Inter-Office Memorandum

To	PARC/SDD	Date	February 26, 1978
From	Ed Taft	Location	Palo Alto
Subject	Alto Time Standard (Edition 3)	Organization	PARC/CSL

XEROX

Filed on: <AltoDocs>AltoTime.Ears, .Press, <AltoSource>AltoTime.Bravo

I am resurrecting a proposal I made nearly two years ago to change the Alto's internal date and time standard. The need for this change has become more pressing since that time, so I would like to see it accomplished as quickly as possible.

A number of software systems will be affected by the change. This memo attempts to enumerate them, but I have undoubtedly missed some (particularly ones in the Mesa and Smalltalk worlds). Fortunately, the consequence of failing to convert immediately is not catastrophic.

The last section of this memo sets forth a schedule for converting from the old standard to the new. My intention is that the conversion be relatively smooth and that old programs continue to work until such time as support for the old standard is revoked.

Change since Edition 1: The internal representation of a time zone is entirely different so as to accomodate fractions of an hour and other oddities.

Change since Edition 2: Current status.

Present Standard

The current Alto time standard is a 32-bit integer denoting the number of seconds since midnight, January 1, 1901, local time (with Daylight Savings Time already applied if it's that time of the year). This standard has two important difficulties.

First, it is location-dependent: a given 32-bit integer represents a different absolute time depending on the time zone within which it is interpreted. With the proliferation of Altos across many time zones and the certainty that they will eventually be able to communicate with each other, it becomes desirable that a time standard be location-independent. Given this, a host in one time zone may obtain the current date and time (as a 32-bit number) from a host in another time zone, without either host having to know the location of the other. This capability is needed now because Webster has just been connected to our present Parc/SDD/XEOS inter-network.

Second, the present time standard is not monotonic: it jumps forward an hour in April and backward an hour in October so as to conform to daylight and standard time. Hence an Alto's clock cannot be maintained correctly simply by counting milliseconds, and for this reason it is not maintained correctly at present. Furthermore, there is a two-hour stretch of absolute time every October within which each 32-bit number is used twice. The

monotonicity of time on a single machine is vital to the correct operation of programs such as the Maxc and IFS backup systems.

New Standard

The new time standard is similar to the present one, but it uses GMT (Greenwich Mean Time) as a basis rather than local time. The choice of GMT is arbitrary, but it is the one adopted by many computer operating systems, including Tenex.

Precisely: the internal time standard is the number of seconds since midnight, January 1, 1901, GMT, represented as a 32-bit integer. Clocks maintained according to this standard will have the same value at any given instant of absolute time, anywhere in the world.

Local Time Conversion

With adoption of this time standard, responsibility for applying time zone and daylight savings time corrections is shifted into the software that converts between internal and external representations. Fortunately, these corrections are relatively simple. In order to perform conversion between the GMT standard and the local external representation, such software needs three pieces of additional information. These are the local time zone (time east or west of Greenwich), the day of the year on or before which daylight savings time takes effect, and the day of the year on or before which standard time is reinstated (these days are adjusted to the next earlier Sunday).

These quantities are quasi-constant, in that the local time zone changes only when a machine (or a disk pack containing the software) is moved into another time zone, and the DST starting and ending dates change only when Congress sees fit to change the applicable laws, as it did a few years ago during the oil embargo. However, the probability of these or similar events is sufficiently high that it would be a mistake to build the parameters into the software as constants.

The requirement, then, is to maintain and distribute local time correction parameters in as automatic manner as possible. This is accomplished by the following strategy:

1. The parameters are maintained permanently in server hosts (Gateways, IFSs, Maxcs, etc.) that are always up and never move. The numbers are "wired-in" in a manner requiring manual intervention to change.

2. Such hosts have time servers that give out the local time parameters as well as the absolute time (GMT) upon request by a host on a directly-connected, geographically restricted network. It is assumed that no single Ethernet spans multiple time zones. Gateways whose interconnections do cross time zones have to know that they must not believe each others' local time parameters.

3. Each Alto maintains these parameters in reserved locations both in main memory and on the disk. Whenever a time server is accessible, the Alto updates the local parameters with the information provided by the server.

The purpose of keeping the local time parameters in each Alto in a relatively permanent form is to ensure their availability (with a high probability of correctness) if not obtainable from other sources. A user of a stand-alone Alto (one not connected to an Ethernet, or whose Ethernet is down, or whose local time server is down or nonexistent) would justifiably be annoyed if every time his Alto "forgot" the time he had to enter the local time zone and DST parameters in addition to the actual time.

Implementation Details

The UNPACKDT and PACKDT procedures in the "Time" software package have been modified to perform corrections for time zone and daylight savings time. They may be used as models for converting other software that performs similar operations. Note that the corrections are applied during the transformation between the "packed" and "unpacked" internal time formats; no special processing is necessary during the transformation between the "unpacked" format and text strings. However, an option has been added to append the time zone to the text string format so as to permit location-independent processing of it. This is important, for example, in the time stamps included in DMS message headers.

This new version of the "Time" package operates correctly only under OS version 14. If you are responsible for any software that deals with time conversion or printing, you are advised to examine carefully the revised documentation in [Maxc1]<AltoDocs>Time.Tty and possibly the revised source code in the dump-format file <AltoSource>TimeSource.Dm.

Two new procedures are added to the Operating System: ReadCalendar and SetCalendar. They are analogous to the existing DayTime and SetDayTime procedures except that they deal in GMT rather than local time. The reason for adding new procedures rather than simply redefining the existing ones to deal in GMT has to do with the transition from the old time standard to the new. This is dealt with in the next section.

Two additional Alto main memory locations are reserved: 570B and 571B (thereby extending by two words the region reserved for time-related information, presently 572B through 577B). These cells contain the following information:

570B	bit 0	Zero if the local time zone is west of Greenwich, one if east.
	bits 1-4	Local time zone in hours east or west of Greenwich. This is an integer in the range 0 to 12. The Pacific time zone is 8 hours west of Greenwich.
	bits 5-6	Unused.
	bits 7-15	Day of the year on or before which Daylight Savings Time takes effect, where 1 = January 1 and 366 = December 31. The software will adjust this number to the nearest preceding Sunday. The standard value is 121 = April 30. The correspondence between numbers and days is based on a leap year. The software corrects for the absence of February 29 during non-leap years.
571B	bit 0	Unused.
	bits 1-6	Additional minutes east or west of Greenwich, in the range 0 to 59. This is usually zero, but there are a few places in the world whose local time is not an integer number of hours from Greenwich.
	bits 7-15	Day of the year on or before which Daylight Savings Time ends. The standard value is 305 = October 31. If Daylight Savings Time is not observed locally, both the start and end dates should be set to 366.

These new cells (like the existing time-related cells) must not be written into by any boot or core-image restoration process.

Copies of these cells are written into magic locations in the Operating System core image (Sys.Boot) by the OS "Install" procedure and by the SetTime subsystem. The locations used are virtual addresses 1375B and 1376B, which actually reside at word addresses 775B and 776B in the file due to the manner in which boot files are organized.

When the OS is booted, it checks to see whether the local time parameter cells in main

memory contain reasonable values. Booting an old OS will set them to zero, which is invalid. If not, it restores them from the magic locations in the OS core image, which most likely have been set to reasonable values by a previous Install or SetTime.

The time servers on the Gateways and Maxcs are modified to accept a new type of time request and to respond with both the present time (GMT) and the local time parameters.

Converting to the New Standard

The following is intended to be an exhaustive list of changes that must be made, in the order required for a smooth transition from the old standard to the new.

1. Release Operating System version 14, which maintains time according to the new standard and that incorporates the new ReadCalendar and SetCalendar procedures. In this version of the OS, the existing DayTime and SetDayTime procedures are modified to subtract or add 8 hours (respectively) to convert between GMT and PST. This temporarily permits existing, unmodified subsystems to continue to obtain local time even though the OS keeps GMT. **OS 14 has been released.**

2. The preceding step provides complete compatibility (I think) for all existing Bcpl programs except Bravo. Bravo does not use the OS DayTime procedure but rather has its own copy of DayTime. Therefore, it is necessary to release (concurrently with OS version 14) a new version of Bravo that contains the revised time software. The consequence of running an old Bravo under a new OS or a new Bravo under an old OS will be that Bravo's "T" command will generate a time string that is wrong by 8 hours. **The new Bravo has been released.**

3. Implement the new-format time server on the Gateways and Maxcs. **This step has been completed.**

4. Release a new version of the SetTime subsystem that deals with either the old or the new standard, depending on the version of the OS it is running under. **The new SetTime has been released.**

5. Release the new version of the "Time" software package, described previously. This package will work correctly only under OS version 14. **The new Time package has been released.**

6. Release new versions of all programs that deal with dates and times. These include Bravo, DDS, FTP, IFS, OfficeTalk, and undoubtedly one or more Mesa subsystems. The necessary modification consists of incorporating or transliterating the new version of the "Time" software package. Programs containing the new package will *not* work under OS versions prior to 14. **The new Bravo has been released. New versions of subsystems released after February 27, 1978 are not guaranteed to work under any OS prior to version 14.**

7. Update Mesa programs that deal with time. **Mesa time-conversion software conforming to the new standard is available. Consult your local Mesa support organization.**

Several aspects of this transition should be noted. First, the compatible DayTime procedure in the OS will cease to work correctly when Daylight Savings Time goes into effect on April 30, 1978. It is not practical for the OS to perform the calculation for correct handling of DST because doing so requires most of the machinery of UNPACKDT. Therefore, we should strive to complete the transition before that time.

Second, the date properties of all existing Alto files will become incorrect by either 7 or 8 hours, depending on whether or not they were written when DST was in effect. I don't think

anyone will care (except on IFS, where I may undertake to convert all the dates). If anyone does care, writing a program to convert the dates on existing files should be straightforward.

Maintainers of Alto subsystems are now requested to begin converting to the new standard. It is desirable that this conversion be completed by March 31, 1978, so as to give users adequate time to update their disks before compatibility with the old time standard ceases to work.