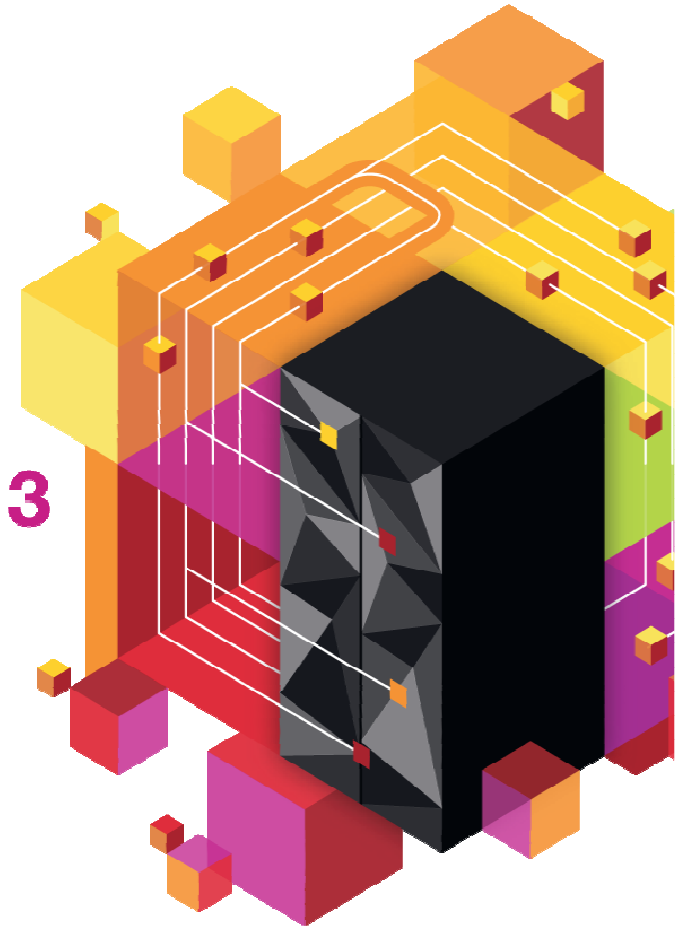


# Université du Mainframe 2013

4-5 avril





# Modernisation des ateliers de développement COBOL\*/PACBASE

*\*Le COBOL est un langage du passé, du présent et d'avenir.....  
COBOL acronyme de COmmon Business Oriented Language*

**Dominique PROVOST**



Global Business Services  
Expert Atelier de développement  
AMS- GBS

[Dominique.provost@fr.ibm.com](mailto:Dominique.provost@fr.ibm.com)

Tel : 01 49 08 68 56

GSM : 06 73 98 26 27

## Université du Mainframe 2013

4-5 avril



# Sommaire

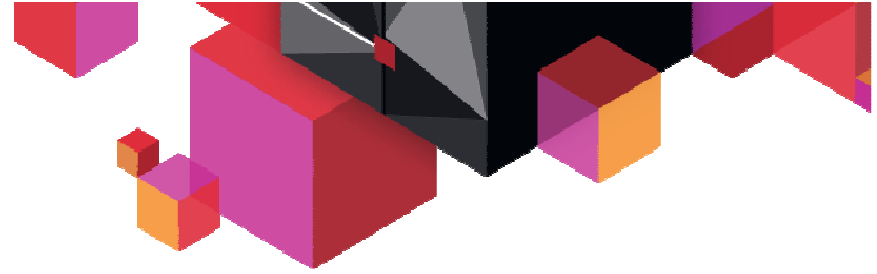
**1. Nos constats**

**2. Principes clés des produits de modernisation  
COBOL/PACBASE (RDZ(RDP)/RPP/RTC)**

**3. Notre conviction, notre vision**

**4. Nos retours d'expériences**

**5. Exemple de conclusions d'un démonstrateur**





## Nos constats

- Ateliers de développement Cobol vieillissants, peu conviviaux, mal adaptés aux stations graphiques
- Univers de développement Cobol / Pacbase (Zos, Unix) et Java ou autre technologie toujours trop cloisonnés

- Difficile d'intégrer dans l'état de nouveaux talents dans les équipes désormais vieillissantes
- Trop de rupture d'acteur dans le développement

- Plan de convergence PACBASE
  - **Migration des développements Pacbase vers les produits de la SDP avant 2015.**
- IBM se doit d'accompagner ses clients pour maîtriser cette transition qui touche le cœur de métier



☞ offre au développeur une plateforme ergonomique, industrialisée, avec une vision intégrée des différents environnements et technologies



✓ Gain de productivité

☞ répond au risque sur les ressources en redynamisant ces activités et en mettant à disposition des clients d'IBM un vivier de ressources polyvalentes et motivées

✓ Accès facilité des ressources Java au développement Cobol

☞ fournit à nos clients PACBASE une solution immédiate, compatible avec la cible, qui fluidifie le passage à la nouvelle organisation du développement



✓ Migration au fil de l'eau et garantie de maintenance du patrimoine



# Sommaire

**1. Nos constats**

**2. Principes clés des produits de modernisation  
COBOL/PACBASE (RDZ/RDP/RPP/RTC)**

**3. Notre conviction, notre vision**

**4. Nos retours d'expériences**

**5. Exemple de conclusions d'un démonstrateur**





# Principes clés de Rational Developer for System z ou for Power

Un des produits de la plateforme JAZZ

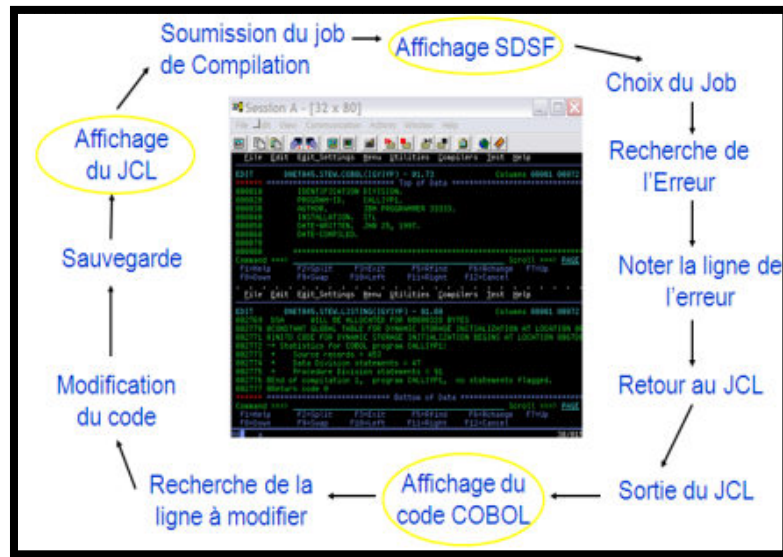
Développement sous RDZ (ou RDP)

- Ergonomie fortement améliorée,
- Homogénéisation des plateformes de développement COBOL/PACBASE / JAVA
- Nouvelles fonctionnalités n'existant pas sous ISPF ou sous unix

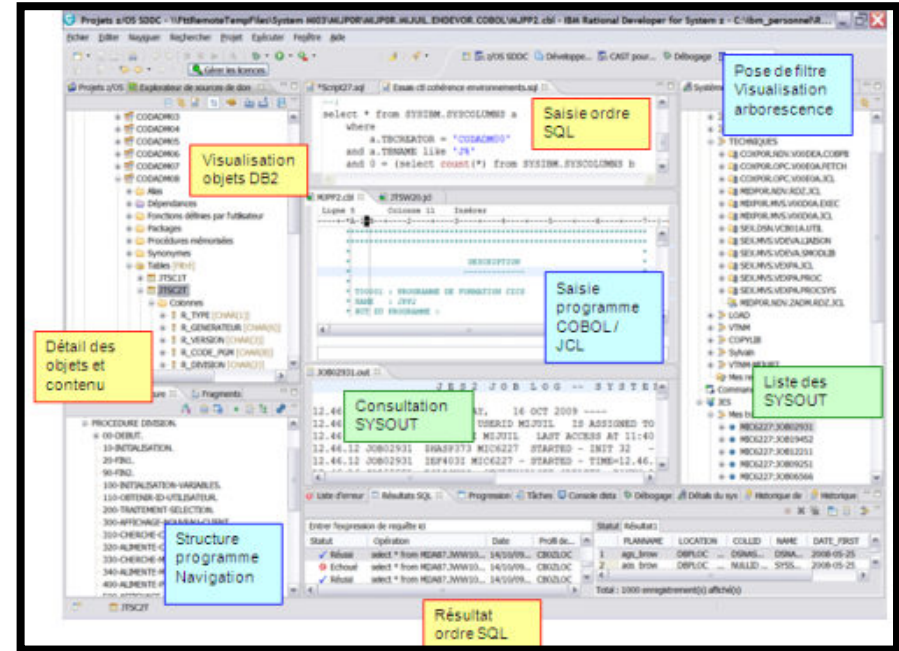
## ▪ Confort dans le travail

## ▪ Gain sur la qualité & la Productivité

- (10-15% de productivité sur les phases de coding et test unitaire)



Développement traditionnel sous ISPF



Développement sous RDZ



# Principes clés de Rational Developer for System z ou for Power

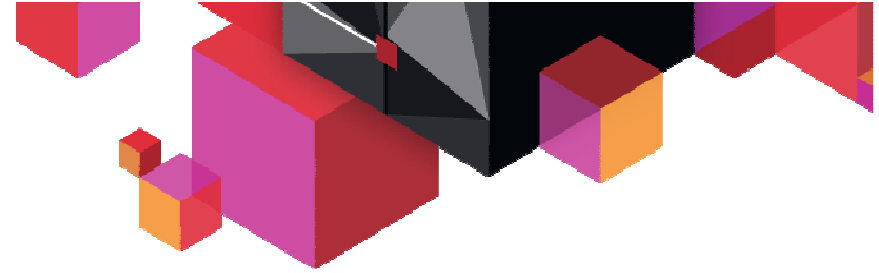
Différentiateurs – Fonctions de productivité

De multiples fonctions d'assistance et d'analyses, pour permettre aux développeurs d'être plus productif

The screenshot displays the IBM Rational Developer for System z interface with several key features highlighted:

- Code Completion:** A context menu is open over the code, showing suggestions for variables like PATMSTR-REC, PARA-NAME, and PARA-POINTER. The word "Complétion" is overlaid on the bottom left.
- Perform Hierarchy:** A red-bordered window titled "Perform Hierarchy" is open, showing a list of program elements such as 300-EXIT, 350-CHECK-LAB-TABLE, and 400-NUMERIC-RANGE-EDITS. The text "Perform Hierarchy" is overlaid on this window.
- Structure View:** A "Structure" window on the right shows a tree view of the project's components, including files and folders like "Définitions globales" and "Fichiers".
- Code Editor:** The main editor shows COBOL code with line numbers and column indicators. A "Remote Systems" window is also visible in the background.





## Principes clés de Rational Developer for System z ou for Power

Différentiateurs – Fonctions de productivité

L'outil de visualisation de contrôle d'exécution permet de mieux maintenir les programmes en place depuis longtemps

The screenshot displays the Rational Developer for System z interface. The top pane shows source code for a program named '300-CALC-EQUIP-COSTS'. The code includes several statements such as 'MOVE "300-CALC-EQUIP-COSTS" TO PARA-NAME.', 'READ PATMSTR INTO PATIENT-MASTER-REC.', and 'CALL 'CLCLEBST' USING CALC-COSTS-REC, CALC-CALL-RET-CODE.'. A context menu is open over the code, with 'Program Control Flow' selected. The bottom pane shows a call graph (Diagramme des appels) with nodes representing program elements like '000-HOUSEKEEPING', '050-LOAD-EQUIPMENT-TABLE', '100-MAINLINE', '200-SEARCH-RTN', '300-CALC-EQUIP-C', '900-CLEANUP', and '1000-ABEND-RTN'. Arrows indicate the flow of control between these elements.

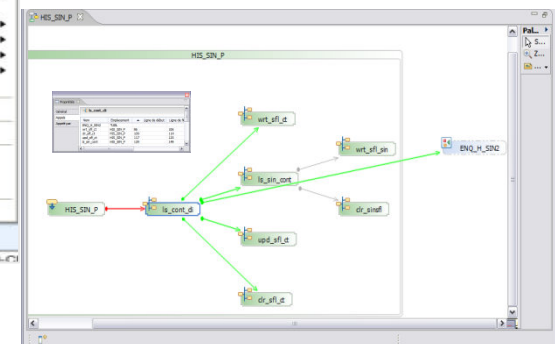
Représentation graphique de la structure d'une application

1) Diagramme des appels

- appels des sous-routines, des procédures et des programmes

2) Diagramme de la structure d'un programme

- modules liés dans un programme ou un programme de service
- relations entre programmes et programmes de service







# Principes clés de Rational Developer for System z ou for Power

Différentiateurs – Fonctions de productivité

Des outils d'identification du code non atteignable et de couverture de code permettent de faciliter la maintenance

Code Coverage Report

Code Coverage Summary

Code coverage report, generated Jun 13, 2012 11:43:24 PM

Element	Coverage	Covered Lines	Total
SAM3	50%	107	
SAM3	46%	72	
USER170.TEST.SYSDEBUG(SAM3).cob	46%	72	
SAM3()	46%	72	
SAM4	61%	35	
USER170.TEST.SYSDEBUG(SAM4).cob	61%	35	
SAM4()	61%	35	

Filter view

Show all Ctrl+W

- Source
- Preprocessor Statements
- Refactor
- Show In
- Open Declaration F3
- Open Perform Hierarchy
- Occurrences in Compilation Unit
- View

Comment Ctrl+/

Uncomment Ctrl+}

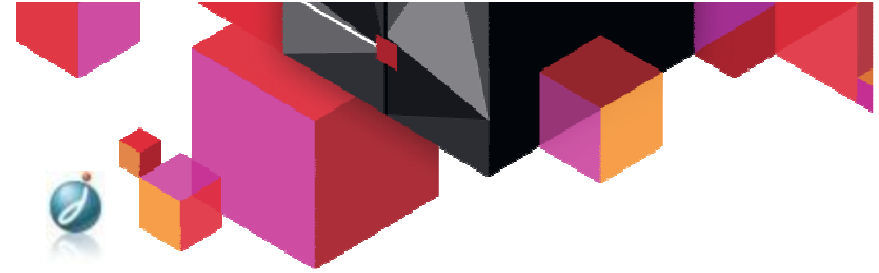
Uppercase all Alt+Shift+K

Hex edit line Alt+Shift+H

Identify Unreachable Code

```
*SAMOS1.cbl
Line 304 Column 1 Insert 1 change
-----+*1-1-B-----2-----3-----4-----5-----6-----7-|-----+
210-PROCESS-CUSTFILE-RECORD.
PERFORM 730-READ-CUSTOMER-FILE.
IF WS-CUST-FILE-EOF = 'Y'
GO TO 210-EXIT.
IF CUST-RECORD-TYPE = 'C'
GO TO 300-EXIT
ADD +1 TO NUM-CUSTOMER-RECS
ELSE
* SUBROUTINE SAMOS2 WILL COLLECT CUSTOMER STATISTICS
CALL 'SAMOS2' USING CUST-REC,
CUSTOMER-BALANCE-STATS
MOVE CUST-ID TO RPT-CUST-ID
MOVE CUST-NAME TO RPT-CUST-NAME
MOVE CUST-OCCUPATION TO RPT-CUST-OCCUPATION
MOVE CUST-ACCT-BALANCE TO RPT-CUST-ACCT-BALANCE
MOVE CUST-ORDERS-YTD TO RPT-CUST-ORDERS-YTD
WRITE REPORT-RECORD FROM RPT-DETAIL AFTER 1
ADD +1 TO NUM-DETAIL-LINES.
IF CUST-RECORD-TYPE = 'P'
ADD +1 TO NUM-PRODUCT-RECS
* SUBROUTINE SAMOS3 WILL COLLECT PRODUCT STATISTICS
CALL 'SAMOS3' USING CUST-REC,
PRODUCT-STATS
ELSE
ADD +1 TO NUM-TOTALS-REQUESTS
GO TO 210-EXIT
ADD +1 TO NUM-TRANSACTIONS.
WRITE REPORT-RECORD FROM RPT-SPACES AFTER 1.
210-EXIT.
EXIT.

300-PROCESS-TOTALS-TRAN.
ADD +1 TO NUM-TOTALS-REQUESTS .
ADD +1 TO NUM-TRANSACTIONS.
WRITE REPORT-RECORD FROM RPT-SPACES AFTER 1.
WRITE REPORT-RECORD FROM RPT-TOTALS-HDR1.
WRITE REPORT-RECORD FROM RPT-TOTALS-HDR2.
GO TO 400-PROCESS-ABEND-TRAN
IF NUM-PRINT-COMPLETED > 0
MOVE SPACES TO RPT-TOTALS-DETAIL
MOVE 'Acct Balance: ' TO RPT-TOTALS-TYPE
MOVE ' Totals: ' TO RPT-TOTALS-TYPE
```



## Principes clés de Rational Developer for System z ou for Power

Un des produits de la plateforme JAZZ

Développement sous RDZ (ou RDP)

Ergonomie fortement améliorée,

**Homogénéisation des plateformes de développement COBOL/PACBASE / JAVA**

Nouvelles fonctionnalités n'existant pas sous ISPF (ou Unix)

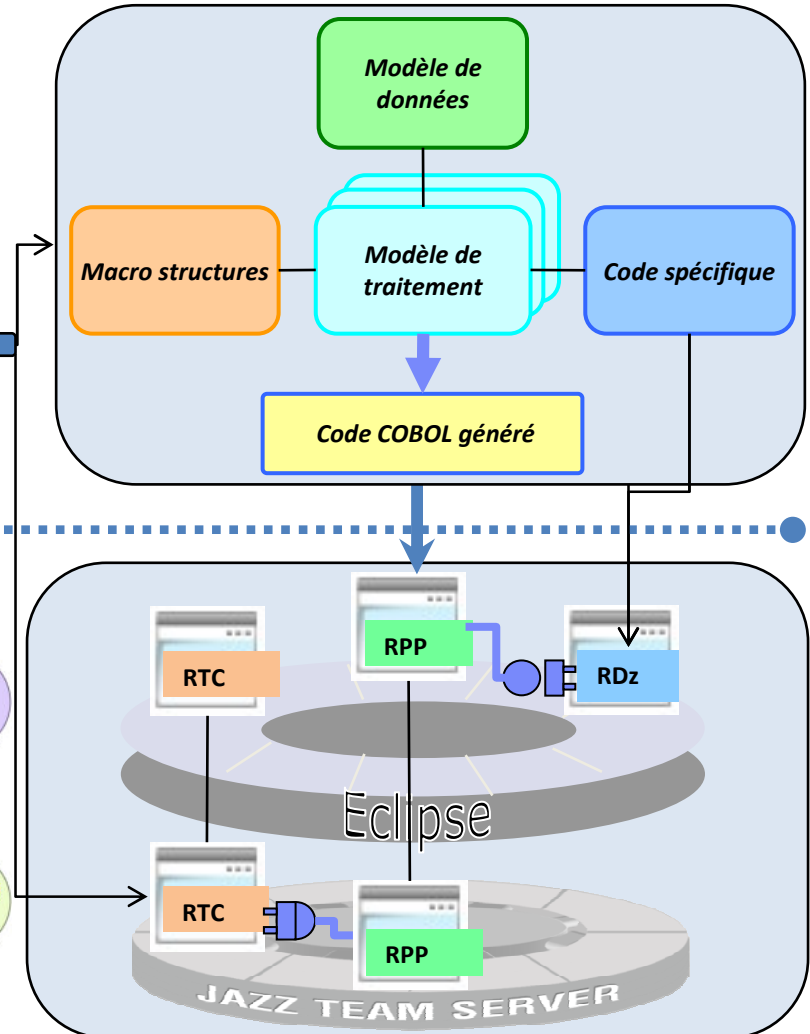
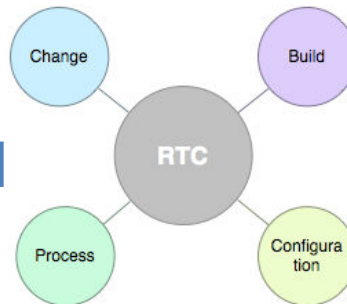
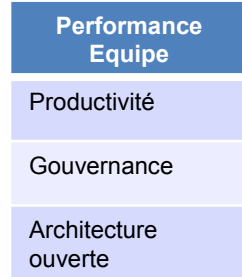
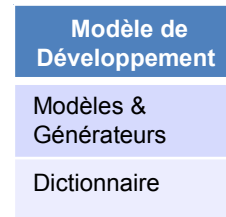
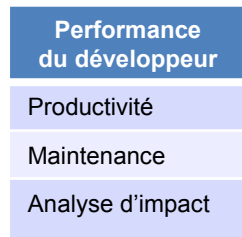
### ▪ Exemple de travaux d'intégration et de modernisation

The screenshot displays the Rational Developer for System z (RDZ) interface. On the left, a Java file named 'DestInfoBSP.java' is open, showing code for handling VHDI messages. The main editor window shows a COBOL program with several lines of code, including 'MOVE X-MESS TO TX-STORAGE', 'MOVE ZERO TO W-LIERR', 'INITIALIZE MD00.', and 'MOVE MM01-NOTRM TO MD02-NOTRM'. The interface includes a 'Débogage' (Debug) toolbar, a 'Variables' window showing 'MD02-NOTRM' and 'MM01-NOTRM', and a 'Points d'arrêt' (Breakpoints) window with four breakpoints set on lines 961, 962, 963, and 972 of the COBOL program.



# Rational Programming Pattern (RPP) Rational Team Concert (RTC)

- Challenges du plan de convergence PACBASE
  - Offrir un moyen de gérer les applications PACBASE ou à développer les nouvelles,
  - Réduire les coûts de migration,
  - Offrir la possibilité d'adopter des pratiques et des outils communs pour PACBASE ou non PACBASE de développement d'applications

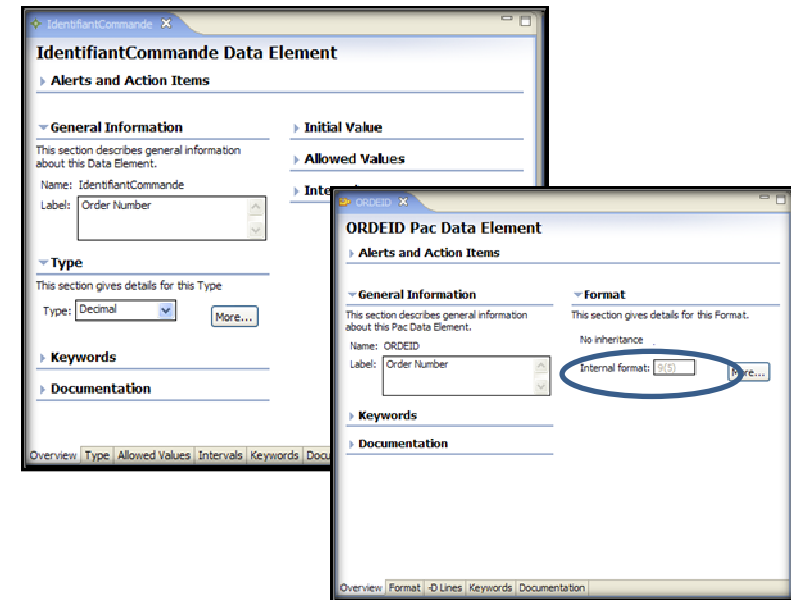




## Principes clés de Rational Programming Pattern (RPP)

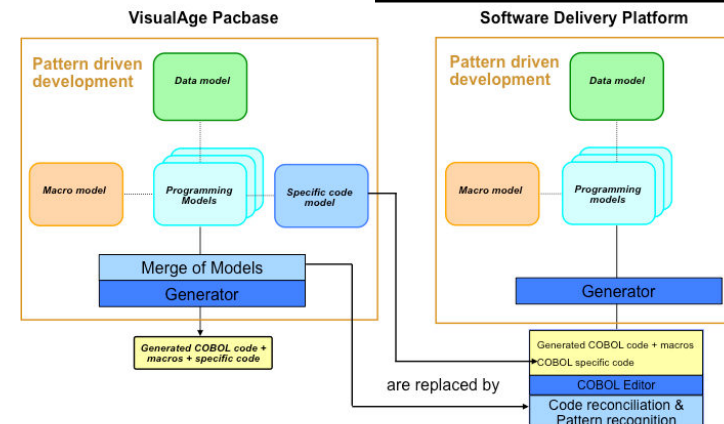


- Un des produits de la plateforme JAZZ
- RPP est un outillage de MDD (Model Driven Développement) à plusieurs facettes
- Ses caractéristiques
- Des modèles « pattern »,
  - basé sur un noyau « kernel », indépendant de la cible (agnostique du langage).
  - Sur ce noyau, définition de facettes précisant la cible
- Des générateurs de code
- Moteur de réconciliation de code




### RPP et le plan de convergence PACBASE

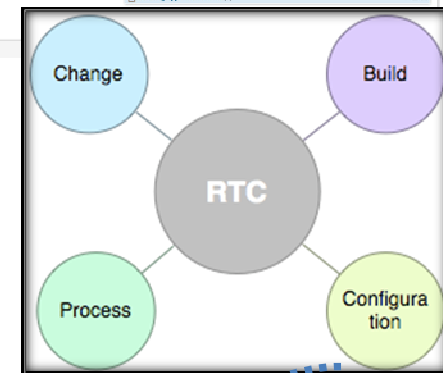
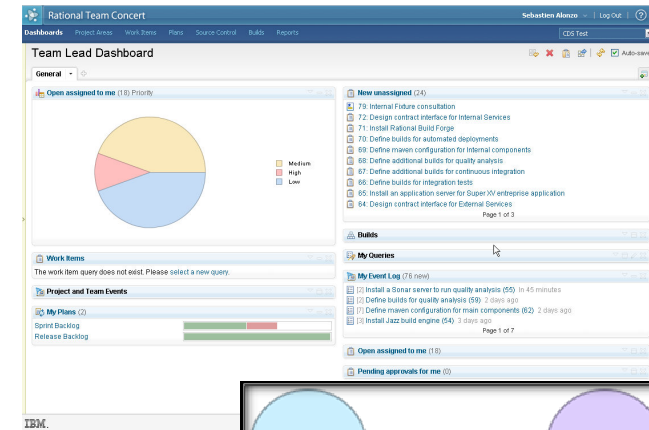
- Outil non dédié avec PACBASE
- PACBASE constitue une des facettes du produit RPP





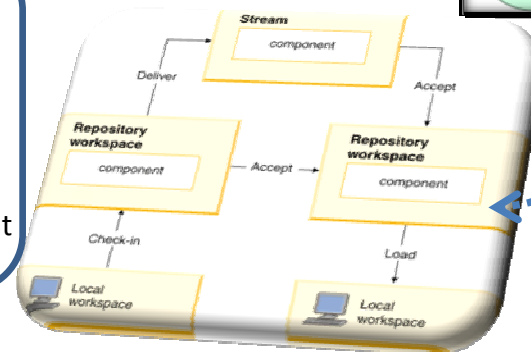
## Principes clés de Rational Team Concert

- Un des produits de la plateforme JAZZ 
- Outil ayant de multiples fonctions
- Outil de développement collaboratif supportant les méthodes agiles et l'intégration continue
  - Planification agile
  - Gestion de WorkItems (tâches, évolutions, anomalies ...)
  - Gestion de Configuration par composant
  - Système de Build automatisé (intégration continue)
  - Production de nombreux rapports
- Gestion de configuration (GCL) parfaitement adaptée aux développements parallèles
- Facteur de productivité important pour les équipes de toutes tailles



### ▪ RTC et le plan de convergence PACBASE

- Outil non dédié à PACBASE
- Utilisation de la fonction de GCL dans le cadre du plan de convergence PACBASE pour assurer les fonctions de versionning et références croisées des modèles PACBASE
- Meilleure collaboration des équipes de développement PACBASE





# Sommaire

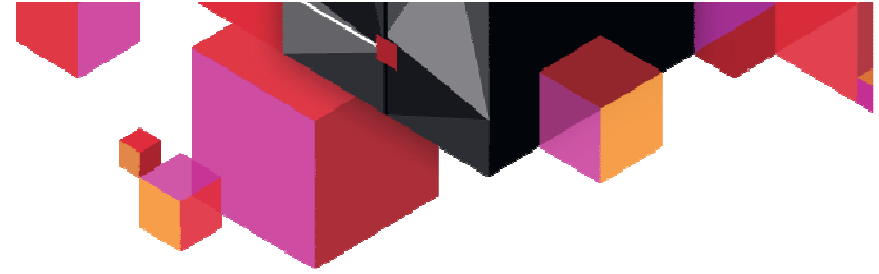
**1. Nos constats**

**2. Principes clés des produits de modernisation  
COBOL/PACBASE (RDZ(RDP)/RPP/RTC)**

**3. Notre conviction, notre vision**

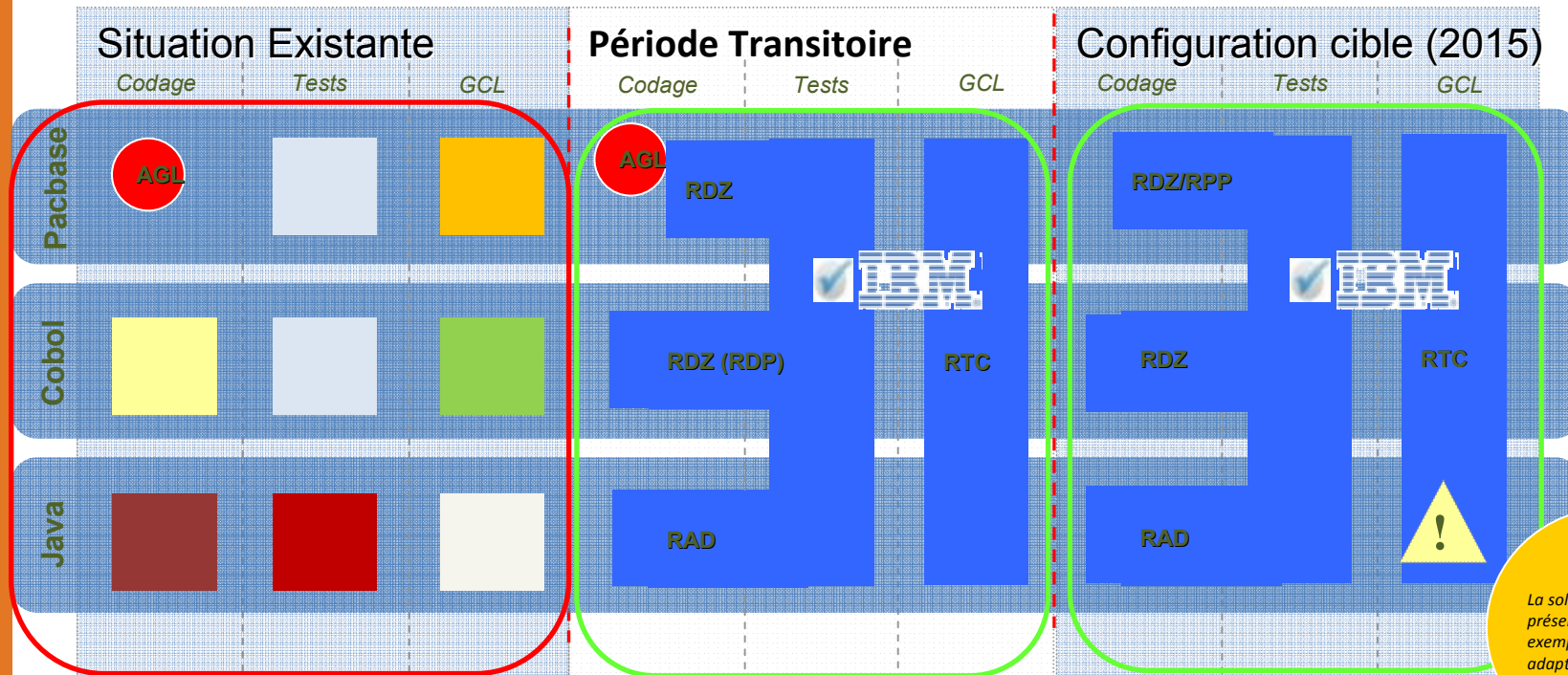
**4. Nos retours d'expériences**

**5. Exemple de conclusions d'un démonstrateur**





## Notre conviction, notre vision



Multitude d'outils ne favorisant pas la communication et l'échange d'information

Rationalisation et Homogénéisation des outils

La solution de principe présentée ici en exemple doit être adaptée à chaque contexte client (GCL)

Dès à présent

1. Modernisation des ateliers de COBOL (Système Zos ou Power (UNIX/AS400)), basé la brique RDZ(RDP), indépendamment du choix du plan de convergence PACBASE
2. Etude de la migration du patrimoine PACBASE (Patrimoine à migrer/ Organisation cible du référentiel/ Outillage existant dans la cible )
3. Usage de la plateforme Jazz (Produits RSM/RAD/RTC) aux développements Java

Une plateforme de développement intégrée basée sur







# Notre conviction, notre vision



**RDz**  
Developper  
For Z/OS

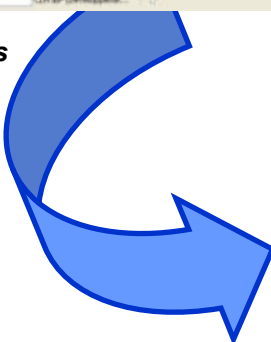
**RDP**  
Developer  
For Power

**PACBASE**

Modernisation  
des ateliers COBOL

La mise en œuvre de RDz(RDp) permet  
de poser la première brique  
logicielle RDZ des outils cible  
PACBASE sur l'ensemble du  
patrimoine COBOL/PACBASE

Ajout fonctionnalités



**RDz**  
Developper  
For Z/OS

**RDP**  
Developer  
For Power

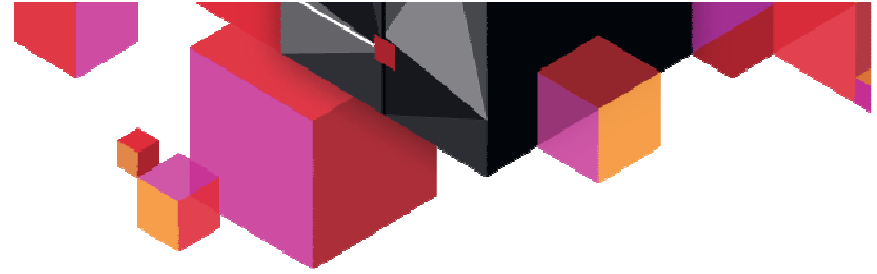
**RPP**  
Programming  
Pattern

**RTC**  
Team  
Concert

Cible du  
Plan de Convergence

Programme	SD	Extensio	OARMOF	Bloc	T	R	S	U	RE	SE	X	Unité	P	S	A	T	O	D	N	E	
IM	TM		LSFOU	O	R																
MD	TM		LSFOU	O	R																
TT	22		2SFOU	O	R																

Transformation vers  
le plan de  
convergence

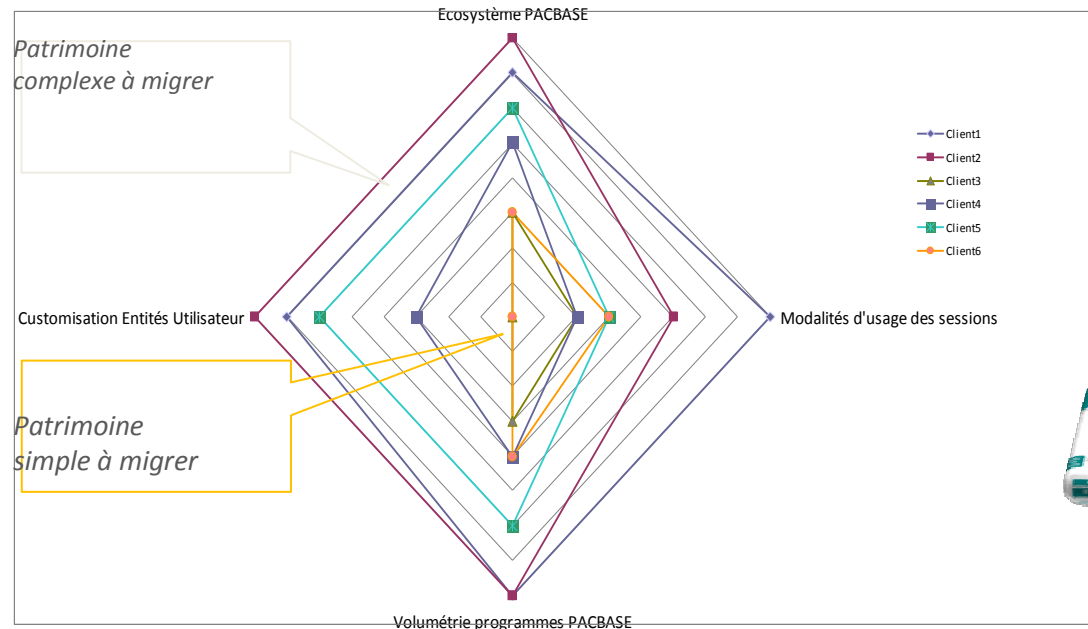


## Notre conviction, notre vision

### Contexte : Complexité de la transformation du patrimoine PACBASE

- S'appuyant sur l'expérience acquise durant l'année 2012 à travers de nombreuses missions d'évaluations menées chez d'autres clients, l'équipe IBM GBS a montré que la mise en œuvre de la transformation du patrimoine PACBASE était propre à chaque client.
- La charge de mise en œuvre de ce plan dépend fortement de 4 axes :
  - ✓ De l'écosystème à adapter,
  - ✓ De la volumétrie des programmes PACBASE à migrer,
  - ✓ De la modalité d'usage des sessions PACBASE,
  - ✓ De l'extension d'usage du modèle PACBASE (utilisation des entités « utilisateur »)

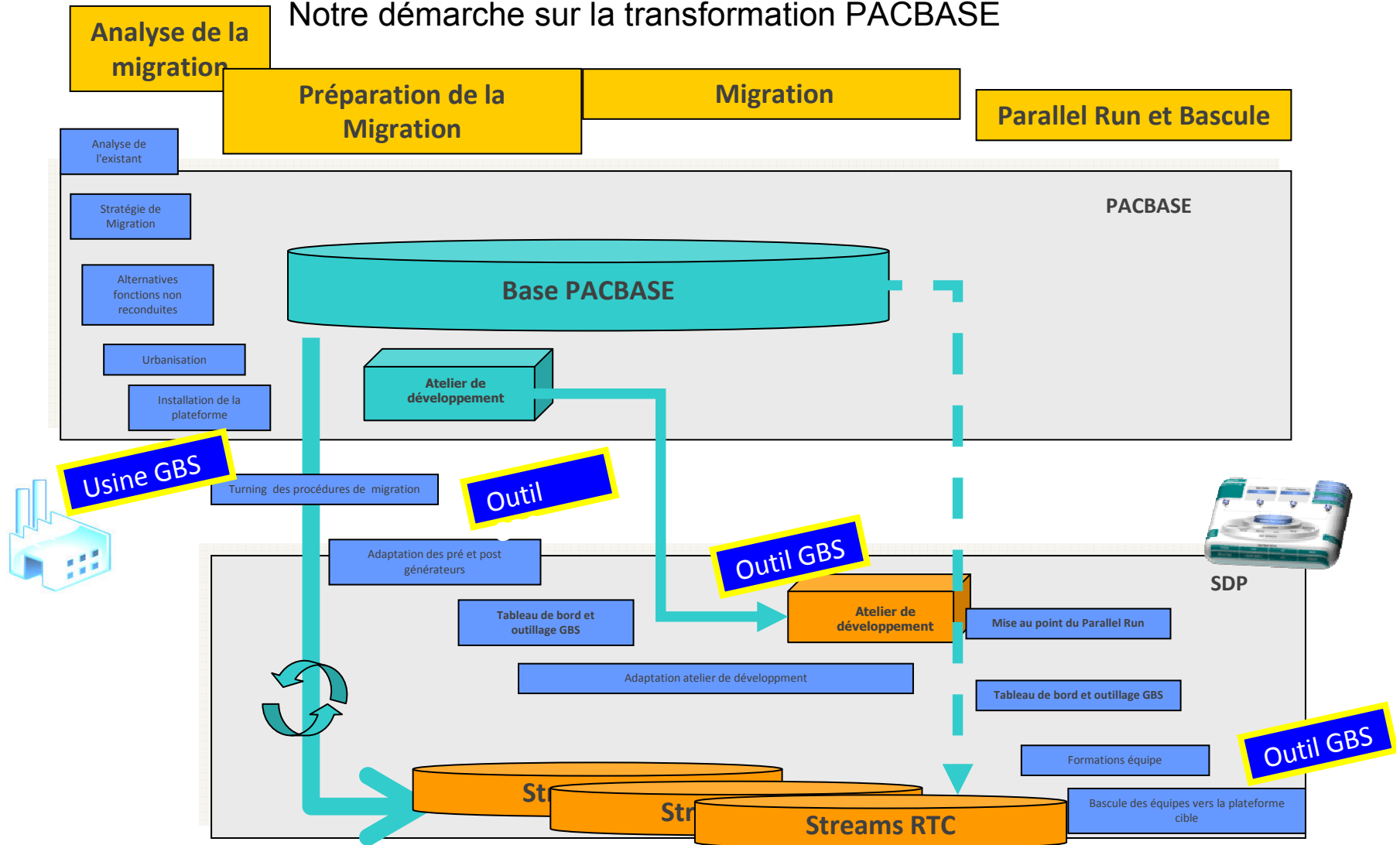
L'analyse ci-dessous, montre les écarts possibles en terme de complexité des patrimoine PACBASE à migrer





# Notre conviction, notre vision

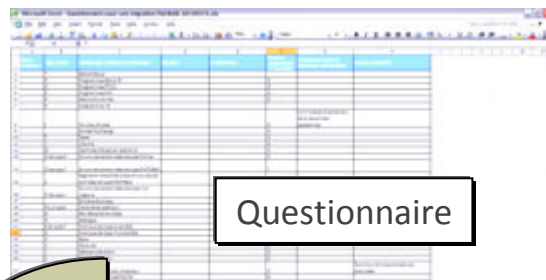
## Notre démarche sur la transformation PACBASE



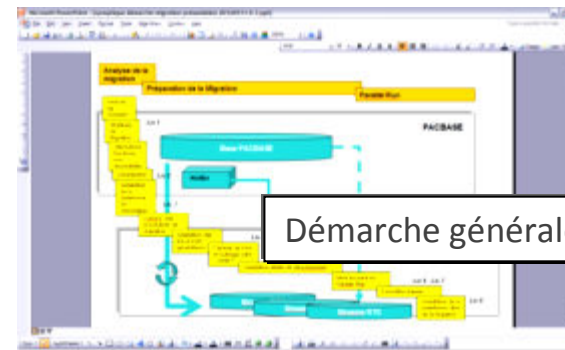


Notre conviction, notre vision

Le Quick Sizing, macro-chiffrage de la transformation PACBASE



Questionnaire

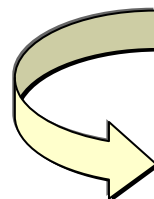


Démarche générale



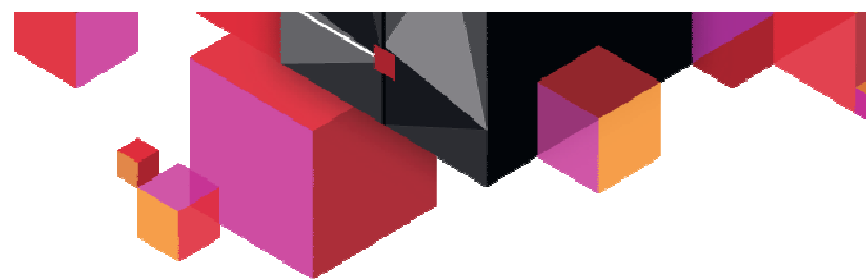
Inducteurs de charge

	A	B
1	Nombre de base	254
2	Nombre de bibliothèques total	84
3	Nombre de bibliothèques actives	84
4	Nombre de projets applicatifs et techniques	6
5	Nombre de programmes (batch + écrans) + clients + serveurs	7800
6	Nombre de sessions actives	10
7	Nombre de type d'entités non reprises ou non révisitées	10
8	Nombre de cas d'utilisation des constantes PACBASE	10



Chiffrage et planification

Module	Inducteur de charge	Prévision	Charge actuelle	Nombre de charge	Prévision	Prévision	Prévision	
Ion						1287	449	
Analyse de la migration							187	731





# Sommaire

**1. Nos constats**

**2. Principes clés des produits de modernisation  
COBOL/PACBASE (RDZ(RDP)/RPP/RTC)**

**3. Notre conviction, notre vision**

**4. Nos retours d'expériences**

**5. Exemple de conclusions d'un démonstrateur**





## Retours d'expérience et missions en cours chez quelques clients

### ■ Grande banque française

#### ■ **Entité Banque de détail, modernisation du poste de travail**

#### ■ *Modernisation de la plateforme COBOL dans un monde Zos*

##### ■ *Version V1 Décembre 2010- Juin 2011*

- Démonstrateur RDZ
- Intégration des outils Compuware
- Outils clients

##### ■ *Version V2.1/V2.2/V2.3 Juin 2012 à ce jour*

- Customisation RDZ
- Formation Client et Usine de développement au Maroc
- Déploiement de RDZ (1200 postes de travail)

#### ■ **Entité Banque de détail à l'international migration PACBASE**

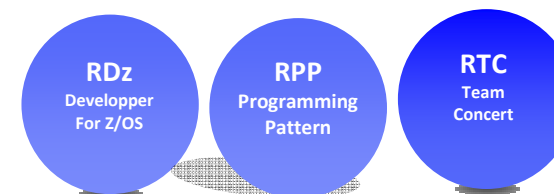
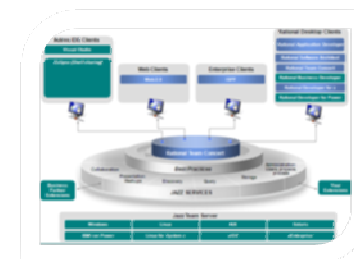
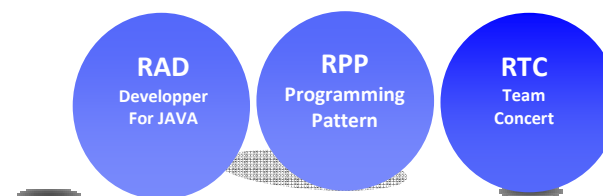
#### ■ *Migration PACBASE, hébergeant le logiciel « maison » dans un environnement Cobol Microfocus/UNIX*

##### ■ *Réalisation du « Quick Analysis » du patrimoine PACBASE*

- Août-Septembre 2012

##### ■ *Réalisation Modernisation et Plan de convergence PACBASE*

- Déploiement RDp d'ici fin 2013
- Migration PACBASE 2eme semestre 2014





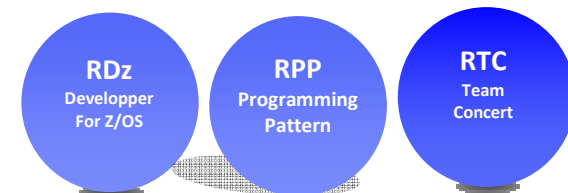
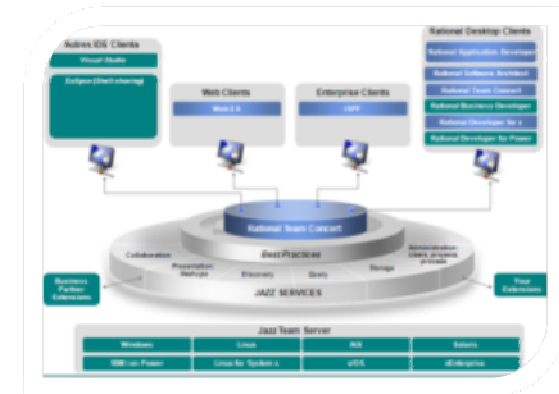
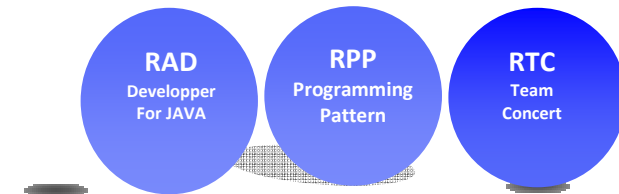
## Retours d'expérience et missions en cours chez quelques clients

### ■ Grand groupe bancaire français

- **Etude de modernisation de la plateforme de développement, projet New Software Factory**
- **Modernisation de la plateforme de développement JAVA/COBOL/PACBASE/WID/.net**
- **Octobre 2011- Avril 2012**
  - Démonstrateur RDZ/RAD/RTC/RPP
    - Intégration des outils Compuware
    - Outils clients
    - Impliquant des équipes clientes
  - Objectif du projet
    - Dossier d'étude définissant la trajectoire de modernisation en tenant compte du plan de convergence PACBASE mais sans l'imposer. Dossier intégrant les différents coûts des paliers de modernisation ( calcul de ROI), présenté au niveau des directions et validé
    - Début de réalisation prévu Avril r 2013

### ■ **Projet « un pour un »**

- **Migration PACBASE dans un monde Zos**
- **Etude de migration PACBASE**
- **Octobre 2011 – Juin 2013**
  - En stand by
  - Reprise en Juin 2013 ( avec la version V9)



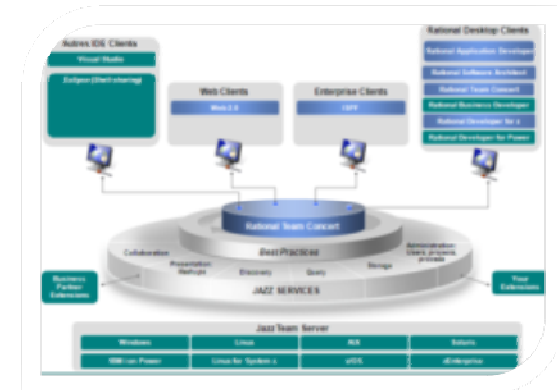
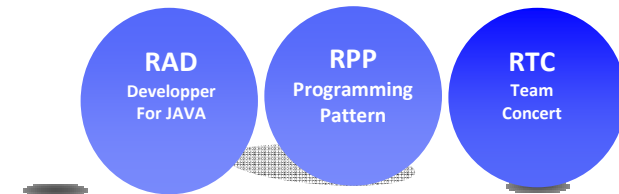




## Retours d'expérience et missions en cours chez quelques clients

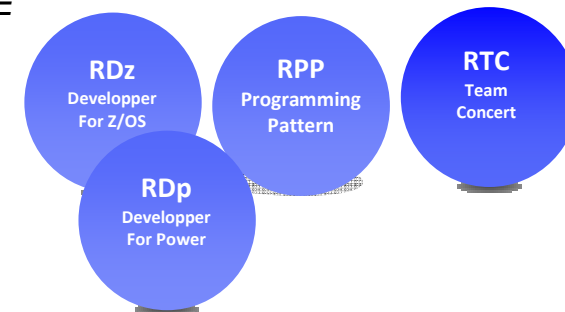
### ■ Grand mutualiste français assurance et banque

- *Réalisation d'un démonstrateur du plan de convergence PACBASE*
- *Caractéristiques du contexte PACBASE*
  - Forte volumétrie du patrimoine applicatif
  - Gestion du changement autour de DSMS
  - A terme remplacement Endeavor par RTC
- *Utilisation de la versions 8.5*
- *Avril 2012- Novembre 2012*
  - Démonstration de l'industrialisation et sécurisation de la chaîne de migration des données (réalisé)
  - Utilisation de la plateforme cible
  - Réalisation de plus 170 vidéos
- *Janvier 2013-Juin 2013*
  - Accompagnement GBS vers la SDP



### ■ Grand mutualiste français assurance et prévoyance

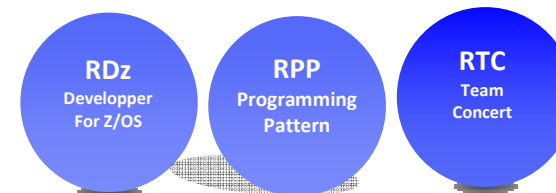
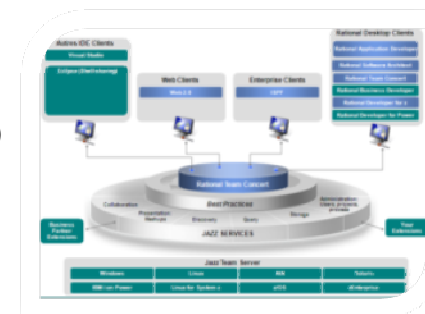
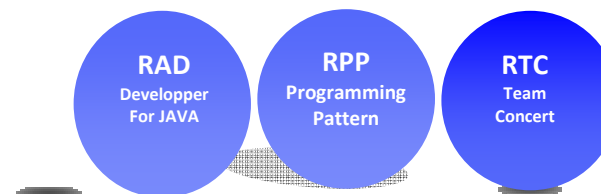
- *Août 2012*
  - *Proposition sur la transformation du Plan de convergence PACBASE*
- *Janvier –Février 2013*
  - *Etude de modernisation des ateliers COBOL intégrant le plan de convergence PACBASE*





## Retours d'expérience et missions en cours chez quelques clients

- **Filiale commune à deux banques françaises dédiée à la gestion des titres**
  - *Aout- Septembre 2012*  
Etude du « Quick Analysis » du patrimoine PACBASE
  - *Février –Juin 2013*  
Réalisation d'un démonstrateur
- **Banque Centrale Européenne**
  - Novembre –Décembre 2012  
Etude du « Quick Analysis » du patrimoine PACBASE
  - Avril 2013- Décembre 2013  
Mise en œuvre des travaux préparatoire (architecture RTC /Déploiement RDZ)
- **Grande banque de détail française**  
Assistance sur la démarche de migration
- **Organisme public français**  
Assistance aux études de coûts de l'adaptation de l'écosystème PACBASE
- **Grande banque Espagnole**  
Expertise sur la démarche de migration
- **Groupe financier Américain**  
Expertise sur la démarche de migration





# Sommaire

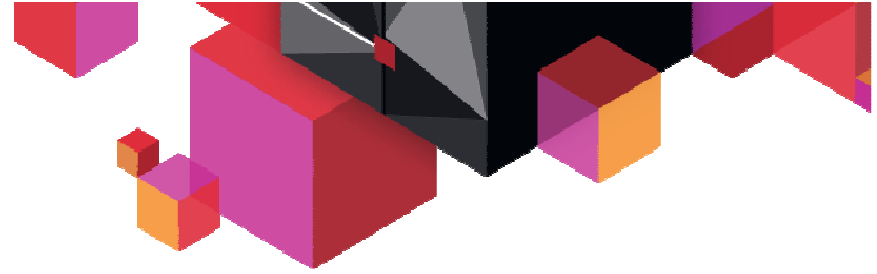
**1. Nos constats**

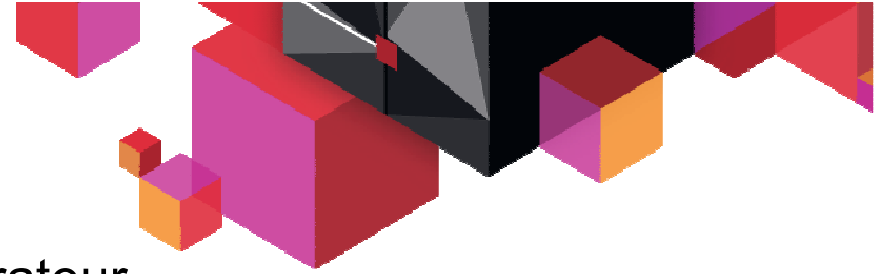
**2. Principes clés des produits de modernisation  
COBOL/PACBASE (RDZ(RDP)/RPP/RTC)**

**3. Notre conviction, notre vision**

**4. Nos retours d'expériences**

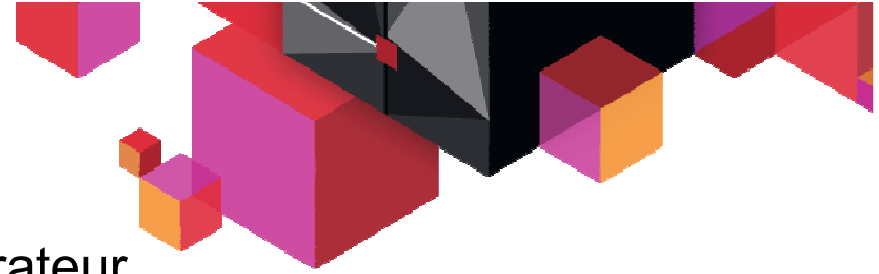
**5. Exemple de conclusions d'un démonstrateur**





## Exemple de conclusions d'un démonstrateur

- **Eléments démontrés**
  - *La migration des données sur le volume réel a été réalisée.*
  - *L'usage de la station permet au développeur de réaliser son activité en mode maintenance .*
  - *Le mécanisme « sparse loading » fonctionne*
  - *La gestion du changement et la gestion du report des changement sont respectivement portées par les work-items et la promotion des work-items, mécanismes standards du produit.*
  - *La gestion build RTC EE permet d'envisager de reproduire l'usage des environnements existants via l'ASSET GBS.*
  - *Les informations du modèle PACBASE sont bien reportées dans les outils cibles*
  - *Possibilité de customiser la station de travail*
  - *Adaptation possible des outils analysés ( avec la disponibilité des API batch MAF)*
- **Evolutions/corrections attendues au niveau produit**
  - *Evolutions de la chaine de la migration (8.5.1 et 8.5.Next)*
  - *Compléter les informations disponibles dans la vision référentiel serveur pour une activité d'analyste (8.5.1)*
  - *Mise en œuvre des scanners en vu de remplacer l'outil Méta-modèle, outil G2S (8.5.Next.Next)*
  - *Contrôler la qualité des programmes en mode serveur (prévu avec la 8.5.Next)*
  - *Démontrer le fonctionnement en mode batch des API MAF (prévu avec la 8.5.Next)*
  - *Disponibilité export csv (prévu avec 8.5.Next)*
- **Etudes complémentaires**
  - *Qualifier la procédure MIBJ*
  - *Identifier le mécanisme de migration de l'historique des changements (DSMS)*
  - *Clarifier le mode de développement de nouveau programme suite au mécanisme de « sparse loading »*
  - *Etudier la granularité des work-items sur tout le cycle de développement*
  - *Etudier l'adaptation du processus de livraison*



## Exemple de conclusions d'un démonstrateur



Migration

Traiter la volumétrie cliente, avoir des temps de migration, reporting et visibilité sur les rejets, reprise d'un historique DSMS lié à la version en cours, spécificité client



Études d'impact

Historique des modifications, consultation du référentiel et références croisées\*



Réalisation fabrication

Gestion d'une demande de changement, éditeur entités modèle, éditeur Cobol, le build unitaire contrôle qualité\*, génération de copy (outil spécifique client, DC\$ Requête), génération du JCL (spécifique client), traçabilité et customisation IDE



Tests unitaires

Procéder aux tests unitaires du développeur



Reports

Report de composants vers l'environnement d'assemblage (Usage DSMS)



Livraison

Création de liste de composants à builder, livraison de version



Interfaces

Commandes de descriptif des entités du modèle, export csv, accessibilité aux informations du modèle cible (RPP/RTC)\*



Opérations transverses

Historisation, Aide en ligne, Thésaurus, Gestion et droits utilisateurs



Fin du document