

IBM System z case study: Con-way Freight
Evolving an SOA strategy built around the System z

Executive Summary

In 1996 Con-way Freight was looking to evolve its legacy applications, it turned to component-based development, a forerunner of SOA, and over the ensuing decade, Con-way's initial mainframe-centric CBD efforts using CA-Gen evolved through Java/J2EE to today's SOA built around System z services. In the process it built reusable services using a variety of IBM WebSphere and Rational tools and System z third party tools. These services address infrastructure and business application needs, allowing Con-way developers to repeatedly enhance its legacy applications running on the System z.

The result: faster and more efficient development, streamlined integration, and cost savings through repeated reuse of services. Con-way Freight has successfully demonstrated the versatility of the System z and its value as a SOA platform.

Business Challenge

Con-way Freight, a major component of Con-way Inc, needed to evolve its existing legacy systems. The aging applications were complex, poorly understood, difficult to change, and badly in need of modernization. Still, the company was not likely to abandon the applications; they sat at the heart of the operation and handled the core business processes.

Since the company wasn't going to get rid of the legacy applications or the mainframe, the challenge facing the solutions team became how to change and modernize the applications to enable the company to better compete going forward. It was 1996 and in this new environment customer satisfaction loomed more important than ever before with transportation and logistics playing a vital role in its customers' operations. In addition, Con-way Freight was committed to offering a broad set of service offerings.

Background

Con-way Inc., the parent of Con-way Freight, bills itself as the premier provider of reliable, regional, inter-regional and nationwide less-than-truckload (LTL) service to customers large and small across its integrated, single North American network of LTL operating locations.

As described by the company, Con-way Inc. provides the full range of transportation and supply chain management services through its three primary subsidiaries — Con-way Freight, Con-way Truckload, and Menlo Worldwide Logistics. It operates on five continents and generates more than \$4 billion in sales from customers in manufacturing, industrial and retail industries. Con-way Freight provides direct service from the largest single LTL service center network in North America.

With more direct delivery points the company is able to reduce transit times while ensuring more reliable on-time performance. In a business world that puts a premium on speed, economy and reliability, Con-way facilitates its

customers' success in their race to market. For nearly 25 years. Conway-Freight's direct, reliable service through 450 locations operating as part of a unified network provides day-definite shipping services throughout the continental U.S. and Canada, as well as Hawaii, Alaska, Puerto Rico and Mexico.

Solution strategy

According to Shibashis Mukherjee and Jim Haek, Con-way Freight Principal Enterprise Architects, and Jeon Rezvani, Senior Manager of Enterprise Architecture and Mainframe Software, an IT team that, at points, would surpass over 100 developers began to tackle the problem of how to modernize mission-critical mainframe applications in a way that would allow them to be changed and changed again as the business changed while still retaining the advantages of the System z at the heart of the operation. Today the solution would be obvious, services and a service-oriented architecture (SOA). In 1996, when the effort began, SOA was not a widely embraced concept, especially where the mainframe was concerned.

The solution evolved in several stages. Initially, the team turned to component-based development (CBD), the leading edge at that time. The CBD approach revolved around large-grain chunks of code, called components, which performed an identifiable task. The components were discrete and could be reusable. In short, CBD was a forerunner for what would become SOA with different terminology.

CBD at Con-way was completely mainframe-oriented. The components were, in effect, breakouts of existing mainframe code and were used to call other mainframe code. Each component had a strictly defined interface. Within the context of the mainframe these initial components could be combined and recombined to provide new or enhanced functionality. The big limitation to this strategy: the components had to be called from the mainframe, which reduced their scope.

The next step in the evolution saw Con-way extending the components to a new environment, such as Java, where they could interact with the Web. The team used J2EE to create Web proxies by wrapping the mainframe components as Enterprise Java Beans (EJBs). This led to the establishment of a middle tier consisting of EJBs, which would later become Web services. However, all the business logic continued to run on the System z.

In the final stage the EJBs along with their methods were wrapped as standard Web services. With this, the team had arrived at SOA as we know it today.

SOA on the System z

SOA initially was conceived and developed with the distributed, open systems world in mind. The hope was that it would allow different, incompatible systems to interoperate. To do this, SOA would define a piece of functionality as a discrete service and present a standard interface to that service. The interface would mask the inner workings of the service. Only a few basic standards, HTTP, XML, and SOAP, were required to call the particular functionality and exchange data. SOA, in effect, was latest iteration of efforts to integrate disparate systems going back to remote procedure calls.

There was no reason why the System z couldn't play this game. In fact, there was every reason for the System z to play a key role in enterprise SOA initiatives. As the Con-way team noted, the enterprise's data already resided on the System z and much of the enterprise's business logic and rules were embedded in applications running on the System z. By 2006 Independent Assessment already was writing and publishing reports on SOA initiatives revolving around the System z and about mainframe tools from IBM, Seagull Software, DataDirect, CA, and others to facilitate the effort.

By then SOA has become one of the hottest IT buzzwords because it promised an easy way to access embedded business logic and functionality as a service and exchange data for the purpose of integrating loosely coupled distributed systems. There was no reason it couldn't be used to unleash valuable

business functionality encapsulated in mainframe data and applications and make it available throughout the enterprise.

In the process, due to SOA's reliance of widely accepted standards, it could expedite the integration of mainframe and distributed systems. Through XML and standards-based messaging disparate systems could exchange data and call functionality without resorting to bulky integration middleware or cobbling together point-to-point integration as had long been the common practice.

Today, with System z applications natively handling XML and standard messaging protocols and key mainframe engines like CICS able to handle the Web natively, the System z can stand at the center of enterprise SOA initiatives, enabling a level of interoperability that previously was difficult and costly to achieve and maintain.

Benefits of SOA

Proponents of SOA typically emphasize two big benefits: 1) cost savings through the reuse of services and 2) better alignment of IT with the business through the ability to change systems more readily in response to changes in the business. Con-way certainly achieved savings through reusable components and then reusable services. Code reuse not only saves money but it speeds development since the code doesn't have to be reinvented repeatedly.

For mainframe shops like Con-way, SOA delivers other benefits that are equally important:

- Ease and speed of integration
- Short path to interoperability of disparate systems
- Ability to more effectively leverage existing software assets, including the business logic in legacy and CICS applications
- Ability to more easily leverage data residing on the mainframe
- Ability to tap the reliability, scalability, performance, and security of the mainframe for distributed applications

The value of reuse and the ability to leverage existing business logic and data are significant although not apparent initially. As Con-way discovered, the

benefits of reuse accelerate over multiple projects in terms of faster development at lower cost.

Con-way Freight IT Infrastructure

Con-way Freight is a long-time mainframe shop. Two System z10 machines reside in the data center, an enterprise class Model 704 as the primary production system and a Model 405 for batch processing and testing.

The mainframes have VM LPARs, which can run multiple instances of Linux, and with an upgrade to the z10 the company acquired another IFL. In addition, it runs all its core business systems, including customer systems, shipments, pickup, and delivery on the System z. It also runs PeopleSoft, which it encapsulated as a business component and now runs as a web service.

Con-way runs WebSphere Application Server, Rational Application Developer, and WebSphere Everyplace (for mobile application access) on a distributed platform, and then DB2, VSAM, CICS, and RACF, along with COBOL on the System Z; it also utilizes tools like CA Gen for back-end development and TIBCO for messaging middleware. It runs both SNA and TCP/IP networking.

Implementation

The team opted for the System z because, in 1996 when the effort began, that was where the key business application logic and data resided. With the System z and CICS, the team felt the system could scale if necessary to more than a million transactions a day. And it delivered the best total cost of ownership at that time and continues to do so today.

The team builds three types of components:

1. Infrastructure components—such as print requests
2. Core business components—such as customer or shipment functions
3. Service components—such as pickup, delivery

By now Con-way has more than 40 reusable components, over half of which are core and service components.

Implementation stretched over a period of that spanned the evolution of SOA technology from the early, pre-Java, CBD to the Web-enabled SOA deployed today. The initial component-based services, each with a precisely defined interface, were limited in scope and had to be called from the mainframe. As the approach evolved, the Con-way team broadened the scope, eventually wrapping Java components as web services.

The team used CA Gen for developing the back-end components. It uses Rational Application Developer and WebSphere Application Server to create and deploy the front end components and web services.

The applications on the back end run assembler and COBOL CICS against VSAM and DB2. Incoming requests are translated from IP to SNA. Con-way intends to deploy the CICS transaction gateway, which will enable it to handle TCP/IP natively. For messaging middleware it uses TIBCO.

SOA governance has proven an ongoing challenge. Initially the SOA developers worked closely with the IT architects to define a methodology, use cases, and core concepts. The result was a loose, informal governance structure that they have had to periodically tweak as issues arose. The lack of a central service repository like WSSR has proved problematic.

**A subset of Con-way
Systems and Software,
on System Z &
distributed**

System z10, models 704
and 405

IFL

CICS,

DB2

SNA, TCP/IP

RACF

WebSphere Application
Server

Rational Application

Developer

WebSphere Everyplace

Oracle

Peoplesoft

CA Gen

TIBCO

Results

Con-way began this effort as a way to streamline development in the face of large amounts of unwieldy legacy code. The team's initial success with CBD demonstrated to top management the value of the component/services approach and won their full backing.

Through the subsequent evolution of the SOA approach the team has been able to demonstrate faster, more efficient development. It has also proven that it can leverage existing legacy code to deliver new business value. With a growing library of reusable components, the team also is achieving high levels of reuse, which saves both time and money. Finally, the SOA approach makes it easier to change systems as the business changes, resulting in a more agile and responsive organization.

Lessons Learned

Over the years, the Con-way team has learned some valuable lessons:

- Define the right granularity of services to maximize reuse
- Tighten up governance to avoid duplicate services and establish the right boundaries
- Win and maintain management support through demonstrated benefits
- Establish a taxonomy of well-defined services
- Develop a services mindset

Still to be fully resolved are the repository and governance issues although the team is working on those.

Independent Assessment analysis

The Con-way Freight experience highlights a key lesson about SOA: that it is a journey undertaken over time, not a one-off project. Similarly, the payoff comes not from the initial project but from successive projects that take advantage of the reusable services developed for earlier projects.

SOA is platform-agnostic, which Con-way demonstrates. SOA may have come out of the distributed systems world, but Con-way's use of the System z proved that the same capabilities that made the System z its core production workhorse make it ideal for SOA. And, by taking advantage IBM's growing System z ecosystem of ISVs, the company found familiar, effective System z tools to aid its SOA efforts from a variety of vendors.

About Independent Assessment

Independent Assessment (<http://www.independentassessment.com>) is the IT and business assessment, analysis, and writing service of Alan Radding, an independent business and IT analyst/writer for over 20 years. It provides independent ROI and TCO analysis, competitive assessment and positioning reports, case studies, and Web content.

Independent Assessment publishes *dancingdinosaur*, the independent blog covering the System z, <http://dancingdinosaur.wordpress.com>