From: Steven Sinofsky
Sent: Sunday, August 18, 1996 4:54 PM
To: Andrew Kwatinetz; Brad Weed; Heikki Kanerva; Ken Dye; Richard Wolf; Sam Hobson
Subject: FW: '97 Tools Vision

-----Original Message-----
From: Bob Muglia
Sent: Friday, August 16, 1996 6.29 AM
To: Developer Tools Division
Cc: IPTD Fulltime Employees; Executive Staff and Direct Reports
Subject: '97 Tools Vision

********** Microsoft Confidential **********
********** Do Not Forward! **********

With beta releases of our '96 tools either underway or rapidly approaching, I thought it was a good time to outline some high-level goals for our '97 tools.

First, it makes sense to look back on what we've accomplished this year. To start with, we have gone from zero to sixty in record time building all new Internet products and incorporating Internet features into our existing tools. Java is a new market and Visual J++ defines the state of the art for Java developers. The ActiveX Control Pad and HTML Layout Control are making a huge splash. VBA is poised to take the application market by storm. DaVinci is ready to add database design across our tools. Internet Studio has been reborn as a key tool for building Internet applications and creating great content for the web. Our other tools - Access, VFP, VB, and VC, all have strong Internet development features. In addition, our multimedia tools are set to go out and help Frontpage and Internet Studio do battle in the Internet tools market.

VB 5, VFP 5, VC 5, and Access 97 are great upgrades for their respective customer sets. Each provides significant performance improvements and new features to address customer needs.

We are getting ready to introduce DevOffice, a product which will define a new category in development tools. As we're nearing the introduction of this product, we now have an official name: "Developer Studio 97".

With the introduction of Developer Studio, we've re-engineered our internal processes. The DevOffice group (I guess I'll call them the Dev Studio group from now on!) is building a product which is focused on solving complex customer scenarios which requires the integration of multiple tools. The individual tool product and component groups are building shared components and single-language focused tools to address particular customer scenarios.

These changes represent the beginning of the process of our movement to the "Office" model for building tools. The teams are working together better then ever before and it is now clear that each group is quite dependent on other teams. The evolution of these processes will continue into 1997. Our '97 product plans will reinforce the message that of each of us will succeed by all working together.

Although we've made great progress this year, the world continues to evolve around us and our competitors are not standing still. In the past six months, Java and in particular the set of Sun API's built around AWT has garnered enormous mindshare in the development community. Without question, the Java platform API's have surpassed the Macintosh as the #2 platform for software development. In concert with this, Netscape has its own offering of platform API's called Netscape One which is also built on Java. Collectively, these two initiatives represent the most serious threat to our core Windows business which Microsoft has seen in years.

The Windows franchise is fueled by application development which is focused on our core APIs. When a developer writes an application to AWT, even if they are using Windows and Visual J++, they are not supporting our platform. Instead, they are furthering Sun's momentum, potentially opening up the opportunity for our competitor to slide in its own operating system offering.

The essence of our tools strategy is to make it easy for developers to write applications for Microsoft platforms. The Internet has changed the rules and opened up opportunities for new competitors. Our development tools need to work hand-in-glove with our platform initiatives to provide unique and compelling solutions for our customers. Fortunately, our Internet strategy brings with it the adoption of a key integration technology which we can exploit to provide these compelling solutions. That technology is of course COM and we can build upon it as the basis for our overall tools vision:

Overall Vision: Language independent rapid application development using common components.

GOVERNMENT EXHIBIT 506
MS98 0137005
CONFIDENTIAL

Our technology is poised to allow developers a new degree of flexibility and effectiveness. The keys to this are the implementation of the system API using language independent COM interfaces combined with rapid application development tools which allow highly effective application development using either VB, VJ, or VC.

The idea is to create a suite of tools which have common capabilities, built from common components. In this world, language becomes the choice of the developer. We are far from that world today. Each of our languages brings with it strengths and weaknesses in it's development environment, forms, database access, wizards, etc. We need to change this by leveraging our strengths across each language.

We are taking a major step towards building our products on common components with Trident, the HTML release of Forms³. As Trident will be built into IE and Windows, it provides a common system service for forms. All of our 97 tools should target Trident as their primary forms package. That doesn't mean that we won't support Ruby or our many other forms packages. They will continue to be required to move forward existing application investments. However, our '97 products should both support and give preference to Forms³ for new applications.

With Forms³, we will also be introducing a new set of COM-based class libraries for application development. You can think of these as a next-generation MFC, but unlike MFC they will work with all of our languages. Initially, there will be classes for drawing, ActiveX Control creation, and database access. Significantly, each language needs to support the creation of components through the common ActiveX Control class. This means any language can create a control and all new controls will be built on a single base class.

On the Server side, we are poised to dramatically improve and simplify the development process for building multi-tier applications. Improvements in the underlying system infrastructure such as DCOM, Viper, and the ActiveX Server provide great opportunities for our developers to build business applications on the server. Here again, the key is using COM-based common components and providing the support for these services in all of our languages. This adds real solid value for our customers because it makes something which has been very difficult to do on any server platform much, much easier. Because it is very high value it will drive increased acceptance of NT Server and provide us with the opportunity to sell more Enterprise tools to our customers.

Over the next year, you can expect to see many of the new system services made available to all our developers through language-independent COM interfaces. This is a significant shift for our customers because it means that as the Windows platform evolves, the majority of APIs which are created are instantly available to all our developer customers. Over time, you can expect to see an ever increasing percentage of the Windows API exposed through language independent COM. Major progress!

As we move toward language independent development, we need to ensure that we focus on rapid application development techniques. Developers should be able to instantly put code behind any object using an approach which is similar to VB. We need to extend this approach to further simplify the process of rapidly building an application. Given this common, RAD approach across languages, we will rapidly evolve to a common IDE because it only makes sense to do things once. We have taken the first step by defining a set of COM interfaces for shell extensions. A key next step is the definition of a cross-product project model.

A similar goal is cross-language debugging. Here again, the first step is to define a common set of debugging interfaces across runtime environments. We have established the core requirements for these interfaces with the Java VM and ActiveX Scripting.

Achieving this vision involves many people within developer tools, as well as others in IPTD and DBSD. It is a compelling objective that will make a huge difference for developers. In addition to this core vision, there are other critical cross-group objectives which we need to address:

- **Great Database support.**
  The majority of our customers use our tool products to create custom solutions. They want to model their business process with the combination of a database, business rules, and graphical UI. Database access is critical because custom application development almost always involves one or more databases, with SQL databases the most important. We should prioritize database support in the following order: MS SQL Server, Oracle, Informix, DB2, then the rest. Our tools have varying degrees of support for database development. Next year, we need to improve our support across the board, making it much more straightforward for any of our core languages to support database development. In particular, we should apply some of the great ideas and technology found in Fox to VB, VJ, and VC. We also need to provide great database support to other servers besides MS SQL, with Oracle taking top priority.

  For programmer access to data, we are building a common set of interfaces codenamed ADO (Active Data Objects). ADO represents the merger of RDO and DAO. It provides a common high-level interface for the majority of database solutions. We are working closely with the Database group in DBSD to provide efficient solutions for accessing both ODBC and OLE DB data.

- **Inter/Intranet content creation and application development.**
  Many customers have begun using Internet technology to build their next generation business applications. This is very attractive because Internet technologies tend to centralize the business process and minimize the deployment and versioning costs. This is consistent with the goals of many customers. Further, widely available browsers mean that a business can reach out and provide new solutions directly to end users, thus increasing the leverage they get from technology.

Thus far, most customers have found that the current browser standards and the tools are lacking for the implementation of significant business solutions. This will change in '97. We should assume that Internet Explorer will gain significant market share and that the extensions and capabilities it supports will be widely available. On our end, we need to drive the maturation process for the tools and provide our customers with a great environment for building applications and easily creating high-quality content for the Web. While we need to support multiple browsers, our focus and effort should clearly target Internet Explorer.

On the content creation side, we have begun the process of pulling together the tools needed to build great looking content on the web. Image Composer, Music Producer, and Hammer can be used to create images, sound, and multimedia content for inclusion in web pages. In '97, we need to take the next step by focusing these tools on the multimedia extensions which are made available in Windows and the Internet Explorer. The platforms group is focused on building a multimedia runtime environment which provides some significant extensions to HTML. Our job is to build easy-to-use tools which author great content for that extended HTML runtime and make it easy for an average user to incorporate rich multimedia content in HTML documents.

With our common forms moving to HTML, in one sense, each of our languages will be able to build applications based on Internet technology. Still, there is a major difference between building a client/server application which uses HTML forms and building an HTTP-based Inter/Intranet solution. A key part of this difference involves the use of the server for the creation of HTML content.

The Internet Information Server is evolving from a file server with the ability to run custom applications to a script driven environment which in turn supports server-side components. The impact of this evolution on the Internet developer is similar to the impact VB has had on the development of Windows applications. Internet Studio targets the creation of IIS applications, so it needs to closely track and influence the evolution of IIS. Our languages should target component creation for this environment.

Internet Studio thus becomes an integration environment for components created by any of the visual languages as well as content, HTML or otherwise.

- **Integrated development of enterprise applications.**
  As our system platforms push upward into larger and more complex enterprise solutions, our tools are called upon to solve problems which previously would have been relegated to either Unix or a mainframe. Many of these solutions involve the integration of components in different languages. Almost all of the solutions involve teams of people; some involve multiple teams collectively making up a large development staff. The solutions often include separation of database, business logic, and UI into multiple tiers. While products like VB EE and VC EE have features to address these issues, we have not yet taken a top down view on the customer scenarios for complex application development and certainly our products today have many seams which make them awkward to use for many of these solutions. A focus on these scenarios and improvement of their integration is a key objective for the Developer Studio group.

  With the move to common, shared components, we need to move into an environment where we build special capabilities for our most demanding, enterprise customers. Those components should be language independent and made available through the Developer Studio product. We should expect that over time, components will move down the food chain and become available to our individual language customers.

  Two things we keep hearing from these high-end customers are integration and team development. The Developer Studio product group will drive the cross-product integration and focus on improving the experience of the user who is developing a solution using many components and potentially, multiple languages. Another key is building a project model and backend store which supports all the standalone products, yet additionally provides an integrated solution. We will need to work closely with both the server group in Anthony Bay's team as well as the repository group to accomplish this.

- **Easier,** more approachable **UI.**
  To grow the tools market, we need to reach out to a broader set of users. Some of these users may be stepping up to our tools from VBA within Office or other applications. A key set of new users are content developers building Internet applications. In either case, an easier, more intuitive user interface is critical to growth.

- **Cross-platform**
  The Internet requires a compelling solution on multiple platforms. Internet application and content developers want to reach the widest set of users regardless of their platform.

  It is clear that Win32 is the primary platform for application development. When discussing other platforms we generally think of the Mac, Windows 3.1, and Unix. While Windows 3.1 still represents a considerable market, from a tools product perspective we are betting that this market will largely transition to Win32 during FY '97. Other then providing support for components in the Windows 3.1 version of IE, we are not making any investment in 16-bit Windows tools.

  As mentioned above, we must accept that in the last six months, the combination of troubles at Apple and the surge of interest in Java has made Java the #2 platform. This is both an opportunity and a challenge to Microsoft. It is an opportunity because we are well positioned to provide the best tools and runtime VM for Java. It is a challenge because when a Java developer writes to AWT, they are writing to Sun APIs and their application can be easily run on competitive platforms. That is the driving reason Sun licensed Java to MS: so they could steal MS developers and

retarget them onto their platforms - such as Solaris or the Network Computer.

So how do we succeed with Java? By building the best possible tools and VM for Java the *language* while at the same time providing Java developers with a more compelling set of APIs. In addition, we need to bring the cross-platform attributes of Java to both VB and VC++.

Trident and the COM class libraries form the core API Microsoft needs to provide to developers of all type, including and maybe especially Java developers. The platform group in IPTD is currently working on plans to make these services available on other operating systems, such as the Mac and Unix. As these APIs will incorporate rapid application development, they will be a very compelling choice for developers.

Cross-platform bytecodes is an essential part of Java's allure. We need to bring this technology to both VB and VC. We will work together with the Java VM team to provide a cross-language bytecode solution. This is technically possible and would be well received by the industry.

Collectively, the combination of a cross-platform API and bytecodes which support any language should provide MS with a strong offering for Internet developers, regardless of their choice of language.

That covers the major customer scenarios which Developer Tools needs to address in '97. I realize that this mail is just the beginning, we need to drill down and create product plans for each individual product and component team. Still, I hope that this outline gives you some high-level objectives to target as you put together these '97 plans.

thanks,
bob

********** Microsoft Confidential **********
********** Do Not Forward! **********

MS98 0137008
CONFIDENTIAL