SZ-1704 - Smarter Development and Testing for IBM® System z®

Sanjay Chandru
Leader, Rational Enterprise Modernization Domain and Accelerators
chandru@us.ibm.com

Agenda

//1 Who is ASIST

//2 Modern Development for zEnterprise

//3 Smarter Testing and Delivery for System z

//4 Conclusion and Q&A

Agenda

//1 Who is ASIST

//2 Modern Development for zEnterprise

//3 Smarter Testing and Delivery for System z

//4 Conclusion and Q&A

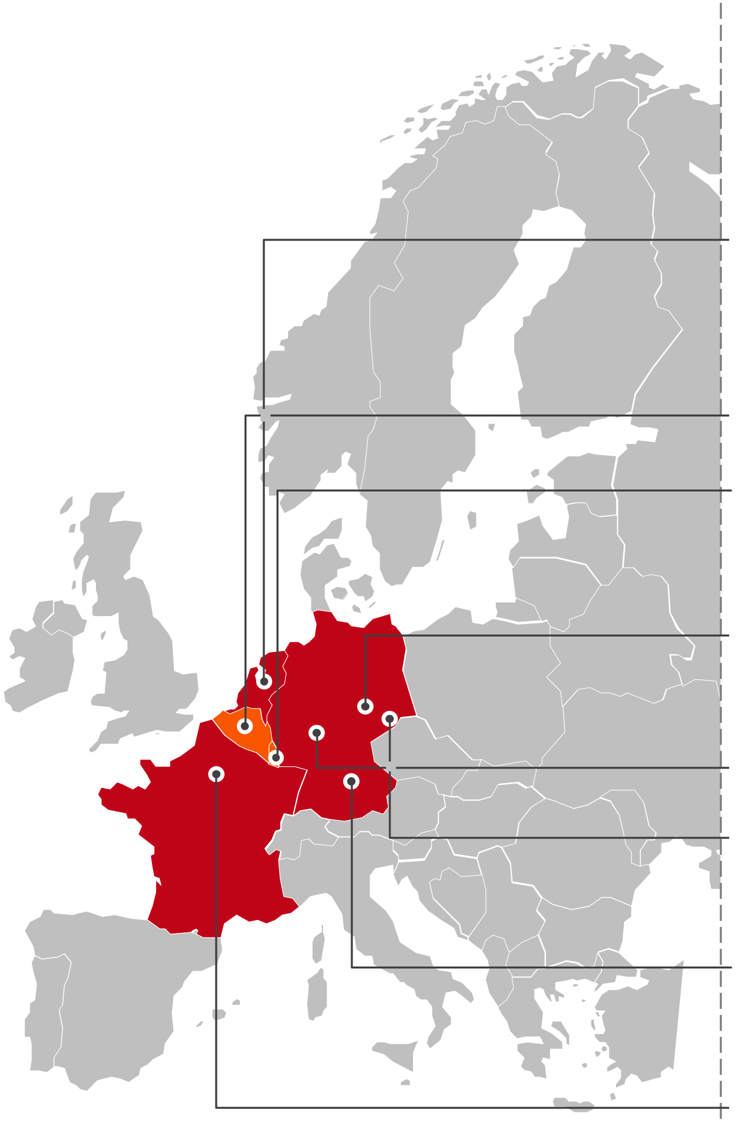
ASIST INTRODUCTION

Who we are

- ☐ Founded in Belgium in 1995
- ☐ Since January 2012 member of the international Softline Group

ASIST INTRODUCTION

Softline Group



softline Group

Softline Solutions Netherlands B.V.

Utrecht

Softline Solutions N.V.
ASIST BVBA

Leuven / Brussels

Luxembourg

Softline Solutions GmbH

Leipzig

Softline Systems & Services GmbH

Frankfurt am Main

Dresden

Prometheus GmbH

Munich

STR S.a.r.l.

Vélizy / Paris

Next NOW! 12/10/2

ASIST INTRODUCTION

Who we are

- ❑ Founded in Belgium in 1995
- ❑ Since January 2012 member of the international Softline Group
- ❑ IBM Premier Business partner
- ❑ Reputable recognition by IBM labs
- ❑ 95 % Certified Consultants
- ❑ SUSE Business partner

ASIST INTRODUCTION

ASIST makes the difference

❑ Knowledge

- In depth knowledge of RBD/EGL, full systems & customer’s legacy

❑ Attitude

- Adaptation to customer’s thinking & practices and organisation culture

❑ Experience

- Hands-on experience in Rational environment: migrations, integrations, development environments, training & coaching

❑ Specialization

- 100% focus on Rational

❑ Location & geographical coverage

- Located in heart of Europe & project coverage around the globe.

❑ Languages

- Consultants with minimum of 3 languages (Dutch, French, English)

Agenda

//1 Who is ASIST

//2 Modern Development for zEnterprise

//3 Smarter Testing and Delivery for System z

//4 Conclusion and Q&A

Modern Development for zEnterprise

Rational Developer for zEnterprise

and

Rational Development and Test Environment for System z

Four key barriers preventing optimal return on IT investments

Decades of application investments

“We don’t understand the effort, risk and impact of modernizing our legacy applications.”

Islands of skills, languages and platforms

“Our skills gap keeps growing. How do we stay current with all the language and technology changes?”

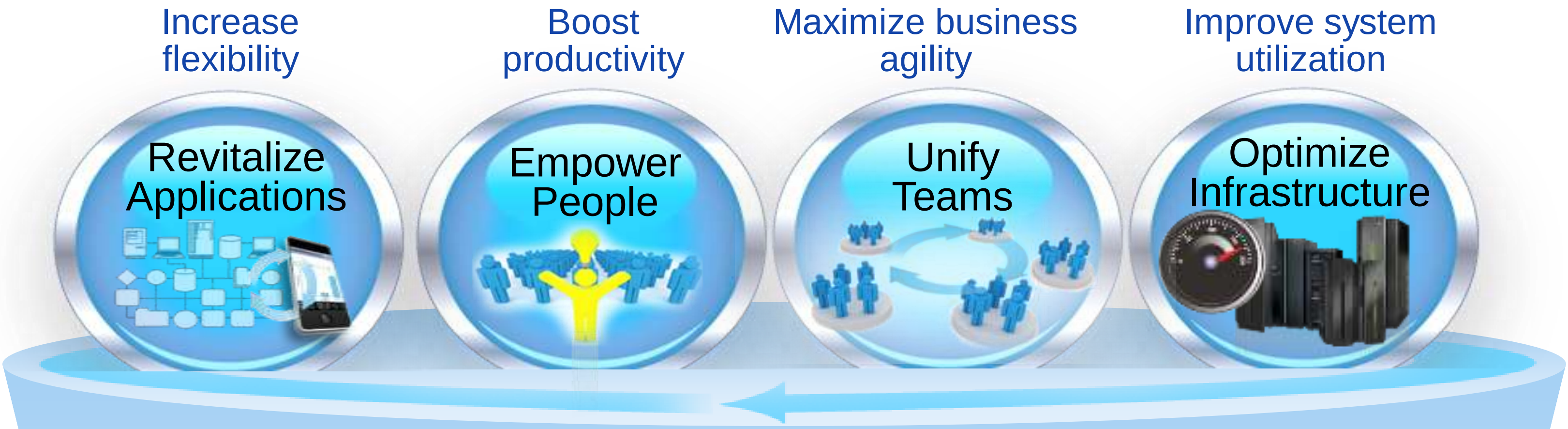
Poorly integrated teams

“We need to enable our teams to collaborate across platforms, languages, and environments.”

Infrastructure inefficiency

“We need a cost effective way to improve our infrastructure efficiency and free up capacity to handle more workload.”

Enterprise Modernization offers a low risk, high return approach



- Increase flexibility by revitalizing existing application portfolios
- Boost productivity and accelerate innovation with modern skills
- Maximize business agility by bridging organizational silos
- Improve system utilization by leveraging hardware capabilities

Jazz®

Business constraints with mainframe development today

Which limits the amount of System z production workload coming online

“Operations tell me it will take two months to get my test system allocated.”

“I can only test my batch applications in offline hours. Online apps consume the 9-5 cycles.”

“It is difficult for my developers to learn the mainframe. Operations controls can prevent experimentation by developers..”

“My development capacity charge-back is consuming my entire budget. I can’t spend on tools.”

“We don’t have the capital budget to obtain more mainframe test resources for my developers.”

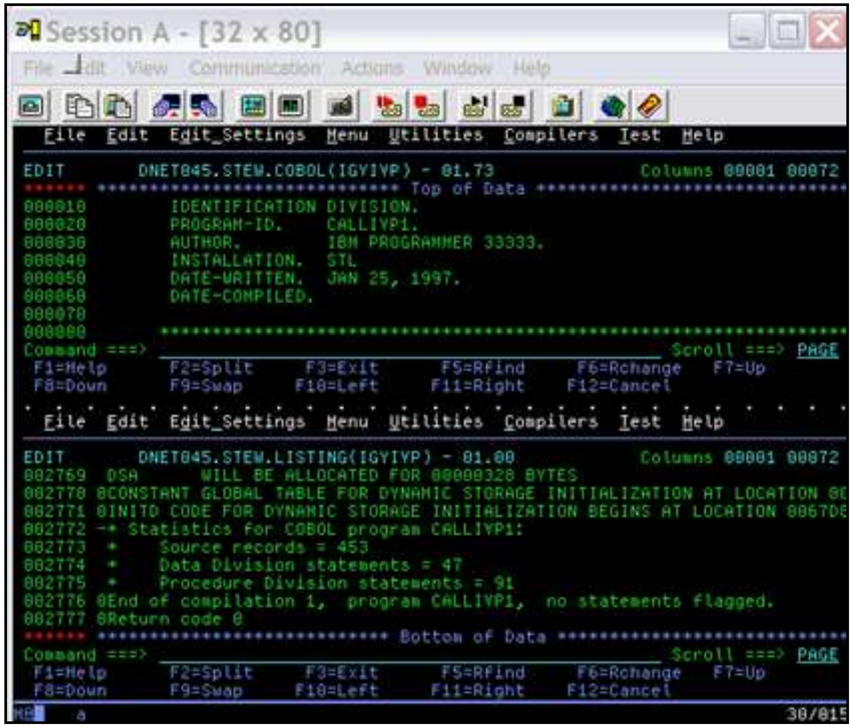
“I can’t even work on Mondays! Production workload kicks me off.”

“I want to try out creating Event Processing and ATOM apps, but my system isn’t scheduled for a CICS update till 2012.”

“The Mainframe isn’t cool anymore.”

Existing development environment

- Multiple screens
- Multiple disparate tools
- 20 x 80 characters of content



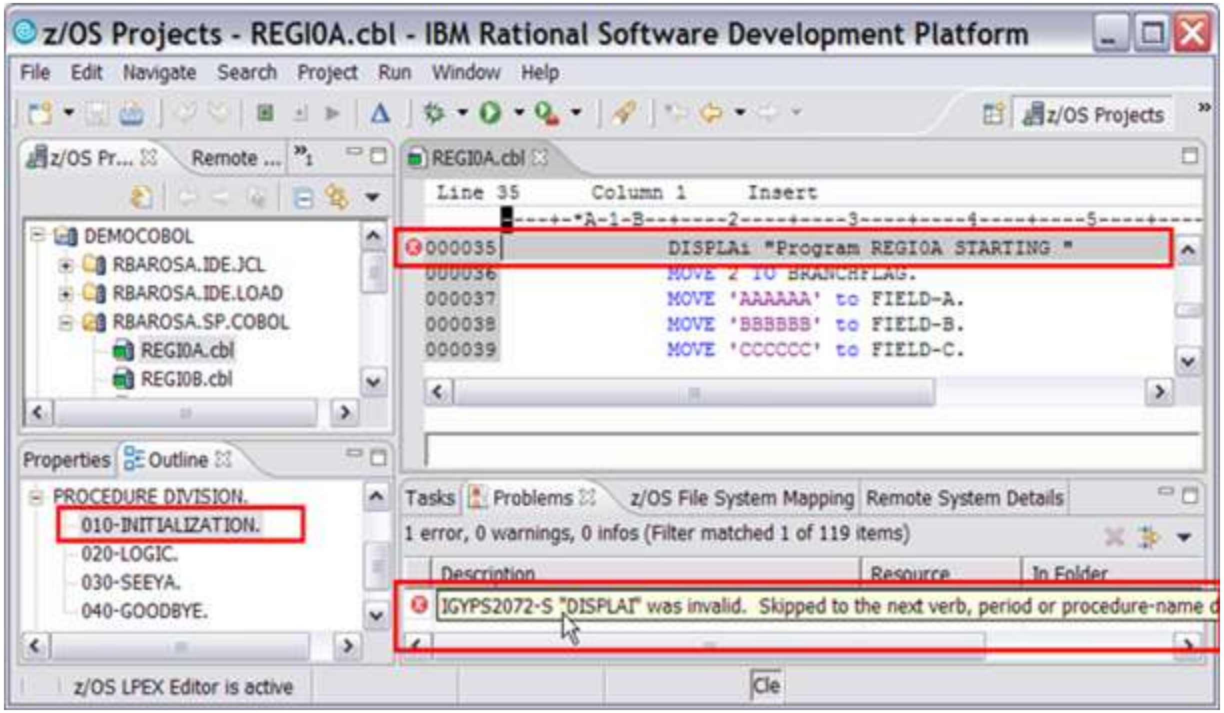
```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       DNET045.STEW.COBOL(IGYIYP) - 01.73              Columns 00001 00072
****** ***************************** Top of Data ******************************
000010        IDENTIFICATION DIVISION.
000020        PROGRAM-ID.    CALLIYP1.
000030        AUTHOR.        IBM PROGRAMMER 33333.
000040        INSTALLATION.  STL
000050        DATE-WRITTEN.  JAN 25, 1997.
000060        DATE-COMPILED.
000070
000080        *****************************************************************
Command ===> ________________________________________________ Scroll ===> PAGE
 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       DNET045.STEW.LISTING(IGYIYP) - 01.00            Columns 00001 00072
002769  DSA      WILL BE ALLOCATED FOR 00000328 BYTES
002770 0CONSTANT GLOBAL TABLE FOR DYNAMIC STORAGE INITIALIZATION AT LOCATION 00
002771 0INITD CODE FOR DYNAMIC STORAGE INITIALIZATION BEGINS AT LOCATION 0067DE
002772 -* Statistics for COBOL program CALLIYP1:
002773  *    Source records = 453
002774  *    Data Division statements = 47
002775  *    Procedure Division statements = 91
002776 0End of compilation 1,  program CALLIYP1,  no statements flagged.
002777 0Return code 0
****** **************************** Bottom of Data ****************************
Command ===> ________________________________________________ Scroll ===> PAGE
 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
MA     a                                                                30/015
```

Steps to validate a change

- Edit source
- Find code line
- Change code
- Exit source
- Find JCL
- Edit JCL
- Submit compile job
- Swap to SDSF
- Select job
- Find error message
- Swap to JCL
- Exit JCL
- Find source file
- <repeat>

Tomorrow's development environment

- Information at your fingertips
- Easy navigation
- Automatic feedback



```
Line 35       Column 1       Insert
        ----+-*A-1-B--+----2----+----3----+----4----+----5----+----
000035                DISPLAi "Program REGIOA STARTING "
000036                MOVE 2 TO BRANCHFLAG.
000037                MOVE 'AAAAAA' to FIELD-A.
000038                MOVE 'BBBBBB' to FIELD-B.
000039                MOVE 'CCCCCC' to FIELD-C.
```

1 error, 0 warnings, 0 infos (Filter matched 1 of 119 items)

IGYPS2072-S "DISPLAI" was invalid. Skipped to the next verb, period or procedure-name d

<u>Steps to validate a change</u>

- Edit source
- Find code line
- Change code w/feedback

Next NOW!

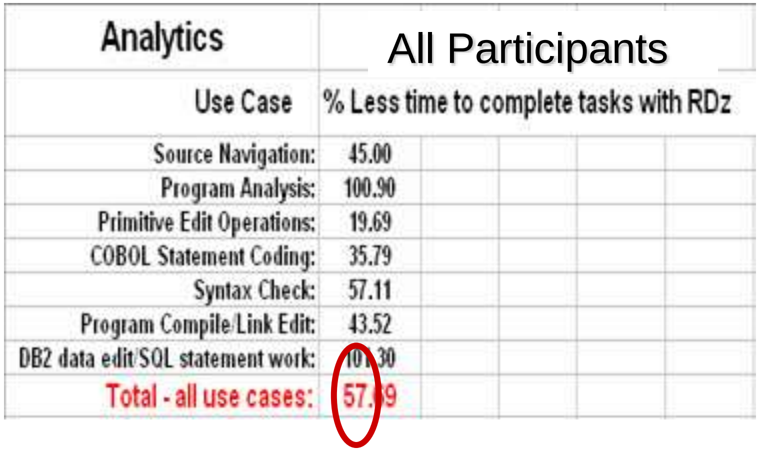
IDE Efficiency Benchmarks

- 100 common (daily) ISPF tasks used during maintenance and support assignments
  - ISPF workflow translated (click-for-click) to RDz development
  - Project participants believed they were trying to find gaps between RDz and ISPF functionality
- Apples-to-Apples and test scripts
- Mix of experienced (veteran) ISPF programmers and new-hire developers
- Assumption was that only new-hire developers would be more productive
  - This turned out to be false

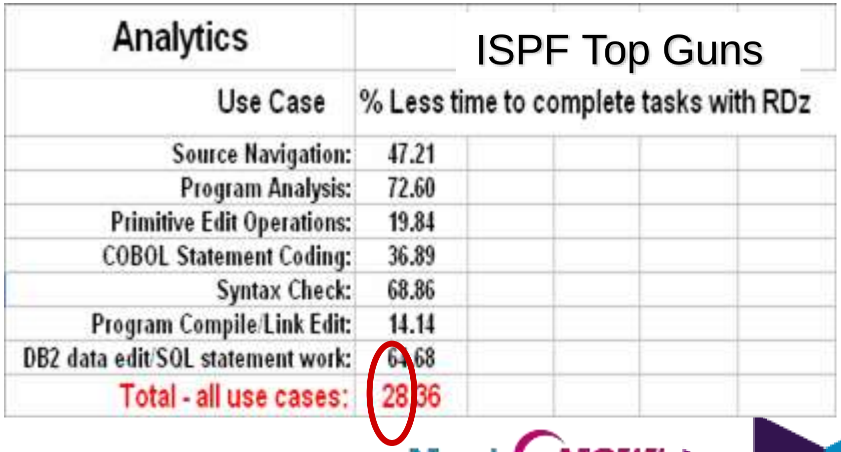
Based on IBM internal productivity study.

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.



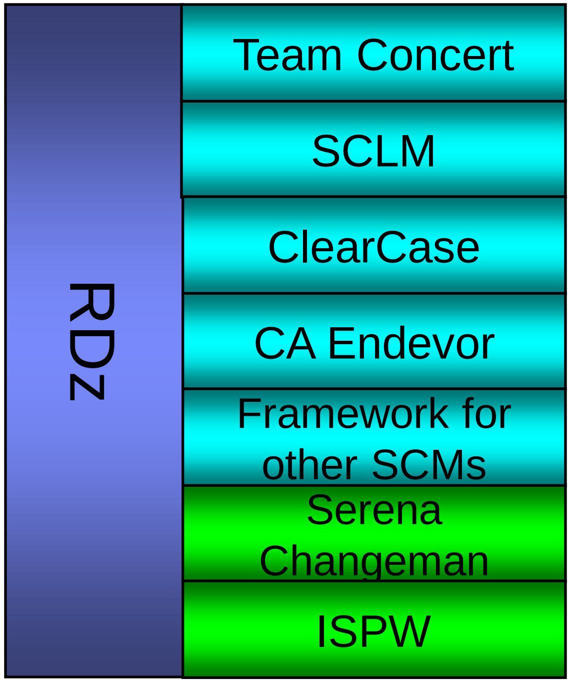
Analytics	All Participants
Use Case	% Less time to complete tasks with RDz
Source Navigation:	45.00
Program Analysis:	100.90
Primitive Edit Operations:	19.69
COBOL Statement Coding:	35.79
Syntax Check:	57.11
Program Compile/Link Edit:	43.52
DB2 data edit/SQL statement work:	101.30
Total - all use cases:	57.69



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:	64.68
Total - all use cases:	28.36

Access source code…

- Integrates with existing SCM systems
- Use existing promotion schemes
- Customize to your process



🟦 IBM Supplied

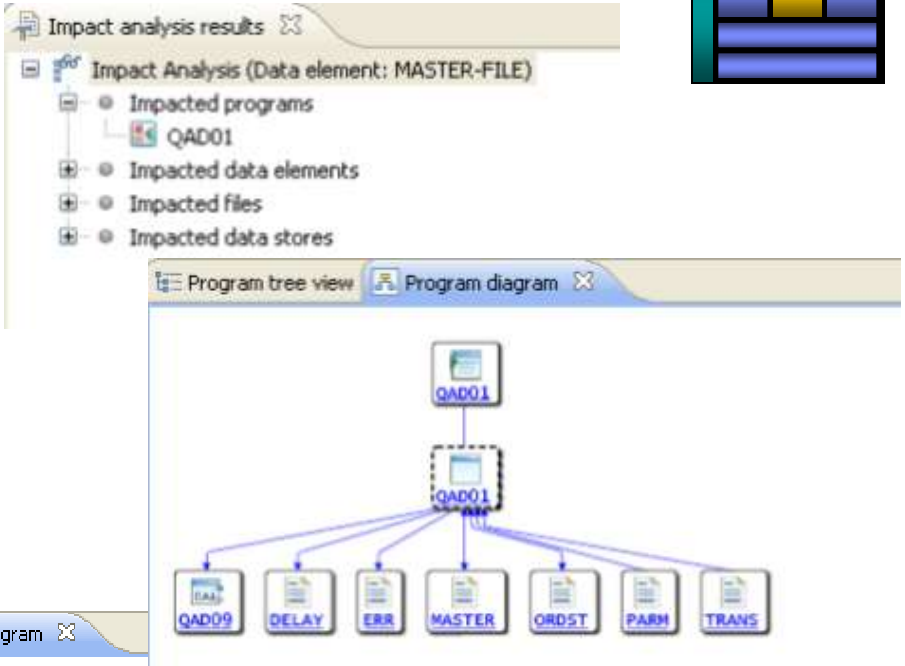
🟩 Vendor Supplied

Create enterprise services…

- Provide standardized access to application assets
- Work across platforms
- Create abstract interfaces for loose coupling
- Vary implementation without affecting consumers
- Ease application reuse

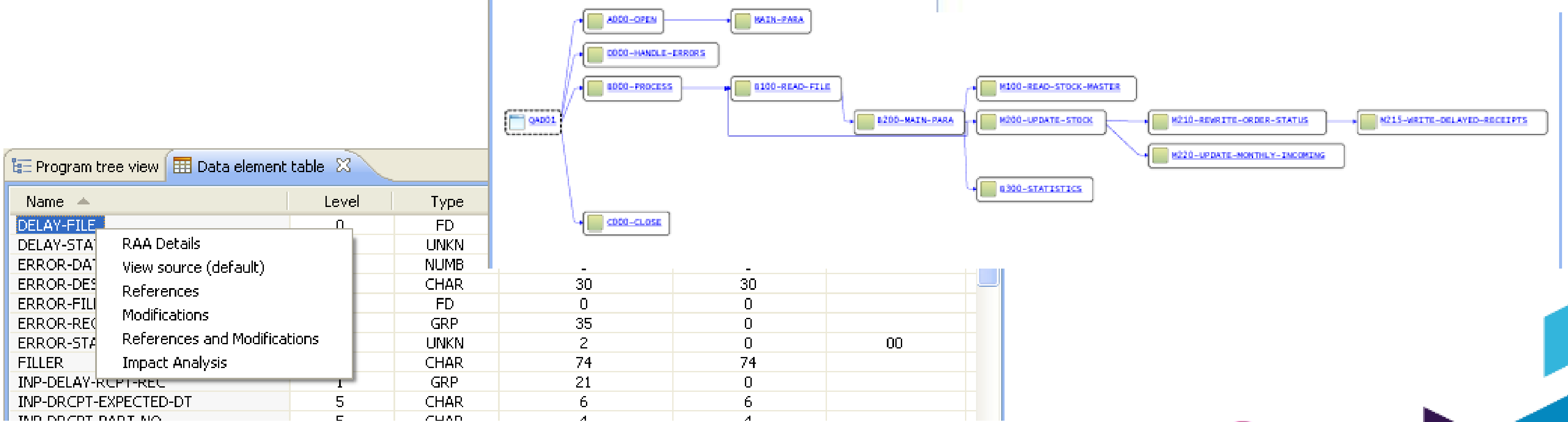
Analyze applications automatically

- Integrate application analysis into development
  - Link code to data and runtime resources
  - Visualize code structure and flow
- Understand the effect of changes
  - Run impact analysis on code changes to determine effected production modules
  - Size testing efforts and create workspaces for changes
  - Locate dead code



Impact analysis results

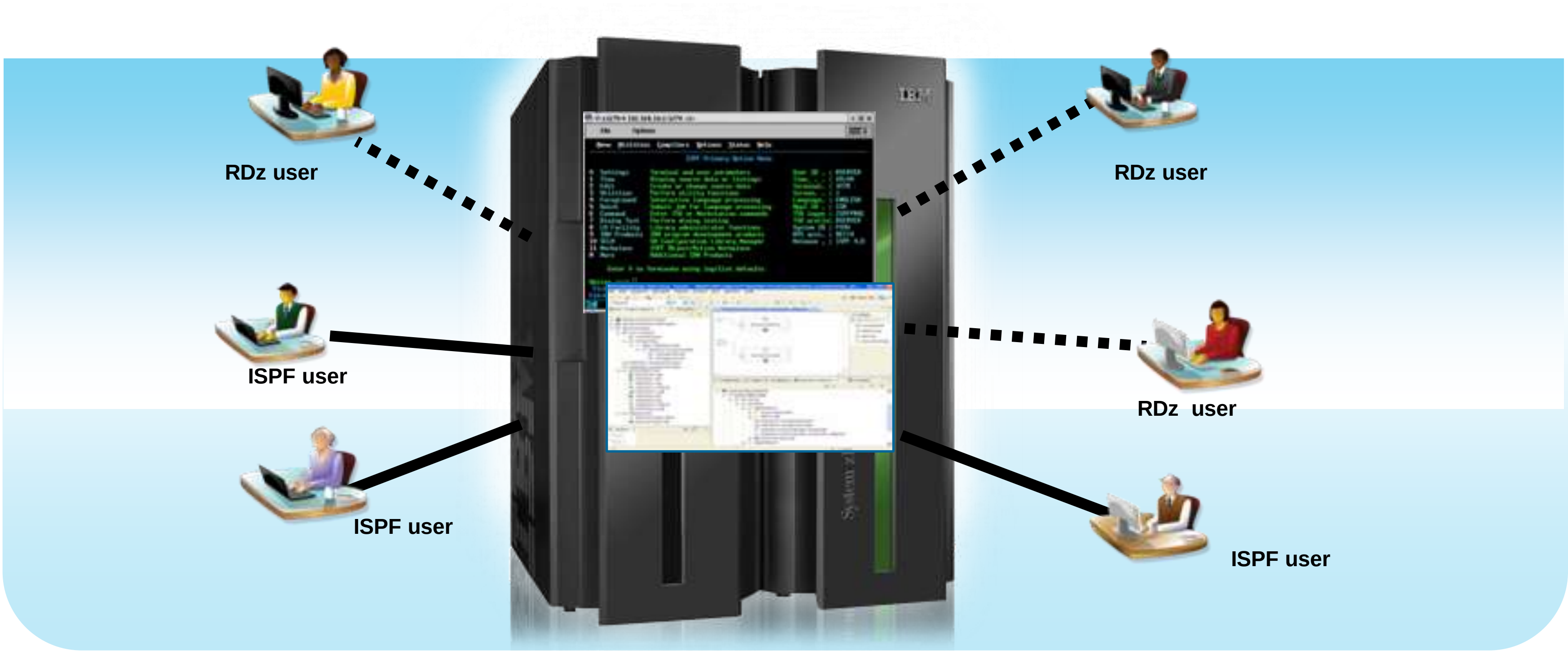
- Impact Analysis (Data element: MASTER-FILE)
  - Impacted programs
    - QAD01
  - Impacted data elements
  - Impacted files
  - Impacted data stores



Name	Level	Type			
DELAY-FILE	0	FD			
DELAY-STA		UNKN			
ERROR-DA		NUMB	-	-	
ERROR-DE		CHAR	30	30	
ERROR-FIL		FD	0	0	
ERROR-REC		GRP	35	0	
ERROR-STA		UNKN	2	0	00
FILLER		CHAR	74	74	
INP-DELAY-RCPT-REC	1	GRP	21	0	
INP-DRCPT-EXPECTED-DT	5	CHAR	6	6	
INP-DRCPT-PART-NO	5	CHAR	4	4	

Mainframe development environment with RDz and ISPF

RDz provides relief…but customers want more



Modern development environments add value:

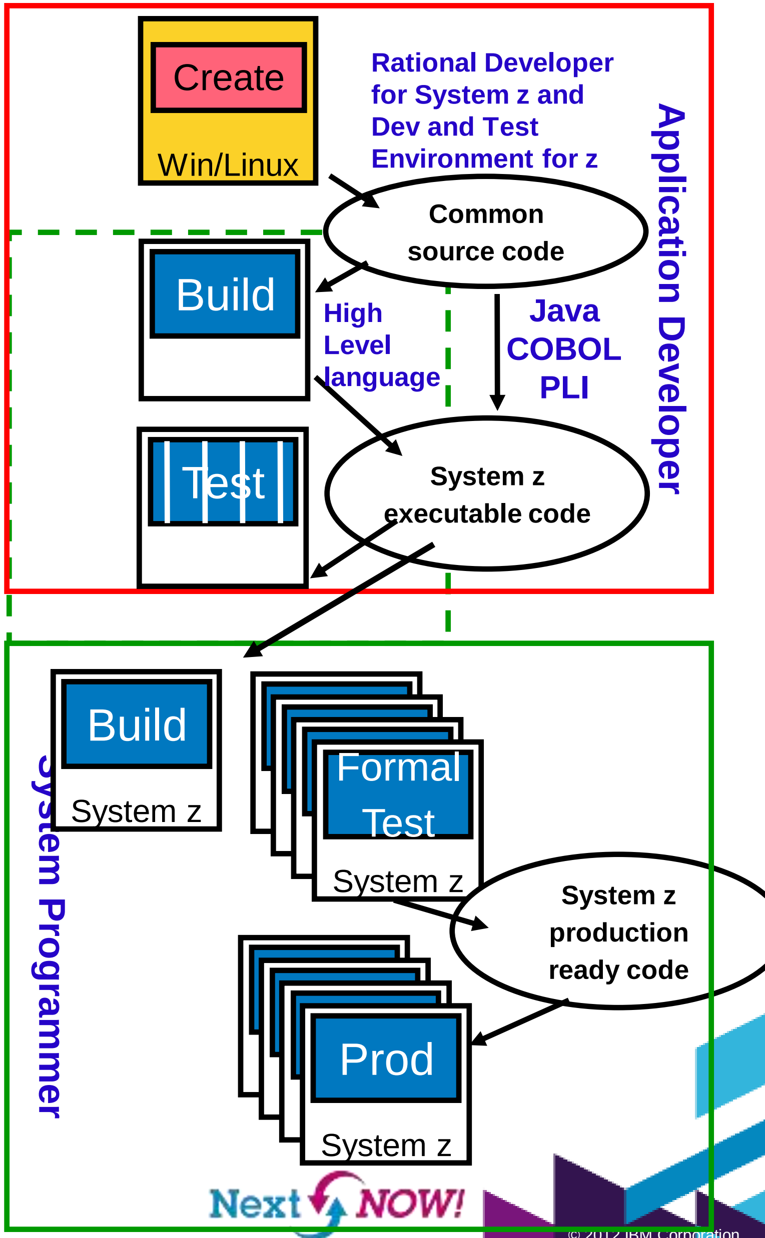
- Productivity enhancing tooling; more attractive tooling for new developers
- Ability to offload some development MIPS usage
- Integration with complete application lifecycle tools

But challenges remain:

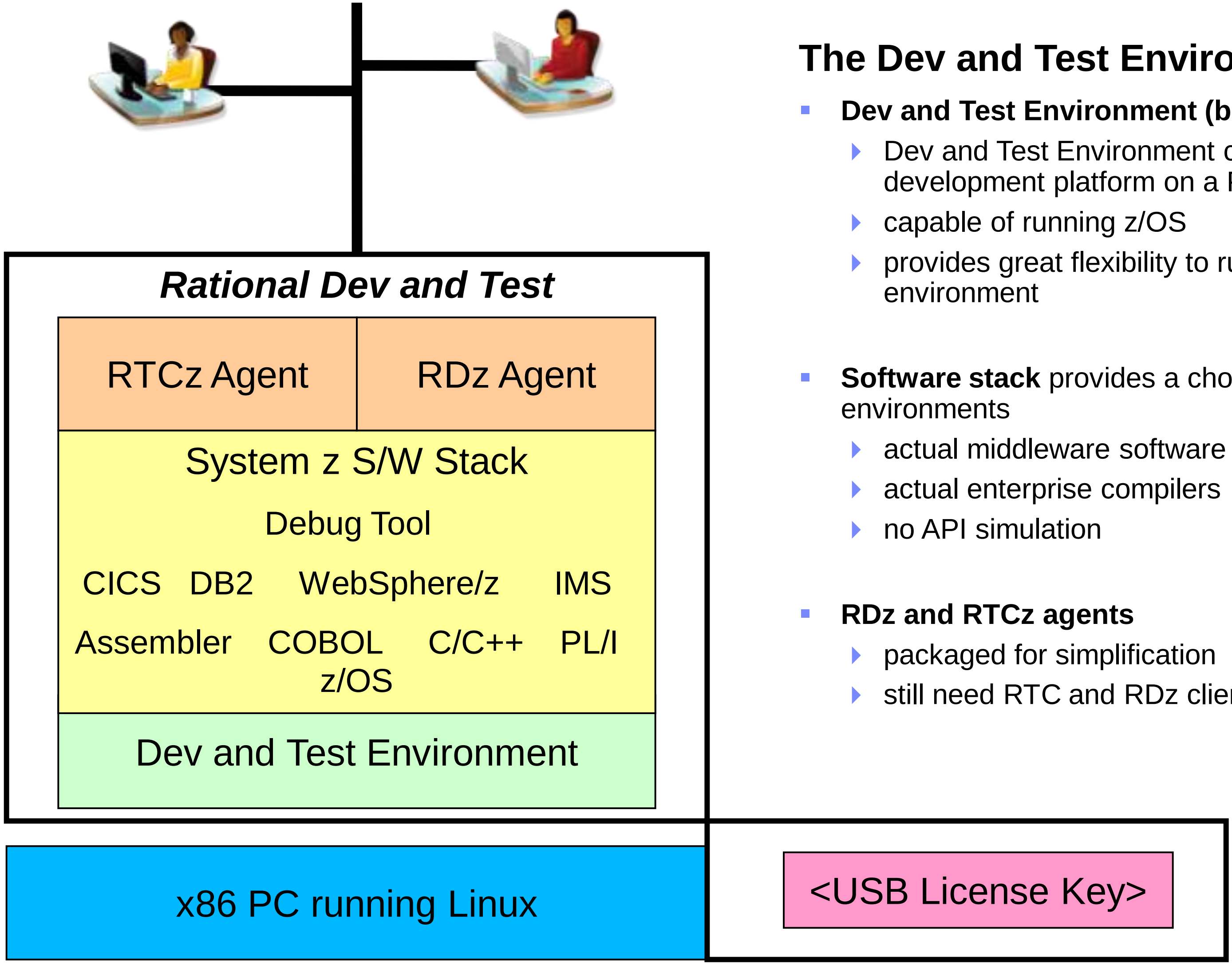
- Business pressures to manage mainframe MIPS usage for development
- Unit test delays caused by dependencies on operations team

Introducing the RD&T Environment

- Provides a local test environment for unit testing System z applications
  - – unit test processing on Linux versus mainframe
- Can enable companies obtain lower cost for development and unit testing
  - – Gives flexibility to Developers/Teams to accomplish unit tests on the mainframe or off
  - – Provides a local System z development/unit test environment to simplify development investments
  - – Frees up more mainframe capacity to production workload usage
- Strengthens development processes through using actual compilers and runtimes in RD&T
  - Use compilers for “true” syntax check/compile using Enterprise COBOL/PLI
  - Use co-processors for CICS/DB2/Custom to cut down on re-implementation efforts on local platform
- Provides flexibility for system programming staff and development
  - System programmers can define common test images for development teams for some centralization of control, but still provide developers with ability to make changes



RD&T Offering Description



Rational Dev and Test

RTCz Agent | RDz Agent

System z S/W Stack

Debug Tool

CICS DB2 WebSphere/z IMS

Assembler COBOL C/C++ PL/I

z/OS

Dev and Test Environment

x86 PC running Linux

<USB License Key>

The Dev and Test Environment consists of:

- **Dev and Test Environment (based on zPDT)**
  - Dev and Test Environment can provide a System z development platform on a PC
  - capable of running z/OS
  - provides great flexibility to run a customized environment
- **Software stack** provides a choice of IBM middleware test environments
  - actual middleware software (including z/OS)
  - actual enterprise compilers
  - no API simulation
- **RDz and RTCz agents**
  - packaged for simplification
  - still need RTC and RDz client license(s) to activate

RD&T limitations

- **The RD&T environment does NOT support all System z function, such as:**
  - Physical Parallel, ESCON®, FCP, FICON® and High Performance FICON channels
  - Coupling links and coupling facilities
  - List-directed IPL
  - External Time Reference (ETR)
  - Server Time Protocol (STP)
  - MIDAWs
  - Logical channel subsystems
  - HiperSockets™
  - Multiple I/O paths per device
  - Not all CHSC functions are supported
  - Some IBM System z Crypto Express2
  - Some IBM 3088 CTC device

- **RD&T does not produce an environment equal to a larger System z.**
  - Some aspects of a larger system are unlikely to be met in any very small environment.
    - Inability to verify and enhance the scalability of a program
    - Inability to run application programs that require hundreds of MIPS.
  - A UT system is not recommended for very fine-level performance tuning that is sensitive to memory location, cache functions, and pipeline optimization.
  - In addition, the UT platform does not nearly have the same quality of service as does a mainframe in terms of availability and connectivity.

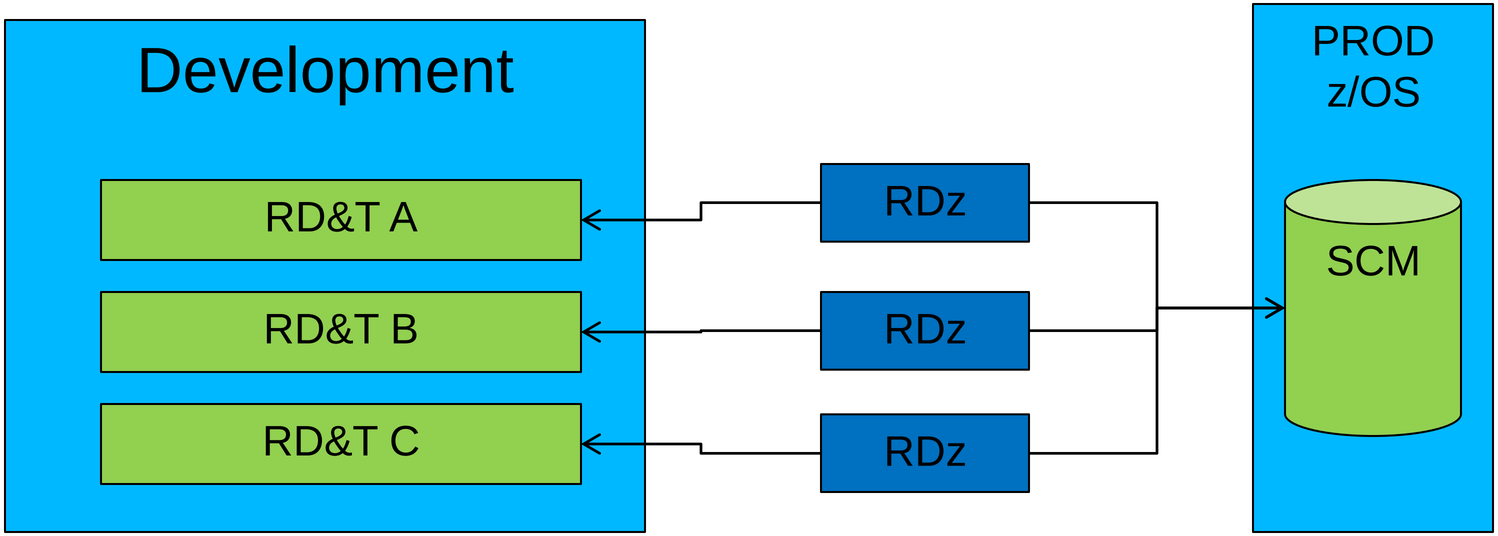
- **Anyone needing any of the function outlined above should consider a traditional System z server.**

How will RD&T help me?

- Are you interested in testing your application changes on your PC/off-mainframe?
  - RD&T provides a x86-based environment where you can test z/OS applications on a desktop or server machine
- Are your development systems overloaded?
  - E.g., My development system is so slow! It takes 1 hour to compile!!
  - E.g., There are not enough resources available for development
  - RD&T provides a low cost mechanism to create additional development capacity without mainframe HW upgrades.
- Do you have difficulty testing application changes? Is too difficult or takes too much time to make changes to your test environment? Are there too many requests for unique test environments than your ops team can fulfill (catastrophic event testing, system failure, impact analysis, etc)?
  - RD&T provides a separate test environment under the development team’s control allowing quick environment changes to be performed by development. The UT environment can be assigned to a small teams for unique test environments allowing more robust testing
- Are you being asked to use fewer MIPS for development
  - RD&T provides a 0-MIPS development environment, allowing additional development resources to be easily allocated

Customer example

Large US Financial Customer - Implementation



- Requirement
  - Provide more responsive System z access for application developers
  - Reduce application software delivery through faster test cycle
  - Provide each application group their own unit test facility
- RD&T Solution
  - Application developers have use of a uniquely defined RD&T feature for System z access
  - RD&T provides a unique test environment for different application development groups
  - Supports multiple RD&T development environments on Intel blade to reduce server hardware
  - Has reduced application development test time

Top 5 Questions

1) What is the maximum number of developers a RD&T server can support?

This can vary depending on the underlying hardware and workload being tested. Desktop/Laptops can typically support 3-5 users. Low end server class machines can support 15-30 users. At this time, IBM has not provided a detailed analysis of RD&T server sizing.

2) How can I get test data for use in RD&T?

Customers can use existing tools like IDCAMS, DB2 utilities, etc. to extract test data and then download it to the RD&T machine. RD&T provides a volume duplication utility that can download an entire 3390 data volume as needed into the UT environment. Facilities such as CICS remote function shipping or DB2 Remote Data Access can also be utilized.

3) Can I run other levels/backlevels of the middleware provided?

Yes, based on existing mainframe entitlement the request can be evaluated and some requests may not be approved. Requires special licensing dispensation from IBM, not automatically entitled with purchase of RD&T.

4) Can I use other IBM tools in the RD&T environment?

Yes, based on existing mainframe entitlement the request can be evaluated and some requests may not be approved. Requires special licensing dispensation from IBM, not automatically entitled with purchase of RD&T.

5) Can I run third party software?

Yes, if the third party license allows this. Customers must work with their software vendor to determine licensing considerations.

Additional Questions

6) Does this require system programming skills?

- Vanilla configuration ready-to-go out of the box
- z/OS does require system programming skills to set up the development and/or testing environments if users want the UT environment to mirror an existing host configuration.
- You can usually set up one box/image and transfer the configurations to another box easily (VMWare style).

7) What about security?

- RACF is installed, but with minimal configuration.
- The sample configuration guide has suggestions for basic security.
- Security is a site choice. The ability to customize z/OS on a platform designed for individuals or small teams may:
  - Provide better testing opportunities
  - Provide customization for individual productivity gains
  - Provide opportunities to learn about z/OS fundamentals

RD&T host machine specifications

- Underlying Linux host
  - Red Hat Enterprise Linux 6.X (RHEL 6.X)
  - OpenSUSE, SLES 11.2
  - Note: IBM does not support installation on other distributions.

- Base machine must have:
  - at least 3 GB of real memory:
    - 1 GB is required for the 64-bit Red Hat or openSUSE Linux
    - 1-2 GB is required for the System z operating system, depending on the size of the development system
  - at least 80 GB free disk space available after Linux
  - Note: RD&T has been tested on:
    - Lenovo ThinkPad W Series
    - IBM System x 3500 M1, 3500 M2, 3650 M1, and 3650 M2

Summary - Benefits of the RD&T Feature

- Increased application quality using the included IBM runtimes for testing. Provides a high fidelity testing environment.
  - Functionality and services more accurately reflect the mainframe
  - Using actual z/OS middleware means less retesting and rework is required when moving from the unit test environment to the quality assurance or pre-production environment.
  - Developers have an isolated test environment to test application changes that can then be easily merged into the next level of testing
- Deployment of the Unit Test feature on a PC lowers development and unit tests costs and allows MIPS to be reallocated for production use.
  - Executes on an x86 Server (see backup charts for details)
  - Utilizing zero development MIPS on the production mainframe for initial application change testing
  - Frees up additional capacity for new workload while reducing line of business development costs and chargeback.
- Provides developers in a single or shared user configuration with increased flexibility and control of the test environment, allowing them to be more productive and improving application delivery times.
  - Can be assigned to a single developer in a laptop configuration, or can support small-scale team environments
  - Environments can be tailored to a single developer or team's runtime needs without altering mainframe testing environments.
  - Can provide a greater level of control for developers to implement quick environment changes without having to involve production operations staff.
  - Developers can perform their first series of tests and regression testing without worrying about causing unexpected errors.

Smarter Testing and Delivery for System z

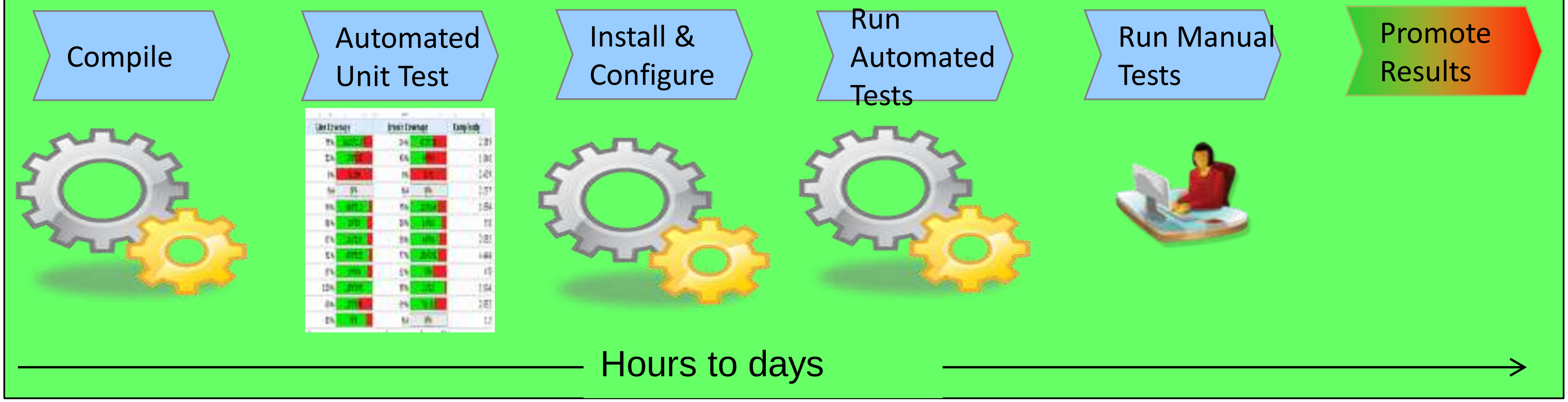
Improve Application Quality
Decrease Delivery Time
Refine Risk Assessment

Customer Comments

- Customer 1 - It takes us 5 to 6 weeks to complete z/OS application testing
  - Over 1000 test cases to run
  - Manual test effort because there are no z/OS automated test tools
  - No end-to-end testing (one tool that does it all)
- Customer 2 – We can not respond to regulatory changes
  - Test cycle takes a minimum of 12 weeks
  - Competition between development teams for testing resources
  - Build and maintain their own test tools. Manual operation.
  - Long batch test runs
- Customer 3
  - Have major z/OS application resource constraints that results in long test cycle
  - Off-Shore development and testing requirements
  - Major application failed due to no functional and performance testing
- **Many Accounts** – “Sorry, we have to cancel our meeting, today, because an application has stopped working”

Testing and Delivery – where are customers today?

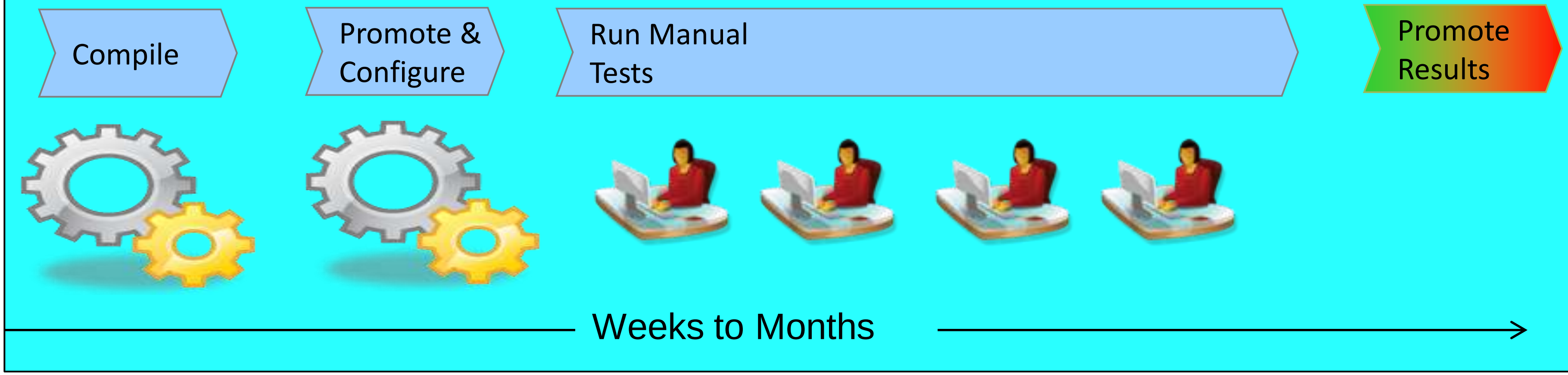
Java / .Net teams



Compile → Automated Unit Test → Install & Configure → Run Automated Tests → Run Manual Tests → Promote Results

Hours to days

Mainframe teams current state **



Compile → Promote & Configure → Run Manual Tests → Promote Results

Weeks to Months

** Feedback from mainframe customers

Cost is a significant driver

80% of development costs are spent identifying and correcting defects!*

During the **CODING** phase

$80/defect

During the **BUILD** phase

$240/defect

During the **QA/TESTING** phase

$960/defect

Once released as a product

$7,600/defect

+

Law suits, loss of customer trust, damage to brand

If admitted or not most development LPARs are managed as if starting here

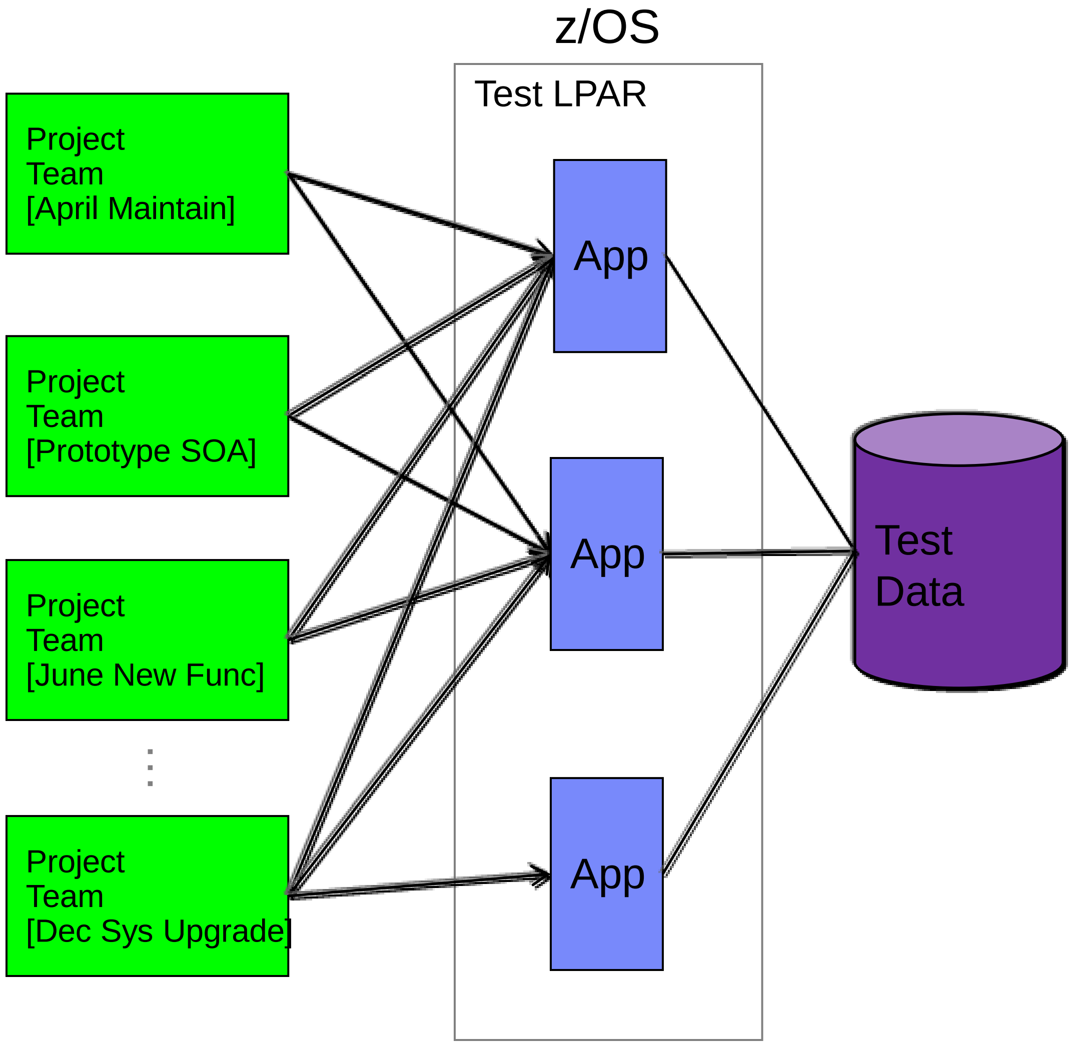
*National Institute of Standards & Technology

Source: GBS Industry standard study

st derived in assuming it takes 8 hrs to find, fix and repair a defect when found in code and unit test. Defect FFR cost for other phases calculated by using the multiplier on a blended rate of $80/hr.

Typical z/OS Testing Architecture

Organized by project team, vertically scaled, sharing resources, limited automation


z/OS
Test LPAR
Project Team [April Maintain]
Project Team [Prototype SOA]
Project Team [June New Func]
⋮
Project Team [Dec Sys Upgrade]
App
App
App
Test Data

Problems Encountered

1. Shared resources combined with overlapping schedules can elicit conflicts, impede innovation and slow code delivery

2. Coordination of environmental changes and releases cause bottlenecks, delays and additional overhead

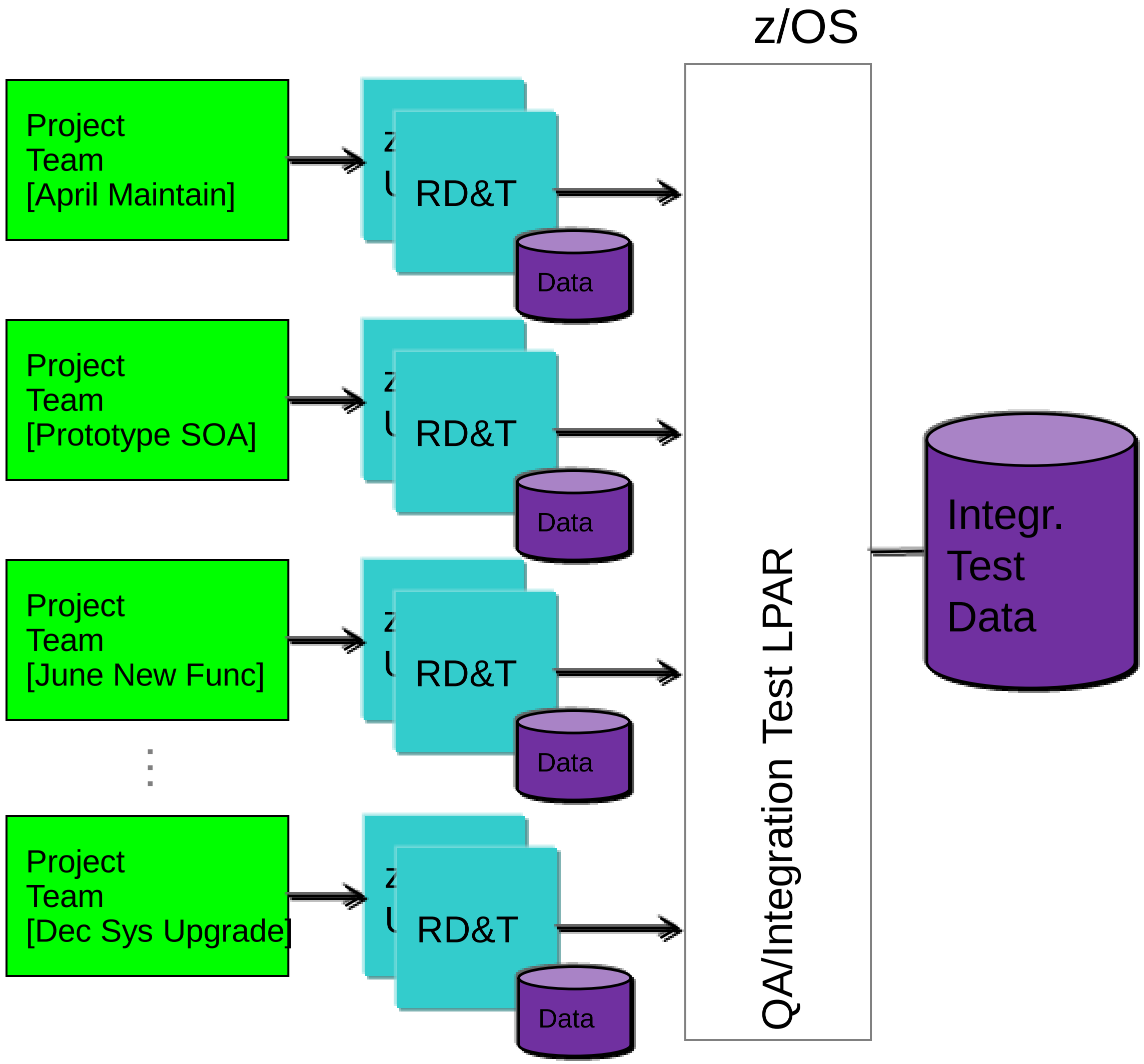
3. Shared test data is difficult to manage and can lead to over testing or incorrect test results

Smarter Testing Strategy

- Increase availability of z/OS testing environment and resources
- Improve quality and lower risk via automation, measurement, and collaboration*
- Focus on what is required for the change at hand, then scale

Testing Organized for Flexibility and Quick Delivery

Organized by application team, horizontally sliced, dedicated resources, highly automated


z/OS
Project Team [April Maintain]
Project Team [Prototype SOA]
Project Team [June New Func]
⋮
Project Team [Dec Sys Upgrade]
z U
RD&T
Data
QA/Integration Test LPAR
Integr. Test Data

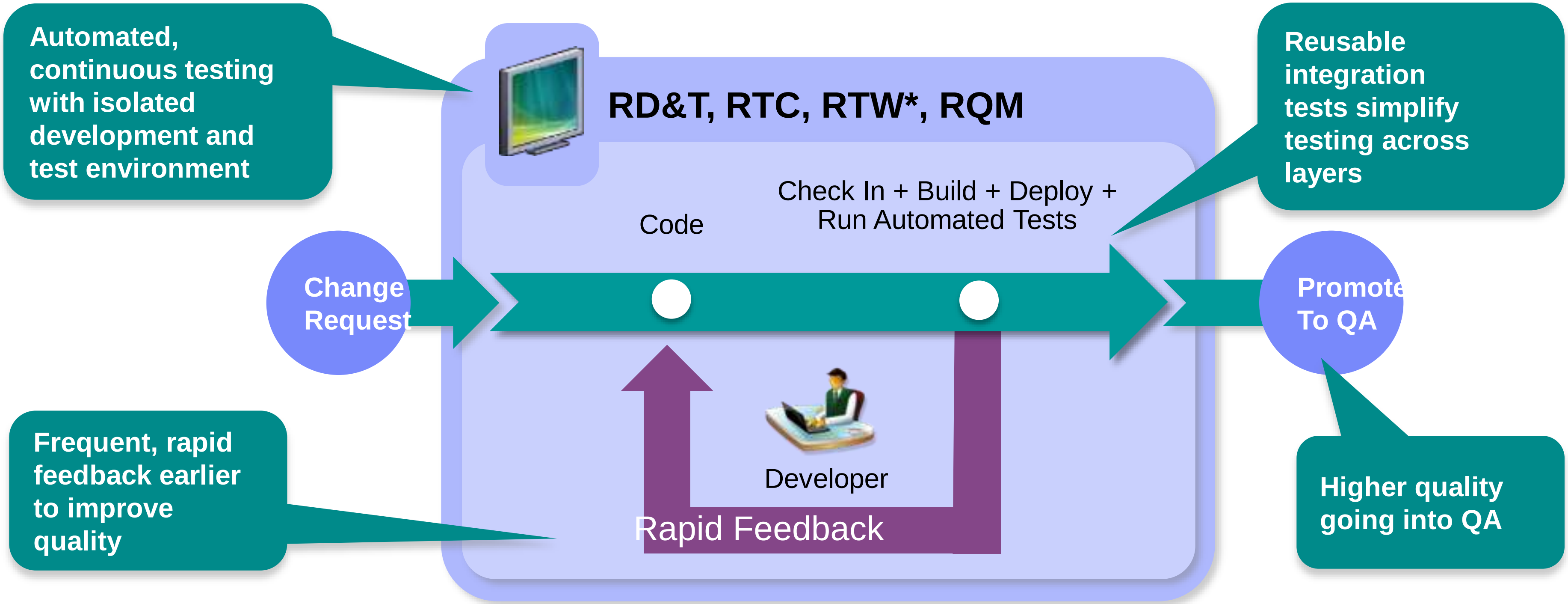
Problems Encountered

1. S [illegible] ined with overla [illegible] g schedu [illegible] can elicit c [illegible] ts, [illegible] de innov [illegible] nd slow [illegible] very
2. Coordination of environmental changes and releases cause bottlenecks, delays and additional overhead
3. S [illegible] d test da [illegible] difficu [illegible] manage a [illegible] can lead to [illegible] testi [illegible] g o [illegible] sults
4. Provisioning, managing, and synchronizing project test environments including data

Next NOW!

IBM Continuous Integration Solution for System z

Reduced delivery time, end-to-end visibility of test activities, safer and faster V2V migrations


Automated, continuous testing with isolated development and test environment
RD&T, RTC, RTW*, RQM
Reusable integration tests simplify testing across layers
Code
Check In + Build + Deploy + Run Automated Tests
Change Request
Promote To QA
Developer
Rapid Feedback
Frequent, rapid feedback earlier to improve quality
Higher quality going into QA

- Rational Team Concert 4.0
- Rational Quality Manager 4.0
- Rational Development and Test Environment for System z 8.5 NEW!
- Rational Testing Workbench powered by Green Hat Technology

Continuous Integration and Typical Mainframe Environments

- Expedite feedback to developers on application quality

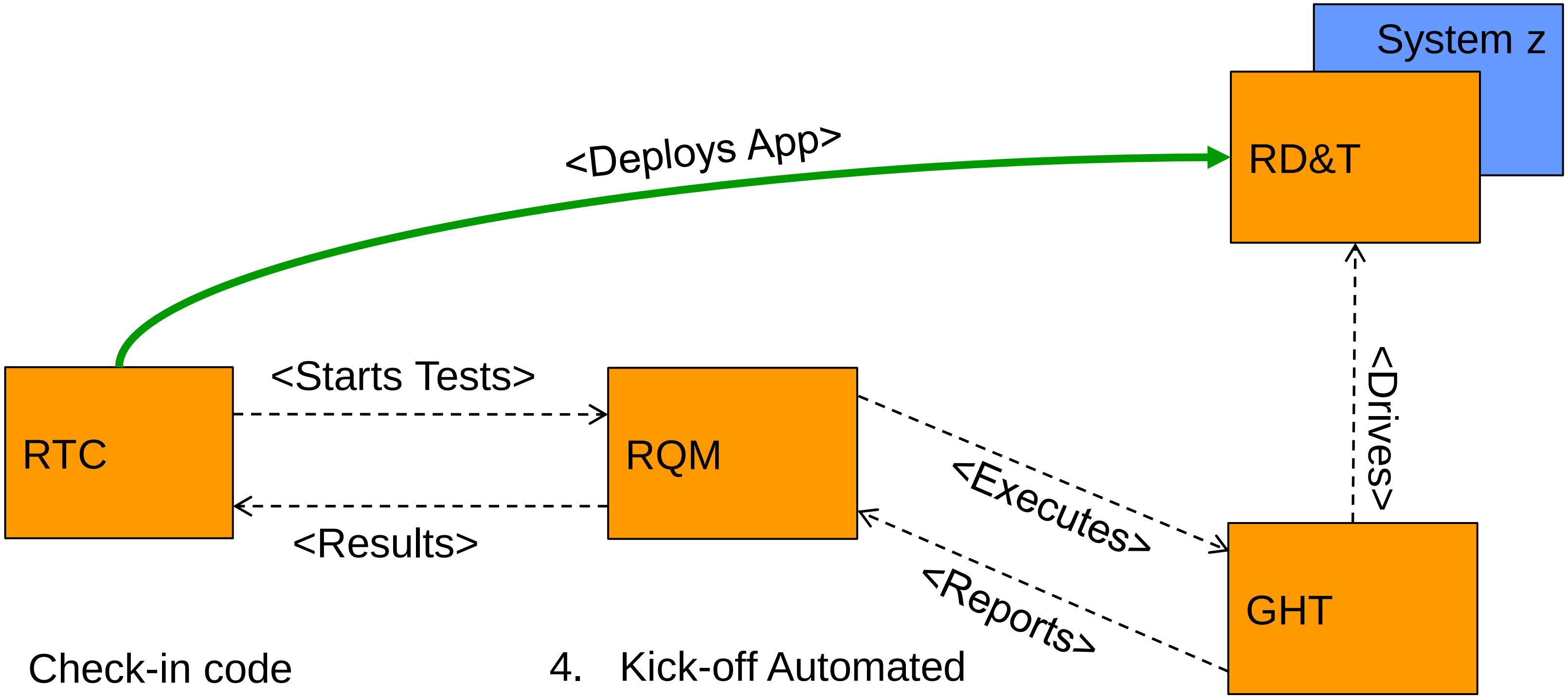
Principles of Continuous Integration

- ✓ Maintain a code repository
- ✓ Automate the build
- ✗ Make the build self-testing
- ? Commit to the baseline every day
- ? Every commit should be built
- ? Keep the build fast
- ✓ Test in a clone of production
- ? Make it easy to get latest deliverables
- ? Everyone can see the latest build results
- ✓ Automate deployment

Solution Components

- Rational Team Concert V4
  - Used for build and deployment
- Rational Quality Manager V4
  - Used for test case management and automated testing
- Rational Development and Test Environments for System z V8.5
  - Used to deploy application onto and run test cases
- Greenhat Tester (Rational Testing Workbench)
  - Used to test the web service interface of the z/OS application

Detailed Continuous Integration for System z Scenario



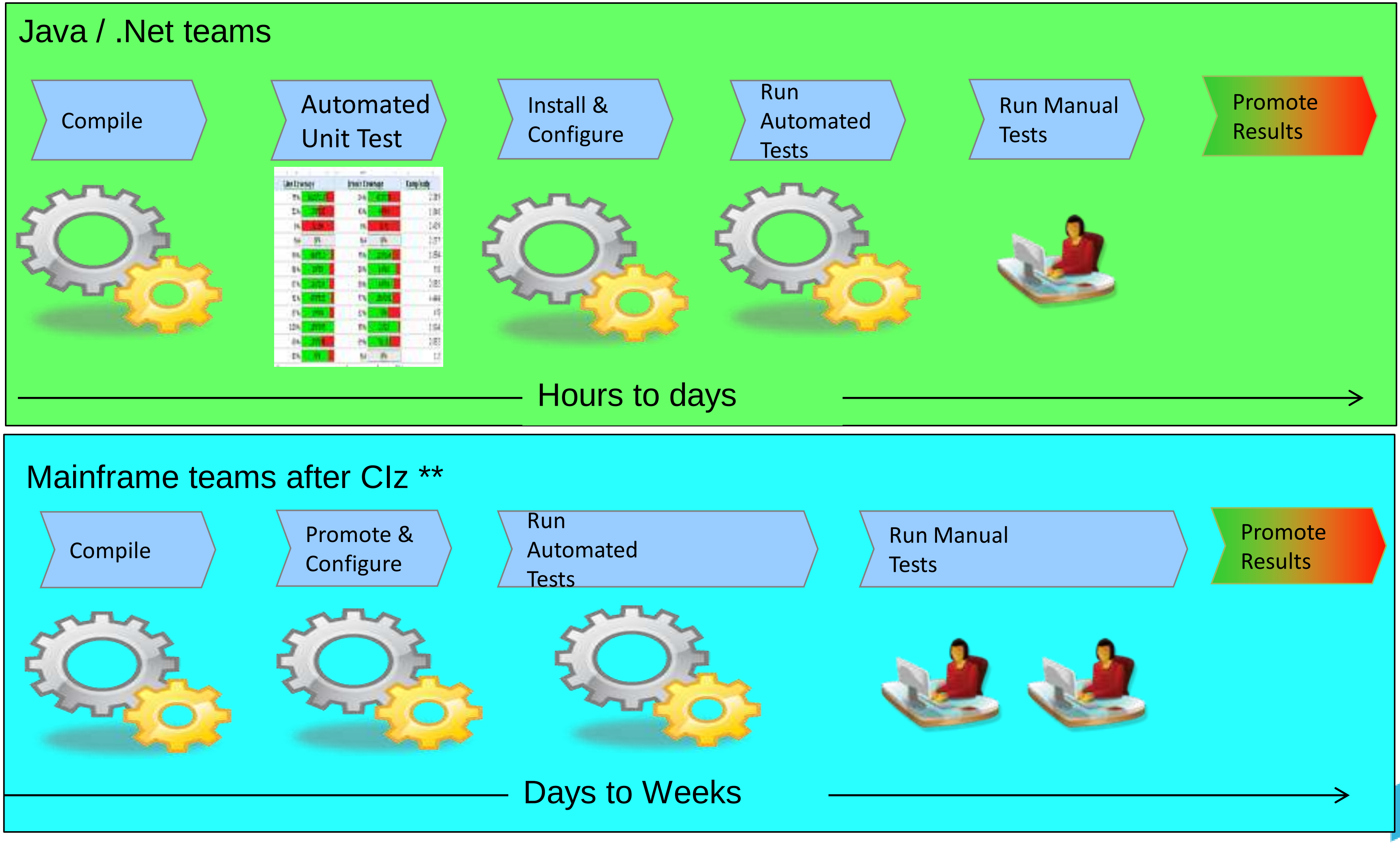
1. Check-in code
2. Build code
3. Deploy build results to RD&T

4. Kick-off Automated Test Plan

5. Run automated interface tests against RD&T or System z
6. Mark RQM execution records Pass/Fail

7. Report test results in dashboard/build results/defect records in RTC.

Testing and Delivery – moving one step forward



Java / .Net teams

Compile

Automated Unit Test

Install & Configure

Run Automated Tests

Run Manual Tests

Promote Results

Hours to days

Mainframe teams after CIz **

Compile

Promote & Configure

Run Automated Tests

Run Manual Tests

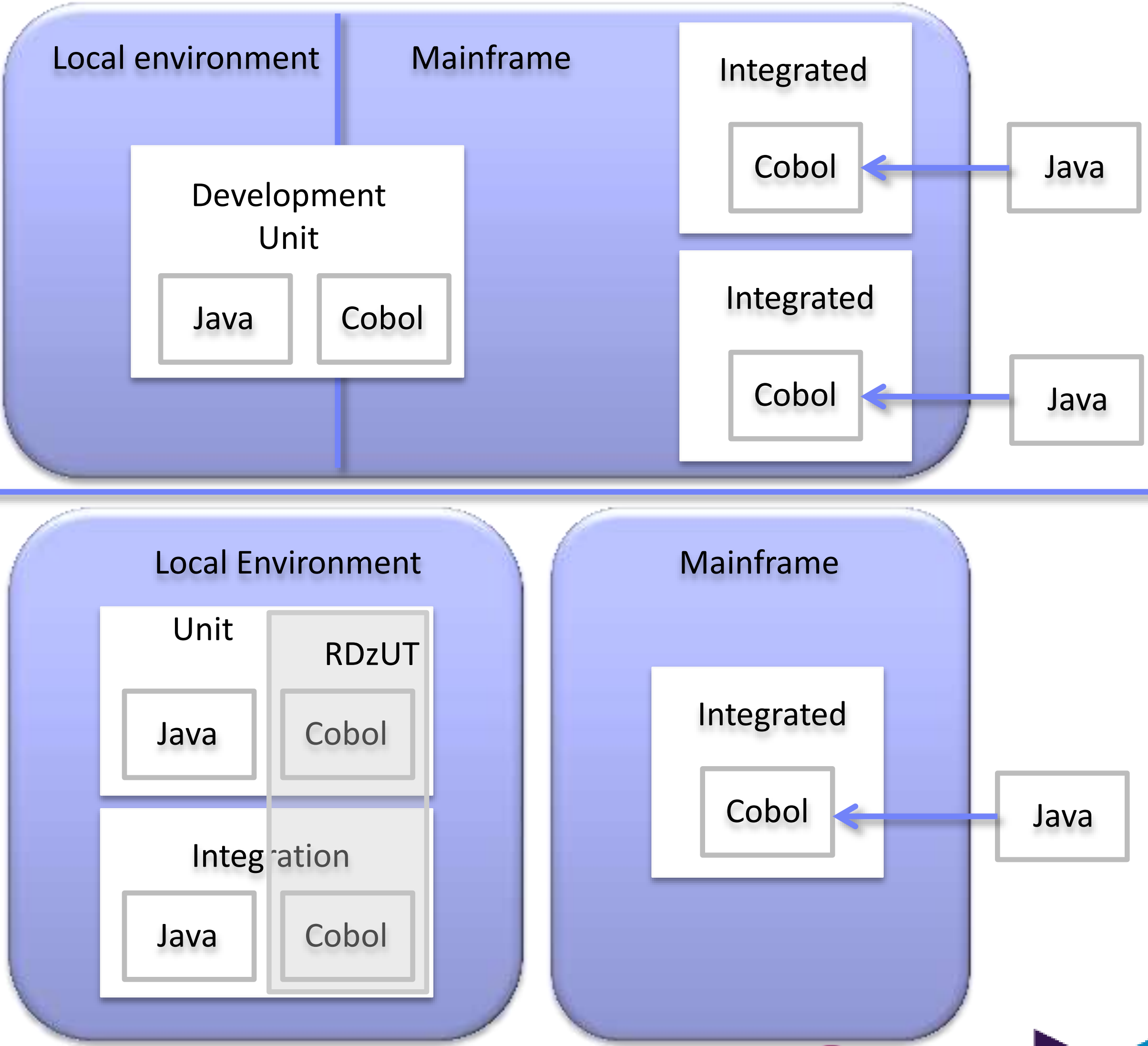
Promote Results

Days to Weeks

Development Environments

Current Model

Proposed Model



Agenda

- //1 Who is ASIST
- //2 Modern Development for zEnterprise
- //3 Smarter Testing and Delivery for System z
- //4 Conclusion and Q&A

Conclusion

SEEING IS BELIEVING

- □ RDz and RD&T Proof of Technologies (POT)
  - ○ Fill out your form
  - ○ Public or Private POT
  - ○ POT’s are free of charge
- □ Offering
  - ○ Customer side Proof of Concept (POC)
  - ○ Prepare a business case based on your current TCO
  - ○ Implementation of the software and technology
  - ○ Tailor-made integration in your current environment
  - ○ Creation of a “Cookbook” = standards- and usage manual
  - ○ Aid for creating your RD&T “appliances”
  - ○ Prepare a plan for continues Testing and Deployment implementation
  - ○ Long term relationship

INTEGRATION SOLUTIONS

INSOURCING SERVICES

Next NOW!

धन्यवाद

Hindi

多謝

Traditional Chinese

ขอบคุณ

Thai

Спасибо

Russian

Thank You

English

Bedankt

Nederlands

شكراً

Arabic

Merci

French

Obrigado

Brazilian Portuguese

Gracias!

Spanish

多谢

Simplified Chinese

Danke

German

நன்றி

Tamil

ありがとうございました

Japanese

감사합니다