CONSOMMER DES SERVICES WEB EN RPG ILE

Les avantages des Services Web ne sont plus à démontrer aujourd'hui, ils sont devenus le nouveau point de convergence des acteurs du marché de l'informatique qui en font un nouveau standard accepté de tous. Le vieux rêve de faire communiquer en temps réel différents programmes issus de différentes plates-formes hétérogènes et ceci quelque soit les distances, est consommé. Que ce soit dans votre intranet pour faire communiquer vos différents applicatifs, ou sur internet pour accéder à un service gratuit ou payant, ils sont l'avenir et leur implémentation au cœur de l'IBM i est devenue une sinécure.

De nombreux articles définissent avec précision ce qu'est un Service Web, leur utilité et la technologie employée, iSeries News a d'ailleurs publié déjà de nombreux articles sur le sujet, les plus remarquables rédigés par Skott Klement et son projet open-source HTTPAPI. Le succès de ce projet a fait réagir les équipes d'IBM à Rochester. Ainsi, la V6R1 nous a apporté un lot important de nouveautés concernant les Services Web, dont nombre d'entre elles ont été portées en V5R4 par PTF.

Nous possédons dorénavant en V5R4 un serveur de Service Web intégré à l'OS qui vous permettra en quelques clics de transformer un programme ILE RPG ou COBOL en Service Web tout en s'affranchissant de déployer Websphère Application Server, mais nous traiterons ce sujet dans un autre article.

La nouveauté qui nous concerne, le portage du projet Apache Fondation Axis C++ version 1.5 dans l'IBM i. Axis est un Framework open-source qui offre en outre la possibilité de générer automatiquement des WSDL correspondant à des classes Java ou inversement à créer des classes Java, C ou C++ sur la base d'un WSDL, et c'est ce dernier point qui va nous intéresser dans cet article. Ainsi, toute la complexité d'accès au programme distant avec le protocole HTTP à travers des messages SOAP est masquée dans ces classes proxy. Il nous reste plus qu'à appeler ces programmes générés pour consommer un Service Web comme indiqué sur la figure 1.

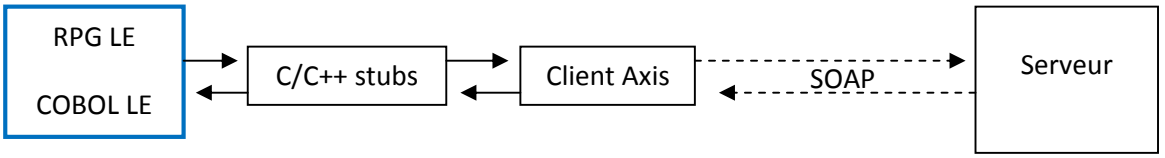
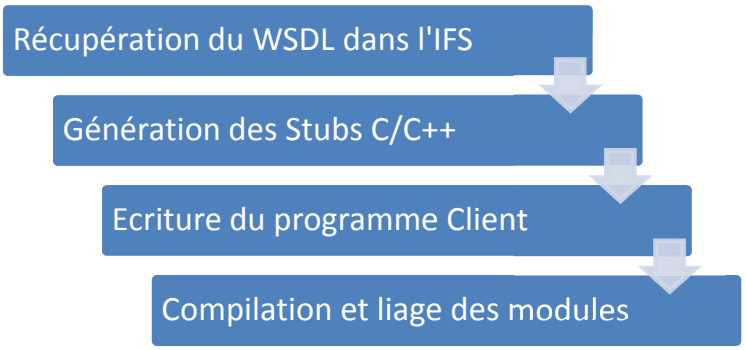


Figure 1

Ainsi, les étapes nécessaires pour consommer un service web sont :



Pour connaître les limitations l'Axis C++ 1.5 dans l'IBM i, je vous conseille de visualiser le README.TXT situé dans le répertoire /Qibm/ProdData/OS/Webservices/V1/Client/. Extraits :

WSDL 1.1. is the only supported WSDL level.
SOAP 1.1 is the only supported over-the-wire protocol
The response from the Server must be in UTF-8 format
…

Récupération dans l'IFS du WSDL

WSDL est un fichier XML qui contient l'emplacement du Service Web sur la toile ainsi que les opérations (méthodes, paramètres et valeurs de retour) que le service propose.

Prenons un exemple avec le Service Web gratuit StockQuote, permettant d'interroger directement la bourse de NewYork (NYSE), vous pouvez le tester directement en ligne sur le lien suivant : http://www.webservicex.net/stockquote.asmx

Pour le tester, on saisi en ligne le code boursier « IBM » et on obtient en réponse le message XML (figure 2)

```
<?xml version="1.0" encoding="utf-8" ?>
<string xmlns="
http://www.webserviceX.NET/"><StockQuotes><Stock><Symbol>IBM</
Symbol><Last>128.63</Last><Date>11/17/2009</Date><Time>4:00p
m</Time><Change>+0.42</Change><Open>127.43</Open><High>12
8.655</High><Low>127.40</Low><Volume>7751233</Volume><MktC
ap>169.0B</MktCap><PreviousClose>128.21</PreviousClose><Percent
ageChange>+0.33%</PercentageChange><AnnRange>69.50 -
128.61</AnnRange><Earns>9.74</Earns><P-E>13.16</P-
E><Name>INTL BUSINESS  MAC</Name></Stock></StockQuotes>
</string>
```

Figure 2

On récupère le fichier WSDL sur http://www.webservicex.net/stockquote.asmx?WSDL et on le place dans un répertoire de l'IFS ex : /home/webservices/ sous le nom de stockquote.wsdl (exemple pour la suite de cet article). Nous avons constaté que le service web répondait bien en le testant directement en ligne, avons récupéré son WSDL dans l'IFS (vous pouvez

d'ailleurs visualiser son contenu avec **DSPF '/home/webservices/stockquote.wsdl'**, nous pouvons maintenant passer à l'étape suivante.

Génération automatique des Stubs en C/C++

Nous allons utiliser la classe Java **wsdl2ws.sh** livrée par Apache Axis afin qu'il nous génère tous les programmes clients nécessaires pour consommer ce Service Web. Nous ne détaillerons pas ici tous les paramètres attendus, mais vous pourrez visualiser l'aide en ligne de cette commande en tapant **wsdl2ws.sh -h** sous QSH dans le répertoire **/QIBM/ProdData/OS/WebServices/V1/client/BIN/**

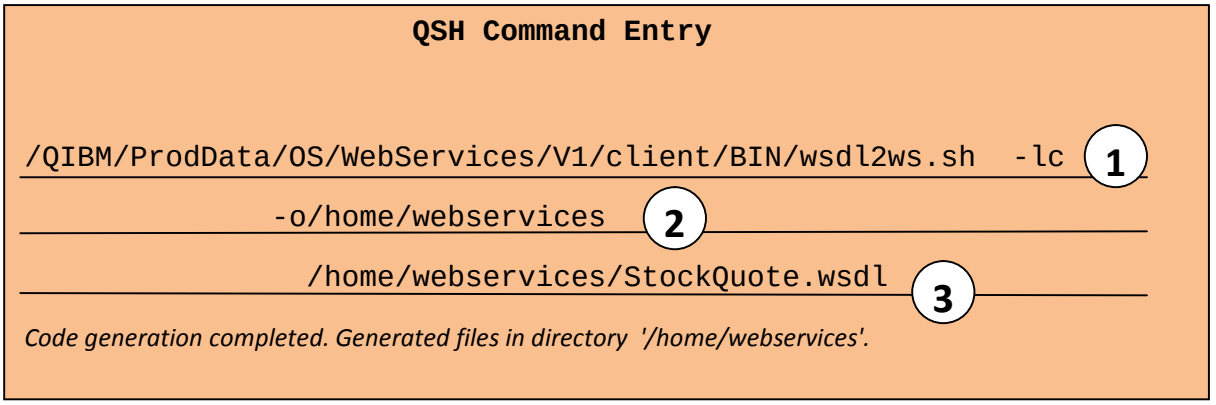


Figure 3

A l'issu de la commande (figure 3), des sources de programmes en C (si *-LC* n'avait pas été indiqué au paramètre ① du C++ aurait été généré) ont été créés dans le répertoire ② à partir du WSDL fourni en ③.

Avec notre WSDL, on récupère StockQuoteSoap.h et StockQuoteSoap.c. En regardant les déclaratives dans le .h on peut voir :

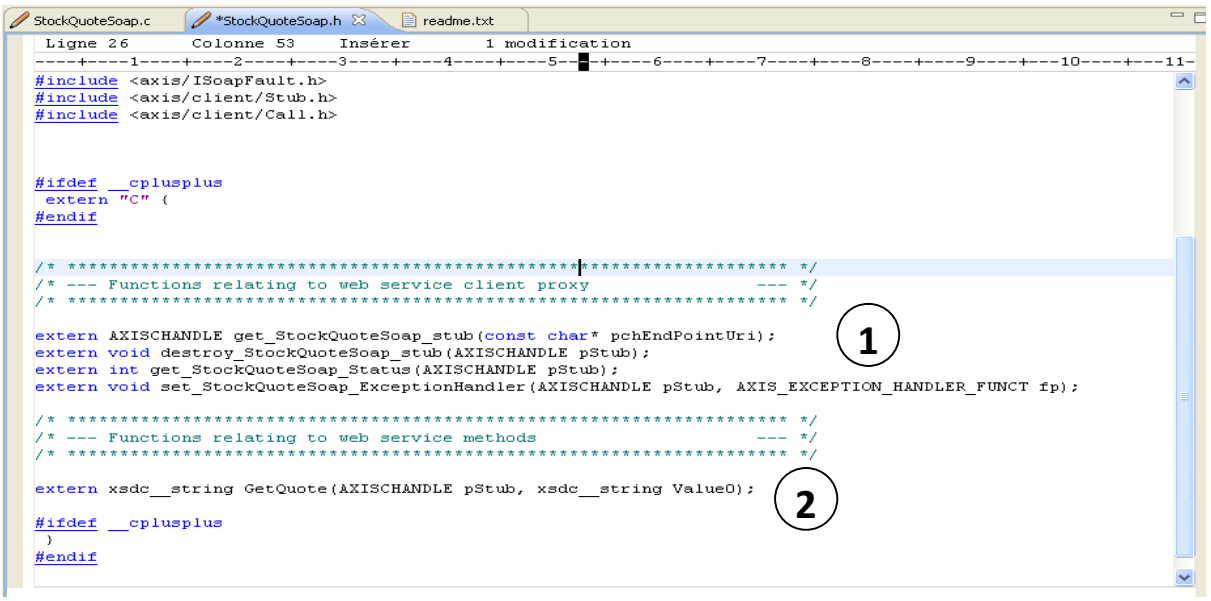


Figure 4

Le Get___Stub permet d'initialiser et ouvrir la connexion, Destroy___Stub de la fermer, Get___Status de tester la dernière opération effectuée et Set___ExceptionHandler de gérer les erreurs SOAP.

Pour consommer notre service web depuis notre RPG, il nous faudra au minimum appeler les fonctions prototypées Get_StockQuoteSoap_stub(), GetQuote() ainsi que la fonction Destroy_StockQuoteSoap_stub(). Mais avec quels paramètres ?

AXISCHANDLE get_StockQuoteSoap_stub(const char pchEndPointUri);*

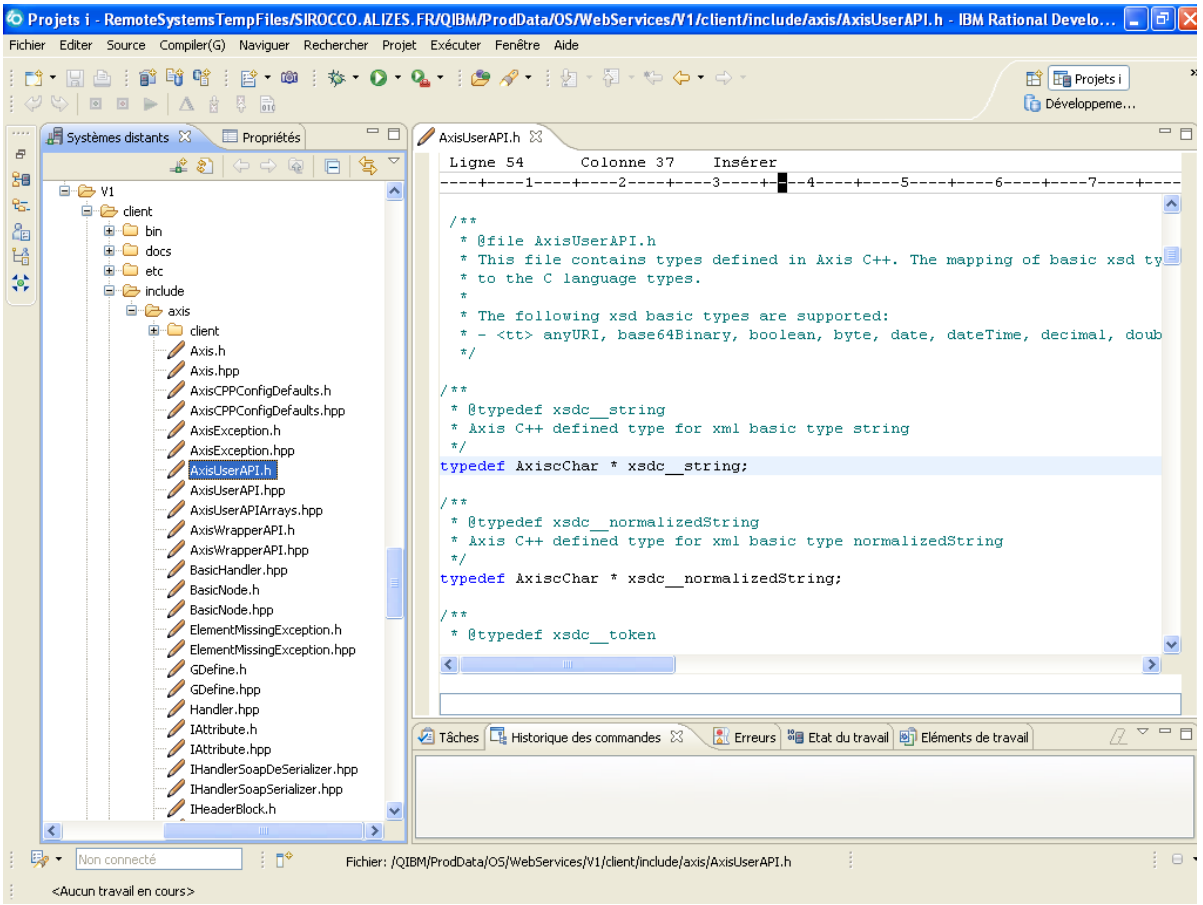
Cette fonction C reçoit en paramètre un **char*** c'est-à-dire un pointeur et renvoie aussi un pointeur à travers le Typdef AXISCHANDLE (il s'agit en fait du handle de la connexion établie).

void destroy_StockQuoteSoap_stub(AXISHANDLE pStub);

Cette fonction reçoit en paramètre le handle (renvoyé par Get___Stub) avec lequel la connexion a été établie, par contre ne renvoie aucune valeur (void).

xsdc__string GetQuote(AXISCHANDLE pStub, xsdc__string Value0);

Cette fonction est la plus importante car il s'agit su service en lui-même et les paramètres varient pour chaque Service Web invoqué. Elle reçoit le handle de la connexion (pointeur) ainsi qu'une variable de type « xsdc__string » et renvoie aussi un « xsdc__string » en retour. Mais ce type n'est pas standard, c'est un type défini dans Axis. S'agit-il d'un char (string) comme il semble le paraître ? Pour cela il faut regarder dans /QIBM/ProdData/OS/WebServices/V1/client/include/axis/AxisUserApi.h, il s'agit en fait aussi d'un pointeur.



Ecriture du programme Client

Maintenant que nous connaissons les fonctions à appeler ainsi que les bons paramètres, nous pouvons écrire notre RPG ILE pour consommer notre premier Service Web.

```
 *---------------------------------------------------------------------
 * Prototypes et interfaces
 *---------------------------------------------------------------------
DgetStub          PR              *   ExtProc('get_StockQuoteSoap_stub')
D pEndpoint                       *   Value
 *
DdestroyStub      PR                  ExtProc('destroy_StockQuoteSoap_Stub')
D AxisHandle                      *   Value
 *
DWebservice       PR              *   ExtProc('GetQuote')
D AxisHandle                      *   Value
D Parameter1                      *   Value
 *
 *---------------------------------------------------------------------
 * Déclarations diverses.
 *---------------------------------------------------------------------
D pResultat       S               *
D Resultat        S           32565   Varying
D WsStubP         S               *
 *---------------------------------------------------------------------
 * Web service logic. The code will attempt to invoke the Web Service
 *---------------------------------------------------------------------
 /FREE
     // Get a Web service stub. The host and port for the endpoint may need
     // to be changed to match host and port of Web service. You can pass
     // *NULL to getStub() if the endpoint in the WSDL file is correct.

     WsStubP = getStub(*NULL); ①
   ② Parameter = 'IBM' + x'00';
     pResultat = WebService(WsStubP : %addr(Parameter)); ③
   ④ Resultat = %str(pResultat);
     destroyStub(WsStubP); ⑤
     *inlr = *on;
```

Figure 5

① Ouverture de la connexion, récupération du handle de connexion dans un pointeur.
② Constitution du paramètre à envoyer, toujours terminer par x'00' pour le C/C++.
③ Appel du Service Web avec le handle de connexion et l'adresse du ou des paramètres. En retour il renvoie un pointeur mémoire ou est stocké le résultat.
④ Récupération du contenu du résultat à partir du pointeur renvoyé.
⑤ Clôture de la connexion en lui passant le handle de connexion, pas de valeur en retour.

N'est-ce pas formidable ? En quelques lignes de code, vous consommez déjà votre premier Service Web en invoquant directement la bourse de New York. Le problème, StockQuote renvoie un fichier XML complet comme montré à la figure 2, il vous faudra soit extraire ces

données par des %scan, %substr etc… ou simplement en utilisant le nouveau code opération XML-INTO de la V5R4. Je vous rassure, tous les Services Web ne renvoient pas forcément un résultat dans un champ caractère contenant du XML. Mais avec StockQuote c'est le cas.
Pour affecter proprement le contenu du XML dans des zones, il vous faudra décrire une Data Structure ayant la même forme que le XML résultant et possédant les mêmes noms de zone.
Le but de cet article n'est pas de vous apprendre à utiliser XML-INTO, ce code opération est bien documenté, mais pour l'exemple, voici ce qu'il faudra rajouter à votre programme :

```
 *----------------------------------------------------------------
 * Data Structure pour XML-INTO
 *----------------------------------------------------------------
D StockQuotes     DS                  Qualified
D   Stock                             LikeDS(Stocks)

D Stocks          DS                  Qualified
D  Symbol                       20
D  Last                         20
D  Date                         20
D  Time                         20
D  Change                       20
D  Open                         20
D  High                         20
D  Low                          20                ①
D  Volume                       20
D  MktCap                       20
D  PreviousClose                20
D  PercentageChange...
D                               20
D  AnnRange                     20
D  Earns                        20
D  P_E                          20
D  Name                         20
 *----------------------------------------------------------------
…

      Dow not (%scan('P-E':Resultat) = 0);
②         Resultat = %Replace('P_E' : Resultat : %scan('P-E':Resultat)  : 3);
      Enddo;
      XML-INTO StockQuotes %XML(Resultat:
                                'case=any +
                                 allowmissing=yes +
                                 allowextra=yes');
```

Figure 6

① J'ai déclaré chacune des zones en caractère de position 20 alphanumérique, mais à vous de préférer les résultats dans d'autres types de zones avec des longueurs différentes.

② Déclarer une zone P-E n'est pas possible en RPG, à cause du caractère spécial « - ». Je substitue donc, avant mon XML-INTO, P-E par P_E qui lui, est accepté.

Compilation et liage des modules

Il faut maintenant compiler le tout, c'est-à-dire les Stubs C/C++ d'Axis ainsi que notre ILE RPG pour en faire un programme unique.

```
CRTCMOD    MODULE(QTEMP/MOD001)
SRCSTMF('/home/webservices/StockQuote.C')
AUT(*USE) ENUM(*INT)                                          ①
INCDIR('/home/webservices'
       '/QIBM/PRODDATA/OS/WEBSERVICES/V1/CLIENT/INCLUDE')


CRTRPGMOD MODULE(QTEMP/MOD002)
SRCFILE(WEBSERVICE/QRPGLESRC) SRCMBR(STOCKQUOTE)              ②


CRTPGM PGM(WEBSERVICE/STOCKQUOTE) MODULE(QTEMP/MOD*)
BNDSRVPGM((QSYSDIR/QAXIS10CC))                                ③
```

Figure 7

① Création du module à partir de la source générée automatiquement par Axis. Dans le mot clé SRCSTMF() on renseigne ou se trouve la source du programme, dans INCDIR() le répertoire ou se trouve StockQuote.h car il est défini en /COPY ainsi que le répertoire des procédures Axis, incluses aussi par un /COPY.

② Création du module de notre RPG ILE.

③ Création du programme définitif en liant les différents modules ainsi que le programme de service QSYSDIR/QAXIS10CC d'Axis.

Conclusion

Consulter la bourse ou la météo en temps réel, envoyer des SMS en masse, contrôler un Swift IBAN/BIC, etc… tout cela est simple à faire en ILE RPG ou COBOL ! Les services SOAP sont nombreux et diversifiés sur la toile, mais pensez-y aussi pour faire communiquer vos applicatifs intranet entre eux surtout s'ils résident sur des plates-formes hétérogènes.

Patrick THOMAS