Notes.net

Iris Today

Jason Dumont and Vinod Seraphin on iNotes Web Access

Interview by
Christie
Williams

Level: All
Works with: Domino 5.0
Updated: 06/27/2001

Recently-announced iNotes Web Access is generating a lot of interest. This month, we interview two important contributors to the iNotes Web Access effort: Product Manager Jason Dumont and Lead Architect Vinod Seraphin. They talk about iNotes Web Access, starting with a discussion of what it is right on down to technical details about how it was designed and implemented.

Note: *You can try out iNotes Web Access right here on Notes.net using our* ***live demo****.*

Also, for the month of November, 2000, Jason and Vinod answered questions about iNotes Web Access in the ***Developer Spotlight Forum****. If you'd like to discuss iNotes Web Access with other users, you can use the* ***iNotes Web Access category*** *in the* ***Notes/Domino Gold Release Forum****.*

Jason Dumont

Let's start with the obvious question: What is iNotes Web Access?
iNotes Web Access is our next generation Web client for Web-based access to Domino messaging and PIM [Personal Information Management] capabilities. It contains functionality, along the lines of mail, calendar, scheduling, group calendar and scheduling, to-do items, and contact management, as well as an integrated notebook for free-form thoughts—similar to the types of things you would put in a journal. It's based on current Web technologies like DHTML, XML, and JavaScript, so that we can produce a very functional, scalable, nice-looking user interface that brings a level of functionality to a Web client that has previously not been available.

Why a Web client?
There are a lot of enterprise customers who are looking to standardize on a browser as their standard client. Sometimes it's to reduce administration and

user training costs, because at this point, most users are familiar with browsers and don't need training, and you can implement and deploy applications very easily through a browser. There's also the situation where customers want to provide options in terms of the types of clients that their employees or customers or business partners can use to access information. So some customers might have a Notes deployment within their office, where you have a more-or-less controlled environment and the IT department has some say as to what goes on those machines; and then they might use iNotes Web Access via a browser either at home, where you have no control over what's on the machine, or at a kiosk or a cafe or anywhere that you need to share device access. The browser implementation is easier to deploy and administer. So, by having the option to have browser access for the same underlying infrastructure, it gives people a much greater variety in terms of when and how they can access a Domino back-end.

What was the main goal for iNotes Web Access?
The main goal was to provide a set of functionality via a browser that was previously not available to browser users in terms of messaging and PIM capabilities; and to do it in a way that would maintain an adequate level of performance to make it a very viable, deployable solution.

How does the product fit in, as part of the iNotes product line and as part of the bigger picture?
If you look at the overall client strategy we have, the three client brands we announced at Lotusphere in January 2000 were Notes, Mobile Notes, and iNotes. iNotes represents generalized client access, which at this point is predominately browser access. And iNotes Web Access is the key product in that area.

We have a product called iNotes Access for Microsoft Outlook specifically for Outlook. And we've got Webmail, which was the initial offering for Web-based access to mail and calendar data and functionality. We should come back to talking about that. And then there's DOLS—Domino Off-Line Services—which is a technology common to iNotes products that gives them off-line capabilities. But in the iNotes product line, iNotes Web Access is the premier product.

What about Webmail? What's the relationship between iNotes Web Access and Webmail?
The biggest complaint about Webmail to date has primarily been one of performance, and there's some significant enhancements in the 5.0.5 release to address that—primarily the areas around the use of Java applets both in the Inbox and when you create a new mail message. There is also some level of feedback from customers that it didn't have a comprehensive set of functionality, at least enough to meet their enterprise needs.

In terms of the level of functionality, iNotes Web Access has a vast improvement in functionality over Webmail. I'd say it has 90+ percent of the most-requested features that were not in Webmail—things like unread marks, spell check, out-of-office agent, unlimited attachment support, and a lot more. And our performance targets are well above the performance levels that folks have seen historically with Webmail.

But, in the initial release of iNotes Web Access, there'll be a limited client platform set. On the client side, it's going to be Win 32 [Windows 32] with Internet Explorer 5.0 and above or Netscape 4.7 and above. So Webmail will continue to exist, if only because of the platform limitations initially with iNotes Web Access.

In subsequent releases of iNotes Web Access, we'll be expanding our platform support. And Netscape 6.0 should be out and available at least on a number of platforms relative to the previous 4.x release, and we'll have enhanced browser support. Once we have full platform coverage for iNotes

Web Access, there will be less of a need for Webmail. So for awhile, there may be organizations that have both iNotes Web Access and Webmail deployed, because of the platforms they are using, but over time, as we expand our platform support, they'll migrate to iNotes Web Access because of the increases in functionality and performance.

The good news is that Notes and Webmail and iNotes Web Access are all based on the same underlying NSF file. So the upgrade path is very simple. Essentially all you're doing is changing the template from which you're inheriting your design. There's really no pain associated with that migration. It's just changing templates.

How does iNotes Web Access relate to the Notes client? Does it compete with the Notes client?
In fact, iNotes Web Access interacts very well with the Notes client. The same user can use both the Notes client and iNotes Web Access, maybe for different situations—the Notes client at the office, iNotes Web Access from on the road or at home. It just really increases the variety of options and access points available to users.

The Notes client is still an extremely viable client choice; anyone who says otherwise is ignoring the fact that it has a very advanced set of features that are just not available in any other client anywhere. When you start looking at the robust, granular security model that developed with the Notes client, for example, you can't mimic that in any scalable way within a browser. The things that you can do within the Notes client around desktop application integration, you can't do. The robust editing capabilities of the Notes client—just all of the things you can do in a Rich Text field like creating tables and sections and so on—are not available on a browser. So there will be many cases where only the Notes client will be suitable.

On the other hand, there will also be situations—for instance, if you've got Domino but you're using it for simple messaging and PIM functionality and don't need the high-end services of the Notes client—where iNotes Web Access is a better client for you. We'll give you the functionality you're looking for in an architecture that requires a no-to-low touch upgrade process.

Because iNotes Web Access is a very thin client, it is a much lighter load administratively. That can be a very significant value proposition. In a large enterprise, when you start to look at the money that can be saved in reduced admin and user training costs, it adds up to millions of dollars.

In addition, it's a client that is more viable in another market area, one that has not generally been able to leverage the Notes client. This market is made up of small to medium businesses who don't have an IT staff to administer Notes clients. Those resources just aren't available. If they're looking to take advantage of these types of collaborative services, without having a dedicated IT staff, they're starting to look at ASPs as a means of providing software as a service or hosted services. And iNotes Web Access fits very well within that model.

And it gets even more powerful in situations where you start to incorporate both the Notes client and iNotes Web Access for different types of access, because you get the best of both worlds.

For customers, it comes down to user style and what makes the most financial sense given the architecture and implementation a company's looking to provide to their users. We're not looking to take options away; we're looking to give options and flexibility to people.

You mentioned that on the client side, this first release is going to support Windows 32 with Internet Explorer 5.0 and above or Netscape 4.7 and above. Can you talk about why you are focusing on this

combination initially, what the differences are, and what the future plans are for client-side support?
It's a combination of factors. The fact that we could share code with some of our other products and so speed up delivery was part of the decision to focus on Windows 32. These products have just in the last few months started porting to other platforms. And right now, outside of Win 32, there are not a lot of other platforms that have browsers that can do what we need them to do.

Even with Netscape, there are issues because their implementations for a whole variety of platforms are not identical. For example, Netscape on OS/2 is significantly different from Netscape on Win 32. So you can't just say "I support Netscape," and that covers everything.

Additionally, current versions of Netscape don't have DOM Level 2 support. DOM Level 2 is one of the ways we're able to do some of the more innovative things within our product, so that added a twist to development. For a lot of strategic and tactical reasons, it wouldn't make sense for us to create a Web client that only supports IE. But it took extra development work to make sure iNotes Web Access would work on Netscape 4.7, although without the DOM Level 2 support, the implementation has to be slightly different. Some of the functionality has to be rendered differently.

Maybe a better way to say it would be that iNotes Web Access is optimized for IE and supports Netscape 4.7. Not to give short treatment to Netscape—it's just from a strict technical viewpoint, they are not currently as robust as IE is, so right now, there's a bit of a gap in functionality. We're looking forward to Netscape 6.0, which will provide DOM Level 2 support.

And for both Netscape and IE, we're watching their platform support carefully. From what I understand, Netscape 6.0 will be only Win 32, Mac, and Linux, which is down considerably from the 17 platforms they supported in the 4.x code. With IE, they've got Win 32 and Mac. And we're on the sidelines watching carefully the activities of browser development for those platforms that are being left to the open source community, like varieties of UNIX or OS/2. Obviously, if a platform doesn't have a viable browser on it for us to do the types of things we want to do, we can't support it.

What server platforms are supported?
We're going to be dealing with all the platforms that Domino is on. For the initial release, we're going to have NT with Service Pack 4 and Windows 2000. There'll be a Solaris version, and—if not on the day of FCS, shortly thereafter—there'll be an AS/400 version as well. And with subsequent releases, again, we're looking to expand our server-side support as well.

How long is it going to take to get to the rest of the platforms, once you get the first release out?
Our current plan is to support the rest of the server platforms with the second release of the product, which should coincide with the next major release of Domino.

Let's move on to some more details about the design goals. You've covered—broadbrush—the general goal you had, but let's take a step down to a little more detail about your other goals.
Again I think performance was number one. And we started to look at what were some of the powerful, popular services of the Notes client that we could deliver to browsers, especially mail and all the PIM functionality. Obviously, the ability to work off-line was a big thing to us; we wanted to get that in.

Client interoperability was important, so that within the same file a user could use both Notes and iNotes Web Access, and among users using either client there could be that interoperability. For example, take a Notes document link. A browser doesn't know a document link from Adam, so we need to take those links and convert them automatically. Or say someone creates a table

in the Notes client. Although you can't currently create a table with the rich text editor in iNotes Web Access, the user should at least be able to read and edit the table, and the Notes user should be able to see the changes, in fidelity without data being lost.

We also looked at making the migration easy. And we wanted to include support for Sametime and some of the other services we had available. We also looked at making sure we have a nice welcome page and that the product in and of itself was "portal friendly," so that you could incorporate its components within any enterprise portal based on any product.

In today's world, so many corporations are global that we knew we had to make sure we had full international language support. And of course, security was also a very big point—not only in terms of making sure transmission of data is secure but also, since it's quite likely this could be a kiosk situation, you need the ability to have a secure log-off, so that you don't need to be concerned if you're looking at your mail and you log off, that the next person who steps up can take a couple clicks and be in your Inbox.

How does iNotes Web Access measure up in terms of performance?
When you start to compare server performance between a Notes client—a robust client with a lot of functionality—and a browser client—a thin client with virtually no functionality—you can't expect them to be at all the same. It's almost apples to oranges, because you're going to an architecture that is really reliant on server-based functionality. The burden of the application now falls primarily on the server. And so that's why it would be unrealistic at this point to expect that you could get a one-to-one ratio between the two scenarios.

With iNotes Web Access, client performance and server scalability are key areas of focus. When you do an apples to apples comparison between two Web clients, such as iNotes Web Access and our Webmail template, I think you'll be happy with the findings. From the very beginning of our development effort, we targeted a performance improvement over our current Webmail template, and by the time we ship iNotes Web Access, that's something our customers can count on receiving.

From your perspective, what are some of the most interesting or innovative things that you've accomplished with iNotes Web Access?
I think some of our design approaches have been pretty cool. We're leveraging things like data islands, which is an XML tag, that allows us to bring more data locally to the client, and so we're reducing the amount of information that's being transmitted between the server and the client.

We've done some very cool things with the creation of a virtual list box, and delivered some enhance components like dynamic outline control. We've used cascading style sheets to render the design of your calendar and then populate the data in there, and we're doing the data feed in such a way that if you display the month of March, we're only going to feed the month of data that you're going to display. And calendar printing is great; we have all the calendar printing options of the Notes 5.0.4 Notes client—Day Runner, Day Timer, Tri-fold, and so on.

Overall, I think the user interface and level of functionality and ability to take these things off-line are just going to blow everyone out of the water.

How will iNotes Web Access be sold and when will it be available?
It will come as part of a Domino release—core functionality—and it's also covered by a Domino CAL [client access license]. And anybody else looking for it can buy it separately as a Web-based CAL. As for availability, we showed it for the first time in September at Lotusphere Berlin, and launched the beta in late October. We're targeting the ship date for early Q1, 2001.
[Editor's note: iNotes Web Access was released in June, 2001, as part of

Notes/Domino release 5.0.8.]

Vinod Seraphin

Jason has already given us an idea about what iNotes Web Access is in general. Can you give us an overview of what iNotes Web Access is from a technological viewpoint?
It's a product that exploits the Dynamic HTML [DHTML] capabilities of Internet Explorer to provide a very usable and enjoyable Web mail and information management experience within a browser without resorting to Java applets or ActiveX controls for its sophisticated UI [user interface]. It leverages and builds on the Domino server extensions added by QuickPlace to support more complex Web page generation. And it uses HTML elements to build its UI.

What were the goals of this project in general and specifically, what were the technological goals?
We wanted to exploit the rich HTML and XML Document Object Model within Internet Explorer 5 to build a very visually appealing and usable Web application that might be used together with the Notes client—perhaps only when roaming or traveling—or at the other extreme, might be the only client for those users looking exclusively for rich mail and PIM capabilities on the Web.

We didn't want to be constrained by the current Notes client user interface, but to bring a fresh new Domino mail and C&S experience to the Web, incorporating the rich calendar views, visual appeal, and the ease of use of Lotus Organizer. And we wanted to innovate further in this area.

We also wanted to address the current shortcomings of our existing Notes Webmail template, by delivering a superior and more enjoyable end-user experience; and it was important to be fully compatible with the existing Domino mail schema and allow Notes client access using this template.

Additionally, we wanted to leverage the work done by the QuickPlace group to deliver this application to the market quickly. And of course we wanted to leverage Domino Off-Line Services (DOLS) to allow this application to run

off-line.

Tell us about the development history a little? What has been your role?
I was with the Lotus Organizer development team since Lotus first acquired it in early 1992. A couple of years back, I led an effort called Lotus Organizer Web Calendar that piqued my interest in Web development. During that time, Microsoft was touting the rich capabilities they were putting into IE, and I was very interested in exploring this to see if it really was possible to build a "rich" Windows-application-like PIM using these technologies. In other words, I wanted to kick the tires of this new development environment to see what really could be done.

But it was only after Organizer 5.0, where I was lead architect, and then helping out on the Notes client's editor team until after R5 shipped that I finally had time to prototype a Web PIM that exploited leading-edge browser technologies. From the start, it used Domino as the Web server. And I soon found out that the R5 mail applets were retrieving data from the Domino server in XML format. This was exposed to all developers in 5.0.2, and it became another cornerstone for my prototype. IE 5 has an XML HTML element (known as an XML island) that let me do what the applets were doing with DHTML.

As I took the prototype around to various development teams to try to get a real development effort underway, I found a lot of interest at Iris. Steve Beckhardt and Mussie Shore were particularly interested in building such a rich application within a browser that implemented mail and PIM features. They were very impressed with what I was able to accomplish within the prototype and asked me to come over to Iris to do this project. The early team was comprised of a mixture of developers from the Lotus Organizer team and from Iris.

That's the history. My official role on the product has been lead architect. I had a big role in building the team and preaching the "vision." Our development team is still a fairly small team, but one which is doing a super job in this space.

What was it like to set out to build a kind of product that hadn't been built before?
It was definitely exciting and very challenging. It also requires a fair amount of thinking "outside the box" and persistence to get over roadblocks. The biggest lesson I've learned is that a good prototype is critical when you're trying to do something radically different. It's important to not only help figure out how to best solve certain technical problems, but also to reinforce that your vision is really attainable. There's the old expression, "a picture is worth a thousand words." Well, a "working prototype" is essential to communicate your vision to others. It then takes a great team of engineers and a lot of hard work and dedication to actually bring it to market in a timely manner.

What were some of the architectural challenges you faced and how did you address them?
One challenge was putting together a development environment that would allow many developers to work on the product at the same time. iNotes Web Access has quite a broad set of features and so needs a team of developers to work on the single "template" at the same time. Doing this from within Domino Designer would have made the template a bottleneck.

Another challenge was to allow writing maintainable code that didn't incur a performance penalty. Some of the keys to writing maintainable code are to use descriptive identifiers and to comment the code well. However, since JavaScript is an interpreted language, both of these characteristics of maintainable code incurs a larger download and execution penalty.

In both these cases, it turns out that the QuickPlace team had encountered

these same challenges and come up with the tools to overcome them. When I found this out, we were able to take their tools and enhance them a bit to meet our needs. To solve the first challenge, we use text-based files that describe the contents of our template. This allows us to use traditional development tools to write and maintain all our client-side JavaScript code. For the second challenge, we use an obfuscator to strip all the comments and remap large identifiers to very minimal character names. This is critical to get client-side performance and helps reduce the load on the server as well.

Let's talk about the technologies being used in a little more detail. Were there considerations in relating iNotes Web Access to our existing technologies that affected how you designed and developed it?
Definitely. As I mentioned earlier, we leveraged the tools the QuickPlace team first put together, and extended the server in a manner very consistent with what they had done. Also, the XML work by the Domino server team is an integral part of how we retrieve data from the server within our views.

How does iNotes Web Access relate to Domino/Notes technology? How do they work together?
iNotes Web Access consists of both client-side code—HTML pages and lots of JavaScript code that is downloaded to the browser as it's needed—and server-side code within the Domino Web server, which pieces together the many subforms and handles the HTTP requests triggered by various user actions. All the client-side code is stored within forms and subforms, which are located within a single database that is shared by all users on the same server. So iNotes Web Access is now an integral part of the Domino server and was built using existing Domino technology.

You mentioned Domino Off-Line Services (DOLS). How is DOLS technology being used?
DOLS is an integral part of the product. With most Web-based applications, you are limited to using them only when you are on-line. DOLS allows iNotes Web Access to be fully usable when off-line. There are very few things, such as Group calendars, that will not be available to you when you are off-line. You'll even be able to send mail, but the message will not be delivered until the next time you go on-line or use the DOLS client application (Lotus iNotes Sync Manager) to synchronize.

And your use of Dynamic HTML?
Dynamic HTML is actually a term that encompasses several browser technologies. It encompasses HTML, XML, CSS (Cascading Style Sheets), JavaScript, the HTML and XML DOM (Document Object Model) and the browser event model. These technologies together allow you to build very “dynamic” Web pages. All the pages in iNotes Web Access make extensive use of these technologies.

Are there really no ActiveX components or Java applets being used within the iNotes UI?
Almost. Besides the core ActiveX components, such as the rich text editor, which are already part of IE 5, the only additional ActiveX component we use is our custom upload control, which allows a much more Windows-like experience manipulating attachments. If you opt to go off-line with DOLS, DOLS uses a small ActiveX component as well to communicate the off-line state and interact with the Lotus iNotes Sync Manager.

If you're used to Windows 32 applications, you’re used to a rich UI. Typically browsers provide a much less robust UI to work with. What were your goals in terms of the UI and how did you create a rich UI for iNotes Web Access?
My goal with iNotes Web Access was to give users the experience they would expect from a well-written Windows application. The goal was never to be constrained to say what typical Web pages look like. When you develop a

Windows application UI, you have a set of standard controls that you can piece together to build various screens. When you consider implementing a rich UI within a browser, the equivalent to standard controls are HTML elements—such as the DIV, SPAN, TABLE, and INPUT elements. With earlier versions of Windows, the set of standard controls were much fewer than what is available today. The better Windows applications, in order to provide a better experience for users, built a library of “custom controls” using standard Windows APIs and the existing base set of controls. With iNotes Web Access, we have done exactly this. We have built a whole set of “custom components” by piecing together HTML elements and programming them with JavaScript and the HTML DOM. We are then able to use these in various places in our UI to make it much more usable.

So you start with the base HTML elements and then build larger more interesting components and screens with these. Windows programs employ an event driven programming model and so do browser pages. With Internet Explorer, it's also possible to influence the tab order between input elements, and add keyboard mnemonics and accelerators. Many of these improvements are within core HTML 4.0, which means in the future this will become a standard supported by all future browsers. In fact, we expect future browsers providing more rich capabilities and Web sites to provide more robust interfaces as a result.

Any details or examples of how you coped with the challenges the browsers presented?
We found that trying to build a UI on Netscape that parallels what we have done in Internet Explorer is very difficult. There are many HTML layout differences between Navigator and Explorer. Navigator also has performance issues with certain HTML constructs such as nested tables. Further, Navigator doesn’t support XML islands, or the ability to add “onclick” handlers to any HTML element. So, for Navigator, we use a more server-intensive approach to build the view HTML and provide more “traditional” HTML mechanisms for selecting entries in a view.

The only way we found we could support Navigator without compromising our goals was to build a very different set of pages for Navigator. To provide an optimal experience on Netscape 4.7, we actually exploit the LAYER tag in specific situations as well.

How did you make the trade-offs when forced to choose between implementing a feature as it is done in the Notes client, versus implementing it in perhaps a new way?
We would take a close look at how the UI is implemented in Notes, Organizer, Windows, and a few competitive products; and then we analyzed what we liked and disliked about the various approaches. We’d keep in mind that the Notes UI in the past has been limited to what is possible within the construction capabilities offered in Domino Designer. Since we are exclusively using pass-thru HTML, we are not constrained in this manner. We are able to take advantage of whatever is available within Internet Explorer 5. We gave our UI designer a lot of freedom to innovate in this area, with the constraint that we had to maintain a compatible NSF storage schema. Certain UI decisions are also based on various performance or scalability implications. For example, we don’t automatically display everyone’s freetime when you click on the Schedule tab. We require an explicit click on the Find Freetime button.

Our team is composed of a mixture of people who have used Notes for many years and those who are very new to using the Notes client. Some of the liveliest debates within our team have been on various UI issues. Our overriding decision maker is usability. I strongly believe we have made decisions that make our interface more usable. We’ve also conducted several usability tests from which we’ve garnered good feedback and we plan to do even more testing down the road.

Over time, iNotes Web Access will replace the existing Webmail template. How is iNotes Web Access different from the current Webmail template?
Our template is actually built using the existing Mail50Ex.ntf as the starting point. This is how we are able to offer support for the traditional Notes client as well. We add certain new iNotes Web Access specific design elements to this template to support our UI. However, almost all of our design elements reside in a shared forms database that is pointed to by our mail template.

As far as features we support with our template in comparison to what the present Webmail template supports, the key additional features we are supporting today (or planning to support prior to ship) include:

- Unread marks
- Rich view management UI
- Jump to (or Quick Find) within a view
- Spell checking messages
- Helper controls to select dates, times, and to set durations
- Much improved support for attachments, including unlimited number of attachments and drag-and-drop files into the attachment control
- Very rich and usable calendar views, including double-click a date to create an entry
- Group calendar features
- Rich contact management UI
- Alarm notification
- Rich calendar printing that generates PDF files
- A Notebook capability (Notes Journal equivalent)
- Mail preferences that let you specify whether new mail will be on top or on the bottom and a check for new mail button
- Dynamic resizing of windows
- An out-of-office auto reply feature and work-hour/off-hour differentiation
- A Gantt-like view for tasks
- Customizable Welcome page
- Context sensitive help
- Secure log off

Most all of these were on the list of the most requested features that were *not* in Webmail, so we know they're important to users.

Are there any features in the current Webmail template that iNotes Web Access doesn't provide?
The only one that comes to mind is the support for managing mail rules. Any existing server-based rules that you may presently have in your mail file will still work, but our initial release will not provide a way to create new rules or edit existing ones. Why? We don't believe this is a feature that the majority of users take advantage of, so it wasn't as high on our priority list for features we wanted to make sure we had in our first release. However, the Out of Office agent was considered very important and is something that will be part of the initial product.

[Editor's note: For more details, you can check out the ***iNotes Web Access/Webmail feature comparison****.]*

What were the most difficult challenges in developing iNotes Web Access?
A few things come to mind. The first is our virtual list (or view) component. It is our most complex client-side component. Views are a big part of Notes, and bringing rich view management to a browser is a very unique feature we have. Traditional Web pages break up large lists into multiple pages. This new view component allows the user to manipulate all the documents in the view from the same page. Portions of the view are retrieved from the server in an XML format and incorporated into the virtual list as needed. You can even adjust the column width by dragging the column border. This is really cool.

Lastly, a big part of being a good PIM is to support rich calendar printouts. I've always been disappointed with the printing capabilities within a browser. PDF files have become a *de facto* standard for superior quality printing on the Web. The IRS [United States Internal Revenue Service] uses this format to make tax forms available to the public and just about every company uses this format to provide rich, printable forms for customers to fill out. Generating such form-quality printouts within HTML is just not feasible today. Hence, we have moved the rich calendar printing code in the Notes client over to the server and modified it to generate PDF files. This brings the richest possible calendar print formats to iNotes Web Access users.

From your perspective, what are some of the coolest technical achievements in iNotes Web Access?
I think there are a lot of cool features in the product. The view component and PDF printouts I just mentioned would both qualify. However, another really cool feature are our calendar views. As you traverse through dates using the next or previous arrows in the view header or the date navigator over on the left side of the page, the entire page isn't reloaded; rather the calendar is updated entirely by client-side code and only the data that belongs in the range of dates displayed is retrieved from the server. The month view also limits each calendar entry to a single line and adds ellipsis to indicate there is more information to be displayed. Resizing the window also readjusts the default dimensions of each cell.

Other cool features are multiple-selection and right-click menus within the view component, our tab control, which displays the different tab panels without a server round-trip. The ability to spell check mail messages is really cool as well. A lot of these features are pretty mundane and nothing out of the ordinary to most Windows application users, but the fact that these are available from a Web page makes it unique.

Let's talk about two critical areas: performance and security. What are your goals for performance? How did performance play into the development of the product?
There are really two aspects to performance—client-side performance and server-side scalability. Of course, we are striving to be the best we can be in both these areas. But there are trade-offs. The user experience is the primary goal, but there can be performance costs for a rich UI. Servers and clients will continue to get faster at a dizzying pace. A good UI already taking advantage of cutting-edge technology and poised to take even more advantage of technology on the horizon is the proper investment to make.

That said, we are quite conscious of both client-side performance and server-side scalability, and I can talk a little about what we are doing in both these areas. There are several things that impact client performance, including paying attention to things like:

- The number of total bytes downloaded from the server to the browser for the page
- The complexity of server-side formulas and logic to generate the page
- The total number of files that comprise the page
- The HTML and client-side logic needed to generate the page

To boost client performance, we strive to separate our "code" from the "data" as best we can. In this manner, most of our code is broken up into external script files, which are downloaded once and then very efficiently cached within the browser's temporary Internet files folder. Our JavaScript code is also obfuscated as I mentioned earlier to minimize page size and boost performance. We also make extensive use of cascading style sheets to reduce the size of the HTML and minimize the sizes of bitmaps on the page. We also try to minimize the need to reload the entire page in many situations, often just getting the new data necessary and building the HTML with client-side code.

Factors impacting server-side scalability include things like:

- The amount of disk access required to generate a page
- The amount of server-side computation required to generate a page
- The amount of contention among server threads processing separate requests
- The number of requests received from each user

To boost server-side scalability, we employ lots of server-side caching of forms and subforms that are shared across all users on the same server. Traditional Domino Web applications have form and subforms in each and every end user database that are not cachable across multiple users. Further, many of our static external files within a Web page are set up to be very efficiently cached by the browser. This should reduce the amount of requests to the server as a user continues to use the application. Lastly, we will be employing key caching of certain data constructs to minimize the computation required to generate subsequent pages and boost both client and server-side performance.

How did you maintain performance while achieving your other goals?
As I said, it's a trade-off and it’s very tough sometimes. Often we implement certain features such as updating server-side unread marks or auto refreshing a view after manipulating a document in the view, and initially we might not consider how it might be done in a more efficient way. The users beta testing our product should know that there is much room for improvement here and in several other areas before we will call this first release done.

Using multiple windows within the UI is not only for usability reasons, but also benefits performance. We will also strategically use hidden frames to make some server-side change without having to totally regenerate our UI from the server. There are several client-side “tricks” we use to try to boost performance and scalability.

There are also certain DOM manipulation techniques that are more efficient than others. We have tried hard to generate DHTML in as efficient a mechanism as possible. Frankly, we have thus far spent most of our energy implementing lots of functionality. We are now about to buckle down and really concentrate on fine-tuning the application for performance and scalability.

What about security? What were the security issues?
The Domino Web server already has a suite of security options. We recommend that you employ at least session-level security within your Web mail servers, but we don't require it. If you opt to use basic authentication, you should know that the browser window caches the username and password, so subsequent access to the same mail file from the same window will not result in an authentication challenge. We offer a log off action that will actually log off the server session as well as try to close the browser window. Internet Explorer further has a security related preference to “Empty Temporary Internet Files folder when browser is closed.” This setting in conjunction with closing the browser windows provides a little more security.

Like Webmail, we are also unable to read encrypted mail from the browser as we would need access to your private key, which is presently stored in your Notes ID file. Look for us to do more in this area in the future.

Looking beyond the first release, can you give us a sense of where the product is heading?
We’ll continue to add more features. We’ll probably add support for managing mail rules, editing calendar entries directly within calendar views, and various drag-and-drop capabilities throughout the product. We’ll also look towards exploiting newer Web technologies, and of course we'll support more platforms as well as the next generation of browsers.

About Jason
Jason Dumont is the Director of Client Strategy & the iNotes Client within Lotus Product Management. In his previous role he was the Senior Manager of the ISV Development Team and was responsible for the strategy and direction of Notes and Domino as a platform for third-party developers. Jason has been with Lotus since 1991, where his first job consisted of sharing a cube with another worker while they stuffed form letters into envelopes. Glamorous, huh? Jason has a great deal of experience with the Notes and Domino application development environment and has been responsible for the creation of the R4 and R5 versions of the Notes and Domino **Application Development Best Practices Guide** and **Inside Notes**, documentation about Notes/Domino architecture.

A frequent speaker at business and technical conferences around the world, Jason holds BS degrees in English and Philosophy from the University of Connecticut. He also spent the better portion of the late 80s and early 90s playing drums in a number of professional touring rock bands. More recently he has been enjoying the return of the NFL season with his wife, Gwen.

About Vinod
Vinod Seraphin has spent the last year as a Software Architect within the Web Applications team at Iris Associates. He was previously with Lotus for eight years, where most recently, he was the Software Architect for Lotus Organizer. He has had key development roles within the Lotus Organizer development team since 1992. A portion of his final year at Lotus was actually spent at Iris, where he was on loan to the Notes client editor team to help fix ship-critical bugs for R5. Prior to Lotus, Vinod worked on a spreadsheet product at Access Technology and office automation products at Data General.

Vinod has an MS in Computer Information Systems from Boston University and a BS in Computer Science and Engineering from MIT. He also is an avid professional sports fan and enjoys softball, skiing, traveling, and spending quality time with his daughters, Audrey and Maddie.