WHY ALTO

Xerox Alto

Why Alto? Butler Lampson's Historic 1972 Memo

Butler Lampson

The following memo, written in December 1972 represents a red letter day in the history of the evolution of personal computing. It is in this document that then PARC scientist and manager Butler Lampson requests support from the Xerox Corporation for funds to support the construction of a number of Alto personal workstations. It is in this memo that then entire history of the Xerox efforts to create an "Architecture for Information" (Peter McCullough, 1969) is made manifest. From this came the networked Altos of the mid 70s that served as a research and development platform that brought us Ethernet, windows/icons interfaces, on-screen full page document creation (WYSIWYG), laser printing, object oriented programming (smalltalk) and many, many other innovations. By 1981 this work had produced such key innovations for Xerox as the Star 8010 workstation and the 9700 high volume laser printer. Needless to say, the revolutions of the 80s and 90s with the Macintosh, Windows, desktop publishing, the Internet and much more owe their direct lineage to the Xerox network created at PARC, and thereby to this document.

For a list of Dr. Lampson's accomplishments see his "Systems" biographical document at Microsoft Research.

XEROX Inter-Office Memorandum

To CSL **Date** December 19, 1972

From Butler Lampson **Location** Palo Alto

Subject Why Alto **Organization** PARC

1. Introduction

This memo discusses the reasons for making a substantial number (10-30) of copies of the personal computer called Alto which has been designed by Chuck Thacker and others. The original motivation for this machine was provided by Alan Kay, who needs about 15-20 'interim Dynabooks' Systems for his education research. Alto has a much broader range of applications than this origin might suggest, however. I will begin by outlining its characteristics, and then go on to consider some of the many exciting uses to which Alto can be put. It turns out that there is some interaction with almost every CSL research program.

2. Characteristics

An Alto system consists of 48-64K 16-bit words of memory (plus parity and perhaps error correction).

A 10 megabyte Diablo disk which transfers one word every 7 us, rotates in 25 ms, and has a track-to-track seek of 8 ms, and worst-case seek of 70 ms.

A 901 line TV monitor whose display surface is almost exactly the size of this page. It is oriented vertically, and is designed to be driven from a bit map in the memory. It takes 32K of memory to fill the display area with a square (825x620) raster. These dots are about 1.4 mils square. It is possible to reduce their width to about 1 mil, which gives an 825x860 raster and 44.3K of memory. The square raster can display 8000 5x7 characters with descenders or 2500 beautiful proportionally-spaced characters.

An undecoded keyboard which allows the processor to determine exactly when each key is depressed or released, and a mouse or other pointing device.

A processor which executes Nova instructions at about 1.5 us/instruction, and can be extended with extra instructions suitable for interpreting Lisp, Bcpl, MPS, or whatever.

A high-bandwidth (10 MHz) communication interface whose details are not yet specified.

Optionally, a fixed-font character generator similar to the one designed and built by Doug Clark. This would save a lot of memory and would permit higher quality characters than can be done with a square raster, but adds no basically new capability. It should cost about $500.

Optionally, a Diablo printer, XGP, or other hardcopy device.

A table about 45" wide and 25" deep to house the machine and mount the display and keyboard.

Most important, a cost of about $lO.5K, which can be reduced to $9.7K by the use of a 2.5 megabyte disk. The cost is about equally split among disk, memory, and everything else. We have spent about twice as much on Maxc per 1974 CSL member.

The system is capable of doing almost any computation which a PDP-10 can do. For most problems it can deliver better performance to the user than a time-shared 10, even if the latter is lightly loaded (obvious exception: lots of floating-point computation). Furthermore, we have under development Lisp, Bcpl, and MPS systems which can run on a Nova and therefore, with slight modification, on Alto. Since most of our own future software work is expected to be done in one of these languages, most of it should be able to run on Alto.

The next paragraph shows that there should be plenty of computing power. Both Lisp and MPS will have some kind of hardware-assisted mapping, as that virtual memory size will not be a problem; a similar arrangement for Bcpl seems feasible, but has not been investigated.

A 64K Alto has as many Lisp cells as 32K of PDP-l0 memory. BBN claims to run Lisp users with an average 25K working set and a 30 ms page fault interval. Forty-two disk tracks hold 256K Lisp cells, and the average access time to a record on one of those tracks is about 32 ms, compared for 17 ms for the 10's drum. Hence, if execution speed on Alto is half the 10 speed or less, paging will cost no more than on the 10 for Lisp programs. It is highly plausible that we can get a Lisp system on an Alto with a few specialized instructions which can deliver half the performance of a Tenex Lisp running in a 32K swap space. Comparable results can be expected for other languages.

3. Applications

All the applications considered here depend on two facts which summarize the contents of the last section:

Alto is more powerful than a VTS terminal connected to Tenex;

Alto is cheap enough that we can buy one for each member of CSL, if that should prove desirable.

a) Distributed computing. We can very easily put in an Aloha-like point-to-point packet network between Alto's, using a coax as the ether (or microwave with a repeater on a hill for home terminals). We can then do a large variety of experiments with dozens of machines. It is easy to try experiments which depend on the independence of the participants as well as those which use specialized components which must cooperate to accomplish anything. In particular, we can set up systems in which each user has his own files and communications is done solely for the

interchange of sharable information, and thus shed some light on the long-standing controversy about the merits of this scheme as against centralized files.

b) Office systems. We can run Peter's Lisp-based NLS-competitor or the xNLS system. The computational overkill of Alto will allow us to concentrate on the capabilities of the system rather than on optimizing its performance. Information gained from this approach should complement that obtained from the multi-user xNLS experiments. It may also be possible to run these on Alto and thus escape from Nova dependency; this possibility requires further investigation.

c) Personal computing. If our theories about the utility of cheap, powerful personal computers are correct, we should be able to demonstrate them convincingly on Alto. If they are wrong, we can find out why. We should, for example, be able to satisfy heavy Lisp users such as Warren and Peter with an Alto. This would also take a big computing load away from Maxc. It should also be quite easy to simulate the hardware configuration of other proposed personal computers (e.g., different memory hierarchies) and thus to validate those designs. This is important because more compact machines will require a much larger investment in engineering development and more precise optimization of the memory system.

d) Graphics. Alto is an excellent vehicle for Bob Flegal's graphics work, and will make the fruits of that work available to a wide community. It can't do Dick Shoup's stuff.

4. Competition

Alto competes with some other things we or SSL are doing. I think this is a good thing, since it encourages the proponents of both approaches to excel. Specifically;

a) VTS can do higher quality characters, has intensity control and blinking, costs half to two-thirds as much if you only want a terminal, and can take advantage of the video switch. It can't do graphics and may suffer from the queueing problems of shared-resource systems (the controlling Nova and communications are shared). And, of course, it is only as good as the computer which uses it.

b) Maxc can compute, and for applications which use existing software, need large working sets, or do lots of multiplications, it will be better. Also, it is known to be good for Lisp, etc., while the suitability of Alto for such large systems remains unproven.

c) Novas which don't have complex interfaces to other hardware (e.g., Toy, XGP) can be replaced by Altos. Those which do, like the Maxc Nova or, probably, the VTS Nova, are secure.

d) The implications of Alto for the local network are unclear.

e) Imlacs are wiped out.