12/10/01
Reed
Exhibit No. 57
LISA A. MOREIRA

MEMO

LOTUS

002206

To: Frank King
Carl Young
Frank Ingari

From: David P. Reed

Subject: Windows and OEMs

Date: July 13, 1989

1 Today

The Intel-based PC operating systems environment is currently in chaos, resulting primarily from a major screwup on the part of IBM and Microsoft in the marketing of OS/2 Presentation Manager. As a result, we may have a significant opportunity to negotiate for what we want. A key to this is understanding what seems to be being proposed. Of course, we have also to decide what we want (which might be different from what we are being offerred as choices).

This note goes over IBM and Compaq's recent signals, and tries to draw conclusions about what we might be able to request and get from Microsoft, IBM, and Compaq.

2 DOS/Windows 3.0 Bundle

We have now heard through 3 independent channels that IBM, Compaq, and Microsoft are considering supplanting DOS with a bundle consisting of DOS and Windows 3.0.

In Compaq's case, this proposal (shared with those who attended their developers' conference) was joined with very strong condemnation of "OS/2 for the 286" and willingness to wait for "OS/2 for the 386" given the availability of Windows 3.0. (see Semmes Walsh's memo)

In IBM's case, the discussions with Tommy Steele have led us to conclude that IBM is somewhat more reluctant, but needs to see a strong argument from use as to what OS/2 gives us that Windows 3.0 does not for our applications. It is clear that they want answers that are definitive, not a "weight of technical evidence" argument. Though Tommy said he wanted us to join him to "defeat Windows in the market", he clearly is looking for a technical differentiation that is obvious to the user. (see Semmes Walsh's note on our phone conversation on 7/11). Were IBM to bundle Windows, it would potentially be viewed by them as a pre-emptive move against Compaq doing so.

One surprise in the IBM conversation was the notion that the bundle would have a subset of the OS/2 PM API instead of the Windows API. Depending on the subset chosen it might be easy or hard to port an application currently under development to Windows/DOS. On the other hand, if the subset is complete enough, it might be possible to build powerful Windows/DOS applications that port directly (maybe even binary compatibly) to OS/2. See last section.

DEFENDANT'S EXHIBIT 667

In the press, Infoworld and PC Week (and various columnists) have picked up the terminology "OS/2 for the 386" and "OS/2 for the 286" and condemned the latter, and encouraging the use of Windows and Windows 3.0 while waiting for the former in late 90/early 91. Some customers have also picked up the notion that Windows 3.0 is as good or better than OS/2 for GUI applications.

Microsoft has been looking for a way to establish Windows. My best guess is that they now have made a good argument to one or more OEMs that Windows 3.0 should be bundled with DOS as the standard, out-of-the-box-O/S for all PCs, since no one makes 8086 machines anymore, and all machines have 1 Meg. Carl, Semmes, and I think that maybe Compaq has swallowed the hook, and that IBM is scared not to follow (but also scared to admit its failure with OS/2).

Why this situation is different from the situation when Compaq bundled Win/386 with its 386 machines is that there are now Windows Apps from Microsoft about to come out the door. Also, of course, Windows 3.0 offers the *promise* of more powerful applications than did Windows 2.0, because of more memory when Windows 3.0 is run on a 386 machine.

3 Issues for Lotus

We have no GUI applications ready to go for Windows 3.0. The level of effort to convert 123/G would be half a team-year or more with the cost of converting to "handle-based memory management" being the bulk of it, but also the windowing environment, though similar in spirit, uses data structures and calls that differ in nearly *every* detail.

1-2-3 release 2.2 will work, but will look old and clunky in comparison with Excel. Similarly for Freelance and Graphwriter.

If a Windows 3.0 bundle replaces DOS, on 386 and 486 platforms, 1-2-3 release 3 will not work at all, due to the lack of protected mode support.

Agenda and Magellan will need to be rewritten for a GUI world, but would work as is in Windows. However, Magellan in its current form would be much less useful in a Windows/DOS bundle environment.

The core of 123/3 would port very easily to Windows, but it would need a lot of work to be a full Windows app (notably, mouse-based input, menus, dialog boxes, and also clipboard/DDE support). 1-2-3/3 would probably barely work on Windows/286 version 3.0, which has 700K of application space in which code and data can be swapped, and additional EMS bank-switched memory for data (we would hope that a combination of Windows-supplied overlay techniques and use of LIM EMS would help us in this environment). It would seem that a 1.5-2.0 Meg Windows/286 3.0 platform would run 1-2-3 release 3 about as well as a 1.0 Meg DOS/286 platform.

4 Differences between Windows 3 and OS/2 PM

OS/2 PM is here today. Windows 3.0 is in its first ISV test release, and still has significant bugs and missing features. Windows 2.10 is commercially available, but does not support applications over about 500K total size, except via very slow code overlay and expanded memory support.

Windows 3.0 takes "less memory" than OS/2. This is due to three things: lower function, integral DOS, and on the 386 platform, a paged virtual memory/virtual machine layer that runs "underneath" Windows 3.0. See below.

In the press, Infoworld and PC Week (and various columnists) have picked up the terminology "OS/2 for the 386" and "OS/2 for the 286" and condemned the latter, and encouraging the use of Windows and Windows 3.0 while waiting for the former in late 90/early 91. Some customers have also picked up the notion that Windows 3.0 is as good or better than OS/2 for GUI applications.

Microsoft has been looking for a way to establish Windows. My best guess is that they now have made a good argument to one or more OEMs that Windows 3.0 should be bundled with DOS as the standard, out-of-the-box-O/S for all PCs, since no one makes 8086 machines anymore, and all machines have 1 Meg. Carl, Semmes, and I think that maybe Compaq has swallowed the hook, and that IBM is scared not to follow (but also scared to admit its failure with OS/2).

Why this situation is different from the situation when Compaq bundled Win/386 with its 386 machines is that there are now Windows Apps from Microsoft about to come out the door. Also, of course, Windows 3.0 offers the *promise* of more powerful applications than did Windows 2.0, because of more memory when Windows 3.0 is run on a 386 machine.

3 Issues for Lotus

We have no GUI applications ready to go for Windows 3.0. The level of effort to convert 123/G would be half a team-year or more with the cost of converting to "handle-based memory management" being the bulk of it, but also the windowing environment, though similar in spirit, uses data structures and calls that differ in nearly *every* detail.

1-2-3 release 2.2 will work, but will look old and clunky in comparison with Excel. Similarly for Freelance and Graphwriter.

If a Windows 3.0 bundle replaces DOS, on 386 and 486 platforms, 1-2-3 release 3 will not work at all, due to the lack of protected mode support.

Agenda and Magellan will need to be rewritten for a GUI world, but would work as is in Windows. However, Magellan in its current form would be much less useful in a Windows/DOS bundle environment.

The core of 123/3 would port very easily to Windows, but it would need a lot of work to be a full Windows app (notably, mouse-based input, menus, dialog boxes, and also clipboard/DDE support). 1-2-3/3 would probably barely work on Windows/286 version 3.0, which has 700K of application space in which code and data can be swapped, and additional EMS bank-switched memory for data (we would hope that a combination of Windows-supplied overlay techniques and use of LIM EMS would help us in this environment). It would seem that a 1.5-2.0 Meg Windows/286 3.0 platform would run 1-2-3 release 3 about as well as a 1.0 Meg DOS/286 platform.

4 Differences between Windows 3 and OS/2 PM

OS/2 PM is here today. Windows 3.0 is in its first ISV test release, and still has significant bugs and missing features. Windows 2.10 is commercially available, but does not support applications over about 500K total size, except via very slow code overlay and expanded memory support.

Windows 3.0 takes "less memory" than OS/2. This is due to three things: lower function, integral DOS, and on the 386 platform, a paged virtual memory/virtual machine layer that runs "underneath" Windows 3.0. See below.

OS/2 PM has few broad market productivity applications, with CUI applications like Paradox, 1-2-3 Release 3, etc. predominant. There is no desktop publishing suite. Windows 3.0 has a broad spectrum of vaporware in the major productivity applications categories (most of the vaporware from Microsoft), and a suite of DTP applications and vertical applications of relatively small, but growing significance.

5 Less memory?

5.1 Lower function

The main weaknesses in Windows over OS/2 have to do with multithread and multiprocess applications.

OS/2 provides preemptive multiprocessing, which dramatically reduces the probablity that one task will prevent another from running.

OS/2 provides for controlled sharing of memory between threads and processes, which allows for high performance in communications-based applications such as LAN applications, DBMS applications, etc.

OS/2 provides for inter-application file locking, so that applications contending for the same files will not accidentally step on each other.

OS/2 makes sure that if an application follows an invalid pointer and fails, the other applications in the system will normally proceed correctly.

Windows 3.0 provides non-preemptive multiprocessing, which means that applications must be carefully designed to give up contro frequently (including releasing pointers) if other applications are to be able to respond to events (such as communications events or other external events).

Windows 3.0 places parts of all applications in one address space, and other parts in inaccessible regions (essentially apps are all divided into resident and transient parts, and all resident parts are present in memory at all times, while transient parts are swapped in and out of the address space when applications receive control). This doesn't allow for much memory protection, or predictable sharing.

Robustness of Windows 3.0 in the face of application bugs is clearly much worse that OS/2 because of lower function in the memory management and processor scheduling areas.

5.2 Integral DOS

Windows 3.0 builds on DOS and the BIOS, using as much as possible of the file system, networking, and memory management as possible. Thus it is quite easy to run DOS applications -- Windows just gets "out of the way" while the DOS app is running. The major trick is to suspend and resume a DOS application, which requires saving and restoring its memory image, screen state, and BIOS/DOS state variables. The save and restore operation on a 286 is tricky only because of "write-only" registers in the EGA adapter, but Windows/286 manages to handle this with heuristics. On a 386, Windows 3 uses the VM8086 mode of the i386 processor to intercept all EGA port accesses, etc. This makes it possible to have multiple DOS boxes, and even to window their screen access.

In OS/2, by contrast, DOS must be simulated by reserving 640K of memory that cannot be used by OS/2, and emulating DOS functions through protected mode code that duplicates the DOS APIs. Obvioulsy this is an imperfect DOS simulation, but more important, it takes about a Meg of memory to provide this function, even though most DOS apps do not use so much. So chalk up 1 Meg of the memory cost of OS/2 to DOS emulation.

5.3 Virtual Machine/Virtual Memory

Windows 3.0 is built in two layers. The lower layer is a machine with a linear address space, the first 1 MB of which contains a DOS address space. On top of this is laid a segmented address space with segments of 1-64K bytes. On the 386, this is done with the paging and virtual machine capability of the 386. The result is significant reduction in the amount of memory that must be consumed to handle little used operating system code, inactive parts of applications, etc. when those are swapped to disk.

In contrast, OS/2 was designed without using any 386 features. It swaps segments, but in fact the operating system seems not to be optimized to minimize memory use by organizing code and data in smaller segments that are organized by usage frequency. The speed of OS/2 is also affected because when memory fills up, segments are reorganized rather than swapped by copying them around to fill up gaps in memory. As memory gets signiuficantly committed, this gets very costly, and I don't believe the proper tuning has been designed.

NOTE HERE: There would seem to be no reason at all why OS/2 could not be made to work on top of the Windows 3.0-style paged-linear-address-space with virtual-DOS-machine bottom layer. This would seem extremely simple, and would make OS/2 smaller and faster on 386 machines.and allow windowing of DOS apps on 386 machines. Instead, IBM and Microsoft have launched the folly of major enhancements to the file system and other significant API changes that will require significant recoding of OS/2 1.1 applications in progress before the market has any applications that need them. (personally, the author cynically believes that this single fact indicates Microsoft's commitment to OS/2's success in the user market, as opposed to its commitment to extracting $'s from IBM).

6 Weaknesses

Weaknesses of the Windows/DOS bundle are mostly in what it interferes with. That is, what you cannot do in the Windows/DOS bundle environment that you can do in either DOS or OS/2 PM easily will be where this environment comes up short.

Applications that use 16-bit DOS extenders that run under DOS today will not run unmodified in this new environment on 386 and 486 machines (they do work on 286 machines, though with reduced amounts of available memory.) Most notable is 1-2-3 release 3.

Applications that use Phar Lap, etc. to use "full 32-bit mode" of the 386 chip will not work in OS/2, and they will not work in Windows 3.0. These applications (though minor) do work today in DOS 3 and DOS 4. Both run in 16-bit protected mode. It is possible that with a lot of work by the DOS Extender crowd, one could write true 32-bit applications. But more likely, the 32-bit linear addressing mode would have to wait till 1991 when "386 OS/2" or "Window 4.0" are fabled to be first shown to ISV's.

In OS/2, by contrast, DOS must be simulated by reserving 640K of memory that cannot be used by OS/2, and emulating DOS functions through protected mode code that duplicates the DOS APIs. Obvioulsy this is an imperfect DOS simulation, but more important, it takes about a Meg of memory to provide this function, even though most DOS apps do not use so much. So chalk up 1 Meg of the memory cost of OS/2 to DOS emulation.

5.3 Virtual Machine/Virtual Memory

Windows 3.0 is built in two layers. The lower layer is a machine with a linear address space, the first 1 MB of which contains a DOS address space. On top of this is laid a segmented address space with segments of 1-64K bytes. On the 386, this is done with the paging and virtual machine capability of the 386. The result is significant reduction in the amount of memory that must be consumed to handle little used operating system code, inactive parts of applications, etc. when those are swapped to disk.

In contrast, OS/2 was designed without using any 386 features. It swaps segments, but in fact the operating system seems not to be optimized to minimize memory use by organizing code and data in smaller segments that are organized by usage frequency. The speed of OS/2 is also affected because when memory fills up, segments are reorganized rather than swapped by copying them around to fill up gaps in memory. As memory gets signiuficantly committed, this gets very costly, and I don't believe the proper tuning has been designed.

NOTE HERE: There would seem to be no reason at all why OS/2 could not be made to work on top of the Windows 3.0-style paged-linear-address-space with virtual-DOS-machine bottom layer. This would seem extremely simple, and would make OS/2 smaller and faster on 386 machines.and allow windowing of DOS apps on 386 machines. Instead, IBM and Microsoft have launched the folly of major enhancements to the file system and other significant API changes that will require significant recoding of OS/2 1.1 applications in progress before the market has any applications that need them. (personally, the author cynically believes that this single fact indicates Microsoft's commitment to OS/2's success in the user market, as opposed to its commitment to extracting $'s from IBM).

6 Weaknesses

Weaknesses of the Windows/DOS bundle are mostly in what it interferes with. That is, what you cannot do in the Windows/DOS bundle environment that you can do in either DOS or OS/2 PM easily will be where this environment comes up short.

Applications that use 16-bit DOS extenders that run under DOS today will not run unmodified in this new environment on 386 and 486 machines (they do work on 286 machines, though with reduced amounts of available memory.) Most notable is 1-2-3 release 3.

Applications that use Phar Lap, etc. to use "full 32-bit mode" of the 386 chip will not work in OS/2, and they will not work in Windows 3.0. These applications (though minor) do work today in DOS 3 and DOS 4. Both run in 16-bit protected mode. It is possible that with a lot of work by the DOS Extender crowd, one could write true 32-bit applications. But more likely, the 32-bit linear addressing mode would have to wait till 1991 when "386 OS/2" or "Window 4.0" are fabled to be first shown to ISV's.

Windows/DOS would have the full Windows print services for Windows apps. which allows support of graphic drawing primitives, Postscript and PCL laser printers, a clipboard, and TIFF file support, for example. CUI applications on the same machine would benefit little from this support, and could not easily use the Windows/DOS queueing, imaging, and clipboard facilities. In fact, applications that use these facilities in imaginative ways in DOS might have a hard time cooperating with Windows use of the printer.

Windows/DOS does not support networking well at all. In particular, multiple apps that communicate through virtual circuits will not work well in a non-preemptive environment. One cannot build robust network apps such as email apps when a single failure in one application can clobber the memory of others; or halt others in unpredictable ways. THis is not to say that OS/2 is the best, but it is at least as good as most UNIX systems, if not better in many ways (such as supporting better timeout, multi-threading, and IPC mechanisms).

The form of interapplication communication and processor multiplexing provided by Windows is not very good for building shared database applications where the server is either a server node on the network, or a background process on the local machine. This would make things like an application that maintains a "real-time" database of stock prices hard to build because it would fight for resources with the application accessing it or other unrelated applications, and might miss trades, etc.

7 What to do?

Obviously begin work that presumes that Windows/DOS of some form will be the next DOS available around 1/1/90. We need to hedge the major incursion that Microsoft applications can make if this happens.

We need to think about what this means to 123/G. If it succeeds in killing a large part of the drive to OS/2 (the desire for multiple, windowed, GUI apps), this move will kill 123/G's market just as it ships.

Having gotten over planning for the worst cases, we need to consider the weaknesses of Windows/DOS as fault lines for carving what we want out of the situation we see in this area.

These weaknesses can be used in several ways to support actions.

First, we can require that they be fixed before Windows becomes an important platform for us. This is a task that is technically infeasible, but can be used to delay and improve the results as much as possible in areas such as OS/2 compatibility, compatibility with our CUI apps by making it possible for them to run in Windows and use Windows print services, support for 32-bit linear address spaces, better network file system and network management support, and better robustness. Some of these would be more beneficial to us than others, depending on what products we choose to port.

Second, we could use them more clearly to define the differentiation between DOS/Windows and OS/2 with PM, by pointing out what the differences prevent us from doing in our DOS applications. This is a hard sell, because all of the weaknesses tend to be "behind-the-scenes" issues that have only indirect effects on the applications environment the user sees. The screens can look identical, and the same demos can be shown on either platform. This is an argment we can only win if the customers are highly technical (like other developers, the software staff of IBM and Compaq, etc.)

My preferred idea today is to acknowledge frankly the reality that many customers see little difference between Windows and OS/2 PM, and in fact see Windows as having a better cost, memory footprint, and suite of applications. If this is so, perhaps the best use of our time is to use Windows 3.0/DOS bundling as a "proof-of-concept" that you can make a success of OS/2 by downsizing, repricing, and incenting developers to go for it by whatever means possible. The OEMs and some of the major ISV's can see the technical limitations and disadvantages of the "alternate road" that Windows/DOS bundle opens up. Thus, they should be receptive to a plan that calls for reduction (not enhancement) of the OS/2 PM API to a mini-API subset, performance tuning of the system aimed at reducing memory footprint, and possibly including the 386/VM layer from Windows 3.0. One goal that might be achieved is that by using some tricks, one could run in a typical 1 Meg machine any one DOS application, by swapping out as much of the O/S as possible, and resuiring that the DOS application terminate before letting the O/S resume.

Repricing this "personal OS/2" platform to be competitive with a DOS/Windows bundle would eliminate a cost.

I think this latter course is the currently preferred outcome. I would sweeten the goal only by pusing for a speedy inclusion of pure 32-bit application-only support (the 386 OS/2 plan seems to be so late because it is a complete rewrite from assembler to C so that the kernel will run in pure-32-bit mode also). The most direct path to this result could get Intel on our side quickly.

The benefits to this path are mostly obvious, but also it is worth noting that OS/2 is strenghtened, and our efforts on GUI can be focussed on one major platform.

My preferred idea today is to acknowledge frankly the reality that many customers see little difference between Windows and OS/2 PM, and in fact see Windows as having a better cost, memory footprint, and suite of applications. If this is so, perhaps the best use of our time is to use Windows 3.0/DOS bundling as a "proof-of-concept" that you can make a success of OS/2 by downsizing, repricing, and incenting developers to go for it by whatever means possible. The OEMs and some of the major ISV's can see the technical limitations and disadvantages of the "alternate road" that Windows/DOS bundle opens up. Thus, they should be receptive to a plan that calls for reduction (not enhancement) of the OS/2 PM API to a mini-API subset, performance tuning of the system aimed at reducing memory footprint, and possibly including the 386/VM layer from Windows 3.0. One goal that might be achieved is that by using some tricks, one could run in a typical 1 Meg machine any one DOS application, by swapping out as much of the O/S as possible, and resuiring that the DOS application terminate before letting the O/S resume.

Repricing this "personal OS/2" platform to be competitive with a DOS/Windows bundle would eliminate a cost.

I think this latter course is the currently preferred outcome. I would sweeten the goal only by pusing for a speedy inclusion of pure 32-bit application-only support (the 386 OS/2 plan seems to be so late because it is a complete rewrite from assembler to C so that the kernel will run in pure-32-bit mode also). The most direct path to this result could get Intel on our side quickly.

The benefits to this path are mostly obvious, but also it is worth noting that OS/2 is strenghtened, and our efforts on GUI can be focussed on one major platform.