IN THE UNITED STATES DISTRICT COURT

FOR THE DISTRICT OF UTAH, CENTRAL DIVISION

NOVELL, INC.,)

Plaintiff,)

vs.) Case No. 2:04-CV-1045 JFM

MICROSOFT CORPORATION,)

Defendant.)

_____)

BEFORE THE HONORABLE J. FREDERICK MOTZ

DATE: NOVEMBER 29, 2011

REPORTER'S TRANSCRIPT OF PROCEEDINGS

JURY TRIAL

REPORTED BY: Patti Walker, CSR, RPR, CP

350 South Main Street, #146, Salt Lake City, Utah 84101

A P P E A R A N C E S

For Plaintiff:

DICKSTEIN SHAPIRO
BY: Paul R. Taskier
Jeffrey M. Johnson
Miriam R. Vishio
1825 Eye Street N.W.
Washington, D.D. 20006

WILLIAMS & CONNOLLY
BY: John E. Schmidtlein
725 Twelfth Street N.W.
Washington, D.C. 20006

SNOW CHRISTENSEN & MARTINEAU
BY: Max D. Wheeler
10 Exchange Place, 11th Floor
Salt Lake City, Utah 84111

NOVELL, INC.
BY: Jim F. Lundberg
1800 South Novell Place, H-544
Provo, Utah 84606

For Defendant:

SULLIVAN & CROMWELL
BY: David B. Tulchin
Steven L. Holley
Sharon L. Nelles
125 Broad Street
New York, New York 10004

MICROSOFT CORPORATION
BY: Steve Aeschbacher
One Microsoft Way
Redmond, Washington 98052

RAY QUINNEY & NEBEKER
BY: James S. Jardine
36 South State Street, #140
Salt Lake City, Utah 84111

I N D E X

SALT LAKE CITY, UTAH; TUESDAY, NOVEMBER 29, 2011; 8:00 A.M.

PROCEEDINGS

THE COURT: Good morning, everybody.

Let's get the jury. It's an amazingly prompt jury.

(Jury present)

THE COURT: You all are going to spoil me. You're all wonderful. You're prompt every morning. I'm going to go back to Baltimore and I'm going to wonder why a juror or two can't be there on time.

Mr. Holley.

MR. HOLLEY: Good morning, Your Honor.

Microsoft calls Robert Muglia to the stand.

ROBERT MUGLIA,

Having been duly sworn, was examined

and testified as follows:

THE CLERK: Please state your full name and spell it for the record.

THE WITNESS: My name is Robert Muglia. M-u-g-l-i-a.

DIRECT EXAMINATION

BY MR. HOLLEY:

Q Good morning, Mr. Muglia.

Could you tell us what your educational background is, starting with graduating from high school?

A Yes. I graduated from high school in Michigan in 1977, and I attended the University of Michigan for four years where I received a bachelor's degree in computer communication science.

Q When did you first start working at the Microsoft Corporation?

A I began at Microsoft in January of 1988.

Q What was your first job at Microsoft?

A I was the first program manager on a database product called SQL server, the first technical person assigned to that product.

Q What does SQL mean?

A SQL stands for structured query language. It is a very commonly used way to perform queries against databases.

Q Now you said that you were a program manager. Can you tell the jury what it means to be a program manager at Microsoft?

A Sure. A program manager is somebody who works with customers and partners, understands what their needs are, and works to develop the technical specifications for a product. Program managers are really responsible for the features and content of any of Microsoft's products.

They also are involved in project management. Typically project management is also done within the program management discipline. So maintaining the schedule and sort

of helping to understand where the product is in terms of its progression through the engineering process.

Program managers don't really define the schedule. That's largely done by the development teams and the test teams. Development teams are the folks that write the code, write the software. Testers validate that it's functional according to the specification. So the program managers kind of watch the project and help it along.

Q Why not just let the software developers do what they want to do on their own?

A Because they wouldn't do the right thing. Software developers, fundamentally their job is to actually do a great architectural design for a product, to understand how to make a product perform as well as it can, and to write the actual code that implements what the product needs to do.

When you are building any product, it's important to understand what's happening in the industry, what your customers need. It's always good to start with the customer, understand what features they want, and also to work more broadly for many products with partners that may have dependency on this product, such as software developers that might build applications on top of an operating system.

Q After being a program manager on the SQL server database product, what did you do next at Microsoft?

A Well, I was -- I was involved in a number of things that were working on server products. We had started to work on a mail product that ultimately became Microsoft Exchange. I did a little bit of work with a server that connected to IBM mainframes.

But my primary job transition that occurred after I left the SQL server group was to run the program management team that was building what we thought would be the next generation operating system called OS/2. This was an operating system we were jointly developing with IBM. And really near the tail end of that project, it had been going on for some number of years, but when I joined that effort, which was roughly 1990 if I'm correct, it was the tail end of the work we were doing there.

Q What happened to Microsoft's involvement with OS/2?

A Well, ultimately OS/2 was not successful in the marketplace. It was -- this was a case where Microsoft and IBM, who had traditionally partnered from the earliest days of the PC, were trying to work together to build this next generation much more powerful operating system called OS/2, but a series of decisions that were made really early in the cycle of the product caused it to not be successful in the market.

You know, one of the major ones, which was a disagreement between Microsoft and IBM, was that IBM was

really focused in the early days on -- and really throughout the OS/2 life cycle to make OS/2 as compatible with mainframe computers as they possibly could. That was, in retrospect, a bad decision because the market viewed PCs very differently than these larger systems that are -- these IBM mainframes. So there were certainly design elements of OS/2 that were very problematic.

The thing that was really difficult, frankly, was having two companies with very different objectives trying to jointly develop something was not a good recipe for success in the technical industry. There are very few cases where that's been successful over time. And the result was the product never worked the way it should work.

It was always a joke that in OS/2 you couldn't print in OS/2, even to the very end getting simple functions like printing to work. So obviously customers wouldn't buy an operating system that didn't do what it was supposed to do, like print.

Q Mr. Muglia, after working on the OS/2 effort, what did you do next at Microsoft?

A I -- after we -- after Microsoft decided to go in separate ways with IBM and focus on Windows as the future of the PC operating system, we had started an effort to and had been working on a next generation operating system that became known as Windows NT. And for a brief period of time

I worked as a staff person for Paul Maritz, who was my boss at the time, helping him to sort out and make some of these strategic decisions.

Following that, I went to work on Windows NT initially in the form of a project that we had launched that was code named Cairo. Ultimately it became really just focused on Windows NT and running the program management group, that group that defines the specifications, doing that for Windows NT.

Q You mentioned something called Cairo, Mr. Muglia, which we've heard a little bit about in the trial. Can you, in a nontechnical way as possible, explain what Cairo was?

A Well, if you put yourself -- I mean it's important to go back in perspective. We're now going back almost 20 years, and in some ways the launch of the idea of Cairo was over 20 years ago. It was a speech that Bill Gates did at a trade show called COMDEX in 1990 where he talked about something that he called information at your fingertips. If you actually go back to that period, you have to recognize that, you know, very few people used PCs. Those that did were generally using character mode operating systems like DOS. So they had to type everything in by hand. There was no real graphical interface -- or at least most people weren't experiencing graphical interfaces. It was in the days before Windows.

So the idea behind Cairo was to use this next generation graphical interface and some ideas like search and some other ideas like being able to browse into the file system to make it very easy to find information. The whole idea was that you have all this information you can store on your PC and it was hard to find it because the computers were very difficult -- they made it difficult for people. And Cairo was an idea that was -- was a project that really focused on making it easier.

Now Cairo was more focused on finding information on your PC. Of course, today we think about finding information anywhere in the world. On the Internet it's a simple thing by searching on Google. But, you know, this was, you know, more than ten years before Google came into existence. And so Cairo was really an idea of focusing on a similar concept of finding things, not in the whole world, but a little bit more contained onto a user's PC.

Q Was an operating system called Cairo ever commercially released by Microsoft?

A No. The project ultimately was not successful, largely, once again, because of some technical decisions that were made early on in the project and because of different efforts that Microsoft undertook, particularly an effort that was underway at roughly the same time as Cairo as code name Chicago, or Windows 95. That ultimately became

the project that succeeded in defining what Windows would look like in the marketplace for many years to come.

But many of the ideas of Cairo did succeed. A lot of the early UI work that we did in Cairo, the user interface work on the way that the user interface of Windows is built, a lot of that pioneering work was done by the Cairo team in those early days of 1990.

If you look today at the Windows user interface, the start menu and the tray at the bottom, the desktop with the icons on it, and in Explorer to be able to browse through the file system, you know, those early concepts were part of the Cairo UI design. They wound up being implemented very differently in the early days of Chicago and Windows 95, but the ideas largely came from Cairo, and then this different implementation that changed the commercial reality. Really, if you look today at Windows 7, it's kind of what -- or Windows XP is, it's quite similar even today. It's evolved, but it's quite similar.

Q Now I believe you said, Mr. Muglia, that after working on the Cairo team, I think you became involved in Windows NT program management. Can you tell us what your role was in that position?

A Right. What happened is that originally the Cairo team -- in the early days when Windows NT was being developed, it was -- it was really a fundamentally very new

thing. It was a next generation operating system that had some very key concepts that had never really existed in the PC space before.

I mean if you go back to DOS and Windows 3.1 and Windows 95, I mean those operating systems were really focused on trying to just get things done quickly for end users. You know, they weren't very reliable. They didn't have any concepts of security. And Windows NT was -- it was birthed from people that came from Digital Equipment Corporation, which -- DEC as it is often known, which was a mini computer company from the 1970s and 1980s.

And there are some great people that worked at Digital and some of them came to Microsoft in the late '80s and started this next generation operating system that became Windows NT, and ultimately became Windows XP and Windows 7. The same operating system that I'm talking about here is the one that you would use today if you're using a Windows XP computer or a Windows 7 computer. It's the same -- you know, a much later generation, but it's really the same foundation that's in there.

And they spent the first few years working on the foundational work to actually make the thing fundamentally boot and run, and that was in the late '80s, early '90s. And the Cairo team was originally separate from the Windows NT group, but it was always meant to run on Windows NT.

Ultimately, I don't remember exactly when, it was probably '92 or '93, the Cairo team folded into the Windows NT team. I, instead of just focusing on the Cairo project, was then running what we call the Windows NT program management focusing on defining the specifications for the product that was released as Windows NT.

Q Mr. Muglia, you referred to two terms in that answer that I would like to explore a little bit. You referred to reliability and security. Can you give us -- not in a very technical way, but can you explain what you mean by those two concepts?

A Sure. There were really -- just for completeness, there were really three fundamental principles that this team from DEC -- it was led by a crusty man who was incredibly smart and does a great job but is often very difficult to deal with by the name of Dave Cutler. Dave is known for, you know, when he gets angry sometimes punching holes in walls and things. So he's not the easiest guy to deal with. But he's a brilliant guy and he's done amazing work over the years. One of those great pioneers in the computer industry in the early days. And Dave really led the NT project. Still works at Microsoft. The guy is 70 years old. He still works at Microsoft right now.

THE COURT: Watch it.

THE WITNESS: I don't think he's punched any holes

in walls for a little while now. He's still there.

And he really had three principles that he started with when he formed Windows NT. One was this concept that it should be reliable, that really when the operating system is running, it should just not go down. It should just work. When you write an application -- you know, the operating system is built by a group at Microsoft, and then applications are built by others. Sometimes they are built by people at Microsoft, but largely they are built by people all around the world, software developers that build all kinds of applications, that those applications can't crash the operating system.

If something goes wrong with one of those applications -- and these things are very complicated, so there are many ways they can go wrong. There are many bugs, it's called, that can be introduced into applications. If one of them fails for some reason, it will fail, but the whole operating system will continue to run.

And so that's the idea of robustness or reliability, that the operating system is solid as a rock, and no matter what happens, the operating system keeps working. And that's generally true. I mean you see it today. Windows is pretty reliable today. We have servers that run for months and months at a time without even needing to be brought down that run Windows.

The second concept that I mentioned was security. This really did come out of the mainframe in the mini computer era. You know, in the early days, anybody could walk up to a PC and just use it. There was no idea -- no concept of logging on and authenticating who you are. And if I put a file on my computer -- this was true in the early days of DOS and Windows, it was true in Windows 95 as well, if I put a file on a computer, you know, somebody else could see it. There was no protection for that. That's the idea of security, that the files that I created are my files and no one else can see them unless I explicitly allow them to see it. That's the core idea of security.

There was a third concept that was fundamental in Windows NT, which we thought would be important in the 1990s. It turned out to not be important, but it looks like it's something very important right now, and that was the -- the third concept was this idea of portability.

Windows NT was designed to run on different kinds of microprocessors, different kinds of computers. Computers -- different manufacturers of computers -- just the chip, the fundamental central processing unit it's called inside a computer, they are different. So, you know, chips made by Intel, which is what all PCs use. Today, if you look, a PC or a Mac uses an Intel chip.

Windows NT was designed to run on those, but it

was also designed to run so they could be ported -- portability, to run on other kinds of microprocessors. And back then there were a lot of microprocessors in existence that we thought might become interesting, microprocessors called MIPS. There was a -- DEC had a chip called Alpha. IBM had and still has a chip called Power PC. Windows NT was designed so it could run on those other architectures.

It turns out that that never turned out to be important because X86 -- the Intel chips have really predominated on the PCs and servers. That's what Windows runs on. But, ironically, now phones run on a different kind of chip called an arm chip. Phones and tablet devices like an iPad, they all run on the arm microprocessor. Microsoft was actually using the fact that Windows NT was designed by Dave, you know, in 1988 to be portable, to actually move Windows in the next version, Windows 8, to run on the arm processor.

So all three of those concepts that Dave created way back in the late '80s are all important. The fact that it's reliable or robust. The fact that it's secure is certainly very important today. It helps protect against viruses and allows users to, you know, keep their data safe. And then the fact that it's portable.

BY MR. HOLLEY:

Q Mr. Muglia, how long were you the director of program

management for the Windows NT project?
A I was in that role until December of 1995.
Q To whom did you report in the hierarchy of Microsoft when you were director of program management for the Windows NT team?
A I reported to Jim Allchin, who was running -- he had originally been running Cairo, and he then ran Windows -- all of Windows NT.
Q Now you mentioned Chicago and Windows NT. How, if at all, did the target customer base for Chicago, Windows 95, differ from the customer base for Windows NT?
A Well, Microsoft embarked on a complicated strategy in the early 1990s, but one that ultimately was quite successful, and that was that we recognized that the ideas that Windows NT had, this idea of being portable and secure and reliable, that that was the future. We needed to build servers that we could create. We also knew that some day that would be what really all PCs would do.

But back in the early 1990s, there was a problem, which was that Windows NT required too much resources for the common computer that people were buying. Again, you've got to put yourself back, you know, 20 years and remember what it was like back then. A PC, you know, in the -- in the early 1990s really had a meg or so of RAM.

Just to give you an idea. You know, if you have an

Android phone, you know, in your pocket, that phone has a gigabyte of storage -- of RAM storage. So it has literally a thousand times more memory than the devices did back then. I mean the phone in your pocket is literally a hundred or a thousand times more powerful than the computers that we were running on at the time. And that was sort of the state of the art, what people could afford in buying a personal computer.

When we looked at Windows NT, we knew it would be the future, but we knew that the hardware that people could afford to buy wasn't really available at the time. And so Microsoft actually built two operating systems at the same time. One was code named Chicago, which eventually was released into the market in 1995 as Windows 95. And that was designed to run at the time -- this is mid 1990s -- on four megabytes of RAM.

Now, remember, I said a phone like an Android phone has a gigabyte of RAM, or a thousand megabytes just to put it in perspective. Your personal computer today has -- if you go to the store and spend, you know, five, 600 bucks on a PC from Dell or Hewlett Packard, that computer will have four gigabytes typically or more of RAM, so 4,000 megabytes, so roughly a thousand times more.

So Chicago was designed to run on four megabytes, and it had a set of tradeoffs associated with that. Windows NT

needed -- we said that it realistically needed 16 megabytes of RAM, and, of course, even that was not really enough. But that a lot of memory back in 1995.

Q Mr. Muglia, can you tell me what you did, just in general terms go through the job that you had at Microsoft after December of 1995 when you stopped running a program management for Windows NT?

A Well, I was very fortunate. I had an opportunity to do some wonderful things working for Microsoft. After my initial work on SQL server and Windows NT, I spent 1996 running our developer tools group, so the group that builds the tools that let developers write software applications.

In '97, I actually ran a set of server products back -- servers are things that run on -- that basically run the back ends of business. Today they run the Internet. Servers are what you use to run the Internet, but also they run business applications. So when people talk about general ledger applications, for example, or e-mail servers or file servers, those are run on servers. And I ran Microsoft's set of server products for a year.

I then spent a couple of years working -- I switched over to the appliance side and ran Microsoft Office, and the product that we built at the time was Office 2000. So Word, Excel, PowerPoint, those products.

I also had a chance to do some early work on our moblie

devices, our Smartphone work. This was early. This was even before the days of Palm, so it was in the early days of these little devices.

I spent about two years on consumer services that focused on what we'll call MSN. So the Microsoft network that is a service that's available today that let's people find information. You know, you can get news and weather and things like that off of MSN. I also ran Hotmail at the time I did that. So Hotmail and our MSN product, which a lot of people use Hotmail as their e-mail system.

And then in 19 -- excuse me, in 2002, I moved back into what was then called the platforms group, and I started a little startup group working on storage products, file servers, again, things that shared files. And then over time picked up more responsibility running our server products. You know, ultimately had Windows server, SQL server. Up until earlier this year, I ran what was called the server and tools business at Microsoft, which was all of our back end server products.

And then I left Microsoft in September and am now working for a company called Juniper Networks. Juniper makes networking systems, routers and switches, things that run the back end of the Internet. When you send mail over the Internet or do a query, go to a Web site and buy something, a good chance the information is traveling across

Juniper equipment. But it's back end stuff that people never really see.

Q Mr. Muglia, what is your position at Juniper Networks?

A I run -- I run the software solutions division reporting to Kevin Johnson, the CEO of the company. So I'm really helping Juniper to branch out into the software business. Juniper has always been a hardware networking company and there was a lot of opportunity to build networking software, and that's what I'm doing is helping to grow this new business for Juniper.

Q Mr. Muglia, I would like to change topics a little bit now that we've been through your background. During the time that you were at Microsoft from 1988 until earlier this year, what, if anything, did Microsoft do to encourage independent software vendors to develop applications to run on Microsoft operating systems?

A Well, from the earliest days, certainly when I was at Microsoft but even before then, Microsoft realized that in order to have a successful operating system business, we needed application developers to build applications that ran on Microsoft operating systems. And so we have always had very active programs working with the -- they are called ISVs, independent software vendors. That's the industry term for it. We've always had programs to provide information to software vendors and to encourage them to use

features and capabilities within Microsoft operating systems.

Q Did Microsoft encourage ISVs to develop applications that competed with Microsoft's own applications?

A Yes, we did. This is sometimes confusing to people. Microsoft had and still has different groups doing different things. So Microsoft has groups building operating systems and it has other groups that build software applications that run on top of Microsoft operating systems.

The software applications, by the way, sometimes run on other operating systems as well, like Office for the Mac has been available for as long as I can remember and it's still the most popular office product on the Apple Macintosh. But certainly Microsoft applications groups build applications that run on Windows.

And Microsoft, even though we had the Office group working to build what we hoped would be the best set of productivity applications for Windows, we also worked with other groups, other companies to build competing applications. Because it's important that if a user wants to use, you know, a product from somebody else, whether it was Lotus or WordPerfect, or whoever it might be, that they could do that.

You know, even today, if you looked at what my old group was, you know, I ran the server groups and we built a

product called SQL server, which is a database, which is that first product I worked on at Microsoft, it's a database product that ran on Windows server. But Oracle and IBM built database products that run on Windows server. And there are open source products that work on Windows server. We would work with Oracle, we would work with IBM, we would work with the open source community to encourage them to build great database products. So that's been true the whole time.

Q Now you mentioned WordPerfect in that answer. Did Microsoft encourage WordPerfect Corporation to write applications to run on Windows?

A Yes, absolutely, we did. Not super successfully, but we did.

You know, Microsoft very much wanted WordPerfect to build a great word processor for Windows. Again, if you go back to the early days of Windows, say the early 1990s, you know, WordPerfect was used by a lot of people on particularly Microsoft's earlier character mode operating system called DOS. And so WordPerfect was very broadly used on DOS.

But it -- you know, as the world was moving to Windows, the world was moving to graphical operating systems, and Microsoft very much wanted WordPerfect to build a great word processor for Windows because it would make Windows more

popular and help to get people to buy more copies of Windows. Unfortunately, WordPerfect really resisted this. They didn't want to help Microsoft make Windows successful. They delayed in their work on Windows.

MR. JOHNSON: Objection, Your Honor, to what WordPerfect and what their thoughts were or strategies were. He doesn't have personal knowledge of that.

THE COURT: You better establish a basis for his knowledge. He can't say what WordPerfect was thinking, but if he interacted with them, then he obviously knew.

MR. HOLLEY: Sure.

BY MR. HOLLEY:

Q Mr. Muglia, in the early 1990s, did you have occasion in your job to pay attention to what ISVs were doing in terms of supporting different platforms?

A Yes, we did -- yes, I did. I worked very closely with a wide variety of ISVs, including WordPerfect, encouraging them to build applications on Windows.

MR. HOLLEY: Your Honor, in light of that testimony, I think Mr. Muglia should be able to continue the answer he was giving before.

THE COURT: Mr. Johnson -- it sounded like it was what WordPerfect was thinking. So just reask the question.

MR. HOLLEY: Fair enough, Your Honor.

//

BY MR. HOLLEY:
Q How, if at all, did WordPerfect support early versions of Windows?
A They didn't.
Q Did you personally do anything to encourage WordPerfect to write for Windows?
A Yes, I did. I talked to WordPerfect people at Microsoft conferences in situations where I had a chance to interact with them. And I traveled to Utah, to Salt Lake, up to the Salt Lake area to visit WordPerfect to encourage them to build Windows applications, and then specifically to work on a technology that we were developing called OLE. But I very much did work directly with folks at WordPerfect.
Q Now you referred in your last answer to OLE, which is something that's come up in the trial. Can you tell us, you know, again, at a fairly general level, what OLE is?
A Sure. Back in -- if you go back to that period again, you know, it was often the case that people might only have a word processor or a spreadsheet. They might not own multiple applications. If you wanted to buy these, they were very expensive. These individual -- a word processor might be four or $500. A spreadsheet might be $500. A database product was typically $700. And there were -- this was the days before Office brought all those things together. And there was no way to build an -- build a

document that had the combination of word processing information and spreadsheet information.

So if you think about a common document that you see today, you know, you might see this in so many ways, people providing financial information or even sometimes more casual documents, it's very common to have the text of the document and then to have a chart embedded in it, or even a picture embedded in it. You can see that all the time today.

Back then, there wasn't a way to do that. You could have text and you could have a chart, but they were separate. And the idea behind object linking and embedding was the technology that let you put a chart that was built by a spreadsheet application into a word processing document. That's really what that technology was about.

And OLE was the technology that Microsoft had developed and ultimately was the one which became successful in enabling the whole industry to build these documents, which are sometimes called compound documents, documents that have different kinds of objects in them. And really they are just presented in a way that makes the information more palatable and easier for the user to understand, the person who's reading the document.

Q Mr. Muglia, which group at Microsoft initially developed the object linking and embedding technology?

A It was originally developed by the Office group. The Office group was the organization that really had the customer problem. The customers wanted to build these compound documents. So Office created the OLE technology.

But we -- after they had done the initial work, we recognized that it was -- and even they recognized that it was important that it be usable by others besides Office. So sure you might want to put a chart from a spreadsheet in a document, but maybe if you've got a drawing that was done by another drawing company, like Corel Draw or something like that, or if you had an architectural diagram that contained the blueprints of something, that maybe you wanted to put some of that inside a document.

So having the ability for third-party ISVs to use this same OLE technology was something that even the Office group recognized was important. And certainly, you know, those of us in the operating system group thought it was particularly important. And so ultimately the technology was moved into the Windows -- actually the Windows NT group and my group began program managing.

Q Now I believe you said you came here to Utah to talk to WordPerfect about OLE. And what did you say?

A Well, at the time there was -- the only product that was really -- the only group -- the only technology, I should say, that was gaining any success in the marketplace

for doing this idea of creating compound documents was OLE. But WordPerfect had been working with a number of other companies on an alternative technology called OpenDoc. And I wanted to encourage WordPerfect to use OLE as they -- as they built their applications. You know, I also in general wanted to encourage WordPerfect to build a great Windows application.

And from my perspective, you know, they were -- although Windows was obviously important, this was a time when I believe I visited them in 1994, so Windows 3.1 had been in the market for some time and was gaining a great deal of success in the marketplace. And, you know, WordPerfect had not released a word processor that had -- I don't even think they released anything. But certainly they hadn't released anything at the time that had gained any market success. From the Windows group's perspective that was really bad. We wanted WordPerfect to build a great product for Windows. And, you know, they just weren't doing it. They were working on other things. They just didn't prioritize it. I don't know what they were thinking, but they just weren't focused on what was important commercially in the market.

It was a real problem for those of us in the Windows group because, you know, we wanted a good word processor from WordPerfect. So I came out to encourage them to build

a great Windows product for Windows 95 and Windows NT. I very specifically went out to talk to them about using this technology called OLE because that technology was going to be very important. If you are building a word processor, that technology was going to be important. And WordPerfect wasn't paying attention to it. And, you know, they just -- they needed to if they were going to be successful. So I just wanted to encourage them.

Q Now you mentioned a technology called OpenDoc. Was that more prevalent than OLE in the early 1990s?

A No, OpenDoc never went anywhere. There were no commercially viable products built on OpenDoc. So WordPerfect was, from my perspective, wasting their time on this technology. And there was -- really, they could use OLE. There was no reason for them not to use OLE. And I wanted them to do what I thought would be best for them and, frankly, it would best for Microsoft too. But, no, OpenDoc went nowhere.

Q I would like to show you a document that's been marked as Plaintiff's Exhibit 502. Mr. Muglia, this document marked as Plaintiff's Exhibit 502, can you tell us what this is, please?

A Well, this is a document that -- I don't know if I ever received this document or not. It doesn't have my e-mail address on it. It would either have my name or typically my

e-mail ID, Bobmu, which was my e-mail -- it was my e-mail at Microsoft. But if you see down below it talks about the agenda and it has Bobmu in it, that would be me. And this would be the meeting -- this was a document that was prepared by Tony Williams, who was the architect for OLE, to Brad Struss, who ran our -- who worked on evangelism, that developed the evangelism group to specifically work with WordPerfect to help get them to do Windows applications, to build Windows apps. It talked about the agenda of that meeting that I was referring to.

Q Now under the heading proposed agenda, Bobmu -- which is the reference to you, correct?

A That's correct.

Q It says, OLE/COM futures. What is OLE/COM? Is that the same thing we've been talking about?

A I mean OLE is the technology I was talking about, which is to build these compound documents, and COM is sort of an underlying technology. It stands for component object model, and it's an underlying technology that enables OLE to work and build these compound documents. And futures is just talking about things that were coming.

Q And at the end of the line next to your name it says Stevem. Who is that a reference to?

A That references Steve Madigan, who was a program manager who worked for me at the time doing -- particularly

working on UI work and user interface work and working on this next generation shell, which was called Cairo.

Q It says, Cairo shell demo. Why were you demonstrating the Cairo shell to WordPerfect in February of 1994?

A Well, in February of 1994, we had believed that this concept -- this idea that Bill had originally spoken about, information at your fingertips, we knew that was important. It turned out to be very important because although what Bill wanted to happen didn't actually happen in terms of the way the products we built came to market, you know, we lived in a world where that vision Bill had in 1990 of information at your fingertips is very true today. And Cairo was our project at the time. Like I say, it ultimately wasn't successful, but it was our project at the time to implement that.

And we were -- in some circumstances, not broadly, but sometimes for really important ISVs that we really cared a lot about, we would show them the work that we were doing in the early days of trying to build this next generation line. And WordPerfect was important to us. They were an important ISV. We wanted them to see where we were taking things in the future. So we did a demo at that -- a demonstration at that meeting here in Utah to show WordPerfect some of the ideas we had where we thought we were going at the time.

Q We've heard a lot in the trial about the concept of

shell extensibility. Was there any extensibility in the Cairo shell at this time?

A Oh, yeah. Part of the idea of Cairo was that it was very customizable by software vendors, and there was a wide amount of options for software vendors to build extensions to Cairo to customize the environment to meet the needs of their applications, the apps that they developed.

Q Was the shell extensibility in Cairo the same as the shell extensibility in Chicago?

A No. No, it wasn't. You know, I mentioned that Microsoft had this challenge that we wanted to build -- we were doing sort of two things at the same time. We were building this -- Dave and his team -- Dave Cutler and his team were building this next generation operating system called Windows NT. And Bill had this idea of doing information at your fingertips, and we were building this very sophisticated set of features and capabilities to implement that vision that we code named Cairo. Those two groups were working together, Windows NT, next generation operating system.

Cairo was this, you know, very futuristic way of doing queries and finding that information really easily. Cairo had a lot of features. Frankly, too many features. Part of the reason Cairo failed was it was too broad for the time and we overshot what we could realistically accomplish. So

Cairo had one approach for doing things.

And, you know, in this same time we were parallel because all of the technology that was being built by the Windows NT group and the Cairo group required computers that were more powerful than what people could buy at the time. In order for us to be commercially viable in the market and to keep the market forward, there was this other group, the Windows group, Chicago group, ultimately the group that built the product called Windows 95 that were taking and evolving Windows in a much more evolutionary approach.

If you go back, there was, you know, the early days of Windows. There was the first viable version of Windows called Windows 3 that I think came out in '91. It was roughly '91, maybe '90 that it came out. And then Windows 3.1, which became a very big commercial success. Well, the same group that built Windows 3.1, which was designed to run on the generation of computers that was affordable at the time, that group was evolving forward and was building this next generation, this next version of Windows that was code named Chicago.

You know, the groups worked -- they shared ideas. So, you know, the ideas that the Cairo team was building were often, you know, adopted by the Windows team. But I mean it's sort of embarrassing really. We didn't get along very well. And that was true at the senior management level and

it was true, you know, by some of the engineers working together. From my perspective, it almost felt like the Windows team, Chicago team was stealing all the good ideas that we came up with in Cairo and building it into their product, but they always did it differently. So they didn't wind up using any of the interface design work that my team was doing.

So if my team was suggesting that an ISV would extend the user interface using a particular set of programming interfaces, the Chicago team would build the same feature, the same capability, but they just did it in a different way. There were sometimes reasons for it, but largely they were just doing it to be expeditious and get something done.

So while things looked the same from a developer perspective, they were quite different. And the teams fought. I mean they fought like cats and dogs on this internally. There were squabbles going on all the time. That happened at my boss's level between Jim Allchin and Brad Silverberg. It happened at my level, which was -- my counterpart was a man by the -- is a man by the name of David Cole. You know, but I still -- he left Microsoft. I still get along with David, but it was a testy relationship at the time. Then it happened below me in my teams and other folks as well.

Q What difference, if any, did it make to you as the

director of Windows NT program management that the Chicago team was implementing features in a different way than the Windows NT team?

A Well, it was awful, because the idea was that we have these two parallel tracks and they are supposed to eventually kind of get to the same place. So, you know, the idea was that when Chicago was released, we would release a version roughly coincident with that of Windows NT that required bigger computers, but that was -- you know, it was released at about the same time. And the idea was that the same applications would run on both.

So if -- you know, let's just say -- you know, let's take Pagemaker. Pagemaker was a commonly used desktop publishing product at the time, still exists in some form or another. It's now an Adobe product. If Pagemaker was released and it ran on Windows 95, the idea was it should run exactly the same on Windows NT. But if the things -- the program interfaces that it calls on Windows 95 are different than the ones it would call on Windows NT, then it couldn't run the same unless the developer built two versions of the product.

So we were developing -- we were developing this work in the Cairo team, doing the leading edge stuff, building -- you know, building this technology. And the Chicago group was doing the same thing. You know, like I say, it felt

like they were stealing our best ideas, but then they would implement them differently, and they were charging ahead. And so we fought.

Q What concerns, if any, Mr. Muglia, did you have about the way in which the Chicago team implemented shell extensibility?

A Well, that was an area of particular disagreement largely because we had two very different design points in terms of what we were trying to accomplish. The Windows NT and Cairo team were building towards this vision that Bill had of information at your fingertips, so we had some very sophisticated mechanisms that would carry the shell. It did require more memory. And the Chicago team were building a very -- they used -- they had a very different code base for the shell. They weren't using any of the code that my team had developed -- or that were developed by the Windows NT team I should say. And they were building a shell that looked a lot like -- looked a lot like the Cairo shell, but, in fact, it was implemented in a completely different way and had a totally different set of interfaces. That's all of the shell extensibility interfaces. And Chicago was filled with shell extensibility interfaces. There were a lot of them -- there still are actually. That shell, in some form or another, still runs in Windows XP and Windows 7.

Q How, if at all, did the principles that you described earlier that Mr. Cutler was trying to implement in Windows NT, reliability, security and portability, affect your attitude toward the Chicago shell extensions?

A Well, it was a problem. You know, if you go back over time, the early days, we just had two totally different implementations. You know, the '93, you know, I think into the '94 time frame, we were building this Cairo shell with its own implementation, and it very much was following, you know, Dave's design points of reliability and security.

So, you know, the idea was that even though you wrote an application that extended the shell, the Cairo shell, if that application does something wrong, if it crashes or stops working, hangs -- that's called a hang where something on the computer just stops, if it does either of those two things, it wouldn't affect the operating system.

And, you know, the Chicago team, they just -- that was not their design point. I mean, again, their design point was being compatible with the existing Windows apps, which there were many at the time, running four megabytes on this relatively small amount of memory. And, you know, they didn't go out -- they didn't seek to build unreliable systems, but the design point they had did build a system which had reliability issues. So if an application misbehaved, if it crashed or hung, it could bring the whole

operating system down.

That was a big problem with the work that was being done in the Chicago shell. It was a big problem, and it was one of the points of significant contention between David Cole and myself.

Q What, if anything, did you say about these concerns you have just expressed to more senior managers at Microsoft?

A Well, it became a significant issue. I mean recognize that this got to the foundation of this key strategy we had, which was, you know, we knew NT was the future because we knew that the principles Dave had laid down were right, but the principles required more horsepower, required more computing power than PCs at the time could run.

We thought Cairo would be super important. And while the idea was important, the implementation turned out not to be over time. But we thought it was important. Meanwhile, we needed the Windows group and Brad Silverberg and David Cole and his band of merry men, they were building something that was important commercially in the marketplace. And, you know, we kind of thought of these guys as cowboys. That's what it felt like to us in the Windows NT group.

Because the strategy of having a short-term operating system, short term as it turned out ran until about 2000 if you look back in history, but having a short-term operating system built on the Windows code base, the Chicago code base

was important for the company. In the long term Windows NT would be important. There was contention about trying to do this in a way that allowed for a graceful transition. And one of the key things was that applications were supposed to be compatible between the two -- the two systems.

And, you know, we were going one way in the Windows NT and they were going another way. So we fought and, you know, it got escalated. Certainly Jim and Brad argued vehemently about it. Paul Maritz was trying to manage these guys. And, you know, Bill Gates ultimately got involved and, you know, helped to sort out some of the technical decisions. So it was very much escalated and there were definitely conversations about this.

Q Are you familiar, Mr. Muglia, with something called the namespace extension APIs?

A Yes, I am. In the Chicago -- in the design of the Chicago shell, as I said, there were a wide variety of APIs that allow software developers to extend the functionality of Windows. You know, you can actually see this, and you see this visibly true today. You know, if you look at the bottom -- if you look at the desktop -- as I say, the Chicago shell that we're talking about, this thing that was in great contention in 1993, that same code is in Windows 7 today. It's changed over time, but it's evolved. But it is the evolution of that code base that went into Windows XP

and Windows 7, because ultimately the Windows NT code base became the code base of Windows for the Chicago shell code base continuing forward.

If you look at the shell, you see at the bottom there is this thing called a tray. It's got the start menu and there's a tray. In the right-hand side of that, there are a series of icons. Those icons get added by ISVs. Those are done by a form of shell extension. You know, if you right click on something in the shell, you can see a context menu. The menu items in there can be extended by ISVs. Those are examples of shell extensions.

Those are things that are very commonly used and extended by ISVs. In fact, they are so commonly used that today they have too many -- too many of them running on many of our computers. It's one of the things that drives people crazy about Windows.

But there was a particular class of extension -- of API called namespace extensions, which for a variety of reasons back in the early '90s became highly contentious, largely because they were -- they were the most problematic in terms of these concepts of reliability. But the namespace extensions are a kind of an API set that turned out to be not very important at all. For all of the discussion back then, they are not used very broadly.

And the only kind of applications that would even

possibly use them -- possibly use them are applications like mail applications, or maybe a desktop publishing system. But, you know, a common word processor spreadsheet would never even -- namespace extensions made no sense for those applications. So they were a subset, a class of the APIs.

Q Let's talk about some of the things you said in that answer. Why is it, in your view, that a word processor or a spreadsheet would not use the namespace extension APIs?

A Well, I mean you've got to understand what these things do. And it's sort of funny because if we look at the way computers work today, they just don't work the way we thought they might have worked back in the 1990s, and certainly they don't work the way the namespace extensions would have allowed.

You know, you see when you -- on a Windows computer, frankly, even a Macintosh, it's quite similar on a Mac, when you want to open a word processing document, let's say you've gotten a document, maybe it was mailed to you and you saved it into the file system or you have a little thumb drive and you put it in, you know, Windows will open a folder, which is literally a view onto the file system, and you can see that document there. And if you just double click on the document, the word processing application or spreadsheet application will open.

The namespace extensions were designed for a very

different idea. They had this idea -- it was this idea, it somewhat came from Bill and information at your fingertips vision, which was a great vision, but there were some things that didn't come true, as I said. The idea of the namespace extension was that you want to use the operating system just to like open a document and view it in there. But when you're doing other things, like your e-mail, you know, working on e-mail, you would do that in the context of the operating system.

So, you know, what the namespace extensions did is they would let you look at visually things that don't exist physically in a file system. As I said, a word processing document or spreadsheet, that actually physically exists in the file system, so you don't need the namespace extensions for that, which is why I say word processors and spreadsheets would never even use them. But the kind of application that might use them, let's say you wanted to look at your e-mail, and, you know, the idea of the namespace extension was, gee, you know, maybe the best tool to use for your e-mail would be Windows itself and to use the Explorer in Windows, the thing that allows you to find files, to actually use that to view your e-mail.

In retrospect it didn't make sense. We don't use that today. Today people use products like Outlook or Notes, or something to -- you know, or Google, or whatever, Gmail to

look at their e-mail. They go to the special purpose applications to view their e-mail. But Bill had this vision that, you know, it's very Bill Gates like. Bill is a brilliant guy, but, you know, he has a particular view on the world and sometimes Bill is a little bit -- you know, he's a little pointy headed for the normal mortal. And he had this view that people would use the operating system Explorer and -- you know, Bill just loves -- like I say, he loves hierarchies and he loves the idea you could browse through things. It just seems so natural to Bill that you would start in file system, you would keep browsing, and eventually you would get to an e-mail message. And you would use the Explorer, the Windows operating system shell, to view your e-mail messages.

Well, that's what the namespace extensions were for. I mean they were for using Windows to view things that didn't physically exist in the file system, that existed -- that existed -- they were there, but they were actually inside another file. So if you technically look at the way Outlook works today, there is a big massive file that Outlook maintains that has all of your e-mail in it. And, you know, if look at that file and the file system, all you see is this big file and you can't do anything with it.

Well, if the namespace extensions had amounted to a hill of beans, you could have gone into the Explorer in

Windows and instead of seeing this big file, you would see e-mail messages and you could open those e-mail messages with the Windows Explorer. That was the idea that Bill had and that's what the namespace extensions were for, they allowed you to use the operating systems Explorer to view, you know, any kind of object, like an e-mail message, or maybe a document management system would use it.

It turned out to not matter. It's not what people want. It's just not what people do. So, you know, although these things were talked about back then, they were never important. They never amounted to anything in the industry. Nobody uses things like this today. And instead people used products like Outlook. I mean if you want to look at your e-mail, you don't use the Windows Explorer, you use Outlook, or you use Gmail, or you use Hotmail, or you use whatever tool -- AOL, or whatever tool you use. That's the way the world wants to work. The namespace extensions were meant for something that never really mattered.

You know, ironically, the funny thing is that the world has gone in the opposite direction. You look at an iPhone, look at an iPad, they don't even have a file system you can see today. You know, we thought back then that everything would be done through the file system.

Well, today, you know, modern, new devices don't even make the file system visible. You don't even see that on

your iPad. Instead you see apps, just like you see Outlook on Windows. That's what the namespace extensions were. You know, we thought they would matter, but they were totally unimportant. And even when we thought they would matter, they would matter to a small class of applications.

Q Mr. Muglia, in an earlier answer you referred to reliability in connection with the namespace extension APIs. Can you tell me what you meant by that, sir?

A Well, you know, this comes back to these principles that Dave had set about, you know, building a system that doesn't crash and apps don't crash. Well, unfortunately, all the Chicago -- the entire Chicago shell was not built with that principle. And, you know, operating systems have ways to -- or at least let's put it this way, a modern operating system like Windows NT, Windows 7 today has ways to isolate the work that applications were developing. So that when they build something, if it goes down, it doesn't bring the operating system down. It's often referred to as a separate process. It's done separately. The operating system can isolate that and protect the rest of itself from it.

Chicago wasn't really built that way. Chicago just didn't have these process boundaries that a modern system like Windows NT had. And because of the memory limitations, there were some viable technical reasons, some real

technical limitations that caused this.

The Chicago shell was designed so that if something goes wrong, that -- you know, it was designed in a way so that when an ISV extended the shell, it did so in the same process that the shell actually ran in. What that means is that if that application crashes, it can crash the shell and it can bring the system down from the end user perspective.

That was true for almost -- I think all of -- if not all of most of the extensions, the shell extensions that Chicago had. And, again, it was a design point. I mean it wasn't like they were actively trying to build an unreliable system. It was really that they were trying to run in four megabytes. And this protection, this process separation takes memory, and they just didn't have the memory. So it was designed so that, you know, things ran in the process.

Well, the worst -- all of these were a problem, but the one that was by far the worst were these namespace extensions because most of the code that a developer would write was just a very simple set of things. Let's take that context menu that I described earlier. When you right click on something, ISVs can add lines to the menu. They can add menu items. So you see this in Windows today. You right click and you see a standard set of functions, cut, copy, paste. Then delete, rename, you might see functions like that. But you also might see a function that says backup

or, you know, a function that says archive. Those are things that would be extended by an ISV. Those are context menu extensions they're called.

Well, the thing about that is it's just a few lines of code. So to write that extension, it might be 20 lines of code. You can blow it on 20 lines of code. You can get 20 lines of code wrong and have an error, but your likelihood of getting this right is pretty high.

The shell extensions, on the other hand, basically ran an entire application in the context of the shell. So it would be like you're running Outlook -- all of Outlook inside the Windows operating system. If Outlook failed, the Windows operating system would crash. You know, Outlook is millions of lines of code.

So while the Chicago shell extensions all have this attribute, they do not follow the robustness principles that Dave had set down. The shell extensions, the -- excuse me, the namespace extensions were the worst of the worst, because they encouraged ISVs to run their entire application in the context of the operating system.

Now, you know, I hate to say it, but applications crash. I've seen many applications crash. I've mentioned Outlook a few times. Outlook, in particular, sometimes hangs and crashes. You know, the idea that you are running all of that in Windows means that -- you know, today if

Windows Outlook fails, at least you can keep doing other things. You can stop Outlook and keep going.

Well, if it was running in the context of the shell, it might bring all of Windows down. That's one of the reasons why I particularly despised these interfaces because -- and was really working hard to get them to be -- you know, to not be shipped because they were so bad. Like I say, they were all not great, but this was the worst of the worst.

Q Mr. Muglia, when you say you were working hard to stop the namespace extension APIs from being used, can you tell us what you did on that score?

A Well, I mean I -- you know, this was -- I did many things. I tried to work with David Cole and the Chicago team. But, like I say, we mostly fought like cats and dogs. I talked to Paul and Jim and Bill to try and encourage them not to do it. I even at various times worked with the Microsoft applications groups to encourage them to take my side and focus on something that would provide much more -- much better service in the long run that would do this in a way that was consistent with the NT principles.

You know, in a way, there were sort of two periods of time. There was the period of time when we were developing the Cairo shell where we had a competing set of interfaces, a competing design. And then there was the time after that when we decided we would not go forward with the Cairo shell

in Windows NT but instead used the Chicago shell.

Then I was just sort of desperately trying to make these things as reliable as I possibly could. You know, I kind of inherited the mess. We were doing everything possible to try and make it as reliable as possible. Whereas Windows NT was designed from the beginning to be reliable, the Chicago shell was not. And so once we decided to use the Chicago shell, we had to try and salvage what we could and sort of make it as reliable as possible.

You know, of the interfaces that the Chicago shell had, by far the worst and the one we just didn't know how we could successfully implement was the namespace extensions.

Q Now, Mr. Muglia, the jury has heard at the trial that there was documentation provided for the namespace extension APIs, something called the M6 beta of the operating system. If these namespace extension APIs were as bad as you say they were, why were they documented in June of 1994?

A Well, that was the question I asked actually. You know, Jim, myself and the Windows NT guys, we sort of thought Brad and David and those guys were a bunch of cowboys. They were going out doing stuff that it didn't seem like -- it always felt duplicitous to us -- we thought duplicitous. And we thought we had an agreement that they wouldn't go ahead and publish this. But, you know, they kind of took the approach that, you know, if they got there

first, you know, they could claim the territory and people would start to use them, and then they could shove it all down our throat.

So they went out and documented these things. And I was pretty angry. I was pretty angry when it happened. Jim was pretty angry. I was pretty angry. But that's what happened. And so there was a big fight internally over the fact they went ahead and published those.

Q Why did you care as a member of the Windows NT team whether or not the Chicago namespace extension APIs had been published to ISVs?

A Well, as I said, we had these two parallel tracks that were supposed to go to the same city. And, you know, these guys were out telling ISVs to use these set of APIs. You know, we had a different set. And basically because they were going to ship first, they were commercially going to get a lot of success. It meant that basically we would have to support what they did. I mean ultimately what happened by them going forward and publishing it, ultimately what it meant was that we were not able to go forward with the Cairo shell as we had planned.

Now to be fair, Cairo had its own problems. I'm not going to tell you that -- you know, if I look back 20/20, that Cairo was a fantastic project and didn't have any challenges of its own, it had plenty. But, you know, the

nail in the coffin of Cairo was publishing these interfaces, the Chicago shell interfaces, all of them, because what it meant was that -- and in particular these namespace extensions, which we didn't know how to support, because what it meant was that we couldn't build the Cairo shell and support all of these things.

So ultimately the Cairo shell was canceled and, you know, the people were merged into the Office team and, you know, we moved forward with Windows NT, but with really leaving the Cairo vision to be done in the future, which ultimately, like I say, sort of happened with the Internet.

So in hindsight, the world kind of caught up to Cairo, but Cairo itself failed. You know, I would say it wasn't one thing, but clearly supporting the Chicago shell extensions was sort of the final straw that broke the camel's back.

Q Now I would like to turn to a slightly different topic, which is a retreat at Mr. Gates's place on the Hood Canal in June of 1993. Did you attend that retreat, Mr. Muglia?

A Yes, I did.

Q I would like to show you what's been marked as Plaintiff's Exhibit 52. Now you are not on this e-mail, Mr. Muglia, and I don't know whether you've ever seen this document before, but I do want to direct your attention to the third page, the page numbered three at the top. It has

this Bates number 9441 at the bottom. It says Bobmu-Cairo. Do you see that?

A Yes, I do.

Q Do you recall making any sort of presentation about Cairo at this retreat?

A Yes, vaguely. It was a long time ago, but, yes, I do. I remember the retreat pretty clearly. I do remember presenting there.

Q Further down the page there is a section entitled notes from breakout presentation, and at the very bottom it says, ship extensible shell in Office. Wire the features we need for Chicago into the Explorer, e.g. mail integration, printman, CPanel, fonts, et cetera. Billg says do it.

Based on your attendance at the retreat, was a plan formulated to remove extensibility from Windows 95 and put it in Microsoft Office?

A No, no, never was. I mean one of the things I would call your attention to is what you just said, which is it says notes from breakout presentations and wrap up. The way a retreat at Microsoft would typically be done is, you know, the first part of the retreat we would have presentations of the team, what the company strategy might be, and, you know, Bill might talk, Paul Maritz might talk, and maybe Jim Allchin and Brad, maybe in something like that I would present what my team is doing in Cairo, and David Cole might

have presented what he was doing in Chicago.

But then very typically -- these retreats are often two days long. Very typically, we would have breakouts. And, you know, the idea behind breakouts was just brainstorming, pure brainstorming, coming up with ideas that might be interesting. So we would break into small groups, you know, three or four people, maybe a few more. Sometimes we would be given topics to focus on, specific areas, you know, maybe changes in technology or competition issues, you know, competitors, or maybe focusing on where the industry is going, whatever it might be. And then these teams were given assignments to come up with ideas. And, you know, they typically would meet for an hour and a half. They are very free flowing, people are just talking and thinking and brainstorming. And then they would take a flip chart and write the results of that, you know, that brainstorming on a flip chart.

You know, this is notes from breakouts. So what this is is notes from, you know, a freewheeling and brainstorming session that a few folks got together to do as a part of this retreat.

So one of the ideas that came up was this idea of an office shell. And, you know, it's an idea. It was thought of, it was discussed in this brainstorming session. It never went anywhere. We never put a project in place.

There was never a team assigned to it. It was never something that went forward in any substantive way. Certainly we never, ever moved on a plan of taking the APIs out of Chicago. As I said, the opposite in fact. Chicago had these APIs, they were published, and certainly went forward. They exist today literally in Windows 7. So that certainly never happened.

But there was never an office shell. There was never a team that built an office shell and it never went anywhere. This was just notes from a brainstorming session that a few folks talked about at this breakout.

Q Mr. Muglia, I would like to show you another document which you have seen at the trail from this retreat, Plaintiff's Exhibit 51. Mr. Muglia, your name is not among the team members listed on the front of this document, but I would like to direct your attention to the page -- they don't have internal numbers, but the slide that has the Bates number 2535292. It's three-quarters of the way through the documents. It's entitled the radical extreme, the office shell. There is a reference here to -- the second bullet point says Chicago shell is nonextensible.

Did that occur?

A No, it did not. I mean I would reference -- you know, again, I don't know where this -- this document I first saw literally yesterday in preparation for my testimony this

morning. I don't recall it at all. But, you know, I would actually prefer to go back to the first page for just a second, if I might.

On the first page, you know, I will note what it says, group one topic. This is exactly what I was describing. I don't know whether this document was from that particular retreat or not, it might -- very well might have been. But, you know, I mentioned that you'd get a set of groups together and they would be given a topic assignment, a group -- a group would get a topic assignment. There might be five groups and they might each get a different topic. As I say, you wrote things up in flip charts. You sometimes wrote them up in PowerPoint slides. This appears to be a PowerPoint presentation that came out of one of these breakout sessions.

So I will just point out to begin with, this is this brainstorming. This appears to be a document that reflects a discussion that was had during a brainstorming session. You can see from this first page the group one topic. This is the topic that group was assigned to. Specifically it was assigned how can Windows -- Chicago help increase the sale of Microsoft apps, how can it increase market share, and how can Microsoft apps help Chicago. In other words, how can the applications group at Microsoft help the Windows group and vice versa, which is certainly something we

discussed.

So now if I refer back to the page that you had referenced, you know, one of the ideas they had apparently was what they entitled the radical extreme, you know, this office shell. And so this, you know, group of -- you can see the names, by the way, of the guys that worked there. Mike Maples, John Lazarus, Tandy Trower, Steve Madigan, David Cole, Chris Graham, Ed Freeze and Nathan Myhrvold. You know, that group of people got together for two hours, had a brainstorming session where they were assigned to think about and think about ideas, you know, in a short period that might be interesting for, you know, the company to do in terms of how Windows can help Office and vice versa. You know, they came up with a number of ideas in this document.

If you look at the full document, it has a whole bunch of ideas. You know, one of the ideas that they chose to say was extremely radical was an office shell, and so they wrote that down. And, you know, it's a fine brainstorming idea that nobody ever got assigned to and never was a plan and we never did anything with.

Q In particular, back on this page 292 at the bottom, did Microsoft ever go forward with the idea of making the Chicago shell nonextensible?

A No. I mean, as I said, we never did that. We

published for a broad set of APIs in Chicago. ISVs used those APIs. They exist in Windows 7 today. Many of those APIs are exactly the same today as were defined in the '94 time frame, '93 time frame, and there are a lot of them.

Q Now I would like to show you, Mr. Muglia, what's been marked as Defendant's Exhibit 49. Is this an e-mail that you wrote in July of 1993?

A Yes, it is.

Q The title is office shell; is that correct?

A Yes, it is.

Q Can you explain to me what you were doing in laying out those three options that you list in this e-mail?

A Well, I mean this was -- you know, this was after that offsite I believe, and we were talking -- this was -- you have to go back to this period where, as I said, the Windows NT group and the Chicago group, you know, were fighting like cats and dogs over these interfaces in the shell. You know, with the Windows -- the Chicago group, you know, having their interfaces that I particularly didn't like because they were different than my Cairo product and they were not robust.

And Chris Graham was a program manager that worked in the Office group. And, you know, one of the ways you can influence an operating system design is based on what the applications need. And certainly the Microsoft applications

group was an important customer to us. And so there was weight -- what the apps group did had important weight in terms of what happened in the operating system. That was true throughout the industry, but it was certainly very true for the Microsoft apps group.

So I was in this mail basically trying to get Chris to get on my side and basically say that they should move forward with the Cairo -- the Cairo plan versus the plan that we were on, which was to use the Chicago shell interfaces. As you see, I present three options here. You know, the first one I called status quo. You know, this was my, you know, recognition of reality, which was the project I was working on was not -- the project I was working on, Cairo, was not on a path to success and the Chicago interfaces were on a path to winning, and I didn't like that at the time. And I was saying that, you know, we could continue to move forward with Chicago, but that would be bad for the Office group because the Chicago interfaces didn't do everything that the Office group wanted to be done.

So there were features that Office wanted in the operating system that Chicago didn't support, again, because they had used design principles of running on a small amount of memory. So I said that was a really bad option.

Then I talk about this office shell and, you know, again, this was just a concept. It was being discussed, but

it was just a concept. And I said, you know, hey, I don't know how this would work organizationally, it was sort of funny in retrospect because I said basically if you're going to do an office shell, it would be all my guys running it, seems kind of funny in retrospect. But that was apparently my perspective at the time, if you read the second paragraph under office shell.

And then I said, basically this isn't a very good idea to build the office shell because you won't be able to get what you need done in time. You won't have the flexibility you would have if you use the Cairo shell.

Then I gave the third option, you know, which was called bet on Cairo. I think my first sentence here says at one point long ago, this was the plan, indicating that, you know, at one point we had been chartered with this, but we're not on a path to succeed. Then I proceeded to, you know, suggest to Chris that going with Chicago -- going with Cairo would be the best plan.

What I was trying to do here in this mail was convince Chris to get on my side and, you know, fight against the Chicago team, to kill the Chicago shell interfaces and use the Cairo interfaces and to bet on the design that my team was doing.

You know, this office shell idea, I said, you know, it's just an idea that was not a very good idea. The

Chicago shell interfaces, they weren't very good. Again, Chicago and Cairo were fighting like cats and dogs and what I was trying to do is get the Office guys to take my side. I didn't succeed, but it was a valiant effort at the time.

Q Now going back to the first option entitled status quo in the paragraph that begins, although this is the current plan, you say, it seems like a bad option no matter how you view it. What did you mean when you said their extensibility mechanism isn't compatible with anything?

A Well, you know, the Office group built OLE, you know, this technology that you get back to object linking and embedding. We used the same -- the Cairo team used the OLE technology for our shell interfaces. So the Office team built OLE for this function of putting like a chart within a word processing document. But those same mechanisms could be used pretty generally and we used that same underlying technology in Cairo to build our shell extensibility.

So I was -- my view was the Cairo shell mechanisms were more compatible with what Office was already doing. Chicago wasn't using. They had their own mechanisms. So I said it wasn't compatible with anything. Like I said, I argued it was a bad idea. At the time I felt very strongly it was a bad idea.

Q Down under the heading office shell, in the third complete paragraph you say, the one thing which must be

understood about this plan is that it is not a plan in which pulls Chicago features into Office for the benefit of our apps. As currently planned, the features one could pull from Chicago are not compelling enough to do the office shell.

What did you mean when you said that the features one could pull from Chicago were not compelling?

A Again, I thought the Chicago shell interfaces were just dog meat and that, you know, they should die and not be used. And I was just arguing that they wouldn't meet the needs of the Office group, and what Office needed to do was very different than that.

So you, know, this was -- using the Chicago features was not a way of getting Office moved ahead. The Office group needed more than that, which is certainly what I believed at the time. They did need different things. In retrospect, a lot of this was all irrelevant. But what I was saying that this was not -- it was not -- you couldn't use Chicago for Office to get a big benefit. It just didn't work that way. Because what Office wanted, Chicago didn't even provide. But the Cairo team, we provided.

Q Which of the three options was what happened in the end?

A The status quo. Chicago went forward. The shell -- the interfaces got published. The world adopted it. The

Cairo shell was killed. Windows NT moved forward using the Chicago shell. Eventually Windows NT became Windows XP and used the Chicago shell. We made it a little more robust over time. We did some things to make it robust, but it's still not well designed. It's still one of the poorest designed parts of the Windows XP and Windows NT environment, but it's compatible. And it's the Chicago shell. All those interfaces that were defined, you know, by the Chicago team had been used by literally hundreds of thousands of ISVs. Virtually every ISV on the planet uses those shell extensibility interfaces as part of the apps they build.

Q Mr. Muglia, I would like to show you what's been marked as Plaintiff's Exhibit 473. You are not on this e-mail chain, but it refers to you. In the bottom e-mail from Mr. Silverberg to Mr. Adler -- first of all, can you tell me who Mr. Adler -- what his job was at time?

A Dennis Adler was the program manager who worked for David Cole on the Chicago team.

Q Mr. Silverberg writes to Mr. Adler, David and I met last week with Bobmu. That's you, correct?

A That's correct, yes.

Q And Jimall. Who was that?

A Jim Allchin, who I worked for.

Q To discuss. We decided that it's A-list and Bob is having a team to determine how to wrap the APIs under Cairo.

Between us two, there is also a chance that Capone won't make Chicago, thanks to OLE2 being so late, or that the Capone we ship is the stand-alone, nonintegrated one. Nevertheless, we decided we would document the shell extensibility after we have finalized the APIs. All agree, however, that we should not use OLE2 for extensibility given how late it's coming in, and we were told we had made the right decision, which we knew all along, but was nice to hear from the others, particularly Bobmu, say it.

Did you agree in September 1993 to withdraw your objections to the namespace extension APIs?

A No. No, I didn't. I think Brad was getting ahead of himself in this mail. And, again, I wasn't on this mail, so I literally first saw it yesterday when I was preparing for my testimony here. I would not say that I had agreed to what Brad was saying here.

Certainly I continued to fight. All I know for sure is that after September of '93, I continued to fight against those namespace extensions. So Brad was reflecting that he had achieved agreement with me and there's probably some things he said that I agreed with because I try to be an agreeable guy sometimes, but I didn't think -- I don't believe I agreed with that, with what he said here. I certainly fought hard afterwards that we should move forward with the namespace extension APIs.

I mean I will point out, it's not 100-percent clear what APIs he's referring to in this mail. It does appear to be the namespace extensions because he's referring to Capone, but it doesn't specifically say that, I believe.

Q Now I would like you to take a look at what's been marked as Defendant's Exhibit 58. Now, again, this is an e-mail chain that you are not on, but it's discussing shell issues, and Mr. Silverberg writes, I talked to Paulma today. Whose e-mail alias was that?

A Paul Maritz.

Q About some shell issues, especially at the global level, such as apps plans and Cairo. He says that the apps group wants to get out of their end of year cycles they are now on and into a mid year cycle. And this means they will do Chicago revs of the apps with Chicago. And they plan to write a bunch of shell extensions to the Chicago V1 shell. This means, of course, that Cairo is going to have to run these Chicago shell extensions.

Why would that be? Why would Chicago have to run -- excuse me, Cairo have to run the Chicago shell extensions?

A Well, it's like I said earlier, Chicago and Cairo were two different trains but they were supposed to arrive at the same city and they were supposed to run the same apps. You know, if you published the Chicago shell extensions and ISVs write to it and they build applications on it, then we would

need to run it under Cairo.

Q Now looking at Mr. Silverberg's e-mail to Mr. Cole, he goes on to say -- this is down in the fifth paragraph -- the sixth paragraph, and this means that we really have to work well with the Cairo guys to develop the extensions so they can support them.

Did that happen?

A No. I mean we were fighting like cats and dogs and we continued to fight until eventually, you know, the Chicago team won the battle and Cairo was canceled. So, no.

Q Why did that battle go on for so long, Mr. Muglia?

A Well, because, you know, you had two -- as I said, there were two different projects that had different goals. Both of them were important. And we were really trying to figure out how to make both of them work.

You know, there was the Windows NT, Cairo goal where the idea of building a long-term operating system base that would last for many years to come. It was important. This idea of information at your fingertips that Bill said was important, and so that's what the Cairo team was working on. There was the reality of four megabyte computers, you know, affordability of PCs at the time, and that's what the Chicago team was doing. And the teams were working on different objectives. And both objectives were valid, but they were -- you know, they were at least partially

incompatible.

I think we could have done a better job. You know, in retrospect, I look back on this time with a great deal of regret that the teams fought so much, that I fought with David and, you know, Brad and Jim fought like crazy.

This e-mail, which I hadn't seen before yesterday because I was never copied on it, it was a fascinating case in point. I mean the e-mail is a mail that Brad sends to a bunch of his guys. I think Steve Madigan was on there, and Steve worked for Jim. So although it doesn't show this, Steve apparently forwarded it to Jim. And Jim just gets so angry in this e-mail.

If you go up to the bottom of the first page, you know, if you look at the beginning of the second paragraph, it says, you know, you both knew this was a critical issue to me. I can't believe that neither of you discussed this with me personally. You know, blah, blah, blah, blah. He just rants and raves, which Jim is known to do sometimes.

Then Paul up above is trying to -- you see Paul is trying to balance these two guys working for him and trying to settle the matter. I had never seen this mail, but it's completely consistent with the battles that were raging at the time. I knew both Brad and Paul, and Brad and Jim very well. I was just at Jim's 60th birthday party two weeks ago. Brad is now working for a venture capital firm and I

certainly meet with him occasionally. We're all good friends. These are good guys, but they fought like crazy back then. Jim and Brad still struggle with each other a little bit, and I think that lasted.

I am glad that I've been able to build relationships with both of those guys and maintain them over time. But, boy, we were fighting in this mail. This mail shows it as clear as day.

Q Did the Windows NT team ever agree with the Chicago team in 1993 or 1994 that the namespace extension APIs were okay?

A Not to my knowledge. I mean you said the Windows NT team. I mean there is a broad set of words there, so I can't tell you that somebody didn't say something, but I ran the Windows NT program management group and I never agreed. I hated those APIs. They were bad APIs. Like I say, they were unreliable.

At first I hated them because I was building the Cairo shell and they were always competition. They were an internal competitor and they were just different in ways that I thought were somewhat arbitrary and poorly designed. Later, once, you know, it became clear that the Chicago shell was going to be the future shell and we decided to put it in Windows NT, you know, I then switched to struggling and focusing on how to make these APIs, which are

fundamentally non-robust, how I could make them as robust as I possibly could.

Most of the APIs we could find work arounds for. They're ugly. They're ugly, ugly, ugly work arounds. You still see this in Windows today. There are times when -- you know, I don't know if you notice this occasionally, but sometimes when you are doing something in Windows, even Windows 7 or Windows XP, you will be just sitting there and all of a sudden the desktop goes away and it disappears. Then about ten seconds later it reappears.

Well, what probably happened then was a third-party add-in that used one of these shell extension APIs, it probably crashed. Maybe not. I say a third party, it could be Microsoft. Microsoft has been known to write code that crashes too. But some application add-in crashed and the shell has to go away and restart that little process. It comes back in a minute.

Well, that was one of the work arounds we put in in Windows NT to make these blasted shell extensions more reliable. But the namespace extension were -- they were just awful because they would run an entire application, millions of lines of code in the context of the shell, so dramatically increasing the unreliability of the system. I just found that unacceptable. I always found it unacceptable. So I fought like crazy right until the end

where at one point we did get them canceled.

If I recall correctly what happened is, you know, Bill had made the decision that the APIs would get canceled. In fact, what happened is they stayed in the system but their behavior was changed. The behavior they have is better than the original behavior design. But I fought from the very beginning to the end on those namespace extension APIs.

Q Now, Mr. Muglia, in your last answer you made a reference to Bill making a decision to do something. To whom were you referring in that answer?

A Bill Gates.

Q And when did you come to learn that Mr. Gates had made a decision regarding the namespace extension APIs?

A I believe it was the fall of 1994. I mean recognize that this argument had been going on for some time. And it's commonplace that at the time in particular, if there was a technical issue where two groups at Microsoft disagreed, that Bill would, you know, act as the judge, so to speak -- excuse me, Your Honor -- act as the judge to decide -- to make a final call on that. So ultimately the case went to the, you know, court and Bill made a ruling in the fall of '94.

Q Now, Mr. Muglia, I would like to show you what's been marked as Defendant's Exhibit 21. Now the bottom e-mail, the earlier in time e-mail in Defendant's Exhibit 21 is from

Mr. Gates entitled shell plans, IShellBrowser, dated October 3rd, 1994. Are you one of the recipients of that e-mail, Mr. Muglia?

A Yes, I am.

Q And the e-mail that appears above, is that an e-mail that you wrote?

A Yes, I did.

Q Can you tell us who the people on -- or what the aliases are on the to line in this e-mail?

A The two that I sent I assume you mean?

Q Yes, sir.

A Karen Brown and Paul Goode worked in our technical documentation team. They are the folks that actually document the interfaces for third-party ISVs. The Windows NT program management group was my organization. It was the team that reported to me that designed the specification for Windows NT. And Leif Pederson was the -- sadly Leif passed away a while ago. Leif was at the time a development manager for the shell team.

Q Mr. Muglia, you write in the first sentence of your e-mail, I am sending this out broadly because of the general interest in the group with regard to this decision.

Why was there general interest in the Windows NT program management group with regard to Mr. Gates's decision?

A Well, because, you know, we were -- at the time this e-mail was sent, you know, we had made the decision that we were going to use the Chicago shell. And we weren't sure what was going to happen to the Cairo shell. The shell team I think was subsequently moved into the Office group and there were still some folks that were working on that and thinking about that. But basically we had realized that at least for the next version of Windows NT, as it turned out for all versions in the future, we would use the Chicago shell. And my team was trying to figure out to make the darn thing robust, how to make this piece of code that was written in Chicago but was not designed to be robust to uphold Dave's principles.

So people cared a lot about those APIs, and they particularly did not like the namespace extensions, which is what the title of this e-mail is. IShellBrowser, that is the actual name of the interface, the so-called namespace extensions. People really worried about how we could possibly make that thing robust.

Q In the second paragraph you say, this is very good news for BSD. What was the acronym BSD?

A Business systems division. It was the Windows NT group, just the name of it.

Q You say, since Bill has decided these interfaces won't be published, NT development does not have to expend

precious energy on implementing these for NT.

What did you mean by that, Mr. Muglia?

A Well, what I meant was that the IShellBrowser interfaces, which fundamentally in their initial design were completely un-robust, and, like I say, had this attribute of running potentially running millions of lines of code in the context of the operating system, which we did not know how to fix. That because Bill decided that we wouldn't publish them, that fortunately we didn't have to spend a lot of time trying to figure out what to do to make it work in a compatible way. We didn't know how to do it. We literally didn't know how we could uphold our principles of robustness and implement these blasted interfaces.

So Bill, fortunately, made the decision of -- the decision that I felt was very strongly backed up technically and one that, you know, ultimately had no commercial impact, from what I could tell, to make this decision not to publish these interfaces.

Q You say in the next paragraph, this also means that the UI model -- does that mean user interface model?

A Yes, it does.

Q For Chicago/Cairo does not put all containers in the scope pane of the Explorer. This is a much more flexible solution which allows for more UI innovation in the future, both in Cairo and in Chicago.

Can you explain what you meant by that, Mr. Muglia?

A Yeah. It's that relatively obscure point that I was describing earlier, which was sort of the gestalt of what these namespace extension interfaces were all about. This idea was, and, again, I think it came from Bill's view of how he thinks the world should be, that, you know, that every user would, you know, go and traverse the file system and use the file system as the way to explore as their application for Bill doing everything inside Windows.

You know, when I say the scope pane, that's the left-hand pane, that's the tree view, that would not put all containers there. In other words, if I have an e-mail folder in my e-mail system, I will never see that in this scope pane of Explorer. Instead, the UI model changed from using the Explorer to do that to one where programs like e-mail run in a separate window, which is what I was saying. That's the way the world works. It turns out that's the way the world wants to work -- things to work. The idea of doing this browsing was never a good idea in the first place. We thought it was important, but it was never an idea that users actually liked.

And what it meant is that the way the user interface was designed meant that applications like an e-mail system, which these namespace extensions were designed to work with, would instead run in its own window. That's the way it

works. If you use Gmail or you use Outlook, or whatever, it's a separate -- it's a separate app. That's the way the world works.

Q Mr. Muglia, to what extent was your opposition to supporting the namespace extension APIs based on your concern that they would be used by companies like Lotus and Novell/WordPerfect?

A Not at all. I mean it literally didn't come up in the discussions. I mean the idea that any of this had anything to do with Novell was not something we discussed. I literally cannot recall ever having that discussion. I remember dozens of conversations with David Cole and Brad Silverberg over these interfaces and the issue of robustness. We fought like cats and dogs, but we never talked about it in the context of competition with Novell or anything like that.

Sure it might have gotten discussed in some retreat by a breakout group or something. Of course, we did talk about competition with Novell. Particularly I talked about it in the context of file server competition. You know, there was competition between Word and WordPerfect, no question about that. But the namespace extensions and Novell, not at all. You know, it was discussed and debated and argued and everything constantly, but it was all about the technical arguments and the fights between the Cairo team and the

Chicago team, that there were different goals, and on and on and on. It wasn't about Novell.

Q Mr. Muglia, I would like to show you one more exhibit, which is Defendant's Exhibit 77.

THE COURT: After that exhibit, we'll take a break.

MR. HOLLEY: Thank you, Your Honor.

BY MR. HOLLEY:

Q Mr. Muglia, this is an e-mail that Mr. Madigan wrote to Mr. Belfiore, you, Mr. Pederson, Frank Artale, and Lit Wong on the 9th of October of 1994. Mr. Madigan writes in the first paragraph, with the recent decision to base the Cairo shell on the Chicago shell code base, we have quite a bit of work to do beyond, quote, just porting, close quote, the Chicago shell. Then skipping, he says, by not publishing the Chicago explorer related interfaces, IShellView, IShellBrowser, this is made significantly easier.

Why would that be so?

A Well, as I said, I mean we needed -- we're going through this process. Ultimately the Cairo shell was not brought forward. The decision was made to use the Chicago shell in the Windows NT code base, which is what Steve was writing in this first sentence here. And as I said, you know, yes, that was true and we're going to use the Chicago code base, so that helped us in compatibility significantly.

But, you know, we were dealing with the fact that the darn thing was not robust. It was not architected to be robust. And we had those principles that Dave laid down, and we felt very important. I still think they were good principles. It was a good call.

And, you know, we had these interfaces that we needed to make the shell that we were putting on Windows NT, the Chicago shell being brought to Windows NT, we needed to make it as robust as possible. There was a lot we could do to make things better. But, boy, those IShellBrowser interfaces, those namespace extensions, they were so bad in their design that we just didn't know how we could implement anything that would be robust for those. So we were very grateful that we didn't have to implement them as they were originally designed. Particularly as they were originally designed. The design was changed at a later point to be more robust, still not fully robust, but better. But thank goodness that ISVs didn't write apps to the original design.

Q Now you have said a couple of times now that the design was changed at a later time --

THE COURT: Mr. Holley, would this be a good time to take a break?

MR. HOLLEY: Sure, Your Honor.

THE COURT: Take 15 minutes.

(Jury excused)

(Recess)