

# CONSOMMER DES SERVICES WEB EN RPG ILE

## Deuxième partie

Nous avons déjà abordé le sujet « Consommer des Services Web en RPG ILE » dans le iSeries News de Janvier 2010.

Pour ce début d'année 2011, IBM nous gratifie d'une amélioration très intéressante sur le sujet. Si vous vous rappelez, l'utilitaire `wSDL2ws.sh` d'Apache Axis nous permettait de générer tous les programmes clients nécessaires pour consommer un Service Web comme montré en figure 1.

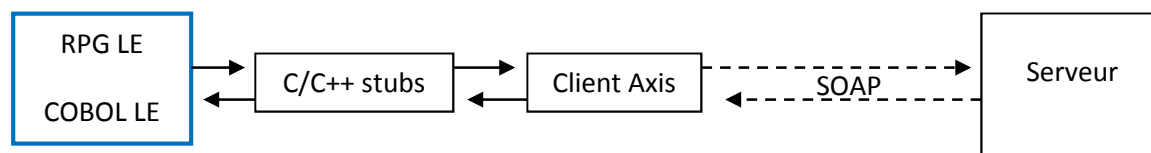


Figure 1

Il ne nous manquait plus qu'à écrire un programme HLL (en RPG ou COBOL) afin d'appeler ces programmes C/C++. La principale difficulté de cet exercice était de savoir lire un programme C/C++ afin de déterminer les bons paramètres attendus par ces « stubs ».

Dorénavant, nous pouvons utiliser le nouvel utilitaire `wSDL2rpg.sh` qui permet de générer les stubs directement en RPGLE facilitant ainsi grandement l'interface entre programmes. De plus, le résultat de la compilation n'est ni plus ni moins qu'un programme de service directement utilisable comme nous allons le voir.

L'utilitaire `wSDL2rpg.sh` est disponible par PTF en V5R4, V6R1 et V7R1 depuis le Janvier 2011.

	V5R41	V6R1	V7R1
PTFs préalables	SI42234	SI42236	SI42235

Source : APAR SE46281

Comme le précédent, cet utilitaire se trouve dans le répertoire de l'IFS à l'emplacement :  
`/qibm/proddata/os/webservices/v1/client/bin`

Reprenons l'exemple du Service Web gratuit **StockQuote** utilisé dans l'article précédent, permettant d'interroger directement la bourse de NewYork (NYSE). Vous pouvez le tester directement en ligne sur le lien suivant : <http://www.websvcx.net/stockquote.asmx> si le serveur n'est pas surchargé comme à son habitude.

Nous commencerons par créer un répertoire dans l'IFS

```
MKDIR '/home/webservices'
```

Puis nous récupérons le fichier WSDL du service web et le plaçons dans l'IFS sous le nom StockQuote.WSDL : <http://www.websvcx.net/stockquote.asmx?WSDL>

Sous QSH il nous suffit de lancer :

```
1. /QIBM/ProdData/OS/WebServices/V1/client/BIN/wsd12rpg.sh
   -t60 -o/home/webservices
   -s/QSYS.LIB/MYLIB.LIB/STOCKQUOTE.SRVPGM
     /home/webservices/StockQuote.wsdl
```

- t : Ce paramètre facultatif permet d'indiquer un timeout en secondes
- o : Répertoire où seront générés les sources des programmes
- s : Répertoire où sera généré le programme de service compilé, ici il s'agit de la bibliothèque MYLIB

Le dernier paramètre représente le fichier WSDL ou son URI ce qui veut dire que l'on aurait très bien pu demander la génération du programme de service et des programmes sources sans copier le fichier WSDL dans l'IFS en utilisant la syntaxe ci-dessous :

```
1. /QIBM/ProdData/OS/WebServices/V1/client/BIN/wsd12rpg.sh
   -t60 -o/home/webservices
   -s/QSYS.LIB/MYLIB.LIB/STOCKQUOTE.SRVPGM
     http://www.websvcx.net/stockquote.asmx?WSDL
```

```
Code generation completed. Generated files in directory
'/home/webservices'.
```

```
Attempting to create service program...
```

```
Service program created. Service program is
'/QSYS.LIB/MYLIB.LIB/STOCKQUOTE.SRVPGM'.
$
```

Les fichiers sources et programmes générés sont

/home/webservices	DESCRIPTION
StockQuoteSoap.c StockQuoteSoap.h	Stub Axis en C++ (Corps et Entête) ayant servi à créer le module WSC0 dans la bibliothèque MYLIB
StockQuoteSoap.cl	Programme CLLE ayant servi à créer le programme de service STOCKQUOTE dans la bibliothèque MYLIB par assemblage des modules WSC0, WSR1 et WSR2
StockQuoteSoap.rpgle StockQuoteSoap.rpgleinc	Procédures du Web Service (Corps et Entête) compilés dans le module WSR1
StockQuoteSoap_util.rpgle StockQuoteSoap_util.rpgleinc	Utilitaires RPGLE (Corps et Entête) ) compilés dans le module WSR2
StockQuoteSoap_xsdtypes.rpgleinc	Types standards de données utilisés en /COPY

Bibliothèque MYLIB	Type	DESCRIPTION
<b>STOCKQUOTE</b>	<b>*SVRPGM</b>	StockQuoteSoap Web service
WSC0	*MODULE	StockQuoteSoap.c
WSR1	*MODULE	StockQuoteSoap.rpgle
WSR2	*MODULE	StockQuoteSoap_util.rpgle

Le seul fichier source qui va nous intéresser par la suite est **StockQuoteSoap.rpgle** ainsi que le programme de service **STOCKQUOTE.SVRPGM**.

Visualisons maintenant le contenu du source **StockQuoteSoap.rpgle** généré automatiquement. Trois procédures y sont définies :

1. stub\_create\_StockQuoteSoap
2. stub\_destroy\_StockQuoteSoap
3. stub\_op\_GetQuote

```

* *****
* RPG Call : stub_create_StockQuoteSoap ①
* parm in : This_t this
* return  : *ON = connected
*          : *OFF = connect failed
* *****
D stub_create_StockQuoteSoap...
D          PR          1N  extproc('stub_create_StockQuoteSoa+
D                               p@')
D this          likeds(This_t)

* *****
* RPG Call : stub_destroy_StockQuoteSoap ②
* parm in : This_t this
* return  : *ON = destroyed
*          : *OFF = destroy failed
* *****
D stub_destroy_StockQuoteSoap...
D          PR          1N  extproc('stub_destroy_StockQuoteSo+
D                               ap@')
D this          likeds(This_t)

* *****
* RPG call : stub_op_GetQuote ③
* parm in : This_t this
* parm in : xsdc__string Value0
* parm out: xsdc__string out
* return  : *ON = success
*          : *OFF = failure
* *****
D stub_op_GetQuote...
D          PR          1N  extproc('GetQuote@')
D this          likeds(This_t)
D Value0        likeds(xsd_string)
D out           likeds(xsd_string)

```

Le `stub_create_xxxx` permet d'initialiser et ouvrir la connexion, un handle de connexion sera renvoyé en retour, `stub_destroy_xxxx` de fermer la connexion, et `stub_op_xxxx` de consommer le service Web.

## Écriture du programme Client

Maintenant que nous connaissons les procédures à appeler, nous pouvons écrire notre RPG ILE afin de consommer notre Service Web.

```

*-----
/copy StockQuoteSoap.rpgleinc ①

d OutputText      s          50
d WsStub          ds          likeds(This_t)
d Input           ds          likeds(xsd_string)
d Result          ds          likeds(xsd_string)

*-----
*-----
C      *ENTRY      PLIST
C      PARM              QUOTE          3
*-----
/free
  clear WsStub;
  WsStub.endpoint = *blanks; ②
  clear input;
  Input.value = %trim(Quote); ③

  if stub_create_StockQuoteSoap(WsStub); ④

    // Invoke the Web service operation.
    if stub_op_GetQuote(WsStub:Input:Result); ⑤
      OutputText = Result.value;
    else;
      OutputText = WsStub.excString;
    endif;

    // Display results.
    dsply OutputText;

    // Destroy Web service stubs.
    stub_destroy_StockQuoteSoap(WsStub); ⑥
  endif;
  *inlr = *on;

```

**Figure 2**

- ① Intégration des prototypes générés automatiquement
- ② On renseigne le « end-point ». En l'occurrence nous aurions pu mettre <http://www.webservicex.net/stockquote.asmx>. \*BLANKS signifie que l'on prend le « end-point » défini directement dans le document WSDL.
- ③ Constitution du paramètre à envoyer.
- ④ Ouverture de la connexion, récupération du handle de connexion dans un pointeur.
- ⑤ Appel du Service Web avec le handle de connexion et les paramètres en entrée et en sortie.
- ⑥ Clôture de la connexion en lui passant le handle de connexion, pas de valeur en retour.

## Compilation et liage des modules

Il faut maintenant compiler ce programme et y intégrer le programme de service généré automatiquement par l'utilitaire comme indiqué en Figure 3.

```
CRTRPGMOD MODULE(QTEMP/STOCKQUOTE) SRCFILE(MYLIB/QRPGLESRC) SRCMBR(*MODULE)
DBGVIEW(*SOURCE) INCDIR('/home/webservices')

CRTPGM PGM(MYLIB/STOCKQUOTE) MODULE(QTEMP/STOCKQUOTE)
BNDSRVPGM((MYLIB/STOCKQUOTE)) ACTGRP(*NEW)AUT(*USE) ENUM(*INT)
```

Figure 3

Il ne nous manque plus qu'à tester notre programme avec un

CALL MYLIB/STOCKQUOTE 'IBM'.

## Echec au liage

En cas d'échec lors du CRTPGM dû au liage à cause d'un problème de CSSID, il vous faudra modifier le fichier source **StockQuoteSoap.rpgleinc** avec RDP/RDI (ou avec la commande EDTF) afin de remplacer les caractères @ par un à.

Puis recompilez le tout en procédant comme montré en figure 4 :

```
- CPYFRMSTMF FROMSTMF('/home/webservices/StockQuoteSoap.cl')
  TOMBR('/QSYS.LIB/MYLIB.LIB/QCLLESRC.FILE/COMPILE.MBR')
- CRTBNDCL PGM(MYLIB/COMPILE) SRCFILE(MYLIB/QCLLESRC)
- CALL MYLIB/COMPILE

- CRTRPGMOD MODULE(QTEMP/STOCKQUOTE) SRCFILE(MYLIB/QRPGLESRC) SRCMBR(*MODULE)
  DBGVIEW(*SOURCE) INCDIR('/home/webservices')

- CRTPGM PGM(MYLIB/STOCKQUOTE) MODULE(QTEMP/STOCKQUOTE)
  BNDSRVPGM((MYLIB/STOCKQUOTE)) ACTGRP(*NEW)AUT(*USE) ENUM(*INT)
```

Figure 4

## Conclusion

Il n'a jamais été aussi facile de consommer un Web Service avec l'utilitaire **wsdl2rpg.sh** pour un programmeur RPG ILE. L'IBM i qui offre déjà un socle technique solide et mature sait aussi faire preuve d'efficacité dans l'univers des nouvelles technologies.

*Patrick THOMAS*

[www.LesCahiersDeLIBMi.fr](http://www.LesCahiersDeLIBMi.fr)