

Consolidation with COBOL and Java

This white paper discusses the topic of how a business with both Traditional and Modern technologies, COBOL and Java, can simplify its development and deployment landscapes.

Development Issues

Organizations in existence for more than 20 years typically have a major portion of the IT inventory developed in COBOL. Development staffs continue to maintain and extend COBOL inventories using a range of tools from historic text editors to modern integrated development environments. Parallel to this activity, new development is being conducted within the Java landscape, and the investment grows as organizations standardize on J2EE while building their Service Oriented Architecture (SOA). There is this dichotomy of development staffs, processes and tools that need to be resolved and that solution is supplied by LegacyJ.

How Do You Describe Relief?

- + Open Systems
- + Integrate Systems & Staff
- + Easy Transition

Virtualization

Virtualization is why organizations are embracing Java so readily. Virtualization is the execution of code on the Java Virtual Machine rather than natively on the physical machine, providing significant benefit to IT organizations. Virtualized code executes on industry standard platforms compliant with J2EE.

And the hardware industry has also responded to Java. Witness IBM's offerings to zSeries customers to reduce Java execution costs:

- z/OS.e is a Java-only operating system, does not support many of the capabilities of standard z/OS. Therefore, it is much less expensive to operate, reducing the customer's expense of operating their software solutions vis-à-vis z/OS software solutions.
- zAAP is a z/OS processor where Java workloads can be off-loaded for execution. zAAP processors are standard IBM processors, yet are marketed outside of the z/OS-visible processor count, reducing the customer's operational costs across both the z/OS hardware and software spectrum.

Other important new technologies are like those that come from Azul Systems (azulsystems.com). Azul offers purpose-built appliances that attach to the customer's existing network and transparently off-loads all Java execution to this purpose-built machine. The result is that IT organizations can benefit from reducing workloads on existing platforms, consolidate servers in the application tier and significantly increasing overall processing throughput.

More and more, one should expect that there will be additional pressure to leverage the virtualization initiative. And of course, organizations that have standardized on J2EE have already begun taking these steps.

The Challenge and COBOL

Organizations with COBOL inventories have the bigger challenge in positioning for SOA than do Java-only organizations. COBOL IT assets are rooted in native hardware and, many times, "proprietary" environments that are not readily able to exploit J2EE. Yet because the COBOL inventory houses the vast majority of business rules executed by IT, access to and from the virtual environment is mandated.

Turning to "connectors" to bridge between the native environment and the virtual environment is an approach that, however, introduces a new level of complexity. This "connector" strategy requires that changes made within the two different technology silos (COBOL and Java) always be reflected within the "connector", consuming staff and budget to remain synchronized.

The better approach is to bring together (consolidate) both technology silos by imbuing COBOL with J2EE compliance. Consolidating the execution environments eliminates “connectors” and associated complexity, freeing staff and budget while also consolidating solution architecture (solutions can now be both COBOL and Java) and development (both staffs can use the same IDE toolset, if desired).

COBOL Becomes J2EE-Compliant

LegacyJ's technology allows developers to continue to define business processes in COBOL while enabling evolution into Java. Compiling the existing COBOL IT assets with the LegacyJ compiler permits deployment in a Java execution environment. COBOL solution compliance with the J2EE standard becomes an easy reality.

The LegacyJ approach is to require as little change to the source as possible. The source of the solution remains COBOL. Thus, development, maintenance and debugging are in COBOL. In other words, the COBOL developer stays within the comfort zone of COBOL expertise and requires little training.

Since the source does not change, becoming J2EE-compliant can be viewed as a natural upgrade of the solution (evolution), not a wholesale solution change. This firmly protects the end-user community from extensive re-training.

Executing in the “Open” Java environment provides access to new Java-enabled technologies. Communication between Java and COBOL is straight-forward and is a simple CALL between programs. Consolidation via LegacyJ technology is the easy path to J2EE compliance.

Extract & Migrate Project

To take the path towards SOA, the ideal candidate should fit LegacyJ's technology profile (see accompanying box). Those fitting the profile should find that the use of LegacyJ technology is straight forward. LegacyJ just re-compiles COBOL for execution in “Open” Java environments.

Using the technology requires an “Extract and Migrate” project consisting of the following steps:

- Extract Cobol source entities from existing platform and import into the LegacyJ Eclipse-based IDE
- Re-platform data definitions to be J2EE-compliant (if not already J2EE-compliant)
- Migrate by compiling and making any need modifications to Cobol source
- Test to validate equivalent execution using existing test cases
- Deploy to any/all J2EE production environments.

COBOL Factors

- COmmon Business-Oriented Language – remains ideal language to describe business processes.
- Extensively used within the business and government sectors
- Existing Compilers compile for execution on platform's native environment
- Compiled executable does not comply with J2EE Standard

Ideal Solution Profile

- Solution Source code in COBOL
- LegacyJ supports 15 COBOL dialects

Ideal Organization Profile

- Desires upgrade rather than re-create as path to J2EE compliance
- Familiarization with RDBMS
- Familiarization with Java
- Familiarization with Application Servers

New Opportunities

Taking the existing COBOL solution to J2EE by using the "Extract and Migrate" path with LegacyJ presents new opportunities. Using these opportunities, the IT organization can:

- Expose COBOL based business rules as a web service.
- Consolidate COBOL and Java programs into a single business solution.
- Repackage COBOL as a J2EE-compliant solution.
- Integrate the development efforts of the COBOL and Java teams, leveraging the best of both business and web skills while improving cross-training and career growth for both teams.

Java is Open

- Platform
- Operating System
- RDBMS
- Application Server

LegacyJ is Open

- Development Language (COBOL & Java)
- Java Platform
- Deployment Method (jar, war, ear)
- Deployment Consolidation with existing jar's, war's, ear's

Taking Action

Adopting J2EE compliance is the important first step. Partnering with LegacyJ is the easy way to make the COBOL business silo a part of this standard. Eliminate the "connectors", taking staff and budget to the "Extract and Migrate" project. Deploy to your end-users an upgrade that now has extendable capabilities.

Partnering with LegacyJ allows gradual controlled evolution, meaning the IT organization can retain existing "look and feel". For the end-user, maintaining consistent behavior is important. Keeping change to a minimum means that the end-users do not require training to use the upgraded solution.

One of the values of LegacyJ is that COBOL has not changed, and the environmental capabilities have been maintained (Screens, Process and as much as possible Data). Taking the pragmatic approach of first moving the existing COBOL code base and getting it functioning in the new environment prior to adding new application extensions is the recommended first project step; it helps reduce project creep. Benefits of this approach overrules those of applying changes to the solution during the initial "Extract and Migrate" effort. A subsequent phase can introduce new features such as GUI, COBOL functions or Java routines as the organization sees fit. Being in Production as a J2EE-compliant solution is the primary objective.

Action Checklist

- Make Decision to Standardize on J2EE
- Partner with LegacyJ
- Eliminate "Connectors"
- Upgrade Existing Solutions to J2EE to take Advantage of "Open"

An Application that is J2EE-compliant means that there are no boundaries between COBOL solutions and other modern solutions within the virtualized environment. COBOL easily calls Java and Java easily calls COBOL. Data exchange is vastly simplified. Consolidation is an action that adds up to success.

Partner with LegacyJ to make COBOL and Java consolidation happen. Visit our website, www.legacyj.com, and download for evaluation our products. Then make the LegacyJ decision.

COBOL Anywhere to Java Everywhere!