Université IBM i

10 et 11 mai 2016 – IBM Client Center de Bois-Colombes

S37 – Appeler vos programmes RPG/COBOL d'où vous voulez !

Mercredi 11 mai – 13h30-15h00

Nathanaël BONNET / Jean-Marie SAAD – Gaia

Gaia

- Conseil et formation IBM i (~~AS/400~~) depuis 1995
  - Inter et intra entreprise
- Base de connaissance en ligne
  - http://know400.gaia.fr
- Organisateur des matinées 400 iday
  - 24 mai à Lyon
  - 31 mai à Paris
  - http://www.gaia.fr/400iday-2
- Contact
  - contact@gaia.fr
  - http://www.gaia.fr
  - http://twitter.com/GaiaFrance

L’IBM i aujourd’hui

- Capital applicatif
  - Un ensemble de programmes, écrits au fil des ans
  - Une implémentation des règles métier
  - Des milliers d’heure de développement et test
  - Vos données métier
- Ce capital doit rester ou devenir un avantage concurrentiel
  - Permettre son ouverture, utilisation depuis les autres applications du SI
  - Afin que vous puissiez développer de nouvelles fonctions business au lieu de réaliser des migrations techniques
  - En s’appuyant sur votre existant et vos compétences
- Vos programmes RPG / COBOL !

Langage PCML

Le langage PCML

- Program Call Markup Language
  - Langage XML
  - Permet de décrire l’interface d’un programme ou d’une procédure
- Fichier XML avec extension .pcml
  - Racine = pcml

```
<pcml version="6.0">
   <!-- RPG module: WS_SRV_OPR  -->
   <!-- created: 2016-04-19-10.58.02 -->
   <!-- source: QTEMP/QSQLTEMP1(WS_SRV_OPR) -->
   <!-- 109 -->
   <struct name="T_OPERATION">
      <data name="ID" type="int" length="4" precision="31" usage="inherit" />
      <data name="ID_CLI" type="int" length="4" precision="31" usage="inherit" />
      <data name="ID_CPT" type="int" length="4" precision="31" usage="inherit" />
      <data name="LIBELLE" type="char" length="25" usage="inherit" />
      <data name="MONTANT" type="packed" length="10" precision="2" usage="inherit" />
      <data name="MAJ" type="timestamp" usage="inherit" />
   </struct>
   <!-- 437 -->
   <program name="ADDOPERATION" entrypoint="ADDOPERATION" returnvalue="integer">...
   <!-- 301 -->
   <program name="GETOPERATIONARRAY" entrypoint="GETOPERATIONARRAY" returnvalue="integer">
      <data name="P_ID" type="int" length="4" precision="31" usage="input" />
      <data name="P_MAXRESULT" type="int" length="2" precision="16" usage="input" />
      <data name="P_OPRARRAY" type="struct" struct="T_OPERATION" count="20" usage="inputoutput" />
      <data name="P_OPRARRAY_LENGTH" type="int" length="4" precision="31" usage="inputoutput" />
      <data name="P_RC" type="char" length="3" usage="inputoutput" />
   </program>
   <!-- 224 -->
   <program name="GETOPERATION" entrypoint="GETOPERATION" returnvalue="integer">...
</pcml>
```

Syntaxe

- Un ensemble de 3 tags
  - **`program`**
    - Un bloc **`<program> … </program>`** permet de décrire l’interface d’un programme
  - **`struct`**
    - Permet de définir une structure (DS).
    - Cette structure peut être utilisée en tant que paramètre d’un programme, ou composante d’une autre structure.
  - **`data`**
    - Permet de définir un champ d’une structure ou d’un paramètre programme.

Tag <program> (1/2)

- Permet de définir le nom et l’emplacement du code exécutable
- Syntaxe

<program name=_"name"_

[entrypoint=_"entry-point-name"_ **]**

[epccsid="ccsid" **]**

[path=_"path-name"_ **]**

[parseorder=_"name-list"_ **]**

[returnvalue=_"{ void | integer }"_ **]**

[threadsafe=_"{ true | false }"_ **]>**

</program>

Tag <program> (2/2)

- **`name`**
  - Nom du programme
- **`entrypoint`**
  - Nom du point d'entrée dans le programme de service cible
- **`path`**
  - Chemin jusqu'à l'objet *PGM ou *SRVPGM
  - La syntaxe est celle de l'IFS pour QSYS :
    `/QSYS.LIB/MALIB.LIB/MONPGM.PGM`

Tag <struct>

- Permet de définir une structure de données
  - Supporte les occurrences

- Syntaxe

```
<struct name="name"
    [ count="{number | data-name }« ]
    [ maxvrm="version-string" ]
    [ minvrm="version-string" ]
    [ offset="{number | data-name }" ]
    [ offsetfrom="{number | data-name | struct-name }" ]
    [ outputsize="{number | data-name }" ]
    [ usage="{ inherit | input | output | inputoutput }" ]>
</struct>
```

Tag <data> (1/3)

- Permet de définir
  - Une zone de structure ou un paramètre de programme ou de procédure

Tag <data> (2/3)

- Syntaxe

```
<data type="{ char | int | packed | zoned | float | byte | struct }"
    [ bidistringtype="{ ST4 | ST5 | ST6 | ST7 | ST8 | ST9 | ST10 | ST11
| DEFAULT }"]
    [ ccsid="{ number | data-name }" ]
    [ chartype="{ onebyte | twobyte }"]
    [ count="{ number | data-name }" ]
    [ init="string" ]
    [ length="{ number | data-name }" ]
    [ maxvrm="version-string" ]
    [ minvrm="version-string" ]
    [ name="name" ]
    [ offset="{ number | data-name }" ]
    [ offsetfrom="{ number | data-name | struct-name }" ]
    [ outputsize="{ number | data-name | struct-name }" ]
    [ passby= "{ reference | value }" ]
```

Tag <data> (3/3)

- Syntaxe (suite)

```
    [ precision="number" ]
    [ struct="struct-name" ]
    [ trim="{ right | left | both | none }" ]
    [ usage="{ inherit | input | output | inputoutput }" ]>
    [ dateformat="{ MDY | DMY | YMD | JUL | ISO | USA | EUR| JIS | CYMD
| CMDY | CDMY | LONGJUL | MY | YM | MYY | YMM }" ]>
    [ dateseparator="{ ampersand, blank | colon | comma | hyphen |
period | slash }" ]>
    [ timeformat="{ HMS | ISO | USA | EUR | JIS | USA | EUR | JIS }" ]>
    [ timeseparator="{ ampersand, blank | colon | comma | hyphen |
period | slash }" ]>
</data>
```

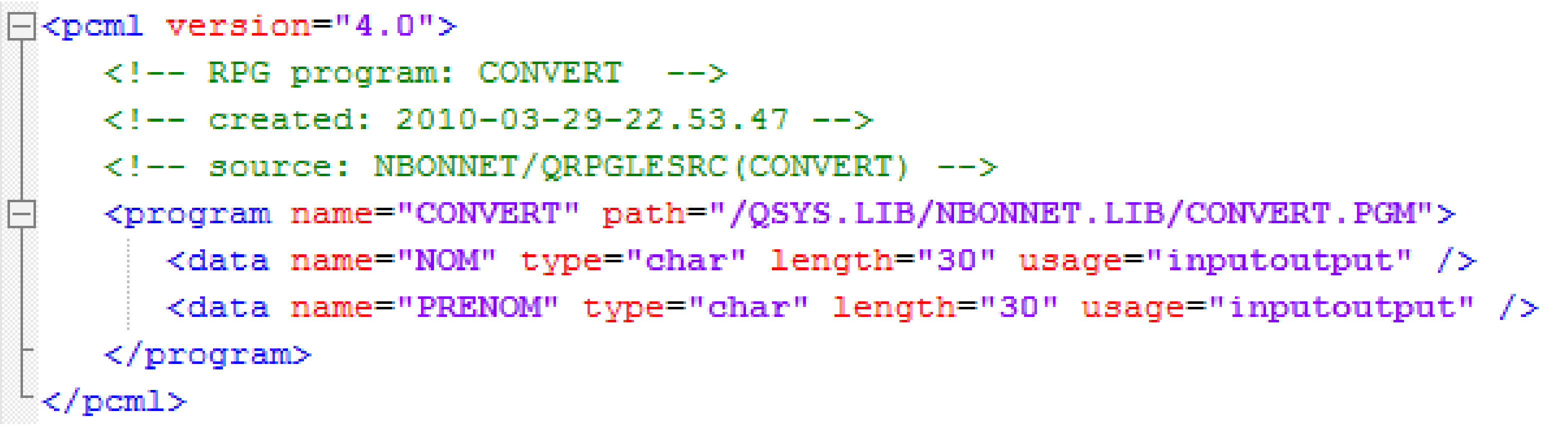
  - Les types date, heure et horodatage sont supportés depuis la 7.2 TR1

- **`Name`**, **`type`**, **`length`** et **`usage`** suffisent la plupart du temps

Exemple programme

- Pour un programme, à défaut d'un prototype, tous les paramètres sont transmis par référence
  - **`inputoutput`**
- Le **`path`** est totalement qualifié
  - Vous pouvez utiliser **`%LIBL%`** :

    `/QSYS.LIB/%LIBL%CONVERT.LIB`



```
<pcml version="4.0">
   <!-- RPG program: CONVERT  -->
   <!-- created: 2010-03-29-22.53.47 -->
   <!-- source: NBONNET/QRPGLESRC(CONVERT) -->
   <program name="CONVERT" path="/QSYS.LIB/NBONNET.LIB/CONVERT.PGM">
      <data name="NOM" type="char" length="30" usage="inputoutput" />
      <data name="PRENOM" type="char" length="30" usage="inputoutput" />
   </program>
</pcml>
```

Exemple procédures

- Pour un module, le **`path`** n’est pas renseigné par défaut
  - Le nom du programme de service n’est pas connu à la compilation du module

```
<pcml version="6.0">
   <!-- RPG module: WS_SRV_OPR  -->
   <!-- created: 2016-04-19-10.58.02 -->
   <!-- source: QTEMP/QSQLTEMP1(WS_SRV_OPR) -->
   <!-- 109 -->
   <struct name="T_OPERATION">
      <data name="ID" type="int" length="4" precision="31" usage="inherit" />
      <data name="ID_CLI" type="int" length="4" precision="31" usage="inherit" />
      <data name="ID_CPT" type="int" length="4" precision="31" usage="inherit" />
      <data name="LIBELLE" type="char" length="25" usage="inherit" />
      <data name="MONTANT" type="packed" length="10" precision="2" usage="inherit" />
      <data name="MAJ" type="timestamp" usage="inherit" />
   </struct>
   <!-- 437 -->
   <program name="ADDOPERATION" entrypoint="ADDOPERATION" returnvalue="integer">
   <!-- 301 -->
   <program name="GETOPERATIONARRAY" entrypoint="GETOPERATIONARRAY" returnvalue="integer">
      <data name="P_ID" type="int" length="4" precision="31" usage="input" />
      <data name="P_MAXRESULT" type="int" length="2" precision="16" usage="input" />
      <data name="P_OPRARRAY" type="struct" struct="T_OPERATION" count="20" usage="inputoutput" />
      <data name="P_OPRARRAY_LENGTH" type="int" length="4" precision="31" usage="inputoutput" />
      <data name="P_RC" type="char" length="3" usage="inputoutput" />
   </program>
   <!-- 224 -->
   <program name="GETOPERATION" entrypoint="GETOPERATION" returnvalue="integer">
</pcml>
```

Génération

- A la compilation pour RPG ILE et COBOL ILE
  - Paramètres **`PGMINFO`** et **`INFOSTMF`** sur les commandes de compilation
    - **`CRTBND[RPG/CBL]`**
    - **`CRT[RPG/CBL]MOD`**
  - **`PGMINFO`**
    - **`*NO`**
    - **`*MODULE`** : pcml stocké dans l'objet compilé lui-même
    - **`*STMF`** : pcml stocké dans un fichier sur l'IFS
    - **`*ALL`** : pcml dans le module et dans l'IFS
  - **`INFOSTMF`**
    - Fichier IFS généré par la compilation
- Il est recommandé de stocker au minimum dans l'objet compilé

Génération

- Option de compilation

```
ctl-opt pgminfo(*pcml:*module) nomain ;
```

- RPGLE

```
CRTRPGMOD ...
          PGMINFO(*PCML *STMF)
          INFOSTMF('/home/NB/module.pcml')
```

- SQLRPGLE

```
CRTSQLRPGI ...
           COMPILEOPT('PGMINFO(*PCML *STMF)
           INFOSTMF(''/home/NB/ws_srv_opr.pcml'')')
```

Limitations

- Paramètres transmis par valeur
  - Int (entier) sur 4 octets (**`10i0`**)
    - Les autres types ne sont pas supportés
- Types de données non supportées
  - Pointeur
  - Pointeur de procédure
  - Entier binaire sur un octet (**`3i0`** et **`3u0`**)
- Valeur de retour pour les procédures
  - Void (aucune)
  - Int (entier) sur 4 octets (**`10i0`**)
- Quelques contraintes sur les DS
  - Pas de recouvrement, les zones doivent se suivre
- Indicateurs
  - Considérés comme **`1A`**
- Ne fonctionne pas pour le CL
  - Mais vous pouvez le créer manuellement !
  - Ce n’est pas très compliqué

IWS – Integrated Web Services

Présentation

- IWS est composé de deux briques
  - Serveur : permet d'exposer un programme ILE sous forme de service web, SOAP ou REST
  - Client : permet à un programme ILE d'appeler un service web, SOAP ou REST
- Le serveur est un serveur Java WebSphere de type Liberty Profile
  - Embarque une application permettant de créer dynamiquement des Web Services
  - L'outil permet l'utilisation d'assistants aussi bien que de scripts shell

SOAP vs REST : principe

- Services Web de type REST
  - Basé sur l'architecture Web
    - HTTP
    - URI
  - Les données en entrée sont encodées soit l'URL, soit en JSON ou XML dans la requête
  - Les données en sortie sont encodées en JSON ou en XML
- Services Web de type WS-*
  - Standards supplémentaires mis en œuvre
    - SOAP (Simple Object Access Protocol)
    - WSDL (Web Service Description Language)
    - UDDI (Universal Description Discovery and Integration)
  - Les données sont encodées en XML
- Performances IWS
  - Comparables

SOAP vs REST : exemple

Démo

```
POST http://neptune:10054/web/services/bankService/bank HTTP/1.1
Accept-Encoding: gzip,deflate
Content-Type: application/soap+xml;charset=UTF-8
Content-Length: 344
Host: neptune:10054
Connection: Keep-Alive
User-Agent: Apache-HttpClient/4.1.1 (java 1.5)

<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope" xmlns:bank="http://bank.univ2016/">
   <soap:Header/>
   <soap:Body>
      <bank:getoperationarray>
         <arg0>
            <p_ID>1</p_ID>
            <p_MAXRESULT>20</p_MAXRESULT>
         </arg0>
      </bank:getoperationarray>
   </soap:Body>
</soap:Envelope>

-------------------------------------------------------------------------------------------------------

GET http://neptune:10054/web/services/bankrest/operations/1/20 HTTP/1.1
Accept-Encoding: gzip,deflate
Host: neptune:10054
Connection: Keep-Alive
User-Agent: Apache-HttpClient/4.1.1 (java 1.5)
```

Démo

- Déployer une procédure ILE en tant que Web Service
  - SOAP
  - REST

SOAP vs REST : usages

- SOAP
  - Plus normatifs (interopérabilité garantie), ils sont très utilisés dans les échanges de flux entre applications de gestion
  - Pour appeler un Web Service SOAP dans une application Android, il faut ajouter des bibliothèques supplémentaires (la validation XML n’est pas prise en charge par le runtime Android + constitution du message SOAP)
- REST
  - Plus légers, ils sont très utilisés dans les applications mobiles, l’Open Data, …
  - A l’inverse, un Web Service REST ne permet pas la validation des données ou le support d’extensions spécifiques (WS-Security par exemple)

Possibilités de IWS - Commun

- Plusieurs instances de serveurs possibles
- Arrêt et démarrage individuel des services
- Authentification basique HTTP
- Sécurisation via un certificat SSL, géré par
  - DCM (Digital Certificate Manager)
  - Par un magasin certificat dans l'IFS (keystore de type JKS, JCEKS, PKCS12, CMS)
- Gestion des traces et WebLog Monitor
  - Permet de déclencher des actions lorsque certains messages apparaissent dans la log
- User et liste de bibliothèques spécifique par service web
- Scripts shell permettant d'automatiser la plupart des actions
  - Création de serveur, déploiement de services, sauvegarde/restauration de serveurs et services, arrêt et démarrage de serveurs et services …
- Support des tableaux en entrée et sortie
  - Y compris les tableaux imbriqués

Possibilités de IWS - REST

- Choix du nom de la ressource et de l'URI
- Choix de la méthode HTTP
- Choix du formalisme de paramètres en entrée
  - **`*QUERY_PARAM`**, **`*PATH_PARAM`**, **`*FORM_PARAM`**, **`*COOKIE_PARAM`**, **`*HEADER_PARAM`**, **`*MATRIX_PARAM`**
  - Un mode « **`Wrap`** »
    - Obligatoire dès que l'on a une structure en entrée. Permet de d'encoder les paramètres sous forme JSON ou XML.
  - Ces possibilités sont combinables paramètre par paramètre
- Possibilité d'indiquer des valeurs par défaut pour les paramètres non reçus
- Choix de l'encodage en sortie : XML ou JSON
- Contrôle des valeurs HTTP : statut et valeurs d'entête
- Accès aux métadonnées suivantes : **`QUERY_STRING`** (paramètres sur l'URL), **`REMOTE_ADDR`** (adresse du client), **`REQUEST_METHOD`** (méthode HTTP), **`REQUEST_URI`**, **`REQUEST_URL`**, **`SERVER_NAME`**, **`SERVER_PORT`**

Possibilités de IWS - SOAP

- Personnalisation de l’espace de nom (namespace)
- Personnalisation du WSDL
- Choix de la version SOAP : 1.1 et/ou 1.2

Synthèse

- Avantages
  - Respect des normes, garantie d’une meilleure interopérabilité
  - Peu de compétences spécifiques nécessaires pour déployer ses premiers services
  - Outils fournis (scripts shell) qui permettent l’exploitation
- Limites
  - Outil totalement automatique : peu de possibilités de personnalisation
  - Pour les web services SOAP, pas de support des extensions les plus utilisées (WS-Security ?)
  - Le retour de données binaires (image, document …) n’est pas supportée
  - Impossibilité de mettre à jour ou d’ajouter une méthode à un service déployé : il faut le recréer
  - Les limitations du PCML
  - Les limitations du ToolKit Java

DB2 – Procédures et fonctions externes

Présentation

- DB2 for i permet l'écriture de routines SQL : procédures et fonctions cataloguées
  - Soit en langage **`PL/SQL`** (**`Procedural Language`**)
  - Soit dans des langages natifs (**`CL`**, **`RPG`**, **`COBOL`**), et un catalogage **`SQL`**

- C'est un moyen particulièrement utilisé pour appeler des programmes (procédures) au travers d'un middleware SQL
  - SQL est un des plus importants standards de l'informatique
  - Sans nécessité de créer de code supplémentaire

Exemple

- Code programme

```
// prototype du programme
dcl-pi *n ;
 p_id                like( T_Operation.id_cpt ) const ;
 p_maxResult         int( 5 ) const ;
 p_rc                like( T_ReturnCode ) ;
end-pi;

// variables locales
dcl-ds  l_operationArray    likeds( T_Operation ) dim( ARRAY_SIZE ) inz ;
dcl-s   l_oprArray_length   int( 10 ) inz;
dcl-s   l_return int(10) ;

// Appel de la procédure getOperationArray
l_return = getOperationArray( p_id :
                              p_maxResult :
                              l_operationArray :
                              l_oprArray_length ) ;

// Result set à renvoyer ?
if ( l_return = SUCCESS and l_oprArray_length > 0 ) ;
  exec sql
    set result sets with return to client
      array :l_operationArray for :l_oprArray_length rows ;
endif ;
```

Exemple

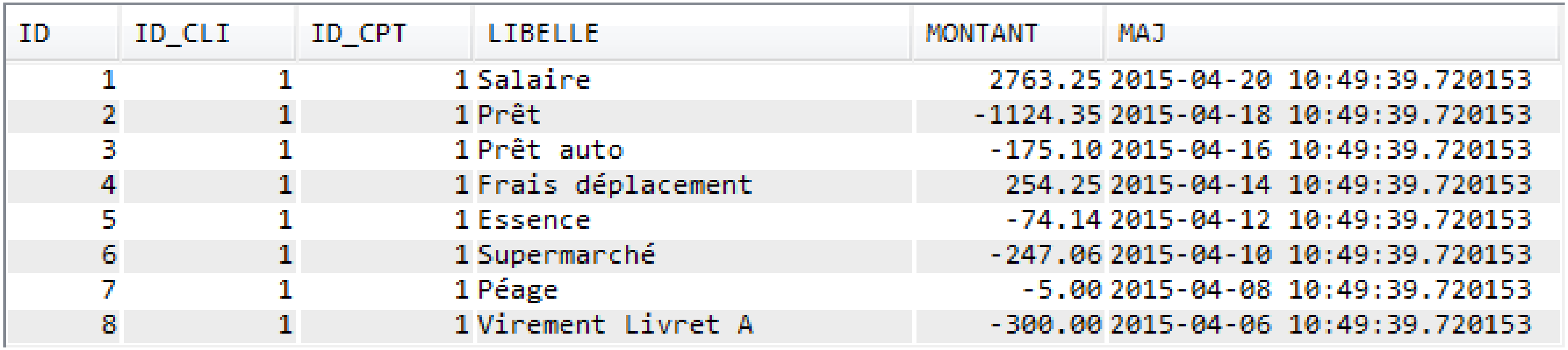
- Code script SQL de catalogage

```
create or replace procedure nb_univ_16.ps_lstoper (
  in    p_id        int,
  in    p_maxResult smallint default 20,
  inout p_rc        char(3)  default '   '
)

language rpgle
external name 'NB_UNIV_16/PS_LSTOPER'
parameter style general
result set 1
program type main
not deterministic ;
```

Exemple

- Appels

```
call ps_lstoper(1, 20, '   ') ;
```



ID	ID_CLI	ID_CPT	LIBELLE	MONTANT	MAJ
1	1	1	Salaire	2763.25	2015-04-20 10:49:39.720153
2	1	1	Prêt	-1124.35	2015-04-18 10:49:39.720153
3	1	1	Prêt auto	-175.10	2015-04-16 10:49:39.720153
4	1	1	Frais déplacement	254.25	2015-04-14 10:49:39.720153
5	1	1	Essence	-74.14	2015-04-12 10:49:39.720153
6	1	1	Supermarché	-247.06	2015-04-10 10:49:39.720153
7	1	1	Péage	-5.00	2015-04-08 10:49:39.720153
8	1	1	Virement Livret A	-300.00	2015-04-06 10:49:39.720153

Paramètres nommés et valeur par défaut

- Il est désormais possible d’indiquer
  - un nom au paramètre lors de l’appel
    - Permet de saisir les paramètres dans n’importe quel ordre
  - une valeur par défaut pour un paramètre
    - C’est-à-dire un paramètre optionnel à l’appel
- Syntaxes possibles

```
call ps_lstoper(1, 20, '   ') ;
call ps_lstoper(p_id => 1, p_maxResult => 20, p_rc => '   ') ;
call ps_lstoper(p_maxResult => 20, p_id => 1, p_rc => '   ') ;
call ps_lstoper(1, 20, '   ') ;
call ps_lstoper(1) ;
call ps_lstoper(1, 20) ;
call ps_lstoper(p_id => 1) ;
call ps_lstoper(p_maxResult => 20, p_id => 1) ;
call ps_lstoper(p_maxResult => 20, p_id => 1, p_rc => DEFAULT) ;
```

Surcharge

- Lisibilité et limiter la surcharge des procédures/fonctions

```
-- catalogage pour programme PS_LSTOPER
create or replace procedure ps_lstoper (
  in    p_id        int,
  in    p_maxResult smallint default 20,
  inout p_rc        char(3)
)

specific_name ps_lstoper
langage rpgle
... ;



-- catalogage pour programme PS_LSTOPER
create or replace procedure ps_lstoper (
  in    p_id        int,
  in    p_maxResult smallint default 20
)

specific_name ps_lstoper_norc
langage rpgle
... ;
```

Même nom de procédure : LST_OPER

Paramètres et nom spécifique distincts

Dernières évolutions PL/SQL

- La clause **`OR REPLACE`** a été ajoutée aux instructions

```
CREATE FUNCTION
CREATE PROCEDURE
```

- Facilite la maintenance des procédures et fonctions

```
CREATE OR REPLACE PROCEDURE xx/xx (yy …) ;
```

  - Au lieu de

```
DROP SPECIFIC PROCEDURE xx/xx ;
CREATE PROCEDURE xx/xx (yy …) ;
```

- Les maintenances sont nécessaires en cas de modification du programme
  - Principalement sur ces paramètres
  - Toutefois, il est recommandé de cataloguer à nouveau les procédures systématiquement pour maintenir les informations SQL associées au programme (**`PRTSQLINF`**)

Synthèse

- Avantages
  - Support en RPG des types de données SQL : **`CLOB`**, **`BLOB`**, **`DBCLOB`**, **`BINARY`**, **`VARBINARY`**, **`XML`**, **`ROWID`**
  - Pas de compétence supplémentaire à acquérir
  - Performance d'un protocole connecté, permettant de faire transiter de nombreuses données
  - Possibilité d'utiliser un certificat SSL pour la communication
  - Catalogage sauvegardé avec les objets programme / programme de service
- Limites
  - Les limitations de SQL
    - Pas de paramètre de type pointeur, pointeur de procédure
    - Pas de paramètre de type structure, tableau et datalink
      - Mais retour 0…n result sets

Technologies Open Source

De nombreuses autres possibilités

- CGI
  - Permet l’appel d’un programme RPG depuis un serveur HTTP Apache
- XMLService
  - Fonctionne en mode HTTP via CGI et DB2 via procédure cataloguée
  - Support de communication pour la plupart des langages
- Java
  - Via le support de JVM pour Power (PASE)
  - Websphere mais également d’autres serveurs Open Source (Glassfish, JBoss, …)
- PHP
  - Porté sur l’IBM i avec une implémentation spécifique : Zend Server
- Node.js
  - Distribué officiellement par IBM via le produit 5770-OPS
- Python
  - Distribué officiellement par IBM via le produit 5770-OPS
- Ruby on Rails (Power Ruby)
  - Distribution Open Source indépendante

Communication

- L’ensemble de ces produits
  - Communique via HTTP
  - Dispose de connecteur spécifique pour l’IBM i
- XMLService s’impose de fait comme un connecteur universel utilisable par n’importe quel langage capable de communiquer via HTTP

CGI – Common Gateway Interface

- Principe
  - Possibilité d’appeler un programme natif via un serveur, c’est-à-dire via une requête HTTP
  - Supporte les programmes
    - C++, REXX, ILE C, ILE RPG, and ILE COBOL
    - AIX via fastCGI
- Cela nécessite
  - De configurer le serveur HTTP afin de permettre le déclenchement de programmes
  - De concevoir des programmes capables de communiquer sur HTTP

    …

CGI – Exemple

- httpd.conf

```
...
Alias /ibm2016/        /home/NB/CGI/
ScriptAliasMatch /ibm2016pgm/(.*)     /qsys.lib/nb_univ_16.lib/$1
...
<Directory /QSYS.LIB/NB_UNIV_16.LIB>
   AllowOverride None
   Options -ExecCGI -FollowSymLinks -SymLinksIfOwnerMatch -Includes
-IncludesNoExec -Indexes -MultiViews
   order allow,deny
   allow from all
   IndexOptions -SelectiveDirAccess
   ServerUserID NB
   SetEnv QIBM_CGI_LIBRARY_LIST "QTEMP;QGPL;NB_UNIV_16;CGIDEV2"
</Directory>
...
```

CGI – Exemple RPG

```
 // pour utilisation CGIDEV2
/include CGIDEV2/qrpglesrc,hspecs
/include CGIDEV2/qrpglesrc,prototypeb
/include CGIDEV2/qrpglesrc,usec
/include CGIDEV2/qrpglesrc,variables3
 // pour utilisation du programme de service
/include ws_srv_inc

// variables globales
dcl-s  l_id              like( T_Operation.id_cpt )              inz ;
dcl-s  l_maxResult       uns( 5 )                                inz ;
dcl-ds l_operationArray  likeds( T_Operation ) dim( ARRAY_SIZE ) inz ;
dcl-s  l_oprArray_length int( 10 )                               inz ;
dcl-s  l_rc              like( T_ReturnCode )                    inz ;
dcl-s  l_return          int(10)                                 inz ;
dcl-s  l_cpt             uns(5)                                  inz ;

// Récupération des paramètres sur la requête : id et maxResult
nbrVars = zhbgetinput(savedquerystring : qusec) ;
l_id = %int( ZhbGetVar( 'id' ) ) ;
l_maxResult = %uns( ZhbGetVar( 'maxResult' ) ) ;

// Appel procéure getOperationArray
l_return = getOperationArray( l_id :
                              l_maxResult :
                              l_operationArray :
                              l_oprArray_length :
                              l_rc ) ;
```

CGI – Exemple RPG

```
// Template HTML
GetHtmlIfs( '/home/NB/CGI/Univ2016.html' : '<as400>' ) ;

// Paramètres en sortie :
updHTMLVar( 'ReturnValue' : %char( l_return ) ) ;
updHTMLVar( 'rc' : l_rc ) ;
updHTMLVar( 'oprArrayLen' : %char( l_oprArray_length ) ) ;
WrtSection( 'entete' ) ;

// Opérations
if l_oprArray_length = 0 ;
  WrtSection( 'vide');
else ;
  WrtSection( 'operation_start');
  for l_cpt = 1 to l_oprArray_length ;
    updHTMLVar( 'id'      : %char( l_operationArray( l_cpt ).id ) ) ;
    updHTMLVar( 'libelle' : l_operationArray( l_cpt ).libelle ) ;
    updHTMLVar( 'montant' : %char( l_operationArray( l_cpt ).montant ) );
    updHTMLVar( 'stamp'   : %char( l_operationArray( l_cpt ).maj ) ) ;
    WrtSection( 'operation' );
  endfor ;
  WrtSection( 'operation_end');
endif ;

// fin
WrtSection( '*fini' );
return;
```

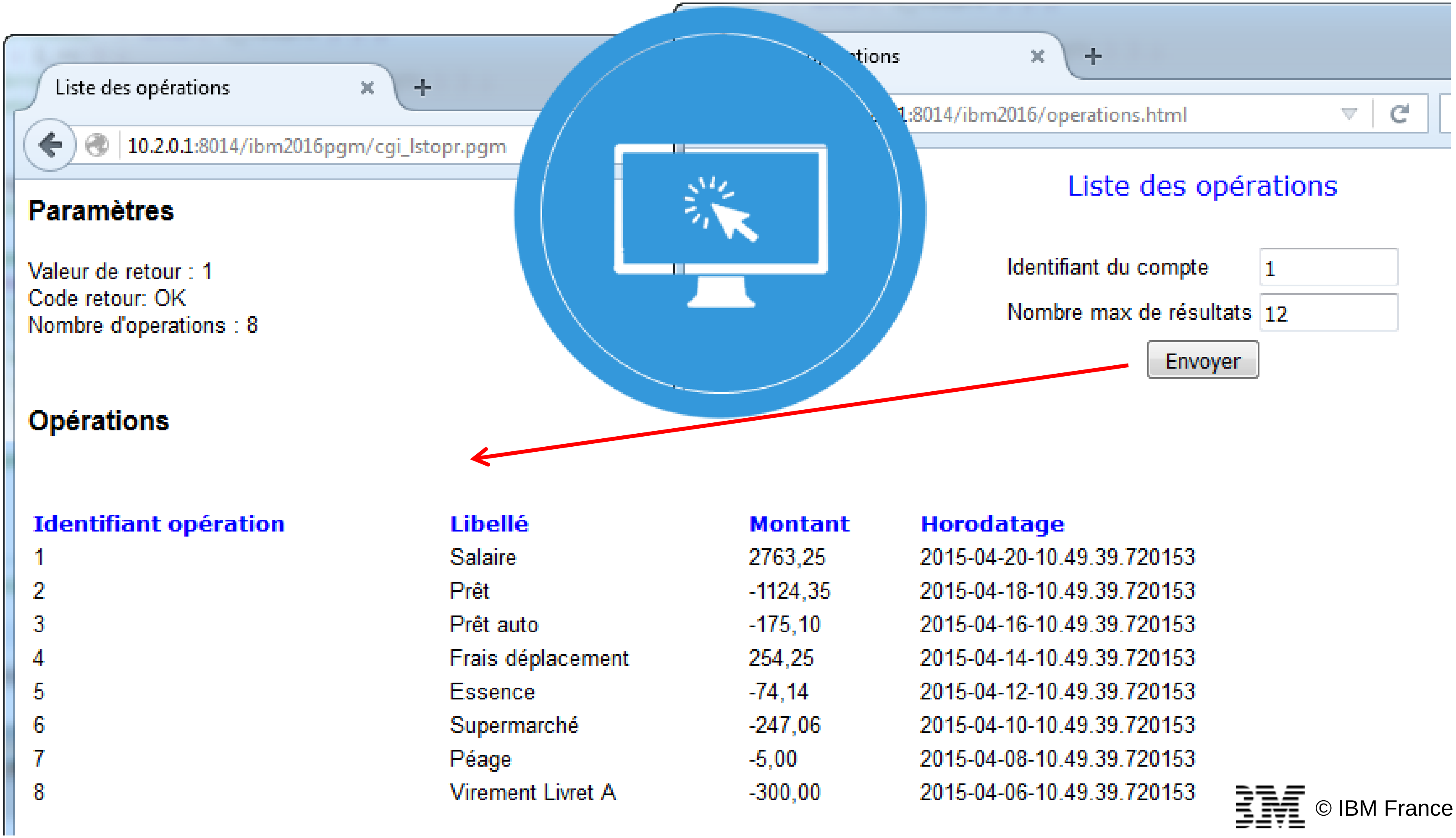
CGI – Exemple « template » HTML

```
<as400>entete                     *** SECTION "ENTETE"
Content-type: text/html
Expires: 0

<HTML>
...

<as400>operation_start            *** SECTION "OPERATION_START"
   <br>
      <table WIDTH=800 BORDER=0>
         <tbody>
            <tr align="left">
               <th><font face="Souvenir LT,Verdana"  color=blue size=2>Identifiant opération</font></th>
               <th><font face="Souvenir LT,Verdana"  color=blue size=2>Libellé</font></th>
               <th><font face="Souvenir LT,Verdana"  color=blue size=2>Montant</font></th>
               <th><font face="Souvenir LT,Verdana"  color=blue size=2>Horodatage</font></th>
            </tr>

<as400>operation                  *** SECTION "OPERATION"
            <tr>
               <td><font face="Helvetica,Helv,Arial" size=2>/%id%/</font></td>
               <td><font face="Helvetica,Helv,Arial" size=2>/%libelle%/</font></td>
               <td><font face="Helvetica,Helv,Arial" size=2>/%montant%/</font></td>
               <td><font face="Helvetica,Helv,Arial" size=2>/%stamp%/</font></td>
            </tr>

<as400>operation_end              *** SECTION "OPERATION_END"
         </tbody>
      </table>
   </body>
</html>
```

CGI – Appel

- Formulaire HTML

  `http://10.2.0.1:8014/ibm2016/operations.html`



Liste des opérations

Identifiant du compte: 1
Nombre max de résultats: 12

Paramètres

Valeur de retour : 1
Code retour: OK
Nombre d'operations : 8

Opérations

Identifiant opération	Libellé	Montant	Horodatage
1	Salaire	2763,25	2015-04-20-10.49.39.720153
2	Prêt	-1124,35	2015-04-18-10.49.39.720153
3	Prêt auto	-175,10	2015-04-16-10.49.39.720153
4	Frais déplacement	254,25	2015-04-14-10.49.39.720153
5	Essence	-74,14	2015-04-12-10.49.39.720153
6	Supermarché	-247,06	2015-04-10-10.49.39.720153
7	Péage	-5,00	2015-04-08-10.49.39.720153
8	Virement Livret A	-300,00	2015-04-06-10.49.39.720153

CGI – Synthèse

- Avantages et inconvénients
  - Technologie installée et bien outillée
  - Nécessite d’écrire des programmes RPG adaptés à un appel via CGI : l’utilisation d’APIs pour permettre la manipulation des variables d’environnements, des requêtes HTTP entrantes et sortantes …
  - La gestion des CCSID peut également être complexe
  - Des outils existent pour faciliter la mise en œuvre des programmes CGI qui nécessitent du code complexe
    - Le plus connu : CGIDEV2

XMLService - Principe

- XMLService
  - Produit RPG Open Source (cf YIPs)
  - Permet l'utilisation de ressources IBM i (programme, commande, SQL …) via un échange de messages XML
    - Stateless ou Statefull
  - Utilisable
    - Via procédures stockées DB2
    - Via HTTP / CGI
  - Utiliser dans la plupart de Toolkits des nouveaux langages Open Source disponibles (PHP, Node.js, Python, Power Ruby)
  - Peut être utilisé par n'importe quel langage sachant communiquer via HTTP ou DB2 (.Net par exemple, mais également RPG…)
- En synthèse, c'est un outil qui automatise ce que nous utilisons depuis longtemps : procédure cataloguée DB2 et CGI

XMLService - ILE

- Via un formulaire HTTP

```
<input type="hidden" name="xmlin" value="<?xml version='1.0'?>
<pgm name='WS_SRV' lib='NB_UNIV_16' func='GETOPERATIONARRAY' mode='ile'>
  <parm comment='id' by='ref' io='in'>
    <data type='10i0'>1</data>
  </parm>
  <parm comment='maxResult' by='ref' io='in'>
    <data type='5u0'>20</data>
  </parm>
  <parm comment='operationArray' io='out' by='ref'>
    <ds dim='20'>
      <data comment='id' type='10i0'></data>
      <data comment='id_cli' type='10i0'></data>
      <data comment='id_cpt' type='10i0'></data>
      <data comment='libelle' type='25a'></data>
      <data comment='montant' type='10p2'></data>
      <data comment='maj' type='26a'></data>
    </ds>
  </parm>
  <parm comment='p_oprArray_length' by='ref' io='out'>
    <data comment='p_oprArray_length' type='10i0'></data>
  </parm>
  <parm io='out' by='ref'>
    <data comment='rc' type='3A'></data>
  </parm>
  <return><data comment='return' type='10i0'></data></return>
</pgm>">
```

XMLService - ILE

- Retourne

```
<?xml version='1.0'?>
<pgm name='WS_SRV' lib='NB_UNIV_16' func='GETOPERATIONARRAY' mode='ile'>
    <parm comment='operationArray' io='out' by='ref'>
        <ds dim='20'>
            <data comment='id' type='10i0'>1</data>
            <data comment='id_cli' type='10i0'>1</data>
            <data comment='id_cpt' type='10i0'>1</data>
            <data comment='libelle' type='25a'>Salaire</data>
            <data comment='montant' type='10p2'>2763,25</data>
            <data comment='maj' type='26a'>2015-04-20-10.49.39.720153</data>
        </ds>
        <ds dim='20'>..
        <ds dim='20'>..
        ...
        <ds dim='20'>..
    </parm>
    <parm comment='p_oprArray_length' by='ref' io='out'>
        <data comment='p_oprArray_length' type='10i0'>8</data>
    </parm>
    <parm io='out' by='ref'>
        <data comment='rc' type='3A'></data>
    </parm>
    <return>
        <data comment='return' type='10i0'>1</data>
    </return>
    <success><![CDATA[+++ success NB_UNIV_16 WS_SRV GETOPERATIONARRAY ]]></success>
</pgm>
```

Démo XMLService

- Démo
  - http://10.2.0.1:2500/univ2016/operations.html
  - http://10.2.0.1:2500/univ2016/getOperationArray.html
  - SOAPUI
  - SQL

XMLService – Synthèse

- Avantages et inconvénients
  - Ne nécessite pas d'adaptation des programmes à appeler
  - Ne nécessite pas de code supplémentaire pour exposer un programme
  - Ne sait pas utiliser PCML
    - Possibilité d'automatiser la construction du template de flux XML à partir du PCML par programmation ?
    - Permet surtout de contourner les limites de PCML
  - Capacités statefull/stateless
  - Capacités HTTP/DB2
  - Performances ?

Java

- Principe
  - Un toolbox natif est fourni pour permettre l'utilisation des différentes ressources IBM i : connecteur JDBC DB2 pour SQL, appel de programme/procédure, exécution de commande, manipulation des différents types d'objets (*DTAARA, …)
    - IWS s'appuie sur cet outil
  - Plusieurs versions disponibles
    - jt400.jar, jt400android
      - Permet de s'appuyer sur PCML
    - jtopen, jtopenlite

Java - Exemple

```
// Objet de connexion à l'IBM i
as400System = new AS400("neptune", "NB", "MonMotDePasse");

try
{
   // Construction ProgramCallDocument
   pcml = new ProgramCallDocument(as400System, "pcml.ws_srv_opr.pcml");
   pcml.setPath("GETOPERATION", "/QSYS.LIB/NB_UNIV_16.LIB/WS_SRV.SRVPGM") ;

   // Alimentation des paramètres input et inputoutput
   pcml.setValue("GETOPERATION.P_ID", new Integer(1));
   pcml.setValue("GETOPERATION.P_OPERATION.ID", new Integer(0));
   pcml.setValue("GETOPERATION.P_OPERATION.ID_CLI", new Integer(0));
   pcml.setValue("GETOPERATION.P_OPERATION.ID_CPT", new Integer(0));
   pcml.setValue("GETOPERATION.P_OPERATION.LIBELLE", "");
   pcml.setValue("GETOPERATION.P_OPERATION.MONTANT", 0) ;
   pcml.setValue("GETOPERATION.P_OPERATION.MAJ", "0001-01-01T00:00:00.000000000") ;
   pcml.setValue("GETOPERATION.P_RC", "");

   // Appel à la procédure
   rc = pcml.callProgram("GETOPERATION");
```

Java - Exemple

```
   // messages d'erreur reçus du serveur ?
   if(rc == false)
   { ... }
   else
   {
      System.out.println("GETOPERATION.P_OPERATION.ID:      " +
         pcml.getValue("GETOPERATION.P_OPERATION.ID"));
      System.out.println("GETOPERATION.P_OPERATION.ID_CLI:  " +
         pcml.getValue("GETOPERATION.P_OPERATION.ID_CLI"));
      System.out.println("GETOPERATION.P_OPERATION.ID_CPT:  " +
         pcml.getValue("GETOPERATION.P_OPERATION.ID_CPT"));
      System.out.println("GETOPERATION.P_OPERATION.LIBELLE: " +
         pcml.getValue("GETOPERATION.P_OPERATION.LIBELLE"));
      System.out.println("GETOPERATION.P_OPERATION.MONTANT: " +
         pcml.getValue("GETOPERATION.P_OPERATION.MONTANT"));
      System.out.println("GETOPERATION.P_OPERATION.MAJ:     " +
         pcml.getValue("GETOPERATION.P_OPERATION.MAJ"));
    }
}
catch (PcmlException e)
{ ... }
```

Java - Synthèse

- Le toolbox Java est le plus complet
  - DB2 SQL
  - RLA
  - Commandes CL
  - Appels de programmes et procédures
  - Pcml
  - Mais aussi la manipulation de nombreux types d'objets natifs
- Une déclinaison
  - Lite : permettant de réduire l'emprunte et d'utiliser les principale fonctions (SQL, appels de programmes/procédures)
  - Android

PHP

- Principe
  - Portage natif pour IBM i par Zend en partenariat avec IBM
  - Implémenter sous forme de serveur : Zend Server
  - Dispose d’une extension XMLToolKit permettant l’accès aux ressources natives IBM i
    - DB2, programmes, procédures, commandes, objets …
    - Basé sur XMLService

PHP - Exemple

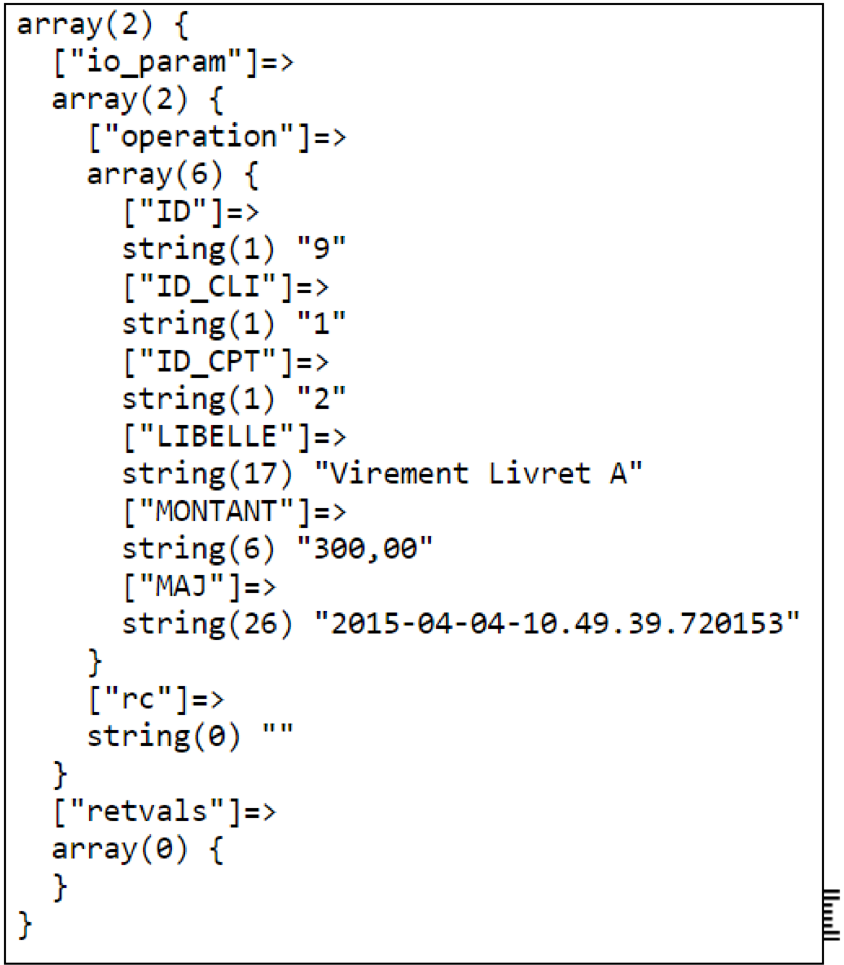
```
// Connexion à la BD
$ToolkitServiceObj = ToolkitService::getInstance('*LOCAL', 'NB', 'MonMotDePasse');
$ToolkitServiceObj->setOptions(
   array('dataStructureIntegrity' => true, 'arrayIntegrity' => true, 'stateless'=>true));

// Alimentation des paramètres pour appel procédure getOperation
$param[] = $ToolkitServiceObj->AddParameterInt32(
   'in', 'identifiant operation', 'id', '9');
$op[] =  $ToolkitServiceObj->AddPArameterInt32(
   'both', 'id', 'ID', '0');
$op[] =  $ToolkitServiceObj->AddPArameterInt32(
   'both', 'id_cli', 'ID_CLI', '0');
$op[] =  $ToolkitServiceObj->AddPArameterInt32(
   'both', 'id_cpt', 'ID_CPT', '0');
$op[] =  $ToolkitServiceObj->AddParameterChar(
   'both', 25, 'LIBELLE', 'LIBELLE','');
$op[] =  $ToolkitServiceObj->AddParameterPackDec(
   'both', 10, 2, 'MONTANT', 'MONTANT', '0.0' );
$op[] =  $ToolkitServiceObj->AddParameterChar(
   'both', 26, 'MAJ', 'MAJ', '0001-01-01-00.00.00.000000');
$param[] = $ToolkitServiceObj->AddDataStruct($op, 'operation') ;
$param[] = $ToolkitServiceObj->AddParameterChar(
   'both', 3, 'Code retour','rc', '   ');
```

http://10.2.0.1:10080/apps/univ2016/getOperation.php

PHP - Exemple

```
// Appel procédure
$OutputParams = $ToolkitServiceObj->PgmCall(
   'WS_SRV', "NB_UNIV_16", $param, NULL, array('func'=>'GETOPERATION') );
var_dump($OutputParams);

// Déconnexion
$ToolkitServiceObj->disconnect();
?>
```

- Retourne



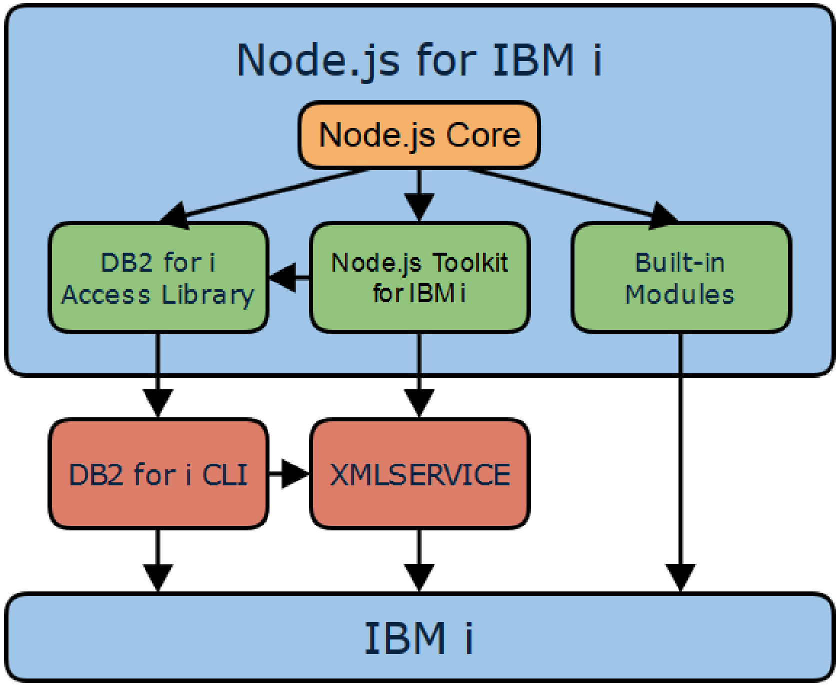
```
array(2) {
  ["io_param"]=>
  array(2) {
    ["operation"]=>
    array(6) {
      ["ID"]=>
      string(1) "9"
      ["ID_CLI"]=>
      string(1) "1"
      ["ID_CPT"]=>
      string(1) "2"
      ["LIBELLE"]=>
      string(17) "Virement Livret A"
      ["MONTANT"]=>
      string(6) "300,00"
      ["MAJ"]=>
      string(26) "2015-04-04-10.49.39.720153"
    }
    ["rc"]=>
    string(0) ""
  }
  ["retvals"]=>
  array(0) {
  }
}
```

PHP - Synthèse

- Pcml
  - Non utilisé par le toolkit
  - Pris en charge par l'ancien toolkit : possibilité d'utiliser cw (compatibility wrapper)
- Toutes les possibilités et limites de XMLService
- Il existe également un connecteur DB2 pour PHP
  - Préférable pour les accès SQL

Node.js

- Principe
  - Distribué sous l'option 1 de 5733OPS (Open Source for IBM i)
  - Projet Open Source basé sur le moteur Javascript V8 de Chrome (Google)
  - Peut être utilisé sous forme d'extenson fastCGI d'une instance de serveur HTTP Apache
  - Dispose des extensions suivantes
    - DB2 for i Node.js
      - Basé sur CLI (Call Level Interface)
    - IBM i ILE object toolkit
      - Basé sur XMLService



Node.js – Exemple iToolkit

- Script js

```
var http = require('http');
var xt = require('/QOpenSys/QIBM/ProdData/Node/os400/xstoolkit/lib/itoolkit');

http.createServer(function (req, res) {
// Connexion
var conn = new xt.iConn("*LOCAL");
var pgm = new xt.iPgm("WS_SRV", {"lib":"NB_UNIV_16", "func":"GETOPERATIONARRAY"});
// Préparation des paramètres
pgm.addParam("1","10i0");   // id
pgm.addParam("20","5u0")    // maxResult
pgm.addParam([              // operationArray
              [0, "10i0"],
              [0, "10i0"],
              [0, "10i0"],
              ["", "25A"],
              [0, "10p2"],
              ["0001-01-01-00.00.00.000000", "26a"]
       ], {"dim":"20"});
pgm.addParam("0","10i0")    // oprArray_length
pgm.addParam("", "3a");     // rc

conn.add(pgm.toXML());
```

Node.js – Exemple iToolkit

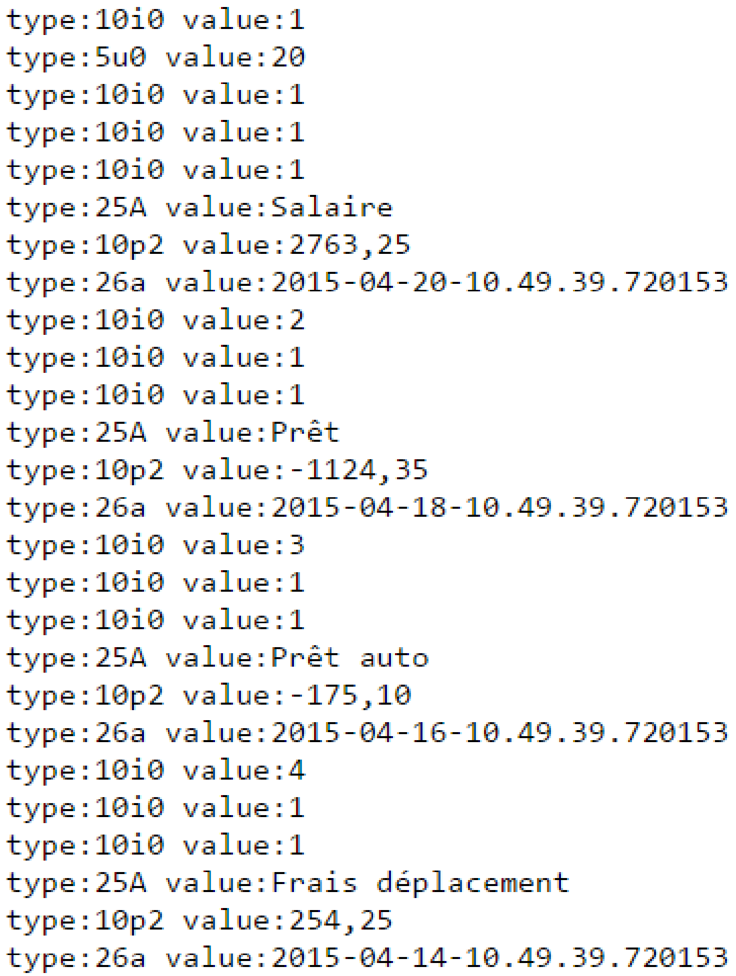
```
// Appel du programme
conn.run(function (rsp) {
// Affichage du résultat
 var results = xt.xmlToJson(rsp);
 res.writeHead(200, {'Content-Type': 'text/plain'});
 results.forEach(function(result, index){
   result.data.forEach(function(data, index2){
     res.write("type:" + data.type + " value:" + data.value + "\n");
   });
 });
 res.end();
})

}).listen(8085, '10.2.0.1');
console.log('Server running at http://10.2.0.1:8085/');
```

- Lancement

```
SBMJOB CMD(QSH CMD('node /home/nb/ibm2016/getOperationArray.js'))
```

Node.js – Exemple iToolkit

- Résultat
  - URL : http://10.2.0.1:8085/



```
type:10i0 value:1
type:5u0 value:20
type:10i0 value:1
type:10i0 value:1
type:10i0 value:1
type:25A value:Salaire
type:10p2 value:2763,25
type:26a value:2015-04-20-10.49.39.720153
type:10i0 value:2
type:10i0 value:1
type:10i0 value:1
type:25A value:Prêt
type:10p2 value:-1124,35
type:26a value:2015-04-18-10.49.39.720153
type:10i0 value:3
type:10i0 value:1
type:10i0 value:1
type:25A value:Prêt auto
type:10p2 value:-175,10
type:26a value:2015-04-16-10.49.39.720153
type:10i0 value:4
type:10i0 value:1
type:10i0 value:1
type:25A value:Frais déplacement
type:10p2 value:254,25
type:26a value:2015-04-14-10.49.39.720153
```

Node.js - Synthèse

- Tous les avantages et limites de XMLService …

Python

- Principe
  - Distribué par le produit 5770 OPS
  - Nécessite l’extension itoolkit permettant l’accès aux ressources IBM i natives
    - Basé sur XMLService
    - SQL, appel de commandes (y compris interactives), de programmes …

Python – Exemple ILE

```
@get('/ile')
def cmd_toolkit():

   itransport = iLibCall()
   itool = iToolKit()
   itool.add(iCmd('chgcurlib', 'CHGCURLIB CURLIB(NB_UNIV_16)'))
   itool.add(
    iSrvPgm('getOperationArray','WS_SRV','GETOPERATIONARRAY')
    .addParm(iData('id','10i0','1'))
    .addParm(iData('maxResult','5u0','20'))
    .addParm(
     iDS('operationArray',{'dim':'20'})
     .addData(iData('ID','10i0','0'))
     .addData(iData('ID_CLI','10i0','0'))
     .addData(iData('ID_CPT','10i0','0'))
     .addData(iData('LIBELLE','25a',''))
     .addData(iData('MONTANT','10p2','0.0'))
     .addData(iData('MAJ','26a','0001-01-01-00.00.00.000000'))
     )
    .addParm(iData('p_oprArray_length','10i0','0'))
    .addParm(iData('rc','3a','   '))
    )
```

Python – Exemple ILE

```
# xmlservice
itool.call(itransport)

# output
chgcurlib = itool.dict_out('chgcurlib')
if 'success' in chgcurlib:
  print (chgcurlib['success'])
else:
  print (chgcurlib['error'])
exit()
```

Python - Synthèse

- Tous les avantages et limites de XMLService …

Ruby on Rails (Power Ruby)

- Principe
  - Portage pour IBM i de Ruby on Rails
  - Utilise une instance de serveur HTTP Apache natif
  - Intègre un framework capable de générer du code : modèle, vue, contrôleur
  - Un connecteur db2 et un toolkit pour XMLService sont est fournis
  - Non porté et distribué par IBM, mais projet par Aaron Bartell (Krengel Technology, Litmis)

Ruby on Rails (Power Ruby) - Exemple

```
# File: getOperationArray.rb
require "./auth"

getOpArr = XMLService::I_SRVPGM.new("WS_SRV","GETOPERATIONARRAY",
                                    $toolkit_test_lib,{'error'=>'on'})
getOpArr << XMLService::I_Int32.new('id', 1)
getOpArr << XMLService::I_Uint16.new('maxResult', 20)
getOpArr << XMLService::I_DS.new("operationArray", 20, 'p_oprArray_length',
            [
              XMLService::I_Int32.new('ID', 0),
              XMLService::I_Int32.new('ID_CLI', 0),
              XMLService::I_Int32.new('ID_CPT', 0),
              XMLService::I_Char.new('LIBELLE', 25, ''),
              XMLService::I_PackedDecimal.new('MONTANT', 10, 2, 0.0),
              XMLService::I_Char.new('MAJ', 26, '0001-01-01-00.00.00.000000')
            ])
getOpArr << XMLService::I_Int32.new('p_oprArray_length', 0, 'p_oprArray_length')
getOpArr << XMLService::I_Char.new('rc', 3, '   ')
getOpArr.setReturn("integer", XMLService::I_Int32.new('id', 0))

# Appel
getOpArr.call
```

Ruby on Rails (Power Ruby) - Exemple

```
# Erreur ?
rc = getOpArr.xmlservice_error
if rc
   puts getOpArr.dump_error
end

# output
puts " p_oprArray_length....#{getOpArr.response.p_oprArray_length}\n"
i = 0
getOpArr.response.operationArray.each do |operationArray|
  i += 1
  puts "operationArray[#{i}]"
  puts "  ID...#{operationArray.ID}"
  puts "  ID_CLI...#{operationArray.ID_CLI}"
  puts "  ID_CPT...#{operationArray.ID_CPT}"
  puts "  LIBELLE...#{operationArray.LIBELLE}"
  puts "  MONTANT...#{operationArray.MONTANT}"
  puts "  MAJ...#{operationArray.MAJ}"
end
```

Ruby on Rails (Power Ruby)

- Tous les avantages et limites de XMLService …

Choisir

Le pire choix … ne pas choisir

- Les langages open source comme Node.js, Python, Java, PHP, Ruby ..
  - Disposent d'une forte communauté d'utilisateurs
  - Avec la mise à disposition rapide des nouvelles versions ou nouvelles fonctionnalités
  - Capacité à trouver rapidement des développeurs
- Attention au niveau d'industrialisation souhaité
  - Ces technologies bougent très régulièrement
  - Dans tous les cas nous n'avons pas la stabilité dans le temps d'un code RPG

Critères

- Support IBM officiel ?
- Utilisation PCML ?
- Communauté ?
- Compétences supplémentaires ?
- Possibilités, personnalisation ?
- Difficultés ?
- Performances
  - Connecté, non connecté

Merci
Q/R

Ressources

- DeveloperWorks
  - https://www.ibm.com/developerworks/community/wikis/home?lang=en#!/wiki/IBM%20i%20Technology%20Updates
- SQL procédure cataloguée
  - http://www.ibm.com/support/knowledgecenter/ssw_ibm_i_72/sqlp/rbafysproeg.htm?lang=fr
- Apache
  - http://www.redbooks.ibm.com/redbooks/pdfs/sg246716.pdf
- CGIDEV2
  - http://www.easy400.net/cgidev2/start
- XMLService
  - http://yips.idevcloud.com/wiki/index.php/XMLService/XMLSERVICE
- JTOpen
  - http://jt400.sourceforge.net/
  - http://www-03.ibm.com/systems/power/software/i/toolbox/
- PHP
  - http://www.zend.com/en/products/server/downloads#IBM%20i
- Node.js
  - https://nodejs.org/en/
- Python
  - https://www.python.org/
- Ruby
  - https://powerruby.com/