



## **Tests fonctionnels**

## **Carte Blanche Partenaires**

### **Manuel utilisateur à l'usage des Utilisateurs de SoapUI**

### **Guide pratique**

#### **Attention :**

***Il est de la responsabilité de chaque utilisateur de s'assurer que la version qu'il utilise est à jour.***

<b>Auteur</b>	<b>Slaheddine BEJAoui</b>
<b>Version</b>	<b>V 1.1</b>
<b>Entité propriétaire</b>	<b>Carte Blanche Partenaires</b>

***Ce document a pour but de décrire la procédure afin de tester la conformité des réponses de services web par rapport à l'attendu.***

### ***Éléments nécessaires :***

***Fichier WSDL : ce fichier décrit la structure nécessaire devant être respecté par les fichiers XML.***

***Requête XML : contenu XML utilisé pour les tests.***

***Soap : protocole orienté service.***

***URL : le lien vers le serveur vers lequel tester l'envoi***

# SOMMAIRE

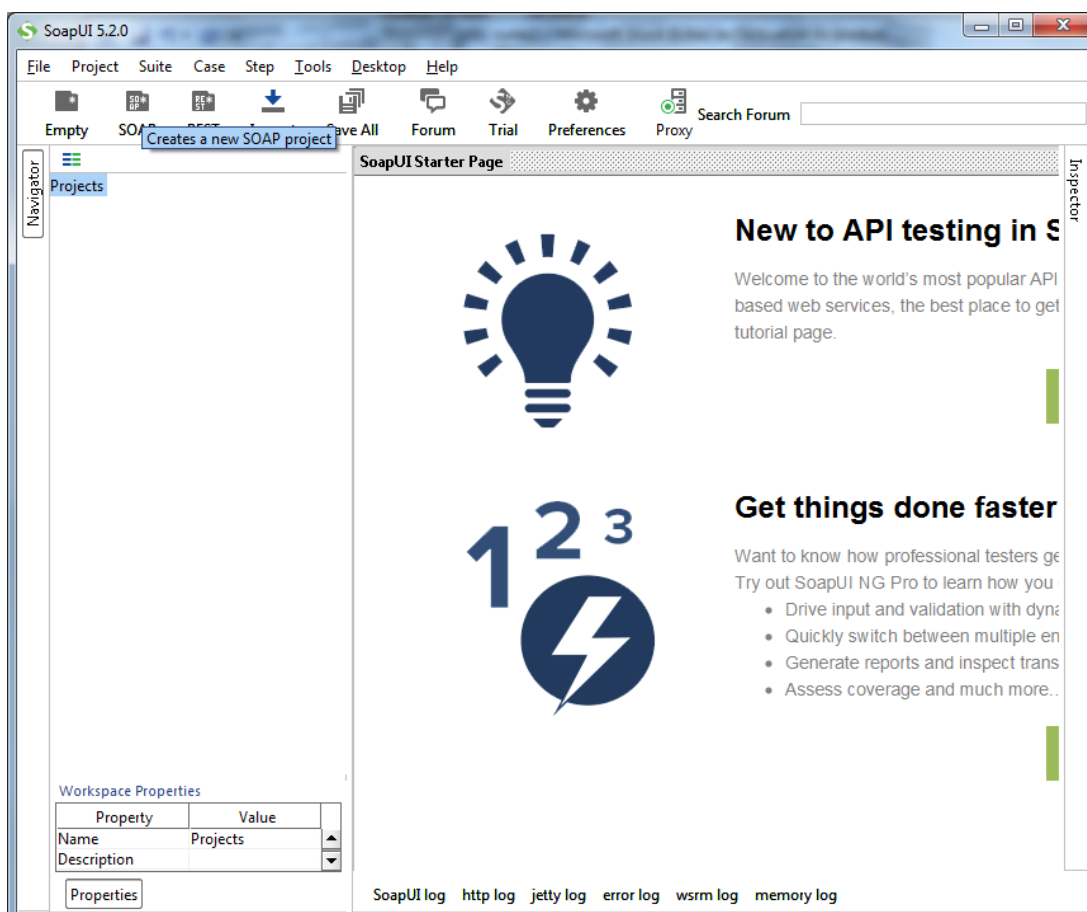
<b>I. PRE-REQUIS .....</b>	<i>Erreur ! Signet non défini.</i>
<b>II. DEBUTER LES TESTS.....</b>	<i>Erreur ! Signet non défini.</i>
<b>III. UTILISATION DU SCRIPT .....</b>	<i>Erreur ! Signet non défini.</i>
<b>IV. POUR ALLER PLUS LOIN.....</b>	<i>Erreur ! Signet non défini.</i>

## I. Pré-requis

SoapUI est un logiciel open source qui permet de tester des webservices dans une architecture orientées services. Il va nous servir à réaliser nos tests fonctionnels dans le cadre de la phase de recette.

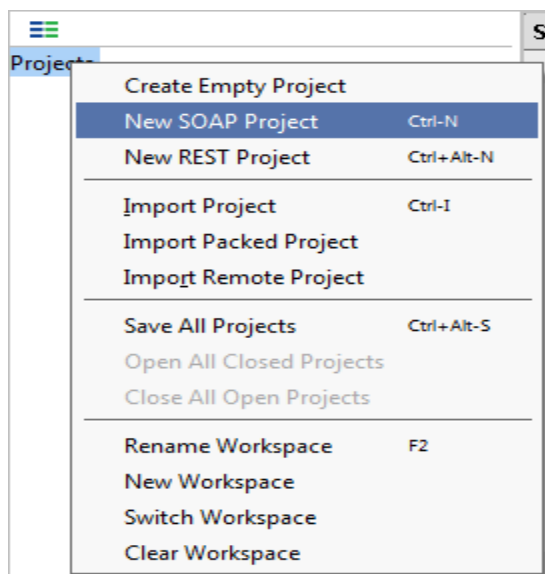
Les projets sous SoapUI sont indispensables à la création de tests. Une fois que le projet créé, vous pouvez créer et dérouler vos tests fonctionnels. Mais nous pouvons faire bien plus avec ce logiciel tel que des tests de charge, des tests de conformité ainsi que des tests de simulation et des tests unitaire notamment à travers les « **MockServices** ». , Mais dans ce tutoriel nous verrons la création de projet Soap et l'ajout d'un WSDL.

### Etape 1 : Lancer SoapUI



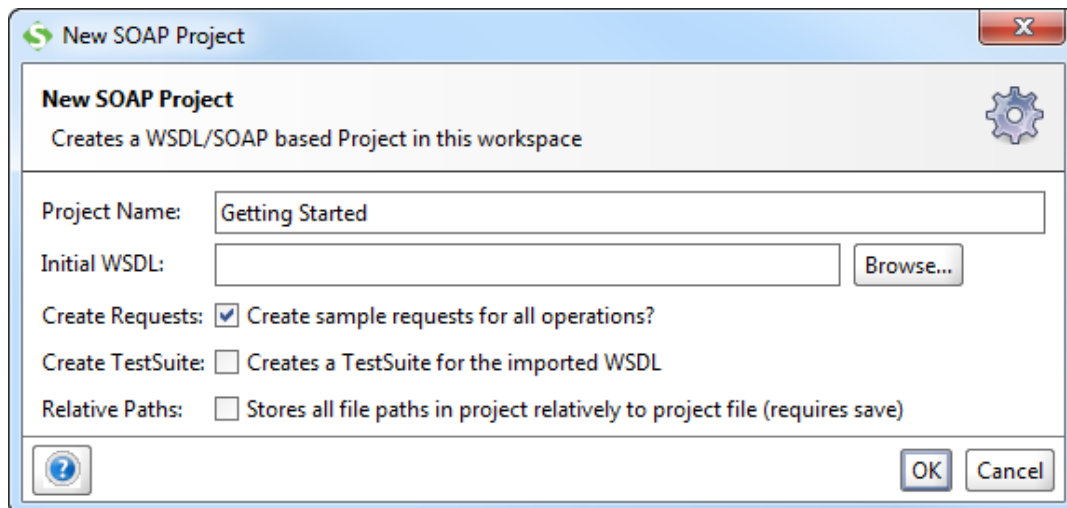
## Etape 2 : Créer un nouveau projet SOAP

1. Dans la barre de navigation, clic droit sur votre espace de travail et sélectionner « **New SOAP Project** » (ou *ctrl-n*)

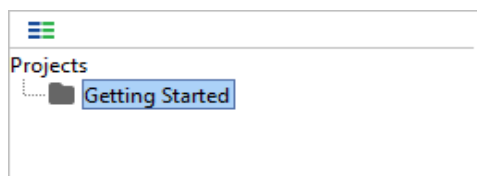


Une fenêtre *New SOAP Project* s'ouvre.

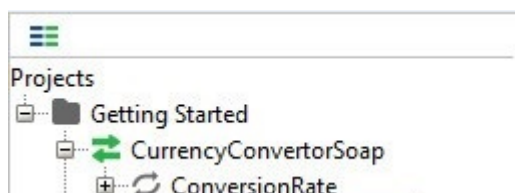
2. Renseignez alors votre nom de projet et charger éventuellement votre url de description du service web (WSDL).



3. Votre projet vient d'être créé



4. Patienter pendant le temps de chargement puis vous devriez voir des flèches vertes apparaître symbolisant que le WSDL a bien été pris en compte.



Vous pouvez double-cliquer sur le dossier de votre projet pour avoir une vue d'ensemble sur celui-ci. Elle vous donnera le chemin de votre projet, le nom du WSDL, un résumé des types de tests que vous avez créé.

**Getting Started**

Overview | TestSuites | WS-Security Configurations | Security Scan Defaults

☒ **Project Summary**

File Path

☒ **Interface Summary**

CurrencyConvertorSoap	<a href="http://www.webservice.com/currencyconvertor.asmx?WSDL">http://www.webservice.com/currencyconvertor.asmx?WSDL</a>
CurrencyConvertorSoap12	<a href="http://www.webservice.com/currencyconvertor.asmx?WSDL">http://www.webservice.com/currencyconvertor.asmx?WSDL</a>

☒ **Test Summary**

TestSuites	0
TestCases	0
TestSteps	0
Assertions	0
LoadTests	0

☒ **SOAP Mock Summary**

WsdIMockServices	0
WsdIMockOperations	0
WsdIMockResponses	0

☒ **REST Mock Summary**

RestMockServices	0
RestMockActions	0
RestMockResponses	0

Mais surtout, vous pourrez créer votre script « **Groovy** » en cliquant sur l'onglet **Load Script** situé en bas de la fenêtre. Ce script se lancera à chaque lancement du projet.

▶ Edit ▼

```

1
2 def testSuiteList = project.getTestSuites()
3 def text = "~"
4 def n=0
5 def OK = false
6
7 def groovyUtils = new com.eviware.soapui.support.GroovyUtils(context)
8 def csvFilePath = "Z:/SoapUI_AudioAMC/test2.csv"
9 context.fileReader = new BufferedReader(new FileReader(csvFilePath))
10
11 rowsData = context.fileReader.readLine()
12 int rowSize = rowsData.size()
13 def libelleKO = new ArrayList();
14

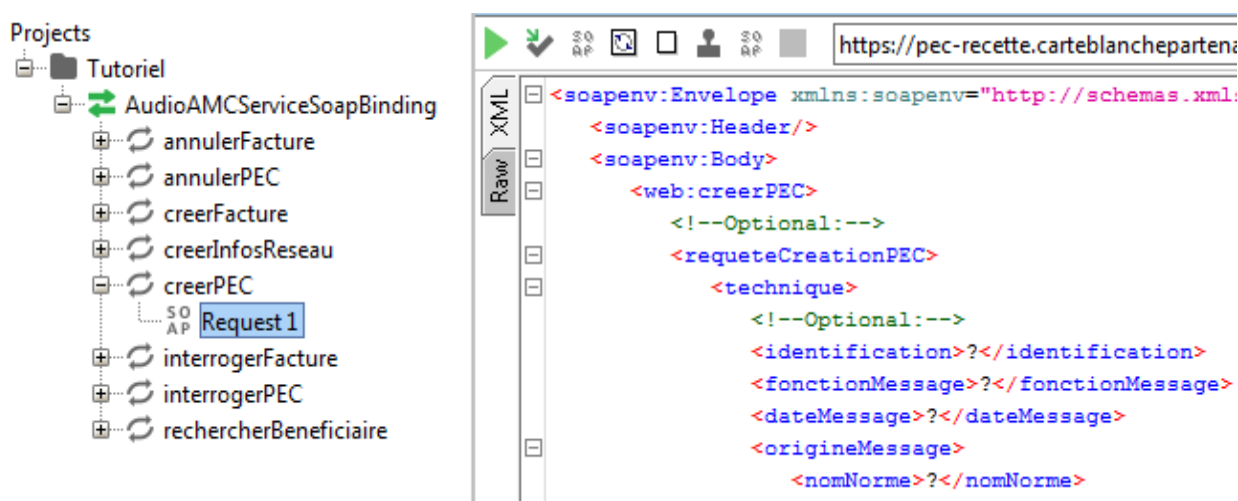
```

Description Properties **Load Script** Save Script

## II. Débuter les tests

### Etape 1 : Créer un modèle de flux

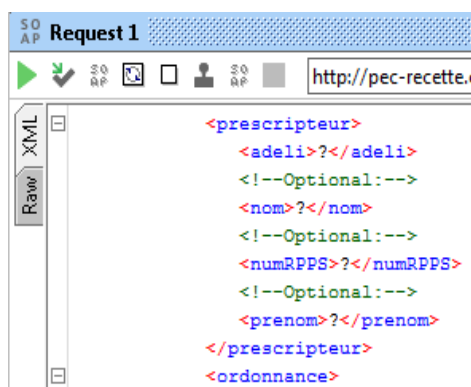
1. Cliquer sur le signe « + » pour faire apparaître la requête XML puis cliquer sur « **Request 1** », la fenêtre d'édition des requêtes s'ouvre :



2. Maintenant, formater le jeu de données avec les valeurs souhaitées. Certains champs de cette fenêtre sont remplis par défaut.



3. L'**URL** indiqué (encadré) doit être modifié par celui qui vous a été fourni afin de pointer vers le bon serveur.

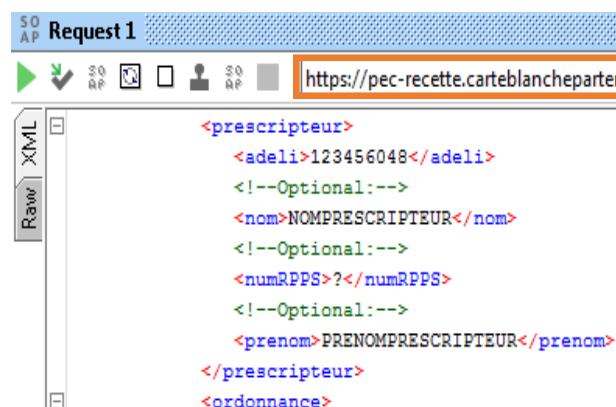


```

Request 1
http://pec-recette.
XML
Raw
<prescripteur>
  <adeli>?</adeli>
  <!--Optional:-->
  <nom>?</nom>
  <!--Optional:-->
  <numRPPS>?</numRPPS>
  <!--Optional:-->
  <prenom>?</prenom>
</prescripteur>
<ordonnance>

```

EXTRAIT D'UNE REQUETE AVANT MODIFICATION



```

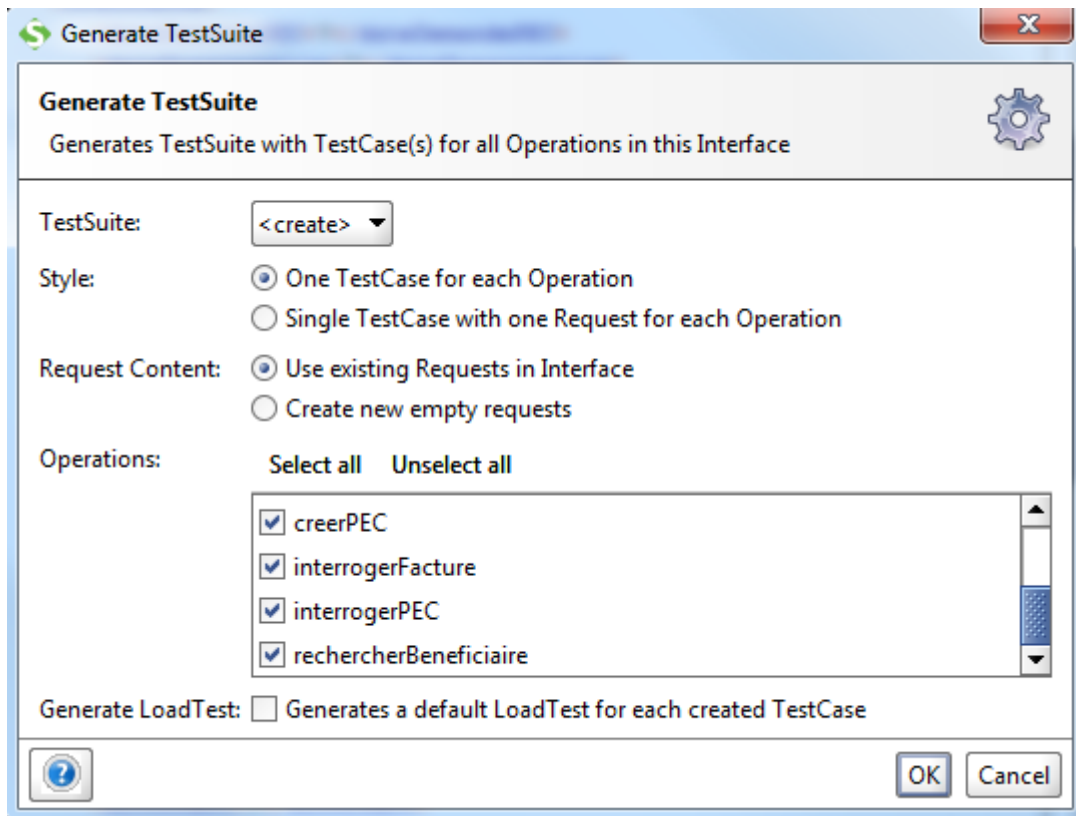
Request 1
https://pec-recette.cartelblancheparter
XML
Raw
<prescripteur>
  <adeli>123456048</adeli>
  <!--Optional:-->
  <nom>NOMPRESCRIPTEUR</nom>
  <!--Optional:-->
  <numRPPS>?</numRPPS>
  <!--Optional:-->
  <prenom>PRENOMPRESCRIPTEUR</prenom>
</prescripteur>
<ordonnance>

```

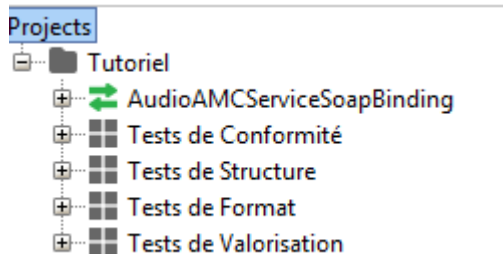
EXTRAIT D'UNE REQUETE APRES MODIFICATION

## Etape 2 : Organiser les tests par catégorie

1. Cliquez droit sur le WSDL puis cliquez sur « **generate TestSuite** ».
2. Cochez « **One TestCase for each Operation** » et « **Use existing Requests in Interface** » comme ci-dessous :



Vous pouvez ensuite choisir le type de flux à inclure dans la suite de tests.



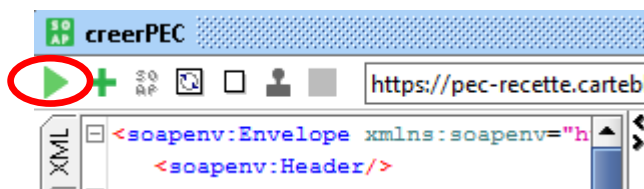
3. Recommencer cette manipulation autant de fois que vous avez de catégorie de tests.

### Etape 3 : Tester les requêtes XML

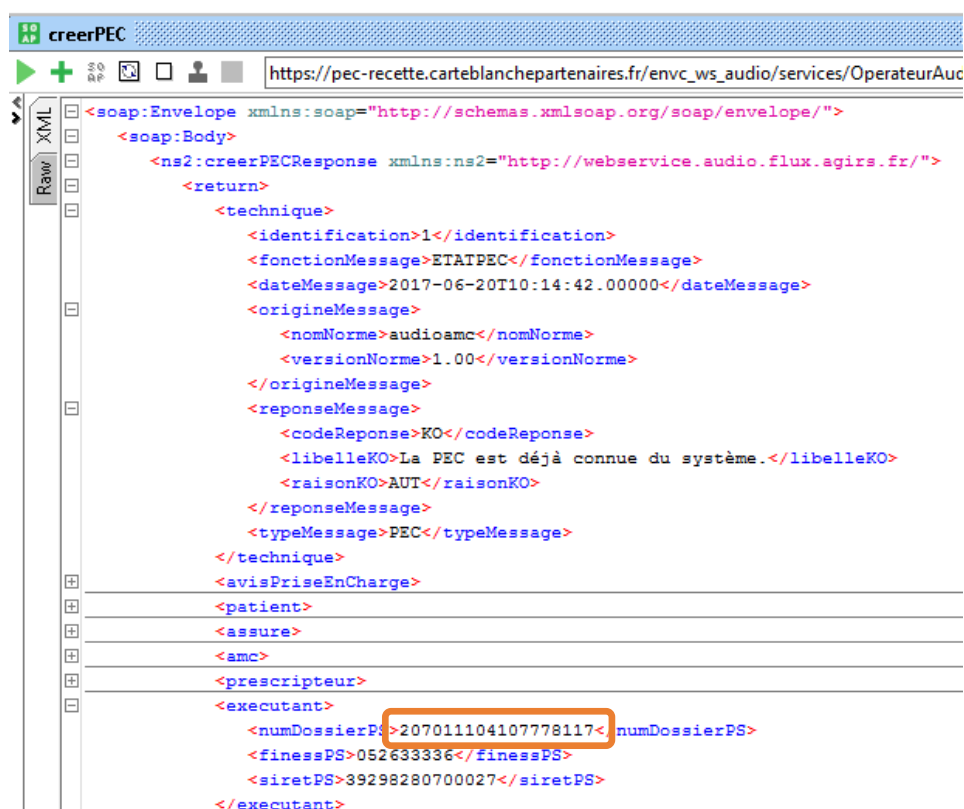
1. Maintenant il ne reste plus qu'à créer vos jeux de données de tests.

Une fois que vous aurez créé tout ce dont vous avez besoin, vous pourrez dès à présent tester vos flux :

2. Pour cela cliquer sur le triangle vert :



La requête est envoyée et le serveur nous envoie un message indiquant si elle est passée ou non dans la partie droite de la fenêtre.



Ici nous remarquons que le codeReponse est « **KO** » c'est-à-dire que le flux n'a pas bien fonctionné. En effet, il n'a pas fonctionné car la PEC créer existe déjà, ce qui nous amène au point 3 :

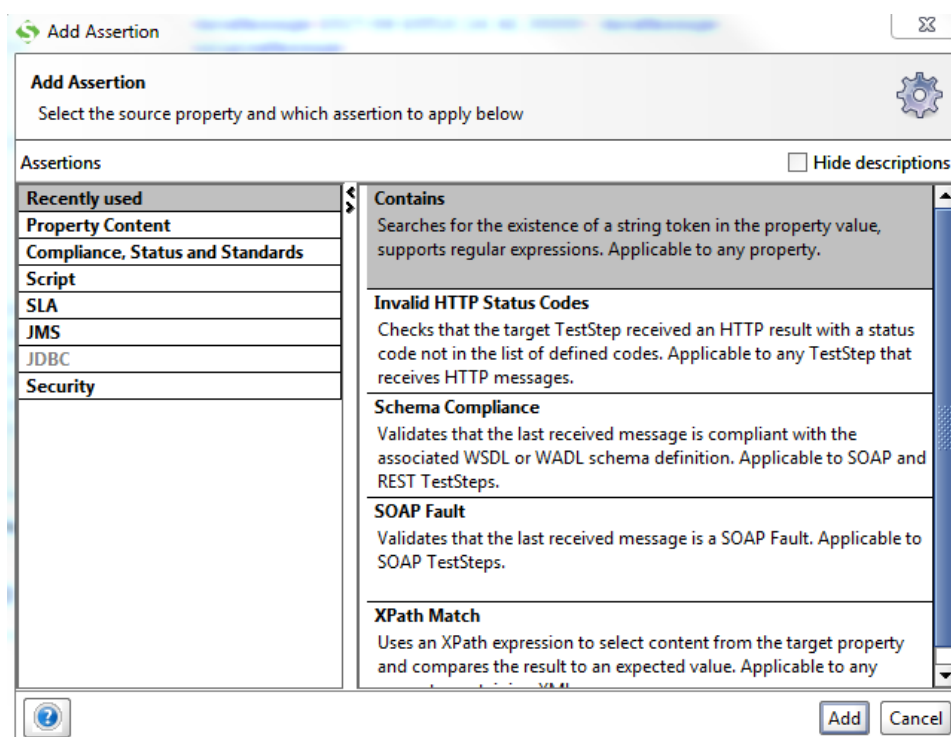
3. Il faut donc changer le **numero de dossier** (« **numDossierPS** »).

#### Etape 4 : Création des mini-tests

Vous pourrez créer vos mini-tests (« **assertions** »). Ici, nous utiliserons l'assertion « **Contains** », il nous servira à tester l'existence d'une valeur dans la réponse XML. Nous pouvons créer par exemple l'assertion du mot « **faultstring** » pour vérifier qu'une erreur de format d'une des balises est bien retournée lorsque le format d'une balise est incorrecte :

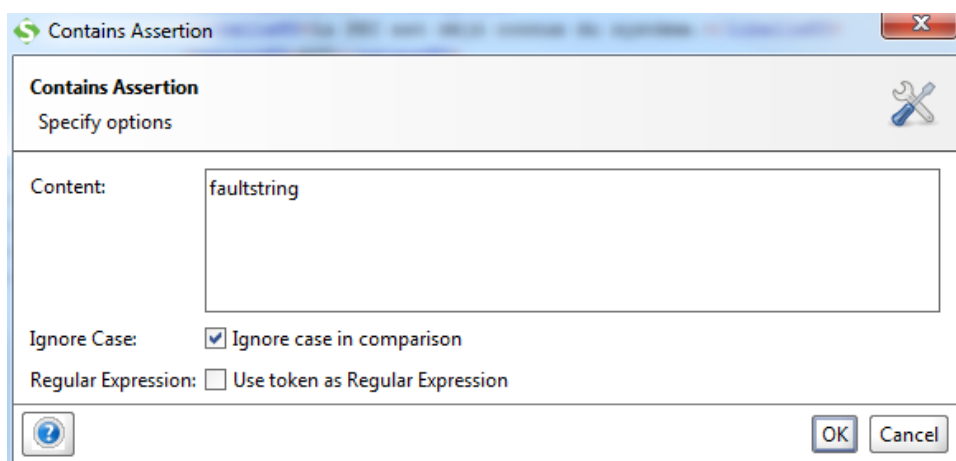


1. Pour cela, cliquer sur le signe « + » en vert puis cliquer sur contains :




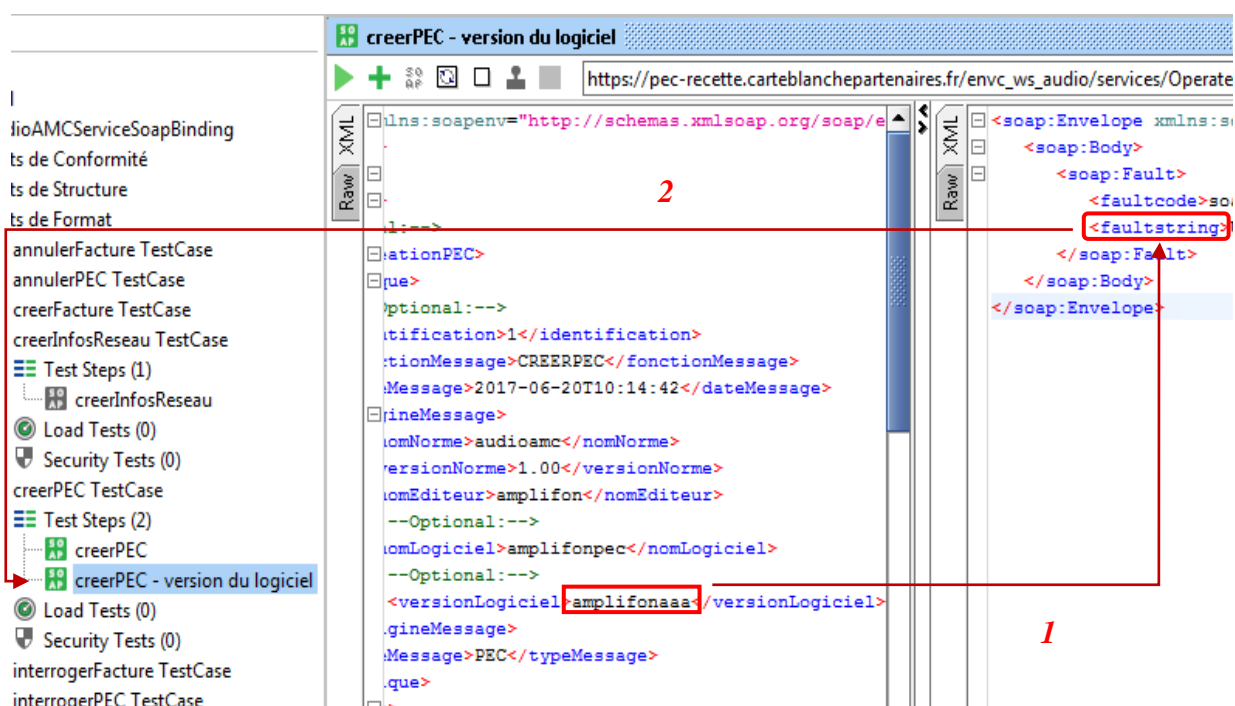
Si l'assertion existe déjà le logiciel vous propose de la renommer sinon il vous ouvre une fenêtre.

2. Renseignez la valeur que vous voulez tester et cocher sur « **Ignore case in comparison** ».



3. Valider en cliquant sur « **OK** ».

Si l'assertion est vérifiée, c'est-à-dire que notre exemple « **faultstring** » est contenu dans la réponse XML, l'icône  s'allumera en vert, sinon il sera rouge.



IV.

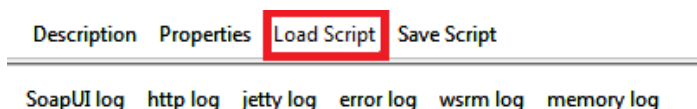
Ici, une erreur de format est détectée car le nombre de caractère maximum pour la version de logiciel a été définis à 10 et le nombre de caractère présent est 11.

## V. Utilisation du script

### Etape 1 : Lancement du script pour le test des flux AudioAMC

Le script concerne à la fois les tests de formatage, de structure et de valorisation. Les tests de structure concernent les tests de présence des balises.

1. Tout d'abord, cliquer sur « **Load Script** » situé en bas de la fenêtre.



Une zone d'édition de texte apparaît, vous avez juste à copier-coller le contenu du fichier « **Script\_groovy\_finale.groovy** » dans cette zone :



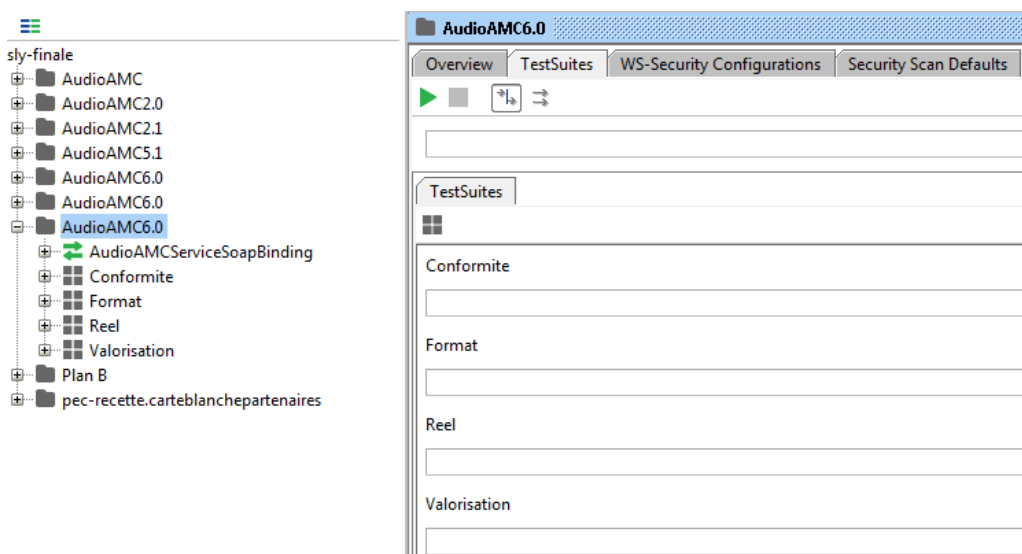
2. Verifier que le chemin du fichier d'erreur est bien celui ou vous avez créer le fichier (ex : « **Z:/SoapUI\_AudioAMC/LibelleErreur-SoapUI.csv** »)

3. Maintenant, il ne vous reste plus qu'à lancer le script en cliquant sur le triangle vert.

## Etape 2 : Lancement de la création des mini-tests

Tout les mini-tests seront ajoutés, maintenant il ne vous reste plus qu'à dérouler tout les tests. Mais avant pensez à changer **tout les numeros de dossiers** (voir astuce) .

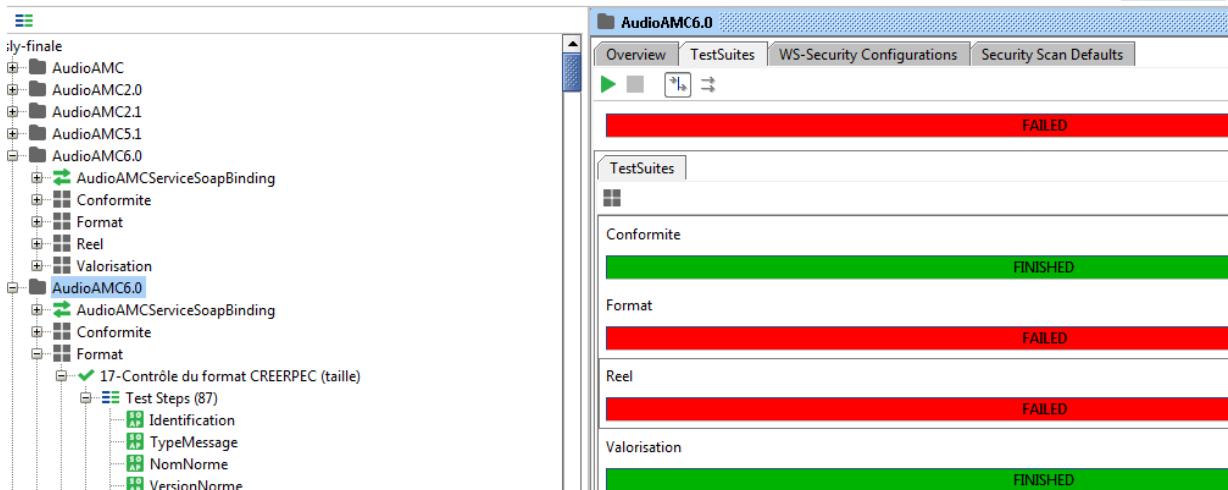
1. Double cliquer sur le projet puis aller sur l'onglet TestSuites :



2. Maintenant, il ne vous reste plus qu'à lancer les tests en cliquant sur le triangle vert.

Dès lors, vous êtes maintenant libre de travailler sur une autre tâche ou aller siroter tranquillement votre soda préféré.

Voici le résultat pour le projet audioAMC du 27/06/2017 :



Vous pouvez voir précisément chaque requête XML pour savoir le détail de ces résultats. Vous pouvez également lancer le test depuis une suite de tests (ex : conformité) ou depuis les cas de tests.

### Etape 2 bis : Utilisation du script dans un nouveau projet

Utilisez le fichier « **Script\_groovy\_finale\_générique.groovy** ».

Pour utiliser le script correctement, il faut dans un premier temps :

1. créer un fichier **csv**
2. indiquer les valeurs des champs suivants :
  - a. catégories de tests (libre)
  - b. numéros de cas de tests (libre)
  - c. messages d'erreur (attendu)
  - d. valeur par défaut (true or false)

Ce dernier champ permet d'utiliser un même message d'erreur pour tout les tests de la suite. Pour cela on met sa valeur à « **true** ».




Categorie de tests	Cas de test	Message d'erreur	Message par défaut
Conformite	1 à 14	OK	true
Format	17 à 28	faultstring	true
Reel	16	La fonction du message est inconnue	false
Reel	29	Devis ou Prise en charge hors TP non géré	false
Reel	30	Devis ou Prise en charge hors TP non géré	false
Reel	31	n'est pas connu chez CBP	false
Reel	32	n'est pas connu chez CBP	false
Reel	33	n'est pas connu chez CBP	false
Reel	34	n'est pas connu chez CBP	false
Reel	35	n'est pas connu chez CBP	false
Reel	36	n'est pas ouvert	false
Reel	49	Le numéro FINESS n'est pas connu de CBP	false
Reel	50	Le numéro FINESS n'est pas connu de CBP	false
Reel	51	Le numéro FINESS n'est pas connu de CBP	false
Reel	52	Le numéro FINESS n'est pas connu de CBP	false
Reel	53	Le numéro FINESS n'est pas connu de CBP	false
Reel	54	Le numéro FINESS n'est pas connu de CBP	false
Reel	55	Le numéro FINESS n'est pas conventionné sur le réseau Carte Blanche	false
Reel	56	Le numéro FINESS n'est pas conventionné sur le réseau Carte Blanche	false

Ici l'exemple prend en compte 3 catégorie de tests : Conformité, Format et Reel. Mais vous pouvez mettre le nom que vous voulez pour chacun des catégories.

Le champ cas de tests permet d'avoir une bonne vue d'ensemble des cas de tests et des messages d'erreur attendus.

Si vous devez tester un même type d'erreur plusieurs fois il est important de renseigner la valeur de la colonne « **message par défaut** » à true.

Exemple : Si la suite de tests (  ) contient que des cas de tests avec un message d'erreur attendu plus ou moins similaire ( ex : « **faultstring** » pour les formats) vous pouvez renseigner « **true** » dans la ligne correspondante et dans la colonne « message par défaut ».

**Veiller juste à avoir le même nom et le même ordre sur vos suites de tests de votre projet SoapUI par rapport à votre fichier csv.**

1. Enfin, il ne vous reste plus qu'à copier-coller le script dans la fenêtre d'édition de script ( « **LoadScript** » ) :

▶ Edit ▼

```

1 def groovyUtils = new com.eviware.soapui.support.GroovyUtils(context)
2
3 // Initialisation des variables
4 def text = "~"
5 def n=0
6 def OK = false
7
8
9 def csvFilePath = "Z:/SoapUI_AudioAMC/LibelleErreur-SoapUI.csv"
10
11
12 def libelleKO = new ArrayList();
13 def categorie = new ArrayList();
14 def testsMultiple = new ArrayList();
15
16 // Lecture du fichier csv
17 context.fileReader = new BufferedReader(new FileReader(csvFilePath))
18 rowsData = context.fileReader.readLine()
19 int rowsize = rowsData.size()
20
21 // Ecriture du contenu dans un tableaux
22 for(int i = 1; i < rowsize; i++)
23 {
24
25     rowdata = rowsData[i]
26     String[] data = rowdata.split(";")
27     //log.info data[2]
28     libelleKO.add(data[2])
29     if(!categorie.contains(data[0])){
30         categorie.add(data[0]);
31         testsMultiple.add(data[3])

```

Description Properties Load Script Save Script

2. Maintenant, appuyer sur le triangle vert puis patienter quelques instants.

Tout les mini-tests seront ajoutés.

Il ne vous reste plus qu'à dérouler tout les tests. Mais avant, pensez à changer tout les **numeros de dossiers PS**. (voir page 14 pour plus de précision sur le lancement des tests)

## VI. Pour aller plus loin

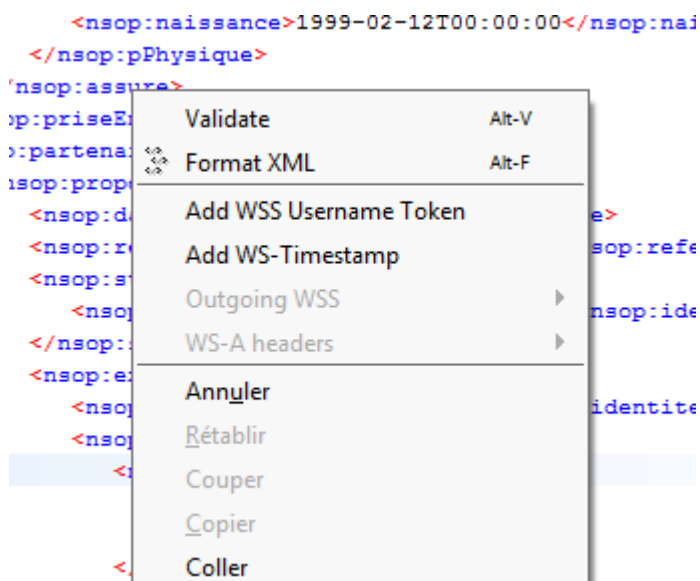
***Lancer le « Test runner »***

***Assertion avec regex***

## Etape 4: Astuces SoapUI

Contrôle du flux :

En cas d'erreur, il est possible de contrôler le flux afin de s'assurer qu'il est bien conforme au WSDL. Pour cela il suffit de faire un clic droit sur le texte du flux envoyé et de sélectionner « **Validate** »



L'outil de chargera de contrôler le flux et renverra les anomalies trouvées en bas de page.

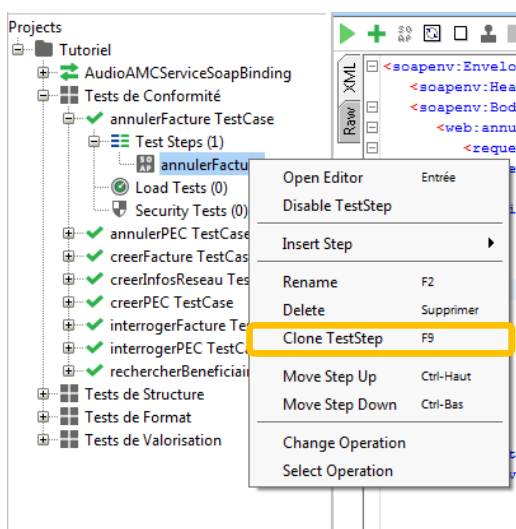
Si le flux est mal indenté, il est possible de restructurer automatiquement en cliquant sur « **Format XML** » sur ce même menu.

### Création des jeux de données (sans script) :

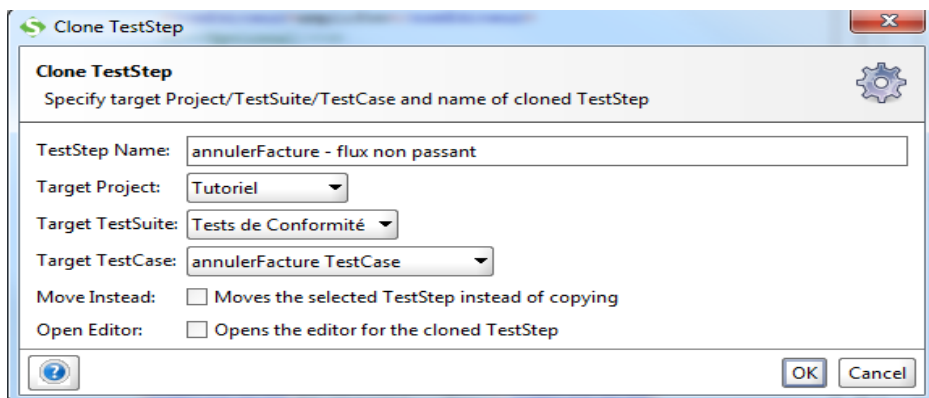
Lorsque vous créez vos jeux de données et s'ils retournent une même catégorie d'erreur (ex : **faultstring**), je vous conseille de créer vos mini-tests (« **assertions** ») dès maintenant.

Ainsi il ne vous reste plus qu'à cloner les requêtes XML pour chaque catégorie de tests et vous aurez toutes les assertions voulues.

1. Pour cela, faites un clic droit sur la requête et sélectionner « **Clone TestStep** » :



Une fenêtre s'ouvre et vous propose de renseigner les caractéristiques du cas de tests cloné :



2. Cliquer sur « **OK** » pour valider.

3. Si vous n'avez pas réellement de catégorie d'erreur, c'est-à-dire que vous ne pouvez classer les erreurs par leur nature, je vous invite à suivre la partie du tutoriel sur les scripts (voir la partie V sur ***l'utilisation du script***).

#### Changement des numéros de dossier:

1. Sauvegarder votre projet dans un dossier précis.
2. Allez dans ce dossier et ouvrez le fichier avec l'éditeur de texte SublimeText.

Dans un premier temps mettez tous les numéros de dossier à nulle :

1. **Ctrl-H** puis entrez « **<numDossierPS>.\*</numDossierPS>** » dans la barre de recherche (« **Find What** »)
2. Ecrire « **<numDossierPS></numDossierPS>** » dans la barre de remplacement (« **Replace With** »).

Dans un second temps, suivre ces étapes :

1. Ctrl-F puis entrez « **<numDossierPS>.\*</numDossierPS>** » dans la barre de recherche.
2. Appuyer sur « **Alt** » puis en maintenant la touche enfoncé appuyé sur la touche « **Shift** »
3. Garder la touche « **Ctrl** » enfoncé et appuyer sur la touche directionnelle gauche jusqu'à arriver dans la balise.
4. Entrez des valeurs aléatoires puis en maintenant la touche « **Ctrl** » et « **Alt** » enfoncé puis appuyé sur la lettre « **i** ».

***Voilà tous vos numéros de dossier de professionnel de santé (« numdossierPS ») de santé sont maintenant différents, vous pouvez utiliser le script sans changer manuellement ces valeurs.***

Suivre les logs du script :

Appuyer sur script log tout en bas de la fenêtre.

Une fenêtre s'ouvre, vous trouverez les informations du déroulement du script

Extraction des messages d'erreur du cahier de recette :

1. Copier le contenu du cahier de recette et coller le dans l'éditeur Notepad++
2. Aller sur l'onglet Rechercher puis sélectionner « **Marquer...** »
3. Dans le champ de recherche taper « **LibelleKO** »
4. Cocher « **Marquer les lignes** »
5. Cliquer sur « **Rechercher tout** »
6. Aller sur l'onglet Rechercher puis sélectionner « **Signet** » et sélectionner « **Supprimer les lignes non marquées** »

Vous voilà avec tous les messages d'erreur, maintenant supprimer tout ce qui ne concerne pas les messages d'erreurs :

1. **Ctrl-h** et copier-coller le contenu que vous voulez supprimer dans la barre de recherche
2. Laissez vide le champ « **remplacez par** ».
3. Appuyer sur « **Remplacer tout** »

Il ne vous reste plus qu'à copier-coller le contenu de votre fichier dans la colonne « **message d'erreur** » du fichier csv « **LibelleErreur-SoapUI.csv** ».