The DevOps approach: Develop and test – quality management in the software development lifecycle

Speaker Name and Title

Overview of Continuous Integration for System z (CIz) a DevOps Solution

Session #4

The continuous integration testing process consists of continuously compiling, inspecting, deploying, and testing source code changes. This improves the likelihood of finding defects earlier in the development cycle, when they can be more easily managed, and it reduces the number of late-stage integration defects that cause development delays and much rework. In many continuous integration environments, this means running a new build whenever code in a source code management repository changes. We'll demonstrate how easy an automated continuous testing can be in the mainframe environment.

Agenda

- Why Do Enterprise Modernization
- The Cost Reasons for CIz and Testing
- What is Agile Development, Continuous Integration and DevOps
- System z CIz Application Testing Solution
  - Increase Test Resource using RD&T
  - Automated Testing using RQM/RTW
- How Virtualization Can Help
- Demo
  - System Overview
  - Required Products
  - System Flow
  - Demo
- Questions

Why Enterprise Modernization for System z

- z/OS Enterprise Modernization Application Development Objective
  - To create faster, better, and cheaper software
    - Faster – to reliably reduce software delivery time from requirements to production
    - Better – software with reduce defect rate
    - Cheaper – reduce application development cost. This includes:
      - Analysis, developer, tester productivity
      - MIP reductions
  - This is the “First Commandment” of Enterprise Modernization Tooling
  - This objective can be measured

- Questions
  - What does each % reduction in delivery time mean to the business?
  - What does fewer defects mean to the business?

The Cost Reasons for CIz and Testing

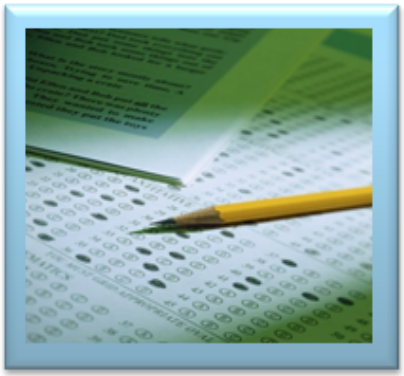
80% of development costs are spent identifying and correcting defects!*

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

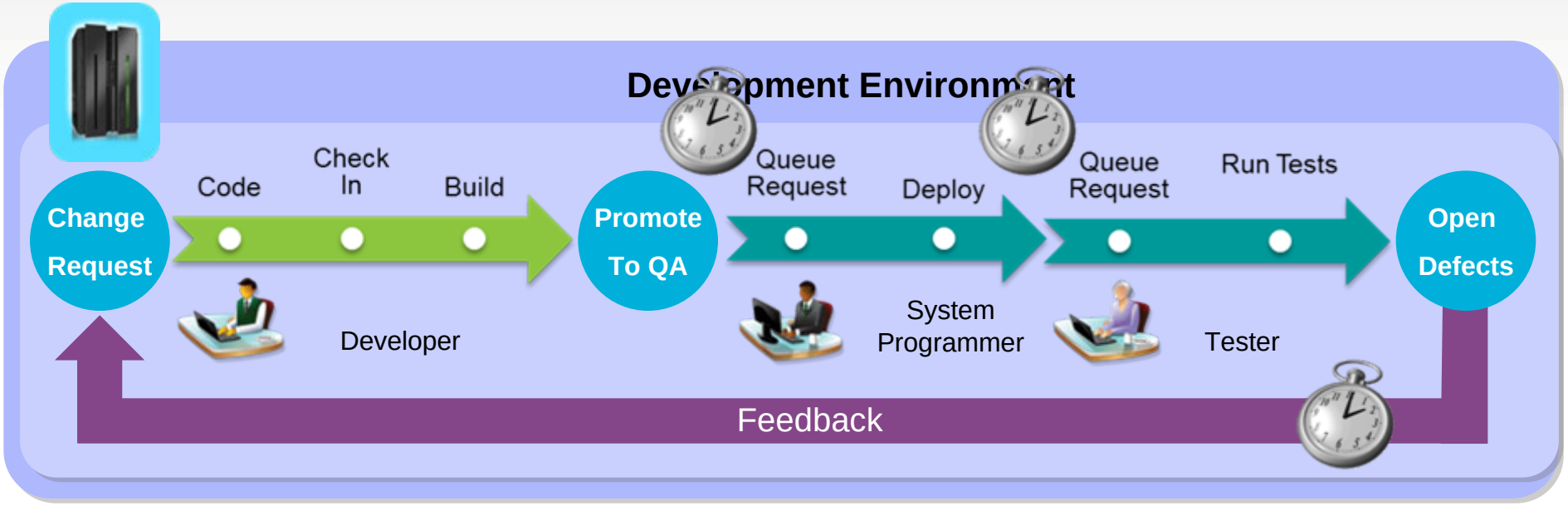
*National Institute of Standards & Technology

Source: GBS Industry standard study

Defect cost 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.

Enterprises want to…

deliver end-to-end application enhancements quickly to stay competitive, trust that complex enterprise systems can be broadly integrated, and bolster confidence in application quality



But…

It takes weeks or even months to test and fix changes due to reliance on manual processes and limited access to test resources

Challenges Meeting Business Time Pressures with Quality Software

*34% of all new IT Projects deploy late**

41% experience delays in integration, configuration and testing of applications

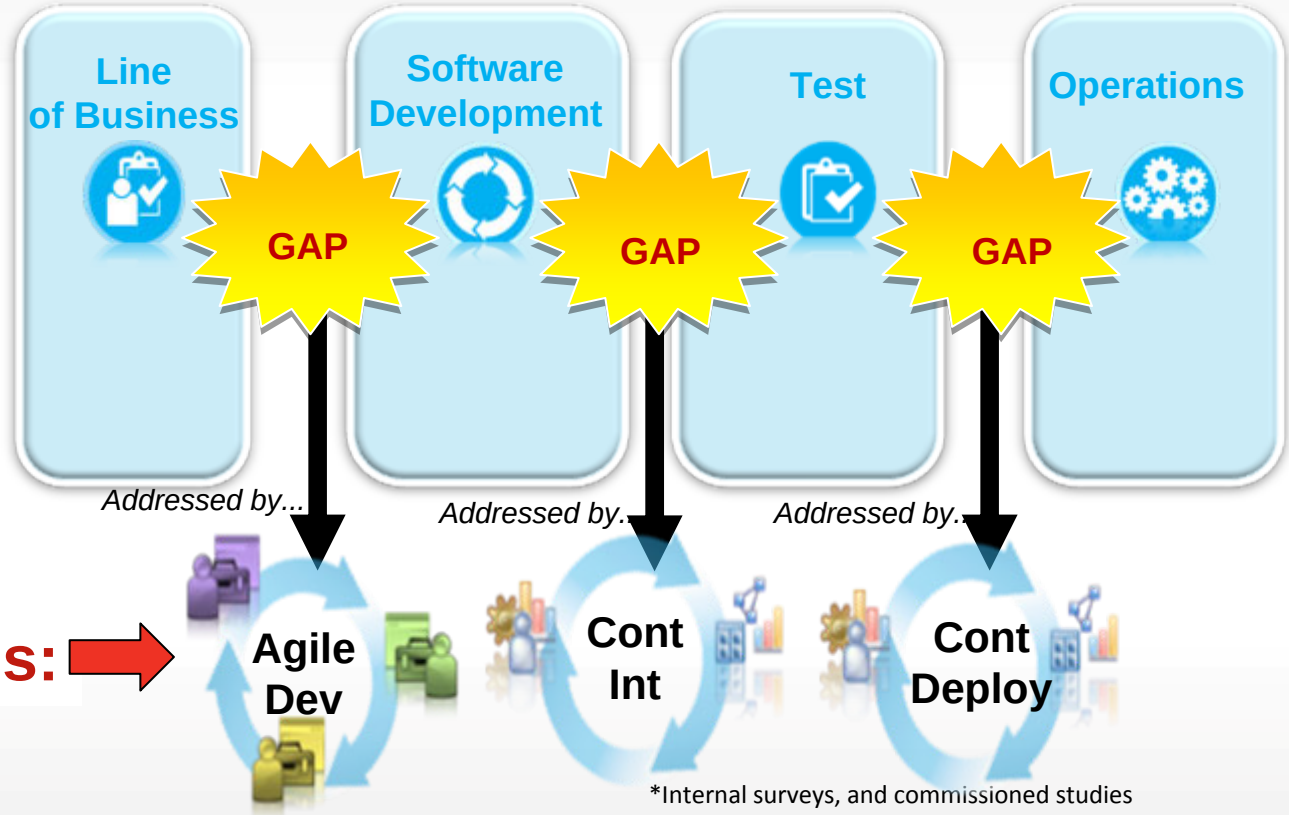
51% applications rolled back due to quality issues escaping into production

45% experience delays due to troubleshooting and fine-tuning issues in production

Up to

4-6 Weeks

to deliver a simple change



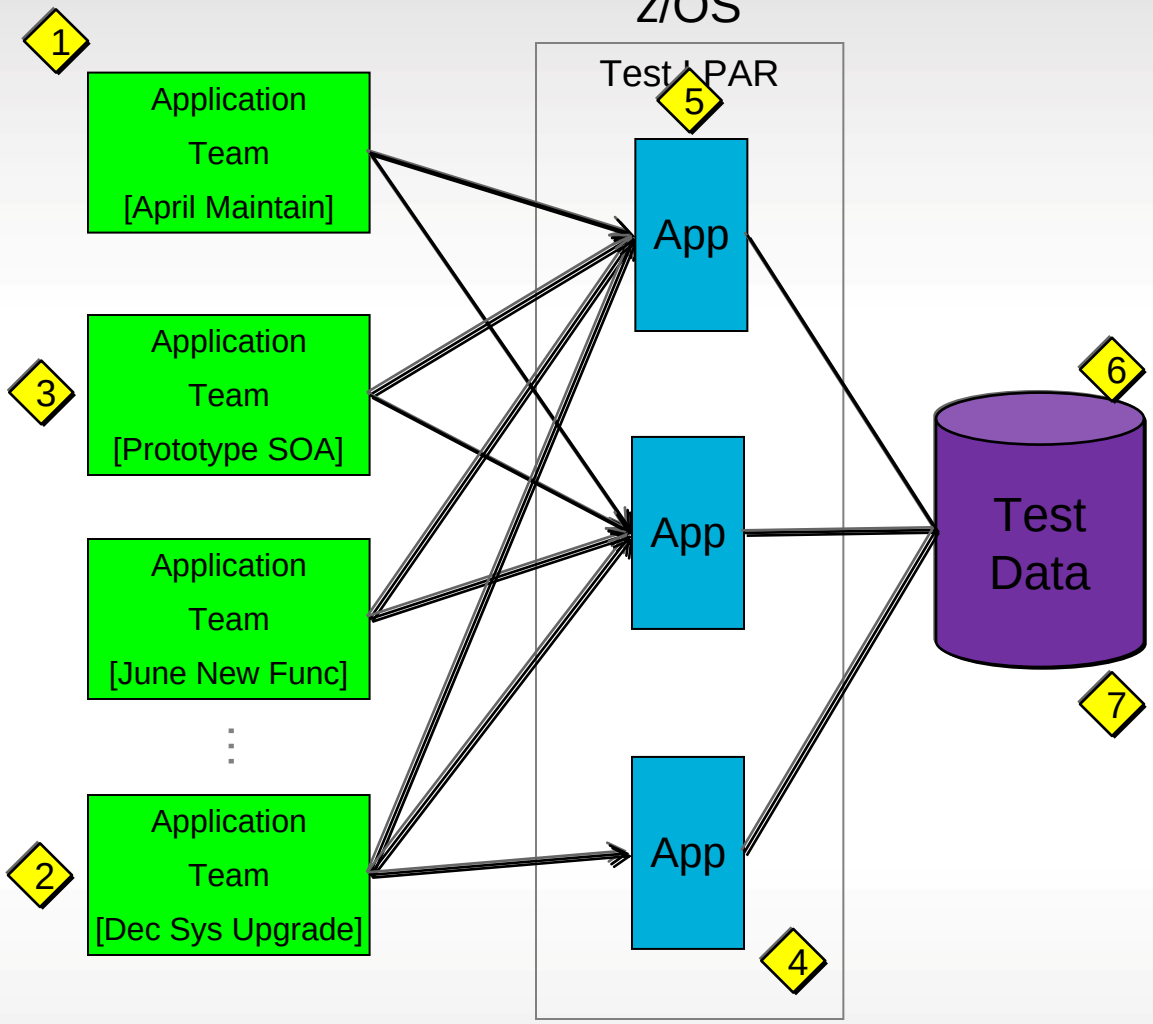
*Internal surveys, and commissioned studies

System z CIz Application Testing Solution

1. Increase the z/OS testing resources
2. Provide automated testing tools to:
   - Developers
   - Testers
   - Q/A
3. Become more agile by integrating requirements to application development and testing (Continuous Integration)

z/OS Testing Observations and Challenges

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

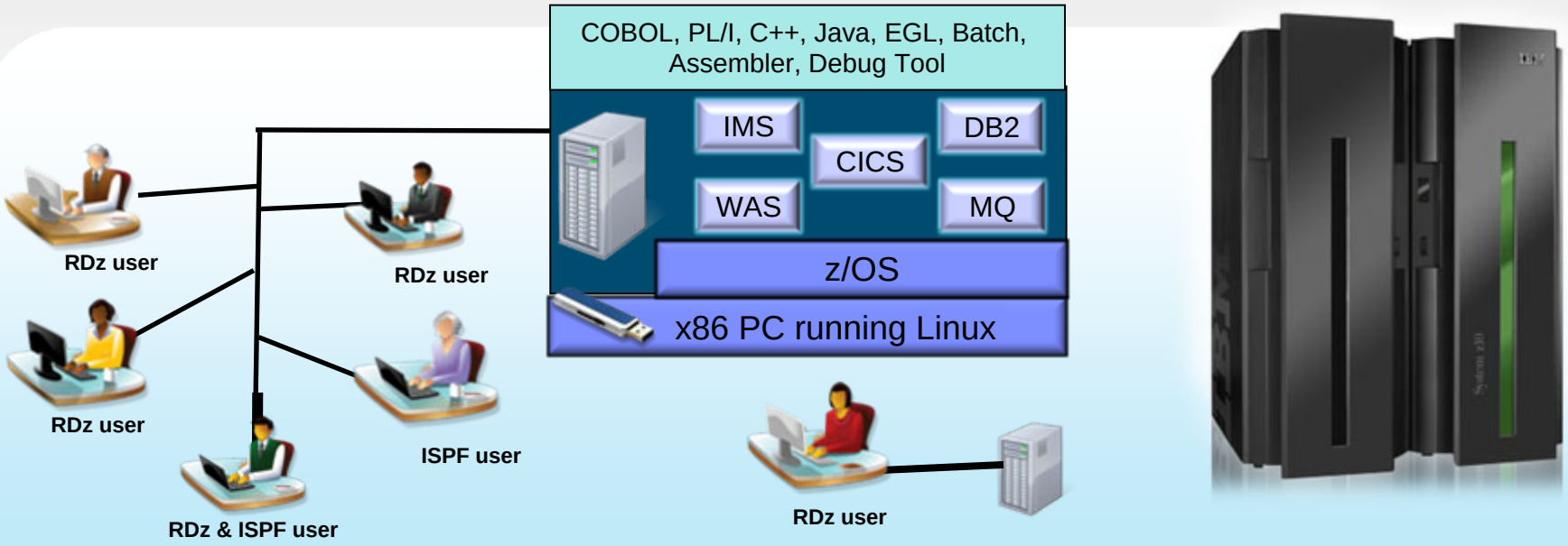


Problems Encountered

1. Teams compete for resources
2. Teams schedules cause overhead in prioritizing work
3. Lack of sandbox environment inhibits innovation and slows initial code delivery
4. Coordination of environmental changes causes bottlenecks at LPAR management
5. Coordination of release components leads to more rework and fewer releases
6. Shared test data takes time to coordinate.
7. Data is one-size-fits-all leading to over-testing

Rational Development and Test Environment for System z

The ultimate in modern application development for System z



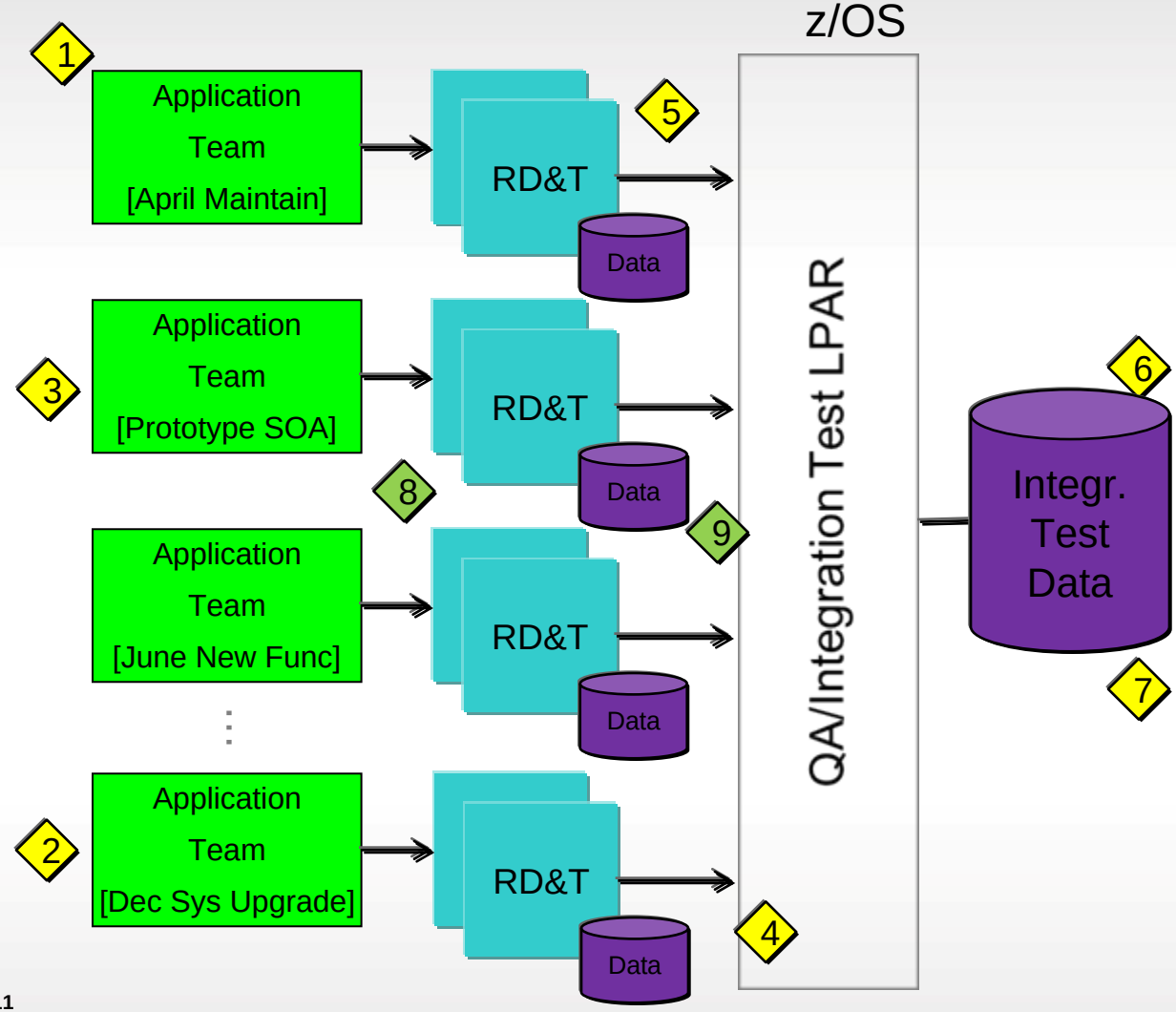
- Liberate developers to rapidly prototype new applications
- Develop and test System z applications anywhere, anytime!
- Free up mainframe development MIPS for production capacity
- Eliminate costly delays by reducing dependencies on operations staff

Note: This Program is licensed only for development , test, and internal training of applications that run on IBM z/OS. The Program may not be used to run production workloads of any kind, nor more robust development workloads including without limitation production module builds, pre-production testing, stress testing, or performance testing.

Increase System z Resources

Organized Testing for Flexibility and Quick Delivery

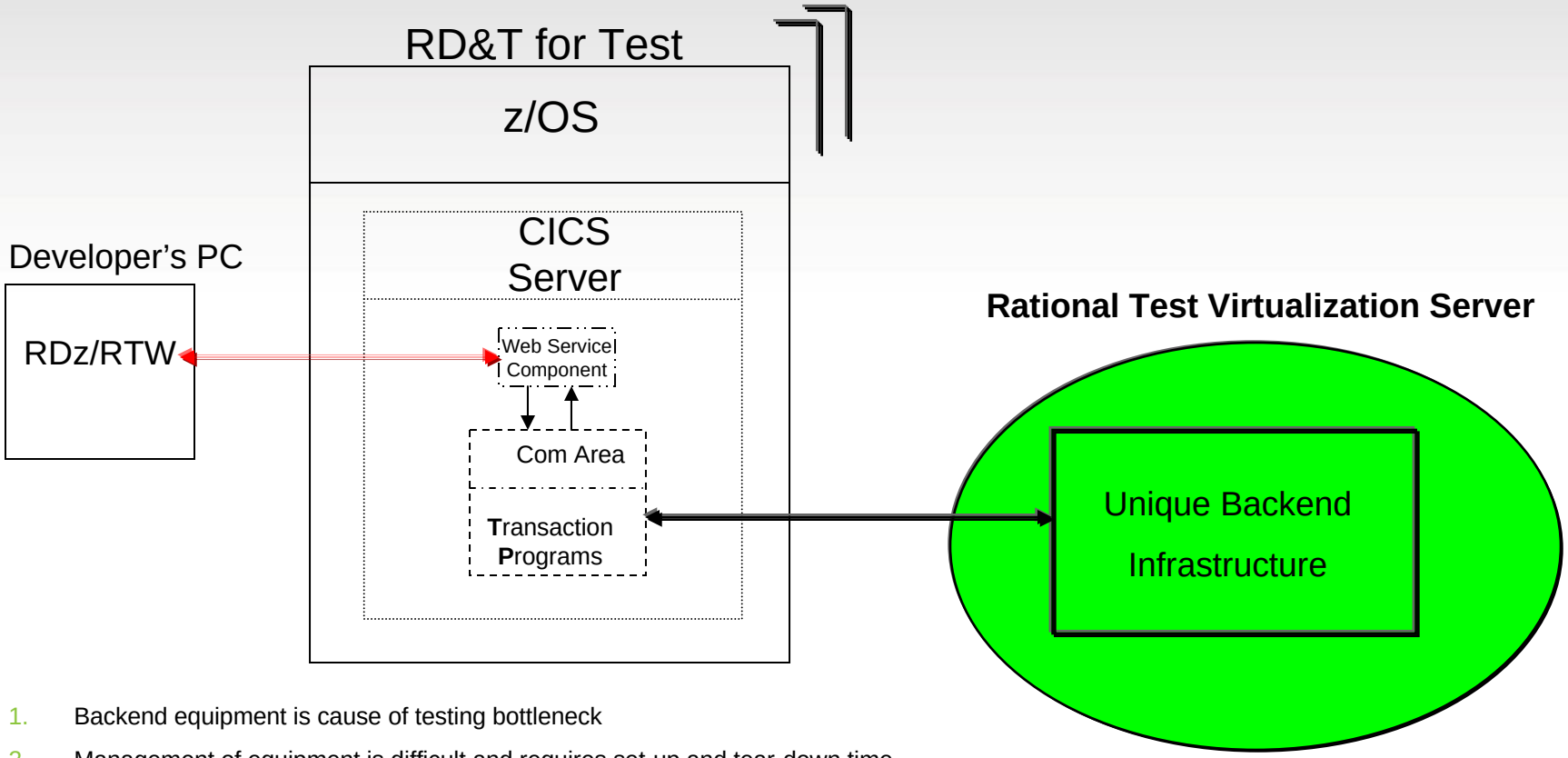
Organized by application team, horizontally sliced, dedicated resources, highly automated



Problems Encountered

1. Teams compete for resources
2. Teams schedules cause overhead in prioritizing work
3. Lack of sandbox environment inhibits innovation and slows initial code delivery
4. Coordination of environmental changes causes bottlenecks at LPAR management
5. Coordination of release components leads to more rework and fewer releases
6. Shared test data takes time to coordinate.
7. Data is one-size-fits-all leading to over-testing

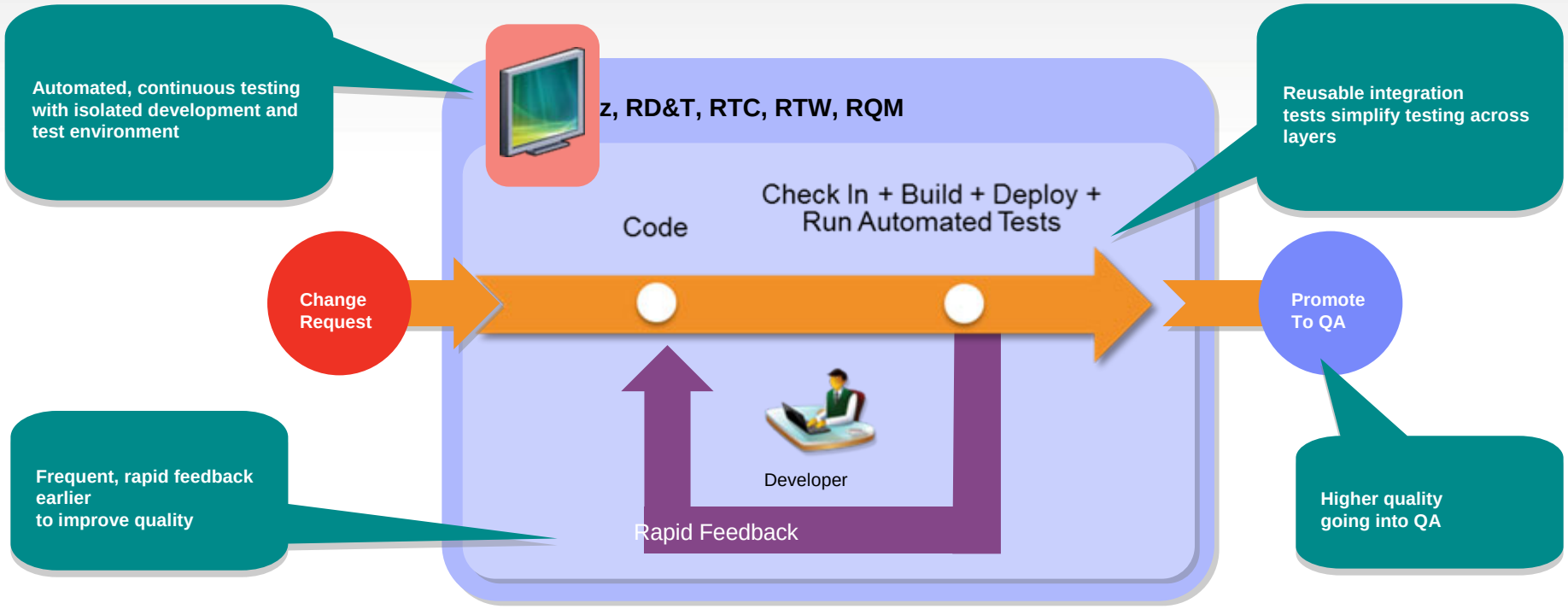
CIz and Rational Test Virtualization Server



1. Backend equipment is cause of testing bottleneck
2. Management of equipment is difficult and requires set-up and tear-down time
3. This can be an inhibitor to testing
4. Rational Test Virtualization Server can resolve infrastructure issues

Improve overall Software Delivery with IBM Rational Solution for Continuous Integration for System z”

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



- Fast, dependable, automatic feedback speeds time to market
- Reduce **operational cost** and maintain continuous integration testing environment with **test virtualization**
- Lower cost of application testing with an off-mainframe z/OS test environment **on a PC**
- Meet **agile** demands with readily available test environment of **current** and **future** releases
- Enables confidence by automatically tracking and promoting code health

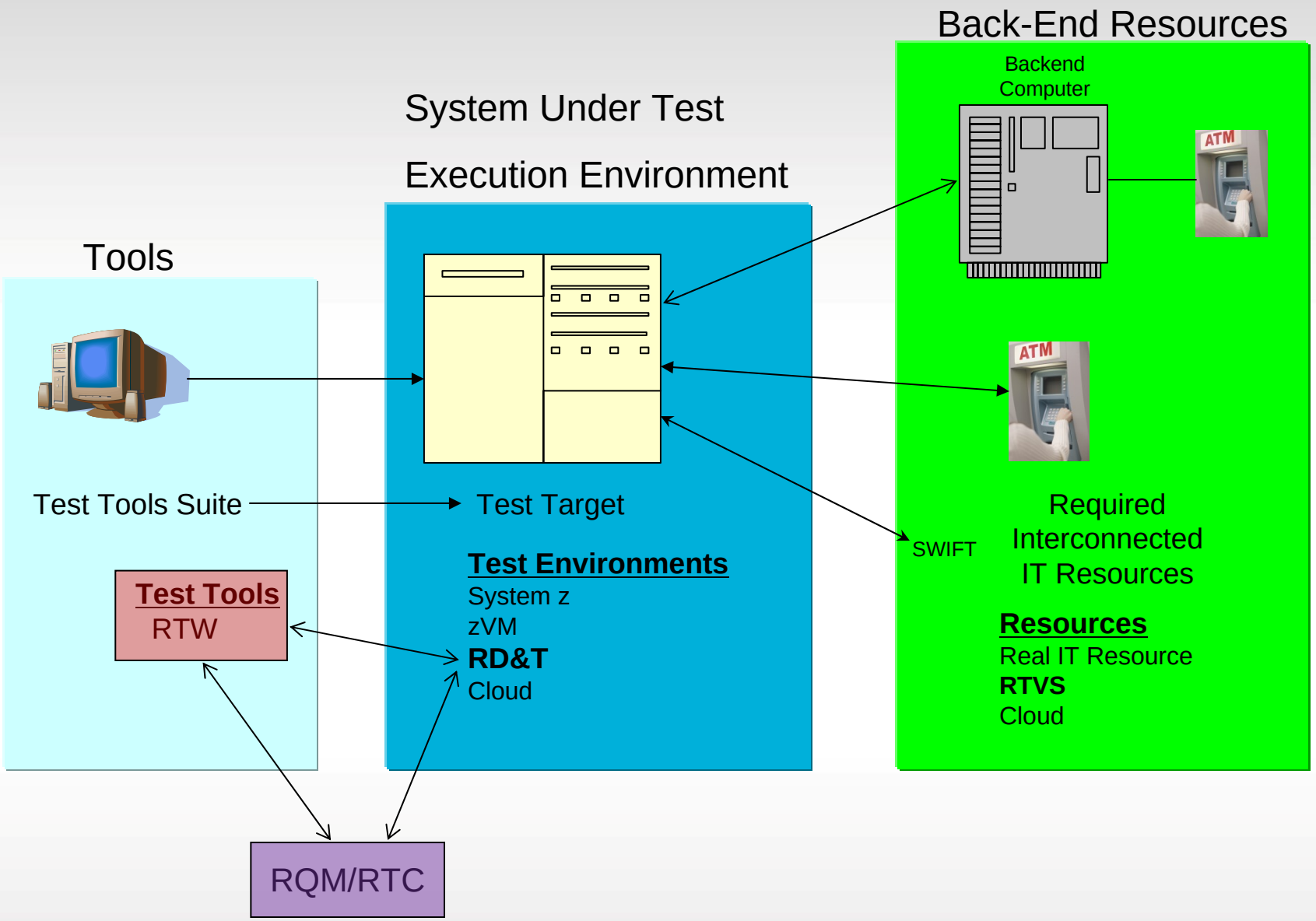
Automated Tooling

- **Jazz Server -** on z/OS or Distributed using WebSphere or Tomcat
  Rational Jazz Server products are:
  - **RTC** – **Rational Team Concert** - manages all aspects of work, such as plans, tasks, revision control, build management and reports
  - **RQM – Rational Quality Manager** - tool used to manage the testing process

- **Client-side**
  Test Machines
  - **RTW** – **Rational Test Workbench** – is a comprehensive test suite that consists of RFT, RIT and RPT components
    - ✓ **RFT** - Rational Functional Tester – a component to do repeatable functional testing of terminal based transactions from a 3270, web or other type of similar devices and Java applications
    - ✓ **RIT** – Rational Integration Tester (a.k.a. Green Hat) – middleware & virtualization component for the functional and performance test of middleware
    - ✓ **Mobile Client – mobile device and end-to-end functional testing**
    - ✓ **RPT** – Rational Performance Tester – Performance testing component
    - ✓ **RTW Agents and Adapters** – ships with RTW and runs on Windows test PCs to integrate RTW with RQM

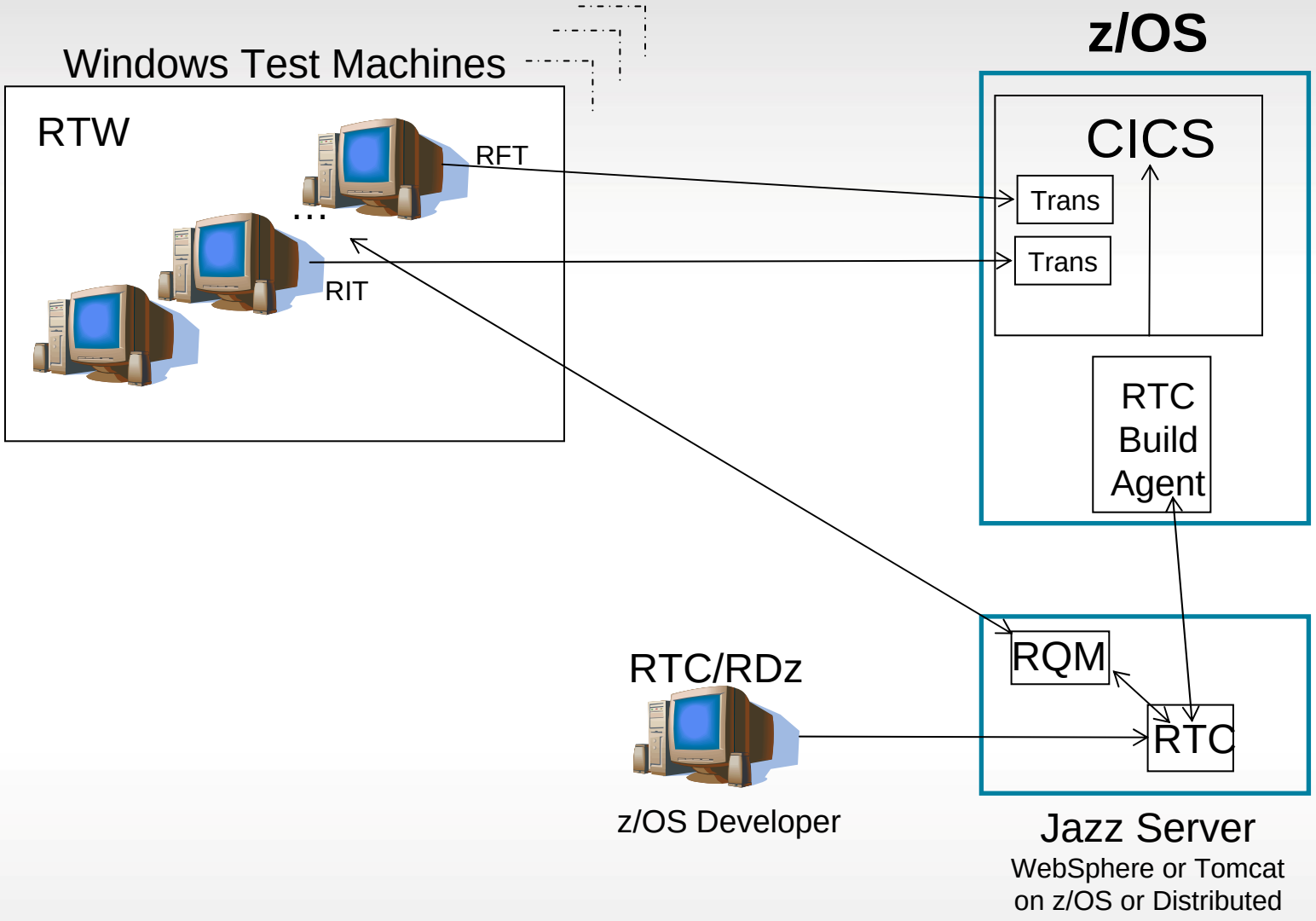
- **Developer PC**
  - **RTC – Rational Team Concert Client** - manage all aspects of work, such as plans, tasks, revision control, build management and reports
  - **RDz** – **Rational Developer for System z** – Application development tool

Clz Testing Products – Where They Fit




DEMO

Demo System Overview



Required Products

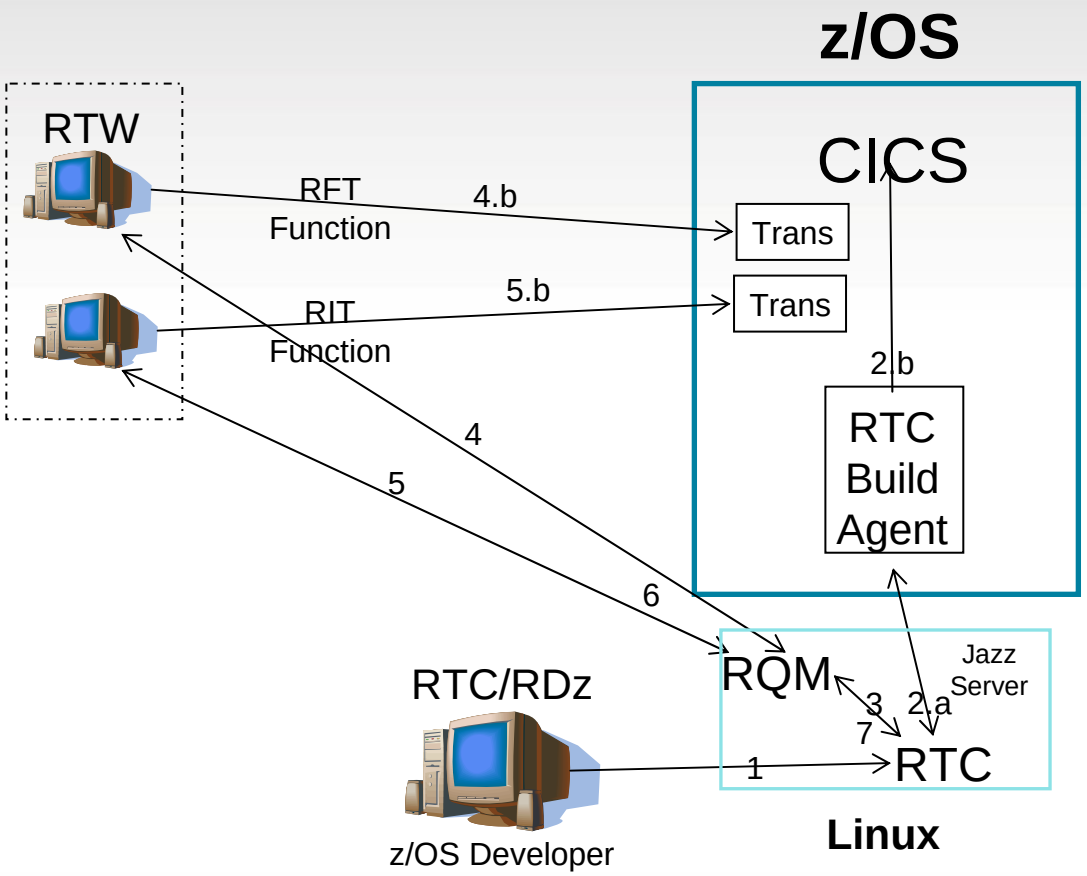
Required Products

- **Jazz Server -** on z/OS or Distributed using WebSphere or Tomcat
  Rational Jazz Server products are:
  - **RTC** – **Rational Team Concert** - manages all aspects of work, such as plans, tasks, revision control, build management and reports
  - **RQM** – **Rational Quality Manager** - tool used to manage the testing process
- **Test Machines**
  - **RTW** – **Rational Test Workbench** – is a comprehensive test suite that consists of:
    - **RFT -** Rational Functional Tester – a component to do repeatable functional testing of terminal based transactions from a 3270, web or other type of similar devices
    - **RIT** – Rational Integration Tester (a.k.a. Green Hat) – middleware & virtualization component that can be used to test Web Services and CICS Transaction APIs
    - **RPT** – Rational Performance Tester – Performance testing component
    - **RTW Agents and Adapters** – ships with RTW and runs on Windows test PCs to integrate RTW with RQM
- **Developer PC**
  - **RTC – Rational Team Concert Client** - manage all aspects of work, such as plans, tasks, revision control, build management and reports
  - **RDz** – **Rational Developer for System z** – application development tool

System Flow

System Flow

1.Developer modifies COBOL Pgm in using RDz and moves it to RTC server and requests a Build
2.RTC server contacts z/OS RTC Build Agent (2.a)for a build and deployment of the transaction to CICS (2.b)
3.On success RTC server will automatically contact RQM server to request a test. Note: RTC and RQM are part of the same Jazz Server
4.RQM contacts RTW/RFT through RFT Adapter to start a web page, 3270, or other device test (3270 in this Use Case, 4.b)
5.RQM contacts RTW/RIT through RIT Agent for CICS Trans API, Web Service, MQ or middleware test (5.b)
6.RTW - RFT or RIT report results back to RQM
7.RQM sends results of the test back to RTC and the developer



Questions