

Leveraging Legacy System Dollars for E-Business

Len Erlikh

Although many firms have rapidly and enthusiastically adopted the distributed architectures, many more are stuck with mainframe-based mission-critical systems that continue to isolate them from their partner, supplier, and customer systems. Indeed, IDC estimates there are more than 10,000 large IBM mainframe sites worldwide with 200 billion lines of legacy code still in use.

Most companies want to transform their applications to meet new business demands, but because legacy systems tend to be unwieldy, monolithic, and inflexible, many firms regard modernization as somewhere between improbable and impossible. Reeling from the Y2K debacle and saddled with years of application backlog, the most these companies can hope for is to keep their legacy system alive.

And keeping it alive is getting more expensive. According to an informal industry poll, 85 to 90 percent of the IS budget goes to legacy system operation and maintenance. It is also becoming harder to find qualified personnel to do the maintenance. All of this makes it difficult to add new functionality and keep up with business requirements.

The ideal solution is to transform legacy systems to newer, more productive platforms so that companies can exploit faster and cheaper development technologies, like Java and XML (Extensible Markup Language). The focus then shifts to functionality, not the infrastructure, which means a company can respond more quickly to its changing business requirements and technology enhancements.

Converting a monolithic legacy system to stand-alone components can turn this source of business knowledge into a competitive edge.

NOT A TRIVIAL PURSUIT

But this legacy transformation isn't trivial, which is why many companies avoid it. The e-business architecture emphasizes just about everything foreign to a legacy system—distributed heterogeneous platforms, component encapsulation, the merging of standards, openness. The challenge is to preserve the wealth of captured business knowledge and have the system fit into the component world of the new e-architecture.

RescueWare, legacy transformation software from Relativity Technologies, breaks business knowledge into stand-alone pieces, or *e-components*. The e-components are basically collections of objects that perform specific business services, have clearly defined application program interfaces (APIs), and are accessible through modern industry-standard protocols.

Because these e-components encapsulate individual business processes and because other components can freely access them, a company can more precisely control individual business processes. This divide-and-conquer approach allows companies to do rapid concurrent development. Each large-scale business process becomes a self-contained unit of manageable size, making it easier to deploy in a Web-based environment.

Legacy transformation in RescueWare begins with understanding what parts of the legacy system are worth transitioning to the e-business

Inside

**Legacy Modernization Resources
Hunting for Business Rules**

world. The software then extracts the business rules from those parts and moves the code, now in components, into the e-architecture or simply back to the old platform. So far, more than 20 firms with legacy systems as large as 2-million-plus lines of code have used RescueWare to transform their legacy systems to an e-business platform.

WHAT'S WORTH SAVING?

The first question any firm should ask is, "Is this system worth the work?" Figure 1 shows the region where legacy transformations make the most sense. Transformation strategies vary according to where in the region a system falls.

For example, you should typically replace a low-quality legacy system that offers generic industry solutions with off-the-shelf enterprise resource planning (ERP) packages. In contrast, a high-quality legacy system that provides a competitive advantage is worth nurturing unless external business pressures dictate change. Accounting, payroll, and HR systems are good candidates for ERP replacement. Risk management, logistics, and insurance rating systems typically offer the best strategic advantages.

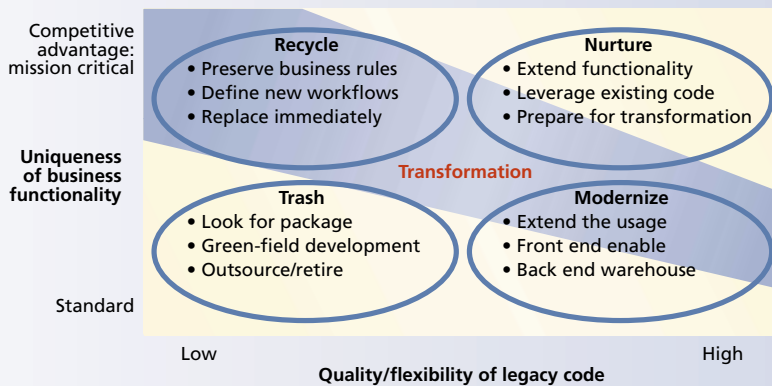
The modernization region in Figure 1 represents any technique used to bring the legacy system into the modern world, which for the most part means fitting it to an e-architecture. The best candidates for modernization are low-quality but strategic legacy systems. Here the best strategy is to recycle the systems by preserving their embedded business rules and organizing the rules into new workflows.

The second-best candidates are high-quality systems with standard functionality. These systems are the most likely candidates for integration with e-business, so you should modernize them by extending their use to an Internet platform. Thus, the systems remain on their present platform but are easily accessible through well-defined APIs.

WHAT ARE THE OPTIONS?

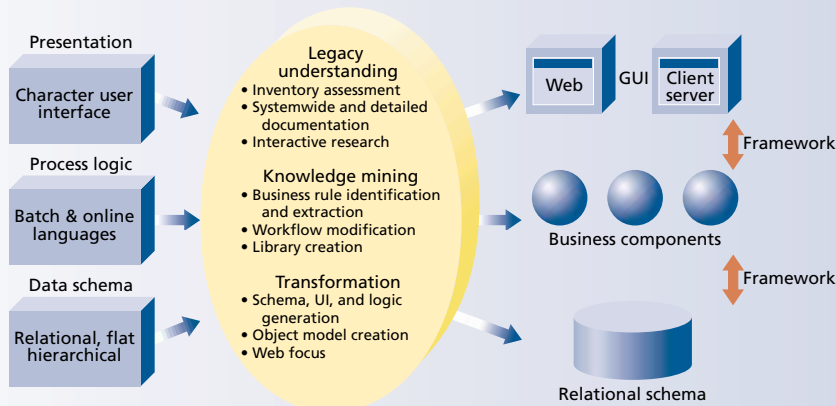
If your legacy system is worth modernizing, you can choose a strategy from three broad categories. The best approach depends on your objectives. The simplest is totally nonintrusive screen scraping, which represents new accessibility. A step up is legacy redeployment, which represents a new platform. The most complex approach is true legacy transformation, which rep-

Figure 1. Four options for handling a legacy system.



Whether you should modernize a legacy system and the method you use depend on the degree of the system's uniqueness to the business and the quality and flexibility of its code. The transformation region shows where legacy transformation makes the most sense. Most firms have at least some code in each category. Strategic business analysis of each system reveals the parts that require the modernization focus.

Figure 2. How RescueWare works.



RescueWare assumes that each legacy application has three main parts: presentation, process logic, and data schema. It automates the process of inventorying and analyzing legacy systems to extract business rules, model data, and partition code. RescueWare then transforms these parts of the system to stand-alone components and generates graphical user interfaces, business logic code, and database definitions.

resents a new business model. Transformation implies a specific set of techniques such as turning the legacy system into e-components.

Accessibility model

Screen scraping is the least expensive and fastest way to extend your legacy system accessibility to the Internet. You can achieve initial results in days, sometimes hours, with no modifications or disruptions to the underlying legacy system. But instant gratification has a price. First, screen-scraped systems can offer only the same functionality as that in the original system. If you start with a poor, rigid legacy system you end up with a poor, rigid screen-scraped system. Second, most legacy systems were developed for highly trained internal end users. Offering the same function to an untrained audience creates instant problems.

Platform model

Companies often face platform issues such as rapidly vanishing Wang or Unisys mainframes, which spur them to redeploy the legacy systems to a new architecture, usually the Internet. The idea is to keep the legacy system in a platform with the same language (Cobol to Cobol, for example) or create a carbon copy of the legacy system on a new language platform (Cobol to Java or C++, for example).

Relative to screen scraping, the new platform approach costs more and takes longer, but at least it gets the company off the existing platform and extends accessibility via the Internet-based architecture. Again, however, the new system is only as good as its predecessor. What was suitable on a monolithic, host-centric platform can be totally inappropriate for the new distributed heterogeneous architecture. A 10,000-line Cobol program is commonplace, but a 10,000-line Java program is a nightmare.

Business model

The new business model approach truly transforms the legacy application into a collection of reusable business components suitable for a particular target e-business platform. The components then become the foundation for the new architecture, and you can partition applications in a variety of ways—fully redeploy components to the new platform or use proxy-based wrappers that keep the components on the host platform but extend their accessibility to Internet technologies. Legacy transformation offers more benefits than the other modernization



Legacy Modernization Resources

Web sites

<http://www.systemtransformation.com>: Managed by Tactical Strategy Group Inc. (founded by William M. Ulrich, former Y2K industry guru).

<http://www.cbdiforum.com>: See “SIG on Legacy Rejuvenation.” CBDi Forum is a spin-off from the Butler Group, a leading European analyst group.

<http://www.nas.nasa.gov/Groups/Tools/Projects/LCM>: NASA’s Legacy Code Modernization special-interest group.

Books

Java for COBOL Programmers, John C. Byrne, Charles River Media, Rockland, Md., 2000.

Migrating Legacy Systems, Michael L. Brodie and Michael Stonebraker, Morgan Kaufmann, San Mateo, Calif., 1995.

Reengineering Legacy Software Systems, Howard W. Miller, Digital Press, Boston, 1998.

Reports

In the US, the Gartner Group, IDC, the Giga Group, and the Hurwitz Group follow legacy modernization and publish research reports on the subject. In Europe, the Butler Group and Bloor Research are good sources.

These reports are available to customers of the Giga Group and the Gartner Group:

From the Giga Group: <http://www.gigaweb.com/Marketing/Default3.asp>

- “Legacy Application Mining to Improve, Renew and Reuse Legacy Assets,” P-0999-008, Stephanie Moore.
- “Post Y2K: Renewing Legacy Applications,” P-1298-004, Liz Barnett.

From the Gartner Group: <http://gartner1.gartnerweb.com/public/static/home/home.html>

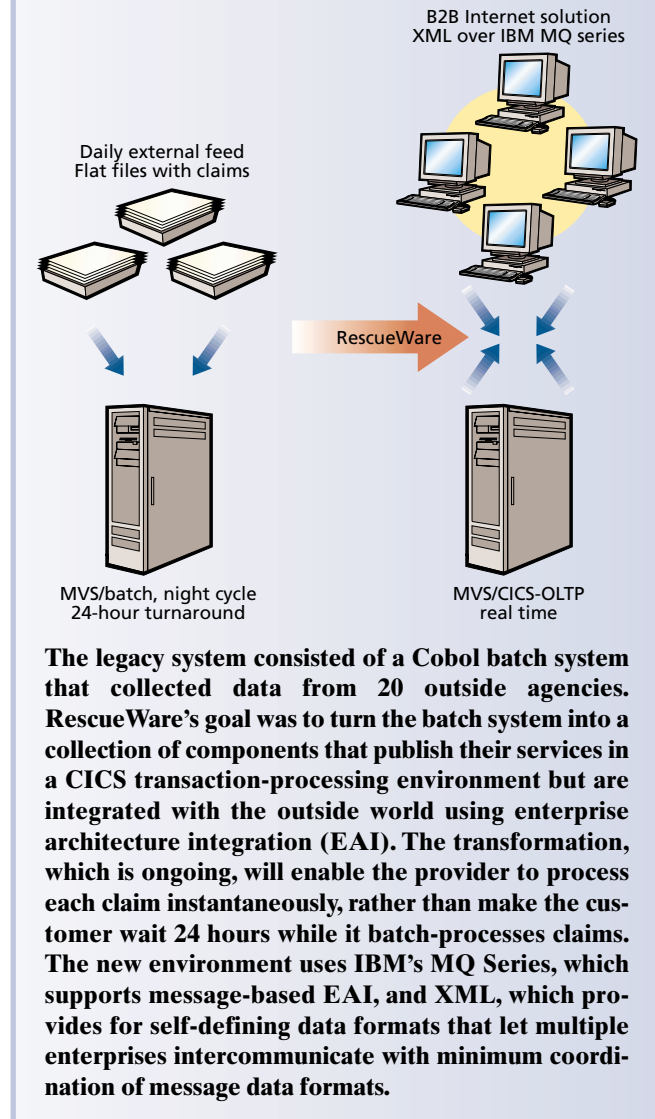
- “Legacy Understanding Is a Many Layered Thing,” D. Vecchio, SPA-10-3203.
- “Legacy Extension: Short-Term Solutions, Future Problems?” D. Vecchio, TG-09-9347.

options, but it is also the most costly and the most difficult to do successfully.

LEGACY TRANSFORMATION: A CASE STUDY

RescueWare (see Figure 2) is designed to help companies transform their legacy systems by automating much of the code analysis and transformation. Legacy transformation has three main parts: help analysts understand what parts of the legacy system are worth transitioning to the e-business world, extract (or mine) the business

Figure 3. How a healthcare provider used RescueWare to transform its claims processing system to e-components.



value from those parts, and move the code into the new world.

In one ongoing RescueWare project, a major US healthcare provider has a strategic Cobol batch system (2 million lines of code) that collects data in flat files from 20 partner agencies. The data feeds from these agencies arrive once a day and contain all patient claims collected during the past 24 hours. The system merges individual flat files into a single transaction file and processes it collectively during a nightly batch cycle. This system gets the job done—it processes individual transactions—but the 24-hour delay it imposes was hindering the company's ability to serve its customers in real time. Moreover, processing

all transactions at the same time was causing throughput strains on an already overloaded nightly batch cycle.

As Figure 3 shows, the provider wanted to transform the unwieldy batch system into a collection of flexible e-components that publish their services in an IBM CICS (Customer Information Control System) transaction-processing environment but are integrated with the outside world using enterprise architecture integration. It could then process individual claims almost instantaneously and would be able to more evenly distribute its workload because claims would arrive throughout the day.

Understanding the system

The first step is to deeply understand the existing batch process as well as individual program structure and algorithms. Understanding a legacy system's structure and operations is critical to transforming it intelligently to the new architecture. This step also helps a company understand its business processes more comprehensively. On the downside, it is time-consuming and costly. A typical legacy system has a million or more lines of code distributed over hundreds, sometimes thousands, of individual programs. Many of these depend on each other via program-to-program calls.

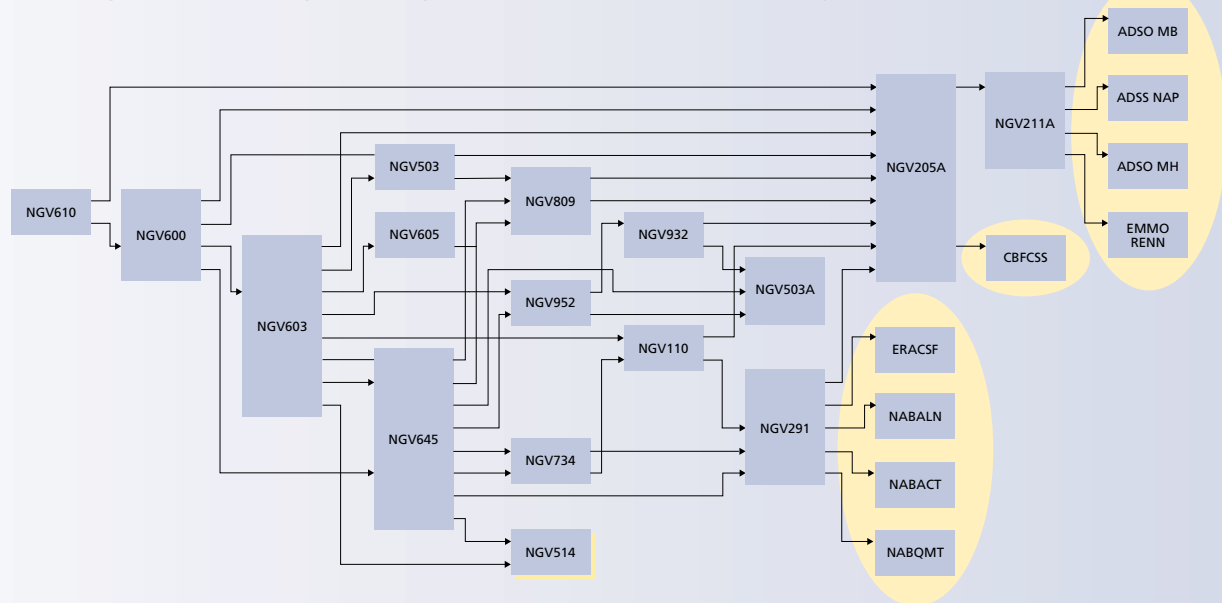
In this phase, RescueWare parses the legacy code and produces reports (inventory, copybook dependency, unresolved external references) and diagrams (program call map, CICS flow map, dataflow diagram) that help a company understand what each program does and why. These products are the foundation for the next step in the transformation process. Process flow diagrams, for example, display program-to-program call/link relationships, which helps IT analysts identify program dependencies, as well as any programs that must interface with the outside world.

Extracting business value

During this phase, RescueWare uses knowledge mining to extract reusable business or e-components from the legacy system. The idea is that business rules—core policies and algorithms that capture the nature of an enterprise's business model—are buried in the legacy system. For the healthcare provider, these business rules include new claim validation, referral requirements, and special-procedure approval. In other applications, business rules might be interest rate calculation, risk assessment policy, or student graduation requirements. Because these rules are fundamental to the business model, they are technology independent.

Identifying e-components. A legacy application system's objects function together in complex ways. To transform a large, complex system, the objects must first be segregated into clearly identified collections, each of which aligns with a business process function or business area.

Figure 4. Using a program call map to identify e-components.



RescueWare generates a series of call maps that are based on its analysis of the legacy code's structure. The IT analyst feeds RescueWare a list of business areas derived from end-user interviews and other sources. RescueWare then attempts to slot each program into a single business area, producing a map of programs (boxes) and calls (lines) for each area. Each call map shows program clusters (tightly connected programs) as well as boundaries and interfaces between clusters. Comparing the maps to the preliminary list of business areas gives analysts enough information to roughly identify e-components. They also use the call maps to identify e-component interdependencies. In the figure, the gold regions represent programs that interface with the outside world (programs in other business areas). This information is important because business areas should have minimum coupling to work well in a component architecture.

IT analysts begin the segregation process by providing RescueWare with a list of business areas derived from end-user interviews and other sources. RescueWare then attempts to slot each program to one of these areas, generating a program call map—a diagram of legacy programs and their process flow that fit a particular business area. Figure 4 shows a sample call map. Analysts refine the list of business areas by reviewing these maps to identify program clusters—programs that are tightly interconnected. This analytical exercise provides enough information for the analysts to roughly identify the e-components. As Figure 4 shows, the diagram uses shading to highlight e-component boundaries and the interfaces to other e-components. Reviewing inventory and complexity reports helps analysts to fine-tune these boundaries.

In the case study, the healthcare provider's analysts were able to identify 12 distinct business areas, which eventually became e-components, including new claims processing, patient referrals, and special procedures.

If a program does not cleanly fit into a business area, the analyst can restructure it for a better fit or replicate it under a different name into multiple business areas. In the

provider example, approximately 10 percent of all programs had to be restructured to provide for a better fit into a specific business area.

A legacy system also generally has components that perform common system utility functions such as error reporting, transaction logging, and data calculation. These usually work at a lower level of abstraction than e-components and so must be factored out. As Figure 3 shows, part of RescueWare's knowledge mining phase is devoted to creating a systemwide reusable utility library. This eliminates processing redundancy and ensures that the system behaves consistently.

Business rule extraction. E-components must have clearly defined APIs for a company to integrate loosely coupled systems. RescueWare uses four business rule extraction techniques to help identify and establish logical e-component APIs.

- Computation-based extraction produces a functional slice of a program that is based on the execution path and the data definitions required to calculate the value for a given variable at a specific point in the program.

Hunting for Business Rules

A business rule is a self-contained section of legacy code that focuses on computing a specific variable or on making a specific binary decision. Once the analyst identifies a business rule through end-user interviews, peer discussions, and good old-fashioned system analysis, the job becomes how to locate the rule in the legacy source code and extract it as a self-contained routine.

If the legacy code is a well-structured program, extraction can be as simple as cut and paste. However, most legacy code programs don't fit this description. Searching for a business rule in even a 4,000-line program, which represents 74 pages of code, is like hunting for the proverbial needle in a haystack. The rule can be scattered throughout the program, most likely intertwined with other pieces of logic. Of the entire program, eight pages might be affected, and two or three lines in those eight pages might be code related to that rule.

You must also consider both the control and the variable computation flows. You must trace every variable encountered in the selected process flow to the origins of its computation. A business rule can have more than one variable controlling its process flow. You need to trace all these variables to the beginning of the program to compute their proper values at the decision points.

Finally, to create a self-contained routine, you must identify and extract all pertinent variable declarations. Although this is a mechanical task, a business rule can involve a large number of variables, making it a labor-intensive and error-prone process.



In the case study, this type of extraction helped separate individual transaction processes that already satisfy immediate business needs, such as claim limits approval and coinsurance validation, from obsolete hardwired batch-processing flows.

- Domain-based extraction produces a functional slice of the program that is based on the fixed value of an input variable. This type of extraction separates individual transaction types into encapsulated e-components with clearly defined APIs. RescueWare uses this type of extraction to partition application logic according to medical-plan type and special-procedure type.
- Structure-based extraction lets the system partition a program into a series of independent business rules on the basis of its physical structure rather than on its underlying business processing. If the legacy system already performs online processing, RescueWare separates the business logic from the user interface and creates additional e-components.
- Global extraction lets analysts apply all the previous techniques across multiple programs, in effect supporting systemwide business rule extraction.

These extraction techniques use advanced mathematical algorithms that automate business rule identification and extraction. All of these techniques produce a syntactically correct stand-alone program that captures a legacy system's business behavior. The "Hunting for Business Rules" sidebar explains why the extraction process is so difficult in legacy systems.

Moving the code

Once RescueWare forms the e-components, it moves them to the new environment, where it publishes them as stand-alone business services.

Another option is to have them remain on the same platform. Companies can deploy the e-components directly on a mainframe as Cobol, PL/I, or Adabas Natural programs, or they can redeploy them into Java, Visual Basic, or C++ objects on distributed platforms. In the case study, e-components will move into a CICS OLTP environment as stand-alone services.

Companies can then use standard application server technologies to package the resulting e-components for distributed access. The technologies can be a combination of callable APIs, wrappers, and proxies—for example, COM/DCOM (Component Object Model/Distributed Component Object Model), CORBA (Common Object Request Broker Architecture) objects, Enterprise JavaBeans, or XML-publishable objects. Companies can also publish the e-components in HTML and component glue languages like JSP and ASP. Finally, RescueWare enables database access via standard mechanisms including ADO and JDBC (Java Database Connectivity).

PROJECT POSSIBILITIES

Business rule extraction demands involving an IT analyst or developer to scrutinize the legacy system up front and to make design decisions when implementing the modern solution. This may sound like a lot of work, but the resulting improvements in productivity, delivery schedules, and accuracy are well worth it.

Labor logistics

In the healthcare provider project, the client opted to divide the transformation into four 500,000-LOC projects, each six months long and done sequentially. Although the subprojects could have proceeded in parallel, the client decided to deal with the technology adoption risk incrementally.

Other clients have chosen to do transformation in parallel subprojects. One client—transforming 1 million lines

of Cobol code distributed over 1,000 programs—has a dedicated three-person team per business area. The teams proceed with minimal interaction (because each business area, or e-component, is self-contained), and the client projects that each team will complete the systemwide analysis and component breakdown in four months. It should take another seven months to extract the business rules, deploy the system to the new technology, and roll it out to the customers.

This shows that transformation is only part of legacy system modernization. I estimate that a typical project is 60 percent development and 40 percent is testing, implementation of the new technology, and rollout to the customers. RescueWare cuts that 60 percent to about 30 percent. In the case study, I estimate that transformation time will go from six months to two.

Language issues

If you don't want to move into a new language but want to find ways of extracting value from existing applications, you can focus on restructuring your existing legacy system. For example, you can mine a 1,000-line Cobol program into three smaller Cobol programs. Each program would perform an individual business function, such as Add Customer, Edit Customer, and Delete Customer. Rescue-

Ware would generate the necessary parameters and calling infrastructure for these components to communicate.

Platform issues

Believe it or not, you can still keep your mainframe. Technology is additive, and mainframes and their applications will exist for some time to come. The question is not whether the hardware is obsolete but rather how you can use the right technology for the right problem to meet all the new stresses that competition and e-business represent.

RescueWare is not a pushbutton product. You don't press a button and convert an entire legacy system to another technology. But it can help you better understand your existing online and batch systems. It can tell you which parts of the system you should transform and which parts you should throw away. ■

Len Erlikh is a cofounder and chief technical officer at Relativity Technologies. Contact him at lerlikh@relativity.com.