Cloud-Enabling Legacy and Mainframe Applications

The Essentials Series

sponsored by Attachmate

Don Jones
Introduction to Realtime Publishers

by Don Jones, Series Editor

For several years now, Realtime has produced dozens and dozens of high-quality books that just happen to be delivered in electronic format—at no cost to you, the reader. We’ve made this unique publishing model work through the generous support and cooperation of our sponsors, who agree to bear each book’s production expenses for the benefit of our readers.

Although we’ve always offered our publications to you for free, don’t think for a moment that quality is anything less than our top priority. My job is to make sure that our books are as good as—and in most cases better than—any printed book that would cost you $40 or more. Our electronic publishing model offers several advantages over printed books: You receive chapters literally as fast as our authors produce them (hence the “realtime” aspect of our model), and we can update chapters to reflect the latest changes in technology.

I want to point out that our books are by no means paid advertisements or white papers. We’re an independent publishing company, and an important aspect of my job is to make sure that our authors are free to voice their expertise and opinions without reservation or restriction. We maintain complete editorial control of our publications, and I’m proud that we’ve produced so many quality books over the past years.

I want to extend an invitation to visit us at http://nexus.realtimepublishers.com, especially if you’ve received this publication from a friend or colleague. We have a wide variety of additional books on a range of topics, and you’re sure to find something that’s of interest to you—and it won’t cost you a thing. We hope you’ll continue to come to Realtime for your educational needs far into the future.

Until then, enjoy.

Don Jones
Introduction to Realtime Publishers

Article 1: Why Your Legacy Application Data Needs to Be Connected to the Cloud

   The Cloud Is Real. The Cloud Is Relevant

   How Your Legacy Apps Are Holding Your Data Hostage

   Connecting Your Legacy Data to the Cloud

      Enable Cloud Application Integration

      Enable External Partnerships

      Enable Internal Users and Integration

   How Your Legacy Apps Can Become the Back-End to Your Own Cloud

Article 2: Connecting Legacy Application Data to the Cloud: Common Mistakes and Misconceptions

   Legacy Apps: Expensive, Mission-Critical, and Change-Averse

   The Top-Down Approach: One Solution to Rule Them All

   The More Moderate Approach: Modifying Your Mainframe Apps

   A Popular Approach: Adding Web Services Capabilities to Your Mainframe

   A Wish List for Cloud-Connected Legacy Data

Article 3: Connect Your Legacy Data to the Cloud—With Less Risk, Time, and Cost

   The Middleman Approach

   Less Risk, Less Time, Less Cost

   Flexibility in Exposing Data

   A Peek Under the Hood at Cloud-Connected Data

   Your Mainframe: A Back-End for Your Cloud
Copyright Statement

© 2011 Realtime Publishers. All rights reserved. This site contains materials that have been created, developed, or commissioned by, and published with the permission of, Realtime Publishers (the “Materials”) and this site and any such Materials are protected by international copyright and trademark laws.

THE MATERIALS ARE PROVIDED “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. The Materials are subject to change without notice and do not represent a commitment on the part of Realtime Publishers its web site sponsors. In no event shall Realtime Publishers or its web site sponsors be held liable for technical or editorial errors or omissions contained in the Materials, including without limitation, for any direct, indirect, incidental, special, exemplary or consequential damages whatsoever resulting from the use of any information contained in the Materials.

The Materials (including but not limited to the text, images, audio, and/or video) may not be copied, reproduced, republished, uploaded, posted, transmitted, or distributed in any way, in whole or in part, except that one copy may be downloaded for your personal, non-commercial use on a single computer. In connection with such use, you may not modify or obscure any copyright or other proprietary notice.

The Materials may contain trademarks, services marks and logos that are the property of third parties. You are not permitted to use these trademarks, services marks or logos without prior written consent of such third parties.

Realtime Publishers and the Realtime Publishers logo are registered in the US Patent & Trademark Office. All other product or service names are the property of their respective owners.

If you have any questions about these terms, or if you would like information about licensing materials from Realtime Publishers, please contact us via e-mail at info@realtimepublishers.com.
Article 1: Why Your Legacy Application Data Needs to Be Connected to the Cloud

An airline wants to cooperate with a credit card company’s rewards program, enabling credit card rewards points to be transferred to the airline’s frequent flier program. That program is managed by a mainframe-based application.

A manufacturing firm wants to outsource their customer relationship management (CRM) software to a software-as-a-service (SaaS) provider but needs to provide real-time data from their mainframe-based inventory and product control systems.

A retailer wants to implement a new distribution center product-shipment system but needs to interface that system to their mainframe-based inventory control application.

It’s happening more and more: Companies in a variety of industries want to do more with their business. They don’t want their mainframe system to hold them back, but they’re not interested in replacing the mainframe, which in most cases does a great job of doing what it was designed to do. They just want to use that data in more places, giving themselves more business agility. In more and more of these cases, they want that mainframe data to be used in “the cloud.”

It’s easy to think of “the cloud” as just another overused IT marketing term. We’re bombarded with cloud-based promises almost every day. The fact, however, is that the cloud is real. Every day, thousands of companies process transactions in cloud-based services. More and more companies are outsourcing selected applications and functionality to the cloud—CRM, business intelligence, email, collaboration, and more. The cloud is giving businesses new options and opportunities to reduce costs, reduce business distractions, and increase their flexibility and agility.

The cloud and its supporting technologies offer an unprecedented level of integration between business partners. Information can be more easily exchanged, validated, and stored. Companies no longer need to rely on slow, manual, batch data transfers conducted via physical media or scheduled FTP transfers. Instead, companies can interact in real-time, exchanging exactly the data they want in a fast, secure fashion. An airline can show you how many frequent flier miles are available to be transferred from your credit card’s rewards account, and an online dinner-reservations service can check on the availability of a table at that hot new restaurant in town.
Cloud technologies have enabled a whole new breed of outsourced service, whether you call it SaaS or some other term. With real-time data exchange, these services are more than just a service that you pay for and consume; they can become an integral part of your infrastructure by accessing key data from your in-house systems—blurring the line between “cloud” and “data center” and providing the right set of services, at the right time, to your business. A cloud-based business intelligence (BI) platform can access sales data from your internal systems, helping you to answer business questions and make smarter decisions going forward.

Cloud technologies can even make it easier to share data between your internal systems by relying on well-understood, industry-standard protocols for communications and data exchange. Your order-tracking system can send shipping information to a shipment-processing system as well as to a credit card merchant system, speeding up and automating order processing and fulfillment. You gain greater abilities to use off-the-shelf systems and components as part of an integrated system—without having to custom-integrate them.

The big problem arises when your legacy data—the data living on your mainframe—becomes involved. Mainframes have typically been big, monolithic, all-inclusive environments. They don’t always work and play well with other systems, and when they do integrate, they like to do so on their own terms. Most mainframe applications were designed well before the advent of the Internet, let alone the concepts that make up the modern cloud. Those legacy applications can hold you back.

How Your Legacy Apps Are Holding Your Data Hostage

The ultimate problem is that, given a desire to integrate that mainframe-based data with other systems, you’re left with few choices, and they all seem like bad ones. Mainframe applications aren’t generally designed to expose application programming interfaces (APIs) so that other applications can interact with their data. Mainframe apps don’t typically provide direct database access, either. Instead, they’re based on ages-old, manual entry screens, typically conducted through terminal emulation software or other user interface layers.

Sure, you could change all that, but reprogramming a mainframe application is a big move. It’s risky because you’re tampering with an application that forms a core part of your business. It’s time-consuming because most companies typically maintain a long backlog of application changes they’d like to make—and because working slowly and carefully is the only way to help reduce the risk of messing with such a mission-critical piece of the business.

In other words, connecting your legacy data with other applications is risky, difficult, and time-consuming. Many companies prefer to simply not do it, meaning they have to content themselves with leaving their data locked up inside those mainframe applications. They have to be happy leaving their data hostage. The good news is that it doesn’t have to be that way.
Connecting Your Legacy Data to the Cloud

In the next article in this series, we'll examine some of the traditional approaches to making that connection—and look at some of the downsides of those approaches. In the final article in this series, we'll see what new techniques and technologies exist to help make that connection with less risk, less time, and less complexity. However, to make sure those new techniques and technologies can meet real-world business needs, we need to spell out some of the reasons businesses want to free their mainframe-based data and bring it to the cloud.

Enable Cloud Application Integration

One major reason companies want to unlock their legacy data is to enable integration with cloud-based applications. As already mentioned, companies are increasingly turning to cloud-based applications for select functionality. Although it’s unreasonable to expect most companies to outsource all of their software, companies are finding value in outsourcing complex, difficult-to-maintain, hard-to-deploy applications that threaten to be a business distraction otherwise. Big enterprise-class applications are a prime target: CRM, enterprise resource planning (ERP), BI, email and collaboration software, and so on.

These applications, however, don’t exist in a vacuum. In order to provide the best functionality and the most value, these cloud applications need to access certain kinds of data, usually maintained in applications that are more core to the business’ operation, which businesses quite reasonably want to keep in their own data centers. That might be a BI system that needs to access product and sales data, or it might be a CRM system that needs to access customer order information. The real trick to moving to these cloud-based applications is in doing so while still giving them the tight integration with in-house data that you’d get if implementing those applications in your own data center.

In companies that rely primarily on PC-based computing, this integration often comes in the form of direct access to databases and applications. Newer applications are often written with data-sharing APIs, and they run on standardized, easily-accessible databases such as Microsoft SQL Server, Oracle, and the like. Those are all easily-accessible using on-site agents and other techniques. However, if your data is locked up inside a mainframe application, getting at it can be much more difficult.

Enable External Partnerships

Companies are increasingly looking to exchange data with each other about their mutual customers—and not just for cross-marketing purposes, as has often been the case in the past. Instead, companies are offering real integration of their services and partnerships, presenting customers with a more seamless experience. Online financial management systems pull bank account transactions in real-time. Customer loyalty programs exchange points between customers. Customer appointments and reservations can be made through popular partner Web sites. These and other scenarios are becoming more and more common.
One approach to creating this kind of connectivity has been to implement custom systems. For example, one popular restaurant reservations Web site requires that restaurants use the site’s own on-premises dining room management software, enabling the site to remotely access table availability and create reservations. That’s suitable for some circumstances, but companies are increasingly interested in creating that kind of integration with their own systems rather than adopting new systems. Such is especially the case with companies that have put one or more mainframe applications at the core of their business.

Yet mainframe applications are, as has been outlined, difficult to integrate with. How can a remote business partner’s systems insert data into your mainframe application when the only way to enter that information is through a terminal emulator and manual entry? How can you ensure that the incoming data is valid, and that it won’t bypass the business controls and validation built-in to your mainframe application? That’s a key challenge in enabling this kind of integration, and something we’ll consider carefully in the forthcoming articles in this series.

Enable Internal Users and Integration
Another kind of integration that companies are wanting more and more is to simply integrate their own internal systems more closely. As companies acquire new, non-mainframe applications and systems, they want those systems to be able to access the core data contained in the mainframe applications that run the business. That kind of integration typically involves a per-project modification of the mainframe application—something that, as has been stated, can be expensive, time-consuming, risky, and complex.

How Your Legacy Apps Can Become the Back-End to Your Own Cloud
What’s really needed is a way to expose mainframe data in a single, consistent fashion. Ideally, that method will include standards-based communications so that a broader range of communications and data transfer is automatically available.

The idea, in fact, is to make your mainframe the back-end for your own private “cloud.” You’ll provide specific interfaces that enable access to your legacy data by external systems—whether those systems are in-house, owned by a cloud computing provider, or owned by a business partner. That’s what we’ll explore in the next two articles.
You’ve decided that you need to connect your mainframe data to the cloud. The previous article in this series outlined several reasons why you might want to do so, and now you’re ready to find out how. In this article, we’ll explore common, traditional approaches—and their downsides. We’ll also look at strategic, tactical, and technical mistakes that companies make when heading down this path so that you can avoid them in your own organization.

Legacy Apps: Expensive, Mission-Critical, and Change-Averse
First, let’s acknowledge that extending a mainframe application in any way can be a frightening proposition. These applications simply aren’t designed for extensibility. In some cases, business logic is contained entirely within text-based user interfaces, making it difficult to automate data entry and retrieval without bypassing all of that business logic.

Mainframe applications typically sit right at the heart of the business. They’re one of the most mission-critical aspects of the business’ technology infrastructure, running the applications that make the business work. Messing with that kind of critical component is extremely risky: One wrong move, and you’re literally out of business.

Let’s look at three traditional approaches that all seek to extend the mainframe application data. For each, we’ll examine the details and downsides, and see which ones meet the most important business criteria for this kind of project. Typically, businesses are looking for a solution that

- is quick and can be implemented for a specific project. Companies are rarely looking for some kind of unified, do-it-all solution, and instead tend to be driven by a specific, time-sensitive need.
- is inexpensive, at least compared with whatever alternatives are available.
- involves as little risk to the business as possible.
- is as reusable as possible so that when future projects come down the road, companies don’t have to start this entire process over again from scratch.
- won’t negatively impact mainframe performance because the mainframe occupies such a critical role within the business.

Let’s see how three traditional approaches measure up.
The Top-Down Approach: One Solution to Rule Them All

With this approach, companies seek to figure out everything they might need to integrate their legacy data into, and to build a solution—either based on the mainframe itself, or integrated from a PC-based server—that will meet all of those needs. On paper, this solution is elegant: Solve the problem once, in a consistent and well-planned way. This might involve creating an entire adjunct to the mainframe application that is capable of communicating with external applications via some data-interchange protocol. Ensure that all the necessary business logic is built-in to that connectivity application, and enable it to access every aspect of the mainframe data.

In reality, this approach is generally one of those projects that never really get off the ground. To begin, it’s a massive undertaking and will generally involve extensive discovery. You’ll need to determine the full extent of the data that will require integration, and you’ll need to consider all the potential ways in which someone might want to integrate with it from external systems. You’ll likely be duplicating a great deal of business logic, giving you another whole mainframe application that has to be maintained (which is certainly a non-trivial concern in and of itself).

It’ll be such a massive, all-hands effort that executive commitment will usually be required, making it much harder to get the necessary approvals to get the project underway. It’ll be a high-profile project, with a lot more eyes on what’s happening and a lot more red tape and process to slow things down. You’ll be messing with the mainframe itself, which is rightly a cause for concern and caution.

The worst aspect of this approach is that it’ll really hold up any specific projects that need this kind of integration. Although your programmers and architects design and build this “unified integration system,” projects that desperately need to access just a few bits of data will simply have to wait—jeopardizing business opportunities and holding back the organization as a whole.
In terms of the common-sense business criteria, how does this “top-down” approach measure up?

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Top-Down Approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time to implement for a specific project</td>
<td>Lengthy</td>
</tr>
<tr>
<td>Expense</td>
<td>High</td>
</tr>
<tr>
<td>Business risk</td>
<td>High</td>
</tr>
<tr>
<td>Reusability</td>
<td>Very good, if well-designed</td>
</tr>
<tr>
<td>Mainframe performance impact</td>
<td>Moderate to high</td>
</tr>
</tbody>
</table>

That’s not looking like an ideal approach.

**The More Moderate Approach: Modifying Your Mainframe Apps**

A more moderate approach is to simply focus on the project at hand. Rather than trying to create a one-size-fits-all integration layer to the mainframe app, you simply modify the mainframe app to accommodate the most immediate need.

This approach is certainly not without its downsides. In a best-case scenario, where you own the source code to the application and have experienced developers to do the work, you’re looking at a time-consuming project that entails an incredible amount of business risk—you are, after all, modifying an application that sits in the most mission-critical position possible. A less-than-best-case scenario will require you to go to the application vendor for modifications or to otherwise outsource the project—which will be expensive. Even if you’re using your own development resources, you’re likely looking at a long wait because most companies have a significant project list stacked up for their mainframe programmers.
If well-written, the modifications can often avoid negative performance impact because they’ll be integrated directly with a native application. However, this approach also offers a lower return on investment (ROI) because this approach is project-specific. The next project that comes along will require its own modifications. Eventually, you’ll have a heavily-modified application that’s been tweaked to meet the needs of each project that comes along. That kind of heavily-customized application carries its own significant business risks in terms of long-term support and operational costs.

How does this approach meet our business criteria?

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Top-Down Approach</th>
<th>Modify the Mainframe App</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time to implement for a specific project</td>
<td>Lengthy</td>
<td>Moderate to Lengthy</td>
</tr>
<tr>
<td>Expense</td>
<td>High</td>
<td>Moderate to High</td>
</tr>
<tr>
<td>Business risk</td>
<td>High</td>
<td>High</td>
</tr>
<tr>
<td>Reusability</td>
<td>Very good, if well-designed</td>
<td>Poor</td>
</tr>
<tr>
<td>Mainframe performance impact</td>
<td>Moderate to high</td>
<td>Low to Moderate</td>
</tr>
</tbody>
</table>

Not so well. It’s still time-consuming, incredibly risky, and even with your own developers doing the work on your own source code, it’s an expensive proposition with poor reusability.

A Popular Approach: Adding Web Services Capabilities to Your Mainframe

Another approach, one made popular in recent years by vendors who play in this space, is to add a Web Services layer to the mainframe itself. Doing so turns the mainframe into a standards-compliant Web Services platform—and that obviously has some advantages. A single implementation project can give you a highly-reusable solution for sharing mainframe data, using industry-standard protocols and techniques.
On the downside, these projects are often extremely expensive because the vendors that implement them know that you’re desperate and that you have few options. Because you’re modifying the mainframe itself, this type of project will again require executive commitment, which may or may not be forthcoming. Project implementations are often lengthy, as you’ll need to create maps and provide business logic between the Web Services layer and the underlying applications and data. In many ways, this approach shares the same disadvantages as the “top-down” approach, although this approach generally provides better cross-project reusability, so there’s a better chance for capturing a good ROI in the long run.

There’s also a real performance risk to the mainframe because you’ll be asking it to perform a kind of work that many mainframe operating systems (OSs) weren’t specifically designed for. You’ll potentially be loading a great deal of additional transactional workload to the system, so you’ll need to be very careful about estimating that traffic and its impact on the core of your business.

Overall, how does this approach stack up?

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Top-Down Approach</th>
<th>Modify the Mainframe App</th>
<th>Web Services on the Mainframe</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time to implement for a specific project</td>
<td>Lengthy</td>
<td>Moderate to Lengthy</td>
<td>Moderate to Lengthy</td>
</tr>
<tr>
<td>Expense</td>
<td>High</td>
<td>Moderate to High</td>
<td>High</td>
</tr>
<tr>
<td>Business risk</td>
<td>High</td>
<td>High</td>
<td>Moderate to High</td>
</tr>
<tr>
<td>Reusability</td>
<td>Very good, if well-designed</td>
<td>Poor</td>
<td>Can be very good</td>
</tr>
<tr>
<td>Mainframe performance impact</td>
<td>Moderate to high</td>
<td>Low to Moderate</td>
<td>Moderate to High</td>
</tr>
</tbody>
</table>

It’s probably the best approach so far, but it’s still expensive, and because it touches the mainframe directly, it’s risky in terms of business impact and in terms of performance hit.
A Wish List for Cloud-Connected Legacy Data

Looking at these three approaches, can we construct a wish list that avoids some of their common disadvantages while perhaps including some of their advantages? Of course we can. We want an approach that

- can be implemented quickly, ideally for a single project.
- can be extended at any time to support additional projects—without having to worry about what those might be up-front.
- is less expensive than modifying the mainframe application directly.
- ideally, lives off of the mainframe. This setup minimizes or eliminates the need for executive-level commitment and reduces the chance for negative business and performance impact.

As you’ll learn in the next article in this series, we can definitely meet these needs.
Article 3: Connect Your Legacy Data to the Cloud—With Less Risk, Time, and Cost

In the previous two articles in this series, we've looked at why companies might want to cloud-enable their mainframe-based data contained in legacy applications, and we've looked at some of the more common, traditional approaches to doing so—and their downsides. In this article, we'll look at newer techniques and technologies that seek to achieve the most important business goals. Specifically, we want an approach that

- can be implemented quickly, ideally for a single project.
- can be extended at any time to support additional projects—without having to worry about what those might be up-front.
- is less expensive than modifying the mainframe application directly.
- ideally, lives off of the mainframe. This approach minimizes or eliminates the need for executive-level commitment, and reduces the chance for negative business and performance impact.

**The Middleman Approach**

The right approach is to use a mainframe-connected, cloud-enabled *middleman*. In other words, an integration system that knows how to talk to your mainframe system's applications and access their data. In addition, this system can expose that data through industry standards such as Web Services, Java Beans, or Microsoft .NET Framework assemblies. In most cases, this middleman will also serve as a conductor of sorts, turning multiple mainframe back-end operations into a single cloud-exposed transaction.

Of course, this approach isn't limited to bringing your mainframe data to the cloud. It can enable many types of interactions with external business partners’ systems, expose mainframe-based data to external applications, and even help integrate systems within your own data center.

As Figure 3.1 shows, this “middleman” integration server can be utilized on a per-project basis, enabling you to quickly spin up integration for your immediate needs. In addition, this server permits longer-term return on your investment (ROI) by easily expanding to include future projects.
Best, this “middleman” approach requires no changes to your mainframe.

**Less Risk, Less Time, Less Cost**

By not modifying your mainframe or its applications in any way, this approach eliminates most of the downsides of the more traditional approaches we examined in the previous article. It’s less likely, for example, that a given project will need executive-level commitment because the mainframe isn’t impacted. You’re not putting yourself onto your mainframe programmers’ long waiting list of projects. You don’t even necessarily need to heavily involve your mainframe team because this middleman solution—as we’ll see—accesses the mainframe just like a user would.
This approach allows you to quickly focus on integration for a single project—and then integrate other projects as they come up in the future. You don’t need an all-encompassing master plan; you can focus on today’s need, but you’re not giving up extensibility, and you’re not starting from scratch with every new little integration project that comes along. Because this approach accesses the mainframe through existing interfaces, it can be implemented very quickly with little or no complex programming. From a performance perspective, it doesn’t impact the mainframe any more than an equivalent number of human beings accessing the same data through your existing interfaces.

Let’s bring back the comparison chart from the previous article, and see how this approach fits:

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Top-Down Approach</th>
<th>Modify the Mainframe App</th>
<th>Web Services on the Mainframe</th>
<th>Middleman Approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time to implement for a specific project</td>
<td>Lengthy</td>
<td>Moderate to Lengthy</td>
<td>Moderate to Lengthy</td>
<td>Short</td>
</tr>
<tr>
<td>Expense</td>
<td>High</td>
<td>Moderate to High</td>
<td>High</td>
<td>Low to moderate</td>
</tr>
<tr>
<td>Business risk</td>
<td>High</td>
<td>High</td>
<td>Moderate to High</td>
<td>Low</td>
</tr>
<tr>
<td>Reusability</td>
<td>Very good, if well-designed</td>
<td>Poor</td>
<td>Can be very good</td>
<td>Very good</td>
</tr>
<tr>
<td>Mainframe performance impact</td>
<td>Moderate to high</td>
<td>Low to Moderate</td>
<td>Moderate to High</td>
<td>Low to moderate</td>
</tr>
</tbody>
</table>

That’s a pretty good picture.

**Flexibility in Exposing Data**

A benefit of this approach is that you don’t have to expose all of your mainframe data—something few companies want to do, anyway. You just expose the pieces of information needed for a particular project. That shortens implementation times, by the way, getting you up and running even more quickly than if you were embarking on an application-wide modification to expose every possible piece of data and process.
With a solution that provides good support for industry standards—Web Services is probably the most popular, although Java Beans and .NET Framework assemblies are also good to have—you’ll have flexibility in how you expose that data, opening it up to other systems. You’ll also open that data to developers experienced with today’s modern, rapid-application development frameworks, platforms, and architectures.

A Peek Under the Hood at Cloud-Connected Data

There are a number of ways in which your mainframe data can be accessed safely and securely, without bypassing business logic. One approach is to simply map data fields to the application screens used for manual data entry. The integration solution can coordinate multiple screens of data so that, for example, what is normally a multi-screen entry operation can be “packaged” as a single Web Services transaction. As Figure 3.2 shows, the integration server is essentially receiving a Web Services transaction, then “manually” entering that data into the same entry screens a human being would use—all automatically, and all behind the scenes.

Figure 3.2: Turning a Web Services transaction into a mainframe operation.
In other cases, you may map fields to a business data layer of the application, utilizing existing business objects and leveraging their embedded business logic to automate the application—using exactly the same components that make the application work today. And you don’t have to map the entire mainframe application; you only map the bits of the application that you need to expose. That makes for faster implementation, easier maintenance, and easier long-term operational support.

Your Mainframe: A Back-End for Your Cloud

What you’ve done is given your mainframe a new lease on life. You’ve extended the ways in which you can use your legacy application data. You’ve freed that data, unlocking it from the application and making it available, through standards-based protocols, wherever and whenever you might need it. You’ve turned your mainframe into a back-end for your own data integration cloud—and you’ve done it without modifying the mainframe in any way.