



IBM Power Systems – IBM i – IBM Rational

Introduction à Rational Open Access: RPG Edition

(RPG Open Access / RPG OA)

Philippe Bourgeois – IBM France

pbourgeois@fr.ibm.com

Plan de la présentation

- Quelques mots sur les offres Rational for Power Systems 2010

- Introduction à RPG Open Access
 - Qu'est-ce que Rational Open Access: RPG Edition ?
 - Principes de fonctionnement
 - Comment est construit un « handler » ?
 - Exemples simples d'applications Open Access

- Quelques mots sur les handlers d'IBM Lab Services

- Annexe - Quelques mots sur le handler de la société looksoftware

Quelques mots sur les offres Rational for Power Systems 2010

- **RD Power - Rational Developer for Power Systems**
 - Développement d'applications pour IBM Power Systems
 - RPG and COBOL Development Tools for i
 - COBOL Development Tools for AIX
 - C/C++ Development Tools for AIX
 - C/C++ Development Tools for Linux on Power

- **RTC Power – Rational Team Concert for Power Systems**
 - Contrôle des sources / versionning, gestion des demandes de changement, travail en équipe, gouvernance des développements

- **Rational Open Access: RPG Edition**
 - Ouverture du langage RPG

Rational Open Access: RPG Edition

Introduction

Qu'est-ce que RPG Open Access ?

- RPG Open Access fournit aux développeurs RPG le moyen d'utiliser le modèle simple et bien connu d'E/S du RPG (codes-opération READ, WRITE, CHAIN, etc.) pour accéder à des ressources et des unités qui ne sont pas directement supportées en RPG :
 - Fichiers XML, CSV, TXT, etc.
 - Services Web
 - Bases de données externes
 - Fichiers base de données cryptés
 - Navigateurs (browsers)
 - Unités mobiles
 - ...

RPG Open Access – Principe de fonctionnement

- 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 un fichier déclaré en « open-access »
 - Pour déclarer un fichier en « open access », on indiquera le mot-clé *handler* au niveau de ce fichier avec comme attribut le nom d'un programme ou d'une procédure ILE, que l'on appellera le « handler »
 - La présence de ce mot-clé *handler* permettra d'indiquer que lorsqu'une opération d'E/S sera réalisée sur ce fichier, ce ne sont pas les routines de l'OS qui seront appelées mais le programme handler
 - 2) Un programme « handler »
 - Qui sera appelé à chaque opération d'E/S sur le fichier déclaré en « open access » au point 1
 - Récupérera un buffer d'informations venant du programme RPG
 - Puis, à partir de ce buffer d'informations, communiquera avec une ressource, une unité ou un programme cible de « rendu »

RPG Open Access – Principe de fonctionnement

- 3) Une ressource, unité ou un programme cible de rendu
 - Le handler peut communiquer directement avec la ressource (par exemple un fichier dans l'IFS)
 - Ou bien faire appel à un programme cible de rendu (rendering program) qui sera spécifique à l'unité (Web, unité mobile...)

- Le produit Rational Open Access: RPG Edition permet de faire le lien entre les composantes 1 et 2

- Il est nécessaire à l'exécution de l'application

RPG Open Access - Principes

Applications RPG

1

```

F définition du handler
D
C      read   fmt1
  :
  :
C      write  fmt2
C
    
```

2

Handler

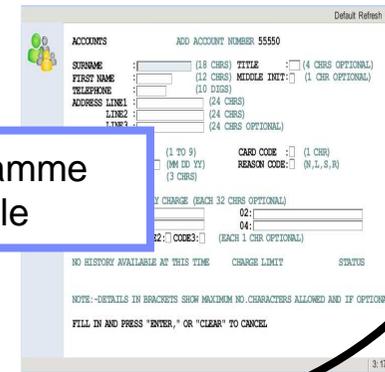
Data

Exécution du programme RPG avec IBM Rational Open Access: RPG Edition

Ressource

Programme cible

3

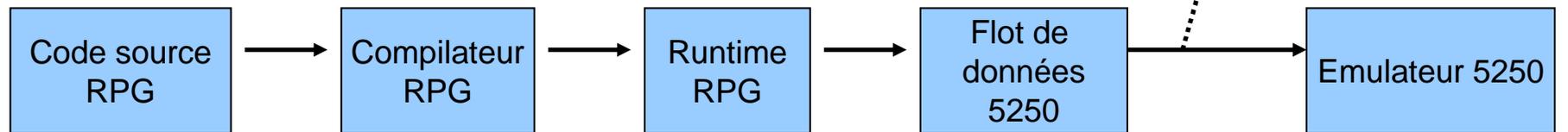


Les handlers et programmes cible sont développés par des éditeurs, des structures de services ou par le client. **RPG Open Access ne fournit pas de handlers**

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

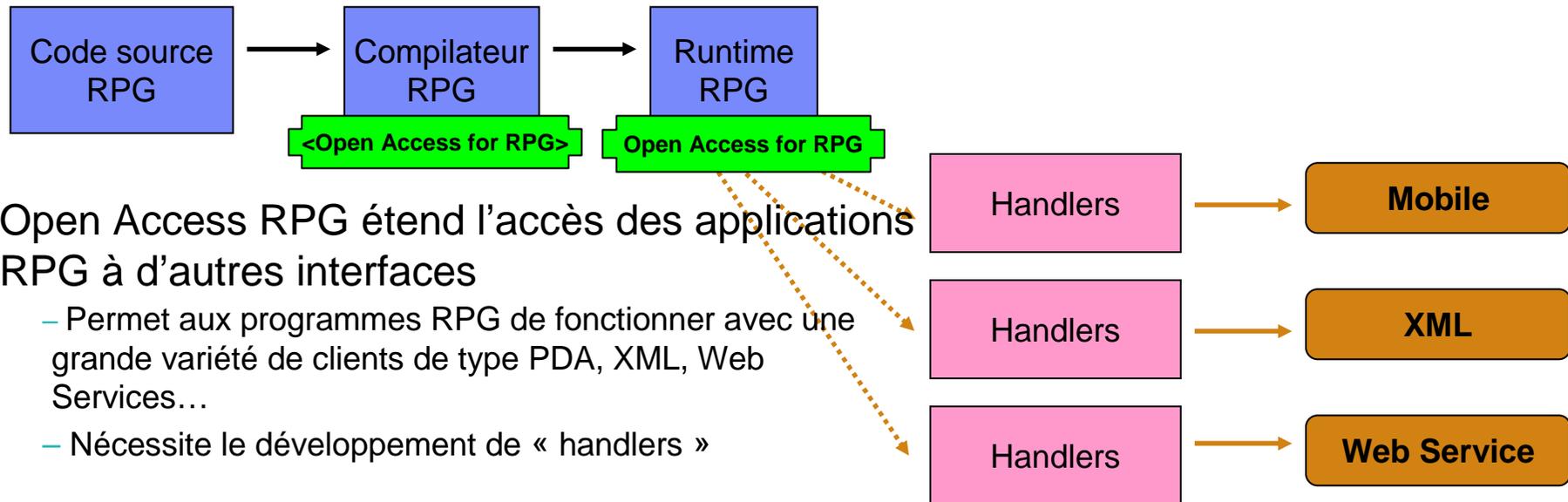
Rational Open Access: RPG Edition versus HATS

- Les applications RPG produisent un flot de données 5250
 - HATS et les outils tiers sont utilisés pour “habiller” ce flot pour les autres interfaces utilisateur



Applications RPG traditionnelles

Applications RPG avec Rational Open Access: RPG Edition



- Open Access RPG étend l'accès des applications RPG à d'autres interfaces
 - Permet aux programmes RPG de fonctionner avec une grande variété de clients de type PDA, XML, Web Services...
 - Nécessite le développement de « handlers »

RPG Open Access – Prérequis - Packaging

- IBM i 7.1 ou 6.1
- En développement :
 - Le mot-clé *handler* n'est pas supporté dans SEU mais uniquement dans RD Power
- En IBM i 6.1, la compilation du programme RPG qui contient le mot-clé *handler* nécessite une PTF :
 - SI39483 sur le 5761-WDS
- A l'exécution :
 - En IBM i 6.1 la PTF SI39480 sur le 5761-SS1 est nécessaire
 - Le produit 5733-OAR (Rational Open Access: RPG Edition) est nécessaire
 - Facturé au groupe logiciel
 - P05 : 464 €
 - P10 : 927 €
 - P20 : 2318 €
 - P30 à P60 : 4637 €

RPG Open Access – Les handlers disponibles

- Les handlers des éditeurs en 2010
 - IBM Lab Services
 - 3 handlers : FFM (File Format Messaging), Services Web et sockets
 - Looksoftware
 - Interfaces client riche, client léger et client mobile
 - Profound Logic
 - Interface client riche Web
 - VAI
 - Interface client riche Web (pour leur ERP uniquement)
 - Rocket Seagull Software
 - Services Web, SOA

RPG Open Access – Quelques détails

1 Le programme RPG

- On indique quels sont les fichiers qui sont en « open access » en spécifiant le mot-clé HANDLER en spécification F
- Le mot-clé HANDLER permet d'indiquer quel est le nom du programme ou du programme de service handler :
 - HANDLER(nom du programme ou du programme de service)
 - Le nom peut être défini en dur ou sous forme de variable

– Exemples :

```
Ffichier1 if      e k disk      handler('bib1/pgm1')
```

```
Ffichier2 cf      e      workstn handler('srvpgm1(proc1)')
```

- Le fichier peut être de type DISK, WORKSTN, PRINTER et il est généralement en corrélation avec le type d'unité cible :
 - Interface utilisateur → fichier WORKSTN
 - Fichier Excel → fichier PRINTER
 - Service Web → Fichier DISK ... / ...
- Le code qui suit est inchangé : on utilise les mêmes codes-opération (WRITE, READ, EXFMT...)

RPG Open Access – Quelques détails

■ 2 Le handler

- C'est un programme ou un programme de service ILE
- Il peut être écrit dans tout langage ILE (RPG, COBOL, CL, C, C++)
- Il gèrera toutes les opérations d'E/S sur les fichiers concernés (READ, WRITE, CHAIN, EXFMT...)
- Les données qui sont passées du/vers le programme RPG final et le handler sont passées sous la forme d'une structure de données
- Cette structure de données est définie dans le membre source QRNOPENACC dans les fichiers source QRPGLSRC, QCBLLSRC et H de la bibliothèque QOAR
- Il est possible de passer au handler des données autres que celles liées aux opérations d'E/S. Il suffit d'indiquer le nom d'une structure de données comme second paramètre (optionnel) du mot-clé handler :

```
Ffichier1 if e k disk handler('bib1/pgm1':ds1)
```

Fichiers en Open Access versus fichiers de type SPECIAL

- Le principe est le même : appel d'un programme externe pour gérer les opérations d'E/S sur le fichier

- Différences
 - Les seules opérations possibles sur un fichier SPECIAL sont celles valables pour un fichier séquentiel (OPEN, READ, WRITE, UPDATE, DELETE, CLOSE, FEOD). Un fichier « Open Access » peut être utilisé pour tout type d'unité (donc codes-opération CHAIN, READE, SETLL... possibles)
 - Avec un fichier SPECIAL, peu d'informations sont passées au handler (type d'opération et enregistrements). Avec un fichier Open Access, des informations comme le noms du fichier et des formats, le nom et le type des zones, etc. sont passées au handler
 - Réciproquement, en retour, avec un fichier SPECIAL, il n'est possible de récupérer que le code retour. Avec un fichier Open Access, il est possible de récupérer le code status RPG, le RRN, la touche de fonction utilisée, etc.
 - Avec un fichier SPECIAL, le handler est obligatoirement un programme. Avec un fichier Open Access, le handler peut être une procédure

RPG Open Access – Comment coder le handler ?

- Documentation



Attachments:

- [Latest version of the documentation: open_access_rpg_edition_5733OAR_2010_08_23.pdf](#)

Chapter 3. Coding the Open Access handler

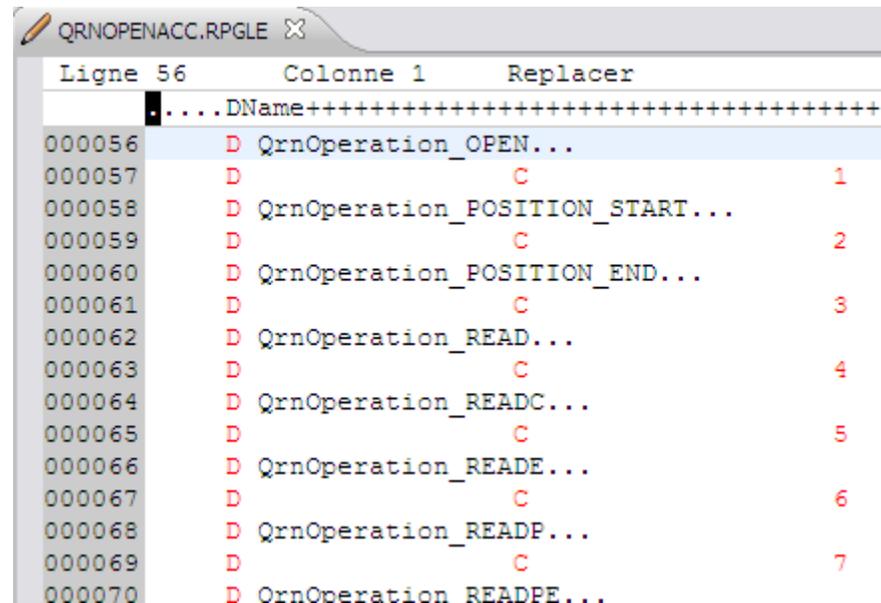
An Open Access handler can be coded in any ILE language.

Library Q0AR has copy files defining the data structures and constants related to the handler parameter for ILE RPG, ILE C, and ILE COBOL. The member name in each source file (QRPGLSRC, H, QCBLLSRC) is QRNOPENACC.

Library Q0AR is the library for product 5733-OAR. The source files can be used without having a license for the product.

RPG Open Access – La DS passée au handler – 1

- Se nomme QrnOpenAccess_T
- Contient :
 - Le type d'opération réalisée sur le fichier (READ, WRITE...)
 - Récupérée sous forme d'une valeur numérique dans la zone *rpgOperation*
 - Des constantes associées à ces valeurs numériques simplifient la lisibilité



```
QrNOENACC.RPGL E X
Ligne 56      Colonne 1      Replacer
.....DName+++++
000056      D QrnOperation_OPEN...
000057      D                      C                      1
000058      D QrnOperation_POSITION_START...
000059      D                      C                      2
000060      D QrnOperation_POSITION_END...
000061      D                      C                      3
000062      D QrnOperation_READ...
000063      D                      C                      4
000064      D QrnOperation_READC...
000065      D                      C                      5
000066      D QrnOperation_READE...
000067      D                      C                      6
000068      D QrnOperation_READP...
000069      D                      C                      7
000070      D OrnOperation READPE...
```

RPG Open Access – La DS passée au handler – 2

- Des zones récupérées par le handler sur le contexte de l'opération
 - Nom des bibliothèque, fichier et membre
 - Est-ce que le fichier est sur clé, est décrit en externe, etc. ?
 - *keyedFile, externalFile, compileFile, recordLevels, etc...*

- Des zones indiquant le « résultat » de l'opération et qui sont alimentées par le handler avant de retourner au programme RPG
 - *rpgStatus* : code d'erreur (0 ou entre 1000 et 2000)
 - *eof* : fin de fichier (sur opération READ)
 - *found* : enregistrement trouvé (sur opération CHAIN, SETLL, SETGT)
 - *equal, rrn, functionKey, deviceFeedback/ioFeedback/openFeedback (INFDS), etc.*

```

/* O  F1-F24, PRINT,ROLLUP etc  */
/*  See QrnFunctionKey_*        */
/*  - Unknown values cause an   */
/*  RPG exception                */
/*                                */
D  functionKey...
D                                3U 0
    
```

Colonne 1	Replacer
..DName+++++	
D QrnFunctionKey_None...	
D	C 0
D QrnFunctionKey_01...	
D	C 1
D QrnFunctionKey_02...	
D	C 2
D QrnFunctionKey_03...	
D	C 3

RPG Open Access – La DS passée au handler – 3

- Un pointeur (*stateInfo*) vers l'adresse d'une structure de données dont l'état sera maintenu lors de tous les appels au handler par le programme RPG
- Des pointeurs (*inputBuffer* et *outputBuffer*) vers des structures de données représentant les buffers d'entrée et sortie du fichier ou du format

```

D          *
/* O   Input buffer          */
/*   NULL if not used by opcode */
/*   - Length is given by   */
/*   inputBufferLen        */
D   inputBuffer...

D          *
/* I   Output buffer        */
/*   NULL if not used by opcode */
/*   - Length is given by   */
/*   outputBufferLen       */
D   outputBuffer...
    
```

- Un pointeur (*namesValues*) vers une structure de données (*QrnNamesValues_T*) contenant la liste des noms, types et valeurs des zones du format

```

/* I/O Alternate version of */
/*   I/O buffer information */
D   namesValues...

D QrnNamesValues_T...
D          DS          QUALIFIED TEMPLATE ALIGN
D   num          10I 0
D   field        LIKEDS(QrnNameValue_T)
D          DIM(32767)

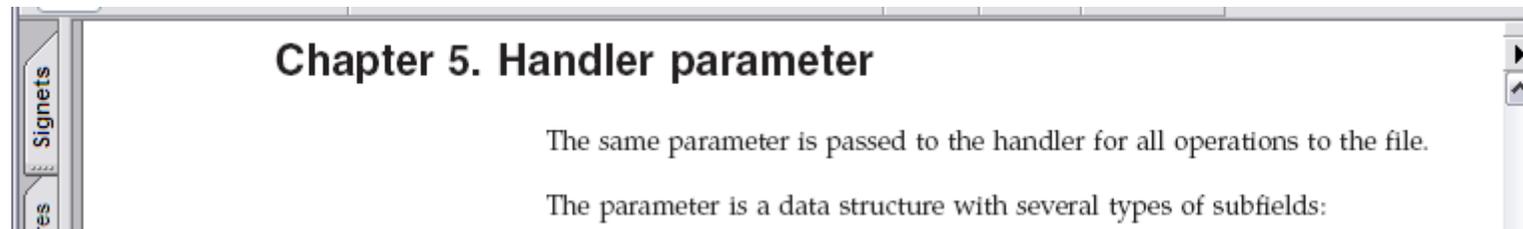
D QrnNameValue_T...
D          DS          QUALIFIED TEMPLATE
/* I   Name from external file */
D   externalName...
D          10A
/* I   Data type of field      */
/*   See QrnDatatype_*        */
D   dataType...
D          3U 0
/* I   Defined length of numeric */
/*   - Decimal: total digits   */
    
```

RPG Open Access – La DS passée au handler – 4

- Il existe donc deux façons d'alimenter / récupérer les données :
 - Via des « record buffers »
 - Via un tableau « Nom / valeur » (nom de la zone / valeur de la zone)

- On choisit en positionnant la valeur *useNamesValues* à *ON ("0") ou à *ON ("1") :
 - *OFF : record buffers (valeur par défaut)
 - *ON : tableau « Nom / Valeur »
 - Ce tableau permet en fait de récupérer :
 - Nom
 - Type
 - Longueur
 - Nombre de décimales
 - Format (date/heure)
 - Séparateur (date/heure)
 - Accepte la valeur indéfinie ?
 - Contient la valeur indéfinie ?
 - CCSID
 - Valeur
 - etc.

RPG Open Access – La DS passée au handler – Doc



The subfields of the main parameter structure

The subfields of the main parameter structure

The name of the main parameter structure is `QrnOpenAccess_T`.

Table 1. Subfields of `QrnOpenAccess_T`

Subfield	Type	Set by	Used by
<code>structLen</code>	UINT4	RPG	Handler
<code>parameterFormat</code>	CHAR(8)	RPG	Handler
<code>userArea</code>	Pointer ¹	RPG	Handler and RPG programmer
<code>stateInfo</code>	Pointer ³	Handler	Handler
<code>recordLevels</code> ⁴	Pointer ¹	RPG	Handler
<code>inputBuffer</code>	Pointer ^{1,2}	Handler	RPG
<code>inputNullMap</code> ⁴	Pointer ^{1,2}	Handler	RPG
<code>outputBuffer</code>	Pointer ^{1,2}	Handler	RPG
<code>outputNullMap</code> ⁴	Pointer ^{1,2}	Handler	RPG

Descriptions of the subfields

blocked

'1' if the file is defined to be blocked in the

Note: This value is provided for information but does not work with blocks of record, but it is used to control whether it block the records from dealing with.

commit

'1' if the file should be opened under commit

compileFile

The library and file that the RPG compiler uses for the description of an externally-described file.

RPG Open Access – 1^{er} exemple – Programme RPG

The screenshot displays the IBM Rational IDE interface. The main window shows the source code of a program named PB_TEST1.RPGLE. The code is as follows:

```
Ligne 1      Colonne 1      Replacer
.....HKeywords+-----
000100      H dftactgrp(*no)
000200      Fclients  if  e          disk  handler('PB_HDLR1(test1)')
000300      F                                usroprn
000400      /free
000500          open clients;
000600          read  clients;
000700          dsply cli_num;
000800          dsply cli_nom;
000900          close clients;
001000          *inlr = *on;
```

Below the code editor, a terminal window titled "Messages du prog" displays the following output:

```
Travail 984841/BOURGEOIS/QPADEV000G démarré
DSPLY  OPEN
DSPLY  READ
DSPLY   10
DSPLY  Client1
DSPLY  CLOSE
```

RPG Open Access – 1^{er} exemple – Programme handler

```

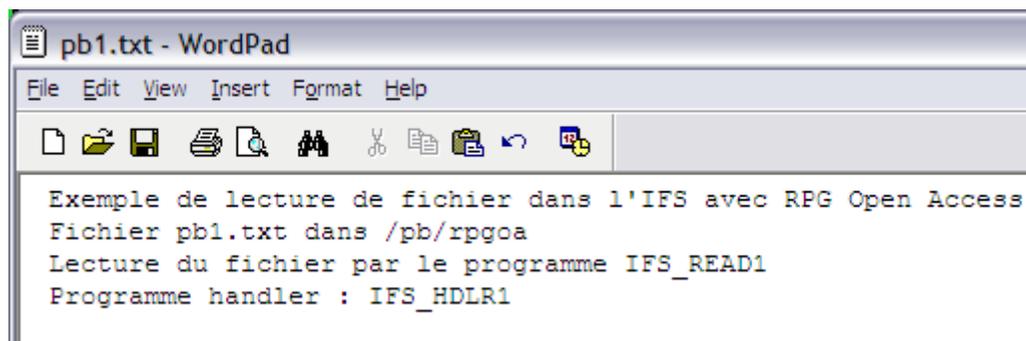
PB_HDLR1.RPGLE
Ligne 1      Colonne 1      Replacer
.....HKeywords+++++
000100      H nomain
000200      /copy qoar/qrpglesrc,qrnopenacc
000300
000400      D test1          pr          extproc('test1')
000500      D ds_oa           likeds(QrnOpenAccess_T)
000600
000700      P test1          b          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 Open Access – 2nd exemple – Programme RPG

Ligne 1	Colonne 1	Replacer
 1 2 3 4 5 6 7 8	
000100	*	Rational Open Access: RPG Edition
000200	*	Programme de lecture d'un fichier dans l'IFS - Utilisation d'un handler
000300	*	Le fichier IFS_F comporte une zone alpha nommée line de longueur 32740
000400	*	Le programme lit les lignes du fichier de l'IFS comme s'il lisait les
000500	*	lignes du fichier DISK et en imprime les 300 premiers caractères
000600		
000700	H	dftactgrp(*no)
000800		
000900	FIFS_F	if e disk handler('IFS_HDLR1(read_ifs)')
001000	F	usropr
001100	FQSYSPT	o f 300 printer
001200		
001300	D ds_ifs	ds 300
001400	/free	
001500	monitor;	
001600	open ifs_f;	
001700	read ifs_f;	
001800	dow not %eof;	
001900	ds_ifs = line;	
002000	write qsysprt ds_ifs;	
002100	read ifs_f;	
002200	enddo;	
002300	on-error 1217;	
002400	dsply 'Erreur lors de l'ouverture/fermeture du fichier IFS';	
002500	on-error 1299;	
002600	dsply 'Tentative d'opération non supportée sur fichier IFS';	
002700	endmon;	
002800	*inlr = *on;	
002900	/end-free	

Ligne 1	Colonne 1	Replacer
 +A* 1 2 3 4 5	
000100	*	Ce fichier ne contient pas de données
000200	*	Il est utilisé comme interface pour le handler
000300	A	R IFS_REC
000400	A	LINE 32740A VARLEN

RPG Open Access – 2nd exemple – Résultat



pb1.txt - WordPad

File Edit View Insert Format Help

Exemple de lecture de fichier dans l'IFS avec RPG Open Access
 Fichier pb1.txt dans /pb/rpgoa
 Lecture du fichier par le programme IFS_READ1
 Programme handler : IFS_HDLR1

```

                                Fichier spoule
Fichier . . . . . :   QSYSPRT                                Page/Ligne  1/1
Contrôle . . . . . :   _____                          Colonnes   1 - 78
Recherche . . . . . :   _____
* . . . + . . . 1 . . . + . . . 2 . . . + . . . 3 . . . + . . . 4 . . . + . . . 5 . . . + . . . 6 . . . + . . . 7 . .
Exemple de lecture de fichier dans l'IFS avec RPG Open Access
Fichier pb1.txt dans /pb/rpgoa
Lecture du fichier par le programme IFS_READ1
Programme handler : IFS_HDLR1
  
```

RPG Open Access – 2nd exemple – Programme handler - 1

```

IFS_READ1.RPGLE  IFS_F.PF  IFS_HDLR1.RPGLE X
Ligne 1          Colonne 1  Replacer
.....1.....2.....3.....4.....5.....6.....7.....8
000100          * Programme handler pour lecture d'un fichier dans l'IFS
000200          * Dans cet exemple, le nom du fichier est défini en dur
000300          * Mais il pourrait être passé en paramètre (par la zone userArea de la DS)
000400          * Hypothèses :
000500          * - le fichier IFS existe
000600          * - les lignes du fichier ne dépassent pas 32740 caractères
000700
000800          H nomain
000900          /copy qoar/qrpglesrc,qrnopenacc
001000          /copy qsysinc/qrpglesrc,ifs
001100
001200          D fichier_ifs      c              '/pb/rpgoa/pb1.txt'
001300
001400          D buffer_t         e ds          extname('IFS_F') template
001500          * (IFS_F est le fichier DISK utilisé dans le programme RPG)
001600
001700          D state_t          ds            qualified template
001800          D code              10i 0
001900          *-----*
002000          * Handler appelé par le programme RPG
002100          D read_ifs         pr            extproc('read_ifs')
002200          D ds_oa            ds            likeds(QrnOpenAccess_T)
002300          *-----*
002400          * Procédures de gestion des fichiers dans l'IFS
002500          D openFile         pr            10i 0
002600          D path              5000a      varying const
002700
002800          D closeFile        pr
002900          D status           10i 0 value
003000

```

RPG Open Access – 2nd exemple – Programme handler - 2

```

IFS_READ1.RPGLE  IFS_F.PF  IFS_HDLR1.RPGLE X
Ligne 31      Colonne 1      Replacer
.....DName+++++++ETDsFrom+++To/L+++IDc.Keywords+++++++
003100      D readFile          pr              n
003200      D status              10i 0 value
003300      D buffer              likeds (buffer_t)
003400      *-----
003500      * Handler appelé par le programme RPG
003600      P read_ifs              b              export
003700      D                          pi
003800      D ds_oa                  likeds (QrnOpenAccess_T)
003900
004000      D state                  ds              likeds (state_t)
004100      D                          based(p_state)
004200      D
004300      D buffer                  ds              likeds (buffer_t)
004400      D                          based(p_buffer)
004500      /free
004600          p_state = ds_oa.stateInfo;
004700
004800          // Ouverture du fichier dans l'IFS
004900      if ds_oa.rpgOperation = QrnOperation_OPEN;
005000          p_state = %alloc(%size(state));
005100          clear state;
005200          ds_oa.stateInfo = p_state;
005300          state.code = openFile(fichier_ifs);
005400          if state.code < 0;
005500              ds_oa.rpgStatus = 1217; // erreur d'ouverture
005600          endif;
005700
005800          // Lecture du fichier dans l'IFS
005900      elseif ds_oa.rpgOperation = QrnOperation_READ;
006000          p_buffer = ds_oa.inputBuffer;
006100          ds_oa.eof = readFile (state.code : buffer);

```

RPG Open Access – 2nd exemple – Programme handler - 3

```

IFS_READ1.RPGLE  IFS_F.PF  IFS_HDLR1.RPGLE X
Ligne 63      Colonne 1      Replacer
.....1.....2.....3.....4.....5.....6.....7.....8
006300      // Fermeture du fichier IFS
006400      elseif ds_oa.rpgOperation = QrnOperation_CLOSE;
006500          closeFile (state.code);
006600          state.code = -1;
006700          dealloc(n) p_state;
006800          ds_oa.stateInfo = *null;
006900
007000          // Si autre opération : erreur
007100      else;
007200          ds_oa.rpgStatus = 1299;
007300      endif;
007400  /end-free
007500  P read_ifs          e
007600
007700  *-----
007800  * Procédure d'ouverture d'un fichier dans l'IFS
007900  P openFile          b
008000  D                  pi          10i 0
008100  D  path              5000a  varying const
008200  /free
008300      // Appel de la procédure open (voir qsysinc/qrpglesrc,ifs)
008400      return open(path : O_RDONLY + O_TEXTDATA);
008500  /end-free
008600  P openFile          e

```

RPG Open Access – 2nd exemple – Programme handler - 4

```

008800 * Procédure de fermeture d'un fichier dans l'IFS
008900 P closeFile      b
009000 D                pi
009100 D  status          10i 0 value
009200 D  rc              s      10i 0
009300 /free
009400 // Appel de la procédure close (voir qsysinc/qrpglesrc,ifs)
009500 rc = close(status);
009600 /end-free
009700 P closeFile      e
009800 -----
009900 * Procédure de lecture d'un fichier dans l'IFS
010000 P readFile       b
010100 D                pi          n
010200 D  status          10i 0 value
010300 D  buffer          likeds(buffer_t)
010400
010500 D  numBytes        s      10i 0
010600 D  z1a             s      1a
010700 D  CR              c      x'0A'
010800 D  LF              c      x'25'
010900 /free
011000 buffer.line = '';
011100 // Appel de la procédure read (voir qsysinc/qrpglesrc,ifs)
011200 numBytes = read(status : %addr(z1a) : %size(z1a));
011300 if numBytes = 0;
011400     return '1'; // fin de fichier atteinte
011500 endif;
011600 dow (numBytes > 0) and (z1a <> CR) and (z1a <> LF);
011700     buffer.line += z1a;
011800     numBytes = read(status : %addr(z1a) : %size(z1a));
011900 enddo;
012000 return '0'; // fin de fichier non atteinte
012100 /end-free
012200 P readFile       e

```

RPG Open Access – Coding des handlers - Pour en savoir plus



The screenshot shows a web browser window displaying a developerWorks Wiki page. The page title is "Rational Open Access. RPG Edition". The breadcrumb trail is "You are in: RPG Cafe > Welcome > Rational Open Access. RPG Edition". The page was updated yesterday at 6:28 PM by user "cafe". The page content describes Rational Open Access: RPG Edition as a way for RPG programmers to use simple and well-understood access resources and devices not directly supported by RPG. It lists several examples of resources: Browsers, Mobile devices, Cloud computing resources, Web services, External databases, and XML files. The page also includes a sidebar with navigation links for "RPG Cafe", "Overview", "Members", "Wiki", and "Blog".

developerWorks > My developerWorks > Connect >

Home Profiles Groups Blogs Bookmarks Activities Files Wikis

This Wiki Search

My developerWorks: Wikis

Public Wikis My Wikis

RPG Cafe

You are in: RPG Cafe > Welcome > Rational Open Access. RPG Edition

Rational Open Access. RPG Edition

Updated yesterday at 6:28 PM by cafe | Tags: openaccess, rpgcafe

Page Actions

(Created on: Apr 14, 2010 6:21 PM by barbara_morris - Last Modified: Jul 27, 2010 1:43 PM by barbara_m)

Rational Open Access: RPG Edition provides a way for RPG programmers to use the simple and well-understood access resources and devices that are not directly supported by RPG. Open Access opens up RPG's file I/O to anyone to write innovative I/O handlers to access other devices and resources such as:

- o Browsers
- o Mobile devices
- o Cloud computing resources
- o Web services
- o External databases
- o XML files

Update 2010-07-27. 2010-08-23:

RPG Open Access – Coding des handlers - Pour en savoir plus



The screenshot displays the IBM Systems magazine website. The top navigation bar includes 'IBM i', 'AIX', 'MAINFRAME', and 'POWER'. The main header features the 'IBM Systems magazine' logo and a 'NEWSLETTERS' section with options to 'SUBSCRIBE', 'CURRENT ISSUE', and 'ARCHIVE'. Below the header is a menu with categories: 'ADMINISTRATOR', 'DEVELOPER', 'TRENDS', 'TIPS & TECHNIQUES', 'CASE STUDIES', 'STORAGE', 'PRODUCT NEWS', 'ENDPGM', and 'BUY'. The main content area shows the breadcrumb 'Developer > RPG' and an 'e-Newsletter Exclusive' tag. The article title is 'Getting a Handle on RPG's Open Access', with a subtitle 'Discover how simple it is to create a generic handler'. The author information is 'July 2010 | by Jon Paris and Susan Gantner'. A photograph of a hand on a steering wheel driving on a road is visible below the text.

RPG Open Access – Coding des handlers - Pour en savoir plus



Getting a Handle on RPG's Open Access

These are the assorted source files for the Open Access for RPG example published in the July 2010 issue of the iSeries EXTRA newsletter. You can [find the article here](#).

The "User" Program

This uses the OAR Handler program [shown below](#). Note that the only difference in the code from a conventional write-disk-file is the use of the HANDLER keyword on the file's F-spec. The second parameter to the handler is used to supply the name of the IFS file to use.

```
H dftactgrp(*no) option(*NoDebugIO : *SrcStmt)

// IFS output file - file definition specifies fields to output
FIFS_OUT1 o e Disk Handler('HND_IFS_J2' : ifs_info1)
```

RPG Open Access – Coding des handlers - Pour en savoir plus



VOLUBIS
CONSEIL ET FORMATION SUR I5/OS ET OS/400

INFOS DU MOIS | LE MOIS DERNIER | PAUSE-CAFÉ | TESTS | LIENS | AF400 | SPLF2 | FORUM | CONTACTS

À LA UNE
Informations du mois / Nouveautés

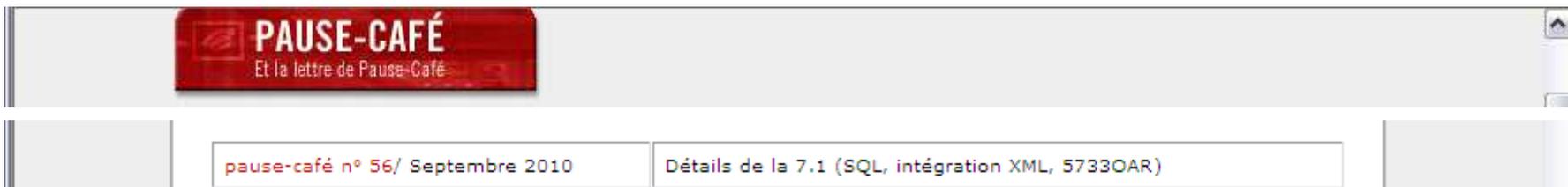
Rational Open Access, RPG edition (5733OAR)

Ce produit permet d'associer à une déclaration de fichier, une routine (écrite ou achetée) qui sera utilisée à la place des routines standard lors des entrées/sorties.

Voici ce que dit l'annonce "Simplifies the development of transaction processing applications for mobile devices and web services when developing with Rational Developer for Power. Removes the requirement for the 5250 data stream"

Pour faire fonctionner ce module (facturable) il faut les PTF **SI39480** et **SI39912**

Voyez dès à présent nos **premiers tests** sur le sujet !



PAUSE-CAFÉ
Et la lettre de Pause-Café

pause-café n° 56/ Septembre 2010	Détails de la 7.1 (SQL, intégration XML, 5733OAR)
----------------------------------	---

Rational Open Access: RPG Edition

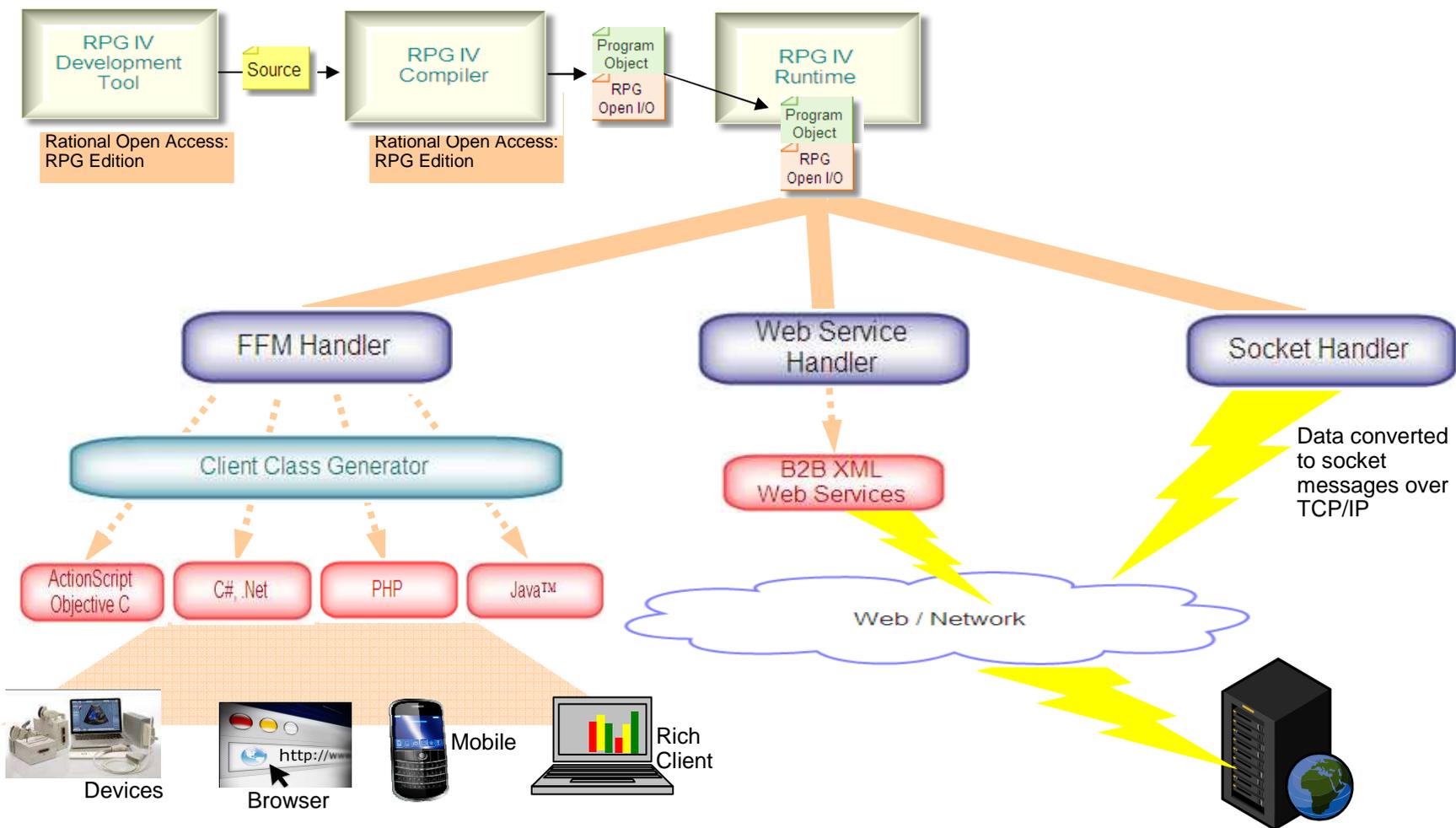
*Quelques mots sur les handlers de
IBM Lab Services*

IBM Lab Services et RPG Open Access

- Les missions / prestations de IBM Lab Services
 - Objectif : Faciliter l'adoption des technologies et des produits IBM
 - Prestations : de service (étude d'architecture, implémentation de solutions, développement, étude de performances...)
 - IBM La Gaude, à côté de Nice
 - Compétences techniques de haut niveau, en relation avec les labs

- IBM Lab Services a développé 3 handlers RPG Open Access :
 - FFM (File Format Messaging) handler
 - Interfaçage d'applications RPG avec des applications Java, PHP, C#
 - Web Services handler
 - Connexion d'applications RPG à des interfaces B2B (appel de Services Web externes, intégration avec des applications externes, etc.)
 - Sockets handler
 - Serveur de sockets TCP/IP

Trois handlers pour interfacer les applications RPG



RPG OA – Quelques mots sur le handler FFM

■ Objectif

- Faciliter la communication (conversations/transactions) entre des programmes RPG et des programmes Java, PHP, C#, etc.

■ Principe

- Le handler FFM est un framework/toolkit permettant de définir des transactions client-serveur basées sur des messages
 - Le modèle recommandé est de définir des transactions « stateless » (événementiel, pools de connexion)
 - Les messages sont définis à partir de fichiers IBM i (fichiers écran, fichiers ICF, etc.)
 - Côté client, le toolkit permet de générer des classes (Java, PHP, C#) à partir des définitions des formats de ces fichiers
 - Côté serveur, le programme RPG utilise des ordres READ et WRITE

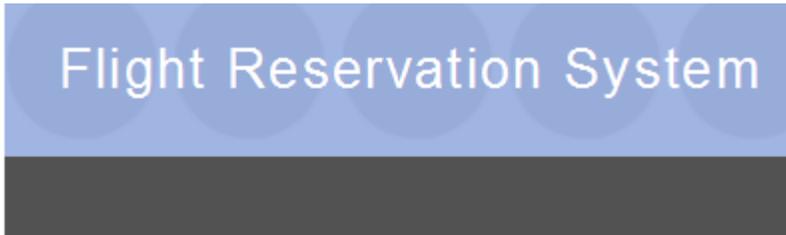
RPG OA – Quelques mots sur le handler FFM

- Le handler FFM :

- Ne permet pas de générer automatiquement des interfaces Web, Web 2.0, mobile, etc.
- Nécessite des compétences RPG pour développer la partie serveur et des compétences Java, PHP ou C# pour développer la partie client
- A pour objectif d'aider les développeurs à mettre en place une architecture client-serveur basée sur des messages :
 - Le toolkit inclut des classes de base pour gérer la communication entre le client et le serveur (connexion, transformation des données, conversations, etc.)
 - Le toolkit permet de générer des classes à partir des formats de fichiers IBM i
 - La conversion des données est gérée par le framework
 - Côté serveur il suffira de développer un programme RPG « dispatcher »
 - Récupération du nom du format et exécution de l'opération associée

RPG OA – Le handler FFM – Exemple

- Application IBM FLGHT400 – Réservation de vols



Ville de départ
 Ville d'arrivée
 Date de départ
 Nombre de billets

- First
 Business
 Economy

création d'une réservation

mise à jour d'une réservation

sélection

AirLine Code: NWA Flight Number: 4457304
 Departure City Code: RCH Departure Time: 10:34 AM
 Arrival City Code: CHG Arrival Time: 12:34 PM

Price: 199,00
 Tax: 7,96
 TotalDue: 206,96

Nom du client

flight number	airline code	departure city code	arrival city
4157301	NWA	RCH	CHG
4257302	NWA	RCH	CHG
4357303	NWA	RCH	CHG
4457304	NWA	RCH	CHG

RPG OA – Le handler FFM – Exemple - Quelques détails

- Exemple de fichier requête
Fichier RQINFO

A	R	RQFLGHTINF		
A		FLGHTNUM	7	
A	R	RQCITYNM		
A		INITIALS	3	
A		FROMTO	1	
A	R	RQCOMPPRC		
A		BASEPRC	3	
A		SERVCLSS	1	
A		TICKETS	3	0
A	R	RQFFLGHTDT		
A		FROM	16	
A		TO	16	
A		MDY	8	
A	R	RQFFRMCITS		
A		CITYNAME	16	
A		LISTTYPE	1	
A		COUNTREQ	10	0

- Exemple de fichier réponse
Fichier RSFLGHTINF

A	R	FLGHTINFR		
A		AIRLINE		3
A		FLIGHT		7
A		FLIGHTDOW		2
A		DEPCITY		3
A		ARRCITY		3
A		DEPTIME		8
A		ARRTIME		8
A		PRICE		3

RPG OA – Le handler FFM – Exemple - Quelques détails

- Côté client – Génération des classes à partir des fichiers IBM i
 - Alimentation des données en entrée à envoyer au READ RPG
 - Récupération des données en sortie envoyées par le WRITE RPG
 - Utilisées en conjonction avec les classes de base

```
Generate Open Access Class (GENCLSOA)

Type choices, press Enter.

Files . . . . . _____ Name, generic*
  Library . . . . . *LIBL Name, *LIBL, *CURLIB
Package or namespace . . . . . _____

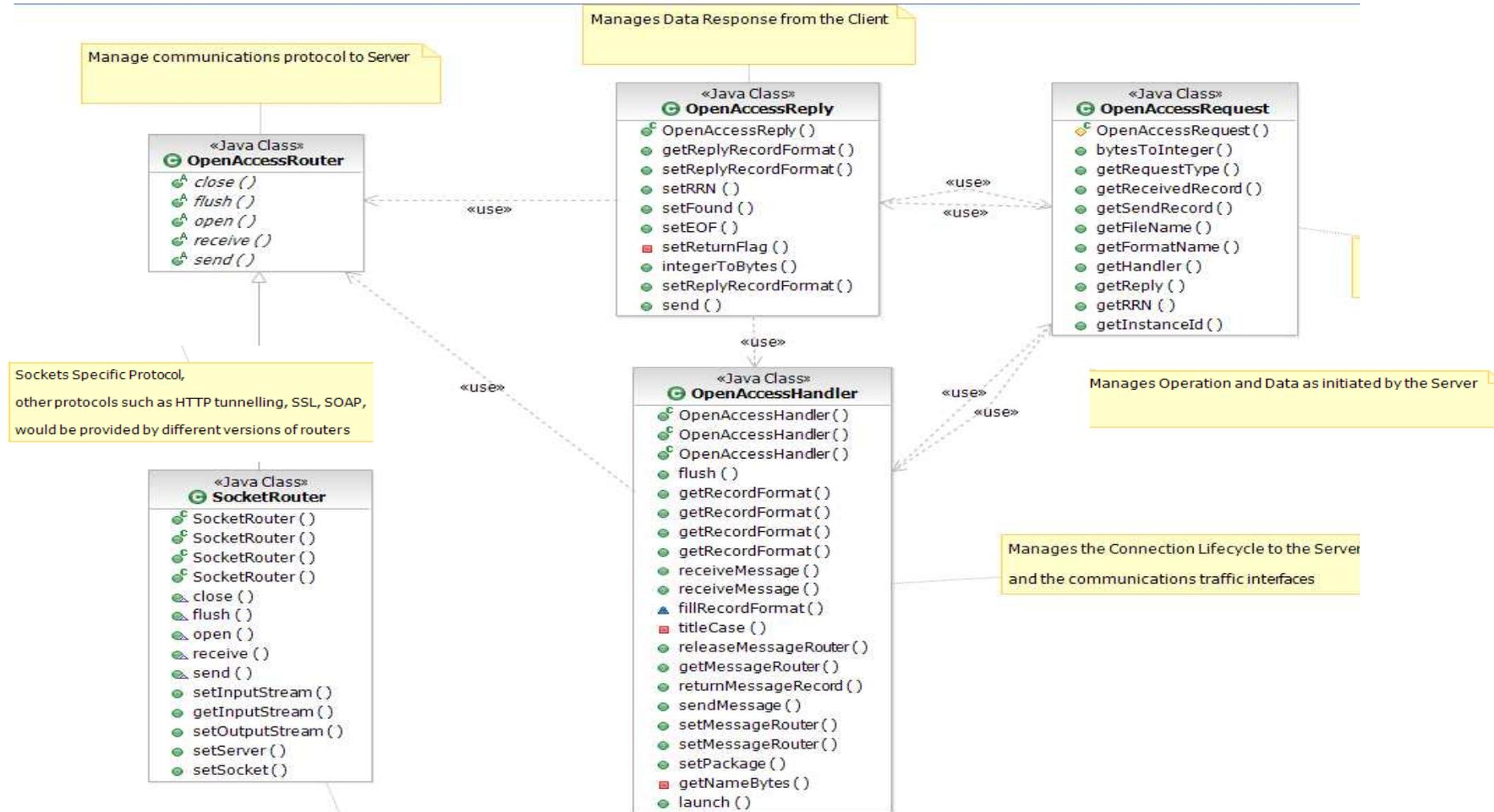
Common package or namespace . . *LANGDFT

Path . . . . . _____

Generated source language . . . *JAVA *JAVA, *CSHARP, *PHP
```

RPG OA – Le handler FFM – Exemple - Quelques détails

- Côté client – Les classes de base fournies avec le handler



RPG OA – Le handler FFM – Exemple - Quelques détails

- Côté serveur - Le programme RPG dispatcher – 1/2

```

F400DISP.RPGLE
Ligne 12      Colonne 79      Replacer
.....1.....2.....3.....4.....5.....6.....7...
000203      * The request file is a multi-format file that contains separate
000204      * formats for each request type
000205      *
000206      fRQINFO      if      e          workstn handler (OAHANDLER)
000207      f          infds (iofeedback)
000208      *
000209      * The response files each contain one format (although they could
000210      * be combined)
000211      *
000600      fRSCITYINFOcf  e          workstn handler (OAHANDLER)
000601      fRSCITYNM   cf  e          workstn handler (OAHANDLER)
000800      fRSCUSTINF  cf  e          workstn handler (OAHANDLER)
000801      fRSCUSTNM   cf  e          workstn handler (OAHANDLER)
001000      fRSFLGHTINFcf e          workstn handler (OAHANDLER)
001001      fRSGETCNBR cf  e          workstn handler (OAHANDLER)
001002      fRSORDSUMM  cf  e          workstn handler (OAHANDLER)
001003      fRSPRICEINFcf e          workstn handler (OAHANDLER)
001004      fRSRESINFO  cf  e          workstn handler (OAHANDLER)
001005      fRSRESFLGHTcf e          workstn handler (OAHANDLER)
001006      fRSUPDORD   cf  e          workstn handler (OAHANDLER)
001200      *
001201      * All the files use the Lab Services handler
001202      *
001301      d OAHANDLER      c          'QZRDOARPG/QZRDOASRV (HANDLER) '
001302      *
  
```

RPG OA – Le handler FFM – Exemple - Quelques détails

- Côté serveur - Le programme RPG dispatcher – 2/2

```

Colonne 5      Replacer
...1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
dou %error; // Exit on error
read(e) rqinfo; // Read from request file
if not %error;
select;
  when iofeedback.formatname = 'RQFLGHTINF'; // Flight Information
    GetFlightInfo (FLGHTNUM:FlightArray(1));
    WrtFlightInfo (FlightArray(1));
  when iofeedback.formatname = 'RQCITYNM'; // City Name
    GetCityName (INITIALS:FROMTO:CITYNAME);
    write(e) citynmr;
  when iofeedback.formatname = 'RQCOMPPRC'; // Calculate Price
    ComputePrice (BASEPRC:SERVCLSS:TICKETS:COMPPRICE:TAX:TOTAL);
    write(e) priceinfr;
  when iofeedback.formatname = 'RQFFLGHTDT'; // Fligh
    FindFlights (FROM:TO:MDY:count:FlightArray);
    for i = 1 to count;
      WrtFlightInfo (FlightArray(i));
    endfor;
  when iofeedback.formatname = 'RQFFRMCITS'; // Find
    FindFromCities (CITYNAME:LISTTYPE:
      Limit (COUNTREQ:%elem(CityArray)):
      count:CityArray);
    for i = 1 to count;
      WrtCityInfo (CityArray(i));
    endfor;
  when iofeedback.formatname = 'RQFINDCUST'; // Find Customers
    FindCustomers (CUSTNAME:LISTTYPE:

```

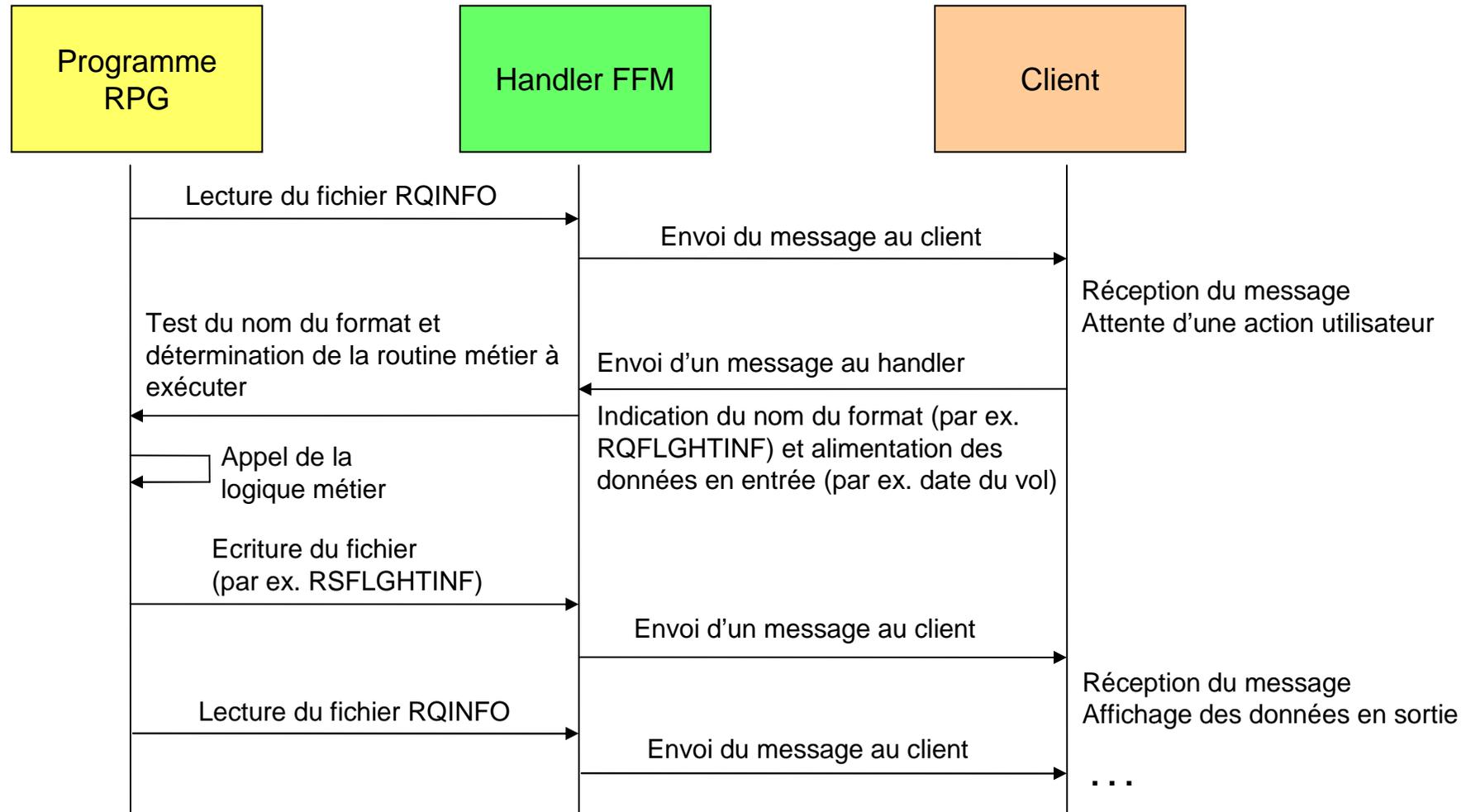
```

p GetFlightInfo b export
*****
d GetFlightInfo pi
d FlightNumber 7 const
d Flight likeds (FlightInfo)
d FlightKey s 9s 0
/free
FlightKey = %dec (FlightNumber:7:0);
chain(e) FlightKey FlightI;
if %found;
  Flight.Airline = AIRLNM;
  Flight.Flight = %char (FLGHTN); ... / ...

```

RPG OA – Le handler FFM – Exemple - Quelques détails

■ Schéma général des flux



Annexe

Rational Open Access: RPG Edition

*Le handler de **looksoftware***

Looksoftware et itheis

Modernisation d'applications
depuis 1995

Pionnier de
l'architecture
dynamique

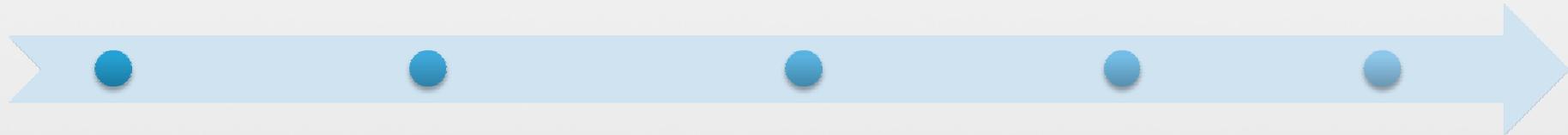
Réutilisation est
notre niche

3000 clients et
partenaires dans
plus de 50 pays

1 Million d'utilisateurs connectés



looksoftware – Les solutions



Reuse

Re-purposing

New

newlook

soarchitect

RPG

lookserver for Open Access

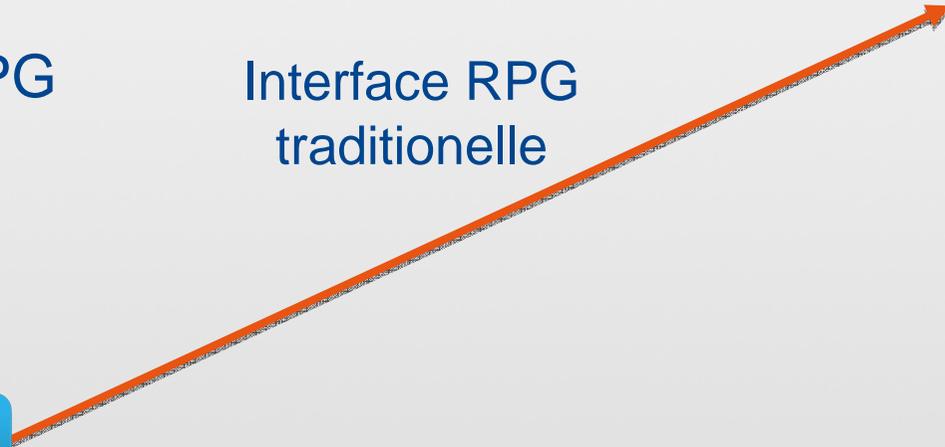
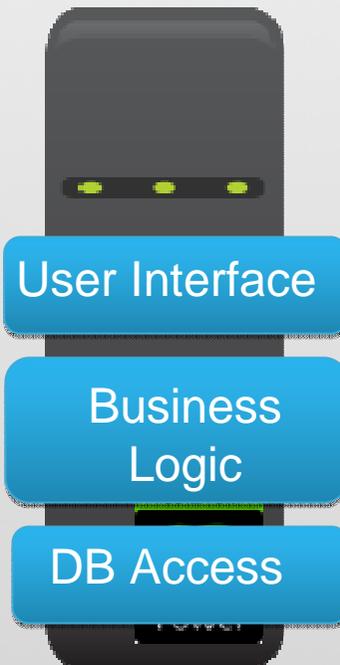
RPG OA

Rational Open Access: RPG Edition

Ecran 5250

Applications RPG

Interface RPG
traditionelle



Rational Open Access: RPG Edition

Applications RPG

```
F      Define the Handler  
D*****  
C*  
C      Exfmt  Dsp1  
C*      :  
C*      :  
C      Exfmt  Dsp2  
C      :  
C
```

Interface RPG
traditionelle

5250 Screens



Browsers



Desktop



Mobile
Devices



Data

Handlers

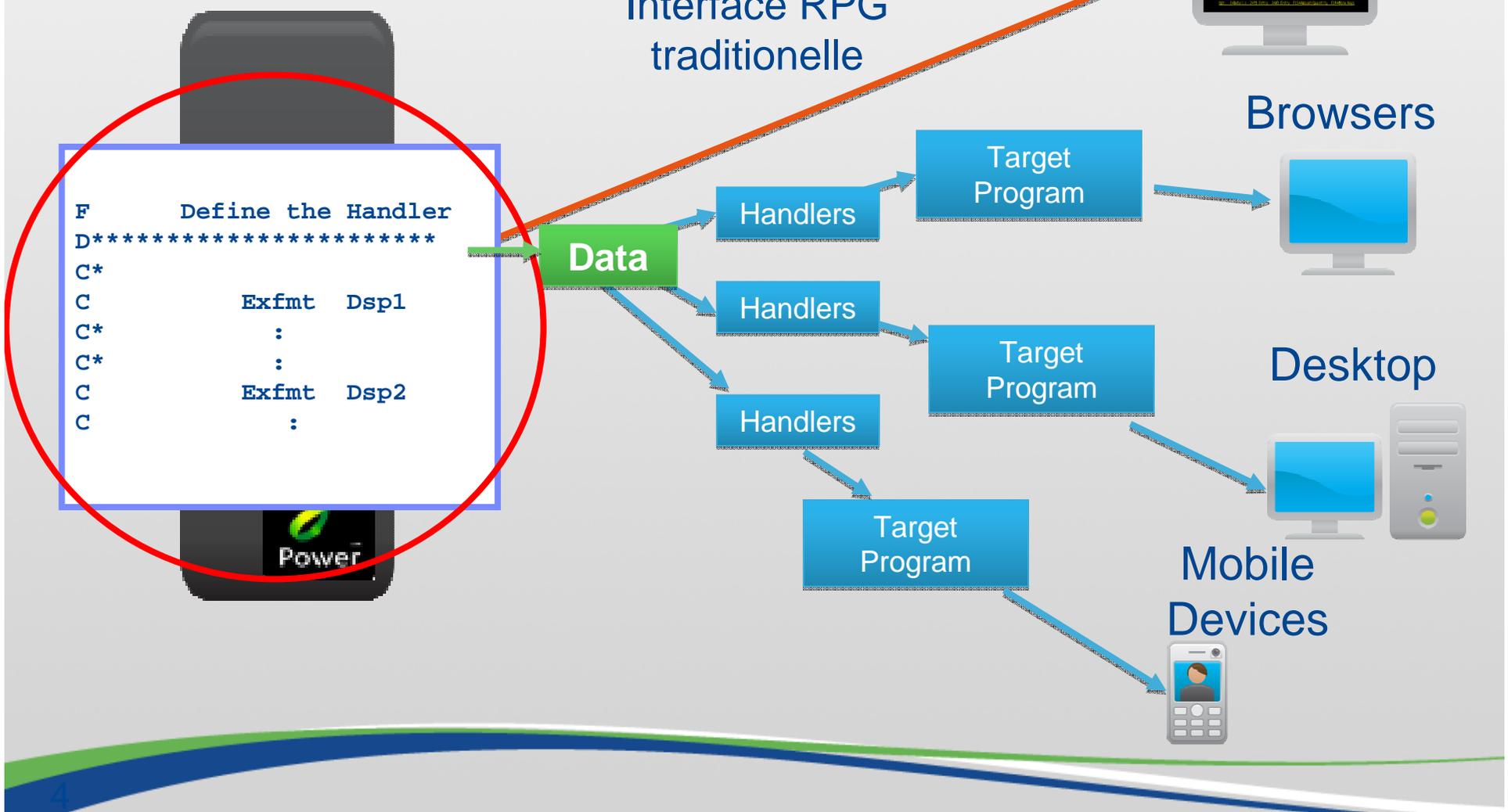
Target
Program

Handlers

Target
Program

Handlers

Target
Program



Rational Open Access: RPG Edition

RPG Applications

```
F      Define the Handler  
D*****  
C*  
C      Exfmt   Dsp1  
C*      :  
C*      :  
C      Exfmt   Dsp1  
C      :  
C
```

Interface RPG
traditionelle

5250 Screens



Browsers



Desktop



Mobile
Devices



Data

looksoftware
Handler

Avant Open Access

- Datastream 5250
- Données supplémentaires recueillies manuellement avec DDM / RPC ou webservice autour du 5250



Avec Open Access

- Display file entier
- Mémoire / buffer
- Sous-fichier entier
- Toutes les touches de fonctions
- Toutes les valeurs de liste
- DDM / RPC / WS toujours disponibles



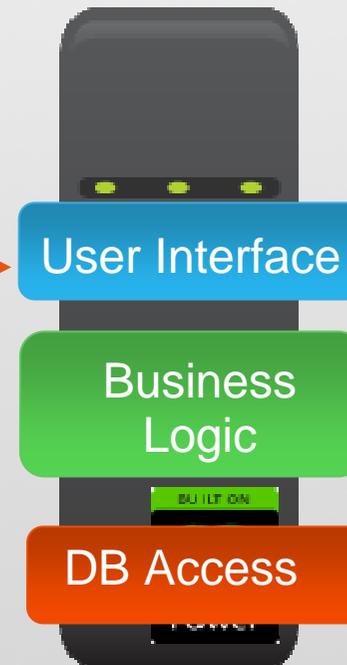
Véritable architecture multi-tiers

Applications
5250 RPG

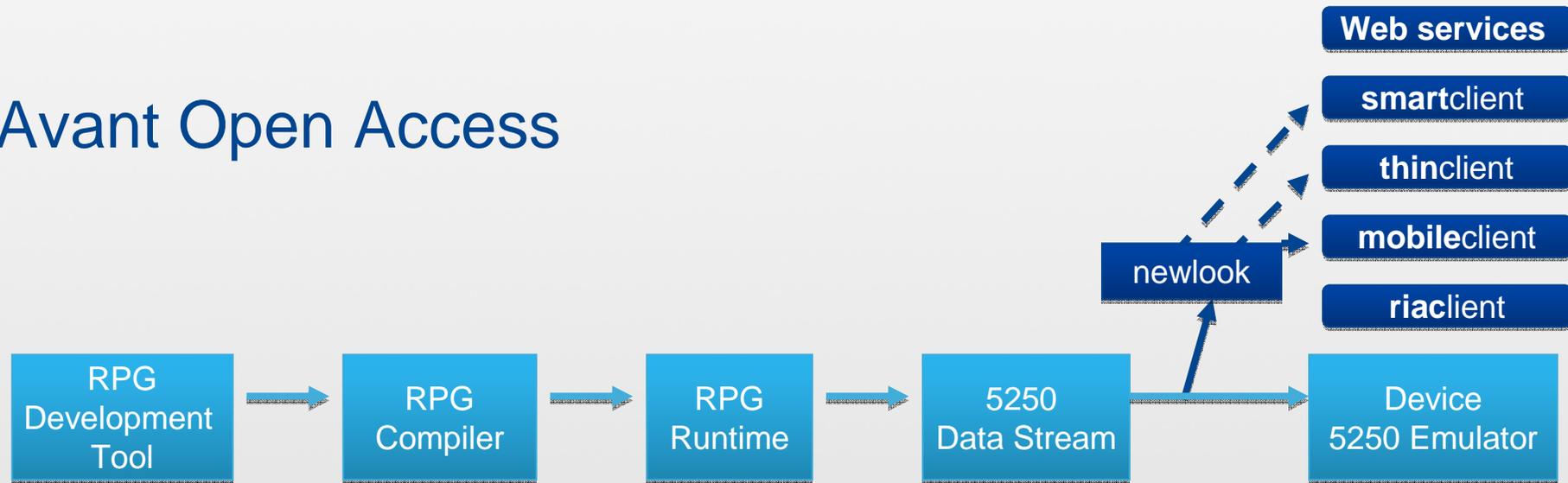


Séparation de la
logique métier et
de l'interface (UI)

Applications
RPG OA



Avant Open Access

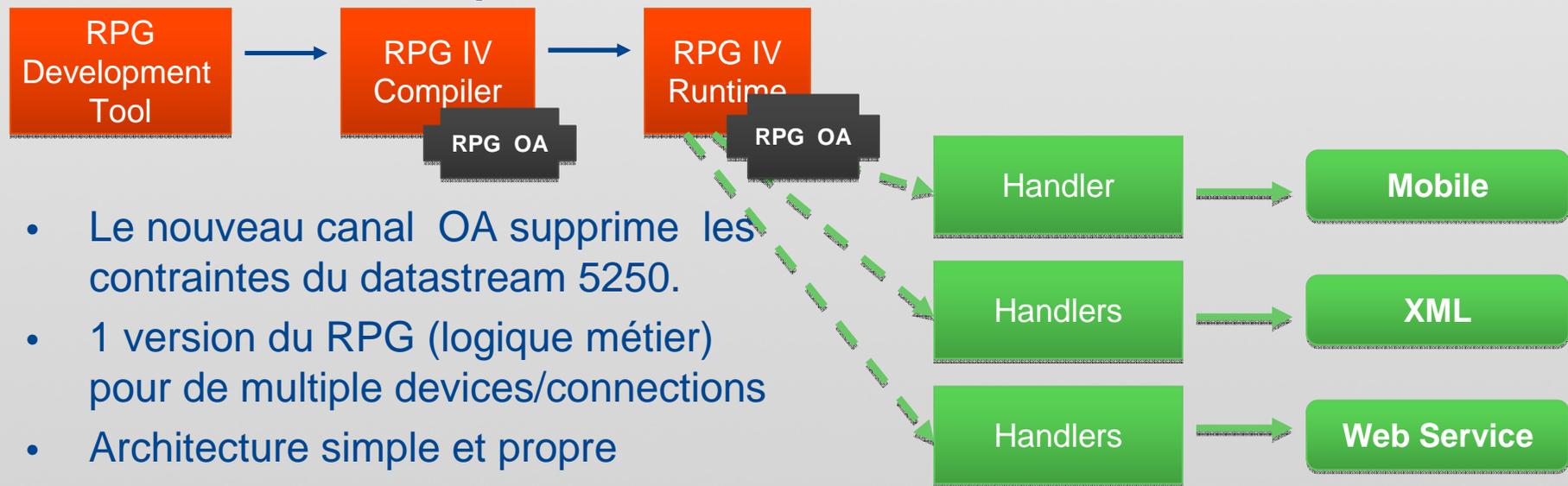


- Contraintes 5250 datastream
 - 24 lignes X 80 colonnes
 - Aucun type de donnée riche
 - Images etc..
- Nouvelles exigences
 - Ecrire nouveaux RPG et 5250 et revamper?
- Une architecture moderne est nécessaire pour soutenir les technologies / dispositifs émergents

Avec Open Access



Le canal natif Open Access

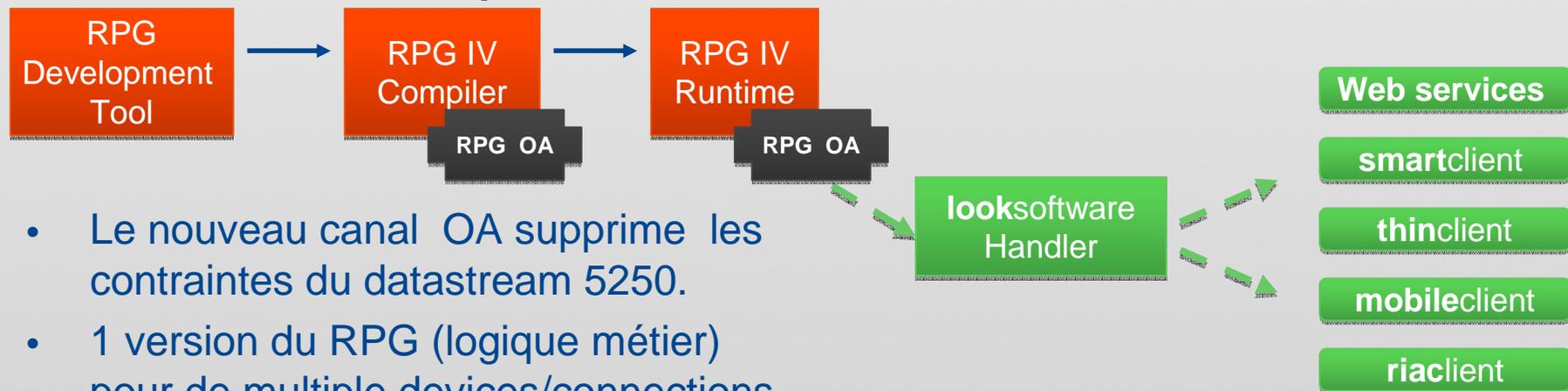


- Le nouveau canal OA supprime les contraintes du datastream 5250.
- 1 version du RPG (logique métier) pour de multiple devices/connections
- Architecture simple et propre

Avec Open Access

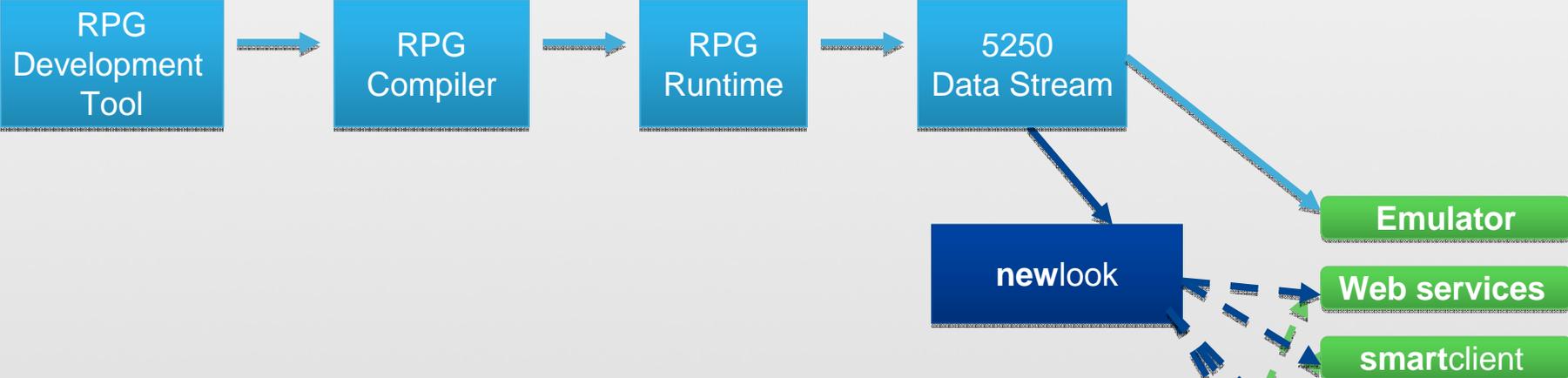


Le canal natif Open Access



- Le nouveau canal OA supprime les contraintes du datastream 5250.
- 1 version du RPG (logique métier) pour de multiple devices/connections
- Architecture simple et propre

Avec Open Access



Le canal natif Open Access



Pourquoi ROA est une avancée majeure ?

- Support RPG natif pour les nouveaux devices des technologies d'aujourd'hui et de demain
- Supprime les contraintes du 5250
- RPG existants deviennent multi-tiers



Pourquoi ROA est une avancée majeure ?

- Futur effectif pour le RPG
- L'architecture du Handler est ouverte
 - N'est pas liée à des technologies d'ISV
- Facile à implémenter
 - Logique métier en RPG
 - Le handler s'occupe de la "plomberie" (connection/device)



Que doit faire le programmeur ?

Looksoftware met a disposition 2 commandes :

- Commande CVTDSPFXML
- Commande CVTRPGROA

Conversion des display file en XML

```
1 - v6 FRS - newlook
File Session Edit View Macros Tools Keyboard Window Help
Cvt DSPF definition to XML (CVTDSPFXML)

Type choices, press Enter.

Display . . . . . _____ Name, *ALL
Library . . . . . _____ Name
Source file . . . . . *OBJD _____ Name, *OBJD, QDDSSRC
Library . . . . . *OBJD _____ Name, *OBJD, *OBJLIB
Source member . . . . . *OBJD _____ Name, *OBJD, *OBJ
To directory . . . . . *DFT _____

_____
_____
_____

Bottom
F3=Exit   F4=Prompt   F5=Refresh   F12=Cancel   F13=How to use this display
F24=More keys
```

Conversion des display file en XML

```
-<fmt name="AIAGIWD1" lib="CUST0004" dspf="AIDGIWD" type="NORMAL" mode="80" X="1" Y="1" width="80" height="24" strRow="1" endRow="23">
  -<fmtKwds>
    <fmtKwd kwd="TEXT" value="Select panel"/>
    <fmtKwd kwd="OVERLAY"/>
    <fmtKwd cond="60" kwd="PUTOVR"/>
    <fmtKwd kwd="PRINT"/>
    <fmtKwd kwd="CA03" value="03"/>
    <fmtKwd kwd="CF04" value="04"/>
    <fmtKwd kwd="HELP" value="29 'HELP key'"/>
    <fmtKwd kwd="CA01" value="29"/>
  </fmtKwds>
  -<inds>
    <ind name="*IN03" use="I"> </ind>
    <ind name="*IN04" use="I"> </ind>
    <ind name="*IN29" use="I"> </ind>
    <ind name="*IN60" use="O"> </ind>
    <ind name="*IN61" use="O"> </ind>
    <ind name="*IN62" use="O"> </ind>
    <ind name="*IN63" use="O"> </ind>
    <ind name="*IN64" use="O"> </ind>
  </inds>
  -<flds>
    -<fld name="SCRHXZ" type="A" use="O" X="2" Y="1" width="79">
      -<fldKwds>
        <fldKwd kwd="REFFLD" value="SCRHXZ XAZLD"/>
        <fldKwd kwd="TEXT" value="Screen headings"/>
      </fldKwds>
    </fld>
    -<fld name="PANEX1" type="A" use="O" X="2" Y="2" width="8">
      -<fldKwds>
        <fldKwd kwd="TEXT" value="Panel"/>
      </fldKwds>
    </fld>
  </flds>
</fmt>
```

UK: Mer 10:24 Melbourne: Mer 19:24 US Central: Mer 04:24 Toronto: Mer 05:24 GMT/UTC: Mer 09:24 Completato

Conversion des RPG en ROA

```
1 - v6 FRS - newlook
File Session Edit View Macros Tools Keyboard Window Help
Cvt RPG src to ROA:add handler (CVTRPGR0A)

Type choices, press Enter.

From file . . . . . QRPGLESRC Name, QRPGLESRC
  Library . . . . . _____ Name
From member . . . . . _____ Name, *ALL
To file . . . . . QROALESRC Name, *SAME, QROALESRC...
  Library . . . . . *SAME Name, *SAME
To member . . . . . *SAME Name, *SAME
Handler service program . . . . > LOOKROA Name, *SAME, LOOKROA
  Library . . . . . > LOOKROA Name, LOOKROA
Handler procedure . . . . . > HANDLER Name, HANDLER

F3=Exit F4=Prompt F5=Refresh F12=Cancel F13=How to use this display
F24=More keys

Bottom
ITA | OVR | 5,37 ? < >
```

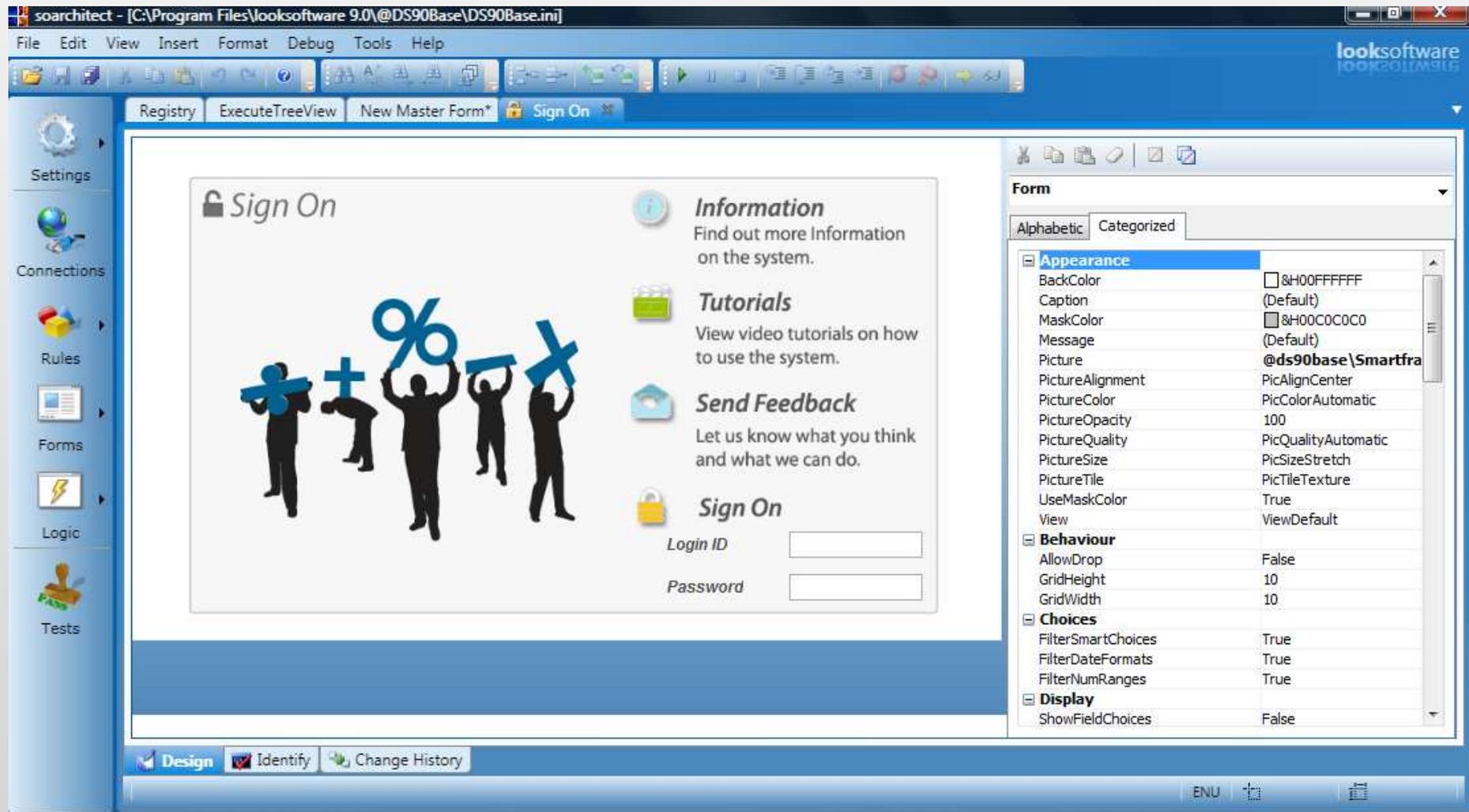
Conversion des RPG en ROA

```
Columns . . . : 6 100 Browse LOOKDEMO/QROALESRC
SEU==> YD1C02RG
FMT * *. 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8 ...+... 9 ...+... 0
***** Beginning of data *****
0001.00 *=====
0002.00 * Member/Customer Maint
0003.00 *
0006.00 *=====
0007.00 Hoption(*nodebugio)
0008.00 *
0009.00 Fyd1c02df cf e workstn
0010.00 F HANDLER('LOOKROA/LOOKROA(HANDLER)')
0011.00 F sfile(sf1:rrn1)
0012.00 F sfile(sf2:rrn2)
0013.00 F sfile(sf3:rrn3)
0014.00 F sfile(sf4:rrn4)
0015.00 F sfile(sf5:rrn5)
0016.00 F sfile(sf6:rrn6)
0017.00 F sfile(sf7:rrn7)
0018.00 F sfile(sf8:rrn8)
0019.00 F infds(info)
0020.00 Fyd1clf1 if e k disk rename(yd1cpr:lfr)
0021.00 Fyd1c uf a e k disk

F3=Exit F5=Refresh F9=Retrieve F10=Cursor F11=Toggle F12=Cancel
F16=Repeat find F24=More keys

(C) COPYRIGHT IBM CORP. 1981, 2007.
```

Le designer WYSIWYG...



... en remplacement de SDA

The screenshot displays the 'soarchitect' application window. The title bar shows the file path: [C:\Program Files\looksoftware 9.0\@RPGOA2\RPGOA.ini]. The menu bar includes File, Edit, View, Insert, Format, Debug, Tools, and Help. The toolbar contains various icons for file operations and development tools. The main workspace is divided into several sections:

- Left Sidebar:** Contains navigation icons for Settings, Connections, Rules, Forms, Logic, and Tests.
- Top Panel:** Displays 'Sales Order History' for 'Charleston Place' with details: Customer Name (Charleston Place), Customer Number (0424999), and Last Purchase (2009-12-23).
- Form Design Area:** Shows a 'Sales Order History' form with tabs for 'Sales Order History', 'Non Stock Orders', and 'Customer Management'. The 'Sales Order History' tab is active, showing a table with columns 'Vnd Order', 'Order No', and 'Description'. Below the table is a large yellow rectangular area.
- Right Panel:** A properties panel for the 'Sales' NavGroup. It has tabs for 'Alphabetic' and 'Categorized'. The 'Appearance' section is expanded, showing properties like BackColor, BackStyle, ButtonForeColor, ButtonSelectedColor, and ButtonBackColor. The 'Behaviour' section shows properties like AllowDrop, Enabled, GroupStart, and Visible. The 'Events' section lists events like OnClick, OnDbClick, OnDrop, OnMouseEnter, and OnMouseExit. The 'Misc' section shows (Name) as 'Sales'.

At the bottom of the window, there are buttons for 'Design' and 'Change History', and a status bar with 'ENU' and other icons.

Pourquoi utiliser le handler de **looksoftware**?

- Solution globale et complète
 - Modernisation du non OA à OA
 - Nouveau + ancien dans la même interfac (UI) et même session
 - Ecrans système + écrans applicatifs dans la même session
- Maximise le potentiel du code natif
 - Utilise RPG natif et DDS
- Handler générique multi-canal
 - Plus qu'un simple browser
 - Rich Windows client, RIA, thin, mobile,
 - Et Service Web
- Integration solide
 - Front-end, back-end, + nouveau

looksoftware™ et

IBM Open Access for RPG

Contact :

itheis

Pascal BLANDIN

pascal.blandin@itheis.com

06 15 16 39 95

