



Cómo desarrollar y probar aplicaciones mainframe sin un mainframe

Alejandro León

Especialista en Soluciones de
Modernización de la Empresa



Cuatro barreras clave impiden un óptimo retorno de la inversión

Décadas de inversiones en aplicaciones



“No entendemos el esfuerzo, el riesgo y el impacto de modernizar nuestras aplicaciones ‘legacy’”

Islas de skills, lenguajes y plataformas



“Nuestros skills gaps siguen creciendo. ¿Cómo podemos permanecer actualizados de todos los cambios en los lenguajes y en las tecnologías?”

Equipos mal integrados



“Necesitamos que nuestros equipos colaboren a través de las distintas plataformas, lenguajes y entornos.”

Ineficiencia de la Infraestructura



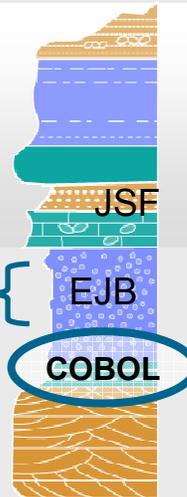
“Necesitamos alguna manera de mejorar nuestra eficiencia y liberar capacidad para poder tener más potencia.”

Las compañías quieren...

*Comprender el impacto y coste de los cambios en sus sistemas de TI.
Unificar, simplificar plataformas y entornos. Mejorar la mantenibilidad.
Hacer que diferentes equipos dispersos geográficamente colaboren.*

¿Qué es lo que se va afectado si cambio esta copy COBOL?

¿Cómo podría mejorar la adopción de los recién llegados?



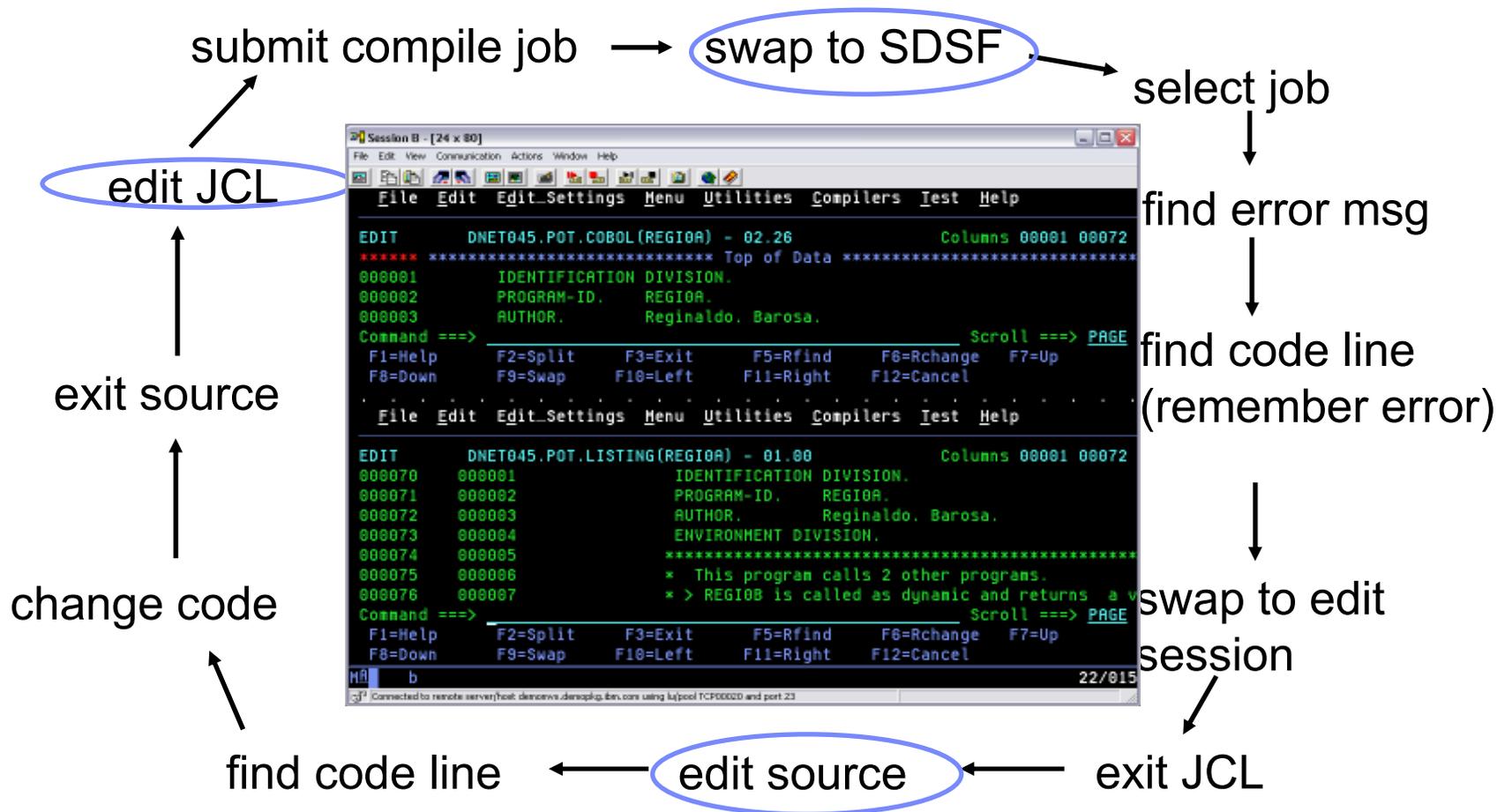
5 billion new COBOL LOC yearly

250 billion LOC of COBOL in existence

Approximately all active code is **80% COBOL**

One million COBOL programs currently active

Desarrollo basado en ISPF

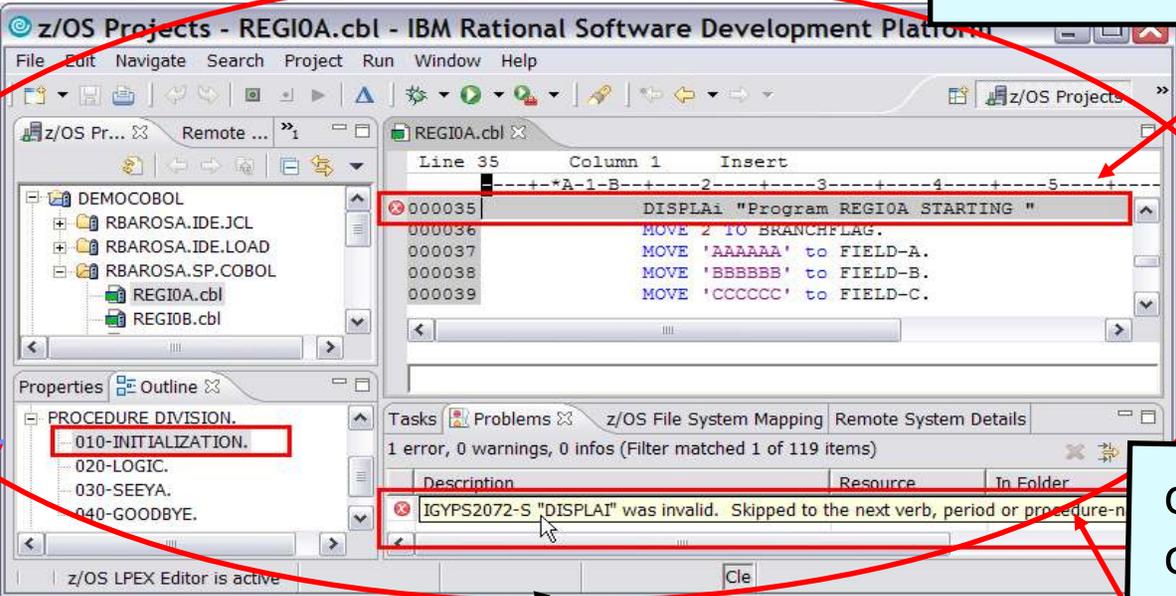


Desarrollo basado en Eclipse

Syntax Check

Edit source

Statement in error



double click on the error

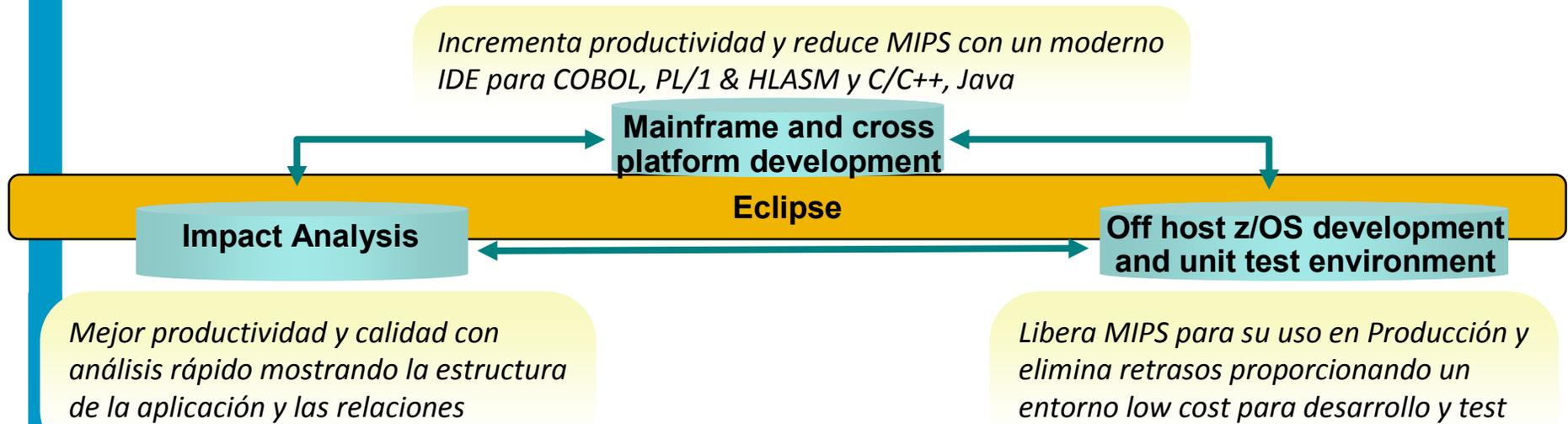
Outline view presents COBOL structure

Error list in Tasks view

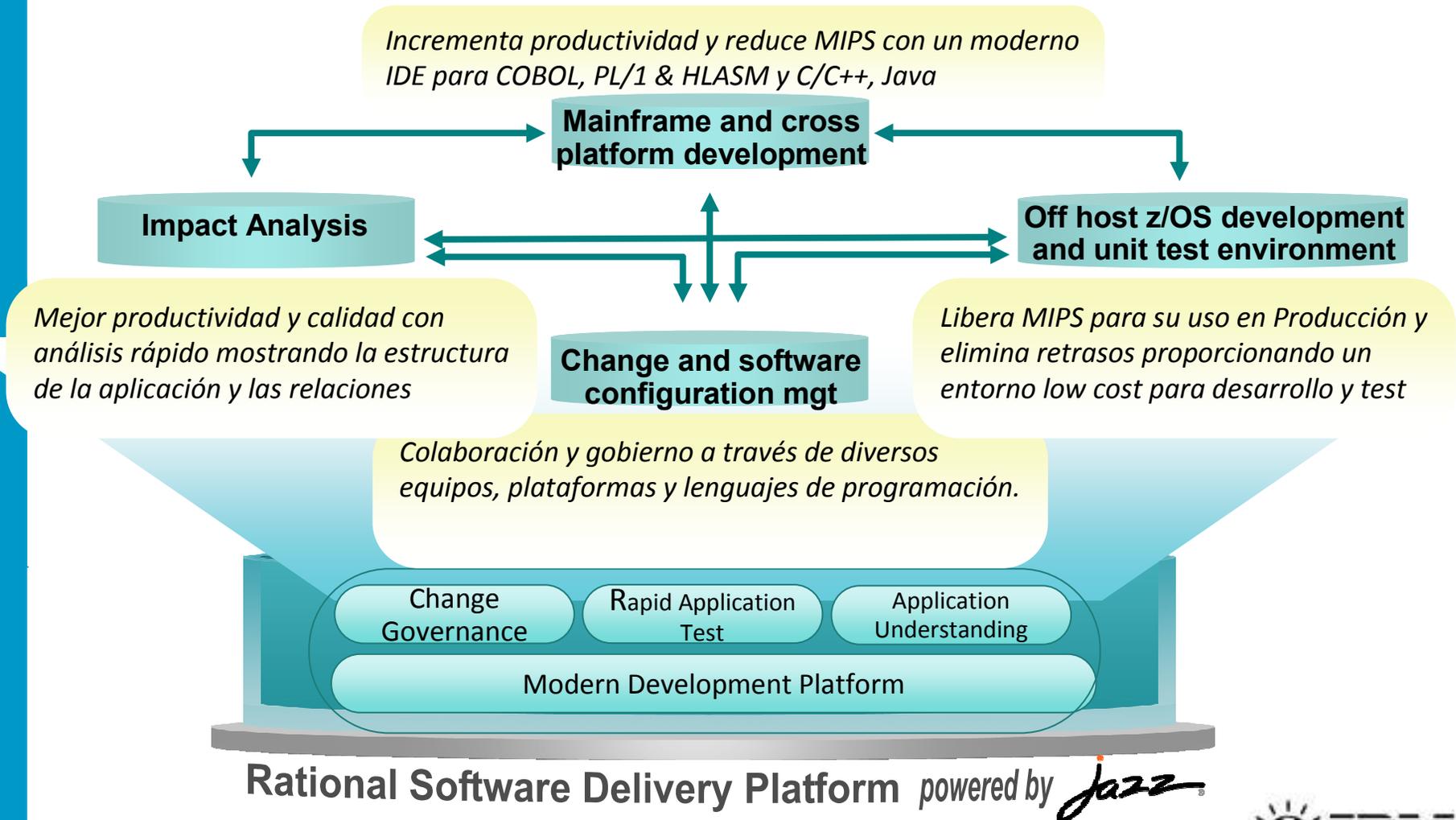
Beneficios: Simplificación del desarrollo en COBOL, PL/I, C/C++ y Java en un mismo entorno de desarrollo.



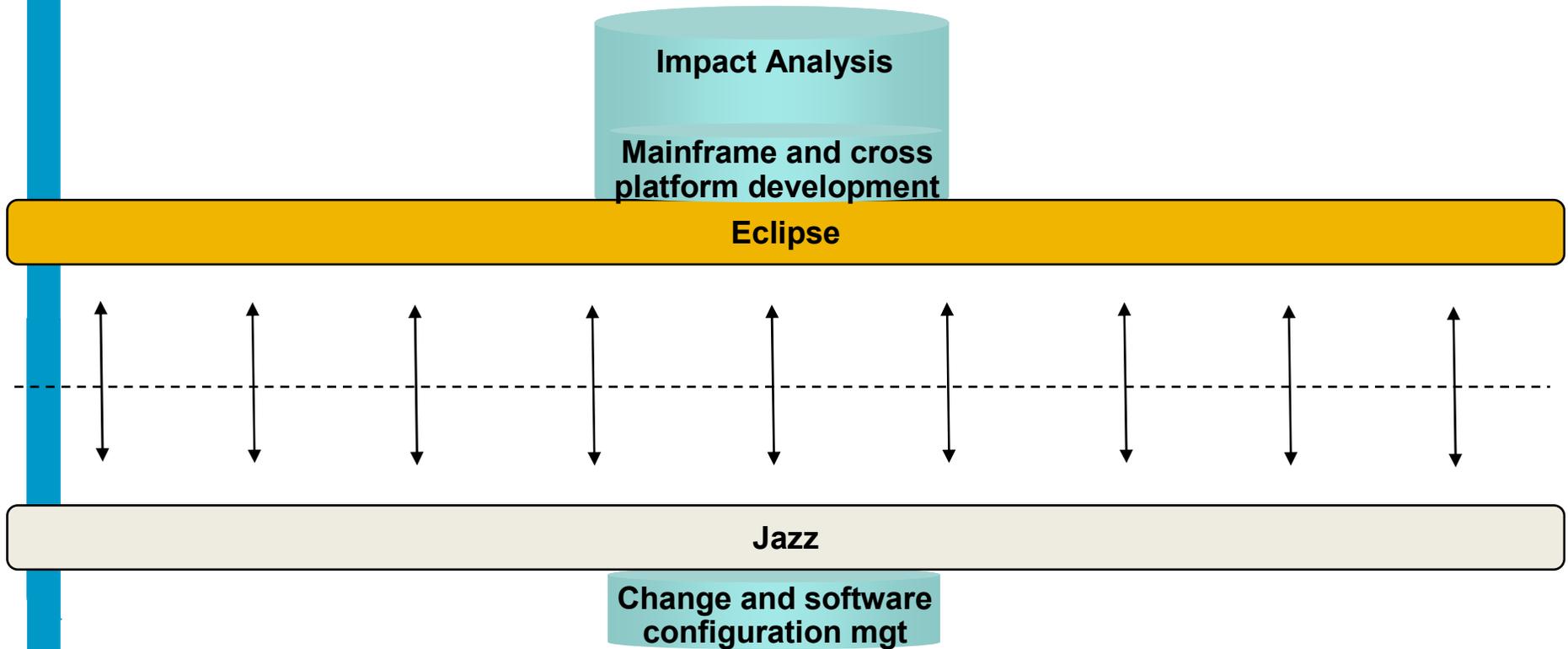
Solución integrada para el desarrollo de System z



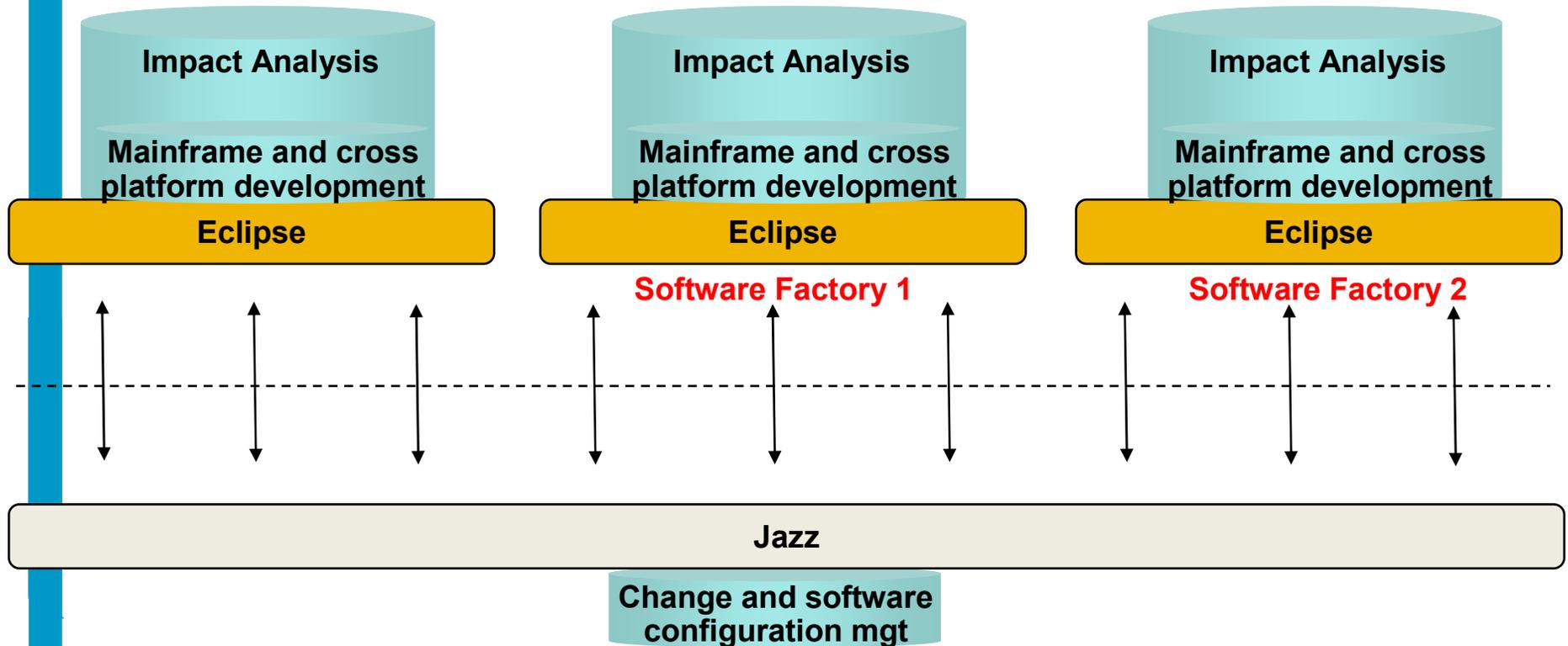
Solución integrada para el desarrollo de System z



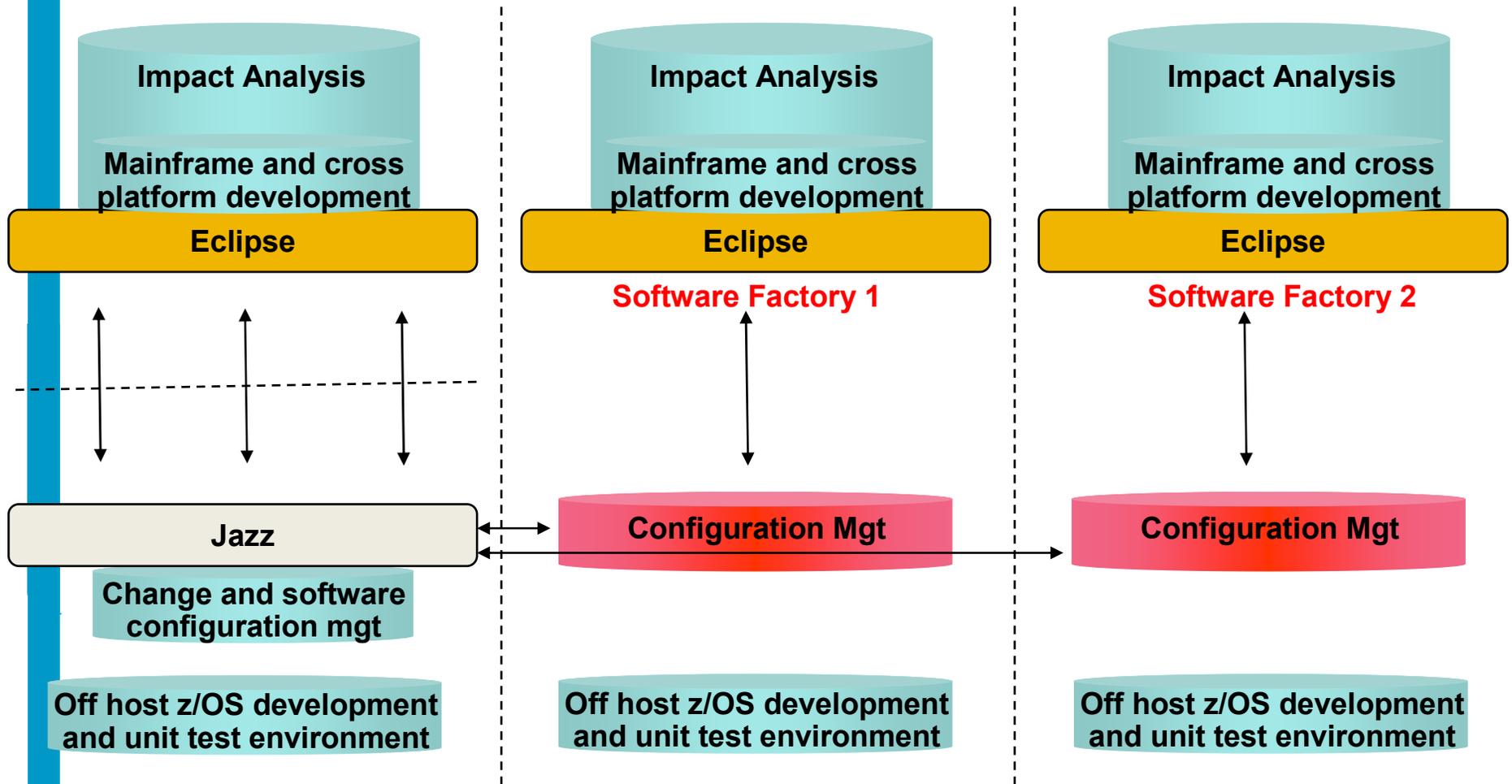
Arquitectura de la Solución (I)



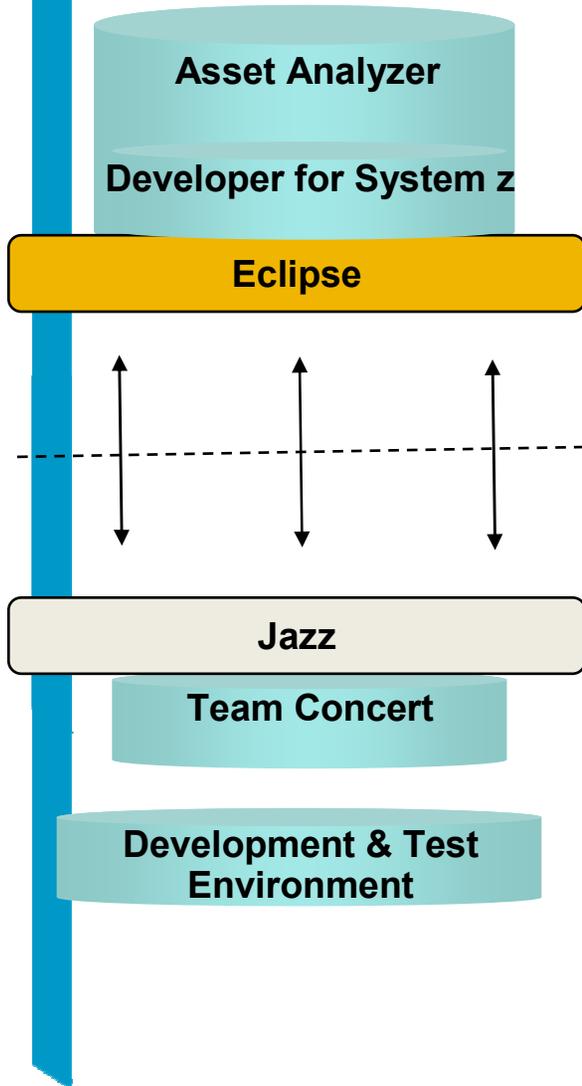
Arquitectura de la Solución (II)



Arquitectura de la Solución (III)



Componentes de la Solución





200x more COBOL transactions/day than **Google searches**

**5 billion new COBOL
LOC yearly**

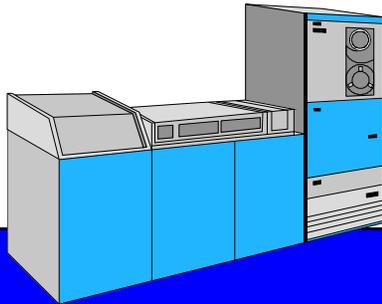
**250 billion LOC of
COBOL** in existence

COBOL training **no longer offered** at most universities

Approximately all active code is
80% COBOL

One million COBOL
programs currently active

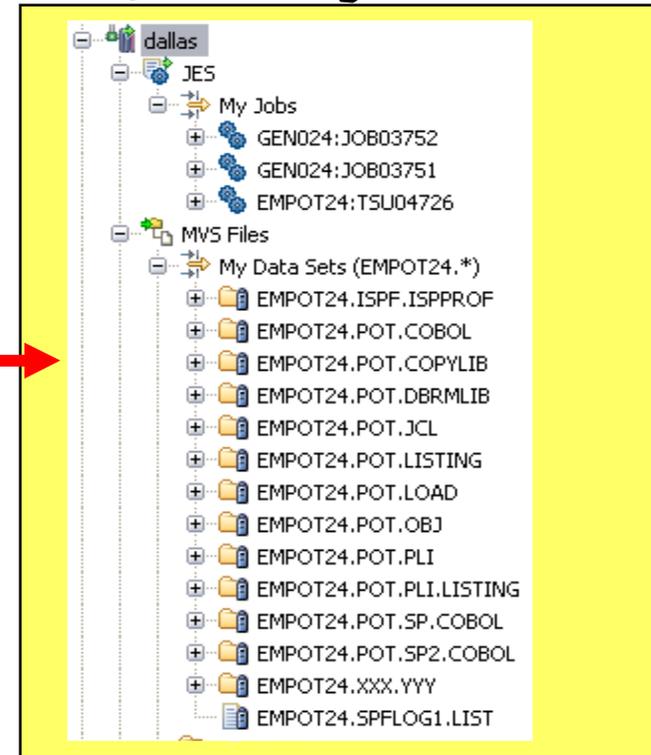
Developer for System z (RDz)



```
GEN024 JOB03751 EMPOT24  
GEN024 JOB03752 EMPOT24
```

```
EMPOT24.HFS  
EMPOT24.ISPF.ISPPROF  
EMPOT24.POT.COBOLE  
EMPOT24.POT.COPYLIB  
EMPOT24.POT.DBRMLIB  
EMPOT24.POT.JCL  
EMPOT24.POT.LISTING  
EMPOT24.POT.LOAD  
EMPOT24.POT.OBJ  
EMPOT24.POT.PLI  
EMPOT24.POT.PLI.LISTING  
EMPOT24.POT.SP.COBOLE  
EMPOT24.POT.SP2.COBOLE  
EMPOT24.SPFLOG1.LIST  
EMPOT24.XXX.YYY
```

'Magic'



Los archivos en el "host" parecen como si estuvieran en la "workstation"



Editor: Asistente de contenido para COBOL - PL/I - C o C++

```
000066 EXIT.  
000067 *  
000068 0150-SECOND-PART.  
000069 MOVE 2 TO BRANCHFLAG.  
000070 ex  
000071 MC  
000072 MC  
000073 MC  
000074 MC  
000075 0200-LOGI  
000076 IF WH...  
000077 * If is LAB2  
000078 MOVE  
000079 CALL
```

0150-SECOND-PART.
MOVE 2 TO BRANCHFLAG.
move
CUST-ADDR1
CUST-CITY
CUST-CTRY
CUST-FN
CUST-LN
CUST-NO
CUST-ST
POTVSAM-RECORD-REC

POTVSAM-FILE
01 POTVSAM-RECORD-REC.
...
03 CUST-ADDR1 PIC X(20).

orce a divide by ZERO
D-FROM-CALLED

divide by ZERO
CALLED

SEND TEXT
SET AUTOINSTALL
SET BRFACTORY

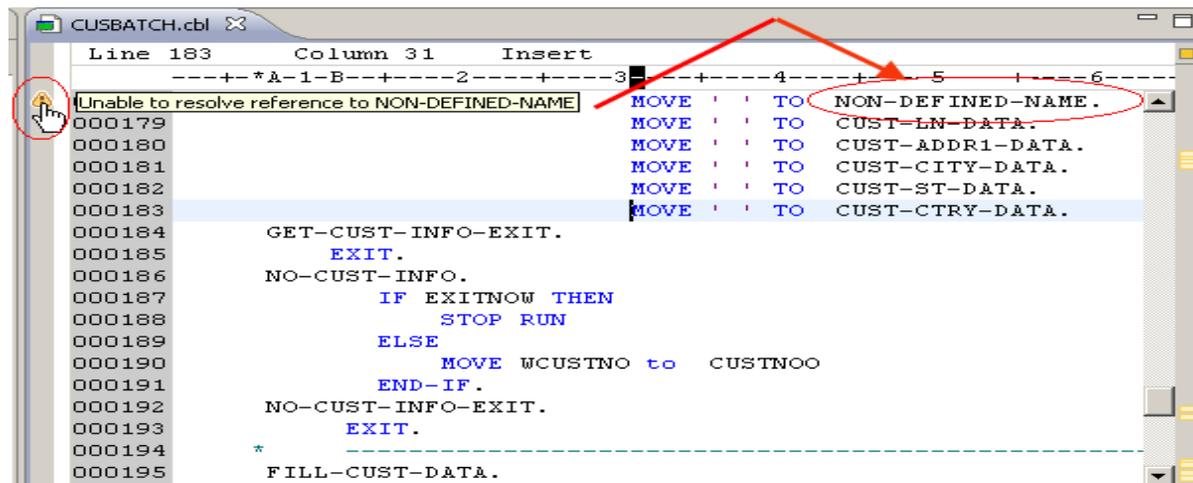
Find all statements and variables (including from COPYBOOKS or INCLUDE)

Beneficio: Desarrolladores finalizan el código de una forma más precisa y eficiente.



Editor: Comprobación en tiempo real de la sintaxis en COBOL

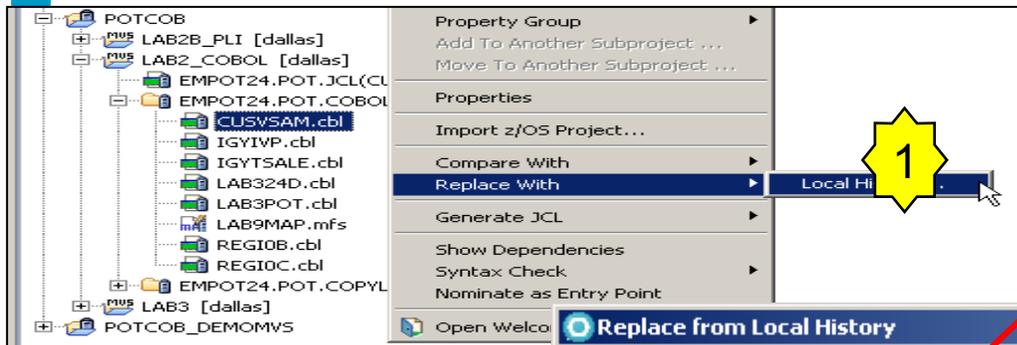
Real-time syntax check without requiring code compile or save



```
Line 183      Column 31      Insert
-----+*A-1-B-----+-----+-----+-----+-----+-----+-----+
Unable to resolve reference to NON-DEFINED-NAME  MOVE  |  |  TO  NON-DEFINED-NAME.
000179                                           MOVE  |  |  TO  CUST-LN-DATA.
000180                                           MOVE  |  |  TO  CUST-ADDR1-DATA.
000181                                           MOVE  |  |  TO  CUST-CITY-DATA.
000182                                           MOVE  |  |  TO  CUST-ST-DATA.
000183                                           MOVE  |  |  TO  CUST-CTRY-DATA.
000184
000185      GET-CUST-INFO-EXIT.
000186      EXIT.
000187      NO-CUST-INFO.
000188          IF EXITNOW THEN
000189              STOP RUN
000190          ELSE
000191              MOVE WCUSTNO to CUSTNOO
000192          END-IF.
000193      NO-CUST-INFO-EXIT.
000194      EXIT.
000195
*-----+-----+-----+-----+-----+-----+-----+-----+
FILL-CUST-DATA.
```

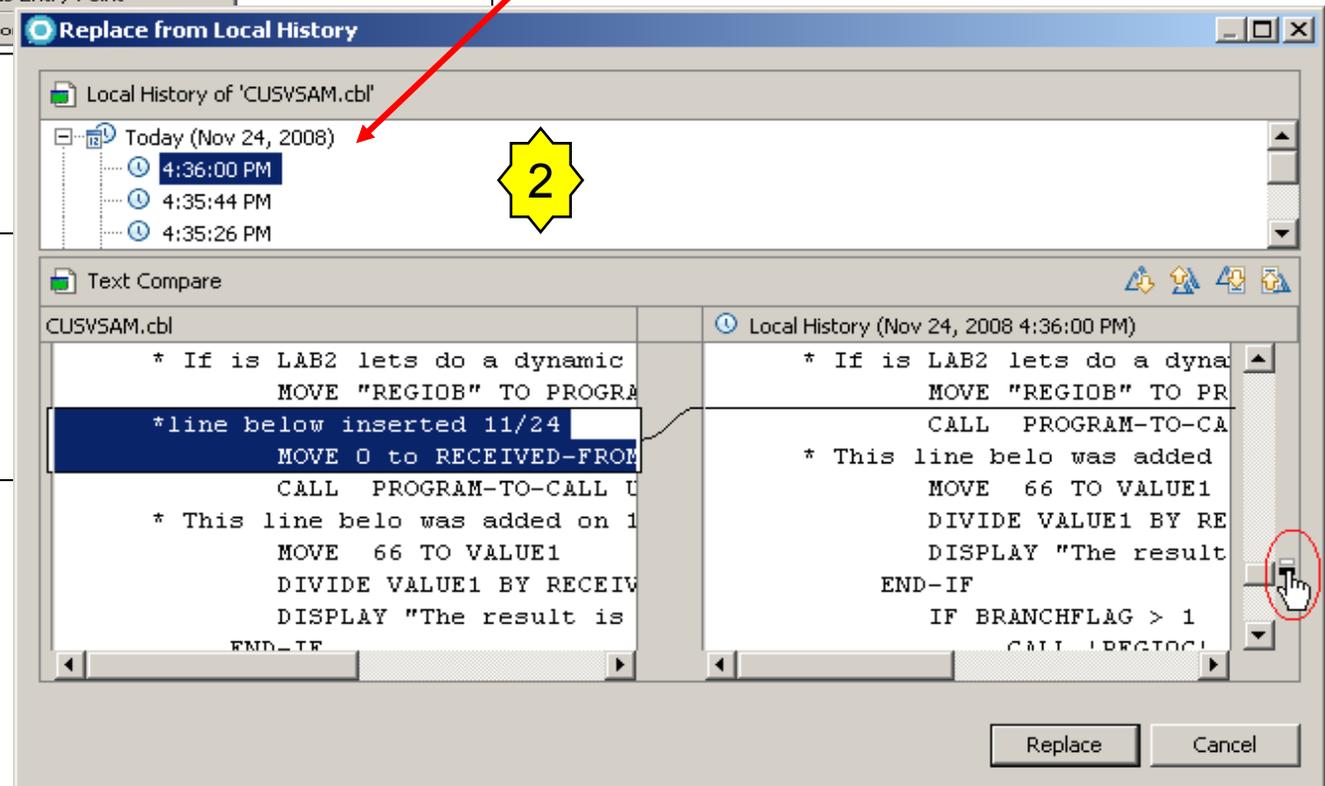
Beneficio: Mejorar la productividad del código.

Funciones de Eclipse útiles disponibles para activos z/OS. Reemplazar con Historia Local.

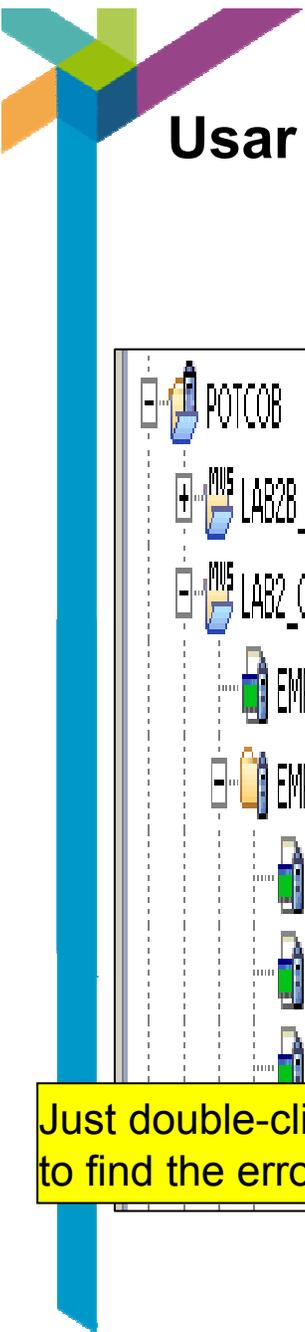


Keep as many local versions as you want and compare with the z/OS current version..

Beneficio: Mejorar la productividad ahorrando en tiempo de recuperación ...



Usar compiladores locales o remotos para verificar la sintaxis



Replace With

Generate JCL

Local Syntax checking..

1

POTCOB

LAB2B_PLI [dallas]

LAB2_COBOL [dallas]

EMPOT24.POT.JCL(CUSV5

EMPOT24.POT.COBO

CUSVSAM.cbl

IGYIVP.cbl

IGYVSALE.cbl

240.cbl

Line 75 Column 1 Insert

```
000071 MOVE 'BBBBBB' to FIELD-B.
000072 MOVE 'CCCCCC' to FIELD-C.
000073 MOVE "LAB2" to WHICH-LAB.
000074 0200-LOGIC.
000075 DSPLAY "Program " PROGRAM-TO-CALL "starting..
000076 IF WHICH-LAB = 'OFF'
000077     "REGIOB" TO PROGRAM-TO-CALL
000078     0 to RECEIVED-FROM-CALLED
000079     PROGRAM-TO-CALL USING RECEIVED-FROM-
000080     MOVE 66 TO VALUE1
```

2

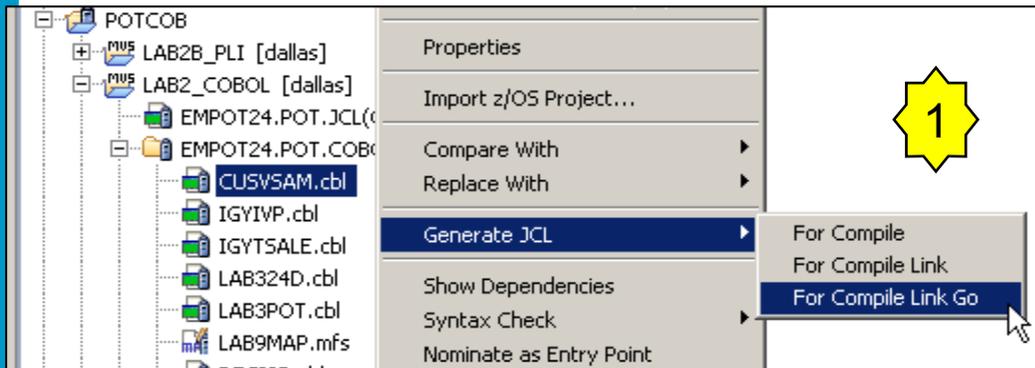
Just double-click to find the error

Beneficios: Mejorar la productividad, poder ahorrar z/OS CPU.

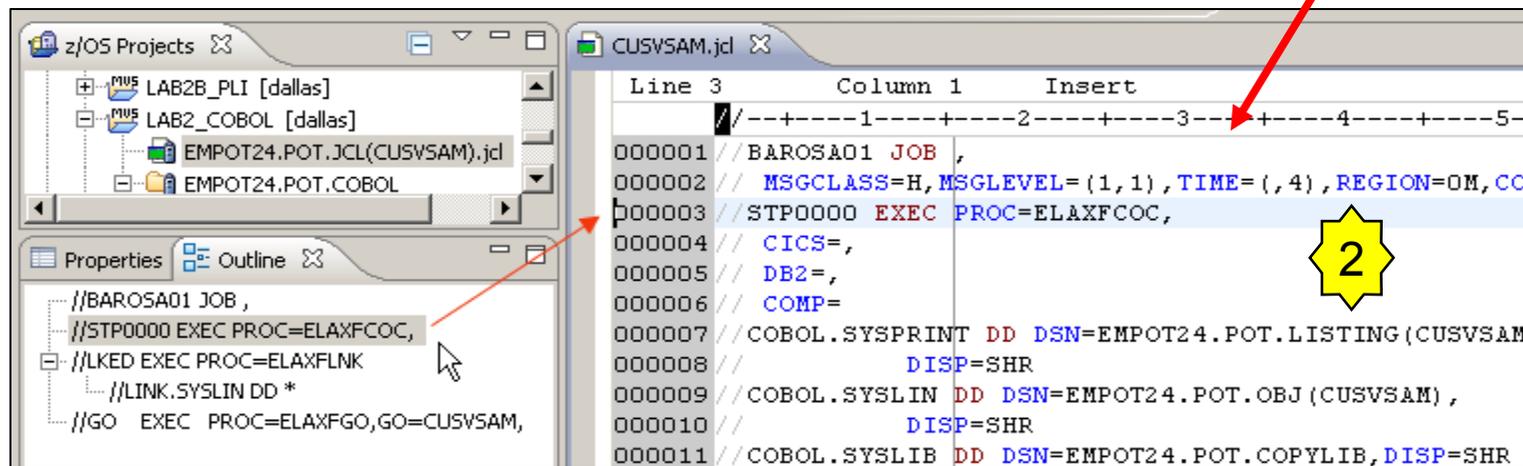
ID	Message	Se...	Line	Location
IGYGR1130	IGYGR1130-I "RECORD KEY" or "ALTERNATE KEY""CUST-NO" w...	0	12	POTCOB/LA
IGYPS2072	IGYPS2072-S "DSPLAY" was invalid. Skipped to the next verb, ...	2	75	POTCOB/LA



Generación y submit de JCL para su ejecución en z/OS



JCL generated from COBOL PL/I or C Code



Beneficio: Desarrolladores enfocados en lógica de negocio y no en escribir JCL: JCL smart editor, Outline...



Debug remoto y local

→ Depurar aplicaciones z/OS desde la workstation mientras se ejecutan en un runtime remoto

Debug - EMPOT24.POT.LISTING(CUSVSAM) - IBM Rational Developer for System z

File Edit Navigate Search Project Data Run Window Help

Debug Remote System... z/OS Proj...

Change contents, etc..

Breakpoints, watchpoints, Jump to, Run to etc..

Variables Breakpoints Registers Monitors Modules

Name	Value
WHICH-LAB	'LAB2'

CUSVSAM [Remote Compiled Application]

390(R) Connection: 192.195.29.60:3971
(Runnable)
SAM : 01
237240 Program: CUSVSAM

IG(CUSVSAM)

Line	Column	Insert	Browse
000081	81	MOVE 2 TO BRANCHFLAG.	
000082	82	MOVE 'AAAAAA' to FIELD-A.	
000083	83	MOVE 'BBBBBB' to FIELD-B.	FIELD-A = 'AAAAAA'
000084	84	MOVE 'CCCCCC' to FIELD-C.	
000085	85	MOVE "LAB2" to WHICH-LAB.	
000086	86	0200-LOGIC.	
000087	87	IF WHICH-LAB = 'LAB2'	

Beneficio: Misma perspectiva Debug para COBOL, PL/I, C, C++, Java, JSP, etc..
→ END to END Debug

Necesita tener instalado el producto z/OS Debug Tool



Monitorizar el Job Output

The screenshot displays the IBM Rational Developer for System z interface. The main window shows the job output for JOB13884.out, which includes a JES2 JOB LOG and a table of job details. A red arrow points from the 'JOB13884.out' tab to the first row of the table. The table lists job details such as Job ID, Job Name, Job Owner, Job Entry Date, Return Code, and Return Info. A context menu is open over the first row, with the 'Purge' option highlighted.

Resource	Job ID	Job Name	Job Owner	Job Entry Date	Return Code	Return Info	System r...	User return...	Return Status	System Na
• DNET0451:JOB13884	JOB13884	DNET0451	DNET045	2009/12/04 13:06:20	U0004	NORMAL		004	COMPLETION	
• DNET0451:JOB13860	JOB13860	DNET0451	DNET045	2009/12/04 12:34:22	U0000	NORMAL			COMPLETION	
• DNET0451:JOB13801	JOB13801	DNET0451	DNET045	2009/12/04 10:22:19	S0CB	ABENDed			ABEND	
• DNET0451:JOB13800	JOB13800	DNET0451	DNET045	2009/12/04 10:19:02	U0004	NORMAL			COMPLETION	
• DNET0451:JOB06679	JOB06679	DNET0451	DNET045	2009/10/26 07:49:06	U0004	NORMAL			COMPLETION	

Beneficio: → Los desarrolladores no tienen que continuamente cambiar entre sistemas para usar System Display y Search Facility (SDSF).
No necesitan TSO ni SDSF.
→ Impresión en local.

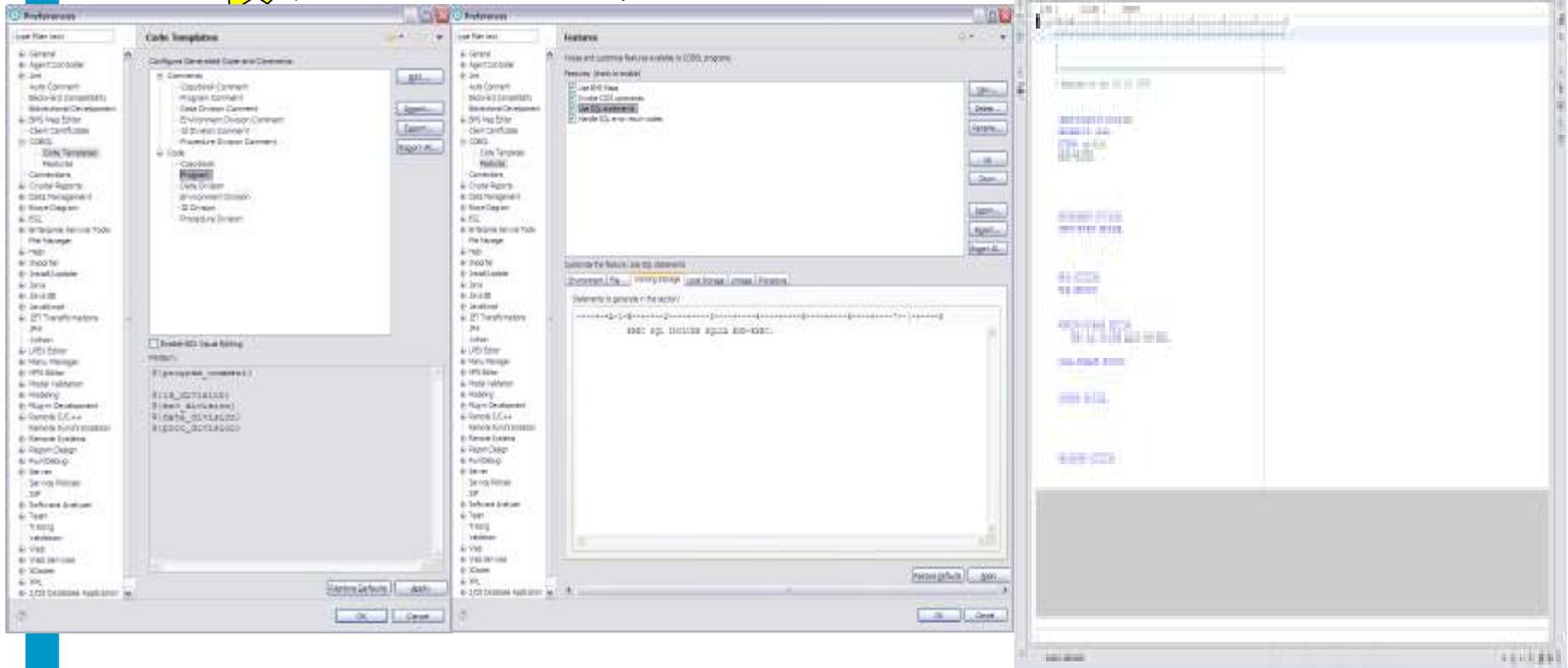
Plantillas para programas

- Ayuda a los programadores a utilizar plantillas pre-definidas.
- Se definen las plantillas y sus características
- El desarrollador las incorpora al crear el programa

Code templates
1

Features
2

Creacion
3





Soporte para preprocesador

Edit Property Group
Edit the properties in the property group

Local COBOL Settings:
Local Link Options

Local COBOL Settings

Local Compiler Options | Local Preprocessor

Pre-processor Options

Preprocessor Name (fully qualified):
C:\Program Files\IBM\SOFT\bin\resolveinc.exe [Browse...]

Preprocessor Description:
resolve INC lines

Preprocessor Arguments:
SRC(\$resource_loc) XML(\$resource_fn_PP.xml) DECK(\$resource_fn_PP.dek) DIR(\$project_loc) OUTPUT(\$buildOutput) SYSL(CopyLib)

Preprocessor Output File Name:
\$resource_fn_PP.dek

Support Error Feedback

Error Feedback XML File Name:
\$resource_fn_PP.xml [Browse...]

Environment Variables (Set Statements):
[Text Area]

< Back Next > Finish Cancel

■ Parámetros del preprocesador

➤ Local:

- ✓ El cliente necesita codificar su propio preprocesador
- ✓ El Preprocesador requiere escribir su fichero de XML y su fichero DEK

(Botón browse para apuntar a ellos)

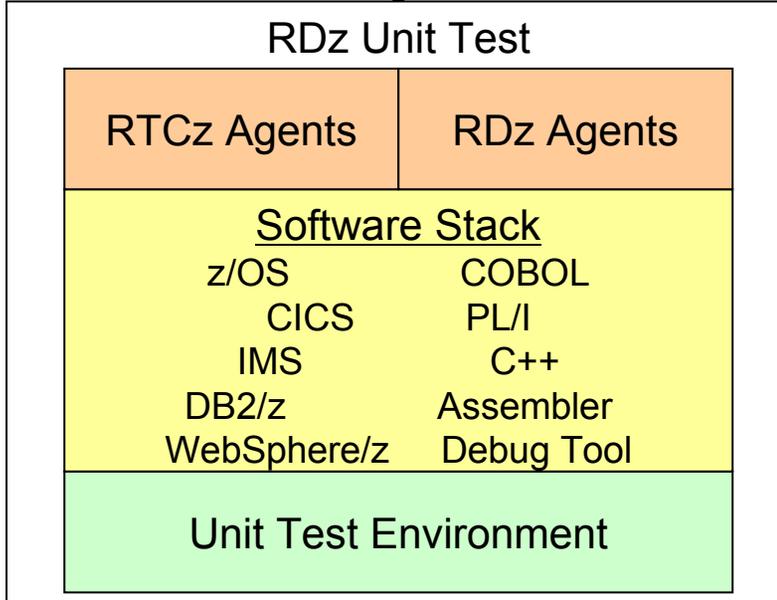
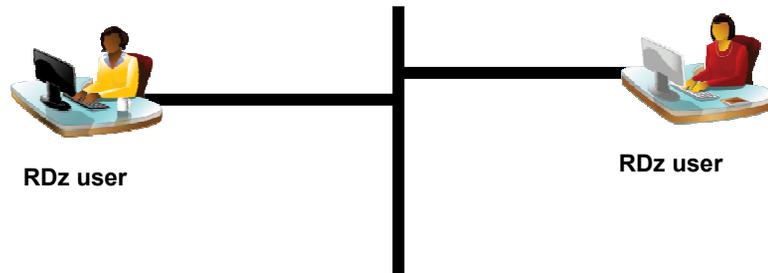
- ✓ Pasar los argumentos necesarios en las opciones

➤ Remoto:

- ✓ Ya codificado por el cliente
- ✓ Añadir pasos en el JCL
- ✓ Los Procs de JCL deberán incluir el paso añadido, y pasar la salida del fichero al compilador



Rational Development & Test Environment (RD&T)



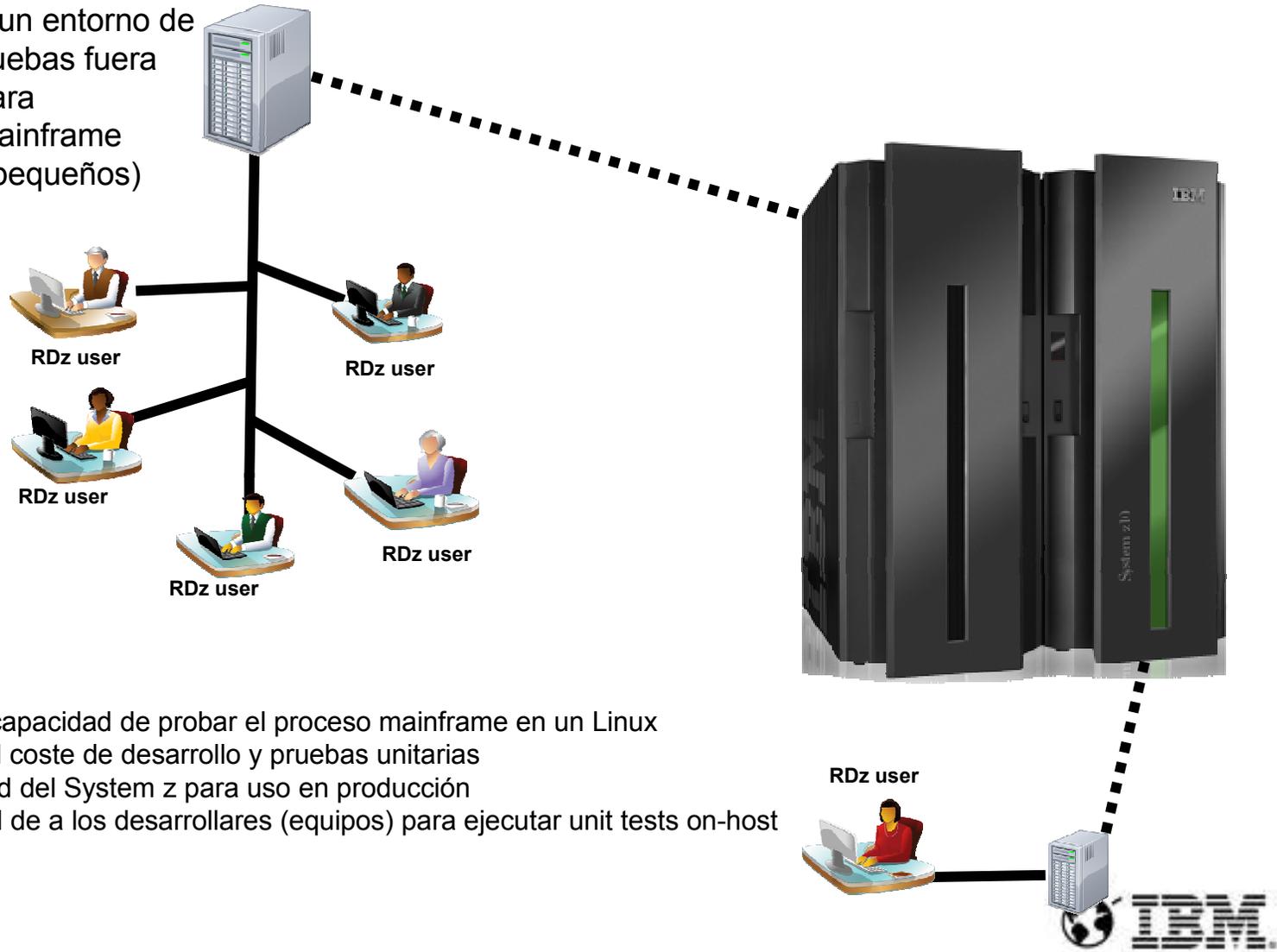
RD&T consiste en lo siguiente:

- Dev&Test – El entorno Unit Test corre sobre un sistema Linux basado en procesador Intel.
 - Puede proporcionar una plataforma de desarrollo de System z y es capaz de ejecutar sistemas operativos z/OS actuales
 - Nota: RD&T, solamente se usa como un sistema de desarrollo y no debe utilizarse (ni está diseñado) para ejecutar workloads de producción.
 - Proporciona gran flexibilidad a la hora de ejecutar un entorno muy configurado
- El stack de software incluido proporciona un middleware de IBM para entorno de prueba
 - Software middleware real (incluyendo z/OS)
 - Compiladores Enterprise reales
 - no simulación de APIs
- Agentes RDz y RTCz
 - Empaquetados para simplificar
 - Necesita la licencia RTC client para activar



Desarrollo actual en Mainframe con RD&T

La RD&T crea un entorno de desarrollo y pruebas fuera de System z para aplicaciones mainframe (para equipos pequeños)



Beneficios:

- Proporciona la capacidad de probar el proceso mainframe en un Linux
- Ayuda a bajar el coste de desarrollo y pruebas unitarias
- Libera capacidad del System z para uso en producción
- Da la flexibilidad de a los desarrollares (equipos) para ejecutar unit tests on-host u off-host

Ventajas de la solución tecnológica (RDz)

- Desarrollo Cobol-PL/1 fuera del Mainframe, resto de tareas (compilación, prueba, ...) en Host



Incorporación de IDE moderno (RDz):

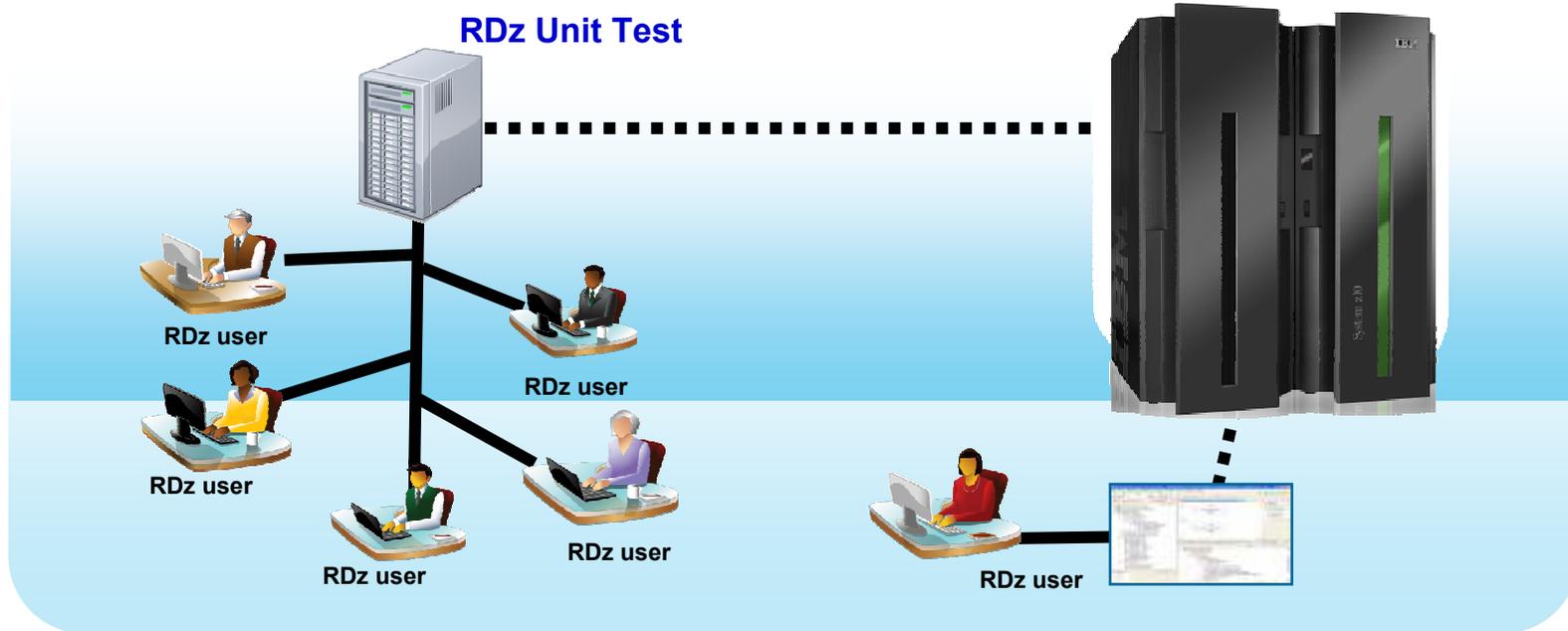
- Mejora de productividad, la herramienta aporta Eficiencia y Seguridad en Desarrollo.
- Reducción de una parte de los MIPS de desarrollo, ya que se trabaja en entorno local
- Integración con otras disciplinas del Ciclo de Vida del Desarrollo

Puntos de mejora:

- MIPS de desarrollo todavía elevados
- Las pruebas generan dependencias y tensiones en operativa de los entornos Mainframe.

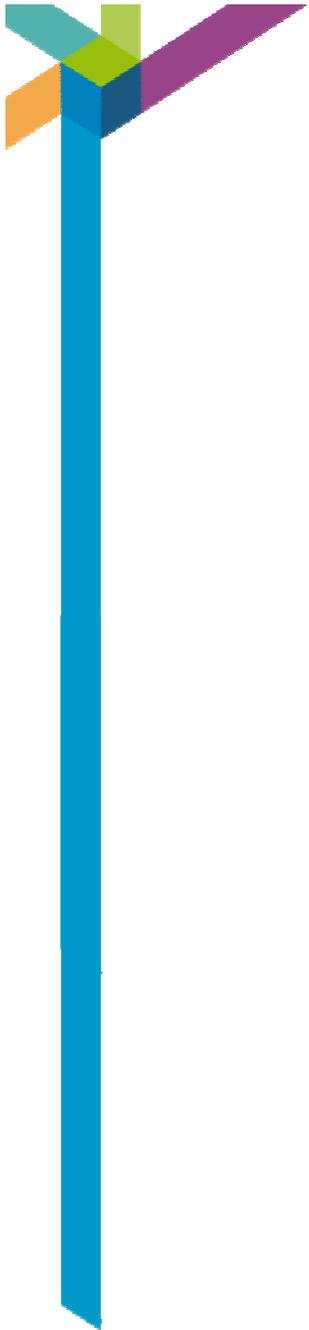
Ventajas de la solución tecnológica (RD&T)

- Desarrollo Cobol-PL/1 y Pruebas Unitarias fuera del Mainframe, reducción relevante de consumos en Host.



Incorporación de IDE moderno (RDz):

- Liberación relevante de MIPS en Desarrollo. Reducción Coste, mejora Producción
- Desarrollo y Test de aplicaciones en entorno más flexible y cercano, no productivo
- RD&T simple de mantener y gestionar (menos costes operativos)



Retorno de la Inversión (R.O.I.)

RDz y RD&T en el ciclo de desarrollo

BENEFICIOS ESPERADOS

➤ **Mejoras en la productividad de desarrollo**

- ✓ Desarrollo a través de un único interfaz
- ✓ Simplificación del proceso de desarrollo
 - Uso de editores avanzados
 - Entorno multitarea que permite actividades simultáneas
 - Disponibilidad de herramientas gráficas de debugging
- ✓ Reducción de la curva de aprendizaje para tecnologías Host
 - ✓ Posibilidad de ampliación futura a otras funcionalidades disponibles

➤ **Optimización del consumo de CPU y recursos Host**

- ✓ Reducción muy importante del consumo TSO
- ✓ Menor número de compilaciones
- ✓ Compilar fuera del mainframe



Benchmarks de eficiencia del IDE (RDz)

- 100 tareas comunes (diarias) de ISPF que se ejecutan durante asignaciones de soporte y mantenimiento.
 - Traducción del workflow ISPF (click-for-click) a desarrollo en RDz
 - Los participantes del proyecto creyeron que estaban intentando encontrar gaps entre la funcionalidad de RDz e ISPF
- “Apples-to-Apples” y test scripts
- Mezcla de experimentados (veteranos) programadores de ISPF y nuevos desarrolladores recién incorporados a plantilla
- La primera impresión fue que solamente los recién llegados serían más productivos...
 - ...y resultó ser falso

Based on IBM internal productivity study.

All performance data contained in this presentation was obtained in the specific operating environment and under the lab conditions and is presented as an illustration only.

Performance obtained in other operating environments may vary and customers should conduct their own testing.

Analytics		All Participants			
Use Case	% Less time to complete tasks with RDz				
Source Navigation:	45.00				
Program Analysis:	100.90				
Primitive Edit Operations:	19.69				
COBOL Statement Coding:	35.79				
Syntax Check:	57.11				
Program Compile/Link Edit:	43.52				
DB2 data edit/SQL statement work:	101.30				
Total - all use cases:	57.19				

Analytics		ISPF Top Guns			
Use Case	% Less time to complete tasks with RDz				
Source Navigation:	47.21				
Program Analysis:	72.60				
Primitive Edit Operations:	19.84				
COBOL Statement Coding:	36.89				
Syntax Check:	68.86				
Program Compile/Link Edit:	14.14				
DB2 data edit/SQL statement work:	61.68				
Total - all use cases:	28.86				





Mándame un email a alejandro.leon@es.ibm.com y te lo haré llegar

Appendix 1- RDz Productivity Benchmarks

This Appendix documents a Benchmark completed in March 2010 that compared two IBM products: RDz v7.6.1 and ISPF v6.0.

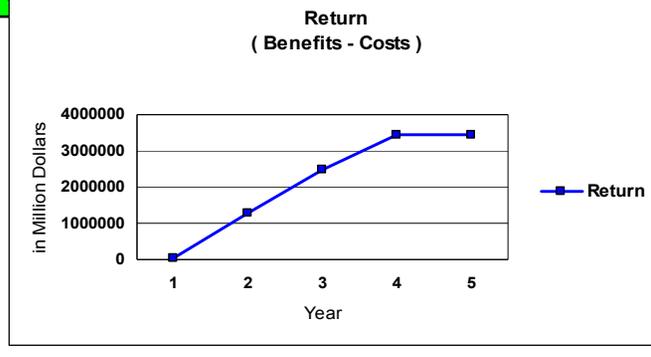
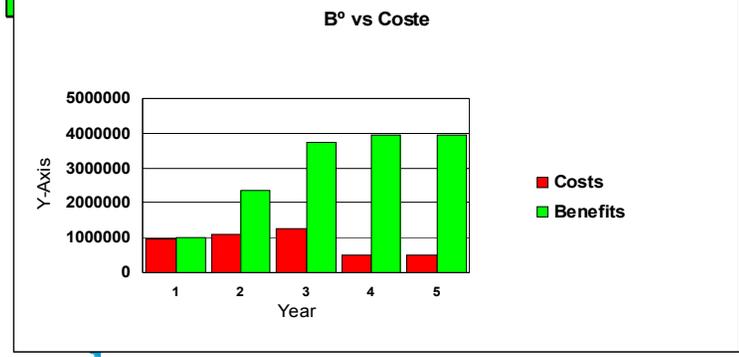
The work performed in the study was a set of tasks normally associated with traditional z/OS COBOL maintenance and production support:

- ▶ Code analysis - paragraph flow and program structure analysis
- ▶ Navigating and modifying existing COBOL programs
- ▶ Adding a small amount of new business logic to an existing program
- ▶ Updating fields in copybooks and modifying code that referenced to updated fields
- ▶ Doing tradition Data Flow analysis
- ▶ Working with SQL and DB2 test data
- ▶ Compiling and linking a program



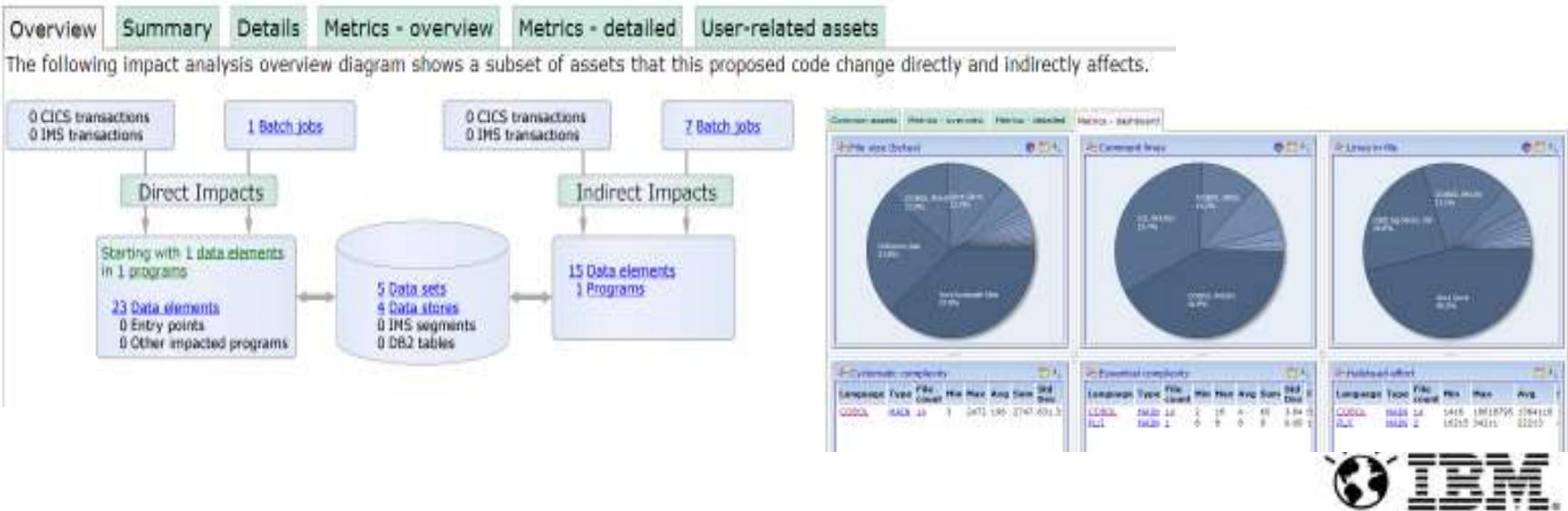
	Year 1	Year 2	Year 3	Year 4	Year 5	Total
	2010	2011	2012	2013	2014	
Rollout RDz	150	175	175			500
Roll Out RD&T	50	50	50			150
Costs						
Software	730.988	823.706	823.706			2.378.400
Subscription	0	146.213	310.950	475.688	475.688	1.408.538
Education/Services	125.000	52.500	52.500			230.000
Setup/Admin Staff	82.500	55.000	55.000	27.500	27.500	247.500
Hardware	27.500	31.250	28.750			87.500
Total Costs	965.988	1.108.669	1.270.906	503.188	503.188	4.351.938
Justification						
Compile CPU Savings	103.726	224.740	345.755	345.755	345.755	1.365.730
TSO CPU Savings	255.528	553.644	851.760	851.760	851.760	3.364.452
Productivity	660.000	1.595.000	2.557.500	2.750.000	2.750.000	10.312.500
Total Justification	1.019.254	2.373.384	3.755.015	3.947.515	3.947.515	15.042.682
Return	53.267	1.264.716	2.484.108	3.444.327	3.444.327	10.690.745

DATOS BASE PARA EL ROI	
Núm Desarrolladores:	
Año 1	150/50
Año 2	175/50
Año 3	175/50
Coste anual por Desarrollador	55.000€
Coste 1 día TSO por Desarrollador	11,06 €
Coste medio de 1 compilación	0,62 €



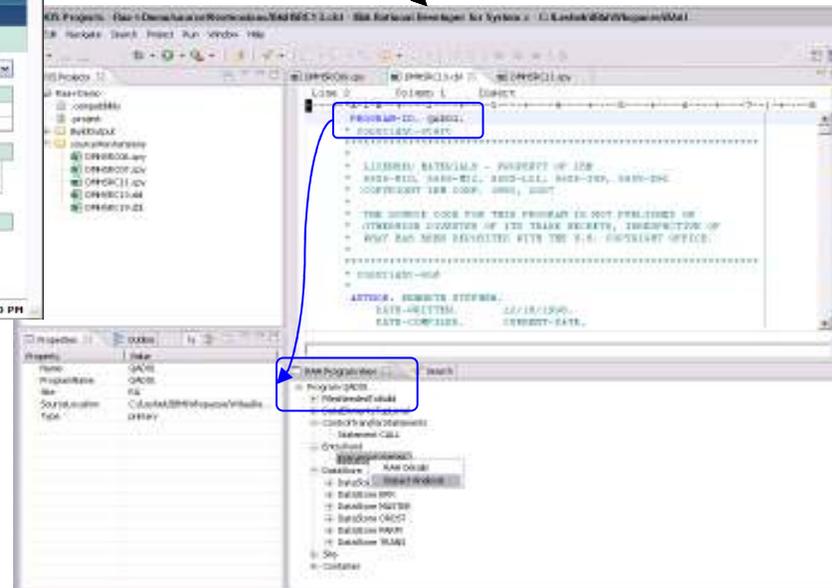
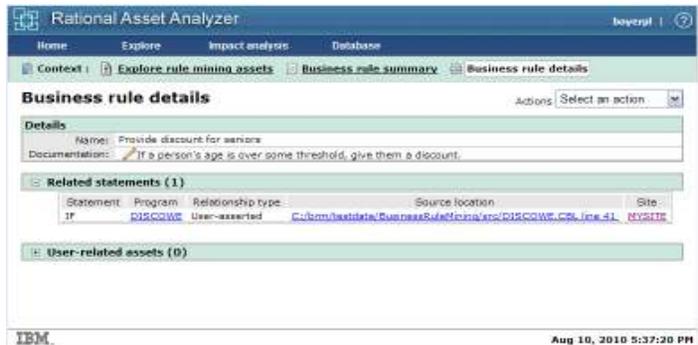
Asset Analyzer (RAA)

- **Acelera la entrega de proyectos respondiendo a las exigencias de negocio**
 - Reduce riesgos, Incrementa la productividad y mejora la calidad de los cambios en las aplicaciones
- Proporciona **control intelectual** de las aplicaciones
 - Identifica las **diferentes tecnologías** que se usan actualmente en la organización – usando dashboards o reports -
 - Comprende **la calidad y la complejidad** de los activos de desarrollo
- Proporciona **transparencia** hacia el desarrollo interno / outsourced
 - Agrupa artefactos en **user-defined groups** llamados Applications para limitar el alcance del área de interés.
 - Usa **varios tipos de diagramas** para entender cómo la aplicación se relaciona (mapa de aplicación)
- **Permite ser configurado e integrado** en los procesos de la organización y entornos de TI



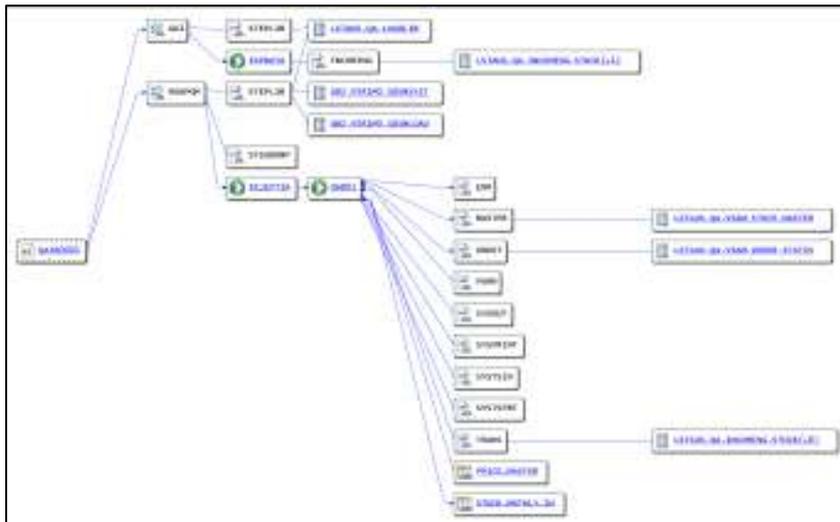
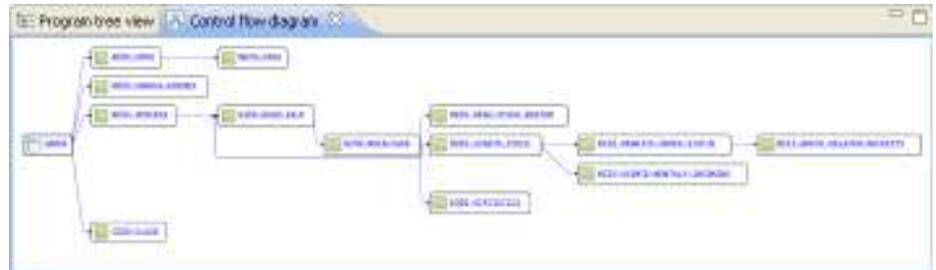
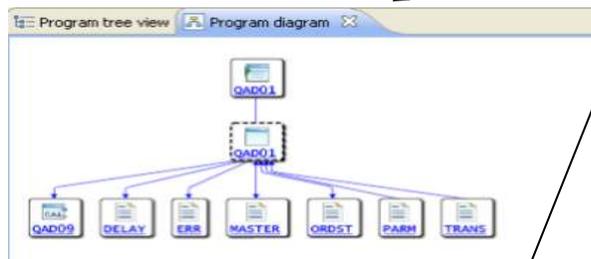
Asset Analyzer (RAA)

- Parte fundamental de la solución para la toma de decisiones ágil
 - Proporciona identificación de las reglas de negocio
 - Permite la captura y la gestión con WebSphere ILOG JRules BRMS
- Interfaces orientadas al Role
 - Interfaz web fácil de usar, para búsquedas y exploración
 - Interfaz eclipse para el Developer, integrada con RDz

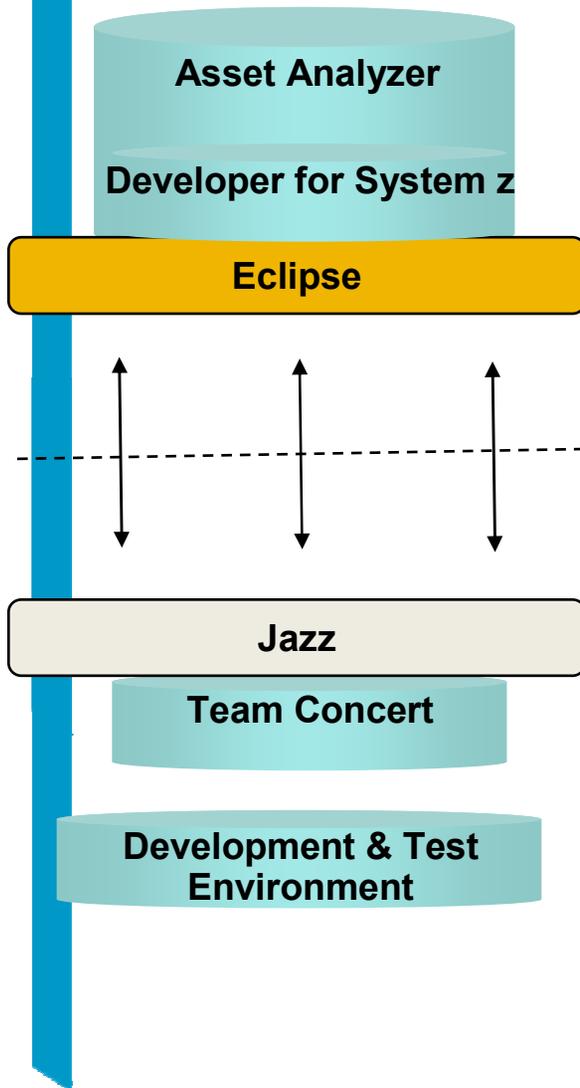


Asset Analyzer (RAA)

- Proporciona una cartografía o mapa de aplicaciones
 - Permite conocer rápidamente los componentes y sus relaciones



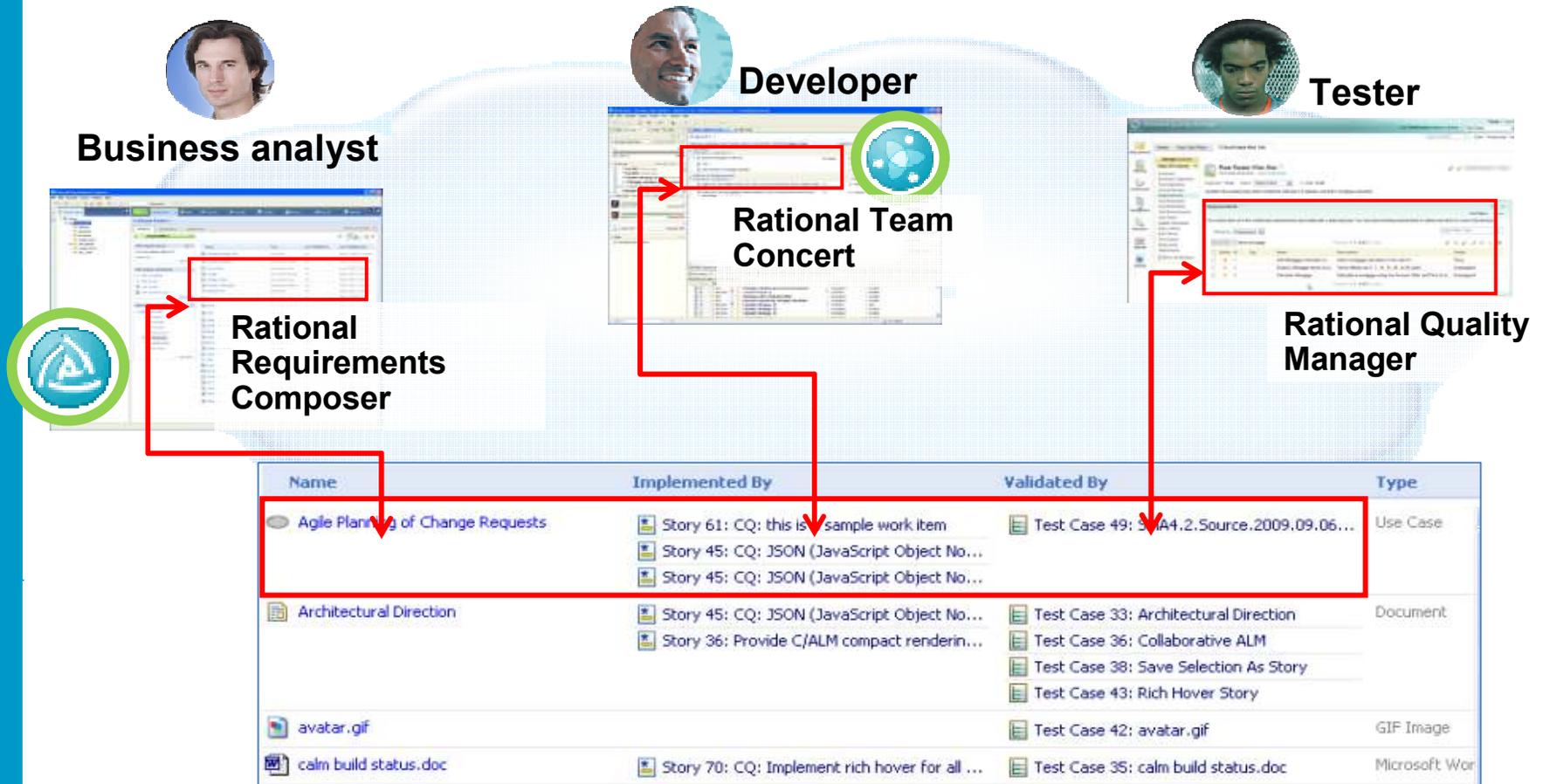
Beneficios de la Solución IBM



- Arquitectura multiplataforma (Windows, Unix, System i, System z)
- Integración a los colaboradores en el ciclo
 - ▶ Colaboración internos/externos en los desarrollos
 - ▶ Medida de productividad y visión de capacidad de la subcontrata, con control de acceso
 - ▶ Selección del mejor proveedor
- Repositorio Único, Plataforma de desarrollo homogénea,
 - ▶ Cobol, PL/I, HLASM, RPG, Java, C, C++, C#... Eclipse, .NET
 - ▶ Más fácil administración
- ▶ Integración de todas las capacidades de desarrollo y trazabilidad
 - ▶ CMMi, Seguridad, Metodologías Agil + Tradicional
 - ▶ Requisitos, Modelos, Configuraciones SW, Releases, Pruebas, Documentación
 - ▶ Sincronización y mayor confiabilidad en los pases a producción

C/ALM: Alinea las tareas de desarrollo y test con el valor del cliente

Collaborative Application Lifecycle Management rompe con los silos de información en pos de una mejor ejecución del proyecto





Muchas gracias

