



Delphi/400

Delphi/400 for PHP

RAD for the Web Reaches
System *i*

by
Brian W. Kelly

Introduction

Application Modernization

In most IT shops, application modernization is not tantamount to a wholesale re-start. The sheer magnitude of such a move would suffocate us all. Rather, application modernization responds to the challenge of tackling external and internal pressures and presenting them as optimally as possible. In essence, it's a much less dramatic process than might initially be thought. The integral functionality aspect, as derived from the not-quite-glamorous green screen applications, would certainly not be sacrificed. To remove them all and move to a browser-based environment with all of its aesthetic and cosmetic bells and whistles would in some ways signal demise. Note however, that this line of thinking must still take into account that there still are a select few who adopt a style-over-substance approach to modernization. They do so at their own peril.

The smart CIO and smart CEO both seek to have an acceptable "modern" system but more importantly, they want systems that support the business needs. If green screen systems began careening towards the verge of extinction, these "C-level" executives would have their work cut out for them to frantically ensure their business would be enabled to survive into the future. Fortunately, the best high level executives are not yet approaching this state of emergency, yet they are fully aware that modernization is necessary to the extent that it advances the health and wellbeing of the business. They may not be inclined to take everything that is green and endow it with every scintilla of that "fashionable" aesthetic, but they know that their IT people should be prepared for the day when they decide to digitally beautify, in essence planting the binary seeds that will one day flourish into towering crystalline trees, sweeping true-color orchids, and posh Hollywood-friendly graphical user interfaces.

So, when considering what the best application plan is for the future, it is helpful to consider the status quo at many legitimate contemporary company websites. A common theme arises from observing them. That theme is that the best application plans are those that are readily able to be implemented. Despite having big dreams and plans that may work well in theory, CEOs and CIOs are especially drawn to things that actually can be realized on a pragmatic scale. Unlike the radical innovators who find themselves so averse to archaic green screens that they want to eradicate the current structure immediately, the higher operating executives are more mindful of the survival and long term prosperity of the goal. Hence, something more practical must be implemented to achieve that objective.

For the CIO whose professional aspiration is to satisfy the CEO, his/her personal mission is to endure for at least another day to advance the cause--for both the firm and the family. Although the job is significant and integral, the CIO must work within a limited and realistic framework that precludes wanton drastic measures. An intelligent CEO and CIO are perpetually implementing one small project at a time, in seriatim, and then reevaluating their products in retrospect.

An ecosystem like the Delphi/400 rapid application development (RAD) environment is indifferent to whether its user desires a modern atmosphere or they are comfortable with the age-old green screen. Its primary concern is advancing forward. As demonstrated in the sections about today's IT landscape and the Delphi/400 development environment, there is only one effective way to complete a large multifaceted task and that is to employ piecemeal measures.

The Importance of Moderation in a Series of Dilemmas

In addition to the restraints that CIOs and CEOs have regarding the inability to redo everything drastically, there is no stable environment on which to attach an intuitive graphical user interface, even if it were the optimal approach. There is currently a prolific wealth of new Web Tools and Protocols emerging by the day. Despite their differences in brand, each proceeds from the same fundamental notions. The distinctions are hence blurred and professional knowledge and understanding of their strengths and weaknesses is at a scarcity. This consigns the C-level executives again to having no clear definitive solution.

Additionally, the current Web culture mandates that a site offer increasingly more "richness" to users. A quick comparison to the most sophisticated sites, typically those owned and operated by large influential corporations, shows that what they employ may not be economically feasible or even desirable for an SMB. The SMB cannot outrace the Road Runner and thus places its executives in a position where they must find the optimal solution with the full knowledge that it would not be viable to implement the most "perfect" or advanced system. They must work within their means. This prohibits any excess or extravagant attention to nuance because larger looming business issues inevitably preempt microscopic concerns.

The lack of unlimited capital makes an already difficult task even more demanding. This is particularly true for those companies who are acclimated to operating small IT shops while producing high quality service and information outputs. These organizations are finding the finer points of the Web to be an onerous chore to understand and manage. Unlike the early days of the interactive revolution when IBM's System/34, System/38, and System/36 were prevalent, there is no singular doorway to the Web that capitalizes on the proficiency of today's System i community.

IBM's Developer Roadmap is Helpful.

IBM's developer roadmap (<http://www-03.ibm.com/systems/i/roadmap>), as helpful as it is, actually demonstrates the difficult choices that CIOs and CEOs must make regarding Web Development. The roadmap has several points of departure likewise several destinations and thus, it is not as simplistic as merely entering one singular point of origin and destination. To use IBM's map, you must first know where you "really" are. Unfortunately, this sounds lots easier than it actually is.

To utilize IBM's Web development roadmap, you must first discern where you are in a Web continuum of choices that are sometimes elusive. In using the roadmap, your location is perhaps the simplest thing to determine.

Once you know your location by identifying your description in the roadmap, the task of determining your destination is even more a quandary to accomplish. Naturally, Delphi/400 qualifies as a destination point in IBM's Roadmap, and it is a very good place to begin your Web experience. But there are enough other destinations, with scant guidance, to help make your map reading effort fruitful. The Roadmap demonstrates nearly every possible choice that IBM approves. But which choice is the best choice for your shop? That question leads to even more questions.

The permutation of possibilities is seemingly infinite because it actually is infinite. To further compound the issue, a new product or Web protocol du jour is in one of various forms of industry evaluation/acceptance at all times. The list of possible tools that are not mentioned in IBM's roadmap is also very sizeable. Consider that IBM lacks room in its roadmap for other viable technologies that are used by System i shops. For instance, there is no room in the roadmap for CGI, the original path to the Web that is still used on many Web sites. Likewise, IBM's own

CGIDEV2, a derivative of CGI with better ease-of-use characteristics, is nowhere to be found. Additionally, PHP, which IBM is clearly favoring today, is nowhere to be found in its Developer's Roadmap. One would conclude that IBM has concluded that to have a map that is reasonably understandable, not all destinations can fit in the space allocated.

Yet, unfortunately for many, such a map does not lead to one's destination. Consider the recent symbiotic relationship of IBM with ZEND, the inventors of PHP, and the ZEND PHP offerings. Though mentioned in IBM's introductory material, PHP (PHP Hypertext Preprocessor), which exists within or functions as the principal engine for an estimated 33% of worldwide websites, is not a destination of the IBM System i Developers Roadmap. Indeed, this is not to say that PHP has been excluded on account of some sort of inherent inferiority or irrelevance. PHP is in fact exceptionally popular with newer developers and college students for many reasons, not the least of which is that it is free and the coding is similar to the C/C++ coding found in many computer science curricula. The point is that the CEO / CIO tool decision-tree presents a staggering number of discrete choices. It's a daunting task for any one person overwhelmed with so many choices to identify the one objectively correct decision.

It would be fair to say there is simply too much information and possibilities to make a fully informed Web development decision. Unless you pay for your own consultant, there is no proverbial babysitter to take you by the hand. However, you may find success by trying to find a fully independent body suggesting that you adopt a particular approach, a particular vendor, and a particular set of tools. Nobody can provide this advice at an independent level. First of all such ombudsman global bodies do not exist and if they did, for the most part, they would not know all the tools themselves. Moreover, without a real effort, they would not know what was best for your specific needs.

Ironically, in the System i Web race, IBM is somewhat of an independent entity, though IBM's own solutions do come first in its roadmap, even though they are out of place in an alphabetical scheme. All other solutions (and admittedly, they are not all there) are listed in alphabetical sequence by vendor. And, yes, in multiple destination categories in IBM's System i Developer Roadmap, you will find Delphi/400 along with other solutions provided by SystemObjects.

Internal Bias is Another Issue to Consider

The objectives of any CIO or CEO determining the right Web methodology are further complicated by internal biases. In many System i SMB shops for instance, a secondary IT department emerged over the years. This evolved primarily to address the desktop PC explosion and later the servers that came in to handle mail, static Web pages, and other one-of-a-kind applications. Therefore, a large number of System i shops that develop, maintain and run the core business activities also have server farms. The server farms address aspects of the business that long ago, the IBM AS/400 IT shop wanted nothing to do with. Though more and more companies have reigned in the two shops and combined them with common leadership, the cultures are very different and between the two it is often difficult to come up with a clear mutual agreement to set a direction that can actually work for the organization. Bias insinuates itself into the process from the outset.

One group knows System i, i5/OS, RPG programming and the DB2 database (mostly DDS & not SQL), and of course interactive 24 X 80 column designed text-only applications that run the business (many home written), etc. The other group knows Windows/Linux/Unix, Visual BASIC or a "C" variety language, SQL Databases (no DDS) - MySQL, Microsoft SQL etc, and of course the new darling, which, by the way, both Delphi platforms support, Microsoft Dot NET. Add to that GUI, & limited Web based applications that touch on more peripheral needs of the business and you have the right picture. We would not be far off the mark to characterize this as a battle of form (Windows & GUI) vs. function (air-tight proven core business applications).

The dilemma of course today is that Web sites need not only great form and navigability, they need to deal with real business processes at every level. When you contemplate the copious array of Internet tools available in light of all of their technical differences and perspectives, it becomes clear that we have a conundrum on our hands. It is difficult to locate someone from either side or even a consultant who understands this reasonably comprehensive starter set of the acronyms representing the tools of the trade for Web development:

.NET, ADO, AJAX, Apache, Applet, AppML, ASP, Blackfish SQL, C, C++, C#, ckdcc, CLX, CORBA, CSS, C# DCOM, DB2, Delphi, DHTML, DJANGO, DOM, DOT NET, Dreamweaver, DTD, Drupal, E4X, ECO, Eclipse, EJB, Flash, FPC, FrontPage, HATS, Hibernate, HTTP, HTML, IAWEB, IIS, J++, J2EE, J2SE, Java, JAVAB, JavaBeans, JavaScript, JBoss, JDBC, JMS, JOnAS, Joomla, JServ, JSF, JSP, LAMP, Mambo, MDD, MySQL, Object Pascal, ODBC, OMG, PLONE, Portlet, PERL, Python, RAD, Rails, RDF, Roxen, RSS, RUBY, Schema, Scoop, Servlet, SMIL, SOA, SOAP, Spring, Struts, SVG, Tomcat, Typo3, UML, VBScript, VCL, WAMP, WAP, WDSC, Web 2.0, WebFace, Web-Service, WebSphere, WMLScript, WSDL, XAMPP, XForms, XHTML, XLink, XML, XPath, XPointer, XQuery, XSL, XML-DOM, XSL-FO, XSLT, Zend_Studio, ZOPE, etc.

Several of these technologies are required for each implementation, and the list is constantly expanding. Most of these must be used in conjunction with others. The approach for existing core applications surely will be different than the approach for the new edge applications which you would like to build to extend the core.

Inevitably, the Internet, the Web, and the infamous Web browser have changed the very shape of what it means to create comprehensive business applications with centralized data repositories. Just like the brick and mortar counterparts, for most users, the Web is everything to everybody: a store, a bank, a book club, and a social plane. For more and more companies and many more to come, the Web has become their primary means of interacting with their full function business applications.

Both types of users find the experience incomplete in one way or another and this precipitates the reasons why companies continually change the look and feel as well as the content and function of their Web sites. It's also why companies who just aren't there yet, get a sense that their very businesses won't continue existing if they do not fix this problem and fix it fast. Whereas brick and mortar firms benefit from their proximity to their customers and rarely have to change the shape of their buildings to compete, Internet firms need a steady stream of architects, designers, re-designers, carpenters, electricians, masons, and of course plumbers. On the Web however, their dress is not blue collar and their functions are quite cerebral. Their tools are often very complex. Which company, organization, or institution can afford not to be an Internet firm?

And Yet the Need for Application Modernization Is Clear

Application modernization is a necessary transformation for any company hoping to assure a competitive edge in its respective market through the long-term vitality of its information systems. The notion predicates on a willingness to invest to gain the ability to adapt the company's core applications to new technologies, and open them up to new market standards, new opportunities, and ultimately new successes.

This is nothing new. Behind the scenes, companies have always had to advance their applications to stay ahead of the opposition, without getting rid of their existing system (in most cases). The system already in place is the product of many evolutions and perhaps even a few minor revolutions in order to enforce the company's business rules. Though some may smear this ability to handle the company's integral core applications as "legacy," companies and organizations would quickly stop functioning if it were not for their in-place IT systems.

Even without the Web, no simplistic vanilla package can replace the years of knowledge wrapped inside those internal systems. Moving to a browser-based system just because it is browser-based is a big net loss. Rewriting an entire system is always far too expensive and time consuming. Yet the status quo cannot be maintained forever.

The System i Partnership -- CodeGear and SystemObjects

In the second year of the IBM PC revolution, already a quarter-century ago, Philippe Kahn brought the Borland Company to life with great turbo, Turbo Pascal, his first product for the new IBM PC. Over time, Borland developed and or acquired a ton of other pioneering products. These included Quattro Pro, SideKick, dBASE (Ashton-Tate), and Paradox (Ansa Software). I worked with Borland products for years after my IBM career and I found them always to be top flight, high quality, and very competitive. In fact, Borland was so much on the mark that in the 1980's, Bill Gates commented that it was, of all competitors, the lone one that worried him most.

Having begun with a program development language, Pascal, a language with almost perfect syntax, Borland leveraged its AD talents to become successful despite Microsoft's quest for domination. Borland, not Microsoft, today is well known for its Integrated Development Environment (IDE) business that consists of software development tools, including what was the award-winning Borland Developer Studio (Delphi®, C++Builder®, and C#Builder®) and JBuilder® product lines). CodeGear, the Borland Company that delivers these products today has an aggressive program for not only staying on top of its game but also for moving innovation in AD to unprecedented heights.

Since 1997, Borland has partnered with SystemObjects, a worldwide IBM business partner headquartered in Paris France to provide the middleware for its products to engage with the IBM System i and its predecessors. As you will see in this white paper, this is very good news for System i developers as the newest Delphi offerings are as good as it gets in application development, and SystemObjects is as good as it gets in ease-of use AD for System i -- and this sentence rightfully ends with a period.

Thanks to this partnership, System i developers now have the same powerful tool set available for Web development and deployment as the builders of the most sophisticated Internet packages in the world have had for years. Quite frankly, the package is built for ISVs, system integrators, VARs and those who demand excellence. I would like to add however that it is also the dream package for small to medium enterprises. Unlike companies that make the user learn a new language for each enhancement, the newly announced Delphi for PHP facility has a very similar interface to the Delphi for Win 32 product set, yet it produces applications for all PHP-enabled operating systems and with the System Objects partnership, this includes i5/OS. Change is only good if it is good.

Let's talk about Delphi for Win32 for just a bit. Formally known as Delphi for Win32, with the 64-bit version and Unicode support just around the corner, this powerful IDE with its Visual Component Library (VCL) enables customers to develop ultra high-performance native Microsoft Windows applications on either Microsoft XP or Vista that not only support both platforms, but are enhanced on Vista. The support for both Microsoft XP and Vista helps developers get a jump-start in leveraging Microsoft's new visual operating system while continuing to support their existing Windows users. But that is only the half of it, and from my perspective, the smaller half.

The big news is that with the same time-proven graphical / visual interface, the all-new VCL for the Web allows developers to rapidly and visually build dynamic data-driven corporate Web applications with rich AJAX user interfaces. By the time we finish this paper, the importance of this statement will be well understood.

While you are learning about all of these wonderful capabilities, it would help to keep in mind that the simplistic client server development platforms of ten years ago that unfortunately drove many companies into bankruptcy, are now viable for the Web. There has been a lot of maturity required of the industry to enable this as well as a lot of browser enhancements. You'll be pleased to know that CodeGear is right on top of this paradigm and in fact, the company is pushing the paradigm that, despite what Larry Ellison declared in 1997 ("Client Server is Dead"), the good notions, and only the

good notions of client server are alive and well, and now they work on the Web. More importantly, the Web versions are better than anyone ever imagined for the single station or multi-station server-bound PC. Again, you and I can thank CodeGear for the evolution, unless of course you haven't yet tuned in. In that case, you may indeed call this a ***revolution***...as it surely is.

Borland / CodeGear is the industry leader in "RAD" tools for Rapid Application Development. The company is by far the best in AD and the partnership with the bright people at SystemObjects makes this phenomenon in application development your best choice for an IDE for the System i. Borland has been at this since it announced its 16-bit Windows 3.1 in 1995. Always ambitious and into continuous improvement in a big way, the company released Delphi 2, just a year later. This supported Windows 95 and the 32-bit Windows environments. Check the news releases in early 2008 and you will see that CodeGear has a lot more ease-of-AD on its way. Leaders lead! With the CodeGear and SystemObjects partnership, the System i community is again enabled for the leading edge.

Delphi/400 Is No Reason To Go Anyplace Else.

In this climate of change, CEOs and CIOs need to know how to work with the new technologies while staying true to the spirit of existing "legacy" applications. Nothing in life worth having is easy. This is where SystemObjects and the Delphi/400 development environment come in. System Objects has helped shed an incredible amount of light in finding the best path for solving the manifold problems facing executive officers in moving into today's contemporary technological standards.

Modernizing your newest System i applications means making technical choices that will still be around years down the road. Whatever strategy you need to implement, your tool choice needs to deliver your vibrant new applications quickly and bug free. They must provide the true look and feel of a modern application while using as much existing and well understood technology as possible. That is why the time tested business notion and stability of DB2 for i5/OS is such a key element in the System i modernization strategy. At the host system level, your AD team does not have to start from scratch to rapidly move ahead. Delphi/400 interfaces with your existing DDS-built databases as well as those that you may build with the SQL Language. As a modern RAD development tool, Delphi/400 is about as good as it gets to get your modernization efforts moving.

What is Delphi/400 and How Can it Aid in Web Application Development?

Delphi/400 is a suite of application modernization tools designed to enable System i application developers to build completely new Web applications or build new Web interfaces to existing applications. For the System/i developer, there is lots of good news. The approach is based on the same notion of holistic application design and user interface / logic separation that System i developers have been using since the box you and I love was once called the System/38

Delphi/400 is the toolset that best addresses the notion of the application factory of rapid application development. It is the natural next step in a progression of tools from those with sophisticated names such as "Intelligent Development Environment," "Componentization," and "Visualization." Yes, It is all of those and more. It does its thing by asking the developer to think about the whole application, not just one Web page at a time. Isn't that how System i developers already think?

Time to Look to a Leader

After many years of hope, with V6R1 of i5/OS and its supporting cast of programs, IBM has yet to provide the long-hinted, if not often promised natural Web interface. After 18 years, for System i developers, name changes come more quickly. Isn't it time to stop waiting? Of course this means that neither RPG developers nor COBOL developers will be replacing their green screen oriented display files with an IBM packaged natural object with HTML or XML GUI. This will not be happening because IBM does not want it to happen. If it is ever to come and there is no reason to believe it will, it won't be for at least two or more years. For Baby Boomers at this point of their careers, that's more like a lifetime.

IBM in recent times programs its major software releases for System i to be released every two years. Some say this is whether the community needs it or not. Clearly with so many System i users on back-levels of the operating system, many have not seen enough value in IBM's recent efforts to make the move to upgrade. The natural IBM RPG interface, the proverbial Web Device Object File, is already "too late" to stop the SMB erosion. Two years from now will just be "too *later*."

Many of the graying baby boomers that I grew up with in this industry have already begun to move their permanent offices to cruise ships and/or exotic islands. To these platform stalwarts, who saw the AS/400 heritage machines as the ultimate technical professional experience, the new office locations will certainly be the ultimate visual experience. Most mature System i developers prefer to switch or quit rather than fight IBM at this point. What's coming to the System i community has already arrived with V6R1 and for developers it happens to be the same as 1978. Nothing is about to change. So, get over it!

Though to many who perform their daily System i tasks without complaint, this is seen as still adequate and superior to other available options, my suggestion is to reevaluate that posture. It is not sufficient for running your business on the Web. It is not good enough professionally for developers to live with half-working tools and half-baked products from the 1980's. More importantly, it is not the deck of cards that the modern System i developer must accept. That is why I am taking the time today to introduce you to a better way. It is long overdue, but it is complete, productive, modern, and best of all, thanks to the CodeGear and System Objects partnership, it is the best development environment in the world. Now, it can be used with the most productive back-room system in the world, the System i.

While others have been waiting, CodeGear has continued to leverage its developer community on to great accomplishments. Incidentally, this community is well over three million strong and they love CodeGear by continually investing in the company's AD wares. CodeGear for its part continually invests in making its products the best of breed. This testimony from a Delphi customer in the same period in which IBM was in the middle of a big System i tool mess shows how much Borland/CodeGear's clients appreciate the company's lead in Rapid Application Development.

"We have had incredible success using Delphi for software development," said Omar Sayed, CEO of Succeed Corporation. "We expect the new features in Delphi 2005 to help us speed development and maximize our existing investments and skills, while making the most of emerging business opportunities like eBay's powerful Web services platform."

Delphi 2007 and the new 2008 capabilities have taken the CodeGear and SystemObjects product set from *international and outstanding* to *out of this world and even better than best*. Yes, these plaudits may seem like hyperbole. However, the level of excellence in function and ease of development and the continual improvement in the product package itself is worthy of all the superlatives I can muster.

Powerful Rapid Application Development for System i

I applaud the work of SystemObjects in taking the Delphi product set and seamlessly providing the System i piece of the puzzle. Why should System i Developers, highly invested in PDM and RPG not be able to move on to an industry standard for Advanced Web Development? These teams partnered more easily than Rational and Rochester and they got this done because the CodeGear Labs and the System Object Labs operated like they were one and the same Lab. In many ways their work is as symbiotic as the way it once was with the Rochester Lab (Operating System i5/OS) and the Toronto Labs (Compiler and AD tools) before Toronto began working for IBM's Software Division and all IBM AD for all IBM platforms had to look the same, for better or for worse.

That which is good for the goose is or rather *should be* good for the gander! While many System i folks were waiting for something that would be AD revolutionary, the fact is that we waited too long. While companies like CodeGear and SystemObjects were making life better for System i developers, IBM was still trying to figure out how it could smuggle EGL into our hearts and minds, and they actually chose to suggest that the language is RPG-like to get us all to fall for the ruse. Those inveiglers.

CodeGear's Delphi is not RPG-like. It is exceedingly good however. And those developers who make their living providing the best of the best for the Web think it is likewise quite good. The product that I suggest for review is in its eleventh version, Delphi 2007. It is not solely System i or it would not survive. It supports the Delphi programming language (Object Pascal) and C++ for the 32 bit Microsoft Windows platform, as well as Delphi and C# for the Microsoft .NET platform. For the System i developer wondering why they cannot get a job in another (non-i) shop, with IBM's share of the market declining, it's time to come to grips with these terms.

Delphi 2007 is superlative. That is a simple sentence representing the exact degree of high confidence that the product inspires. It's not simply because it is a strongly typed language. Its visual tools make "real work" a thing of the past. Its built-in visual component library to avoid using code for common Web functions such as lists (subfiles), prompts, text fields, pull-downs etc. has not yet been met or matched by other IDEs. Nobody else has these things.

CodeGear's Delphi is the clear leader in the tooling game. More importantly CodeGear makes its changes incrementally so that instead of you learning the new way and having IBM rip the product apart and put it back together incompletely, Delphi has been an AD staple for over twenty years. In fact, CodeGear has over 3.2 million users of its products in 29 countries. That certainly puts the company among the world's top tool providers—and the only one focused exclusively on tools—and it shows. Quite frankly, I am glad they asked me for this evaluation.

How Does Delphi/400 Get its Job Done?

You bring up the Visual, Componentized Intelligent Development Environment on your workstation. Unlike WDS, you do not have to wait five minutes. If you want a nice title on the page, you bring in a nice title component. For a text box heading, you drag and drop a heading and type in the text. If you want a list or a drop down or another type of output or input component, you find it on the right side of your screen and you bring it with your mouse to the design image area.

To provide logic, on the same panel, there is an editor facility somewhat like VARPG that pre-populates the sections that you need and you just add the logic. You use Object Pascal, built with one of the clearest and easiest to understand syntax of all computer languages. When you are

finished, you tell the tool to build the application (may have one or many pages like a display file) and then you can test it right on your workstation by bringing up an inboard browser. When it looks the way you want, you deploy it to the System i. It's just about that simple. And by the way, it works....

But what if you want to change the application after it is created. Just like any source application, all of the components are available for you to change within the Delphi environment. You rebuild the application, test it, and redeploy it. It sounds like how you would do it—even on green screen. That is because it is the best way! The bottom line is that you do not have to work with WebSphere tools to get your new applications or your modified applications up and running. If you are using the PHP version then you can take the code and move it to any PHP platform including System i. The Delphi400 for PHP product does this for you.

One of the sharp criticisms of System i modernization tools is that the client is often left with little more than a dressed up 24 x 80 screen template regardless of how powerful the browser itself may be in extending the horizontal and vertical limits (and other limits) of the application look and feel. The Delphi/400 tooling has no such limitation. Moreover, the application that is created has no such limitation. With this tool, you get all the Web has to offer.

The standard controls for the developer to use to specify data features are fashioned after SDA so there is no need to learn 40 tools in order to tailor the behavior of your Web pages in your application. You simply choose numeric, alpha, edit code, values, comparisons, etc. Without making this a highly technical paper for example, the tool can create full applications, not just single pages. The only thing close to it runs on green screen and it is called a 5250 display file. With the VCL, this stuff is Web-ready and when you use this tooling, your applications are Web-ready.

Since the application GUI is built with AJAX components, there is a lot more that can be done in the browser to ensure the validity of the data as well as the user experience. For example, when users begin to enter information in the field of a Web application created with the Delphi/400 tooling, the developer gets to pick the right component for the job. There is no macro or program-like coding needed to provide the editing for say a certain customer name or a customer number. In this way, users can search for and locate specific customer records in DB2/400 very quickly using the built-in ODBC drivers. The AJAX technology also is very handy for creating F4 "pop-up" screens. Note that this refers not to the annoying pop up that your browser does not permit anymore but pop ups that enhance the overall entry experience.

Please note that Delphi/400 is not a generator product; it is a visual IDE and much more. The typical System i RPG/COBOL developer wants control of their applications and as a real AD tool, without waiting for IBM to build this for the System i, you can use it today with your System i.

CodeGear and SystemObjects built it for you.

How many times have you been out to a Web site, booking a flight or a rent-a-car, and you experienced the handy dandy calendar object? Instead of having to get a paper calendar or use the calendar on your PC, along with a number of other tooling components, Delphi/400 has its own calendar object. When you define a field as a date field, your Web users get to use the calendar object rather than having to type the date. That is a big hole in a converted 5250 application and the hole is patched with Delphi/400 so much so that the new road actually looks better than the original paving.

Can Delphi Be Successful for Your Organization?

The Delphi project was destined for success from its inception. The chief architect behind the project was Anders Hejlsberg, who also developed its predecessor Turbo Pascal. As all geniuses, Hejlsberg eventually got restless and later was stolen by Microsoft in 1996. There he worked on Visual J++, and was a key participant in the creation of the Microsoft .NET Framework, becoming the chief designer of C#.

Borland formed CodeGear in late 2006. Delphi 2007 was released on March 16, 2007. It was the first Delphi release by CodeGear. Furthermore, it was the first version of Delphi since version 7 that only allowed compilation of native 32-Bit Windows Applications. Most System i professionals don't care about the Windows part of this but the fact is that Delphi does quite well with Windows developers. The new release features include support for the Visual Component Library (VCL) for Windows Vista, making the product further capable of phenomenally advanced visuals.

Delphi gets its developer productivity from the CodeGear RAD Studio 2007. This component is a general purpose, rapid application development (RAD) Windows application development tool. It is capable of producing both native and managed .NET binaries that execute on x86 operating systems. The System i interface facilities enable its VCL for the System i Web environment. Native developers can write either Delphi or C++ code, and .NET developers can write code in Delphi for .NET. If none of this makes sense to you, it is because System i people typically only see what is necessary for the System i. Yes, all of these capabilities are highly desirable. System i developers have had to depend on IBM bringing the platform to the Web at the same level as the Web already is—not some level that those who have ignored the Web hopes that it may be. The good news in Delphi/400 is that even though IBM has yet to deliver Web AD, System i developers no longer have to be wishing and hoping. Delphi/400 is here, now!

RAD Studio 2007 can build Web-based applications, full-fledged web sites, thin clients, fat clients, application servers that those clients can access, web services (both clients and servers), windowed applications, ActiveX controls, code libraries accessible by any native or .NET programming language, and multi-threaded applications for running complex embedded systems and VCL for System i. In short, RAD Studio 2007 can meet the needs of any developer writing any application for Windows and .NET with interfaces to the System i from SystemObjects. It takes a long time and some very serious effort to reach any "virtual programming walls" while developing with CodeGear. The System i community has been dealing with one wall after another in AD for years and a trip to Delphi is all one needs to get over the wall.

What About PHP?

Delphi for PHP is an IDE for PHP. It provides true RAD functionality for the PHP environment. If you are experienced in PHP, you know how cryptic some of the tools can be. This product features Delphi or Visual Basic like form designers, an integrated debugger (based on Apache web server). It also includes the VCL library ported to PHP. Support for Web 2.0 features, such as AJAX, makes it a unique IDE and one in which your shop can rapidly deploy business critical Web applications with the same confidence as your "tried and true" green screen traditional applications.

Finally, CodeGear has established Delphi for the "application factory" environment of the future. For Delphi, the future is now. The company sees the application factory notion as a game-changer for AD for the Web. Think back to the advent of the original IDEs (integrated development environments), which, by the way, CodeGear invented. They enabled a profound improvement in productivity. Some years later, IDEs were deemed commodities and productivity improvement was

thought to have peaked. Along came component-based design and visualization. With respect, while all this was happening, in the System i environment, the IBM tools were getting further and further from leading edge.

Once again, a meaningful improvement in productivity was enabled by such new innovations. The methodology behind application factories represents the next wave. A common theme you typically observe with innovations in design that lead to major productivity improvement, whether it be software, electrical or mechanical design, is a practical leap in abstraction. CodeGear is already there.

The application factory approach will enable this practical leap in design abstraction. Just as with native applications, using Delphi, this approach will enable System i developers to think at the application level first. In addition, this approach will help address one of the biggest issues facing companies today--efficient knowledge transfer. All of the knowledge will have the code project as its repository.

Companies struggle to harness intellectual capital due to changing project teams, organizational churn, turnover, and the necessity for distributed development. By capturing logic and intent in the application, companies will be able make significant improvements about how they leverage knowledge across their organizations. The result will be the creation of application factories, repositories of truly reusable software assets that can effectively be leveraged across an organization. Isn't that how you would see it becoming if you had it your way? Isn't that the idea of object based systems such as the System i?

CodeGear continues to apply this concept of application factories in products the company will be launching this year. How can you fight it? The answer is that there is no need to fight. They have the right idea. The bottom line is that CodeGear is where System i shops wish IBM was today.

With the SystemObjects partnership, Delphi/400 and Delphi/400 for PHP can be deployed in your shop and used by your team to get out of the stagnant IBM development environment that has taken us all to the great fields of Nowhere.

Let us all now get into the arena of the future with many successes in many industries already. I would recommend using the RAD IDE that the best solution providers in the industry select because they know better... The company name just happens to be CodeGear.

There are many more items that SystemObjects and CodeGear anticipated you would need for your Web Development and therefore, the company has already included them in Delphi/400. If I listed them all in serial fashion, you would have less incentive to delight in visiting them at www.systemobjects.com. May you enjoy the trip to Paris?

Best wishes in your Web implementation efforts.

Brian W. Kelly

About the author of this white paper:

Brian W. Kelly was an IBM Senior Systems Engineer (SE) for 30 years, and has spent over a decade as a System i consultant based in Wilkes-Barre / Scranton, Pennsylvania. He is also an author of dozens of AS/400, iSeries, and System i books and numerous articles. He serves as an assistant professor at Marywood University, which uses the OS/400 and i5/OS platform and teaches courses using the platform as well. Kelly is also one of the contributing technical authors to IT Jungle's "The Four Hundred" and "Four Hundred Guru" newsletters. Kelly has written extensively about the specifics of a natural GUI interface for System i and its development tools and compilers. He also provides counsel regarding the future shape of System i GUI facilities to IBM's Rational Development Lab team as well as to IBM's Systems Executives.

You can read about the advanced internals of the System i in Brian Kelly's best selling book, [The All Everything Machine](#). Google the book title and add Kelly if need be and you will find a number of spots on the Web to claim your own copy.