pour plusieurs politiques.

3.5.2 Réponse « VerifyResponse »

La fonction `reVerify` retourne le même message que la fonction `verify`, se référer au § 3.4.2

3.6 Fonction « Enhance »

La fonction `enhance` s'applique à des documents xml contenant des signatures XAdES. Elle permet de :

- ✓ Vérifier une signature particulière pouvant nécessiter des pré-vérifications ou vérifier un document entier en vérifiant toutes ses signatures
- ✓ Vérification de toutes les signatures basées sur la même politique
- ✓ Transformer toutes les signatures XAdES en XAdES-T ou -A

3.6.1 Requête « EnhanceRequest »

Le format de la requête de validation et transformation XAdES est une variation du format générique (Cf. 3.1). L'élément `Options` est spécifique à la requête.

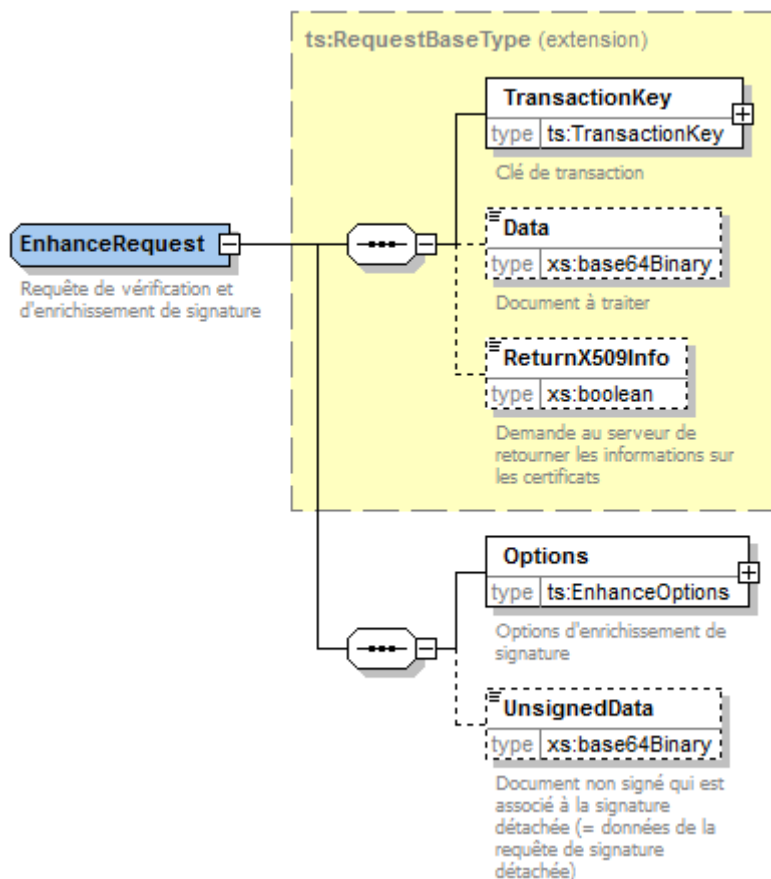


Figure 25 – Requête d'enrichissement de signature

- ✓ **UnsignedData** : Ce champ permet fournir le document non signé utilisé pour générer la signature détachée afin de pouvoir vérifier la signature.

3.6.1.1 « Options » de type « EnhanceOptions »

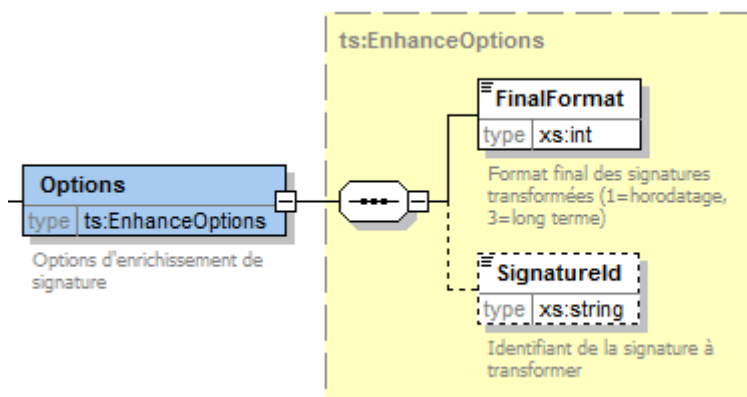


Figure 26 – Options d'enrichissement de signature

L'élément Options contient les éléments suivants :

- ✓ FinalFormat : correspond au format de transformation souhaité dans le cadre d'une transformation (enrichissement) PAdES ou XAdES en prenant en compte les contraintes suivantes :

- | | |
|---------------|---|
| FinalFormat=1 | La signature en entrée est du type XAdES-BES/EPES ou PAdES-BES/EPES
La signature en sortie est enrichie avec un jeton d'horodatage pour être en XAdES-T ou PAdES-T |
| FinalFormat=2 | Inutilisé |
| FinalFormat=3 | La signature en entrée est du type XAdES-T ou PAdES-T
La signature en sortie est enrichie avec la chaîne de certification du certificat de signature et leurs listes de révocation. La signature est alors transformée en XAdES-A ou PAdES-LTV |

- ✓ SignatureId : identifiant de la signature à vérifier et transformer au sein du document

3.6.2 Réponse « EnhanceResponse »

Le format de la réponse à une requête de transformation XAdES est une extension du format générique (Cf.3.3.2). L'élément « Report » est identique à l'élément « Report » (Cf : 3.4.2.1).

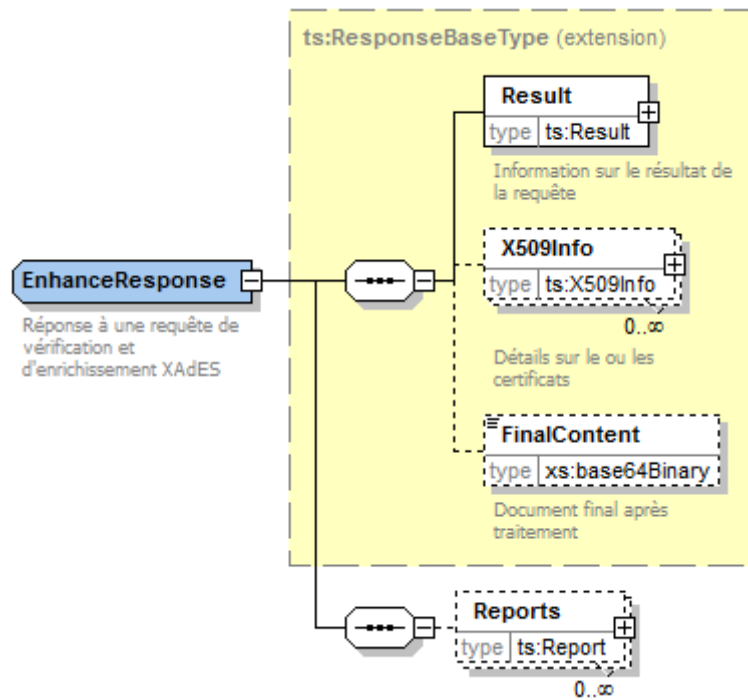


Figure 27 - Réponse à une requête d'enrichissement

4. EXEMPLES

4.1 Création de signature

4.1.1 Requête

Voici un exemple d'appel de TrustyServer pour demander la signature d'un document. Cet appel est construit avec les données suivantes :

- ✓ **Mon_Application** : nom de l'application tel qu'elle est enregistrée comme client dans TrustyBox ;
- ✓ **PolitiqueSignature_RGS** : nom d'une politique de signature déclarée dans TrustyBox ;
- ✓ **REQ-01234** : identifiant arbitraire fixé par l'application pour cette requête ;
- ✓ **CachetValidation** : nom du profil de signature déclaré dans la politique de signature ;
- ✓ «Donnée à signer» : information à signer, dont la conversion en base64 est «RG9ubsOpZSDDoCBzaWduZXI=»

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ts="http://com.cs.server.web.serveur/services">
  <soapenv:Header/>
  <soapenv:Body>
    <ts:SignRequest>
      <TransactionKey>
        <AppId>Mon_Application</AppId>
        <policyId>PolitiqueSignature_RGS</policyId>
        <requestId>REQ-01234</requestId>
      </TransactionKey>
      <Data>RG9ubsOpZSDDoCBzaWduZXI=</Data>
      <Options>
        <ProfileId>CachetValidation</ProfileId>
      </Options>
    </ts:SignRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

4.1.2 Réponse

Voici une réponse qui pourrait être retournée par TrustyServer Sign à la requête précédente :

- ✓ **0** : Résultat indiquant que la signature s'est bien passée ;
- ✓ **REQ-01234** : identifiant de la requête fixée par l'application dans l'appel ;
- ✓ **RG9uw6lIHNPz27DqWU=** : Les données signées par TrustySever Sign au format base64.

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns3:SignResponse xmlns:ns3="http://com.cs.server.web.serveur/services">
      <Result>
        <ResultMajor>0</ResultMajor>
        <requestId>REQ-01234</requestId>
      </Result>
      <FinalContent>RG9uw611IHNpZ27DqWU=</FinalContent>
    </ns3:SignResponse>
  </soap:Body>
</soap:Envelope>
```

4.2 Vérification de signature

4.2.1 Requête

Voici un exemple d'appel de TrustyServer pour demander la vérification de signature d'un document. Cet appel est construit avec les données suivantes :

- ✓ **Mon_Application** : nom de l'application tel qu'elle est enregistrée comme client dans TrustyBox ;
- ✓ **PolitiqueSignature_RGS** : nom d'une politique de signature déclarée dans TrustyBox ;
- ✓ **6789ABCD** : identifiant arbitraire fixé par l'application pour cette requête ;
- ✓ **CachetValidation** : identifiant de la signature à vérifier, correspondant à un nom du profil de signature déclaré dans la politique de signature ;
- ✓ Les données en signer sont dans le fichier « document_signe.xml » et incluses par référence.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:ts="http://com.cs.server.web.serveur/services">
  <soapenv:Header/>
  <soapenv:Body>
    <ts:VerifyRequest>
      <TransactionKey>
        <AppId>Mon_Application</AppId>
        <policyId>PolitiqueSignature_RGS</policyId>
        <requestId>6789ABCD</requestId>
      </TransactionKey>
      <Data><xop:Include href="cid:document_signe.xml"
        xmlns:xop="http://www.w3.org/2004/08/xop/include"/></Data>
      <ReturnX509Info>>false</ReturnX509Info>
      <Options>
        <ReturnSignatureInfo>>false</ReturnSignatureInfo>
        <SignatureId>CachetValidation</SignatureId>
      </Options>
    </ts:VerifyRequest>
```

```
</soapenv:Body>
</soapenv:Envelope>
```

4.2.2 Réponse

Voici une réponse qui pourrait être retournée par TrustyServer Sign à la requête précédente :

- ✓ **0** : Résultat indiquant que la signature s'est bien passée ;
- ✓ **6789ABCD** : identifiant de la requête fixée par l'application dans l'appel ;
- ✓ **CachetValidation** : identifiant de la signature vérifiée ;
- ✓ Les détails de la signature vérifiée :
 - Statut de la signature : 0 (valide)
 - Statut du certificat du signataire : 0 (valide)
 - Statut du jeton d'horodatage : 0 (valide)
- ✓ Pas d'enrichissement de la signature.

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns3:VerifyResponse xmlns:ns2="http://www.w3.org/2000/09/xmldsig#"
      xmlns:ns3="http://com.cs.server.web.serveur/services">
      <Result>
        <ResultMajor>0</ResultMajor>
        <requestId>6789ABCD</requestId>
      </Result>
      <Reports>
        <signatureId>CachetValidation</signatureId>
        <VerificationDetails>
          <SignatureStatus>0</SignatureStatus>
          <X509Status>0</X509Status>
          <TimeStampStatus>0</TimeStampStatus>
        </VerificationDetails>
        <EnhancementDetails>
          <EnhancementStatus>0</EnhancementStatus>
        </EnhancementDetails>
      </Reports>
    </ns3:VerifyResponse>
  </soap:Body>
</soap:Envelope>
```

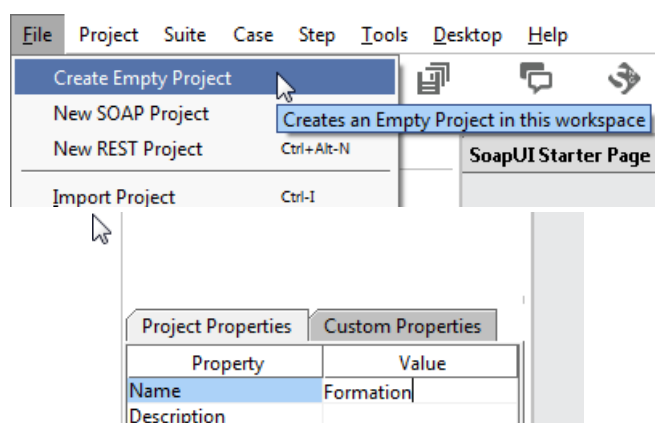

5. MISE EN ŒUVRE DE SOAPUI

SoapUI est un outil gratuit, disponible sur internet qui peut être utilisé pour effectuer des tests d'appels webservice vers un service TrustyServer Sign.

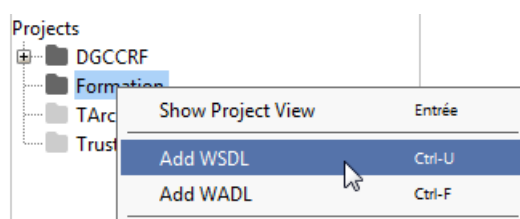
Le chapitre présente la façon de mettre en œuvre SoapUI pour effectuer un test d'appel à la fonction de signature et de vérification de signature.

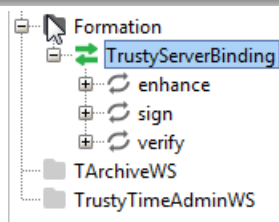
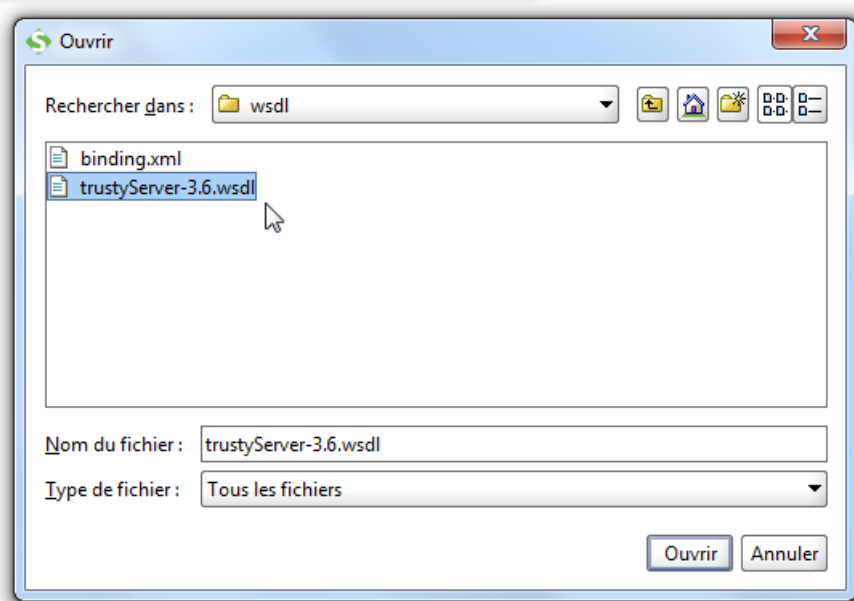
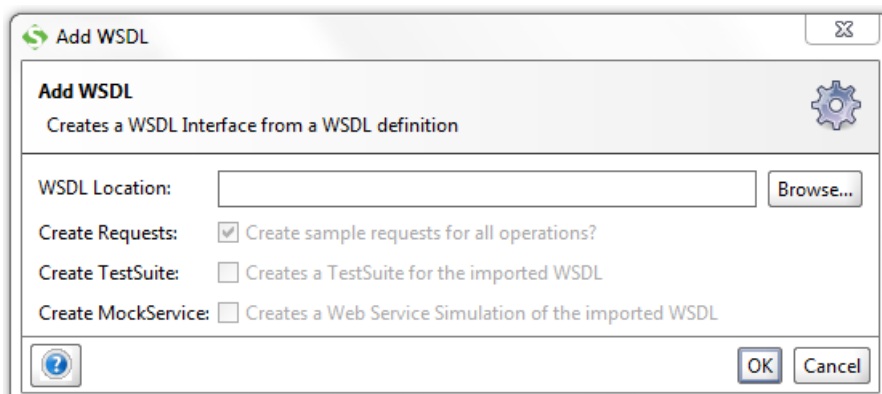


5.1 Créer le projet

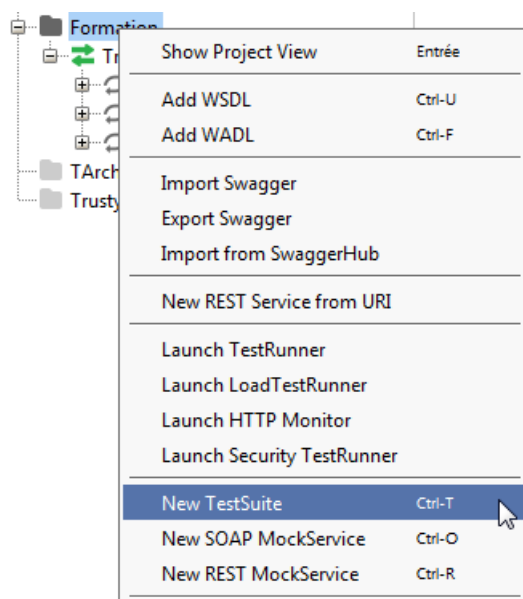


5.2 Ajouter la définition du .wsdl

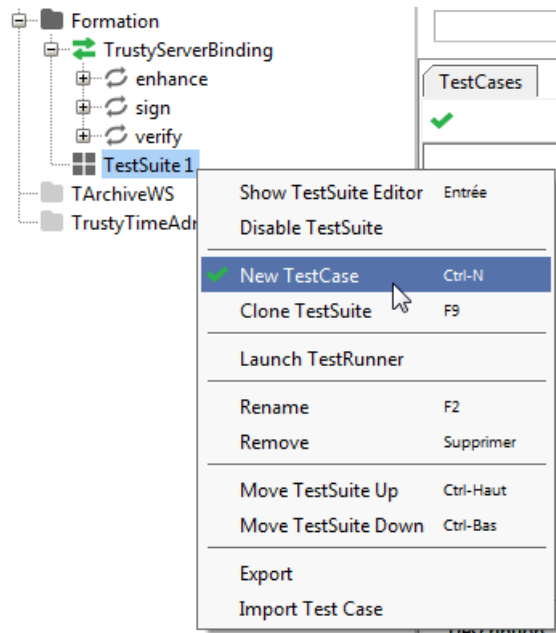




5.3 Déclarer une suite de tests

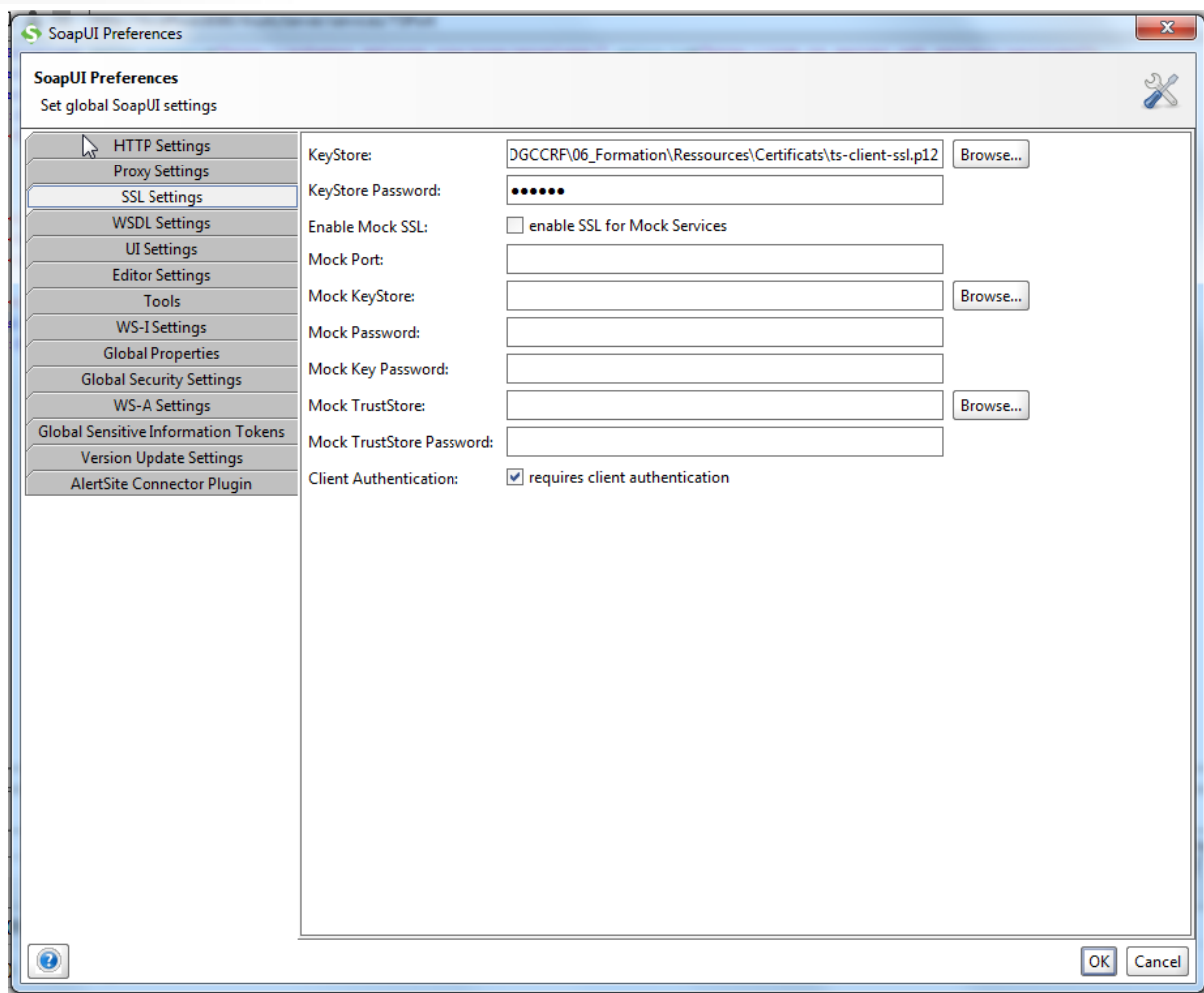


5.4 Déclarer un cas de test



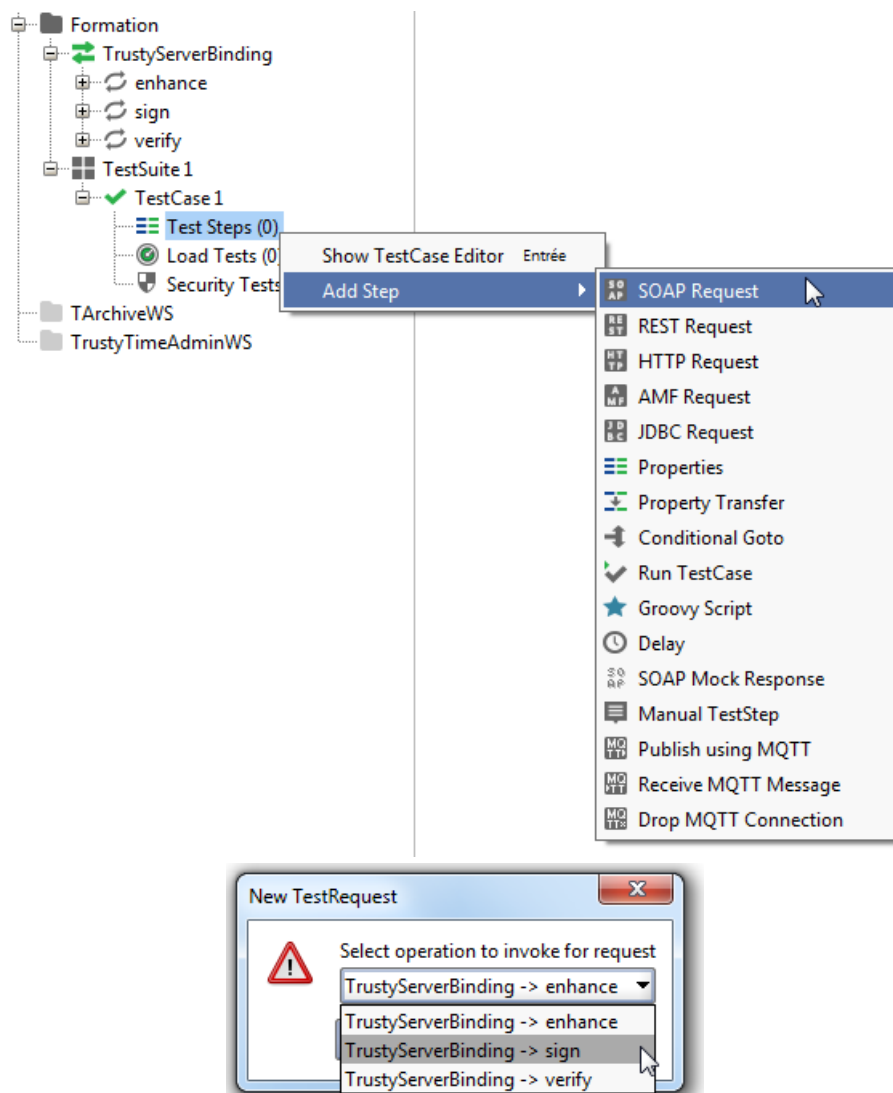
5.5 Déclarer l'appel HTTPS





5.6 Signature

5.6.1 Déclarer une étape de test



Dans la requête proposée par défaut par l'outil, modifier les éléments suivants (en gras) :

```

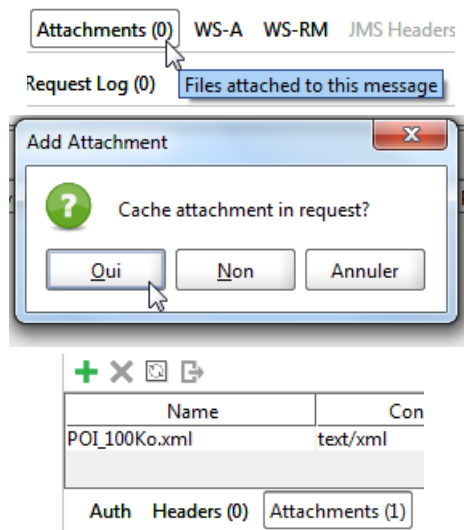
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:ts="http://com.cs.server.web.serveur/services">
  <soapenv:Header/>
  <soapenv:Body>
    <ts:SignRequest>
      <TransactionKey>
        <AppId>CACHET_TEST</AppId>
        <policyId>Signature Datas</policyId>
        <requestId>uniqueIdTest</requestId>
      </TransactionKey>
    </ts:SignRequest>
  </soapenv:Body>
</soapenv:Envelope>
  
```

```
<Data><xop:Include href="cid:POI_100Ko.xml"
xmlns:xop="http://www.w3.org/2004/08/xop/include"/></Data>
<Options>
  <ProfileId>Client</ProfileId>
</Options>
</ts:SignRequest>
</soapenv:Body>
</soapenv:Envelope>
```

Les paramètres d'appel sont :

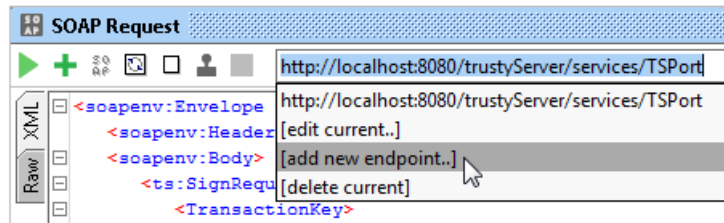
- ✓ AppId : le nom du client
- ✓ policyId : le nom de la politique
- ✓ requestId : un identifiant unique attribué par le système en interface
- ✓ ProfileId : le nom du profil à utiliser

La requête contient le fichier à signer :

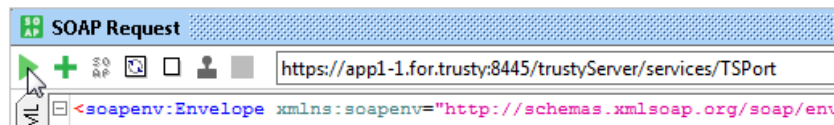
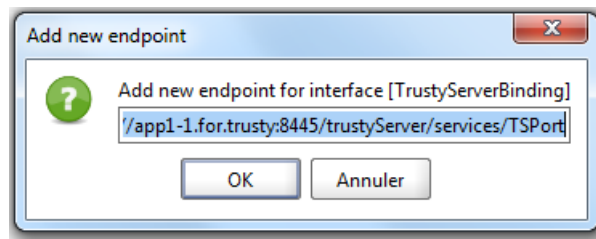


```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:ts="http://com.cs.server.web.serveur/services">
  <soapenv:Header/>
  <soapenv:Body>
    <ts:SignRequest>
      <TransactionKey>
        <AppId>TEST</AppId>
        <policyId>PDF_1</policyId>
        <requestId>uniqueIdTest</requestId>
      </TransactionKey>
      <Data><xop:Include href="cid:POI_100ko.xml" xmlns:xop="http://www.w3.org/2004/08/xop/include"/></Data>
      <Options>
        <ProfileId>PRF_1</ProfileId>
      </Options>
    </ts:SignRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

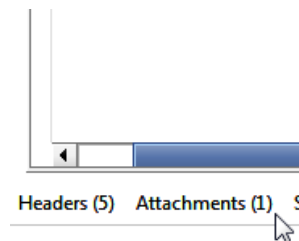
5.6.2 Lancer la requête



<https://app1-1.for.trusty:8445/trustyServer/services/TSPort>



5.6.3 Récupérer la réponse

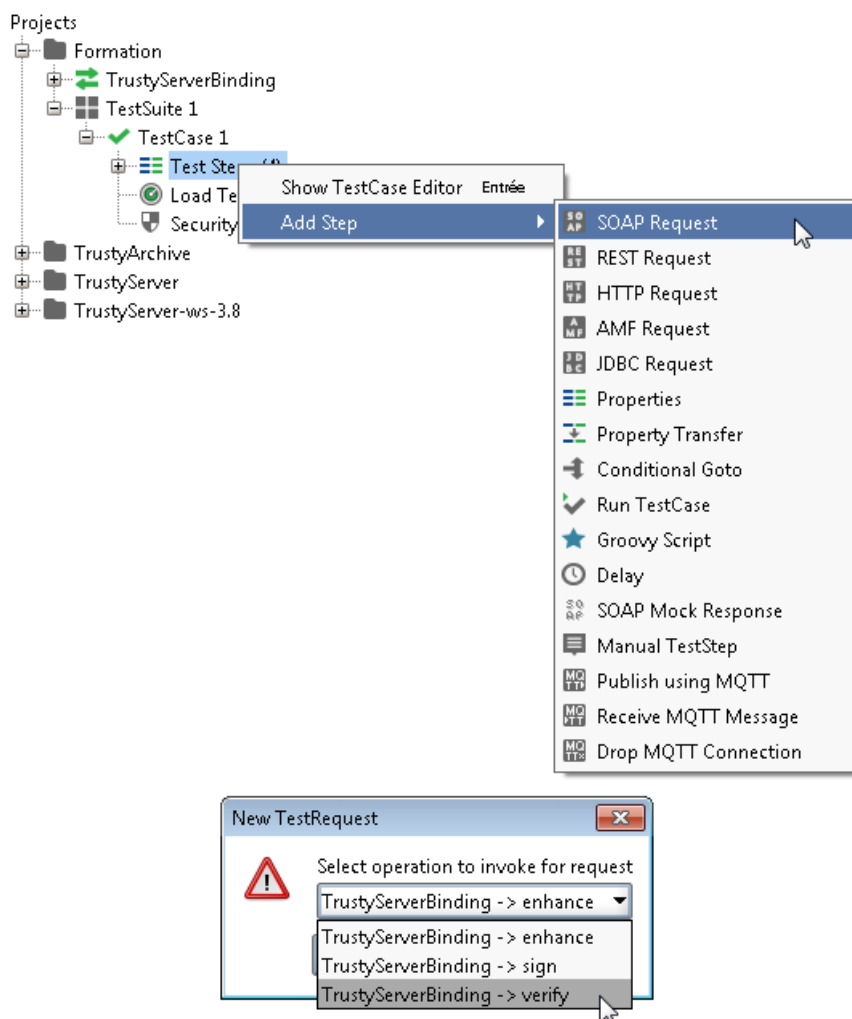


Name	Content type	Size	Part
dfc0070b-561b-4...	text/plain	94382	dfc0070b-561b-4... XOP

Headers (5) Attachments (1) SSL Info (2 certs) WSS (0) JMS (0)

5.7 Vérification de signature

5.7.1 Déclarer une étape de test



Dans la requête proposée par défaut par l'outil, modifier les éléments suivants (en gras) :

```

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ts="http://com.cs.server.web.serveur/services">
  <soapenv:Header/>
  <soapenv:Body>
    <ts:VerifyRequest>
      <TransactionKey>
        <AppId>TEST_4096</AppId>
        <policyId>XADES_BES_ENVELOPPANT</policyId>
        <requestId>uniqueIdTest</requestId>
      </TransactionKey>
      <Data><xop:Include href="cid:XADES_BES_ENVELOPPANT-PRF_1.xml"
xmlns:xop="http://www.w3.org/2004/08/xop/include"/></Data>
      <ReturnX509Info>>false</ReturnX509Info>
    </ts:VerifyRequest>
  </soapenv:Body>
</soapenv:Envelope>
  
```

```

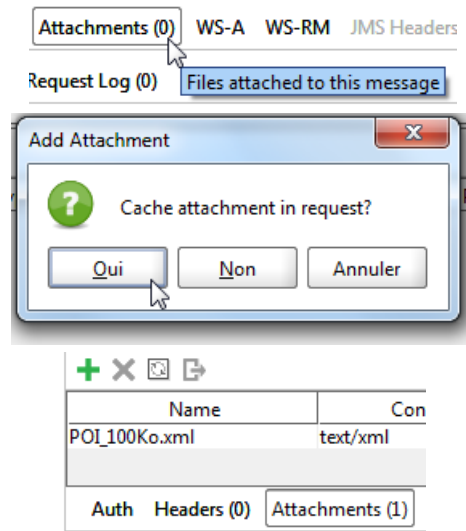
<Options>
  <ReturnSignatureInfo>false</ReturnSignatureInfo>
  <SignatureId>PRF_1</SignatureId>
</Options>
</ts:VerifyRequest>
</soapenv:Body>
</soapenv:Envelope>

```

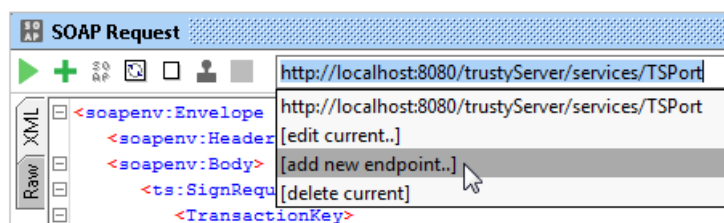
Les paramètres d'appel sont :

- ✓ Appld : le nom du client
- ✓ policyId : le nom de la politique
- ✓ requestId : un identifiant unique attribué par le système en interface
- ✓ ProfileId : le nom du profil à utiliser

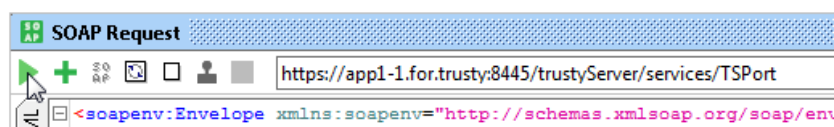
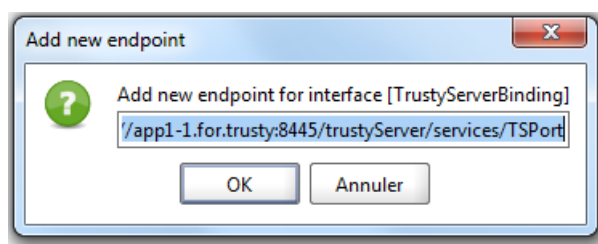
La requête contient le fichier à signer :



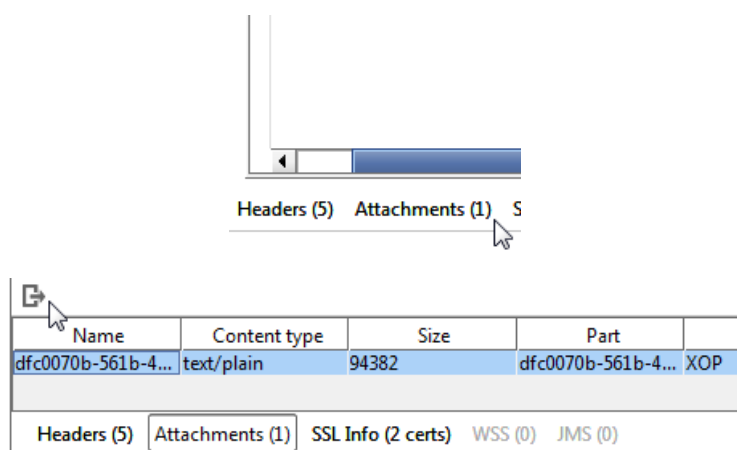
5.7.2 Lancer la requête



<https://app1-1.for.trusty:8445/trustyServer/services/TSPort>



5.7.3 Récupérer la réponse



6. CODES D'ERREUR

Le tableau ci-dessous présente les différents codes d'erreur qui peuvent être retournés dans le champ `ResulMessage` lors d'un appel à une fonction de service.

Code	Descriptif	Cause et résolution du problème
[TSRV-108]	Le service de signature n'est pas opérationnel	Le service de signature n'est pas fonctionnel car sa configuration est incomplète ou erronée, ou bien la bi-clé est en cours de renouvellement. Réessayer ultérieurement ou contacter l'administrateur du service.
[TSRV-150]	L'application cliente <> n'est pas autorisée à exécuter la requête	L'application identifiée dans la requête n'a pas le droit d'interroger le service de signature. Corriger l'identifiant de l'application.
[TSRV-160]	aucune signature XML dans le document	La signature du document XML à vérifier n'a pas été trouvée dans le document. Vérifier le document XML envoyé.
[TSRV-170]	la politique de signature <> n'est pas enregistrée	La politique de signature référencée dans la requête n'existe pas sur le service de signature. Corriger le nom de la politique.
[TSRV-171]	la politique <> ne peut être invoquée par l'application <>	L'application identifiée dans la requête n'a pas le droit d'utiliser la politique référencée. Utiliser une autre politique.
[TSRV-172]	le format de la politique <> est inadapté	La politique référencée dans la requête n'est pas prévue pour traiter le format de document envoyé. Utiliser une autre politique.
[TSRV-180]	requête invalide : <>	Le format de la requête est invalide. Vérifier la requête envoyée et sa compatibilité avec la version du service de signature.
[TSRV-210]	client non authentifié	Le client envoyant la requête n'est pas authentifié auprès du service de signature. Vérifier que la connexion au service de signature se fait bien en https avec authentification mutuelle.
[TSRV-211]	échec d'authentification d'un client	Le client envoyant la requête n'a pas réussi à s'authentifier auprès du service de signature Analyser la cause fournie dans le message et corriger l'authentification.
[TSRV-220]	la transformation <> n'existe pas	La transformation de signature demandée n'existe pas sur le service de signature. Corriger la transformation de signature demandé ou vérifier la compatibilité avec la version du service de signature.
[TSRV-287]	Erreur interne du service de signature	La signature n'a pas pu être traitée correctement sur le serveur. Réessayer ultérieurement ou contacter

Code	Descriptif	Cause et résolution du problème
		l'administrateur du service.

Versions successives



Version	Date	Motif
3.10	3/11/2017	§ 3.6, options d'enrichissement
3.9	2/11/2016	Ajout du § 5 de mise en œuvre de SoapUI
3.8	05/09/2016	Précisions sur le champ « FinalFormat » (fonction « Enhance ») Fonction et processus de vérification différée
3.7	22/09/2015	Ajout du DN du signataire §3.3.2.2 Ajout pour la signature détachée §3.3.1, §3.4.1 Ajout des exemples §4
3.6	21/07/2015	Retrait du document de référence [Politiques] Révision des codes d'erreur
3.5	02/04/2015	Ajout de SignerInfo.ArchiveId
3.4	23/03/2015	Mise à jour mineure
3.3	03/02/2015	Mise à jour des interfaces Web services
3.0	15/11/2014	Mise à jour des interfaces Web services
2.0	25/06/2012	Mise à jour des interfaces
1.0	25/01/2011	Création du document