From: Yuval Neeman
Sent: Wednesday, October 02, 1996 11:00 AM
To: Developer Tools Strategy; Peter Kukol; Patrick Dussud; Brad Lovering; David Stutz; Victor Stone; Ben Slivka; Adam Bosworth; Bob Muglia
Subject: RE: Think Week (Long!)

Let me show my ignorance,
I downlaod a Java (or applet) and the app offers me the the ability to change the font and background color of the UI. I expect subsequent usage of the app to have use these defaults.
Aslo to use Adam's example I have some data access app, initialy i need to tell it what database type I want use (Oracle) and the specific database, after I use it for a while guess what, there is a new oracle driver (it now actualy works...) and I want to tell my app to use the new - specific driver.

Where is this information stored ?
- In code in a class file on my local drive
- On the server in some user personalization gizmo

It has to be persisted somewhere in the file system ir in some configuration database. So where is the magic. How is uninstall or upgrade simpler ?

Yuval.

From: Bob Muglia
Sent: Tuesday, October 01, 1996 9:56 PM
To: Developer Tools Strategy; Peter Kukol; Patrick Dussud; Brad Lovering; David Stutz; Victor Stone; Ben Slivka; Adam Bosworth
Subject: FW: Think Week (Long!)

fyi...

bob

From: Aaron Contorer
Sent: Monday, September 30, 1996 12:37 PM
To: Bob Muglia
Subject: FW: Think Week (Long!)

It is my understanding that this issue falls entirely under your group, but please let me know if there is any way I could help, or any people I should talk to to better understand this for Bill.

—Original Message—
From: Bill Gates
Sent: Monday 30 September 1996 11:25 AM
To: Aaron Contorer
Subject: FW: Think Week (Long!)

This scares the hell out of me.

Its still very unclear to me what our OS will offer to Java client applications code that will make them unique enough to preserve our market position.

Understanding this is so important that it deserves top priority.

ATTORNEYS ONLY

—Original Message—
From: Adam Bosworth
Sent: Sunday, September 29, 1996 2:09 PM
To: Aaron Contorer
Cc: Bill Gates; Gary Burd; Brad Silverberg; John Ludwig; Ben Slivka; Bob Muglia
Subject: Think Week (Long!)

MSS 0065262
CONFIDENTIAL

I think it is important to understand that Java is not just a language. If it were just a language, it would not be a threat to us. We would and could easily just build the best implementation of this language and be done. It is, however, much more. It is an alternative to COM. Now I realize that this is, even for me, a provocative statement. I'm going to

try to back it up. First I'll use some examples and then discuss the issue in a slightly more wide-ranging fashion.

Examples:
When listening to the Denali folks explain to various Web-masters why it doesn't run on Unix, I hear the statement that COM isn't on Unix. Ignoring whether this is or isn't true, Java is on Unix and requires no dealing with setup, install, de-install, or anything else. Thus it is really easy to understand how a system for dynamically authoring Web pages on the server that depended upon Java objects rather than COM ones would have wider appeal. In Trident I'm asked how the local Dataset gets brought across the wire. I answer dataSource controls and can demo this. But, it assumes that some piece of code is permanently available on my machine to do ODBC. I'm asked why I don't just read the data over the wire using JDBC. I answer that I want to and will. Why? Because the JDBC class can be dynamically loaded, executed directly against the wire format coming over the same wire (Java is good at this, the right classes are there), and then discarded. No install. Nothing sitting in my registry file. Works on any platform. So the fact that JDBC can be loaded across the wire essentially makes it a wire level protocol as far as I'm concerned in as much as I never need to keep custom code on my machine that talks to a specific wire protocol. Next, let's consider authoring a calender control. Up to now I've been telling folks that COM works here and Applet's don't because I want to be able in Trident to sink the events of the control like DateChanged fired when the user clicks on the control and COM provides an architecture both for me sinking the events and for a RAD tool discovering what these events are and Applet's don't. This isn't true anymore. With LiveConnect I can simply pass in a script function in my page as a paramteter of the Applet and the Applet can fire it. Or with JavaBeans, I can easily write one extra property in the Applet (EventSink) set my one new method SetHandleEvents, and then when I want to fire the event I just write a single line of code this.EventSink->DateChanged(NewDate). (The syntax may be slightly off). To implement the same think in our OLE Controls, I must allocate a conection point container, add a connection point, add a pointer to the sink to the connection point, marshall the arguments into the form that the Invoke method of the IDispatch interface expects, and then fire it. This is actually quite a lot of hard code which provides no runtime benefit in real life. Thus it is a lot harder and bigger to write than the Java equivalent. Sure we could use the same architecture from C++, but that's the point. This is an architecture, not just a language. Another example. We were told that it would be easy to write controls in Java. It was just a language. Well, considering the importance to us, I'd assume if it were easy we'd be doing it by now. But as of this writing, there isn't even clear agreement about how you will write such a thing because the Java folks are afraid/embarrased to expose the raw plumbing to the Java programmer because it is so much harder than the Borland Javabeans alternatives. So right now we have nothing. I literally cannot write today a calendar control in Java that will fire events to my script in Trident. Netscape is shipping today a browser that can. This cannot stand and, if it does, obviously I'll be forced to support LiveConnect with great speed in Trident because the alternative would be that Netscape would interact better with Java Components than I do.

General Issue:
A key general issue that Gary Burd is better equipped to explain than I am is that Java has a competitive advantage in simplicity of implementation precisely because the architecture isn't language neutral. They have brilliantly used the language to solve problems where we must (in C++) write lots of explicit code to do what essentially amounts to runtime plumbing and this code is hard to write and hard to understand. This shows up in every facet of the language. Indeed, the way to sink events in AWT is still simpler. You subclass the appropriate method of the object. Period. No wiring. JavaBeans has taken this to a new level by adding equivalents to our TypeInfos and our EventBinding ConnectionPoints that are lightyears simpler to implement and will almost certainly run a lot faster because they are essentially not overhead. All of these benefit from the fact that the objects can be dynamically loaded and don't require install/setup and you have only to look at Office to understand what a total nightmare Setup is for us at this point. This single fact, dynamic loading without setup/install/registration conflicts is alone a major asset for Java. The next issue is the classes. Java people have a zeitgeist which mandates that widgets built in Java run anywhere. This may limit them (no direct draw), but at the end of the day they do have access to a bitplane in AWT and many of them have simply written all their controls at that level bypassing all the slowness and inneficiency of the rest of AWT and using the bitplane as a universal portable layer. This works. This is what BOngo from Marimba and the forms that NetScape bought from a spin-off of the Nextstep folks and others are doing. At the same time, AWT 1.1 is about to come out and it is, by all accounts, a lot faster and better than AWT 1.0. We cannot fight this with complex classes that only work on Windows 32 bit platforms alone. We must also be the best provider of these AWT classes and the ones who then quietly offer "extensions" that do realize either the power of Windows (DirectDraw) or of Trident, but in ways that seduce rather than collide. We must acknowledge that Java competes with COM in order to understand what to do about it, not just put our heads in the sand. And we must actually go the last step and make certain that it is really easy to build dynamic extensions to our frameworks whether they are Denali or Trident using Java even if that means doing it differently than we would for C++. This doesn't mean just making it easy to put Java behind a Trident Page to construct a component as Bobmu is planning with Trics. It means also making it easy to build components in Java that either have nothing to do with Trident (they extend Denali) or just extend Trident but don't reuse it like DataSourceControls or sliders or ChartControls or animatedart.

Concerned

Adam Bosworth

From: Aaron Contorer
Sent: Friday, September 27, 1996 12:55 PM
To: Adam Bosworth
Subject: re: memos you are writing

Best is to email them to both me and Bill.

If in time, I will include paper printout in to-read stack, which increases probability Bill will get to them promptly. But even if not in time, Bill says he's quite interested so I'm not worried that he won't read them soon.

I apologize for the quite short notice for this Think Week. I know you are super busy.

-Aaron