THE CLIPPER GROUP
Navigator™
Navigating Information Technology Horizons℠
Published Since 1993 Report #TCG2013024LI December 13, 2013

Agile Application Development on System z — Is It Keeping Up with Your Business?

Analyst: Stephen D. Bartlett

Management Summary

For the sake of argument, let me claim that *Football*, as played in North America, and *Futbol*, everywhere else, are by far the most popular sports at the professional level. More people attend the games/matches, watch or listen to their broadcasts, or follow their favorite teams most avidly (sometimes resulting in "heated" discussions). Track and field sports, on the other hand, get little notice except perhaps during the quadrennial Olympiad. At that time some of the world's most gifted and skilled men and women athletes seem to stretch the boundaries of human physiology and willpower in the pursuit of excellence.

An excellent example of this prowess is the most exciting 4x100 meter relay race. It combines the speed and power of the four fastest sprinters in each nation, puts them on the same track, and having them pass a baton sequentially, ending when the fourth runner completes the last 100 meter leg. The most |successful team is NOT necessarily the one with the four fastest sprinters, but the team that also is able to pass the baton most efficiently, that is when the passer is still at his/her maximum speed and the recipient is just achieving his/hers. A few tenths or less of second can make the difference between a Gold and a Silver medal, or none at all.

How does this apply to IT operations or more specifically to the development and delivery of software applications (or extensions thereto) that support the business needs of the enterprise? **Those enterprises with development and operations staffs that are the speediest AND most efficient at doing this will most likely allow their team, the enterprise, to win the "Gold".** Moreover this must be achieved while delivering code that is efficient (low resource use) and reliable (bug free or tolerant of errors) – and managing this within the tight constraints imposed by limited IT budgets.

However, there are hurdles (pun intended) to overcome. A typical *System z* with z/*OS* development environment follows well-established sequential "waterfall" development methodologies that largely are manual – completing the coding cycle before moving to test cycle(s), eventually moving to staging the phase-in, and finally into live production. During this time, the development staff is at arm's length from the operations staff and, though very dependent on them, usually has conflicting objectives. Twelve-to-eighteen month project implementations are commonplace – the traditional approach is time consuming and expensive. **Unfortunately, this application deployment model is not sustainable in a world that is driven by quickly appearing and disruptive ideas and technologies.** These ideas and technologies drive enormous expansion of empowered users, cloud use, big data, social media, as well as the use of intelligent/connected devices. These users increasingly demand continuous access, a quality experience, and with little hesitation, rapidly will abandon your enterprise's services if their expectations are not met. Users also expect timely updates and added features with changes happening in days or weeks not months or years.

IN THIS ISSUE

Mainframes are the base technology that host the "systems of record"[1] and usually serve as the enterprise's operational hub. Its regular heartbeat is the ultimate determination of continuing good. **In order for these systems of record to continue to meet the challenges of many rapidly-emerging and need-it-now requirements, a new paradigm must be adopted that emphasizes rapid prototyping, continuous development and testing, and much tighter integration between the development and operations staffs.** The common goal is to facilitate the adoption of business processes and new features that will be responsive to user's demands (whether internal or external), and to do this within a cycle that is measured in days and weeks, not months and years.

Not claiming invention, IBM mainframe advocates are addressing these issues aggressively by fostering and facilitating the adoption of new software technologies coined as *DevOps*. This is the amalgam of *development* and *operations* (the latter being the production side of the data center) as an integrated team, supported by a common set of tools and methodologies that substantially

automates the process of delivering new software functionality – from concept to final delivery to the target audience. Modern enterprises that have faced and acknowledged any or all of these rapid development and implementation challenges will need to address them quickly, or risk continuing erosion of their control of important processing that is tied to the systems of record. If an IBM mainframe is central to your enterprise and you would like to increase the value of that investment to your enterprise, please read on.

Challenges in Today's Mainframe Development Environment

How do you know that you might have a problem? What symptoms should a CIO be looking for? One is likely to be that your peer executives are complaining that the application software supporting a new or revised business process is taking much too long to be implemented, costs too much, and, when delivered, falls short in meeting the functional requirements. In desperation, a business general manager or CMO might threaten to turn to an outside source, or worse, might, on their own authority, chose to implement a solution within their own operations (and, often these days, hosted via public Internet service providers) and outside your control.

In another possible symptom within your own domain, development managers constantly are bickering with the operations staff over the lack of development and test resources, whether hardware and software is configured appropriately, and whether it is available when needed. No doubt, they often have conflicting objectives, the successful achievement on which they will be measured. Developers strive to deliver new applications and new features as quickly as possible, but they frequently are faced with the lack of appropriate test resources. Meanwhile, the goal for data center operations staff is application and data availability and delivery performance. This is accomplished through governance and uninterrupted delivery of services.

Among the mainframe development teams, there usually are conflicts among the various projects. Maintaining test environments that meet the needs of each project is complex and almost always resource constrained. For instance, one team may be doing maintenance on an ERP application, another is installing new analytics capabilities, and a third may be working on a prototype for new mobile connectivity, each having unique test environments – different middleware, suitable test data, and performance profiles and each with systems or parts of systems to maintain. Look closely and you likely will find old toolsets and processes, usually acquired and deployed over several decades. *Why so old?* Because they seem to work, continuing to deliver rigorously controlled software that dependably delivers on the core values of the mainframe environment – security, reliability, resiliency, and high performance – but usually at a higher cost. Tools are different. Bringing new people on board is difficult. Procedures, while functional, are slow.

Under the same roof, usually there is another group of developers, equally conscientious, equally skilled, but born into newer technologies. They don't trust a server that they could not lift, are likely to be adroit in Python, Perl, Java, JavaScript, SOAP, REST, and C/C++, appear to embrace only "open" technologies, and have never heard of (much less worked with) a hierarchical DBMS. These are the folk whose work usually is focused on the front end of the business process flow[2], the customer-facing portion of an

[1] Systems of record process and manage the information of business transactions.

[2] For more on business process management and IBM BPM on System z, see **The Clipper Group Navigator** dated October 16, 2013, entitled *Why More of Your Mission-Critical*

application, often called the "systems of engagement" and the most visible to those looking from the outside in.

Which of these groups is most loved and most valued? Neither, of course, because both are essential to the enterprise, but these "development silos" usually (and unfortunately) present more opportunities for conflict. Seemingly (according to popular culture), mainframe developers are a species that is about to become extinct while the "new-agers" are undisciplined and have no appreciation for the systems knowledge accumulated over the last half-century regarding the care and delivery of the systems of record.

From the point of view of the mainframe developers, the keepers of the "sacred" systems of record, the focus of the front-end developers leaves them ignorant of the many components that make up the application and data delivery chain.[3] This results in the lack of any appreciation of the impact that front-end systems will have on the demands for mainframe resources, and, thus, the costs of providing the mainframe resources required. Where mainframe charge-back systems are at the heart of cost-allocation methodologies, and thus the careful conservation of resources, this concept tends to be little-known or appreciated in the distributed environment. Indeed, mainframe charge-back can even drive work to other systems to avoid the charge-back on the internal bookkeeping. All (operations, "distributed" development, and "mainframe" development) parties are guilty when it comes to understanding how applications flow from end-to-end, quickly assigning blame to the opposite party for poor achievement of an application's function and performance goals. All of these issues have both technical and organizational effects that need to be addressed.

Goals of an Agile Development Process

If your teams have any of these symptoms, it is likely that this is impacting growth and sustainability of your enterprise. Some new thinking most likely is required; in some cases, some traditions may need to be retired. **At a high level, the essential goals are:**

- **Accelerate speed of delivery.**
- **Improve operational and organizational efficiency.**
- **Deliver the highest quality product, to consumers, every time.**

This sounds simple enough, but perhaps some additional concepts are merited.

- ***Application software*** represents business processes that have been implemented and delivered through automation. Usually, the more automated they are, the faster and more reliable they will be.
- ***DevOps*** is the joining of software development processes with IT deployment (operations) processes, which are implemented and automated using software, commonly referred to as "tools".[4] The more automated the processes are, the faster, more efficient, more standardized, and more reliable the processes will be.

The adoption of DevOps principles and appropriate software tools to support these principles can help organizations achieve the goals set out above.

The earlier reference to development silos, if they do exist within your IT organization, may have a hidden benefit in that teams focused on development for the systems of engagement already may have adopted an *agile development and deployment model*. The origin of DevOps dates back to 2009[5], when it was recognized that within more entrepreneurial organizations there was a strong intersection of interests among the software engineering, quality assurance, and technology operations communities that was essential to the continuous delivery and rapid frequency of software releases in support of new business processes.[6] Clearly, this is a model that is likely to be found at companies like Google, Facebook, and Netflix.

Establishing the Beachhead

The goal of DevOps is not to start an internecine war, but to establish the norm for engaging the strengths among all of the stakeholders. **What this will require is the integration of the goals and processes of all the people involved** – mainframe developers, front-end developers, the

Business Should Be Processed on IBM System z, and available at http://www.clipper.com/research/TCG2013019.pdf.

[3] The same is true, in reverse, of course.

[4] The obvious metaphor is the skilled craftsman or artisan who possesses and carefully maintains a set of tools that are essential to the building or repairing of objects within his or her particular specialty. Never ever mess with their tools!

[5] See http://en.wikipedia.org/wiki/DevOps for more on DevOps.

[6] Those familiar with the automotive industry may notice the underlying principles similar to Toyota's innovative Just-in-Time (JIT) manufacturing model that began its evolution in the late 1940's, based on the earlier fundamental thinking of Fredrick Taylor and the Gilbreths.

change and release team, and the operations teams, those responsible for the day-to-day delivery of the enterprise's application portfolio, regardless of its location. **This "integration of the teams" may be the most challenging aspect of any innovations that may be introduced by DevOps methods and tools because this integration likely will test longstanding philosophies.** This is not a technical issue, but deals with the essential values and belief systems of a number of individuals and their departments; in larger organizations there even may be more conflict. **Some minds will be closed to any change that challenges their knowledge, habits, and beliefs. The key is to focus on utilizing strengths rather than what is adversarial.** Nevertheless, some reorganization may be required; in fact, it may be desirable.

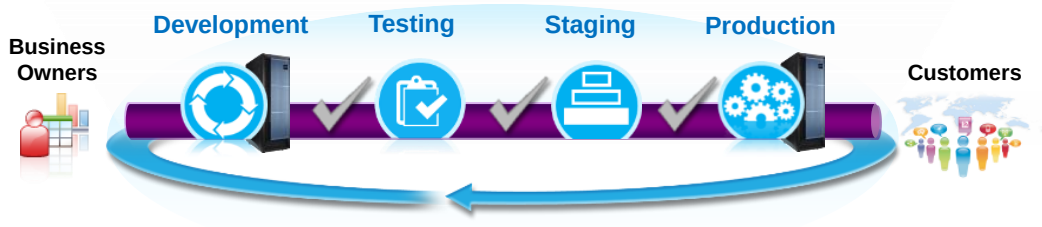
Examining the Tools and Processes

An examination of the software used by the developers to manage their processes will almost certainly reveal a proliferation of toolsets. This is a direct result of the isolated development silos that have evolved over time. There likely will be little in common between the mainframe shop and the distributed shop and the tools likely come from different vendors. The location of the teams may have an effect; plus the newer the operation, the less likely it is to have tools in common with the central data center. And beyond that generality, the toolsets used may be an amalgam from several vendor and homegrown sources. **The consequences of these non-uniform environmental characteristics are that every interface presents an opportunity for incompatibilities, delays, and the loss of essential information.** These are the enemies of faster development cycle times, improved quality, and efficient resource utilization because they result in miscommunication and lost information.

Furthermore, this suggests that the **toolsets used for development and deployment need to be supported cross-platform (across heterogeneous servers and operating environments) and, thus, the tools need to be computing-environment agnostic and also must support cross-platform implementations. Beyond that, they must satisfy the needs and provide visibility to all the stakeholders: mainframe and distributed platform developers, system and database administrators, the operations staffs, and their management.** This may seem to suggest shades of "Big Brother", but visibility also provides the basis for sharing success, a much stronger motivator.

Any parts of the process that require manual intervention – such as hand-offs to a different department, waiting for resources to be allocated, gaining an approval to move to the next phase, or any of a number of other circumstances – will introduce serious and costly delays that must be eliminated or substantially reduced through the introduction of more automated processes and procedures. **One benefit of this automation is that it makes the monitoring of the process simpler and more visible. This facilitates the improved visibility of the development and deployment processes and problems and issues**

Exhibit 1 — The Solution: A Continuous Delivery Pipeline



Ensure applications are production-ready throughout the lifecycle and can be released at any time while minimizing rollback due to quality issues

- Validate on more production-like conditions earlier
- Automate hand-offs/promotions to increase velocity through the different stages
- Standardization on processes and assets between Dev and Ops
- Automated monitoring and dashboarding of quality and performance against service level agreements at multiple stages

Source: IBM

tend to be identified quickly.

In general, you need to keep what is effective and eliminate sources of delay, redundancy and waste. Mainframe developers and their operational counterparts have evolved over a long period the strictest disciplines for change and release management. They encompass procedures that enforce strict governance of security, auditability, load testing, and recovery and roll-back, if required. Any changes being considered must include these essential elements and they must be pervasive and encompass the entire information flow. These elements need to be celebrated and worked into the automation of processing, not discarded.

Overly ambitious efforts often become the seeds of expensive, and unfortunately, very visible failures. This premise establishes the rationale for breaking work into smaller and more manageable pieces – attempting to satisfy every requirement in one grand effort invites disaster. In mainframe land, its history is littered with such events, but neither has the distributed world escaped unscathed. *Behind schedule* and *over budget* are mutual-shared, interrelated problems. **Smaller program elements result in faster testing, the need for fewer leaner test beds, and more frequent releases with higher quality.** (See the "process pipeline" shown in Exhibit 1, at the bottom of the previous page.)

Quality Processes Require Quality Information

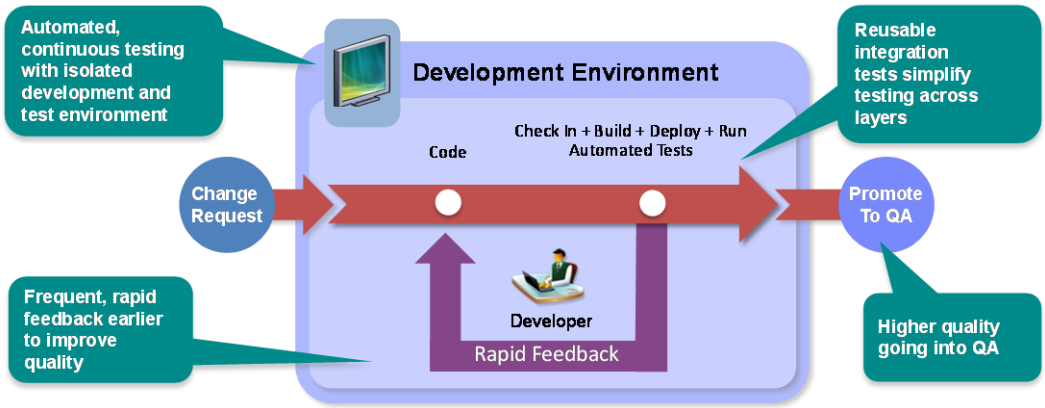
Enterprise applications have hundreds if not thousands of disparate parts. They encompass a view of the flows from the business owners (COO, CFO, CMO, CIO, and business line managers) to the enterprise's customers (and partners) and back again, while the business owners struggle to satisfy their customers' increased needs and enrich their experiences. The "process intermediaries" are all those that constitute the application development, test, and production systems of the enterprise. They are called to enable rapid evolution of deployed business services while being challenged to reduce risk, reduce costs, and improve the quality across the entire portfolio. They need a solution that embodies a *Continuous Delivery Process* that enables:

- Collaboration across disciplines
- Development and test against production-like system environments
- Frequent, repeatable, reliable deployment
- Continuous validation of operational quality

As was shown in the pipeline in Exhibit 1 on page 4, the process must accommodate the needs of developers, testers, and operations and the information flow between them. It embodies pervasive automation, versioning, testing, tracking, instrumentation, visibility through the dashboarding the entire process, and extensive feedback loops. No part of an application is subject to exclusion, whether it is within the systems of record on the mainframe or a system of engagement on a distributed platform. **DevOps is about**

Exhibit 2 — The Solution: Continuous Integration

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



- Fast, dependable, automatic feedback speeds time to market
- Lower cost of application testing using off-mainframe z/OS test environment
- Enables confidence by automatically tracking and promoting code health

Source: IBM

accelerated, continuous delivery of software innovation.

Enabling DevOps for System z

The keys to success are "continuous integration" and "rapid feedback". Expectedly, IBM understands these issues and offers a comprehensive suite of integrated solutions based on its *Rational* family of software development and management products. *Wait, will it be necessary to install, maintain, and pay for a whole lot of new z/OS software? No!* Most of what is described in the remainder of this paper is supported on a number of non-System z environments and built to run on x86 platforms. However, most also are supported on *Linux on System z*, virtual Linux servers hosted on the *Integrated Facility for Linux (IFL)* specialty engine that is available on all zEnterprise systems products. They effectively offload the consumption of standard MIPS and thus will have no impact on z/OS license charges and also will be very familiar to most mainframers. These tools support the development, deployment, and testing of applications which run on many platforms, including those which run on System z (z/OS and Linux on System z).

Conceptual View of IBM's Vision

The conceptual view of an exceptional development environment encompasses "continuous integration" that delivers rapid delivery, end-to-end visibility of all testing, and high quality and faster upgrades. (See Exhibit 2, at the bottom of the previous page.) For the purposes of this paper, illustrative examples of IBM solution offerings that support the conceptual view are divided into four broad categories – automation, communications, quality, and testing – in order to focus on their core capabilities.

Automation

IBM offers solutions that automate consistent build, configuration, and deployment processes across all phases. These capabilities and more are included in ***IBM Rational Team Concert (RTC)*** **product, which helps teams collaborate for faster software delivery in four ways.**

- ***Enhancing team collaboration*** with integrated features including work-item, build, and software configuration management. Integrated work-item tracking, source-control management, continuous builds, iteration planning, and configurable process support all make collaboration easier.
- ***Providing high visibility into project activities and team progress*** via multilevel dashboards and reporting features. At-a-glance and real-time views improve usability.
- ***Facilitating planning and execution of agile or formal projects*** with planning tools and templates. Consistent processes help improve software quality.
- ***Helping to improve productivity*** with advanced source control management for geographically distributed teams. Sharing, comparing, and managing software changes among teams and across repositories keep everyone aware.

The server portion of the RTC solution may be installed in a z/OS environment, but most users will choose any of a number of server environments commonly available from IBM, including Linux on System z, AIX, and Linux and Windows Server on System x, as well as servers from other vendors.

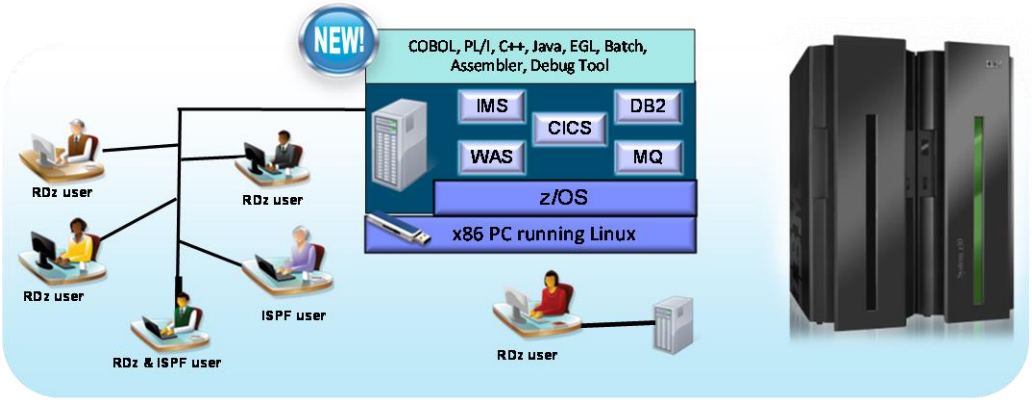
Communications

The goal here is to improve communications and collaboration with cross-platform release planning. These capabilities are provided by the *IBM Collaborative Lifecycle Management (CLM)*. CLM brings together requirements management, quality management, change and configuration management, project planning and tracking via a single, shared platform. It includes Rational Team Concert, previously described, coupled with the *Rational Quality Manager (RQM)*, and the *Rational Requirements Composer (RRC)*.

- **RQM helps teams share information about test plans, test cases, test results, and test environments uniformly, use automation to accelerate project schedules and to report on metrics for informed release decisions.** Collaboration is fostered by sharing project information and status updates seamlessly so that team members can synchronize teamwork throughout the lifecycle.
- **RRC empowers teams to define, manage and report on requirements in a lifecycle development project.** It provides a broad, flexible approach that enables extended teams to collaborate, clarify and quickly achieve consensus about requirements throughout the project lifecycle – as they develop business-driven solutions.

Exhibit 3 — 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 and test 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.

Source: IBM

Quality

Enforcement of quality standards is an absolutely essential goal for any project, regardless of size. Quality cannot be "tested in"; it must be integral to the entire process. IBM addresses these issues, again through advanced automation not only via the Rational Quality Manager (RQM) described above, but also via other tools including the *Rational Test Workbench (RTW).*

The RTW provides a comprehensive test automation solution for mobile applications, regression testing, integration technologies and performance and scalability testing. It helps build intelligent and interconnected enterprise applications that can be deployed on traditional and cross-platform cloud infrastructures. Usually, test cycle times are reduced significantly, resulting in the movement of integration testing to earlier in the development lifecycle.

The Rational Test Workbench provides test automation for all types of applications through the use of physical device emulation, including mobile platforms. Additionally, RTW offers the following.

- ***Simplifies test creation*** with storyboard testing and code-free test authoring.
- ***Facilitates quick development of complex performance test scenarios*** with script-less, visual, performance test and workload models.
- ***Provides earlier, end-to-end continuous integration testing*** across hardware, software and cloud-based dependencies.
- ***Emulates workloads accurately*** so that server workloads that represent realistic user scenarios can be created.
- ***Is extensible and supports standards and protocols*** to help meet the challenges of cross-platform testing environments.

Testing

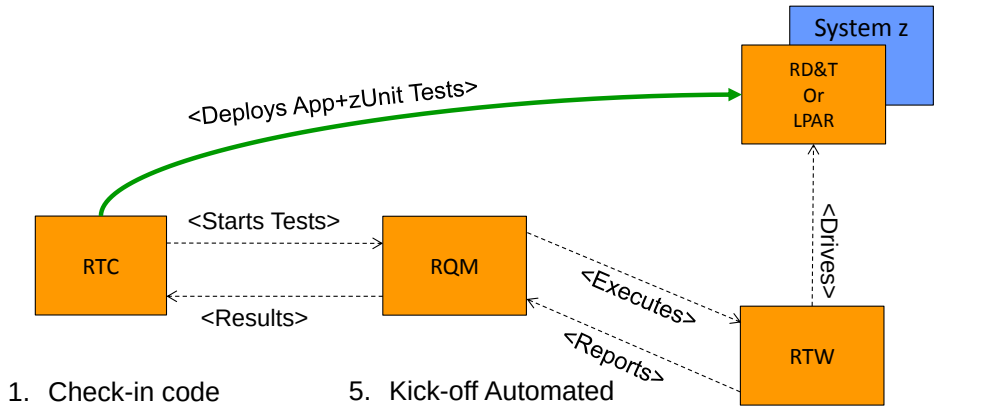
Common arguments in mainframe development shops usually contain one or more of these "not invented here" themes.

- No capacity is available to automate testing.
- Maintenance of lots of z/OS system images is complex, too time-consuming, and eats up our skilled resources.
- Current waterfall methodologies have worked for two decades, so why should we change it?

As suggested earlier in this paper, these are not only technical issues but ones that also test a development team's ability to recognize a compelling need and adapt to the changes required. IBM offers a solution that has been tried, tested, and proven in the real world.[7]

[7] The essential elements have been available and in use for more than four years by a large number of Independent Soft-

Exhibit 4 — Detailed Continuous Integration for System z Scenario



1. Check-in code
2. Build code and zUnit tests
3. Deploy build results to Test Environment
4. Execute zUnit Tests
5. Kick-off Automated Test Plan
6. Run automated interface tests against Test Environment
7. Mark execution records Pass/Fail in Test Execution Manager
8. Report test results in dashboard/build results/defect records in CI server.

Source: IBM

IBM's Out of the Box Thinking

IBM's *Rational Development and Test Environment for System z (RD&T)*[8] provides an environment for mainframe application development and testing where the operating systems, middleware, and software can run on x86 platforms, without the need for actual System z mainframe resources.

This software is based on the *IBM System z Personal Development Tool (zPDT)* that provides an emulated System z architecture with virtual I/O and devices that significantly promotes fast development and test cycling of the most common mainframe application environments. The RD&T allows and delivers the following.

- ***An application development and testing environment*** that can improve development, infrastructure availability, and flexibility with no effect on mainframe resources.
- ***Mixed workload support*** that can help reduce development costs and can span multiple platforms.
- ***A high-fidelity development and test environment*** that can streamline the delivery of production-ready applications in concert with other tools, such as RQM.
- ***Provides current levels of the IBM z/OS middleware*** (including *IBM CICS, IBM IMS, IBM DB2, IBM WebSphere, COBOL, PL/I, REXX, C/C++, Java,* and *high-level assembler (HLASM)*). This facilitates access to new run-time capabilities for development and testing purposes. See Exhibit 3, at the top of the previous page.

This tool commonly is installed on a local x86 server, such as an *IBM System x*, but is capable of running on laptop for exceptional portability and low provisioning cost. Of course, direct testing on the mainframe on an LPAR configured in the target environment is not precluded and may be desired in the final stages of testing prior to full deployment.

Most of the tools described above are part of integrated solutions from IBM, including the *IBM Integrated Solution for System z Development (ISDz)* and *the IBM Continuous Integration Solution for System z (CIz)*. These both provide comprehensive bundling at a lower cost than if the products are licensed individually. For an illustration of how several of the tools described here fit and work together see Exhibit 4, above.

ware Vendors (ISVs) who have been implementing a broad range of products for z/OS on System z.

[8] The Rational Development and Test Environment for System z only can be used for development, test, employee education, or demonstration of applications that run on z/OS. It may not be used for production workloads of any kind, nor for robust development workloads including without limitation production module builds, pre-production testing, stress testing, or performance testing.

Even Better with Latest Compilers

Backed by IBM's longstanding commitment to forward compatibility of hardware and software, IBM compilers improve programmer productivity by exploiting advances in hardware and performance optimization, without requiring special coding or source code changes. They let the programmer focus on the business logic of the application – in the following ways.

- Integrating existing business-critical applications with dynamic Web applications
- Supporting interoperability with Java and promote the exchange and usage of data with XML
- Optimizing the performance of commercial and high-performance computing workloads
- Reducing CPU cycles consumed for a given task, maximizing your IBM hardware investments

Conclusion

There is no point in minimizing the organizational and, perhaps less so, the technical challenges presented in this paper – primarily because the larger the organization, the greater the likelihood of poor collaboration among different groups. Tribal knowledge is perpetuated in manual processes and small group conversations and these processes often have poorly-defined or no metrics by which they can be monitored and measured. There are fundamental obstacles resulting from the conflict between the need for rapid test and development and the absolute requirement to protect the integrity and performance of the current suite of applications.

Initiating adoption of DevOps principles and the necessary software tools to achieve them has the promise of effective and continuous delivery of innovation of an enterprise's business processes across all platforms and applications used by the enterprise. These principles embody a common set of practices that focus on maximizing value, automation of manual processes and any other overhead activities that tend to impede progress. This enables continuous learning through feedback loops that extend out to the end-user/customer, and improves visibility across the organization by using meaningful measurement and monitoring tools.

Enterprises that have mainframe systems of record as their operational hub, even those that have robust development and deployment operations, are not immune to the realities of the accelerating demand for on-demand, meaningful business services. They are likely to be at risk falling behind the "innovation curve", if their operations always are in "catch up" mode. Adoption of DevOps principles and practices holds the greatest promise in the continuing quest to establish and maintain innovation leadership, and to allow their enterprise team to win the "Gold". It now is up to you!

About The Clipper Group, Inc.

The Clipper Group, Inc., now in its twenty-first year, is an independent publishing and consulting firm specializing in acquisition decisions and strategic advice regarding complex, enterprise-class information technologies. Our team of industry professionals averages more than 25 years of real-world experience. A team of staff consultants augments our capabilities, with significant experience across a broad spectrum of applications and environments.

- ***The Clipper Group can be reached at 781-235-0085 and found on the web at www.clipper.com.***

About the Author

Stephen D. (Steve) Bartlett **is a Senior Contributing Analyst for The Clipper Group.** Mr. Bartlett's interests include enterprise solutions including servers, system software, middleware, their underlying technologies, and their optimal deployment for responsiveness to emerging business requirements. In 2010, he joined the Clipper team after over 42 years with the IBM Corporation as an account and program manager in large system sales, product development, strategy, marketing, market research, and finance. During that time, he received several awards for his contributions in innovative market research and contributions to the business. He has a B.S. from Rensselaer Polytechnic Institute, and an M.S. from Union College.

- ***Reach Steve Bartlett via e-mail at steve.bartlett@clipper.com or at 845-452-4111.***

Regarding Trademarks and Service Marks

Disclosures

Officers and/or employees of The Clipper Group may own as individuals, directly or indirectly, shares in one or more companies discussed in this bulletin. Company policy prohibits any officer or employee from holding more than one percent of the outstanding shares of any company covered by The Clipper Group. The Clipper Group, Inc., has no such equity holdings.

After publication of a bulletin on *clipper.com*, The Clipper Group offers all vendors and users the opportunity to license its publications for a fee, since linking to Clipper's web pages, posting of Clipper documents on other's websites, and printing of hard-copy reprints is not allowed without payment of related fee(s). Less than half of our publications are licensed in this way. In addition, analysts regularly receive briefings from many vendors. Occasionally, Clipper analysts' travel and/or lodging expenses and/or conference fees have been subsidized by a vendor, in order to participate in briefings. The Clipper Group does not charge any professional fees to participate in these information-gathering events. In addition, some vendors sometime provide binders, USB drives containing presentations, and other conference-related paraphernalia to Clipper's analysts.

Regarding the Information in this Issue

The Clipper Group believes the information included in this report to be accurate. Data has been received from a variety of sources, which we believe to be reliable, including manufacturers, distributors, or users of the products discussed herein. The Clipper Group, Inc., cannot be held responsible for any consequential damages resulting from the application of information or opinions contained in this report.