



Moving Your COBOL Assets to Microsoft .NET:
Fujitsu NetCOBOL for .NET

— Fujitsu Software

► Hurwitz Report



Moving Your COBOL Assets to Microsoft .NET: Fujitsu NetCOBOL for .NET

— Fujitsu Software

iii Executive Summary

This white paper outlines the options available on the market and discusses a new solution — Fujitsu Software's NetCOBOL for .NET — and its approach to moving COBOL assets over to Microsoft's .NET platform.

1 Extending Code Assets to New Architectures

Hurwitz Group strongly recommends that companies look for a solution that leverages current COBOL assets and unifies them with new development.

4 A New Solution for COBOL Migration: Fujitsu NetCOBOL for .NET

Microsoft's .NET framework provides an environment that enables a new level of integration between multiple programming languages.

5 A Cost-Saving Scenario for Migrating COBOL Assets to Microsoft's .NET Platform

Hurwitz Group has created a model for illustrating a cost comparison for extracting business logic from COBOL applications so that it can be componentized and moved to the .NET platform.

7 Real-World Example: Affiliated Network Services

The move to the NetCOBOL for .NET and the Microsoft .NET platform has cut ANS's development time in half.

9 Conclusion

Microsoft's .NET platform provides the framework to bring legacy assets forward so that they can be used in new application development.

A Hurwitz Group white paper written for:

Fujitsu Software Corporation
3055 Orchard Drive
San Jose, CA 95134-2005
Tel: 800 545 6774 or 408 428 0300
cobol@netcobol.com
www.netcobol.com

Published by:
Hurwitz Group, Inc.
111 Speen Street, Framingham, MA 01701 ► Telephone: 508 872 3344 ► Fax: 508 872 3355
Email: info@hurwitz.com ► Web: www.hurwitz.com

June 2002

© Copyright 2002, Hurwitz Group, Inc.
All rights reserved. No part of this report may be reproduced or stored in a retrieval system
or transmitted in any form or by any means, without prior written permission.

EXECUTIVE SUMMARY

Using the Web as a distributed computing platform helps extend the reach and value of corporate computing assets by improving access to information and services for customers and partners regardless of where they are. Through this path, companies can more flexibly deliver compelling service, but the real value comes when web initiatives complement an existing technology infrastructure and lower the cost of maintaining those legacy environments. For the future success of web development and delivery of services, enterprises must focus on simplified integration with existing IT assets to improve overall interoperability while keeping total cost of ownership low. This focus on leveraging the current technical assets and developer resources drives savings and extends the value locked up in core legacy transaction systems.

One of the barriers that companies face in moving their legacy assets into their web-based development initiatives is the expense associated with migrating applications. The first step, then, is to evaluate options so that they can make a sound business decision. This white paper outlines the options available on the market and discusses a new solution — Fujitsu Software's NetCOBOL for .NET — and its approach to moving COBOL assets over to Microsoft's .NET platform.

Extending Code Assets to New Architectures

Hurwitz Group research shows that many enterprises are stymied by their inability to best utilize their legacy assets and often don't have the appropriate tools available to help ensure success. This white paper covers the key issues associated with getting the most out of legacy assets. It characterizes the challenges impacting companies as they modernize application architectures, including aligning legacy with Web Services and specifically .NET. Examination of a baseline business case assists in understanding the legacy extension issue. Through highlighting options and positing a ROI-based model, we present information that can serve as a guide to making positive choices for improved legacy value. To begin, consider these common business situations:

- ▶ A Midwestern transportation company with hundreds of field offices across the country needs to provide all its distributed employees with access to core billing and freight management applications that are housed on a centralized mainframe. The company needs to tie these mainframe applications into the Web so that the mainframe data can be used in conjunction with newly developed applications for customer pricing. This modernization effort is geared toward improving field office coordination by using the Web to facilitate collaboration. Ultimately, this company sees providing this information as a Web Service as being essential to its long-term application development goals.
- ▶ A municipal law enforcement agency needs to arm its field agents with critical data that can be accessed on remote PCs. The agency needs to design a web-based system that pulls in text-based records information from the current mainframe application and couples this with graphical images to provide the agents with on-the-spot criminal record history. The focus is to better arm officers to make improved decisions in the field under extreme time and technology constraints. Business leaders in this organization also want to move to a Web Services model to further improve the availability of critical field information.

In the preceding examples, both companies want to extend the reach of their core mainframe COBOL applications and migrate them to the Web and Web Services to give their employees a richer set of information resources to do their jobs more effectively. While realizing this benefit, both companies must closely examine how they can lower their costs by unifying their mainframe environments with their new development initiatives on the Web. Both are challenged with the business need to extend information and yet lower the cost of the overall technology infrastructure that supports their goals. They need to effectively bridge the gulf between their legacy systems and new systems development so that all their business-critical data can be brought to bear without added overhead.

Increasingly, Hurwitz Group sees many organizations facing this type of challenge: extend more services to the Web while keeping IT costs low. Hurwitz Group strongly recommends that companies in this situation look for a solution that leverages current COBOL assets and unifies them with new development.

Why should enterprises look to extend rather than build from scratch? Legacy applications drive many of the core operational facets of business and drive vital business processes that are central to your business's value. Given the critical nature of these applications, you cannot afford to rip out and rewrite all your applications in a new language to take advantage of Web technology. While the demise of COBOL is frequently discussed, COBOL programs contain critical business logic and data that must be maintained and brought forward to further serve your enterprise.

The COBOL programming language was developed over 40 years ago specifically for developing business applications. And it continues to drive applications that involve large amounts of data, such as payroll, inventory, logistics, and insurance ratings systems. With its English-like syntax, COBOL is highly readable and easily documented. It also provides a standard approach because a COBOL program must be structured to contain defined program elements. COBOL is highly portable, enabling the core syntax for writing applications on mainframes to be used on PCs with no changes. With the widespread use of COBOL in business-critical operations, the language certainly isn't going away. What many companies are determining is how best to integrate their COBOL-based systems to connect into new Web development initiatives.

Many enterprises are working on integrating their COBOL assets into their Web applications. Some are adopting Web Services to carry out this integration. In its Web Services PRO Study of 2002, Hurwitz Group uncovered Web Services adoption and implementation plans, and the top two business benefits cited by respondents were:

- ▶ Web Services would enable their company to use its IT investments more effectively.
- ▶ Web Services would enable better integration of their company's business processes.

Following close behind in priority was the need to adopt Web Services to help the companies differentiate themselves competitively. They were concerned that if they did not adopt Web Services computing, they risked being bypassed by their competition.

Hurwitz Group believes that Web Services is important to the future of the enterprise computing environment. Companies are making strategic choices about their future Web Services plans and the platforms for developing and deploying Web Services. Over half of the companies in the Hurwitz Group survey responded that they viewed Microsoft's .NET platform as a strategic platform for Web Services development. While .NET provides the framework for

integrating legacy applications with new applications, companies also need a strategy for bringing their legacy code into the .NET architecture.

How will these companies embrace Web Services and .NET while they still make effective use of their COBOL assets? This is precisely the question that the transportation company and municipal law enforcement agency previously mentioned are grappling with. In answering this question, Hurwitz Group recommends looking for benefits in these areas:

- ▶ Moving from mainframes to PCs to lower hardware costs.
- ▶ Reducing the operational costs of maintaining COBOL code. Years of modification and patching have resulted in COBOL applications with monolithic structures that are inflexible.
- ▶ Making the business logic contained in legacy code portable so that it can be reused and extended throughout the enterprise. This enables the code to participate in additional deployment and delivery options such as wireless devices, PCs, and web applications.
- ▶ Synchronizing new development efforts with legacy assets to ensure that the value contained within legacy systems is brought forward.

To gain these financial benefits, organizations must evaluate their options and determine which approach best addresses their needs and fits their budget requirements. Approaches include:

- ▶ **Conducting a complete rewrite of legacy applications to a new development language.** In taking this approach, companies would need to justify the time and cost of writing new applications that replicate the functionality in existing legacy applications. If the company decided to rewrite the applications in Java, for example, and did not have enough Java programming expertise in-house, it would need to factor in the cost of hiring Java programmers, who typically command high salaries because of the limited pool of skilled Java developers on the market. Or if the company chose to train its existing development staff on Java, it would need to factor in the cost of training classes, as well as the ramp-up time after training, for each developer involved.
- ▶ **Creating hard-coded interfaces between legacy applications and web applications.** While this approach provides a short-term fix for accessing legacy data, it introduces scalability and maintenance complexities. If any changes were made to the application, the hard-coded connection would no longer work. Handling a few of these connections would not be a big problem, but when organizations have multiple one-to-one integrations that need to be modified by hand whenever a change is

made, a maintenance issue arises. IT resources to support and maintain the integration code would be required, and if a company has a large number of point-to-point integrations, this approach could take dollars and time away from more strategic development projects. This strategy also does nothing about moving the COBOL applications away from costly mainframe charges.

- ▶ **Migrating legacy code to new development environments.** This approach enables the integrity of the business logic contained within the code to be maintained while also allowing it to interact with new technology on an object level. Essentially, it digs into the existing legacy code and identifies the core business logic of each application as well as the business process relationships among different legacy applications. Once this logic and these processes are identified, they can be exposed as a Web Service and componentized so that they can be shared and reused in multiple applications.

A New Solution for COBOL Migration: Fujitsu NetCOBOL for .NET

Microsoft's .NET framework provides an environment that enables a new level of integration between multiple programming languages. Microsoft's development environment for .NET, Visual Studio.NET, includes the common language runtime (CLR) facility, which allows code developed in different languages to be integrated to build applications. As a result, developers with different coding specialties, such as COBOL, C#, and Visual Basic.NET, can work together in the same development environment. It also facilitates code reuse to extend business assets throughout the enterprise.

The .NET platform has been written from the ground up to support XML Web Services. The technology enables software to be delivered as a set of services, with XML serving as an open standard format for transferring data on the Web. Fujitsu's NetCOBOL for .NET contains a COBOL compiler that enables a company to extend valuable COBOL assets to participate in the .NET platform, where it can be integrated with other languages and tools within the .NET framework.

Main features of the product include:

- ▶ Support for all COBOL file types, which protects the integrity and the structure of data
- ▶ Integration with the Visual Studio .NET development environment, which enables COBOL programmers to create COBOL applications using Microsoft's IDE
- ▶ The ability for COBOL developers to create ASP.NET web
- ▶ Applications and services using COBOL as the scripting language.
- ▶ Ability to create rich graphical user interfaces for a company's applications with Windows Forms

- ▶ Support for XML Web Services to provide a productive pathway that enables COBOL code assets to participate in Web Services application deployments

The process for moving COBOL code is straightforward. The NetCOBOL compiler takes procedural COBOL code and outputs Microsoft Intermediate Language code (MSIL). The code is then packaged as an MSIL object that can be executed in the .NET platform by the CLR. This provides flexibility for COBOL assets, which can now interact with other languages within the .NET framework and be used to build Web Services. With NetCOBOL, developers can bring in the rich business logic from their COBOL programs and enable it to interact with and be deployed on new technologies, such as web applications, handheld devices, and web pages.

In addition to the technical benefits of moving COBOL code into the .NET environment, which enables greater code reuse and lower costs for integration, NetCOBOL offers a richer collaboration environment for development teams. NetCOBOL provides COBOL developers with a productive transition to Microsoft's Visual Studio .NET development environment; they can develop COBOL applications within Visual Studio. This capability reduces the training time for developers because they are building on their current skill sets rather than learning an entirely new programming language.

Most development teams consist of team members using different programming languages and different development environments to do their work. This practice often forms a separation between COBOL developers and Windows developers because their tasks do not overlap and they cannot easily share code assets. With Visual Studio .NET, organizations that use the environment for Windows development can unite their development team under a central IDE. Because the CLR facility supports over 20 programming languages, development organizations can consolidate their developers under a unified IDE, enabling developers to share a common workspace and providing an easier migration path for fusing code assets.

A Cost-Saving Scenario for Migrating COBOL Assets to Microsoft's .NET Platform

Hurwitz Group has created a model for illustrating a cost comparison for extracting business logic from COBOL applications so that it can be componentized and moved to the .NET platform (see Table 1). This model comes from interviews with many companies executing on legacy strategies. In the model, the percentage of time allocated to each task is an aggregated estimate that has come from the interviews. Often, Hurwitz Group spoke to companies that jumped too quickly into the process and found themselves overwhelmed by the complexity, which has led to numerous project failures. While the allocated percentages may not specifically reflect your organizational efforts, Hurwitz Group recommends a solid requirements analysis

prior to beginning such a project. The baseline scenario involves a project team of two senior COBOL developers working on an application consisting of 100,000 lines of code. The tasks associated with the project are:

- ▶ Building knowledge of the source application
- ▶ Identifying the business rules within the application
- ▶ Extracting the rules by hand
- ▶ Transferring knowledge of applications between a COBOL developer and a Windows developer
- ▶ Replatforming the application to run on .NET

Table 1. Cost-Comparison Migration Model: "As-Is" Baseline	
Basic Assumptions	
Number of hours per year	2,080
Average salary + benefits per developer	\$95,000
Hourly salary per developer	\$46
Labor Assumptions	
Number of developers working on the app	2
Number of lines a developer can fix per hour	21
Portion of time spent:	
Building knowledge of source app	10%
Identifying business rules	25%
Building relationship map for business rules	25%
Extracting rules by hand	25%
Transferring knowledge for .NET expert	5%
Replatforming	10%
Total	100%
Scenario Assumptions	
Lines of code in the app	100,000
Total Hours and Labor Costs	
Number of hours to complete project	2,381
Cost to complete the project	\$108,745

The baseline breaks out percentages for time spent on each task, calculates the number of hours necessary to complete the project, and provides the total project cost of \$108,745.

Then we compared this scenario with one that involved conducting the same process with Fujitsu NetCOBOL for .NET (see Table 2). In the new scenario, we take a conservative approach where using Fujitsu's product, and the development team is able to reduce the time spent on the project tasks from 2,381 hours to 1,339 to achieve an overall project savings of \$47,576. Time savings resulted because the developers were able to migrate their existing COBOL code using Fujitsu's compiler that can directly target the .NET platform, rather than having to extract the code by hand, and hand it off to a .NET developer to handle the replatforming tasks to run the application in .NET.

Table 2. Cost-Comparison Migration Model: With Fujitsu NetCOBOL for .NET

	"As Is"		Reduction with Fujitsu			Total Project Savings	
	Hours	Cost (\$)	Reduction (%)	Hours	Cost (\$)	Hours	Cost (\$)
Building knowledge of source app	238	10,875	25	179	8,156	60	2,719
Identifying business rules	595	27,186	50	298	13,593	298	13,593
Building relationship map for business rules	595	27,186	50	298	13,593	298	13,593
Extracting rules by hand	595	7,186	50	298	13,593	298	13,593
Transferring knowledge for .NET expert	119	5,437	25	89	4,078	30	1,359
Replatform	238	10,875	25	179	8,156	60	2,719
Total	2,381	108,745	–	1,339	61,169	1,042	47,576

The scenario in Table 2 illustrates how Fujitsu's product can potentially reduce the time and cost associated with migrating COBOL applications to Microsoft .NET. It does not take into account the full process of migrating and testing applications to ensure that the integrity of the application is verified on the .NET platform.

Real-World Example: Affiliated Network Services

Affiliated Network Services (ANS) is a healthcare benefits service and technology company in Chicago, Illinois. One of its main products, ANSLink, enables healthcare providers, such as dentists, to determine patient insurance eligibility via the Web. Dentists can look up a patient's coverage details and determine what procedures will be covered by that patient's health plan, submit the claim to the insurance carrier, and track its status via the system. ANS also works with insurance carriers to interface with their systems to access patient data and enable electronic claims submission and electronic funds transfer between healthcare providers and insurance carriers.

ANS wanted to use the latest technology to drive its business and made the decision to move to the .NET platform. Previously, ANS had been using COBOL, but limitations of current data transfer schemes and standard web protocols made the company seek alternatives, and it decided to move to an XML-based system. In evaluating what language to use in conjunction with its XML development, ANS investigated moving away from COBOL and considered recoding in a different language. ANS instead chose Fujitsu's NetCOBOL because it provided tight integration with the .NET platform, enabled the company to leverage its current COBOL assets, and move it forward to the Web Services development paradigm.

The move to the NetCOBOL for .NET and the Microsoft .NET platform has cut ANS's development time in half. .NET allows it to reuse program components across applications and interface different applications into the application server and database server for better cohesion throughout the entire computing architecture.

Integration is easier and less expensive through the use of XML for data transfer. Some of the biggest issues with moving to Web computing are the quality of data transfer and the performance pitfalls of moving large amounts of data over the Web. Some corporations have concerns about allowing outside parties access to their internal network. Through the use of XML templates, ANS can set up a controlled means for defining the type of data that it is transferring and how it accesses it. Once developed, the XML templates can be modified to exchange data with different partner and customer systems.

ANS is also finding that it can better serve its customers. It has enhanced its ability to pull data into its applications and can push out data more easily to its clients. While some of its clients still require batch processing, ANS now has the ability to conduct real-time transactions with clients that have adopted XML into their systems. The more insurance carriers adopt XML, the easier it will be for ANS to exchange data with them and accelerate healthcare transactions.

Overall, ANS has found that the move to NetCOBOL and .NET has been a painless process that has enabled the company to expand its technological reach and agility in conducting business. It has cut development time and costs while also making the development and integration process easier.

Conclusion

In today's competitive environment, companies must put all their assets to work for their business. With the majority of the world's data and business logic still residing in legacy applications, enterprises need to implement a strategic solution to bring those rich assets forward to fuel new computing initiatives. Before embarking on a legacy migration project, companies need to assess their business needs and examine how their technology infrastructure is supporting their business strategy. They need to assess how their systems will support future business growth and enable competitive differentiation and then weigh their options for legacy application migration.

The Web Services computing model is gaining acceptance in the market and being adopted by organizations as a solution to help solve the key problem of integrating their computing assets. Microsoft's .NET platform provides the framework to bring legacy assets forward so that they can be used in new application development.

Fujitsu NetCOBOL for .NET provides a compelling solution that can lower the cost of legacy code migration as well as an easy-to-use environment that is very approachable to COBOL developers. It enables organizations to leverage their COBOL assets and provides the flexibility to deploy them on the most cost-effective, business-critical deployment target, whether that is PCs, wireless devices, or Web applications. Moving your COBOL assets into the .NET platform will give your organization a new level of interoperability within your computing infrastructure and bring the value of your business assets further into your extended enterprise, enabling you to better serve your customers.



About Hurwitz Group

Hurwitz Group, an analyst, research, and consulting firm, is a recognized leader in identifying and articulating the business value of technology. Known for its real-world experience, consultative style, and pragmatic approach, Hurwitz Group provides strategic guidance to its clients by delivering analysis, market research, custom content, and consulting services. Clients include Global 2000, software, services, systems, and investment companies.