IBM's Strategy for System i Modernization: A Live Debate

A Webcast from System iNetwork and Databorough

Panelists:

Bob Cancilla, IBM. Bob has spent 30 years managing large-scale system-development projects and technology for both large insurance companies and independent software-development companies and has been involved with AS/400 Internet technology since its inception.

Ted Tritchew, IBM. Ted works for IBM Global Business Services in Canada as a consultant in SOA and Web services practice.

Bleddyn Williams, Morpheus UK. Bleddyn has been working with the iSeries and System i community for 14 years and is the co-founder and lead technical architect of Internet solutions company Morpheus.

Mark Tregear, Databorough UK. Mark is co-founder of Databorough and a lead designer of X-Analysis and its extensions. He has worked on X-Analysis development for the past 15 years and on the AS/400 since its launch.

Stuart Milligan, Databorough. Stuart is the marketing director for Databorough and the author of an IBM Redbook, Modernizing and Improving the Maintainability of RPG Applications Using X-Analysis Version 5.6.

In this exciting live debate and discussion (May 31, 2007) about IBM's strategy for System i modernization, IBM's Bob Cancilla clarifies IBM's strategy for companies who rely on legacy System i applications for their mission-critical business operations.

Stuart Milligan (SM): The background to this debate stems from discussions we've been having with our customers worldwide over the past several years about modernizing System i applications. A number of different technologies, opinions on exactly what route should be taken, strategic views on how modernization should take place, and technical solutions are being propagated by different companies. At the same time, IBM has been putting out a series of roadmaps and changes to those roadmaps over the past few years, and the overwhelming feedback that we've been getting from the marketplace is that while the roadmap lays out the technological structure within which modernization can take place, it's not entirely clear what IBM's strategy has been and still is in terms of the modernization of System i applications.

Databorough's been involved — really from an agnostic point of view — in this particular subject for quite a long time. We have an interest, obviously, in that the core product set that we offer is aimed at anybody who has a legacy application. And we certainly have a number of opinions that Mark Tregear, our chairman, will be putting forward today. But largely we're more interested in the overall strategy that IBM has for System i customers — which technologies and solutions are part of their strategy and where the strategy fits from a commercial point of view. What we hope to achieve with this live Web debate is to answer some of the questions that are still hanging with regard to System i modernization strategy

from IBM's perspective and to also play out, if you like, some of the actual debates that will be taking place internally in your organizations.

As for the format for the presentation, Bob Cancilla will present the IBM case, followed by Bleddyn Williams, Ted Tritchew, and Mark Tregear making counter-arguments and asking Bob direct questions. Then we will open up the questions to the Webcast participants.

Bob Cancilla (BC): I want to preface these remarks with the fact that what I'm presenting is IBM Rational's modernization strategy and that, as many of you know, a great many of the tools that you have been using on your System i were transferred to the Rational brand from WebSphere as of March 1 of this year. That includes WDSc [WebSphere Development Studio client]; the compilers — RPG and Cobol, as well as C and C++; and the WebFacing and HATS technology, WDHT [WebFacing Deployment Tool with HATS Technology].

The fundamental issue we're talking about today is modernization and a changing world. The reality is that the world is changing, that businesses are changing. Where once a business was a self-contained entity, today we're seeing global businesses. We're seeing manufacturing done in China, the Philippines, throughout the world, from U.S. companies. We're seeing insurance companies outsource major business departments to India and now China. Where once you used a standard EDI interface with your business partners and trading partners and suppliers, we're now seeing a global interaction with systems around the world.

We embrace the fact that there are large existing systems, many written in RPG, some in Cobol. We've also seen that customers are now no longer just System i shops. (Back in 1990, when I first got involved with the AS/400, everybody was an AS/400 shop. We had 5250 terminals and private networks.) Today we have Web, GUI, SOA [service-oriented architecture], B2B, and Web services applications. In fact, if we look at the System i, where once we had an AS/400 that ran OS/400, the machine today runs i5/OS. It runs AIX. It runs Linux. It integrates Intel technologies to provide Windows support. The world has become complex.

There is no one solution for every customer.

So how do we deal with multiple technologies and increasing crises in terms of human skills and the skills of developers trying to embrace all of these technologies? IBM certainly endorses Java-based technology, based on our Eclipse platform, and Rational tooling. But we also recognize that open source is playing a role. System i is promoting the use of PHP [Hypertext Preprocessor], and we certainly think that is an option for many of our customers. The System i community is vast, with several hundred thousand System i customers. There is no one solution for every customer. So what we're trying to do in terms of our strategy is to come up with a roadmap. You're not going to see a new roadmap from Rational; you've seen the System i roadmap, and that's a good approach to all of the various alternatives that can be predicted. What we propose to do in Rational is to work with you to leverage our tools and technology and build a custom roadmap that meets your business requirements, that meets your budget, that meets your skills.

The other thing about modernization is that we're not coming in and advocating that you convert all your RPG or Cobol apps to something new. That just doesn't make good business sense. One of the key ingredients of modernization as we see it is to discover what you have, understand what you have today, map your business requirements to your existing system, identify opportunities to enhance and modernize, and move forward progressively over time. Time can vary, depending on the nature of your business and the requirements of your system, from a very short-term, very rapid modernization project to a project or strategy that could last perhaps 10 years.

As I mentioned earlier, Rational now owns the RPG compilers, WDSc, and WDHT legacy tools. I can assure you that we will continue to enhance and support existing technologies as long as there is a significant customer base using them. We have, however, introduced a new product we call Rational Business Developer, based on Enterprise Generation Language [EGL], and we view that as the path to the future.

We offer basically five options, or five patterns, for modernization. One is rapid Web enablement with our WDHT product, which combines Web basics and HATS. Putting a modern interface, either Web, or GUI, or Web services, over your existing 5250 application is a tactical solution to rapidly move you from a 5250-based environment to a modern GUI-based environment.

But while you're doing that, we also encourage you to look at EGL as your strategic modernization tool and development language for modern technology.

Second, we encourage you to Web-enable: Build Web-enabled and new apps in EGL, leverage your database, call existing programs.

Third, we look at the option of creating service wrappers over vendor-provided programs, over existing callable RPG or Cobol. Another, fourth, option, and this is where Databorough comes into play, is to refactor monolithic RPG or Cobol programs — where you have a 5250 UI [user interface] business-logic database access all in the same program — into a Model II or MVC-type [Model-View-Controller] environment, separating the UI from the business logic, from the database access. You can use EGL to build modern apps, Web services, and SOA, or just build modern Web or GUI applications leveraging the business logic in your existing code.

The final option is to work with some of our partners, such as PKS Systems in Germany or Cornerstone in Belgium, who

have products that will convert RPG to EGL if that meets your requirements.

We are not saying that you should pick any one of these strategies or options. In fact, you may need all of them. The key is to mix and match and build a strategy for you, build a roadmap for you, that meets your budget, your business requirements, and the skills of your people.

One point about EGL and Rational Business Developer: You can learn it rapidly. Learning curves are generally a week to get up to speed, about a month to become expert. It's a tool, and one of the key issues that we see in the System i environment is that in Java, you have a professional programmer/developer that focuses on the technology and the skills of building application code. In the System i environment, we see that the RPG developer is more of a business analyst whose focus is solving business problems. EGL is a modern alternative to RPG that utilizes the same skills. Our objective is to hide the technology and provide equivalent functionality to what you've been able to do for years within RPG.

So pick from our five modernization options, build a roadmap, and move forward. Leverage partner tools such as Databorough's X-Analysis to understand what you have, build your migration plan, and move forward progressively over time and meet the needs of your business with our tools. Rational offers a complete suite of application tools. We also offer Rational Software Architect, which is a UML-based [Unified Modeling Language] modeling tool, so we integrate modeling into the mix. Rational Application Developer is our pathway to Java for advanced Java development. EGL integrates the work done by these Java developers in a manner that a business developer can engage in. We also manage the total lifecycle with our RequisitePro for requirements, definition, and documentation; ClearQuest for process management; and ClearCase for software change management. And we bring a complete suite of testing tools, from building a test plan to a manual product, a manual tester that helps you test, as well as automated progression testing tools.

Bleddyn Williams (BW): I guess from the perspective of someone who's been doing Web development around the iSeries for a long time, one of the key issues we see is that customers, and people who work with the iSeries, still don't understand the message that IBM is trying to put out. People are always looking for direction. We get a lot of questions from people who've got a lot of RPG code that's sat there on their AS/400, and they're really looking for information on what they can do. Having these strategies in a way sounds almost like having a roadmap. And people are looking for examples on how they can do things. You're always going to feel more comfortable if someone else has already taken the path and has done all the hard work, and you can then look at what's gone on and match that against where you need to go.

I think IBM tends to have so many products. Bob was just talking about things from a Rational point of view, but within IBM, you also get all the other competing products. If you start to bring Portal in, then you get some of the latest development tools. And then you get into things like Portal Factory, and you get into all these different pieces from IBM, and it can be a very confusing message to customers. I think with a lot of the development that's going on within IBM, it also then adds another challenge for your typical AS/400 shop, that they have to then look after things like WebSphere. It's something I've worked with since Version 1, so I know how good it is, but I also know some of the challenges that can be there, especially if you're coming at it with no experience.

You're always going to feel more comfortable if someone else has already taken the path and has done all the hard work.

So there's all this complexity that goes along with the changes. A lot of people out there would be very comfortable with their RPG development and their green-screen development environments. From our perspective, the challenge has always been trying to help them move across. I think when you go to a big piece of PC-based software, the problem I've always seen, with things like WDSc, is they contain an awful lot of components. We need to show the customer and the people who are going to work with those products exactly what they can do for them. And they're almost overwhelmed with the technology that's out there, so they end up getting confused about what they perhaps should be using and doing.

BC: IBM does offer a vast array of options. I like to focus on business development. Ninety-five percent of our internal development within IBM is done in Java. We understand exactly the benefits, and the challenges, of embracing Java technology. EGL provides support for the Portal that you mentioned. We're looking at the Portal Factory tooling. Right now we don't support that, but we do support the development of portlets in EGL, so we integrate with Portal.

I think that the key issue is that today's world has two levels of technology. First, there is the advanced technologist, and I suspect that many customers have some Java developers on staff. In my view, Java is the assembly language of the twenty-first century. You can do literally anything in Java. We integrate

Java with EGL so that we can focus on the core business developer, who represents the other level of technology. Our objective in Rational is to focus on solving business problems.

In terms of the roadmap, and of the five key options for modernization that I've laid out, rapid Web enablement with WDHT or WebFacing, Web enablement, new applications, leveraging existing database or code, service wrappers, refactoring, and conversion are the elements of building a roadmap. But while I agree with you, Bleddyn, I don't agree that I could produce a generic roadmap that will meet the needs of any particular customer.

BW: I think it's not just a case of it being generic, but it needs to contain, for people to understand, some examples of what they can achieve and how they go about doing it.

SM: In response to that, an important point to make is that one of the more underutilized and unknown areas of IBM's kitbag, if you like, is the IBM Redbook system. The IBM Redbook system is designed primarily to achieve the objective that you've described. I think what a lot of people don't realize is how these Redbooks are produced. What happens is that people go and spend time in the various development centers, either in Rochester, Raleigh, or other IBM locations. There, they take practical examples using the technology and real-world situations, and they work with the various solutions from the various IBM products to actually deliver on a methodology that will actually be, practically, implementation. For example, X-Analysis has a Redbook on how to modernize System i systems, and that describes the overall process. So perhaps the response to the point you're making, from IBM's perspective, would be for people to understand and utilize areas such as the Redbook system more effectively.

BW: That would go from an IBM point of view, as well, because they'll then understand more, especially at the SMB level, especially at the lower level, of what people are trying to achieve. And that's for me what it comes down to: having easy-to-use tools and just keeping it simple.

SM: What I pick up from the points that you're making, Bleddyn, is that the complexity of the environment in which they're delivering complex object-oriented applications, or Web applications, on large, complex System i legacy applications is a big challenge. I think where Bob's coming from is that IBM and the Rational tooling is centralizing a lot of that functionality in a single toolset, and maybe add to the mix of that the fact that the Redbook system will be more and more effectively deployed in actually providing practical guides to using these technological tools to provide solutions for the average iSeries shop.

BW: From a Web development point of view, it seems that the only strategy at the moment is to do everything in PHP.

BC: I think that PHP is excellent technology, and I think that there is a place for PHP in terms of rapid Web enablement. Quite frankly, though, I see some challenges in the fact that it's open source and the fact that many commercial customers have constraints in embracing open-source technologies. I think it's terrific that System i has embraced PHP, just as it did Perl early on, and other technology. It's one more choice.

BW: I agree. It's just that at the moment, it sounds as though it's the only choice you should be making, which I think is just quite confusing.

Ted Tritchew (TT): I think any modernization strategy has to take into account that technology is changing really quickly right now, especially on the UI side, what with things like Ajax and Microsoft's XAML [Security Assertion Markup Language], with Microsoft even pushing to replace HTML with a UI format-definition language. So the only thing you can be certain of is the "Once I Pick A" technology — the minute I write my first line of code, it's going to be obsolete. Therefore, I think I'll just end up having another legacy migration problem.

EGL is heading in the right direction, but does it really go far enough?

There needs to be consideration for future proofing, and I think generator technology like EGL is one way to do it. But I think in general we need tools that are a lot more business oriented. EGL is heading in the right direction, but does it really go far enough? I remember EGL was based on VisualAge Generator, which kind of ages me, I guess, but it's an interesting tool. I haven't used it for a long time. When I was at SSA, we realized that our portfolio of apps really had only 10 or 15 application patterns, and they just kept getting used over and over again. We found that we were building the same apps a lot of different times using different technologies, whether it was RPG, Java, or PowerBuilder.

I know that most modernization tools are based on the premise of applications getting written over and over again and recognizing these patterns. So what I'd like to see is a more business-oriented set of tools that allows application developers to configure applications rather than handcrafting code, which is very difficult. All the building blocks are there: You have SOA; you have semantic Web technologies like RDF [Resource Description Framework] and OWL [Web Ontology Language], which now allow us to actually define the semantics of apps. And I think Rational should really be heading down the road of providing a lot more business-oriented applications, which is really kind of the root of the iSeries.

BC: I think those are outstanding points. First of all, I absolutely agree regarding the constantly changing technology, and that's one of the benefits of EGL. We've had a question posted here from one of the attendees, "What's the future of EGL?" IBM is investing significant amounts of money in EGL. Our team is one of the largest development teams in the Rational organization, and we are investing in EGL as a cross-platform solution. We support the i, z, p, and xSeries environments. One of the key issues is that we are constantly monitoring the development of open-source technologies. IBM is aggressively involved in the open-source development universe as participants. As a technology emerges, we will embrace it, support it, and extend EGL to handle it. One of the key issues is having a core language that builds to the future.

In terms of building blocks and patterns, I totally agree with you. We are just now discussing this very issue in our planning, and you will begin to see an EGL community materialize — an open, public community — where both IBM and others will be encouraged to contribute to the environment and provide the type of ongoing environment that only a community can build.

SM: I think a couple of points that I'd like to make, just briefly, on some of the things that Ted was talking about is that we've been dealing with and discussing forward-engineering technologies. But I think one of the other issues that I've seen missing out of IBM's message from a modernization perspective over the past number of years is not really just about the tools that you're going to need in order to build new things, but really how you're going to take what you've got and take it forward. For instance, if you take the paradigm of modern development being a Model II architecture, Model-View-Controller, and companies have large monolithic programs, that's a challenge. The reality is that the overall scope of the challenge that people face in a modernization context is not necessarily just one of technology. It's not about all the fancy technologies that are available to do things once you've got a better architected application. It's more in the initial phases: Where do you start? Where are we right now?

Mark Tregear (MT): I'd like to second the points made by both Bob and Ted in their different ways. Our key product, X-Analysis, does support a rather specific roadmap. What we do is see the iSeries as effectively a repository of that business-logic information and patterns that you need to move forward. Ted was making the point that the technological side of system design is moving ahead very, very rapidly. It's unwise to choose any particular technology and commit to it, and I'd second that.

The correct thing to do, in our view, is to try and separate out from the analysis of your current iSeries system the patterns that Ted refers to. In other words, patterns of screen interaction, the start of transactions, information like the data model definition information, and to port that effectively to your target tool. Ted mentioned a whole range of technologies, but they all work in a pretty good way, providing they're provided with clean information. The challenge is, as I say, to process your existing iSeries system and separate it into, effectively, a view, controller, or presentation layer that consists of patent information — in other words, standard descriptions of how the programs should work — not the detail, DDS, indicators, and all these iSeries technological details, but an overview description. And that's handled basically by object-oriented programming records.

It's unwise to choose any particular technology and commit to it.

The larger part — the bottom part of the iceberg that sits on an iSeries application — is all the business logic that's sitting there embedded in your RPG application. And what we've found happens in real projects is that there is such a depth of iSeries business functionality, it's quite complex. A little of that is just taken across raw into a new technology environment without refactoring. We actually have a bigger maintenance problem potentially than maintaining it in its conventional RPG form. So our tool applies analysis and refactoring methods to separate the business logic components out of a current RPG or Cobol system and turn them into business-rule modules.

And why that seconds Bob Cancilla's arguments is that what we find is that in the first instance, those logic components are best expressed in RPG. It's best to take the presentation layer, in our view, totally across into new technology, use the latest tools, and have the skilled technologists, people like Bleddyn and Ted, working on it. In the short run, it's better to leave those legacy logics over on the iSeries side of the equation.

But the community, the iSeries community, has been struggling to find a portable technology that will actually take those business-logic components and maintain them in a more robust and rigorous way than is allowed by straightforward Java programming. People do take legacy logic and convert it all to Java, but it's a heavy-duty deal. And it's our expectation and our belief that EGL is the best route forward currently offered for taking legacy logic into the new environment.

BC: I think you've stated the case perfectly, Mark.

SM: The question I would pose, which maybe we skirted over a little bit, is you've been talking about the fact that the software group has taken over the control of the compilers and the various tooling the iSeries shops have been utilizing for a lot of years. Do you want to talk a little bit about what you see as

the future of RPG from your perspective? Is it an important factor? Are you going to invest in it moving forward?

BC: As for the future of RPG, we have thousands of existing customers who are using RPG as their core development language. The strategy that we're putting forth is that we will definitely continue to invest in RPG. Don't look for modern UIs; don't look for GUI applications being built in RPG. That's not likely to happen. Do look for closed integration between EGL and RPG. I see that most customers are going to be working in both EGL and RPG over the future — and PHP, for that matter. So will we continue to invest? You bet we will continue to invest. IBM is not going to leave any customer behind. As long as there is a significant install base of RPG out there, we will continue to enhance and support the product that builds RPG and the compiler itself.

Regarding EGL and PHP: You know, this is a matter of personal choice, I believe, and it's a matter of investing in an emerging technology versus IBM technology. And I think that the real answer is to take a look at EGL, compare for yourself, make the decision for yourself. Consider the driving factors behind the two languages; consider that one is an IBM product. We have taken the initiative to standardize EGL; we're working with the OMG [Object Management Group] to create a language standard, a public language standard, to encourage other vendors to embrace EGL. We are looking at contributing something from EGL, though we don't know exactly what that's going to be, to open source. As you know, it was IBM Rational's team that donated Eclipse to the open-source community. We make major contributions, so we will be contributing something to the open-source community in terms of EGL.

SM: I just want to clarify a point that you made. You're saying that EGL will in time be opened up to the open-source community, so other vendors will be able to make enhanced contributions or enhancement contributions toward the actual code base itself. With regard to PHP elements, I think the point you made earlier on was that EGL will actually be able to consume and run alongside PHP, but it is designed for different purposes.

Does IBM's modernization strategy offer help to customers with an RPG II code base?

BC: Yes. Plus, EGL is a complete end-to-end development environment and language, including support for batch and 5250. It's a complete environment, whereas PHP tends to focus on its Web enablement.

MT: In our experience of doing these sort of transformations and dealing with the iSeries community in general, we would say that the larger half of the problem is the legacy-logic-analysis half, the dealing with the business-logic side of the picture rather than the screen interface and presentation layer. PHP is a presentation-layer technology, as are many of the things that Ted mentions. The key productivity aids that we find useful to the iSeries community are essentially analysis tools, the ability to be able to visualize the RPG, or the future EGL system, at a higher level — code analysis, data flow diagrams, data model diagrams, structure charts of the system, the ability to be able to chase through references and edit your code at a much higher level. And that's a tool-based approach, which we think will give a great productivity boost regardless of whether one's working in RPG or in EGL. That's very much what's documented in the IBM Redbook on this modernization topic.

SM: RPG is a fantastic language with which to solve business problems and describe business solutions to business problems at a language level and then generate applications from it. Therefore, by definition, the legacy application is a definition of business logic described in RPG. So the point you're making, Mark, is that if you want to modernize your system, you may put different user interfaces on it, but the true modernization is to reutilize that large amount of business logic that's buried in the system. And that's largely an analytical process rather than a technological one. Would that be a summary of what you're saying?

MT: Absolutely. RPG is best used as a database and logic language, not as a screen-interface language, I think, if you want to have tools that make the best use of its capabilities.

SM: Perhaps the point that Bob's making, then, is that business applications are really a description of the business logic and business process manifested in a piece of code. And EGL is an excellent technology with which to do that. As Bob has pointed out quite clearly, you can use these languages to do many more things — Web services, graphical user interfaces, Web applications. You're not constrained to one particular technical implementation of your business logic. EGL can do any one of those, which then fits into the point I think that Ted was making in that if you have a higher-level way of describing what your business process is, and then let the application generate the implementation code itself, it means that you then can keep the architecture of your business system sacred and the integrity of it moving forward. And then let the technologies come and go as they please because they are the ones that change most rapidly.

I would like to ask a question here. Bob, does IBM's modernization strategy offer help to customers with an RPG II code base?

BC: Since March 1, Rational has been aggressively planning what we're going to do with the products we got from WebSphere, including the RPG compilers. The answer is we will absolutely embrace RPG II, and we will embrace System/36 compatibility, and we will embrace System/38 compatibility. Our goal is to leave no customer behind in our modernization mission. Look in about a year for specific IBM offerings, but I believe that with our partners, we can address much of this today. I'll let Mark speak to what they offer, but IBM will be there also. We will be providing tools and mechanisms to move you up to the current levels of RPG and will embrace it with our partners in terms of other modernization approaches.

MT: I have something to add to that. If you adopt an automatic refactoring strategy, the fact that your old code is in RPG II is in fact less of a problem than you might think because, as Ted's amply described, the presentation layer is going to be refactored from patterns anyway. In other words, we're going to recognize that it's such-and-such a program, a display program or an edit program, and rebuild the interface anyway. So whether that was in RPG II or RPG IV, it doesn't really matter. And in regard to the other primary characteristic you get with RPG II code, that the database is not externalized, our own software in fact automatically modernizes that so that it is dealing with an external database and thus is in a much better position to being migrated.

SM: One question has come through asking us to explain and compare EGL and PHP. One thing I can say to our audience is that we can supply some links to overviews of the different technologies and where they might fit together, and we'd be very happy to talk to you. Various people in this presentation can help you with understanding those various technologies and where they might fit in.