Définition des Webservices Paiement par identifiant Systempay v1.26.3

Version 2.1d

Historique du document

Version	Auteur	Date	Commentaires
2.1d	Lyra-Network	31/05/2013	Précisions apportées sur le champ extendedErrorCode Ajout des codes retour autorisation
2.1c	Lyra-Network	31/05/2013	Précisions apportées sur la fonction subscriptionModify
2.1b	Lyra-Network	21/01/2013	Précisions apportées sur le champ cvv Précisions sur les codes d'erreur Précision sur le champ paymentMethod Maintien de la session HTTP lors de l'enchainement de plusieurs requêtes Ajout d'exemples de fichiers xml
2.1a	Lyra-Network	18/06/2012	Modification des champs présents dans le type identResponse
2.1	Lyra-Network	18/06/2012	Nouvelle version du wsdl. Ajout du service customerReactivate
2.0a	Lyra-Network	06/04/2012	Ajout précision sur les dates
2.0	Lyra-Network	15/06/2010	V2
1.1	Lyra-Network	07/08/2010	Ajout services.
1.0	Lyra-Network	07/06/2010	Version initiale.

SOMMAIRE

1. Présentation

Ce document présente les webservices identifiant qui permettent d'automatiser les actions réalisables manuellement depuis l'outil de gestion de caisse commerçant concernant les abonnements.

Ces webservices ont été développés suivant le protocole SOAP (Simple Object Access Protocol) et sont décrits par le fichier wsdl suivant :

https://paiement.systempay.fr/vads-ws/ident-v2.1?wsdl

Afin de sécuriser les échanges, les webservices (SOAP) sont cryptés grâce au protocole HTTPS. De plus un mécanisme de signature a été mis en place afin de valider et d'authentifier l'échange des données.

2. Description des types

2.1. Date

Les champs 'date' doivent suivre les recommandations W3C (http://www.w3.org/TR/NOTE-datetime).
Par exemple pour une date d'expiration en décembre 2012, le format correct devrait-être 2012-12-31T00:00:00+00:00.
Attention :
Lors du calcul de signature, le format des champs de type 'date' est YYYYmmDD.
(cf. chapitre 4)

2.2. identResponse

Nom du champ	Type	Description
errorCode	int	Code d'erreur cf.2.13
extendedErrorCode	String	Précision sur le code d'erreur
transactionStatus	int	Statut de la transaction
identId	String	Identifiant du compte carte
transactionId	String	Identifiant de la transaction
subscriptionId	String	Identifiant de l'abonnement
paymentWarranty	paymentWarranty	Garantie du paiement.
amount	Long	Montant de la transaction
authNumber	String	Numéro d'autorisation
authorizationDate	Date	Date et heure de la demande d'autorisation
timestamp	long	Timestamp permettant la génération de signature unique
signature	String	Signature de la réponse (cf. §4)

La signature permet de valider l'intégrité de la réponse, le calcul de cette signature se fait en prenant les paramètres dans l'ordre suivant :

timestamp, errorCode, extendedErrorCode, identId, transactionId, subscriptionId, transactionStatus, , **paymentWarranty**, **amount**, **authNumber**, **authorizationDate**

Remarque :
Les champs paymentWarranty, amount, authNumber, authorizationDate ne seront renvoyés que si une transaction a été créée.

2.3. paymentWarranty

Ce type indique l'état de la garantie du paiement. Les valeurs possibles sont définies ci-dessous :

Valeur	Description
NO	Pas de garantie
YES	Garantie du paiement

2.4. createPaiementIdentInfo

Ce type permet de décrire les paramètres pour une création de transaction par identifiant.

Nom du champ	Type	Description	Obligatoire
Détail transaction			
shopId	String	Identifiant de la boutique	✓
transmissionDate	Date	Date et heure **UTC** de la transaction exprimée au format W3C (2012-06-08T08:16:43+00:00). Représente la date et l'heure de la requête. Si la valeur de ce champ est trop éloignée de l'heure actuelle, la requête sera rejetée (errorCode 6)	✓
transactionId	String	Identifiant de la transaction **sur 6 chiffres**. Cet identifiant doit être unique sur une même journée.	✓
paymentMethod	String	Source du paiement: - EC: Commerce Électronique - MOTO : commande par mail ou téléphone - CC : Centre d'appel OTHER : autre canal de vente **Remarque :** La valeur **EC** impose qu'un contrat E-commerce **VADS (ERT 24)** soit associé à votre boutique. Toutes les autres valeurs nécessitent un contrat VAD (ERT 20). Dans le cas contraire, la requête sera rejetée avec un code d'erreur 15 (cf. 2.13)	✓
orderId	String	Référence de la commande	✓
orderInfo1	String	Description libre de la commande	
orderInfo2	String	Description libre de la commande	
orderInfo3	String	Description libre de la commande	
amount	Long	Montant de la transaction en plus petite unité monétaire (en centimes pour euro)	✓
devise	int	Code de la devise suivant la norme ISO 4217, (978 pour EURO)	✓
presentationDate	Date	Ce champ permet de définir la date de remise de la transaction. Si le nombre de jours entre la date de remise demandée et la date actuelle est supérieur à 7 jours, une autorisation d'1 euro sera réalisée le jour de la transaction. Ceci afin de vérifier la validité de la carte. L'autorisation pour le montant total sera effectuée entre 7 jours et 0 jour avant la date de remise, en fonction du paramétrage de votre boutique (Avec ou Sans autorisations anticipées). Si vous souhaitez être notifié du résultat de cette demande d'autorisation, vous devez souscrire l'option 'appel étendu de l'URL serveur'.	
manualValidation	int	Mode de validation des paiements : 0= Automatique ; 1= Manuelle	
Détail carte			
cardIdent	String	Identifiant du compte carte	✓
cvv	String	Cryptogramme visuel à 3 chiffres (ou 4 pour Amex : 4DBC) **Ce champ est obligatoire lorsque la carte dispose d'un cryptogramme visuel et que l'internaute l'a saisi.** **Remarque :** **Certaines cartes ne possédant pas de CVV, le champ cvv est optionnel dans la méthode create.**	
contractNumber	String	Numéro de contrat commerçant. Si ce champ est renseigné, veillez à utiliser le bon contrat en fonction du réseau de la carte. Par exemple, le contrat CB ne peut être utilisé pour une transaction AMEX.	

Divers			
ctxMode	String	Contexte de sollicitation de la plateforme de paiement ("TEST", "PRODUCTION")	✓
comment	String	Commentaire « libre »	

2.5. customerModifyInfo

Ce type permet de modifier les paramètres d'un identifiant.

Nom du champ	Type	Description	Obligatoire
customerId	String	Référence client	
customerTitle	String	Civilité	
customerStatus	custStatus	Cf.2.6	
customerName	String	Nom du client	
customerPhone	String	Numéro de téléphone	
customerMobilePhone	String	Numéro de téléphone portable	
customerMail	String	Adresse de courrier électronique	
customerAddressNumber	String	Numéro de rue	
customerAddress	String	Adresse	
customerDisctrict	String	Quartier	
customerZipCode	String	Code postal	
customerCity	String	Ville	
customerState	String	Etat	
customerCountry	String	Pays	
customerLanguage	String	Langue du client (norme ISO 639-1 sur 2 caractères).	

2.6. custStatus

Ce type décrit le type de client. Les valeurs possibles sont définies ci-dessous :

Valeur	Description
PRIVATE	Particulier
COMPANY	Entreprise

2.7. customerInfo

Ce type permet de décrire un identifiant.

Nom du champ	Type	Description
errorCode	int	Code d'erreur cf.**Erreur ! Source du renvoi introuvable.**
customerId	String	Référence client
customerTitle	String	Civilité
customerName	String	Nom du client
customerPhone	String	Numéro de téléphone
customerMobilePhone	String	Numéro de téléphone mobile
customerMail	String	Adresse de courrier électronique
customerAddress	String	Adresse
customerZipCode	String	Code postal
customerCity	String	Ville
customerState	String	Etat
customerCountry	String	Pays
customerLanguage	String	Langue du client (norme ISO 639-1 sur 2 caractères).
customerCreationDate	Date	Date de création de l'identifiant
customerCancelDate	Date	Date de résiliation de l'identifiant
cardNumber	String	Numéro de carte masqué
cardExpirationDate	Date	Date d'expiration de la carte
cardType	String	Type de la carte (CB / Visa / Mastercard / …)
autoNum	String	Numéro de l'autorisation
autoDate	Date	Date de l'autorisation
timestamp	long	Timestamp permettant la génération de signature unique
signature	String	Cf. ci-dessous

La signature permet de valider l'intégrité de la réponse, le calcul de cette signature se fait en prenant les paramètres dans l'ordre suivant :

errorCode, customerId, customerTitle, customerName, customerPhone, customerMobilePhone, customerMail, customerAddress, customerZipCode, customerCity, customerState, customerCountry, customerLanguage, customerCreationDate, customerCancelDate, cardNumber, cardExpirationDate, cardType, autoNum, autoDate, timestamp

2.8. subscriptionCreationInfo

Ce type permet de décrire les paramètres pour une création d'abonnement par identifiant.

Nom du champ	Type	Description	Obligatoire
shopId	String	Identifiant de la boutique	✓
transmissionDate	Date	Date et heure **UTC** de la transaction exprimée au format W3C (2012-06-08T08:16:43+00:00). Représente la date et l'heure de la requête. Si la valeur de ce champ est trop éloignée de l'heure actuelle, la requête sera rejetée (errorCode 6)	✓
subscriptionId	String	Identifiant de l'abonnement	
effectDate	Date	Date d'effet de l'abonnement	✓
paymentMethod	String	Source du paiement: - EC: Commerce Électronique - MOTO : commande par mail ou téléphone - CC : Centre d'appel OTHER : autre canal de vente **Remarque :** La valeur **EC** impose qu'un contrat E-commerce **VADS (ERT 24)** soit associé à votre boutique. Toutes les autres valeurs nécessitent un contrat VAD (ERT 20). Dans le cas contraire, la requête sera rejetée avec un code d'erreur 15 (cf. 2.13)	✓
orderId	String	Référence de la commande	✓
orderInfo1	String	Description libre de la commande	
orderInfo2	String		
orderInfo3	String		
amount	Long	Montant de la transaction exprimé dans la plus petite unité monétaire (en centimes pour Euro)	✓
devise	int	Code de la devise suivant la norme ISO 4217, (978 pour EURO)	✓
initialAmount	Long	Montant de ou des premières occurrences	
occInitialAmount	int	Nombre d'occurrence initiale	
subscriptionDescr	String	Description (RRULE) de la règle d'abonnement Cf. vads_sub_desc de la doc Guide_d'implémentation_formulaire_ Paiement_V2_Abonnements	✓
subscriptionDescrHR	String	Description textuelle de la règle d'abonnement	
manualValidation	int	Mode de validation des paiements : 0= Automatique ; 1= Manuelle	
cardIdent	String	Identifiant du compte carte	✓
contractNumber	String	Numéro de contrat commerçant. Si ce champ est renseigné, veillez à utiliser le bon contrat en fonction du réseau de la carte. Par exemple, le contrat CB ne peut être utilisé pour une transaction AMEX.	
ctxMode	String	Contexte de sollicitation de la plateforme de paiement ("TEST", "PRODUCTION")	✓
comment	String	Commentaire « libre »	

2.9. subscriptionModifyInfo

Ce type permet de modifier les paramètres d'un abonnement.

Nom du champ	Type	Description	Obligatoire
effectDate	Date	Date d'effet de l'abonnement	
paymentMethod	String	Source du paiement: - EC: Commerce Électronique - MOTO : commande par mail ou téléphone - CC : Centre d'appel OTHER : autre canal de vente **Remarque :** La valeur **EC** impose qu'un contrat E-commerce **VADS (ERT 24)** soit associé à votre boutique. Toutes les autres valeurs nécessitent un contrat VAD (ERT 20). Dans le cas contraire, la requête sera rejetée avec un code d'erreur 15 (cf. 2.13)	
amount	Long	Montant de la transaction en plus petite unité monétaire	
devise	int	Devise (Code monnaie ISO 4217, Euro : 978)	
initialAmount	Long	Montant de ou des premières occurrences	
occInitialAmount	int	Nombre d'occurrence initiale	
subscriptionDescr	String	Description (RRULE) de la règle d'abonnement Cf. vads_sub_desc de la doc Guide_d'implémentation_formulaire_ Paiement_V2_Abonnements	
subscriptionDescrHR	String	Description textuelle de la règle d'abonnement	
manualValidation	int	Mode de validation des paiements : 0= Automatique ; 1= Manuelle	
contractNumber	String	Numéro de contrat commerçant	

2.10. subscriptionInfo

Ce type permet de décrire un abonnement.

Nom du champ	Type	Description
errorCode	int	Code d'erreur
shopId	String	Identifiant de la boutique
transmissionDate	Date	Date de transaction
subscriptionId	String	Identifiant de l'abonnement
effectDate	Date	Date d'effet de l'abonnement
cancelDate	Date	Date de résiliation de l'abonnement
paymentMethod	String	Source du paiement : - "E_COMMERCE" : e-Commerce - "MAIL_OR_TELEPHONE" : mail ou téléphone - "CALL_CENTER" : centre d'appel - "OTHER" : autres
orderId	String	Référence de la commande
orderInfo	String	Description libre de la commande
orderInfo2	String	
orderInfo3	String	
amount	Long	Montant de la transaction en plus petite unité monétaire
devise	int	Code de la devise (Code monnaie ISO 4217, Euro : 978)
initialAmount	Long	Montant de ou des premières occurrences
occInitialAmount	int	Nombre d'occurrence initiale
subscriptionDescr	String	Description (RRULE) de la règle d'abonnement Cf. vads_sub_desc de la doc Guide_d'implémentation_formulaire_Paiement_V2_Abonnements
subscriptionDescrHR	String	Description textuelle de la règle d'abonnement
pastOccNb	Int	Nombre d'occurrences échues
totalOccNb	int	Nombre d'occurrences total
manualValidation	int	Mode de validation des paiements : 0= Automatique ; 1= Manuelle
cardIdent	String	Identifiant du compte carte
contractNumber	String	Numéro de contrat commerçant utilisé
ctxMode	String	Contexte de sollicitation de la plateforme de paiement ("TEST", "PRODUCTION")
timestamp	long	Timestamp permettant la génération de signature unique
signature	String	Cf. ci-dessous

La signature permet de valider l'intégrité de la réponse, le calcul de cette signature se fait en prenant les paramètres dans l'ordre suivant :

errorCode, shopId, transmissionDate, subscriptionId, effectDate, cancelDate, paymentMethod, orderId, orderInfo, orderInfo2, orderInfo3, amount, devise, initialAmount, occInitialAmount, subscriptionDescr, subscriptionDescrHR, pastOccNb, totalOccNb, manualValidation, cardIdent, contractNumber, ctxMode, timestamp

2.11. identCreationInfo

Ce type permet de décrire les paramètres pour une création d'identifiant.

Nom du champ	Type	Description	Obligatoire Création	Obligatoire Update
Détail transaction				
shopId	String	Identifiant de la boutique	✓	✓
transmissionDate	Date	Date et heure **UTC** de la transaction exprimée au format W3C (2012-06-08T08:16:43+00:00). Représente la date et l'heure de la requête. Si la valeur de ce champ est trop éloignée de l'heure actuelle, la requête sera rejetée (errorCode 6).	✓	✓
cardIdent	String	Identifiant du compte carte		✓
paymentMethod	String	Source du paiement: - EC: Commerce Électronique - MOTO : commande par mail ou téléphone - CC : Centre d'appel - OTHER : autre canal de vente **Remarque :** La valeur **EC** impose qu'un contrat E-commerce **VADS (ERT 24)** soit associé à votre boutique. Toutes les autres valeurs nécessitent un contrat VAD (ERT 20). Dans le cas contraire, la requête sera rejetée avec un code d'erreur 15 (cf. 2.13)		
Détail carte				
cardNumber	String	Numéro de carte	✓	
cardNetwork	String	Réseau de carte ("AMEX", "CB", "MASTERCARD", "VISA", "MAESTRO", "E-CARTEBLEUE")	✓	
cardExpirationDate	Date	Date expiration de la carte. On conseille de positionner l'heure à 00:00:00UTC.	✓	
cvv	String	Cryptogramme visuel à 3 chiffres (ou 4 pour Amex : 4DBC) **Ce champ est obligatoire lorsque la carte dispose d'un cryptogramme visuel et que l'internaute l'a saisi.** **Remarque :** **Certaines cartes ne possédant pas de CVV, le champ cvv est optionnel dans la méthode create.**		
contractNumber	String	Numéro de contrat commerçant. Si ce champ est renseigné, veillez à utiliser le bon contrat en fonction du réseau de la carte. Par exemple, le contrat CB ne peut être utilisé pour une transaction AMEX.		
threeDsResult	threeDsResult	Cf.2.12		
Détail porteur				
customerId	String	Référence client		
customerTitle	String	Civilité		
customerName	String	Nom du client		
customerStatus	custStatus	Cf.2.6		
customerPhone	String	Téléphone client		
customerMobilePhone	String	Téléphone mobile client		
customerMail	String	Adresse de courrier électronique	✓	
customerAddressNumber	String	Numéro de rue		
customerAddress	String	Adresse		

customerDistrict	String	Quartier		
customerZipCode	String	Code postal		
customerCity	String	Ville		
customerState	String	Etat		
customerCountry	String	Pays		
customerLanguage	String	Langue client (Code ISO 639-1, sur 2 caractères)		
customerIP	String	Adresse IP		
Divers				
ctxMode	String	Contexte de sollicitation de la plateforme de paiement ("TEST", "PRODUCTION")	✓	✓
comment	String	Commentaire « libre »		

2.12. threeDSResult

Ce type permet de décrire les paramètres de retour 3DS.

Nom du champ	Type	Description	Obligatoire
brand	String	Brand de la carte ("VISA" ou "MASTERCARD")	✓
enrolled	String	Statut enrôlement porteur : • "Y" : Enrôlé • "N" : Non enrôlé • "U" : Inconnu	✓
authStatus	String	Statut authentification : • "Y" : Authentifié 3DS • "N" : Erreur Authentification • "U" : Authentification impossible • "A" : Essai d'authentification	
eci	String	Indicateur de commerce Electronique	
xid	String	Numéro de transaction 3DS	
cavv	String	Certificat de l'ACS	
cavvAlgorithm	String	Algorithme de vérification de l'authentification du porteur (CAVV) : • "0" : HMAC • "1" : CVV • "2" : CVV_ATN • "3" : Mastercard SPA	

2.13. Description des codes erreur

Les codes d'erreurs **en gras** sont détaillés sur la page suivante.

Error Code	Description	Error Code	Description
0	**Action réalisée avec succès**	62	Paramètre 'orderInfo2' invalide
1	**Action non autorisée. Assurez-vous que l'offre souscrite auprès de Systempay incluse l'utilisation des web services identifiant ainsi que la gestion des comptes clients. Vérifiez que le mode de sollicitation de la plateforme (ctxMode) soit correct**	63	Paramètre 'orderInfo3' invalide
2	Transaction non trouvée	64	Paramètre 'PaymentMethod' invalide
3	Transaction pas dans le bon état	65	Paramètre 'cardNumber' invalide
4	Transaction existe déjà	66	Paramètre 'contractNumber' invalide
5	Mauvaise signature	67	Paramètre 'customerId' invalide
6	Mauvaise date	68	Paramètre 'customerTitle' invalide
10	Mauvais montant	69	Paramètre 'customerName' invalide
11	Mauvaise devise	70	Paramètre 'customerPhone' invalide
12	Type de carte inconnu	71	Paramètre 'customerMail' invalide
13	Paramètre 'date d'expiration' invalide	72	Paramètre 'customerAddress' invalide
14	Paramètre 'cvv' invalide	73	Paramètre 'customerZipCode' invalide
15	**Contrat inconnu**	74	Paramètre 'customerCity' invalide
16	Paramètre 'Numéro de carte' invalide	75	Paramètre 'customerCountry' invalide
17	Identifiant non trouvé	76	Paramètre 'customerLanguage' invalide
18	Identifiant non valide (Résilié, …)	77	Paramètre 'customerIp' invalide
19	Subscription non trouvée	78	Paramètre 'customerSendMail' invalide
20	Subscription non valide	79	Paramètre 'customerMobilePhone' invalide
21	Identifiant déjà existant	80	Paramètre 'subPaiementType' invalide
22	Création d'identifiant refusé	81	Paramètre 'subReference' invalide
40	Identifiant purgé	82	Paramètre 'initialAmount' invalide
50	Plage non trouvée	83	Paramètre 'occInitialAMount' invalide
51	Paramètre 'shopId' invalide	84	Paramètre 'effectDate' invalide
52	Paramètre 'transmissionDate' invalide	85	Paramètre 'state' invalide
53	Paramètre 'transactionId' invalide	90	Paramètre 'enrolled' invalide
54	Paramètre 'ctxMode' invalide	91	Paramètre 'authStatus' invalide
55	Paramètre 'comment' invalide	92	Paramètre 'eci' invalide
56	Paramètre 'autoNb' invalide	93	Paramètre 'xid' invalide
57	Paramètre 'autoDate' invalide	94	Paramètre 'cavv' invalide
58	Paramètre 'presentationDate' invalide	95	Paramètre 'cavvAlgo' invalide
59	Paramètre 'newTransactionId' invalide	96	Paramètre 'brand' invalide
60	Paramètre 'validationMode' invalide	98	Paramètre 'requestId' invalide
61	Paramètre 'orderId' invalide	99	Autre erreur
		999	Donnée sensible détectée

Une donnée sera considérée comme 'sensible' si sa longueur est comprise entre 13 et 19 chiffres, et si sa valeur peut être interprétée comme un numéro de carte bancaire.

Précisions sur les codes d'erreurs

- **ErrorCode 0 :**

Indique que l'action demandée a été réalisée avec succès, traduisant ainsi que le format de la requête est correct.

Remarque :
Dans le cas d'une création de paiement (méthode create) ce code d'erreur ne doit pas être confondu avec le champ transactionStatus qui est le seul à donner le résultat du paiement.

Ainsi on pourra avoir un errorCode à 0 et un transactionStatus à 8, correspondant à la création d'une transaction dont la demande d'autorisation a été refusée.

- **ErrorCode 1 :**

Indique que vous n'avez pas souscrit une offre Systempay permettant d'utiliser les webservices.

- **ErrorCode 15 :**

Indique un défaut au niveau du contrat commerçant.
Plusieurs cas possibles :

- La valeur transmise dans la requête ne correspond à aucun contrat enregistré sur la boutique (shopId),
- Il n'y a pas de contrat enregistré sur la boutique,
- Le contrat spécifié est clôturé,

Aucun contrat ne correspond au type de contrat nécessaire pour effectuer le paiement. C'est le cas si vous ne possédez pas de contrat VAD (ERT 20) et que **paymentMethod** est valorisé à **MOTO**, **CC** ou **OTHER** dans votre requête.

2.14. Liste des codes retour autorisation

Valeur	Description
00	transaction approuvée ou traitée avec succès
02	contacter l'émetteur de carte
03	accepteur invalide
04	conserver la carte
05	ne pas honorer
07	conserver la carte, conditions spéciales
08	approuver après identification
12	transaction invalide
13	montant invalide
14	numéro de porteur invalide
15	*Emetteur de carte inconnu*
17	*Annulation client*
19	*Répéter la transaction ultérieurement*
20	Réponse erronée (erreur dans le domaine serveur)
24	*Mise à jour de fichier non supportée*
25	*Impossible de localiser l'enregistrement dans le fichier*
26	*Enregistrement dupliqué, ancien enregistrement remplacé*
27	*Erreur en « edit » sur champ de lise à jour fichier*
28	*Accès interdit au fichier*
29	*Mise à jour impossible*
30	erreur de format
31	identifiant de l'organisme acquéreur inconnu
33	date de validité de la carte dépassée
34	suspicion de fraude
38	*Date de validité de la carte dépassée*
41	carte perdue
43	carte volée
51	provision insuffisante ou crédit dépassé
54	date de validité de la carte dépassée
55	*Code confidentiel erroné*
56	carte absente du fichier
57	transaction non permise à ce porteur
58	transaction interdite au terminal
59	suspicion de fraude
60	l'accepteur de carte doit contacter l'acquéreur
61	*montant de retrait hors limite*
63	règles de sécurité non respectées
68	réponse non parvenue ou reçue trop tard
75	*Nombre d'essais code confidentiel dépassé*
76	*Porteur déjà en opposition, ancien enregistrement conservé*
90	*arrêt momentané du système*
91	émetteur de cartes inaccessible
94	transaction dupliquée
96	mauvais fonctionnement du système
97	échéance de la temporisation de surveillance globale
98	serveur indisponible routage réseau demandé à nouveau
99	*incident domaine initiateur*

3. Description des méthodes

3.1. create

Cette fonction permet de créer les paiements à partir d'un identifiant.
Cette fonction prend en entrée les paramètres suivants :

Nom du champ	Type	Description	Obligatoire
createInfo	createPaiementIdentInfo	cf. 2.4	✓
wsSignature	String	Signature (cf. ci-dessous)	✓

Le calcul de la signature se fait en prenant les paramètres dans l'ordre suivant :
shopId, transmissionDate, transactionId, paymentMethod, orderId, orderInfo1, orderInfo2, orderInfo3, amount, devise, presentationDate, manualValidation, cardIdent, contractNumber, ctxMode, comment

Cette fonction retourne une réponse du type **identResponse** (cf. 2.2).

NB :
Le code d'erreur étendu (extendedErrorCode) n'est pas renseigné pour cette fonction.

3.2. customerModify

Cette fonction permet de modifier les informations d'un identifiant.
Cette fonction prend en entrée les paramètres suivants :

Nom du champ	Type	Description	Obligatoire
shopId	String	Identifiant de la boutique	✓
cardIdent	String	Identifiant du compte carte à modifier	✓
custInfo	customerModifyInfo	Cf. 2.5	✓
ctxMode	String	Contexte de sollicitation de la plateforme de paiement ("TEST", "PRODUCTION")	✓
wsSignature	String	Signature (cf. détail ci-dessous)	✓

Le calcul de la signature se fait en prenant les paramètres dans l'ordre suivant :
shopId, cardIdent, custInfo.customerId, custInfo.customerTitle, custInfo.customerName, custInfo.customerPhone, custInfo.customerMobilePhone, custInfo.customerMail, custInfo.customerAddress, custInfo.customerZipCode, custInfo.customerCity, custInfo.customerState, custInfo.customerCountry, custInfo.customerLanguage, ctxMode

Dans le champ custInfo, les champs non renseignés ne sont pas modifiés.
Cette fonction retourne une réponse du type **identResponse** (cf.2.2).

NB :
Le code d'erreur étendu (extendedErrorCode) et le statut de la transaction (transactionStatus) ne sont pas renseignés pour cette fonction.

3.3. customerCancel

Cette fonction permet de résilier un identifiant à une date donnée.
Cette fonction prend en entrée les paramètres suivants :

Nom du champ	Type	Description	Obligatoire
shopId	String	Identifiant de la boutique	✓
cardIdent	String	Identifiant du compte carte à résilier	✓
date	Date	Date de résiliation de l'identifiant	✓
ctxMode	String	Contexte de sollicitation de la plateforme de paiement ("TEST", "PRODUCTION")	✓
wsSignature	String	Signature (cf. ci-dessous)	✓

Le calcul de la signature se fait en prenant les paramètres dans l'ordre suivant :
shopId, cardIdent, date, ctxMode

Cette fonction retourne une réponse du type **identResponse** (cf. 2.2).

NB :
Le code d'erreur étendu (extendedErrorCode) et le statut de la transaction (transactionStatus) ne sont pas renseignés pour cette fonction.

3.4. customerReactivate

Cette fonction permet de réactiver un identifiant.
Cette fonction prend en entrée les paramètres suivants :

Nom du champ	Type	Description	Obligatoire
shopId	String	Identifiant de la boutique	✓
cardIdent	String	Identifiant du compte carte à réactiver	✓
ctxMode	String	Contexte de sollicitation de la plateforme de paiement ("TEST", "PRODUCTION")	✓
wsSignature	String	Signature (cf. ci-dessous)	✓

Le calcul de la signature se fait en prenant les paramètres dans l'ordre suivant :
shopId, cardIdent, ctxMode

Cette fonction retourne une réponse du type **identResponse** (cf.2.2).

3.5. customerInfo

Cette fonction permet d'interroger un identifiant pour en connaître ses différents attributs.
Cette fonction prend en entrée les paramètres suivants :

Nom du champ	Type	Description	Obligatoire
shopId	String	Identifiant de la boutique	✓
cardIdent	String	Identifiant du compte carte	✓
ctxMode	String	Contexte de sollicitation de la plateforme de paiement ("TEST", "PRODUCTION")	✓
wsSignature	String	Signature (cf. ci-dessous)	✓

Le calcul de la signature se fait en prenant les paramètres dans l'ordre suivant :
shopId, cardIdent, ctxMode

Cette fonction retourne une réponse du type **customerInfo** (cf.2.7).

3.6. subscriptionCreate

Cette fonction permet de créer des abonnements a partir d'un identifiant.
Cette fonction prend en entrée les paramètres suivants :

Nom du champ	Type	Description	Obligatoire
createInfo	SubscriptionCreationInfo	Cf. 2.8	✓
wsSignature	String	Signature (cf. ci-dessous)	✓

Le calcul de la signature se fait en prenant les paramètres dans l'ordre suivant :
shopId, transmissionDate, subscriptionId, effectDate, paymentMethod, orderId, orderInfo1, orderInfo2, orderInfo3, amount, devise, initialAmount, occInitialAmount, subscriptionDescr, subscriptionDescrHR, manualValidation, cardIdent, contractNumber, ctxMode, comment

Cette fonction retourne une réponse du type **identResponse** (cf.2.2).

NB :
Le code d'erreur étendu (extendedErrorCode) et le statut de la transaction (transactionStatus) ne sont pas renseignés pour cette fonction.

3.7. subscriptionModify

Cette fonction permet de modifier les informations d'un abonnement.
Attention un abonnement ne peut être modifié si la date d'effet de ce dernier est dépassée. La date d'effet correspond au paramètre « effectDate » qui a été définit lors de la création d'une récurrence.
Cette fonction prend en entrée les paramètres suivants :

Nom du champ	Type	Description	Obligatoire
shopId	String	Identifiant de la boutique	✓
cardIdent	String	Identifiant du compte carte	✓
subscriptionId	String	Identifiant de l'abonnement à modifier	✓
modificationInfo	subscriptionModifyInfo	Cf. 2.9	✓
ctxMode	String	Contexte de sollicitation de la plateforme de paiement ("TEST", "PRODUCTION")	✓
wsSignature	String	Signature (cf. ci-dessous)	✓

Le calcul de la signature se fait en prenant les paramètres dans l'ordre suivant :
shopId, cardIdent, subscriptionId, modificationInfo.effectDate, modificationInfo.paymentMethod, modificationInfo.amount, modificationInfo.devise, modificationInfo.initialAmount, modificationInfo.occInitialAmount, modificationInfo.subscriptionDescr, modificationInfo.subscriptionDescrHR, modificationInfo.manualValidation, modificationInfo.contractNumber, ctxMode

Dans le champ modificationInfo, les champs non renseignés ne sont pas modifiés.
Cette fonction retourne une réponse du type **identResponse** (cf.2.2).

NB :
Le code d'erreur étendu (extendedErrorCode) et le statut de la transaction (transactionStatus) ne sont pas renseignés pour cette fonction.

3.8. subscriptionCancel

Cette fonction permet de résilier un abonnement à une date donnée.
Cette fonction prend en entrée les paramètres suivants :

Nom du champ	Type	Description	Obligatoire
shopId	String	Identifiant de la boutique	✓
cardIdent	String	Identifiant du compte carte	✓
subscriptionId	String	Identifiant de l'abonnement à résilier	✓
date	Date	Date de résiliation de l'identifiant	✓
ctxMode	String	Contexte de sollicitation de la plateforme de paiement ("TEST", "PRODUCTION")	✓
wsSignature	String	Signature (cf. ci-dessous)	✓

Le calcul de la signature se fait en prenant les paramètres dans l'ordre suivant :
shopId, cardIdent, subscriptionId, date, ctxMode

Cette fonction retourne une réponse du type **identResponse** (cf.2.2).

NB :
Le code d'erreur étendu (extendedErrorCode) et le statut de la transaction (transactionStatus) ne sont pas renseignés pour cette fonction.

3.9. subscriptionInfo

Cette fonction permet d'interroger un abonnement pour en connaître ses différents attributs.
Cette fonction prend en entrée les paramètres suivants :

Nom du champ	Type	Description	Obligatoire
shopId	String	Identifiant de la boutique	✓
cardIdent	String	Identifiant du compte carte	✓
subscriptionId	String	Identifiant de l'abonnement à consulter	✓
ctxMode	String	Contexte de sollicitation de la plateforme de paiement ("TEST", "PRODUCTION")	✓
wsSignature	String	Signature (cf. ci-dessous)	✓

Le calcul de la signature se fait en prenant les paramètres dans l'ordre suivant :
shopId, cardIdent, subscriptionId, ctxMode

Cette fonction retourne une réponse du type **subscriptionInfo** (cf.2.10).

3.10. identCreate

Cette fonction permet de créer un identifiant.
Cette fonction prend en entrée les paramètres suivants :

Nom du champ	Type	Description	Obligatoire
createInfo	identCreationInfo	Cf. 2.11	✓
wsSignature	String	Signature (cf. ci-dessous)	✓

Le calcul de la signature se fait en prenant les paramètres dans l'ordre suivant :
shopId, transmissionDate, cardIdent, paymentMethod, cardNumber, cardNetwork, cardExpirationDate, cvv, contractNumber, threeDsResult, customerId, customerTitle, customerName, customerPhone, customerMobilePhone, customerMail, customerAddress, customerZipCode, customerCity, customerState, customerCountry, customerLanguage, customerIP, ctxMode, comment

Note : si le paramètre threeDsResult est non renseigné, la valeur prise en compte dans le calcul de signature est vide (comme tous les champs non renseignés), par contre si il est renseigné il est calculé de la sorte : brand+enrolled+authStatus+eci+xid+cavv+cavvAlgorithm soit par exemple :
VISA+Y+Y++XidXidXidXidXidXidXidXidX+CavvCavvCavvCavvCavvCavvCavv+2

Cette fonction retourne une réponse du type **identResponse** (cf. 2.2).

NB :
En cas d'échec de la création de l'identifiant, le code d'erreur étendu (extendedErrorCode) sera valorisé avec le code retour d'autorisation (cf.2.14).

3.11. identUpdate

Cette fonction permet de mettre à jour un identifiant (y compris les données de cartes).
Cette fonction prend en entrée les paramètres suivants :

Nom du champ	Type	Description	Obligatoire
updateInfo	IdentCreationInfo	Cf. 2.11	✓
wsSignature	String	Signature (cf. ci-dessous)	✓

Le calcul de la signature se fait en prenant les paramètres dans l'ordre suivant :

shopId, transmissionDate, cardIdent, paymentMethod, cardNumber, cardNetwork, cardExpirationDate, cvv, contractNumber, threeDsResult, customerId, customerTitle, customerName, customerPhone, customerMobilePhone, customerMail, customerAddress, customerZipCode, customerCity, customerState, customerCountry, customerLanguage, customerIP, ctxMode, comment

Note : si le paramètre threeDsResult est non renseigné, la valeur prise en compte dans le calcul de signature est vide (comme tous les champs non renseignés), par contre si il est renseigné il est calculé de la sorte : brand+enrolled+authStatus+eci+xid+cavv+cavvAlgorithm soit par exemple :
VISA+Y+Y++XidXidXidXidXidXidXidXidX+CavvCavvCavvCavvCavvCavvCavv+2

Cette fonction retourne une réponse du type **identResponse** (cf.2.2).

NB :
En cas d'échec de la création de l'identifiant, le code d'erreur étendu (extendedErrorCode) sera valorisé avec le code retour d'autorisation (cf.2.14).

3.12. identCreateAndPay

Cette fonction permet de créer un identifiant.
Cette fonction prend en entrée les paramètres suivants :

Nom du champ	Type	Description	Obligatoire
identInfo	identCreationInfo	Cf. 2.11	✓
payInfo	createPaiementIdentInfo	Cf. 2.4	✓
wsSignature	String	Signature (cf. ci-dessous)	✓

Certaines informations sont dupliquées dans les différents champs, s'ils sont renseignés différemment, seule la première valeur sera prise en compte.

Le calcul de la signature se fait en prenant les paramètres dans l'ordre suivant :

identInfo.shopId, identInfo.transmissionDate, identInfo.cardIdent, identInfo.paymentMethod, identInfo.cardNumber, identInfo.cardNetwork, identInfo.cardExpirationDate, identInfo.cvv, identInfo.contractNumber, identInfo.threeDsResult, identInfo.customerId, identInfo.customerTitle, identInfo.customerName, identInfo.customerPhone, identInfo.customerMobilePhone, identInfo.customerMail, identInfo.customerAddress, identInfo.customerZipCode, identInfo.customerCity, identInfo.customerState, identInfo.customerCountry, identInfo.customerLanguage, identInfo.customerIP, identInfo.ctxMode, identInfo.comment, payInfo.shopId, payInfo.transmissionDate, payInfo.transactionId, payInfo.paymentMethod, payInfo.orderId, payInfo.orderInfo1, payInfo.orderInfo2, payInfo.orderInfo3, payInfo.amount, payInfo.devise, payInfo.presentationDate, payInfo.manualValidation, payInfo.cardIdent, payInfo.contractNumber, payInfo.ctxMode, payInfo.comment

Note : si le paramètre threeDsResult est non renseigné, la valeur prise en compte dans le calcul de signature est vide (comme tous les champs non renseignés), par contre si il est renseigné il est calculé de la sorte : brand+enrolled+authStatus+eci+xid+cavv+cavvAlgorithm soit par exemple :
VISA+Y+Y++XidXidXidXidXidXidXidXidX+CavvCavvCavvCavvCavvCavvCavv+2

Cette fonction retourne une réponse du type **identResponse** (cf.2.2).

NB :
En cas d'échec de la création de l'identifiant, le code d'erreur étendu (extendedErrorCode) sera valorisé avec le code retour d'autorisation (cf.2.14).

Exemple de fichier xml généré lors de l'appel de la méthode identCreateAndPay

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV='http://schemas.xmlsoap.org/soap/envelope/'
xmlns:ns1='http://v2.ident.ws.vads.lyra.com/'>
<SOAP-ENV:Body>
        <ns1:identCreateAndPay>
                <identInfo>
                        <shopId>48337286</shopId>
                        <transmissionDate>2013-01-18T15:48:59+00:00</transmissionDate>
                        <cardIdent>identifant-acheteur</cardIdent>
                        <paymentMethod>EC</paymentMethod>
                        <cardNumber>49700000000000</cardNumber>
                        <cardNetwork>VISA</cardNetwork>
                        <cardExpirationDate>2015-09-28T00:00:00+00:00</cardExpirationDate>
                        <cvv>123</cvv>
                        <customerId>ref-client</customerId>
                        <customerTitle>Mr</customerTitle>
                        <customerName>Nom</customerName>
                        <customerStatus>COMPANY</customerStatus>
                        <customerPhone>0123456789</customerPhone>
                        <customerMobilePhone>0612345678</customerMobilePhone>
                        <customerMail>test.systempay@gmail.com</customerMail>
                        <customerAddressNumber>8</customerAddressNumber>
                        <customerAddress>rue du test</customerAddress>
                        <customerZipCode>31000</customerZipCode>
                        <customerCity>Toulouse</customerCity>
                        <customerState>Haute-Garonne</customerState>
                        <customerCountry>FR</customerCountry>
                        <customerIP>127.0.0.1</customerIP>
                        <ctxMode>TEST</ctxMode>
                </identInfo>
                <payInfo>
                        <shopId>48337286</shopId>
                        <transmissionDate>2013-01-18T15:48:59+00:00</transmissionDate>
                        <transactionId>605399</transactionId>
                        <paymentMethod>EC</paymentMethod>
                        <orderId>ref-commande</orderId>
                        <amount>135</amount>
                        <devise>978</devise>
                        <manualValidation>0</manualValidation>
                        <cardIdent>identifant-acheteur</cardIdent>
                        <cvv>123</cvv>
                        <ctxMode>TEST</ctxMode>
                </payInfo>
                <wsSignature>7602291123d8481fa3f7de01028ec6f2376e546f</wsSignature>
</ns1:identCreateAndPay>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Calcul de signature correspondant :

48337286+20130118+identifant-acheteur+EC+4970100000000000+VISA+20150928+123+++ref-client+Mr+Nom+0123456789+0612345678+test.systempay@gmail.com+rue du test+31000+Toulouse+Haute-Garonne+FR++127.0.0.1+TEST++48337286+20130118+605399+EC+ref-commande++++135+978++0+identifant-acheteur++TEST++*certificat*

3.13. identCreateAndSubscribe

Cette fonction permet de créer un identifiant.
Cette fonction prend en entrée les paramètres suivants :

Nom du champ	Type	Description	Obligatoire
identInfo	identCreationInfo	Cf. 2.11	✓
subscriptionInfo	subscriptionCreationInfo	Cf.2.8	✓
wsSignature	String	Signature (cf ci-dessous)	✓

Certaines informations sont dupliquées dans les différents champs, s'ils sont renseignés différemment, seule la première valeur sera prise en compte.

Le calcul de la signature se fait en prenant les paramètres dans l'ordre suivant :

identInfo.shopId, identInfo.transmissionDate, identInfo.cardIdent, identInfo.paymentMethod, identInfo.cardNumber, identInfo.cardNetwork, identInfo.cardExpirationDate, identInfo.cvv, identInfo.contractNumber, identInfo.threeDsResult, identInfo.customerId, identInfo.customerTitle, identInfo.customerName, identInfo.customerPhone, identInfo.customerMobilePhone, identInfo.customerMail, identInfo.customerAddress, identInfo.customerZipCode, identInfo.customerCity, identInfo.customerState, identInfo.customerCountry, identInfo.customerLanguage, identInfo.customerIP, identInfo.ctxMode, identInfo.comment,subscriptionInfo.shopId, subscriptionInfo.transmissionDate, subscriptionInfo.subscriptionId, subscriptionInfo.effectDate, subscriptionInfo.paymentMethod, subscriptionInfo.orderId, subscriptionInfo.orderInfo1, subscriptionInfo.orderInfo2, subscriptionInfo.orderInfo3, subscriptionInfo.amount, subscriptionInfo.devise, subscriptionInfo.initialAmount, subscriptionInfo.occInitialAmount, subscriptionInfo.subscriptionDescr, subscriptionInfo.subscriptionDescrHR, subscriptionInfo.manualValidation, subscriptionInfo.cardIdent, subscriptionInfo.contractNumber, subscriptionInfo.ctxMode, subscriptionInfo.comment

Note : si le paramètre threeDsResult est non renseigné, la valeur prise en compte dans le calcul de signature est vide (comme tous les champs non renseignés), par contre si il est renseigné il est calculé de la sorte : brand+enrolled+authStatus+eci+xid+cavv+cavvAlgorithm soit par exemple :
VISA+Y+Y++XidXidXidXidXidXidXidXidX+CavvCavvCavvCavvCavvCavvCavv+2

Cette fonction retourne une réponse du type **identResponse** (cf.2.2).

NB :
En cas d'échec de la création de l'identifiant, le code d'erreur étendu (extendedErrorCode) sera valorisé avec le code retour d'autorisation (cf.2.14).

Exemple de fichier xml généré lors de l'appel de la méthode identCreateAndSubscribe

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ns1="http://v2.ident.ws.vads.lyra.com/">
<SOAP-ENV:Body>
	<ns1:identCreateAndSubscribe>
		<createInfo>
			<shopId>48337286</shopId>
			<transmissionDate>2013-01-18T16:08:02+00:00</transmissionDate>
			<paymentMethod>EC</paymentMethod>
			<cardNumber>4970100000000000</cardNumber>
			<cardNetwork>CB</cardNetwork>
			<cardExpirationDate>2017-10-28T00:00:00+00:00</cardExpirationDate>
			<cvv>123</cvv>
			<customerId>ref-client</customerId>
			<customerTitle>Mr</customerTitle>
			<customerName>Nom</customerName>
			<customerStatus>COMPANY</customerStatus>
			<customerPhone>0123456789</customerPhone>
			<customerMobilePhone>0612345678</customerMobilePhone>
			<customerMail>test.systempay@gmail.com</customerMail>
			<customerAddressNumber>8</customerAddressNumber>
			<customerAddress>rue du test</customerAddress>
			<customerZipCode>31000</customerZipCode>
			<customerCity>Toulouse</customerCity>
			<customerState>Haute-Garonne</customerState>
			<customerCountry>FR</customerCountry>
			<customerIP>127.0.0.1</customerIP>
			<ctxMode>TEST</ctxMode>
		</createInfo>
		<subscriptionInfo>
			<shopId>48337286</shopId>
			<transmissionDate>2013-01-18T16:08:02+00:00</transmissionDate>
			<effectDate>2013-01-31T16:08:03+00:00</effectDate>
			<paymentMethod>EC</paymentMethod>
			<orderId>ref-commande</orderId>
			<amount>3490</amount>
			<devise>978</devise>
			<initialAmount>2990</initialAmount>
			<occInitialAmount>6</occInitialAmount>
				<subscriptionDescr>RRULE:FREQ=MONTHLY;BYMONTHDAY=18;COUNT=24</subscriptionDescr>
			<subscriptionDescrHR>29.90€ TTC/ mois pendant 6 mois puis 34.90€ TTC/ mois. Durée d\'engagement de
24mois</subscriptionDescrHR>
			<manualValidation>0</manualValidation>
			<ctxMode>TEST</ctxMode>
		</subscriptionInfo>
		<wsSignature>520201672926ac86a11173755a03afcd9244b6f2</wsSignature>
	</ns1:identCreateAndSubscribe>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Calcul de signature correspondant :

48337286+20130118++EC+4970100000000000+CB+20171028+123+++ref-client+Mr+Nom+0123456789+0612345678+test.systempay@gmail.com+rue du test+31000+toulouse+Haute-Garonne+FR++127.0.0.1+TEST++48337286+20130118++20130131+EC+ref-commande++++3490+978+2990+6+RRULE:FREQ=MONTHLY;BYMONTHDAY=18;COUNT=24+29.90€ TTC/ mois pendant 6 mois puis 34.90€ TTC/ mois. Durée d\'engagement de 24mois+0+++TEST++*certificat*

3.14. identCreatePayAndSubscribe

Cette fonction permet de créer un identifiant.
Cette fonction prend en entrée les paramètres suivants :

Nom du champ	Type	Description	Obligatoire
identInfo	IdentCreationInfo	cf. 2.11	✓
payInfo	CreatePaiementIdentInfo	cf. 2.4	✓
subscriptionInfo	SubscriptionCreationInfo	cf. 2.8	✓
wsSignature	String	Signature (cf. ci-dessous)	✓

Certaines informations sont dupliquées dans les différents champs, s'ils sont renseignés différemment, seule la première valeur sera prise en compte.

Le calcul de la signature se fait en prenant les paramètres dans l'ordre suivant :

identInfo.shopId, identInfo.transmissionDate, identInfo.cardIdent, identInfo.paymentMethod, identInfo.cardNumber, identInfo.cardNetwork, identInfo.cardExpirationDate, identInfo.cvv, identInfo.contractNumber, identInfo.threeDsResult, identInfo.customerId, identInfo.customerTitle, identInfo.customerName, identInfo.customerPhone, identInfo.customerMobilePhone, identInfo.customerMail, identInfo.customerAddress, identInfo.customerZipCode, identInfo.customerCity, identInfo.customerState, identInfo.customerCountry, identInfo.customerLanguage, identInfo.customerIP, identInfo.ctxMode, identInfo.comment, payInfo.shopId, payInfo.transmissionDate, payInfo.transactionId, payInfo.paymentMethod, payInfo.orderId, payInfo.orderInfo1, payInfo.orderInfo2, payInfo.orderInfo3, payInfo.amount, payInfo.devise, payInfo.presentationDate, payInfo.manualValidation, payInfo.cardIdent, payInfo.contractNumber, payInfo.ctxMode, payInfo.comment, subscriptionInfo.shopId, subscriptionInfo.transmissionDate, subscriptionInfo.subscriptionId, subscriptionInfo.effectDate, subscriptionInfo.paymentMethod, subscriptionInfo.orderId, subscriptionInfo.orderInfo1, subscriptionInfo.orderInfo2, subscriptionInfo.orderInfo3, subscriptionInfo.amount, subscriptionInfo.devise, subscriptionInfo.initialAmount, subscriptionInfo.occInitialAmount, subscriptionInfo.subscriptionDescr, subscriptionInfo.subscriptionDescrHR, subscriptionInfo.manualValidation, subscriptionInfo.cardIdent, subscriptionInfo.contractNumber, subscriptionInfo.ctxMode, subscriptionInfo.comment

Note : si le paramètre threeDsResult est non renseigné, la valeur prise en compte dans le calcul de signature est vide (comme tous les champs non renseignés), par contre si il est renseigné il est calculé de la sorte : brand+enrolled+authStatus+eci+xid+cavv+cavvAlgorithm soit par exemple :
VISA+Y+Y++XidXidXidXidXidXidXidXidX+CavvCavvCavvCavvCavvCavvCavv+2

Cette fonction retourne une réponse du type **identResponse** (cf.2.2).

NB :
En cas d'échec de la création de l'identifiant, le code d'erreur étendu (extendedErrorCode) sera valorisé avec le code retour d'autorisation (cf.2.14).

Exemple de fichier xml généré lors de l'appel de la méthode identCreatePayAndSubscribe.

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV='http://schemas.xmlsoap.org/soap/envelope/' xmlns:ns1='http://v2.ident.ws.vads.lyra.com/'>
<SOAP-ENV:Body>
	<ns1:identCreatePayAndSubscribe>
		<identInfo>
			<shopId>48337286</shopId>
			<transmissionDate>2013-01-18T16:32:14+00:00</transmissionDate>
			<paymentMethod>EC</paymentMethod>
			<cardNumber>4970100000000000</cardNumber>
			<cardNetwork>CB</cardNetwork>
			<cardExpirationDate>2019-10-28T00:00:00+00:00</cardExpirationDate>
			<cvv>123</cvv>
			<customerId>ref-client</customerId>
			<customerTitle>Mr</customerTitle>
			<customerName>Nom</customerName>
			<customerStatus>PRIVATE</customerStatus>
			<customerPhone>0123456789</customerPhone>
			<customerMobilePhone>0612345678</customerMobilePhone>
			<customerMail>test.systempay@gmail.com</customerMail>
			<customerAddressNumber>8</customerAddressNumber>
			<customerAddress>rue du test</customerAddress>
			<customerZipCode>31000</customerZipCode>
			<customerCity>Toulouse</customerCity>
			<customerState>Haute-Garonne</customerState>
			<customerCountry>FR</customerCountry>
			<customerIP>127.0.0.1</customerIP>
			<ctxMode>TEST</ctxMode>
		</identInfo>
		<payInfo>
			<shopId>48337286</shopId>
			<transmissionDate>2013-01-18T16:32:14+00:00</transmissionDate>
			<transactionId>631347</transactionId>
			<paymentMethod>EC</paymentMethod>
			<orderId>ref-cmd</orderId>
			<amount>5000</amount>
			<devise>978</devise>
			<manualValidation>0</manualValidation>
			<ctxMode>TEST</ctxMode>
		</payInfo>
		<subscriptionInfo>
			<shopId>48337286</shopId>
			<transmissionDate>2013-01-18T16:32:14+00:00</transmissionDate>
			<effectDate>2013-01-18T16:32:14+00:00</effectDate>
			<paymentMethod>EC</paymentMethod>
			<orderId>ref-cmd</orderId>
			<amount>3490</amount>
			<devise>978</devise>
			<initialAmount>2990</initialAmount>
			<occInitialAmount>6</occInitialAmount>
			<subscriptionDescr>RRULE:FREQ=MONTHLY;BYMONTHDAY=18;COUNT=24</subscriptionDescr>
			<subscriptionDescrHR>29.90€ TTC/ mois pendant 6 mois puis 34.90€ TTC/ mois. Durée d\'engagement de
24mois</subscriptionDescrHR>
			<manualValidation>0</manualValidation>
			<ctxMode>TEST</ctxMode>
		</subscriptionInfo>
		<wsSignature>668f0e40a754f42baa438c390f8d1792677ec70c</wsSignature>
	</ns1:identCreatePayAndSubscribe>
</SOAP-ENV:Body></SOAP-ENV:Envelope>
```

Calcul de signature correspondant :

48337286+20130118++EC+497010XXXXXX0000+CB+20191028+XXX+++ref-client+Mr+Nom+0123456789+0612345678+test. systempay@gmail.com+rue du test+31000+Toulouse+Haute-Garonne+FR++127.0.0.1+TEST++48337286+20130118+631347+EC+ref-cmd++++5000+978++0+++TEST++48337286+20130118++20130118+EC+ref-cmd++++3490+978+2990+6+RRULE:FREQ=MONTHLY;BYMONTHDAY=18;COUNT=24+29.90€ TTC/ mois pendant 6 mois puis 34.90€ TTC/ mois. Durée d\'engagement de 24mois +0+++TEST++*certificat*

4. Signature

Un certificat est nécessaire pour dialoguer avec la plateforme de paiement. Il est mis à disposition de toutes les personnes habilitées à la consultation des certificats dans votre outil de gestion de caisse à l'emplacement suivant : Paramètres / Boutique / Certificat. Il existe deux certificats différents : un pour la plateforme de test et un pour la plateforme de production.

La signature sera générée comme suit :

- Création d'une chaîne de caractère représentant la concaténation des paramètres, séparés par le caractère "+".
- Ajout à cette chaîne d'un "certificat " numérique (de test ou de production selon le contexte).
- Hachage de la chaîne résultante avec l'algorithme SHA1.

La plateforme de paiement effectuera obligatoirement la vérification de la signature. Il est de la responsabilité du commerçant de vérifier à son tour la signature transmise en retour.

- L'ordre des champs doit être respecté.
- Les champs de type **numérique** ne doivent pas avoir de 0 à gauche du digit le plus significatif.
- Les champs de type **bool** prennent les valeurs suivantes :
  - 1 pour vrai (true)
  - 0 pour faux (false)
- Les champs de type **String** non renseignés seront vides.
- Les champs de type **dateTime** doivent être formatés de la manière suivante : **YYYYMMDD** en **UTC**.

Exemple pour le 1er Février 2012 : 20120201.

Attention aux heures, par exemple le 23/02/2012 00:30:00 heure française (GMT+1 ou GMT+2) donne 20100222.

En mode TEST, en cas de mauvais calcul de signature, le code erreur de la fonction renvoie 5, la chaine de caractère utilisée pour la signature côté serveur est alors renvoyée dans le champ **extendedErrorCode**.

---

5. Maintien de la session HTTP entre 2 requêtes.

Important :

L'architecture de la plateforme de paiement reposant sur un ensemble de serveurs avec répartition de charge, il est nécessaire que chaque requête concernant un même paiement dans un laps de temps très court, soient réalisées avec la même session HTTP afin d'assurer la continuité du processus.

Pour cela, à chaque requête, une session est créée coté serveur.
L'ID de la session est renvoyé dans l'entête HTTP de la réponse. Il devra être retourné dans les requêtes suivantes afin que la requête soit traitée par le même serveur, évitant ainsi à votre requête d'être rejetée car la transaction ne serait pas encore disponible sur les autres serveurs.

5.1. Exemple d'application

Vous souhaitez créer un paiement à remettre dans 30 jours en mode de validation manuelle. Une fois le paiement accepté vous décidez de changer la date de remise pour le lendemain et de valider la transaction.

- Vous appelez le web service de création de paiement (create).
- La plateforme vérifie la présence d'un ID de session dans l'en-tête HTTP de votre requête. Comme rien n'a été précisé, une nouvelle session et un nouvel ID son créés. Systempay procède ensuite au traitement de votre requête et envoie sa réponse en indiquant dans l'en-tête HTTP l'identifiant de session attribué ainsi que le nom du serveur ayant traité la requête:

Exemple d'entête de requête :

```
POST /vads-ws/v2.1 HTTP/1.1
Host: paiement.systempay.fr
Connection: Keep-Alive
User-Agent: PHP-SOAP/5.2.10
Content-Type: text/xml; charset=utf-8
SOAPAction: ""
Content-Length: 482
```

Exemple d'entête de réponse :

```
HTTP/1.1 200 OK
Date: Thu, 25 Oct 2012 09:48:06 GMT
Server: Apache
Set-Cookie: JSESSIONID=00A19F16AD158A3C4862EBB84896E9FE.sirisvad2; Path=/vads-ws; Secure; HttpOnly
Access-Control-Allow-Origin: *
Vary: Accept-Encoding
Connection: close
Transfer-Encoding: chunked
Content-Type: text/xml;charset=UTF-8
```

- Dans les en-têtes HTTP de la réponse, vous récupérez le cookie **JSESSIONID** :
- Vous initialisez le cookie JSESSIONID de votre en-tête HTTP.
- Vous appelez le web service de modification et de validation du paiement (modifyAndValidate).

Exemple d'entête de requête :

```
POST /vads-ws/v3 HTTP/1.1
Host: paiement.systempay.fr
Connection: Keep-Alive
User-Agent: PHP-SOAP/5.2.10
Content-Type: text/xml; charset=utf-8
SOAPAction: ""
Content-Length: 482
Cookie: JSESSIONID=00A19F16AD158A3C4862EBB84896E9FE.sirisvad2;
```

- La plateforme vérifie la présence d'un ID de session dans l'en-tête HTTP de votre requête.
  La requête est ensuite envoyée au serveur ayant généré la session.
  Si la session existe, elle est réutilisée, sinon une nouvelle session et un nouvel identifiant seront créés.
  Systempay procède au traitement de la requête et envoie sa réponse.

Remarque :
L'ID de session sera renvoyé dans l'en-tête HTTP de la réponse uniquement dans le cas où une nouvelle session a été créée.
Il est donc conseillé de tester systématiquement la présence du cookie dans l'en-tête HTTP de la réponse et d'utiliser l'ID de session présent avant d'enchainer un autre appel (getInfo par exemple).

Exemple d'en-tête HTTP de réponse utilisant la même session:

```
HTTP/1.1 200 OK
Date: Thu, 25 Oct 2012 09:48:07 GMT
Server: Apache
Secure; HttpOnly
Access-Control-Allow-Origin: *
Vary: Accept-Encoding
Connection: close
Transfer-Encoding: chunked
Content-Type: text/xml;charset=UTF-8
```

(Pas d'information sur l'identifiant de session)

Exemple d'en-tête HTTP de réponse avec une nouvelle session:

```
HTTP/1.1 200 OK
Date: Thu, 25 Oct 2012 09:48:06 GMT
Server: Apache
Set-Cookie: JSESSIONID=F6D2CCC8B075077843724ADDDE887EE2.sirisvad2; Path=/vads-
ws; Secure; HttpOnly
Access-Control-Allow-Origin: *
Vary: Accept-Encoding
Connection: close
Transfer-Encoding: chunked
Content-Type: text/xml;charset=UTF-8
```

(Présence du Set-Cookie précisant l'identifiant de la nouvelle session)

5.2. Exemple d'implémentation en PHP

Pour récupérer l'en-tête HTTP de la réponse vous devez utiliser la fonction __getLastResponseHeaders()

```
/* La méthode ci-dessous permet de récupérer l'entête HTTP de la réponse */
/* $client étant une instance du client SOAP utilisé pour appeler les WS */
$header = $client->__getLastResponseHeaders();
```

Pour récupérer ensuite l'identifiant de session vous pouvez utiliser le code suivant :

```
/* Dans la chaîne de caractère obtenue, nous recherchons la présence de l'ID de la
session HTTP, stockée dans l'élément "JSESSIONID" : */

if(preg_match("#JSESSIONID=([A-Za-z0-9\.]+)#",$header, $matches)){
      $JSESSIONID = $matches[1];;
}
```

Pour initialiser le cookie vous devez utiliser la fonction __setCookie()

```
$cookie= $JSESSIONID;
$client->__setCookie('JSESSIONID', $cookie);
```