IBM Power Systems - IBM i

Modernisation, développement d'applications et DB2 sous IBM i
Technologies, outils et nouveautés 2013-2014

13 et 14 mai 2014 – IBM Client Center Paris, Bois-Colombes

S26 – Saviez-vous que vous pouviez faire cela en RPG ?

Mercredi 14 mai – 15h15-16h45

Philippe Bourgeois – IBM France

Plan de la présentation

- 1. Synthèse des nouveautés V3R1 à V7R2
- 2. Appel de fonctions C et de méthodes Java
- 3. Accès à des bases de données externes
- 4. Lecture de flux XML
- 5. Invocation de Services Web
- 6. Développement Web avec CGI
- 7. Ouverture avec RPG Open Access

1. Synthèse des nouveautés V3R1 à V7R2

V3R1

- Apparition du RPG IV
- Noms de longueur 10
- Support des minuscules
- Déclaration en spécifications D
- Expressions, code-opération EVAL
- Fonctions intégrées (BIF – Build In Functions)
- Gestion des dates
- Support de l'ILE
- Outil de conversion RPG/400 → RPG IV

V3R2/V3R6 – V3R7

- V3R2/V3R6
  - Nouveaux types de données : entiers signés (I) et non-signés (U)
  - Procédures prototypées
    - Création de ses propres fonctions
    - Déclaration de variables locales
    - Modules multi-procédures
- V3R7
  - Support des valeurs indéfinies (ALWNULL - %NULLIND)
  - Nouveau type de données : virgule flottante (F)
  - Nouvelles fonctions intégrées (%EDITC…, %INT…, %SCAN, %ABS)
  - Support des noms de longueur > 10
  - Compilation conditionnelle
  - Support des /COPY imbriqués

V3R2/V3R6 – Exemple de sous-procédure

PR_CALCJ4.RPGLE

```
Ligne 2        Colonne 1      Remplacement
      .....DName++++++++++ETDsFrom+++To/L+++IDc.Keywords+++++++++++++
000001      H datfmt(*eur) dftactgrp(*no) actgrp(*new)
000203      D date1            s                d
000206      D jourSemaine      pr               1  0
000208      D                                   d   const
000211      C     *entry         plist
000212      C                    parm                    date1
000213       /free
000215        dsply jourSemaine(date1 + %days(1));
000219        *inlr = *on;
000409       /end-free
000410        //-------------------------------------------------------
000411        // Calcul du jour de la semaine (1=Lundi, 2=Mardi, etc.)
000901      P jourSemaine      b
000902      D                  pi               1  0
000904      D  datew                            d   const
000906      D  un_dimanche     c                   d'11.03.2012'
000907      D  nbjw            s                7  0
000908      D  jourw           s                1  0
000909       /free
000911        nbjw = %diff(datew:un_dimanche:*d);
000912        jourw = %rem(nbjw:7);
000913        if jourw < 1;
000914          jourw = jourw + 7;
000915        endif;
000916        return jourw;
000917       /end-free
000918      P                  e
```

V4R2

- Options de compilation en spécification H
- Zones de longueur variable (varying)
- Nouveau type de données : indicateur (N)
- Nouvelles fonctions intégrées pour remplacer les indicateurs
  - %EOF, %FOUND, %ERROR, %EQUAL, %STATUS, %OPEN
- « Nommage » des indicateurs des DSPF : mot-clé INDDS
- Nouvelles fonctions intégrées : %CHAR, %REPLACE

V4R4

- Nouveaux codes-opération : EVALR, LEAVESR
- Boucles avec FOR / ENDFOR
- OVERLAY(ds:*NEXT)
- Nouvelles fonctions intégrées : %XFOOT, %DIV
- Support d'Unicode (type de données UCS-2)
- H option(*SRCSTMT:*NODEBUGIO)

V5R1

- Instructions de traitement en format libre (/free et /end-free)
- Structures de données qualifiées (QUALIFIED)
- Interception des erreurs par MONITOR
- Possibilité d'appeler des méthodes Java
- Nouvelles fonctions intégrées de gestion des dates (%DIFF, %DAYS, %YEARS…, %SUBDT, %DATE…)
- Nouveau code-opération ELSEIF
- Directives de compilation conditionnelles
- Nouvelles fonctions intégrées : %XLATE, %LOOKUPxx, %OCCUR, %CHECK, %CHECKR
- Ouverture dynamique des fichiers (EXTFILE, EXTMBR)
- Définition d'une DS par rapport à un format (LIKEREC)

V5R2

- Conversion alpha vers numérique avec %DEC, %INT…
- Opérateurs d'affectation raccourcis (a+=b;)
- Clé fractionnée en format libre et fonction %KDS
- Les sources RPGLE peuvent être placés dans l'IFS
- Support du PCML
- Mise à jour sélective avec la fonction %FIELDS
- DS en zone résultat des codes-opération CHAIN, READx…
- Imbrication des tableaux et DS à tout niveau
- Support des DTAARA qualifiées par le nom de la bibliothèque

V5R3

- Choix des caractères à supprimer dans les fonctions %TRIMx
- Conversion des dates/heures/horodates en numérique avec %DEC
- Accès à un sous-ensemble d'un tableau avec %SUBARRAY
- Suppression des espaces des valeurs des paramètres d'une procédure : OPTIONS(*TRIM)
- Possibilité d'indiquer les zones à inclure des DS externes : EXTNAME(fic{:fmt}{:*ALL|*INPUT|*OUTPUT|*KEY})
- La longueur maxi d'une zone numérique passe à 63 digits

V5R4

- Nouveau code-opération EVAL-CORR
- Lecture de flux XML : XML-INTO, XML-SAX, %XML et %HANDLER
- Support des instructions SQL en format libre (exec sql xxx)
- Transmission des valeurs nulles pour les paramètres de procédures : OPTIONS(*NULLIND)

V6R1

- Support des fichiers locaux dans les procédures
- Support du multi-threading
- Formats des fichiers qualifiés: mot-clé QUALIFIED
- Mot-clé LIKEFILE en spécification F
  - Récupération des attributs d'un autre fichier
  - Passage d'un fichier en paramètre d'une procédure
- DS ou fichier « modèle » : mot-clé TEMPLATE
- Compilation dynamique des fichiers : mot-clé EXTDESC
- Indication de la procédure principale : H MAIN(xxx)
- Possibilité de stocker le PCML dans l'objet *MODULE
- Possibilité de différer le chargement d'un programme de service
- Extension des limites

V7R1

- Nouvelle fonction intégrée %SCANRPL
- Nouveau paramètre *MAX pour la fonction %LEN
- Nouvelle fonction intégrée %PARMNUM pour connaître le numéro de passage d'un paramètre défini comme optionnel
- Tri et recherche sur des tableaux de structures de données
- Tri des tableaux en ascendant ou descendant : SORTA(A) et SORTA(D)
- Support des noms longs dans les DS externes : mot-clé ALIAS
- Conversion Unicode implicite des paramètres d'une procédure
- Prototypes des procédures optionnels dans les sources qui implémentent ces procédures
- Protection de la vue listing lors du lancement du débogage (paramètre de compilation DBGENCKEY)
- Nouveau RPG Free-Form (spécifications H, F, D, C et P en format libre)

V7R1 – Le nouveau RPG Free-Form

- Annoncé en décembre 2013 en 7.1 (PTF SI51094)
- En standard en 7.2
- Spécifications H, F, D, C et P en format libre
- Supporté à partir de RDi 9.0.1

```
ctl-opt option(*srcstmt:*nodebugio);
 dcl-f  fic1;
 dcl-s  z1 char(10);
 dcl-ds ds1 likerec(f1);
 dcl-f  fic2 printer;
 read fic1 ds1;
 . . . / . . .
 *inlr = *on;
dcl-proc p1;
  . . . / . . .
end-proc;
```

Plus de /free et /end-free

On peut mixer
spécifications F et D

H : CTL-OPT
F : DCL-F
D : DCL-S, DCL-DS, …
P : DCL-PROC

Objectifs
- Faciliter l'apprentissage du langage RPG
- En faire un langage « actuel »

Conversion de code RPG vers RPG Free-Form
- ARCAD Transformer RPG
- Linoma RPG Toolbox

V7R1 - Le nouveau RPG Free-Form – ARCAD Transformer RPG

- Outil de conversion automatique de code RPG IV vers du code RPG Free-Form
- Conversion unitaire ou de masse
  - Plugin RDi V9
  - Commande batch ACVTRPGFRE

```
H dftactgrp(*no) bnddir('QC2LE')

D runcmd          pr             5i 0 extproc('system')
D  cmd                            *   value options(*string)

D errmsgid        s              7    import('_EXCP_MSGID')
D returncode      s              5i 0
```

```
Free-Form++++++++++++++++++++++++++++++++++++++++++++++++
ctl-opt dftactgrp(*no) bnddir('QC2LE');

dcl-pr runcmd int(5) extproc('system');
  cmd pointer value options(*string);
end-pr;

dcl-s errmsgid char(7) import('_EXCP_MSGID');
dcl-s returncode int(5);
```

```
                    Convert RPGLE to Free form (ACVTRPGFRE)

  Type choices, press Enter.

  Source file  . . . . . . . . . . . SRCFILE      > QRPGLESRC
    Library  . . . . . . . . . . . .                >   SAMPLEx
  Source member  . . . . . . . . . . SRCMBR       > HSR200
  Source type  . . . . . . . . . . . SRCTYPE        RPGLE
  Type of object . . . . . . . . . . OBJTYPE        *PGM
  Convert calculation specs. . . . . CVTCLCSPEC     *FREE
  Convert declaration specs. . . . . CVTDCLSPEC   > *YES
  Destination source file  . . . . . TOSRCFILE    > QRPGLESRC
    Library  . . . . . . . . . . . .                >   SAMPLEx
  Member to receive output . . . . . TOSRCMBR     > HSR200FULL
  Replace the existing member  . . . REPLACE        *NO
```

- L’outil permet également de faire du « rajeunissement » de code

V7R2

- Support du CCSID pour les zones alphanumériques
- Augmentation de la précision des zones de type TIMESTAMP
- Contrôle de la longueur retournée par la fonction %SUBDT
- Support de valeurs avec des décimales lors de l'ajout de secondes avec la fonction %SECONDS
- Possibilité de récupérer la différence entre 2 horodates en secondes avec positions décimales
- Nouveaux types de données supportés dans le PCML (dates, heures, horodates et entiers non signés sur 8 octets)

V7R2 – Support du CCSID

- Avertissement en cas de problèmes de conversion de CCSID
  - Nouveau mot-clé en spécification H : **CCSIDCVT**
    - ***LIST** : indique les pertes potentielles de données lors de la compilation
    - ***EXCP** : indique les pertes potentielles de données lors de l'exécution
  - Est disponible par PTF en 7.1 et 6.1

- Support du CCSID (uniquement en 7.2) :
  - Support du mot-clé CCSID pour les zones alphanumériques
  - Option pour indiquer que les sous-zones d'une DS externe récupèrent le CCSID des zones externes
  - Option pour indiquer que l'on souhaite ouvrir un fichier BD sans conversion du CCSID
  - Conversion implicite du CCSID lors de la concaténation en format libre (la conversion implicite était déjà supportée pour l'affectation, la comparaison et les paramètres)

RPG – Support du CCSID

- Exemple – Support de multiples CCSID

```
ctl-opt CCSID(*char:1147);
  dcl-s z1 char(1);
  dcl-s z2 char(50) CCSID(*UTF8);
  dcl-s z3 char(10) CCSID(*JOBRUN);
```

- Exemple – Récupération dans une DS du CCSID du fichier BD

```
A       R F1
A         Z1                2A     CCSID(297)
A         Z2               25A     CCSID(1208)

dcl-ds ds1       likerec(F1);
dcl-ds ds2       likerec(F1) CCSID(*EXACT);
```

- Pour que les données alphanumériques des fichiers base de données en soient pas converties dans le CCSID du job à l'ouverture des fichiers :
  - `ctl-opt OPENOPT(*NOCVTDATA)` ou `CCSID(*EXACT);`
  - `dcl-f fic1 DATA(*NOCVT);`

RPG – Autres nouveautés

- Augmentation de la précision des zones de type TIMESTAMP
  - 0 à 12 positions pour les microsecondes

```
dcl-s ts0 timestamp(0); // 2014-05-13-01.02.03
dcl-s ts3 timestamp(3); // 2014-05-13-01.02.03.789
dcl-s ts12 timestamp(12); // 2014-05-13-01.02.03.78912345678
D ts5             s               z 5
```

- Troisième paramètre dans la fonction %SUBDT pour contrôler la longueur de la zone extraite
  - `%char(%subdt(d:*years:4))` renvoie `'2014'`
  - `%char(%subdt(d:*years))` renvoie `'0000002014'`

- Support de valeurs avec des décimales lors de l'ajout de secondes avec la fonction %SECONDS
  - `t1 = t1 + %seconds(1,5);`

- Possibilité de récupérer la différence entre 2 horodates en secondes avec positions décimales

```
diff1 = %diff(z_fin : z_debut : *SECONDS : 6); // diff1 =
1922,483019
```

2. Appel de fonctions C et de méthodes Java

Appel de fonctions C

- IBM fournit tout un ensemble de fonctions codées en langage C, accessibles sous forme de procédures par tout langage ILE

- Ces procédures sont disponibles dans des programmes de service QC2* enregistrés dans le répertoire de liage QC2LE :
  - QC2UTIL1/2/3 : fonctions C standard et utilitaires
    - Fonctions mathématiques, gestion des dates et heures, conversion de chaînes de caractères, recherche et tri, etc.
  - QC2IFS : gestion de fichiers et répertoires IFS
    - Ouverture/fermeture, lecture, écriture, etc.
    - Remarque : les APIs système « Integrated File System APIs » dans la catégorie « Unix-Type APIs » permettent une gestion plus facile de l'IFS
  - etc.

- Pour les visualiser :
  - WRKBNDDIRE QSYS/QC2LE
  - DSPSRVPGM SRVPGM(…) DETAIL(*PROCEXP)

Appel de fonctions C – Documentation

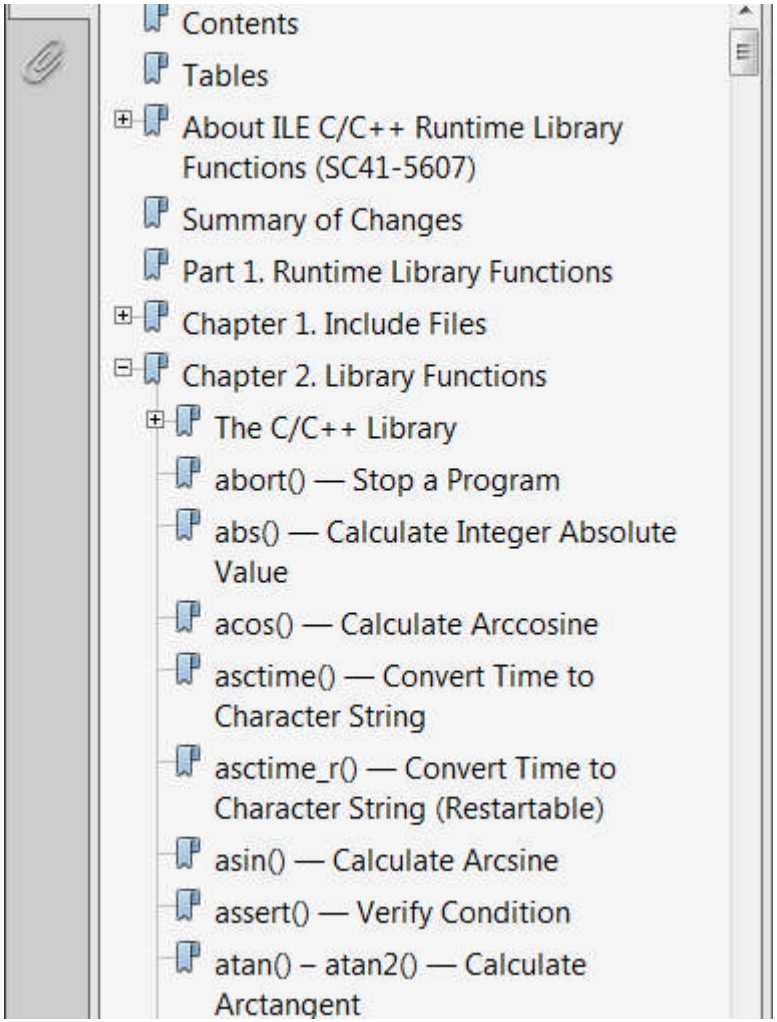
IBM

IBM i

ILE C/C++ Runtime Library Functions

7.1

SC41-5607-04



Appel de fonctions C – Exemples

- Génération d'un nombre aléatoire

rand(), rand_r() — Generate Random Number

Format

```
#include <stdlib.h>
int rand(void);
int rand_r(unsigned int *seed);
```

C_RAND.RPGLE

```
Ligne 7       Colonne 3     Replacer
      ....+....  Free-Form+++++++++++++++++++++++++++++++++++++++++++++
000100        ctl-opt dftactgrp(*no) bnddir('QC2LE');
000101
000102        dcl-pr random_nbr int(5) extproc('rand') end-pr;
000103
000104          dsply random_nbr;
000105          *inlr = *on;
```

OU

C_RAND2.RPGLE

```
Ligne 9       Colonne 1     Replacer
      ....+....  Free-Form+++++++++++++++++++++++++++++++++++++++++++++
000100      H dftactgrp(*no) bnddir('QC2LE')
000101
000102      D random_nbr      pr             5i 0 extproc('rand')
000103
000104       /free
000105         dsply random_nbr;
000106         *inlr = *on;
000107       /end-free
```

```
DSPLY                 16838
DSPLY                  5758
DSPLY                 10113
DSPLY                 17515
```

Appel de fonctions C – Exemples

- Appel d'une commande CL

system() — Execute a Command

Format

```
#include <stdlib.h>
int system(const char *string);
```

```
H dftactgrp(*no) bnddir('QC2LE')

D runcmd          pr             5i 0 extproc('system')
D  cmd                            *   value options(*string)

D errmsgid        s              7    import('_EXCP_MSGID')
D returncode      s              5i 0
```

C_SYSTEM.RPGLE

```
Ligne 22      Colonne 1      Replacer
      .....  Free-Form++++++++++++++++++++++++++++++++++++++++
000100       ctl-opt dftactgrp(*no) bnddir('QC2LE');
000101
000102       dcl-pr runcmd int(5) extproc('system');
000103         cmd pointer value options(*string);
000105       end-pr;
000106
000107       dcl-s errmsgid char(7) import('_EXCP_MSGID');
000108       dcl-s returncode int(5);
000109
000110         returncode = runcmd('DLTF PB/FIC1');
000111         if returncode = 0;
000112           dsply 'Fichier supprimé';
000113         else;
000114           select;
000115           when errmsgid = 'CPF2105';
000116             dsply 'Fichier non trouvé';
000117            when errmsgid = 'CPF3202';
000118             dsply 'Fichier en cours d''utilisation';
000119           endsl;
000120         endif;
000121         *inlr = *on;
```

OU

Appel de méthodes Java

- Le support de Java est fourni avec l'IBM i (produit JV1)
- La JVM permet l'exécution de code Java
- Le RPG peut appeler du code Java en utilisant les APIs JNI (Java Native Interface)
  - Par l'appel d'une procédure prototypée
- Pour prototyper l'appel d'une méthode Java, il faut connaître
  - Le nom complet de la classe Java
  - Le nom du constructeur
  - Le nom de la méthode
  - Le type des paramètres
  - Le type de la valeur de retour
  - Les équivalences entre les types Java et RPG
- L'assistant « Appel de méthode Java » de RDi facilite la tâche en créant le code nécessaire

Appel de méthodes Java

- Déclaration d'une classe Java
  - `D var1  s  O  CLASS(*JAVA:'nom_de_la_classe')`
  - Peut être définie en tant que variable, paramètre d'une procédure ou valeur de retour d'une procédure
- Déclaration d'une méthode Java
  - `D m1  pr  xx  EXTPROC(*JAVA:'nom_de_la_classe':`
    `'nom_méthode' ou *CONSTRUCTOR)`
  - xx étant le type de données retourné
- Le CLASSPATH ainsi que la version de Java peuvent être définies par des variables d'environnement (WRKENVVAR)
- La JVM sera démarrée à l'appel du programme

Appel de méthodes Java – Exemple

```
000001      H dftactgrp(*no)
000104      D s1              s                 o   class(*java:'java.lang.String')
000106      D s2              s                 o   class(*java:'java.lang.String')
000107      D
000127      D new_String      pr                o   extproc(*java:'java.lang.String':
000130      D                                       *constructor)
000131      D                                       class(*java:'java.lang.String')
000132      D   var1                          50a   options(*varsize) const
000133      D
000134      D cmpIgnoreCase   pr                n   extproc(*java:'java.lang.String':
000135      D                                       'equalsIgnoreCase')
000136      D  anotherstring                    o   class(*java:'java.lang.String')
000140      D                                       const
000156       /free
000157         s1 = new_String('Evolutions du RPG');
000158         s2 = new_String('EVOLUTIONS du RPG');
000159
000160         if (cmpIgnoreCase(s1:s2));
000161           dsply 'Les chaînes s1 et s2 sont identiques';
000162         else;
000163           dsply 'Les chaînes s1 et s2 sont différentes';
000164         endif;
000165
000166        *inlr = *on;
```

Class String

java.lang.Object
java.lang.String

String

```
public String(byte[] bytes)
```

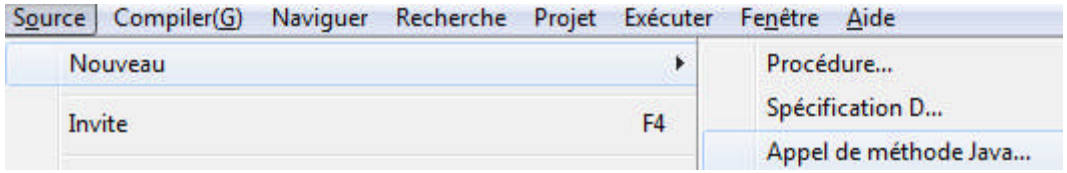
equalsIgnoreCase

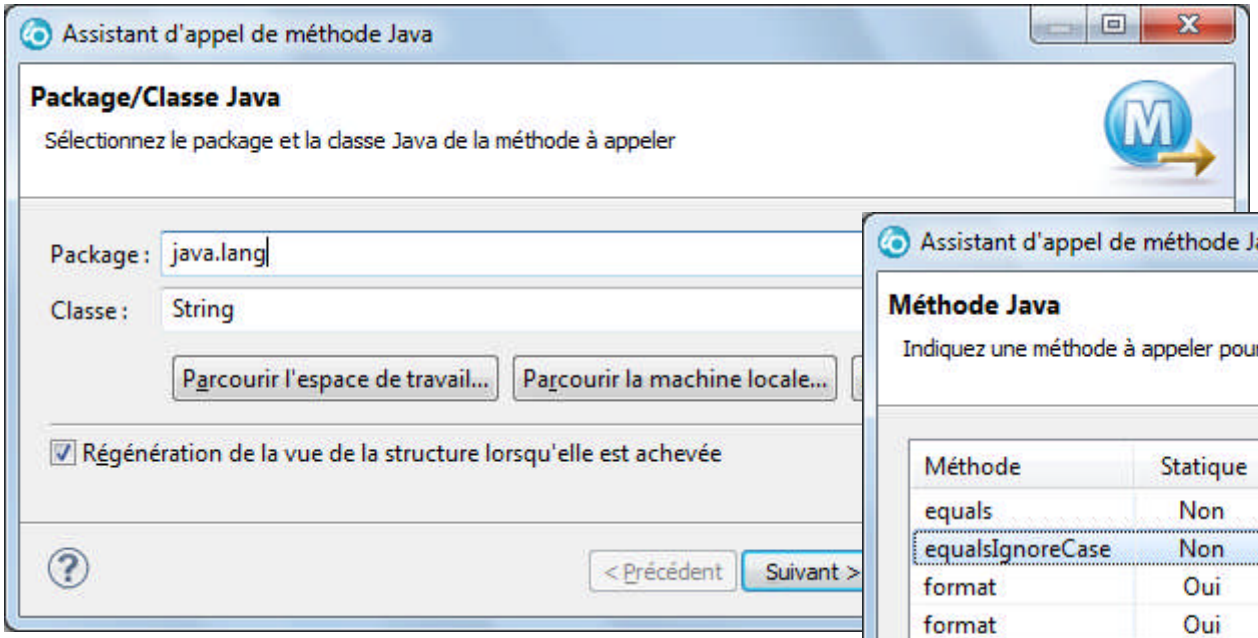
```
public boolean equalsIgnoreCase(String anotherString)
```

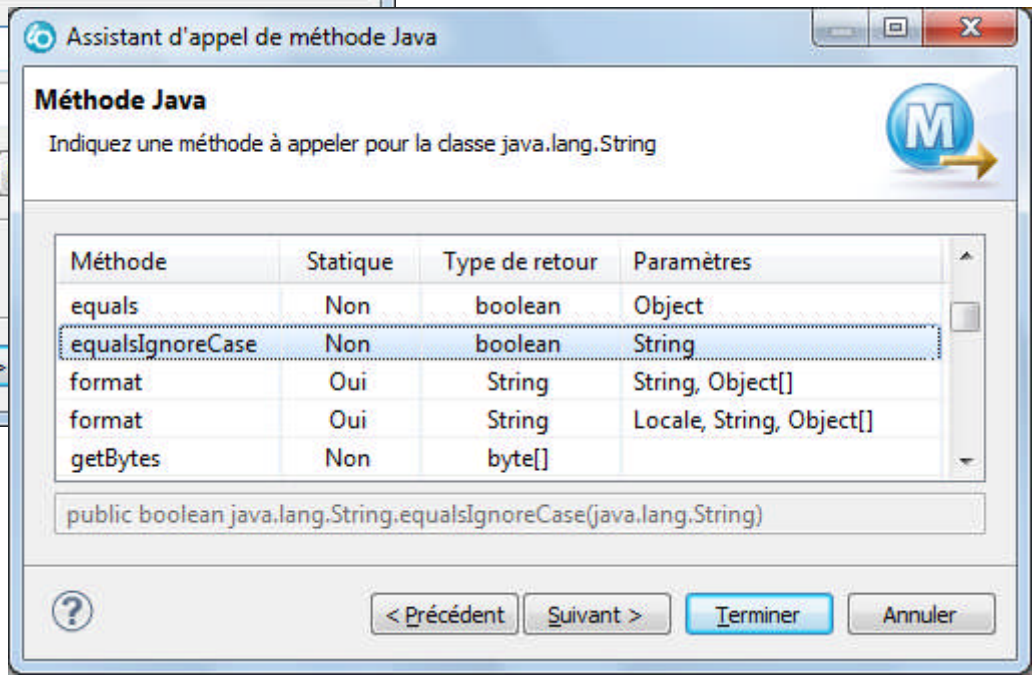
```
s1 = new_String('Evolutions du RPG');
s2 = new_String('EVOLUTIONS duu RPG');
```

```
La machine JVM est IBM Technology for Java. PID (1650)
DSPLY  Les chaînes s1 et s2 sont identiques
DSPLY  Les chaînes s1 et s2 sont différentes
```

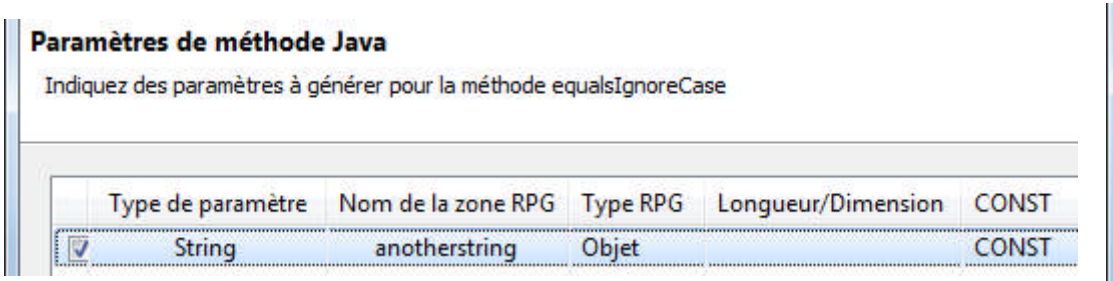
Appel de méthodes Java – L'assistant de RDi – 1/2

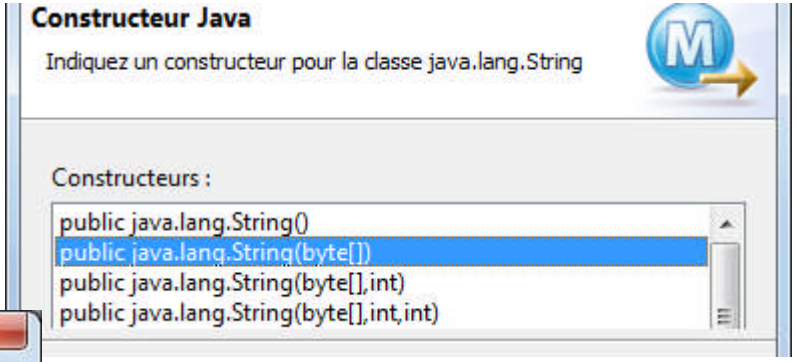


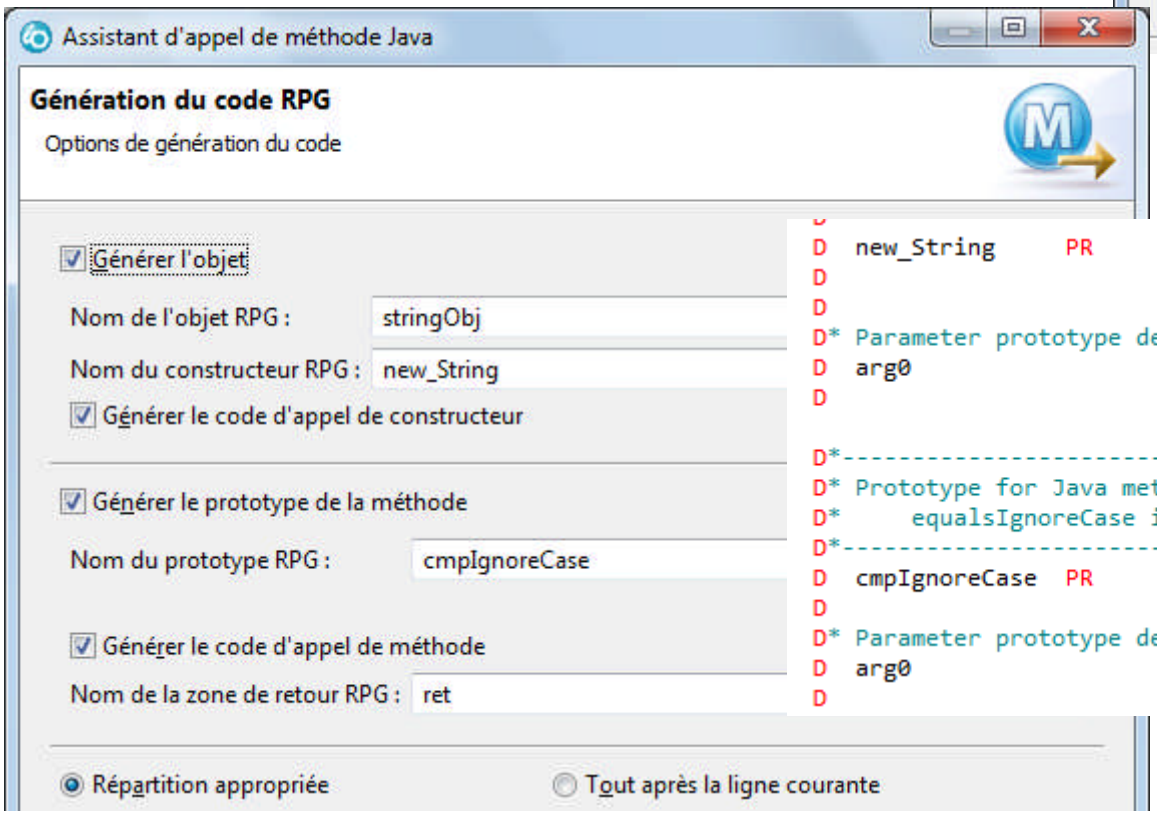




Appel de méthodes Java – L'assistant de RDi – 2/2








```
D  new_String     PR                  O   EXTPROC(*JAVA : 'java.lang.String'
D                                         : *CONSTRUCTOR)
D                                        CLASS(*JAVA : 'java.lang.String' )
D* Parameter prototype declaration for Java type: byte[]
D  arg0                           50A   VARYING
D                                        CONST

D*---------------------------------------------------------
D* Prototype for Java method:
D*     equalsIgnoreCase in class String in package java.lang
D*---------------------------------------------------------
D  cmpIgnoreCase  PR                  N   EXTPROC(*JAVA : 'java.lang.String'
D                                         : 'equalsIgnoreCase' )
D* Parameter prototype declaration for Java type: String
D  arg0                               O   CLASS(*JAVA : 'java.lang.String' )
D                                        CONST
```

. . . / . . .

3. Accès à des bases de données externes

Le projet JDBCR4

- Les drivers JDBC de type 4 des bases de données MySQL, SQL Server, Oracle, DB2 LUW, etc. sont développés en Java
- Ces drivers sont fournis sous forme de fichiers jar (xxx : numéro de version) :
  - MySQL : mysql-connector-java-xxx-bin.jar
  - SQL Server : sqljdbc.jar ou jtds-xxx.jar
  - Oracle : ojdbcxxx.jar
  - IBM DB2 LUW : db2jcc.jar
  - A placer dans l'IFS – Modifier le CLASSPATH (WRKENVVAR)
- Il est possible d'exécuter des méthodes Java à partir de RPG (voir chapitre précédent)
- Le projet JDBCR4 de Scott Klement permet d'accéder facilement à ces bases de données externes
  - Par la fourniture d'un programme de service à créer
    - http://systeminetwork.com/files/RpgAndJdbc.zip

Le projet JDBCR4

- Après création du programme de service et ajout de celui-ci dans un répertoire de liage il faut débuter les programmes RPG par :

```
H dftactgrp(*no) bnddir('nom_du répertoire_de_liage')
 /copy jdbc_h
```

- Vous pouvez alors utiliser les procédures RPG suivantes :
  - JDBC_Connect() : connexion à la base de données
  - JDBC_ExecUpd() : exécution d'un ordre autre que SELECT
  - JDBC_ExecQry() : exécution d'un ordre SELECT
    - Renvoie un ResultSet qui pourra être lu par les procédures suivantes :
      - JDBC_nextRow() : lit la ligne suivante
      - JDBC_getCol() : retourne la valeur d'une colonne par son numéro
      - JDBC_getColByName() : retourne la valeur d'une colonne par son nom
      - JDBC_freeResult() : ferme le ResultSet
  - JDBC_PrepStmt(), JDBC_ExecPrepUpd(), JDBC_ExecPrepQry() et JDBC_FreePrepStmt pour les instructions préparées et JDBC_setXXX pour alimenter les marqueurs de ces instructions
- Il en existe d'autres (accès aux métadonnées, appel de procédures stockées, commit/rollback…)

Le projet JDBCR4 – Exemple

```
000001     H dftactgrp(*no) bnddir('JDBCR4')
000002      /copy jdbc_h
000003     D user             s              30a   inz('root')
000004     D password         s              30a   inz('xxxxxxx')
000005     D conn             s                    like(Connection)
000006     D films            s                    like(ResultSet)
000007     D dsfilm         e ds                   extname(dvd)
000156      /free
000157       conn = JDBC_Connect('com.mysql.jdbc.Driver':
000158                           'jdbc:mysql://localhost/db1':
000159                           %trim(user):
000160                           %trim(password));
000161
000162       films = JDBC_ExecQry(conn:'select titre, annee, duree from pb.dvd');
000163
000164       dow JDBC_nextRow(films);
000165         titre = JDBC_getCol(films:1);
000166         annee = %int(JDBC_getCol(films:2));
000167         duree = %int(JDBC_getCol(films:3));
000171       enddo;
000172
000173       JDBC_freeResult(films);
000174       JDBC_close(conn);
000175       *inlr = *on;
```

4. Lecture de flux XML

RPG et XML

- La lecture d'un flux XML est supportée en RPG natif
  - IBM i 7.1, 6.1 et 5.4
- Code-opération XML-INTO
  - Permet d'alimenter une DS depuis un flux XML
    - Le flux XML peut provenir d'une variable ou d'un fichier dans l'IFS
    - La DS peut être simple ou à n dimensions
- Code-opération XML-SAX
  - Permet de parcourir un flux XML de façon événementielle
    - Utilisation d'une procédure dite « handler » qui sera appelée à chaque événement XML

Le code-opération XML-INTO

- Syntaxe :
  - `XML-INTO nom_DS %XML(flux_XML : 'options')`
  - Flux_XML :
    - Une variable (charactère ou UCS-2) contenant du XML
    - Un fichier IFS (donné sous forme de constante ou de variable)
  - Options : comportement du parser XML
    - trim=all/none
      - Enlève ou non les blancs des éléments avant leur affectation auprès des variables
    - allowmissing=no/yes
      - Peut-il y avoir des sous-zones de DS n'ayant pas d'équivalent en tant qu'élément XML ?
    - allowextra=no/yes
      - Peut-il y avoir des éléments XML n'ayant pas d'équivalent dans la DS ?
    - etc.

Le code-opération XML-INTO – Exemple

XML_INTO1.RPGLE

```
Ligne 25      Colonne 62     Replacer
 .....  Free-Form+++++++++++++++++++++++++++++++++++++++++++++++++++++
     D i               s              5i 0
     D j               s                   like(i)

     D films           ds                  qualified
     D  film                               likeds(film_t) dim(10)
     D  nbfilm                        5i 0

     D film_t          ds                  qualified template
     D  titre                        40a
     D  annee                         4a
     D  acteur                       10a   dim(5)
     D  nbacteur                      5i 0

      /free
       xml-into films %xml('/pb/xml/liste_films.xml':
                           'doc=file path=films +
                            countprefix=nb');
       for i=1 to films.nbfilm;
         dsply films.film(i).titre;
         dsply films.film(i).annee;
         for j=1 to films.film(i).nbacteur;
           dsply films.film(i).acteur(j);
         endfor;
       endfor;
       *inlr = *on;
```

liste_films.xml

```xml
<?xml version="1.0" encoding="UTF-8" ?>
<films>
    <film titre="MATRIX" annee="1999">
        <acteur>REEVES</acteur>
        <acteur>MOSS</acteur>
    </film>
    <film titre="SPIDERMAN" annee="2002">
        <acteur>MAGUIRE</acteur>
        <acteur>DUNST</acteur>
        <acteur>DAFOE</acteur>
    </film>
</films>
```

```
DSPLY  MATRIX
DSPLY  1999
DSPLY  REEVES
DSPLY  MOSS
DSPLY  SPIDERMAN
DSPLY  2002
DSPLY  MAGUIRE
DSPLY  DUNST
DSPLY  DAFOE
```

Utilisation d'un handler – XML-INTO et XML-SAX

- Avec XML-INTO :
  - `XML-INTO %HANDLER(nom_proc) %XML(flux_XML:'options')`
  - Construit un tableau de N éléments XML et le passe à la procédure (nom_proc) qui sera appelée autant de fois que nécessaire. N occurences seront transmises à chaque appel sauf éventuellement lors du dernier appel. Permet une lecture séquentielle et non complète du flux XML

- Avec XML-SAX :
  - `XML-SAX %HANDLER(nom_proc) %XML(flux_XML:'options')`
  - La procédure sera appelée pour chaque événement XML
  - Par exemple 10 fois dans l'exemple suivant

```
<film titre="MATRIX"><acteur>REEVES</acteur></film>
```

  - A utiliser si l'on souhaite distinguer les attributs des éléments, rechercher combien il existe d'occurrences de tel élément, etc.

Utilisation d'un handler avec XML-INTO – Exemple

```
H dftactgrp(*no)
D film_t          ds                  qualified template
D  titre                        40a
D  annee                         4a
D  acteur                       10a   dim(5)

D commarea        s                  1a
D i               s                  5i 0
 /free
   xml-into %handler(xml_handler:commarea)
            %xml('/pb/xml/liste_films.xml':
                 'doc=file path=films/film +
                 allowmissing=yes');
   *inlr = *on;
 /end-free
```

```
P xml_handler     b
D                 pi             10i 0
D  commarea                       1a
D  films                                dim(1) likeds(film_t) const
D  nbfilms                        10i 0 value

D numpassage      s                2  0 static
D msg             s               50
 /free
   numpassage = numpassage + 1;
   msg = 'Passage n° ' + %char(numpassage);
   dsply msg;
   for i=1 to nbfilms;
     dsply films(i).titre;
     dsply films(i).annee;
   endfor;
   return 0;
 /end-free
P  xml handler    e
```

```
DSPLY  Passage n° 1
DSPLY  MATRIX
DSPLY  1999
DSPLY  Passage n° 2
DSPLY  SPIDERMAN
DSPLY  2002
```

```
DSPLY  Passage n° 1
DSPLY  MATRIX
DSPLY  1999
DSPLY  SPIDERMAN
DSPLY  2002
```

```
D  films                          dim(10) likeds(film_t) const
```

5. Invocation de Services Web

Les Services Web – Rappel

- Les Services Web sont des programmes métier invocables à partir du Web :
  - Pas d'interface utilisateur
  - Utilise les protocoles standards du Web
  - Sont indépendants de la plateforme et du langage
  - Permet l'interopérabilité des applications
  - Sont de type SOAP ou REST

- Un Service Web SOAP est décrit par un fichier WSDL
  - Web Service Definition Language
  - Définit les opérations (les procédures, les fonctions) que l'on peut invoquer, les types de données transmises, les protocoles utilisés ainsi que la localisation du service (URI / URL)
  - Nécessaire et suffisant pour invoquer (consommer) un Service Web

L'Integrated Web Services (IWS) Client for IBM i

- Intégré à l'IBM i (7.1, 6.1 ou 5.4) – Groupe de PTFs HTTP
- Commandes shell qui permettent de générer, à partir d'un fichier WSDL, des procédures RPG (ou C) qui permettront d'invoquer le Service Web
  - L'outil génère également le programme CL pour construire les objets ILE nécessaires (modules et programme de service) et exécute ce programme
- Ces procédures pourront ensuite être appelées à partir de programmes ILE RPG /COBOL
- Quelques restrictions (SOAP 1.1 uniquement…)

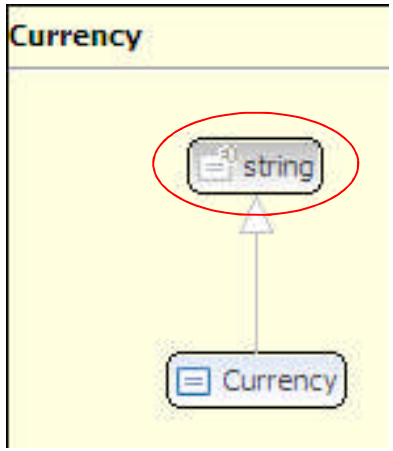
Invocation d'un Service Web avec IWS Client – Exemple

Currency Convertor

Description

Get conversion rate from one currency to another currency

CurrencyConvertorSoap		
ConversionRate		
input	FromCurrency	Currency
	ToCurrency	Currency
output	ConversionRateResult	double



ConversionRate

FromCurrency string

EUR

ToCurrency string

USD

Statut

Corps

ConversionRateResponse

ConversionRateResult (double) : 1.2895

Invocation d'un Service Web avec IWS Client – Exemple

- Génération des procédures RPG et du programme de service par l'exécution du shell **wsdl2rpg.sh**

```
> wsdl2rpg.sh -o/pb/AS488FR/CurrencyConvertor
                -s/QSYS.LIB/SOADEMO.LIB/CYCSRV.SRVPGM
                   -pCYC /pb/AS488FR/CurrencyConvertor/CurrencyConvertor.wsdl

  Code generation completed. Generated files in directory
  '/pb/AS488FR/CurrencyConvertor'.

  Attempting to create service program...

  Service program created. Service program is
  '/QSYS.LIB/SOADEMO.LIB/CYCSRV.SRVPGM'.
  $
```

Invocation d'un Service Web avec IWS Client – Exemple

- Le code des procédures RPG générées
  - Création de la « connexion »

```
D stub_create_CurrencyConvertorSoap...
D                 PR                  1N   extproc('stub_create_CurrencyConve+
D                                          rtorSoap@')
D this                                     likeds(This_t)
```

  - Appel de l'opération ConversionRate

```
D stub_op_ConversionRate...
D                 PR                  1N   extproc('ConversionRate@')
D this                                     likeds(This_t)
D Value0                                   likeds(xsd_string)
D Value1                                   likeds(xsd_string)
D out                                      likeds(xsd_double)
```

  - Suppression de la « connexion »

```
D stub_destroy_CurrencyConvertorSoap...
D                 PR                  1N   extproc('stub_destroy_CurrencyConv+
D                                          ertorSoap@')
D this                                     likeds(This_t)
```

Invocation d'un Service Web avec IWS Client – Exemple

- Le programme final

```
H bnddir('CYCBNDDIR') dftactgrp(*no) actgrp(*caller)
 /copy /AS488FR/CurrencyConvertor/CurrencyConvertorSoap.rpgleinc
D wsStub          ds                  likeds(this_t)
D fromDevise      ds                  likeds(xsd_string)
D toDevise        ds                  likeds(xsd_string)
D taux            ds                  likeds(xsd_double)
D msg             s             52a
D devise1         s              3a
D devise2         s              3a

C     *entry        plist
C                   parm                    devise1
C                   parm                    devise2
 /free
   wsStub.endpoint = *blanks;
   fromDevise.value = %trim(devise1);
   toDevise.value   = %trim(devise2);

   if stub_create_CurrencyConvertorSoap(wsStub)= *on;
     if stub_op_ConversionRate(wsStub:fromDevise:toDevise:taux) = *on;
      msg = 'Le taux de conversion de ' + devise1 + ' vers '
          + devise2 + ' est de : ' + %char(%dec(taux.value:10:4));
     endif;
    stub_destroy_CurrencyConvertorSoap(wsStub);
   endif;
   if wsStub.excOccurred = *on;
    msg = wsStub.excString;
   endif;
   dsply msg;
   *inlr = *on;
```

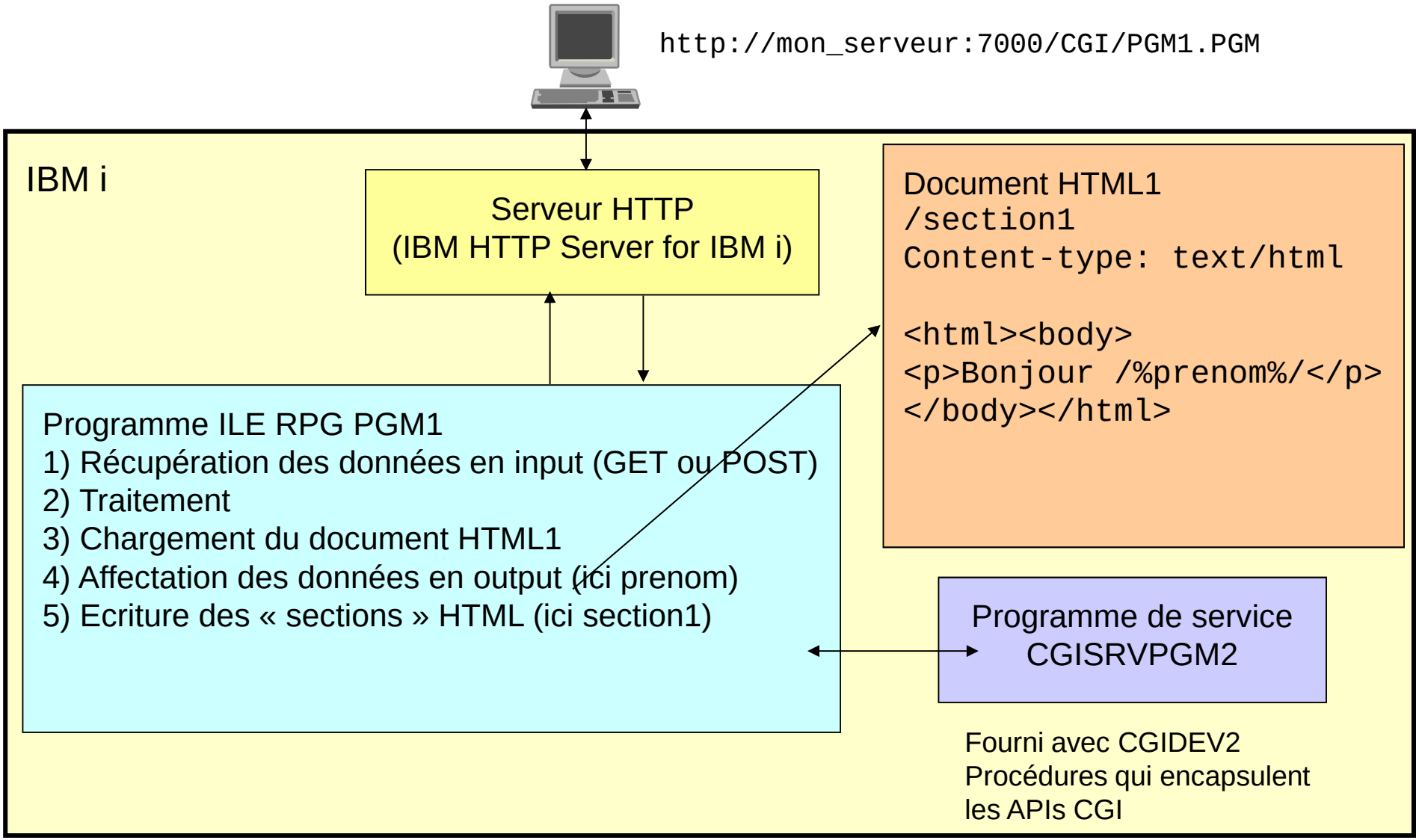
```
CALL PGM(CYCSOAP) PARM(EUR USD)
DSPLY  Le taux de conversion de EUR vers USD est de :  1.293
```

6. Développement Web avec CGI

Qu'est-ce que CGI ?

- CGI (Common Gateway Interface) permet de développer des applications Web en intégrant des APIs de gestion du HTML dans des programmes écrits en ILE RPG/COBOL
- CGIDEV2
  - Bibliothèque de fonctions permettant de faciliter le développement d'applications Web en CGI et ILE RPG (HTML externe, validation des données, gestion des messages, etc.)
  - Gratuit - Développé et maintenu par IBM Lab Services
  - Téléchargement, tutoriaux et exemples sur le site EASY400.NET
- Pour pouvoir exécuter des applications CGI seul le serveur HTTP de l'IBM i (57xx-DG1 – IBM HTTP Server for IBM i, fourni en standard avec l'IBM i) est nécessaire
- Des frameworks open-source basés sur CGIDEV2 permettent de faciliter le développement

CGI – Architecture (avec CGIDEV2)



CGI – Exemple – 1/4 – Formulaire de saisie

CGI_Formulaire.html

```
1  <HTML>
2  <HEAD>
3  <TITLE>Comparaison des solutions CGI, Java, PHP et EGL sous IBM i</TITLE>
4  </HEAD>
5  <BODY>
6  <H1>Gestion des DVD vidéo - CGI</H1>
7  <FORM METHOD=POST ACTION="/CGIDEV2P/CGI_CONTRL.PGM">
8  <H3>Saisissez un code DVD
9  <INPUT TYPE="text" NAME="code_dvd" SIZE="5"></H3>
10 <INPUT TYPE="submit" VALUE="OK">
11 </FORM>
12 </BODY>
13 </HTML>
```

CGI – Exemple – 2/4 – Contrôleur

CGI_CONTRL.RPGLE

```
Ligne 20      Colonne 1      Replacer
      ....+....1....+....2....+....3....+....4....+....5....+....6..
000100     H dftactgrp(*NO) bnddir('CGIDEV2':'CGI_MODELE')
000200
000300     D/copy prototypeb
000400     D/copy usec
000401     D qrystr          s           32767    varying
000402     D dvdCode         s              10i 0
000403     D dvdTitre        s              60
000500
000501     D getDvdTitre     pr             60
000502     D                                10i 0
000503
000600     C/free
000601         ZhbGetInput(qrystr:qusec);
000602         dvdCode = %int(ZhbGetVar('code_dvd'));
000604         dvdTitre = getDvdTitre(dvdCode);
000700         gethtmlIFS('/CgidevExtHtml/CGI_Vue.html':'<IBMi>');
000800         updHTMLvar('code_dvd':%char(dvdCode));
000801         updHTMLvar('titre_film':dvdTitre);
000900         wrtsection('top *fini');
001000         *inlr = *on;
```

CGI – Exemple 3/4 – Modèle

CGI_MODELE.RPGLE

```
Ligne 12      Colonne 30    Replacer
      ...PName++++++++++..B.....O.............Keywords++++++++++
000501     D getDvdTitre     pr                  60
000502     D                                     10i 0
000503
000504     P getDvdTitre     b                   export
000505     D                 pi                  60
000506     D  code                               10i 0
000600      /free
000601           return 'Le titre du film numéro ' + %char(code) +
000602                  ' a été défini en dur';
001000           *inlr = *on;
001001      /end-free
001100     P                 e
```

CGI – Exemple – 4/4 – Vue

CGI_Vue.html

CGI_Vue.ht...

```
<IBMi>top
Content-type: text/html

<HTML>
<TITLE>Comparaison des solutions CGI, Java, PHP et EGL sous IBM i</TITLE>
<BODY>
<H1>Gestion des DVD vidéo - CGI</H1>
<H3>Code Dvd : /%code_dvd%/ </H3>
<H3>Titre du film : /%titre_film%/ </H3>
</BODY>
</HTML>
```

Comparaison des solutions CGI, Java...

Gestion des DVD vidéo - CGI

Code Dvd : 50

Titre du film : Le titre du film numéro 50 a été défini en dur

7. Ouverture avec RPG Open Access

Qu'est-ce que RPG Open Access ?

- RPG Open Access (RPG OA) permet aux programmes RPG d'accéder facilement aux technologies et interfaces modernes :
  - Une ligne de code à ajouter dans les programmes
  - Un accès, en utilisant les codes-opération que vous connaissez bien (READ, CHAIN, UPDATE…), à des ressources telles que :
    - Interfaces autres que le 5250 (client Web, client mobile…)
    - Fichiers dans l'IFS
    - Services Web
    - DB2 for i mais en SQL
    - Bases de données externes…

- RPG OA est fourni avec les compilateurs ILE – PTFs nécessaires :
  - A la compilation
    - V7R1 : SI45902 et SI45903
    - V6R1 : SI45904

  A l'exécution :
  - V7R1 : SI45905
  - V6R1 : SI45906

RPG OA – Principes

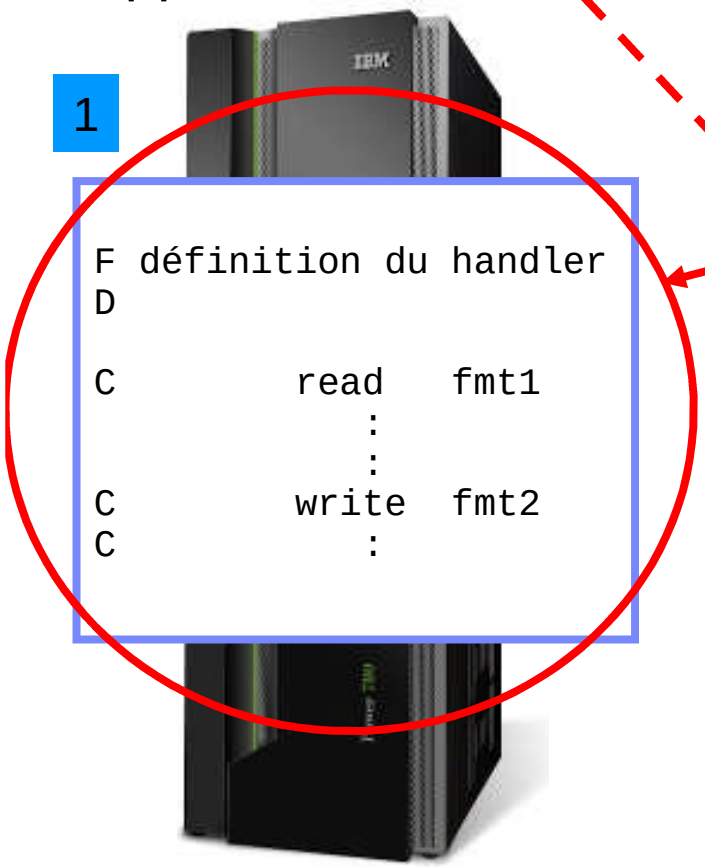
- Une application Open Access a trois composantes :
  - 1) Un programme RPG qui utilise les codes-opération d'E/S classiques du RPG (READ, WRITE, CHAIN…) sur des fichiers déclarés en « open-access » (mot-clé "handler")

```
Fclients   if   e                 disk    handler('PB_HDLR1(test1)')
```

  - 2) Un programme « handler qui gèrera les opération d'E/S sur les fichiers déclarés en « open access » au point 1
  - 3) Un programme d'accès aux ressources (fichiers IFS, interface Web, bases de données externes…)

RPG OA – Principes

Applications RPG



Data

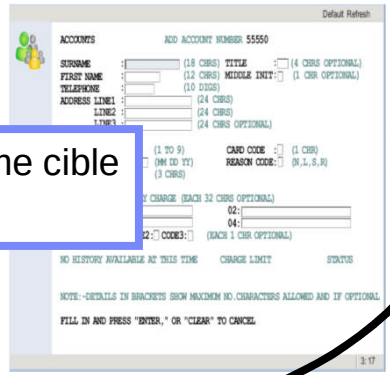
2

Handler

Ressource

Programme cible

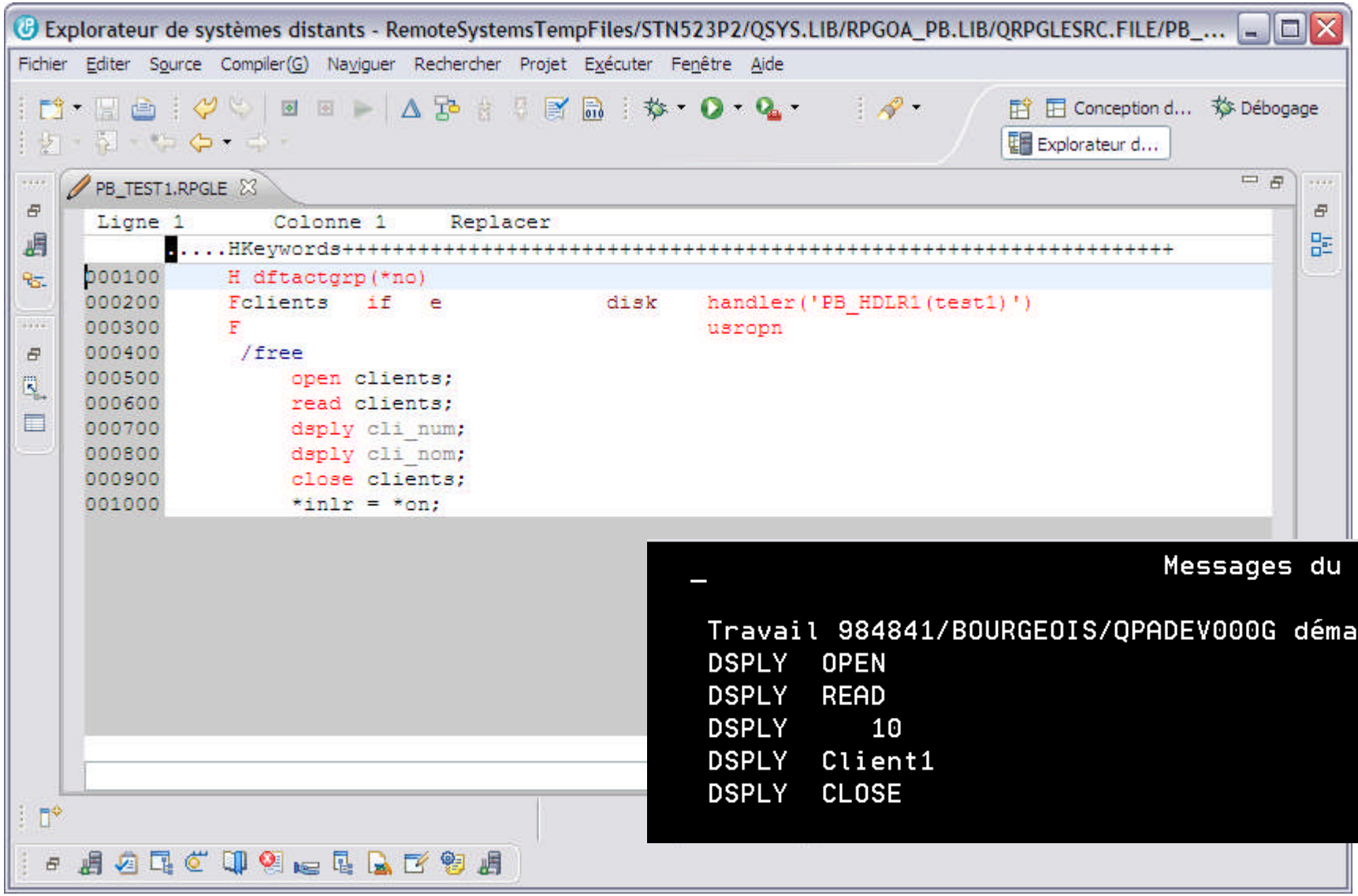
3



Les handlers et programmes cible sont développés par des éditeurs (looksoftware, Profound Logic, Rocket Seagull, PKS, RJS, VAI) ou par le client. IBM ne fournit pas de handlers

Le développeur continue à développer en RPG
Il fait appel, de façon transparente, aux procédures du handler

RPG OA – Exemple – Programme RPG

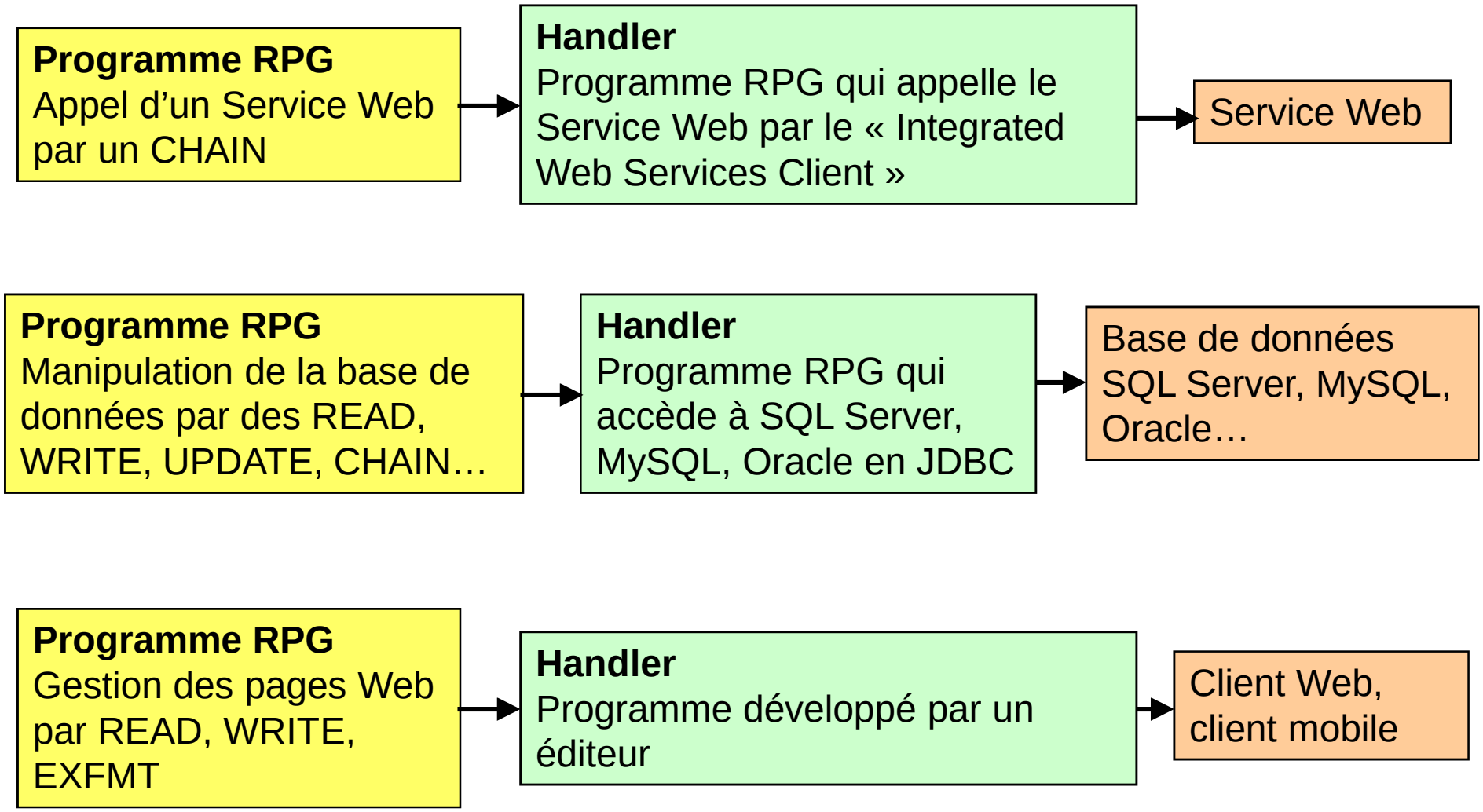


RPG OA – Exemple – Programme handler

PB_HDLR1.RPGLE

```
Ligne 1       Colonne 1     Replacer
      ....HKeywords+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
000100     H nomain
000200      /copy qoar/qrpglesrc,qrnopenacc
000300
000400     D test1           pr                  extproc('test1')
000500     D   ds_oa                             likeds(QrnOpenAccess_T)
000600
000700     P test1           b                   export
000800     D                 pi
000900     D   ds_oa                             likeds(QrnOpenAccess_T)
001000     D
001100     D client_ds     e ds                  extname('CLIENTS') qualified
001200     D                                     based(p_client_ds)
001300      /free
001400         if ds_oa.rpgOperation = QrnOperation_OPEN;
001500          dsply 'OPEN';
001600
001700         elseif ds_oa.rpgOperation = QrnOperation_CLOSE;
001800          dsply 'CLOSE';
001900
002000         elseif ds_oa.rpgOperation = QrnOperation_READ;
002100          dsply 'READ';
002200          p_client_ds = ds_oa.inputBuffer;
002300          client_ds.cli_num = 10;
002400          client_ds.cli_nom = 'Client1';
002500
002600         else;
002700            ds_oa.rpgStatus = 1299;
002800         endif;
002900      /end-free
003000     P test1           e
```

RPG OA – Exemples d'utilisation



RPGOA – Exemple de handler

developerWorks > Technical topics > IBM i > Technical library >

Decoupling RPG database IO using Rational Open Access: RPG Edition

A fresh start for RPG, ILE, and SQL

Moving from a DDS to SQL database on DB2 for i can be accomplished without changing a single line of program code or recompiling a program. In this article I will describe how to use Rational Open Access: RPG Edition to take advantage of advanced data centric programming techniques only available via SQL programming.

14 Comments

Daniel R. Cruikshank (dcrank@us.ibm.com), Senior Consultant, IBM

06 September 2011

Also available in Chinese Spanish

+ Table of contents

IBM i customers, worldwide, are now using SQL to define and access data. With each new release, IBM continues to provide a wealth of new function and capability that can only be leveraged with SQL. A list of these enhancements are described in the DB2 Modernization white paper.

RPG Open Access Sample code

The RPG Open Access sample code is now online at: http://ibm.com/systems/i/db2/db2code.html

These new SQL based applications are taking advantage of modern data centric development concepts that stress the importance of pushing more of the business rules and logic into the database.

RPG – Pour en savoir plus

- Evénements Modernisation IBM i
  - 8 et 9 avril 2013
    - S7 - RPG et IFS
    - S12 - Profitez des fonctionnalités de JAVA dans vos programmes RPG
  - 5 et 6 avril 2012
    - S5 - RPG – Transformez vos sous-routines en procédures ILE
    - S10 - RPG – Comment accéder à d'autres bases de données ?
    - S13 - RPG – Les 10 choses qu'un développeur RPG doit connaître
    - S15 - XML sous IBM i - Le point sur les solutions
    - S23 - Des Services Web dans vos programmes RPG et COBOL
  - 16 et 17 mai 2011
    - S10 - RPG, nouveautés 6.1 et 7.1
    - S13 - RPG – Comment utiliser au mieux les types de données récents ?