RPG Display Modernization Gets Wings

Convert old 5250 screens to modern GUI formats in a few easy steps.

Written by Thomas M. Stockwell

Last year, ASNA introduced its ASNA Wings modernization tool, which quickly and efficiently converts the old 5250 green-screen legacy display files into .NET ASPX Web forms. ASNA Wings has been out for a year now, so I made use of an opportunity to take a look at the product under the guidance of Anne Ferguson and Roger Pence. Ferguson is ASNA's co-founder and president, and Pence is ASNA's U.S. Customer Service and Education Director.

First Questions: Why and What Value?

The first question I asked was about the business case for Wings. Why would a company that had invested heavily over the years in RPG now suddenly *need* to modernize its green-screens? Sure, millions of lines of code are out there pumping out 5250 data streams to dumb emulated terminals, but unless there's a solid business case for going through a lengthy conversion to a newer "modern" graphical user interface (GUI), how could an IT manager or CTO justify the effort and expense?

Their answer was exactly what I had hoped to hear.

Personnel Turnover and Efficiency

RPG is a great language for running mission-critical applications, they said. But fewer and fewer RPG programmers are being trained. So as current RPG programmers leave the organization, management needs to identify a strategy that can preserve investments in the current RPG applications, while finding a means of making them more efficient for users who have cut their teeth on PC-based "point-and-click" GUIs.

Screen-scraping programs used to offer this kind of strategy, but as users evolve their skill sets in GUIs, and as the GUIs themselves advance, there's always a nagging feeling that even those services are grossly inadequate to satisfy the user community. The problem was exacerbated when Internet browsers became the standard user interface.

Pence was adamant about this point. "ASNA Wings is *not* a screen-scraper! It's a modernization tool, supported by an IBM-supported architecture, to *convert* green-screen display formats into fully viable Microsoft .NET ASPX file formats."

Underpinnings of Success

With these points in mind, I was extremely interested in how ASNA Wings was approaching the problem and how the technical underpinnings were configured.

For instance, if a manager is going to invest in a tool for modernization, it's clear to me that it needs to be both easy to learn and quick to implement, and it must lead the organization to a strategy that is both resilient and cost-effective. At the same time, the tool needs to preserve the current infrastructure, sustaining past investments in the RPG source code. Finally, any modernization technology also needs to be *extensible* so that applications can be further enhanced within the expanding paradigm of Web-based services that are currently exploding in the world's browser interfaces. This includes, but is not limited to, XML, JavaScript, HTML5, and AJAX.

Could ASNA Wings meet all these requirements?

Pence said, "Yes! Definitely!"

Rational Use of Rational Open Access RPG Edition

ASNA Wings uses the IBM Rational Open Access RPG Edition (OAR). This is an IBM licensed program enhancement to RPG that permits an ILE RPG program to access new devices and resources that are not directly supported by RPG. OAR enables the ILE RPG programs to interact with Web browsers, mobile devices, services, cloud resources, XML files, or spreadsheets. OAR uses a well-understood I/O model within ILE RPG to bypass the green-screen display and/or printer file formats, creating a new data stream that can be intercepted by external resources. The advantage of OAR is the transparent manner by which it is implemented in ILE RPG source code.

Simple Code Addition to RPG

In ASNA Wings' case, only one new F-specification entry needs to be added, called "Handler('ASNAWINGS')." After the source is recompiled, I/O is sent directly to the ASNA Wings module, which is then responsible for managing the display format for the data stream. (BTW, Ferguson says that the next release of ASNA Wings will perform this operation automatically.) The advantages of this technical strategy are very simple:

- There is an absolute minimum amount of RPG source modification (one added F-specification).
- Program logic flow is completely preserved and unaltered.
- IBM's support profile for the ILE RPG language is sustained.

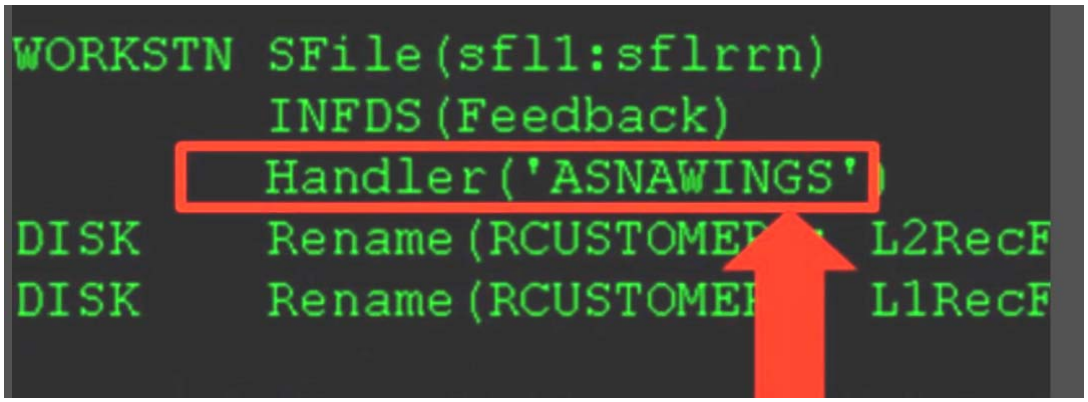

Figure 1: Redirect output through OAR to ASNA Wings.

Furthermore, with the addition of two more IBM-supported F-specs, the code can be *conditionally compiled*, spawning two program objects: one for the traditional green-screen user and one for the GUI user.

This means that, using library lists, a user in department A can use the old green-screen display interface they may prefer, while the user in department B can use the GUI through a browser, such as IE or Firefox or Opera or even the Apple Safari browser. Either way, they'll be running the exact same ILE RPG business logic. It's a neat architectural enhancement for ILE RPG, and if you want to know more about OAR, you can find it at the IBM Rational Web site.

Converting the Display Formats

Rational Open Access RPG Edition is the key pipeline that transforms the green-screen data stream into something that can be useful to other services. It bypasses the 5250 display file formats and sends a structured stream to the ASNA Wings module. But in order for the ASNA Wings module to understand how to use that data, a conversion of the original display file formats into .NET ASPX file formats is required. This is where ASNA's deep experience with .NET—through the use of its famous ASNA Monarch services—pays substantial dividends.

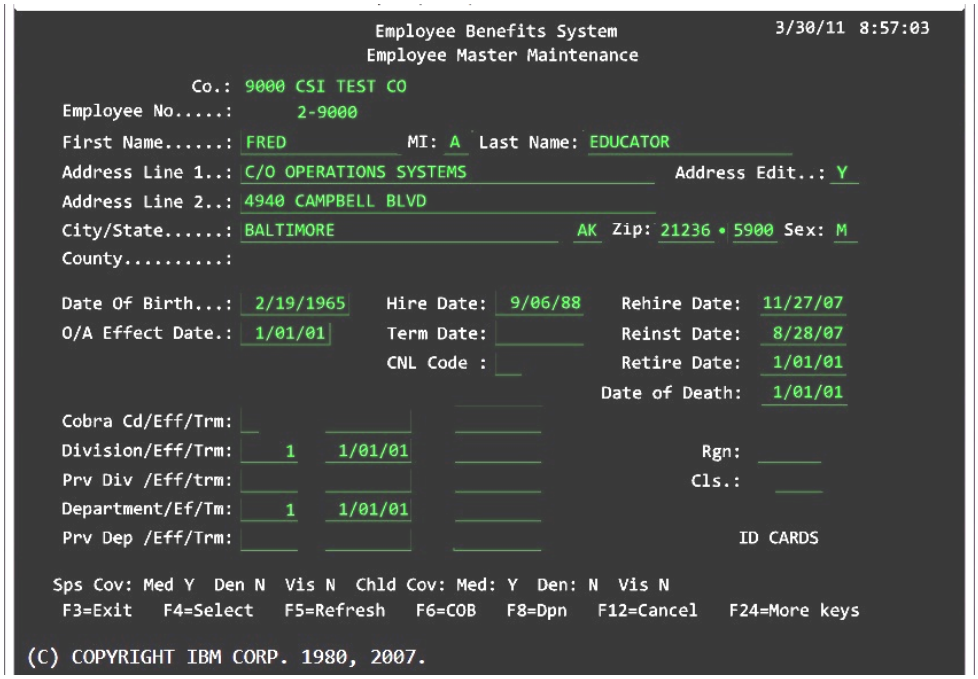

Figure 2: A typical green-screen display looks like this before conversion.

ASNA Wings provides an integrated display file source converter that imports the original DSPFM file formats from the IBM i and then outputs Microsoft .NET ASPX file formats. It provides templates and CSS formats so that the look and feel of the resulting Web pages can be later (optionally) enhanced by someone with .NET and CSS training. The conversion is a quick, one-step, one-time process.

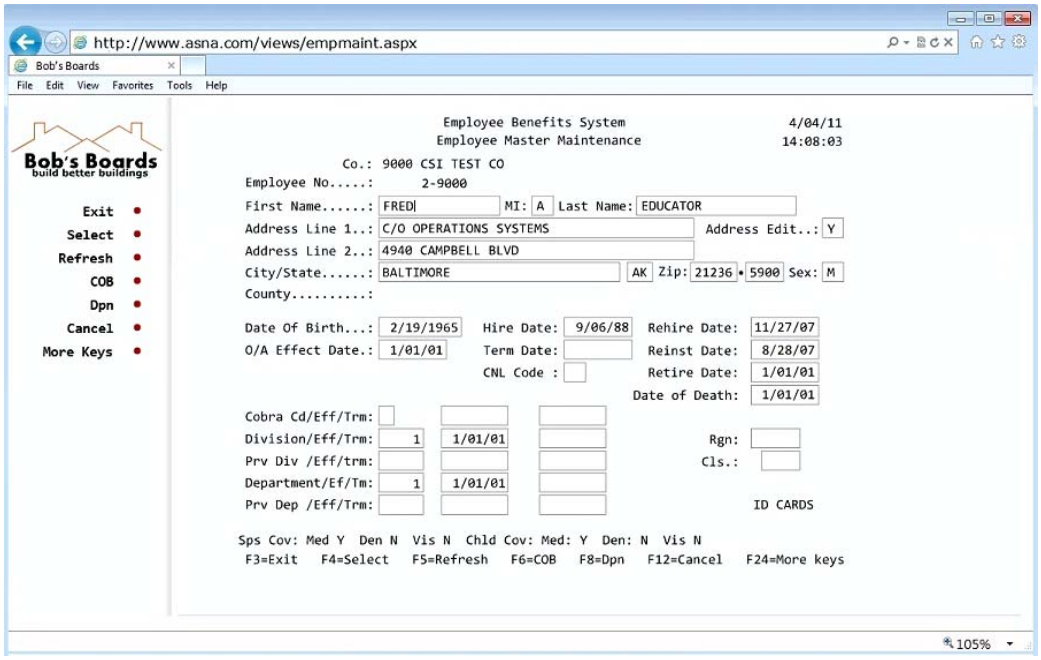

Figure 3: After conversion, the display of raw output has been transformed to a fully functional GUI.

Preserving Investment While Enhancing the GUI

The resulting ASPX formats are completely available for further enhancement through Microsoft's Visual Studio. The conversion process is branched so that a person doing the conversion (a *modernizer* in ASNA parlance) can output code for ASNA Visual RPG for .NET, Microsoft VB.NET, or Microsoft C#.

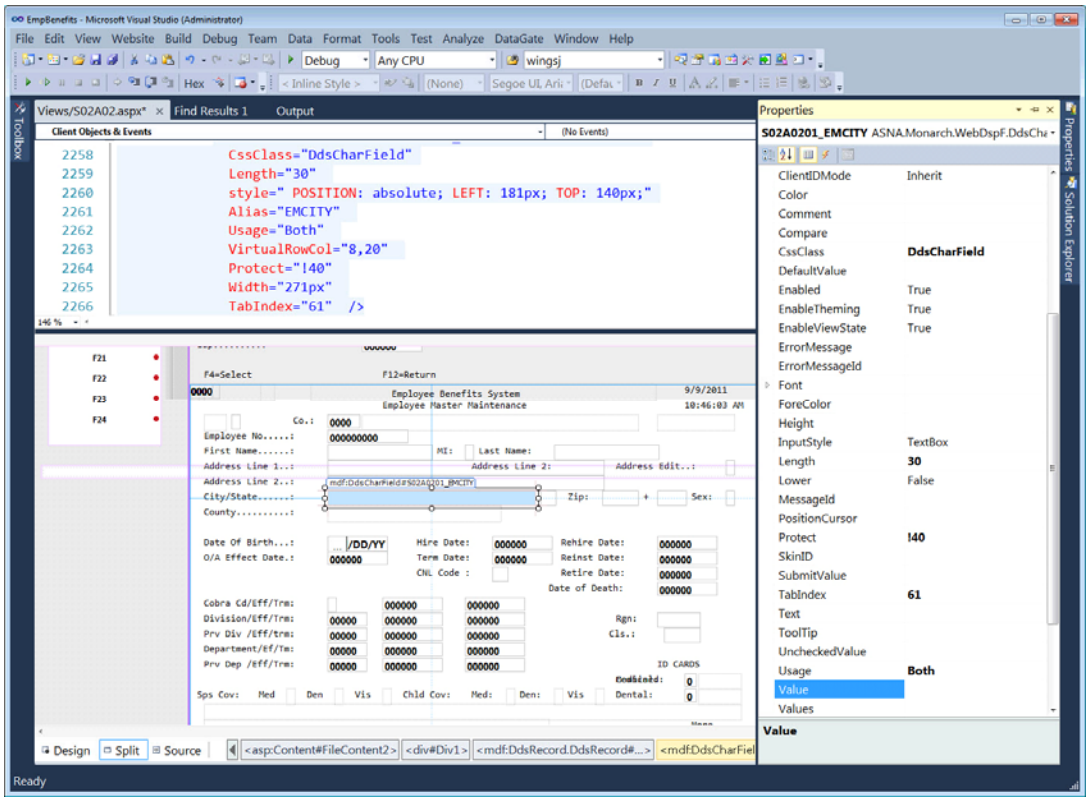

Figure 4: Enhance the converted screen with MS Visual Studio.

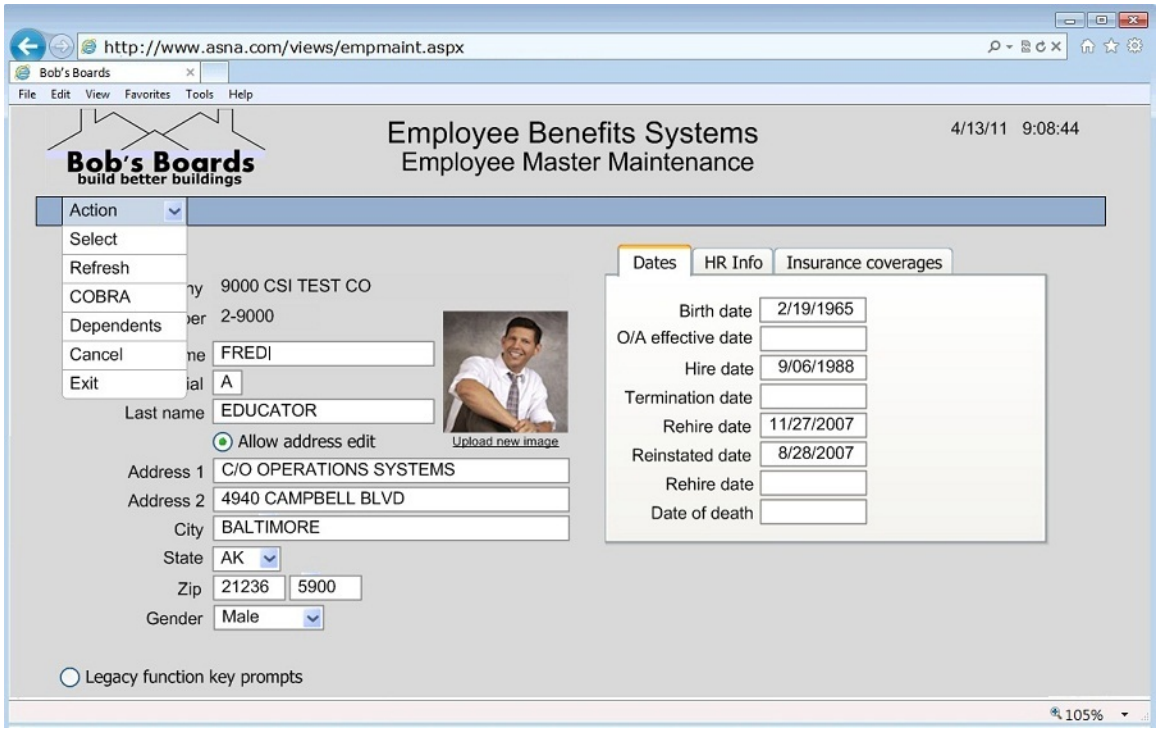

Figure 5: After customization and enhancements, the display looks like this.

This functionality offers a number of advantages for a customer who is building a strategy for application modernization:

- It permits the organization to use programmers—who may have little or no experience with ILE RPG but have skills in VB.NET or C# or ASNA Visual RPG—to maintain the presentation layer of the application without ever touching the code back on the IBM i.
- It permits the organization to enhance the presentation layer with additional services through .NET (such as AJAX, JavaScript, etc.) transparently. Again, the programmer will never touch the core business logic of the ILE RPG application. So the investment in RPG is preserved.
- It offers the organization a future transitional path for a complete migration off the IBM i through ASNA Monarch migration suite. That's because everything that ASNA Wings creates is fully reusable for ASNA Monarch.

But do you need to plan for such a complete migration? Absolutely not, said Ferguson. ASNA Wings is a complete process in and of itself. ASNA has hundreds of ASNA Wings customers who have no desire or plan for abandoning the IBM i. They just want to modernize the user's efficiency with a GUI.

Speeding Toward Modernization

So how fast and efficient is the ASNA Wings DSPFM conversion process? In a word: Quick!

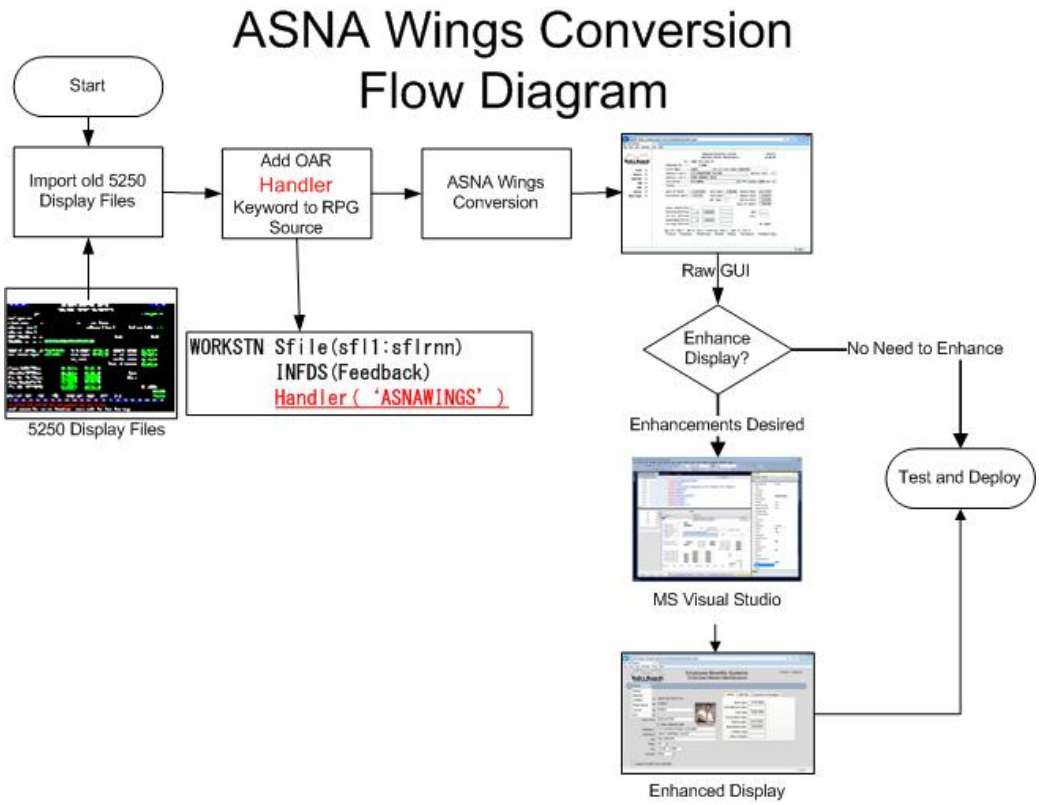

Figure 6: This is the ASNA Wings conversion flow diagram.

Using the integrated ASNA Wings Handler—an integral part of the ASNA Wings configuration—the modernizer chooses the display file source code with a single point-and-click action, and the source file is seamlessly converted into a basic ASPX file in seconds. The conversion can use pre-determined, customizable templates (called "Masters" in ASNA parlance) that can add standardized look-and-feel elements, such as color codes, logos, etc. Meanwhile, all display positioning indicators are preserved by the process, such as program-controlled cursor positioning indicators, meaning that the functionality of the original DSPFM file is transparently converted as well.

It takes a few seconds longer to compile the ASPX into something that .NET can use, but—all in all—it's a fully transparent process.

As-Needed Conversion

But what about those seldom-used RPG modules and subroutines and display formats that don't deserve the attention of an ASNA Wings modernizer? What happens when a .NET program calls one of these modules in the course of processing a user's actions? Do those programs still function?

The answer is "yes." ASNA Wings merely calls the routine or module through an MS IIS server, and it's served to the display through the ASNA emulator. The original green-screen display splashes on the user's machine, just as it would in a normal emulated session. Of course, the user is suddenly back in the world of green-screen functionality of CMD keys and cursor movements, but it's straightforward and obvious.

This means that an application modernization project can be controlled and staged in segments with little or no forethought to your modernizing personnel resources: Start with the most heavily used applications, and then work smoothly and casually toward completion through the entire application suite. You can stop worrying that you're going to break something or interrupt your users' workflow.

This also means you can segment the workload of your modernizers: those who can add the single RPG F-spec (again, this will occur automatically in a future release), those who can convert the display formats to ASPX formats, and those who can enhance the ASPX for added functionality or customization. It's the perfect project for those new programming resources who need to get a handle on the overall application base that runs your business without being bogged down by RPG-specific concerns. They can use the tools they are already familiar with, while making visible progress on the modernization tasks.

This "as-wanted/as-needed" approach to modernization also makes the most sense for companies that have limited time or resources. It speeds the modernization process, putting real results in the hands of users much more quickly than many other modernization schemes. It's quick, easy to explain to users, and understandable to everyone.

Change Management

And since ASNA Wings is fully compatible with IBM-supported change management processes, you can build out your modernization plan in a controlled and traceable manner. Best of all, it's reversible because neither your ILE RPG business logic nor the original display file source code is ever really touched.

Display Modernization Gets Wings

RPG mission-critical applications need display modernization. Our users demand it, but our budgets prohibit it. Nonetheless, each day shops around the world lose their precious RPG skills through personnel displacements.

Until recently, the technical challenges of modernizing our old 5250 displays required full conversion of the RPG source code with painstaking manipulation of the underlying display logic. IT had few choices beyond emulation, screen-scraping technologies, or full source conversion.

ASNA Wings offers a better solution. You can convert old 5250 screens to modern GUI formats in a few easy steps, with limited personnel, positioning your organization for further enhancements or full conversion to a Microsoft .NET environment, leaving the mission-critical RPG source logic untouched and undisturbed. It offers a cost-effective path that maximizes your personnel resources without the pain of other conversion technologies. It lets your modernization projects take wing. ASNA Wings!

For more information about ASNA Wings, visit www.ASNA.com.