From: Chris Jones [chrisjo]
Sent: Thursday, November 03, 1994 6:14 PM
To: Adam Bosworth; Bill Bliss; Bill Gates; Bob Atkinson; Bob Muglia; Brad Silverberg; Brian Fleming; Brian MacDonald; Bruce Jacobsen; Chris Jones; Chris Peters; Chris Zimmerman; Chris Williams (WPG); Darryl Rubin; David Cole; David Stutz; David Vaskevitch; Dawn Trudeau; Denis Gilbert; Duane Campbell; Ed Fries; Eliyezer Kohen; Hal Berenson; Hollace Kennedy; Jim Allchin; John Ludwig; Jon De Vaan; Lewis Levin; Mike Maples; Paul Osborne; Pete Higgins; Peter Spiro; Pat Helland; Phil Bernstein; Peter Pathe; Rob Price; Roger Heinen; Scott Randell; Steven Sinofsky; Tanj (John G.) Bennett; Tom Evslin; Tom Reeve; Tony Williams
Cc: Chris Jones; Paul Maritz
Subject: Agenda and Pre-Readings for Offsite

Here's the agenda for the Nov. 8th and 9th offsite. It will be held in the Vashon room. Please block off your schedules (if you haven't already) for 9:00-5:30 on Nov. 8th and 9:00-1:45 on Nov. 9th.


AGENDA.DOC

You should plan on attending both days as the purpose of these meetings is to get people up to speed on the key plans of different groups, find cross group synergies, and generate new thinking about ways we can work together to address customer needs.

Additionally, here are some pre-readings, which some of you may have already seen before. The first is a memo from BillG on changes in the framework of computing.


SEACHAN.DOC

The second is a document from DavidV on the future of databases and computing.


SIGMOD94.DOC

The third is a set of slides from "Phase 1" of this process, held 9/16/94. Some people have seen these, but I've included them for those folks who have not. You can find these on \\NTSRVR\INFO\plan94. The slides are fairly long and detailed, but try to drill down in areas where you have questions or particular interest. The following document gives an overview of each of these presentations, their authors, and the contents.


OUTLINE.DOC

Please let me know if you have questions or comments.

Thanks,

Chris

Nov. 8/9 Offsite Agenda

[illegible]		
9:00-10:00	Overview of Goals and Priorities	*PaulMa, BillG*
10:00-11:30	Business Systems (Server, OLE, Workgroup)	*BobMu*
11:30-11:45	Break	
11:45-1:15	DDT (Part 1: Server Side)	*DavidV*
1:15-2:15	Lunch	
2:15-3:45	DDT (Part 2: Client Side)	*HalB*
3:45-4:00	Break	
4:00-5:30	Personal Systems (Client, Communications)	*JohnLu*
[illegible]		
9:00-10:30	Office	*JonDe, SteveSi*
10:30-10:45	Break	
10:45-11:30	Consumer Division (Titles, Games)	*BruceJ*
11:30-12:15	Consumer Division (Productivity, Utopia)	*TomR*
12:15-1:15	Extra Time (if necessary)	
1:15-1:45	Wrap Up	*BillG, PaulMa*

To: Steve Ballmer, Mike Maples, Pete Higgins, Jeff Raikes, Bernard Vergnes, Richard Fade, Joachim Kempin, Mike Brown, Hank Vigil, Lewis Levin, Chris Peters, Peter Pathe, John Neilson, Brian Fleming
From: Bill Gates
Cc: Executive staff, Executive staff direct reports
Date: October 6, 1994

"Sea change" brings Opportunity

Among our future challenges is the high percentage of office workers and homes who will already have an "Office" solution and are no longer candidates to be new users. Already in a number of our large accounts we have seen major sales years when Office is widely deployed and then a drop in sales to a much lower level at least for the DAD products. The solution to this is to get more revenue from our installed based. If we can get high percentages of our users to buy upgrades our business will thrive. The challenge there is "adequacy". Some people feel we have already gotten to the point where most users will not benefit from updated Office applications. Although we can do a better job on this in the short run "adequacy" will limit our penetration. However, over the next decade I believe we will see several "Sea changes" which will drive major waves of upgrades. This is an optimistic point of view that struck me during this Think Week. Its new thinking - at least for me and I think it leads to exciting opportunities.

An imperfect analogy is the consumer electronics industry which has seen major waves not only of hardware sales but software sales including old titles as new formats like CD come along.

Starting sometime after 1990 the move to graphical computing has been a "Sea change". Although the vast majority of Wordperfect user would have said their product was quite adequate at the start of the "Sea change" every year a higher percentage of those have moved across to either Windows Wordperfect or Windows Word. Because it took several tries to fully exploit graphical word processing and match up with the latest operating system users who switched by 1992 will have bought on average at two major upgrades. The graphical computing sea change has played out over a period of 6 years creating immense share and leadership opportunity for the software company that saw it coming and helped make it happen (Microsoft). By 1996 Office users will spend an insignificant amount of money on DOS applications and even the diminishing installed base will know they are "dated".

Arguably the shift to an integrated Office approach is another "Sea Change" which we help caused and benefited from. However it is not as clear cut or total as the move to graphical interface. No matter what you consider the start date of this shift, 6 years after its start there will still be significant numbers of users buying and using standalone word-processing and spreadsheets. Anything we can do to drive the Office percentage up is very helpful to our strategy.

I believe we are in the midst of another major sea change which is the move to electronic communication with office documents. In the past PC software users created most of their own input and did their output to a printer. During this decade a very high percentage of input will come across private networks (another terms for corporate LAN/WAN) and public networks (including Internet and online services). The information coming across the private network will include business information created to review sales, budgets, personnel, customer service and every other aspect of the business. Word must become a great authoring and reading tool for electronic documents. Excel must blow away the competition in being a viewer for corporate data by tighter integration to databases and extensions of features like pivot tables. We need to make sure public networks include lots of documents best viewed with Office. The product approach for this is complex and multifaceted including things like supersetting Internet features and providing free subset readers. The basic point however is that users expectation of what Office applications will do is changing and 3-4 years from now anyone forced to use the software we have today would find it completely inadequate for dealing with the electronic world.

This sea change like others provides opportunities for new challengers as well as our familiar rivals. Extended Web viewers from startups will grow to provide Word with new competition. These competitors will ridicule the number of commands and features Word brings from its past and suggest it is not the right tool for the new usage model. Embarrassingly we find ourselves somewhat behind on of our old rivals in providing both the system (replication, security) and application (views with categories, @ expressions, multivalued fields, flowing forms) elements for basic workgroup sharing and so Lotus is recognized as a leader in moving corporations into the benefits of corporate wide information sharing. We can move out in front of this sea change but it will require a focus and an overhauling of parts of our interface and coordination between systems and DAD beyond what we have had in the past.

In a recent meeting on Office96 there was a discussion of whether the priority should be designing for our installed base or for our competitors installed base or new users. Some math relating the size of these groups, potential penetration and price suggested a focus on the installed base. Although its an interesting calculation it is absolutely the wrong framework to consider our choices in. We believe this "Sea change" is inevitable and are willing to bet all of our success on it. We must optimize for being the best product for these new scenarios even if that means causing disruption in our user interface or compatibility that will cause existing users to wait longer to buy an upgrade. Very few users will switch to a competitive product for non-"Sea Change" related features (unless all of their cohorts are using another product but that is the subject of another memo). Due to the "Sea Change" they will buy an upgrade - the only questions are whose and when. Winning the "whose" is far more important than winning the when. In the early 1990's Lotus surveyed their installed base and found limited desire for graphical interface. By the time it showed up in the surveys it took them too long to respond and users were willing to switch. Microsoft bet on the "Sea change". It takes even more guts to bet on the "Sea Change" when you are the market leader but it is the only way to position yourself for massive upgrades.

Lets do some math on the "Sea change" opportunity. Our installed base has not peaked. My exhortation about studying the saturation phenomena is not to say I believe we are at the peak. In some countries we have only scratched the surface of the new user potential. However we should understand the potential for new users at least on a per country basis. Lets say over the next 2-3 year we get our high end applications installed base up to over 24M users. Lets assume that during the peak 4 years of a sea change 30% of those users buy 3 $150 upgrades and 30% buy 2 and 30% buy 1. This generates $1.6B per year which is almost the size of our current business. During those years we will also be deriving revenue from new users, addons, and new products. With the kind of discounts we are providing right now the $150 might seem high however an upgrade which provide "Sea change" benefits is worth more than an upgrade which only provides more functionality without a "Sea change". Calling these changes to the product "upgrades" may be misleading both internally and externally. We want to draw on our installed base but we want to take them somewhere new.

The "Sea change" to electronic information sharing is a particularly important one because it will bring us closer to our customers. It will also bring our competitors and free software closer to our customers. The effort to learn about upgrades and to install them will be much lower than it is today. Lots of low cost and free software will be easily distributed. Memory and disk size will outrun even our prodigious ability to create demanding software making it easy for developers who are don't spend as much time optimizing to provide adequate products. Although its something to be watchful of I don't think new entrants will be able to redefine the categories enough to take Office out of the mainstream. The value of having the best software will be even greater because of the new scenarios.

Electronic information sharing is not the final "Sea change" that we can see ahead. Microsoft has always assumed that hardware advances will be incredibly rapid and that assumption still holds true. It is critical that we look out ahead to see what other "Sea change"s are coming. There is no rule that says only one takes place at a time.

One "Sea Change" that is still at least three years away but probably not more than 6 years away is the move to extensive use of voice input. This will catch on even more rapidly than graphics interface did. This

will have a deep effect on Office. This is one we should be spending time on today. If a computer had perfect speech recognition how would we choose to work with it? What combination of keyboard, pointing and speech would we use? Of course the early speech devices will be imperfect so we will have to pass lots of context to the voice recognition module from our applications.

I still believe strongly that once a tablet sized computer has the right accuracy and physical characteristics that pen based computing will be successful and that either mainstream applications will address this or a new category will emerge. I believe that linguistic understanding and expert systems will find there way into productivity software over the next decade. I am sure Nathan will provide further thoughts on "Sea Changes" to come.

These "Sea changes" will not affect only Office. Windows also has the opportunities and challenges of all of these changes.

A static view of the world of technology badly misleads one in understanding the value of our installed base and continued commitment to renew our products. I have been guilty of this myself. Articulating the "Sea changes" that we are betting on and preparing the company for them is the most important and exciting part of our work.

Database in Crisis and Transition

A Technical Agenda for the Year 2000

Are Databases and Database technology at the center of the information rich computer world of the future, or, ironically, are databases about to become literally irrelevant just as that future arrives? If there is a problem, does the solution require more of the same kind of research and development that has characterized the last two decades in database land, or is there a completely different set of questions, in addition to the standard ones, that need to be considered to ensure that databases stay vital, relevant and central to the world? Where are databases going, and how is different from whence they came?

Clearly, at one level, databases and database technology are one of the key pillars of the modern computer world. Banks, manufacturing companies, airlines, and organizations of all sizes critically depend on their dbms's every day and in every way. Database products like DB2, Oracle, SQLserver, IMS, and their kindred like CICS, account for billions of dollars in revenue each year. Finally, no computer science department, no large research group, no serious consulting firm, no systems vendor would be considered complete without at least one major database group.

At the same time, the database industry often seems oddly out of step with the rate of change in the rest of the world; a community caught in some kind of *back to the future* time warp. For instance, most of the same organizations running the biggest relational and non-relational database systems now have more data sitting on desktops than in their db's. This desktop data, often highly structured in nature, is controlled by everything but database management systems. In fact, by any objective measure the number one database in the Western World is, not DB2, Oracle, or even IMS, but rather 1-2-3, with Excel hard on its heels. Yet, where is the database architecture, academic or otherwise that explicitly describes the integration of desktop datastores into latter day federated systems?

Continuing to examine the dimensions of this reality warp, we find a sharp distinction between distributed database the theory versus distributed database, the reality. In the theoretical world, distributed databases are completely understood. Techniques like two phase commit, federated database schemas, and partitioned queries represent largely solved problems, and distributed databases aren't even that interesting to talk about. Out in the cold, hard commmercial world, not only are distributed databases exceedingly hard to find; most practitioners consider them to not even be feasible. Which is right? Trivial or impossible? Where is the balanced view that pulls these two contradictory perspectives together?

Moving closer to the center of the database universe is the issue of the data model. Here both the academic literature and the commercial vendors are more or less equally out of touch with reality. In one corner of the ring are all the relational vendors, representing a healthy $2B industry, convinced that their products own the world. To attend a client / server seminar is to come away convinced that SQL and RDBMS's are the path to *all* production data past, present and future. In the other corner of the ring, preparing to slug it out with the reigning champ, is the OODBMS community, strongly committed to the notion of persistent languages, complex data structure and sophisticated navigation. Must we choose? Are we witnessing a generational shift in progress? Perhaps, but what about those two other contenders silently sitting in the other two corners? Which corners, you ask? Well, over on the left are our old friends IMS, IDMS, and that gang. According to Gartner group these old buddies still are responsible for almost half of all the world's production data. Wait there's more . . . In the same corner with IMS is VSAM, RMS, and more recently Btrieve. Between them the

David Vaskevitch

version of: March 17, 1994

ISAM's and pre-relational databases turn out to hold about 75% of the data really used to run organizations. Do we hear reality calling yet? If not, we need only look in the fourth corner of the ring to see those spreadsheets and other desktop database stores seeking our attention. While everybody argues about where the *real* or the *most important* production data, the desktops have grown in size and importance to the point where they hold at least as much data as all the servers, mainframes, and minis combined.

The question to ask about all these different data storer is *will the real, true database please stand up?* The database community will insist that relational databases are the present, and objects in some form are the future. Does that mean that network and hierarchical dbms's are not databases after all? And, if no database worthy of the name provides the simplicity and functionality of an ISAM, a spreadsheet, or a Paradox, what does that mean? Would the users of a spreadsheet agree that their data is not really data and their database is not one after all?

To see the dilemna we really face in its true perspective, let's consider one last reality confounding conundrum. To most database professionals, distributed data and two phase commit (2PC) go hand in hand. It is a characteristic of large systems built around two phase commit protocols that failures can easily cause the entire system to grind to a halt. That is, if there many interlocking transactions, and lots of cross node dependencies, then when nodes and communications links go down, major parts of the network will go down until the nodes / links come back up. Of course, 2PC guarantees that the whole system will produce the right results in the end, but along the way, the system can seem very fragile. The obvious conclusion is that a centralized system with duplicate hardware would have better uptime characteristics. *Isn't there something deeply counter intuitive about the idea that the distributed solution might be less robust than the centralized one?* Surely there must be some way of building distributed database systems that will be more, not less, robust than centralized ones.

All of these problems susggest that there might be some major new ways of thinking about databases that the database community might consider adopting. The result would be a paradigm shift in the direction of increasing relevance. Relevance?

The core issue behind each of the problems described above is that database technology, as it has evolved, while highly useful, is in danger of becoming irrelevant to the majority of computer users in the next century. Just as computers are ready to truly change society, just as they are on the verge of truly widespread adoption, databases might end up on the sidelines of the resulting picture. How could this be? Simply:

- If most data sits on desktops (and in notebooks) in data bases that are not databases, and
- If most production data sits either in non database stores, or again on desktops, and
- If the highly distributed computers of the future still don't have adequate distributed database technology, and
- If the distributed databases that do exist are not truly robust,

then users and developers will find other ways of managing data. They won't call the result a database, but that's what it will be. And, we, the database profession will have made ourselves obsolete.

An Agenda for the Decade

At Microsoft we are working rethinking the very meaning of the term *database*. As part of that redefinition, we have an agenda of important problems that need solution either by us or by partners we can work with. Some of these problems are described below. In a way, besides being important problems, they are also a kind of challenge to the database community as a whole.

- **Component Databases** Ironically, databases are the last major preserve of monolithic, closed design. A decision to use a particular dbms is also a decision to accept a way of managing disk space, buffers, an access method, a security scheme, a query language, an api, and

more. In short, every database, relational, object oriented, or otherwise, is its own self contained world.

The first challenge for the decade is to redesign databases around the concept of layered, cooperating, components.

- **Open Databases** Component databases, with published interfaces, are, by definition, open databases. A query processor can retrieve data from record providers of all kinds. Many kinds of query providers can be written. A spreadsheet can masquerade as a database by acting like the right kind of component. A geographical query processor can retrieve data from an underlying store just as fully as a relational query processor.

  In an open database environment, individuals can create and maintain budgets in spreadsheets, running as spreadsheets. Yet, a CFO can consolidate data across many spreadsheets (and project managers and databases) using a classical relational query tool, and seeing the whole thing as truly a database running as a database.

  In an open database world, displaying data on a map, and handling geographical queries is just as easy as handling hierarchical and navigational queries, which in turn is as easy as handling hypertext queries. What you see depends only on the query tool, the underlying data is equally accessible no matter which tool you pick.

- **Distributed Databases** How many databases will there be in the year 2000? How many computers? More than millions, actually hundreds of millions. This implies that databases have to be highly distributed. This in turn means first, databases completely self installing, self managing. Secondly it means that coordination between databases must be automatic and highly robust. But most of all, it means the distributed infrastructure must scale extremely well.

- **Processes, More Than Tasks** Classical databases and TP Monitors handles transactions and tasks that occur in real time. By definition, a transaction is viewed as an atomic activity, a single event, that either occurs in its entirey or is made to not occur at all. The real world though, is built out of *sequences of tasks* that occur over very long periods of time. Databases and the infrastructure that surrounds them must be designed to handled long running sequences of transactions.

  A network designed entirely around coordinated transactions is, indeed, less robust than a centralized system. A network, on the other hand, designed around sequences of tasks, is far more robust than a centralized system. The key is to have infrastructure that makes those sequences easy to build, reliable, and robust.

- **Rich Data Models** Normalized data is fine, when the design calls for normalizetion. Often though, more complex record structures are both more natural and more efficient. In the same way, representing many to many relationships often has risks, but it often has benefits too. Twenty years of real experience teaches us that sometimes normalized tables are right and sometimes not; databases in the future must offer that choice.

- **Databases, Not Languages** Once consequence of the component database model is that underlying database becomes a distinct and separate component from any higher level language environment. Today most modern databases are tighly bound to either SQL or some object oriented language like C++ / SmallTalk. This type of binding has strong advantages for many applications, but there are other cases where the developer simply wants the use of a database manager without being forced to pick a particular language, object model or development framework. In the component world of the future, this kind of separation becomes possible and natural. Once consequence is that a whole new class of record providers becomes possible,

each inheriting all the higher level environments. Thus the developer of a new type of project manager, for instance, by exposing appropriate methods can be a record providing component. The higher level language environments, whether SQL based or object oriented, can then tap into this record provider just as well as they do into any other.

- Navigation *and* Queries Today object oriented databases support one style of navigation; network dbms's and ISAM's support another. Relational databases, on the other hand, provide queries and set based operations. A direct consequence of the component database model is that the developer (and user) no longer has to make a choice. Lower level database components provide the same navigational capabilities as ISAM's. Higher level query processors can skew in either the set oriented or the pointer navigational direction or both. The user can choose.

  Just as importantly though, part of our agenda for the next decade has to be to recognize that both element by element, navigational style processing *and* set oriented, query based computation are equally valid. Often, the query based approach is the best way to specify a set of records in the first place, while at the same time, navigational operation is the only way to then work with the resulting data in a fashion sufficiently rich to meet the needs of complex applications. The sooner we give up on the idea of forcing a choice, the better.

- Server, Desktop, Laptop Hundreds of milllions of servers describes only a small part of the distribution model for the future. Each of the multitudinous servers will support dozens of desktops. And, many of the desktops, will really be computers that are often disconnected to become notebooks and laptops. Even today, as much data sits on desks, laps, and under arms, as on servers; in the future this ratio will shift even away from the server. In a world with billions of databases, how do we think about replication, distributed transactions, processes and the like? What does the administrative model need to look like, and how does it operate in a totally decentrlaized environment?

  One tempting, but wrong view is to think of these personal databases as somehow simple, small, or trivial. Features like online backup, real transactions, and so on might not be required, right? Take online backup . . . Do you backup your computer regularly? Would you be willing to do so if the process was completely automatic and didn't prevent you from working while the backup was in process? Is that possible without online backup? How about transactions and recoverability? Do we really believe users *want* to lose data? Can we guarantee they won't without such facilities? Yet, the whole thing has to be so simple that even a garage mechanic or taxi driver can install the database and run it without getting help ever. Quite a challenge? Absolutely, and now's the time to start thinking about it.

- Thanks for the Memory Imagine a really big server supporting several hundred personal computers. Perhaps the server has 100M or 200M of memory. How much does each workstation have? If the answer is 25M - 50MB, then how much memory do the workstations in the aggregate have? Here's a situation where the aggregate personal computer memory, at, say, 5,000MB totally dwarves the server's memory. How do we design database systems to really take advantage of this situation? OODB's do some of this, quite well actually, but how well do they do at managing large queries where the work could be divided up across several machines? And, what about relational databases; how much advantage of a two level memory architecture do they make?

  Another way of thinking about this problem is to ask: where do applications keep their private data structures? Certainly not in any classical database: too slow and rigid. Perhaps in an OODB. A goal of the nineties should be to have all applications shift a major part of their currently private data structures to the stewardship of a database manager. Why

bother? To simplify queries, management of concurrency, provide recoverability, and so on.

2001: A Database Oddysey

A salesman is about to take a day trip to another city. As he undocks his notebook computer, it refreshes his database one last time before disconnecting. On the plane, the salesman completes a territory analysis using a spreadsheet, develops an action plan using a project manager, and then decides on the top twenty accounts to use with a classical graphical query tool. Each of these tools works directly with the underlying database sitting in his machine.

On landing, his geographical query processor puts up a street map, shows where the top twenty prospects are located, and highlights the best route for making it through the day. Although the mapping program is intensely navigational (no pun intended), the saleman sees it as just another tool accessing his data in a very natural fashion. During the course of the day, the saleman makes several presentations, enters some orders, and updates a few customer records.

Throughout the day, as he rents a car, buys meals, and completes other transactions, his wallet computer (nee credit card) tracks all the transactions for him. Communicating with the notebook computer, the wallet computer also keeps the salesman's expense report constantly up-to-date.

Returning home, the salesman docks his notebook so that it can talk to his house server. Sorting through mail, he finds that his daughter was invited to a birthday party which conflicted with a dentist appointment. The server, talking to the dentist's office computer moved the appointment to eliminate the conflict, and confirmed the daughter's appointment based on it's knowledge of the close nature of the friendship. All of the underlying transaction coordination was, of course totally invisible. Reviewing monthly expenses, before having dinner, the salesman finds that his wallet computer has already updated the house server, and all his expenses are already reflected in his personal accounting system.

Finally, the next morning,t he salesman returns to the office, docks his notebook computer and starts working. The orders and customer changes entered the previous day are sent to the server which in turn communicates changes it has received back to the notebook. Along the way, the salesman receives the results of a historical marketing analysis he had launched two days ago, which involved collecting data from all over the world, collating and then massaging it. The server, data in hand, sends the final result to the notebook for subsequent analysis.

2001 is only seven years away. Is there any part of this scenario we would want to not have be true by then? Can we build it now? Clearly, if this scenario comes true, 2001 is a world where databases are truly ubiquitous, highly relevant, and quite different from those we know today. That is our challenge.

Outline of 9/16/94 Offsite [Phase 1]

Document	Title and description	Presenter
 ovw994.ppt	**Three year plan objectives** An overview of objectives for the process and expectations for this meeting.	Paul Maritz
 form0915.ppt	**Forms³** An outline of the plans for Forms³, Microsoft's shared forms technology.	Adam Bosworth
 txtplan.ppt	**Text** Outlines the recommendations and plans for text sharing.	Howard Campbell
 oledbovr.ppt	**OLE DB** Overview of OLE DB, a specification for defining database tools as COM components, along with an implementation schedule.	Tanj Bennet
 ole3yr.ppt	**OLE** Details on the short and long term plans of the OLE team, including COM, Storage, and new Linking and Embedding interfaces.	Nat Brown
 sys994.ppt	**Operating Systems** Future plans and goals for Windows, both client (Windows 95) and server (Windows NT).	Paul Maritz
 office.ppt	**Office** Office plans, both short term (95 and 96), and longer term (beyond 98).	Chris Peters
 stutz.ppt	**MS Dev** Outlines the goals of MS Dev, Microsoft's integrated develop environment which combines VB, Access, and Fox.	David Stutz
 94-3yr.ppt	**Overall Recommendations** Summary of key points along with recommendations about where we should focus our energies as a company.	Greg Whitten

EMAIL ATTACHMENTS WILL NOT PRINT