PLAINTIFF'S EXHIBIT 349 Comes v. Microsoft

112

From gregw Tue Jul 31 23:56:40 1990
To: billg darrylr jeffr mikemap
Cc: bradsi jabeb lloydfr tonyw
Subject: Re: GO threat
Date: Tue Jul 31 23:42:48 1990

The L&E stuff was not meant to be a real architecture. Our applications could not respond to a real architecture. The AppDT work will form the basis of a real OO architecture - a robust extensible data model which works with the AFX view models.

Our instances can't be viewed as containers of information today. This means that it is very difficult to implement indexing and content querying. The system would be forced into understanding file formats (we know this is not workable).

(the remainder is long)

The GO machine brings home the following point in a big way.

Once we have implemented enough interesting data types and viewers using
?
our OO frameworks and interfaces, there is no need for DOS and Windows as we know it. Instead, the file system can be replaced by a simple memory manager with a backing store to yield persistence. The notion of processes and applications disappears replaced by a single address space with concurrent threads of activity. On the notebook, detatched from rest of the world, the security of separate processes is unnecessary. There is still a need for concurrency controlled resource management (memory and screen real-estate).

Why bother with DOS apps or Windows apps as we know them, the DOS apps don't interoperate and the Windows apps are not much better.

These apps are easier to write - no file formats and I/O (only in-memory storage), few format conversions (enough to support content and queries), natural container-containee relationships, garbage collection, objects have well defined behavior (implement a set of protocols). Objects that can be queried support the content protocols. The system can enumerate all objects. If it makes sense to have a container which knows about all instances of a particulr type, this is easy to implement and install.

The GO UI is probably the least interesting part of the product from a technical perspective. Like Hypercard and some aspects of Toolbook, it will show how far graphics art can take you. Of all of our products Windows 3.0 is perhaps the best, but it doesn't come close to these other examples.
?

GO is scary but they are a small player attaching themselves to a limited hardware platform. Their distributed machine plans are interesting and very focused.

In our business there are the following things that are important -

EXHIBIT

X 531240
CONFIDENTIAL

1. ownership of the information type implementation
2. location of the information
3. diversity of information types
4. end-user ability to integrate information into new components

We need to be taking control of these 4 things with our system and application strategies.

GO (or any new platform) is going to have a hard time addressing these 3 issues. They are completely dependent on making the hardware platform compelling soon after the initial introductions. We need to understand what we think are the compelling features and have a response in the form of product and strategy. We won't be able to get a product (an OEM to support us) until we have a compelling strategy to sell.

?

The OO architected system is the key part of that strategy (we need to same pitch to go against New Wave except forget the NT-OS/2 heavy duty features for notebook computers). We need to be able to demonstrate that handwriting does not require new UI concepts and looks by making existing apps work with minimal changes. Most of the apps that people say that notebook computers need are keyboard apps that we are missing today. Other apps like the math equation app are just brain-dead - what ever their implementation, it is unlikely that it can be effectively reused at a low level in other places in the system - high level reuse is easier but the right application contexts need to be found.

What do we have going for us -

1. handwriting is neat but not as reliable as a keyboard for entry
2. handwriting computers will have keyboard options
3. with a keyboard DOS apps can run
4. handwriting/pen interface techniques integrate smoothly into Windows apps
5. information is naturally exchanged with the primary location (no unreliable format converters - using the same application)
6. diversity of applications for our environments
7. long term strategy that makes sense on the three interesting machine environments - notebook, workstation, and server. Keep the picture simple. The GO solution is weak on interoperability with the

   ?

   workstation and server.
8. new hardware fits into the big picture of office work - take home and travel (take the information with you)
9. huge ISV support for Windows apps with huge base (can evangelize when software and hardware are ready - need it soon)
10. ... (we have some more going for us - left to the reader as an exercise)

What are we missing -

1. low-end personal organizer apps (windows desktop needs these also)
2. low-end information types (we're dieing for windows works 2.0)
3. OEM hardware (may be we have it)
4. some of the below

What do we have going against us -

1. higher cost due to larger memory requirements
   (need to figure out if this is offset by lower software costs - we
   have a 2 machine/1 user licensing problem)
2. compelling features of GO UI and software
3. very focused competitors - we need a good economic model for their
   businesses - the financial bootstrapping process for them is complicated
   and could be interrupted by successfully using our advantages above
?
4. confusing endorsements by OEMs like IBM
5. ... (there must be more)

The bottom-line is that we have a compelling alternative to the GO machine.
We are having real difficulties in articulating it.

Enough said by me - use it as you like - I have other things to do.

```
|>From billg Tue Jul 31 21:39:11 1990
|Subject: GO threat
|Date: Tue Jul 31 21:36:15 1990
|
|In reviewing some sketchy stuff on the GO machine it is clear that the
|threat posed by GO is as much an integration threat as a handwriting threat.
|By using an object oriented approach they allow for searching, hypertext
|linking, an index and table of contents across all data types. They allow
|new objects to be added easily in their framework. Their shell is quite visual
|-- with foler tabs and pages. We should try and learn more about it. Meanwhile
|it is time for our L&E stuff to deal with linking and sorting. I cant believe
|we dont have this as part of our architecture when a real architecture would al
l
|allow for these things. Our handwriting group should write up what they know.
|ACtually the esther dyson article does a good job explaining what should be don
e
?
|(GO doesnt do all of it).
```

X 531242
CONFIDENTIAL