Interview: Jeff Eisen

Interview by Betsy Kosheff

[Editor's note: This interview resides in "Notes Today", the technical Webzine located on the http://www.notes.net Web site produced by Iris Associates, the developers of Domino/Notes.]

If you liked the Notes 4.5 client's support for Java applets, you can thank Jeff Eisen. In addition to leading this effort at Iris, now he's working on the implementation of native HTML support in Notes. Learn more about these topics and the next steps for Java applications and Notes.

Why should I care about Notes and Java?
Well, several reasons. As of Notes 4.5, you can execute Java applets from directly within the Notes client environment; you can cut and paste Java applets into Notes mail messages for secure distribution; you can grant additional security rights to your Java applets by using the new Notes trusted host and trusted proxy features; and later on, in upcoming versions of Notes, you will be able to let your Java applications take advantage, including remotely accessing, of the Notes object store, security and other object-based application services.

Which Java efforts are you working on?
Initially, I've been focused on supporting Java on the Notes front end, embedding Java applets into the Notes client. So, for example, if you come across a Web page that has a Java applet in it, the Notes browser that is part of Notes 4.5 can now deal with it. That involved doing the work needed for our browser and Notes documents to understand Java applets, so that if you have an applet in a Web page and you copy it and paste it into a mail message, Notes knows what to do with it.

Going forward, we are working on improving the front-end support of Java, including using newer and better versions of Java, such as Java 1.1. More importantly, perhaps, we are working on providing Java access to the Notes back end -- that is, we will be supporting agents written in Java in addition to our current LotusScript support.

Jeff focuses on making sure that Domino/Notes handles Java in a secure way.

But what about your customer's current investment in agents written in LotusScript?
We are committed to continuing and improving LotusScript support in Notes. However, we realize that for many of our customers, particularly those with significant C, C++, or Java programming resources (because Java is pretty similar to C++), Java may be a more comfortable language to use than LotusScript. For that reason, we are providing a choice.

What do you think of the state of Java today?

Most of the Java you see out on the Web today is sort of toy applications. The real applications for intranets and the Internet are coming. But first, we need better security to work from; we need much more granularity in allowing more operations, and not allowing operations, as the case may be. It's a real trade-off to give users as much power as you can, but also make it so that relatively naive users can't accidentally pick some really bad options that could lead to a security breach.

Isn't Java a secure language?
That's sort of an overstatement. Java security is only as good as the embedding environment makes it. It's a language in which the embedding application -- Notes or Netscape Navigator, or whatever -- controls the security. So, if a Java applet tries to access a file on a machine and it's deemed to be an insecure operation, Java asks the embedding application, "Can I do this? " Perhaps we say no, or further prompt the user. This is possible because we've installed the Java security manager, which is one of the Java classes. It's called Java lang.SecurityManager, and it controls everything the applet can do, up to a point. For example, applets can't access files on your local hard disk. Beyond a certain set of things you're not permitted to do, however, it's then up to the browser or application developer for the embedding application to determine what to allow or disallow.

What would be an example of a security function Notes could allow, beyond what Java customarily allows?
In Notes 4.5, there's a new security model that allows trusted hosts and trusted proxies. We did this because Notes is popular in intranet environments, where applets often need to have more rights than those written to run over the Internet.

So is this a supplement to the Java security?
Yes. One of the security rules that Java has in general is that, if an applet comes from a certain host, it can only have network communications with the same host. This is to avoid obvious problems like applets coming in from outside the firewall poking around on different machines and shipping proprietary corporate data outside. What we did was introduce the concept of a trusted host, so we allow hosts to talk to each other, if desired. So, you can explicitly indicate right in a Notes location record your trusted hosts, either by listing them or using a wildcard. This is particularly useful in an intranet environment, where you commonly need to have a set of trusted hosts so you can enable applets to grab information off one machine and ship it to another.

Is this a long-term solution?
It's hard to say. When we support digital signatures for Java in Notes, you will be better able to control the security of Java applets. However, in an intranet environment, it still may prove easier to use and manage a trusted hosts lists than to deal with signing all your applets.

How would you define a trusted proxy?
It's provided as an additional value field in the Notes location record so that you can indicate a proxy known to be trustworthy. This makes it easier to surf the net from within a firewall.

For example, as applets come in from the Web, it's sometimes hard to figure out what is the originating host. Let's say I'm inside the firewall trying to read a page on Evil-Machine.com. I can't directly talk to that machine because I'm inside the firewall, so I have to go through a proxy server. Notes tells the proxy, "give me a connection to Evil-Machine.com." But let's say it's also an evil, untrustworthy proxy, so the next time I say "give me a connection to Evil-Machine.com" it could give me a connection to a different machine. I can't really tell who it's giving me a connection to; it could be two different machines if it's not a well-behaved, trusted proxy. From a Java user perspective, I think they're the same machine and, yet, they 're going to communicate with each other and pass information, which, of course, Java tries to forbid. In Netscape Navigator, you just get a security exception. In Notes, in the Java applet security section of the location record, you can have the ability to check, yes, I do trust the proxy, so I know it's not going to give me different hosts each time I use the same name.

If you don't trust the proxy, you can play it safe by using network IP addresses rather than names. That way it is tougher (though still not impossible) for the proxy to spoof you. This is necessary when using Netscape Navigator where you can't specify that you trust the proxy.

So, trusted hosts are good reasons to use the Notes browser if you're building an intranet?
Right.

Having worked with LotusScript, how do you find Java as an embedded language?
Java is pretty good for people who are writing Java programs, especially self-contained applets, but because it's cross-platform, it's somewhat of a lowest common denominator approach-- it can't take advantage of cool GUI things on the Mac, for example. But I'm coming from the other side, the business of embedding Java, and it's not as far along as LotusScript in that regard. Java is really in its infancy. It's gotten somewhat better with the recently released Java 1.1, but it still has a long way to go. Of course, LotusScript was designed from the ground up to be an embedded language. It has a very definite API for people who have a product and want to put LotusScript into their product. Don't get me wrong -- for people focused on applets it's just great, but it just hasn't been so well thought out from an application developers' perspective. The result is that you tend to do a lot of not-so-clean things just to get it embedded. I've talked to other people who have embedded Java into other products, both GUI and non-GUI products, and we compare war stories all the time.

What kind of Java war stories?
Well, you can't just exit Java. When Java is exiting, it thinks it owns the world. It calls the C function "exit," and then it shuts down your application. Notes doesn't like that, so at first, we tried not exiting, just sort of killing it out from under it. This works on some operating systems, but not others. In Windows 95, for example, it causes your systems resources to not get freed. If we didn't work around this problem, if you started up and shut down Notes a few times, you'd have to reboot Windows 95.

There are lots of things that are very different on each of the Java-supported platforms, and you just need to learn the gotchas. Like how you parent a Java AWT, which is their windowing toolkit for having a Java window inside a non-Java window. It's a totally different experience on each of the platforms.

What advice do you have for Sun that will make Java better from an application developer's perspective?
Well, many of my complaints from working with Java 1.0.2 have been addressed in Java 1.1. But, I still think they have a ways to go. Clean shutdowns are still difficult, for example. Also, I don't think support for minimal configuration machines (such as machines with only standard VGA displays) is very good. We are still having to workaround limitations such as these.

On which platforms will Notes support Java?
Win 32, which is NT and 95, plus NT Alpha, OS/2, and various UNIX flavors -- Solaris, both SPARC and Intel, AIX and HP.

What's the next step for Java applications and Notes?
At this stage, it's hard to write a Notes application in Java because Notes apps tend to be workflow-based and make heavy use of LotusScript. We support Java as an applet, but soon you'll have the ability to do things like tell Notes to open a database and get this mail and forward it.

Cool. How does this work?
You will be able to write Notes agents in Java, similarly to the way you write them today in LotusScript. The Notes object hierarchy will be as close to the current LotusScript hierarchy (NotesDatabase, NotesSession, etc.) as is practical so it should be a pretty easy transition for current LotusScript Notes programmers.

In the Notes Release 5 timeframe, there will probably not be a GUI support in Notes for developing these agents (such as a Java mode in the programmer's pane). So, the agents will have to be developed externally to Notes. However, we are aggressively working on GUI integration for future versions of Notes.

How will you integrate Notes security with Java in the future?
In the new version of Java coming out, there is more security as part of the Java spec, for example, there are digital signatures of Java applets. We don't currently support this, but when Java has it, we will, and we'll offer some integrated UI with the general Notes security UI.

What other projects are you working on?
I'm also working on the conversion needed to store Notes documents as native HTML documents, rather than the Notes-only format, so that they can live on any server. HTML and MIME aren't as rich description languages as Notes, so they won't support all the features Notes can. And HTML isn't dynamic, yet anyway, while Notes excels at handling dynamic information.

But we want to do this so we can de-couple the product better to enable people to buy Notes client and server technology on an a la carte basis.

Will native HTML support improve speed?
For some things, but probably not for others. It should speed up the browsing of Web pages, for example, because it will cut down on time spent on format conversions. However, for normal editing, saving, and loading documents, the speed will probably be about the same.

Speed is not the primary reason why we are working on native HTML support in Notes. Priorities such as better fidelity in conversions and better WYSIWYG for Internet communication where HTML is the standard is driving this effort. Native HTML support will also make it easier to decouple the Notes client and server so a non-Notes based e-mail client will be able to use the Notes server and vice versa (Notes client with non-Notes server).

Will you support dynamic HTML?
We are certainly looking into it for the future.

What about JavaScript?
Well, personally, I'm not a big fan of JavaScript. There's a lot of confusion out there about the differences between Java and JavaScript In reality, they are totally unrelated except for their names. JavaScript is a scripting language for HTML originating from Netscape with C or Java-like syntax and with some object-oriented features. Microsoft is instead pushing a scripting language with a Visual Basic syntax. It is still unclear what HTML scripting language will win in the marketplace. I suppose I'm rooting for Microsoft here because I think a BASIC language, such as Visual Basic or LotusScript, makes a pretty good language for scripting that can be used by the average HTML author.

Microsoft and Netscape have both pledged support for basic Web standards like HTML and Java and seem to be extending these in ways that will give them competitive advantage. Will Iris do the same? What are you doing to help evolve HTML and Java?
Our efforts are actually a much bigger effort than just the team at Iris. With the help and power of our parent companies, both Lotus and IBM, we are playing a significant role in the evolving standards for HTML and Java. For example, we have played a lead role in the JavaBeans specifications, an "object model" for Java.

BIOGRAPHY
Jeff Eisen joined Iris in July 1995 to lead the effort to embed Java applets into the Notes client. Previously, he worked at Lotus, where he focused on the platform-dependent layer of LotusScript. In addition to being the resident LotusScript expert at Iris, he is currently implementing native HTML support into the Notes client.