REDACTED

From gregw Wed Aug 1 11:49:24 1990
To: darrylr
Cc: billg mikemap tonyw
Subject: Re: GO threat
Date: Wed Aug 01 11:34:08 1990
Mail-Flags: 0000

> 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).
>
> All that is needed to solve this problem is a standardized set of methods for enumerating content terms and positions that any file type could supply as a dll. There is no need for the system to understand file formats. We don't need an oo framework to solve the content indexing problem today.

This solution is obvious. There are problems related to concurrency control/deadlock and in-memory instances. Solve the first and ignore the last (that is the file system way).

Printing/rendering can be done the same way. The same approach makes print servers for the different file formats trivial and viable for multitasking. Our products don't do it this way because 1) we like waiting for the hourglass or 2) we think that the print code is much faster being memory based. By the time networks are in the picture 2) is less of an issue.

X 188560 CONFIDENTIAL

You do enough of these OO interfaces as above and you have an OO framework and an architecture.

I suppose that I should have said L&E was not a complete architecture instead of real. L&E is really centered around the container/containee relationship and not the nature of the container itself. We are looking at ways to increase

the flexibility of the container/containee relationship that has simple yet seful application to compound documents - like hypertext, animation, annotation.

The content and enumeration protocols are independent of the L&E. Along with the L&E work we support protocol negotiation that would allow content information to be passed along the activated link. I thought that this work was being done in the data storage task force and document library.

From darrylr Wed Aug 1 09:48:46 1990
To: gregw
Cc: billg mikemap tonyw
Subject: Re: GO threat
Date: Wed Aug 01 09:45:24 1990
Mail-Flags: 0000

> >From gregw Tue Jul 31 23:56:47 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.

don't what this kind of statement means. L&E is the whole way that we're going to implement compound document features for the next 2-4 years. The in situ L&E extensions are supposed to make this work as seamlessly for the user as anything based on a "real" architecture. Additions to the design to handle things like hyperlinking are trivial and we need to do them. If we view the design as just a hack then of course we'll never look at it the right way and fix it to do what we need. If we view it as the strategic way that non-oo apps participate in the oo world, we can smooth out the rough spots in the design and make it good. As far as ISV's are concerned in the foreseeable future, l&e will *be* our real architecture.

> 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).

All that is needed to solve this problem is a standardized set of methods for enumerating content terms and positions that any file type could supply as a dll. There is no need for the system to understand file formats. We don't need an oo framework to solve the content indexing problem today.

| 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.

This may work in a standalone environment like a notebook computer, but nobody has ever figured out how to make it work in a network environment where you need access to diverse remote resources, with sharing and security controls. I don't assume you're mean to imply our network and standalone environments should be different.

| 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.

I've read that they are backing away from selling hardware and are instead trying to license their software to hardware oems. If this is true they are becoming another systems software company. That makes the threat a lot more real.

. think the response we need to the Go threat is to make sure we have a response in our software to anything that people will like about theirs. The stuff about hyperlinks and sorting will need to be addressed. Could tony spend some time understanding what Go has done and, in conjunction with bill's earlier feedback on l&e, prepare a list of recommendations for l&e.

REDACTED

X 188562
CONFIDENTIAL

REDACTED

X 188563
CONFIDENTIAL

REDACTED

X 188564
CONFIDENTIAL

REDACTED

X 188565
CONFIDENTIAL

REDACTED

X 188566
CONFIDENTIAL

REDACTED

X 188567
CONFIDENTIAL

REDACTED

X 188568
CONFIDENTIAL