

# **PUND-IT RESEARCH**

## **Marketplace Update**

### **Rationally Speaking**

#### **How IBM's Rational EGL Benefits Mainframe SOA**

**January 2007**

Pund-IT, Inc.  
Hayward, CA  
U.S.A. 94541

Phone: 510-909-0750  
Fax: 510-886-4937  
charles@pund-it.com  
[www.pund-it.com](http://www.pund-it.com)

# Rationally Speaking – How IBM's Rational EGL Benefits Mainframe SOA

By Charles King, Pund-IT, Inc.

For many IT vendors and their enterprise customers, Service Oriented Architecture (SOA) represents the next best opportunity to gain maximum advantage from IT infrastructure investments. The reasons for this are simple and compelling. SOA opens doors to reusing existing IT assets for services supporting new applications and processes, thus allowing organizations to enhance IT/business performance, maximize their return on investment (ROI), and delay, or even avoid costly IT purchases.

If SOA and IBM's signature System z mainframe solutions seem ideally suited for one another, in many ways they are. IBM mainframes represent the Quality of Service (QoS) gold standard of IT platforms, and have been in common use for decades powering core business applications across global regions and industries. But while newer IBM System z solutions offer specialty co-processor options for cutting-edge Web- and Java-specific applications, many consider older mainframes to be bastions of traditional enterprise computing. For these systems, flexible SOA technologies provide the means to teach the world's most powerful transactional workhorses a new set of business-critical tricks.

However, some practical impediments stand in the way. SOA efforts require mainframe developers to leverage unfamiliar Java-based tools to intersect with traditional CICS and IMS transactional environments. This situation is rife with technical and cultural challenges, since success also requires cooperation between typically autonomous COBOL and Java development teams. As a result, mainframe customers who wish to pursue SOA reuse opportunities are ripe for better approaches. Enter IBM's Rational Enterprise Generation Language (EGL) solutions, a modular programming model that enforces good development practices and asset reuse, but does not force deployment compromises or require a full blown paradigm shift in developers' behavior.

## ***Challenges to Mainframe SOA Development***

The challenges IBM System z customers face during the process of service transformation fall into three general areas:

- **Existing Applications** – The vast majority of mainframe applications, which tend to be monolithic entities, are not written with reuse or modularity in mind. As a result, while they capture/leverage an enormous amount of business knowledge they are also difficult to repurpose for new processes or requirements. In addition, since maintaining and extending these applications is expensive they tend to absorb a significant portion of their owners IT budgets. Given the importance of such applications, organizations may endeavor to rewrite them in order to enhance their flexibility, extensibility, and componentization. However, such efforts tend to be highly costly and complex since success depends on leveraging developers' business domain knowledge and few "legacy" developers are capable of rewriting applications using Java and SOA.
- **Platforms and Middleware** – Though most people tend to think of IT in current terms, a longer view reveals that platforms and middleware are in a constant state of evolution. This situation can be aggravated as companies consolidate via downsizings, acquisitions, and mergers. The result is a proliferation of disparate computing and middleware platforms, increasing complexity in lugubriously merged IT infrastructures, and the fragmentation of skills among disposed IT staff and developers. These issues all contribute to rising IT costs and compromised development processes that can hamper a company's ability to respond effectively to shifting business requirements.
- **Staffing Skills** – As IT becomes ever more complex, computing professionals and the tools they use have become increasingly specialized, decreasing the portability of developers' skills. Not only do CICS developers and Web developers follow separate paths but they also travel in mutually exclusive vehicles. The result? Business process fragmentation and sub-optimal governance due to siloed applications and developers stranded on platform-centric "islands." In addition, there is a widening disconnect between IT and LOB managers, a problem exacerbated as aging baby boomer professionals are replaced by younger managers with little if any knowledge of or interest in the inherent difficulties and costs of properly maintaining and supporting legacy technologies.

## ***The Rational Difference and IBM's System z***

What is Rational EGL? In short, EGL is a simplified, easy to learn, high level programming language that allows developers to quickly write full-function programs designed to solve specific business problems. In EGL,

middleware programming and Java/ Java EE details are hidden from view, so even developers with minimal Web technology experience can easily deliver data to browsers. Developers write their business logic in EGL, which is used to generate Java or COBOL. Then runtime artifacts can be deployed to whatever runtime platform has been targeted for the application. Along with capturing notable business efficiencies, EGL also reduces developers' training costs and increases their productivity, crucial points for companies striving to efficiently adopt emerging Java EE and Web Services standards.

EGL offers critical support to mainframe customers by helping them address numerous SOA development and adoption challenges. Most importantly, EGL integrates with and easily extends existing COBOL applications via simple Call COBOL functions that can be used to build services, exchange messages, and natively interact with Web pages. With EGL, there are no wizards or seams between Java and COBOL, so there is no need for programmers to marshal parameters and map data types, to understand or learn Java JCA, or to use special adapters. As a result, programmers can create new CICS and IMS services quickly and easily. Additionally, since like Java and COBOL tools EGL tools are totally integrated with the Eclipse workbench, mainframe developers can seamlessly shift from one language to another.

At the same time, EGL is a modern programming language designed for organizations' most up-to-date computing requirements. Because of its seamless integration with Java-based frameworks (e.g. JSF) and in-built service functionality, EGL is a valuable tool for re-using legacy applications in new SOA-based offerings. Developers concerned about being shoehorned into deployment compromises will appreciate EGL's native deployment to CICS, IMS, and WAS, and its full support of mainframe batch programs. But EGL's benefits do not end with IT staff. Experienced and college-educated developers can typically achieve EGL proficiency in a matter of weeks, costing companies far less time and money than ground-up Java and Java EE re-training.

## ***Rational EGL in Real World Mainframe SOA***

Good enough, but what sorts of tangible effects can Rational EGL have on mainframe development efforts? Here are two examples:

1. IBM conducted an internal study aimed at benchmarking Rational EGL productivity by using the EGL language, the EGL tooling contained in Rational Application Developer and the SUN Pet Store specifications. Compared against the time measured in a previous study where the Pet Store application was developed simply with hand coding in Java using a Java IDE with no power tools, using EGL and EGL tools demonstrated significant reductions in development time and effort.
2. In addition, Rational EGL efficiencies have real world implications. For example, a noted European banking institution approached IBM for a way to unify application development across mainframe and UNIX platforms and WAS and IMS transaction managers. The company was pursuing an active acquisition strategy and looking to better leverage its assets and effectively achieve cost reductions through new synergies and IT/staff integration.

To achieve this, IBM helped the customer shift from a monolithic application environment to component-based architecture utilizing browser-based and open systems, product factories, and multi-channel deployments. A core group of about 250 mainframe developers who trained in EGL were re-classified as multi-employable developers for UNIX and mainframe, and then were deployed in teams in various European locales and India. These teams develop solutions and business components for deployment and reuse within each company subsidiary. Overall, by using Rational EGL solutions, IBM's customer expects to achieve a significant reduction in time to development and adoption of valuable new business services that run regardless of platform, flexibly delivering value across an evolving organization.

## ***Mission Accomplished?***

The reasons for the growing interest in and deployment of SOA-based solutions are simple and compelling. SOA provides the engine for organizations to reuse existing IT assets for services supporting new applications and processes, enhance IT/business performance, maximize their return on investment (ROI), and delay, or even avoid costly IT purchases. At the same time, SOA and IBM's signature System z mainframe solutions are ideally suited for one another, since SOA technologies provide the means to leverage mainframe platforms to support new business-critical service offerings. While some practical impediments stand in the way of mainframe SOA development, IBM's Rational EGL enables highly effective development practices and asset reuse, and does not force compromises or significant shifts in developers' behavior.

EGL supports mainframe customers by helping them address SOA development and adoption challenges, naturally integrating with and easily extending existing COBOL applications. At the same time, EGL is a mod-

ern programming language designed for the most up-to-date computing requirements, thus allowing businesses to support a variety of new Java-based front end applications while back-end processes continue to run on CICS. This blending of new and traditional technologies into business-centric services provides organizations the means to gain numerous technical and business benefits and to improve the performance and efficiency of IT developers. Overall, we believe that IBM's Rational EGL is a powerful tool that enforces good software development practices, good software architecture processes, and good reuse results. Enterprises that wish to pursue SOA deployments and gain maximum advantage from their mainframe systems would be well-advised to investigate IBM's Rational EGL.

© 2007 Pund-IT, Inc. All rights reserved.

*About Pund-IT, Inc.*

*Pund-IT emphasizes understanding technology and product evolution and interpreting the effects these changes will have on business customers and the greater IT marketplace. This report is the result of sponsored research developed by Pund-IT, Inc., which believes its findings are objective and represent the best analysis available at the time of publication.*