

Take IBM WebSphere offline with IBM Lotus Expeditor

Abstract: In this podcast we will discuss how Web applications can be run locally by using the IBM Lotus® Expeditor client and its tooling. Lotus Expeditor software extends the Java(TM) Platform, Enterprise Edition (Java EE) programming model to managed clients on laptops, desktops and mobile devices.

Transcription

DUTCHER: Welcome to this IBM Lotus® Expeditor Podcast discussing “Extending IBM WebSphere® applications with IBM Lotus Expeditor software.” I'm David Dutcher, Worldwide Technical Sales Support for Enterprise Access, and I'm joined today by Angus McIntyre, product manager for Lotus Expeditor software.

McINTYRE: Hi, David. Thanks for having me here today.

DUTCHER: Angus, explain this podcast series. What's this series all about, and why should developers care?

McINTYRE: David, this is fourth in this series of podcasts on extending applications out through to the edge of network. It shows IT organizations and application developers that they can reuse both their skills and their applications that they have running in production today and extend those applications out through to people that need information – the end user

Lotus Expeditor provides a “fit for purpose” application desktop that can give the people the exact information that they need to help them accelerate the business process. The software that works the way people think to get them through business processes faster.

This series is really in six parts. [1] We started out with an Introduction to Lotus Expeditor. [2] We talked about taking portal offline with Lotus Expeditor. [3] And also business process automation with offline forms in Lotus Expeditor.

[4] We, this series [podcast] is about taking WebSphere offline with Lotus Expeditor. [5] We'll follow up with mobile access for PDAs and smartphones using Lotus Expeditor. [6] And then finally, we'll end off with Expeditor as an open alternative to Microsoft® .NET

[Framework]. And we'll even talk a bit about [Microsoft] Silverlight™.

DUTCHER: For IT organizations, what problems does Lotus Expeditor [software] solve?

McINTYRE: You know, if you take a look at end users and you go into the desktops that they look at, any one business process may have five or six -- or 10 or 15 - - different applications open at any one time.

Look at our own desktops, right? It's sometimes a safari in terms of searching for the information that you need... information that is really needed at your fingertips.

For call center employees or car rental agents or bank tellers, they use a wide variety of applications, and they need that information right at their fingertips in a coordinated fashion so that they can get through the business process very quickly.

The ways that IT organizations can use this to solve this business problem.... Well, we've got to look at what we really want to take to end users and provide them. You want them to be hidden from the complexity of searching through multiple applications by having a comprehensive desktop.

You want to consolidate all that information together into one composite desktop -- or mashup -- that has all the information that's "fit for purpose" -- for that process, for that task at hand.

You want to make that information roles-based, so based on if it's a withdrawal or a deposit, et cetera, you have all the information that's needed on the desktop in front of that bank teller.

It's all about tying business processes out through to the people that feed these business processes on a day-by-day basis. It's an understanding that people reuse applications, like the 5250/3270 screen [they] have been running for years and years now. How do you bring that forward and integrate it into a desktop application that forms a comprehensive multi-channel access to information in business processes?

And finally, how do you secure this information both on the client and back through the

business processes? You take a look at the things that [Lotus] Expeditor provides to the end users, when you reuse applications within Expeditor. I talked about 5250. You can also have [Microsoft] Visual Basic®, Web-based applications, native applications, browser plug-ins, Eclipse™ RCP- or SWT- based applications [that] can be integrated together. [Editor's note: Eclipse is an open source community committed to implementation of a universal development platform.]

This reuse of applications makes the environment familiar to end users. You're reducing end user training because you're reusing software. It's a productive user interface.

You can actually take out the actual URLs that end users can browse to. So you're creating a fit for purpose, comprehensive desktop -- a composite desktop -- that's free from any distraction. And then you're allowing applications to integrate at the desktop level.

Finally, you have to realize that your end users are not technicians. Expeditor [software] is convenient to use because it's centrally managed. It's multi-user enabled. It also has secure elements in terms of local authentication and then access in through to LDAP2 back-end-server-based applications.

And in conclusion, you're essentially extending WebSphere or WebSphere Portal applications out through this composite desktop. It's a great open alternative to Microsoft .NET.

DUTCHER: Sounds powerful. So what types of end users is Expeditor suitable for?

McINTYRE: Well, we're starting to be used in many different scenarios. Let's look at the people who are always on, always connected: the call center agents or point of sale.

Let's take a contact center agent in a call center. They're looking up customer information, suggesting services, answering questions. Providing them a fit for purpose desktop with all of the information integrated together gets people through that queue faster.

Point of sale. Looking up account information. Looking for orders. And even activations of services. These people are trying to move people through the storefront very quickly. These people are always on, always connected, but they need a focus to move people through the queue very quickly.

Retail kiosks -- something that sits in the shopping mall where people walk up to and access information. It could be even an ATM in terms of a kiosk application.

Knowledge workers -- Knowledge workers that go offline for long periods of time, let's say on an airplane. And they still need to see business process information while they're working offline. Knowledge workers are handled with this.

And then finally, field service people. People that work on work orders and time spent on job, out in the field using PDAs, et cetera.

Expeditor runs the complete connection spectrum. For always on, always connected call center, we provide a rich, compelling and responsive end-user interface. On the desktop/laptop that provides virtually all the information right down to the semi-connected field service personnel that are out there that are filling out the work orders on a PDA and then synchronizing the data as they come back in and get network connectivity.

Finally, applications running on WebSphere are probably using the model view controller design pattern. They just needed to be extended out to through to the rich client platform. Expeditor provides that rich client platform. The APIs that we will discuss further in this podcast... We'll go through and show how familiar this programming environment is to WebSphere application developers.

DUTCHER: Okay, you've mentioned reuse of skills and reuse of applications. How does Expeditor accomplish this?

McINTYRE: David, it's broken up into three sections. When you look at it, as I mentioned before, model view controller.

1] Let's start at the access layer. You want to create a compelling end user experience, but you need the capability through middleware to take that experience and move it out through to the desktop. So virtually all of the APIs that WebSphere application

developers are used to -- things like JDBC for relational database access, SOAP for access to Web services, the Web Services Descriptor Language, the Web services security -- these are all integrated into Expeditor. To take the application offline, the assured messaging capabilities through JMF in micro broker and MQE are also provided within the environment. [Editor's note: MQE is IBM WebSphere MQ Everyplace.] So Expeditor really enables the extension of existing applications through known middleware APIs. Developers know the APIs that they're using today and they are just extending them out through to the client.

[2] Let's look at interaction. Right. Once the data gets there, how do people look at it? Well, currently today a lot of the applications are servlet JSP. Expeditor has a servlet JSP engine on the client. So if you provide enough information on the client so that page can be page compiled and rendered, it will render locally. Servlet JSP is well known programming model to WebSphere developers, and it's fully enabled within the Expeditor environment.

I mentioned in the other podcast that there's about 2.6 million Eclipse developers. They understand JDBC and RCP application development. Expeditor is built on that programming model, so people that are building plug-ins for Eclipse today will well understand the capabilities within Expeditor and be able to deliver rich client user interfaces to the back-end of business processes that they already have.

Expeditor can also be used as a client integration platform in terms of the interaction layer where we have the capability of doing [Microsoft] ActiveX® components in the client and integrate them with the back end systems as well.

[3] Finally, from a management perspective, Expeditor uses the OSGI™ services platform. And what this provides is the capability to remotely provision, maintain, update and remove software from a central location, remove it from the client from that central location.

Expeditor has also elements of security within it. Single sign on. Desktop lock down where you can control URL access or the ability to prohibit end-users from adding any plug-ins. They all must come from the central provisioning server.

So it's really about extending WebSphere applications out through the edge of the

network, desktop, laptops, tablets and then using the embedded rich client platforms for Eclipse out through PDAs and smartphones, as well.

DUTCHER: Well, Angus from a developer's point of view, what functions does Expeditor provide?

McINTYRE: David, it provides five basic functionalities. [1] Web. When you need Web-based access in a controlled Web browser environment so that you can lock down URL access to the end-users. You can browse any Web application inside Expeditor.

[2] From a rich client perspective, it's a rich client platform. You have optimal end user experience exploiting native platform look and feel.

[3] It's a composition platform -- the ability to help enable a seamless integration, and enabling composite applications integrated on the desktop.

[4] It's a semi-connected environment -- The ability to have full application support at the client when you're disconnected from the server and then synchronize with the application and data as you come back into connectivity range.

[5] And finally, client management. The ability to install, configure, maintain and remove software... in maintaining both the platform and the application.

So let's go these in order.

[1] Let's start at Web. Expeditor is a controlled Web browser environment, so it supports virtually all full feature Web content, HTML, dynamic HTML, JavaScript, Ajax, et cetera. Also browser extensions or plug-ins can be installed into the Expeditor environment -- Flash, Flex, PDF, et cetera.

Applets can also run within the environment, as well as Microsoft ActiveX controls, Microsoft Visual Basic applications, Microsoft Excel spreadsheets. Those are available on Microsoft Windows 2000 or Windows XP only, but they can be embeddable within an environment and integrated with the other applications on the client desktop. As I mentioned before, providing a locked-down browser environment to reduce end-user distraction.

[2] It's a rich environment; that's the second feature. It enables full drag and drop, keyboard integration, 2D and vector graphics capability, hardware device integration in terms of barcode scanners, et cetera, and it really protects the application layer from low level OS changes.

Finally, it enables platform look and feel so an application that's written to Windows 2000 looks like an Windows 2000 application. That same application if it's run on XP looks like XP. And if it's ported to Linux it looks like a Linux based application as well. That's the magic of SWT being mapped to native controls.

[3] Expeditor from a composition capability -- Mashups across pre-existing and new application types. The content that I mentioned before -- Swing, AWT, native, .NET applications, Web-based applications, portlets and forms. Rich client platform based RCP applications, Flash, Flex, text terminal, 5250, et cetera.

You know, inside this composite desktop that you compose fit for purpose for the end user, you have local data integration between these client applications and local event integration between the applications as well. Eclipse application developers will understand this data and event integration and our property broker makes that happen.

You can also create new views on existing applications, reface those applications out. You enable complete desktop management. You can optimize real estate, control the look and feel, you can even brand the workstation to your own brand or even to a business partner or a customer brand of yours.

And then finally, application reuse. When you're reusing all the applications and the mashups, you're reducing the need for end user training, which is probably the most expensive part of a roll out.

[4] Disconnection. Lotus Expeditor works in a semi-connected environment, with full application support at the client.

When you take applications offline, you need certain compute power local on the client. Take the network cable, disconnect it from your machine -- you'll get a "404 Web server not there." It's by taking the Web container locally -- the servlet JSP engine locally --

that we can render those pages, provided there's enough information on the client to do that rendering.

Then we have a local data store, and transactional messaging forwarding the information to back-end systems to provide access and make people productive virtually any time, anywhere... And then they have a synchronization capability as they come back online.

[5] From a managed client perspective, the Expeditor server has been tested over the course of tens of thousands of users. So from a central location you can provision, maintain, update and remove software from that end client workstation.

You can also lock down the composite applications in terms of browser-based control or even the end user's capability to download plug-ins into that environment. The whole notion here is that OSGI is used on the client side to do that the resolving of the application, are all the piece parts there on the client so that the application can even be evoked.

So Expeditor has an integrated server that administrators can use to deploy, maintain, update and remove client software from the machine.

[6] Finally, mobile. And here's where I want to talk a bit about mobility. We do have an entire podcast on this, but I'll mention it now. Don't start with a WebSphere application and then think you're going to shrink it down into a Windows mobile device.

You start small using the ERCP and you add the middleware elements that you want -- either transactional messaging or relational data. Don't start big and work down. Start small and work up. And that's where ERCP (and the ESWT plays) takes place. So Expeditor has tools that can enable you to build applications that fit for purpose on mobile devices as well.

DUTCHER: Angus, you just talked about the rich client platform. What value can Expeditor deliver on top of the open source project?

McINTYRE: Sure. That's a very good question, one that comes up a lot.

Let's start at the access layer, right. Eclipse RCP provides the integration capability between applications.

Expeditor augments this with... a managed browser capability so you can control the URLs that the end users go and look at when over the course of their workday, removing distractions from them.

There's a portlet container that ships with the Expeditor client. So you can view portlets offline in a controlled fashion as you're disconnected from the environment.

There's a composite application infrastructure which is composed of an application launcher. So this controls the life cycles of applications and the capability of integration between the applications as they live on the composite desktop.

And then as I mentioned before, restricted workbench -- the ability to lock down the workbench and have the workbench being used by multiple users.

In the access layer... we augment with the middleware that connects the client through to the server. The Web container for servlet and JSPs, the portlet container lives here. There's a directory in terms of LDAP capability on the back end through JNDI. There's a services layer for Web services, as well as data and transactional services and Web services, Web services client.

The property broker is in this layer as well -- to do application-to-application integration. There's a network aware capability within Expeditor, so it understands when their network attachment is available, so that it can automatically synchronize applications with WebSphere as it comes back online.

There's also a set of security elements within in here in terms of a Jazz framework and key store, a local key store capability, so you can store credentials locally and be authenticated to the system as you're offline.

And then there's an accounts API that allow you to go in and have multiple users go against multiple back end systems as well. That along with the enterprise management agents provides an inventory of what users and what groups have which versions of which bundled features on each client.

So it's really an overall client to server managed environment here. And it's been tested against the following clients: Windows XP Professional, Tablet, XP Home, Windows 2000, Red Hat and Novell Linux desktop.

If we go back to the device, remember we need to make the device a bit smaller because it needs to be fit for purpose.

Portlets don't really play in the device infrastructure so the portlet container doesn't live there. In this case, you have browser plug-ins, along with a rich client platform, you have browser plug-ins for Web-based applications, and then the application launcher to invoke and remove applications in terms of the life cycle capability.

In the middleware row, it's a lot similar to the one there, so there's the Web container, the directory elements, the services in terms of data transactions and Web services capability along with that enterprise management agent for inventory control.

On the client side platforms that we do support, it's Windows Mobile 2003 Second Edition, Windows Mobile Version 5PDA and new in Expeditor 6.1.1 which is Win[dows]CE 5.0.

I must also mention here that our programming model is symmetric so you can run the ERCP version of this on a Windows and Linux desktop. You can get a complete end-to-end programming model here by starting small, you can run these applications still on the desktop systems.

DUTCHER: Can you share with us an example of where this is being used?

McINTYRE: Sure, David.

If you look at where Expeditor is currently being used in terms of extending WebSphere applications -- banking here in Canada. A bank just finished rolling it out to wealth managers.

The wealth managers essentially take prospects offline, work with their clients and then resynchronize those as they come back with their laptop/desktop systems and synchronize it with the back-end WebSphere application server using middleware in terms of message queuing et cetera.

That same bank is now in phase two where they're rolling it into their call center, extending the same back-end business processes, but refacing it so that their call center employees have a fit for purpose desktop that provides exactly the right information to accelerate people through the business process while they're doing phone banking.

The next step that they will take along this journey is essentially going through and deploying it to teller workstations so that the tellers have access to the same applications.

Other areas where this is being used -- take a look at [it] from a mobility standpoint. Openstream... They have a wide range of field force mobility applications in terms of work orders, asset management, time spent on jobs, et cetera, all built on top of Expeditor on a PDA for field force mobility.

And that's used also in production on the west coast.

It's used within a government organization where you essentially provision to the device the current commodity prices. The commodity traders essentially take these offline and work with these prices offline and to capture orders in rural markets in India in Uttar Pradesh.

What they do is once they've taken the orders, they come back in, resynchronize those orders at the end of the day and get the next day's pricing so that they can take back and the market goes along the next day with new orders being taken and the latest and greatest prices being pushed through to the devices.

DUTCHER: Well, Angus, how is Expeditor using within IBM as an integration platform for client computing?

McINTYRE: That's a great question. We're "eating our own cooking." We're using integration -- an integration platform inside Expeditor -- to take virtually all of our client-based products and make them work together.

Let's talk about these products and how they use Lotus Expeditor software capabilities.

First, inside the [Lotus] Notes® 8 client, Expeditor is used under the covers to integrate SOA applications into the advanced collaborative desktop provided by Notes 8. So WebSphere applications under the covers with the APIs that I mentioned earlier are hooking into client side collaboration to produce the next high level of high-performance work environment.

Host integration transformation services as we talked about, you know, 5250/3270, business transformation. Runs on top of Expeditor and could be hosted with/inside that composite application integration on the desktop.

IBM system servers, both the System I [IBM iSeries®] and System p [IBM pSeries®] use the Expeditor infrastructure to deliver a more compelling user experience to the consoles of those hardware systems.

We talked about in another podcast the forms-based offerings from Lotus. Electronic forms for people on the move, the ability to do transactional messaging to synchronize forms and forms data. [IBM] WebSphere Commerce uses this to provide an online e-business compelling end user experience for people that are out using the commerce server.

[IBM Lotus] Sametime... for a Unified Communication and Collaboration offering. Sametime runs on top of Expeditor to integrate Web services into the collaborative environment that's provided by Sametime.

You know, Expeditor is used in this case underneath the covers. Expeditor can also be used in conjunction with Sametime to bring that real-time business communication and collaboration into line of business applications.

Application servers and portals. You're extending, as we mentioned, right up to the front applications beyond the data center out through to these rich client platforms. It's being used both by WebSphere and by portal as an open alternative to Microsoft .NET Framework.

DUTCHER: You know programming tools are really important to developers. What type of tools are recommended for use with the Lotus Expeditor?

McINTYRE: David, any Eclipse 3.2 IDE can be used with the Expeditor toolkit. We created it as a well-behaved plug in so that it can plug in to toolsets such as Rational® Application Developer. It augments the toolkit with Rational or the Eclipse-based IDE with the capability of building bundles for provisioning out through to the client.

We've been tested with other visual building software such as Instantiations Window Builder Pro, new in 6.1 is portlets created by [IBM] WebSphere Portlet Factory also run within the Expeditor client environment in their portlet container.

This augments the already available ability to host JSR168 portlets inside the Expeditor environment, portlets that are created within the Rational Application Developer RAD 7 toolkit.

We have to remember that Expeditor is aimed at IT shops that have both Java and Eclipse skills. That's not really hard to find since Eclipse has been downloaded to over 60 million platforms and analysts estimate that there are over 2.6 million Eclipse based programmers.

All of these will get Expeditor right out of the box. We have a four-day course offline that you can download and use locally at your leisure off of the Expeditor Web site. Download the information.

If you already understand Java and Eclipse, you have a two-day head start. The rest of the course is how to create applications and extend existing applications out into the client based environment.

DUTCHER: Sounds great, Angus. Where do I go to get more information?

McINTYRE: Sure. David, people should really start at ibm.com/lotus/mobile. Or, contact their Lotus sales representatives. You can also join us for the remaining two podcasts which again are, Mobile Applications with Lotus Expeditor, and finally, Lotus Expeditor as an open alternative to Microsoft .NET and a look at Silverlight.

DUTCHER: Thanks Angus. We really appreciate your sharing this insight. And thanks to those of you listening to this podcast today. Please visit us at ibm.com/lotus/mobile. And we hope you plan to join us for other podcasts in this series.

Thanks again.

[END OF SEGMENT]

Editor's note: In this podcast, "Expeditor" refers to IBM Lotus® Expeditor software.