From: Paul Johnson
To: Microsoft ATR
Date: 1/16/02 7:58am
Subject: Microsoft Settlement

Introduction
============

I write to comment on the proposed settlement between the US Department of Justice and Microsoft (the Proposal). I believe that the Proposal makes progress in the right direction, but does not go far enough.

I am a British subject, and must therefore beg forgiveness for intruding on a case being heard in the United States. However Microsoft is a multinational company, and holds a similar market position in both the US and Britain. If it is implemented then the Proposal will have essentially the same impact on both our countries. Finally I note that the European Union is considering its own action to curb the Microsoft monopoly, but is awaiting the resolution of the US case. I therefore feel that I have a legitimate interest in the outcome of this case, and submit my comments accordingly.

I hold a degree in Computing Systems from University College Cardiff (now part of the University of Cardiff). I have been a practicing computer scientist ever since. I have been following the Microsoft case with great interest since it started.

Summary
=======

* Microsoft holds a dominant position throughout the software industry. A remedy which deals exclusively with "middleware" is not sufficient. All Microsoft software should be covered. Microsoft is a repeat offender and there is no reason to think that this case has changed its character, so remedies must be carefully drafted to avoid leaving loopholes.

* There should be no restrictions on pricing or product tying. Microsoft should be left free to develop and sell its products as it sees fit. The only exception to this are the rules which cover OEMs ability to include competing products instead of Microsoft ones.

* Microsoft's monopoly position is founded on its control of proprietary interfaces. Microsoft products are linked through a network of proprietary interfaces, making it difficult for competitors to produce software that will inter-operate with Microsoft software. If the proprietary interfaces were published then competitors could produce software that competed directly with Microsoft without the expensive and error-prone process of reverse engineering.

* These proprietary interfaces are in the form of file formats, network protocols and APIs. All three need to be made available to competing products.

* Where two Microsoft products work together the interface between them can best be made available by setting up a "Chinese wall" between the development groups responsible for them, and then requiring Microsoft to publish all the technical data that is exchanged between these groups.

* Where one copy of a product communicates with other copies of the same product (such as when an MS word document is sent to another MS Word user) the file format or communication protocol should be published in a form which allows independent verification that the product conforms to the published description.

* Special consideration should be taken of Open Source Software development over the questions of cost, trade secret status and patent licensing.

* The "security related" exception to disclosure should be narrowed to include only keys, passwords and similar security tokens.

Microsoft's Position

Microsoft currently holds a dominant position in the computer software industry, and as I shall show below it maintains this position through control of proprietary interfaces.

I believe that a fair and effective remedy should destroy the competitive advantage that Microsoft gains through its control of interfaces, but still allow it to compete and innovate on equal terms with its competitors.

(In the longer term I would suggest that legislation be created to require all software companies above a certain size to publish the details of their interfaces, and thereby create a truly level playing field in the software market. However that is not the subject of this note.)

Over the past decade Microsoft has repeatedly demonstrated a willingness to evade or ignore regulations aimed at curbing its monopoly power. There is no reason to expect this behaviour to change. Therefore any effective remedy must be drafted to block not only the past misdeeds of Microsoft but any it might devise in the future. The rules under which Microsoft is to operate must be unambiguous and, as far as possible, free from the need to make value judgements as to whether Microsoft has fulfiled its obligations sufficiently.

Product Tying

The current case was originally concerned with the alleged tying of Microsoft Internet Explorer with Windows 95, in violation of anti-trust law. However the list of features which users expect to find in an operating system has evolved over time, and continues to do so. A previous example concerns "disk defragmenters", which optimise the arrangement of data on a disk in order to speed up access. Before Windows 95 these programs were sold separately by competitors to Microsoft. When Windows 95 was released it included a disk defragmenter. The competing companies could no longer sell their existing products, but there was no public outcry because disk defragmentation is generally considered to be a function of the operating system. Similarly when Microsoft first bundled Internet Explorer with Windows a web browser was considered an application. Today most consumers would expect to find it bundled with the operating system.

Suppose that ten years ago Microsoft had been effectively prevented from adding new features to Windows: today a modern PC would have to include a dozen or more small packages of software that would be more economically produced and sold as a single product. Computer vendors would have to purchase and integrate all of these small packages, and buyers would have to cope with a bewildering checklist of small but important items that they would have to ensure their computer included. The variation in quality and operation of these packages would present serious challenges to the use and maintenance of different PCs. Such a situation would not be in the interests of the consumer.

Thus a fair and effective remedy cannot enjoin Microsoft from ever bundling new functionality in its products, even when a market for that functionality already exists and is serviced by third party products.

US anti-trust law deals with this point by requiring that product tying of this sort be of benefit to consumers, and prohibiting predatory pricing. However this principle is of little help in the software market:

* The "benefit to consumers" test largely fails because if Microsoft adds a feature to a package then it saves the consumer the trouble of buying and installing extra software to provide that feature. Thus a strong argument can always be made in favour of any particular feature being added.

* There is no "fair" price for software in the sense that there is for physical products (i.e. the unit cost plus a reasonable profit) because there is no unit cost. The cost of software is entirely in its original development. Left to themselves software vendors will set a price which maximises their income, but there is no link between this price and the cost of development. Any plan to regulate Microsoft by imposing fair prices must therefore remove entirely its right to set its own prices, and this in turn will require it to negotiate a price for software with the regulator before starting development. This is highly unlikely to benefit consumers. But if Microsoft is free to set prices, even to set them at zero, then it can effectively tie products by distributing free add-ons at the point of sale.

Therefore I must reluctantly conclude that regulating Microsoft's ability to tie products is likely to do more harm than good, and should not be included in the final remedy. Microsoft should be left free to determine what functionality is included in each of its products.

The Proposal also sets rules for the related issue of the "Desktop". This properly prevents Microsoft from ensuring that its products are more prominent on the desktop than those of its competitors. Such user interface concerns are important, but are not the subject of this note.

Interfaces

The Proposal concentrates on the "Application Programmer Interfaces" (APIs) to Microsoft "Middleware" (a vaguely defined term, roughly meaning software that sits between the operating system and the applications employed by end users).

The Proposal is right to concentrate on interfaces. Microsoft has always used proprietary interfaces to manipulate the market and lock out competition. To illustrate how this works, suppose Microsoft sells products Foo and Bar that communicate via a proprietary interface. I purchase Foo, and subsequently want the added functionality of Bar. There may be many competitors in the market for Bar, but they are effectively excluded from my consideration because their products cannot communicate with Foo.

Similarly if copies of Foo communicate with each other through a proprietary interface then anyone wishing to work with me must also purchase a copy of Foo. This creates a "network externality" that ensures that even in a competitive market the best option for an individual consumer is the product with the largest market share, since this brings them into the largest population of potential collaborators.

By creating a web of proprietary interfaces, both between products and between its customers, Microsoft has ensured that it is locked into its market in a way that has never before been possible. It is this

stranglehold on the market for software that must be broken. Since Microsoft has used its control of proprietary interfaces to achieve this, it is on interfaces that any effective remedy must concentrate.

The focus of the Proposal on "middleware" is misguided. It excludes applications and operating systems, which are the two areas where the monopoly power of Microsoft most needs to be restricted. Furthermore its vague definition creates too much opportunity for Microsoft to redefine critical interfaces as something other than "middleware", leading at best to argument and delay.

Examples

It is worth looking at two of these interfaces to see how they lock Microsoft into the market.

* Microsoft Office is the leading "office productivity suite". There are competitors, but they are critically hampered because their users cannot reliably exchange documents with MS Office users. Some degree of inter-operability does exist, but this has been enabled by painstaking "reverse engineering": the competitor can only learn about document formats by inspecting the files created by Office and trying to deduce how each part of the document is encoded in the file. This process is expensive and error-prone, and Microsoft can always introduce new features faster than they can be reverse engineered. As a result no existing competitor to Office can reliably import a complex document. Consumers know this, and therefore avoid these competitors. This prevents the competitors from gaining market share, no matter how good their products might otherwise be.

* The Kerberos security protocol was developed by MIT and has now become an important component of many systems. Microsoft included Kerberos support in Windows 2000, but with a small change. Kerberos is an "authentication" protocol: it guarantees that the parties to a transaction are who they say they are. Microsoft added authorisation data to the protocol. This meant that Windows 2000 would only grant access to shared files and printers if the Kerberos "ticket" presented by the user had been issued by a Windows 2000 server. This appears to have been an attempt to lock competitors (including the freely available MIT server) out of the market for Kerberos authentication products. In response to a public outcry within the computer industry Microsoft first insisted that the format of its extra data was a trade secret, and then released the format on its web site under a "click-through" license under which the recipient promised to keep its contents a secret. I will return to this strange license later in the section on Open Source Software.

The net effect of this web of proprietary interfaces is to make any mix of Microsoft and competing products less functional than a pure Microsoft solution. A pure non-Microsoft solution is not usually possible, either because Microsoft has driven the competition into the ground or because there is a need to communicate with others who are using Microsoft. Hence the only choice is between a pure Microsoft solution and a mix. In a world which is dominated by Microsoft there can only be level competition if the interfaces to Microsoft software are equally open to all competitors.

Files, Protocols and APIs

There are three types of interface which an effective remedy must address: files, network protocols, and APIs.

Files stored on disk are an important repository of value for any computer user. The ability to read this data and exchange it with

others is the most important requirement for any new software. Therefore Microsoft shoud be required to disclose the file formats for all its software. This will enable competitors to create software which reliably works with files created by Microsoft software. The main immediate effect of this will be to enable competitors of Microsoft Office to compete on a level playing field. In the longer term it will prevent Microsoft from using the proprietary file format of any popular application to gain a monopoly position through market lock-in.

Similarly, protocols used to communicate over networks should be opened up. The Kerberos example above illustrates how even seemingly minor proprietary extensions can create strong market lock-in. As the Internet becomes increasingly important so the use of proprietary protocols will become an important method for Microsoft to maintain its monopoly position unless it is stopped.

APIs are a much more complicated issue than files and protocols. For every file format or network protocol used by Microsoft there are thousands of "function calls", the basic element of APIs. Function calls are used both within a single product and between products. There is no simple way to distinguish the function calls which are made within a product and those made between products unless the products in question are designed to work separately as well as together. Microsoft has already used this fact to obfuscate the question of whether Internet Explorer is intrinsically integrated with Windows 95. It can be expected to use this tactic again in the future. Since it is not feasible to use product tying rules to prevent this (see above), I suggest that Microsoft be required to identify every API which is used to communicate between software in two different products, and disclose that API in full. The smallest unit of "API" to be disclosed should be the "DLL" (Dynamically Linked library). In Windows a DLL is a single file which provides collection of functions to other software. Making DLLs atomic for disclosure purposes will encourage Microsoft to keep the APIs for communication between products distinct from the APIs within products, thereby reducing the work required by competitors who wish to offer competing products which offer the same APIs.

Disclosure Mechanisms

Detail

The Proposal has nothing to say about what level of detail will be included in the interface descriptions. This issue is not trivial.

For programmers, the ultimate description of what a function within an API does is the source code which implements that function, which leads programmers to say "use the Source, Luke" when faced with a detailed technical query about a piece of sotware.

However the inspection of source code is not always practical, either because the code in question is proprietary (as in this case), or just because it would take too long to understand. Hence developers routinely produce documentation which describes the functions in an API in a more readable form.

The Proposal seems to envisage this kind of documentation being made publicly available. However there does not appear to be any incentive to Microsoft to make this documentation complete or accurate, other than enforcement by the courts. Since this kind of document can never be 100% complete or accurate the question will arise as to whether it is good enough. If Microsoft acts true to form it will inevitably argue that its documentation is indeed good enough even though it is not, and will carry on arguing this until it becomes a moot point.

To avoid this problem I suggest that Microsoft be required to erect "Chinese walls" between the development groups working on different products. Only published documentation may be exchanged between these groups. Hence if Microsoft wishes to sell two products which work together it can only do so if it also informs its competitors how to make products which will can work just as effectively.

The remaining problem on detail is the file formats and protocols used when one copy of a product communicates with other copies of the same product. The Chinese wall system will not work here. However since this problem is restricted to file formats and protocols the problem of ensuring the adequacy of documentation is much smaller. Established techniques (such as BNF grammars and state machines) can completely describe file formats and protocols, and these can be used as the basis of an unarguable technical finding that either the software or the documentation is defective. This is not a complete solution to the problem, but it should level the playing field sufficiently to allow competition.

Publication and Open Source

Since this case started Open Source Software (OSS), such as the Linux operating system, has become a significant competitor to Microsoft. Therefore any effective remedy must take account of the special requirements of OSS development over normal commercial software development. The primary issues here are costs, trade secrets, and patents.

Costs:

Whatever disclosure mechanism is chosen for interface descriptions, it must be within the financial reach of open source developers. A subscription of several hundred dollars a year, such as is required for the Microsoft Developer Network, is trivial for a competing software company but a major hurdle for a volunteer developer working on OSS. Given that interface descriptions must be prepared for competitors, there is no reason why they should not be distributed for free over the web rather than only made available to an exclusive club.

Trade Secrets:

Microsoft must not be allowed to pretend that these interface descriptions are trade secrets, as it tried to do with its extension to Kerberos. Because OSS packages include the full source code they inevitabley reveal the full details of their operation to any programmer who downloads them. If Microsoft can claim trade secret status on an interface it can effectively block any OSS package from using that interface, since to do so would reveal the "secret" of its operation. This appears to have been the objective of the click-through license on the Kerberos extensions (see above). The "Samba" project (www.samba.org) has reverse-engineered the Microsoft file and printer sharing protocols, allowing non-Microsoft systems to gain access to resources on Microsoft systems. An updated version of Samba for Windows 2000 is being prepared which will need to inter-operate with the Windows 2000 Kerberos extensions. If these extensions are considered trade secrets then it would be impossible for the Samba project to work with these extensions, and a key component in any mix of Microsoft and non-Microsoft computers would be crippled.

Patents:

Microsoft has not made much use of patents to protect its market, prefering to rely on proprietary interfaces. However if it is prevented from using proprietary interfaces it may decide to use patented ones instead.

When Microsoft next introduces a new interface, especially a network protocol, it would be a simple matter to obtain a patent covering the operation of that interface. At that point any competitor wishing to inter-operate with Microsoft products using that interface would have to license it from Microsoft. The usual solution in such situations is to require licenses on "Reasonable And Non-Discriminatory" (RAND) terms. However even RAND terms require payment. OSS developers are unable to offer payment. Therefore an effective remedy must require Microsoft to license its patents on RAND terms to commercial software vendors and on Royalty Free terms to Open Source projects.

Incidentally, Microsoft has described OSS as "un-American" and "an intellectual property destroyer". These descriptions try to tar OSS developers with the same brush as software pirates. This is incorrect. Software pirates selfishly take the work of others and use it without paying. OSS developers take their own work and permit others to use it for free. This is a wholly generous act, fully in keeping with the American ideals of volunteerism and service to one's community.

Security Details

The Proposal includes a broad exception for "security related" information. However Microsoft could argue that almost any interface, especially APIs and communication protocols, is "security related" if it is used to carry any kind of authorisation or authentication information. Indeed, it made exactly this argument when it initally refused to reveal its extensions to Kerberos. Therefore the exception for security related information must be narrowly drawn.

Fortunately this is not a major problem. It is a basic principle of computer security that would-be intruders will eventually learn the operational details of your security mechanism, either by reverse engineering or by other less legitmate means. Any security which depends on the intruders remaining ignorant of these details is known as "security through obscurity", and regarded by security practitioners as inadequate at best. Therefore the only items which should need to be kept secure are the keys or passwords which operate the software. These can be easily changed if they are compromised. Hence if security interfaces are well designed then they will not need to be kept secret. And if they are not well designed then Microsoft should be required to remedy the fault rather than keep this fact secret.

Conclusion

The proposed Settlement would have little effect upon the business practices of Microsoft. If adopted in its current form them the result will be no change to the behaviour of Microsoft, and yet another prolonged court case in another five or ten years.

Any effective settlement must concentrate on opening up the markets that Microsoft has effectively closed by its use of proprietary interfaces, file formats and protocols.

I hereby respectfully submit these comments for your consideration,

Paul Johnson.