To: Office Productivity Hallway
Cc: Jeff Raikes; Bill Gates; Steve Ballmer
Re: Creating Office.NET
Date: October 23, 2000

CREATING OFFICE.NET

Next steps in creating a vision for productivity

INTRODUCTION

This memo is about creating the next generation of Office. Not a vision statement, this memo outlines the business situation and the clear direction we are taking the Office product, and the bets we are making (a vision statement will follow soon). This memo is also about creating a new product—one that takes the enormous success of Office and melds it with new functionality and new technologies to create an exciting new product. We will call this product Office.NET. Office.NET is the essential set of tools and services that empower individuals to get their work done with a personal computer.

We are in the midst of creating an incredibly exciting release as we round the final corner for Office 2002. As has been said many times, shipping Office 2002 on time with the quality both we and our customers demand is our first priority. It might seem a little confusing to hear how critical it is that we ship Office 2002 because it is so important, yet at the same time we need to start work on the next release. The truth is that we need to do both of these. Microsoft's business is counting on us releasing a new product early in 2001. At the same time, something we have long pondered is coming true; customers are not as excited about "traditional" Office productivity as they are about new features and services that fall outside the Office "box." Office 2002 is about closing the books on a generation of Office and starting afresh.

Starting afresh is different than starting over. We are starting on Office.NET with the benefits of over 100 million happy customers who not only use Office, but rely on Office every single day of their working lives. We have the benefits of years of experience in optimizing the customer experience for our "desktop" software and designing, building, and testing great applications for end-users. But times have changed and what customers once valued is no longer top of mind; the next cool IntelliSense feature, or the latest and greatest "basic use" feature simply isn't enough to cause a customer to go out and buy a new copy of Office or cause a large organization to deploy thousands of seats. We know, absolutely, that we are catching the current and the trailing edge customers with Office 2002—Office.NET is about catching the leading edge customers. It is these leading edge customers that, back in 1990 with the introduction of Windows Office, caught the wave that carried us to be one of the most successful businesses ever in the history of business. It is no small job to follow that success. There is an enormous responsibility to Microsoft and to our customers to design and build the productivity tools that will continue to be contributors to the ever-more efficient economy—tools that help people get their work done with less effort, more profit, and a little bit of fun!

As you read this memo you will almost certainly have questions—you might want more details on a specific point; you might think something is a particularly silly idea; and you might wonder what this means to you and your role on the team. Not only are all of these legitimate questions, but they are in a sense the purpose of this memo. Office.NET will require more creativity, more entrepreneurial thinking, and more "change the world" energy than we have had recently. Building Office.NET is going to be a radical departure, but one filled with payoff—for Microsoft, for customers, and for those that step up and accept the challenges of building an incredibly important piece of software—Microsoft Office.NET.

DEFINING SUCCESS

As we begin planning Office.NET and building the vision, we should look at what it means for us to be successful. With the early versions of Office, success was defined by being smaller, faster, and having more features than our competitors. As customers became more sophisticated success was measured by which products shared more consistent user-interface and which applications had the richest extensibility model. The early releases of Office also made some big bets on some important technologies such as the graphical interface. This was hardly a no-brainer (in fact, one of Applications' best developers even quit over the decision to bet on GUI, if you can believe that). As Windows achieved success, we moved our big bets to shared code and ultimately to the web. Along the way we made a few bets that might not have paid off as well as we would have liked, but overwhelmingly our bets have paid off. Customers have bought and come to rely on the innovations in Office, or at the very least some of our bets have reminded customers that we take risks and are pushing to develop new technologies and bring them to market (sometimes called "thought leadership" or "clippit").

Success for Office.NET will be defined in very broad terms—Office.NET is the first release of many to come. This is our "V1" release, much like Office 4.0 was the first "real" Windows Office back when Office was defined, by customers and the market, to be a simple toolbar called "MOM." Sometimes it takes several product iterations to really deliver on a complete vision and no one is going to say we will get everything right and include everything we want in the first release. We want to learn from the marketplace over several releases.

Success for Office.NET has the following characteristics:

- Customers move beyond the view that Office is "just" a word processor, spreadsheet, email client, graphics, web authoring, and database. **Office.NET adds whole new services and applications to the toolset that we build and sell as Office.** Think of the service and services elements of Office.NET as "puzzle pieces." When we release Office.NET people will use our product to get work done in new ways that they might not have thought of and certainly did not think of using Office. **Office.NET is not "Office 11."**

- Customers not only use our new suite of hosted services but customers come to rely on Office.NET services as a critical element of getting their work done. Office.NET services are not about gimmicks or "dumb PC/internet tricks" but about being simple, elegant, and useful additions to getting work done. For customers, **Office.NET is about saving hours, not mere seconds.**

- The glue that holds Office.NET together is integration and integration is what makes the value of our product greater than the sum of the pieces. **Customers using Office.NET see an unprecedented integration between their tasks**—whether those tasks are Office.NET services, desktop productivity tasks, browser-based services, third-party services affiliated with Office.NET, or Microsoft's own MSN services. Integration is the key that allows a customer to solve real-life problems, such as sharing a document with a partner outside the firewall or merging a work calendar and a private calendar. Many would say that one beauty of the internet is the elegance at which a large number of valuable tools interoperate saving time and effort—we will bring that elegance to Office.NET's services.

- **Office.NET provides a new level of "customer service"** by keeping the software updated, enriched, and "running" for customers. No longer will customers feel like they are "cut off" from Microsoft after they buy the product or feel like they have to wait a year for a 30MB patch to fix things. Of course, Office.NET doesn't change this from the first day a customer gets the product, but we will over time build up the service relationship. Customers no longer view buying Office as a one-time transaction, but rather **customers subscribe to Office.NET because Microsoft is making a commitment to back our software and services with the highest level of support possible**.

- Customers who use Office.NET can do so with full faith and confidence that **Office.NET provides a safe, secure, private, and reliable service**. We will go to extremes to insure that customers can trust their important work to the tools and services offered by Microsoft. Everyday one hundred million people trust their work to Office, so we're in a good position to extend this trust to a new level of support. This will not come easy, but we will make it so by making it the highest priority in everything we do.

- **Office.NET is good for business.** The great American philosopher, Steve Martin, once had a moment of enlightenment when he realized "it's a profit game." For most of the history of Office, it has been more than good enough to maintain a clear focus on improving our engineering and building products that more often than not continued along the path of incremental improvement and that led to an amazing business. Office.NET is about building a new product and selling this product in new ways. We are making these choices because everything we know says that they will be good for business—just as we thought building Windows applications was going to be good for business. We will run a service business with the same focus on efficiency and cost that we have had in building our packaged product business.

(REALLY) BIG BETS

Office.NET will make some *big bets*. We love big bets. Office was a big bet—who would have thought putting all the "categories" in one box would have caught on. Office 95 was a big bet on 32 bits. Office 97 was a big bet on shared code. Office 2000 was a big bet on the web and cost of ownership.

Office 2002 is a (renewed) big bet on end-users and a big bet on Exchange 2000. Each release has a small number of big bets—everyone building Office.NET can assume a few things and take these as axioms. The big bets we are making for Office.NET are the biggest we have ever made:

- **Services**. In the broadest sense a service is something Microsoft offers as part of Office.NET that goes beyond what a single customer can do by themselves on an isolated PC—*a service is work done by Microsoft on behalf of, for, or in partnership with a customer*. Being a service means we store information for customers so they can get to it from any location. We understand the needs of customers and can proactively offer up suggestions or help. We provide mechanisms for customers to connect with their partners, co-workers, friends, and family through servers that facilitate communication. A service means that a customer's copy of Office keeps running, and gets repaired by an update service we run. The best way to know that we've built a successful service is that if it isn't there customers will know right away and we've let them down. **We're betting that we can flawlessly provide incredibly useful services for our customers.**

- **Subscription model for business**. As we have discussed over the past several years, our business has been moving from "boxes" to "licenses." Today these licenses are *perpetual*, which means we sell the customer the right to use the product forever, but our relationship with the customer essentially ends, except for the largest customers that buy support contracts. We are betting that a significant number of customers will find a subscription model for Office.NET very attractive. By lowering the cost of entry, enabling a higher level of support, improving the product over time, and operating services we can provide a very significant value to customers. **The subscription model is a natural extension of the services model and we're betting that subscriptions are good for business and good for customers.**

- **New code.** For Office.NET to succeed we must build entirely new features in a service-centric manner. One need look no further than successful services such as WebEx, to see that scenarios that we used to think of as being "baked" into Office can be built using new code in a new way (on the server with client-code that makes the experience better). **We are betting that the best way to build new services for customers is by building new code.** We know that the compatibility and "legacy baggage" of Office makes it extraordinarily difficult for us to add new features, and ironically makes it very hard for customers to discover these features as well. **We are betting that we will do almost no work in existing code**—we will minimize this work to the features that vastly improve the core experience for every customer or features that make services vastly better in ways that the market would not have predicted. Of course FrontPage and Outlook, as part of Office 2002 and Office.NET, are applications that are already exclusively focused on enhancing service experiences. We will continue this work, but with a focus on the services that we operate rather than arbitrary third party services.

- **Bandwidth.** We are betting on connected customers. We will not dumb down services for the lowest common denominator, but we will also not assume an always-on T1 connection. We will be as economical in the use of bandwidth as we are in the use of RAM or disk space, which means we will treat bandwidth as a limited resource but one that is also necessary to get the job done. Bandwidth increases at different rates around the world, so we will not necessarily build all of our services in every language—just as we did not provide the latest releases of our Windows applications in every language until there was a critical mass of hardware and OS use around the world. Customers with DSL or cable modems will be able to sign up on a clean machine and get Office.

Customers with 56K dialup connections will be able to use our services such that we are competitive with other offerings. Document creation and email are tasks that can take place offline, and we should not do anything to minimize those capabilities. **We are betting on the increasing availability of bandwidth and that customers can connect to the internet when necessary.**

- **New productivity experiences with services**. As strange as it may seem, not a day goes by where a customer does not offer ways to improve our applications. Yet at the same time we see the results in the marketplace that point to a business that cannot grow if our product strategy is to meet the needs of those expressive customers. **We are betting that entirely new productivity experiences are an opportunity to grow our business and provide an exciting offering for customers.**

- **Microsoft's .NET Platform**. The greatest success Microsoft has had has been when there has been an effort to develop a new platform and new applications for that platform. This is not easy—in fact this is enormously difficult. That is why customers place a high value on such work. **Office.NET is betting on the key elements of the .NET platform.** Some of these elements are not done yet, so we will make them ready for prime time. We will provide the input that will make these platform services real and valuable.

These are the big bets that we will make for Office.NET. Some will pay off handsomely in the first release. And some will take time to gain traction. And it might even be the case that after trying for a while we look at a bet we made and back-off a little bit. That is okay, and in fact the Office team has shown enormous flexibility in making bets—both in sticking to them through challenging engineering and in adjusting our priorities in response to the market.

GROWING OUR BUSINESS

By any measure, the Office business has been an incredibly successful business. In the span of a decade, Office has changed the world, the industry, and built an amazing track record of business success. Yet, we cannot rest on our laurels; old ways of doing things are not a foolproof recipe for future success. It's clear that the Office business is at a critical crossroads today as we plan Office.NET. Several basic aspects of our business point to the challenges we must address:

- We have very high worldwide penetration (legal and pirated) for Office (mostly 97, and about 30% 2000) across Home, SORG, MORG, and LORG. Legal ownership, however, is still low in many geographies and segments.

- As penetration increased, revenue growth has steadily dropped over the last 6 years, from 30+% a year then to nearly 0% revenue growth today.

- Upgrades are not getting easier to win. The market believes that Office is already packed with more functionality than they can use, and that just adding more "traditional" features to Office is not likely to warrant frequent upgrades.

It is worth emphasizing one point above all others—our business is not growing in revenue. This is shocking and has happened relatively quickly. Although we have been on steady decline in our growth rate, we are now at the point where our business could actually be *shrinking* just due to the increased cost structure (i.e. salary increases). This requires nothing short of a radical approach to increasing our revenue and profits—we are in the business of making products customers value enough on which to spend their hard earned dollars or limited budgets.

There are several tried and true paths to increasing revenue and profits. Profits can be increased by lowering costs. Office is an incredibly lean and efficient organization and there is not much to change, certainly not on the scale of revenue we are talking about, but we will continue to be mindful of our cost structure. Revenue can be increased by one of two ways: increasing the price customers pay for the product (a very delicate situation), or finding new sources of revenue by finding new customers or developing new products to sell. We are going to emphasize the latter.

With Office.NET we are tackling nothing less than the reinvention of the Office product and business. In the context of Microsoft's .NET initiative, we will assume connectivity to redefine what Office can do for customers. In fact it is important to consider not just Office the box but Office the *concept*, the suite of tools necessary for an end-user to use a PC for doing work. Much of this new functionality will be delivered as a Microsoft service, with new revenue models. These new services must first and foremost be very compelling to individuals. By really understanding what the individual needs and building compelling services that meet those needs, we create the basis for a sound and broad business. Delighting the individual is of course a core part of our Office heritage we can build on.

Our business has historically tracked a number of key metrics around revenue. One of the most important has been the *dollars per PC* or $/PC. This metric is very easy to calculate and looks at Microsoft's yearly revenue (overall or for a product category like suites) and divides that into the total number of PCs sold that year. Our sales force is very focused on this goal, and because of the high value customers place on Office, Office is a key driver to improve $/PC. Yet despite our best efforts, this metric has been on a steady decline. This means more and more PCs are being sold and through a combination of factors we are getting less revenue. Many PCs are second and third machines, which don't necessarily garner a new license for Office. Many PCs are being sold to corporations which buy volume licenses of Office at a discount. And many PCs are being sold to consumers that mostly use the internet though they might use an older or pirated copy of Office. To make these numbers concrete, Microsoft generates about $120/PC for our best and largest corporate customers because they buy the most software and most corporate PCs have Microsoft software. For a medium sized business we generate about $54/PC. For a small business we generate $24/PC. For a home user we generate $13/PC.

Where is the opportunity? There is opportunity in two places. First, we must solve important problems for our corporate customers that will make it easy for them to justify spending dollars on software. When we think about the types of services we will offer—roaming important settings around the globe, hosting meetings between individuals and partners, or coordinating schedules across corporations it is easy to see how valuable these are. In fact, many of these services are currently used by employees at large corporations and we are in a unique position to offer "enterprise ready" versions of these services. Even if they are hosted on our servers, corporations will use them if they solve a problem cheaply and efficiently. It will not be enough for us to have some nicely integrated services, however. Integration is absolutely necessary, but not sufficient to garner the leadership we need. We must also provide innovative and unique services, which our large customers view as thought leadership. They want to know that we are recreating the way their "knowledge workers" use their PCs and tools and doing so in a way that makes businesses more responsive to customers and more profitable. And of course, we must continue to insure that the desktop components of Office are easy to deploy and manage.

Second, we have an opportunity to sell more software to medium and small businesses, and to work-at-home individuals. We know that Office is being used in these segments. We also know that piracy is high. And we know that sometimes the price of Office is a slight barrier, particularly for home and small business, even though the price of Office continues to decline. The subscription model for Office offers the advantage of a *pay as you go* license which as we know lowers the cost of entry and decreases the cash requirements, both of which are important to small business. Think of the simple math that if only a small percentage of customers purchased an upgrade because of the lower subscription entry price, we are actually getting more from that customer than today's $/PC in the home market in just a single month or two. That's a huge revenue upside and customers reap all the benefits.

Office.NET will be an upgrade offering for our customers. From a business perspective, what matters most is getting customers to use and continue to use our services. We do not think it is prudent to offer services as distinct from the "client code", yet at the same time it is clear that for services to be successful they must work for the majority of browser-based scenarios (kiosk machines, a machine at a friend's house, or at the library) or for machines with an older version of Office. When a customer subscribes to Office.NET they will own an Office.NET license which allows them the right to use the services and the new applications, but we will not force them to use the new applications (just like AOL does not force subscribers to use the latest AOL client). Since services cannot be pirated this is much less of a business problem than it might have been in the past, and since Office.NET's client side applications (Word, Excel, etc.) cannot run without being registered and validated against a service account (as our annuity model allows) they are secured as well. Of course we are interested in showing off the best ways to solve a problem, so it is certain that some services will be better, or only available, when you are running the new Office.NET applications. We should not compel ourselves to implement Office 2000 or Office 97 add-ins to enable the full service experience, but we should also consider this as a viable mechanism for enabling services in Office.NET's clients so that we do not have to implement the integration in core code. We should also insure that core scenarios, such as storing files on the web, continue to work with Office 2000 and/or Office 2002.

Office.NET is an important part of Microsoft's .NET strategy and is also the biggest opportunity we have to provide a new experience for customers while also providing a set of tools that allow them to get their work done. The .NET strategy is quite similar to the Windows strategy—one group is providing a platform along with some applications that show off the platform, and one group is providing a robust set of full-featured applications that build on that platform. The .NET platform is slated to provide some key services such as billing, passport authentication, and profile storage, along with new user-interface elements, all of which are important base level features of Office.NET. Built on top of that platform to demonstrate and "prove" the platform will be NetDocs technology as a basic service. Our job is to build the full-featured service that will be used most broadly on top of that platform, learning from the experiences of the NetDocs team. This is similar to how the Office applications learned from the experiences of Write as early versions of Windows were built, and also how the Excel team really drove Windows to be a robust platform. One addition is that Microsoft will offer "free" services via MSN. These are not really free since they are supported by partnerships and advertising, but these services form the entry level of services. There will be some overlap and some seams, but the goal is to provide a base level of services with advertiser-supported MSN, an enhanced level of subscription services that includes the NetDocs technology and .NET platform elements, and then the broadly marketed and used level of subscription services that we provide in Office.NET also built using elements of the .NET platform.

We have several specific goals for the business of Office which are going to be met by the Office.NET plan:

1) Sustain and grow the overall revenue for our products, including Office. This means continuing to sell packaged product as appropriate, continuing to drive license sales as needed. It also means we will create new sources of revenue such as an Office storefront hosted on OfficeUpdate which allows us to sell Microsoft software or provide a conduit for third parties.

2) Move a significant portion of our revenue base to a subscription/annuity model, which includes maintaining today's large customers that are on enterprise agreements. This is how we will make money on Office.NET. A subscription/annuity model has the potential for more revenue in $/PC than our current model, though with an increase in costs to us. That is a bet we are making.

3) Improve our efforts to fight piracy by enforcing the proper use of our core executables but by also having killer services that people want to use, which are inherently impossible to pirate (AOL doesn't have a piracy problem!)

4) We must operate our services efficiently. Our success at entering this new business depends on our ability to develop services that scale very well and do not require us to have inordinate resources per-user or per-server. It is acceptable to have pay-per-use services for very high-priced or resource intensive services, but we must pay careful attention to the entire package of services we offer customers and the cost structure of those services. We will carefully track the total cost of a service, the cost per user, and monitor important issues around scalability.

PLANNING OUR FOCUS

One of the most impressive aspects of the Office product development process is the focus we have on developing a clear product vision, prioritizing features within that vision, and then delivering a product that meets the goals and adheres to the tenets of that vision. With a muddled vision, a muddled product is guaranteed.

The process of creating a vision, like so many creative endeavors, is chaotic and filled with uncertainty. This is a necessary element of insuring that the vision is a synthesis of the best ideas and clearest strategy we can muster. Over the next 8-12 weeks, through January, our team will be crafting the vision for Office.NET. Our goal is to have a vision well before we RTM Office 2002 so that specification writing can begin in earnest and we are ready to begin Office.NET with a smooth transition from Office 2002. This is not an easy task.

In order to facilitate creating the vision, program management is organizing planning and brainstorming around key *focus areas*. These are not the vision areas but will serve to structure our thinking and brainstorming. These focus areas are not disconnected silos, but should be viewed as inter-related. Features in one focus area will depend on features in another. These are representative of the work that has been going on through the summer and fall at various offsites and other forums. The focus areas are designed to illustrate the breadth of possibilities of Office.NET while insuring that we have a solid framework and a shared understanding of some specifics about Office.NET.

ACCESSING MY INFORMATION FROM ANYWHERE, ANY TIME

We all have experienced the inconvenience of leaving a file on a PC at work or being unable to get to a mail message from a friend's house because they don't have access to the Microsoft corporate network. These are problems that can be solved by internet technologies and in fact many new services already exist that make it somewhat easy to accomplish these things from Office 2000. For example, fusionOne allows you to synchronize (relatively transparently) your mail and PIM information to a secured service which you can access from any browser anywhere, or numerous "hard drives in the sky" such as driveway or xdrive allow the saving and loading of files to secured servers. We will call these services that utilize internet-based storage *cloud services* and refer to storing information in the cloud. Being able to access information from anywhere and any time is Microsoft's mission and Office.NET will offer cloud based storage that makes this possible for our customers.

Once we enable customers to store data on our secured servers we can also store data on behalf of customers. Office.NET will make it (even) easier to store your user settings and profile to the cloud. In fact, it should be easy to walk up to any machine and easily switch to your Office personality (and you might even have more than one Office personality, one for home and one for work). Your settings, customizations, spelling dictionaries should all easily migrate to the local machine. MSN Explorer has done an implementation of this for your favorites and other browser settings which allows for easy roaming. Office.NET will expand the idea of a profile to include much richer information than we do today in HKCU, such as your specific industry interests, subscriptions to Office assistance and newsletters, and a whole host other account information.

Of course everything that is stored in the cloud must be secure, reliable, and private—our customers are paying customers and they require this.

The focus of accessing information anywhere goes well-beyond just storing information in servers. It also includes being smart about creating new possibilities for networked information. For example, today a mail message is a "static document" even though we know the person reading a message can often connect to a server we don't take advantage of that. Imagine that instead of opening up a mail message with a copy of a list of people attending a meeting or a list of bugs to be fixed, you opened up a mail message that went to an internet-server and obtained the latest view of that data. Imagine that this message also contained buttons that allowed validated users to add, remove, or modify the list. These *always live* documents, for example as seen in Zaplet, are an exciting possibility that is truly unique in a service-focused environment.

An Office.NET customer is one that uses more than one PC and is also a customer that uses other computing devices for getting work done, including cell phones and handheld. Office.NET provides us the opportunity to solve problems customers face today in managing their information across these products. The most obvious problem is getting information such as your phone numbers from Outlook to a phone, for example. But again there is so much opportunity because of our service focus. For example, if Office.NET hosts your schedule (or a copy of it) in the cloud it is possible to deliver reminders to a cell phone by just selecting an option at the time you create the appointment. If someone changes a document in your team's shared workspace, your subscription can notify a pager. Many of these scenarios are being worked on in isolation but we have an opportunity to provide a complete suite of these features that also integrate with the desktop experience that will still account for the bulk of a customer's computing experience.

CREATING A PERSONALIZED OFFICE EXPERIENCE

In the days before the internet, computing was represented by standardization by IT departments. The internet has created a world where customization and personalization are not just features, but required features. Customers expect a service to be tailored to them. Office.NET provides a personalized experience that allows customers to be more effective and more efficient in getting their work done because Office.NET can provide the right tools and services to customers based on their own preferences or based on an intelligent understanding of what is important.

First and foremost, Office.NET customers should have a home base that allows them to control their subscription to Office. This isn't their default home page or a portal to an array of information, but it is a place where the varieties of Office.NET services come together for a customer and where customers know to come when they want to manage their service level agreement with Microsoft.

We also know that Office customers like a starting place when they use Office and they like that starting place to be small and readily available. Perhaps the most canonical representation of this was the original Microsoft Office Manager (MOM) which allowed for easy access to the most common applications. Office.NET can offer customers this type of home base, without the burden of yet another portal page, by building an internet enabled MOM. A marketplace example of this is the Yahoo! Companion, which is a MOM-like toolbar that roams with you and at the same time builds an ongoing personalized view of your use of Yahoo!

At the same time we know that many customers clamor for a "heads up" display of their work. This display can include important mail and PIM data, data from third party applications on their secured intranet, and also data from a variety of sources such as MSN news. Today in Office 2002 we provide an architecture that supports a digital dashboard and the beginnings of a service offering with customizable webparts. A subscription offers us the chance to tune these offerings based on actual use as well as the chance to offer relevant webparts to customers based on their personalized profile.

Personalizing Office.NET means getting the most out of the services, and the content, available through Office.NET. With information about the past activities of a customer we can offer helpful new services or provide information on updates to services (and just as importantly this information can help us to focus our energies on updating the service). When a service becomes very popular, such as a template for a calendar in December, we should bubble that to the top of the site and interested customers can subscribe to these types of notifications. Services allow us to offer the power of the instrumented version, but for all of our customers, not just a select few. In fact a recent article in *The Economist* even postulated that "instant feedback may go some way towards explaining why web-based software is now so much more innovative than desktop productivity software."

Perhaps no area of Office is in need of personalization more than supporting the technical support and user-assistance needs of our customers. Today every support incident, whether on the phone, email, or a BBS, is a game of 20 blind questions—what OS, what version, what hardware, what Tools Options, what is the error message, etc. With a service approach and personalized information about customers we can dramatically alter the support process—and this alone is worth a subscription for many customers. We can provide assistance that is tuned to the skill level of a customer or we can provide industry specific assistance written by experts in the field.—no longer are we limited by what we can squeeze on a CD or install locally. In Office 2002 we introduce DAD Watson (DW) that is the first step in this direction and an incredibly important first step. There are many ideas on how to improve, and further personalize, this experience. Related to DW is the ability of providing the always-up-to-date client and again we have an early start on this with OfficeUpdate's AutoUpdate. This feedback is crucial to the development of Office.NET over time. Every service aspect of Office.NET will incorporate a feedback mechanism that will be consistent across the product. There is ample opportunity to deepen our relationship with subscribers and insure that every interaction with customers *closes the loop*.

BUILDING EFFECTIVE COMMUNITIES AND TEAMS

Our target customer for Office.NET is the individual. Yet every individual does their work as part of a larger whole. Whether the customer needs to share information with friends, co-workers, or partners at another organization, the need to work with others is inherent in using Office.NET. What was great about the PC revolution was that it was all mine—but that quickly hits a wall when you want to share information with other people or create a place for your team to virtually congregate. Of course email, file servers, and storing documents in the cloud facilitate the basics, but building a community around Office.NET means that the product gets better as you allow more of your co-workers to participate in your projects and work, and as more people participate, the community grows stronger.

The cornerstone of a great community is a place for people of like interests or job responsibilities to congregate. With SharePoint (Office Web Server), Office 2002 provides that for customers. Yet we have a great deal of work to do to build a service offering with these critical features, and we will do that for Office.NET. The combination of a personalized place on the internet and the ability to invite fellow subscribers (or just people with a Passport) makes building a community easy and "out of the box."

Meetings are one important aspect of teams, whether the team members want to admit it or not. The pervasive uses of email and information sharing via the web have radically changed corporate meetings. Where people used to meet to update each other on status or to blandly replay the team's recent activities, meetings are now held to take that information and make decisions or plot a course of action. Meetings now require the most up to date information, so it is not uncommon for individuals to use networks and laptops to access this data during the meeting. And the need to get results of a meeting to many others as quickly as possible means that recording action items or notes on a laptop is a common scenario. Yet our tools can make this very difficult. We make it very hard to keep all the important elements of a meeting together (notes, presentations, handouts, action items, or the attendees names/contact information) and we do not help in the communication of results or follow-through in any substantial way. Office.NET combined with a team web can provide a revolutionary mechanism for making this form of team communication more efficient.

Sometimes members of a community or team are separated by geography, or just a desire not to leave the office. The internet provides the infrastructure that makes it possible to support electronic meetings and we have not taken advantage of this in a way that customers would call "effortless." Today's client-focused tools such as NetMeeting provide a taste of this experience, but services such as WebEx solve the really hard parts of the problem such as establishing a meeting place, keeping a record of the meeting, working through a firewall, and providing the infrastructure for broadcasting to any number of attendees at any time, or simultaneously. The opportunity to provide a service that allows meetings to be more interactive, have built-in feedback, as well as an archival record is an exciting addition for Office.NET.

Finally, instant messaging (IM) is an incredible phenomenon that long ago swept through though home market and is now becoming an integral part of the workplace. The opportunity to integrate IM throughout the core Office.NET experience is huge and provides customers with a new view of how to do even the most simple tasks, such as deciding ownership of an action item, giving (and recording) an approval to a memo, or reviewing and revising a document.

GROWING NEW OPPORTUNITIES FOR OFFICE

We sometimes take the contents of the Office box for granted or even assume the contents are fixed. This has not ever been the case as over the years we have greatly expanded the definition of what it means to be Office—we've added applications such as Outlook and FrontPage and changed the definition of categories by including, for example, data analysis in Excel. A major focus of Office.NET is to broaden the role of Office in everyday work. The way we will accomplish this is by providing new services, and features, as part of Office.NET. The key to adding these new services is that we are creating an opportunity for Office to become an even more important part of a customer's computing experience.

This planning focus area proposes that we dream big dreams and think "out of the box" about the new software, not just services but applications, we could write that will allow us to enter new businesses and most importantly generate increased revenue.

In terms of services, there are many possibilities. We will investigate performing routine services that are point services in today's internet world. We might offer to scan files stored in the cloud for viruses. We might offer to convert a document from a rich Office format to information that can be summarized on WAP devices. We might detect corrupt files you store in the cloud and repair them on the server. We might expand our offering by partnering with experts in fields and offering our subscribers easy access to experts that can help prepare your presentation or financial model, or even edit a manuscript. We might create an entire "affinity" program for third parties much like our eServices, with the addition that we charge the third party for the presence within our service. There are numerous services being built by Microsoft that target people getting work done, and Office.NET will expand to incorporate elements of those when appropriate. This will be especially important when services expose important data using formats such as XML that we can consume or when those services accept as inputs our Office formats of HTML and binary. One example is the bCentral service targeted at small businesses—businesses that are big users of Office. Office should be able to provide reporting and analysis tools for the customer and financial data that is part of bCentral, for example.

Office.NET also gives us an opportunity to attack the "vertical markets" that we have always held back from because the cost of distributing these solutions broadly on a CD was too high. We have been partnering with SAP for some time now, but we can and should do more. We can provide an SAP gateway that will securely work with your intranet's SAP data or we could just provide SAP components that integrate Office and your company's SAP data that you can download and seamlessly use with Office.NET at your desktop. There are many enterprise applications, such as Siebel, Great Plains, etc., that provide great business opportunities for us if we do the right level of partnership.

We will also explore exciting new applications, whether those applications are client code, server code, or a combination. Two important areas to consider are supporting the tablet PC and reinventing the desktop database.

The tablet PC is going to be an incredible opportunity for our customers—it provides a form factor that allows for quick note-taking, high mobility, and great readings. Office should provide world-class software that makes that better. One application we will consider is building a "thought processor" (an old buzzword from the innovative product *More*). This tool would be used as the primary application on a tablet PC that allows you to take notes, record your thoughts, and arrange them into meaningful structure—all using a pen or a keyboard when available. And when the time is right, this application would seamlessly work with Word, Outlook, PowerPoint, and Excel to move your thoughts to the best tool to finish the job.

The desktop database has been largely unchanged since the advent of the PC. Office.NET needs to bring the desktop database and the web together in an incredibly novel manner. The *list* is a fundamental data type used in the office, and yet we have not done much to make this easier—especially if that list is shared among a group of people. SharePoint (OWS) created a list data type for use in team web sites, but today it lacks the most basic feature such as joins or duplicate removal, as well as reporting or analysis (though it can integrate with Excel). There is so much power in this concept that many might be willing to give up the power of Excel and Access to use these features. We should enhance the list type by providing the powerful tools we have in Excel and Access today, but doing so in a completely new service-centric model. For example, today a huge number of databases have phone numbers in them yet we provide no help when an area code changes—there is an excellent opportunity for us to provide not just a solution to this single problem, but a vast array of richer data types that have been on the wish list for customers for a generation (validated addresses, real-world data such as population, standard schema, etc.)

A NOTE ABOUT "TRADITIONAL" FEATURES

We will continue to get lots of demand from our customers to add "traditional" feature improvements, but we are making a bet on a new type of product (a service focused product) and a new set of scenarios beyond typical "document creation." But as we said, even if we grant 100% of those customer requests it is clear we will not be doing the right thing for our business. Just because we have a very strong focus on services, does not mean we will be inflexible to the point where we turn away great end-user features just because they do not have a clear services component. However, we also know the engineering realities of changing too much of our existing code, so we will not allocate a fixed amount of schedule time to these types of features. And, the bar will be super high, similar to our Office 95 efforts (where we limited our development to a small number of high-appeal, high-profile features like background spell checking). First, we will look to our services-targeted focus areas as the source of "traditional" innovation, where in the course of solving end-to-end scenarios we are likely to introduce basic functionality enhancements independent of services (such as when we improved basic scenarios because of our Web and TCO focus of Office 2000). Beyond that, we will think carefully about which features we consider. They must appeal fundamentally to a high percentage of Office users and have a well-understood impact on the code. At our first check point, we will review Office.net along "traditional" lines and evaluate our plan to see if we have enough or need to add more. The implications of this are twofold. First, we should not be thinking that postponed Office 2002 bugs will get fixed for Office.NET, and we must also avoid the temptation to use MM0 as a way to add features that are not consistent with our Office.NET goals.

INNOVATING OUR TEAM STRUCTURE

To build Office.NET we will focus on a number of core scenarios. In the past, the overwhelming focus of our effort was the "EXE" or "MSO." This is no longer the case, and as such we will need to adjust our processes and our team organization to have the right level of focus on these new scenarios.

Yikes…a re-org. Yes we will organize our team to address new challenges. Office has a history of first defining a new problem, second understanding the solution we are aiming for, and **then** re-organizing to solve the problem. We are still months from a solid understanding of our solution space and product vision, and thus not ready to announce a reorganization. There are a few principles that I hold very strongly regarding any organizational changes. First and foremost, form follows function. We don't just reorganize to shuffle the deck or make sure everyone is awake—changes in organization are hard and they take time, so they need to make sense. Second, changing the organization is something that everyone is involved in, not just the senior managers or VPs. When Office reorganizes it is a chance for everyone to take a breather and look around for something new to do, and team members encouraged to do so. Third, reorganization affords the chance to more clearly define ownership and boundaries so that everyone can be more productive and more clearly understand the successful partnerships that make up Office.NET. And finally, reorganization is not about "winners" or "losers" but about doing our best to easily and efficiently build the product for which we are aiming

The key asset of the Office organization is the people that make up our team. Everyone will have an important new role in building Office.NET. At the same time, many people will have new skills to learn and new opportunities to grow. In that sense this is a little bit scary. But everyone that is on our team is here because of the "brains" or "IQ" that they bring, not because of specific technical skills they had when they arrived their first day of work. Web and service development are not harder or easier than what we do, but they are different; just like C and Windows were not harder or easier than whatever we each did in college, but they were different. Our development efforts will not be simply writing static HTML pages; as anyone who has written real service offerings will tell you, it is the combination of those pages along with high-performance, scalable, robust server code that really make the difference between a successful offering and a failure. From managers, to new hires, throughout our organization we will have new things to learn, but our core assets, those brains and IQ, are the primary reason this transition will be achievable.

Engineering Office.NET is going to be different than any past release. We are simply not building Office 11. We will no longer think in terms of Apps and MSO. Those are old terms and an old way of thinking. When you hear "integrate with the apps" or "put that in MSO" then you are talking about our past releases, but not Office.NET. Office.NET is about services, and although not everyone will be working on server code, everyone will be working on features that enable services, integrate with services, make services better, as well implement services. A core to success in building services is that when you work on a service your team owns that customer experience. This does not mean you have no dependencies—recall the value of integration—but it does mean that a service has to stand on its own and deliver real value out of the gate.

At the same time, we cannot lose sight of the assets that are Office. The EXEs and DLLs are how customers experience the product and how documents are created and edited, but as we know they are also hard to modify and hard to ship. Each team will continue to manage and own the appropriate parts of traditional Office code, and will insure two things. First, we must have an appropriate level of development and testing on this code and second we must invest only in features that vastly improve the experience of using services. We will be working on a tight schedule and touching a lot of our core code is not compatible with such a schedule.

It is important for us to focus on very clear customer views of the product. This used to be the box or the application and over time we organized around shared code and aggregations of related apps. Office.NET is about services so we will use that as a way to structure our efforts. Just as word processing, graphics, or spreadsheet defined core experiences, we will define new experiences that will last many releases and we expect will become major businesses. These are experiences that are broadly applicable to Office.NET customers: Core experience, Digital Communication, Community, Web Data, Sustaining Engineering, and Opportunities for developers and partners.

- **Core experience.** The core experience defines those scenarios that cross all of the scenarios in the product. In some ways, you can think of the core experience as the service version of MSO. Some of the features our core experience will include enabling the subscription business model, supporting web deployment and setup, customer profiles and settings, managing my personalization information, and the content provided to customers on an ongoing basis. The core experience is all about really getting "Office" right and insuring that we have a consistent look, feel, and operation across the product. What we call *operations* is an incredibly key element of the core experience and probably much more sophisticated than you might imagine if you have not been part of some of the operational elements of Office 2002. As we plan and spec Office.NET many will learn and become involved in the operational elements of their features. Ops is not something that you can throw a feature to, but is something that must be considered from the inception of a feature. We will increase substantially our investment in operations as we create services. From a customer experience point of view, operations is as much a part of Office.NET as any "feature."

- **Digital Communication.** Digital communication is about building the necessary services to support scenarios such as interactive and productive online (and offline) meetings and note-taking and follow-up. These scenarios are all about building new areas where Office.NET can add substantial value and provide solutions to many common problems faced by our customers.

- **Email, calendaring, and personal information.** Email is such an important aspect of digital communication that we will make this a separate scenario so that we focus on improving the email experience in deep and innovative ways. We will focus on access to email and PIM data from browsers and synchronization of data with other devices, as well as providing intelligent mechanisms to manage that information. We will also make it easy to share this valuable information with others in a secure and robust manner.

- **Community.** Today each installation of Office stands by itself, connected by a few strands of shared files and email messages. Office.NET builds a notion of a community that is part of the experience for every Office customer. A community allows customers to have a place where you can store your private information as well as information you want others to see as you grant them permission. You can create views of the information on your community, or even integrate that information with data in other communities. And of course a community can be customized with a powerful tool like FrontPage.

- **Web Data.** The web is an easy place for customer to get at data, but a nearly impossible place for customers to put their data. Whether it is a list of customers or products, or important financial information, we will make it easy for customers to use a hosted service to maintain their data. We will also provide ways for customers to create databases and more important to incorporate data from other sources on the web (sources we provide, or sources available as XML). We will also provide ways to view and report on that data through our tools that can consume XML or display XML results. There is a huge opportunity to pioneer completely new scenarios around databases that enable average customers to finally tap the power of the relational model.

- **Sustaining Engineering.** The MOSE team has been engaged in real-time engagement with our customers since Office 2000 shipped, and in a services world this type of work becomes even more critical. As we learn from supporting DW we will see an increasingly critical role our sustaining efforts will assume for customers. The ability for us to diagnose critical problem areas and deliver updates to customers in a seamless manner will ultimately determine our success at offering the most understood element of service—keep Office free of major bugs and keep it running. As we focus our efforts on code outside the EXEs, MOSE will also assume the critical role of delivering customer satisfaction updates and helping us to win customers by removing deployment blockers.

- **Opportunities for developers and partners.** Office has always provided a programmability platform for customers and Office.NET will as well. The key difference is that we will shift from a model that allows customizing our client code to a model that allows customizing the service experience. We will need to find ways to offer developers an opportunity to add value to services by adding some new code or content to a subscriber's profile. We also have a huge opportunity to integrate partners into the subscription process where partners can provide vertical services, expertise, or additional features. There are many challenging issues we face in providing these opportunities, but it is a key metric of our success. Building opportunities for developers and partners is hard, but we will focus energy on this experience.

As we develop the vision for the product and have a deeper understanding of Office.NET's features, the specifics of the organization will become clear. There is much work to be done and time is better spent understanding what we need to work on rather than the organization of that, since form will follow function.

BASIC TENETS

As with every release, we will have some basic tenets. A tenet is something that we assume to be true and will not spend time discussing or reconsidering.

- **Quality and reliability wins over features, every single time.** We have spent much time during planning this release learning about operations and how important it is that we understand these new complexities. We will take this learning and it put it practice as we make design and implementation decisions that have quality and reliability as the first priority.

- **Simple and efficient services**. We are going to build services that are efficient in their use of server resources. Just as with any type of resource, it is easy to consume too much in our zeal of accomplishing a goal. The multiplier of having a service that requires too much in the way of server CPU, disk, or bandwidth is extreme and will quickly erode our profits. We must think about simple and elegant services that scale because they are clever in their use of software, not extreme in their use of hardware. We will adhere to simplicity on the client side and in the browser as well. Simple pages load fast and are robust—Yahoo! has a 40K limit on pages (3 seconds) and so will we.

- **Crawl, walk, run.** We will not try to do everything in a single release, but we will build in feedback mechanisms so we know where to focus our efforts in the future. We will start off by building a service and operating it reliably and at scale. We will then focus on integration with the Office applications. And finally, we will create new types of services that build on that foundation. We did not build MSO in 1990, so there is no reason to think that our first service-oriented product should have the depth of architecture or features that we show in Office 2002.

- **Don’t touch core code.** We will build Office.NET assuming that we can do so without invasive changes to our existing code base. As we evolve our ideas and create the vision, we will look to add features to existing code that absolutely maximize the value of the services (not just enhance them a bit) and do so in a way that does not jeopardize backward compatibility or our own ability to ship in a predictable manner. Clearly some elements of Office 2002 are already focused exclusively on working with services and we will continue to move this code forward.

- **Shipping in a short time, and on time, matters.** We have not set the schedule yet for the first release of Office.NET but the working goal is a schedule that runs for 12 months from the time we start the schedule. We can hit this schedule if we focus on building in new code bases, not building quantity of code but quality, and working in our existing code only when we have the most bang for the buck.

- **Rolling release plan.** We are not going to immediately transition to an “internet time” or rolling release schedule. The first release of Office.NET is about building a critical mass of services and releasing them all at once. As we mature our services offerings we will of course choose to stagger our release schedule, which will be especially important once we start leveraging the feedback loop we will build into our services, but we need to learn the ropes and we need to gain a critical mass of service offerings.

- **Platform requirements.** Office.NET services need to be available to people wherever they may be. We will assume that customers have access to an Internet Explorer 5.0/Navigator 4.x browser for any PC browser-based services. The core Office applications will run on the same system requirements as Office 2002. When we use .NET services on the client, it is acceptable to require the same OS requirements as the .NET service without providing a down-level version of that experience. Office.NET is a forward looking release, so we will not be hardcore about providing everything on older platforms. We are going to be hardcore about being able to get to crucial *information* from any machine, anywhere—that is a core element of the Office.NET experience.

- **Web sign-up experience.** We will support a reasonable web purchase experience where someone can register and download the client-side part of Office.NET (perhaps incrementally if that is appropriate). A good example of software that accomplishes this is MusicMatch where you receive a key to "activate" the product. Of course the download size will be significant, but we will set a goal of making that size something that takes about an hour on a reasonable DSL or cable modem connection (100MB of download). We will have an "up and running" metric for customers that do not have Office.NET on their machines (and we will do everything we can to support offering free CDs and pre-installation of our code, assuming we can overcome the piracy issues).

- **Building the right code.** Our goal is not to build just server code or just client code, but to solve the customer problem we set out to solve in the best way possible. We need to be excellent engineers.

- **Hosted by Microsoft.** Office.NET services are hosted by Microsoft. We are not building a server application that can be installed inside the firewall. Instead our services should function flawlessly through corporate firewalls. We will offer features that corporations will find attractive such as isolating their users from others, keeping their users identity invisible, and providing some levels of customization such as a branded site. Users who use Office.NET at home will appreciate being able to combine views of their personal information with their work information, such as their calendars.

- **Appropriate worldwide product.** We are betting on bandwidth and on services, which might be an aggressive bet in some markets. But this is a big bet and we are betting that over time, probably a much shorter time than it took the GUI to catch on, the worldwide marketplace will be ready for Office.NET. We will look to local markets and where we become confident there is a unique local business need and we feel we can build a wise business around a market-specific offering, we will exploit that opportunity. Our default is to build and operate a worldwide Office.NET, since our experience has been that when we do an excellent job identifying market-specific features we can generalize them to the worldwide market. As we have done with Office features in the past, we might also decide that some elements of Office.NET are offered only in our larger markets.

- **Microsoft's .NET Platform.** Office.NET will be on Microsoft's .NET platform where applicable. We will be the first customers of many of the .NET innovations so it is clear we will have some work to do in driving the requirements and making sure .NET is real. We will not reinvent or duplicate services that are being done by .NET, such as billing and passport logon. We will also find clever new uses of .NET services such as the MSN team did by using Communities to store user's Favorites. The .NET platform will also include new user-interface elements, some of which we are pioneering in Office 2002, such as speech that we will continue to support.

COMPETITIVE LANDSCAPE

On the market today there are an incredible number of innovative products and services targeting the "knowledge worker." In the past we would have said that these products do not necessarily compete with Office because they do not offer a word processor, spreadsheet, etc. But we are redefining Office with Office.NET and many products and services—those that are critical to getting work done and are broadly applicable—now compete with Office.

The Office team has always done a fantastic job focusing on competitors, understanding not just the technology but how customers perceive a competitive product and ultimately how to build a better offering. We must again embrace this methodology and competitive zeal that perhaps has been toned down because of the relative strength of the traditional Office suite in the marketplace.

Product planning has been tracking a list of interesting products and has been creating competitive demonstrations and memos. This information can be found on http://ngo/lists/competitors/allitems.htm.

The following products and services are ones that I feel are particularly compelling. These are a set of products that everyone should try out—the web is great because you can try out a product in a few minutes. You should try to use these products or incorporate them in your daily work. I think you will be surprised how easy it is to become "addicted" to a powerful service offering that really solves a problem or provides a unique value.

Product or Service	Cool Aspects To Notice
Yahoo!	Demonstrates better than any service the power of integration. The companion.yahoo.com add-on shows how you can provide a single entry point without the complexities of a home page, and this entry point is customized as part of your "service" using a very nice web-based customization. Check out the unified address book across a broad range of services such as the briefcase, email, calendar, greeting cards, etc. Be sure to note the consistent and simple interface across an incredible range of services.
fusionOne	"Sync is everything" is the advertising of this company. The service replicates your important email and contact information from Outlook to your handheld, cell-phone, or another PC. It also maintains an up to date copy on the web so you can access this information from any browser.
GuruNet	Like SmartTags, GuruNet provides "right click" access to relevant information about a word or phrase in a document (or anywhere on the screen). Professional writers rave about Guru because it brings together the best of the web within the context of Word.
Evite	Most of us have probably already been invited to an Evite gathering. The simplicity with which you can create an event, invite people with any email address, and build a community around the event (who is attending, what to bring, etc.) is awesome. This type of service shows how there is opportunity in melding work and personal life since everyone wants to have a single calendar. It also happens to be incredibly useful for coordinating with outside partners and customers.
WebEx	Rather than taking the task of broadcasting presentations and building it into PowerPoint, WebEx took broadcasting presentations and created a service, which also happens to use PowerPoint if you'd like. WebEx hosts meetings, software product demonstrations, and allows for feedback on presentations. It is becoming incredibly popular within our industry and with people that do corporate training.

Product or Service	Cool Aspects To Notice
Zaplet	Zaplets are an example of “always live documents.” When you use the Zaplet service you create a database (a flat list of issues, people, or anything you want—similar to an OWS list). You can mail that list to anyone and upon opening the message the reader sees the most current contents of the list as it is fetched from the service site (or a cached copy of the list if they are offline). And if the recipient has permission he/she can click on a button to add, edit, or delete items directly from within the message. Zaplet is focused on core business processes, even though they demonstrate their service for consumers.
AnyDay	AnyDay is one of many services that provide a hosted calendar where you can store your appointments, schedule meetings, and also synchronize your calendar with a handheld device.
WebBasis	WebBasis is the latest breed of integrated calendaring and email hosted as a browser based service. It includes a whole suite of services (messaging, virus scanning, encryption, secure email, etc.) One interesting element of WebBasis is that is allows corporate customers to “brand” the site for their own use when used by employees.
Napster	What more can be said about Napster? If you haven’t tried it, please do—but don’t steal any music! One thing to notice is that there is nothing in Napster that is inherently tied to music, at least technically. What is there for the Office.NET user in a peer-to-peer (sorry, P2P) product like Napster? There are several file sharing products out there already, such as Aimster that integrate more traditional scenarios with P2P.
America Online + StarOffice	It goes without saying that the ultimate service product (in terms of customers, profits, revenue) combined with a desktop suite spells trouble for Office. It is important for everyone to keep a close eye on developments in StarOffice and if you have not tried the product then you should—Star is quickly becoming quite compatible with Office and for some customers a viable alternative to Microsoft Office.

NEXT STEPS

This document provides a “turn of the crank” so to speak and describes what the Office.NET product and service strategy is all about. It is hard to overstate the opportunities ahead of us:

- Office.NET will offer customers a radically useful new set of tools for getting their work done.

- Office.NET will grow the Office business and will be more connected to customers through a feedback loop than any other product we have worked on.

- Office.NET will offer everyone a chance to learn new skills and push their own abilities in the development process. Everyone will become an entrepreneur as we develop new products and services and bring them to market.

- Office.NET will be more fun to work on than any previous release of Office. Shipping is a fun part of working on Office, and Office.NET is a product that will allow us, ultimately, to ship more features to our customers on a more regular basis.

There are still many unknowns beyond the primary focus of this memo which is framing the feature set and honing in on the vision of Office.NET. In particular there is much work to be done on building the development, testing, and releases processes that will enable us to deliver a new product and a functional, scalable, and reliable set of services in a predictable manner. I am not underestimating this task and it will take significant time to re-engineer our core processes around Office.NET.

To better prepare for the challenges of Office.NET there are some tasks that everyone should spend time on over the next months, while builds are compiling or Office 2002 is installing.

- **Technical training.** Since there is so much new technology to learn and so many new skills that we will need to build services, we are going to provide some "structured" training on core .NET technologies such as Active Server Pages (and ASP+), Windows 2000 Server, and SQL Server. Many on the team have significant experience in some of these areas, so for them this might be a little redundant. This technical aspect of this training is being coordinated by Antoine Leblond and will be offered at the start of 2001. All developers and testers will be eligible, and there will be room for some program managers as well.

- **XML**. One of the core technologies being used by the entire Microsoft .NET initiative is XML. Office.NET will make extensive use of XML as well. XML is not magic—it is just a text file after all—but it provides a level of consistency and sharing across disparate parts of Microsoft. XML is also going to be the basis for cross-group protocols and data sharing, sort of a universal clipboard meta-format. There is much to be learned for developers and testers in the area of XML, and program managers should become familiar with expressing data formats and schema in XML. There are dozens of resources on MSDN and many third party books on XML that are worth looking into.

- **Live the web.** In order for us to do a world-class job on Office.NET we must build a truly web-centric experience. A web-centric experience is not browser-only, server-only, or Microsoft-only. But rather it is the best experience you can deliver for a customer that solves a customer problem. There are many excellent examples in the market place that show how the web "perspective" is different than the "Microsoft" perspective at solving customer problems. The web is not better at everything, and sometimes forcing the web everywhere is just plain klunky. But the best way to become fluent in the language of the web is to live the web—schedule meetings on Evite, look up topics on Guru, keep your files in a Yahoo! Briefcase, don't carry a laptop on your next trip, host a meeting on WebEx, use IM to get work done. There are dozens of ways for you to experience the services world and I would encourage everyone to do so.

And finally, over the next months every single person on our team will need to do three things:

1) Focus on shipping the highest quality on-time release of Office ever with Office 2002.

2) Exhibit some patience as everyone works hard to pull together all the details of Office.NET. This is a tough job—Office is a hugely important part of Microsoft and we will work deliberately at every step as we engineer the vision and processes for Office.NET. We, and certainly I, do not have all the answers and there will be some unknowns and some bets we make along the way.

3) Participate in the creation of the Office.NET vision by working with program management and product planning. Office.NET will be great because it will be a product of everyone on the team.

# # # # #