Microsoft Corporation
One Microsoft Way
Redmond, WA 98052-6399
Tel 206 882 8080
Telex 160520
Fax 206 883 8101

PLAINTIFF'S EXHIBIT 1019 Comes v. Microsoft

To: ABumsoss, Edf
Chrisg, V2: please coordinate a response to the request ("The chart") on the last page.
thx, Pete

Microsoft Memo

To: Mike Maples, Darryl Rubin
From: Bill Gates
Date: October 14, 1991
cc: Jeff Raikes, Pete Higgins, Aaron Getz, Greg Whitten, Nathan Myhrvold, Jim Allchin
Subject: Applications Strategy

MIPS

While I can't guarantee high volumes, we have decided to take the risk and support the MIPS platform. I hope that we are in good shape with our applications. I understand that Excel will ship for MIPS in '92. I hope that PowerPoint, Project, and Visual Basic will also ship in that time frame. I'm worried if Word doesn't ship till 1993.

I am still a little unclear if we are going to have a common source code to achieve this. What is the current plan for the number of .EXEs we are expecting to have for our applications? My current understanding is that unless the 32-bit version is quite a bit faster, we are planning to have a 286 version compiled with the Microsoft C compiler not using much pcode, a 32-bit version for Windows NT on Intel, and a 32-bit version for Windows NT on MIPS. If the application considers the 32-bit speed particular important versus 286 compatibility, they can use the subset of Win32 to create a fast version for Windows DOS on the 386.

Mac Alignment

The products which do not already have an aligned version out of the Mac should look very hard at switching to AFX as part of their strategy, instead of trying to create the Mac version in another way. Cirrus, Mail and Works should be moved on top of AFX as soon as we can. This will enable us to get the other benefits as well. If this is unrealistic, I want to understand why.

It is interesting to contemplate the future of the Mac. It appears that the port to RISC is going to take precedence over new features for the Mac. We certainly don't know much about System 8. The lack of new features makes it easy for us to exploit the Mac while we are supporting Windows 3. Is the implementation of OLE-2 on the Macintosh well understood?

Once we get to Windows 4, we will have a real dilemma. We won't be able to create an application that really exploits Windows 4 and still have an aligned product on the Mac. We will not be able to duplicate the Win4 functionality on the Mac. However, the only apps with really hard-core exploitation of Win4 are the Workgroup applications.

Sound

Sound hardware will be much more widespread over the next few years. We need to decide whether our applications should do anything more than just support OLE? For example, should our applications play sounds when different events occur?

We should describe the legal set of inputs at different places in the interface in a tighter way, so that we are ready for sound input. This idea of specifying input sets very strictly is a common point for pen input and sound input. The research work done in the Jensen/Heidorn group (Natural Language Recognition) might help us with input specifications. Marlin should probably spend time with them on this topic.

Pen

I feel we are not doing very much to exploit the pen. Carr's book on PenPoint makes some excellent points about how applications should work differently if a pen is available. Certainly our drawing, equation, and publishing applications need to change. Our lack of good annotation capabilities really shows up in the pen environment. I am worried that a competitor could use pen to our disadvantage. I want to see a memo discussing great note taking features and assessing where they belong in our products.

CD Versions

I would like to see us have our applications shipped on CD, along with a good deal of value-added material and the documentation. If our viewer work makes progress our user education groups will be authoring on-line and print based material electronically, so that delivering it on a CD will be reasonable. Word is now taking 15 megabytes of disk space with all of its features and customers will welcome using CDs to get the latest version to their network. I don't know how to prioritize value added materials versus just getting the applications available.

Group Environment

Another area where I am afraid we are behind the competition is in workgroup use of our productivity applications. AaronG wrote up some common scenarios for group work recently. This is valuable reading for people considering the topic. I think Workgroup Applications is thinking about these issues and will be able to build some wonderful Win4 applications. I worry that the mainstream applications will not be rich enough in this area. Group work is more than multiple people trying to change the same thing at the same time. That is only one of the scenarios described in Aaron's memo. Darryl's recent memo on annotation is a fine discussion of an important workgroup scenario.

Sharing Code

Our lack of sharing a common long text control for handling comment like text is creating a real problem. I was unhappy that text fields in Excel annotations, Excel text boxes, Project annotations, Mail text fields, and Visual Basic text fields do not have important features because they are not shared work. Do we have a concrete plan to fix this?

Our lack of sharing tables objects is also a real problem. Word and Excel have different features. It is nonsense to say that they are serving different needs. The idea of merging cells together is a great one that Word supports well and Excel needs to support. The whole justification, layout model within cells in Excel is getting more complicated, partly as a result of not sharing table concepts. I think we will wind up adding tables to PowerPoint, Notebook, Publisher and perhaps even Chart or Draw and that is going to make to make the situation

worse. I think Excel 5 and WinWord 3 can bring tables closer together and support better interchange, but it will be the generation after that when we will use code sharing to get commonality.

Our lack of sharing expression evaluation work is another real problem. Look at the immense amount of work going into doing engineering and scientific extensions to Excel. Why shouldn't our language users (Visual Basic and C) get that same functionality? Why shouldn't the formatting capabilities in our languages match Excel pictures? I think that Object Basic should bring these things together. I am concerned about Object Basic in general. I asked to have a spec to read for Think Week and was told that there was none. Has MacroManager become that much of a distraction?

Object Oriented Project

I firmly believe that we should create an object-oriented, page-oriented application that has the functionality of Publisher, Draw, PowerPoint and the Notebook through a from-scratch effort using AFX/Composer. I am not sure whether Works should be considered part of this effort or not. Probably it shouldn't be, since the initial four is already a lot to encompass. Is this what Bob Cook is working on? Is he alone? I am willing to fully staff this effort, but the ideal is to leverage off of the Composer work to get it to the point where people believe Composer will work well. Then we should bring in engineers from one or several of the development groups. This application should demonstrate that OLE 2 command exposure and great internationalization support comes largely for free if the right tools are being used.

OLE 2/Win 4

I am very concerned that we don't understand this work well enough. We don't know what the user interface looks like. We don't know how much benefit will accrue to applications.

Ideally, we want to have a clearly defined set of work for taking advantage of OLE 2 and Win4. If this work is done, applications and users will get many great and obvious benefits. We would create a clear time frame for this work and make sure that applications make this a priority. This will have a similar affect to Windows 3. Our application group can help users, systems and itself all at the same time, by really taking advantage early of exciting new system features (without any preferential treatment, just the right focus and execution).

In regards to the requirements for OLE 2, I think changing all file references to be object references is very desirable and not very hard. I think changing embedded object handling to use the Docfile subroutines is also very desirable and not that hard. I'm not sure what the user interface work will be, because I still don't think that we have figured it out. There will be more drag and drop, the current plan for in-situ editing is still bothering me. I was impressed that GO was able to support in-situ fairly cleanly by forcing people to use their framework, but I still have not figured out how they avoided all of the other hard user interface problems with in-situ editing.

I think exposing commands may be hard. Partly this is because our goal is that people should prefer to use the generalized macro language rather than the application-specific language for any new macros. Maybe this is not achievable? Is the IDT work separate from the Command

Exposure work? Do we have an object model for the state of our key applications written down? Jim Allchin is asking people to define interfaces and run them through the NIDL compiler before publishing them. I would like to see this for the command interfaces, at least.

Do you think great implementations of all of the OLE 2 features will be a major competitive edge for us? I think it will be if we come up with more interesting servers, get third party developers using this stuff and integrate the external command handling. I want us to have as much advantage over a non-OLE 2 application as we had over a non-Windows application. I know this is asking for a lot.

I think the work for Win4 will be less than that for OLE 2. Each application will have to decide whether to do this as one effort or as two. Windows 4 requires applications to expose properties and support long names. It also requires some interface clean up. Certainly installation will change and pop-ups will become uniform. The clipboard will go away and become just a default place that something is moved to, rather than special commands. Delete and Move (which are cut today) will need to be separated.

Basic

My current thinking is that MacroManager should be bundled with the system and that our basic language should not. I don't think it is worth the distraction to make embedded Basic available before we do the OLE 2 work in our applications. We should get focused on building IDTs that are competitive with embedded languages as soon as possible. As long as other ISVs don't all cluster around a single solution such as Softbridge or Borland, and as long as we can execute well, we can decide to license the technology to other companies after we get the feature. I don't think the language technology will ever be cheap to ISVs. I think we will eventually have a solution that small ISVs will like, which will be based on a royalty scheme. The architecture for common commands should be available to everyone.

I don't think we should make objects like charting part of the free runtimes of our development tools. Already I hear cases where people who would have bought Excel for all of their users decide instead to write a VB application. This means we get nothing for each user. This does not mean that we can't ship the functionality with the development tool. But if we do, we have to make clear it can't be shipped along with generated applications.

The Chart

I would like to see what the work plans and schedules are for all of our application products in the following areas. (I want to know in what versions of each package what level of support will be offered.)

- 32-bit versions / MIPS Support
- Mac Alignment
- Sound Support
- Pen Support
- CD Versions
- Work Group Uses

- OLE 2 Command Architecture
- OLE 2 User Interface
- Win 4 Support